aws

Linux インスタンス用ユーザーガイド

Amazon Elastic Compute Cloud

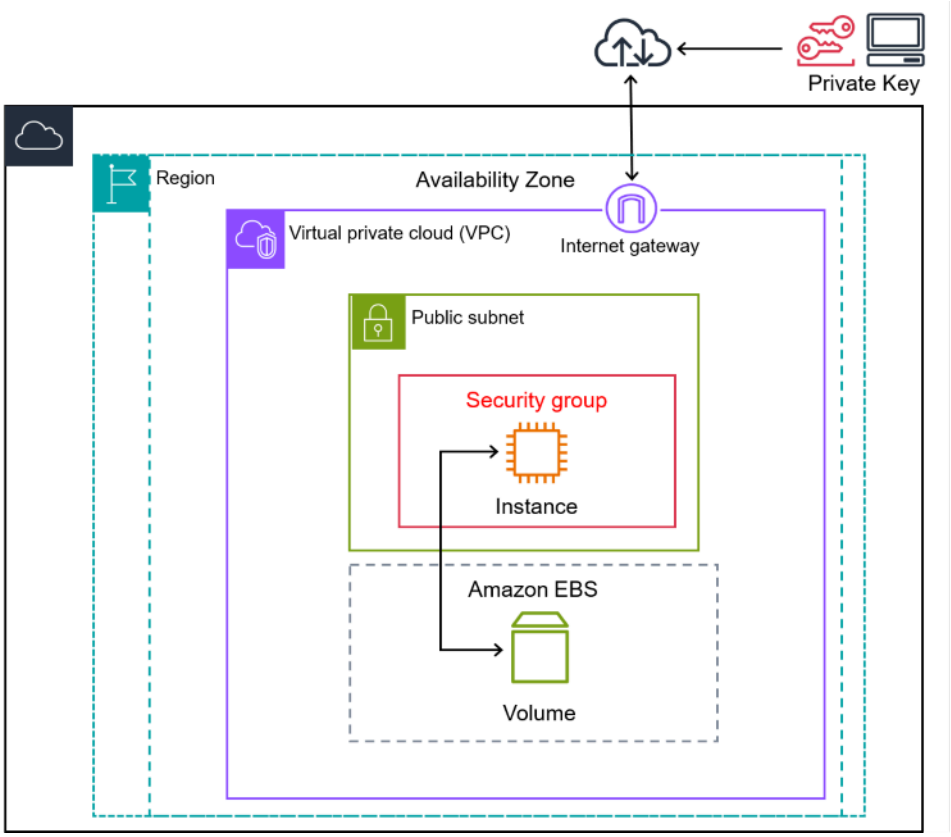
Amazon Elastic Compute Cloud: Linux インスタンス用ユーザーガイド

Table of Contents

Amazon EC2 とは

Amazon Elastic Compute Cloud (Amazon EC2) は、Amazon Web Service (AWS) クラウドでオンデマンドのスケーラブルなコンピューティングキャパシティーを提供します。Amazon EC2 を使用することで、ハードウェアのコストを削減できます。これによりアプリケーションの開発とデプロイを迅速に行うことができます。Amazon EC2 を使用すると、必要な数 (またはそれ以下) の仮想サーバーの起動、セキュリティおよびネットワーキングの構成、ストレージの管理ができます。月次または年次の処理やウェブサイトのトラフィックの急増など、計算量の多いタスクを処理するためのキャパシティーを追加 (スケールアップ) できます。使用量が減った場合は、キャパシティーを再び減らす (スケールダウン) こともできます。

次の図は、Amazon Virtual Private Cloud (VPC) 内にデプロイされた Amazon EC2 インスタンスの基本アーキテクチャを示しています。この例では、EC2 インスタンスはリージョンのアベイラビリティーゾーン内にあります。EC2 インスタンスは、送受信トラフィックを制御する仮想ファイアウォールであるセキュリティグループで保護されます。プライベートキーはローカルコンピュータ、パブリックキーはインスタンスに保存されます。どちらのキーも、ユーザーの ID を証明するためのキーペアとして指定されます。このシナリオでは、インスタンスは Amazon EBS ボリュームによってバックアップされます。VPC は、インターネットゲートウェイを使用してインターネットと通信します。Amazon VPC の詳細については、「Amazon VPC ユーザーガイド」を参照してください。



> **Tip**
>
> このユーザーガイドでは、Amazon EC2 で Linux ベースのインスタンスを実行するための固有の情報を提供します。EC2 で Windows ベースのインスタンスを実行するのに役立つ情報については、「Windows インスタンス用 EC2 ユーザーガイド」を参照してください。

Amazon EC2 は、マーチャントまたはサービスプロバイダーによるクレジットカードデータの処理、ストレージ、および伝送をサポートしており、Payment Card Industry (PCI) Data Security Standard (DSS) に準拠していることが確認されています。PCI DSS の詳細 (AWS PCI Compliance Package のコピーをリクエストする方法など) については、「PCI DSS レベル 1」を参照してください。

Amazon EC2 に関する技術的なガイダンスをお探しの場合は、「AWS re:Post」をお試しください。

クラウドコンピューティングの詳細については、「クラウドコンピューティングとは」を参照してください。

トピック

Amazon EC2 の機能

Amazon EC2 には次の高度な機能があります。

インスタンス

仮想サーバー。

Amazon マシンイメージ (AMI)

サーバーに必要なコンポーネントをパッケージ化した、インスタンス用に事前に設定されているテンプレート(オペレーティングシステムおよび追加のソフトウェアを含む)。

インスタンスのタイプ

インスタンス用の CPU、メモリ、ストレージ、ネットワーキングキャパシティーのさまざまな設定。

キーペア

インスタンス用の安全なログイン情報。AWS はパブリックキー、ユーザーはプライベートキーを安全な場所に保存します。

インスタンスストアボリューム

インスタンスを停止、休止、または終了するときに削除される一時データ用のストレージボリューム。

Amazon EBS ボリューム

Amazon Elastic Block Store (Amazon EBS) を使用したデータ用の永続的ストレージボリューム。

リージョン、アベイラビリティーゾーン、ローカルゾーン、AWS Outposts、Wavelength Zone

インスタンスや Amazon EBS ボリュームなどのリソース用の、複数の物理的な場所。

セキュリティグループ

インスタンスに到達できるプロトコル、ポート、送信元 IP の範囲、およびインスタンスが接続できる宛先 IP の範囲を指定できる仮想ファイアウォール。

Elastic IP アドレス

動的なクラウドコンピューティング用の静的 IPv4 アドレス。

タグ

作成し、Amazon EC2 リソースに割り当てることができるメタデータ。

仮想プライベートクラウド (VPC)

AWS クラウドの他の部分から論理的に分離された、作成可能な仮想ネットワーク。オプションで、これらの仮想ネットワークを独自のネットワークに接続できます。

Amazon EC2 のすべての機能の詳細については、「Amazon EC2 の機能」を参照してください。

AWS でウェブサイトを運営するためのオプションについては、「ウェブホスティング」を参照してください。

Amazon EC2 の使用を開始する

次のトピックは、Amazon EC2 の使用を開始する際に役立ちます。EC2 を使用するように設定したら、チュートリアル: Amazon EC2 Linux インスタンスの開始方法 を通してインスタンスを起動、接続、クリーンアップできます。残りのトピックでは、EC2 の高度な機能に関する詳細について説明します。

EC2 インスタンスを設定および使用する

- Amazon EC2 を使用するようにセットアップする
- チュートリアル: Amazon EC2 Linux インスタンスの開始方法
- Linux インスタンスへの接続
- ファイルの転送

Amazon Lex V2 の基本について学ぶ

- インスタンスと AMI
- リージョンとゾーン

- インスタンスタイプ
- タグ

ネットワークとセキュリティについて読む

- キーペア
- セキュリティグループ
- Elastic IP アドレス
- 仮想プライベートクラウド

ストレージオプションを確認する

- Amazon EBS
- インスタンスストア

Linux チュートリアルを詳しく見る

- AWS Systems Manager を使用して EC2 インスタンスでコマンドをリモートで実行する
- Amazon Linux 2 での LAMP のインストール
- Amazon Linux 2 での SSL/TLS の設定
- WordPress ブログをホストする

EC2 のトラブルシューティング

- EC2 インスタンスのトラブルシューティング
- AWS re:Post

関連サービス

インスタンスやボリュームなど、Amazon EC2 のリソースは Amazon EC2 を使用して直接プロビジョニングできます。さらに、次のようなその他の AWS サービスを使用して、EC2 リソースをプロビジョニングできます。

- Amazon EC2 Auto Scaling

アプリケーションの負荷を処理するために適切な数の Amazon EC2 インスタンスがあることを確認できます。

- AWS CloudFormation

  テンプレートを使用した AWS リソースのモデル化および設定に役立ちます。

- AWS Elastic Beanstalk

  基盤となるインフラストラクチャについて理解していなくても、AWS クラウド にアプリケーションをデプロイして管理できます。

- AWS OpsWorks

  Chef と Puppet を使用して、Amazon EC2 インスタンス全体でサーバーを設定、デプロイ、管理する方法を自動化できます。

- EC2 Image Builder

  カスタマイズされたセキュアで最新のサーバーイメージの作成、管理、デプロイを自動化します。

- AWS Launch Wizard

  個々の AWS リソースを手動で識別およびプロビジョニングすることなく、サードパーティアプリケーション用の AWS リソースのサイズ設定、設定、デプロイを行えます。

その他の関連サービス

- Amazon Lightsail

  ウェブサイトやウェブアプリケーションを構築するために、Amazon Lightsail を使用して基本的なクラウドリソースをデプロイおよび管理できます。実際のユースケースに合わせ Amazon EC2 と Lightsail の機能を比較するには、「Amazon Lightsail または Amazon EC2」を参照してください。

- Elastic Load Balancing

  アプリケーションの着信トラフィックを複数の インスタンスに自動的に分散できます。

- Amazon Relational Database Service (Amazon RDS)

  クラウド内でマネージドリレーショナルデータベースを簡単に設定、運用、およびスケールできます。EC2 インスタンス上でデータベースをセットアップできますが、Amazon RDS には、ソフトウェアのパッチ処理、バックアップ、バックアップの保存など、データベース管理タスクを処理できるという利点があります。

- Amazon Elastic Container Service (Amazon ECS)

  コンテナ化されたアプリケーションを EC2 インスタンスのクラスターにデプロイ、管理、スケーリングできます。

- Amazon Elastic Kubernetes Service (Amazon EKS)

  AWS で Kubernetes アプリケーションを実行します。

- Amazon CloudWatch

  インスタンスと Amazon EBS ボリュームをモニタリングできます。

- Amazon GuardDuty

  不正な、または悪意のある可能性がある EC2 インスタンスの使用を検出します。

- AWS Backup

  Amazon EC2 インスタンスとそれらにアタッチされている Amazon EBS ボリュームのバックアップを自動化できます。

Amazon EC2 へのアクセス

次のインターフェイスを使用して、Amazon EC2 インスタンスを作成および管理できます。

Amazon EC2 コンソール

Amazon EC2 インスタンスおよびリソースを作成、管理するためのシンプルなウェブインターフェイス。AWS アカウントにサインアップ済みの場合は、AWS Management Console にサインインし、コンソールのホームページから [EC2] を選択することで、Amazon EC2 コンソールにアクセスできます。

AWS Command Line Interface

コマンドラインシェルでコマンドを使用して AWS サービスとやり取りを行えます。Windows、Mac、Linux でサポートされています。AWS CLI の詳細については、「AWS Command Line Interface ユーザーガイド」を参照してください。「AWS CLI コマンドリファレンス」で Amazon EC2 のコマンドを確認できます。

AWS Tools for PowerShell

AWS SDK for .NET から公開されている機能に基づいて構築された PowerShell モジュールのセットです。Tools for PowerShell では、PowerShell のコマンドラインから AWS リソースのオ

ペレーションのスクリプトを作成できます。使用を開始する方法については、『AWS Tools for Windows PowerShellユーザーガイド』を参照してください。「AWS Tools for PowerShell コマンドレットリファレンス」で、Amazon EC2 のコマンドレットを確認できます。

AWS CloudFormation

Amazon EC2 は、AWS CloudFormation を使用したリソースの作成をサポートしています。AWS リソースを説明するテンプレートを JSON または YAML 形式で作成すると、AWS CloudFormation はそれらのリソースをプロビジョニングして設定します。CloudFormation テンプレートを再利用して、同じリージョンとアカウント内でも、複数のリージョンとアカウント内でも、同じリソースを複数回プロビジョニングできます。サポートされている Amazon EC2 のリソースタイプとプロパティの詳細については、「AWS CloudFormation ユーザーガイド」の「EC2 リソースタイプのリファレンス」を参照してください。

Query API

Amazon EC2 はクエリ API を提供します。このリクエストは、HTTP 動詞 (GET または POST) とクエリパラメータ `Action` で記述する HTTP または HTTPS リクエストです。Amazon EC2 の API アクションの詳細については、Amazon EC2 API Referenceの「アクション」を参照してください。

AWS SDK

HTTP または HTTPS を介してリクエストを送信する代わりに、言語固有の API を使用してアプリケーションを構築することを希望する場合に備えて、AWS には、ソフトウェアデベロッパー向けのライブラリ、サンプルコード、チュートリアル、その他のリソースが用意されています。これらのライブラリには、リクエストの暗号化署名、リクエストの再試行、エラーレスポンスの処理などのタスクを自動化する基本機能が用意されているので、開発を簡単に始められます。詳細については、「」と「 AWS で構築するツール」を参照してください 。

Amazon EC2 の料金表

Amazon EC2 では、次の料金オプションが提供されています。

無料利用枠

Amazon EC2 は無料で始めることができます。無料利用枠のオプションについては、「AWS 無料利用枠」を参照してください。

オンデマンドインスタンス

インスタンスの使用に対し秒単位 (最低時間は 60 秒) で課金され、長期契約や前払い金は不要です。

Savings Plans

1〜3 年の期間、1 時間 につき USD で、定期的な使用量を守ることにより Amazon EC2 コストを削減できます。

Reserved Instances

1〜3 年の期間、インスタンスタイプとリージョンを含む特定のインスタンス設定を守ることにより Amazon EC2 コストを削減できます。

Spot Instances

未使用の EC2 インスタンスをリクエストして、Amazon EC2 コストを大幅に削減できます。

Dedicated Hosts

オンデマンドで、または Savings Plan の一部として、専用の物理 EC2 サーバーを使用することでコストを削減できます。既存のサーバーバウンドソフトウェアライセンスを使用して、コンプライアンス要件を満たすための支援を受けることができます。

On-Demand Capacity Reservations

任意の期間、特定のアベイラビリティーゾーンの EC2 インスタンス用にキャパシティーを予約します。

1 秒単位の請求

未使用の分および秒単位のコストを請求から排除します。

Amazon EC2 の課金および料金の詳細なリストと購入モデルの詳細については、「Amazon EC2 の料金表」を参照してください。

見積もり、請求、コストの最適化

AWS ユースケースの見積もりを作成するには、AWS Pricing Calculator を使用します。

請求を表示するには、AWS Billing and Cost Management コンソールで請求およびコスト管理ダッシュボードに移動します。請求書には、料金の明細が記載された使用状況レポートへのリンクが記載されています。AWS アカウントの請求の詳細については、AWS Billing and Cost Management ユーザーガイドを参照してください。

AWS の請求、アカウント、イベントについてご質問がある場合は、AWS サポートにお問い合わせください。

プロビジョニングされたサンプル環境の費用を計算するには、「クラウドエコノミクスセンター」を参照してください。プロビジョニングされた環境のコストを計算するときは、EBS ボリュームのスナップショットストレージなどの付随的コストを必ず含めてください。

AWS Trusted Advisor を使用して、AWS 環境のコスト、セキュリティ、およびパフォーマンスを最適化できます。

Amazon EC2 を使用するようにセットアップする

Amazon EC2 インスタンスを初めて作成するための準備を整えるには、このセクションのタスクを完了します。

1. AWS アカウントへのサインアップ
2. 管理ユーザーの作成
3. キーペアを作成する
4. セキュリティグループの作成

完了したら、Amazon EC2 の開始方法のチュートリアルに進むことができます。

AWS アカウントへのサインアップ

AWS アカウントがない場合は、以下のステップを実行して作成します。

AWS アカウント にサインアップするには

1. https://portal.aws.amazon.com/billing/signup を開きます。
2. オンラインの手順に従います。

   サインアップ手順の一環として、通話呼び出しを受け取り、電話のキーパッドを使用して検証コードを入力するように求められます。

   AWS アカウントにサインアップすると、AWS アカウントのルートユーザーが作成されます。ルートユーザーには、アカウントのすべての AWS のサービスとリソースへのアクセス権があります。セキュリティのベストプラクティスとして、管理ユーザーに管理アクセスを割り当て、ルートユーザーアクセスが必要なタスクを実行する場合にのみ、ルートユーザーを使用してください。

サインアップ処理が完了すると、AWS からユーザーに確認メールが送信されます。https://aws.amazon.com/ の アカウント をクリックして、いつでもアカウントの現在のアクティビティを表示し、アカウントを管理することができます。

管理ユーザーの作成

AWS アカウント にサインアップしたら、AWS アカウントのルートユーザー をセキュリティで保護し、AWS IAM Identity Center を有効にして管理ユーザーを作成します。日常的なタスクには、管理ユーザーを使用し、ルートユーザーを使用しないようにします。

AWS アカウントのルートユーザーをセキュリティで保護する

1. ルートユーザー を選択し、AWS アカウント のメールアドレスを入力して、アカウント所有者として AWS Management Console にサインインします。次のページでパスワードを入力します。

   ルートユーザーを使用してサインインする方法については、「AWS サインイン User Guide」の「Signing in as the root user」を参照してください。
2. ルートユーザーの多要素認証 (MFA) を有効にします。

   手順については、「IAM ユーザーガイド」の「AWS アカウント のルートユーザーの仮想 MFA デバイスを有効にする (コンソール)」を参照してください。

管理ユーザーを作成する

1. IAM アイデンティティセンターを有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Center の有効化」を参照してください。
2. IAM アイデンティティセンターで、管理ユーザーに管理アクセス権を付与します。

   IAM アイデンティティセンターディレクトリ をアイデンティティソースとして使用するチュートリアルについては、「AWS IAM Identity Center ユーザーガイド」の「IAM アイデンティティセンターディレクトリ デフォルトでのユーザーアクセスの設定」を参照してください。

管理ユーザーとしてサインインする

- IAM アイデンティティセンターのユーザーとしてサインインするには、IAM アイデンティティセンターのユーザーの作成時に E メールアドレスに送信されたサインイン URL を使用します。

  IAM アイデンティティセンターのユーザーを使用してサインインする方法については、「AWS サインイン ユーザーガイド」の「AWS アクセスポータルにサインイン」を参照してください。

キーペアを作成する

AWS では公開キー暗号化を使用して、お客様のインスタンスのログイン情報の安全性を保護します。Linux インスタンスにはパスワードがありませんが、キーペアを使用することでインスタンスに安全にログインできます。インスタンスを起動するときにキーペアの名前を指定し、SSH を使ってログインする際のプライベートキーを指定します。。

キーペアをまだ作成していない場合は、Amazon EC2 コンソールを使用して作成できます。複数の AWS リージョン でインスタンスを起動する予定がある場合は、各リージョンでキーペアを作成する必要があります。リージョンの詳細については、「リージョンとゾーン」を参照してください。

キーペアを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キーペア] を選択します。
3. [キーペアの作成] を選択します。
4. [Name (名前)] に、キーペアのわかりやすい名前を入力します。Amazon EC2 は、キー名として指定した名前にパブリックキーを関連付けます。キー名には、最大 255 文字の ASCII 文字を含めることができます。先頭または末尾にスペースを含めることはできません。
5. [キーペアタイプ]を使用する場合には、[RSA]または[25519]を選択します。Windows インスタンスでは、ED25519 キーはサポートされていません。
6. [Private key ファイル形式] に、プライベートキーを保存する形式を選択します。OpenSSH で使用できる形式でプライベートキーを保存するには、[pem] を選択します。プライベートキーを PuTTY で使用できる形式で保存するには、[ppk] を選択します。
7. [キーペアの作成] を選択します。
8. ブラウザによって秘密キーファイルが自動的にダウンロードされます。ベースファイル名は、キーペアの名前として指定した名前で、ファイル名拡張子は選択したファイル形式によって決まります。プライベートキーファイルを安全な場所に保存します。

   > **Important**
   >
   > プライベートキーのファイルを保存できるのは、このタイミングだけです。

9. macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は、自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

```
chmod 400 key-pair-name.pem
```

これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については、エラー: Unprotected Private Key File (保護されていないプライベートキーファイル) を参照してください。

詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンス を参照してください。

セキュリティグループの作成

セキュリティグループは、関連付けられたインスタンスのファイアウォールとして動作し、インバウンドトラフィックとアウトバウンドトラフィックの両方をインスタンスレベルでコントロールします。SSH を使用して、IP アドレスからインスタンスに接続できるようにするためのルールをセキュリティグループに追加します。さらに、任意の場所からのインバウンドおよびアウトバウンドの HTTP アクセスおよび HTTPS アクセスを可能にするルールを追加できます。

複数の AWS リージョン でインスタンスを起動する予定がある場合は、各リージョンでセキュリティグループを作成する必要があります。リージョンの詳細については、「リージョンとゾーン」を参照してください。

前提条件

ローカルコンピューターのパブリック IPv4 アドレスが必要です。Amazon EC2 コンソールのセキュリティグループエディタは、パブリック IPv4 アドレスを自動的に検出できます。別の方法として、インターネットブラウザで検索文字列として私の「 IP アドレスは何ですか?」を使用するか、次のサービス: Check IP を使用することもできます。インターネットサービスプロバイダー (ISP) 経由で、またはファイアウォールの内側から静的 IP アドレスなしで接続する場合は、クライアントコンピュータで使用されている IP アドレスの範囲を見つける必要があります。

カスタムのセキュリティグループは、次のいずれかの方法で作成できます。

Console

最小限の権限でセキュリティグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. 上部のナビゲーションバーで、セキュリティグループの AWS リージョン を選択します。セキュリティグループはリージョンに固有であるため、キーペアを作成したリージョンと同じリージョンを選択する必要があります。
3. 左のナビゲーションペインで セキュリティグループを選択します。
4. [セキュリティグループの作成] を選択します。
5. [基本的な詳細] で、次の操作を行います。

   a. 新しいセキュリティグループの名前と説明を入力します。覚えやすい名前 (ユーザー名など) を使用し、その後に _SG_ を続け、さらにリージョン名を続けます。たとえば、me_SG_uswest2 などです。

   b. [VPC] リストで、リージョンのデフォルト VPC を選択します。
6. [インバウンドルール] で、インスタンスに到達する特定のトラフィックを許可するルールを作成します。例えば、HTTP および HTTPS によるトラフィックを受け入れるウェブサーバーには、次のルールを使用します。その他の例については、「さまざまなユースケースのセキュリティグループのルール」を参照してください。

   a. [Add rule] を選択します。タイプ] で HTTP] を選択します。[送信元] で、任意の IPv4 アドレスからの受信 HTTP トラフィックを許可するには [Anywhere-IPv4] を選択し、任意の IPv6 アドレスからの受信 HTTP トラフィックを許可するには [Anywhere-IPv6] を選択します。

   b. [Add rule] を選択します。[Type] で、[HTTPS] を選択します。[送信元] で、任意の IPv4 アドレスからの受信 HTTPS トラフィックを許可するには [Anywhere-IPv4] を選択し、任意の IPv6 アドレスからの受信 HTTPS トラフィックを許可するには [Anywhere-IPv6] を選択します。

   c. [Add rule] を選択します。[タイプ] で [SSH] を選択します。[送信元] で、次のいずれかの操作を行います。

      - ローカルコンピュータのパブリック IPv4 アドレスを自動的に追加するには、[My IP] を選択します。
      - [Custom] を選択して、コンピュータまたはネットワークのパブリック IPv4 アドレスを CIDR 表記で指定します。CIDR 表記で個々の IP アドレスを指定するには、ルーティングサフィックス /32 を追加します (203.0.113.25/32 など)。会社もしくはルーターにより、特定の範囲からアドレスを割り当てられている場合、その範囲全体 (203.0.113.0/24など) を指定します。

> **Warning**
>
> セキュリティ上の理由から、[SSH] のルールを使用する場合は、[送信元] で [Anywhere-IPv4] または [Anywhere-IPv6] を選択しないでください。これを設定すると、インターネット上のすべての IP アドレスから、ご使用のインスタンスへのアクセスが可能になります。この状態は、テスト環境での短時間の使用であれば許容できますが、実稼働環境においては安全ではありません。

7. [アウトバウンドルール] では、デフォルトのルールをそのまま使用し、すべてのアウトバウンドトラフィックを許可します。
8. [セキュリティグループの作成] を選択します。

AWS CLI

AWS CLI を使用してセキュリティグループを作成すると、すべてのアウトバウンドトラフィックを許可するアウトバウンドルールがセキュリティグループに自動的に追加されます。インバウンドルールは自動的に追加されないため、追加する必要があります。

この手順では、create-security-group および authorize-security-group-ingress AWS CLI コマンドを組み合わせてセキュリティグループを作成し、指定されたインバウンドトラフィックを許可するインバウンドルールを追加します。次の手順の代わりに、コマンドを個別に実行して最初にセキュリティグループを作成し、そのセキュリティグループにインバウンドルールを追加する方法があります。

セキュリティグループを作成し、そのセキュリティグループにインバウンドルールを追加するには

次のように create-security-group および authorize-security-group-ingress AWS CLI コマンドを使用します。

```
aws ec2 authorize-security-group-ingress \
    --region us-west-2 \
    --group-id $(aws ec2 create-security-group \
        --group-name myname_SG_uswest2 \
        --description "Security group description" \
        --vpc-id vpc-12345678 \
        --output text \
```

```
        --region us-west-2) \
    --ip-permissions \

IpProtocol=tcp,FromPort=80,ToPort=80,IpRanges='[{CidrIp=0.0.0.0/0,Description="HTTP
from anywhere"}]' \

IpProtocol=tcp,FromPort=443,ToPort=443,IpRanges='[{CidrIp=0.0.0.0/0,Description="HTTPS
from anywhere"}]' \

IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges='[{CidrIp=172.31.0.0/16,Description="SSH
from private network"}]' \

IpProtocol=tcp,FromPort=22,ToPort=22,IpRanges='[{CidrIp=203.0.113.25/32,Description="SSH
from public IP"}]'
```

内容:

- `--region` — インバウンドルールを作成するリージョンを指定します。
- `--group-id` — `create-security-group` コマンドと次のパラメータを指定して、セキュリティグループを作成します。
  - `--group-name` — 新しいセキュリティグループの名前を指定します。覚えやすい名前 (ユーザー名など) を使用し、その後に _SG_ を続け、さらにリージョン名を続けます。例えば、myname_SG_uswest2 と指定します。
  - `--description` — セキュリティグループが許可しているトラフィックについての説明を入力します。
  - `--vpc-id` — リージョンのデフォルト VPC を指定します。
  - `--output` — コマンドの出力形式として `text` を指定します。
  - `--region` — セキュリティグループを作成するリージョンを指定します。インバウンドルールで指定したリージョンと同じである必要があります。
- `--ip-permissions` — セキュリティグループに追加するインバウンドルールを指定します。この例のルールは、どこからでも HTTP および HTTPS トラフィックを受け入れるウェブサーバー、またはプライベートネットワーク (会社またはルーターが特定の範囲からアドレスを割り当てる場合) および指定されたパブリック IP アドレス (コンピューターまたはネットワークのパブリック IPv4 アドレスの CIDR 表記など) から SSH トラフィックを受け入れるウェブサーバーを対象としています。

> **Warning**
>
> セキュリティ上の理由から、SSH 用のルールで CidrIp に 0.0.0.0/0 を指定しないでください。これを設定すると、インターネット上のすべての IP アドレスから、ご使用のインスタンスへのアクセスが可能になります。この状態は、テスト環境での短時間の使用であれば許容できますが、実稼働環境においては安全ではありません。

PowerShell

AWS Tools for Windows PowerShell を使用してセキュリティグループを作成すると、すべてのアウトバウンドトラフィックを許可するアウトバウンドルールがセキュリティグループに自動的に追加されます。インバウンドルールは自動的に追加されないため、追加する必要があります。

この手順では、New-EC2SecurityGroup および Grant-EC2SecurityGroupIngress AWS Tools for Windows PowerShell コマンドを組み合わせてセキュリティグループを作成し、指定されたインバウンドトラフィックを許可するインバウンドルールを追加します。次の手順の代わりに、コマンドを個別に実行して最初にセキュリティグループを作成し、そのセキュリティグループにインバウンドルールを追加する方法があります。

セキュリティグループを作成するには

New-EC2SecurityGroup および Grant-EC2SecurityGroupIngress AWS Tools for Windows PowerShell コマンドを次のように使用します。

```
Import-Module AWS.Tools.EC2
New-EC2SecurityGroup -GroupName myname_SG_uswest2 -Description 'Security group
 description' -VpcId vpc-12345678 -Region us-west-2 | `
    Grant-EC2SecurityGroupIngress `
    -GroupName $_ `
    -Region us-west-2 `
    -IpPermission @(
        (New-Object -TypeName Amazon.EC2.Model.IpPermission -Property @{
                IpProtocol = 'tcp';
                FromPort   = 80;
                ToPort     = 80;
                Ipv4Ranges = @(@{CidrIp = '0.0.0.0/0'; Description = 'HTTP from
 anywhere'})
            }),
        (New-Object -TypeName Amazon.EC2.Model.IpPermission -Property @{
```

```
                IpProtocol = 'tcp';
                FromPort   = 443;
                ToPort     = 443;
                Ipv4Ranges = @(@{CidrIp = '0.0.0.0/0'; Description = 'HTTPS from
anywhere'})
            }),
        (New-Object -TypeName Amazon.EC2.Model.IpPermission -Property @{
                IpProtocol = 'tcp';
                FromPort   = 3389;
                ToPort     = 3389;
                Ipv4Ranges = @(
                    @{CidrIp = '172.31.0.0/16'; Description = 'RDP from private
network'},
                    @{CidrIp = '203.0.113.25/32'; Description = 'RDP from public
IP'}
                    )
            })
    )
```

セキュリティグループの場合:

- -GroupName — 新しいセキュリティグループの名前を指定します。覚えやすい名前 (ユーザー名など) を使用し、その後に _SG_ を続け、さらにリージョン名を続けます。例えば、myname_SG_uswest2 と指定します。
- -Description — セキュリティグループが許可しているトラフィックについての説明を入力します。
- -VpcId — リージョンのデフォルト VPC を指定します。
- -Region — セキュリティグループを作成するリージョンを指定します。

インバウンドルールの場合:

- -GroupName — 作成するセキュリティグループを参照するように $_ を指定します。
- -Region — インバウンドルールを作成するリージョンを指定します。セキュリティルールで指定したリージョンと同じである必要があります。
- -IpPermission — セキュリティグループに追加するインバウンドルールを指定します。この例のルールは、どこからでも HTTP および HTTPS トラフィックを受け入れるウェブサーバー、またはプライベートネットワーク (会社またはルーターが特定の範囲からアドレスを割り当てる場合) および指定されたパブリック IP アドレス (コンピューターまたはネットワー

クのパブリック IPv4 アドレスの CIDR 表記など) から RDP トラフィックを受け入れるウェブサーバーを対象としています。

> **Warning**
>
> セキュリティ上の理由から、RDP 用のルールで `CidrIp` に `0.0.0.0/0` を指定しないでください。これを設定すると、インターネット上のすべての IP アドレスから、ご使用のインスタンスへのアクセスが可能になります。この状態は、テスト環境での短時間の使用であれば許容できますが、実稼働環境においては安全ではありません。

詳細については、「Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ」を参照してください。

チュートリアル: Amazon EC2 Linux インスタンスの開始方法

このチュートリアルを使用して、Amazon Elastic Compute Cloud (Amazon EC2) の使用を開始できます。Linux/ インスタンスを起動、接続、使用する方法について学習します。インスタンスは、AWS クラウド内の仮想サーバーです。Amazon EC2 を使用して、インスタンスで実行されるオペレーティングシステムとアプリケーションをセットアップし、設定することができます。

AWS にサインアップすると、AWS 無料利用枠 を使用して、Amazon EC2 の使用を開始できます。AWS アカウント を作成してから 12 か月間が経過しておらず、Amazon EC2 の無料利用枠を使い切っていない場合、無料利用枠の特典で利用できるオプションを提案します。そうすると、このチュートリアルを料金の発生なしでご利用になれます。それ以外の場合、インスタンスを起動したときから、インスタンスを削除するまで (このチュートリアルの最終タスク) 、アイドル状態のままでも標準の Amazon EC2 使用料が発生します。

関連チュートリアル

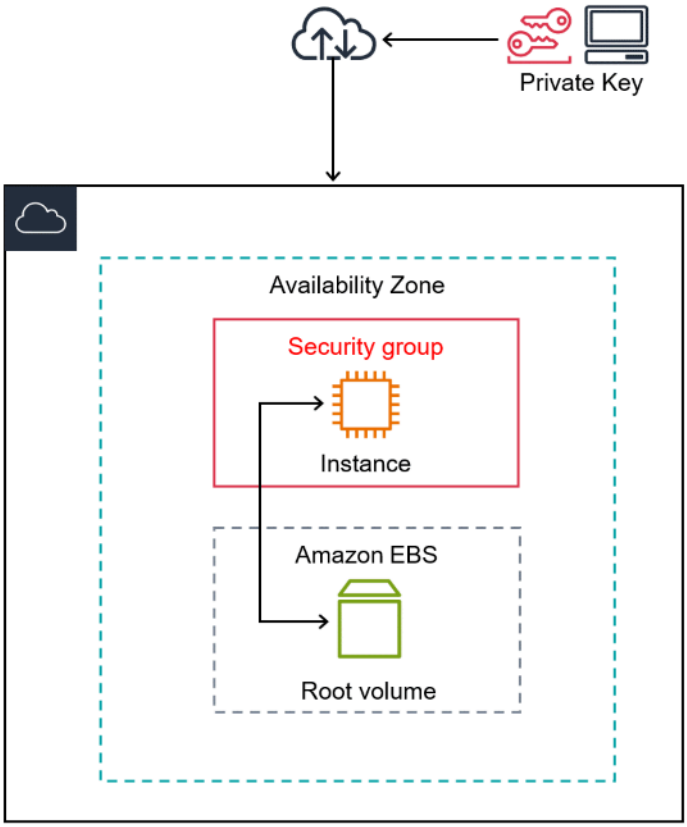
- Windows インスタンスを起動する場合は、Windows インスタンスの Amazon EC2 ユーザーガイド のチュートリアル「Amazon EC2 Windows インスタンスの開始方法」を参照してください。
- コマンドラインを使用する方法については、AWS Command Line Interface ユーザーガイドのチュートリアル「AWS CLI を介した Amazon EC2 の使用」を参照してください。

コンテンツ

概要

このチュートリアルで起動されるインスタンスは Amazon EBS-backed インスタンスです (つまり、ルートボリュームは EBS ボリュームです)。インスタンスが実行されるアベイラビリティーゾーンは、指定するか、Amazon EC2 によって自動的に選択されます。アベイラビリティーゾーンとは、各 AWS リージョン 内に存在する、複数の独立した口ケーションです。アベイラビリティーゾーンは、独立したデータセンターと考えることができます。

インスタンスを起動するときは、キーペア (アイデンティティを証明するため) とセキュリティグループ (着信トラフィックと発信トラフィックを制御する仮想ファイアウォールとして機能する) を指定して、インスタンスをセキュリティで保護します。インスタンスに接続するときは、インスタンスの起動時に指定したキーペアのプライベートキーを指定する必要があります。



前提条件

開始する前に、Amazon EC2 を使用するようにセットアップする の手順を完了するようにしてください。

ステップ 1: インスタンスを起動する

以下の手順で説明しているように、AWS Management Console を使用して Linux/ インスタンスを起動できます。このチュートリアルは、最初のインスタンスをすばやく起動できるようにすることを目的としています。そのため、使用できるすべてのオプションを扱っているわけではありません。詳細オプションについては、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。インスタンスを起動するその他の方法については、「インスタンスの起動」を参照してください。

インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. EC2 コンソールダッシュボードの [インスタンスの起動] ボックスで、[インスタンスの起動] を選択します。
3. [Names and tags] (名前とタグ) の [Name] (名前) には、インスタンス用にわかりやすい名前を入力します。
4. [Application and OS Images (Amazon Machine Image]) (アプリケーションと OS イメージ (Amazon マシンイメージ) で、次の作業を行います。

   a. [Quick Start] (クイックスタート) を選択してから、Amazon Linux を選びます。これが、インスタンスのオペレーティングシステム (OS) です。

   b. [Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から、HVM バージョンの Amazon Linux 2 を選択し、。これらの AMI は [無料利用枠対象] とマークされていることに注意してください。[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) はインスタンスのテンプレートとして機能する基本設定です。

   > **Note**
   >
   > AL2023 は Amazon Linux 2 の後継です。詳細については、「Amazon EC2 コンソールを使用した AL2023 の起動」を参照してください。

5. [Instance type] (インスタンスタイプ) の [Instance type] (インスタンスタイプ) リストからインスタンスのハードウェア構成を選択できます。`t2.micro` インスタンスタイプを選択します。デフォルトではこれが選択されています。`t2.micro` インスタンスタイプは無料利用枠の対象です。`t2.micro` が利用できないリージョンでは、無料利用枠で `t3.micro` インスタンスを使用できます。詳細については、「AWS 無料利用枠」を参照してください。

6. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) には、セットアップ中に作成したキーペアを選択します。

   > **Warning**
   >
   > [Proceed without a key pair (Not recommended)] (キーペアなしで続行 (非推奨) は選択しないでください。キーペアなしでインスタンスを起動すると、インスタンスに接続できません。

7. [Network settings] (ネットワーク設定) の横にある [Edit] (編集) を選択します。Security group name (セキュリティグループ名) には、ウィザードで作成して選択したセキュリティグループが表示されます。このセキュリティグループを使用するか、または次のステップを使用して設定を行うときに作成したセキュリティグループを選択できます。

   a. [Select existing security group] (既存のセキュリティグループを選択) を選択します。

   b. [Common security groups] (共通セキュリティグループ) から、既存のセキュリティグループのリストからセキュリティグループを選択します。

8. インスタンスの他の構成設定については、デフォルトの選択のままにします。

9. Summary (概要) パネルでインスタンス設定の要約を確認します。準備が完了したら、[Launch instance] (インスタンスを起動) を選択します。

10. 確認ページは、インスタンスが起動中であることを通知します。[View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。

11. インスタンス画面で、起動のステータスを確認できます。インスタンスの起動には短時間かかります。インスタンスを起動すると、その初期状態は `pending` です。インスタンスがスタートすると、その状態は `running` に変わり、公開 DNS 名を受け取ります。[Public IPv4 DNS] (パブリック IPv4 DNS) 列が非表示になっている場合は、右上隅にある設定アイコン () をクリックし、[Public IPv4 DNS] (パブリック IPv4 DNS) をオンにした上で [Confirm] (確認) をクリックします。

12. インスタンスに接続可能になるまでには、数分かかることがあります。インスタンスのステータスチェックが正常に終了したことを確認します。この情報は [ステータスチェック] 列に表示されています。

ステップ 2: インスタンスに接続する

Linux インスタンスに接続するにはいくつかの方法があります。詳細については、「Linux インスタンスへの接続」を参照してください。

> **Important**
>
> .pem ファイルがあるキーペアで起動し、コンピュータから SSH アクセスを許可するセキュリティグループで起動していない限り、インスタンスに接続することはできません。インスタンスに接続できない場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

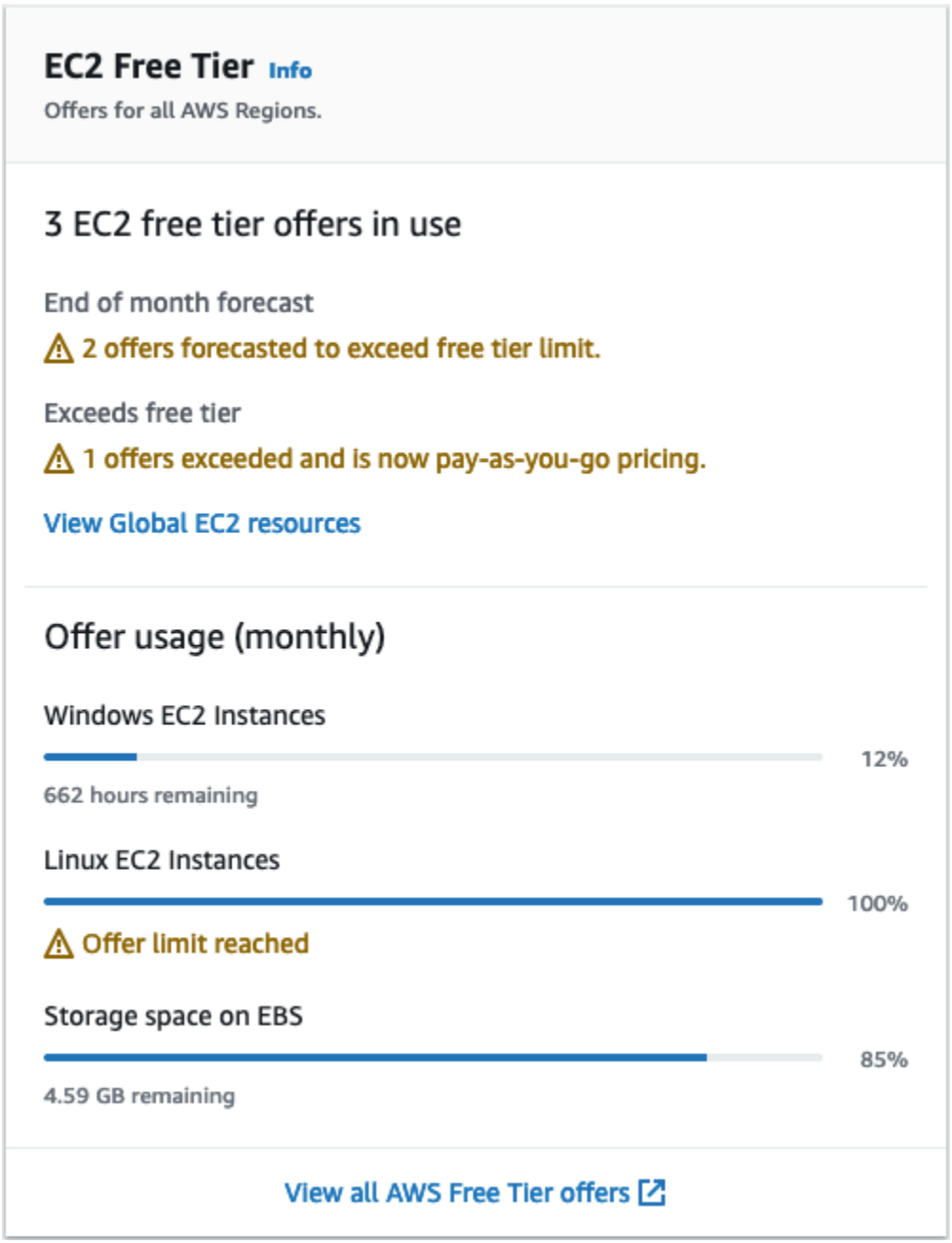
ステップ 3: 無料利用枠の使用状況の追跡

AWS のお客様になってから 12 か月未満で、無料利用枠の制限内であれば、Amazon EC2 は無料で使用できます。予期せぬ請求を避けるには、無料利用枠の使用状況を追跡することが重要です。無料利用枠の制限を超える場合、標準の従量課金制料金が発生します。

> **Note**
>
> 12 か月以上ご利用いただいている AWS カスタマーの場合は、無料利用枠の利用資格がなくなり、以下の手順で説明する [EC2 無料利用枠] ボックスも表示されなくなります。

無料利用枠の使用状況を追跡するには

1. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
2. [EC2 無料利用枠] ボックス (右上) を参照してください。



3. [EC2 無料利用枠] ボックスで、次のように無料利用枠の使用状況を確認します。

   - [使用中の EC2 無料利用枠のオファー] の警告に注意してください。
     - [月末の予測] — 現在の使用パターンを継続する場合、今月は料金が発生することを警告しています。
     - [無料利用枠超過] — 無料利用枠の制限を超え、すでに料金が発生していることを警告しています。

- [オファーの使用量 (月額)] で、Linux インスタンス、Windows インスタンス、EBS ストレージの使用状況を書き留めてください。このパーセンテージは、今月使用した無料利用枠の制限を示します。100% の場合、それ以降の使用には料金が発生します。

> **Note**
>
> この情報は、インスタンスを作成した後にのみ表示されます。ただし、使用状況情報はリアルタイムでは更新されず、1 日 3 回更新されます。

4. さらなる料金の発生を避けるには、現在料金が発生しているリソース、または無料利用枠の使用制限を超えた場合に料金が発生するリソースをすべて削除してください。

   - インスタンスを削除する手順については、このチュートリアルの次のステップに記載しています。
   - 料金が発生する可能性のあるリソースが他のリージョンにあるかどうかを確認するには、[EC2 無料利用枠] ボックスで [グローバル EC2 リソースを表示] を選択して [EC2 グローバルビュー] を開きます。詳細については、「Amazon EC2 Global View」を参照してください。

5. AWS 無料利用枠 にあるすべての AWS のサービス のリソース使用量を表示するには、[EC2 無料利用枠] ボックスの下部にある [すべての AWS 無料利用枠 オファーを表示] を選択します。詳細は、「AWS 料金ユーザーガイド」の「AWS 無料利用枠 を使用する」を参照してください。

ステップ 4: インスタンスをクリーンアップする

このチュートリアル用に作成したインスタンスを使用した操作が終了したら、インスタンスを終了してクリーンアップする必要があります。クリーンアップする前にこのインスタンスでやることがある場合は、「次のステップ」を参照してください。

> **Important**
>
> インスタンスを終了するということは、実質的には、そのインスタンスを削除するということです。いったん終了したインスタンスに再接続することはできません。

AWS 無料利用枠 外でインスタンスを起動した場合は、インスタンスのステータスが `shutting down` または `terminated` に変わるとインスタンスの課金が停止します。後のためにインスタンス

を維持したいが料金を発生させたくない場合は、インスタンスを停止して後で再び開始できます。詳細については、インスタンスの停止と起動 を参照してください。

インスタンスを終了するには

1. ナビゲーションペインで、[インスタンス] を選択します。インスタンスの一覧で、インスタンスを選択します。
2. [Instance state (インスタンスの状態)]、[Terminate instance (インスタンスの終了)] の順に選択します。
3. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択します。

   Amazon EC2 によって、インスタンスがシャットダウンおよび終了します。インスタンスの終了後、インスタンスはしばらくの間コンソールに表示されたままですが、エントリは自動的に削除されます。終了したインスタンスを自分でコンソールディスプレイから削除することはできません。

次のステップ

インスタンスを起動した後で、次の演習の一部を行ってみるといいかもしれません。

- Run コマンドを使用してリモートに EC2 インスタンスを管理する方法を説明します。詳細については、AWS Systems Manager ユーザーガイドの「AWS Systems ManagerRun Command」を参照してください。
- 使用量が無料利用枠を超えた場合に通知する CloudWatch アラームの設定。詳細については、AWS Billing ユーザーガイドの「AWS の無料利用枠の使用量の追跡」を参照してください。
- EBS ボリュームの追加。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの作成」を参照してください。
- LAMP スタックのインストール。詳細については、「Amazon Linux 2 での LAMP のインストール」を参照してください。
- インスタンス購入オプションについてご覧ください。詳細については、「インスタンス購入オプション」を参照してください。
- インスタンスタイプに関するアドバイスを取得します。詳細については、「新しいワークロードのインスタンスタイプに関する推奨事項の取得」を参照してください。

Amazon EC2 のベストプラクティス

Amazon EC2 の利点を最大限に高めるために、以下のベストプラクティスを実践することをお勧めします。

セキュリティ

- AWS リソースおよび API へのアクセス管理は、可能な限り ID プロバイダーおよび IAM ロールによる ID フェデレーションを使用します。詳細については、『IAM ユーザーガイド』の「IAM ポリシーの作成」を参照してください。
- セキュリティグループに対して、最小権限となるルールを適用します。詳細については、「セキュリティグループのルール」を参照してください。
- 定期的にインスタンスのオペレーティングシステムやアプリケーションに対してパッチ処理、更新、および保護を行います。AL2023 のアップデートの詳細については、「AL2023 ユーザーガイド」の「AL2023 のアップデート」を参照してください。Amazon Linux 2 または Amazon Linux AMI の更新の詳細については、Linux インスタンス用 Amazon EC2 ユーザーガイド の「ソフトウェアの管理 (Linux インスタンスの場合)」を参照してください。
- Amazon Inspector を使用して、Amazon EC2 インスタンスを自動的に検出し、ソフトウェアの脆弱性やネットワークへの意図しない公開がないかスキャンします。詳細については、Amazon Inspector ユーザーガイドを参照してください。
- AWS Security Hub コントロールを使用して、Amazon EC2 リソースをセキュリティのベストプラクティスやセキュリティ基準に照らして監視します。セキュリティハブの詳細については、「AWS Security Hub ユーザーガイド」の「Amazon Elastic Compute Cloud の管理」を参照してください。

ストレージ

- データの永続性、バックアップ、および復元に対するルートデバイスタイプの影響について理解します。詳細については、「ルートデバイスのストレージ」を参照してください。
- オペレーティングシステム用およびデータ用として個別に Amazon EBS ボリュームを使用します。データのボリュームがインスタンス終了後も保持されることを確認します。詳細については、「インスタンスの終了時にデータを保持する」を参照してください。
- インスタンスで一時データの格納に使用できるインスタンスストアを使用します。インスタンスを停止、休止、または終了すると、インスタンスストアに格納されたデータは削除されることに注意

してください。データベースストレージにインスタンスストアを使用する場合は、耐障害性を確保するレプリケーション係数が設定されたクラスターがあることを確認します。

- EBS ボリュームとスナップショットを暗号化します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

リソース管理

- AWS リソースを追跡および識別するために、インスタンスメタデータおよびリソースのカスタムタグを使用します。詳細については、「インスタンスメタデータとユーザーデータ」および「Amazon EC2 リソースのタグ付け」を参照してください。
- Amazon EC2 の現在の制限を表示します。制限の引き上げに対するリクエストは、制限の引き上げが必要となる前に計画してください。詳細については、「Amazon EC2 の Service Quotas」を参照してください。
- AWS Trusted Advisor では、お客様の AWS 環境を検査し、コスト削減、システムの可用性とパフォーマンスの向上、セキュリティギャップの封鎖につながるレコメンデーションをお知らせします。詳細については、AWS Support ユーザーガイドの AWS Trusted Advisorを参照してください。

バックアップと復旧

- 定期的に Amazon EBS スナップショットを使用して EBS ボリュームをバックアップし、インスタンスから Amazon Machine Image (AMI) を作成して、それ以降にインスタンスを起動するためのテンプレートとして設定を保存します。このユースケースの実現に役立つ AWS サービスの詳細については、「AWS Backup」と「Amazon Data Lifecycle Manager」を参照してください。
- 複数のアベイラビリティゾーンにアプリケーションの重要なコンポーネントをデプロイし、データを適切にレプリケートします。
- インスタンスが再開したときに、動的な IP アドレスを処理するアプリケーションを設計します。詳細については、「Amazon EC2 インスタンスの IP アドレス指定」を参照してください。
- イベントを管理し、対応します。詳細については、「Amazon EC2 のモニタリング」を参照してください。
- フェイルオーバーを処理する準備が整っていることを確認します。基本的な解決策として、手動でネットワークインターフェイスをアタッチすることも、代替インスタンスに Elastic IP アドレスを関連付けることもできます。詳細については、「Elastic Network Interface」を参照してください。自動化されたソリューションとして Amazon EC2 Auto Scaling を使用できます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」を参照してください。

- インスタンスと Amazon EBS ボリュームを復元するプロセスを定期的にテストして、データとサービスが正常に復元されるようにします。

ネットワーク

- アプリケーションの TTL (有効時間) 値を IPv4 と IPv6 で 255 に設定します。小さい値を使用すると、アプリケーショントラフィックの送信中に TTL が期限切れになり、インスタンスの到達可能性の問題が発生する危険性があります。

Linux を実行する Amazon EC2 インスタンスのチュートリアルです。

次のチュートリアルでは、Linux を実行する EC2 インスタンスを使用して一般的なタスクを実行するための方法を解説します。AWS では、Amazon Linux 2023、Amazon Linux 2、および Amazon Linux AMI が利用できます。詳細については、「Amazon Linux 2023」、「Amazon Linux 2」、および「Amazon Linux AMI」を参照してください。動画チュートリアルについては、「AWS の講習動画とラボ」を参照してください。

> **Note**
>
> AL2023 のチュートリアルについては、「Amazon Linux 2023 ユーザーガイド」の「チュートリアル」を参照してください。

チュートリアル

- LAMP のインストール
- SSL/TLS の設定
- WordPress ブログをホストする

LAMP のインストール

このセクションでは、Amazon EC2 インスタンスに Apache ウェブサーバーを PHP および MariaDB とともにインストールする方法を説明します。

> **Note**
>
> AL2023 LAMP チュートリアルについては、「Amazon Linux 2023 ユーザーガイド」の「チュートリアル: AL2023 で LAMP サーバーをインストールする」を参照してください。

以下への LAMP のインストール

- Amazon Linux 2 での LAMP のインストール
- Amazon Linux への LAMP のインストール

Amazon Linux 2 での LAMP のインストール

次の手順では、Apache ウェブサーバーを PHP と MariaDB (コミュニティによって開発された MySQL の派生版) のサポートとともに Amazon Linux 2 インスタンスにインストールします (LAMP ウェブサーバーまたは LAMP スタックとも呼ばれます)。このサーバーを使用して静的ウェブサイトをホストしたり、データベースとの情報の読み取りと書き込みを行う動的な PHP アプリケーションをデプロイしたりできます。

> **Important**
>
> Ubuntu や Red Hat Enterprise Linux などの別のディストリビューションに LAMP ウェブサーバーを設定しようとすると、このチュートリアルの通りにはなりません。Amazon Linux AMI については、「Amazon Linux への LAMP のインストール」を参照してください。Ubuntu については、Ubuntu コミュニティドキュメントの ApacheMySQLPHP を参照してください。その他のディストリビューションについては、それぞれのドキュメントを参照してください。

オプション: オートメーション を使用してこのチュートリアルを完了する

以下のタスクを行う代わりに AWS Systems Manager オートメーションを使用してこのチュートリアルを完了するには、オートメーションドキュメントである AWSDocs-InstallALAMPServer-AL2 を実行します。

タスク

ステップ 1: LAMP サーバーを準備する

前提条件

- このチュートリアルでは、インターネットからアクセス可能なパブリック DNS 名を持つ、Amazon Linux 2 を使用する新しいインスタンスをすでに起動していることを前提にしています。詳細については、「ステップ 1: インスタンスを起動する」を参照してください。また、セキュリティグループを設定して、SSH (ポート 22)、HTTP (ポート 80)、HTTPS (ポート 443) 接続を有効にしている必要もあります。前提条件の詳細については、Linux インスタンス用のインバウンドトラフィックの承認 を参照してください。
- 次の手順により、Amazon Linux 2 で利用可能な最新の PHP バージョンがインストールされます。現在のバージョンは php8.2 です。このチュートリアルで説明されている以外の PHP アプリケーションを使用する場合は、php8.2 と互換性を確認する必要があります。

LAMP サーバーを準備するには

1. インスタンスに接続します。
2. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。この処理には数分かかりますが、最新の更新とバグ修正を確実に適用することが重要です。

   -y オプションを指定すると、確認メッセージを表示せずに更新をインストールします。インストール前に更新を検査する場合は、このオプションを省略できます。

   ```
   [ec2-user ~]$ sudo yum update -y
   ```

3. mariadb10.5 Amazon Linux Extras リポジトリをインストールして、MariaDB パッケージの最新バージョンを取得します。

   ```
   [ec2-user ~]$ sudo amazon-linux-extras install mariadb10.5
   ```

   sudo: amazon-linux-extras: command not found というエラーが表示された場合、インスタンスは Amazon Linux 2 AMI で起動されていません (おそらく、代わりに Amazon Linux AMI を使用しています)。次のコマンドを使用して、Amazon Linux のバージョンを表示できます。

   ```
   cat /etc/system-release
   ```

4. php8.2 Amazon Linux Extras リポジトリをインストールして、Amazon Linux 2 向け PHP パッケージの最新バージョンを取得します。

```
[ec2-user ~]$ sudo amazon-linux-extras install php8.2
```

5. これでインスタンスが最新状態になったので、Apache ウェブサーバー、MariaDB、および PHP ソフトウェアパッケージをインストールできます。yum インストールコマンドを使用すると、複数のソフトウェアパッケージと関連するすべての依存関係を同時にインストールできます。

```
[ec2-user ~]$ sudo yum install -y httpd
```

次のコマンドを使用して、これらのパッケージの現在のバージョンを表示できます。

```
yum info package_name
```

6. Apache ウェブサーバーを起動します。

```
[ec2-user ~]$ sudo systemctl start httpd
```

7. systemctl コマンドを使用して、システムがブートするたびに Apache ウェブサーバーが起動するように設定します。

```
[ec2-user ~]$ sudo systemctl enable httpd
```

httpd が有効であることは、次のコマンドを実行して確認できます。

```
[ec2-user ~]$ sudo systemctl is-enabled httpd
```

8. インバウンド HTTP (ポート 80) 接続をインスタンスに許可するセキュリティルールを追加していない場合には、このルールを追加します。デフォルトでは、起動時に [launch-wizard-*N*] セキュリティグループがインスタンスに設定されます。このグループには SSH 接続を許可する単一のルールが含まれます。

   a. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

   b. [インスタンス] を選択し、該当するインスタンスを選択します。

   c. [セキュリティ] タブで、インバウンドルールを表示します。次のルールが表示されます。

```
Port range   Protocol     Source
```

```
22          tcp         0.0.0.0/0
```

> **Warning**
>
> `0.0.0.0/0` を使用すると、すべての IPv4 アドレスからインスタンスへの、SSH によるアクセスが許可されます。これはテスト環境で短時間なら許容できますが、実稼働環境で行うのは安全ではありません。本番環境では、特定の IP アドレスまたは特定のアドレス範囲にのみ、インスタンスへのアクセスを限定します。

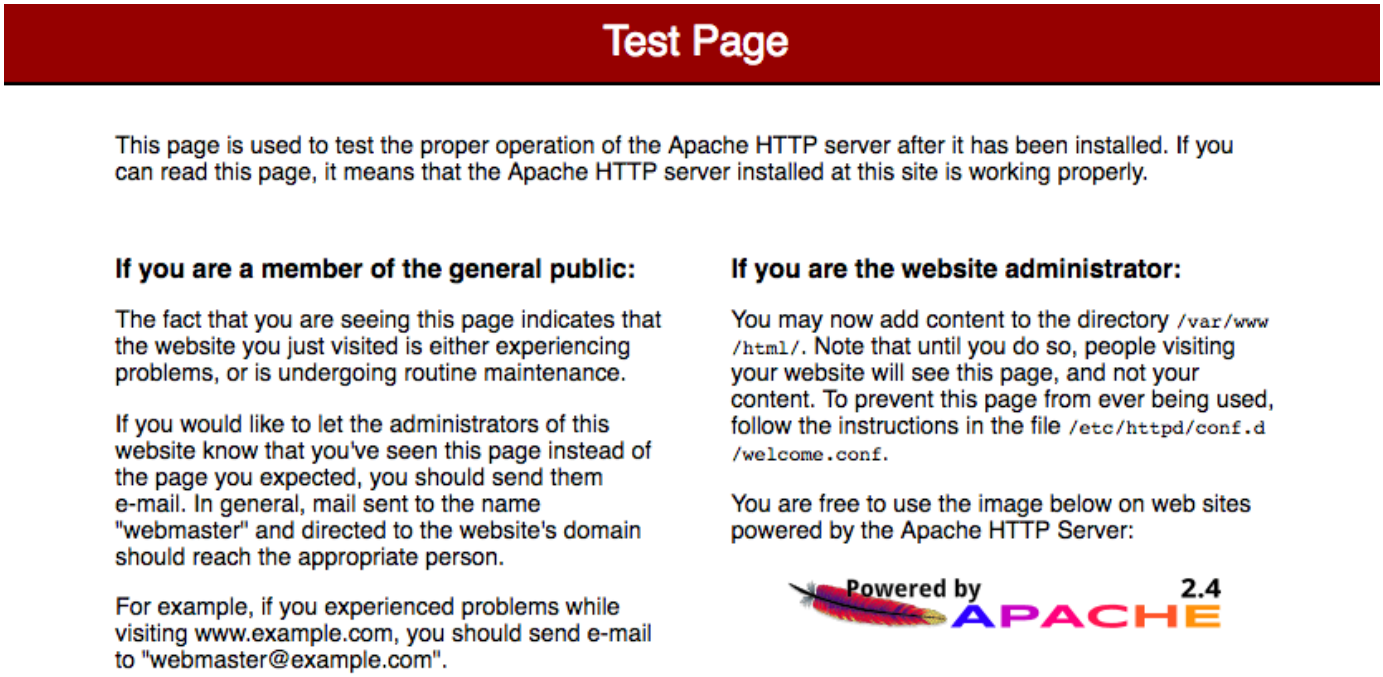
d. セキュリティグループのリンクを選択します。「セキュリティグループへのルールの追加」の手順を使用して、次の値で新しいインバウンドセキュリティルールを追加します。

- [Type]: HTTP
- [Protocol]: TCP
- [Port Range]: 80
- [Source]: Custom

9. ウェブサーバーをテストします。ウェブブラウザで、インスタンスのパブリック DNS アドレス (またはパブリック IP アドレス) を入力します。`/var/www/html` にコンテンツがない場合、Apache テストページが表示されます。インスタンスのパブリック DNS は、Amazon EC2 コンソールを使用して取得できます ([Public DNS] 列を確認します。この列が表示されない場合は、[Show/Hide Columns] (歯車型のアイコン) をクリックして、[Public DNS] を選択します)。

   インスタンスのセキュリティグループに、ポート 80 での HTTP ラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

> **Important**
>
> Amazon Linux を使用していない場合は、それらの接続を許可するようにインスタンスのファイアウォールを設定する必要があるかもしれません。ファイアウォールの設定方法の詳細については、ディストリビューション用のドキュメントを参照してください。



Apache httpd は、Apache ドキュメントルートと呼ばれるディレクトリに維持されるファイルを提供します。Amazon Linux Apache ドキュメントルートは `/var/www/html` であり、デフォルトでは root によって所有されます。

`ec2-user` アカウントがこのディレクトリで複数のファイルを操作することを許可するには、ディレクトリの所有権とアクセス許可を変更する必要があります。このタスクを行うには、複数の方法があります。このチュートリアルでは、`ec2-user` を apache グループに追加し、apache ディレクトリの所有権を `/var/www` グループに付与し、グループへの書き込み権限を割り当てます。

ファイルの許可を設定するには

1. ユーザー (この場合は `ec2-user`) を apache グループに追加します。

```
[ec2-user ~]$ sudo usermod -a -G apache ec2-user
```

2. ログアウトし、再度ログインして新しいグループを選択し、メンバーシップを確認します。

   a. ログアウトします (exit コマンドを使用するか、ターミナルウィンドウを閉じます)。

```
[ec2-user ~]$ exit
```

b. apache グループのメンバーシップを検証するには、インスタンスに再接続して次のコマンドを実行します。

```
[ec2-user ~]$ groups
ec2-user adm wheel apache systemd-journal
```

3. /var/www とそのコンテンツのグループ所有権を apache グループに変更します。

```
[ec2-user ~]$ sudo chown -R ec2-user:apache /var/www
```

4. グループの書き込み許可を追加して、これからのサブディレクトにグループ ID を設定するには、/var/www とサブディレクトのディレクトリ許可を変更します。

```
[ec2-user ~]$ sudo chmod 2775 /var/www && find /var/www -type d -exec sudo chmod
 2775 {} \;
```

5. グループ書き込み許可を追加するには、/var/www とサブディレクトリのファイル許可を再帰的に変更します。

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
```

ここで、ec2-user (および apache グループの将来のメンバー) は、Apache ドキュメントルートでファイルを追加、削除、編集できるようになります。したがって、静的ウェブサイトや PHP アプリケーションなどのコンテンツを追加できます。

ウェブサーバーを保護するには (オプション)

HTTP プロトコルを実行するウェブサーバーは、送受信したデータのトランスポートセキュリティを提供しません。ウェブブラウザを使用して HTTP サーバーに接続すると、閲覧した URL、受信したウェブページのコンテンツ、送信した HTML フォームの内容 (パスワードなど) はすべて、ネットワーク経路上のだれでも傍受できるようになります。ウェブサーバーを保護するためのベストプラクティスとして、SSL/TLS 暗号化でデータを保護する HTTPS (HTTP Secure) のサポートをインストールしてください。

サーバーで HTTPS を有効にする方法については、「Amazon Linux 2 での SSL/TLS の設定」を参照してください。

ステップ 2: LAMP サーバーをテストする

サーバーがインストールおよび実行されており、ファイルのアクセス許可が正しく設定されている場合、ec2-user アカウントは、インターネットから使用できる /var/www/html ディレクトリに PHP ファイルを作成できます。

LAMP サーバーをテストするには

1. Apache ドキュメントルートで PHP ファイルを作成します。

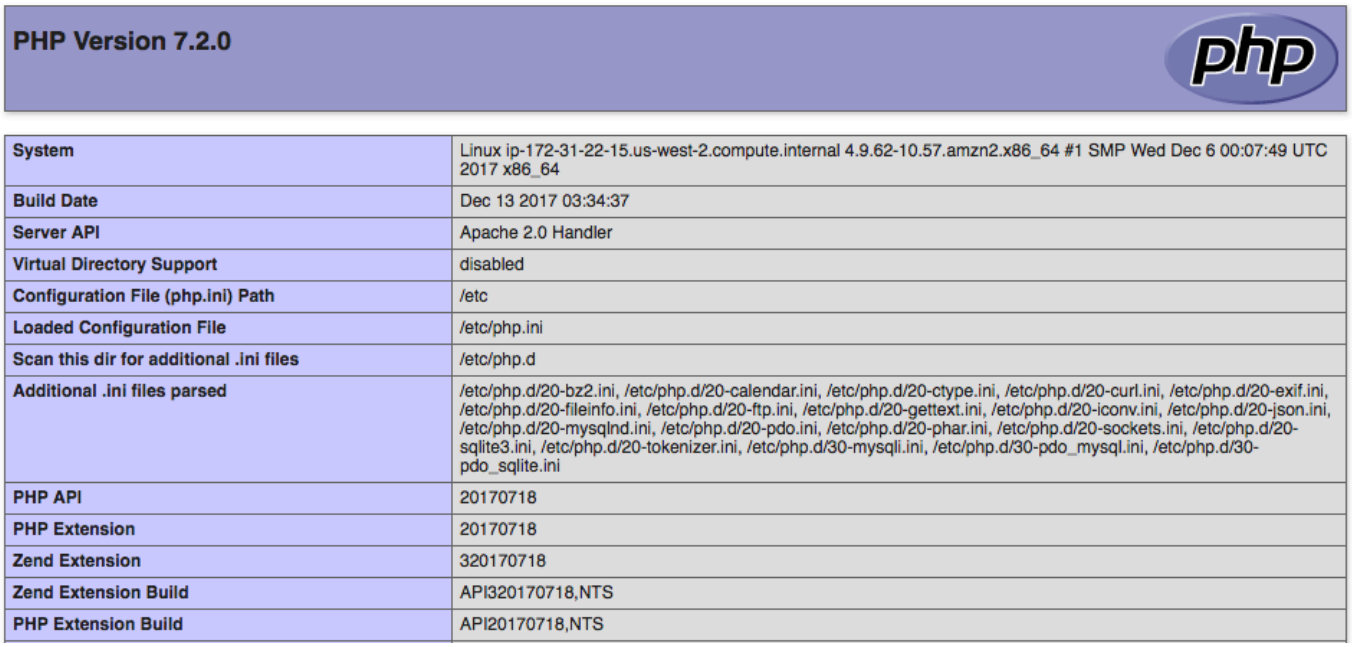
   ```
   [ec2-user ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
   ```

   このコマンドを実行しようとしたときに「許可が拒否されました」というエラーが表示された場合は、ログアウトし、再度ログインして、ファイルの許可を設定するには で設定した正しいグループ許可を取得します。

2. ウェブブラウザで、作成したファイルの URL を入力します。この URL は、インスタンスのパブリック DNS アドレスにスラッシュとファイル名を追加したものです。次に例を示します。

   ```
   http://my.public.dns.amazonaws.com/phpinfo.php
   ```

   PHP 情報ページが表示されるはずです。



PHP Version 7.2.0

System	Linux ip-172-31-22-15.us-west-2.compute.internal 4.9.62-10.57.amzn2.x86_64 #1 SMP Wed Dec 6 00:07:49 UTC 2017 x86_64
Build Date	Dec 13 2017 03:34:37
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-json.ini, /etc/php.d/20-mysqlnd.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini
PHP API	20170718
PHP Extension	20170718
Zend Extension	320170718
Zend Extension Build	API320170718,NTS
PHP Extension Build	API20170718,NTS

このページが表示されない場合は、前のステップで /var/www/html/phpinfo.php ファイルが正しく作成されたことを確認します。次のコマンドで、必要なパッケージがすべてインストールされたことを確認することもできます。

```
[ec2-user ~]$ sudo yum list installed httpd mariadb-server php-mysqlnd
```

必要なパッケージのいずれかが出力に表示されていない場合は、sudo yum install *package* コマンドを使ってインストールします。また、php7.2 と lamp-mariadb10.2-php7.2 のエキストラが amazon-linux-extras のコマンド出力で有効になっていることを確認してください。

3. phpinfo.php ファイルを削除します。これは有用な情報であることもありますが、セキュリティ上の理由から、インターネット上で公表しないでください。

```
[ec2-user ~]$ rm /var/www/html/phpinfo.php
```

これで、完全に機能する LAMP ウェブサーバーを設定しました。/var/www/html の Apache ドキュメントルートにコンテンツを追加する場合、そのコンテンツはインスタンスのパブリック DNS アドレスで表示できます。

ステップ 3: データベースサーバーをセキュリティで保護する

MariaDB サーバーのデフォルトのインストールには、テストおよび開発に役立ついくつかの機能がありますが、実稼働サーバーでは無効にするか削除する必要があります。mysql_secure_installation コマンドを使用すると、ルートパスワードを設定し、安全でない機能をインストールから削除する手順が案内されます。MariaDB サーバーを使用する予定がない場合でも、この手順を実行することが推奨されます。

MariaDB サーバーをセキュリティで保護するには

1. MariaDB サーバーを起動します。

```
[ec2-user ~]$ sudo systemctl start mariadb
```

2. mysql_secure_installation を実行します。

```
[ec2-user ~]$ sudo mysql_secure_installation
```

   a. プロンプトが表示されたら、ルートアカウントのパスワードを入力します。

i. 現在のルートパスワードを入力します。デフォルトでは、ルートアカウントにはパスワードが設定されていません。Enter キーを押します。

ii. 「**Y**」と入力してパスワードを設定し、安全なパスワードを 2 回入力します。安全なパスワード作成の詳細については、「https://identitysafe.norton.com/password-generator/」を参照してください。このパスワードは必ず安全な場所に保管します。

MariaDB のルートパスワードの設定は、データベースを保護するための最も基本的な手段にすぎません。データベース駆動型アプリケーションを構築またはインストールする必要がある場合、通常はそのアプリケーションのデータベースサービスユーザーを作成します。ルートアカウントは、データベース管理以外には使用しないでください。

b. 「**Y**」と入力して匿名ユーザーアカウントを削除します。

c. 「**Y**」と入力してリモートルートログインを無効にします。

d. 「**Y**」と入力してテストデータベースを削除します。

e. 「**Y**」と入力して権限テーブルを再ロードし、変更を保存します。

3. (オプション) MariaDB サーバーをすぐに使用する予定がない場合は、これを停止します。再び必要になったときには再起動できます。

```
[ec2-user ~]$ sudo systemctl stop mariadb
```

4. (オプション) ブート時に毎回 MariaDB サーバーを起動させる場合は、次のコマンドを入力します。

```
[ec2-user ~]$ sudo systemctl enable mariadb
```

ステップ 4: (オプション) phpMyAdmin をインストールする

phpMyAdmin は、EC2 インスタンスで MySQL データベースを表示して編集するために使用できる、ウェブベースのデータベース管理ツールです。Amazon Linux インスタンスで phpMyAdmin をインストールして設定するには、以下の手順に従ってください。

> **Important**
>
> Apache で SSL/TLS を有効にしていない場合、LAMP サーバーへのアクセスに phpMyAdmin を使用することは推奨されません。そのようにすると、データベース管理者のパスワードや他のデータは、インターネット上を安全ではない状態で送信されます。開発者によるセキュリティ関連の推奨事項については、「Securing your phpMyAdmin installation」を参照して

ください。EC2 インスタンスでのウェブサーバーの保護に関する一般的な情報については、「Amazon Linux 2 での SSL/TLS の設定」を参照してください。

pMyAdmin をインストールするには

1. 必要な依存ファイルをインストールします。

```
[ec2-user ~]$ sudo yum install php-mbstring php-xml -y
```

2. Apache を再起動します。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

3. php-fpm を再起動します。

```
[ec2-user ~]$ sudo systemctl restart php-fpm
```

4. /var/www/html で Apache ドキュメントルートに移動します。

```
[ec2-user ~]$ cd /var/www/html
```

5. https://www.phpmyadmin.net/downloads で最新の phpMyAdmin リリース用のソースパッケージを選択します。ファイルディレクトリをインスタンスにダウンロードするには、次の例のようにリンクをコピーして wget コマンドに貼り付けます。

```
[ec2-user html]$ wget https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-languages.tar.gz
```

6. phpMyAdmin フォルダを作成し、次のコマンドでパッケージを展開します。

```
[ec2-user html]$ mkdir phpMyAdmin && tar -xvzf phpMyAdmin-latest-all-languages.tar.gz -C phpMyAdmin --strip-components 1
```

7. *phpMyAdmin-latest-all-languages.tar.gz* Tarball を削除します。

```
[ec2-user html]$ rm phpMyAdmin-latest-all-languages.tar.gz
```

8. (オプション) MySQL サーバーが実行中ではない場合は、今すぐ起動します。

```
[ec2-user ~]$ sudo systemctl start mariadb
```

9. ウェブブラウザで、phpMyAdmin のインストール URL を入力します。この URL は、インスタンスのパブリック DNS アドレス (または、パブリック IP アドレス) にスラッシュとインストールディレクトリを追加してものです。次に例を示します。

   ```
   http://my.public.dns.amazonaws.com/phpMyAdmin
   ```

   phpMyAdmin ログインページが表示されます。

10. 前に作成した root ユーザー名と MySQL のルートパスワードを使って、phpMyAdmin インストールにログインします。

    インストールは、サービス開始前に設定する必要があります。次の手順に従って、設定ファイルを手動で作成することから始めるのをお勧めします。

a. 最小の設定ファイルから開始するには、お気に入りのテキストエディタを使用して新しいファイルを作成し、`config.sample.inc.php` の内容をそのファイルにコピーします。
b. `index.php` を含む phpMyAdmin ディレクトリに、ファイルを `config.inc.php` として保存します。
c. 追加のセットアップについては、phpMyAdmin のインストール手順の「セットアップスクリプトの使用」セクションにある「ファイル作成後の手順」を参照してください。

phpMyAdmin の使用に関する情報は、「phpMyAdmin ユーザーガイド」を参照してください。

トラブルシューティング

このセクションでは、新しい LAMP サーバーの設定時に発生する可能性がある一般的な問題の解決案を提供します。

ウェブブラウザを使用してサーバーに接続できません。

以下のチェックを行って、Apache ウェブサーバーが実行されていて、アクセス可能であるかどうかを確認します。

- ウェブサーバーが実行されていますか?

  httpd が有効であることは、次のコマンドを実行して確認できます。

  ```
  [ec2-user ~]$ sudo systemctl is-enabled httpd
  ```

  httpd プロセスが実行されていない場合は、LAMP サーバーを準備するには に記載されているステップを繰り返します。
- ファイアウォールは正しく設定されていますか?

  インスタンスのセキュリティグループに、ポート 80 での HTTP ラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

HTTPS を使用してサーバーに接続できない

以下のチェックを行って、Apache ウェブサーバーが HTTPS をサポートするように設定されているかどうかを確認します。

- ウェブサーバは正しく設定されていますか?

  Apache をインストールすると、サーバーは HTTP トラフィック用に設定されます。HTTPS をサポートするには、サーバーで TLS を有効にし、SSL 証明書をインストールします。詳細については、Amazon Linux 2 での SSL/TLS の設定 を参照してください。

- ファイアウォールは正しく設定されていますか?

  インスタンスのセキュリティグループに、ポート 443 で HTTPS トラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

関連トピック

インスタンスへのファイルの転送、またはウェブサーバーへの WordPress ブログのインストールの詳細については、次のドキュメントを参照してください。

- WinSCP を使用した Linux インスタンスへのファイルの転送
- SCP クライアントを使用した Linux インスタンスへのファイルの転送
- Amazon Linux 2 での WordPress ブログのホスト

このチュートリアルで使用されているコマンドおよびソフトウェアの詳細については、次のウェブページを参照してください。

- Apache ウェブサーバー: http://httpd.apache.org/
- MariaDB データベースサーバー: https://mariadb.org/
- PHP プログラミング言語: http://php.net/
- chmod コマンド: https://en.wikipedia.org/wiki/Chmod
- chown コマンド: https://en.wikipedia.org/wiki/Chown

ウェブサーバーのドメイン名の登録、または、既存のドメイン名をこのホストに移す方法についての詳細は、『Amazon Route 53 デベロッパーガイド』の「Amazon Route 53 のドメインとサブドメインの作成と移行」を参照してください。

Amazon Linux への LAMP のインストール

次の手順では、Apache ウェブサーバーを PHP と MySQL のサポートとともに Amazon Linux インスタンスにインストールします (LAMP ウェブサーバーまたは LAMP スタックとも呼ばれます)。このサーバーを使用して静的ウェブサイトをホストしたり、データベースとの情報の読み取りと書き込みを行う動的な PHP アプリケーションをデプロイしたりできます。

> **Important**
>
> Ubuntu や Red Hat Enterprise Linux などの別のディストリビューションに LAMP ウェブサーバーを設定しようとすると、このチュートリアルの通りにはなりません。Amazon Linux 2 については、「Amazon Linux 2 での LAMP のインストール」を参照してください。Ubuntu については、Ubuntu コミュニティドキュメントの ApacheMySQLPHP を参照してください。その他のディストリビューションについては、それぞれのドキュメントを参照してください。

オプション: オートメーション を使用してこのチュートリアルを完了する

以下のタスクを行う代わりに AWS Systems Manager オートメーションを使用してこのチュートリアルを完了するには、オートメーションドキュメントである AWSDocs-InstallALAMPServer-AL を実行します。

タスク

- ステップ 1: LAMP サーバーを準備する
- ステップ 2: LAMP サーバーをテストする
- ステップ 3: データベースサーバーをセキュリティで保護する
- ステップ 4: (オプション) phpMyAdmin をインストールする
- トラブルシューティング
- 関連トピック

ステップ 1: LAMP サーバーを準備する

前提条件

このチュートリアルでは、インターネットからアクセス可能なパブリック DNS 名を持つ、Amazon Linux AMI を使用する新しいインスタンスをすでに起動していることを前提にしています。詳細につ

いては、「ステップ 1: インスタンスを起動する」を参照してください。また、セキュリティグループを設定して、SSH (ポート 22)、HTTP (ポート 80)、HTTPS (ポート 443) 接続を有効にしている必要もあります。前提条件の詳細については、Linux インスタンス用のインバウンドトラフィックの承認 を参照してください。

Amazon Linux AMI を使用して LAMP ウェブサーバーをインストールして起動するには

1. インスタンスに接続します。
2. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。この処理には数分かかりますが、最新の更新とバグ修正を確実に適用することが重要です。

   -y オプションを指定すると、確認メッセージを表示せずに更新をインストールします。インストール前に更新を検査する場合は、このオプションを省略できます。

   ```
   [ec2-user ~]$ sudo yum update -y
   ```
3. これでインスタンスが最新状態になったので、Apache ウェブサーバー、MySQL、PHP ソフトウェアパッケージをインストールできます。

   > **Important**
   >
   > アプリケーションによっては、次の推奨のソフトウェア環境と互換性がない場合があります。これらのパッケージをインストールする前に、LAMP アプリケーションと互換性があることを確認してください。問題がある場合には、代替環境のインストールが必要になることがあります。詳細については、「サーバーで実行するアプリケーションソフトウェアに、インストールされている PHP バージョンまたは他のソフトウェアとの互換性がありません。」を参照してください。

   yum install コマンドを使用すると、複数のソフトウェアパッケージと関連するすべての依存関係を同時にインストールできます。

   ```
   [ec2-user ~]$ sudo yum install -y httpd24 php72 mysql57-server php72-mysqlnd
   ```

   No package *package-name* available エラーが表示された場合、インスタンスは Amazon Linux AMI で起動されていません (おそらく、代わりに Amazon Linux 2 を使用しています)。次のコマンドを使用して、Amazon Linux のバージョンを表示できます。

```
cat /etc/system-release
```

4. Apache ウェブサーバーを起動します。

```
[ec2-user ~]$ sudo service httpd start
Starting httpd:                                            [  OK  ]
```

5. chkconfig コマンドを使用して、システムがブートするたびに Apache ウェブサーバーが起動するように設定します。

```
[ec2-user ~]$ sudo chkconfig httpd on
```

chkconfig コマンドでは、それを使用してサービスを正常に有効にしたときに確認メッセージは一切表示されません。

httpd が有効であることは、次のコマンドを実行して確認できます。

```
[ec2-user ~]$ chkconfig --list httpd
httpd           0:off   1:off   2:on    3:on    4:on    5:on    6:off
```

ここ,httpd、onランレベル 2、3、4、5 (見たいものです)。

6. インバウンド HTTP (ポート 80) 接続をインスタンスに許可するセキュリティルールを追加していない場合には、このルールを追加します。デフォルトでは、起動時に [launch-wizard-*N*] セキュリティグループがインスタンスに設定されます。このグループには SSH 接続を許可する単一のルールが含まれます。

   a. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

   b. [インスタンス] を選択し、該当するインスタンスを選択します。

   c. [セキュリティ] タブで、インバウンドルールを表示します。次のルールが表示されます。

```
Port range   Protocol     Source
22           tcp          0.0.0.0/0
```

> **Warning**
>
> `0.0.0.0/0` を使用すると、すべての IPv4 アドレスからインスタンスへの、SSH によるアクセスが許可されます。これはテスト環境で短時間なら許容できますが、

実稼働環境で行うのは安全ではありません。本番環境では、特定の IP アドレスまたは特定のアドレス範囲にのみ、インスタンスへのアクセスを限定します。

d. セキュリティグループのリンクを選択します。「セキュリティグループへのルールの追加」の手順を使用して、次の値で新しいインバウンドセキュリティルールを追加します。

- [Type]: HTTP
- [Protocol]: TCP
- [Port Range]: 80
- [Source]: Custom

7. ウェブサーバーをテストします。ウェブブラウザで、インスタンスのパブリック DNS アドレス (またはパブリック IP アドレス) を入力します。インスタンスのパブリック DNS アドレスを取得するには、Amazon EC2 コンソールを使用します。`/var/www/html` にコンテンツがない場合、Apache テストページが表示されます。ドキュメントルートに追加したコンテンツは、テストページからではなくインスタンスのパブリック DNS アドレスからアクセスできます。

   インスタンスのセキュリティグループに、ポート 80 での HTTP ラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

   Amazon Linux を使用していない場合は、それらの接続を許可するようにインスタンスのファイアウォールを設定する必要があるかもしれません。ファイアウォールの設定方法の詳細については、ディストリビューション用のドキュメントを参照してください。

Apache httpd は、Apache ドキュメントルートと呼ばれるディレクトリに維持されるファイルを提供します。Amazon Linux Apache ドキュメントルートは `/var/www/html` であり、デフォルトでは root によって所有されます。

```
[ec2-user ~]$ ls -l /var/www
total 16
drwxr-xr-x 2 root root 4096 Jul 12 01:00 cgi-bin
drwxr-xr-x 3 root root 4096 Aug  7 00:02 error
drwxr-xr-x 2 root root 4096 Jan  6  2012 html
drwxr-xr-x 3 root root 4096 Aug  7 00:02 icons
drwxr-xr-x 2 root root 4096 Aug  7 21:17 noindex
```

`ec2-user` アカウントがこのディレクトリで複数のファイルを操作することを許可するには、ディレクトリの所有権とアクセス許可を変更する必要があります。このタスクを行うには、複数の方法が

あります。このチュートリアルでは、ec2-user を apache グループに追加し、apache ディレクトリの所有権を /var/www グループに付与し、グループへの書き込み権限を割り当てます。

ファイルの許可を設定するには

1. ユーザー (この場合は ec2-user) を apache グループに追加します。

```
[ec2-user ~]$ sudo usermod -a -G apache ec2-user
```

2. ログアウトし、再度ログインして新しいグループを選択し、メンバーシップを確認します。

   a. ログアウトします (exit コマンドを使用するか、ターミナルウィンドウを閉じます)。

```
[ec2-user ~]$ exit
```

   b. apache グループのメンバーシップを検証するには、インスタンスに再接続して次のコマンドを実行します。

```
[ec2-user ~]$ groups
ec2-user wheel apache
```

3. /var/www とそのコンテンツのグループ所有権を apache グループに変更します。

```
[ec2-user ~]$ sudo chown -R ec2-user:apache /var/www
```

4. グループの書き込み許可を追加して、これからのサブディレクトにグループ ID を設定するには、/var/www とサブディレクトのディレクトリ許可を変更します。

```
[ec2-user ~]$ sudo chmod 2775 /var/www
[ec2-user ~]$ find /var/www -type d -exec sudo chmod 2775 {} \;
```

5. グループ書き込み許可を追加するには、/var/www とサブディレクトリのファイル許可を再帰的に変更します。

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0664 {} \;
```

ここで、ec2-user (および apache グループの将来のメンバー) は、Apache ドキュメントルートでファイルを追加、削除、編集できるようになります。したがって、静的ウェブサイトや PHP アプリケーションなどのコンテンツを追加できます。

(オプション) ウェブサーバーの保護

HTTP プロトコルを実行するウェブサーバーは、送受信したデータのトランスポートセキュリティを提供しません。ウェブブラウザを使用して HTTP サーバーに接続すると、閲覧した URL、受信したウェブページのコンテンツ、送信した HTML フォームの内容 (パスワードなど) はすべて、ネットワーク経路上のだれでも傍受できるようになります。ウェブサーバーを保護するためのベストプラクティスとして、SSL/TLS 暗号化でデータを保護する HTTPS (HTTP Secure) のサポートをインストールしてください。

サーバーで HTTPS を有効にする方法については、「Amazon Linux での SSL/TLS の設定」を参照してください。

ステップ 2: LAMP サーバーをテストする

サーバーがインストールおよび実行されており、ファイルのアクセス許可が正しく設定されている場合、`ec2-user` アカウントは、インターネットから使用できる `/var/www/html` ディレクトリに PHP ファイルを作成できます。

LAMP ウェブサーバーをテストするには

1. Apache ドキュメントルートで PHP ファイルを作成します。

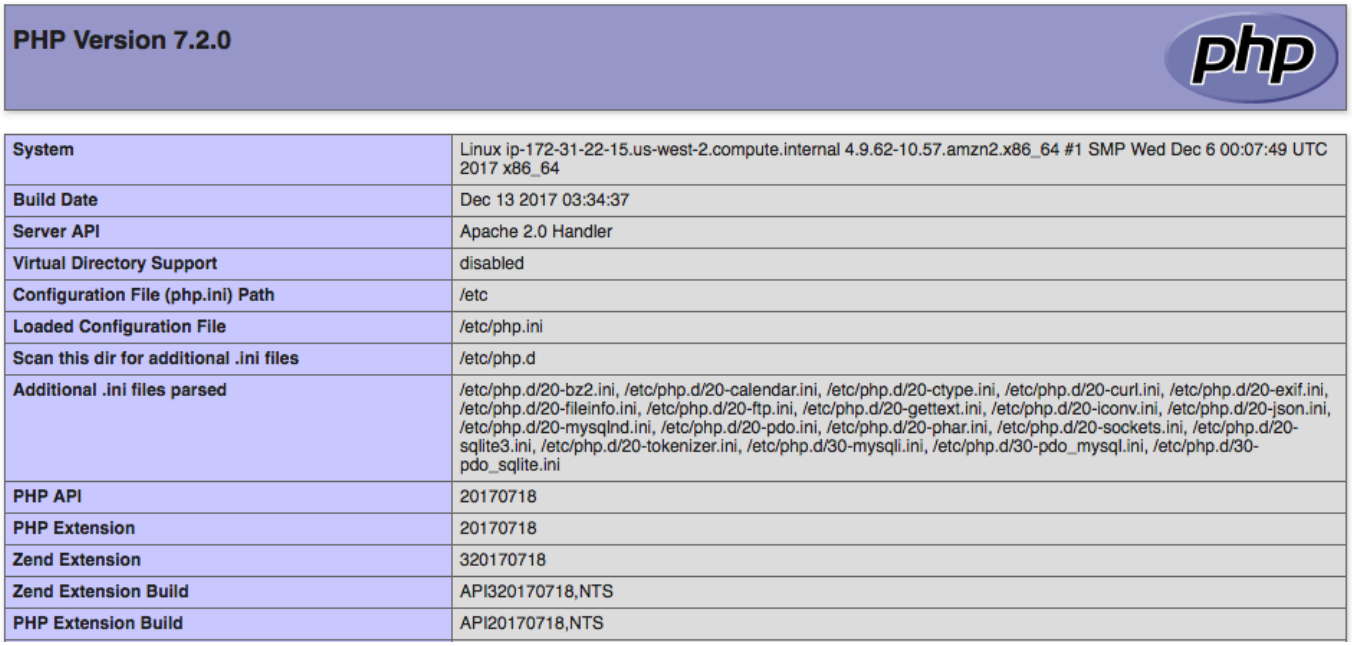
   ```
   [ec2-user ~]$ echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
   ```

   このコマンドを実行しようとしたときに「許可が拒否されました」というエラーが表示された場合は、ログアウトし、再度ログインして、ステップ 1: LAMP サーバーを準備する で設定した正しいグループ許可を取得します。

2. ウェブブラウザで、作成したファイルの URL を入力します。この URL は、インスタンスのパブリック DNS アドレスにスラッシュとファイル名を追加したものです。次に例を示します。

   ```
   http://my.public.dns.amazonaws.com/phpinfo.php
   ```

   PHP 情報ページが表示されるはずです。



PHP Version 7.2.0

System	Linux ip-172-31-22-15.us-west-2.compute.internal 4.9.62-10.57.amzn2.x86_64 #1 SMP Wed Dec 6 00:07:49 UTC 2017 x86_64
Build Date	Dec 13 2017 03:34:37
Server API	Apache 2.0 Handler
Virtual Directory Support	disabled
Configuration File (php.ini) Path	/etc
Loaded Configuration File	/etc/php.ini
Scan this dir for additional .ini files	/etc/php.d
Additional .ini files parsed	/etc/php.d/20-bz2.ini, /etc/php.d/20-calendar.ini, /etc/php.d/20-ctype.ini, /etc/php.d/20-curl.ini, /etc/php.d/20-exif.ini, /etc/php.d/20-fileinfo.ini, /etc/php.d/20-ftp.ini, /etc/php.d/20-gettext.ini, /etc/php.d/20-iconv.ini, /etc/php.d/20-json.ini, /etc/php.d/20-mysqlnd.ini, /etc/php.d/20-pdo.ini, /etc/php.d/20-phar.ini, /etc/php.d/20-sockets.ini, /etc/php.d/20-sqlite3.ini, /etc/php.d/20-tokenizer.ini, /etc/php.d/30-mysqli.ini, /etc/php.d/30-pdo_mysql.ini, /etc/php.d/30-pdo_sqlite.ini
PHP API	20170718
PHP Extension	20170718
Zend Extension	320170718
Zend Extension Build	API320170718,NTS
PHP Extension Build	API20170718,NTS

このページが表示されない場合は、前のステップで /var/www/html/phpinfo.php ファイルが正しく作成されたことを確認します。次のコマンドで、必要なパッケージがすべてインストールされたことを確認することもできます。2 番目の列のパッケージのバージョンが、この出力例に一致する必要はありません。

```
[ec2-user ~]$ sudo yum list installed httpd24 php72 mysql57-server php72-mysqlnd
Loaded plugins: priorities, update-motd, upgrade-helper
Installed Packages
httpd24.x86_64                          2.4.25-1.68.amzn1                   @amzn-
updates
mysql56-server.x86_64                   5.6.35-1.23.amzn1                   @amzn-
updates
php70.x86_64                            7.0.14-1.20.amzn1                   @amzn-
updates
php70-mysqlnd.x86_64                    7.0.14-1.20.amzn1                   @amzn-
updates
```

必要なパッケージのいずれかが出力に表示されていない場合は、sudo yum install ***package*** コマンドを使ってインストールします。

3. phpinfo.php ファイルを削除します。これは有用な情報であることもありますが、セキュリティ上の理由から、インターネット上で公表しないでください。

```
[ec2-user ~]$ rm /var/www/html/phpinfo.php
```

ステップ 3: データベースサーバーをセキュリティで保護する

MySQL サーバーのデフォルトのインストールには、テストおよび開発に役立ついくつかの機能がありますが、実稼働サーバーでは無効にするか削除する必要があります。mysql_secure_installation コマンドを使用すると、ルートパスワードを設定し、安全でない機能をインストールから削除する手順が案内されます。MySQL サーバーを使用する予定がない場合でも、この手順を実行することが推奨されます。

データベースサーバーをセキュリティで保護するには

1. MySQL サーバーを起動します。

```
[ec2-user ~]$ sudo service mysqld start
Initializing MySQL database:
...

PLEASE REMEMBER TO SET A PASSWORD FOR THE MySQL root USER !
...

Starting mysqld:                                           [  OK  ]
```

2. mysql_secure_installation を実行します。

```
[ec2-user ~]$ sudo mysql_secure_installation
```

   a. プロンプトが表示されたら、ルートアカウントのパスワードを入力します。

      i. 現在のルートパスワードを入力します。デフォルトでは、ルートアカウントにはパスワードが設定されていません。Enter キーを押します。

      ii. 「**Y**」と入力してパスワードを設定し、安全なパスワードを 2 回入力します。安全なパスワード作成の詳細については、「https://identitysafe.norton.com/password-generator/」を参照してください。このパスワードは必ず安全な場所に保管します。

      MySQL のルートパスワードの設定は、データベースを保護するための最も基本的な手段にすぎません。データベース駆動型アプリケーションを構築またはインストールする必要がある場合、通常はそのアプリケーションのデータベースサービスユーザーを作成します。ルートアカウントは、データベース管理以外には使用しないでください。

   b. 「**Y**」と入力して匿名ユーザーアカウントを削除します。

   c. 「**Y**」と入力してリモートルートログインを無効にします。

d. 「**Y**」と入力してテストデータベースを削除します。

e. 「**Y**」と入力して権限テーブルを再ロードし、変更を保存します。

3. (オプション) MySQL サーバーをすぐに使用する予定がない場合は、これを停止します。再び必要になったときには再起動できます。

```
[ec2-user ~]$ sudo service mysqld stop
Stopping mysqld:                                           [  OK  ]
```

4. (オプション) ブート時に毎回 MySQL サーバーを起動させる場合は、次のコマンドを入力します。

```
[ec2-user ~]$ sudo chkconfig mysqld on
```

これで、完全に機能する LAMP ウェブサーバーを設定しました。`/var/www/html` の Apache ドキュメントルートにコンテンツを追加する場合、そのコンテンツはインスタンスのパブリック DNS アドレスで表示できます。

ステップ 4: (オプション) phpMyAdmin をインストールする

pMyAdmin をインストールするには

phpMyAdmin は、EC2 インスタンスで MySQL データベースを表示して編集するために使用できる、ウェブベースのデータベース管理ツールです。Amazon Linux インスタンスで phpMyAdmin をインストールして設定には、以下の手順に従ってください。

> **Important**
>
> Apache で SSL/TLS を有効にしていない場合、LAMP サーバーへのアクセスに phpMyAdmin を使用することは推奨されません。そのようにすると、データベース管理者のパスワードや他のデータは、インターネット上を安全ではない状態で送信されます。開発者によるセキュリティ関連の推奨事項については、「Securing your phpMyAdmin installation」を参照してください。

> **Note**
>
> Amazon Linux パッケージの管理システムは、現在のところ PHP 7 環境における phpMyAdmin の自動インストールをサポートしていません。このチュートリアルでは、phpMyAdmin を手動でインストールする方法を説明します。

1. SSH を使用して EC2 インスタンスにログインします。
2. 必要な依存ファイルをインストールします。

```
[ec2-user ~]$ sudo yum install php72-mbstring.x86_64 -y
```

3. Apache を再起動します。

```
[ec2-user ~]$ sudo service httpd restart
Stopping httpd:                                            [  OK  ]
Starting httpd:                                            [  OK  ]
```

4. `/var/www/html` で Apache ドキュメントルートに移動します。

```
[ec2-user ~]$ cd /var/www/html
[ec2-user html]$
```

5. https://www.phpmyadmin.net/downloads で最新の phpMyAdmin リリース用のソースパッケージを選択します。ファイルディレクトリをインスタンスにダウンロードするには、次の例のようにリンクをコピーして wget コマンドに貼り付けます。

```
[ec2-user html]$ wget https://www.phpmyadmin.net/downloads/phpMyAdmin-latest-all-languages.tar.gz
```

6. phpMyAdmin フォルダを作成し、次のコマンドを使用してパッケージを展開します。

```
[ec2-user html]$ mkdir phpMyAdmin && tar -xvzf phpMyAdmin-latest-all-languages.tar.gz -C phpMyAdmin --strip-components 1
```

7. *phpMyAdmin-latest-all-languages.tar.gz* Tarball を削除します。

```
[ec2-user html]$ rm phpMyAdmin-latest-all-languages.tar.gz
```

8. (オプション) MySQL サーバーが実行中ではない場合は、今すぐ起動します。

```
[ec2-user ~]$ sudo service mysqld start
Starting mysqld:                                           [  OK  ]
```

9. ウェブブラウザで、phpMyAdmin のインストール URL を入力します。この URL は、インスタンスのパブリック DNS アドレス (または、パブリック IP アドレス) にスラッシュとインストールディレクトリを追加してものです。次に例を示します。

```
http://my.public.dns.amazonaws.com/phpMyAdmin
```

   phpMyAdmin ログインページが表示されます。

10. 前に作成した root ユーザー名と MySQL のルートパスワードを使って、phpMyAdmin インストールにログインします。

インストールは、サービス開始前に設定する必要があります。phpMyAdmin を設定するには、手動で設定ファイルを作成する、設定コンソールを使用する、またはその両方の方法を組み合わせることができます。

phpMyAdmin の使用に関する情報は、「phpMyAdmin ユーザーガイド」を参照してください。

トラブルシューティング

このセクションでは、新しい LAMP サーバーの設定時に発生する可能性がある一般的な問題の解決案を提供します。

ウェブブラウザを使用してサーバーに接続できません。

以下のチェックを行って、Apache ウェブサーバーが実行されていて、アクセス可能であるかどうかを確認します。

- ウェブサーバーが実行されていますか?

  httpd が有効であることは、次のコマンドを実行して確認できます。

  ```
  [ec2-user ~]$ chkconfig --list httpd
  httpd           0:off   1:off   2:on    3:on    4:on    5:on    6:off
  ```

  ここ,httpd、onランレベル 2、3、4、5 (見たいものです)。

  httpd プロセスが実行されていない場合は、ステップ 1: LAMP サーバーを準備する に記載されているステップを繰り返します。
- ファイアウォールは正しく設定されていますか?

  インスタンスのセキュリティグループに、ポート 80 での HTTP ラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

サーバーで実行するアプリケーションソフトウェアに、インストールされている PHP バージョンまたは他のソフトウェアとの互換性がありません。

このチュートリアルでは、最新バージョンの Apache HTTP Server、PHP、MySQL をインストールすることをお勧めします。追加の LAMP アプリケーションをインストールする前に、そのシステム

条件を調べて、インストール済みの環境と互換性があることを確認してください。最新バージョンの PHP がサポートされていない場合は、以前サポートされていた設定にダウングレードできます (全体的に安全です)。PHP は、複数のバージョンを並行してインストールすることもできます。これにより、最小限の労力で特定の互換性の問題を解決できます。複数バージョンの PHP の優先順位を設定する方法については、「Amazon Linux AMI 2016.09 Release Notes」を参照してください。

ダウングレード方法

このチュートリアルの十分にテストされた前バージョンでは、次のコア LAMP パッケージを推奨しています。

- httpd24
- php56
- mysql55-server
- php56-mysqlnd

このチュートリアルの始まりの推奨に従って最新のパッケージをインストールした場合は、まずこのパッケージとその他の依存ファイルを次のようにアンインストールする必要があります。

```
[ec2-user ~]$ sudo yum remove -y httpd24 php72 mysql57-server php72-mysqlnd perl-DBD-MySQL57
```

次に、代替環境をインストールします。

```
[ec2-user ~]$ sudo yum install -y  httpd24 php56 mysql55-server php56-mysqlnd
```

後で推奨環境にアップグレードすることを決定した場合は、まずカスタマイズされたパッケージと依存関係を削除する必要があります。

```
[ec2-user ~]$ sudo yum remove -y  httpd24 php56 mysql55-server php56-mysqlnd perl-DBD-MySQL56
```

これで、前に説明した最新のパッケージをインストールできます。

関連トピック

インスタンスへのファイルの転送、またはウェブサーバーへの WordPress ブログのインストールの詳細については、次のドキュメントを参照してください。

- WinSCP を使用した Linux インスタンスへのファイルの転送
- SCP クライアントを使用した Linux インスタンスへのファイルの転送
- Amazon Linux 2 での WordPress ブログのホスト

このチュートリアルで使用されているコマンドおよびソフトウェアの詳細については、次のウェブページを参照してください。

- Apache ウェブサーバー: http://httpd.apache.org/
- MySQL データベースサーバー: http://www.mysql.com/
- PHP プログラミング言語: http://php.net/
- chmod コマンド: https://en.wikipedia.org/wiki/Chmod
- chown コマンド: https://en.wikipedia.org/wiki/Chown

ウェブサーバーのドメイン名の登録、または、既存のドメイン名をこのホストに移す方法についての詳細は、『Amazon Route 53 デベロッパーガイド』の「Amazon Route 53 のドメインとサブドメインの作成と移行」を参照してください。

SSL/TLS の設定

このセクションでは、Amazon EC2 インスタンスで実行されているウェブサーバーで SSL/TLS を設定する方法を説明します。

> **Note**
>
> AL2023 SSL/TLS チュートリアルについては、「Amazon Linux 2023 ユーザーガイド」の「チュートリアル: AL2023 で SSL/TLS を設定する」を参照してください。

以下での SSL/TLS の設定

- Amazon Linux 2 での SSL/TLS の設定
- Amazon Linux での SSL/TLS の設定

Amazon Linux 2 での SSL/TLS の設定

Secure Sockets Layer/Transport Layer Security (SSL/TLS) は、ウェブサーバーとウェブクライアントの間に、転送中のデータが傍受されないように保護する、暗号化されたチャネルを確立します。このチュートリアルでは、Amazon Linux 2 と Apache ウェブサーバーを使用して EC2 インスタンスに、SSL/TLS のサポートを手動で追加する方法を説明します。このチュートリアルでは、ロードバランサーを使用していないことを前提としています。Elastic Load Balancing を使用している場合は、代わりに AWS Certificate Manager の証明書を使用して、ロードバランサーで SSL オフロードを設定できます。

歴史的経緯から、ウェブの暗号化は、単純に SSL と呼ばれることが少なくありません。ウェブブラウザでは今でも SSL がサポートされていますが、後継プロトコルである TLS プロトコルの方が攻撃を受けにくくなります。Amazon Linux 2 では、すべてのバージョンの SSL でデフォルトでサーバー側のサポートを無効にしています。セキュリティ標準化団体は、TLS 1.0 は安全でないとみなしています。TLS 1.0 および TLS 1.1 は、2021 年 3 月に正式に非推奨になりました。このチュートリアルは、TLS 1.2 を有効にすることを前提としたガイダンスです。TLS 1.3 は 2018 年に完成し、基盤となる TLS ライブラリ (このチュートリアルでは OpenSSL) がサポートされ、かつ、有効である限り、Amazon Linux 2 で利用できます。クライアントは 2023 年 6 月 28 日までに TLS 1.2 以降をサポートしている必要があります。最新の暗号化基準の詳細については、「RFC 7568」および「RFC 8446」を参照してください。

このチュートリアルでは、現代のウェブ暗号化を単に TLS と呼びます。

> **Important**
>
> これらの手順は Amazon Linux 2 で使用するためのものです。また、新しい Amazon EC2 インスタンスを使用して開始するものと仮定します。異なるディストリビューションを実行している EC2 インスタンス、または古いバージョンの Amazon Linux 2 を実行しているインスタンスをセットアップしようとすると、このチュートリアルの一部の手順が上手くいかないことがあります。Amazon Linux AMI については、「Amazon Linux での SSL/TLS の設定」を参照してください。Ubuntu については、Ubuntu 上の OpenSSL に関するコミュニティドキュメントを参照してください。Red Hat Enterprise Linux については、以下を参照してください。Apache HTTP Web サーバーの設定。その他のディストリビューションについては、それぞれのドキュメントを参照してください。

> **Note**
>
> あるいは、AWS Nitro Enclaves 向けの AWS Certificate Manager (ACM) を使用することもできます。これは、パブリックおよびプライベートの SSL/TLS 証明書を(AWS Nitro Enclaves を使用している Amazon EC2 インスタンスで実行されている) ウェブアプリケーションおよびサーバーで使用できるようにする、エンクレーブアプリケーションです。Nitro Enclaves は、SSL/TLS 証明書やプライベートキーなどの機密性の高いデータを保護し、安全に処理するために、分離されたコンピューティング環境を作成できる Amazon EC2 の機能です。
> Nitro Enclaves 向け ACM では、Amazon EC2 Linux インスタンスで実行する nginx を使用することで、プライベートキーの作成、証明書とプライベートキーの配布、および証明書の更新を実行します。
> Nitro Enclaves 向け ACM を使用するには、エンクレーブ対応の Linux インスタンスを使用する必要があります。
> 詳細については、AWS Nitro Enclaves ユーザーガイドの「What is AWS Nitro Enclaves?」および「AWS Certificate Manager for Nitro Enclaves」を参照してください。

コンテンツ

前提条件

このチュートリアルを開始する前に、次のステップを完了してください。

- EBS-backed Amazon Linux 2 インスタンスを起動します。詳細については、「ステップ 1: インスタンスを起動する」を参照してください。
- インスタンスが以下の TCP ポートで接続を受け付けるようにセキュリティグループを設定します。
  - SSH (ポート 22)
  - HTTP (ポート 80)
  - HTTPS (ポート 443)

詳細については、「Linux インスタンス用のインバウンドトラフィックの承認」を参照してください。

- Apache ウェブサーバーをインストールします。手順については、「チュートリアル: Amazon Linux 2 に LAMP ウェブサーバーをインストールする」を参照してください。必要なのは httpd パッケージおよび対応する従属コンポーネントのみです。PHP および MariaDB に関連する手順は無視してかまいません。
- ウェブサイトの識別と認証を行うため、TLS の公開鍵基盤 (PKI) ではドメインネームシステム (DNS) を使用します。EC2 インスタンスを使用してパブリックウェブサイトをホストするには、ウェブサーバーのドメイン名を登録するか、既存のドメイン名を Amazon EC2 ホストに移す必要があります。これについては、ドメイン登録および DNS ホスティングに関するサードパーティのサービスが多数存在します。Amazon Route 53 を使用することもできます。

ステップ 1: サーバーで TLS を有効にする

オプション: オートメーション を使用してこのチュートリアルを完了する

以下のタスクを行う代わりに AWS Systems Manager オートメーションを使用してこのチュートリアルを完了するには、オートメーションドキュメントを実行します。

この手順では、自己署名のデジタル証明書を使用して、Amazon Linux 2 で TLS をセットアップします。

> **Note**
>
> 自己署名証明書はテスト用であり、本稼働環境では使用できません。インターネットに自己署名証明書を公開すると、サイトへの訪問者にセキュリティ警告が表示されます。

サーバーで TLS を有効にするには

1. インスタンスに接続し、Apache が実行されていることを確認します。

   ```
   [ec2-user ~]$ sudo systemctl is-enabled httpd
   ```

   返される値が「enabled」でない場合、Apache を起動し、システムブート時に毎回起動されるように設定します。

```
[ec2-user ~]$ sudo systemctl start httpd && sudo systemctl enable httpd
```

2. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。この処理には数分かかりますが、最新の更新とバグ修正を確実に適用することが重要です。

> **Note**
>
> -y オプションを指定すると、確認メッセージを表示せずに更新をインストールします。インストール前に更新を検査する場合は、このオプションを省略できます。

```
[ec2-user ~]$ sudo yum update -y
```

3. これでインスタンスが最新状態になったため、Apache モジュール mod_ssl をインストールして TLS サポートを追加します。

```
[ec2-user ~]$ sudo yum install -y mod_ssl
```

次のファイルがインスタンスに作成されました。このファイルは、セキュアサーバーの設定とテスト用の証明書の作成に使用します。

- `/etc/httpd/conf.d/ssl.conf`

  mod_ssl の設定ファイル。このファイルには、暗号化キーと証明書の場所、許可する TLS プロトコル、受け入れる暗号化アルゴリズムを Apache に指示するディレクティブが含まれています。

- `/etc/pki/tls/certs/make-dummy-cert`

  サーバーホスト用の自己署名 X.509 証明書とプライベートキーを生成するためのスクリプト。この証明書は、TLS を使用するように Apache が正しくセットアップされているかどうかをテストする場合に役立ちます。アイデンティティは証明されないため、本稼働環境では使用しないでください。本稼働環境で使用すると、ウェブブラウザで警告が表示されます。

4. テスト用に自己署名のダミー証明書とキーを生成するためのスクリプトを実行します。

```
[ec2-user ~]$ cd /etc/pki/tls/certs
sudo ./make-dummy-cert localhost.crt
```

/etc/pki/tls/certs/ ディレクトリに新しいファイル localhost.crt が生成されます。指定されたファイル名は、SSLCertificateFile の /etc/httpd/conf.d/ssl.conf ディレクティブで割り当てたデフォルトの名前と一致します。

このファイルには、自己署名証明書と証明書のプライベートキーのいずれも含まれます。Apache では、証明書とキーを PEM 形式にする必要があります。これは、次の短縮化された例のように、"BEGIN" 行と "END" 行で囲まれた Base64 エンコードの ASCII 文字で構成されます。

```
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQD2KKx/8Zk94m1q
3gQMZF9ZN66Ls19+3tHAgQ5Fpo9KJDhzLjOOCI8u1PTcGmAah5kEitCEc0wzmNeo
BCl0wYR6G0rGaKtK9Dn7CuIjvubtUysVyQoMVPQ97ldeakHWeRMiEJFXg6kZZ0vr
GvwnKoMh3DlK44D9dX7IDua2PlYx5+eroA+1Lqf32ZSaAO0bBIMIYTHigwbHMZoT
...
56tE7THvH7vOEf4/iUOsIrEzaMaJ0mqkmY1A70qQGQKBgBF3H1qNRNHuyMcPODFs
27hDzPDinrquSEvoZIggkDMlh2irTiipJ/GhkvTpoQlv0fK/VXw8vSgeaBuhwJvS
LXU9HvYq0U6O4FgD3nAyB9hI0BE13r1HjUvbjT7moH+RhnNz6eqqdscCS09VtRAo
4QQvAqOa8UheYeoXLdWcHaLP
-----END PRIVATE KEY-----

-----BEGIN CERTIFICATE-----
MIIEazCCA1OgAwIBAgICWxQwDQYJKoZIhvcNAQELBQAwgbExCzAJBgNVBAYTAi0t
MRIwEAYDVQQIDAlTb21lU3RhdGUxETAPBgNVBAcMCFNvbWVDaXR5MRkwFwYDVQQK
DBBTb21lT3JnYW5pemF0aW9uMR8wHQYDVQQLDBZTb21lT3JnYW5pemF0aW9uYWxV
bml0MRkwFwYDVQQDDBBpcC0xNzItMzEtMjAtMjM2MSQwIgYJKoZIhvcNAQkBFhVy
...
z5rRUE/XzxRLBZOoWZpNWTXJkQ3uFYH6s/sBwtHpKKZMzOvDedREjNKAvk4ws6F0
CuIjvubtUysVyQoMVPQ97ldeakHWeRMiEJFXg6kZZ0vrGvwnKoMh3DlK44D9dlU3
WanXWehT6FiSZvB4sTEXXJN2jdw8g+sHGnZ8zCOsclknYhHrCVD2vnBlZJKSZvak
3ZazhBxtQSukFMOnWPP2a0DMMFGYUHOd0BQE8sBJxg==
-----END CERTIFICATE-----
```

ファイル名および拡張子は利便性のためであり、機能には影響しません。例えば、cert.crt または cert.pem などのファイル名で証明書を呼び出すことができます。ただし、ssl.conf ファイルの関連ディレクティブが同じ名前を使用している場合に限ります。

> **Note**
>
> デフォルトの TLS ファイルを独自にカスタマイズしたファイルに置き換える場合は、PEM 形式であることを確認してください。

5. ルートユーザーとしてお好みのテキストエディタ (vim、nanoなど) を使用して `/etc/httpd/conf.d/ssl.conf` ファイルを開き、次の行をコメントアウトします。ダミーの自己署名証明書にも同じキーが含まれているためです。次のステップに進む前にこの行をコメントアウトしないと、Apache サービスは起動に失敗します。

```
SSLCertificateKeyFile /etc/pki/tls/private/localhost.key
```

6. Apache を再起動します。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

> **Note**
>
> 前述のとおり、TCP 443 番ポートが EC2 インスタンスでアクセス可能であることを確認してください。

7. Apache ウェブサーバーではポート 443 経由で HTTPS (セキュア HTTP) がサポートされるようになっています。これをテストするには、ブラウザの URL バーに、**`https://`** というプレフィックスを指定して、EC2 インスタンスの IP アドレスまたは完全修飾ドメイン名を入力します。

   信頼されていない自己署名ホスト証明書を使用してサイトに接続しようとしているため、ブラウザには一連のセキュリティ警告が表示されることがあります。これの警告を無視し、サイトに進みます。

   サーバーで TLS を正しく設定できていれば、Apache のデフォルトのテストページが開きます。これで、ブラウザとサーバーの間でやり取りされるすべてのデータが暗号化されるようになります。

> **Note**
>
> サイト訪問者に対して警告画面が表示されないようにするには、暗号化だけではなく、サイト所有者のパブリック認証を行うための信頼された CA 署名証明書を取得する必要があります。

ステップ 2: CA 署名証明書を取得する

CA 署名証明書を取得するには、次の手順に従います。

- プライベートキーから証明書署名リクエスト (CSR) を作成します。
- 作成した CSR を認証機関 (CA) に送信します。
- 署名付きホスト証明書を入手する
- 証明書を使用するように Apache を設定します

自己署名 TLS X.509 ホスト証明書は、暗号化技術上は CA 署名証明書と同じです。これらの相違は数学的なものではなく、社会的なものです。CA では、最低でもドメイン所有権を検証してから申請者に証明書を発行することを保証しています。そのため、各ウェブブラウザには、ブラウザベンダーが信頼する CA のリストが含まれています。X.509 証明書は主に、プライベートサーバーキーに対応するパブリックキーと、このパブリックキーに暗号で関連付けられている CA による署名で構成されています。HTTPS 経由でブラウザがウェブサーバーに接続すると、サーバーは、信頼された CA のリストをブラウザが確認できるように、証明書を提示します。Signerがリストに含まれている場合や、他の信頼された署名者の信頼チェーンを通じてアクセス可能である場合、ブラウザはサーバーと、高速暗号化データチャネルのネゴシエーションを行い、ページをロードします。

証明書には、リクエストの確認作業が必要であり、一般的に費用がかかるため、各社を比較することをお勧めします。いくつかの CA では、基本レベル証明書が無料で提供されます。これらの CA で最も注目すべきは Let's Encrypt プロジェクトです。このプロジェクトでは、証明書の作成および更新プロセスの自動化もサポートしています。Let's Encrypt 証明書の使用の詳細については、「Certbot の取得」を参照してください。

商業グレードのサービスを提供する予定がある場合は、AWS Certificate Manager は良い選択肢です。

ホスト証明書の基盤にはキーがあります。2019 年時点で、政府および業界グループは、2030 年まで、ドキュメントを保護するための RSA キーに 2048 ビットの最小キー (モジュロ) サイズを使用す

ることを推奨しています。Amazon Linux 2 で OpenSSL によって生成されるデフォルトのモジュラスサイズは 2048 ビットです。つまり、CA 署名証明書に適しています。次の手順では、モジュラスサイズを大きくする、別の暗号化アルゴリズムを使用するなど、キーのカスタマイズが必要な場合のオプションのステップを提供しています。

> **Important**
>
> CA 署名ホスト証明書を取得するための手順は、登録およびホスト済みの DNS ドメインを所有している場合を除き、使用しません。

CA 署名証明書を取得するには

1. インスタンスに接続して、/etc/pki/tls/private/ に移動します。サーバーの TLS 用プライベートキーは、このディレクトリに格納されます。既存のホストキーを使用して CSR を生成する場合は、ステップ 3 に進みます。
2. (オプション) 新しいプライベートキーを生成します。キー設定のいくつかのサンプルを次に示します。生成されたキーのどれもウェブサーバーで機能しますが、実装されるセキュリティの強度とタイプはそれぞれ異なります。

   - 例 1: デフォルトの RSA ホストキーを作成します。結果として生成されるファイル **custom.key** が、2048 ビットの RSA プライベートキーです。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key
     ```

   - 例 2: これより大きなモジュラサイズを使用して、より強力な RSA キーを作成します。結果として生成されるファイル **custom.key** が、4096 ビットの RSA プライベートキーです。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key 4096
     ```

   - 例 3: パスワードで保護された 4096 ビット暗号化 RSA キーを作成します。結果のファイル、**custom.key** は、AES-128 暗号で暗号化された 4096 ビットの RSA プライベートキーです。

     > **Important**
     >
     > キーを暗号化するとセキュリティを強化できますが、暗号化キーにはパスワードが必要であるため、暗号化に依存するサービスを自動的に開始することはできません。こ

のキーを使用するたびに、SSH 接続でパスワード (前述の例では、"abcde12345") を指定する必要があります。

```
[ec2-user ~]$ sudo openssl genrsa -aes128 -passout pass:abcde12345 -out
 custom.key 4096
```

- 例 4: 非 RSA 暗号を使用してキーを作成します。RSA 暗号化は、2 つの大きな素数の積に基づくパブリックキーのサイズのために、比較的遅くなる可能性があります。ただし、非 RSA 暗号化方式を使用する TLS 用のキーを作成することも可能です。同等レベルのセキュリティを提供する場合は、楕円曲線の計算に基づいたキーのほうが小さく計算処理も高速です。

  ```
  [ec2-user ~]$ sudo openssl ecparam -name prime256v1 -out custom.key -genkey
  ```

  結果は、prime256v1 (OpenSSL でサポートされる "名前付き曲線") を使用した 256 ビットの楕円曲線プライベートキーです。暗号化強度は (NIST によると) 2048 ビットの RSA キーよりやや優れています。

  > **Note**
  >
  > すべての CA で、楕円曲線ベースのキーに対して RSA キーと同じレベルのサポートが提供されているわけではありません。

新しいプライベートキーには、制限の厳しい所有権とアクセス権を設定します (所有者 = root、グループ = root、所有者のみの読み取り/書き込み)。コマンドは次の例のようになります。

```
[ec2-user ~]$ sudo chown root:root custom.key
[ec2-user ~]$ sudo chmod 600 custom.key
[ec2-user ~]$ ls -al custom.key
```

上記のコマンドにより、次のような結果が得られます。

```
-rw------- root root custom.key
```

適切なキーを作成し、設定できたら、CSR を作成できます。

3. 好みのキーを使用して CSR を作成します。次の例では **custom.key** を使用しています。

```
[ec2-user ~]$ sudo openssl req -new -key custom.key -out csr.pem
```

OpenSSL によりダイアログが開かれ、次の表に示されている情報の入力が求められます。基本的なドメイン検証済みホスト証明書については、[共通名] 以外のフィールドはすべてオプションです。

名前	説明	例
国名	2 文字の ISO 略称 (国名コード)。	US (= 米国)
州名	あなたが所属する組織の所在地の州または県。省略不可です。	ワシントン
市区町村	市など、組織の場所。	シアトル
組織名	組織の正式名称。組織名は、省略不可です。	Example Corp
部門名	組織に関する追加情報 (存在する場合)。	Example Dept
共通名	この値は、ユーザーがブラウザに入力する必要のあるウェブアドレスと正確に一致します。通常、これはプレフィックス付きのホスト名またはエイリアスによるドメイン名 (**www.example.com** の形式) を意味します。自己署名証明書を使用し、DNS 解決なしでテストを行う場合、共通名の構成要素はホスト名のみになる場合があります。CA では、***.example.com** などのワイルドカード名を許容する、よりコストの高い証明書も用意されています。	www.example.com
E メールアドレス	サーバー管理者の E メールアドレス。	someone@example.com

最後に、OpenSSL により、オプションのチャレンジパスワードが求められます。このパスワードは CSR と、ユーザーと CA の間のトランザクションのみに適用されるため、このフィールド

と、もう 1 つのオプションフィールドである、オプションの会社名については、CA の推奨事項に従ってください。CSR のチャレンジパスワードは、サーバー操作には影響しません。

結果として生成されるファイル **csr.pem** には、パブリックキー、パブリックキーのデジタル署名、入札したメタデータが含まれています。

4. CA に CSR を送信します。この作業は通常、テキストエディタで CSR ファイルを開く動作と、内容をウェブフォームにコピーする動作で構成されています。このとき、証明書に適用する 1 つ以上のサブジェクト代替名 (SAN) を指定するように求められることがあります。共通名が **www.example.com** の場合、有効な SAN は **example.com** になります (逆も同様です)。サイトへの訪問者がこれら名前のいずれかを入力すると、エラーなしの接続が提示されます。CA のウェブフォームで許可される場合は、SAN のリストに共通名を含めます 一部の CA では自動的に含められます。

   リクエストが承認されると、CA によって署名された新しいホスト証明書が届きます。CA の信頼チェーンを完成するために必要な、追加の証明書が含まれている中間証明書ファイルをダウンロードするよう指示されることもあります。

   > **Note**
   >
   > 多様な用途向けに複数の形式のファイルを送信してくる CA もあります。このチュートリアルでは、PEM 形式の証明書ファイルのみ使用してください。PEM 形式のファイルには通常、`.pem` または `.crt` ファイル拡張子が使用されます (ただし、常にこれらの拡張子が使用されるわけではありません)。どのファイルを使用すべきかわからない場合は、テキストエディタでファイルを開き、以下の行で始まる 1 つ以上のブロックを含むファイルを見つけてください。
   >
   > ```
   > - - - - -BEGIN CERTIFICATE - - - - -
   > ```
   >
   > ファイルの末尾は次のような行になっている必要があります。
   >
   > ```
   > - - - -END CERTIFICATE - - - - -
   > ```
   >
   > 以下に示すように、コマンドラインでファイルをテストすることもできます。
   >
   > ```
   > [ec2-user certs]$ openssl x509 -in certificate.crt -text
   > ```

これらの行がファイルに表示されていることを確認してください。.p7b、.p7c、または類似のファイル拡張子で終了するファイルは使用しないでください。

5. 新しい CA 署名証明書と任意の中間証明書を /etc/pki/tls/certs ディレクトリに配置します。

> **Note**
>
> EC2 インスタンスに新しい証明書をアップロードする方法は複数ありますが、最も簡単でわかりやすい方法は、テキストエディタ (vi、nano、またはメモ帳など) をローカルコンピュータとインスタンスの両方で開いて、両者の間でファイルの内容をコピーして貼り付けることです。EC2 インスタンス内でこれらの操作を実行する際には、root [sudo] アクセス許可が必要です。こうすることで、許可やパスに問題があるかどうかをすぐに確認できます。ただし、内容をコピーする際に行を追加したり、内容を変更したりしないでください。

/etc/pki/tls/certs ディレクトリの中から、ファイルの所有者、グループ、アクセス権の設定が制限の厳しい Amazon Linux 2 のデフォルト (所有者 = root、グループ = root、所有者のみの読み込み/書き込み可) と一致することを確認します。以下の例では、使用するコマンドを示しています。

```
[ec2-user certs]$ sudo chown root:root custom.crt
[ec2-user certs]$ sudo chmod 600 custom.crt
[ec2-user certs]$ ls -al custom.crt
```

これらのコマンドによって、次の結果が得られます。

```
-rw------- root root custom.crt
```

中間証明書ファイルのアクセス権は、比較的厳しくありません (所有者 = root、グループ = root、所有者による書き込み可、グループによる読み取り可、その他による読み取り可)。以下の例では、使用するコマンドを示しています。

```
[ec2-user certs]$ sudo chown root:root intermediate.crt
[ec2-user certs]$ sudo chmod 644 intermediate.crt
[ec2-user certs]$ ls -al intermediate.crt
```

これらのコマンドによって、次の結果が得られます。

```
-rw-r--r-- root root intermediate.crt
```

6. CSR の作成に使用したプライベートキーを /etc/pki/tls/private/ ディレクトリに配置します。

> **Note**
>
> EC2 インスタンスにカスタムキーをアップロードする方法は複数ありますが、最も簡単でわかりやすい方法は、テキストエディタ (vi、nano、メモ帳など) をローカルコンピュータとインスタンスの両方で開いて、両者の間でファイルの内容をコピーして貼り付けることです。EC2 インスタンス内でこれらの操作を実行する際には、root [sudo] アクセス許可が必要です。こうすることで、許可やパスに問題があるかどうかをすぐに確認できます。ただし、内容をコピーする際に行を追加したり、内容を変更したりしないでください。

/etc/pki/tls/private ディレクトリの中から、次のコマンドを使用してファイルの所有者、グループ、アクセス許可の設定が制限の厳しい Amazon Linux 2 のデフォルト (所有者 = root、グループ = root、所有者のみの読み込み/書き込み可) と一致することを確認します。

```
[ec2-user private]$ sudo chown root:root custom.key
[ec2-user private]$ sudo chmod 600 custom.key
[ec2-user private]$ ls -al custom.key
```

これらのコマンドによって、次の結果が得られます。

```
-rw------- root root custom.key
```

7. 新しい証明書とキーファイルに合わせるには、/etc/httpd/conf.d/ssl.conf を編集します。

   a. CA 署名のホスト証明書のパスとファイル名を Apache の SSLCertificateFile ディレクティブで指定します。

      ```
      SSLCertificateFile /etc/pki/tls/certs/custom.crt
      ```

b. 中間証明書ファイル (この例では `intermediate.crt`) を受け取ったら、Apache の `SSLCACertificateFile` ディレクティブを使用して、次のファイルのパスとファイル名を指定します。

```
SSLCACertificateFile /etc/pki/tls/certs/intermediate.crt
```

> **Note**
>
> 一部の CA では、ホスト証明書と中間証明書を組み合わせて 1 つのファイルを作成するため、この `SSLCACertificateFile` ディレクティブは必要ありません。CA が提供している手順を参照してください。

c. プライベートキー (この例では `custom.key`) のパスとファイル名を Apache の `SSLCertificateKeyFile` ディレクティブで指定します。

```
SSLCertificateKeyFile /etc/pki/tls/private/custom.key
```

8. `/etc/httpd/conf.d/ssl.conf` を保存して、Apache を再起動します。

```
[ec2-user ~]$ sudo systemctl restart httpd
```

9. サーバーをテストするには、ブラウザの URL バーにドメイン名を入力し、プレフィックス `https://` を指定します。ブラウザによって、エラーが生成されることなく、HTTPS 経由でテストページがロードされます。

ステップ 3: セキュリティ設定をテストして強化する

TLS が運用可能になりパブリックに公開されたら、実際の安全性をテストする必要があります。セキュリティセットアップの詳細な分析を無料で行うことのできる Qualys SSL Labs などのオンラインサービスを使用すると簡単です。その結果に基づき、受け入れるプロトコル、優先する暗号化方式、除外する暗号化方式を制御することによって、デフォルトのセキュリティ設定を強化するかどうかを決定できます。詳細については、「Qualys のスコアの計算方法」を参照してください。

> **Important**
>
> サーバーのセキュリティを確保するには、実際のテストが非常に重要です。小さな設定エラーによって、深刻なセキュリティ侵害やデータの損失が生じる可能性があります。調査

や新たな脅威に応じて、推奨されるセキュリティ管理方法は常に変化するため、適切なサーバー管理を行うには、定期的なセキュリティ監査が不可欠です。

Qualys SSL Labs のサイトで、サーバーの完全修飾ドメイン名を **www.example.com** という形式で入力します。約 2 分後に、サイトに関するグレード (A から F) と、結果の詳細な内訳が届きます。以下の表は、Amazon Linux 2 でのデフォルトの Apache 設定およびデフォルトの Certbot 証明書と同じ設定を使用しているドメインのレポートをまとめたものです。

総合評価	B
証明書	100%
プロトコルサポート	95%
キー交換	70%
暗号強度	90%

概要は設定がほとんど正常であることを示していますが、詳細レポートでは、いくつかの潜在的な問題が指摘されています。重大度の高い順に以下に示します。

RC4 暗号は、特定の古いブラウザでの使用がサポートされています。暗号は、暗号化アルゴリズムの計算の中核です。TLS データストリームの暗号化に使用される高速の暗号化方式である RC4 は、いくつかの重大な脆弱性を持つことで知られています。従来のブラウザをサポートするもっともな理由がない限り、この暗号化方式を無効にする必要があります。

✗旧バージョンの TLS がサポートされています。設定では TLS 1.0 (すでに廃止されています) と TLS 1.1 (廃止予定) がサポートされています。2018 年以降は、TLS 1.2 のみ推奨されています。

前方秘匿性は完全にサポートされていません。前方秘匿性は、プライベートキーから派生した一時 (エフェメラル) セッションキーを使用して暗号化を行う、アルゴリズムの機能です。これは、攻撃者がウェブサーバーの長期的なプライベートキーを所有していても、HTTPS データを復号できないことを意味します。

TLS 設定を修正し、将来への対応性を確保するには

1. 設定ファイル `/etc/httpd/conf.d/ssl.conf` を開き、行頭に # を付けて以下の行をコメントアウトしてください。

```
#SSLProtocol all -SSLv3
```

2. 次のディレクティブを追加します。

```
#SSLProtocol all -SSLv3
SSLProtocol -SSLv2 -SSLv3 -TLSv1 -TLSv1.1 +TLSv1.2
```

このディレクティブにより、SSL バージョン 2、3、および TLS バージョン 1.0、1.1 が明示的に無効化されます。これで、サーバーでは、TLS 1.2 以外を使用した、クライアントとの暗号化された接続の受け入れが拒否されます。ディレクティブに含める指定が多くなるほど、サーバーの動作に対する設定内容が明確に伝わります。

> **Note**
>
> このようにして、TLS バージョン 1.0 および 1.1 を無効にすると、ごく一部の古くなったウェブブラウザによるサイトへのアクセスがブロックされるようになります。

許可された暗号のリストを変更するには

1. 設定ファイル /etc/httpd/conf.d/ssl.conf で、**SSLCipherSuite** ディレクティブを含むセクションを探し、行頭に # を付けて既存の行をコメントアウトします。

```
#SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
```

2. 明示的な暗号スイートと、前方秘匿性を優先し、安全でない暗号を禁止する暗号順序を指定します。ここで使用される SSLCipherSuite ディレクティブは、Mozilla SSL Configuration Generatorの出力に基づいています。これは、お客様のサーバーで実行されている特定のソフトウェアに合わせて TLS 設定を調整します。(詳細については、Mozilla の有益なリソース「Security/Server Side TLS」を参照してください。) まず、以下のコマンドの出力を使用して、Apache と OpenSSL のバージョンを確認します。

```
[ec2-user ~]$ yum list installed | grep httpd

[ec2-user ~]$ yum list installed | grep openssl
```

例えば、返された情報が Apache 2.4.34 および OpenSSL 1.0.2 である場合、これをジェネレーターに入力します。"最新" 互換性モデルを選択すると、SSLCipherSuite ディレクティブが作

成されます。このディレクティブは、積極的にセキュリティを適用しますが、ほとんどのブラウザで使用できます。ソフトウェアで最新互換性モデルがサポートされていない場合は、ソフトウェアを更新するか、"中間" の構成を選択します。

```
SSLCipherSuite ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:
ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:
ECDHE-ECDSA-AES256-SHA384:ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-
RSA-AES128-SHA256
```

選択された暗号化方式の名前には、ECDHE が含まれています (Elliptic Curve Diffie-Hellman Ephemeral の略語です)。ephemeral は前方秘匿性を示します。また、これらの暗号化方式では、RC4 はサポートされていません。

デフォルトや、内容が見えない簡単なディレクティブに依存するのではなく、暗号化方式の明示的なリストを使用することをお勧めします。

生成されたディレクティブを `/etc/httpd/conf.d/ssl.conf` にコピーします。

> **Note**
>
> ここでは読みやすくするために数行に分けて示していますが、このディレクティブは、`/etc/httpd/conf.d/ssl.conf` にコピーする際に、暗号化方式名の間をコロンのみ (スペースなし) で区切り、1 行に指定する必要があります。

3. 最後に、次の行について、行頭の # を削除してコメント解除します。

```
#SSLHonorCipherOrder on
```

このディレクティブは、(この場合) 前方秘匿性をサポートするものも含めて、ランクの高い暗号化方式を優先するようサーバーに強制します。このディレクティブが有効になると、サーバーは、セキュリティの弱い暗号化方式に戻る前に、セキュリティが強力な接続を確立しようとします。

これらの手順がいずれも完了したら、変更内容を `/etc/httpd/conf.d/ssl.conf` に保存し、Apache を再起動します。

Qualys SSL Labs でドメインをもう一度テストすると、RC4 脆弱性やその他の警告は解決し、次のようなサマリレポートが出力されます。

総合評価	A
証明書	100%
プロトコルサポート	100%
キー交換	90%
暗号強度	90%

OpenSSL の更新ごとに、新しい暗号化方式が導入され古い暗号化方式のサポートが削除されます。EC2 の Amazon Linux 2 インスタンスを最新の状態に維持し、セキュリティに関する OpenSSL からの告知に注意して、技術分野の報道でセキュリティ面の新しい脆弱性に関するレポートを警戒してください。

トラブルシューティング

- パスワードを指定しないと Apache ウェブサーバーが起動しません

  これは、パスワードで保護された暗号化プライベート サーバー キーをインストールした場合は正常な動作です。

  暗号化とパスワードの要件をキーから削除できます。デフォルトディレクトリに `custom.key` という暗号化プライベート RSA キーがあり、そのパスワードが **abcde12345** であるとすると、EC2 インスタンスで次のコマンドを実行し、このキーの非暗号化バージョンを生成してください。

```
[ec2-user ~]$ cd /etc/pki/tls/private/
[ec2-user private]$ sudo cp custom.key custom.key.bak
[ec2-user private]$ sudo openssl rsa -in custom.key -passin pass:abcde12345 -out
 custom.key.nocrypt
[ec2-user private]$ sudo mv custom.key.nocrypt custom.key
[ec2-user private]$ sudo chown root:root custom.key
[ec2-user private]$ sudo chmod 600 custom.key
[ec2-user private]$ sudo systemctl restart httpd
```

パスワードが求められずに Apache が起動するようになります。

- sudo yum install -y mod_ssl を実行するとエラーが発生します。

  SSL に必要なパッケージをインストールすると、次のようなエラーが表示されることがあります。

  ```
  Error: httpd24-tools conflicts with httpd-tools-2.2.34-1.16.amzn1.x86_64
  Error: httpd24 conflicts with httpd-2.2.34-1.16.amzn1.x86_64
  ```

  これは、通常 EC2 インスタンスが Amazon Linux 2 を実行していないことを意味します。このチュートリアルでは、公式の Amazon Linux 2 AMI から新しく作成されたインスタンスのみをサポートしています。

Amazon Linux での SSL/TLS の設定

Secure Sockets Layer/Transport Layer Security (SSL/TLS) は、ウェブサーバーとウェブクライアントの間に、転送中のデータが傍受されないように保護する、暗号化されたチャネルを確立します。このチュートリアルでは、Amazon Linux AMI と Apache ウェブサーバーを使用して EC2 インスタンスに、SSL/TLS のサポートを手動で追加する方法を説明します。このチュートリアルでは、ロードバランサーを使用していないことを前提としています。Elastic Load Balancing を使用している場合は、代わりに AWS Certificate Manager の証明書を使用して、ロードバランサーで SSL オフロードを設定できます。

歴史的経緯から、ウェブの暗号化は、単純に SSL と呼ばれることが少なくありません。ウェブブラウザでは今でも SSL がサポートされていますが、後継プロトコルである TLS プロトコルの方が攻撃を受けにくくなります。Amazon Linux AMI は、デフォルトですべてのバージョンの SSL をサーバー側でサポートすることを無効にします。セキュリティ標準化団体は、TLS 1.0 は安全でないとみなしています。TLS 1.0 および TLS 1.1 は、2021 年 3 月に正式に非推奨になりました。このチュートリアルは、TLS 1.2 を有効にすることを前提としたガイダンスです。TLS 1.3 は 2018 年に完成し、基盤となる TLS ライブラリ (このチュートリアルでは OpenSSL) がサポートされ、かつ、有効である限り、Amazon Linux 2 で利用できます。クライアントは 2023 年 6 月 28 日までに TLS 1.2 以降をサポートしている必要があります。最新の暗号化基準の詳細については、「RFC 7568」および「RFC 8446」を参照してください。

このチュートリアルでは、現代のウェブ暗号化を単に TLS と呼びます。

Important

これらの手順はAmazon Linux AMI で使用するためのものです。他のディストリビューションのインスタンスで LAMP ウェブサーバーをセットアップする場合、このチュートリアルの一部の手順は使用できません。Amazon Linux 2 については、「Amazon Linux 2 での SSL/TLS の設定」を参照してください。Ubuntu については、Ubuntu 上の OpenSSL に関するコミュニティドキュメントを参照してください。Red Hat Enterprise Linux については、以下を参照してください。Apache HTTP Web サーバーの設定。その他のディストリビューションについては、それぞれのドキュメントを参照してください。

Note

あるいは、AWS Nitro Enclaves 向けの AWS Certificate Manager (ACM) を使用することもできます。これは、パブリックおよびプライベートの SSL/TLS 証明書を(AWS Nitro Enclaves を使用している Amazon EC2 インスタンスで実行されている) ウェブアプリケーションおよびサーバーで使用できるようにする、エンクレーブアプリケーションです。Nitro Enclaves は、SSL/TLS 証明書やプライベートキーなどの機密性の高いデータを保護し、安全に処理するために、分離されたコンピューティング環境を作成できる Amazon EC2 の機能です。
Nitro Enclaves 向け ACM では、Amazon EC2 Linux インスタンスで実行する nginx を使用することで、プライベートキーの作成、証明書とプライベートキーの配布、および証明書の更新を実行します。
Nitro Enclaves 向け ACM を使用するには、エンクレーブ対応の Linux インスタンスを使用する必要があります。
詳細については、AWS Nitro Enclaves ユーザーガイドの「What is AWS Nitro Enclaves?」および「AWS Certificate Manager for Nitro Enclaves」を参照してください。

コンテンツ

前提条件

このチュートリアルを開始する前に、次のステップを完了してください。

- Amazon Linux AMI. を使用して EBS-backed インスタンスを起動します。詳細については、「ステップ 1: インスタンスを起動する」を参照してください。
- インスタンスが以下の TCP ポートで接続を受け付けるようにセキュリティグループを設定します。
  - SSH (ポート 22)
  - HTTP (ポート 80)
  - HTTPS (ポート 443)

  詳細については、「Linux インスタンス用のインバウンドトラフィックの承認」を参照してください。
- Apache ウェブサーバーをインストールします。手順については、「チュートリアル: Amazon Linux への LAMP ウェブサーバーのインストール」を参照してください。必要なのは http24 パッケージおよび対応する従属コンポーネントのみです。PHP および MySQL に関連する手順は無視してかまいません。
- ウェブサイトの識別と認証を行うため、TLS の公開鍵基盤 (PKI) ではドメインネームシステム (DNS) を使用します。EC2 インスタンスを使用してパブリックウェブサイトをホストするには、ウェブサーバーのドメイン名を登録するか、既存のドメイン名を Amazon EC2 ホストに移す必要があります。これについては、ドメイン登録および DNS ホスティングに関するサードパーティのサービスが多数存在します。Amazon Route 53 を使用することもできます。

ステップ 1: サーバーで TLS を有効にする

オプション: オートメーション を使用してこのチュートリアルを完了する

以下のタスクを行う代わりに AWS Systems Manager を使用してこのチュートリアルを完了するには、オートメーションドキュメントを実行します。

この手順では、自己署名のデジタル証明書を使用して、Amazon Linux で TLS をセットアップします。

> **Note**
>
> 自己署名証明書はテスト用であり、本稼働環境では使用できません。インターネットに自己署名証明書を公開すると、サイトへの訪問者にセキュリティ警告が表示されます。

サーバーで TLS を有効にするには

1. インスタンスに接続し、Apache が実行されていることを確認します。

   ```
   [ec2-user ~]$ sudo service httpd status
   ```

   必要であれば、Apache を起動します。

   ```
   [ec2-user ~]$ sudo service httpd start
   ```

2. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。この処理には数分かかりますが、最新の更新とバグ修正を確実に適用することが重要です。

   > **Note**
   >
   > -y オプションを指定すると、確認メッセージを表示せずに更新をインストールします。インストール前に更新を検査する場合は、このオプションを省略できます。

   ```
   [ec2-user ~]$ sudo yum update -y
   ```

3. これでインスタンスが最新状態になったため、Apache モジュール mod_ssl をインストールして TLS サポートを追加します。

   ```
   [ec2-user ~]$ sudo yum install -y mod_ssl
   ```

   次のファイルがインスタンスに作成されました。このファイルは、セキュアサーバーの設定とテスト用の証明書の作成に使用します。

`/etc/httpd/conf.d/ssl.conf`

mod_ssl の設定ファイル。このファイルには、暗号化キーと証明書の場所、許可する TLS プロトコル、受け入れる暗号化アルゴリズムを Apache に指示する「ディレクティブ」が含まれています。

`/etc/pki/tls/private/localhost.key`

Amazon EC2 ホスト用に自動生成された 2048 ビットの RSA プライベートキー。インストール時には、自己署名ホスト証明書を生成するために OpenSSL によってこのキーが使用されます。また、このキーを使用して、認証局 (CA) に送信する証明書署名リクエスト (CSR) を生成することもできます。

`/etc/pki/tls/certs/localhost.crt`

サーバーホスト用に自動生成された、自己署名の X.509 証明書。この証明書は、TLS を使用するように Apache が正しくセットアップされているかどうかをテストする場合に役立ちます。

`.key` ファイルと `.crt` はどちらも PEM 形式であり、この短縮化された証明書の例のように、「BEGIN」行と「END」行で囲まれ Base64 でエンコードされた ASCII 文字で構成されます。

```
-----BEGIN CERTIFICATE-----
MIIEazCCA1OgAwIBAgICWxQwDQYJKoZIhvcNAQELBQAwgbExCzAJBgNVBAYTAi0t
MRIwEAYDVQQIDAlTb21lU3RhdGUxETAPBgNVBAcMCFNvbWVDaXR5MRkwFwYDVQQK
DBBTb21lT3JnYW5pemF0aW9uMR8wHQYDVQQLDBZTb21lT3JnYW5pemF0aW9uYWxV
bml0MRkwFwYDVQQDDBBpcC0xNzItMzEtMjAtMjM2MSQwIgYJKoZIhvcNAQkBFhVy
...
z5rRUE/XzxRLBZOoWZpNWTXJkQ3uFYH6s/sBwtHpKKZMzOvDedREjNKAvk4ws6F0
WanXWehT6FiSZvB4sTEXXJN2jdw8g+sHGnZ8zCOsclknYhHrCVD2vnBlZJKSZvak
3ZazhBxtQSukFMOnWPP2a0DMMFGYUHOd0BQE8sBJxg==
-----END CERTIFICATE-----
```

ファイル名および拡張子は利便性のためであり、機能には影響しません。証明書は、`cert.crt` ファイルの関連ディレクティブが同じ名前を使用している限り、`cert.pem`、`ssl.conf`、またはその他のファイル名で呼び出すことができます。

> **Note**
>
> デフォルトの TLS ファイルを独自にカスタマイズしたファイルに置き換える場合は、PEM 形式であることを確認してください。

4. Apache を再起動します。

```
[ec2-user ~]$ sudo service httpd restart
```

5. Apache ウェブサーバーではポート 443 経由で HTTPS (セキュア HTTP) がサポートされるようになっています。これをテストするには、ブラウザの URL バーに、**https://** というプレフィックスを指定して、EC2 インスタンスの IP アドレスまたは完全修飾ドメイン名を入力します。信頼されていない自己署名ホスト証明書を使用してサイトに接続しようとしているため、ブラウザには一連のセキュリティ警告が表示されることがあります。

   これの警告を無視し、サイトに進みます。サーバーで TLS を正しく設定できていれば、Apache のデフォルトのテストページが開きます。これで、ブラウザとサーバーの間でやり取りされるすべてのデータが安全に暗号化されるようになります。

   サイト訪問者に対して警告画面が表示されないようにするには、暗号化だけではなく、サイト所有者のパブリック認証を行うための証明書を取得する必要があります。

ステップ 2: CA 署名証明書を取得する

CA 署名証明書を取得するには、次の手順に従います。

- プライベートキーから証明書署名リクエスト (CSR) を作成します。
- 作成した CSR を認証機関 (CA) に送信します。
- 署名付きホスト証明書を入手する
- 証明書を使用するように Apache を設定します

自己署名 TLS X.509 ホスト証明書は、暗号化技術上は CA 署名証明書と同じです。これらの相違は数学的なものではなく、社会的なものです。CA では、最低でもドメイン所有権を検証してから申請者に証明書を発行することを保証しています。そのため、各ウェブブラウザには、ブラウザベンダーが信頼する CA のリストが含まれています。X.509 証明書は主に、プライベートサーバーキーに対応するパブリックキーと、このパブリックキーに暗号で関連付けられている CA による署名で構成さ

れています。HTTPS 経由でブラウザがウェブサーバーに接続すると、サーバーは、信頼された CA のリストをブラウザが確認できるように、証明書を提示します。Signerがリストに含まれている場合や、他の信頼された署名者の信頼チェーンを通じてアクセス可能である場合、ブラウザはサーバーと、高速暗号化データチャネルのネゴシエーションを行い、ページをロードします。

証明書には、リクエストの確認作業が必要であり、一般的に費用がかかるため、各社を比較することをお勧めします。いくつかの CA では、基本レベル証明書が無料で提供されます。これらの CA で最も注目すべきは Let's Encrypt プロジェクトです。このプロジェクトでは、証明書の作成および更新プロセスの自動化もサポートしています。Let's Encrypt 証明書の使用の詳細については、「Certbot の取得」を参照してください。

商業グレードのサービスを提供する予定がある場合は、AWS Certificate Manager は良い選択肢です。

ホスト証明書の基盤にはキーがあります。2017 年時点で、政府および業界グループは、2030 年まで、ドキュメントを保護するための RSA キーに 2048 ビットの最小キー (モジュロ) サイズを使用することを推奨しています。Amazon Linux で OpenSSL によって生成されるデフォルトのモジュラスサイズは 2048 ビットです。つまり、自動生成された既存のキーは、CA 署名証明書に適しています。モジュラスサイズを大きくする、別の暗号化アルゴリズムを使用するなど、キーのカスタマイズが必要な場合は、次に示す代替手順に従ってください。

CA 署名ホスト証明書を取得するための手順は、登録およびホスト済みの DNS ドメインを所有している場合を除き、使用しません。

CA 署名証明書を取得するには

1. インスタンスに接続して、/etc/pki/tls/private/ に移動します。これは、サーバーの TLS 用プライベートキーが格納されているディレクトリです。既存のホストキーを使用して CSR を生成する場合は、ステップ 3 に進んでください。
2. (オプション) 新しいプライベートキーを生成します。キー設定のいくつかのサンプルを次に示します。生成されたキーのどれもウェブサーバーで機能しますが、セキュリティの実装方法 (および強度) はそれぞれ異なります。

   - 例 1: デフォルトの RSA ホストキーを作成します。結果として生成されるファイル **custom.key** が、2048 ビットの RSA プライベートキーです。

     ```
     [ec2-user ~]$ sudo openssl genrsa -out custom.key
     ```

   - 例 2: これより大きなモジュラサイズを使用して、より強力な RSA キーを作成します。結果として生成されるファイル **custom.key** が、4096 ビットの RSA プライベートキーです。

```
[ec2-user ~]$ sudo openssl genrsa -out custom.key 4096
```

- 例 3: パスワードで保護された 4096 ビット暗号化 RSA キーを作成します。結果のファイル、**custom.key** は、AES-128 暗号で暗号化された 4096 ビットの RSA プライベートキーです。

  > **Important**
  >
  > キーを暗号化するとセキュリティを強化できますが、暗号化キーにはパスワードが必要であるため、暗号化に依存するサービスを自動的に開始することはできません。このキーを使用するたびに、SSH 接続でパスワード (前述の例では、"abcde12345") を指定する必要があります。

```
[ec2-user ~]$ sudo openssl genrsa -aes128 -passout pass:abcde12345 -out
 custom.key 4096
```

- 例 4: 非 RSA 暗号を使用してキーを作成します。RSA 暗号化は、2 つの大きな素数の積に基づくパブリックキーのサイズのために、比較的遅くなる可能性があります。ただし、非 RSA 暗号化方式を使用する TLS 用のキーを作成することも可能です。同等レベルのセキュリティを提供する場合は、楕円曲線の計算に基づいたキーのほうが小さく計算処理も高速です。

```
[ec2-user ~]$ sudo openssl ecparam -name prime256v1 -out custom.key -genkey
```

  結果は、prime256v1 (OpenSSL でサポートされる "名前付き曲線") を使用した 256 ビットの楕円曲線プライベートキーです。暗号化強度は (NIST によると) 2048 ビットの RSA キーよりやや優れています。

  > **Note**
  >
  > すべての CA で、楕円曲線ベースのキーに対して RSA キーと同じレベルのサポートが提供されているわけではありません。

新しいプライベートキーには、制限の厳しい所有権とアクセス権を設定します (所有者 = root、グループ = root、所有者のみの読み取り/書き込み)。コマンドは次のようになります。

```
[ec2-user ~]$ sudo chown root.root custom.key
[ec2-user ~]$ sudo chmod 600 custom.key
[ec2-user ~]$ ls -al custom.key
```

上のコマンドを実行すると、次のような結果になります。

```
-rw------- root root custom.key
```

適切なキーを作成し、設定できたら、CSR を作成できます。

3. 好みのキーを使用して CSR を作成します。次の例では、**custom.key** を使用しています。

```
[ec2-user ~]$ sudo openssl req -new -key custom.key -out csr.pem
```

OpenSSL によりダイアログが開かれ、次の表に示されている情報の入力が求められます。基本的なドメイン検証済みホスト証明書については、[共通名] 以外のフィールドはすべてオプションです。

名前	説明	例
国名	2 文字の ISO 略称 (国名コード)。	US (= 米国)
州名	あなたが所属する組織の所在地の州または県。省略不可です。	ワシントン
市区町村	市など、組織の場所。	シアトル
組織名	組織の正式名称。組織名は、省略不可です。	Example Corp
部門名	組織に関する追加情報 (存在する場合)。	Example Dept
共通名	この値は、ユーザーがブラウザに入力する必要のあるウェブアドレスと正確に一致します。通常、これはプレフィックス付きのホスト名またはエイリアスによるドメイン名 (**www.example.com** の形式) を意味します。自己署名証明書を使用し、DNS 解決なしでテストを行う場合、共通名の構成要素は	www.example.com

名前	説明	例
	ホスト名のみになる場合があります。CA では、***.example.com** などのワイルドカード名を許容する、よりコストの高い証明書も用意されています。	
E メールアドレス	サーバー管理者の E メールアドレス。	someone@example.com

最後に、OpenSSL により、オプションのチャレンジパスワードが求められます。このパスワードは CSR と、ユーザーと CA の間のトランザクションのみに適用されるため、このフィールドと、もう 1 つのオプションフィールドである、オプションの会社名については、CA の推奨事項に従ってください。CSR のチャレンジパスワードは、サーバー操作には影響しません。

結果として生成されるファイル **csr.pem** には、パブリックキー、パブリックキーのデジタル署名、入札したメタデータが含まれています。

4. CA に CSR を送信します。この作業は通常、テキストエディタで CSR ファイルを開く動作と、内容をウェブフォームにコピーする動作で構成されています。このとき、証明書に適用する 1 つ以上のサブジェクト代替名 (SAN) を指定するように求められることがあります。共通名が **www.example.com** の場合、有効な SAN は **example.com** になります (逆も同様です)。サイトへの訪問者がこれら名前のいずれかを入力すると、エラーなしの接続が提示されます。CA のウェブフォームで許可される場合は、SAN のリストに共通名を含めます 一部の CA では自動的に含められます。

   リクエストが承認されると、CA によって署名された新しいホスト証明書が届きます。CA の信頼チェーンを完成するために必要な、追加の証明書が含まれている中間証明書ファイルをダウンロードするよう指示されることもあります。

   > **Note**
   >
   > 多様な用途向けに複数の形式のファイルを送信してくる CA もあります。このチュートリアルでは、PEM 形式の証明書ファイルのみ使用してください。PEM 形式のファイルには通常、`.pem` または `.crt` 拡張子が使用されます (ただし、常にこれらの拡張子が使用されるわけではありません)。どのファイルを使用すべきかわからない場合は、テキストエディタでファイルを開き、以下の行で始まる 1 つ以上のブロックを含むファイルを見つけてください。

```
- - - - -BEGIN CERTIFICATE - - - - -
```

ファイルの末尾は次のようになっている必要があります。

```
- - - -END CERTIFICATE - - - - -
```

次のように、コマンドラインでファイルを確認することもできます。

```
[ec2-user certs]$ openssl x509 -in certificate.crt -text
```

これらの行がファイルに表示されていることを確認してください。.p7b、.p7c、または類似のファイル拡張子で終了するファイルは使用しないでください。

5. 新しい CA 署名証明書と任意の中間証明書を /etc/pki/tls/certs ディレクトリに配置します。

> **Note**
>
> EC2 インスタンスにカスタムキーをアップロードする方法は複数ありますが、最も簡単でわかりやすい方法は、テキストエディタ (vi、nano、メモ帳など) をローカルコンピュータとインスタンスの両方で開いて、両者の間でファイルの内容をコピーして貼り付けることです。EC2 インスタンス内でこれらの操作を実行する際には、root [sudo] アクセス許可が必要です。こうすることで、許可やパスに問題があるかどうかをすぐに確認できます。ただし、内容をコピーする際に行を追加したり、内容を変更したりしないでください。

/etc/pki/tls/certs ディレクトリの中から、次のコマンドを使用してファイルの所有者、グループ、アクセス許可の設定が制限の厳しい Amazon Linux のデフォルト (所有者 = root、グループ = root、所有者のみの読み込み/書き込み可) と一致することを確認します。

```
[ec2-user certs]$ sudo chown root.root custom.crt
[ec2-user certs]$ sudo chmod 600 custom.crt
[ec2-user certs]$ ls -al custom.crt
```

上のコマンドを実行すると、次のような結果になります。

```
-rw------- root root custom.crt
```

中間証明書ファイルのアクセス権は、比較的厳しくありません (所有者 = root、グループ = root、所有者による書き込み可、グループによる読み取り可、その他による読み取り可)。コマンドは次のようになります。

```
[ec2-user certs]$ sudo chown root.root intermediate.crt
[ec2-user certs]$ sudo chmod 644 intermediate.crt
[ec2-user certs]$ ls -al intermediate.crt
```

上のコマンドを実行すると、次のような結果になります。

```
-rw-r--r-- root root intermediate.crt
```

6. カスタムキーを使用して CSR を作成し、ホスト証明書を取得した場合は、/etc/pki/tls/private/ ディレクトリから古いキーを削除するか、名前を変更して、同ディレクトリに新しいキーをインストールします。

> **Note**
>
> EC2 インスタンスにカスタムキーをアップロードする方法は複数ありますが、最も簡単でわかりやすい方法は、テキストエディタ (vi、nano、メモ帳など) をローカルコンピュータとインスタンスの両方で開いて、両者の間でファイルの内容をコピーして貼り付けることです。EC2 インスタンス内でこれらの操作を実行する際には、root [sudo] 権限が必要です。こうすることで、許可やパスに問題があるかどうかをすぐに確認できます。ただし、内容をコピーする際に行を追加したり、内容を変更したりしないでください。

/etc/pki/tls/private ディレクトリの中から、ファイルの所有者、グループ、アクセス権の設定が制限の厳しい Amazon Linux のデフォルト (所有者 = root、グループ = root、所有者のみの読み込み/書き込み可) と一致することを確認します。コマンドは次のようになります。

```
[ec2-user private]$ sudo chown root.root custom.key
[ec2-user private]$ sudo chmod 600 custom.key
[ec2-user private]$ ls -al custom.key
```

上のコマンドを実行すると、次のような結果になります。

```
-rw------- root root custom.key
```

7. 新しい証明書とキーファイルに合わせるには、/etc/httpd/conf.d/ssl.conf を編集します。

   a. CA 署名のホスト証明書のパスとファイル名を Apache の SSLCertificateFile ディレクティブで指定します。

   ```
   SSLCertificateFile /etc/pki/tls/certs/custom.crt
   ```

   b. 中間証明書ファイル (この例では intermediate.crt) を受け取ったら、Apache の SSLCACertificateFile ディレクティブを使用して、次のファイルのパスとファイル名を指定します。

   ```
   SSLCACertificateFile /etc/pki/tls/certs/intermediate.crt
   ```

   > **Note**
   >
   > 一部の CA では、ホスト証明書と中間証明書を組み合わせて 1 つのファイルを作成するため、このディレクティブは必要ありません。CA が提供している手順を参照してください。

   c. プライベートキーのパスとファイル名を Apache の SSLCertificateKeyFile ディレクティブで指定します。

   ```
   SSLCertificateKeyFile /etc/pki/tls/private/custom.key
   ```

8. /etc/httpd/conf.d/ssl.conf を保存して、Apache を再起動します。

```
[ec2-user ~]$ sudo service httpd restart
```

9. サーバーをテストするには、ブラウザの URL バーにドメイン名を入力し、プレフィックス https:// を指定します。ブラウザによって、エラーが生成されることなく、HTTPS 経由でテストページがロードされます。

ステップ 3: セキュリティ設定をテストして強化する

TLS が運用可能になりパブリックに公開されたら、実際の安全性をテストする必要があります。セキュリティセットアップの詳細な分析を無料で行うことのできる Qualys SSL Labs などのオンラインサービスを使用すると簡単です。その結果に基づき、受け入れるプロトコル、優先する暗号化方式、除外する暗号化方式を制御することによって、デフォルトのセキュリティ設定を強化するかどうかを決定できます。詳細については、「Qualys のスコアの計算方法」を参照してください。

> **Important**
>
> サーバーのセキュリティを確保するには、実際のテストが非常に重要です。小さな設定エラーによって、深刻なセキュリティ侵害やデータの損失が生じる可能性があります。調査や新たな脅威に応じて、推奨されるセキュリティ管理方法は常に変化するため、適切なサーバー管理を行うには、定期的なセキュリティ監査が不可欠です。

Qualys SSL Labs のサイトで、サーバーの完全修飾ドメイン名を **www.example.com** という形式で入力します。約 2 分後に、サイトに関するグレード (A から F) と、結果の詳細な内訳が届きます。概要は設定がほとんど正常であることを示していますが、詳細レポートでは、いくつかの潜在的な問題が指摘されています。次に例を示します。

RC4 暗号は、特定の古いブラウザでの使用がサポートされています。暗号は、暗号化アルゴリズムの計算の中核です。TLS データストリームの暗号化に使用される高速の暗号化方式である RC4 は、いくつかの重大な脆弱性を持つことで知られています。従来のブラウザをサポートするもっともな理由がない限り、この暗号化方式を無効にする必要があります。

✗旧バージョンの TLS がサポートされています。設定では TLS 1.0 (すでに廃止されています) と TLS 1.1 (廃止予定) がサポートされています。2018 年以降は、TLS 1.2 のみ推奨されています。

TLS 設定を修正するには

1. テキストエディタで設定ファイル `/etc/httpd/conf.d/ssl.conf` を開き、行頭に # を付けて以下の行をコメントアウトしてください。

   ```
   #SSLProtocol all -SSLv3
   #SSLProxyProtocol all -SSLv3
   ```

2. 次のディレクティブを追加します。

   ```
   SSLProtocol -SSLv2 -SSLv3 -TLSv1 -TLSv1.1 +TLSv1.2
   ```

```
SSLProxyProtocol -SSLv2 -SSLv3 -TLSv1 -TLSv1.1 +TLSv1.2
```

これらのディレクティブにより、SSL バージョン 2、3、および TLS バージョン 1.0、1.1 が明示的に無効化されます。これで、サーバーでは、TLS 1.2 以外を使用した、クライアントとの暗号化された接続の受け入れが拒否されます。ディレクティブに含める指定が多くなるほど、サーバーの動作に対する設定内容が明確にわかりやすくなります。

> **Note**
>
> このようにして、TLS バージョン 1.0 および 1.1 を無効にすると、ごく一部の古くなったウェブブラウザによるサイトへのアクセスがブロックされるようになります。

許可された暗号のリストを変更するには

1. 設定ファイル /etc/httpd/conf.d/ssl.conf を開き、**SSLCipherSuite** と **SSLProxyCipherSuite** を設定するためのコメントアウトされた例のセクションを見つけます。

```
#SSLCipherSuite HIGH:MEDIUM:!aNULL:!MD5
#SSLProxyCipherSuite HIGH:MEDIUM:!aNULL:!MD5
```

これらの設定はそのままにして、その下に以下のディレクティブを追加します。

> **Note**
>
> ここでは読みやすくするために数行に分けて示していますが、これらの 2 つのディレクティブは 1 行に指定する必要があります。暗号化方式名はスペースで区切りません。

```
SSLCipherSuite ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-SHA384:ECDHE-
ECDSA-CHACHA20-POLY1305:
ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-SHA384:
ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:AES:!
aNULL:!eNULL:!EXPORT:!DES:
!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-SHA:!EDH-RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-
SHA
```

```
SSLProxyCipherSuite ECDHE-ECDSA-AES256-GCM-SHA384:ECDHE-RSA-AES256-GCM-
SHA384:ECDHE-ECDSA-CHACHA20-POLY1305:
ECDHE-RSA-CHACHA20-POLY1305:ECDHE-ECDSA-AES128-GCM-SHA256:ECDHE-RSA-AES128-GCM-
SHA256:ECDHE-ECDSA-AES256-SHA384:
ECDHE-RSA-AES256-SHA384:ECDHE-ECDSA-AES128-SHA256:ECDHE-RSA-AES128-SHA256:AES:!
aNULL:!eNULL:!EXPORT:!DES:
!RC4:!MD5:!PSK:!aECDH:!EDH-DSS-DES-CBC3-SHA:!EDH-RSA-DES-CBC3-SHA:!KRB5-DES-CBC3-
SHA
```

これらの暗号化方式は、OpenSSL でサポートされている暗号化方式の長いリストのうち、一部です。これらは、以下の条件に応じて選択され、順序付けられています。

- 前方秘匿性のサポート
- Strength
- スピード
- 特定の暗号化方式、その後に暗号化方式のファミリー
- 許可されている暗号化方式、その後に拒否されている暗号化方式

ランクの高い暗号化方式の名前には、ECDHE が含まれています (Elliptic Curve Diffie-Hellman Ephemeral など)。ephemeral は前方秘匿性を示します。また、RC4 は現在、リストの最後に近い位置にあり、禁止された暗号化方式に含まれています。

デフォルトや、内容が見えない簡単なディレクティブに依存するのではなく、暗号化方式の明示的なリストを使用することをお勧めします。ここに示されている暗号化方式リストは、多数考えられるリストの 1 つに過ぎません。例えば、前方秘匿性よりスピードを重視したリストが必要になることもあります。

古いクライアントをサポートする必要性が予測される場合は、DES-CBC3-SHA 暗号化スイートを許可することができます。

OpenSSL の更新ごとに、新しい暗号化方式が導入され古い暗号化方式が廃止されます。EC2 の Amazon Linux インスタンスを最新の状態に維持し、セキュリティに関する OpenSSL からの告知に注意して、技術分野の報道でセキュリティ面の新しい脆弱性に関するレポートを警戒してください。

2. 次の行について、"#" を削除してコメント解除します。

```
#SSLHonorCipherOrder on
```

このコマンドは、(この場合) 前方秘匿性をサポートするものも含めて、ランクの高い暗号化方式を優先するようサーバーに強制します。このディレクティブが有効になると、サーバーは、セキュリティの弱い暗号化方式に戻る前に、セキュリティが強力な接続を確立しようとします。

3. Apache を再起動します。Qualys SSL Labs でドメインをもう一度テストすると、RC4 の脆弱性がなくなったことがわかります。

トラブルシューティング

- パスワードを入力しないと Apache ウェブサーバーが起動しません

  これは、パスワードで保護された暗号化プライベート サーバー キーをインストールした場合は正常な動作です。

  暗号化とパスワードの要件をキーから削除できます。デフォルトディレクトリに `custom.key` という暗号化プライベート RSA キーがあり、そのパスワードが **abcde12345** であるとすると、EC2 インスタンスで次のコマンドを実行し、このキーの非暗号化バージョンを生成してください。

```
[ec2-user ~]$ cd /etc/pki/tls/private/
[ec2-user private]$ sudo cp custom.key custom.key.bak
[ec2-user private]$ sudo openssl rsa -in custom.key -passin pass:abcde12345 -out
 custom.key.nocrypt
[ec2-user private]$ sudo mv custom.key.nocrypt custom.key
[ec2-user private]$ sudo chown root.root custom.key
[ec2-user private]$ sudo chmod 600 custom.key
[ec2-user private]$ sudo service httpd restart
```

  パスワードが求められずに Apache が起動するようになります。

WordPress ブログをホストする

このセクションでは、Amazon EC2 インスタンスに WordPress ブログをインストールおよび設定し、その安全性を確立する方法を説明します。

> **Note**
>
> AL2023 WordPress チュートリアルについては、「Amazon Linux 2023 ユーザーガイド」の「チュートリアル: AL2023 で WordPress ブログをホストする」を参照してください。

以下で WordPress ブログをホストする

- Amazon Linux 2 での WordPress ブログのホスト

Amazon Linux 2 での WordPress ブログのホスト

次の手順では、お客様の Amazon Linux 2 インスタンスで、WordPress ブログのインストール、構成を実行し、安全性を確立します。このチュートリアルは、WordPress ブログをホストするウェブサーバーを完全に制御する (これは従来のホスティングサービスでは一般的なことではありません) という点で、Amazon EC2 を使用するための優れた手引きになります。

サーバーに対するソフトウェアパッケージの更新とセキュリティパッチの維持は、お客様の責任となります。ウェブサーバー構成との直接的な対話操作を必要としない、より自動化された WordPress をインストールする場合、AWS CloudFormation サービスは、迅速に始められる WordPress テンプレートを提供します。詳細については、 AWS CloudFormation ユーザーガイドの「開始方法」を参照してください。Windows インスタンスで WordPress ブログをホストする場合は、Windows インスタンスの Amazon EC2 ユーザーガイド の「Amazon EC2 Windows インスタンスへの WordPress ブログのデプロイ」を参照してください。データベースが疎結合化された高可用性のソリューションが必要な場合は、AWS Elastic Beanstalk デベロッパーガイドの「高可用性の WordPress ウェブサイトをデプロイする」を参照してください。

> **Important**
>
> これらの手順は Amazon Linux 2 で使用するためのものです。その他のディストリビューションの詳細については、各ドキュメントを参照してください。このチュートリアルの多くの手順は、Ubuntu インスタンスには使用できません。Ubuntu インスタンスでの WordPress のインストールについては、Ubuntu のドキュメントの「WordPress」を参照してください。CodeDeploy を使用して、Amazon Linux、macOS、または Unix システムでこのタスクを実行することもできます。

トピック

前提条件

このチュートリアルは、Amazon Linux 2 の Amazon Linux 2 での LAMP のインストール の Amazon Linux への LAMP のインストール にあるすべての手順に従って PHP とデータベース (MySQL または MariaDB) をサポートしている、機能的なウェブサーバーを使って、Amazon Linux インスタンスを起動していることが前提となります。このチュートリアルでは、セキュリティグループで HTTP および HTTPS トラフィックを許可するように設定する手順や、ウェブサーバー用にファイルアクセス許可が正しく設定されていることを確認する手順も示します。セキュリティグループへのルール追加の詳細については、セキュリティグループへのルールの追加 を参照してください。

Elastic IP アドレス (EIP) は、WordPress ブログのホストに使用しているインスタンスに関連付けることを強くお勧めします。これにより、インスタンスのパブリック DNS アドレスが変更されて、インストールが破損することを防止できます。ドメイン名を所有していてそのドメインをブログに使用する場合、EIP アドレスをポイントするようにドメイン名の DNS レコードを更新できます (これを行うには、ドメイン名レジストラに問い合わせてください)。実行中のインスタンスに関連付けられた EIP アドレスを無料で 1 つ取得できます。詳細については、「Elastic IP アドレス」を参照してください。

ブログのドメイン名がまだない場合は、Route 53 にドメイン名を登録し、そのドメイン名にインスタンスの EIP アドレスを関連付けることができます。詳細については、Amazon Route 53 デベロッパーガイド の「Amazon Route 53 を使用したドメイン名の登録」を参照してください。

WordPress のインストール

オプション: オートメーション を使用してこのチュートリアルを完了する

以下のタスクを行う代わりに AWS Systems Manager オートメーションを使用してこのチュートリアルを完了するには、オートメーションドキュメントを実行します。

インスタンスに接続して、WordPress インストールパッケージをダウンロードします。

WordPress インストールパッケージをダウンロードして解凍するには

1. wget コマンドを使って、最新の WordPress インストールパッケージをダウンロードします。次のコマンドを実行すると、最新リリースが必ずダウンロードされます。

```
[ec2-user ~]$ wget https://wordpress.org/latest.tar.gz
```

2. インストールパッケージを解凍します。インストールフォルダは、wordpress という名前のフォルダに解凍されます。

```
[ec2-user ~]$ tar -xzf latest.tar.gz
```

WordPress インストール用にデータベースユーザーとデータベースを作成するには

WordPress インストールは、ブログの投稿、ユーザーコメントなどの情報をデータベースに格納する必要があります。この手順を実行すると、ブログのデータベースを作成するのに役立ち、このデータベースに対して情報の読み取りや保存を許可されたユーザーにも有用です。

1. データベースサーバーを起動します。

   - ```
     [ec2-user ~]$ sudo systemctl start mariadb
     ```

2. データベースサーバーに root ユーザーとしてログインします。メッセージが表示されたら、データベース root パスワードを入力します。これは通常の root システムパスワードと異なることもあれば、データベースサーバーのセキュリティ確保を実行していない場合は、空のときもあります。

   データベースサーバーのセキュリティを確保していない場合、セキュリティ確保を行うことは重要です。詳細については、MariaDB サーバーをセキュリティで保護するには (Amazon Linux 2) を参照してください。

```
[ec2-user ~]$ mysql -u root -p
```

3. MySQL データベースのユーザーとパスワードを作成します。WordPress インストールは、これらの値を使って、MySQL データベースと通信を行います。

   ユーザー用に強力なパスワードを作成してください。パスワードに一重引用符 (') を使用しないでください。この文字は前述のコマンドを中断させるためです。安全なパスワードの作成の詳細については、「http://www.pctools.com/guides/password/」を参照してください。既存のパス
```

ワードを再利用しないでください。また、このパスワードは必ず安全な場所に保管してください。

一意のユーザー名とパスワードを入力して、次のコマンドを入力します。

```
CREATE USER 'wordpress-user'@'localhost' IDENTIFIED BY 'your_strong_password';
```

4. データベースを作成します。wordpress-db など、データベースにはわかりやすい名前を使用します。

> **Note**
>
> 次のコマンドのデータベース名を囲む区切り記号は、「バックティック」と呼ばれています。バックティック (`) キーは通常、標準キーボードの Tab キーの上に配置されています。バックティックは必ずしも必要ではありませんが、データベース名では使用できない文字 (ハイフンなど) の代わりに使用できます。

```
CREATE DATABASE `wordpress-db`;
```

5. データベースに対して、以前作成した WordPress ユーザーに対する完全な権限を付与します。

```
GRANT ALL PRIVILEGES ON `wordpress-db`.* TO "wordpress-user"@"localhost";
```

6. すべての変更を有効にするため、データベース権限をフラッシュします。

```
FLUSH PRIVILEGES;
```

7. mysql クライアントを終了します。

```
exit
```

wp-config.php ファイルの作成と編集を行うには

WordPress インストールフォルダには、wp-config-sample.php という名前の構成ファイル例が格納されています。この手順では、このファイルをコピーして、特定の構成に合うように編集します。

1. wp-config-sample.php ファイルを wp-config.php という名前でコピーします。この操作を実行すると、新しい構成ファイルが作成され、元のファイルがバックアップとしてそのまま保持されます。

```
[ec2-user ~]$ cp wordpress/wp-config-sample.php wordpress/wp-config.php
```

2. お好みのテキスト工ディタ (wp-config.php、nano など) を使って vim ファイルを編集し、インストール用の値を入力します。お好みのテキストエディタがない場合、nano が初心者に適しています。

```
[ec2-user ~]$ nano wordpress/wp-config.php
```

   a. DB_NAME を定義する行を探して、database_name_here を Step 4 の WordPress インストール用にデータベースユーザーとデータベースを作成するには で作成したデータベース名に変更します。

```
define('DB_NAME', 'wordpress-db');
```

   b. DB_USER を定義する行を探して、username_here を Step 3 の WordPress インストール用にデータベースユーザーとデータベースを作成するには で作成したデータベースユーザーに変更します。

```
define('DB_USER', 'wordpress-user');
```

   c. DB_PASSWORD を定義する行を探して、password_here を Step 3 の WordPress インストール用にデータベースユーザーとデータベースを作成するには で作成した強力なパスワードに変更します。

```
define('DB_PASSWORD', 'your_strong_password');
```

   d. Authentication Unique Keys and Salts というセクションを見つけます。これらの KEY と SALT の値は、WordPress ユーザーがローカルマシンに保存したブラウザクッキーに対する暗号化レイヤーを提供します。基本的に、ここで長くてランダムな値を指定すると、サイトのセキュリティが向上します。https://api.wordpress.org/secret-key/1.1/salt/ にアクセスして、ランダムに生成されるキーセット値を取得し、wp-config.php ファイルにコピーして貼り付けることができます。PuTTY 端末にテキストを貼り付けるには、テキストを貼り付ける場所にカーソルを置き、PuTTY 端末内でマウスを右クリックします。

セキュリティキーの詳細については、「https://wordpress.org/support/article/editing-wp-config-php/#security-keys」にアクセスしてください。

> **Note**
>
> 次の値はサンプル専用です。これらの値を実際のインストールには使わないでください。

```
define('AUTH_KEY',         ' #U$$+[RXN8:b^-L 0(WU_+ c+WFkI~c]o]-bHw+)/
Aj[wTwSiZ<Qb[mghEXcRh-');
define('SECURE_AUTH_KEY',  'Zsz._P=l/|y.Lq)XjlkwS1y5NJ76E6EJ.AV0pCKZZB,*~*r ?
6OP$eJT@;+(ndLg');
define('LOGGED_IN_KEY',    'ju}qwre3V*+8f_zOWf?{LlGsQ]Ye@2Jh^,8x>)Y |;(^[Iw]Pi
+LG#A4R?7N`YB3');
define('NONCE_KEY',        'P(g62HeZxEes|LnI^i=H,[XwK9I&[2s|:?0N}VJM%?;v2v]v+;
+^9eXUahg@::Cj');
define('AUTH_SALT',        'C$DpB4Hj[JK:?{ql`sRVa:{:7yShy(9A@5wg+`JJVb1fk%_-
Bx*M4(qc[Qg%JT!h');
define('SECURE_AUTH_SALT', 'd!uRu#}+q#{f$Z?Z9uFPG.${+S{n~1M&%@~gL>U>NV<zpD-@2-
Es7Q1O-bp28EKv');
define('LOGGED_IN_SALT',   ';j{00P*owZf)kVD+FVLn-~ >.|Y%Ug4#I^*LVd9QeZ^&XmK|
e(76miC+&W&+^0P/');
define('NONCE_SALT',       '-97r*V/cgxLmp?Zy4zUU4r99QQ_rGs2LTd%P;|
_e1tS)8_B/,.6[=UK<J_y9?JWG');
```

e. ファイルを保存し、テキストエディタを終了します。

WordPress ファイルを Apache ドキュメントルートの下にインストールするには

- インストールフォルダの解凍、MySQL データベースとユーザーの作成、WordPress 構成ファイルのカスタマイズが終了したため、インストールファイルをウェブサーバーのドキュメントルートにコピーし、インストールスクリプトを実行して、インストールを終了する準備ができました。これらのファイルの場所は、ウェブサーバーの実際のルートで WordPress ブログを使用できるようにするかどうか (*my.public.dns.amazonaws.com* など)、またはルートの下のサブディレクトリやフォルダに格納するか (*my.public.dns.amazonaws.com/blog* など) によって異なります。

- WordPress をドキュメントルートで実行する場合は、WordPress のインストールディレクトリのコンテンツを次のようにコピーします (ただし、ディレクトリ自体はコピーしません)。

```
[ec2-user ~]$ cp -r wordpress/* /var/www/html/
```

- WordPress をドキュメントルートの下の別のディレクトリで実行する場合、まず、そのディレクトリを作成してから、そこにファイルをコピーします。この例では、WordPress はディレクトリ blog から実行されます。

```
[ec2-user ~]$ mkdir /var/www/html/blog
[ec2-user ~]$ cp -r wordpress/* /var/www/html/blog/
```

> **Important**
>
> セキュリティ上の理由から、次の手順にすぐに進まない場合は、Apache ウェブサーバー (httpd) を直ちに停止してください。インストールを Apache ドキュメントルートの下に移動すると、WordPress インストールスクリプトは保護されなくなり、Apache ウェブサーバーが実行している場合、攻撃者はブログへのアクセス権を取得する可能性があります。Apache ウェブサーバーを停止するには、sudo systemctl stop httpd コマンドを入力します。次の手順に移動する場合、Apache ウェブサーバーを停止する必要はありません。

WordPress がパーマリンクを使用できるようにするには

WordPress のパーマリンクが正しく機能するには Apache の .htaccess ファイルを使用する必要がありますが、Amazon Linux ではデフォルトで有効になっていません。Apache ドキュメントルートですべての上書きできるようにするには、次の手順を使用します。

1. お好みのテキスト工ディタ (httpd.conf や nano など) で、vim ファイルを開きます。お好みのテキスト工ディタがない場合、nano が初心者に適しています。

```
[ec2-user ~]$ sudo vim /etc/httpd/conf/httpd.conf
```

2. <Directory "/var/www/html"> で始まるセクションを見つけます。

```
<Directory "/var/www/html">
    #
```

```
    # Possible values for the Options directive are "None", "All",
    # or any combination of:
    #   Indexes Includes FollowSymLinks SymLinksifOwnerMatch ExecCGI MultiViews
    #
    # Note that "MultiViews" must be named *explicitly* --- "Options All"
    # doesn't give it to you.
    #
    # The Options directive is both complicated and important.  Please see
    # http://httpd.apache.org/docs/2.4/mod/core.html#options
    # for more information.
    #
    Options Indexes FollowSymLinks

    #
    # AllowOverride controls what directives may be placed in .htaccess files.
    # It can be "All", "None", or any combination of the keywords:
    #   Options FileInfo AuthConfig Limit
    #
    AllowOverride None

    #
    # Controls who can get stuff from this server.
    #
    Require all granted
</Directory>
```

3. 上のセクションの AllowOverride None 行を AllowOverride *All* に変更します。

> **Note**
>
> このファイルには複数の AllowOverride 行があります。必ず <Directory "/var/www/html"> セクションの行を変更してください。

```
AllowOverride All
```

4. ファイルを保存し、テキストエディタを終了します。

PHP グラフィック描画ライブラリを Amazon Linux 2 にインストールするには

PHP 用の GD ライブラリを使用すると、イメージを変更することができます。ブログのヘッダーイメージをトリミングする必要がある場合は、このライブラリをインストールします。インストールするバージョンの phpMyAdmin は、このライブラリの特定の最小バージョン (バージョン 7.2 など) を必要とする場合があります。

PHP グラフィック描画ライブラリを Amazon Linux 2 にインストールするには、次のコマンドを使用します。例えば、LAMP スタックをインストールする一環として amazon-linux-extras から php7.2 をインストールした場合、このコマンドは PHP グラフィック描画ライブラリのバージョン 7.2 をインストールします。

```
[ec2-user ~]$ sudo yum install php-gd
```

インストールしたバージョンを検証するには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo yum list installed php-gd
```

出力例を次に示します。

```
php-gd.x86_64                  7.2.30-1.amzn2           @amzn2extra-php7.2
```

Apache ウェブサーバーのファイル許可を修正するには

WordPress で使用できる機能の中には、Apache ドキュメントルートへの書き込み権限が必要なものがあります (管理画面を使った、メディアのアップロードなど)。まだ設定していない場合は、次のグループのメンバーシップおよびアクセス許可を適用します (プロセスの詳細は「LAMP ウェブサーバーチュートリアル」を参照)。

1. /var/www とそのコンテンツのファイル所有権を apache ユーザーに付与します。

   ```
   [ec2-user ~]$ sudo chown -R apache /var/www
   ```

2. /var/www とそのコンテンツのグループ所有権を apache グループに付与します。

   ```
   [ec2-user ~]$ sudo chgrp -R apache /var/www
   ```

3. /var/www およびそのサブディレクトリのディレクトリ許可を変更してグループの書き込み許可を設定し、将来のサブディレクトリにグループ ID を設定します。

   ```
   [ec2-user ~]$ sudo chmod 2775 /var/www
   ```

```
[ec2-user ~]$ find /var/www -type d -exec sudo chmod 2775 {} \;
```

4. /var/www およびそのサブディレクトリのファイル許可を繰り返し変更します。

```
[ec2-user ~]$ find /var/www -type f -exec sudo chmod 0644 {} \;
```

> **Note**
>
> WordPress を FTP サーバーとして使用する場合も、これよりも制限の少ないグループ設定が必要になります。これを実行するには、「WordPress で推奨されている手順とセキュリティ設定」を参照してください。

5. Apache ウェブサーバーを再起動して、新しいグループと許可を有効にします。

- ```
  [ec2-user ~]$ sudo systemctl restart httpd
  ```

Amazon Linux 2 で WordPress インストールスクリプトを実行する

WordPress をインストールする準備ができました。使用するコマンドは、オペレーティングシステムによって異なります。この手順のコマンドは、Amazon Linux 2 で使用するためのものです。

1. systemctl コマンドを使って、httpd サービスとデータベースサービスがシステムブート時に起動することを確認します。

```
[ec2-user ~]$ sudo systemctl enable httpd && sudo systemctl enable mariadb
```

2. データベースサーバーが実行中であることを確認します。

```
[ec2-user ~]$ sudo systemctl status mariadb
```

データベースサービスが実行されていない場合は、起動します。

```
[ec2-user ~]$ sudo systemctl start mariadb
```

3. Apache ウェブサーバー (httpd) が実行中であることを確認します。

```
[ec2-user ~]$ sudo systemctl status httpd
```

httpd サービスが実行されていない場合は、起動します。
```

```
[ec2-user ~]$ sudo systemctl start httpd
```

4. ウェブブラウザで WordPress ブログの URL を入力します (インスタンスのパブリック DNS アドレス、または blog フォルダに続くアドレス)。WordPress インストールスクリプトが表示されます。WordPress のインストールに必要な情報を入力します。[WordPress のインストール] を選択して、インストールを完了します。詳細については、WordPress のウェブサイトの Step 5: Run the Install Script を参照してください。

次のステップ

WordPress ブログをテストしたら、設定の更新を検討します。

カスタムドメイン名を使用する

EC2 インスタンスの EIP アドレスに関連付けられたドメイン名がある場合、EC2 パブリック DNS アドレスの代わりにその名前を使用するようにブログを設定できます。詳細については、WordPress ウェブサイトの「サイトの URL の変更」を参照してください。

ブログを設定する

読者にパーソナライズされた体験を提供するため、さまざまなテーマやプラグインを使用するようにブログを設定できます。ただし、インストールプロセスで問題が発生してブログ全体が失われることがあります。インストール中に問題が発生した場合もブログを復元できるように、テーマやプラグインを員ストーする前にインスタンスのバックアップ Amazon マシンイメージ (AMI) を作成しておくことを強くお勧めします。詳細については、「独自の AMI の作成」を参照してください。

容量を増やす

WordPress ブログが人気になり処理能力やストレージを増やす必要がある場合は、次のステップを検討してください。

- インスタンスストレージ領域を拡張する。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS Elastic Volumes」を参照してください。
- MySQL データベースを Amazon RDS に移動して、サービスが持つ容易にスケールする機能を活用する。

インターネットトラフィックのネットワークパフォーマンスを向上させる

ブログにより世界中のユーザーからのトラフィックが増加すると予想される場合は、AWS Global Accelerator をご検討ください。Global Accelerator を使用すると、ユーザーのクライアントデバイスと AWS で実行中の WordPress アプリケーションとの間で、インターネットトラフィックのパフォーマンスを向上でき、低レイテンシーを実現できます。Global Accelerator では、AWS グローバルネットワークを使用して、トラフィックをクライアントから最も近い AWS リージョンにある正常なアプリケーションエンドポイントに送信します。

WordPress の詳細

WordPress の詳細については、「http://codex.wordpress.org/」にある WordPress Codex ヘルプ文書を参照してください。インストールのトラブルシューティングの詳細については、https://wordpress.org/support/article/how-to-install-wordpress/#common-installation-problems にアクセスしてください。WordPress ブログのセキュリティ向上の詳細については、https://wordpress.org/support/article/hardening-wordpress/ にアクセスしてください。WordPress ブログを最新状態に維持する方法の詳細については、https://wordpress.org/support/article/updating-wordpress/ にアクセスしてください。

ヘルプ! パブリック DNS 名が変更されたため、ブログが壊れました

WordPress のインストールは、EC2 インスタンスのパブリック DNS アドレスを使用して自動的に設定されます。インスタンスを停止および再開した場合、パブリック DNS アドレスが変更され (Elastic IP アドレスに関連付けられている場合を除く)、ブログが存在しなくなった (または別の EC2 インスタンスに割り当てられた) アドレスにあるリソースを参照することになるため、ブログは機能しなくなります。問題および考えられるいくつかの解決策の詳細については、https://wordpress.org/support/article/changing-the-site-url/ で説明されています。

WordPress のインストール時にこの状況が発生した場合、WordPress の wp-cli コマンドラインインターフェイスを使用する以下の手順でブログを復元できる可能性があります。

wp-cli を使用して WordPress のサイト URL を変更するには

1. SSH を使って EC2 インスタンスに接続します。
2. インスタンスの古いサイト URL と新しいサイト URL を書き留めます。古いサイト URL は、WordPress をインストールした時点での EC2 インスタンスのパブリック DNS 名と考えられます。新しいサイト URL は、EC2 インスタンスの現在のパブリック DNS 名です。古いサイト URL が不明な場合、次のコマンドで curl を使用して調べることができます。

```
[ec2-user ~]$ curl localhost | grep wp-content
```

古いパブリック DNS 名への参照が出力に表示されます。次に例を示します (古いサイト URL は赤色になっています)。

```
<script type='text/javascript' src='http://ec2-52-8-139-223.us-
west-1.compute.amazonaws.com/wp-content/themes/twentyfifteen/js/functions.js?
ver=20150330'></script>
```

3. 次のコマンドを使って wp-cli をダウンロードします。

```
[ec2-user ~]$ curl -O https://raw.githubusercontent.com/wp-cli/builds/gh-pages/
phar/wp-cli.phar
```

4. 次のコマンドを使って、WordPress インストールの古いサイト URL を検索し、置き換えます。EC2 インスタンスの古いサイト URL と新しいサイト URL、および WordPress のインストールパス (通常は `/var/www/html` または `/var/www/html/blog`) を置き換えます。

```
[ec2-user ~]$ php wp-cli.phar search-replace 'old_site_url' 'new_site_url' --path=/
path/to/wordpress/installation --skip-columns=guid
```

5. ウェブブラウザで、WordPress ブログの新しいサイト URL を入力し、サイトが再び正しく動作していることを確認します。それ以外の場合、詳細については、https://wordpress.org/support/article/changing-the-site-url/ および https://wordpress.org/support/article/how-to-install-wordpress/#common-installation-problems を参照してください。

Amazon マシンイメージ (AMI)

Amazon マシンイメージ (AMI) は、AWS がサポートおよび管理するイメージで、インスタンスの起動に必要な情報を提供します。インスタンスを起動するときは、AMI を指定する必要があります。同じ設定で複数のインスタンスが必要な場合は、1 つの AMI から複数のインスタンスを起動できます。さまざまな設定のインスタンスが必要なときは、各インスタンスをそれぞれ異なる AMI から起動できます。

AMI には次が含まれています。

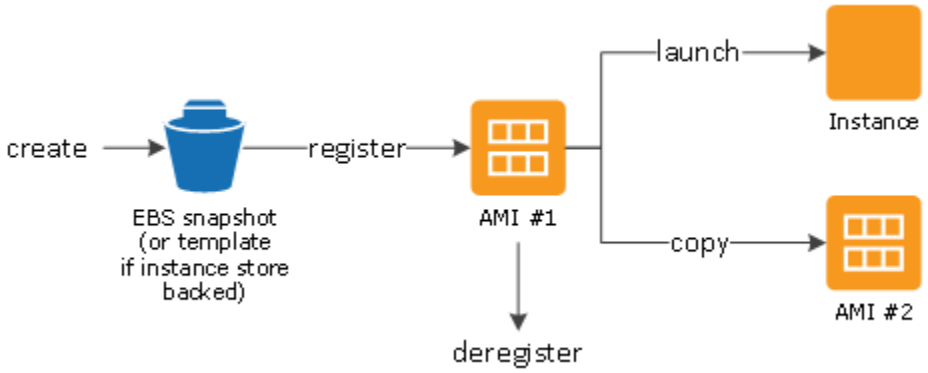
- 1 つまたは複数の Amazon Elastic Block Store (Amazon EBS) スナップショット、または、instance-store-backed AMI、インスタンスのルートボリュームのテンプレート (オペレーティングシステム、アプリケーションサーバー、アプリケーションなど)
- AWS アカウントが AMI を使用してインスタンスを起動可能にするための起動許可
- インスタンスの起動時にインスタンスにアタッチするボリュームを指定するブロックデバイスマッピング

Amazon マシンイメージ (AMI) のトピック

- AMI の請求情報について
- Amazon Linux
- ユーザー提供カーネル
- Amazon Linux 2 MATE デスクトップ接続を設定する
- AMI クォータ

AMI の使用

次の図は AMI のライフサイクルをまとめたものです。AMI を作成し、登録したら、それを使用して新しいインスタンスを起動できます (AMI 所有者から起動許可を与えられた場合、AMI からインスタンスを起動することもできます)。AMI は同じ AWS リージョン内でコピーすることも、異なる AWS リージョンにコピーすることもできます。不要になった AMI は登録を解除できます。



ご自分のインスタンスの基準に一致する AMI を検索できます。AWS が提供する AMI、またはコミュニティが提供する AMI を検索できます。詳細については、AMI タイプ および Linux AMI の検索 を参照してください。

AMI からインスタンスを起動したら、インスタンスに接続できます。インスタンスに接続したら、そのインスタンスを他のサーバーとまったく同じように使用できます。インスタンスの起動、接続、使用に関する詳細については、チュートリアル: Amazon EC2 Linux インスタンスの開始方法 を参照してください。

独自の AMI の作成

既存の AMI からインスタンスを作成して、インスタンスをカスタマイズ (例えば、インスタンスにソフトウェアをインストール) した後に、更新した設定をカスタム AMI として保存することができます。この新しいカスタム AMI から起動されるインスタンスには、AMI の作成時に追加したカスタマイズが含まれます。

AMI の作成プロセスは、インスタンスのルートストレージデバイスにより決まります。インスタンスのルートボリュームは、Amazon Elastic Block Store (Amazon EBS) ボリュームまたはインスタンスストアボリュームのどちらかです。ルートデバイスボリュームの詳細については、Amazon EC2 インスタンスのルートボリューム を参照してください。

- Amazon EBS-backed AMI を作成するには、「Amazon EBS-backed Linux AMI を作成する」を参照してください。
- Instance Store-Backed AMI を作成するには、「instance store-backed Linux AMI を作成する」を参照してください。

AMI には分類や管理のために任意のタグを付けられます。詳細については、Amazon EC2 リソースのタグ付け を参照してください。

AMI の購入、共有、販売

AMI を作成したら、自分だけがそれを使用できるようにプライベートとして保存したり、AWS アカウントの指定リストと共有したりできます。コミュニティで利用できるように、カスタム AMI を公開することもできます。安全で信頼性が高く、便利な AMI を作成して、一般公開する手順はきわめて単純で、いくつかのシンプルなガイドラインにしたがうだけです。共有 AMI の作成および使用方法の詳細については、共有 AMI を参照してください。

Red Hat のような組織のサービス契約に付属する AMI など、サードパーティーから AMI を購入できます。また、AMI を作成し、他の Amazon EC2 ユーザーに販売することもできます。AMI の購入と販売に関する詳細については、有料 AMI を参照してください。

AMI の登録の解除

AMI の利用が終わったら、その登録を解除できます。AMI の登録を解除すると、その AMI を使用して新しいインスタンスを起動できなくなります。その AMI から起動された既存のインスタンスは影響を受けません。詳細については、「AMI の登録の解除」を参照してください。

AL2023 と Amazon Linux 2

Amazon Linux の最新リリースである AL2023 は、Amazon EC2 向けに最適化されており、Amazon EC2 ユーザーに追加コストなしで提供されます。AL2023 の特徴には、予測可能なリリース間隔、頻繁な更新、長期サポートなどがあります。

AL2023 の機能および AL2023 AMI の起動の詳細については、以下を参照してください。

- AL2023 の機能
- AL2023 の使用を開始する

Amazon Linux 2 は、Amazon EC2 で実行されるアプリケーションに、安定した、安全で高性能な実行環境を提供します。Amazon Linux 2 EOL の詳細については、「Amazon Linux 2 に関するよくある質問」を参照してください。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

AMI タイプ

次の特性に基づき、使用する AMI を選択できます。

- リージョン (「リージョンとゾーン」を参照)
- オペレーティングシステム
- アーキテクチャ (32 ビットまたは 64 ビット)
- 起動許可
- ルートデバイスのストレージ

起動許可

AMI の所有者は、起動許可を指定することで可用性を決定します。起動許可は次のように分類されます。

起動アクセス許可	説明
パブリック	所有者はすべての AWS アカウントに起動許可を与えます。

起動アクセス許可	説明
明示的	所有者は特定の AWS アカウント、組織、または組織単位 (OU) に起動許可を与えます。
暗示的	所有者には AMI の暗示的起動許可があります。

Amazon や Amazon EC2 コミュニティではさまざまなパブリック AMI を提供しています。詳細については、共有 AMI を参照してください。デベロッパーは自分の AMI に料金を請求できます。詳細については、有料 AMI を参照してください。

ルートデバイスのストレージ

すべての AMI が Amazon EBS-Backed と Instance Store-Backed のいずれかに分類されます。

- Amazon EBS-backed AMI – AMI から起動されるインスタンスのルートデバイスが、Amazon EBS スナップショットから作成される Amazon Elastic Block Store (Amazon EBS) ボリュームであることを意味します。
- Amazon instance store-backed AMI - AMI から起動したインスタンスのルートデバイスは、Amazon S3 に保存されたテンプレートから作成されたインスタンスストアボリュームです。

詳細については、Amazon EC2 インスタンスのルートボリューム を参照してください。

次の表では、2 種類の AMI を使用した場合の重要な相違点をまとめています。

特徴	Amazon EBS-backed AMI	Amazon instance store-backed AMI
インスタンスの起動時間	通常 1 分以内	通常 5 分以内
ルートデバイスのサイズ制限	64 TiB**	10 GiB
	EBS ボリューム	インスタンスストアボリューム

特徴	Amazon EBS-backed AMI	Amazon instance store-backed AMI
ルートデバイスボリューム		
データの永続性	デフォルトでは、インスタンスを終了するとルートボリュームは削除されます。* EBS ボリュームにある他のデータはすべて、インスタンスの終了後もデフォルトで保持されます。	インスタンスストアボリューム上のデータは、インスタンスの存続中のみ使用できます。
変更	インスタンスの停止中に、インスタンスタイプ、カーネル、RAM ディスク、およびユーザーデータが変更可能	インスタンスの属性は、インスタンスを削除するまで固定。
料金	インスタンスの使用量、EBS ボリューム、また、EBS スナップショットとして保存した AMI に対して料金が発生します。	インスタンスの使用量や Amazon S3 に保存した AMI に対して料金が発生します。
AMI の作成/バンドル	単一のコマンドまたは呼び出しを使用	AMI ツールをインストールして使用する必要があります
停止状態	停止状態になっている場合があります。インスタンスが停止して実行されていない場合でも、ルートボリュームは Amazon EBS で保持されます。	実行中もしくは終了のどちらの場合でも、インスタンスを停止状態にすることができない

* デフォルトでは、EBS ルートボリュームには `DeleteOnTermination` フラグが `true` に設定されています。このフラグを変更し、終了後もボリュームを保持する方法については、「永続的ルートボリュームへの変更」を参照してください。

** `io2` EBS ブロックエクスプレスのみでサポートされています。詳細については、「Amazon EBS ユーザーガイド」の「プロビジョンド IOPS SSD Block Express ボリューム」を参照してください。

AMI のルートデバイスタイプの判別

コンソールを使用して AMI のルートデバイスタイプを判別するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI] をクリックした後、AMI を選択します。
3. 次のように、[Details] (詳細) タブで [Root device type] (ルートデバイスタイプ) の値を確認します。

   - `ebs` — これは EBS-Backed AMI です。
   - `instance store` — これは instance store-backed AMI です。

コマンドラインを使用して AMI のルートデバイスタイプを判別するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- describe-images (AWS CLI)
- Get-EC2Image (AWS Tools for Windows PowerShell)

停止状態

ルートデバイスの EBS ボリュームを持つインスタンスを停止することはできますが、ルートデバイス用のインスタンスストアボリュームを持つインスタンスを停止することはできません。

停止すると、インスタンスの実行が停止します (ステータスが `running` から `stopping` を経て `stopped` に進む)。停止したインスタンスは Amazon EBS で保持されるため、再起動できます。stopping (停止) は terminating (終了) と異なります。terminated インスタンスは再起動できません。ルートデバイス用のインスタンスストアボリュームを持つインスタンスは停止できないため、実行中か終了のいずれかになります。インスタンスが停止している場合に何が行われ、何を実行できるかの詳細については、「インスタンスの停止と起動」を参照してください。

デフォルトのデータストレージと永続性

ルートデバイスにインスタンスストアボリュームを持つインスタンスでは、自動的にインスタンスストアが利用できます (ルートボリュームにルートパーティションが含まれ、追加のデータを保存できます)。1 つまたは複数の EBS ボリュームをアタッチすることで、永続的ストレージをインスタンスに追加できます。インスタンスストアボリューム上のデータは、インスタンスが失敗または終了すると、削除されます。詳細については、インスタンスストアボリュームとデータライフタイム を参照してください。

ルートデバイスに Amazon EBS を持つインスタンスには、自動的に EBS ボリュームがアタッチされます。ボリュームは、他のボリュームと同様に、ボリュームのリストに表示されます。ほとんどのインスタンスタイプでは、ルートデバイスの EBS ボリュームを持つインスタンスは、デフォルトでインスタンスストアボリュームを保持しません。ブロックデバイスマッピングを使用して、インスタンスストアボリュームまたは追加の EBS ボリュームを追加できます。詳細については、ブロックデバイスマッピング を参照してください。

作成時刻

Amazon EBS-backed AMI から起動するインスタンスは、instance store-backed AMI から起動するインスタンスよりも速く起動します。instance store-backed AMI からインスタンスを起動するときは、Amazon S3 からすべてのパートを取得しないとインスタンスを利用できません。Amazon EBS-backed AMI の場合、インスタンスの起動に必要な部分だけをスナップショットから取得するとインスタンスを利用できます。ただし、ルートデバイスに EBS ボリュームを使用するインスタンスのパフォーマンスは、残りの部分がスナップショットから取得され、ボリュームにロードされる少しの時間、遅くなります。インスタンスを停止し、再起動する場合は、状態が EBS ボリュームに保存されているため早く起動します。

AMI の作成

Instance Store-Backed Linux AMI を作成するには、Amazon EC2 AMI ツールを使用して、当該のインスタンス上でインスタンスから AMI を作成する必要があります。

AMI の作成は、Amazon EBS Backed の AMI の方がはるかに簡単です。`CreateImage` API アクションは、Amazon EBS-backed AMI を作成して登録します。AWS Management Console にも、実行中のインスタンスから AMI を作成できるボタンがあります。詳細については、Amazon EBS-backed Linux AMI を作成する を参照してください。

課金方法

Instance Store-Backed の AMI の場合、インスタンスの使用量と Amazon S3 への AMI の保存に対して課金されます。Amazon EBS でバックアップされた AMI の場合、インスタンスの使用料、EBS ボリュームストレージおよび使用量、AMI の EBS スナップショットとしての保存に対して課金されます。

Amazon EC2 Instance Store-Backed の AMI の場合、AMI をカスタマイズしたり、新しい AMI を作成したりするたびに、各 AMI のすべての部分が Amazon S3 に保存されます。そのため、カスタマイズした各 AMI のストレージフットプリントは、AMI の完全なサイズになります。Amazon EBS-Backed の AMI の場合、AMI をカスタマイズしたり、新しい AMI を作成したりするたびに、変更のみが保存されます。そのため、最初の AMI の後にカスタマイズする後続の AMI のストレージフットプリントははるかに小さくなり、AMI ストレージ料金が少なくなります。

ルートデバイスに EBS ボリュームを使用しているインスタンスが停止した場合、インスタンスの使用については課金されませんが、ボリュームストレージについては引き続き課金されます。インスタンスを起動した時点で、最低 1 分間分の使用料が課金されます。1 分経過した後は、使用した秒数のみ課金されます。例えば、インスタンスを 20 秒間実行して停止した場合は、1 分間分課金されます。インスタンスを 3 分 40 秒実行した場合は、ちょうど 3 分 40 秒間分課金されます。インスタンスがアイドル状態で残っていて、そのインスタンスに接続していない場合でも、実行中のインスタンスに対して、1 秒ごとに最低 1 分間分の使用料が課金されます。

Linux AMI 仮想化タイプ

Linux Amazon マシンイメージ では、2 つの仮想化タイプ (準仮想化 (PV) およびハードウェア仮想マシン (HVM)) のどちらかを使用します。PV AMI と HVM AMI の主な違いは、起動の方法と、パフォーマンス向上のための特別なハードウェア拡張機能 (CPU、ネットワーク、ストレージ) を利用できるかどうかという点です。

最適なパフォーマンスを得るために、インスタンスを起動するときには、現行世代のインスタンスタイプと HVM AMI を使用することをお勧めします。現行世代のインスタンスタイプの詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。旧世代のインスタンスタイプを使用中で、アップグレードする場合は、「アップグレードパス」および インスタンスタイプを変更する を参照してください。

以下の表では、HVM と PV AMI を比較しています。

	HVM	PV
説明	HVM AMI は、完全に仮想化された一連のハードウェアを備えており、イメージのルートブロックデバイスのマスターブートレコードを実行することによって起動します。この仮想化タイプでは、ベアメタルハードウェア上でオペレーティングシステムが動作するのと同様に、修正を行わなくても仮想マシン上でオペレーティングシステムを直接実行することができます。Amazon EC2 ホストシステムでは、ゲストに提供されている基盤となるハードウェアの一部またはすべてがエミュレートされます。	PV AMIs は、PV-GRUB と呼ばれる特別なブートローダーを使用して起動します。このブートローダーによって起動サイクルが開始され、イメージの `menu.lst` ファイルで指定されているカーネルがチェーンロードされます。準仮想化のゲストは、仮想化を明示的にサポートしていないホストハードウェアで実行できます。従来、PV のゲストは HVM のゲストよりも多くの場合にパフォーマンスが向上しました。ただし、HVM 仮想化の機能強化や HVM AMI で PV ドライバが利用可能になったことにより、このようなパフォーマンスの向上はなくなりました。PV-GRUB の詳細や Amazon EC2 での使用方法については、「ユーザー提供カーネル」を参照してください。
ハードウェア拡張のサポート	はい。PV のゲストとは異なり、HVM のゲストは、ホストシステム上の基盤となるハードウェアへの高速なアクセスを可能にするハードウェア拡張を利用できます。Amazon EC2 で使用できる CPU 仮想化拡張機能の詳細については、Intel のウェブ	いいえ。拡張ネットワークや GPU 処理などの特別なハードウェア拡張を利用することはできません。

	HVM	PV
	サイトの「Intel Virtualization Technology」を参照してください。HVM AMI は、拡張ネットワーキングと GPU 処理を利用する場合に必要です。専用のネットワークや GPU デバイスに命令を伝達するには、OS がネイティブハードウェアプラットフォームにアクセスできる必要があります。HVM 仮想化ではこのアクセスが可能です。詳細については、「Linux での拡張ネットワーキング」および「Linux 高速コンピューティングインスタンス」を参照してください。	
サポートされるインスタンスタイプ	すべての現行世代のインスタンスタイプは HVM AMI をサポートします。	次の旧世代のインスタンスタイプは、PV AMI をサポートします: C1、C3、M1、M3、M2、および T1。現行世代のインスタンスタイプは PV AMI をサポートしません。
サポートされているリージョン	すべてのリージョンで HVM インスタンスがサポートされています。	アジアパシフィック (東京)、アジアパシフィック (シンガポール)、アジアパシフィック (シドニー)、欧州 (フランクフルト)、欧州 (アイルランド)、南米 (サンパウロ)、US East (N. Virginia)、米国西部 (北カリフォルニア)、米国西部 (オレゴン)

	HVM	PV
検索方法	コンソールまたは describe-images コマンドを使用して、AMI の仮想化タイプが hvm に設定されていることを確認します。詳細については、「Linux AMI の検索」を参照してください。	コンソールまたは describe-images コマンドを使用して、AMI の仮想化タイプが paravirtual に設定されていることを確認します。詳細については、「Linux AMI の検索」を参照してください。

PV on HVM

従来、準仮想化のゲストはストレージやネットワークの操作については、HVM のゲストよりも高いパフォーマンスを実現していました。これは、準仮想化のゲストでは I/O 用の特別なドライバー (ネットワークとディスクのハードウェアをエミュレートする際のオーバーヘッドが回避されます) を活用することができたためです。これに対して、HVM のゲストでは、エミュレートされたハードウェアに対する命令を変換する必要がありました。現在では、PV ドライバーを HVM のゲストで利用できるようになりました。このため、準仮想化された環境で実行するためのができないオペレーティングシステムでも、これらのドライバーを使用することで、ストレージやネットワークの I/O でパフォーマンスの向上を確認することができます。このような PV on HVM ドライバーを使用すると、HVM のゲストで、準仮想化のゲストと同じまたはより優れたパフォーマンスを実現できます。

ブートモード

コンピュータが起動して最初に実行されるソフトウェアが、プラットフォームの初期化を行い、そのプラットフォーム固有の操作を実行するためのオペレーティングシステム用のインタフェースを提供する必要があります。

Amazon EC2 では、統合拡張ファームウェアインターフェイス (UEFI) とレガシー BIOS の、2 種類のブートモードソフトウェアがサポートされます。

AMI で使用可能なブートモードパラメータ

AMI のブートモードパラメータ値には、uefi、legacy-bios または uefi-preferred のどちらかを指定できます。AMI ブートモードパラメータの設定はオプションです。ブートモードパラメータがない AMI の場合、これらの AMI から起動されるインスタンスでは、インスタンスタイプごとのデフォルトのブートモード値が使用されます。

AMI ブートモードパラメータの目的

AMI ブートモードパラメータは、インスタンスの起動時に使用するブートモードを Amazon EC2 に通知します。ブートモードパラメータが `uefi` に設定されている場合、EC2 は UEFI でのインスタンスの起動を試みます。オペレーティングシステムが UEFI をサポートするように設定されていない場合、インスタンスの起動が失敗します。

UEFI Preferred ブートモードパラメータ

`uefi-preferred` ブートモードパラメータを使用して、UEFI とレガシー BIOS の両方をサポートする AMI を作成できます。ブートモードパラメータが `uefi-preferred` に設定されている場合、インスタンスタイプごとの EFI がサポートされている場合、インスタンスは UEFI での起動になります。インスタンスタイプが UEFI をサポートしていない場合、インスタンスはレガシー BIOS で起動されます。

> **Warning**
>
> UEFI セキュアブートなどの一部の機能は、UEFI で起動するインスタンスでのみ使用できます。UEFI をサポートしないインスタンスタイプで `uefi-preferred` AMI ブートモードパラメータを使用すると、インスタンスはレガシー BIOS として起動し、UEFI 依存機能は無効になります。UEFI に依存する機能の可用性を重視する場合は、AMI ブートモードパラメータを `uefi` に設定します。

インスタンスタイプごとのデフォルトのブートモード

- Graviton インスタンスタイプ: UEFI
- Intel および AMD インスタンスタイプ: レガシー BIOS

UEFI で Intel および AMD インスタンスタイプを実行中

Most Intel and AMD instance types は UEFI とレガシー BIOS の両方で実行できます。UEFI を使用するには、ブートモードパラメータを `uefi` または `uefi-preferred` に設定した AMI を選択し、その AMI に含まれるオペレーティングシステムで、UEFI をサポートするための設定を行う必要があります。

ブートモードのトピック

- インスタンスの起動

- AMI のブートモードパラメータを定義する
- インスタンスタイプがサポートしているブートモードを確認する
- インスタンスのブートモードを決定する
- オペレーティングシステムのブートモードを特定する
- AMI のブートモードを設定する
- UEFI 変数
- UEFI セキュアブート

インスタンスの起動

インスタンスの起動には、UEFI またはレガシー BIOS のブートモードがあります。

トピック

- 制限事項
- 考慮事項
- UEFI でインスタンスを起動するための要件

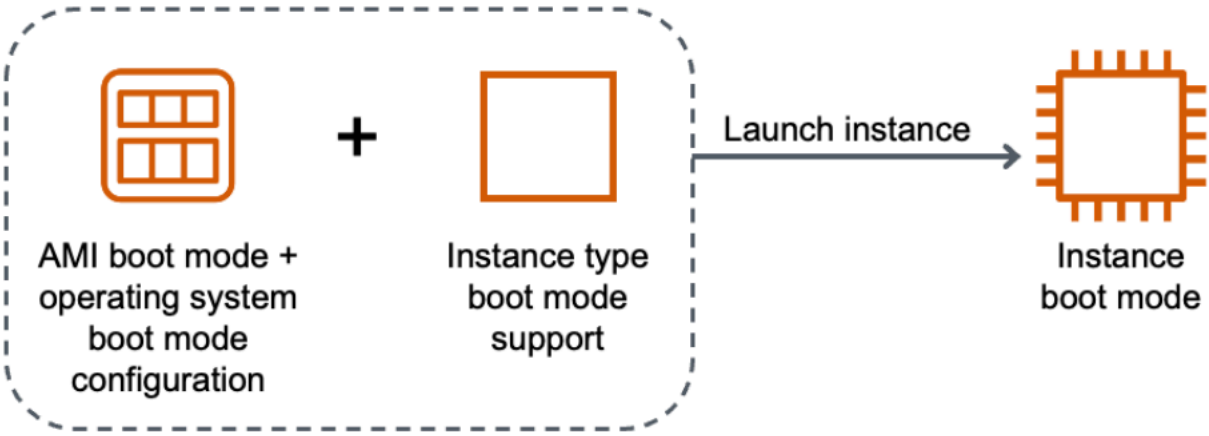
制限事項

Local Zones および Wavelength Zone で、あるいは AWS Outposts を使用する場合、UEFI ブートはサポートされません。

考慮事項

インスタンスの起動には、以下を考慮します。

- インスタンスのブートモードは、次の図に示すように、AMI の設定、その中に含まれるオペレーティングシステム、およびインスタンスタイプによって決まります。



次の表は、インスタンスのブートモード (インスタンス起動モードの結果列で示される) が AMI のブートモードパラメータ (列 1)、AMI に含まれるオペレーティングシステムのブートモード設定 (列 2)、およびインスタンスタイプのブートモードサポート (列 3) の組み合わせによって決定されることを示しています。

AMI ブートモードパラメータ	オペレーティングシステムのブートモード設定	インスタンスタイプのブートモードサポート	インスタンスのブートモードの結果
UEFI	UEFI	UEFI	UEFI
レガシー BIOS	レガシー BIOS	レガシー BIOS	レガシー BIOS
UEFI Preferred	UEFI	UEFI	UEFI
UEFI Preferred	UEFI	UEFI とレガシー BIOS	UEFI
UEFI Preferred	レガシー BIOS	レガシー BIOS	レガシー BIOS
UEFI Preferred	レガシー BIOS	UEFI とレガシー BIOS	レガシー BIOS
ブートモードが指定されていません - ARM	UEFI	UEFI	UEFI

AMI ブートモードパラメータ	オペレーティングシステムのブートモード設定	インスタンスタイプのブートモードサポート	インスタンスのブートモードの結果
ブートモードが指定されていません - x86	レガシー BIOS	UEFI とレガシー BIOS	レガシー BIOS

- デフォルトのブートモード:
  - Graviton インスタンスタイプ: UEFI
  - Intel および AMD インスタンスタイプ: レガシー BIOS
- レガシー BIOS に加えて UEFI をサポートする Intel および AMD インスタンスタイプ:
  - AWS Nitro System に構築されている (ベアメタルインスタンス、DL1、G4ad、P4、u-3tb1、u-6tb1、u-9tb1、u-12tb1、u-18tb1、u-24tb1 および VT1 以外の) すべてのインスタンス

特定のリージョンで UEFI をサポートする、Linux の現在利用可能なインスタンスタイプを表示するには

利用可能なインスタンスタイプは、AWS リージョン ごとに異なります。リージョンで利用可能であり、UEFI をサポートしているインスタンスタイプを確認するには、`--region` パラメータを指定しながら describe-instance-types コマンドを使用します。`--region` パラメータを省略すると、デフォルトのリージョンがリクエストに使用されます。`--filters` パラメータを含めることで結果の範囲を UEFI をサポートするインスタンスタイプに規定し、`--query` パラメータを含めることで出力の範囲を `InstanceType` の値に規定します。

AWS CLI

```
$ aws ec2 describe-instance-types --filters Name=supported-boot-mode,Values=uefi --
query "InstanceTypes[*].[InstanceType]" --output text | sort

a1.2xlarge
a1.4xlarge
a1.large
a1.medium
a1.metal
a1.xlarge
c5.12xlarge
```

```
...
```

PowerShell

```
PS C:\> Get-EC2InstanceType | `
 Where-Object {$_.SupportedBootModes -Contains "uefi"} | `
 Sort-Object InstanceType | `
 Format-Table InstanceType -GroupBy CurrentGeneration

CurrentGeneration: False

InstanceType
------------
a1.2xlarge
a1.4xlarge
a1.large
a1.medium
a1.metal
a1.xlarge

CurrentGeneration: True

InstanceType
------------
c5.12xlarge
c5.18xlarge
c5.24xlarge
c5.2xlarge
c5.4xlarge
c5.9xlarge
...
```

特定のリージョンについて、UEFI Secure Boot をサポートし不揮発性変数を保持する、Linux の利用可能なインスタンスタイプを表示するには

現在、ベアメタルインスタンスは UEFI Secure Boot および不揮発性変数をサポートしていません。describe-instance-types コマンドは、上記の例での説明と同じように実行します。ただし、`Name=bare-metal,Values=false` フィルターを使用してベアメタルインスタンスを除外します。UEFI Secure Boot の詳細については、「UEFI セキュアブート」を参照してください。

AWS CLI

```
$ aws ec2 describe-instance-types --filters Name=supported-boot-mode,Values=uefi
 Name=bare-metal,Values=false --query "InstanceTypes[*].[InstanceType]" --output
 text | sort

a1.2xlarge
a1.4xlarge
a1.large
a1.medium
...
```

PowerShell

```
PS C:\> Get-EC2InstanceType | `
    Where-Object { `
        $_.SupportedBootModes -Contains "uefi" -and `
        $_.BareMetal -eq $False
        } | `
    Sort-Object InstanceType  | `
    Format-Table InstanceType, SupportedBootModes, BareMetal,
 @{Name="SupportedArchitectures";
 Expression={$_.ProcessorInfo.SupportedArchitectures}}

InstanceType     SupportedBootModes  BareMetal SupportedArchitectures
------------     ------------------  --------- ----------------------
a1.2xlarge       {uefi}                  False arm64
a1.4xlarge       {uefi}                  False arm64
a1.large         {uefi}                  False arm64
a1.medium        {uefi}                  False arm64
a1.xlarge        {uefi}                  False arm64
c5.12xlarge      {legacy-bios, uefi}     False x86_64
c5.18xlarge      {legacy-bios, uefi}     False x86_64
```

UEFI でインスタンスを起動するための要件

UEFI 起動モードでインスタンスを起動するには、以下の手順に従って、UEFI をサポートするインスタンスタイプを選択し、AMI とオペレーティングシステムを UEFI 用に設定する必要があります。

インスタンスタイプ

インスタンスを起動する際は、UEFI をサポートするインスタンスタイプを選択する必要があります。詳細については、「インスタンスタイプがサポートしているブートモードを確認する」を参照してください。

AMI

インスタンスを起動する際は、UEFI 用に設定された AMI を選択する必要があります。AMI の設定は次に従います。

- オペレーティングシステム – AMI に含まれるオペレーティングシステムです。UEFI の使用を設定する必要があり、この設定がない場合はインスタンスの起動に失敗します。詳細については、「オペレーティングシステムのブートモードを特定する」を参照してください。
- AMI ブートモードパラメータ - AMI のブートモードパラメータは `uefi` または `uefi-preferred` に設定します。詳細については、「AMI のブートモードパラメータを定義する」を参照してください。

AWS では、Graviton ベースのインスタンスタイプ向けに UEFI をサポートするように設定された Linux AMI のみを提供します。他の UEFI インスタンスタイプで Linux を使用するには、AMI を設定した上で、その AMI を、VM Import/Export 経由または CloudEndure 経由でインポートする必要があります。

サポートされている Windows AMI については、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「UEFI でインスタンスを起動するための要件」を参照してください。

AMI のブートモードパラメータを定義する

AMI ブートモードパラメータの設定はオプションです。AMI のブートモードパラメータ値には、`uefi`、`legacy-bios` または `uefi-preferred` のどちらかを指定できます。

一部の AMI には、ブートモードパラメータがありません。AMI にブートモードパラメータがない場合、AMI から起動されるインスタンスでは、インスタンスタイプごとのデフォルト値が使用されます。Graviton ではこの設定は `uefi` となり、Intel および AMD インスタンスタイプでは `legacy-bios` となります。

Console

AMI のブートモードパラメータを確認するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで [AMI] をクリックした後、AMI を選択します。
3. [ブートモード] フィールドを調べます。

   - uefi という値は、AMI が UEFI をサポートしていることを示します。
   - uefi-preferred という値は、AMI が UEFI とレガシー BIOS の両方をサポートしていることを示します。
   - 値がない場合、AMI から起動されるインスタンスは、インスタンスタイプのデフォルト値を使用します。

インスタンスの起動時に AMI のブートモードパラメータを確認するには (コンソール)

インスタンスの起動ウィザードを使用してインスタンスを起動する場合、AMI を選択するステップで、[ブートモード] フィールドを表示します。詳細については、「アプリケーションと OS イメージ (Amazon マシンイメージ)」を参照してください。

AWS CLI

AMI のブートモードパラメータを確認するには (AWS CLI)

describe-images オペレーションを使用して、AMI のブートモードを確認します。

```
$ aws ec2 describe-images --region us-east-1 --image-id ami-0abcdef1234567890

{
    "Images": [
        {
         ...
            ],
            "EnaSupport": true,
            "Hypervisor": "xen",
            "ImageOwnerAlias": "amazon",
            "Name": "UEFI_Boot_Mode_Enabled-Windows_Server-2016-English-Full-
Base-2020.09.30",
            "RootDeviceName": "/dev/sda1",
            "RootDeviceType": "ebs",
            "SriovNetSupport": "simple",
            "VirtualizationType": "hvm",
            "BootMode":
 "uefi"
        }
    ]
```

```
}
```

出力で、BootMode フィールドは AMI のブートモードが表示します。uefi という値により、その AMI が UEFI をサポートしていることを示します。uefi-preferred の値により、その AMI が UEFI とレガシー BIOS の両方をサポートしていることを示されます。値がない場合、AMI から起動されるインスタンスは、インスタンスタイプのデフォルト値を使用します。

PowerShell

AMI のブートモードパラメータを確認するには (Tools for PowerShell)

Get-EC2Image コマンドレットを使用して、AMI のブートモードを確認します。

```
PS C:\> Get-EC2Image -Region us-east-1 -ImageId ami-0abcdef1234567890 | Format-List
 Name, BootMode, TpmSupport

Name       : TPM-Windows_Server-2016-English-Full-Base-2023.05.10
BootMode   : uefi
TpmSupport : v2.0
```

出力で、BootMode フィールドは AMI のブートモードが表示します。uefi という値により、その AMI が UEFI をサポートしていることを示します。uefi-preferred の値により、その AMI が UEFI とレガシー BIOS の両方をサポートしていることを示されます。値がない場合、AMI から起動されるインスタンスは、インスタンスタイプのデフォルト値を使用します。

インスタンスタイプがサポートしているブートモードを確認する

インスタンスタイプがサポートしているブートモードを確認するには AWS CLI または Tools for PowerShell を使用できます。

インスタンスタイプがサポートしているブートモードを確認するには

インスタンスタイプがサポートしているブートモードを確認するには次の方法を使用できます。

AWS CLI

インスタンスタイプがサポートしているブートモードを確認するには describe-instance-types コマンドを使用できます。--query パラメータを含めることで、出力をフィルタリングできます。この例では、出力がフィルタリングされ、サポートされているブートモードのみが返されます。

次の例では、m5.2xlarge が UEFI ブートモードとレガシー BIOS ブートモードの両方をサポートしていることを示しています。

```
aws ec2 describe-instance-types --region us-east-1 --instance-types m5.2xlarge --
query "InstanceTypes[*].SupportedBootModes"
```

正常な出力:

```
[
    [
        "legacy-bios",
        "uefi"
    ]
]
```

次の例は、t2.xlarge がレガシー BIOS のみをサポートしていることを示しています。

```
aws ec2 describe-instance-types --region us-east-1 --instance-types t2.xlarge --
query "InstanceTypes[*].SupportedBootModes"
```

正常な出力:

```
[
    [
        "legacy-bios"
    ]
]
```

PowerShell

インスタンスタイプがサポートしているブートモードを確認するには Get-EC2InstanceType (Tools for PowerShell) コマンドレットを使用できます。

次の例では、m5.2xlarge が UEFI ブートモードとレガシー BIOS ブートモードの両方をサポートしていることを示しています。

```
Get-EC2InstanceType -Region us-east-1 -InstanceType m5.2xlarge | Format-List
 InstanceType, SupportedBootModes
```

正常な出力:

```
InstanceType       : m5.2xlarge
SupportedBootModes : {legacy-bios, uefi}
```

次の例は、t2.xlarge がレガシー BIOS のみをサポートしていることを示しています。

```
Get-EC2InstanceType -Region us-east-1 -InstanceType t2.xlarge | Format-List
 InstanceType, SupportedBootModes
```

正常な出力:

```
InstanceType       : t2.xlarge
SupportedBootModes : {legacy-bios}
```

インスタンスのブートモードを決定する

インスタンスのブートモードは、Amazon EC2 コンソールの [ブートモード] フィールドに表示され、AWS CLI の currentInstanceBootMode パラメータによって表示されます。

インスタンスの起動時、そのブートモードパラメータの値は、インスタンスの起動に使用された AMI のブートモードパラメータの値によって決まります。

- uefi のブートモードパラメータを持つ AMI は、uefi の currentInstanceBootMode パラメータを持つインスタンスを作成します。
- legacy-bios のブートモードパラメータを持つ AMI は、 legacy-bios の currentInstanceBootMode パラメータを持つインスタンスを作成します。
- uefi-preferred のブートモードパラメータを持つ AMI は、インスタンスタイプが UEFI をサポートしている場合はという uefi の currentInstanceBootMode パラメータを持つインスタンスを作成します。それ以外の場合は、legacy-bios の currentInstanceBootMode パラメータがのインスタンスを作成します。
- ブートモードのパラメータ値を持たない AMI は、AMI アーキテクチャが ARM か x86 か、サポートされているインスタンスタイプのブートモードによって決まる currentInstanceBootMode パラメータ値を持つインスタンスを作成します。デフォルトのブートモードは、Graviton インスタンスタイプでは uefi、Intel と AMD インスタンスタイプでは legacy-bios です。

Console

インスタンスのブートモードを確認するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. [詳細] タブを開き、[ブートモード] フィールドを確認します。

AWS CLI

インスタンスのブートモードを確認するには (AWS CLI)

インスタンスのブートモードを決定するには describe-instances を使用します。インスタンスの作成に使用された AMI のブートモードを確認することもできます。

```
$ aws ec2 describe-instances --region us-east-1 --instance-ids i-1234567890abcdef0

{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-0e2063e7f6dc3bee8",
                    "InstanceId": "i-1234567890abcdef0",
                    "InstanceType": "m5.2xlarge",
                    ...
                    },
                   "BootMode": "uefi",
                   "CurrentInstanceBootMode": "uefi"
                }
            ],
            "OwnerId": "1234567890",
            "ReservationId": "r-1234567890abcdef0"
        }
    ]
}
```

PowerShell

インスタンスのブートモードパラメータを確認するには (Tools for PowerShell)

インスタンスのブートモードを決定するには Get-EC2Image コマンドレットを使用します。インスタンスの作成に使用された AMI のブートモードを確認することもできます。

Get-EC2Image (AWS Tools for Windows PowerShell)

```
(Get-EC2Instance -InstanceId i-1234567890abcdef0).Instances | Format-List BootMode,
 CurrentInstanceBootMode, InstanceType, ImageId

BootMode                : uefi
CurrentInstanceBootMode : uefi
InstanceType            : c5a.large
ImageId                 : ami-0265446f88eb4021b
```

出力では、次のパラメータがブートモードを説明しています。

- BootMode - インスタンスの作成に使用された AMI のブートモード。
- CurrentInstanceBootMode - 起動時または起動時にインスタンスを起動するために使用される起動モード。

オペレーティングシステムのブートモードを特定する

Amazon EC2 のブートモードは AMI のブートモードに従います。このブートモードがインスタンスの起動に使用されます。インスタンスのオペレーティングシステムが UEFI 用に設定されているかどうかを確認するには、SSH もしくは を使用してインスタンスに接続する必要があります。

インスタンスのオペレーティングシステムのブートモードを特定するには

1. SSH を使用しての Linux インスタンスへの接続
2. オペレーティングシステムのブートモードを表示するには、以下のいずれかを実行します。

   - 以下のコマンドを実行します。

     ```
     [ec2-user ~]$ sudo /usr/sbin/efibootmgr
     ```

     UEFI ブートモードで起動されたインスタンスで想定される出力

     ```
     BootCurrent: 0001
     Timeout: 0 seconds
     BootOrder: 0000,0001
     ```

```
Boot0000* UiApp
Boot0001* UEFI Amazon Elastic Block Store vol-xyz
```

- 次のコマンドを実行して、/sys/firmware/efi ディレクトリが存在するか確認します。このディレクトリは、インスタンスが UEFI を使用して起動する場合のみ存在します。このディレクトリが存在しない場合、このコマンドは Legacy BIOS Boot Detected を返します。

```
[ec2-user ~]$ [ -d /sys/firmware/efi ] && echo "UEFI Boot Detected" || echo
 "Legacy BIOS Boot Detected"
```

UEFI ブートモードで起動されたインスタンスで想定される出力

```
UEFI Boot Detected
```

レガシー BIOS ブートモードで起動されたインスタンスで想定される出力

```
Legacy BIOS Boot Detected
```

- 次のコマンドを実行して、dmesg 出力に EFI が 含まれていることを確認します。

```
[ec2-user ~]$ dmesg | grep -i "EFI"
```

UEFI ブートモードで起動されたインスタンスで想定される出力

```
[    0.000000] efi: Getting EFI parameters from FDT:
[    0.000000] efi: EFI v2.70 by EDK II
```

AMI のブートモードを設定する

register-image コマンドを使用して AMI を作成する際に、AMI のブートモードを uefi、legacy-bios または uefi-preferred に設定することができます。

AMI ブートモードを uefi-preferred に設定すると、インスタンスは次のように起動します。

- UEFI とレガシー BIOS の両方をサポートするインスタンスタイプ (例えば m5.large など) の場合、インスタンスは UEFI を使用して起動します。
- レガシー BIOS のみをサポートするインスタンスタイプ (例えば m4.large など) の場合、インスタンスはレガシー BIOS を使用して起動します。

> **Note**
>
> AMI ブートモードを `uefi-preferred` に設定した場合、オペレーティングシステムは UEFI と Legacy BIOS の両方を起動する機能をサポートしている必要があります。現在、register-image コマンドを使用して NitrotPM と UEFI Preferred の両方をサポートする AMI を作成することはできません。

> **Warning**
>
> UEFI セキュアブートなどの一部の機能は、UEFI で起動するインスタンスでのみ使用できます。UEFI をサポートしないインスタンスタイプで `uefi-preferred` AMI ブートモードパラメータを使用すると、インスタンスはレガシー BIOS として起動し、UEFI 依存機能は無効になります。UEFI に依存する機能の可用性を重視する場合は、AMI ブートモードパラメータを `uefi` に設定します。

既存のレガシー BIOS ベースのインスタンスを UEFI に、または既存の UEFI ベースのインスタンスをレガシー BIOS に変換するには、いくつかの手順を実行する必要があります。まず、選択したブートモードをサポートするように、インスタンスのボリュームとオペレーティングシステムを変更します。次に、ボリュームのスナップショットを作成します。最後に、register-image を使用して、スナップショットから AMI を作成します。

create-image コマンドを使用して AMI のブートモードを設定することはできません。create-image を使用すると、AMI には、その作成に使用される EC2 インスタンスのブートモードが継承されます。例えば、レガシー BIOS で実行されている EC2 インスタンスから AMI を作成する場合、その AMI のブートモードは `legacy-bios` として設定されます。ブートモードが `uefi-preferred` に設定された AMI を使用して起動された EC2 インスタンスから AMI を作成すると、作成された AMI のブートモードも `uefi-preferred` に設定されます。

> **Warning**
>
> AMI ブートモードパラメータを設定しても、オペレーティングシステムは指定されたブートモードに自動的に変更されません。これらのステップに進む前に、まずインスタンスのボリュームとオペレーティングシステムに対し、選択したブートモードを使用しての起動をサポートするよう適切な変更を行う必要があります。これを行わないと、作成された AMI は使

用不能になります。ここでの変更内容は、オペレーティングシステムにより異なります。詳細については、オペレーティングシステムのマニュアルを参照してください。

AMI のブートモードを設定するには (AWS CLI)

1. インスタンスのボリュームとオペレーティングシステムに対し、選択したブートモードでの起動をサポートするための適切な変更を加えます。ここでの変更内容は、オペレーティングシステムにより異なります。詳細については、オペレーティングシステムのマニュアルを参照してください。

   > **Note**
   >
   > この手順を実行しないと、AMI は使用不可能になります。

2. インスタンスのボリューム ID を、describe-instances コマンドを使用して確認します。次のステップでは、このボリュームのスナップショットを作成します。

   ```
   aws ec2 describe-instances --region us-east-1 --instance-ids i-1234567890abcdef0
   ```

   正常な出力

   ```
   ...
               "BlockDeviceMappings": [
                           {
                               "DeviceName": "/dev/sda1",
                               "Ebs": {
                                   "AttachTime": "",
                                   "DeleteOnTermination": true,
                                   "Status": "attached",
                                   "VolumeId": "vol-1234567890abcdef0"
                               }
                           }
                           ...
   ```

3. ボリュームのスナップショットを作成するには、create-snapshot コマンドを使用します。前のステップで取得したボリューム ID を使用します。

   ```
   aws ec2 create-snapshot --region us-east-1 --volume-id vol-1234567890abcdef0 --
   description "add text"
   ```

正常な出力

```
{
 "Description": "add text",
 "Encrypted": false,
 "OwnerId": "123",
 "Progress": "",
 "SnapshotId": "snap-01234567890abcdef",
 "StartTime": "",
 "State": "pending",
 "VolumeId": "vol-1234567890abcdef0",
 "VolumeSize": 30,
 "Tags": []
}
```

4. 前のステップで出力されたスナップショット ID を書き留めます。
5. スナップショットの作成状況が completed になるまで待ってから、次のステップに進みます。スナップショットの状態は、describe-snapshots コマンドを使用して照会できます。

```
aws ec2 describe-snapshots --region us-east-1 --snapshot-ids snap-01234567890abcdef
```

出力例

```
{
    "Snapshots": [
        {
            "Description": "This is my snapshot",
            "Encrypted": false,
            "VolumeId": "vol-049df61146c4d7901",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2019-02-28T21:28:32.000Z",
            "Progress": "100%",
            "OwnerId": "012345678910",
            "SnapshotId": "snap-01234567890abcdef",
...
```

6. 新しい AMI を作成するには、register-image コマンドを使用します。前のステップで記録したスナップショット ID を使用します。

- 起動モードを UEFI に設定するには、コマンドに --boot-mode パラメータを追加して uefi を値として指定します。

```
aws ec2 register-image \
   --region us-east-1 \
   --description "add description" \
   --name "add name" \
   --block-device-mappings "DeviceName=/dev/
sda1,Ebs={SnapshotId=snap-01234567890abcdef,DeleteOnTermination=true}" \
   --architecture x86_64 \
   --root-device-name /dev/sda1 \
   --virtualization-type hvm \
   --ena-support \
   --boot-mode uefi
```

- 起動モードを uefi-preferred に設定するには、コマンドに --boot-mode パラメータを追加して uefi-preferred を値として指定します。

```
aws ec2 register-image \
   --region us-east-1 \
   --description "add description" \
   --name "add name" \
   --block-device-mappings "DeviceName=/dev/
sda1,Ebs={SnapshotId=snap-01234567890abcdef,DeleteOnTermination=true}" \
   --architecture x86_64 \
   --root-device-name /dev/sda1 \
   --virtualization-type hvm \
   --ena-support \
   --boot-mode uefi-preferred
```

正常な出力

```
{
"ImageId": "ami-new_ami_123"
}
```

7. 新しく作成した AMI が、前のステップで指定したブートモードに設定されていることを確認するには、describe-images コマンドを使用します。

```
aws ec2 describe-images --region us-east-1 --image-id ami-new_ami_123
```

正常な出力

```
{
  "Images": [
   {
   "Architecture": "x86_64",
   "CreationDate": "2021-01-06T14:31:04.000Z",
   "ImageId": "ami-new_ami_123",
   "ImageLocation": "",
   ...
   "BootMode": "uefi"
   }
   ]
}
```

8. 新しく作成した AMI を使用して、新しいインスタンスを起動します。

   AMI ブートモードが uefi または legacy-bios の場合、この AMI から作成されたインスタンスは AMI と同じブートモードになります。AMI ブートモードが uefi-preferred の場合、インスタンスタイプが UEFI をサポートしていれば、インスタンスは UEFI を使用して起動します。それ以外の場合、インスタンスはレガシー BIOS を使用して起動します。詳細については、「考慮事項」を参照してください。

9. 新しいインスタンスが想定通りのブートモードになっているかは、describe-instances コマンドにより確認できます。

UEFI 変数

ブートモードが UEFI に設定されているインスタンスを起動すると、変数の key-value ストアが作成されます。このストアは、UEFI およびインスタンスオペレーティングシステムで UEFI 変数を格納するために使用できます。

UEFI 変数は、ブートローダーとオペレーティングシステムにより使用されるもので、システムの起動初期における処理を指定します。これにより、オペレーティングシステムは、(ブート順序や UEFI Secure Boot キーの管理など) ブートプロセスに関する特定の設定を行えます。

> **Warning**
>
> インスタンス (およびインスタンス上で実行されている可能性のある任意のソフトウェア) に接続できるユーザー、またはインスタンスで GetInstanceUefiData API を使用するアクセス許可を持つユーザーは誰でも変数を読み取ることができます。パスワードや個人識別情報などの機密データを UEFI 変数ストアに保存しないでください。

UEFI 変数の永続性

- 2022 年 5 月 10 日以前に起動されたインスタンスの UEFI 変数は、再起動または停止時に消去されます。
- 2022 年 5 月 11 日以降に起動されたインスタンスの場合、不揮発性として設定された UEFI 変数であれば、再起動および停止/開始時にも保持されます。
- ベアメタルインスタンスの場合は、インスタンスの停止/開始オペレーションの全体を通して、不揮発性の UEFI 変数は保持されません。

UEFI セキュアブート

UEFI Secure Boot は、長期の使用実績がある Amazon EC2 の安全なブートプロセスを基に構築されたものです。これには詳細な防御が追加されているので、ユーザーは、起動後も持続する脅威からソフトウェアを保護できます。インスタンスが起動できるのは、暗号化キーで署名されたソフトウェアのみになります。キーは、UEFI の不揮発性変数ストアのキーデータベースに保存されています。UEFI Secure Boot は、インスタンスのブートフローが不正な変更を受けることを防止します。

トピック

- UEFI Secure Boot のしくみ
- UEFI Secure Boot サポートを使用して Linux インスタンスを起動する
- Linux インスタンスで UEFI Secure Boot が有効化されているかどうかを検証する
- UEFI Secure Boot をサポートする Linux AMI の作成
- AWS バイナリ BLOB が作成されるしくみ

UEFI Secure Boot のしくみ

UEFI Secure Boot は、UEFI の中で規定された機能であり、これにより、ブートチェーンの状態を検証することができます。ファームウェア自体による初期化後は、暗号を使用した検証が行わ

れた UEFI バイナリコードのみを、実行するように設計されています。これらのバイナリコードには、UEFI ドライバーやメインブートローダーに加え、ブートチェーンによりロードされるコンポーネントも含まれます。

UEFI Secure Boot では、信頼チェーンで使用される 4 つのキーデータベースが指定されています。このデータベースは、UEFI の変数ストアに格納されています。

信頼チェーンには、以下が含まれます。

プラットフォームキー (PK) データベース

PK データベースは信頼チェーンのルートに置かれます。これには、キー交換キー (KEK) データベースを更新する際に信頼チェーンで使用される、単一のパブリック PK キーが含まれています。

PK データベースを変更するためには、プライベート PK キーを使用して、その更新リクエストに署名する必要があります。この変更処理には、空の PK キーの書き込みによる PK データベースの削除も含まれます。

キー交換キー (KEK) データベース

KEK データベースでは、公開 KEK キーがリストされています。これらのキーは、署名データベース (db) と拒否リストデータベース (dbx) を更新する際に、信頼チェーンが使用します。

パブリック KEK データベースを変更するには、プライベート PK キーを使用して、更新のリクエストに署名をする必要があります。

署名 (DB) データベース

db データベースには、すべての UEFI ブートバイナリを検証するために信頼チェーンが使用する、パブリックキーとハッシュがリストされています。

db データベースを変更するには、プライベート PK キーまたはプライベート KEK キーを使用して、更新リクエストに署名する必要があります。

署名拒否リスト (dbx) データベース

dbx データベースは、信頼されていないパブリックキーとバイナリハッシュをリストします。このリストは、信頼チェーンが失効ファイルとして使用します。

dbx データベースは、常に、他のすべてのキーデータベースよりも優先されます。

dbx データベースを変更するには、プライベート PK キーまたはプライベート KEK キーを使用して、その更新リクエストに署名する必要があります。

UEFI フォーラム (https://uefi.org/revocationlistfile) には、既知の問題のあるバイナリコードと証明書を多数リストした dbx が公開されており、いつでも使用できます。

> **Important**
>
> UEFI Secure Boot は、任意の UEFI バイナリでシグニチャ検証を適用します。UEFI Secure Boot 内での UEFI バイナリの実行を許可するには、上記で説明したいずれかのプライベート db キーを使用して、そのバイナリに署名します。

デフォルトでは、UEFI Secure Boot は無効になっており、システムは `SetupMode` になっています。`SetupMode` の状態であるシステムでは、暗号による署名なしですべてのキー変数の更新が可能です。PK が設定されると、UEFI Secure Boot が有効化されるとともに、SetupMode が取り消されます。

UEFI Secure Boot サポートを使用して Linux インスタンスを起動する

以下の前提条件の下でインスタンスを起動する際、インスタンスは UEFI Secure Boot データベースに基づき UEFI ブートのバイナリコードを自動的に検証します。UEFI Secure Boot は、インスタンスを起動した後に設定することもできます。

> **Note**
>
> UEFI Secure Boot は、インスタンスとそのオペレーティングシステムについて、ブートフローが変更されないように保護します。通常、UEFI Secure Boot の設定は、AMI の一部に含まれています。ベースの AMI とは異なるパラメータで (例えば、AMI 内の `UefiData` を変更して) 新しい AMI を作成する場合には、UEFI Secure Boot を無効にできます。

Linux インスタンスの前提条件

AMI

UEFI Secure Boot が有効になっている AMI が必要です。

Amazon Linux は、AL2023 リリース 2023.1 から UEFI セキュアブートをサポートしています。ただし UEFI セキュアブートは、デフォルトの AMI では有効になっていません。詳細については、「AL2023 ユーザーガイド」の「UEFI Secure Boot」を参照してください。古いバージョン

の Amazon Linux AMI では、UEFI セキュアブートは有効ではありません。サポートされている AMI を使用するには、独自の Linux AMI でいくつかの設定手順を実行する必要があります。詳細については、「UEFI Secure Boot をサポートする Linux AMI の作成」を参照してください。

インスタンスタイプ

- サポート対象: UEFI をサポートするすべての仮想インスタンスタイプは、UEFI Secure Boot もサポートします。UEFI Secure Boot をサポートするインスタンスタイプについては、考慮事項を参照してください。
- サポートなし: ベアメタルインスタンスタイプは、UEFI Secure Boot をサポートしていません。

Linux インスタンスでの前提条件については、「Amazon EC2 Linux インスタンス用ユーザーガイド」の「UEFI Secure Boot サポートを使用して Windows インスタンスを起動する」を参照してください。

Linux インスタンスで UEFI Secure Boot が有効化されているかどうかを検証する

Linux インスタンスが UEFI Secure Boot に対して有効になっているかどうかを確認するには `mokutil` ユティリティを使用できます。`mokutil` がインスタンスにインストールされていない場合は、これをインストールする必要があります。Amazon Linux のインストール手順については、「Amazon Linux インスタンスでのソフトウェアパッケージのインストール」を参照してください。その他のディストリビューションについては、それぞれのドキュメントを参照してください。

Linux インスタンスが UEFI Secure Boot に対して有効になっているかどうかを確認するには

インスタンスの `root` から、次のコマンドを実行します。

```
mokutil --sb-state
```

正常な出力:

- UEFI Secure Boot が有効な場合、出力には `SecureBoot enabled` が含まれます。
- UEFI Secure Boot が有効化されていない場合は、出力に `SecureBoot disabled` または `Failed to read SecureBoot` が含まれます。

Windows インスタンスが有効化されているかどうかを確認するには、「Amazon EC2 Windows インスタンス用ユーザーガイド」の「Verify whether a Windows instance is supported for UEFI Secure

Boot」(Windows インスタンスが UEFI Secure Boot をサポートしているかどうかを確認する) を参照してください。

UEFI Secure Boot をサポートする Linux AMI の作成

以下の手順で、カスタムメイドのプライベートキーを使用して、Secure Boot 用に独自の UEFI 変数ストアを作成する方法について説明します。Amazon Linux は、AL2023 リリース 2023.1 から UEFI セキュアブートをサポートしています。詳細については、「AL2023 ユーザーガイド」の「UEFI Secure Boot」を参照してください。

> **Important**
>
> 以下に示した、UEFI Secure Boot をサポートする AMI を作成するための手順では、上級ユーザーのみを対象としています。これらの手順を使用するには、SSL および Linux ディストリビューションのブートフローに関する十分な知識が必要です。

前提条件

- 以下のツールを使用します。
  - OpenSSL – https://www.openssl.org/
  - efivar – https://github.com/rhboot/efivar
  - efitools – https://git.kernel.org/pub/scm/linux/kernel/git/jejb/efitools.git/
  - get-instance-uefi-data AWS CLI コマンド
- Linux インスタンスは、UEFI ブートモードをサポートする Linux AMI で起動され、不揮発性データを使用している必要があります。

UEFI Secure Boot キーを使用せずに新規で作成されたインスタンスには、`SetupMode` が適用されます。この状態で、独自のキーを登録することが可能です。一部の AMI では UEFI Secure Boot が事前設定されており、既定のキーを変更することはできません。AMI のキーを変更する場合は、その AMI をベースに、新らたな AMI を作成する必要があります。

変数ストア内のキーを伝達するには 2 つの方法があり、それぞれを、以下の オプション A とオプション B の中で説明します。オプション A では、実際のハードウェアのフローを模倣することで、インスタンス内からこれを行う方法について説明します。オプション B では、AMI の作成時に base64 でエンコードされたファイルとして渡される、バイナリ BLOB を作成する方法について説明

します。どちらのオプションでも、最初に、信頼チェーンに使用するためのキーペアを 3 つ作成する必要があります。

UEFI Secure Boot をサポートする Linux AMI を作成するには、まず 3 つのキーペアを作成し、オプション A またはオプション B のいずれかを実行します。

- 3 つのキーペアを作成する
- オプション A: インスタンス内から変数ストアにキーを追加する
- オプション B: 値が事前に設定済みの変数ストアを含むバイナリ BLOB を作成する

> **Note**
>
> ここでの手順は、Linux AMI を作成する場合にのみ使用が可能です。Windows AMI が必要な場合は、サポートされている Windows AMI のいずれかを使用します。詳細については、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「UEFI Secure Boot サポートを使用して Windows インスタンスを起動する」を参照してください。

3 つのキーペアを作成する

UEFI Secure Boot は、プラットフォームキー (PK)、キー交換キー (KEK)、署名データベース (db) という 3 つのキーデータベースに基づいおり、これらのデータベースは信頼チェーンで使用されます。

インスタンスでは、これらの各キーを作成する必要があります。UEFI Secure Boot 標準が有効な形式でパブリックキーの準備を行うには、それぞれのキー用に証明書を作成します。DER では SSL 形式 (バイナリエンコード用の形式) を定義しています。その後、各証明書を UEFI 署名リストに変換します。このリストはバイナリ形式で、UEFI Secure Boot による解析が可能です。最後に、関連するキーで各証明書に署名します。

トピック

- キーペアの作成を準備するには
- キーペア 1: プラットフォームキー (PK) を作成します。
- キーペア 2: キー交換キー (KEK) を作成します
- キーペア 3: 署名データベース (db) を作成します
- ブートイメージ (カーネル) にプライベートキーで署名する

キーペアの作成を準備するには

キーペアを作成する前に、キー生成処理で使用するための、グローバルで一意の識別子 (GUID) を作成します。

1. インスタンスに接続します。
2. シェルプロンプトで、次のコマンドを実行します。

```
uuidgen --random > GUID.txt
```

キーペア 1: プラットフォームキー (PK) を作成します。

PK は UEFI Secure Boot インスタンスの信頼のルートです。プライベート PK は KEK を更新するために使用されます。このキーはその後、認証されたキーを署名データベース (db) に追加するためにも使用されます。

キーペアの作成には、X.509 標準が適用されます。標準の詳細については、ウィキペディアで「X.509」を参照してください。

PK を作成するには

1. キーを作成します。変数 PK に名前を付ける必要があります

```
openssl req -newkey rsa:4096 -nodes -keyout PK.key -new -x509 -sha256 -days 3650 -
subj "/CN=Platform key/" -out PK.crt
```

以下の各パラメータが指定されます。

- `-keyout PK.key` – プライベートキーファイル。
- `-days 3650` – 証明書が有効な日数。
- `-out PK.crt` – UEFI 変数の作成に使用される証明書。
- CN=*Platform key* – キーの共通名 (CN)。ここでは、*Platform key* の代わりに組織の独自の名前を入力できます。

2. 証明書を作成します。

```
openssl x509 -outform DER -in PK.crt -out PK.cer
```

3. 証明書を UEFI 署名リストに変換します。

```
cert-to-efi-sig-list -g "$(< GUID.txt)" PK.crt PK.esl
```

4. UEFI 署名リストに (自己署名の) プライベート PKで署名します。

```
sign-efi-sig-list -g "$(< GUID.txt)" -k PK.key -c PK.crt PK PK.esl PK.auth
```

キーペア 2: キー交換キー (KEK) を作成します

プライベート KEK は db にキーを追加するために使用されます。このデータベースは、システム上で起動が許可された署名のリストです。

KEK を作成するには

1. キーを作成します。

```
openssl req -newkey rsa:4096 -nodes -keyout KEK.key -new -x509 -sha256 -days 3650 -
subj "/CN=Key Exchange Key/" -out KEK.crt
```

2. 証明書を作成します。

```
openssl x509 -outform DER -in KEK.crt -out KEK.cer
```

3. 証明書を UEFI 署名リストに変換します。

```
cert-to-efi-sig-list -g "$(< GUID.txt)" KEK.crt KEK.esl
```

4. プライベート PK で署名リストに署名します。

```
sign-efi-sig-list -g "$(< GUID.txt)" -k PK.key -c PK.crt KEK KEK.esl KEK.auth
```

キーペア 3: 署名データベース (db) を作成します

db リストには、システム上で起動することが認証されているキーが記載されています。このリストを変更する場合は、プライベート KEK が必要です。ブートイメージは、このステップで作成したプライベートキーで署名されます。

db を作成するには

1. キーを作成します。

```
openssl req -newkey rsa:4096 -nodes -keyout db.key -new -x509 -sha256 -days 3650 -
subj "/CN=Signature Database key/" -out db.crt
```

2. 証明書を作成します。

```
openssl x509 -outform DER -in db.crt -out db.cer
```

3. 証明書を UEFI 署名リストに変換します。

```
cert-to-efi-sig-list -g "$(< GUID.txt)" db.crt db.esl
```

4. プライベート KEK を使用して署名リストに署名します。

```
sign-efi-sig-list -g "$(< GUID.txt)" -k KEK.key -c KEK.crt db db.esl db.auth
```

ブートイメージ (カーネル) にプライベートキーで署名する

Ubuntu 22.04 の場合、以下のイメージに署名が必要です。

```
/boot/efi/EFI/ubuntu/shimx64.efi
/boot/efi/EFI/ubuntu/mmx64.efi
/boot/efi/EFI/ubuntu/grubx64.efi
/boot/vmlinuz
```

イメージに署名するには

イメージに署名するには、以下の構文を使用します。

```
sbsign --key db.key --cert db.crt --output /boot/vmlinuz /boot/vmlinuz
```

> **Note**
>
> 署名は、すべての新しいカーネルに対して行う必要があります。通常、*/boot/vmlinuz* は、最近インストールしたカーネルへのシンボリックリンクとなります。

ブートチェーンおよび必要なイメージは、ディストリビューションのドキュメントで確認してください。

[1] ArchWiki コミュニティの皆さんから提供された、すべての作業に感謝します。PK の作成、KEK の作成、DB の作成、およびイメージへの署名のためのコマンドは、Creating keys から提供されており、ArchWiki メンテナンスチームおよび (または) ArchWiki のコントリビューターによって作成されたものです。

オプション A: インスタンス内から変数ストアにキーを追加する

3 つのキーペアの作成が完了すると、以下の手順により、インスタンスに接続し、インスタンス内から変数ストアにキーを追加できるようになります。

オプション A の手順:

- ステップ 1: UEFI Secure Boot をサポートするインスタンスを起動する
- ステップ 2: UEFI Secure Boot をサポートするようにインスタンスを設定する
- ステップ 3: インスタンスから AMI を作成する

ステップ 1: UEFI Secure Boot をサポートするインスタンスを起動する

次の前提条件の下でインスタンスを起動することで、UEFI Secure Boot をサポートするようにインスタンスを構成する準備が整います。インスタンスでの UEFI Secure Boot のサポートは、起動時にみ有効化が可能で、その後に有効にすることはできません。

前提条件

- AMI - Linux AMI では、UEFI ブートモードをサポートする必要があります。AMI が UEFI ブートモードをサポートしていることを確認するには、AMI ブートモードパラメータで uefi を指定する必要があります。詳細については、「AMI のブートモードパラメータを定義する」を参照してください。

  AWS では、Graviton ベースのインスタンスタイプ向けに UEFI をサポートするように構成された Linux AMI のみを提供します。AWS では、UEFI ブートモードをサポートする x86_64 Linux AMI を提供していません。すべてのアーキテクチャで UEFI ブートモードをサポートするように独自の AMI を構成できます。UEFI ブートモードをサポートするように AMI を設定するには、独自の AMI に対して、複数の設定手順を実行する必要があります。詳細については、「AMI のブートモードを設定する」を参照してください。
- インスタンスタイプ — UEFI をサポートするすべての仮想インスタンスタイプは、UEFI Secure Boot もサポートします。ベアメタルインスタンスタイプは UEFI Secure Boot をサポートしていません。UEFI Secure Boot をサポートするインスタンスタイプについては、考慮事項 を参照してください。

- UEFI Secure Boot の立ち上げ後に、インスタンスの起動を行います。UEFI Secure Boot のサポートが可能なのは、2022 年 5 月 10 日 (UEFI Secure Boot リリース日) より後に起動されたインスタンスのみです。

インスタンスを起動したら、UEFI データが存在するかどうかを調べ、UEFI Secure Boot のサポートを設定できる状態であることを確認します (ステップ 2 に進みます)。UEFI データが見つかれば、不揮発性データが保持されていることになります。

インスタンスがステップ 2 に進める状態かどうかを確認するには

get-instance-uefi-data コマンドを使用して、インスタンス ID を指定します。

```
aws ec2 get-instance-uefi-data --instance-id i-0123456789example
```

出力に UEFI データが含まれていれば、そのインスタンスはステップ 2 に進める状態です。空の出力が表示される場合、そのインスタンスで UEFI Secure Boot をサポートする設定は行えません。この状態は、UEFI Secure Boot サポートが利用可能になる前に起動されたインスタンスで発生します。この場合、新しいインスタンスを起動して再試行します。

ステップ 2: UEFI Secure Boot をサポートするようにインスタンスを設定する

インスタンスの UEFI 変数ストアにキーペアを登録します。

> **Warning**
>
> ブートイメージへの署名は、キーの登録後に行う必要があります。そうしないと、インスタンスを起動できなくなります。

署名済みの UEFI 署名リスト (PK、KEK、および db) は、作成後に UEFI ファームウェアに登録する必要があります。

PK 変数に対する書き込みは、以下の場合にのみ行うことができます。

- PK が未登録 (SetupMode 変数が 1) の状態。これは、次のコマンドを使用して確認します。1 または 0 のどちらかが出力されている。

  ```
  efivar -d -n 8be4df61-93ca-11d2-aa0d-00e098032b8c-SetupMode
  ```

- 新しい PK が、既存の PK のプライベートキーによって署名されている。

UEFI 変数ストアにキーを登録するには

インスタンスで以下のコマンドを実行する必要があります。

SetupMode が有効 (値が 1) になっていれば、インスタンスで以下のコマンドを実行することでキーを登録できます。

```
[ec2-user ~]$ efi-updatevar -f db.auth db
```

```
[ec2-user ~]$ efi-updatevar -f KEK.auth KEK
```

```
[ec2-user ~]$ efi-updatevar -f PK.auth PK
```

UEFI Secure Boot が有効になっていることを確認するには

UEFI Secure Boot が有効であることを確認するには、「Linux インスタンスで UEFI Secure Boot が有効化されているかどうかを検証する」に示した手順に従います。

この段階で、get-instance-uefi-data CLI コマンドにより UEFI 変数ストアをエクスポートすることが可能です。あるいは、次のステップに進みブートイメージに署名して、それを UEFI Secure Boot-対応のインスタンスで再起動できます。

ステップ 3: インスタンスから AMI を作成する

インスタンスから AMI を作成するには、コンソールまたは `CreateImage` API、CLI、または SDK を使用します。コンソールでの手順については、「Amazon EBS-backed Linux AMI を作成する」を参照してください。API での手順については、「CreateImage」を参照してください。

> **Note**
>
> `CreateImage` API は、インスタンスの UEFI 変数ストアを AMI に自動的にコピーします。コンソールは `CreateImage` API を使用します。この AMI を使用してインスタンスを起動した場合、インスタンスにも AMI と同じ UEFI 変数ストアが含まれます。

オプション B: 値が事前に設定済みの変数ストアを含むバイナリ BLOB を作成する

3 つのキーペアの作成後、事前に値が設定され UEFI Secure Boot キーが指定された変数ストアを含む、バイナリ BLOB を作成できます。

> **Warning**
>
> ブートイメージには、キーを登録する前に署名を行う必要があります。そうしないと、インスタンスを起動できなくなります。

オプション B の手順:

- ステップ 1: 変数ストアを新規で作成するか既存の変数ストアを更新する
- ステップ 2: AMI の作成時にバイナリ BLOB をアップロードする

ステップ 1: 変数ストアを新規で作成するか既存の変数ストアを更新する

python-uefivars ツールを使用すると、インスタンスを実行せずに変数ストアをオフラインで作成できます。このツールでは、キーから新しい変数ストアを作成できます。現在、このスクリプトでは、EDK2 形式、AWS 形式、および上位レベルのツールで編集しやすい JSON 表現がサポートされています。

インスタンスを実行せずに変数ストアをオフラインで作成するには

1. 次のリンクからツールをダウンロードします。

   ```
   https://github.com/awslabs/python-uefivars
   ```

2. 次のコマンドを実行して、キーから新しい変数ストアを作成します。これにより、base64 でエンコードされたバイナリ BLOB が、*your_binary_blob* bin として作成されます。このツールでは、-I パラメータ経由でバイナリ BLOB を更新することもできます。

   ```
   ./uefivars.py -i none -o aws -O your_binary_blob.bin -P PK.esl -K KEK.esl --db
    db.esl --dbx dbx.esl
   ```

ステップ 2: AMI の作成時にバイナリ BLOB をアップロードする

register-image により、UEFI 変数ストアデータを渡します。--uefi-data パラメータではバイナリ BLOB を指定し、また --boot-mode パラメータでは uefi を指定します。

```
aws ec2 register-image \
    --name uefi_sb_tpm_register_image_test \
```

```
    --uefi-data $(cat your_binary_blob.bin) \
    --block-device-mappings "DeviceName=/dev/sda1,Ebs=
{SnapshotId=snap-0123456789example,DeleteOnTermination=true}" \
    --architecture x86_64 \
    --root-device-name /dev/sda1 \
    --virtualization-type hvm \
    --ena-support \
    --boot-mode uefi
```

AWS バイナリ BLOB が作成されるしくみ

次の手順により、AMI の作成中に UEFI Secure Boot 変数をカスタマイズすることができます。この手順で使用される KEK は、2021 年 9 月現在のものです。Microsoft により KEK が更新された場合は、その最新の KEK を使用する必要があります。

AWS でバイナリ BLOB を作成するには

1. 空の PK 署名リストを作成します。

   ```
   touch empty_key.crt
   cert-to-efi-sig-list empty_key.crt PK.esl
   ```

2. KEK 証明書をダウンロードします。

   ```
   https://go.microsoft.com/fwlink/?LinkId=321185
   ```

3. UEFI 署名リスト (`siglist`) で KEK 証明書をラップします。

   ```
   sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output
    MS_Win_KEK.esl MicCorKEKCA2011_2011-06-24.crt
   ```

4. Microsoft から db 証明書をダウンロードします。

   ```
   https://www.microsoft.com/pkiops/certs/MicWinProPCA2011_2011-10-19.crt
   https://www.microsoft.com/pkiops/certs/MicCorUEFCA2011_2011-06-27.crt
   ```

5. db 署名リストを生成します。

   ```
   sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output
    MS_Win_db.esl MicWinProPCA2011_2011-10-19.crt
   sbsiglist --owner 77fa9abd-0359-4d32-bd60-28f4e78f784b --type x509 --output
    MS_UEFI_db.esl MicCorUEFCA2011_2011-06-27.crt
   ```

```
cat MS_Win_db.esl MS_UEFI_db.esl > MS_db.esl
```

6. 次のリンクから、更新された dbx 変更リクエストをダウンロードします。

```
https://uefi.org/revocationlistfile
```

7. 前のステップでダウンロードした dbx 変更リクエストには、既に Microsoft KEK を使用した署名が行われているので、これを抽出または解凍する必要があります。以下のリンクを使用できます。

```
https://gist.github.com/out0xb2/f8e0bae94214889a89ac67fceb37f8c0
```

```
https://support.microsoft.com/en-us/topic/microsoft-guidance-for-applying-secure-boot-dbx-update-e3b9e4cb-a330-b3ba-a602-15083965d9ca
```

8. uefivars.py スクリプトを使用して UEFI 変数ストアを作成します。

```
./uefivars.py -i none -o aws -O uefiblob-microsoft-keys-empty-pk.bin -P ~/PK.esl -K
 ~/MS_Win_KEK.esl --db ~/MS_db.esl  --dbx ~/dbx-2021-April.bin
```

9. バイナリ BLOB と UEFI 変数ストアを確認します。

```
./uefivars.py -i aws -I uefiblob-microsoft-keys-empty-pk.bin -o json | less
```

10. 再度、同じツールに渡すと、BLOB を更新できます。

```
./uefivars.py -i aws -I uefiblob-microsoft-keys-empty-pk.bin -o aws -O uefiblob-
microsoft-keys-empty-pk.bin -P ~/PK.esl -K ~/MS_Win_KEK.esl --db ~/MS_db.esl  --dbx
 ~/dbx-2021-April.bin
```

正常な出力

```
Replacing PK
Replacing KEK
Replacing db
Replacing dbx
```

Linux AMI の検索

インスタンスを起動するには、インスタンスの起動元となる AMI を選択する必要があります。AMI を選択するときに、起動するインスタンスに関して、次の要件を検討します。

- リージョン - AMI ID は各 AWS リージョンで固有です。
- オペレーティングシステム
- アーキテクチャ: 32 ビット (i386)、64 ビット (x86_64)、または 64 ビット ARM (arm64)
- ルートデバイスタイプ: Amazon EBS またはインスタンスストア
- プロバイダー (Amazon Web Services など)
- 追加のソフトウェア (SQL Server など)

Windows AMI の検索方法については、「Windows インスタンスの Amazon EC2 ユーザーガイド」の「Windows AMI の検索」を参照してください。https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/finding-an-ami.html

AL2023 AMI を検索するには、「AL2023 ユーザーガイド」の「AL2023 の使用を開始する」を参照してください。

Ubuntu AMI の検索方法については、「EC2 AMI ロケータ」を参照してください。

RedHat AMI の検索方法については、RHEL ナレッジベース記事を参照してください。

[Linux] AMI トピックの検索

- Amazon EC2 コンソールを使用した Linux AMI の検索
- AWS CLI を使用した AMI の検索
- Systems Manager を使用して最新の Amazon Linux AMI を検索するには
- Systems Manager パラメータを使用した AMI の検索

Amazon EC2 コンソールを使用した Linux AMI の検索

Amazon EC2 コンソールを使用して、Linux/ AMIを検索できます。インスタンス起動ウィザードを使用してインスタンスを起動するときに、AMI のリストから目的のインスタンスを選択できます。また、[Images] (イメージ) ページを使用して使用可能なすべての AMI を検索することもできます。AMI ID は各 AWS リージョンで固有です。

インスタンス起動ウィザードを使用して Linux AMI を検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択します。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. コンソールダッシュボードで [インスタンスの作成] を選択します。
4. (新しいコンソール) [Application and OS Images (Amazon Machine Image)] (アプリケーションと OS イメージ (Amazon マシンイメージ)にある [Quick Start] (クイックスタート) を選択して、インスタンスのオペレーティングシステム (OS) を選択し、その後[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) で、よく使用されている AMI のいずれかをリストから選択します。必要な AMI が表示されていない場合は、[Browser more AMIs] (その他の AMI を閲覧) を選択して、AMI の全カタログを参照します。詳細については、アプリケーションと OS イメージ (Amazon マシンイメージ) を参照してください。

   (旧コンソール) [Quick Start] (クイックスタート) タブで、よく使用されている AMI のいずれかをリストから選択します。使用する AMI が表示されていない場合は、[My AMIs] (マイ AMI)、[AWS Marketplace]、[Community AMIs] (コミュニティ AMI) のいずれかのタブを開き、目的の AMI を探します。詳細については、ステップ 1: Amazon Machine Image (AMI) を選択する を参照してください。

AMI ページを使用して Linux AMI を検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択します。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. ナビゲーションペインで [AMIs] を選択します。
4. (オプション) フィルターと検索オプションを使用すると、表示される AMI を限定して、条件に一致する AMI のみを表示できます。例えば、AWS が提供するすべての Linux AMI を表示するには、[Public images] (パブリックイメージ) をクリックします。次に、検索オプションを使用して、AMI のリストに表示される範囲を指定します。

   検索バーをクリックし、メニューから [Owner alias] (所有者エイリアス)、[=] 演算子の順に選択し、値として [Amazon] を選択します。検索バーをもう一度クリックし、[Platform] (プラットフォーム)、[=] 演算子の順に選択し、表示されたリストからオペレーティングシステムを選択します。

5. (オプション) [設定] アイコンを選択して、ルートデバイスタイプなど、表示するイメージ属性を選択します。あるいは、一覧から AMI を選択し、[Details] (詳細) タブにそのプロパティを表示できます。
6. AMI を選択する前に、その AMI が Instance Store-Backed と Amazon EBS-Backed のどちらであるかを確認し、その違いを認識しておくことが重要です。詳細については、「ルートデバイスのストレージ」を参照してください。
7. この AMI からインスタンスを起動するには、インスタンスを選択し、[イメージからのインスタンスの起動] を選択します。コンソールを使用してインスタンスを起動する方法については、新しいインスタンス起動ウィザードを使用してインスタンスを起動する を参照してください。まだインスタンスを起動する準備ができていない場合は、後で使用するために AMI ID を記録します。

AL2023 AMI の見つけ方については、「AL2023 ユーザーガイド」の「AL2023 の使用を開始する」を参照してください。

AWS CLI を使用した AMI の検索

Amazon EC2 の AWS Systems Manager コマンドを使用すると、要件に一致する Linux AMI のみを表示できます。要件に一致する AMI が見つかったら、インスタンスの起動に使用できるようにその ID をメモしておきます。詳細については、[AWS Command Line Interface User Guide] (ユーザーガイド) の[Launch your instance] (インスタンスを起動) を参照してください。

describe-images コマンドは、フィルタリングパラメータをサポートしています。例えば、Amazon が所有するパブリック AMI を表示するのに `--owners` パラメーターを使用します。

```
aws ec2 describe-images --owners self amazon
```

Amazon EBS-backed AMI のみを表示するには、上記のコマンドに以下のフィルターを追加します。

```
--filters "Name=root-device-type,Values=ebs"
```

> **Important**
>
> describe-images コマンドから `--owners` フラグを省略すると、所有権に関係なく、起動許可を持つすべてのイメージが返されます。

Systems Manager を使用して最新の Amazon Linux AMI を検索するには

Amazon EC2 では、AWS で保持されるパブリック AMI 用の AWS Systems Manager パブリックパラメータが提供されており、インスタンスの起動時に使用することができます。例えば、EC2 で提供されている `/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2` パラメータはすべてのリージョンで使用でき、特定のリージョンにある最新バージョンの Amazon Linux 2 AMI を常にポイントします。

AWS Systems Manager を使用して最新の AL2023 AMI を検索するには、「AL2023 の使用を開始する」を参照してください。

Amazon EC2 AMI のパブリックパラメータは、次のパスから使用できます。

`/aws/service/ami-amazon-linux-latest`

現在の AWS リージョンにあるすべての Linux および 向け AMI を表示するには、次の AWS CLI コマンドを実行します。

```
aws ssm get-parameters-by-path --path /aws/service/ami-amazon-linux-latest --query
 "Parameters[].Name"
```

パブリックパラメータを使用してインスタンスを作成するには

次の例では、EC2 で提供されているパブリックパラメータを使用して、最新の Amazon Linux 2 AMI で `m5.xlarge` インスタンスを作成します。

このパラメータをコマンドで指定するには、`resolve:ssm:public-parameter` 構文を使用します。`resolve:ssm` は標準のプレフィクス、`public-parameter` はパブリックパラメータのパスと名前です。

この例では、`--count` パラメータと `--security-group` パラメータは含まれていません。`--count` はデフォルトで 1 になります。デフォルトの VPC とデフォルトのセキュリティグループがある場合は、これらが使用されます。

```
aws ec2 run-instances
    --image-id resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-
gp2
    --instance-type m5.xlarge
    --key-name MyKeyPair
```

詳細については、AWS Systems Manager ユーザーガイドの「パブリックパラメータを使用する」および「AWS Systems Manager Parameter Store を使用して最新の Amazon Linux AMI ID をクエリする」「」を参照してください。

Systems Manager パラメータを使用した AMI の検索

コンソールで EC2 インスタンス起動ウィザードを使用してインスタンスを起動する場合は、リストから AMI を選択するか、AMI ID を指す AWS Systems Manager パラメータを選択します。オートメーションコードを使用してインスタンスを作成する場合は、AMI ID の代わりに Systems Manager パラメータを指定できます。

Systems Manager パラメータは、Systems Manager パラメータストアで作成できるユーザー定義のキーと値のペアです。パラメータストアは、アプリケーションの設定値を外部化するための一元的なストアを提供します。詳細については、AWS Systems Manager ユーザーガイドの「AWS Systems Manager Parameter Store」を参照してください。

AMI ID をポイントするパラメータを作成する場合は、データ型に `aws:ec2:image` を指定してください。このデータ型を指定すると、パラメータの作成時または変更時にパラメータ値が AMI ID として検証されます。詳細については、AWS Systems Manager ユーザーガイドの「Amazon マシンイメージ ID のパラメータのネイティブサポート」を参照してください。

Systems Manager パラメータのトピック

- ユースケース
- アクセス許可
- 制限事項
- Systems Manager パラメータを使用したインスタンスの起動

ユースケース

Systems Manager パラメータを使用して AMI ID を指すようにすると、ユーザーがインスタンスの起動時に適切な AMI を簡単に選択できるようになります。Systems Manager パラメータにより、オートメーションコードのメンテナンスを簡素化することもできます。

ユーザーの利便性の向上

特定の AMI を使用してインスタンスを作成する必要があり、その AMI を定期的に更新する場合は、AMI を見つけるために Systems Manager パラメータを選択するようにユーザーに求めることを

お勧めします。ユーザーに Systems Manager パラメータを選択するように求めると、インスタンスの起動に最新の AMI が使用されるようになります。

例えば、組織内で毎月最新のオペレーティングシステムとアプリケーションパッチが適用された AMI の新しいバージョンを作成するとします。また、ユーザーに最新バージョンの AMI を使用してインスタンスを作成するように求めるとします。ユーザーに最新バージョンを確実に使用させるには、適切な AMI ID をポイントする Systems Manager パラメータ (例: `golden-ami` など) を作成することができます。新しいバージョンの AMI を作成するたびに、パラメータの AMI ID の値を更新して常に最新の AMI をポイントするようにします。ユーザーは毎回同じ Systems Manager パラメータを選択し続けるため、AMI の定期的な更新について知る必要はありません。AMI に Systems Manager パラメータを使用すると、ユーザーはインスタンスの起動に適切な AMI を簡単に選択できるようになります。

オートメーションコードの保守の簡素化

オートメーションコードを使用してインスタンスを作成する場合は、AMI ID の代わりに Systems Manager パラメータを指定できます。新しいバージョンの AMI を作成したら、最新の AMI を指すようにパラメータの AMI ID の値を変更できます。パラメータを参照するオートメーションコードは、新しいバージョンの AMI を作成するたびに修正する必要はありません。これにより、オートメーションのメンテナンスが簡素化されるため、デプロイのコストの削減に役立ちます。

> **Note**
>
> Systems Manager パラメータが指す AMI ID を変更しても、実行中のインスタンスは影響を受けません。

アクセス許可

インスタンス起動ウィザードで AMI ID を指す Systems Manager パラメータを使用する場合は、IAM ポリシーに `ssm:DescribeParameters` と `ssm:GetParameters` を追加する必要があります。`ssm:DescribeParameters` は、ユーザーに Systems Manager パラメータを表示および選択する許可を付与します。`ssm:GetParameters` は、ユーザーに Systems Manager パラメータの値を取得する許可を付与します。また、特定の Systems Manager パラメータへのアクセスを制限することもできます。詳細については、EC2 起動インスタンスウィザードの使用 を参照してください。

制限事項

AMI と Systems Manager パラメータはリージョンに固有です。リージョン間で同じ Systems Manager パラメータ名を使用するには、それぞれのリージョンで同じ名前の Systems Manager パラ

メータを作成します (例: `golden-ami` など)。それぞれのリージョンの Systems Manager パラメータでそのリージョンの AMI をポイントします。

Systems Manager パラメータを使用したインスタンスの起動

コンソールまたは AWS CLI を使用してインスタンスを作成できます。AMI ID を指定する代わりに、AMI ID をポイントする AWS Systems Manager パラメータを指定できます。

New console

Systems Manager パラメータを使用して Linux AMI を検索するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択します。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. コンソールダッシュボードで [インスタンスの作成] を選択します。
4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMIs] (その他の AMI を閲覧する) を選択します。
5. 検索バーの右側にある矢印ボタンを選択し、[Search by Systems Manager parameter] (Systems Manager パラメータで検索) を選択します。
6. [Systems Manager parameter (Systems Manager パラメータ)] でパラメータを選択します。対応する AMI ID が [Currently resolves to] (現在、以下に解決されています) の下に表示されます。
7. [検索] を選択します。AMI ID に一致する AMI がリストに表示されます。
8. リストから AMI を選択し、[Select (選択)] を選択します。

インスタンス起動ウィザードを使用してインスタンスを起動する方法の詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

Old console

Systems Manager パラメータを使用して Linux AMI を検索するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択します。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. コンソールダッシュボードで [インスタンスの作成] を選択します。

4. [Search by Systems Manager parameter (Systems Manager パラメータで検索)] (右上) を選択します。
5. [Systems Manager parameter (Systems Manager パラメータ)] でパラメータを選択します。対応する AMI ID が [Currently resolves to (現在対応するもの)] の横に表示されます。
6. [検索] を選択します。AMI ID に一致する AMI がリストに表示されます。
7. リストから AMI を選択し、[Select (選択)] を選択します。

インスタンス起動ウィザードを使用して AMI からインスタンスを起動する方法の詳細については、「ステップ 1: Amazon Machine Image (AMI) を選択する」を参照してください。

AMI ID の代わりに AWS Systems Manager パラメータを使用してインスタンスを作成するには (AWS CLI)

次の例では、Systems Manager パラメータの `golden-ami` を使用して `m5.xlarge` インスタンスを作成します。このパラメータは AMI ID をポイントします。

このパラメータをコマンドで指定するには、`resolve:ssm:/parameter-name` 構文を使用します。この場合、`resolve:ssm` は標準のプレフィクス、`parameter-name` は一意のパラメータ名です。パラメータ名では、大文字と小文字が区別されることに注意してください。パラメータ名のバックスラッシュは、パラメータが階層の一部である場合にのみ必要です (例: `/amis/production/golden-ami`)。パラメータが階層の一部でない場合は、バックスラッシュを省略できます。

この例では、`--count` パラメータと `--security-group` パラメータは含まれていません。`--count` はデフォルトで 1 になります。デフォルトの VPC とデフォルトのセキュリティグループがある場合は、これらが使用されます。

```
aws ec2 run-instances
    --image-id resolve:ssm:/golden-ami
    --instance-type m5.xlarge
    ...
```

特定のバージョンの AWS Systems Manager パラメータを使用してインスタンスを作成するには (AWS CLI)

Systems Manager パラメータでは、バージョンがサポートされています。パラメータの各バージョンには、一意のバージョン番号が割り当てられます。パラメータのバージョンは、`resolve:ssm:parameter-name:version` のように参照できます。`version` は一意のバー

ジョン番号です。デフォルトでは、パラメータのバージョンを指定しない場合は最新バージョンが使用されます。

次の例では、バージョン 2 のパラメータを使用します。

この例では、--count パラメータと --security-group パラメータは含まれていません。--count の場合、デフォルトは 1 です。デフォルトの VPC とデフォルトのセキュリティグループがある場合は、そのデフォルトが使用されます。

```
aws ec2 run-instances
    --image-id resolve:ssm:/golden-ami:2
    --instance-type m5.xlarge
    ...
```

AWS が提供するパブリックパラメータを使用してインスタンスを起動するには

Amazon EC2 では、AWS が提供するパブリック AMI 用の Systems Manager パブリックパラメータを使用できます。例えば、パブリックパラメータ /aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-gp2 はすべてのリージョンで利用でき、常にリージョン内の Amazon Linux 2 AMI の最新バージョンを指しています。

```
aws ec2 run-instances
    --image-id resolve:ssm:/aws/service/ami-amazon-linux-latest/amzn2-ami-hvm-x86_64-
gp2
    --instance-type m5.xlarge
    ...
```

共有 AMI

共有 AMI は、デベロッパーが作成し、他のデベロッパーが利用できるようにした AMI です。Amazon EC2 を始める最も簡単な方法は、必要なコンポーネントが含まれている共有 AMI を使用して、カスタムコンテンツを追加することです。独自の AMI を作成し、他のユーザーと共有することもできます。

共有 AMI は、ご自分の判断で使用してください。Amazon は、他の Amazon EC2 ユーザーとの間で共有される AMI の統合性や安全性を保証できません。そのため、共有 AMI を取り扱う際は、ご自分のデータセンターに外部のコードをデプロイすることを検討する場合と同じように、充分な注意を払う必要があります。検証済みのプロバイダーなど、信頼できるソースから AMI を取得することをお勧めします。

検証済みプロバイダー

Amazon EC2 コンソールでは、Amazon または検証済み Amazon パートナーが所有するパブリック AMI には [Verified provider] (検証済みプロバイダー) のマークが付されます。

また、describe-images AWS CLI コマンドを使用して、検証済みプロバイダーからのパブリック AMI を識別することもできます。Amazon または検証済みパートナーが所有するパブリックイメージには、`amazon` または `aws-marketplace` のいずれかのエイリアス所有者が存在します。CLI 出力では、これらの値が `ImageOwnerAlias` について表示されます。他のユーザーは、AMI にエイリアスを設定できません。これを利用すれば、Amazon または検証済みパートナーから AMI を簡単に見つけられます。

検証済みプロバイダーになるには、AWS Marketplace で販売者として登録する必要があります。登録が完了すると、AMI を AWS Marketplace で一覧表示できます。詳細については、AWS Marketplace 販売者ガイドの「販売者としての開始方法」および「AMI ベースの製品」を参照してください。

共有 AMI のトピック

- 共有 AMI の検索
- AMI の公開
- AMI を特定の組織または組織単位と共有する
- 特定の AWS アカウントとの AMI の共有
- お客様の AWS アカウント と AMI の共有をキャンセルする
- ブックマークの使用
- 共有 Linux AMI のガイドライン

他のトピックに関する情報をお探しの場合は

- AMI の作成の詳細については、「Instance Store-Backed Linux AMI の作成」または「Amazon EBS-Backed Linux AMI の作成」または「」を参照してください。
- AWS Marketplace でのアプリケーションの構築、配信、保守の詳細については、AWS Marketplace ドキュメントをご参照ください。

共有 AMI の検索

Amazon EC2 コンソールまたはコマンドラインを使用して、共有 AMI を検索できます。

AMI はリージョンのリソースです。共有 AMI (パブリックまたはプライベート) を検索するときには、それの共有元のリージョンから実行する必要があります。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、「AMI のコピー」を参照してください。

トピック

- 共有 AMI の検索 (コンソール)
- 共有 AMI (AWS CLI) を見つけます。
- 共有 AMI の使用

共有 AMI の検索 (コンソール)

コンソールを使用して、共有しているプライベート AMI を見つけるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMIs] を選択します。
3. 最初のフィルタで、[Private images] を選択します。お客様が共有しているすべての AMI が一覧表示されます。詳細な検索を行うには、[Search] (検索) バーを選択し、メニューに用意されたフィルターオプションを使用します。

コンソールを使用して、共有しているパブリック AMI を見つけるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMIs] を選択します。
3. 最初のフィルタで、[Public images] を選択します。詳細な検索を行うには、[Search] (検索) フィールドを選択し、メニューに用意されたフィルターオプションを使用します。

コンソールを使用して、Amazon 共有パブリック AMI を見つけるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMIs] を選択します。
3. 最初のフィルタで、[Public images] を選択します。
4. [Search] (検索) フィールドを選択し、表示されるメニューオプションから、[Owner alias] (所有者エイリアス)、[=]、[amazon] の順に選択して、Amazon のパブリックイメージのみを表示します。

コンソールを使用して検証済みプロバイダーから共有パブリック AMI を見つけるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI Catalog] (AMI カタログ) を選択します。
3. [コミュニティ AMI] を選択します。
4. [Verified provider] (検証済みプロバイダー) のラベルは、Amazon または検証済みパートナーからの AMI を示します。

共有 AMI (AWS CLI) を見つけます。

AMI を一覧表示するには、-describe-images コマンド (AWS CLI) を使用します。次の例のように、興味のある種類の AMI に絞って一覧表示できます。

例: すべてのパブリック AMI を一覧表示します。

次のコマンドは、所有しているパブリック AMI を含むすべてのパブリック AMI を一覧表示します。

```
aws ec2 describe-images --executable-users all
```

例: 明示的な起動許可を持つ AMI を一覧表示する

次のコマンドを使用すると、お客様が明示的な起動許可を持つ AMI が一覧表示されます。このリストには、お客様が所有する AMI は含まれていません。

```
aws ec2 describe-images --executable-users self
```

例: 検証済みプロバイダーが所有する AMI を一覧表示する

次のコマンドは、検証済みプロバイダーが所有する AMI を一覧表示します。検証済みプロバイダー (Amazon または検証済みパートナー) が所有するパブリック AMI には、アカウントフィールドで amazon または aws-marketplace として表示されるエイリアス所有者が存在します。これは、検証済みプロバイダーからの AMI を簡単に見つけるのに役立ちます。他のユーザーは、AMI にエイリアスを設定できません。

```
aws ec2 describe-images \
    --owners amazon aws-marketplace \
    --query 'Images[*].[ImageId]' \
    --output text
```

例: アカウントが所有する AMI を一覧表示する

次のコマンドを実行すると、指定した AWS アカウント が所有する AMI が一覧表示されます。

```
aws ec2 describe-images --owners 123456789012
```

例: フィルタを使用してスコープ AMI

表示される AMI の数を減らすには、フィルタを使用して、興味のある種類の AMI に限定して表示します。例えば、次のフィルタを使用すると、EBS-backed AMI のみが表示されます。

```
--filters "Name=root-device-type,Values=ebs"
```

共有 AMI の使用

共有 AMI を使用する前に、次の手順を実行して、インスタンスへの好ましくないアクセスを許可する認証情報が第三者により事前にインストールされていないことと、機密データを第三者に送信する可能性があるリモートロギングが事前設定されていないことを確認します。システムセキュリティ改善についての詳細は、AMI で使用される Linux ディストリビューションの文書を確認してください。

インスタンスへのアクセスを誤って失わないように、SSH セッションを 2 つ開始して、見覚えのない認証情報を削除し、その後も SSH を使用してインスタンスにログインできることが確認されるまで、2 つ目のセッションを開いておくことをお勧めします。

1. 未許可のパブリック SSH キーを特定し、無効にします。ファイル内の唯一のキーは、AMI の起動に使用したキーである必要があります。次のコマンドを使用すると、authorized_keys ファイルが見つかります。

   ```
   [ec2-user ~]$ sudo find / -name "authorized_keys" -print -exec cat {} \;
   ```

2. ルートユーザーにはパスワードベースの認証を無効にします。sshd_config ファイルを開き、次のように PermitRootLogin 行を編集します。

   ```
   PermitRootLogin without-password
   ```

   または、ルートユーザーとしてインスタンスにログインする機能を無効にできます。

   ```
   PermitRootLogin No
   ```

sshd サービスを再起動します。

3. インスタンスにログインできるユーザーが他にないか確認します。スーパーユーザー権限を持つユーザーが特に危険です。不明のアカウントがあれば、そのパスワードを削除するか、ロックします。
4. 開いていても使用していないポートと、着信接続をリスニングしている実行中のネットワークサービスをチェックします。
5. 事前設定されているリモートロギングを防ぐには、既存の設定ファイルを削除し、`rsyslog` サービスを再起動してください。例:

```
[ec2-user ~]$ sudo rm /etc/rsyslog.conf
[ec2-user ~]$ sudo service rsyslog restart
```

6. すべての cron ジョブが正当であることを確認します。

セキュリティ上のリスクとして考えられるパブリック AMI を発見した際には、AWS セキュリティチームにご連絡ください。詳細については、「AWS セキュリティセンター」を参照してください。

AMI の公開

AMI をすべての AWS アカウント と共有することで公開できます。

AMI がパブリックに共有されないようにしたい場合は、AMI のパブリックアクセスをブロックできます。これにより、AMI を公開しようとするあらゆる試みがブロックされ、不正アクセスや AMI データの悪用を防ぐのに役立ちます。パブリックアクセスのブロックを有効にしても、既に公開されている AMI には影響しないことに注意してください。AMI は引き続き公開されています。

特定のアカウントのみが AMI を使用してインスタンスを起動可能にする方法については、「特定の AWS アカウントとの AMI の共有」を参照してください。

トピック

- 考慮事項
- すべての AWS アカウントで AMI を共有 (パブリックに共有)
- AMI へのパブリックアクセスをブロックする

考慮事項

AMI を公開する前に、以下の点を検討してください。

- 所有権 — AMI を公開するには、お客様の AWS アカウント がその AMI を所有している必要があります。
- リージョン – AMI はリージョンのリソースです。共有した AMI は、共有したリージョンでのみ使用できます。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、「AMI のコピー」を参照してください。
- パブリックアクセスをブロック – AMI をパブリックに共有するには、AMI をパブリックに共有する各リージョンで AMI のパブリックアクセスのブロックを無効にする必要があります。AMI をパブリックに共有した後で、AMI のパブリックアクセスのブロックを再度有効にして、AMI がそれ以上パブリックに共有されないようにできます。
- 公開できない AMI - 次のコンポーネントが含まれる AMI は公開できません (ただし、AMI を特定の AWS アカウント と共有することはできます)。
  - 暗号化されたボリューム
  - 暗号化されたボリュームのスナップショット
  - 製品コード
- 機密データが公開されないようにする - AMI を共有するときに機密データが公開されないようにするには、「共有 Linux AMI のガイドライン」のセキュリティ考慮事項を読み、推奨アクションに従います。
- 使用 — AMI を共有する場合、ユーザーは AMI からのインスタンスのみを起動できます。AMI はそれを削除、共有、または変更することはできません。ただし、AMI を使用してインスタンスを起動した後は、起動したインスタンスから AMI を作成できます。
- 自動非推奨 – すべてのパブリック AMI を非推奨にする日はデフォルトで AMI 作成日の 2 年後になっています。非推奨にする日は 2 年より前の日付に設定できます。非推奨にする日を取り消す場合や、非推奨にする日をもっと先の日付に変える場合は、AMI を特定の AWS アカウント とのみ共有するようにして、AMI を非公開にする必要があります。
- 請求 — 他の AWS アカウント がお客様の AMI を使用してインスタンスを起動しても、お客様には請求されません。AMI を使用してインスタンスを起動するアカウントには、起動するインスタンスに対して請求されます。

すべての AWS アカウントで AMI を共有 (パブリックに共有)

AMI を公開すると、コンソールの [コミュニティ AMI] で使用できるようになります。これには、EC2 コンソールの左側のナビゲーターにある [AMI カタログ] から、またはコンソールを使用してインスタンスを起動するときにアクセスできます。AMI は、公開してから [Community AMIs] に表示されるまでに、しばらく時間がかかることもあります。

Console

[To make an AMI public]

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMIs] (AMI) を選択します。
3. リストから AMI を選択し、[Actions] (アクション) から [Edit AMI permissions] (AMI 権限の編集) を選択します。
4. [AMI の可用性] で、[パブリック] を選択します。
5. [Save changes] (変更の保存) をクリックします。

AWS CLI

各 AMI には、所有者以外でその AMI を使用してインスタンスを起動できる AWS アカウント を制御する `launchPermission` プロパティがあります。AMI の `launchPermission` プロパティを変更することで、AMI を公開したり (この場合、すべての AWS アカウント に起動許可が与えられます) 、指定した AWS アカウント とのみ AMI を共有したりすることができます。

AMI の起動許可を持っているアカウントの一覧に対してアカウント ID の追加または削除ができます。AMI を公開するには、`all` グループを指定します。パブリック起動許可と明示的起動許可の両方を指定できます。

[To make an AMI public]

1. 次のように、modify-image-attribute コマンドを使用して、指定した AMI の `launchPermission` リストに `all` グループを追加します。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Add=[{Group=all}]"
```

2. AMI の起動許可を確認するには、describe-image-attribute コマンドを使用します。

```
aws ec2 describe-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

3. (オプション) AMI をプライベートに戻すには、その起動許可から `all` グループを削除します。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Remove=[{Group=all}]"
```

AMI へのパブリックアクセスをブロックする

AMI がパブリックに共有されないようにするために、AMI のパブリックアクセスをブロックできます。この設定はアカウントレベルで有効になっていますが、AMI がパブリックに共有されないようにする AWS リージョン ごとに有効にする必要があります。

パブリックアクセスのブロックを有効にすると、AMI を公開しようとする試みは自動的にブロックされます。ただし、既にパブリック AMI がある場合は、引き続き公開されます。

AMI をパブリックに共有したい場合は、パブリックアクセスのブロックを無効にする必要があります。共有が完了したら、AMI が意図せずパブリックに共有されないように、パブリックアクセスのブロックを再度有効にするのがベストプラクティスです。

管理者ユーザーのみが AMI のパブリックアクセスのブロックを有効または無効にできるように、IAM アクセス許可を管理者ユーザーに制限できます。

トピック

- デフォルト設定
- 必要な IAM 許可
- AMI のパブリックアクセスのブロックを有効にする
- AMI のパブリックアクセスのブロックを無効にする
- AMI のパブリックアクセスのブロック状態を表示する

デフォルト設定

[AMI のパブリックアクセスをブロック] 設定は、アカウントが新規か既存か、およびパブリック AMI の有無に応じて、デフォルトで有効または無効になります。次のテーブルは、デフォルト設定の一覧です。

AWS アカウント	AMI のデフォルト設定ではパブリックアクセスをブロックします。
新しいアカウント	有効
パブリック AMI のない既存のアカウント [1]	有効
1 つ以上のパブリック AMI がある既存のアカウント	無効

[1] 2023 年 7 月 15 日以降のアカウントに 1 つ以上のパブリック AMI があった場合、その後すべての AMI を非公開にしたとしても、[AMI のパブリックアクセスをブロック] はデフォルトで無効になっています。

必要な IAM 許可

AMI のパブリックアクセスのブロックを使用するには、以下の IAM アクセス許可が必要です。

- `EnableImageBlockPublicAccess`
- `DisableImageBlockPublicAccess`
- `GetImageBlockPublicAccessState`

AMI のパブリックアクセスのブロックを有効にする

AMI がパブリックに共有されないようにするには、AMI のパブリックアクセスのブロックを有効にします。AMI がパブリックに共有されないようにする AWS リージョン ごとに、AMI のパブリックアクセスのブロックを有効にする必要があります。既にパブリック AMI がある場合は、引き続き公開されます。

Console

指定したリージョンで AMI のパブリックアクセスのブロックを有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、AMI のパブリックアクセスをブロックするリージョンを選択します。

3. ダッシュボードが表示されていない場合は、ナビゲーションペインで [EC2 ダッシュボード] を選択します。
4. [アカウントの属性] で [データ保護とセキュリティ] を選択します。
5. [AMI のパブリックアクセスをブロック] で [管理] を選択します。
6. [新しいパブリック共有をブロック] のチェックボックスを選択してから、[更新] を選択します。

> **Note**
>
> API がこの設定を行うには、最大 10 分かかる場合があります。この間、値は [新しいパブリック共有が可能] になります。API が設定を完了すると、値は自動的に [新しいパブリック共有をブロック中] に変更されます。

AWS CLI

指定したリージョンで AMI のパブリックアクセスのブロックを有効にするには

enable-image-block-public-access コマンドを使用して、AMI のパブリックアクセスのブロックを有効にするリージョンを指定します。`--image-block-public-access-state` パラメータでは、`block-new-sharing` を指定します。

```
aws ec2 enable-image-block-public-access \
    --region us-east-1 \
    --image-block-public-access-state block-new-sharing
```

正常な出力

```
{
    "ImageBlockPublicAccessState": "block-new-sharing"
}
```

> **Note**
>
> API がこの設定を行うには、最大 10 分かかる場合があります。この間に get-image-block-public-access-state コマンドを実行すると、レスポンスは `unblocked` になります。API が設定を完了すると、レスポンスは `block-new-sharing` になります。

AMI のパブリックアクセスのブロックを無効にする

アカウント内のユーザーが AMI をパブリックに共有できるようにするには、アカウントレベルでパブリックアクセスのブロックを無効にします。AMI がパブリックに共有できるようにする AWS リージョン ごとに、AMI のパブリックアクセスのブロックを無効にする必要があります。

Console

指定したリージョンで AMI のパブリックアクセスのブロックを無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、AMI のパブリックアクセスのブロックを無効にするリージョンを選択します。
3. ダッシュボードが表示されていない場合は、ナビゲーションペインで [EC2 ダッシュボード] を選択します。
4. [アカウントの属性] で [データ保護とセキュリティ] を選択します。
5. [AMI のパブリックアクセスをブロック] で [管理] を選択します。
6. [新しいパブリック共有のブロック] のチェックボックスの選択を解除してから、[更新] を選択します。
7. 確認を求められたら、「**confirm**」と入力してから、[パブリック共有の許可] を選択します。

> **Note**
>
> API がこの設定を行うには、最大 10 分かかる場合があります。この間、値は [新しいパブリック共有をブロック中] になります。API が設定を完了すると、値は自動的に [新しいパブリック共有が可能] に変更されます。

AWS CLI

指定したリージョンで AMI のパブリックアクセスのブロックを無効にするには

disable-image-block-public-access コマンドを使用して、AMI のパブリックアクセスのブロックを無効にするリージョンを指定します。

```
aws ec2 disable-image-block-public-access --region us-east-1
```

正常な出力

```
{
   "ImageBlockPublicAccessState": "unblocked"
}
```

> **Note**
>
> API がこの設定を行うには、最大 10 分かかる場合があります。この間に get-image-block-public-access-state コマンドを実行すると、レスポンスは `block-new-sharing` になります。API が設定を完了すると、レスポンスは `unblocked` になります。

AMI のパブリックアクセスのブロック状態を表示する

AMI のパブリックアクセスのブロック状態を表示すると、AMI のパブリック共有がアカウントでブロックされているかどうかを確認できます。AMI のパブリック共有がブロックされているかどうかを確認するには、それぞれの AWS リージョン で状態を確認する必要があります。

Console

指定したリージョンで AMI のパブリックアクセスのブロック状態を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、AMI のパブリックアクセスのブロック状態を表示するリージョンを選択します。
3. ダッシュボードが表示されていない場合は、ナビゲーションペインで [EC2 ダッシュボード] を選択します。
4. [アカウントの属性] で [データ保護とセキュリティ] を選択します。
5. [AMI のパブリックアクセスをブロック] で [パブリックアクセス] フィールドを確認します。値は [新しいパブリック共有をブロック中] または [新しいパブリック共有が可能] です。

AWS CLI

指定したリージョンで AMI のパブリックアクセスのブロック状態を取得するには

get-image-block-public-access-state コマンドを使用して、AMI のパブリックアクセスのブロック状態を取得するリージョンを指定します。

```
aws ec2 get-image-block-public-access-state --region us-east-1
```

期待される出力 – 値は block-new-sharing または unblocked です。

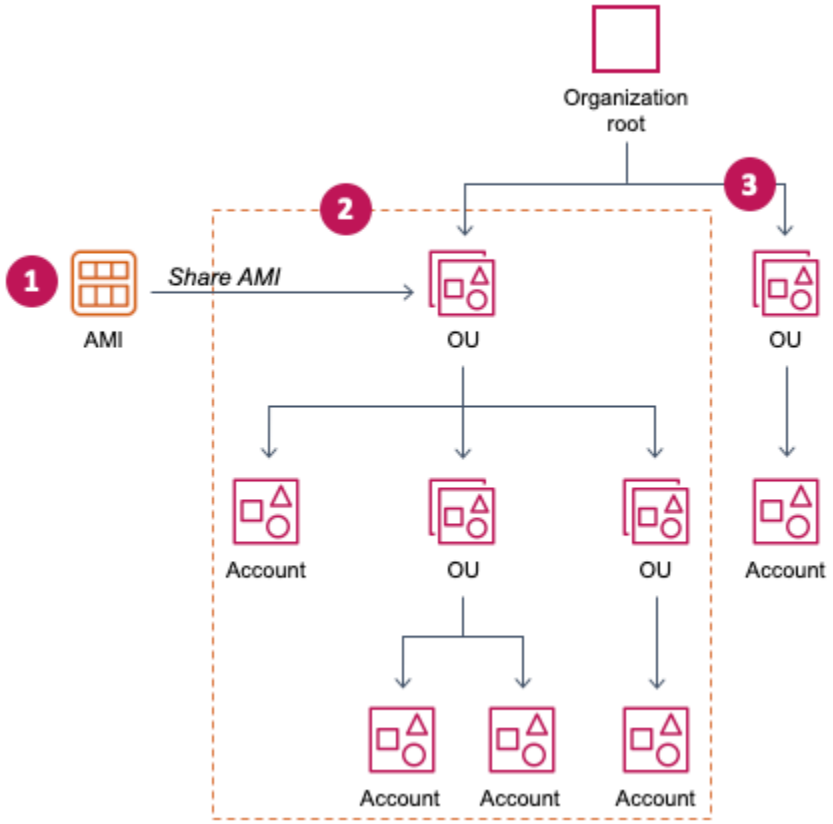
```
{
    "ImageBlockPublicAccessState": "block-new-sharing"
}
```

AMI を特定の組織または組織単位と共有する

AWS Organizations は、作成し一元管理する組織に、複数の AWS アカウント を統合するためのアカウント管理サービスです。AMI は、特定のアカウントと共有するだけでなく、組織または組織単位 (OU) と共有することもできます。

組織とは、AWS アカウント を統合して一元管理するために作成するエンティティのことです。アカウントを階層ツリーのような構造に編成して、ルートを最上部に置いて組織単位をその組織ルート下にネストすることができます。各アカウントは、ルートに直接追加するか、階層内の OU のいずれかに配置することができます。詳細については、「AWS Organizations ユーザーガイド」の「AWS 組織の用語およびコンセプト」を参照してください。

AMI を組織または OU と共有すると、すべての子アカウントが AMI にアクセスできます。例えば、次の図では、AMI は最上位レベルの OU と共有されています (1 の数字の矢印で示されます)。その最上位レベルの OU の下にネストされているすべての OU やアカウント (2 の数字の点線で示したもの) も AMI にアクセスできます。点線の外側にある組織や OU (数字の 3 で示されている) のアカウントは、AMI が共有されている OU の子供ではないため、AMI へのアクセス権はありません。



考慮事項

特定の組織または組織単位で AMI を共有する場合は、以下について検討してください。

- 所有権 — AMI を共有するには、お客様の AWS アカウント がその AMI を所有している必要があります。
- 共有制限 — AMI の所有者は、メンバーではない組織や OU を含め、任意の組織または OU と AMI を共有できます。

  リージョン内で AMI を共有できるエンティティの最大数については、「Amazon EC2 Service Quotas」をご覧ください。
- タグ - ユーザー定義タグ (AMI にアタッチするタグ) は共有できません。AMI を共有する場合、ユーザー定義タグは AMI が共有されている組織または OU のどの AWS アカウント にも使用できません。
- ARN 形式 — コマンドで組織または OU を指定する場合は、正しい ARN 形式を必ず使用してください。ID のみを指定するとエラーになります。例えば、`o-123example` や `ou-1234-5example` を指定するとエラーになります。

正しい ARN 形式:

- 組織の ARN: `arn:aws:organizations::account-id:organization/organization-id`
- OU ARN: `arn:aws:organizations::account-id:ou/organization-id/ou-id`

コードの説明は以下のとおりです。

- *account-id* は 12 桁の管理アカウント番号で、例えば、123456789012 となります。管理アカウント番号がわからない場合は、管理アカウント番号を含む ARN を取得するための組織または組織単位を記述できます。詳細については、ARN を入手する を参照してください。
- *organization-id* は組織 ID であり、例えば、o-123example となります。
- *ou-id* は組織単位 ID であり、例えば、ou-1234-5example となります。

ARN の形式の詳細については、「IAM ユーザーガイド」の「Amazon リソースネーム (ARN)」を参照してください。

- 暗号化とキー — 暗号化されていないスナップショットと暗号化されたスナップショットによってバックアップされた AMI を共有できます。
  - 暗号化されたスナップショットは、カスタマーマネージド型キーを使用して暗号化する必要があります。デフォルトの AWS マネージドキーで暗号化されたスナップショットでバックアップされた AMI を共有することはできません。
  - 暗号化されたスナップショットによってバックアップされた AMI を共有する場合、スナップショットの暗号化に使用されたカスタマーマネージドキーの使用を組織または OU に許可する必要があります。詳細については、組織と OU に KMS キーの使用を許可する をご参照ください。
- リージョン – AMI はリージョンのリソースです。共有した AMI は、共有したリージョンでのみ使用できます。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、AMI のコピー を参照してください。
- 使用 — AMI を共有する場合、ユーザーは AMI からのインスタンスのみを起動できます。AMI はそれを削除、共有、または変更することはできません。ただし、AMI を使用してインスタンスを起動した後は、起動したインスタンスから AMI を作成できます。
- 請求 — 他の AWS アカウント がお客様の AMI を使用してインスタンスを起動しても、お客様には請求されません。AMI を使用してインスタンスを起動するアカウントには、起動するインスタンスに対して請求されます。

組織と OU に KMS キーの使用を許可する

暗号化されたスナップショットによってバックアップされた AMI を共有する場合、組織または OU がスナップショットの暗号化に使用された AWS KMS keys の使用を許可する必要もあります。

`aws:PrincipalOrgID` および `aws:PrincipalOrgPaths` キーを使用して、リクエストを行っているプリンシパルの AWS Organizations パスをポリシー内のパスと比較します。そのプリンシパルは、ユーザー、IAM ロール、フェデレーションユーザー、または AWS アカウント ルートユーザーです。ポリシーでは、この条件キーによって、リクエスタが AWS Organizations で指定された組織ルートまたは OU 内のアカウントメンバーであることが保証されます。その他の条件ステートメントの例については、「IAM ユーザーガイド」の「aws:PrincipalOrgID」と「aws:PrincipalOrgPaths」を参照してください。

キーポリシーの編集の詳細については、「AWS Key Management Service 開発者ガイド」の「他のアカウントのユーザーに KMS キーの使用を許可する」を参照してください。

組織または OU に KMS キーを使用するアクセス権限を付与するには、次のステートメントをキーポリシーに追加します。

```
{
    "Sid": "Allow access for organization root",
    "Effect": "Allow",
    "Principal": "*",
    "Action": [
        "kms:Describe*",
        "kms:List*",
        "kms:Get*",
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*"
    ],
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:PrincipalOrgID": "o-123example"
        }
    }
}
```

KMS キーを複数の OU と共有するには、次の例のようなポリシーを使用します。

```
{
        "Sid": "Allow access for specific OUs and their descendants",
        "Effect": "Allow",
        "Principal": "*",
        "Action": [
            "kms:Describe*",
            "kms:List*",
            "kms:Get*",
            "kms:Encrypt",
            "kms:Decrypt",
            "kms:ReEncrypt*",
            "kms:GenerateDataKey*"
        ],
        "Resource": "*",
        "Condition": {
            "StringEquals": {
                "aws:PrincipalOrgID": "o-123example"
            },
            "ForAnyValue:StringLike": {
                "aws:PrincipalOrgPaths": [
                    "o-123example/r-ab12/ou-ab12-33333333/*",
                    "o-123example/r-ab12/ou-ab12-22222222/*"
                ]
            }
        }
}
```

AMI の共有

Amazon EC2 コンソールまたは AWS CLI を使用して AMI を組織または OU と共有できます。

AMI の共有 (コンソール)

コンソールを使用して AMI を組織または OU と共有するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI] を選択します。
3. リストで AMI を選択し、[Actions] (アクション) から [Edit AMI permissions] (AMI 権限の編集) を選択します。
4. [AMI availability] (AMI の利用状況) で、[Private] (プライベート) を選択します。

5. [Shared organizations/OUs] (共有組織/OU) の隣で、[Add organization/OU ARN] (組織/OU ARN を追加) を選択します。
6. [Organization/OU ARN] (組織/OU ARN) で、AMI を共有する組織 ARN または OU ARN を入力し、[Share AMI] (AMI の共有) を選択します。ID だけでなく、完全な ARN を指定する必要があることに注意してください。

   この AMI を複数の組織または OU と共有するには、この手順を繰り返して、必要なすべての組織または OU を追加します。

   > **Note**
   >
   > AMI を共有するために、AMI の参照先の Amazon EBS スナップショットを共有する必要はありません。共有する必要があるのは AMI 自体だけです。起動の際に、参照先の Amazon EBS スナップショットへのインスタンスアクセスが自動的に提供されます。ただし、AMI が参照するスナップショットを暗号化するために使用した KMS キーは共有する必要があります。詳細については、組織と OU に KMS キーの使用を許可する をご参照ください。

7. 完了したら、[Save changes] (変更を保存) を選択します。
8. (オプション) AMI を共有した組織または OU を表示するには、リストから AMI を選択し、[Permissions] (アクセス許可) タブをクリックし、[Shared organizations/OUs] (共有組織/OU) までスクロールします。共有されている AMI を見つけるには、「共有 AMI の検索」を参照してください。

AMI の共有 (AWS CLI)

AMI を共有するには、modify-image-attribute コマンド (AWS CLI) を使用します。

AWS CLI を使用して AMI を組織と共有するには

modify-image-attribute コマンドを使用すると、指定した組織に対し、指定した AMI の起動許可が与えられます。ID だけでなく、完全な ARN を指定する必要があることに注意してください。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission
 "Add=[{OrganizationArn=arn:aws:organizations::123456789012:organization/
o-123example}]"
```

AWS CLI を使用して AMI を OU と共有するには

[modify-image-attribute] コマンドを使用すると、指定した OU に対し、指定した AMI の起動許可が与えられます。ID だけでなく、完全な ARN を指定する必要があることに注意してください。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission
 "Add=[{OrganizationalUnitArn=arn:aws:organizations::123456789012:ou/o-123example/
ou-1234-5example}]"
```

> **Note**
>
> AMI を共有するために、AMI の参照先の Amazon EBS スナップショットを共有する必要はありません。共有する必要があるのは AMI 自体だけです。起動の際に、参照先の Amazon EBS スナップショットへのインスタンスアクセスが自動的に提供されます。ただし、AMI が参照するスナップショットを暗号化するために使用した KMS キーは共有する必要があります。詳細については、組織と OU に KMS キーの使用を許可する をご参照ください。

AMI の共有を停止する

Amazon EC2 コンソールまたは AWS CLI を使用して AMI を組織または OU と共有することを停止できます。

AMI の共有を停止する (コンソール)

コンソールを使用して AMI を組織または OU と共有することを停止するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI] を選択します。
3. リストで AMI を選択し、[Actions] (アクション) から [Edit AMI permissions] (AMI 権限の編集) を選択します。
4. [Shared organizations/OUs] (共有組織/OU) で、AMI の共有を停止する組織または OU を選択し、[Remove selected] (選択を削除) を選択します。
5. 完了したら、[Save changes] (変更を保存) を選択します。

6. (オプション) AMI の組織または OU との共有の停止を確認するには、リストから AMI を選択し、[Permissions] (アクセス許可) タブをクリックし、[Shared organizations/OUs] (共有組織/OU) までスクロールします。

AMI の共有を停止する (AWS CLI)

AMI の共有を停止するには、[modify-image-attribute] または [reset-image-attribute] コマンド (AWS CLI) を使用します。

AWS CLI を使用して AMI を組織または OU と共有することを停止するには

[modify-image-attribute] コマンドを使用すると、指定した組織から指定した AMI の起動許可が削除されます。ARN を指定する必要があることに注意してください。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission
 "Remove=[{OrganizationArn=arn:aws:organizations::123456789012:organization/
o-123example}]"
```

AWS CLI を使用してすべての組織、OU、および AWS アカウント と AMI の共有を停止するには

reset-image-attribute コマンドを使用すると、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。

```
aws ec2 reset-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

> **Note**
>
> AMI が共有されている組織または OU 内にある場合、特定のアカウントと AMI の共有を停止することはできません。アカウントの起動権限を削除して AMI の共有を停止しようとすると、Amazon EC2 は成功メッセージを返します。ただし、AMI は引き続きアカウントと共有されます。

AMI が共有されている組織と OU を表示する

Amazon EC2 コンソールまたは AWS CLI を使用して、AMI を共有した組織および OU を確認することができます。

AMI が共有されている組織と OU を表示する (コンソール)

コンソールを使用して AMI を共有した組織および OU を確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI] を選択します。
3. リストで AMI を選択し、[Permissions] (アクセス許可) タブをクリックし、[Shared organizations/OUs] (共有組織/OU) までスクロールします。

   共有されている AMI を見つけるには、「共有 AMI の検索」を参照してください。

AMI が共有されている組織と OU を表示する (AWS CLI)

どの組織や OU と AMI を共有しているかは、describe-image-attribute コマンド (AWS CLI) と `launchPermission` 属性で確認できます。

AWS CLI を使用して AMI を共有した組織および OU を確認するには

describe-image-attribute コマンドは、指定した AMI の `launchPermission` 属性を説明し、その AMI を共有している組織や OU を返します。

```
aws ec2 describe-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

レスポンスの例

```
{
    "ImageId": "ami-0abcdef1234567890",
    "LaunchPermissions": [
        {
            "OrganizationalUnitArn": "arn:aws:organizations::111122223333:ou/
o-123example/ou-1234-5example"
        }
    ]
```

```
}
```

ARN を入手する

組織と組織単位 ARN には、12 桁の管理アカウント番号が含まれています。管理アカウント番号がわからない場合は、組織と組織単位を記述して、それぞれの ARN を取得できます。以下の例では、123456789012 は管理アカウント番号です。

ARN を取得する前に、組織と組織単位を記述する権限が必要です。次のポリシーで、これらの権限が付与されます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "organizations:Describe*"
            ],
            "Resource": "*"
        }
    ]
}
```

組織の ARN を取得するには

組織の ARN だけを返すには、describe-organization コマンドを使用し、`--query` パラメータを `'Organization.Arn'` に設定します。

```
aws organizations describe-organization --query 'Organization.Arn'
```

レスポンスの例

```
"arn:aws:organizations::123456789012:organization/o-123example"
```

組織単位の ARN を取得するには

組織単位の ARN だけを返すには、describe-organizational-unit コマンドを使用し、OU ID を指定し、`--query` パラメータを `'OrganizationalUnit.Arn'` に設定します。

```
aws organizations describe-organizational-unit --organizational-unit-
id ou-1234-5example --query 'OrganizationalUnit.Arn'
```

レスポンスの例

```
"arn:aws:organizations::123456789012:ou/o-123example/ou-1234-5example"
```

特定の AWS アカウントとの AMI の共有

AMI を公開せず、特定の AWS アカウント とだけ共有することもできます。これに必要なものは AWS アカウント ID のみです。

AWS アカウント ID は、AWS アカウント を一意に識別する 12 桁の数値です (012345678901 など)。詳細については、AWS Account Management リファレンスガイドの「AWS アカウント 識別子の表示」を参照してください。

考慮事項

特定の AWS アカウント で AMI を共有する場合は、以下について検討してください。

- 所有権 — AMI を共有するには、お客様の AWS アカウント がその AMI を所有している必要があります。
- 共有制限 – リージョン内で AMI を共有できるエンティティの最大数については、「Amazon EC2 Service Quotas」をご覧ください。
- タグ - ユーザー定義タグ (AMI にアタッチするタグ) は共有できません。AMI を共有する場合、ユーザー定義タグは AMI が共有されている AWS アカウント では使用できません。
- 暗号化とキー — 暗号化されていないスナップショットと暗号化されたスナップショットによってバックアップされた AMI を共有できます。
  - 暗号化されたスナップショットは、KMS キーを使用して暗号化する必要があります。デフォルトの AWS 管理キーで暗号化されたスナップショットでバックアップされた AMI を共有することはできません。
  - 暗号化されたスナップショットによってバックアップされた AMI を共有する場合、スナップショットの暗号化に使用された KMS キーの使用を AWS アカウント に許可する必要があります。詳細については、「組織と OU に KMS キーの使用を許可する」を参照してください。暗号化にカスタマーマネージドキーを使用する際に、Auto Scaling インスタンスの起動に必要となるキーポリシーを設定するには、「Amazon EC2 Auto Scaling ユーザーガイド」の「暗号化ボリュームで使用するために必要な AWS KMS key ポリシー」を参照してください。

- リージョン – AMI はリージョンのリソースです。共有した AMI は、そのリージョンでのみ使用できます。AMI を他のリージョンで利用できるようにするには、AMI をそのリージョンにコピーし、共有します。詳細については、AMI のコピー を参照してください。
- 使用 — AMI を共有する場合、ユーザーは AMI からのインスタンスのみを起動できます。AMI はそれを削除、共有、または変更することはできません。ただし、AMI を使用してインスタンスを起動した後は、インスタンスから AMI を作成できます。
- 共有 AMI のコピー — 別のアカウントのユーザーが共有 AMI をコピーする場合は、AMI をバックアップするストレージに対する読み取り権限をそのユーザーに付与する必要があります。詳細については、「アカウント間のコピー」を参照してください。
- 請求 — 他の AWS アカウント がお客様の AMI を使用してインスタンスを起動しても、お客様には請求されません。AMI を使用してインスタンスを起動するアカウントには、起動するインスタンスに対して請求されます。

AMI の共有 (コンソール)

コンソールを使用して明示的な起動許可を与えるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMI] を選択します。
3. リストで AMI を選択し、[Actions] (アクション) から [Edit AMI permissions] (AMI 権限の編集) を選択します。
4. [Private] (プライベート) を選択します。
5. [Shared accounts] (共有アカウント) で、[Add account ID] (アカウント ID の追加) を選択します。
6. AWS アカウント ID には、AMI を共有したい AWS アカウント ID を入力し、[Share AMI] (AMI の共有) を選択します。

   この AMI を複数のアカウントで共有するには、必要なアカウント ID がすべて追加されるまでステップ 5 と 6 を繰り返します。

> **Note**
>
> AMI を共有するために、AMI の参照先の Amazon EBS スナップショットを共有する必要はありません。共有する必要があるのは AMI 自体だけです。起動の際に、参照先の Amazon EBS スナップショットへのインスタンスアクセスが自動的に提供されます。ただし、AMI が参照するスナップショットを暗号化するために使用した KMS キー は共有

する必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットの共有」を参照してください。

7. 完了したら、変更を保存 を選択します。
8. (オプション) AMI を共有した AWS アカウント ID を表示するには、リストから AMI を選択し、[Permissions] (アクセス許可) タブを開きます。共有されている AMI を見つけるには、「共有 AMI の検索」を参照してください。

AMI の共有 (AWS CLI)

AMI を共有するには、次の例のように modify-image-attribute コマンド (AWS CLI) を使用します。

明示的な起動許可を与えるには

次のコマンドを実行すると、指定した AWS アカウント に対し、指定した AMI の起動許可が与えられます。次の例では、例の AMI ID を有効な AMI ID に置き換え、12 桁の AWS アカウント ID を *account-id* に置き換えます。

```
aws ec2 modify-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Add=[{UserId=account-id}]"
```

> **Note**
>
> AMI を共有するために、AMI の参照先の Amazon EBS スナップショットを共有する必要はありません。共有する必要があるのは AMI 自体だけです。起動の際に、参照先の Amazon EBS スナップショットへのインスタンスアクセスが自動的に提供されます。ただし、AMI が参照するスナップショットを暗号化するために使用した KMS キー は共有する必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットの共有」を参照してください。

アカウントに与えた起動許可を取り消すには

次のコマンドを実行すると、指定した AWS アカウント から指定した AMI の起動許可が削除されます。次の例では、例の AMI ID を有効な AMI ID に置き換え、12 桁の AWS アカウント ID を *account-id* に置き換えます。

```
aws ec2 modify-image-attribute \
```

```
    --image-id ami-0abcdef1234567890 \
    --launch-permission "Remove=[{UserId=account-id}]"
```

すべての起動許可を取り消すには

次のコマンドを使用すると、指定した AMI からパブリック起動許可と明示的起動許可がすべて削除されます。AMI の所有者には常に起動許可が与えられるため、このコマンドの影響を受けないことにご注意ください。次の例では、サンプルの AMI ID を有効な AMI ID に置き換えます。

```
aws ec2 reset-image-attribute \
    --image-id ami-0abcdef1234567890 \
    --attribute launchPermission
```

お客様の AWS アカウント と AMI の共有をキャンセルする

AMI の起動許可にアカウントを追加することで、Amazon マシンイメージ (AMI) を特定の AWS アカウント と共有できます。AMI が AWS アカウント と共有されていて、そのアカウントとの共有が不要になった場合は、AMI の起動許可からアカウントを削除できます。この操作は、`cancel-image-launch-permission` AWS CLI コマンドを実行して行うことができます。このコマンドを実行すると、指定した AMI の起動許可から AWS アカウント が削除されます。

例えば、共有された未使用または廃止予定の AMI を含むインスタンスを起動する可能性を減らすために、AMI をアカウントで共有することをキャンセルする場合があります。AMI をアカウントと共有することをキャンセルすると、describe-images の出力や EC2 コンソールの AMI リストには表示されなくなります。

トピック

- 制限事項
- アカウントと AMI の共有をキャンセルする
- アカウントと共有されている AMI を見つける

制限事項

- お客様の AWS アカウント とだけ共有されている AMI の起動許可からアカウントを削除できます。組織または組織単位 (OU) と共有されている AMI の起動許可からアカウントを削除したり、パブリック AMI へのアクセスを削除したりすることに `cancel-image-launch-permission` を使用することはできません。

- AMI の起動許可からアカウントを完全に削除することはできません。AMI の所有者は、お客様のアカウントと再び AMI を共有できます。
- AMI はリージョンのリソースです。cancel-image-launch-permission の実行時には、AMI が配置されているリージョンを指定する必要があります。コマンドの中でリージョンを指定するか、AWS_DEFAULT_REGION 環境変数を使用します。
- AWS CLI および SDK のみが、AMI の起動許可からのアカウントの削除をサポートしています。EC2 コンソールは現在このアクションに対応していません。

アカウントと AMI の共有をキャンセルする

> **Note**
>
> お客様のアカウントと AMI の共有をキャンセルすると、元に戻すことはできません。AMI へのアクセスを回復するには、AMI 所有者がお客様のアカウントと AMI を共有する必要があります。

AWS CLI

お客様の AWS アカウント と AMI の共有をキャンセルするには

cancel-image-launch-permission コマンドを使用して、AMI の ID を指定します。

```
aws ec2 cancel-image-launch-permission \
    --image-id ami-0123456789example \
    --region us-east-1
```

正常な出力

```
{
    "Return": true
}
```

PowerShell

AWS Tools for PowerShell を使用して AMI がお客様の AWS アカウント と共有されるのをキャンセルするには

Stop-EC2ImageLaunchPermission コマンドを使用して、AMI の ID を指定します。

```
Stop-EC2ImageLaunchPermission `
    -ImageId ami-0123456789example `
    -Region us-east-1
```

正常な出力

```
True
```

アカウントと共有されている AMI を見つける

お客様の AWS アカウント と共有されている AMI を見つけるには、「共有 AMI の検索」を参照してください。

ブックマークの使用

パブリック AMI を作成した場合、あるいは AMI を別の AWS アカウント と共有した場合は、ブックマークを作成できます。ブックマークを作成すると、ユーザーは自分のアカウントですばやく AMI にアクセスし、インスタンスを起動できます。これにより AMI リファレンスを簡単に共有できるため、時間をかけず、使用する AMI を見つけることができます。

AMI はパブリックであるか、ブックマークの送信先ユーザーと共有している必要があります。

AMI のブックマークを作成するには

1. 次の情報が含まれる URL を入力します。region には AMI のリージョンを指定します。

   ```
   https://console.aws.amazon.com/ec2/v2/home?
   region=region#LaunchInstanceWizard:ami=ami_id
   ```

   例えば、この URL は、米国東部 (バージニア北部) us-east-1 リージョンの ami-0abcdef1234567890 AMI からインスタンスを起動します。

   ```
   https://console.aws.amazon.com/ec2/v2/home?region=us-
   east-1#LaunchInstanceWizard:ami=ami-0abcdef1234567890
   ```

2. AMI を使用するユーザーにリンクを配信します。
3. ブックマークを使用するには、リンクを選択するか、そのリンクをコピーしてブラウザに貼り付けます。launch wizardが開きます。AMI が既に選択されています。

共有 Linux AMI のガイドライン

攻撃対象領域を縮小し、作成する AMI の信頼性を向上させるためには、次のガイドラインを使用します。

> **Important**
>
> セキュリティのガイドラインのリストは、いずれも完全ではありません。共有 AMI を注意深く作成し、機密データが漏洩される可能性について十分考慮してください。

コンテンツ

AWS Marketplace の AMI を構築する場合は、AWS Marketplace 販売者ガイドの「AMI 構築のベストプラクティス」で、ガイドライン、ポリシー、ベストプラクティスをご参照ください。

AMI の安全な共有についての詳細は、次の記事を参照してください。

- パブリック AMI を安全に共有し使用する方法
- パブリック AMI の公開: セキュリティ強化とクリーンアップの要件

使用する前に AMI ツールを更新する

Instance Store-Backed の AMI の場合、使用する前に、AMI で Amazon EC2 AMI 作成ツールをダウンロードして、アップグレードすることをお勧めします。これにより、共有 AMI に基づく新しい AMI に最新の AMI ツールが与えられます。

Amazon Linux 2 では、aws-amitools-ec2 パッケージをインストールし、次のコマンドで PATH に AMI ツールを追加します。Amazon Linux AMI の場合、デフォルトで aws-amitools-ec2 パッケージが既にインストールされています。

```
[ec2-user ~]$ sudo yum install -y aws-amitools-ec2 && export PATH=$PATH:/opt/aws/bin
 > /etc/profile.d/aws-amitools-ec2.sh && . /etc/profile.d/aws-amitools-ec2.sh
```

次のコマンドを使用して AMI ツールをアップグレードします。

```
[ec2-user ~]$ sudo yum upgrade -y aws-amitools-ec2
```

他のディストリビューションの場合は、AMI ツールが最新版であることを確認してください。

ルートユーザーのパスワードベースのリモートログインを無効にする

パブリック AMI に固定のルートパスワードを使用することは、セキュリティの面で危険であり、すぐに知られるおそれがあります。初回ログイン後にパスワードを変更するようにユーザーに依存していますが、変更されるまでの一瞬の間にパスワードが悪用される危険性があります。

この問題を解決するには、ルートユーザーのパスワードベースのリモートログインを無効にします。

ルートユーザーのパスワードベースのリモートログインを無効にするには

1. テキストエディタで /etc/ssh/sshd_config ファイルを開き、次の行を見つけ出します:

   ```
   #PermitRootLogin yes
   ```

2. 行を次のように変更します:

   ```
   PermitRootLogin without-password
   ```

   この設定ファイルの場所は、ディストリビューションに応じて、または OpenSSH を実行していない場合は、異なることがあります。このような場合は、関連資料を参照してください。

ローカルルートアクセスを無効にする

共有 AMI を使用する際のベストプラクティスは、直接ルートログインを無効にすることです。これを行うには、実行中のインスタンスにログインし、次のコマンドを発行します。

```
[ec2-user ~]$ sudo passwd -l root
```

> **Note**
>
> このコマンドが sudo の使用に影響を及ぼすことはありません。

SSH ホストキーペアの削除

パブリック AMI から派生した AMI を共有する場合は、/etc/ssh にある既存の SSH ホストキーペアを削除します。これにより、他のユーザーがお客様の AMI を使用してインスタンスを起動したときに、SSH は、新しい固有の SSH キーペアを生成するように強制されるため、セキュリティが強化され、「中間者」攻撃の可能性を減らします。

システムにある次のすべてのキーファイルを削除します。

- ssh_host_dsa_key
- ssh_host_dsa_key.pub
- ssh_host_key
- ssh_host_key.pub
- ssh_host_rsa_key
- ssh_host_rsa_key.pub
- ssh_host_ecdsa_key
- ssh_host_ecdsa_key.pub
- ssh_host_ed25519_key
- ssh_host_ed25519_key.pub

次のコマンドを使用して、これらのファイルをすべて確実に削除できます。

```
[ec2-user ~]$ sudo shred -u /etc/ssh/*_key /etc/ssh/*_key.pub
```

> **Warning**
>
> **shred** などの安全な削除ユーティリティでは、ストレージメディアからファイルのすべてのコピーを削除できない可能性があります。ファイルの非表示のコピーは、ジャーナルファイルシステム (Amazon Linux のデフォルト ext4 を含む)、スナップショット、バック

アップ、RAID、および一時キャッシュによって作成することができます。詳細については、**shred** に関するドキュメントを参照してください。

Important

パブリック AMI から既存の SSH ホストキーペアを削除することを忘れた場合、ルーチン監査プロセスから、AMI のインスタンスを実行するすべての顧客に向けて、セキュリティ上のリスクがある可能性について通知されます。短い猶予期間の後に、AMI にプライベートのマークが付けられます。

パブリックキー認証情報のインストール

パスワードを使用したログインを防ぐように AMI を構成したら、ユーザーが別のメカニズムを使用してログインできるようにしておく必要があります。

ユーザーは、Amazon EC2 を使用すると、インスタンスの起動時にパブリックプライベートキーペア名を指定できます。`RunInstances` API 呼び出し (またはコマンドライン API ツール) で有効なキーペア名を指定すると、パブリックキー (`CreateKeyPair` または `ImportKeyPair` の呼び出し後に Amazon EC2 がサーバー上に保持するキーペアの一部) を、インスタンスメタデータに対する HTTP Query を介してインスタンスで使用できるようになります。

SSH を使用してログインするには、AMI が起動時にキー値を取得し、それを `/root/.ssh/authorized_keys` (または AMI 上のその他のユーザーアカウントの同等項目) に付加する必要があります。ユーザーはキーペアを使用して AMI のインスタンスを起動し、ルートパスワードを入力せずにログインできます。

Amazon Linux や Ubuntu を初めとする多くのディストリビューションでは、`cloud-init` パッケージを使用して、設定されたユーザーのパブリックキー認証情報を挿入します。`cloud-init` をサポートしていないディストリビューションの場合は、システムスタートアップスクリプト (例: `/etc/rc.local`) に次のコードを追加して、起動時にルートユーザーに対して指定したパブリックキーを取り込みます。

Note

次の例では、IP アドレス http://169.254.169.254/ はリンクローカルアドレスであり、インスタンスからのみ有効です。

IMDSv2

```
if [ ! -d /root/.ssh ] ; then
        mkdir -p /root/.ssh
        chmod 700 /root/.ssh
fi
# Fetch public key using HTTP
TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-
token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-
data/public-keys/0/openssh-key > /tmp/my-key
if [ $? -eq 0 ] ; then
        cat /tmp/my-key >> /root/.ssh/authorized_keys
        chmod 700 /root/.ssh/authorized_keys
        rm /tmp/my-key
fi
```

IMDSv1

```
if [ ! -d /root/.ssh ] ; then
        mkdir -p /root/.ssh
        chmod 700 /root/.ssh
fi
# Fetch public key using HTTP
curl http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key > /tmp/my-key
if [ $? -eq 0 ] ; then
        cat /tmp/my-key >> /root/.ssh/authorized_keys
        chmod 700 /root/.ssh/authorized_keys
        rm /tmp/my-key
fi
```

この設定は、あらゆるユーザーに適用できます。root ユーザーに限定する必要はありません。

> **Note**
>
> この AMI に基づいたインスタンスを再バンドルすると、起動時に使用されたキーが組み込まれます。キーへの組み込みを阻止するには、authorized_keys ファイルの を空にする (ファイルを削除する) か、このファイルを再バンドルから除外します。

sshd DNS チェックの無効化 (オプション)

sshd DNS チェックを無効にすると、sshd セキュリティが若干低下します。ただし、DNS の解決策が失敗した場合は、SSH ログインが引き続き機能します。sshd チェックを無効にしなかった場合、DNS の解決策が失敗すると、すべてのログインが阻止されます。

sshd DNS チェックを無効にするには

1. テキストエディタで /etc/ssh/sshd_config ファイルを開き、次の行を見つけ出します:

```
#UseDNS yes
```

2. 行を次のように変更します:

```
UseDNS no
```

> **Note**
>
> この設定ファイルの場所は、ディストリビューションに応じて、または OpenSSH を実行していない場合は、異なることがあります。このような場合は、関連資料を参照してください。

自身の保護

共有する AMI に、機密性のあるデータやソフトウェアは保管しないことをお勧めします。共有 AMI を起動するユーザーは、それを再バンドルしたり、自分のものとして登録したりできる可能性があります。以下のガイドラインに従って、見落としやすいセキュリティ上のリスクを回避してください:

- --exclude *directory* で ec2-bundle-vol オプションを使用して、バンドル操作に含めたくない機密情報が入っているディレクトリおよびサブディレクトリをスキップすることをお勧めします。特に、イメージをバンドルするときに、すべてのユーザー所有の SSH パブリックキー/プライベートキーペアおよび SSH authorized_keys ファイルを除外します。Amazon パブリック AMI で、これらのファイルは、ルートユーザーの場合は /root/.ssh、通常のユーザーの場合は /home/*user_name*/.ssh/ に配置されています。詳細については、「ec2-bundle-vol」を参照してください。

- バンドルの前に必ずシェル履歴を削除してください。同じ AMI で複数のバンドルのアップロードを試行すると、シェル履歴にアクセスキーが含まれます。次の例は、インスタンス内からのバンドルの前に実行される最後のコマンドとなる必要があります。

```
[ec2-user ~]$ shred -u ~/.*history
```

> **Warning**
>
> 上記の警告で示した **shred** の制限は、ここにも適用されます。
> bash は、終了時に現在のセッション履歴をディスクに書き込むことに注意してください。~/.bash_history を削除後にインスタンスをログアウトし、再度ログインすると、~/.bash_history が再作成され、前のセッション中に実行されたすべてのコマンドが含まれています。
> bash 以外の他のプログラムもディスクに履歴を書き込むため、注意して不要な dot ファイルと dot ディレクトリを削除または除外します。

- 実行中のインスタンスをバンドルするには、プライベートキーと X.509 証明書が必要です。これらの証明書およびその他の証明書を、バンドルされていない場所 (インスタンスストアなど) に書き込みます。

有料 AMI

有料 AMI は、デベロッパーから購入できる AMI です。

Amazon EC2 は AWS Marketplace と統合されており、デベロッパーは自身が開発した AMI を他の Amazon EC2 ユーザーに有償で提供したり、インスタンスにサポートを提供したりできます。

AWS Marketplace は、EC2 インスタンスの起動に使用できる AMI など、AWS で実行されるソフトウェアを購入できるオンラインストアです。AWS Marketplace AMI はデベロッパーツールなどカテゴリ別に整理されており、ユーザーは要件に適合する製品を見つけることができます。AWS Marketplace の詳細については、AWS Marketplace のウェブサイトを参照してください。

有料 AMI からのインスタンスの起動は、他の AMI からのインスタンスの起動と同じです。追加パラメータは必要ありません。インスタンスは、AMI の所有者が設定した料金と、Amazon EC2 で m1.small インスタンスタイプを実行する場合の 1 時間あたりの料金など、関連ウェブサービスの標準使用料に基づいて課金されます。税金が加算されることもあります。有料 AMI の所有者は、特定のインスタンスがその有料 AMI から起動されたかどうかを確認できます。

> **Important**
>
> Amazon DevPay は新しい販売者または製品の受付を停止しました。現在は AWS Marketplace が、ソフトウェアとサービスを AWS で販売しており、唯一の統一された e コマースプラットフォームとなっています。AWS Marketplace でソフトウェアをデプロイし販売する方法については、AWS Marketplace での販売を参照してください。AWS Marketplace は Amazon EBS-Backed の AMI をサポートしています。

コンテンツ

AMI の販売

AWS Marketplace を使用して AMI を販売できます。AWS Marketplace では体系的に買い物をすることができます。また AWS Marketplace は、Amazon EBS-Backed AMI、リザーブドインスタンス、スポットインスタンスなどの AWS 機能もサポートしています。

AWS Marketplace で AMI を販売する方法の詳細については、AWS Marketplace での販売を参照してください。

有料 AMI の検索

購入できる AMI を検索する方法はいくつかあります。例えば、AWS Marketplace、Amazon EC2 コンソール、コマンドラインを使用できます。あるいは、デベロッパーが有料 AMI に関する情報をお客様にお知らせすることがあります。

コンソールを使用した有料 AMI の検索

コンソールを使用して有料 AMI を見つけるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMIs] を選択します。
3. 最初のフィルタで、[パブリックイメージ] を選択します。
4. 検索バーで、[Owner alias] (所有者エイリアス)、[=]、[aws-marketplace] の順に選択します。
5. 製品コードがわかっている場合は、[Product code] (製品コード)、[=] の順に選択し、製品コードを入力します。

AWS Marketplace を使用して有料 AMI を検索する

AWS Marketplace を使用して有料 AMI を見つけるには

1. を開く。。AWS Marketplace
2. 検索フィールドにオペレーティングシステムの名前を入力し、検索ボタン (虫眼鏡) を選択します。
3. 検索結果をさらに絞るには、カテゴリまたはフィルタを利用します。
4. 各製品には、製品タイプ (AMI または Software as a Service) のラベルが付けられています。

AWS CLI を使用した有料 AMI の検索

次の describe-images コマンド (AWS CLI) を使用して、有料 AMI を見つけることができます。

```
aws ec2 describe-images
    --owners aws-marketplace
```

このコマンドは、有料 AMI の製品コードなど、各 AMI を説明するさまざまな詳細を返します。describe-images からの出力には、次のような製品コードのエントリがあります:

```
"ProductCodes": [
    {
        "ProductCodeId": "product_code",
        "ProductCodeType": "marketplace"
    }
```

```
],
```

製品コードがわかっている場合は、結果を製品コードでフィルタリングすることができます。次の例は、指定された製品コードを持つ最新の AMI を返します。

```
aws ec2 describe-images
    --owners aws-marketplace \
    --filters "Name=product-code,Values=product_code" \
    --query "sort_by(Images, &CreationDate)[-1].[ImageId]"
```

有料 AMI の購入

AMI を使用してインスタンスを起動するには、有料 AMI にサインアップする (購入する) 必要があります。

通常、有料 AMI の販売者は、価格や購入サイトへのリンクなど、AMI に関する情報を提供します。リンクをクリックすると、最初に AWS へのログインが求められます。ログイン後、AMI を購入できます。

コンソールを使用した有料 AMI の購入

Amazon EC2 Launch Wizard を使用して有料 AMI を購入できます。詳細については、AWS Marketplace インスタンスの起動 を参照してください。

AWS Marketplace を使用した製品の登録

AWS Marketplace を使用するには AWS アカウントが必要です。AWS Marketplace 製品からインスタンスを起動するには、Amazon EC2 サービスを利用するためのサインアップと、インスタンスを起動する製品のサブスクリプションが必要です。AWS Marketplace の製品を受信登録するには、2 つの方法があります。

- AWS Marketplace ウェブサイト: 1-Click デプロイメント機能で、事前に設定したソフトウェアをすばやく起動できます。
- Amazon EC2 Launch Wizard : AMI を検索し、ウィザードからインスタンスを直接起動できます。詳細については、AWS Marketplace インスタンスの起動 を参照してください。

インスタンスの製品コードの取得

インスタンスの AWS Marketplace 製品コードは、インスタンスメタデータを使用して取得できます。メタデータの取得については、インスタンスメタデータとユーザーデータ を参照してください。

製品コードを取得するには、次のコマンドを使用します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/product-codes
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/product-codes
```

インスタンスに製品コードが含まれる場合、Amazon EC2 はそれを返します。

有料サポートの使用

Amazon EC2 は、デベロッパーがソフトウェア (またはそれに由来する AMI) のサポートを提供できるように手配します。デベロッパーは、お客様がサインアップして使用できるサポート製品を提供することができます。サポート製品にサインアップすると、デベロッパーはお客様に製品コードを渡します。お客様はそのコードをご自分の AMI に関連付ける必要があります。これにより、デベロッパーは、ユーザーのインスタンスがサポート対象であることを確認できます。また、お客様が製品からインスタンスを実行すると、デベロッパーが定めた製品の利用規約にしたがい、お客様に課金されます。

> **Important**
>
> リザーブドインスタンス とともにサポート製品を使用することはできません。お客様は常に、サポート製品の販売者が指定した価格を支払います。

製品コードと自分の AMI を関連付けるには、次のコマンドの 1 つを使用します。ami_id は AMI の ID で、product_code は製品コードです。

- modify-image-attribute (AWS CLI)

```
aws ec2 modify-image-attribute --image-id ami_id --product-codes "product_code"
```

- Edit-EC2ImageAttribute (AWS Tools for Windows PowerShell)

```
PS C:\> Edit-EC2ImageAttribute -ImageId ami_id -ProductCode product_code
```

一度設定した製品コード属性を変更したり削除したりすることはできません。

有料およびサポートされる AMI の請求書

有料またはサポートされた AMI の使用料金がお客様のクレジットカードに請求され、その金額を記載した E メールが毎月末に届きます。これは通常の Amazon EC2 使用料金とは別に請求されます。詳細については、AWS Marketplace 購入者ガイドの「製品の支払い」をご参照ください。

AWS Marketplace サブスクリプションを管理する

AWS Marketplace ウェブサイトでは、サブスクリプションの詳細の確認、使用に関するベンダー指示の表示、サブスクリプションの管理などを行うことができます。

サブスクリプションの詳細を確認するには

1. AWS Marketplace にログインします。
2. [Your Marketplace Account] を選択します。
3. [Manage your software subscriptions] を選択します。
4. 現在のすべてのサブスクリプションが表示されます。実行中のインスタンスに接続するためのユーザー名など、製品の使用に関する特定の取扱説明を表示するには、[Usage Instructions] を選択します。

AWS Marketplace のサブスクリプションをキャンセルするには

1. サブスクリプションによって実行されていたすべてのインスタンスを終了したことを確認します。
   a. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
   b. ナビゲーションペインで、[インスタンス] を選択します。

c. インスタンスを選択し、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) の順に選択します。

d. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択します。

2. AWS Marketplace にログインし、[Your Marketplace Account] (自分の Marketplace アカウント)、[Manage your software subscriptions] (ソフトウェアサブスクリプションの管理) の順に選択します。

3. [Cancel subscription] を選択します。取り消しの確認を求めるプロンプトが表示されます。

> **Note**
>
> 受信登録をキャンセルすると、その AMI からインスタンスを起動できなくなります。その AMI を再度使用するには、AWS Marketplace ウェブサイトまたは Amazon EC2 コンソールの起動ウィザードを使用して、その AMI を再度サブスクライブする必要があります。

AMI ライフサイクル

独自の AMI を作成、コピー、およびバックアップしたり、非推奨または登録解除の準備ができるまで維持したりすることができます。

コンテンツ

AMI を作成する

Amazon EBS-back Linux AMI と インスタンスストアでサポートされている AMI を作成できます。

トピック

- Amazon EBS-backed Linux AMI を作成する
- instance store-backed Linux AMI を作成する

Windows AMI を作成する方法については、「カスタム Windows AMI を作成する」を参照してください。

Amazon EBS-backed Linux AMI を作成する

Amazon EBS-Backed Linux AMI を作成するには、既存の Amazon EBS-Backed Linux AMI から起動したインスタンスから始めます。例えば、AWS Marketplace から取得した AMI、AWS Server Migration Service か VM Import/Export を使用して作成した AMI、またはユーザーがアクセス可能なその他の任意の AMI です。ニーズに合わせてインスタンスをカスタマイズしたら、新しい AMI を作成し、登録します。新しい AMI を使用して、カスタマイズした新しいインスタンスを起動できます。

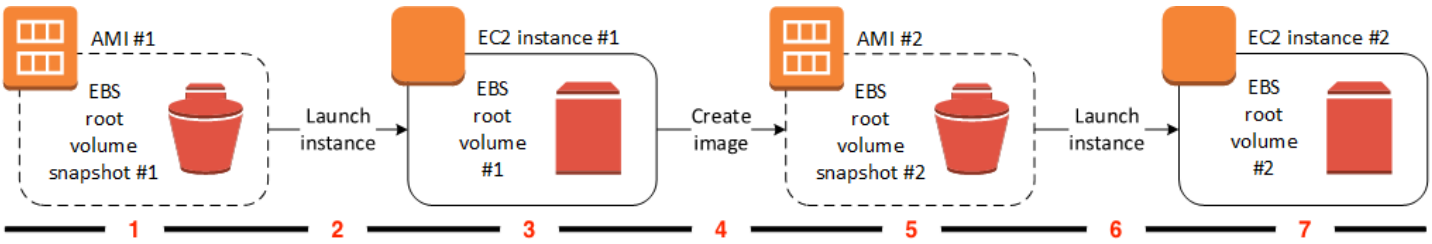
以下に説明された手順は、暗号化された Amazon Elastic Block Store (Amazon EBS) ボリューム (ルートボリュームを含む) でバックアップされた Amazon EC2 インスタンスにも、暗号化されていないボリューム同様に機能します。

AMI の作成プロセスは、Instance Store-Backed AMIs の場合とは異なります。Amazon EBS-backed インスタンスと instance store-backed インスタンスの違いの詳細と、インスタンスのルートデバイスタイプを判別する方法については、「ルートデバイスのストレージ」を参照してください。instance store-backed Linux AMI の作成に関する詳細については、「instance store-backed Linux AMI を作成する」を参照してください。

Amazon EBS-backed Windows AMI の作成の詳細については、「Windows インスタンスの Amazon EC2 ユーザーガイド」の「Amazon EBS-backed Windows AMI の作成」を参照してください。

Amazon EBS-Backed AMIs の作成の概要

次の図は、実行中の EC2 インスタンスから Amazon EBS-backed AMI を作成するプロセスをまとめたものです。既存の AMI から開始して、インスタンスを起動してカスタマイズし、そこから新しい AMI を作成し、最後に新しい AMI のインスタンスを起動します。図表内の数字は、次の説明の数値と一致します。



1 — AMI #1: 既存の AMI から始める

作成する AMI に似た既存の AMI を検索します。例えば、AWS Marketplace から取得した AMI、AWS Server Migration Service か VM Import/Export を使用して作成した AMI、またはユーザーがアクセス可能なその他の任意の AMI です。この AMI を必要に応じてカスタマイズします。

図表内の EBS ルートボリュームスナップショット #1 は、AMI が Amazon EBS-backed AMI であり、ルートボリュームに関する情報がこのスナップショットに格納されていることを示します。

2 — 既存の AMI からインスタンスを起動する

AMI を設定する方法は、新しい AMI のベースとなる AMI からインスタンスを起動し、インスタンスをカスタマイズすることです (図表内の 3)。次に、カスタマイズを含む新しい AMI を作成します (図表内の 4)。

3 — EC2 インスタンス #1: インスタンスをカスタマイズする

インスタンスに接続し、必要に応じてカスタマイズします。新しい AMI には、これらのカスタマイズが含まれます。

インスタンスで次のアクションを実行して、インスタンスをカスタマイズできます。

- ソフトウェアやアプリケーションをインストールする
- データをコピーする
- 起動時間を短縮するために一時ファイルの消去、ハードディスクのデフラグ、占有領域の開放処理を行う。
- 追加の EBS ボリュームをアタッチする

4 – イメージを作成する

インスタンスから AMI を作成する際に、Amazon EC2 がインスタンスをシャットダウンしてから AMI を作成するのは、インスタンス上のすべての動作を停止し、作成プロセス中に一貫した状態が保たれるようにするためです。インスタンスが一貫した状態にあり、適切に AMI を作成できる場合、インスタンスの電源を落として再起動しないように、Amazon EC2 に指定できます。XFS

などの一部のファイルシステムでは、アクティビティのフリーズおよびフリーズ解除が可能なため、インスタンスを再起動しなくてもイメージを安全に作成できます。

AMI 作成プロセスの間、Amazon EC2 はインスタンスのルートボリュームとインスタンスにアタッチされているその他の EBS ボリュームのスナップショットを作成します。AMI の登録を解除してスナップショットを削除するまで、スナップショットは課金の対象となります。インスタンスにアタッチされるいずれかのボリュームが暗号化されている場合、新しい AMI は、Amazon EBS 暗号化をサポートするインスタンスでのみ正常に起動します。

ボリュームのサイズによっては、AMI 作成プロセスの完了に数分かかる場合があります (最長で 24 時間かかることもあります)。AMI を作成する前に、ボリュームのスナップショットを作成しておくと、効率が向上する可能性があります。この方法では、AMI を作成する際に作成する必要があるのは小さい差分スナップショットのみになるため、プロセスがよりすばやく完了します (スナップショット作成の合計時間は同じです)。

5 — AMI #2: 新しい AMI

プロセスが完了すると、新しい AMI と、インスタンスのルートボリュームから作成されたスナップショット (スナップショット #2) が得られます。ルートデバイスボリュームに加えて、インスタンスストアボリュームまたは EBS ボリュームをインスタンスに追加した場合、新しい AMI のブロックデバイスマッピングにこれらのボリュームの情報が含まれます。

Amazon EC2 では AMI は自動的に登録されます。

6 – 新しい AMI からインスタンスを起動する

新しい AMI を使用してインスタンスを起動できます。

7 — EC2 インスタンス #2: 新しいインスタンス

ユーザーが新しい AMI を使用してインスタンスを起動すると、Amazon EC2 はスナップショットを使用して、そのインスタンスのルートボリュームのために新しい EBS ボリュームを作成します。インスタンスのカスタマイズ時に、インスタンスストアボリュームまたは EBS ボリュームを追加した場合、新しい AMI のブロックデバイスマッピングにこれらのボリュームの情報が含まれ、新しい AMI から起動するインスタンスのブロックデバイスマッピングに自動的にこれらのボリュームの情報が含まれます。新しいインスタンスのブロックデバイスマッピングに指定されているインスタンスストアボリュームは新しく、AMI の作成に使用したインスタンスのインスタンスストアボリュームからのデータは含まれていません。EBS ボリュームのデータは永続的です。詳細については、ブロックデバイスマッピング を参照してください。

EBS-backed AMI から新しいインスタンスを作成する場合、本稼働環境に移す前にそのルートボリュームと追加 EBS ストレージの両方を初期化する必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの初期化」を参照してください。

インスタンスからの Linux AMI の作成

AWS Management Console またはコマンドラインを使用して、AMI を作成できます。

Console

AMI を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. AIM を作成されるインスタンスを選択し、[Actions] (アクション)、[Image and templates] (イメージとテンプレート) の順に選択し、[Create image] (イメージの作成) をクリックします。

   > **Tip**
   > このオプションが無効になっている場合、そのインスタンスは Amazon EBS-Backed インスタンスではありません。

4. [Create image] (イメージの作成) ページで、次の情報を指定します。

   a. [Image name] (イメージ名) に、最大 127 文字までイメージの一意の名前を入力します。

   b. [Image description] (イメージの説明) に、最大 255 文字までイメージの説明を入力します (オプション)。

   c. [No reboot] (再起動しない) の場合は、[Enable] (有効化) チェックボックスをオフ (デフォルト) のままにするか、オンにします。

      - [再起動しない] の [有効化] チェックボックスがオフの場合、Amazon EC2 が新しい AMI を作成するときに、データの保存中にアタッチされたボリュームのスナップショットを取得できるようにインスタンスを再起動して、一貫性のある状態を維持します。
      - [再起動しない] の [有効化] チェックボックスがオンの場合、Amazon EC2 が新しい AMI を作成しても、インスタンスはシャットダウンおよび再起動されません。

> **Warning**
>
> [No reboot] (再起動しない) を選択した場合、作成されたイメージの、ファイルシステムの整合性は保証されません。

d. [Instance volumes] (インスタンスボリューム) - 次のとおり、ルートボリュームを変更し、Amazon EBS およびインスタンスストアボリュームを追加できます。

   i. ルートボリュームは、最初の行で定義されます。

      - ルートボリュームのサイズを変更するには、[サイズ] に必要な値を入力します。
      - [終了時に削除] を選択した場合、この AMI から作成されたインスタンスを終了すると、EBS ボリュームが削除されます。[終了時に削除] をオフにした場合は、インスタンスを終了しても、EBS ボリュームは削除されません。詳細については、インスタンスの終了時にデータを保持する を参照してください。

   ii. EBS ボリュームを追加するには、[Add Volume] を選択します (これにより、新しい行が追加されます)。[ストレージタイプ] で [EBS] を選択し、行のフィールドに入力します。作成した AMI からインスタンスを起動すると、追加したボリュームは自動的にそのインスタンスにアタッチされます。空のボリュームはフォーマットしてマウントする必要があります。スナップショットベースのボリュームはマウントする必要があります。

   iii. インスタンスストアボリュームを追加するには、「AMI へのインスタンスストアボリュームの追加を参照してください。その後新しい AMI からインスタンスを起動すると、追加されたボリュームは自動的に初期化されてマウントされます。これらのボリュームには、AMI の作成に使用された実行中のインスタンスのインスタンスストアボリュームのデータは含まれません。

e. タグ - AMI とスナップショットに同じタグを付けることも、異なるタグを付けることもできます。

   - AMI とスナップショットに同じタグを付けるには、[イメージとスナップショットに一緒にタグを付ける] を選択します。AMI と作成されるすべてのスナップショットには、同じタグが適用されます。
   - AMI とスナップショットに異なるタグを付けるには、[イメージとスナップショットに個別にタグを付ける] を選択します。AMI と作成されるスナップショットには、異な

るタグが適用されます。ただし、すべてのスナップショットに同じタグが付けられます。各スナップショットに異なるタグを付けることはできません。

(オプション) タグを追加するには、[Add tag] を選択し、そのタグのキーと値を入力します。各タグについて、これを繰り返します。

f. AMI を作成する準備ができたら、[Create image] (イメージの作成) を選択します。

5. 作成中に AMI のステータスを表示するには

a. ナビゲーションペインで [AMI] を選択します。

b. フィルタを [Owned by me] (自分が所有) に設定し、リストから AMI を探します。

最初は、ステータスは `pending` ですが、数分後 `available` に変わります。

6. (オプション) 新しい AMI に作成されたスナップショットを表示するには:

a. 前のステップで特定した AMI の ID をメモします。

b. ナビゲーションペインで、[Snapshots] を選択します。

c. フィルターを [Owned by me] (自分が所有) に設定し、新しい AMI ID のスナップショットを [Description] (説明) 列で検索します。

ユーザーがこの AMI からインスタンスを起動すると、Amazon EC2 はこのスナップショットを使用して、ルートデバイスボリュームを作成します。

AWS CLI

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- create-image (AWS CLI)
- New-EC2Image (AWS Tools for Windows PowerShell)

スナップショットからの Linux AMI の作成

インスタンスのルートデバイスボリュームのスナップショットがある場合、AWS Management Console またはコマンドラインを使用して、そのスナップショットから AMI を作成できます。

Console

スナップショットから AMI を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Snapshots] を選択します。
3. AMI の作成元になるスナップショットを選択してから、[Actions] (アクション)、[Create image from snapshot] (スナップショットからイメージを作成) の順に選択します。
4. [スナップショットからイメージを作成] ページで、次の情報を指定します。
   a. [Image name] (イメージ名) に、イメージのわかりやすい名前を入力します。
   b. [Description] (説明) に、イメージの簡単な説明を入力します。
   c. [Architecture] (アーキテクチャ) で、イメージアーキテクチャを選択します。32 ビットの場合は [i386]、64 ビットの場合は [x86_64]、64 ビット ARM の場合は [arm64]、64 ビット macOS の場合は [x86_64] をそれぞれ選択します。
   d. [Root device name] (ルートデバイス名) に、ルートデバイスボリュームに使用するデバイス名を入力します。詳細については、Linux インスタンスでのデバイス名 を参照してください。
   e. [Virtualization type] (仮想化タイプ) で、この AMI から起動されたインスタンスで使用する仮想化タイプを選択します。詳細については、Linux AMI 仮想化タイプ を参照してください。
   f. (準仮想化の場合のみ) [Kernel ID] (カーネル ID) で、イメージのオペレーティングシステムのカーネルを選択します。インスタンスのルートデバイスボリュームのスナップショットを使用している場合、元のインスタンスと同じカーネル ID を選択します。不明な場合は、デフォルトのカーネルを使用してください。
   g. (準仮想仮想化の場合のみ) [RAM disk ID] (RAM ディスク ID) で、イメージの RAM ディスクを選択します。カーネルを選択した場合は、サポートするドライバーとともに特定の RAM ディスクを選択しなければならない可能性があります。
   h. [ブートモード] では、イメージのブートモードを選択するか [デフォルトを使用] を選択し、この AMI でインスタンスを起動したときにインスタンスタイプでサポートされているブートモードで起動するようにします。詳細については、「AMI のブートモードを設定する」を参照してください。
   i. (オプション) [ブロックデバイスマッピング] で、ルートボリュームをカスタマイズし、データボリュームを追加します。

ボリュームごとに、サイズ、タイプ、パフォーマンス特性、終了時の削除の動作、および暗号化ステータスを指定できます。ルートボリュームについては、サイズをスナップショットのサイズより小さくすることはできません。ボリュームタイプには、汎用 SSD gp3 がデフォルトで選択されています。

j. (オプション) [タグ] で、新しい AMI に 1 つ以上のタグを追加できます。(オプション) タグを追加するには、[Add tag] を選択し、そのタグのキーと値を入力します。各タグについて、これを繰り返します。

k. AMI を作成する準備ができたら、[Create image] (イメージの作成) を選択します。

AWS CLI

コマンドラインを使用してスナップショットから AMI を作成するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- register-image (AWS CLI)
- Register-EC2Image (AWS Tools for Windows PowerShell)

作成した AMI からのインスタンスの起動

インスタンスまたはスナップショットから作成した AMI からインスタンスを起動できます。

新しい AMI からインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Images (イメージ)] で、[AMIs (AMI)] を選択します。
3. フィルタを [Owned by me (自分の所有)] に設定し、AMI を選択します。
4. [AMI からインスタンスを起動する] を選択します。
5. デフォルト値をそのまま使用するか、インスタンス起動ウィザードでカスタム値を指定します。詳細については、新しいインスタンス起動ウィザードを使用してインスタンスを起動する を参照してください。

instance store-backed Linux AMI を作成する

インスタンスの起動時に指定する AMI によってルートデバイスボリュームのタイプが決まります。

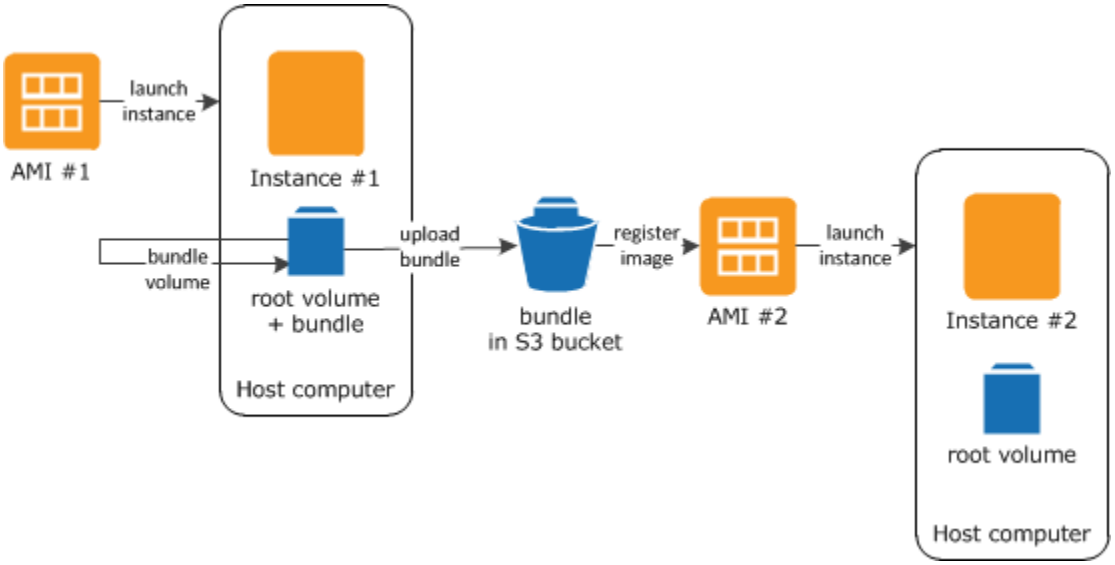
Instance Store-Backed Linux AMI を作成するには、既存の Instance Store-Backed Linux AMI から起動したインスタンスから始めます。ニーズに合わせてインスタンスをカスタマイズしたら、ボリュームをバンドルし、新しい AMI を登録します。新しい AMI を使用して、カスタマイズした新しいインスタンスを起動できます。

> **Important**
>
> インスタンスストアボリュームをルートデバイスとしてサポートするインスタンスタイプは C1、C3、D2、G2、I2、M1、M2、M3、R3、X1 のみです。

AMI の作成プロセスは、Amazon EBS-backed AMI の場合とは異なります。Amazon EBS-Backed インスタンスと Instance store-Backed インスタンスの違いの詳細と、インスタンスのルートデバイスタイプを判別する方法については、「ルートデバイスのストレージ」を参照してください。Amazon EBS-backed Linux AMI を作成する必要がある場合は、「Amazon EBS-backed Linux AMI を作成する」を参照してください。

Instance Store-Backed AMI の作成プロセスの概要

次の図は、Instance Store-Backed インスタンスから AMI を作成するプロセスをまとめたものです。



最初に、作成する AMI と同様の AMI からインスタンスを起動します。インスタンスに接続し、それをカスタマイズできます。インスタンスのカスタマイズが終わったら、それをバンドルできます。バンドルプロセスが完了するには数分間かかります。プロセスが完了すると、イメージマニフェスト (`image.manifest.xml`) とルートボリューム用のテンプレートを含むファイル (`image.part.xx`)

で構成されるバンドルが作成されます。次に、バンドルを Amazon S3 バケットにアップロードし、AMI を登録します。

> **Note**
>
> instance store-backed Linux AMI の S3 バケットにオブジェクトをアップロードするには、バケットで ACL を有効にする必要があります。有効にしない場合、Amazon EC2 はアップロードするオブジェクトに ACL を設定できません。宛先のバケットが S3 オブジェクトの所有権のバケット所有者強制設定を使用している場合、ACL が無効になるため、この方法は使えません。詳細については、「S3 オブジェクトの所有権を使用したアップロードされたオブジェクトの所有権の管理」を参照してください。

お客様が新しい AMI を使用してインスタンスを起動すると、Amazon はユーザーが Amazon S3 にアップロードしたバンドルを使用してインスタンスのルートボリュームを作成します。Amazon S3 のバンドルで使用されるストレージ領域については、お客様がその領域を削除するまでアカウントに料金が発生します。詳細については、AMI の登録の解除 を参照してください。

ルートデバイスボリュームに加えて、インスタンスストアボリュームをインスタンスに追加した場合、新しい AMI のブロックデバイスマッピングにこれらのボリュームの情報が含まれ、新しい AMI から起動するインスタンスのブロックデバイスマッピングに自動的にこれらのボリュームの情報が含まれます。詳細については、ブロックデバイスマッピング を参照してください。

前提条件

AMI を作成するには、最初に次のタスクを完了する必要があります。

- AMI ツールをインストールします。詳細については、AMI ツールのセットアップ を参照してください。
- AWS CLI をインストールします。詳細については、「AWS Command Line Interface のセットアップ」を参照してください。
- バンドルに S3 バケットがあり、バケットに ACL が有効になっていることを確認します。ACL の設定の詳細については、「ACL の設定」を参照してください。
  - AWS Management Console を使用して S3 バケットを作成するには、https://console.aws.amazon.com/s3/ で Amazon S3 コンソールを開き、[Create Bucket] を選択します。
  - AWS CLI で S3 バケットを作成するには、「mb」コマンドを使用できます。インストールしている AMI ツールのバージョンが 1.5.18 以降の場合は、`ec2-upload-bundle` コマンドを使用

して S3 バケットを作成することもできます。詳細については、「ec2-upload-bundle」を参照してください。

- AWS アカウント ID があることを確認します。詳細については、「AWS アカウント管理リファレンスガイド」の「View AWS アカウント identifiers」を参照してください。
- AWS CLI を使用するのに必要な認証情報があることを確認します。詳細については、AWS Account Management リファレンスガイドの「AWS アカウントのベストプラクティス」を参照してください。
- X.509 証明書および対応するプライベートキーがあることを確認します。
  - X.509 証明書を作成する必要がある場合は、「署名証明書の管理」を参照してください。X.509 証明書とプライベートキーは、AMI の暗号化/復号に使用されます。
  - [中国 (北京)] `$EC2_AMITOOL_HOME/etc/ec2/amitools/cert-ec2-cn-north-1.pem` 証明書を使用します。
  - [AWS GovCloud (米国 – 西部)] `$EC2_AMITOOL_HOME/etc/ec2/amitools/cert-ec2-gov.pem` 証明書を使用します。
- インスタンスに接続し、カスタマイズします。例えば、ソフトウェアとアプリケーションをインストールしたり、データをコピーしたり、一時ファイルを削除したり、Linux 設定を変更したりできます。

タスク

- AMI ツールのセットアップ
- Instance Store-Backed Amazon Linux インスタンスからの AMI の作成
- Instance Store-Backed Ubuntu インスタンスからの AMI の作成
- instance store-backed AMI を Amazon EBS-backed AMI への変換

AMI ツールのセットアップ

AMI ツールを使用して、Instance Store-Backed Linux AMIs を作成および管理できます。ツールを使用するには、Linux インスタンスにインストールする必要があります。AMI ツールは RPM として使用できるとともに、RPM をサポートしていない Linux ディストリビューションでは .zip ファイルとして使用できます。

RPM を使用して AMI ツールを設定するには

1. yum などの Linux ディストリビューション用のパッケージマネージャを使用して Ruby をインストールします。次に例を示します。

```
[ec2-user ~]$ sudo yum install -y ruby
```

2. wget や curl などのツールを使用して RPM ファイルをダウンロードします。次に例を示します。

```
[ec2-user ~]$ wget https://s3.amazonaws.com/ec2-downloads/ec2-ami-tools.noarch.rpm
```

3. 次のコマンドを使用して RPM ファイルの署名を確認する:

```
[ec2-user ~]$ rpm -K ec2-ami-tools.noarch.rpm
```

上のコマンドは、ファイルの SHA1 および MD5 ハッシュが OK. であることを示しています。ハッシュが NOT OK であることをコマンドが示している場合、次のコマンドを使用してファイルのヘッダー SHA1 および MD5 ハッシュを表示します。

```
[ec2-user ~]$ rpm -Kv ec2-ami-tools.noarch.rpm
```

次に、ファイルのヘッダー SHA1 および MD5 ハッシュを、以下の検証済み AMI ツールハッシュと比較し、ファイルの正統性を確認します。

- ヘッダー SHA1: a1f662d6f25f69871104e6a62187fa4df508f880
- MD5: 9faff05258064e2f7909b66142de6782

ファイルのヘッダー SHA1 および MD5 ハッシュが検証済み AMI ツールハッシュと一致する場合、次のステップに進みます。

4. 次のコマンドを使用して RPM をインストールします。

```
[ec2-user ~]$ sudo yum install ec2-ami-tools.noarch.rpm
```

5. ec2-ami-tools-version コマンドを使用してインストールした AMI ツールを検証します。

```
[ec2-user ~]$ ec2-ami-tools-version
```

> **Note**
>
> [cannot load such file -- ec2/amitools/version (LoadError)] などのロードエラーを受信した場合は、次のステップを実行し、AMI ツールをインストールした場所を RUBYLIB パスに追加します。

6. (オプション) 前のステップでエラーが発生した場合、AMI ツールをインストールした場所を RUBYLIB パスに追加します。

   a. 追加するパスを調べるには、次のコマンドを実行します。

   ```
   [ec2-user ~]$ rpm -qil ec2-ami-tools | grep ec2/amitools/version
   /usr/lib/ruby/site_ruby/ec2/amitools/version.rb
   /usr/lib64/ruby/site_ruby/ec2/amitools/version.rb
   ```

   上記の例では、以前のロードエラーから失われたファイルは /usr/lib/ruby/site_ruby および /usr/lib64/ruby/site_ruby にあります。

   b. 前のステップの場所を RUBYLIB パスに追加します。

   ```
   [ec2-user ~]$ export RUBYLIB=$RUBYLIB:/usr/lib/ruby/site_ruby:/usr/lib64/ruby/site_ruby
   ```

   c. ec2-ami-tools-version コマンドを使用してインストールした AMI ツールを検証します。

   ```
   [ec2-user ~]$ ec2-ami-tools-version
   ```

zip ファイルを使用して AMI ツールを設定するには

1. Ruby をインストールし、apt-get など、Linux ディストリビューション用のパッケージマネージャを使用して解凍します。次に例を示します。

   ```
   [ec2-user ~]$ sudo apt-get update -y && sudo apt-get install -y ruby unzip
   ```

2. wget や curl などのツールを使用して .zip ファイルをダウンロードします。次に例を示します。

   ```
   [ec2-user ~]$ wget https://s3.amazonaws.com/ec2-downloads/ec2-ami-tools.zip
   ```

3. /usr/local/ec2 など、適切なインストールディレクトリにファイルを解凍します。

```
[ec2-user ~]$ sudo mkdir -p /usr/local/ec2
$ sudo unzip ec2-ami-tools.zip -d /usr/local/ec2
```

.zip ファイルには、フォルダ (ec2-ami-tools-*x.x.x*) が含まれます。ここで、*x.x.x* はツールのバージョン番号 (例: ec2-ami-tools-1.5.7) です。

4. EC2_AMITOOL_HOME 環境変数を、ツールのインストールディレクトリに設定します。次に例を示します。

```
[ec2-user ~]$ export EC2_AMITOOL_HOME=/usr/local/ec2/ec2-ami-tools-x.x.x
```

5. ツールを PATH 環境変数に追加します。次に例を示します。

```
[ec2-user ~]$ export PATH=$EC2_AMITOOL_HOME/bin:$PATH
```

6. ec2-ami-tools-version コマンドを使用してインストールした AMI ツールを検証できます。

```
[ec2-user ~]$ ec2-ami-tools-version
```

署名証明書の管理

AMI ツールの特定のコマンドでは、デジタル署名用証明書 (X.509 証明書とも呼ばれる) が必要です。証明書を作成し、AWS にアップロードする必要があります。例えば、証明書の作成に OpenSSL などのサードパーティ製のツールを使用できます。

デジタル署名用証明書を作成するには

1. OpenSSL をインストールおよび設定します。
2. プライベートキーを openssl genrsa コマンドを使用して作成し、出力を .pem ファイルで保存します。2048 ビットまたは 4096 ビット RSA キーの作成を推奨しています。

```
openssl genrsa 2048 > private-key.pem
```

3. openssl req コマンドを使用して、証明書を作成します。

```
openssl req -new -x509 -nodes -sha256 -days 365 -key private-key.pem -outform PEM -
out certificate.pem
```

証明書を AWS にアップロードするには、upload-signing-certificate コマンドを使用します。

```
aws iam upload-signing-certificate --user-name user-name --certificate-body
 file://path/to/certificate.pem
```

ユーザーの証明書を一覧表示するには、list-signing-certificates コマンドを使用します。

```
aws iam list-signing-certificates --user-name user-name
```

ユーザーのデジタル署名用証明書を無効化または再有効化するには、update-signing-certificate コマンドを使用します。次のコマンドは証明書を無効にします。

```
aws iam update-signing-certificate --certificate-id OFHPLP4ZULTHYPMSYEX7O4BEXAMPLE --
status Inactive --user-name user-name
```

証明書を削除するには、delete-signing-certificate コマンドを使用します。

```
aws iam delete-signing-certificate --user-name user-name --certificate-
id OFHPLP4ZULTHYPMSYEX7O4BEXAMPLE
```

Instance Store-Backed インスタンスからの AMI の作成

次の手順では、instance store-backed インスタンスから instance store-backed AMI を作成します。開始する前に、必ず「前提条件」を参照してください。

トピック

- Instance Store-Backed Amazon Linux インスタンスからの AMI の作成
- Instance Store-Backed Ubuntu インスタンスからの AMI の作成

Instance Store-Backed Amazon Linux インスタンスからの AMI の作成

このセクションでは、Amazon Linux インスタンスからの AMI の作成について説明します。以下の手順は、他の Linux ディストリビューションを実行するインスタンスでは機能しない可能性があります。Ubuntu 固有の手順については、「Instance Store-Backed Ubuntu インスタンスからの AMI の作成」を参照してください。

AMI ツールの使用準備を整えるには (HVM インスタンスのみ)

1. AMI ツールでは、GRUB のレガシーが正常に起動する必要があります。次のコマンドを使用して GRUB をインストールします。

```
[ec2-user ~]$ sudo yum install -y grub
```

2. 次のコマンドを使用して、パーティション管理パッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y gdisk kpartx parted
```

Instance Store-Backed Amazon Linux インスタンスから AMI を作成するには

この手順では、「前提条件」に記載された前提条件が満たされていることを前提としています。

次のコマンドでは、*###############*をユーザー自身の情報で置き換えます。

1. インスタンスに認証情報をアップロードします。Amazon ではこれらの認証情報を使用して、お客様と Amazon EC2 だけがお客様の AMI にアクセスできるようにします。

   a. 次のように、認証情報のための一時ディレクトリをインスタンスに作成します。

   ```
   [ec2-user ~]$ mkdir /tmp/cert
   ```

   それにより、作成したイメージから認証情報を除外できます。

   b. scp などの安全なコピーツールを使用して、コンピュータからインスタンスの /tmp/cert ディレクトリに X.509 証明書と対応するプライベートキーをコピーします。次の -i *my-private-key*.pem コマンドの scp オプションは、X.509 プライベートキーではなく、SSH でインスタンスに接続するために使用するプライベートキーです。次に例を示します。

   ```
   you@your_computer:~ $ scp -i my-private-key.pem /
   path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem /
   path/to/cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem ec2-
   user@ec2-203-0-113-25.compute-1.amazonaws.com:/tmp/cert/
   pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  717     0.7KB/s   00:00
   cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  685     0.7KB/s   00:00
   ```

または、これらがプレーンテキストファイルの場合、証明書とキーをテキストエディタで開き、コンテンツを /tmp/cert の新しいファイルにコピーできます。

2. インスタンス内から ec2-bundle-vol コマンドを実行して、Amazon S3 にアップロードするバンドルを準備します。-e オプションを指定して、認証情報を保存するディレクトリを除外します。デフォルトでは、バンドルプロセスで機密情報を含んでいる可能性があるファイルを除外します。ファイルには、*.sw、*.swo、*.swp、*.pem、*.priv、*id_rsa*、*id_dsa*、*.gpg、*.jks、*/.ssh/authorized_keys、*/.bash_history などがあります。これらのファイルをすべて含めるには、--no-filter オプションを使用します。これらのファイルの一部を含めるには、--include オプションを使用します。

> **Important**
>
> AMI バンドルプロセスは、デフォルトで、ルートボリュームを表す /tmp ディレクトリに、圧縮され暗号化された一連のファイルを作成します。バンドルを格納するのに十分な空きディスク領域が /tmp にない場合、-d */path/to/bundle/storage* オプションを使用して、バンドルを格納する別の場所を指定する必要があります。インスタンスによっては、エフェメラルストレージが /mnt または /media/ephemeral0 にマウントされて使用可能になっている場合があります。または、バンドルを格納する新しい Amazon EBS ボリュームを作成、アタッチ、およびマウントすることもできます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの作成」を参照してください。

   a. ec2-bundle-vol コマンドは、root として実行する必要があります。ほとんどのコマンドで、sudo を使用することでアクセス許可を昇格させることができますが、この場合は、環境変数を維持するために sudo -E su を実行する必要があります。

      ```
      [ec2-user ~]$ sudo -E su
      ```

      これで、bash プロンプトにより root ユーザーとして識別されるようになったことと、root シェルにいることを示すハッシュタグにドル記号が置き換えられたことに注意してください。

      ```
      [root ec2-user]#
      ```

b. AMI のバンドルを作成するには、次のように ec2-bundle-vol コマンドを実行します。

```
[root ec2-user]# ec2-bundle-vol -k /tmp/cert/pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c /tmp/cert/cert-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u 123456789012 -r x86_64 -e /tmp/cert --
partition gpt
```

> **Note**
>
> 中国 (北京) および AWS GovCloud (米国 – 西部) リージョンについては、--ec2cert パラメータを使用し、前提条件に従って証明書を指定します。

イメージの作成には数分かかります。このコマンドが完了したら、/tmp (またはデフォルト以外の) ディレクトリにバンドルが含まれます (image.manifest.xml、および複数の image.part.*xx* ファイル)。

c. root シェルを終了します。

```
[root ec2-user]# exit
```

3. (オプション) インスタンスストアをさらに追加するには、AMI 用の image.manifest.xml ファイルで、ブロックデバイスマッピングを編集します。詳細については、ブロックデバイスマッピング を参照してください。

a. image.manifest.xml ファイルのバックアップを作成します。

```
[ec2-user ~]$ sudo cp /tmp/image.manifest.xml /tmp/image.manifest.xml.bak
```

b. image.manifest.xml ファイルの形式を変更し、読み取りと編集が簡単になるようにします。

```
[ec2-user ~]$ sudo xmllint --format /tmp/image.manifest.xml.bak > /tmp/
image.manifest.xml
```

c. テキストエディタで image.manifest.xml のブロックデバイスマッピングを編集します。次の例は、ephemeral1 インスタンスストアボリュームの新しいエントリを示しています。

> **Note**
>
> 無効な種類のファイルの一覧については、「ec2-bundle-vol」を参照してください。

```
<block_device_mapping>
  <mapping>
    <virtual>ami</virtual>
    <device>sda</device>
  </mapping>
  <mapping>
    <virtual>ephemeral0</virtual>
    <device>sdb</device>
  </mapping>
  <mapping>
    <virtual>ephemeral1</virtual>
    <device>sdc</device>
  </mapping>
  <mapping>
    <virtual>root</virtual>
    <device>/dev/sda1</device>
  </mapping>
</block_device_mapping>
```

d. image.manifest.xml ファイルを保存し、テキストエディタを終了します。

4. バンドルを Amazon S3 にアップロードするには、次のように ec2-upload-bundle コマンドを実行します。

```
[ec2-user ~]$ ec2-upload-bundle -b my-s3-bucket/bundle_folder/bundle_name -m /tmp/
image.manifest.xml -a your_access_key_id -s your_secret_access_key
```

> **Important**
>
> US East (N. Virginia) 以外のリージョンで AMI を登録するには、--region オプションと、すでにターゲットリージョンに存在するバケットパス、またはターゲットリージョンで作成できる一意のバケットパスの両方でターゲットリージョンを指定する必要があります。

5. (オプション) バンドルを Amazon S3 にアップロードしたら、次の /tmp コマンドを使用して、インスタンスの rm ディレクトリからバンドルを削除できます。

```
[ec2-user ~]$ sudo rm /tmp/image.manifest.xml /tmp/image.part.* /tmp/image
```

> **Important**
>
> -d */path/to/bundle/storage* で Step 2 オプションを使用してパスを指定した場合は、/tmp ではなくそのパスを使用します。

6. AMI を登録するには、次のように register-image コマンドを実行します。

```
[ec2-user ~]$ aws ec2 register-image --image-location my-s3-
bucket/bundle_folder/bundle_name/image.manifest.xml --name AMI_name --
virtualization-type hvm
```

> **Important**
>
> ec2-upload-bundle コマンドでリージョンを以前に指定した場合は、このコマンドでもう一度そのリージョンを指定します。

Instance Store-Backed Ubuntu インスタンスからの AMI の作成

このセクションでは、インスタンスストアボリュームをルートボリュームとして使用する Ubuntu Linux インスタンスからの AMI の作成について説明します。以下の手順は、他の Linux ディストリビューションを実行するインスタンスでは機能しない可能性があります。Amazon Linux 固有の手順については、「Instance Store-Backed Amazon Linux インスタンスからの AMI の作成」を参照してください。

AMI ツールの使用準備を整えるには (HVM インスタンスのみ)

AMI ツールでは、GRUB のレガシーが正常に起動する必要があります。ただし、Ubuntu は GRUB 2 を使用するように設定されています。インスタンスで GRUB のレガシーを使用しているかどうか確認し、使用していない場合はインストールして設定する必要があります。

AMI ツールが正常に機能するためには、HVM インスタンスにパーティションツールがインストールされている必要もあります。

1. GRUB Legacy (バージョン 0.9*x* 未満) をインスタンスにインストールする必要があります。GRUB Legacy が存在していることを確認し、必要な場合はインストールしてください。

   a. GRUB インストールのバージョンを確認します。

   ```
   ubuntu:~$ grub-install --version
   grub-install (GRUB) 1.99-21ubuntu3.10
   ```

   この例では、GRUB バージョンが 0.9*x* 以上のため、GRUB Legacy をインストールする必要があります。Step 1.b に進みます。GRUB Legacy が既にある場合、「Step 2」までスキップできます。

   b. 次のコマンドを使用して grub パッケージをインストールします。

   ```
   ubuntu:~$ sudo apt-get install -y grub
   ```

2. お使いのディストリビューションのパッケージマネージャを使用して、次のパーティション管理パッケージをインストールします。

   - gdisk (ディストリビューションによっては代わりにパッケージ gptfdisk が呼び出される場合があります)。
   - kpartx
   - parted

   次のコマンドを使用します。

   ```
   ubuntu:~$ sudo apt-get install -y gdisk kpartx parted
   ```

3. インスタンスのカーネルパラメータを確認します。

   ```
   ubuntu:~$ cat /proc/cmdline
   BOOT_IMAGE=/boot/vmlinuz-3.2.0-54-virtual root=UUID=4f392932-ed93-4f8f-
   aee7-72bc5bb6ca9d ro console=ttyS0 xen_emul_unplug=unnecessary
   ```

   カーネルおよびルートデバイスのパラメータ ro、console=ttyS0、および xen_emul_unplug=unnecessary を書き留めます。オプションは異なる場合があります。

4. /boot/grub/menu.lst でカーネルエントリを確認してください。

   ```
   ubuntu:~$ grep ^kernel /boot/grub/menu.lst
   ```

```
kernel  /boot/vmlinuz-3.2.0-54-virtual root=LABEL=cloudimg-rootfs ro console=hvc0
kernel  /boot/vmlinuz-3.2.0-54-virtual root=LABEL=cloudimg-rootfs ro single
kernel  /boot/memtest86+.bin
```

console パラメータが hvc0 をポイントしている (ttyS0 ではない) こと、および xen_emul_unplug=unnecessary パラメータが未指定であることに注意してください。ここでも、オプションは異なる場合があります。

5. /boot/grub/menu.lst ファイルを任意のテキストエディタで (vim や nano など) で編集して、コンソールを変更し、先ほど確認したパラメータをブートエントリに追加します。

```
title           Ubuntu 12.04.3 LTS, kernel 3.2.0-54-virtual
root            (hd0)
kernel          /boot/vmlinuz-3.2.0-54-virtual root=LABEL=cloudimg-rootfs
 ro console=ttyS0 xen_emul_unplug=unnecessary
initrd          /boot/initrd.img-3.2.0-54-virtual

title           Ubuntu 12.04.3 LTS, kernel 3.2.0-54-virtual (recovery mode)
root            (hd0)
kernel          /boot/vmlinuz-3.2.0-54-virtual root=LABEL=cloudimg-rootfs ro
 single console=ttyS0 xen_emul_unplug=unnecessary
initrd          /boot/initrd.img-3.2.0-54-virtual

title           Ubuntu 12.04.3 LTS, memtest86+
root            (hd0)
kernel          /boot/memtest86+.bin
```

6. カーネルエントリに適切なパラメータが含まれていることを確認します。

```
ubuntu:~$ grep ^kernel /boot/grub/menu.lst
kernel  /boot/vmlinuz-3.2.0-54-virtual root=LABEL=cloudimg-rootfs ro console=ttyS0
 xen_emul_unplug=unnecessary
kernel  /boot/vmlinuz-3.2.0-54-virtual root=LABEL=cloudimg-rootfs ro single
 console=ttyS0 xen_emul_unplug=unnecessary
kernel  /boot/memtest86+.bin
```

7. (Ubuntu 14.04 以降のみ) Ubuntu 14.04 で起動する Instance Store-Backed Ubuntu AMI は GPT のパーティションテーブルおよび /boot/efi にマウントされた別の EFI のパーティションを使用します。ec2-bundle-vol コマンドはこの起動パーティションをバンドルしません。そのため、次の例に示すように EFI のパーティションの /etc/fstab エントリをコメントアウトする必要があります。

```
LABEL=cloudimg-rootfs   /        ext4   defaults        0 0
#LABEL=UEFI     /boot/efi       vfat    defaults        0 0
/dev/xvdb       /mnt    auto    defaults,nobootwait,comment=cloudconfig 0       2
```

Instance Store-Backed Ubuntu インスタンスから AMI を作成するには

この手順では、「前提条件」に記載された前提条件が満たされていることを前提としています。

次のコマンドでは、*############*をユーザー自身の情報で置き換えます。

1. インスタンスに認証情報をアップロードします。Amazon ではこれらの認証情報を使用して、お客様と Amazon EC2 だけがお客様の AMI にアクセスできるようにします。

   a. 次のように、認証情報のための一時ディレクトリをインスタンスに作成します。

   ```
   ubuntu:~$ mkdir /tmp/cert
   ```

   それにより、作成したイメージから認証情報を除外できます。

   b. scp などの安全なコピーツールを使用して、コンピュータからインスタンスの /tmp/cert ディレクトリに X.509 証明書とプライベートキーをコピーします。次の -i *my-private-key*.pem コマンドの scp オプションは、X.509 プライベートキーではなく、SSH でインスタンスに接続するために使用するプライベートキーです。次に例を示します。

   ```
   you@your_computer:~ $ scp -i my-private-key.pem /
   path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem /
   path/to/cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem ec2-
   user@ec2-203-0-113-25.compute-1.amazonaws.com:/tmp/cert/
   pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  717     0.7KB/s   00:00
   cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  685     0.7KB/s   00:00
   ```

   または、これらがプレーンテキストファイルの場合、証明書とキーをテキストエディタで開き、コンテンツを /tmp/cert の新しいファイルにコピーできます。

2. インスタンスから ec2-bundle-vol コマンドを実行して、Amazon S3 にアップロードするバンドルを準備します。-e オプションを指定して、認証情報を保存するディレクトリを除外します。デフォルトでは、バンドルプロセスで機密情報を含んでいる可能性があるファイルを除外します。ファイルには、*.sw、*.swo、*.swp、*.pem、*.priv、*id_rsa*、*id_dsa*、*.gpg、*.jks、*/.ssh/

authorized_keys、*/.bash_history などがあります。これらのファイルをすべて含めるには、--no-filter オプションを使用します。これらのファイルの一部を含めるには、--include オプションを使用します。

> **Important**
>
> AMI バンドルプロセスは、デフォルトで、ルートボリュームを表す /tmp ディレクトリに、圧縮され暗号化された一連のファイルを作成します。バンドルを格納するのに十分な空きディスク領域が /tmp にない場合、-d */path/to/bundle/storage* オプションを使用して、バンドルを格納する別の場所を指定する必要があります。インスタンスによっては、エフェメラルストレージが /mnt または /media/ephemeral0 にマウントされて使用可能になっている場合があります。または、バンドルを格納する新しい Amazon EBS ボリュームを作成、アタッチ、およびマウントすることもできます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの作成」を参照してください。

a. ec2-bundle-vol コマンドは、root として実行する必要があります。ほとんどのコマンドで、sudo を使用することでアクセス許可を昇格させることができますが、この場合は、環境変数を維持するために sudo -E su を実行する必要があります。

```
ubuntu:~$ sudo -E su
```

これで、bash プロンプトにより root ユーザーとして識別されるようになったことと、root シェルにいることを示すハッシュタグにドル記号が置き換えられたことに注意してください。

```
root@ubuntu:#
```

b. AMI のバンドルを作成するには、次のように ec2-bundle-vol コマンドを実行します。

```
root@ubuntu:# ec2-bundle-vol -k /tmp/cert/pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c /tmp/cert/cert-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u your_aws_account_id -r x86_64 -e /tmp/
cert --partition gpt
```

> **Important**
>
> Ubuntu 14.04 以降の HVM インスタンスの場合、`--partition mbr` フラグを追加して起動手順を正しくバンドルします。それ以外の場合は、新しく作成された AMI は起動しません。

イメージの作成には数分かかります。このコマンドが完了したら、tmp ディレクトリにバンドルが含まれます (image.manifest.xml、および複数の image.part.*xx* ファイル)。

c. root シェルを終了します。

```
root@ubuntu:# exit
```

3. (オプション) インスタンスストアをさらに追加するには、AMI 用の image.manifest.xml ファイルで、ブロックデバイスマッピングを編集します。詳細については、ブロックデバイスマッピング を参照してください。

   a. image.manifest.xml ファイルのバックアップを作成します。

   ```
   ubuntu:~$ sudo cp /tmp/image.manifest.xml /tmp/image.manifest.xml.bak
   ```

   b. image.manifest.xml ファイルの形式を変更し、読み取りと編集が簡単になるようにします。

   ```
   ubuntu:~$ sudo xmllint --format /tmp/image.manifest.xml.bak > /tmp/
   image.manifest.xml
   ```

   c. テキストエディタで image.manifest.xml のブロックデバイスマッピングを編集します。次の例は、*ephemeral1* インスタンスストアボリュームの新しいエントリを示しています。

   ```
       <block_device_mapping>
         <mapping>
           <virtual>ami</virtual>
           <device>sda</device>
         </mapping>
         <mapping>
           <virtual>ephemeral0</virtual>
   ```

```
        <device>sdb</device>
      </mapping>
      <mapping>
        <virtual>ephemeral1</virtual>
        <device>sdc</device>
      </mapping>
      <mapping>
        <virtual>root</virtual>
        <device>/dev/sda1</device>
      </mapping>
    </block_device_mapping>
```

d. image.manifest.xml ファイルを保存し、テキストエディタを終了します。

4. バンドルを Amazon S3 にアップロードするには、次のように ec2-upload-bundle コマンドを実行します。

```
ubuntu:~$ ec2-upload-bundle -b my-s3-bucket/bundle_folder/bundle_name -m /tmp/
image.manifest.xml -a your_access_key_id -s your_secret_access_key
```

> **Important**
>
> US East (N. Virginia) 以外のリージョンで AMI を登録する予定の場合、--region オプションと、すでにターゲットリージョンに存在するバケットパス、またはターゲットリージョンで作成できる一意のバケットパスの両方でターゲットリージョンを指定する必要があります。

5. (オプション) バンドルを Amazon S3 にアップロードしたら、次の /tmp コマンドを使用して、インスタンスの rm ディレクトリからバンドルを削除できます。

```
ubuntu:~$ sudo rm /tmp/image.manifest.xml /tmp/image.part.* /tmp/image
```

> **Important**
>
> -d */path/to/bundle/storage* で Step 2 オプションを使用してパスを指定した場合、/tmp ではなく以下と同じパスを使用します。

6. AMI を登録するには、次のように AWS CLI の register-image コマンドを実行します。

```
ubuntu:~$ aws ec2 register-image --image-location my-s3-
bucket/bundle_folder/bundle_name/image.manifest.xml --name AMI_name --
virtualization-type hvm
```

> **Important**
>
> ec2-upload-bundle コマンドでリージョンを以前に指定した場合は、このコマンドでもう一度そのリージョンを指定します。

7. (Ubuntu 14.04 以降) /etc/fstab の EFI エントリをコメント解除します。それ以外の場合、実行中のインスタンスは再起動できません。

instance store-backed AMI を Amazon EBS-backed AMI への変換

Instance Store-Backed Linux AMI は、Amazon EBS-Backed Linux AMI に変換できます。

> **Important**
>
> Instance Store-Backed Windows AMI から Amazon EBS-Backed Windows AMI への変換、および所有していない AMI の変換はできません。

Instance Store-Backed AMI を Amazon EBS-backed AMI に変換するには

1. Amazon EBS-backed AMI から Amazon Linux インスタンスを起動します。詳細については、新しいインスタンス起動ウィザードを使用してインスタンスを起動する を参照してください。Amazon Linux インスタンスには、AWS CLI および AMI ツールがプリインストールされています。
2. Instance Store-Backed AMI をバンドルするのに使用した X.509 プライベートキーをインスタンスにアップロードします。Amazon はこのキーを使用して、お客様と Amazon EC2 だけがお客様の AMI にアクセスできるようにします。

   a. 次のように、X.509 プライベートキーのインスタンスに一時ディレクトリを作成します。

   ```
   [ec2-user ~]$ mkdir /tmp/cert
   ```

b. scp などの安全なコピーツールを使用して、コンピュータから /tmp/cert ディレクトリに X.509 プライベートキーをコピーします。次のコマンドの *my-private-key* パラメータは、SSH でインスタンスに接続するために使用するプライベートキーです。例:

```
you@your_computer:~ $ scp -i my-private-key.pem /
path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem ec2-
user@ec2-203-0-113-25.compute-1.amazonaws.com:/tmp/cert/
pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem  100%  717     0.7KB/s   00:00
```

3. 環境変数を設定して、AWS CLI を使用します。詳細については、「キーペアの作成」を参照してください。

   a. (推奨) AWS アクセスキー、シークレットキーおよびセッショントークンの環境変数を設定します。

```
[ec2-user ~]$ export AWS_ACCESS_KEY_ID=your_access_key_id
[ec2-user ~]$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
[ec2-user ~]$ export AWS_SESSION_TOKEN=your_session_token
```

   b. AWS アクセスキーおよびシークレットキーの環境変数を設定します。

```
[ec2-user ~]$ export AWS_ACCESS_KEY_ID=your_access_key_id
[ec2-user ~]$ export AWS_SECRET_ACCESS_KEY=your_secret_access_key
```

4. 新しい AMI の Amazon Elastic Block Store (Amazon EBS) ボリュームを準備します。

   a. create-volume コマンドを使用して、インスタンスと同じアベイラビリティーゾーンに空の EBS ボリュームを作成します。コマンド出力のボリューム ID を書き留めてください。

> **Important**
>
> この EBS ボリュームは、元のインスタンスストアのルートボリュームと同じサイズ以上である必要があります。

```
[ec2-user ~]$ aws ec2 create-volume --size 10 --region us-west-2 --
availability-zone us-west-2b
```

   b. attach-volume コマンドを使用して、Amazon EBS-Backed インスタンスにボリュームをアタッチします。

```
[ec2-user ~]$ aws ec2 attach-volume --volume-id volume_id --instance-
id instance_id --device /dev/sdb --region us-west-2
```

5. バンドルのフォルダを作成します。

```
[ec2-user ~]$ mkdir /tmp/bundle
```

6. /tmp/bundle コマンドを使用して、Instance Store-Backe AMI のバンドルを ec2-download-bundle にダウンロードします。

```
[ec2-user ~]$ ec2-download-bundle -b my-s3-bucket/bundle_folder/bundle_name -m
 image.manifest.xml -a $AWS_ACCESS_KEY_ID -s $AWS_SECRET_ACCESS_KEY --privatekey /
path/to/pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -d /tmp/bundle
```

7. ec2-unbundle コマンドを使用して、バンドルからイメージファイルを再作成します。

   a. バンドルフォルダにディレクトリを変更します。

```
[ec2-user ~]$ cd /tmp/bundle/
```

   b. ec2-unbundle コマンドを実行します。

```
[ec2-user bundle]$ ec2-unbundle -m image.manifest.xml --privatekey /path/to/pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem
```

8. バンドルを解除したイメージから新しい EBS ボリュームにファイルをコピーします。

```
[ec2-user bundle]$ sudo dd if=/tmp/bundle/image of=/dev/sdb bs=1M
```

9. バンドルを解除した新しいパーティションのボリュームを調査します。

```
[ec2-user bundle]$ sudo partprobe /dev/sdb1
```

10. ブロックデバイスの一覧を表示してマウントするデバイス名を選択します。

```
[ec2-user bundle]$ lsblk
NAME         MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
/dev/sda    202:0    0   8G  0 disk
##/dev/sda1 202:1    0   8G  0 part /
/dev/sdb    202:80   0  10G  0 disk
```

```
##/dev/sdb1 202:81   0  10G  0 part
```

この例では、マウントするパーティションは /dev/sdb1 ですが、デバイス名はおそらく異なります。ボリュームが仕切られていない場合は、マウントするデバイスは /dev/sdb に似ています (デバイスパーティションの末尾に数値なし)。

11. 新しい EBS ボリュームのマウントポイントを作成し、ボリュームをマウントします。

```
[ec2-user bundle]$ sudo mkdir /mnt/ebs
[ec2-user bundle]$ sudo mount /dev/sdb1 /mnt/ebs
```

12. EBS ボリュームの /etc/fstab ファイルを任意のテキストエディタ (vim や nano など) で開き、インスタンスストア (エフェメラル) ボリュームのエントリがあれば削除します。EBS ボリュームが /mnt/ebs に取付けられるため、fstab ファイルは /mnt/ebs/etc/fstab にあります。

```
[ec2-user bundle]$ sudo nano /mnt/ebs/etc/fstab
#
LABEL=/     /           ext4    defaults,noatime  1   1
tmpfs       /dev/shm    tmpfs   defaults        0   0
devpts      /dev/pts    devpts  gid=5,mode=620  0   0
sysfs       /sys        sysfs   defaults        0   0
proc        /proc       proc    defaults        0   0
/dev/sdb        /media/ephemeral0       auto    defaults,comment=cloudconfig    0
     2
```

この例では、最後の行を削除する必要があります。

13. ボリュームをアンマウントし、インスタンスからデタッチします。

```
[ec2-user bundle]$ sudo umount /mnt/ebs
[ec2-user bundle]$ aws ec2 detach-volume --volume-id volume_id --region us-west-2
```

14. 次のように、新しい EBS ボリュームから AMI を作成します。

    a. 新しい EBS ボリュームのスナップショットを作成します。

```
[ec2-user bundle]$ aws ec2 create-snapshot --region us-west-2 --description
 "your_snapshot_description" --volume-id volume_id
```

    b. スナップショットが完了していることを確認します。

```
[ec2-user bundle]$ aws ec2 describe-snapshots --region us-west-2 --snapshot-
id snapshot_id
```

c. 元の AMI で使用されたプロセッサアーキテクチャ、仮想化タイプ、カーネルイメージ (aki) を、describe-images コマンドを使用して特定します。このステップでは、元の Instance Store-Backed AMI の AMI ID が必要です。

```
[ec2-user bundle]$ aws ec2 describe-images --region us-west-2 --image-id ami-id
 --output text
IMAGES x86_64 amazon/amzn-ami-pv-2013.09.2.x86_64-s3 ami-8ef297be amazon
 available public machine aki-fc8f11cc instance-store paravirtual xen
```

この例では、アーキテクチャは x86_64 で、カーネルイメージ ID は aki-fc8f11cc です。次のステップでこれらの値を使用します。前述のコマンドの出力では ari ID もリストされますので、これも書き留めます。

d. 新しい EBS ボリュームのスナップショット ID と前のステップで書き留めた値を使用して、新しい AMI を登録します。前述のコマンド出力に ari ID がリストされていた場合は、その ID を次のコマンドで --ramdisk-id *ari_id* を使用して指定します。

```
[ec2-user bundle]$ aws ec2 register-image --region us-west-2 --
name your_new_ami_name --block-device-mappings DeviceName=device-
name,Ebs={SnapshotId=snapshot_id} --virtualization-type paravirtual --
architecture x86_64 --kernel-id aki-fc8f11cc --root-device-name device-name
```

15. (オプション) 新しい AMI からインスタンスを起動できることをテストした後で、この手順で作成した EBS ボリュームを削除できます。

```
aws ec2 delete-volume --volume-id volume_id
```

AMI ツールリファレンス

AMI ツールコマンドを使用して、Instance Store-Backed Linux AMI を作成および管理できます。ツールをセットアップする方法は、「AMI ツールのセットアップ」を参照してください。

アクセスキーの詳細については、「AWS Account Management リファレンスガイド」の「AWS アカウントのベストプラクティス」を参照してください。

コマンド

- ec2-ami-tools-version
- ec2-bundle-image
- ec2-bundle-vol
- ec2-delete-bundle
- ec2-download-bundle
- ec2-migrate-manifest
- ec2-unbundle
- ec2-upload-bundle
- AMI ツール用の一般的なオプション

ec2-ami-tools-version

説明

AMI ツールのバージョンについて説明します。

Syntax

ec2-ami-tools-version

出力

バージョン情報。

例

このコマンド例では、使用中の AMI ツールのバージョン情報を表示します。

```
[ec2-user ~]$ ec2-ami-tools-version
1.5.2 20071010
```

ec2-bundle-image

説明

ループバックファイル内に作成されるオペレーティングシステムイメージから instance store-backed Linux AMI を作成します。

Syntax

ec2-bundle-image -c *path* -k *path* -u *account* -i *path* [-d *path*] [--ec2cert *path*] [-r *architecture*] [--productcodes *code1*,*code2*,...] [-B *mapping*] [-p *prefix*]

オプション

`-c, --cert` パス

ユーザーの PEM エンコード RSA パブリックキー証明書ファイル。

必須: はい

`-k, --privatekey` パス

PEM エンコードされる RSA キーファイルへのパス。このバンドルをバンドル解除するには、このキーを指定する必要があるため、安全な場所に保管してください。このキーは AWS アカウントに登録されている必要はありません。

必須: はい

`-u, --user` アカウント

ダッシュのない、ユーザーの AWS アカウント ID。

必須: はい

`-i, --image` パス

バンドルするイメージへのパス。

必須: はい

`-d, --destination` パス

バンドルを作成するディレクトリ。

デフォルト: `/tmp`

必須: いいえ

`--ec2cert` パス

イメージマニフェストの暗号化に使用される Amazon EC2 X.509 パブリックキー証明書へのパス。

us-gov-west-1 および cn-north-1 リージョンではデフォルト以外のパブリックキー証明書を使用し、その証明書へのパスは、このオプションで指定する必要があります。証明書へのパスは、AMI ツールのインストール方法によって異なります。Amazon Linux の場合、証明書の場所は /opt/aws/amitools/ec2/etc/ec2/amitools/ です。「AMI ツールのセットアップ」の RPM または ZIP ファイルから AMI ツールをインストールした場合、証明書の場所は $EC2_AMITOOL_HOME/etc/ec2/amitools/ です。

必須: us-gov-west-1 および cn-north-1 リージョンのみ。

-r, --arch アーキテクチャ

イメージアーキテクチャ。コマンドラインでアーキテクチャを指定しない場合、バンドルの開始時に入力を求められます。

有効な値: i386 | x86_64

必須: いいえ

--productcodes code1、code2、...

登録時にイメージにアタッチする、カンマ区切りの製品コード。

必須: いいえ

-B, --block-device-mapping マッピング

インスタンスタイプが指定されたデバイスをサポートする場合に、この AMI のインスタンスにブロックデバイスを公開する方法を定義します。

キーと値のペアのカンマ区切りのペアを指定します。名キーは仮想名であり、各値は対応するデバイス名です。仮想名には以下が含まれています。

- ami — インスタンスによって判断されるルートファイルシステムデバイス
- root — カーネルによって判断されるルートファイルシステムデバイス
- swap — インスタンスによって判断されるスワップデバイス
- ephemeralN — N 番目のインスタンスストアボリューム

必須: いいえ

-p, --prefixprefix

バンドル済み AMI ファイルのファイル名プレフィクス。

デフォルト: イメージファイルの名前。例えば、イメージパスが /var/spool/my-image/version-2/debian.img である場合、デフォルトのプレフィクスは debian.img です。

必須: いいえ

--kernel kernel_id

廃止済み。カーネルを設定するには、register-image を使用します。

必須: いいえ

--ramdisk ramdisk_id

廃止。必要に応じて、register-image を使用して RAM ディスクを設定します。

必須: いいえ

出力

バンドルプロセスのステージとステータスを記述するステータスメッセージ。

例

この例は、ループバックファイルで作成されたオペレーティングシステムイメージから、バンドルされた AMI を作成します。

```
[ec2-user ~]$ ec2-bundle-image -k pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c cert-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u 111122223333 -i image.img -d bundled/ -r x86_64
Please specify a value for arch [i386]:
Bundling image file...
Splitting bundled/image.gz.crypt...
Created image.part.00
Created image.part.01
Created image.part.02
Created image.part.03
Created image.part.04
Created image.part.05
Created image.part.06
Created image.part.07
Created image.part.08
Created image.part.09
Created image.part.10
Created image.part.11
Created image.part.12
Created image.part.13
Created image.part.14
Generating digests for each part...
Digests generated.
```

```
Creating bundle manifest...
ec2-bundle-image complete.
```

ec2-bundle-vol

説明

インスタンスのルートデバイスボリュームを圧縮、暗号化、署名することで、instance store-backed Linux AMI を作成します。

Amazon EC2 はインスタンスから製品コード、カーネル設定、RAM ディスク設定、およびブロックデバイスマッピングを継承しようとします。

デフォルトでは、バンドルプロセスで機密情報を含んでいる可能性があるファイルを除外します。ファイルには、`*.sw`、`*.swo`、`*.swp`、`*.pem`、`*.priv`、`*id_rsa*`、`*id_dsa*`、`*.gpg`、`*.jks`、`*/.ssh/authorized_keys`、`*/.bash_history` などがあります。これらのファイルをすべて含めるには、`--no-filter` オプションを使用します。これらのファイルの一部を含めるには、`--include` オプションを使用します。

詳細については、instance store-backed Linux AMI を作成する を参照してください。

Syntax

ec2-bundle-vol -c *path* -k *path* -u *account* [-d *path*] [--ec2cert *path*] [-r *architecture*] [--productcodes *code1*,*code2*,...] [-B *mapping*] [--all] [-e *directory1*,*directory2*,...] [-i *file1*,*file2*,...] [--no-filter] [-p *prefix*] [-s *size*] [--[no-]inherit] [-v *volume*] [-P *type*] [-S *script*] [--fstab *path*] [--generate-fstab] [--grub-config *path*]

オプション

`-c, --cert` パス

ユーザーの PEM エンコード RSA パブリックキー証明書ファイル。

必須: はい

`-k, --privatekey` パス

ユーザーの PEM エンコード RSA キーファイルへのパス。

必須: はい

`-u, --user` アカウント

ダッシュのない、ユーザーの AWS アカウント ID。

必須: はい

`-d, --destination` 送信先

バンドルを作成するディレクトリ。

デフォルト: `/tmp`

必須: いいえ

`--ec2cert` パス

イメージマニフェストの暗号化に使用される Amazon EC2 X.509 パブリックキー証明書へのパス。

`us-gov-west-1` および `cn-north-1` リージョンではデフォルト以外のパブリックキー証明書を使用し、その証明書へのパスは、このオプションで指定する必要があります。証明書へのパスは、AMI ツールのインストール方法によって異なります。Amazon Linux の場合、証明書の場所は `/opt/aws/amitools/ec2/etc/ec2/amitools/` です。「AMI ツールのセットアップ」の RPM または ZIP ファイルから AMI ツールをインストールした場合、証明書の場所は `$EC2_AMITOOL_HOME/etc/ec2/amitools/` です。

必須: `us-gov-west-1` および `cn-north-1` リージョンのみ。

`-r, --arch` アーキテクチャ

イメージアーキテクチャ。コマンドラインでこれを指定しない場合、バンドルの開始時に入力を求められます。

有効な値: `i386 | x86_64`

必須: いいえ

`--productcodes` code1、code2、...

登録時にイメージにアタッチする、カンマ区切りの製品コード。

必須: いいえ

`-B, --block-device-mapping` マッピング

インスタンスタイプが指定されたデバイスをサポートする場合に、この AMI のインスタンスにブロックデバイスを公開する方法を定義します。

キーと値のペアのカンマ区切りのペアを指定します。名キーは仮想名であり、各値は対応するデバイス名です。仮想名には以下が含まれています。

- `ami` — インスタンスによって判断されるルートファイルシステムデバイス
- `root` — カーネルによって判断されるルートファイルシステムデバイス
- `swap` — インスタンスによって判断されるスワップデバイス
- `ephemeralN` — N 番目のインスタンスストアボリューム

必須: いいえ

`-a, --all`

リモートでマウントされたファイルシステムのディレクトリを含めて、すべてのディレクトリをバンドルします。

必須: いいえ

`-e, --exclude` directory1、directory2、…

バンドルオペレーションから除外する絶対ディレクトリパスとファイルのリスト。このパラメータは `--all` オプションを上書きします。除外を指定すると、パラメータとともにリストされたディレクトリとサブディレクトリは、ボリュームにバンドルされません。

必須: いいえ

`-i, --include` file1、file2、…

バンドルオペレーションに含めるファイルのリスト。指定されたファイルは、それ以外の場合は AMI から除外されます。これは、機密情報が含まれる可能性があるためです。

必須: いいえ

`--no-filter`

指定した場合、AMI からファイルは除外されません。これは、機密情報が含まれる可能性があるためです。

必須: いいえ

`-p, --prefix` prefix

バンドル済み AMI ファイルのファイル名プレフィクス。

デフォルト: `image`

必須: いいえ

`-s, --size` サイズ

作成するイメージファイルの MB (1024 * 1024 バイト) 単位のサイズ。最大サイズは 10240 MB です。

デフォルト: 10240

必須: いいえ

`--[no-]inherit`

イメージがインスタンスのメタデータを継承するかどうかを示します (デフォルトでは継承します)。`--inherit` を有効にし、インスタンスメタデータにアクセスできない場合、バンドルは失敗します。

必須: いいえ

`-v, --volume` ボリューム

バンドルを作成する、マウントされたボリュームへの絶対パス。

デフォルト: ルートディレクトリ (/)。

必須: いいえ

`-P, --partition` type

ディスクイメージでパーティションテーブルを使用するかどうかを示します。パーティションテーブルタイプを指定しない場合、デフォルトでは、該当する場合はボリュームの親ブロックデバイスで使用されるタイプになります。それ以外の場合、デフォルトは `gpt` です。

有効な値: `mbr` | `gpt` | `none`

必須: いいえ

`-S, --script` スクリプト

バンドルの直前に実行するカスタマイズスクリプト。スクリプトでは単一の引数である、ボリュームのマウントポイントが予期されます。

必須: いいえ

`--fstab` パス

イメージにバンドルする fstab へのパス。これを指定しない場合、Amazon EC2 は /etc/fstab をバンドルします。

必須: いいえ

`--generate-fstab`

Amazon EC2 で提供される fstab を使用してボリュームをバンドルします。

必須: いいえ

`--grub-config`

イメージにバンドルする別の grub 設定ファイルへのパス。デフォルトでは、`ec2-bundle-vol` は `/boot/grub/menu.lst` または `/boot/grub/grub.conf` が、クローンされたイメージ上に存在することを想定します。このオプションにより、別の grub 設定ファイルへのパスを指定することができ、このファイルはデフォルトに上書きしてコピーされます (存在する場合)。

必須: いいえ

`--kernel` kernel_id

廃止済み。カーネルを設定するには、register-image を使用します。

必須: いいえ

`--ramdiskramdisk_id`

廃止。必要に応じて、register-image を使用して RAM ディスクを設定します。

必須: いいえ

出力

バンドルのステージとステータスを説明するステータスメッセージ。

例

この例では、ローカルマシンのルートファイルシステムのスナップショットを圧縮、暗号化、署名することで、バンドルされた AMI を作成します。

```
[ec2-user ~]$ ec2-bundle-vol -d /mnt -k pk-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -c
 cert-HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -u 111122223333 -r x86_64
  Copying / into the image file /mnt/image...
```

```
  Excluding:
       sys
       dev/shm
       proc
       dev/pts
       proc/sys/fs/binfmt_misc
       dev
       media
       mnt
       proc
       sys
       tmp/image
       mnt/img-mnt
  1+0 records in
  1+0 records out
  mke2fs 1.38 (30-Jun-2005)
  warning: 256 blocks unused.

  Splitting /mnt/image.gz.crypt...
  Created image.part.00
  Created image.part.01
  Created image.part.02
  Created image.part.03
  ...
  Created image.part.22
  Created image.part.23
  Generating digests for each part...
  Digests generated.
  Creating bundle manifest...
  Bundle Volume complete.
```

ec2-delete-bundle

説明

Amazon S3 ストレージから、指定されたバンドルを削除します。バンドルを削除した後で、対応する AMI からインスタンスを起動することはできません。

Syntax

ec2-delete-bundle -b *bucket* -a *access_key_id* -s *secret_access_key* [-t *token*] [--url *url*] [--region *region*] [--sigv *version*] [-m *path*] [-p *prefix*] [--clear] [--retry] [-y]

オプション

`-b, --bucket` bucket

バンドルされた AMI に続いてオプションの '/' 区切りパスプレフィクスを含む Amazon S3 バケットの名前

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: はい

`-s, --secret-key` secret_access_key

AWS シークレットアクセスキー。

必須: はい

`-t, --delegation-token` トークン

AWS リクエストに渡す委任トークン。詳細については、「一時的なセキュリティ認証情報の使用」を参照してください。

必須: 一時的なセキュリティ認証情報を使用している場合のみ。

デフォルト: `AWS_DELEGATION_TOKEN` 環境変数の値 (設定されている場合)。

`--region`リージョン

リクエスト署名で使用するリージョン。

デフォルト: `us-east-1`

必須: 署名バージョン 4 を使用する場合は必須

`--sigvversion`

リクエストに署名するときに使用する署名バージョン。

有効な値: 2 | 4

デフォルト: 4

必須: いいえ

`-m, --manifest`パス

マニフェストファイルへのパス。

必須: `--prefix` または `--manifest` のどちらかを指定する必要があります。

`-p, --prefix`prefix

バンドルされた AMI ファイル名プレフィクス。プレフィクス全体を指定します。例えば、プレフィクスが image.img である場合は、`-p image.img` ではなく `-p image` を使用します。

必須: `--prefix` または `--manifest` のどちらかを指定する必要があります。

`--clear`

指定されたバンドルを削除した後で空の場合は、Amazon S3 バケットを削除します。

必須: いいえ

`--retry`

すべての Amazon S3 エラーで、オペレーションあたり最大 5 回まで自動的に再試行します。

必須: いいえ

`-y, --yes`

すべてのプロンプトへの答えが [yes] であると自動的に想定します。

必須: いいえ

出力

Amazon EC2 は、削除プロセスのステージとステータスを示すステータスメッセージを表示します。

例

この例では、Amazon S3 からバンドルを削除します。

```
[ec2-user ~]$ ec2-delete-bundle -b DOC-EXAMPLE-BUCKET1 -a your_access_key_id -
s your_secret_access_key
Deleting files:
DOC-EXAMPLE-BUCKET1/
image.manifest.xml
DOC-EXAMPLE-BUCKET1/
```

```
image.part.00
DOC-EXAMPLE-BUCKET1/
image.part.01
DOC-EXAMPLE-BUCKET1/
image.part.02
DOC-EXAMPLE-BUCKET1/
image.part.03
DOC-EXAMPLE-BUCKET1/
image.part.04
DOC-EXAMPLE-BUCKET1/
image.part.05
DOC-EXAMPLE-BUCKET1/image.part.06
Continue? [y/n]
y
Deleted DOC-EXAMPLE-BUCKET1/image.manifest.xml
Deleted DOC-EXAMPLE-BUCKET1/image.part.00
Deleted DOC-EXAMPLE-BUCKET1/image.part.01
Deleted DOC-EXAMPLE-BUCKET1/image.part.02
Deleted DOC-EXAMPLE-BUCKET1/image.part.03
Deleted DOC-EXAMPLE-BUCKET1/image.part.04
Deleted DOC-EXAMPLE-BUCKET1/image.part.05
Deleted DOC-EXAMPLE-BUCKET1/image.part.06
ec2-delete-bundle complete.
```

ec2-download-bundle

説明

指定された instance store-backed Linux AMIs を Amazon S3 ストレージからダウンロードします。

Syntax

ec2-download-bundle -b *bucket* -a *access_key_id* -s *secret_access_key* -k *path* [--url *url*] [--region *region*] [--sigv *version*] [-m *file*] [-p *prefix*] [-d *directory*] [--retry]

オプション

`-b, --bucket` バケット

バンドルが存在する Amazon S3 バケットの名前。この後に、オプションで '/' 区切りのパスプレフィクスが続きます。

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: はい

`-s, --secret-key` secret_access_key

AWS シークレットアクセスキー。

必須: はい

`-k, --privatekey` パス

マニフェストの復号に使用されるプライベートキー。

必須: はい

`--url` url

Amazon S3 サービスの URL。

デフォルト: `https://s3.amazonaws.com/`

必須: いいえ

`--region` region

リクエスト署名で使用するリージョン。

デフォルト: `us-east-1`

必須: 署名バージョン 4 を使用する場合は必須

`--sigv` バージョン

リクエストに署名するときに使用する署名バージョン。

有効な値: 2 | 4

デフォルト: 4

必須: いいえ

`-m, --manifest` ファイル

マニフェストファイル名 (パスなし)。マニフェスト (`-m`) またはプレフィクス (`-p`) を指定することをお勧めします。

必須: いいえ

`-p, --prefix prefix`

バンドル済み AMI ファイルのファイル名プレフィクス。

デフォルト: `image`

必須: いいえ

`-d, --directory ディレクトリ`

ダウンロードしたバンドルが保存されているディレクトリ。ディレクトリが存在している必要があります。

デフォルト: 現在の作業ディレクトリ。

必須: いいえ

`--retry`

すべての Amazon S3 エラーで、オペレーションあたり最大 5 回まで自動的に再試行します。

必須: いいえ

出力

ダウンロードプロセスの多様な段階ステータスを示すメッセージが表示されます。

例

この例では、bundled ディレクトリを作成 (Linux mkdir コマンドを使用) し、Amazon S3 *DOC-EXAMPLE-BUCKET1* バケットからバンドルをダウンロードします。

```
[ec2-user ~]$ mkdir bundled
[ec2-user ~]$ ec2-download-bundle -b DOC-EXAMPLE-BUCKET1/bundles/bundle_name
 -m image.manifest.xml -a your_access_key_id -s your_secret_access_key -k pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -d mybundle
Downloading manifest image.manifest.xml from DOC-EXAMPLE-BUCKET1 to mybundle/
image.manifest.xml ...
Downloading part image.part.00 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.00 ...
Downloaded image.part.00 from DOC-EXAMPLE-BUCKET1
Downloading part image.part.01 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.01 ...
Downloaded image.part.01 from DOC-EXAMPLE-BUCKET1
```

```
Downloading part image.part.02 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.02 ...
Downloaded image.part.02 from DOC-EXAMPLE-BUCKET1
Downloading part image.part.03 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.03 ...
Downloaded image.part.03 from DOC-EXAMPLE-BUCKET1
Downloading part image.part.04 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.04 ...
Downloaded image.part.04 from DOC-EXAMPLE-BUCKET1
Downloading part image.part.05 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.05 ...
Downloaded image.part.05 from DOC-EXAMPLE-BUCKET1
Downloading part image.part.06 from DOC-EXAMPLE-BUCKET1/bundles/bundle_name to
 mybundle/image.part.06 ...
Downloaded image.part.06 from DOC-EXAMPLE-BUCKET1
```

ec2-migrate-manifest

説明

別のリージョンをサポートするように instance store-backed Linux AMI (証明書、カーネル、RAM ディスクなど) を変更します。

Syntax

```
ec2-migrate-manifest -c path -k path -m path {(-a access_key_id -
s secret_access_key --region region) | (--no-mapping)} [--ec2cert
ec2_cert_path] [--kernel kernel-id] [--ramdisk ramdisk_id]
```

オプション

`-c, --cert` パス

ユーザーの PEM エンコード RSA パブリックキー証明書ファイル。

必須: はい

`-k, --privatekey` パス

ユーザーの PEM エンコード RSA キーファイルへのパス。

必須: はい

`--manifest` パス

マニフェストファイルへのパス。

必須: はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: 自動マッピングを使用する場合は必須です。

`-s, --secret-key` secret_access_key

AWS シークレットアクセスキー。

必須: 自動マッピングを使用する場合は必須です。

`--region` region

マッピングファイル内で検索するリージョン。

必須: 自動マッピングを使用する場合は必須です。

`--no-mapping`

カーネルと RAM ディスクの自動マッピングを無効にします。

移行中、Amazon EC2 は、コピー先リージョン用に設計されたカーネルと RAM ディスクで、マニフェストファイルのカーネルと RAM ディスクを置き換えます。`--no-mapping` パラメータを指定しない場合、`ec2-migrate-bundle` は `DescribeRegions` および `DescribeImages` オペレーションを使用して、自動化されたマッピングを実行します。

必須: 自動マッピングに使用される `-a`、`-s`、および `--region` オプションを指定しない場合は必須です。

`--ec2cert` パス

イメージマニフェストの暗号化に使用される Amazon EC2 X.509 パブリックキー証明書へのパス。

`us-gov-west-1` および `cn-north-1` リージョンではデフォルト以外のパブリックキー証明書を使用し、その証明書へのパスは、このオプションで指定する必要があります。証明書へのパスは、AMI ツールのインストール方法によって異なります。Amazon Linux の場合、証明書の場所は `/opt/aws/amitools/ec2/etc/ec2/amitools/` です。「AMI ツールのセットアップ」の ZIP ファイルから AMI ツールをインストールした場合、証明書の場所は `$EC2_AMITOOL_HOME/etc/ec2/amitools/` です。

必須: us-gov-west-1 および cn-north-1 リージョンのみ。

--kernel kernel_id

選択するカーネルの ID。

> **Important**
>
> カーネルと RAM ディスクではなく PV-GRUB を使用することをお勧めします。詳細については、ユーザー提供カーネル を参照してください。

必須: いいえ

--ramdisk ramdisk_id

選択する RAM ディスクの ID。

> **Important**
>
> カーネルと RAM ディスクではなく PV-GRUB を使用することをお勧めします。詳細については、ユーザー提供カーネル を参照してください。

必須: いいえ

出力

バンドルプロセスのステージとステータスを記述するステータスメッセージ。

例

この例では、my-ami.manifest.xml マニフェストで指定された AMI を米国から EU にコピーします。

```
[ec2-user ~]$ ec2-migrate-manifest --manifest my-ami.manifest.xml
 --cert cert-HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem --privatekey pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBZQ55CLO.pem --region eu-west-1

Backing up manifest...
Successfully migrated my-ami.manifest.xml It is now suitable for use in eu-west-1.
```

ec2-unbundle

説明

instance store-backed Linux AMI からバンドルを再作成します。

Syntax

ec2-unbundle -k *path* -m *path* [-s *source_directory*] [-d *destination_directory*]

オプション

`-k, --privatekey` パス

PEM エンコードされる RSA キーファイルへのパス。

必須: はい

`-m, --manifest` パス

マニフェストファイルへのパス。

必須: はい

`-s, --source` source_directory

バンドル含むディレクトリ。

デフォルト: 現在のディレクトリ。

必須: いいえ

`-d, --destination` destination_directory

AMI をバンドル解除するディレクトリ。宛先ディレクトリが存在している必要があります。

デフォルト: 現在のディレクトリ。

必須: いいえ

例

この Linux および UNIX の例では、`image.manifest.xml` ファイルに指定された AMI をバンドル解除します。

```
[ec2-user ~]$ mkdir unbundled
$ ec2-unbundle -m mybundle/image.manifest.xml -k pk-
HKZYKTAIG2ECMXYIBH3HXV4ZBEXAMPLE.pem -s mybundle -d unbundled
$ ls -l unbundled
total 1025008
-rw-r--r-- 1 root root 1048578048 Aug 25 23:46 image.img
```

出力

バンドル解除プロセスの多様な段階ステータスを示すメッセージが表示されます。

ec2-upload-bundle

説明

instance store-backed Linux AMI のバンドルを Amazon S3 にアップロードし、アップロードされたオブジェクトで適切なアクセスコントロールリスト (ACL) を設定します。詳細については、instance store-backed Linux AMI を作成する を参照してください。

> **Note**
>
> instance store-backed Linux AMI の S3 バケットにオブジェクトをアップロードするには、バケットで ACL を有効にする必要があります。有効にしない場合、Amazon EC2 はアップロードするオブジェクトに ACL を設定できません。宛先のバケットが S3 オブジェクトの所有権のバケット所有者強制設定を使用している場合、ACL が無効になるため、この方法は使えません。詳細については、「S3 オブジェクトの所有権を使用したアップロードされたオブジェクトの所有権の管理」を参照してください。

Syntax

ec2-upload-bundle -b *bucket* -a *access_key_id* -s *secret_access_key* [-t *token*] -m *path* [--url *url*] [--region *region*] [--sigv *version*] [--acl *acl*] [-d *directory*] [--part *part*] [--retry] [--skipmanifest]

オプション

`-b, --bucket` バケット

バンドルを保存する Amazon S3 バケットの名前。その後にオプションで '/' 区切りのパスプレフィクスが続きます。バケットが存在しない場合、バケット名を使用できる場合はバケットが作

成されます。さらに、バケットが存在せず、AMI ツールのバージョンが 1.5.18 以降の場合、このコマンドはバケットの ACL を設定します。

必須：はい

`-a, --access-key` access_key_id

AWS アクセスキー ID。

必須: はい

`-s, --secret-key` secret_access_key

お客様の AWS シークレットアクセスキー。

必須: はい

`-t, --delegation-token` トークン

AWS リクエストに渡す委任トークン。詳細については、「一時的なセキュリティ認証情報の使用」を参照してください。

必須: 一時的なセキュリティ認証情報を使用している場合のみ。

デフォルト: `AWS_DELEGATION_TOKEN` 環境変数の値 (設定されている場合)。

`-m, --manifest` パス

マニフェストファイルへのパス。マニフェストファイルはバンドルプロセス中に作成され、バンドルを含むディレクトリにあります。

必須: はい

`--url` url

廃止済み。バケットの場所が (`--region` ではなく) EU に制約されない限り、代わりに `eu-west-1` オプションを使用します。`--location` フラグは、その特定の場所の制限を対象にする唯一の方法です。

Amazon S3 エンドポイントサービスの URL。

デフォルト: `https://s3.amazonaws.com/`

必須: いいえ

`--region` region

宛先の S3 バケットに対してリクエスト署名で使用するリージョン。

- バケットが存在せず、リージョンを指定しない場合、ツールは (us-east-1 で) 場所の制約のないバケットを作成します。
- バケットが存在せず、リージョンを指定した場合、ツールは指定したリージョンでバケットを作成します。
- バケットが存在し、リージョンを指定しない場合、ツールはバケットの場所を使用します。
- バケットが存在し、リージョンとして us-east-1 を指定した場合、ツールはエラーメッセージなしでバケットの実際の場所を使用し、一致する既存のファイルは上書きされます。
- バケットが存在し、バケットの実際の場所に一致しない (us-east-1 以外の) リージョンを指定した場合、ツールはエラーで終了します。

バケットが (EU ではなく) eu-west-1 の場所に制約されている場合は、代わりに --location フラグを使用します。--location フラグは、その特定の場所の制限を対象にする唯一の方法です。

デフォルト: us-east-1

必須: 署名バージョン 4 を使用する場合は必須

--sigv バージョン

リクエストに署名するときに使用する署名バージョン。

有効な値: 2 | 4

デフォルト: 4

必須: いいえ

--acl acl

バンドルされたイメージのアクセスコントロールリストのポリシー。

有効な値: public-read | aws-exec-read

デフォルト: aws-exec-read

必須: いいえ

-d, --directory ディレクトリ

バンドルされた AMI 部分を含むディレクトリ。

デフォルト: マニフェストファイルを含むディレクトリ (-m オプションを参照)。

必須: いいえ

--part パート

指定された部分とそれ以降のすべての部分のアップロードを開始します。例えば、--part 04。

必須: いいえ

--retry

すべての Amazon S3 エラーで、オペレーションあたり最大 5 回まで自動的に再試行します。

必須: いいえ

--skipmanifest

マニフェストをアップロードしません。

必須: いいえ

--location の場所

廃止済み。バケットの場所が (--region ではなく) EU に制約されない限り、代わりに eu-west-1 オプションを使用します。--location フラグは、その特定の場所の制限を対象にする唯一の方法です。

宛先 Amazon S3 バケットの場所の制約。バケットが存在し、バケットの実際の場所に一致しない場所を指定する場合、ツールはエラーで終了します。バケットが存在し、場所を指定しない場合、ツールはバケットの場所を使用します。バケットが存在しない場合に場所を指定すると、ツールは、指定した場所でバケットを作成します。バケットが存在せず、場所を指定しない場合、ツールは (us-east-1 で) 場所の制約のないバケットを作成します。

デフォルト: --region を指定した場合、場所はその指定したリージョンに設定されます。--region を指定しない場合、場所はデフォルトで us-east-1 になります。

必須: いいえ

出力

Amazon EC2 は、アップロードプロセスのステージとステータスを示すステータスメッセージを表示します。

例

この例では、image.manifest.xml マニフェストで指定されたバンドルをアップロードします。

```
[ec2-user ~]$ ec2-upload-bundle -b DOC-EXAMPLE-BUCKET1/bundles/bundle_name -m
 image.manifest.xml -a your_access_key_id -s your_secret_access_key
Creating bucket...
Uploading bundled image parts to the S3 bucket DOC-EXAMPLE-BUCKET1 ...
Uploaded image.part.00
Uploaded image.part.01
Uploaded image.part.02
Uploaded image.part.03
Uploaded image.part.04
Uploaded image.part.05
Uploaded image.part.06
Uploaded image.part.07
Uploaded image.part.08
Uploaded image.part.09
Uploaded image.part.10
Uploaded image.part.11
Uploaded image.part.12
Uploaded image.part.13
Uploaded image.part.14
Uploading manifest ...
Uploaded manifest.
Bundle upload completed.
```

AMI ツール用の一般的なオプション

AMI ツールのほとんどで、以下の任意のパラメータを使用できます。

`--help, -h`

ヘルプメッセージを表示します。

`--version`

バージョンと著作権表記を表示します。

`--manual`

手動のエントリを表示します。

`--batch`

インタラクティブなプロンプトを制約するバッチモードで実行します。

--debug

問題のトラブルシューティング時に役立つ可能性がある情報を表示します。

AMI を変更する

AMI の説明や共有プロパティなどの Amazon マシンイメージ (AMI) 属性の限定セットを変更できます。ただし、AMI コンテンツ (ボリュームバイナリデータ) は変更できません。AMI コンテンツを変更するには、新しい AMI を作成する必要があります。

> **Important**
>
> EBS-backed AMI のコンテンツ (ボリュームバイナリデータ) は、それらをバックアップするスナップショットが不変であるため変更できません。また、コンテンツが署名されているため、インスタンスストアバッキング (S3 バッキング) AMI のコンテンツ (ボリュームバイナリデータ) を変更することはできません。署名が一致しないとインスタンスの起動が失敗します。

変更できる AMI 属性については、「Amazon EC2 API リファレンス」の「ModifyImageAttribute」を参照してください。

以下のトピックでは、Amazon EC2 コンソールおよび AWS CLI を使用する方法と AMI の属性を変更する方法について説明します。

- AMI の公開
- AMI を特定の組織または組織単位と共有する
- 特定の AWS アカウントとの AMI の共有
- 有料サポートの使用
- AMI を設定する

AMI のコピー

AWS リージョン内またはリージョンをまたいで Amazon マシンイメージ (AMI) をコピーできます。Amazon EBS-backed AMI と instance store-backed AMI のいずれもコピーできます。暗号化されたスナップショットで EBS-backed AMI をコピーし、コピープロセス中に暗号化ステータスを変更することもできます。共有されている AMI をコピーすることができます。

ソース AMI をコピーすると、見た目は同じでもまったく別のターゲット AMI とも呼ばれる新しい AMI になります。ターゲット AMI にはそれ独自の AMI ID があります。ソース AMI は、ターゲット AMI に影響を及ぼさずに変更または登録解除できます。逆の場合も同様です。

EBS-backed AMI を使用すると、それぞれのバックアップするスナップショットは、同一だが区別されるターゲットスナップショットにコピーされます。AMI を新しいリージョンにコピーすると、スナップショットは完全な (増分ではない) コピーになります。暗号化されていないバッキングスナップショットを暗号化するか、新しい KMS キーに暗号化すると、スナップショットは完全な (増分ではない) コピーになります。以降の AMI のコピーオペレーションでは、バッキングスナップショットの増分コピーが作成されます。

コンテンツ

考慮事項

- AMI をコピーするためのアクセス許可 – IAM ポリシーを使用すると、AMI をコピーするためのアクセス許可をユーザーに付与したり、それを拒否したりできます。`CopyImage` アクション用に指定されたリソースレベルのアクセス権限は、新しい AMI にのみ適用されます。ソース AMI に対しては、リソースレベルのアクセス許可を付与できません。
- 起動許可と Amazon S3 バケット許可 – AWS は、起動許可と Amazon S3 バケット許可をソース AMI から新しい AMI にコピーしません。コピー操作が完了すると、起動許可と Simple Storage Service (Amazon S3) バケット許可を新しい AMI に適用できます。
- タグ – コピーできるのは、ソース AMI にアタッチされているユーザー定義の AMI タグだけです。システムタグ (`aws:` プレフィックスが付いている) や、他の AWS アカウント がアタッチしたユーザー定義タグはコピーされません。AMI をコピーするときに、ターゲット AMI とそのバックアップするスナップショットに新しいタグをアタッチできます。

- 共有された AWS Marketplace AMI - `CopyImage` アクションは、別のアカウントから共有された AWS Marketplace AMI のコピーには対応していません。代わりに、別のアカウントに AWS Marketplace AMI をコピーする場合は、次の操作を行う必要があります。AWS Marketplace AMI を他のアカウントと共有し、そのアカウントで AWS Marketplace AMI を使用して EC2 インスタンスを起動します。これで、`CreateImage` アクションを使用してインスタンスから AMI を作成できます。新しい AMI にはすべての AWS Marketplace コードが保持されます。このプロセスは AWS Marketplace AMI から直接または間接的に派生したすべての AMI にも適用されることに注意してください。インスタンスから AMI を作成する方法の詳細については、「」と「Amazon EBS-backed Linux AMI を作成する」を参照してください。

コスト

AMI のコピーには課金されません。ただし、標準のストレージ料金とデータ転送料金が適用されます。EBS-backed AMI をコピーする場合は、追加の EBS スナップショットのストレージに対して料金が発生します。

IAM アクセス許可

EBS-backed AMI または instance store-backed AMI をコピーするには、次の IAM アクセス許可が必要です。

- `ec2:CopyImage` – AMI をコピーするアクセス許可。EBS-backed AMI の場合、AMI のバックアップするスナップショットをコピーするアクセス許可も付与します。
- `ec2:CreateTags` – ターゲット AMI にタグ付けするアクセス許可。EBS-backed AMI の場合、ターゲット AMI のバックアップするスナップショットにタグ付けするアクセス許可も付与します。

instance store-backed AMI をコピーする場合は、追加で次の IAM アクセス許可が必要です。

- `s3:CreateBucket` – 新しい AMI のターゲットリージョンに S3 バケットを作成するアクセス許可
- `s3:GetBucketAcl` – ソースバケットの ACL アクセス許可を読み取るアクセス許可
- `s3:ListAllMyBuckets` – ターゲットリージョンで AMI の既存の S3 バケットを検出するアクセス許可
- `s3:GetObject` – ソースバケットのオブジェクトを読み取るアクセス許可
- `s3:PutObject` – ターゲットバケットにオブジェクトを書き込むアクセス許可

- `s3:PutObjectAcl` – ターゲットバケットの新しいオブジェクトのアクセス許可を書き込むアクセス許可

EBS-backed AMI をコピーし、ターゲット AMI とスナップショットにタグ付けするための IAM ポリシーの例

次のポリシー例では、EBS-backed AMI をコピーし、ターゲット AMI とそのバックアップするスナップショットにタグを付けるアクセス許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "PermissionToCopyAllImages",
        "Effect": "Allow",
        "Action": [
            "ec2:CopyImage",
            "ec2:CreateTags"
        ],
        "Resource": "arn:aws:ec2:*::image/*"
    }]
}
```

EBS-backed AMI をコピーし、新しいスナップショットのタグ付けを拒否する IAM ポリシーの例

`ec2:CopySnapshot` アクセス許可は、`ec2:CopyImage` アクセス許可を取得すると自動的に付与されます。これには、ターゲット AMI の新しいバックアップするスナップショットにタグ付けするアクセス許可が含まれます。新しいバックアップするスナップショットにタグ付けするアクセス許可は、明示的に拒否できます。

次のポリシー例では、EBS-backed AMI をコピーするアクセス許可を付与しますが、ターゲット AMI の新しいバックアップするスナップショットのタグ付けは拒否します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Effect": "Allow",
            "Action": [
                "ec2:CopyImage",
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*::image/*"
```

```
        },
        {
            "Effect": "Deny",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:::snapshot/*"
        }
    ]
}
```

instance store-backed AMI をコピーし、ターゲット AMI にタグ付けするための IAM ポリシーの例

次のポリシー例では、指定されたソースバケットの instance store-backed AMI を指定されたリージョンにコピーし、ターゲット AMI にタグ付けするアクセス許可を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "PermissionToCopyAllImages",
            "Effect": "Allow",
            "Action": [
                "ec2:CopyImage",
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*::image/*"
        },
        {
            "Effect": "Allow",
            "Action": "s3:ListAllMyBuckets",
            "Resource": [
                "arn:aws:s3:::*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "s3:GetObject",
            "Resource": [
                "arn:aws:s3:::ami-source-bucket/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:CreateBucket",
```

```
                "s3:GetBucketAcl",
                "s3:PutObjectAcl",
                "s3:PutObject"
            ],
            "Resource": [
                "arn:aws:s3:::amis-for-account-in-region-hash"
            ]
        }
    ]
}
```

AMI ソースバケットの Amazon リソースネーム (ARN) を検索するには、https://console.aws.amazon.com/ec2/ で Amazon EC2 コンソールを開き、ナビゲーションペインで [AMI] を選択し、[Source] 列でバケット名を特定します。

> **Note**
>
> `s3:CreateBucket` アクセス許可は、初めて instance store-backed AMI を個々のリージョンにコピーするときにのみ必要です。その後、リージョンですでに作成された Amazon S3 バケットは、そのリージョンにコピーする将来のすべての AMIs に保存されます。

AMI のコピー

AWS Management Console、AWS Command Line Interfaceまたは SDK、または Amazon EC2 API を使用して AMI をコピーするできます。これはらいずれも `CopyImage` アクションをサポートしています。

前提条件

コピーする AMI を作成または取得します。Amazon EC2 コンソールを使用すると、AWS が提供するさまざまな AMI を検索できます。詳細については、「Amazon EBS-backed Linux AMI を作成する」および「AMI の検索」「」を参照してください。

Console

AMI をコピーするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. コンソールのナビゲーションバーから、AMI を含むリージョンを選択します。

3. ナビゲーションペインで、[AMI] を選択し、リージョンで利用できる AMI のリストを表示します。
4. コピーする AMI が表示されない場合は、別のフィルターを選択します。AMI は、[自己所有]、[プライベートイメージ]、[パブリックイメージ]、および [無効化されたイメージ] でフィルタリングできます。
5. コピーする AMI を選択して、[アクション]、[AMI のコピー] の順に選択します。
6. [Copy AMI] (AMI のコピー) ページで、次の情報を指定します。

   a. [AMI copy name] (AMI コピー名) : 新しい AMI の名前。この名前にはオペレーティングシステム情報を含めることができます (この情報は AMI の詳細としては表示されません)。

   b. [AMI copy description] (AMI コピーの説明) : デフォルトでは、オリジナルからコピーを見分けられるように、ソース AMI に関する情報が説明に含まれています。この説明は必要に応じて変更できます。

   c. [Destination Region] (送信先リージョン) : AMI をコピーするリージョン。詳細については、「リージョン間のコピー」を参照してください。

   d. [Copy tags] (タグのコピー): このチェックボックスを選択すると、AMI のコピー時にユーザー定義の AMI タグが含まれます。システムタグ (aws: プレフィックスが付いている) や、他の AWS アカウント がアタッチしたユーザー定義タグはコピーされません。

   e. (EBS-backed AMI のみ) [AMI コピーの EBS スナップショットを暗号化]: ターゲットスナップショットを暗号化するか、別のキーを使用して再暗号化する場合は、このチェックボックスを選択します。デフォルトで暗号化を有効にしている場合は、[AMI コピーの EBS スナップショットを暗号化] チェックボックスが選択され、クリアできません。詳細については、「暗号化とコピー」を参照してください。

   f. (EBS-backed AMI のみ) [KMS キー]: ターゲットスナップショットを暗号化するための KMS キー。

   g. [タグ]: 新しい AMI と新しいスナップショットに同じタグを付けることも、異なるタグでタグ付けすることもできます。

      - 新しい AMI と新しいスナップショットに同じタグを付けるには、[イメージとスナップショットに対し一緒にタグを付けます] を選択します。新しい AMI と作成されるすべてのスナップショットには、同じタグが適用されます。
      - 新しい AMI と新しいスナップショットに異なるタグを付けるには、[イメージとスナップショットに対し個別にタグを付けます] を選択します。新しい AMI と作成され

るスナップショットには、異なるタグが適用されます。ただし、作成されるすべての新しいスナップショットには同じタグが付けられることに注意してください。新しいそれぞれのスナップショットに異なるタグを付けることはできません。

(オプション) タグを追加するには、[Add tag] を選択し、そのタグのキーと値を入力します。各タグについて、これを繰り返します。

h. AMI をコピーする準備ができたら、[AMI のコピー] を選択します。

新しい AMI の初期ステータスは `Pending` です。ステータスが `Available` になると、AMI のコピー操作は完了です。

AWS CLI

AWS CLI を使用して AMI をコピーするには

AMI は、copy-image コマンドを使用してコピーできます。コピー元リージョンおよび送信先リージョンの両方を指定する必要があります。コピー元のリージョンは、`--source-region` パラメータを使用して指定します。`--region` パラメータまたは環境変数を使用して送信先リージョンを指定できます。詳細については、「AWS コマンドラインインターフェイスの設定」を参照してください。

(EBS-backed AMI のみ) コピー時にターゲットスナップショットを暗号化する場合は、`--encrypted` および `--kms-key-id` の追加のパラメータを指定する必要があります。

コマンドの例については、「AWS CLI コマンドリファレンス」の「copy-image」の「例」を参照してください。

PowerShell

Tools for Windows PowerShell を使用して AMI をコピーするには

AMI は、Copy-EC2Image コマンドを使用してコピーできます。コピー元リージョンおよび送信先リージョンの両方を指定する必要があります。コピー元のリージョンは、`-SourceRegion` パラメータを使用して指定します。`-Region` パラメータまたは `Set-AWSDefaultRegion` コマンドを使用して送信先リージョンを指定できます。詳細については、「AWS リージョンの指定」を参照してください。

(EBS-backed AMI のみ) コピー時にターゲットスナップショットを暗号化する場合は、`-Encrypted` および `-KmsKeyId` の追加のパラメータを指定する必要があります。

保留中の AMI コピーオペレーションの中止

保留中の AMI のコピーは、AWS Management Console またはコマンドラインを使用して停止することができます。

Console

コンソールを使用して AMI のコピー操作を中止するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーのリージョンセレクターから対象のリージョンを選択します。
3. ナビゲーションペインで [AMIs] を選択します。
4. コピーを中止する AMI を選択し、[アクション]、[AMI を登録解除] を選択します。
5. 確認を求めるメッセージが表示されたら、[Deregister AMI] (AMI の登録解除) を選択します。

Command line

コマンドラインを使用して AMI コピー操作を中止するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- deregister-image (AWS CLI)
- Unregister-EC2Image (AWS Tools for Windows PowerShell)

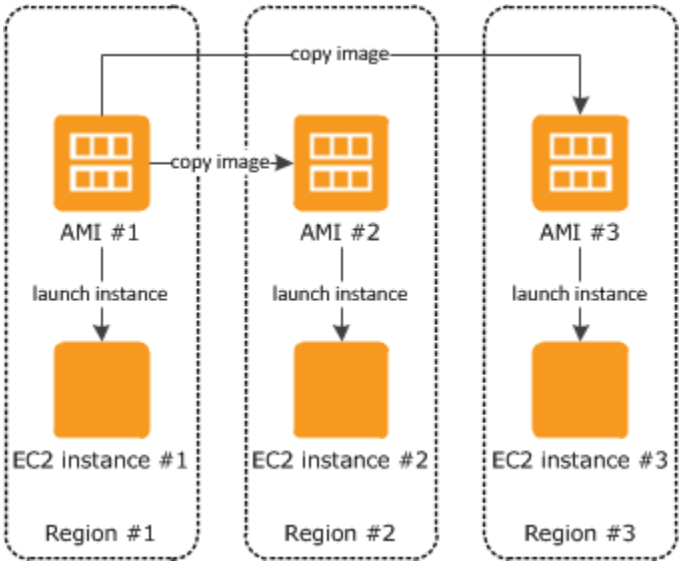
リージョン間のコピー

地理的に分散したリージョンに AMI をコピーすると、次のような利点があります。

- 一貫性のあるグローバルなデプロイメント: 1 つのリージョンから別のリージョンに AMI をコピーすることで、一貫性のあるインスタンスを同じ AMI から別のリージョンに起動できます。
- スケーラビリティ: ユーザーの場所にかかわらず、ユーザーのニーズに合ったグローバルアプリケーションをより簡単に設計できます。
- パフォーマンス: アプリケーションを配布したり、アプリケーションの重要なコンポーネントをユーザーの近くに配置したりすることでパフォーマンスを向上できます。また、インスタンスの種類やその他の AWS サービスなど、リージョン固有の機能を活用することもできます。

- 高可用性: アプリケーションを設計し、AWS リージョン全体にわたってデプロイして可用性を高めることができます。

次の図は、ソース AMI と異なるリージョンにある 2 つのコピーされた AMI、およびそこから起動される EC2 インスタンスの関係を示します。AMI からインスタンスを起動すると、AMI が存在する同じリージョンに存在します。ソース AMI を変更し、それらの変更をターゲットリージョンの AMIs に反映させる場合、ソース AMI をターゲットリージョンに再度コピーする必要があります。



instance store-backed AMI を最初にリージョンにコピーするときに、そのリージョンにコピーされた AMIs に Amazon S3 バケットを作成します。そのリージョンにコピーするすべての Instance Store-Backed AMIs が、このバケットに保存されます。バケット名の形式は次のとおりです: amis-for-#####-in-#####-#### 例: `amis-for-123456789012-in-us-east-2-yhjmxvp6`。

前提条件

AMI をコピーする前に、ソース AMI のすべてのコンテンツが、異なるリージョンでの実行をサポートするように更新されていることを確認する必要があります。例えば、データベース接続文字列や同様のアプリケーション設定データが、適切なリソースを指すように更新する必要があります。それ以外の場合、対象のリージョンの新しい AMI から起動したインスタンスは元のリージョンのリソースをまだ使用している可能性があり、それによりパフォーマンスとコストに影響が及ぶことがあります。

制限事項

- コピー先のリージョンには、AMI の同時コピーが 100 個までという制限があります。

- 準仮想化 (PV) AMI がサポートされていないリージョンに、PV AMI をコピーすることはできません。詳細については、「Linux AMI 仮想化タイプ」を参照してください。

アカウント間のコピー

別の AWS アカウントと AMI を共有できます。AMI の共有は AMI の所有権には影響しません。所有しているアカウントには、リージョンのストレージ料金が適用されます。詳細については、特定の AWS アカウントとの AMI の共有 を参照してください。

自分のアカウントと共有された AMI をコピーした場合、アカウントのコピー先の AMI の所有者は自分になります。コピー元の AMI の所有者には、Amazon EBS または Amazon S3 の標準転送料金が課金され、コピー先の AMI の所有者には、コピー先リージョンのストレージ料金が課金されます。

リソースのアクセス許可

AMI を別のアカウントから共有した場合、この AMI をコピーするには、ソース AMI の所有者がこの AMI をバックアップするストレージの読み取り許可を付与する必要があります。ストレージは、関連 EBS スナップショット (Amazon EBS-backed AMI の場合) か、関連 S3 バケット (instance store-backed AMI の場合) のどちらかです。共有 AMI に暗号化されたスナップショットがある場合、所有者はキーも共有する必要があります。リソースのアクセス許可の付与についての詳細は、EBS スナップショットの場合は、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットの共有」を、S3 バケットの場合は、「Amazon Simple Storage Service ユーザーガイド」の「Amazon S3 での Identity and Access Management」を参照してください。

> **Note**
>
> AMI をタグ付きでコピーするには、ソース AMI の起動許可が必要です。

暗号化とコピー

次の表は、各種 AMI コピーのシナリオにおける暗号化サポートを示します。暗号化されたスナップショットを生成するために暗号化されていないスナップショットをコピーすることはできますが、暗号化されていないスナップショットを生成するために暗号化されたスナップショットをコピーすることはできません。

シナリオ	説明	サポート対象
1	非暗号化から非暗号化	はい
2	暗号化から暗号化	はい
3	非暗号化から暗号化	はい
4	暗号化から非暗号化	いいえ

> **Note**
>
> CopyImage アクション中の暗号化は Amazon EBS-backed AMIs にのみ適用されます。Instance Store-Backed AMI はスナップショットに依存しないため、コピーを使用して暗号化ステータスを変更することはできません。

デフォルト (暗号化パラメータを指定しない) では、AMI をバックアップするスナップショットは元の暗号化ステータスとともにコピーされます。暗号化されていないスナップショットにバックアップされた AMI をコピーすると、やはり暗号化されていない同一のターゲットスナップショットになります。暗号化されたスナップショットにバックアップされている AMI をコピーすると、コピー先でもスナップショットが同じ AWS KMS キーで暗号化されます。複数のスナップショットにバックアップされた AMI をコピーした場合、デフォルトでは、元の暗号化ステータスが各ターゲットスナップショットで維持されます。

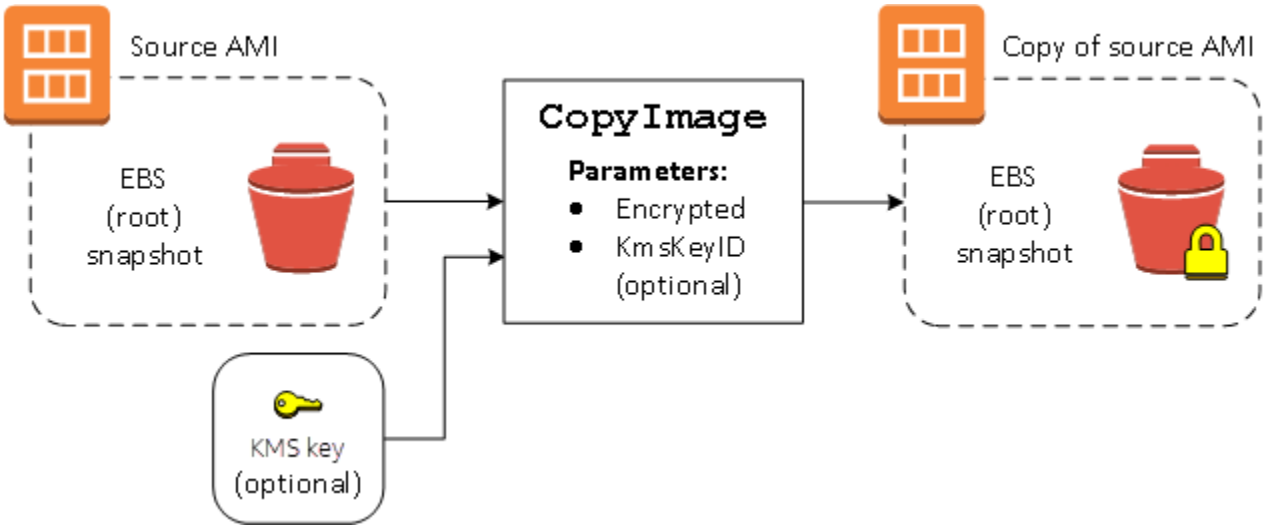
AMI をコピー中に暗号化パラメータを指定した場合、バックアップスナップショットを暗号化または再暗号化できます。以下の例は、ターゲット AMI の暗号化状態を変更するために CopyImage アクションに暗号化パラメータを提供する、デフォルトではないケースを示しています。

暗号化されていないソース AMI の暗号化されたターゲット AMI へのコピー

このシナリオでは、暗号化されていないルートスナップショットでバックアップされた AMI は、暗号化されたルートスナップショットを持つ AMI にコピーされます。CopyImage アクションは、カスタマーマネージド型キーなど、2 つの暗号化パラメータで呼び出されます。その結果、ルートスナップショットの暗号化ステータスが変更され、ターゲット AMI はソーススナップショットと同じデータを含むルートスナップショットにバックアップされますが、指定されたキーを使用して暗号化

されます。両方の AMI でスナップショットのストレージコストと、いずれかの AMI から起動するインスタンスの料金が発生します。

> **Note**
>
> デフォルトで暗号化を有効にすると、AMI 内のすべてのスナップショットで `Encrypted` パラメータを `true` に設定したのと同じ効果があります。



`Encrypted` パラメータを設定すると、このインスタンスの単一のスナップショットが暗号化されます。`KmsKeyId` パラメータを指定しない場合は、デフォルトのカスタマーマネージド型キーを使用して、スナップショットのコピーが暗号化されます。

暗号化されたスナップショットを持つ AMIs のコピーの詳細については、「EBS-backed AMI での暗号化の利用」を参照してください。

S3 を使用して AMI を保存および復元する

Amazon マシンイメージ (AMI) を Amazon S3 バケットに保存し、AMI を別の S3 バケットにコピーして、S3 バケットから復元できます。S3 バケットを使用して AMI を保存および復元することで、AMI をある AWS パーティションから別のパーティション (例えば、主要な商用パーティションから AWS GovCloud (US) パーティション) にコピーできます。AMI を S3 バケットに保存することで、AMI のアーカイブコピーを作成することもできます。

S3 を使用した AMI の保存および復元のサポート対象の API は、`CreateStoreImageTask`、`DescribeStoreImageTasks`、および `CreateRestoreImageTask` です。

CopyImage は、AWS パーティション内の AMI のコピーの際に使用することが推奨される API です。ただし、CopyImage は、AMI を別のパーティションにコピーできません。

AWS パーティションの詳細については、「IAM ユーザーガイド」の「Amazon リソースネーム (ARN)」ページの「#######」を参照してください。

> **Warning**
>
> AWS パーティションまたは AWS リージョン間でデータを移動する場合、適用されるすべての法令およびビジネス要件 (適用される政府の規制およびデータ所在地に関する要件を含みますが、これらに限られません) を確実に遵守してください。

トピック

ユースケース

保存 API と復元 API を使用して、次の操作を実行します。

- ある AWS パーティションから別の AWS パーティションに AMI をコピーする
- AMI のアーカイブコピーを作成する

ある AWS パーティションから別の AWS パーティションに AMI をコピーする

S3 バケットを使用して AMI を保存および復元することで、ある AWS パーティションから別のパーティションに、またはある AWS リージョンから別のリージョンに AMI をコピーできます。次の例では、主要な商用パーティションから AWS GovCloud (US) パーティションに、具体的には us-east-2 リージョンから us-gov-east-1 リージョンに AMI をコピーします。

あるパーティションから別のパーティションに AMI をコピーするには、次の手順に従います。

- `CreateStoreImageTask` を使用して、現在のリージョンの S3 バケットに AMI を保存します。この例では、S3 バケットは `us-east-2` にあります。コマンドの例については、S3 バケットに AMI を保存する をご参照ください。
- `DescribeStoreImageTasks` を使用して、保存タスクの進行状況をモニタリングします。タスクが完了すると、オブジェクトが S3 バケットに表示されます。コマンドの例については、AMI 保存タスクの進行状況を記述する をご参照ください。
- 任意の手順を使用して、保存された AMI オブジェクトをターゲットパーティションの S3 バケットにコピーします。この例では、S3 バケットは `us-gov-east-1` にあります。

> **Note**
>
> パーティションごとに異なる AWS 認証情報が必要なため、S3 オブジェクトをあるパーティションから別のパーティションに直接コピーすることはできません。パーティション間で S3 オブジェクトをコピーするプロセスは、このドキュメントの対象外です。例として、次のコピープロセスを提供していますが、お客様のセキュリティ要件を満たすコピープロセスを使用する必要があります。
>
> - パーティション間で 1 つの AMI をコピーするためのコピープロセスはシンプルです。ソースバケットから中間ホスト (EC2 インスタンスやラップトップなど) にオブジェクトをダウンロードし、中間ホストからターゲットバケットにオブジェクトをアップロードするだけです。プロセスの各段階で、パーティションの AWS 認証情報を使用します。
> - より持続的な使用のために、コピーを管理するアプリケーションの開発をご検討ください。S3 マルチパートダウンロードとアップロードを使用することも考慮に値します。

- `CreateRestoreImageTask` を使用して、ターゲットパーティションの S3 バケットから AMI を復元します。この例では、S3 バケットは `us-gov-east-1` にあります。コマンドの例については、S3 バケットから AMI を復元する をご参照ください。
- その状態が使用可能になるタイミングを確認するために、AMI を記述して復元タスクの進行状況をモニタリングします。また、スナップショットを記述することで、復元される AMI を構成するスナップショットの進行状況 (%) をモニタリングすることもできます。

AMI のアーカイブコピーを作成する

AMI を S3 バケットに保存することで、AMI のアーカイブコピーを作成できます。コマンドの例については、S3 バケットに AMI を保存する をご参照ください。

AMI は S3 内の 1 つのオブジェクトにパックされ、すべての AMI メタデータ (共有情報を除く) は、保存された AMI の一部として保持されます。AMI データは、ストレージプロセスの一環として圧縮されます。簡単に圧縮できるデータを含む AMI は、S3 で小さめのオブジェクトとなります。コストを削減するために、より安価な S3 ストレージ階層を使用できます。詳細については、Amazon S3 ストレージクラスおよび Amazon S3 の料金をご参照ください。

AMI ストアと復元 API の仕組み

S3 を使用して AMI を保存および復元するには、次の API を使用します。

- `CreateStoreImageTask` – AMI を S3 バケットに保存する
- `DescribeStoreImageTasks` – AMI 保存タスクの進行状況を示す
- `CreateRestoreImageTask` – S3 バケットから AMI を復元する

API の仕組み

- CreateStoreImageTask
- DescribeStoreImageTasks
- CreateRestoreImageTask

CreateStoreImageTask

CreateStoreImageTask API は、AMI を S3 バケット内の単一のオブジェクトとして保存します。

API は、AMI とそのスナップショットからすべてのデータを読み取るタスクを作成し、S3 マルチパートアップロードを使用して S3 オブジェクトにデータを保存します。API は、リージョン固有でない AMI メタデータの大部分を含む AMI のすべてのコンポーネント、および AMI に含まれるすべての EBS スナップショットを取得し、S3 内の単一のオブジェクトにパックします。データは、S3 で使用される領域の量を削減するために、アップロードプロセスの一環として圧縮されるので、S3 内のオブジェクトは AMI 内のスナップショットのサイズの合計よりも小さくなる可能性があります。

この API を呼び出すアカウントに AMI タグとスナップショットタグが表示されている場合、それらは保持されます。

S3 のオブジェクトは AMI と同じ ID を持っていますが、.bin 拡張子が付いています。AMI 名、AMI の説明、AMI の登録日、AMI の所有者アカウント、および保存オペレーションのタイムスタンプといったデータも S3 オブジェクトに S3 メタデータタグとして保存されます。

タスクを完了するのにかかる時間は、AMI のサイズによって異なります。また、タスクがキューに入れられているため、進行中の他のタスクの数にも依存します。DescribeStoreImageTasks API を呼び出すことで、タスクの進行状況を追跡できます。

進行中のすべての AMI のサイズの合計は、アカウントごとに 600 GB の EBS スナップショットデータに制限されます。進行中のタスクが制限未満になるまで、それ以降のタスクの作成は拒否されます。例えば、100 GB のスナップショットデータを持つ AMI と 200 GB のスナップショットデータを持つ別の AMI が現在保存されている場合、別のリクエストが受け入れられます。これは、進行中の合計が 300 GB で、制限未満であるためです。ただし、800 GB のスナップショットデータを持つ 1 つの AMI が現在保存されようとしている場合は、タスクが完了するまでそれ以降のタスクは拒否されます。

DescribeStoreImageTasks

DescribeStoreImageTasks API は、AMI 保存タスクの進行状況を示します。指定した AMI のタスクを記述できます。AMI を指定しない場合、過去 31 日間に処理されたすべての保存イメージタスクのページ分割されたリストが表示されます。

各 AMI タスクについて、応答では、タスクが `InProgress`、`Completed`、または `Failed` のいずれであるかが示されます。`InProgress` のタスクの場合、応答では、進行状況がパーセンテージで示されます。

タスクは時系列の逆順にリストされます。

現時点では、前月のタスクのみを表示できます。

CreateRestoreImageTask

CreateRestoreImageTask API は、CreateStoreImageTask リクエストを使用して以前に作成された S3 オブジェクトから AMI を復元するタスクを開始します。

復元タスクは、保存タスクが実行されたリージョンと同じリージョンまたは別のリージョンで実行できます。

AMI オブジェクトの復元ソースである S3 バケットは、復元タスクがリクエストされたリージョンと同じリージョンに存在する必要があります。AMI はこのリージョンで復元されます。

AMI は、保存された AMI の値に対応する名前、説明、ブロックデバイスマッピングなどのメタデータとともに復元されます。このアカウントの名前は、リージョン内の AMI に対して一意である必要があります。名前を指定しない場合、新しい AMI は元の AMI と同じ名前になります。AMI は、復元プロセス時に生成される新しい AMI ID を取得します。

AMI 復元タスクを完了するのにかかる時間は、AMI のサイズによって異なります。また、タスクがキューに入れられているため、進行中の他のタスクの数にも依存します。タスクの進行状況は、AMI (describe-images) またはその EBS スナップショット (describe-snapshots) を記述することで確認できます。タスクが失敗すると、AMI とスナップショットは失敗の状態に移行されます。

進行中のすべての AMI のサイズの合計は、アカウントあたり 300 GB (復元後のサイズに基づく) の EBS スナップショットデータに制限されます。進行中のタスクが制限未満になるまで、それ以降のタスクの作成は拒否されます。

制限事項

- AMI を保存するには、AWS アカウント が AMI とそのスナップショットを所有しているか、AMI とそのスナップショットをアカウントと直接共有する必要があります。公開されているだけの AMI は保存できません。
- これらの API を使用して保存できるのは、EBS-backed AMI だけです。
- 準仮想化 (PV) AMI はサポートされていません。
- 保存可能な AMI の上限サイズ (圧縮前) は、5,000 GB です。
- 保存イメージリクエストのクォータ: 進行中の 600 GB の保存作業 (スナップショットデータ)。
- 復元イメージリクエストのクォータ: 進行中の 300 GB の復元作業 (スナップショットデータ)。
- 保存タスク中は、スナップショットを削除してはならず、保存を実行する IAM プリンシパルにはスナップショットへのアクセス権が必要です。それ以外の場合は、保存プロセスが失敗します。
- 同じ S3 バケットに AMI の複数のコピーを作成することはできません。
- S3 バケットに保存されている AMI は、元の AMI ID では復元できません。AMI エイリアシングを使用すると、これを軽減できます。
- 現在、保存 API と復元 API は、AWS Command Line Interface、AWS SDK、および Amazon EC2 API を使用する場合にのみサポートされます。Amazon EC2 コンソールを使用して AMI を保存および復元することはできません。

コスト

S3 を使用して AMI を保存および復元する場合、保存 API と復元 API で使用されるサービス、およびデータ転送について料金が発生します。API は、S3 と EBS Direct API を使用します (これらの API がスナップショットデータにアクセスするために内部的に使用されます)。詳細については、Amazon S3 の料金および Amazon EBS の料金をご参照ください。

AMI のセキュリティ保護

保存 API と復元 API を使用するには、S3 バケットと AMI が同じリージョンに存在する必要があります。AMI のコンテンツを保護するために十分なセキュリティをもって S3 バケットが確実に設定されていること、および AMI オブジェクトがバケット内に残っている限り、セキュリティが確実に維持されるようにすることが重要です。これを実行できない場合は、これらの API の使用はお勧めしません。S3 バケットへのパブリックアクセスが許可されていないことを確認します。必須ではありませんが、AMI を保存する S3 バケットのためにサーバー側の暗号化を有効にすることをお勧めします。

S3 バケットに適切なセキュリティを設定する方法については、次のセキュリティトピックをご参照ください。

- Amazon S3 ストレージへのパブリックアクセスのブロック
- Amazon S3 バケット向けのサーバー側のデフォルトの暗号化動作の設定
- AWS Config ルールの s3-bucket-ssl-requests-only に準拠するには、どの S3 バケットポリシーを使用すればよいですか?
- Amazon S3 サーバーアクセスログ記録の有効化

AMI スナップショットが S3 オブジェクトにコピーされると、データは TLS 接続を介してコピーされます。暗号化されたスナップショットを使用して AMI を保存できますが、スナップショットは保存プロセスの一部として復号されます。

S3 を使用して AMI を保存および復元するためのアクセス権限

IAM プリンシパルが Amazon S3 を使用して AMI を保存または復元する場合は、必要な許可を付与する必要があります。

次のポリシーの例には、IAM プリンシパルが保存タスクと復元タスクを実行できるようにするために必要なすべてのアクションが含まれています。

特定のリソースへのアクセス権のみをプリンシパルに付与する IAM ポリシーを作成することもできます。ポリシーの例については、「IAM ユーザーガイド」の「AWS リソースのアクセス管理」を参照してください。

> **Note**
>
> AMI を構成するスナップショットが暗号化されている場合、またはアカウントの暗号化がデフォルトで有効になっている場合は、IAM プリンシパルに KMS キーを使用するための許可が付与されている必要があります。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:DeleteObject",
                "s3:GetObject",
                "s3:ListBucket",
                "s3:PutObject",
                "s3:PutObjectTagging",
                "s3:AbortMultipartUpload",
                "ebs:CompleteSnapshot",
                "ebs:GetSnapshotBlock",
                "ebs:ListChangedBlocks",
                "ebs:ListSnapshotBlocks",
                "ebs:PutSnapshotBlock",
                "ebs:StartSnapshot",
                "ec2:CreateStoreImageTask",
                "ec2:DescribeStoreImageTasks",
                "ec2:CreateRestoreImageTask",
                "ec2:GetEbsEncryptionByDefault",
                "ec2:DescribeTags",
                "ec2:CreateTags"
            ],
            "Resource": "*"
        }
    ]
}
```

AMI 保存 API および復元 API を使用する

トピック

S3 バケットに AMI を保存する

AMI (AWS CLI) を保存するには

create-store-image-task コマンドを使用します。AMI の ID と、AMI を保存する S3 バケットの名前を指定します。

```
aws ec2 create-store-image-task \
    --image-id ami-1234567890abcdef0 \
    --bucket myamibucket
```

正常な出力

```
{
  "ObjectKey": "ami-1234567890abcdef0.bin"
}
```

AMI 保存タスクの進行状況を記述する

AMI 保存タスクの進行状況を記述するには (AWS CLI)

describe-store-image-tasks コマンドを使用します。

```
aws ec2 describe-store-image-tasks
```

正常な出力

```
{
  "AmiId": "ami-1234567890abcdef0",
  "Bucket": "myamibucket",
```

```
  "ProgressPercentage": 17,
  "S3ObjectKey": "ami-1234567890abcdef0.bin",
  "StoreTaskState": "InProgress",
  "StoreTaskFailureReason": null,
  "TaskStartTime": "2021-01-01T01:01:01.001Z"
}
```

S3 バケットから AMI を復元する

AMI (AWS CLI) を復元するには

create-restore-image-task コマンドを使用します。describe-store-image-tasks 出力からの S3ObjectKey および Bucket の値を使用して、AMI のオブジェクトキーと AMI のコピー先の S3 バケットの名前を指定します。復元された AMI の名前も指定します。このアカウントの名前は、リージョン内の AMI に対して一意である必要があります。

> **Note**
>
> 復元された AMI は、新しい AMI ID を取得します。

```
aws ec2 create-restore-image-task \
    --object-key ami-1234567890abcdef0.bin \
    --bucket myamibucket \
    --name "New AMI Name"
```

正常な出力

```
{
   "ImageId": "ami-0eab20fe36f83e1a8"
}
```

S3 のファイルパスを使用する

AMI を保存および復元するときは、次の方法でファイルパスを使用できます。

- AMI を S3 に保存する場合、ファイルパスをバケット名に追加できます。システム内部では、バケット名からパスが分離され、AMI を保存するために生成されたオブジェクトキーにそのパスが追加されます。完全なオブジェクトパスは、API 呼び出しからのレスポンスに表示されます。

- AMI を復元する場合、オブジェクトキーパラメータを使用できるので、オブジェクトキー値の先頭にパスを追加できます。

AWS CLI および SDK を使用するときは、ファイルパスを使用できます。

例: AMI の保存と復元にファイルパスを使用する (AWS CLI)

次の例では、まず、バケット名にファイルパスを追加して、AMI を S3 に保存します。次に、オブジェクトキーパラメータの先頭にファイルパスを追加して、S3 から AMI を復元します。

1. AMI を保存します。--bucket には、次のようにバケット名の後にファイルパスを指定します。

```
aws ec2 create-store-image-task \
    --image-id ami-1234567890abcdef0 \
    --bucket myamibucket/path1/path2
```

正常な出力

```
{
  "ObjectKey": "path1/path2/ami-1234567890abcdef0.bin"
}
```

2. AMI を復元します。--object-key には、前のステップの出力からの、ファイルパスを含む値を指定します。

```
aws ec2 create-restore-image-task \
    --object-key path1/path2/ami-1234567890abcdef0.bin \
    --bucket myamibucket \
    --name "New AMI Name"
```

AMI を非推奨にする

AMI の使用を避けることで、それが古く、使用すべきではないことを示せます。AMI が非推奨となる将来の日付を特定し、AMI が使用期限切れになるタイミングを知ることも可能です。例えば、現在有効な管理が行われていない AMI の使用を避けたり、新しいバージョンで置き換えられている AMI を避けたりすることができます。新しいユーザーが古い AMI を使用することを防止するため、デフォルトで、非推奨の AMI は AMI のリストに表示されていません。ただし、既存のユーザーおよび起動サービス (起動テンプレートや Auto Scaling グループなど) では、ID を指定することで、非推

奨の AMI を引き続き使用できます。AMI を削除して、ユーザーとサービスが使用できないようにするには、その AMI を登録解除します。

AMIが非推奨となった後は、以下が実施されます。

- AMI ユーザーの場合、非推奨の AMI は (ID を指定した場合や、非推奨の AMI を表示する必要があると指定した場合を除き) DescribeImages API 呼び出しに表示されなくなります。AMIの 所有者に対しては、非推奨の AMI は引き続き DescribeImages API 呼び出しに表示されます。
- AMI ユーザーは、非推奨の AMI をは EC2 コンソール経由で選択できなくなります。例えば、非推奨の AMI は、インスタンスの起動ウィザードの AMI カタログに表示されません。AMI 所有者の EC2 コンソール上には、非推奨の AMI が引き続き表示されます。
- AMI ユーザーで、非推奨となった AMI の ID がわかっている場合は、API、CLI、または SDK により、非推奨の AMI を使用しながらインスタンスの起動を継続することができます。
- 起動テンプレートや Auto Scaling グループなどの起動サービスは、非推奨の AMI を引き続き参照できます。
- 今後非推奨となる予定の AMI を使用して起動された EC2 インスタンスは影響を受けず、停止、起動、および再起動が可能です。

プライベート AMI とパブリック AMI の両方を非推奨にすることができます。

Amazon Data Lifecycle Manager EBS-backed AMI ポリシーを作成して、EBS-backed AMI の廃止を自動化することもできます。詳細については、「AMI ライフサイクルの自動化」を参照してください。

> **Note**
>
> すべてのパブリック AMI を非推奨にする日をデフォルトで AMI 作成日の 2 年後とします。非推奨にする日は 2 年より前の日付に設定できます。非推奨にする日を取り消す場合や、非推奨にする日をもっと先の日付に変える場合は、AMI を特定の AWS アカウントとのみ共有するようにして、AMI を非公開にする必要があります。

トピック

- 非推奨 AMI の詳細表示
- AMI の非推奨のキャンセル

コスト

AMI を非推奨にしても、その AMI は削除されません。AMI 所有者には、その AMI のスナップショットのための料金が引き続き請求されます。スナップショットの支払いを停止するには、AMI 所有者は、登録解除により AMI を削除する必要があります。

制限事項

- AMI を非推奨にするには、AMI の所有者である必要があります。

AMI を非推奨にする

AMI を非推奨にする日時を指定することができます。この手順を実行するには、AMI の所有者である必要があります。

Console

AMI を特定の日付に非推奨にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI を選択し、[Actions] (アクション)、[Manage AMI Deprecation] (AMI 非推奨を管理) の順に選択します。複数の AMI を選択して、複数の AMI の同じ非推奨日を一度に設定できます。
5. [Enable] (有効化) のチェックボックスをオンにして、非推奨となった日時を入力します。

   非推奨日の上限は 10 年後ですが、パブリック AMI は例外で、上限は作成日から 2 年です。過去の日付を指定することはできません。
6. [Save] を選択します。

AWS CLI

AMI を特定の日付に非推奨にするには

enable-image-deprecation コマンドを使用します。AMI の ID、ならびに、その AMI を非推奨にする日時を指定します。秒の値を指定した場合は、Amazon EC2 により最も近い分に丸められます。

deprecate-at の上限は 10 年後ですが、パブリック AMI は例外で、上限は作成日から 2 年です。過去の日付を指定することはできません。

```
aws ec2 enable-image-deprecation \
    --image-id ami-1234567890abcdef0 \
    --deprecate-at "2021-10-15T13:17:12.000Z"
```

正常な出力

```
{
  "Return": "true"
}
```

最終起動時間

LastLaunchedTime は、AMI が最後にインスタンスの起動に使用された時間を示すタイムスタンプです。インスタンスを起動するために最近使用されていない AMI は、非推奨や登録解除の対象となる可能性が高いです。

> **Note**
>
> - インスタンスを起動するために AMI が使用される場合、その発生から 24 時間経過した後に報告されます。
> - lastLaunchedTime データは、2017 年 4 月以降に使用が可能になっています。

Console

AMI の最終起動時間を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。

4. AMI を選択し、[Last launched time] (最終起動時間) フィールドにチェックを入れます (AMI の横にあるチェックボックスを選択した場合、このフィールドは [Details] (詳細) タブにあります)。このフィールドには、インスタンスの起動のために最後に AMI が使用された日時が表示されます。

AWS CLI

AMI の最終起動時間を表示するには

describe-image-attribute コマンドを実行し、`--attribute lastLaunchedTime` を指定します。このコマンドを実行するには、AMI の所有者である必要があります。

```
aws ec2 describe-image-attribute \
    --image-id ami-1234567890example \
    --attribute lastLaunchedTime
```

出力例

```
{
    "LastLaunchedTime": {
        "Value": "2022-02-10T02:03:18Z"
    },
    "ImageId": "ami-1234567890example",
}
```

非推奨 AMI の詳細表示

AMI が非推奨になった日時を表示し、すべての AMI を非推奨になった日付でフィルタリングできます。AWS CLI を使用して、過去の日付で非推奨になっている、すべての AMI についての詳細を表示することもできます。

Console

AMI が非推奨になった日付を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側のナビゲーターで [AMI] をクリックした後、AMI を選択します。
3. [Deprecation time] (非推奨となった時刻) フィールドにチェックを入れます (AMI の横にあるチェックボックスを選択した場合は、[Details] (詳細) タブにあります)。このフィールドに

は、AMI の非推奨の日次が表示されます。フィールドが空の場合は、AMI は非推奨ではありません。

非推奨になった日付で AMI をフィルタリングするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) または [Private images] (プライベートイメージ) を選択します (プライベートイメージには、共有されている AMI のほかに、所有している AMI も含まれます)。
4. 検索バーで **`Deprecation time`** と入力し (文字を入力すると、[Deprecation time] (非推奨となった時刻) のフィルターが表示されます)、演算子と日時を選択します。

AWS CLI

describe-images コマンドを使用してすべての AMI を表示する場合、その結果は、AMI ユーザーに対するものと AMI 所有者に対するもので異なります。

- AMI ユーザーの場合:

  デフォルトでは、すべての AMI を describe-images コマンドにより表示した場合でも、自分が共有しているものの所有はしていない非推奨の AMI は、そのコマンドの結果に表示されません。これは、デフォルトが `--no-include-deprecated` であるためです。非推奨の AMI を結果に含めるには、`--include-deprecated` パラメータを指定します。

- AMI の所有者である場合:

  describe-imagesコマンドを使用して、すべての AMI を表示すると、所有しているすべての AMI (非推奨の AMI を含む) が結果に表示されます。`--include-deprecated` パラメータを指定する必要はありません。また、`--no-include-deprecated` を指定しても、所有している非推奨の AMI を結果から除外することはできません。

非推奨となった AMI については、結果に `DeprecationTime` フィールドが表示されます。

> **Note**
>
> 非推奨の AMI は、そこに過去の日付が表示されている AMI です。非推奨となる日付が将来に設定されている場合、その AMI はまだ非推奨とはなっていません。

すべての非推奨の AMI を含めながらすべての AMI を詳細表示するには

ユーザーが所有していない非推奨の AMI をすべて結果に含めるには、describe-images コマンドを使用して、`--include-deprecated` パラメータを指定します。

```
aws ec2 describe-images \
    --region us-east-1 \
    --owners 123456example
    --include-deprecated
```

AMI が非推奨となった日付を表示するには

describe-images コマンドを実行する際に、AMI の ID を指定します。

AMI ID とともに `--no-include-deprecated` を指定しても、非推奨の AMI が結果に返されることに注意してください。

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-ids ami-1234567890EXAMPLE
```

正常な出力

`DeprecationTime` フィールドには、AMI が非推奨にされる予定の日付が表示されます。AMI を非推奨にすることが設定されていない場合、`DeprecationTime` フィールドは出力には表示されません。

```
{
    "Images": [
        {
            "VirtualizationType": "hvm",
            "Description": "Provided by Red Hat, Inc.",
            "PlatformDetails": "Red Hat Enterprise Linux",
```

```
            "EnaSupport": true,
            "Hypervisor": "xen",
            "State": "available",
            "SriovNetSupport": "simple",
            "ImageId": "ami-1234567890EXAMPLE",
            "DeprecationTime": "2021-05-10T13:17:12.000Z"
            "UsageOperation": "RunInstances:0010",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-111222333444aaabb",
                        "DeleteOnTermination": true,
                        "VolumeType": "gp2",
                        "VolumeSize": 10,
                        "Encrypted": false
                    }
                }
            ],
            "Architecture": "x86_64",
            "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-
GP2",
            "RootDeviceType": "ebs",
            "OwnerId": "123456789012",
            "RootDeviceName": "/dev/sda1",
            "CreationDate": "2019-05-10T13:17:12.000Z",
            "Public": true,
            "ImageType": "machine",
            "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
        }
    ]
}
```

AMI の非推奨のキャンセル

AMI を非推奨とすることをキャンセルできます。これにより、[Deprecation time] (非推奨となった時刻) フィールド (コンソール) から日時が削除され、または describe-images 出力 (AWS CLI) から DeprecationTime フィールドが削除されます。この手順を実行するには、AMI の所有者である必要があります。

Console

非推奨となっている AMI を復旧するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI を選択し、[Actions] (アクション)、[Manage AMI Deprecation] (AMI 非推奨を管理) の順に選択します。複数の AMI を選択して、複数の AMI の非推奨を一度にキャンセルできます。
5. [Enable] (有効化) チェックボックスをオフにして、[Save] (保存) を選択します。

AWS CLI

非推奨となっている AMI を復旧するには

AMI の ID を指定しながら、disable-image-deprecation コマンドを実行します。

```
aws ec2 disable-image-deprecation \
    --image-id ami-1234567890abcdef0
```

正常な出力

```
{
  "Return": "true"
}
```

AMI の無効化

AMI を無効にして、インスタンスの起動に使用されないようにできます。無効な AMI から新しいインスタンスを起動することはできません。無効化された AMI を再度有効にして、インスタンスの起動時に再び使用できるようにすることができます。

> **Warning**
>
> AMI を無効にすると、AMI のすべての起動権限が削除されます。

AMI が無効になっている場合:

- AMI の状態は `disabled` に変わります。
- 無効化された AMI は共有できません。AMI が公開されていたか、以前に共有されていた場合は、非公開になります。AMI が AWS アカウント、組織または組織単位で共有されていた場合、無効になっている AMI にはアクセスできなくなります。
- 無効化された AMI は、デフォルトで DescribeImages API 呼び出しに表示されません。
- 無効化された AMI は [自分が所有] コンソールフィルタには表示されません。無効になっている AMI を検索するには、[無効化されたイメージ] コンソールフィルタを使用してください。
- 無効化された AMI は、EC2 コンソールのインスタンス起動時に選択できません。たとえば、無効化された AMI はインスタンスの起動ウィザードの AMI カタログに表示されません。また、起動テンプレート作成時にも表示されません。
- 起動テンプレートや Auto Scaling グループなどの起動サービスは、無効化された AMI を引き続き参照できます。無効化された AMI からのそれ以降のインスタンスの起動は失敗するため、使用可能な AMI のみを参照するように、起動テンプレートと Auto Scaling グループを更新することをお勧めします。
- 今後無効化される予定の AMI を使用して起動された EC2 インスタンスは影響を受けず、停止、起動、および再起動が可能です。
- 無効になっている AMI に関連するスナップショットは削除できません。関連するスナップショットを削除しようとすると `snapshot is currently in use` エラーになります。

AMI が再び有効になると:

- AMI の状態が `available` に変わり、インスタンスの起動に使用できるようになります。
- AMI は共有できます。
- AMI を無効にしたときに AMI にアクセスできなくなった AWS アカウント、組織、および組織単位は、自動的にアクセスを回復できませんが、AMI を再び共有することは可能です。

プライベート AMI とパブリック AMI の両方を無効化できます。

トピック

- コスト
- 前提条件
- 必要な IAM 許可

- AMI の無効化
- 無効化された AMI の説明
- 無効化された AMI を再度有効にする

コスト

AMI を無効化しても、その AMI は削除されません。AMI が EBS ベースである場合は、AMI の EBS スナップショットの料金を引き続きお支払いいただきます。AMI を残しておきたい場合は、スナップショットをアーカイブすることでストレージコストを削減できる場合があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットのアーカイブ」を参照してください。AMI とそのスナップショットを保持したくない場合は、AMI を登録解除し、スナップショットを削除する必要があります。詳細については、「Amazon EBS-backed AMI のクリーンアップ」を参照してください。

前提条件

AMI を無効または再度有効にするには、AMI の所有者である必要があります。

必要な IAM 許可

AMI を無効化する、および再度有効化するには、次の IAM 権限が必要です。

- `ec2:DisableImage`
- `ec2:EnableImage`

AMI の無効化

AMI は EC2 コンソールまたは AWS Command Line Interface (AWS CLI) を使用して無効にできます。この手順を実行するには、AMI の所有者である必要があります。

Console

AMI を無効化するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。

4. AMI を選択し、[アクション]、[AMI を無効にする] の順に選択します。複数の AMI を選択し、まとめて無効化することもできます。
5. [AMI を無効にする] ウィンドウで、[AMI を無効にする] を選択します。

AWS CLI

AMI を無効化するには

disable-image コマンドを使用して、AMI の ID を指定します。

```
aws ec2 disable-image --image-id ami-1234567890abcdef0
```

正常な出力

```
{
  "Return": "true"
}
```

無効化された AMI の説明

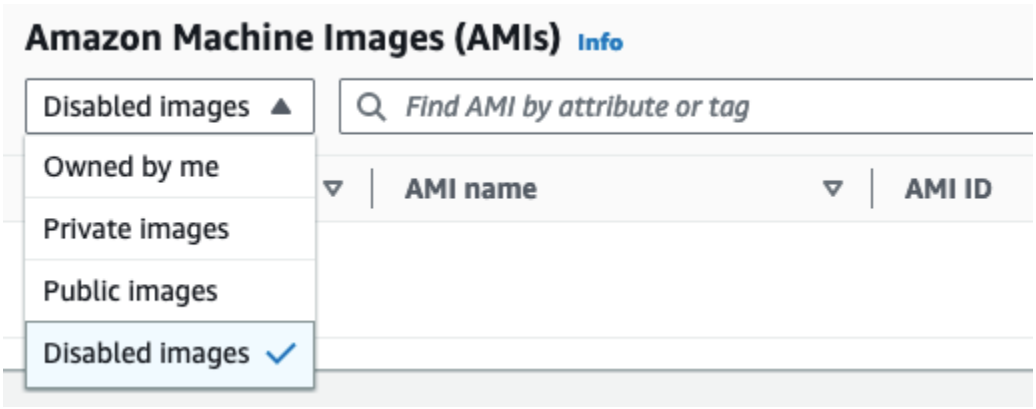
無効化された AMI は EC2 コンソールと AWS CLI を使用して表示できます。

無効化された AMI を表示するには AMI 所有者である必要があります。無効化された AMI は非公開になるため、所有者以外には表示されません。

Console

無効化された AMI を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで [AMI] を選択します。
3. フィルターバーから [無効化された画像] を選択します。



AWS CLI

デフォルトでは、describe-images コマンドを使用してすべての AMI を記述しても、無効化された AMI は結果に表示されません。これは、デフォルトが `--no-include-disabled` であるためです。無効化された AMI を結果に含めるには、`--include-disabled` パラメータを指定する必要があります。

すべての無効化された AMI を含めながらすべての AMI を詳細表示するには

describe-images コマンドを使用して `--include-disabled` パラメータを指定すると、他のすべての AMI に加えて無効化された AMI も取得できます。オプションで、所有している AMI のみを取得するように `--owners self` を指定できます。

```
aws ec2 describe-images \
    --region us-east-1 \
    --owners self
    --include-disabled
```

無効な AMI の ID を指定しても `--include-disabled` は指定しない場合、無効な AMI が結果で返されます。

```
aws ec2 describe-images \
    --region us-east-1 \
    --image-ids ami-1234567890EXAMPLE
```

無効になっている AMI のみを取得するには

`--filters Name=state,Values=disabled` を指定します。`--include-disabled` も指定する必要があります。指定しないとエラーが返されます。

```
aws ec2 describe-images \
    --include-disabled \
    --filters Name=state,Values=disabled
```

出力例

State のフィールドには AMI の状態が表示されます。disabled は AMI が無効になっていることを示します。

```
{
    "Images": [
        {
            "VirtualizationType": "hvm",
            "Description": "Provided by Red Hat, Inc.",
            "PlatformDetails": "Red Hat Enterprise Linux",
            "EnaSupport": true,
            "Hypervisor": "xen",
            "State": "disabled",
            "SriovNetSupport": "simple",
            "ImageId": "ami-1234567890EXAMPLE",
            "DeprecationTime": "2023-05-10T13:17:12.000Z"
            "UsageOperation": "RunInstances:0010",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-111222333444aaabb",
                        "DeleteOnTermination": true,
                        "VolumeType": "gp2",
                        "VolumeSize": 10,
                        "Encrypted": false
                    }
                }
            ],
            "Architecture": "x86_64",
            "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-
GP2",
            "RootDeviceType": "ebs",
            "OwnerId": "123456789012",
            "RootDeviceName": "/dev/sda1",
            "CreationDate": "2019-05-10T13:17:12.000Z",
            "Public": false,
            "ImageType": "machine",
```

```
            "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
        }
    ]
}
```

無効化された AMI を再度有効にする

無効化された AMI を再度有効にすることができます。この手順を実行するには、AMI の所有者である必要があります。

Console

無効化した AMI を再度有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで [AMI] を選択します。
3. フィルターバーから [無効化された画像] を選択します。
4. AMI を選択し、[アクション]、[AMI を有効化] の順に選択します。複数の AMI を選択し、まとめて再有効化を選択することもできます。
5. [AMI を有効化] ウィンドウで、[有効化] を選択します。

AWS CLI

無効化した AMI を再度有効にするには

enable-image コマンドを使用して、AMI の ID を指定します。

```
aws ec2 enable-image --image-id ami-1234567890abcdef0
```

正常な出力

```
{
  "Return": "true"
}
```

AMI スナップショットをアーカイブする

無効化した EBS ベースの AMI に関連付けられているスナップショットをアーカイブできます。これにより、使用頻度が低く、長期間保持する必要がある AMI に関連付けられたストレージコストを削減できます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットのアーカイブ」を参照してください。

AMI に関連付けられたスナップショットをアーカイブするには

1. AMI を無効にします。
2. スナップショットをアーカイブします。

AMI が無効で、関連付けられたスナップショットがアーカイブされている間は AMI を使用できません。

アーカイブされたスナップショットを使用して無効化された AMI を復元するには

1. AMI に関連付けられているアーカイブされたスナップショットを復元します。
2. AMI を有効にします。

AMI の登録の解除

AMI の利用が終わったら、その登録を解除できます。AMI の登録を解除すると、それを使用して新しいインスタンスを起動できなくなります。

AMI の登録を解除しても、その AMI から作成済みのインスタンスやその AMI の作成プロセスで作成されたスナップショットには影響はありません。これらのスナップショットのインスタンスとストレージコストには、引き続き使用料が発生します。したがって、その使用が終了した際には、インスタンスを終了させ、スナップショットを削除する必要があります。

AMI のクリーンアップのための手順は、動作に Amazon EBS を使用しているか、インスタンスストアを使用しているかで異なります。詳細については、AMI のルートデバイスタイプの判別 を参照してください。

コンテンツ

- 最終起動時間

考慮事項

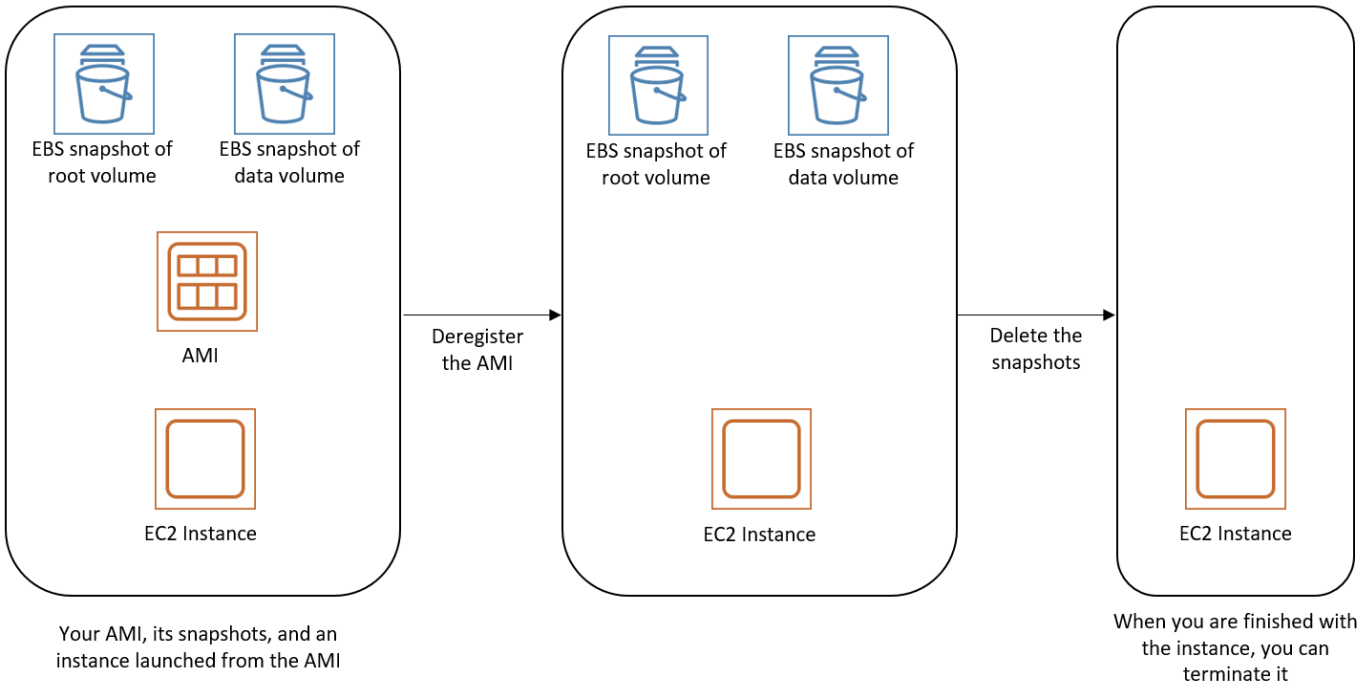
AMI の登録解除には、次の考慮事項が適用されます。

- アカウントが所有していない AMI の登録を解除することはできません。
- Amazon EC2 を使った AWS Backup サービスで管理されている AMI の登録解除はできません。代わりに、AWS Backup を使用して、バックアップボールト内の対応するリカバリポイントを削除します。詳細については、「AWS Backup デベロッパーガイド」の「Deleting backups」(バックアップの削除) を参照してください。

Amazon EBS-backed AMI のクリーンアップ

Amazon EBS-backed AMI の登録を解除しても、その AMI の作成プロセス中にインスタンスのボリューム 用に作成したスナップショットには影響しません。スナップショットのストレージは引き続き課金されます。そのため、使用が終わったスナップショットは削除することをお勧めします。

次の図に、Amazon EBS-backed AMI のクリーンアッププロセスを示します。



Amazon EBS-backed AMI をクリーンアップするには、次のいずれかの方法を使用します。

Console

Amazon EBS-backed AMI をクリーンアップするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. AMI の登録を解除する
   a. ナビゲーションペインで [AMIs] を選択します。
   b. フィルターバーから [自己所有] を選択して使用可能な AMI を一覧表示するか、[無効化されたイメージ] を選択して無効になっている AMI を一覧表示します。
   c. 登録を解除する AMI を選択し、その ID を書き留めます — これは、削除するスナップショットを次のステップで見つけるのに役立ちます。
   d. [Actions] (アクション)、[Deregister AMI] (AMI の登録解除) の順に選択します。確認を求めるメッセージが表示されたら、[Deregister AMI] (AMI の登録解除) を選択します。

   > **Note**
   > コンソールで AMI がリストから削除されるまで、数分ほどかかります。ステータスを更新するには、[Refresh] を選択します。

3. 不要になったスナップショットの削除
   a. ナビゲーションペインで、[Snapshots] を選択します。
   b. 削除するスナップショットを選択します ([Description] (説明) 列で前のステップから AMI ID を検索)。
   c. [Actions] (アクション)、[Delete snapshot] (スナップショットの削除) の順に選択します。確認を求めるメッセージが表示されたら、[削除] を選択します。
4. (オプション) インスタンスの終了

   AMI から起動したインスタンスの使用が終わったら、それを削除できます。

   a. ナビゲーションペインで、[Instances] (インスタンス) をクリックした上で、終了するインスタンスを選択します。
   b. [Instance state (インスタンスの状態)]、[Terminate instance (インスタンスの終了)] の順に選択します。確認を求めるメッセージが表示されたら、[終了] を選択します。

AWS CLI

以下の手順に従い、Amazon EBS-backed AMI をクリーンアップします。

1. AMI の登録を解除する

   deregister-image コマンドを使用して AMI の登録を解除します。

   ```
   aws ec2 deregister-image --image-id ami-12345678
   ```

2. 不要になったスナップショットの削除

   delete-snapshotコマンドを使用して、不要になったスナップショットを削除します。

   ```
   aws ec2 delete-snapshot --snapshot-id snap-1234567890abcdef0
   ```

3. インスタンスの削除 (オプション)

   AMI から起動したインスタンスの使用を終えた場合には、terminate-instances コマンドによりそのインスタンスを削除できます。

   ```
   aws ec2 terminate-instances --instance-ids i-12345678
   ```

PowerShell

以下の手順に従い、Amazon EBS-backed AMI をクリーンアップします。

1. AMI の登録を解除する

   AMI の登録を解除するには、Unregister-EC2Imageコマンドレットを使用します。

   ```
   Unregister-EC2Image -ImageId ami-12345678
   ```

2. 不要になったスナップショットの削除

   不要になったスナップショットは、Remove-EC2Snapshotコマンドレットを使用して削除します。

   ```
   Remove-EC2Snapshot -SnapshotId snap-12345678
   ```

3. インスタンスの削除 (オプション)

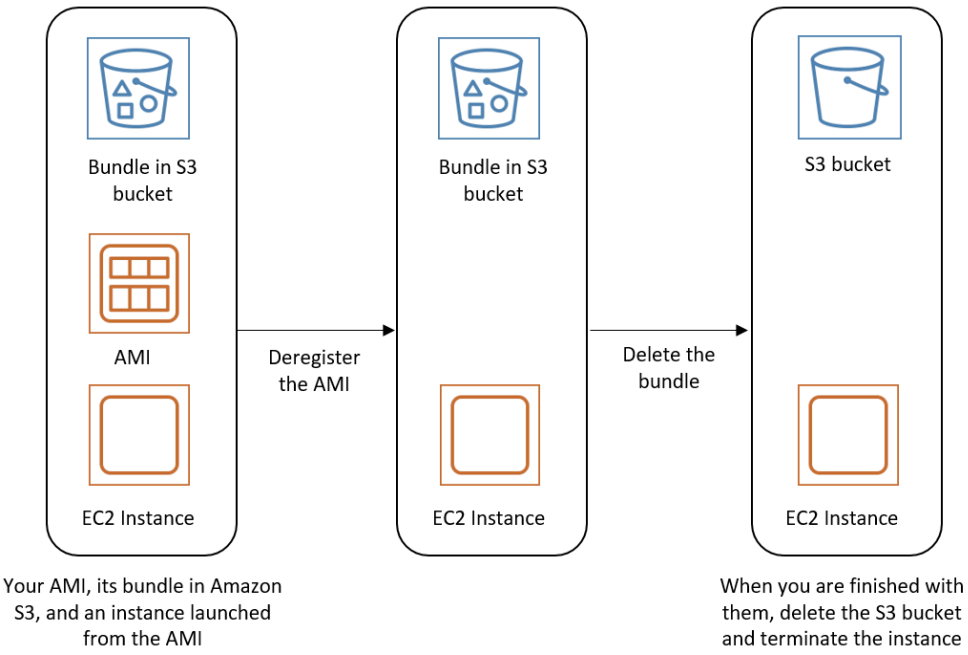
   AMI から起動したインスタンスの使用を終えた場合には、Remove-EC2Instance コマンドレットによりそのインスタンスを削除できます。

```
Remove-EC2Instance -InstanceId i-12345678
```

Instance Store-Backed AMI のクリーンアップ

Instance Store-Backed AMI の登録を解除しても、AMI の作成時に Amazon S3 にアップロードしたファイルには影響しません。Amazon S3 のそのファイルの使用に対しては引き続き課金されます。そのため、使用が終わったら、それらのファイルは削除することをお勧めします。

次の図は、Instance Store-Backed AMI のクリーンアッププロセスをまとめたものです。



Instance Store-Backed AMI をクリーンアップするには

1. 次のように、deregister-image コマンドを使用して AMI の登録を解除します。

```
aws ec2 deregister-image --image-id ami_id
```

2. 次のように ec2-delete-bundle (AMI ツール) コマンドを使用して、Amazon S3 のバンドルを削除します。

```
ec2-delete-bundle -b myawsbucket/myami -a your_access_key_id -
s your_secret_access_key -p image
```

3. (オプション) AMI から起動したインスタンスの使用が終わったら、次のように、terminate-instances コマンドを使用してそれを終了します。

```
aws ec2 terminate-instances --instance-ids instance_id
```

4. (オプション) バンドルをアップロードした Amazon S3 バケットの使用が終わったら、バケットを削除できます。Amazon S3 バケットを削除するには、Amazon S3 コンソールを開き、バケットを選択してから、[Actions]、[Delete] の順に選択します。

最終起動時間

`LastLaunchedTime` は、AMI が最後にインスタンスの起動に使用された時間を示すタイムスタンプです。インスタンスを起動するために最近使用されていない AMI は、登録解除や非推奨の対象となる可能性が高いです。

> **Note**
>
> - インスタンスを起動するために AMI が使用される場合、その発生から 24 時間経過した後に報告されます。
> - `lastLaunchedTime` データは、2017 年 4 月以降に使用が可能になっています。

Console

AMI の最終起動時間を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーターで [AMI] を選択します。
3. フィルターバーから、[Owned by me] (自己所有) を選択します。
4. AMI を選択し、[Last launched time] (最終起動時間) フィールドにチェックを入れます (AMI の横にあるチェックボックスを選択した場合、このフィールドは [Details] (詳細) タブにあります)。このフィールドには、インスタンスの起動のために最後に AMI が使用された日時が表示されます。

AWS CLI

AMI の最終起動時間を表示するには

describe-image-attribute コマンドを実行し、--attribute lastLaunchedTime を指定します。このコマンドを実行するには、AMI の所有者である必要があります。

```
aws ec2 describe-image-attribute \
    --image-id ami-1234567890example \
    --attribute lastLaunchedTime
```

出力例

```
{
    "LastLaunchedTime": {
        "Value": "2022-02-10T02:03:18Z"
    },
    "ImageId": "ami-1234567890example",
}
```

EBS-backed AMI ライフサイクルの自動化

Amazon Data Lifecycle Manager を使用して、Amazon EBS-backed AMI とそのバッキングスナップショットの作成、保持、コピー、非推奨、登録削除を自動化できます。詳細については、「Amazon Data Lifecycle Manager」を参照してください。

EBS-backed AMI での暗号化の利用

Amazon EBS スナップショットを使用した AMI は Amazon EBS 暗号化の利点を活かすことができます。データおよびルートボリュームの両方のスナップショットを暗号化して AMI にアタッチできます。インスタンスを起動し、完全な EBS 暗号化サポートも含めてイメージをコピーできます。これらのオペレーションの暗号化パラメータは、AWS KMS が利用できるすべてのリージョンでサポートされています。

暗号化された EBS ボリュームを持つ EC2 インスタンスは、他のインスタンスと同様に AMIs から起動します。また、暗号化されていない EBS スナップショットでバックアップされている AMI からインスタンスを起動するとき、起動中に一部またはすべてのボリュームを暗号化できます。

EBS ボリュームと同様に、AMI のスナップショットはデフォルトの AWS KMS key または指定したカスタマーマネージド型キーで暗号化できます。いずれの場合も、選択した KMS キー を使用するためのアクセス権限が必要です。

暗号化されたスナップショットを持つ AMI は、AWS アカウント間で共有できます。詳細については、共有 AMI を参照してください。

EBS-backed AMI トピックでの暗号化

- インスタンスの起動シナリオ
- イメージコピーのシナリオ

インスタンスの起動シナリオ

AMI から Amazon EC2 インスタンスを起動するには、`RunInstances` または直接 Amazon EC2 API や CLI を使用して、AWS Management Console アクションを実行します。その際、ブロックデバイスマッピングから提供されるパラメータを指定します。ブロックデバイスマッピングに関する詳細については、「ブロックデバイスマッピング」を参照してください。AWS CLI からブロックデバイスマッピングを制御する例については、「EC2 インスタンスを起動、リスト、および終了する」を参照してください。

デフォルトでは、明示的な暗号化パラメータがない場合、AMI のソーススナップショットから EBS ボリュームを復元しているときに、`RunInstances` アクションは AMI のソーススナップショットの既存の暗号化状態を維持します。デフォルトでの暗号化が有効になっている場合、AMI から作成したすべてのボリュームが暗号化されます (作成元のスナップショットが暗号化されているかどうかは関係ありません)。デフォルトでの暗号化が有効にされていない場合、インスタンスは AMI の暗号化状態を維持します。

インスタンスを起動し、同時に、暗号化パラメータを指定して、新しい暗号化状態を生成されるボリュームに適用することもできます。そのため、以下の動作が観察されます。

暗号化パラメータなしでの起動

- デフォルトでの暗号化が有効にされている場合を除き、暗号化されていないスナップショットは、暗号化されていないボリュームに復元されます。デフォルトでの暗号化が有効にされている場合は、新しく作成されるすべてのボリュームが暗号化されます。
- 所有する暗号化されたスナップショットは、同じ KMS キー に暗号化されるボリュームに復元されます。

- 所有していない (例えば、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているボリュームに復元されます。

デフォルトの動作は、暗号化パラメータを指定してオーバーライドできます。利用できるパラメータは、Encrypted と KmsKeyId です。Encrypted パラメータのみを設定すると、次のような結果になります。

Encrypted を設定し、**KmsKeyId** を指定しない場合のインスタンス起動動作

- 暗号化されていないスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されている EBS ボリュームに復元されます。
- 所有する暗号化されたスナップショットは、同じ KMS キー により暗号化された EBS ボリュームに復元されます。(つまり、Encrypted パラメータには効果がありません。)
- 所有していない (つまり、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているボリュームに復元されます。(つまり、Encrypted パラメータには効果がありません。)

Encrypted と KmsKeyId 両方のパラメータを設定すると、暗号化オペレーションにデフォルトではない KMS キー を指定できます。結果として次のように動作します。

Encrypted と **KmsKeyId** が両方設定されたインスタンス

- 暗号化されていないスナップショットは、指定された KMS キー により暗号化された EBS ボリュームに復元されます。
- 暗号化されたスナップショットは、元の KMS キー ではなく、指定された KMS キー に暗号化された EBS ボリュームに復元されます。

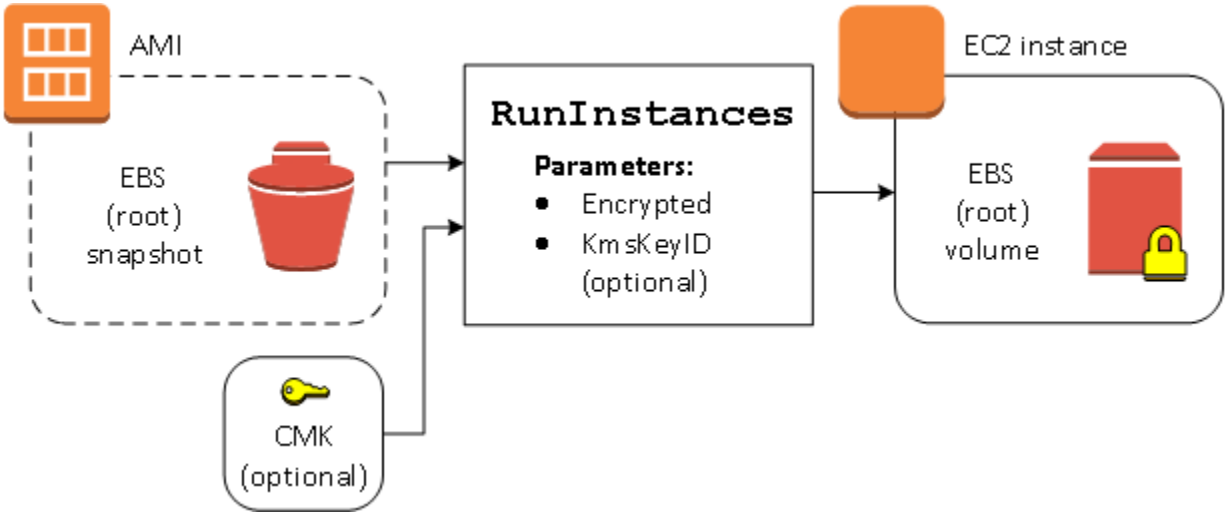
Encrypted パラメータも設定せずに KmsKeyIdを送信するとエラーが発生します。

以下のセクションでは、デフォルトではない暗号化パラメータを使用して AMI からインスタンスを起動する例を示します。これらの各シナリオでは、RunInstances アクションに指定するパラメータにより、スナップショットからボリュームを復元中に暗号化の状態が変化します。

コンソールを使用して AMI からインスタンスを起動する方法については、「インスタンスの起動」を参照してください。

起動時にボリュームを暗号化する

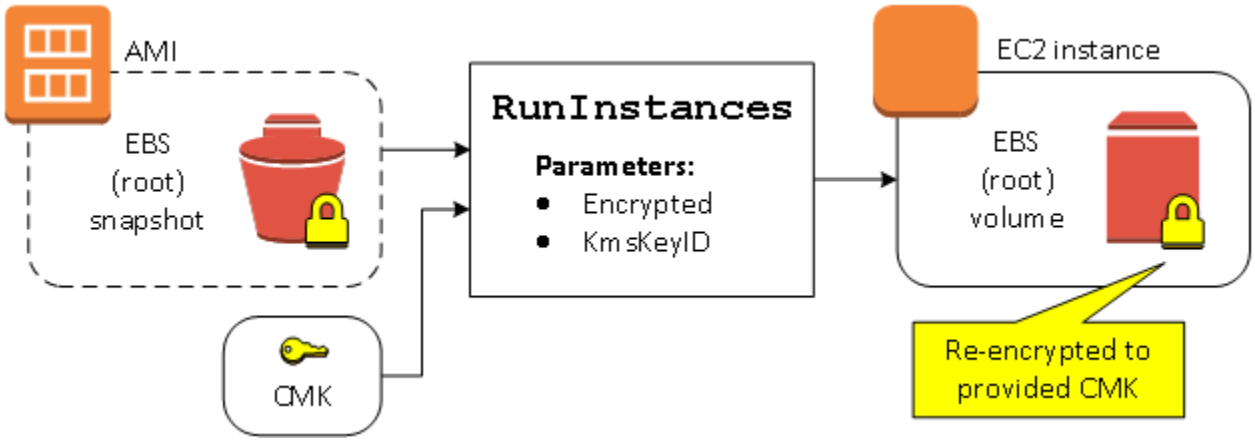
この例では、暗号化されていないスナップショットでバックアップされた AMI を使用して、暗号化された EBS ボリュームのある EC2 インスタンスを起動します。



`Encrypted` パラメータのみを使用すると、このインスタンスのボリュームが暗号化されます。`KmsKeyId` パラメータの指定はオプションです。KMS キー ID を指定しない場合、AWS アカウントのデフォルト KMS キーを使用して、ボリュームを暗号化します。所有する別の KMS キー にボリュームを暗号化するには、`KmsKeyId` パラメータを指定します。

起動時にボリュームを再暗号化する

この例では、暗号化されたスナップショットでバックアップされた AMI を使用して、新しい KMS キー により暗号化された EBS ボリュームのある EC2 インスタンスを起動します。

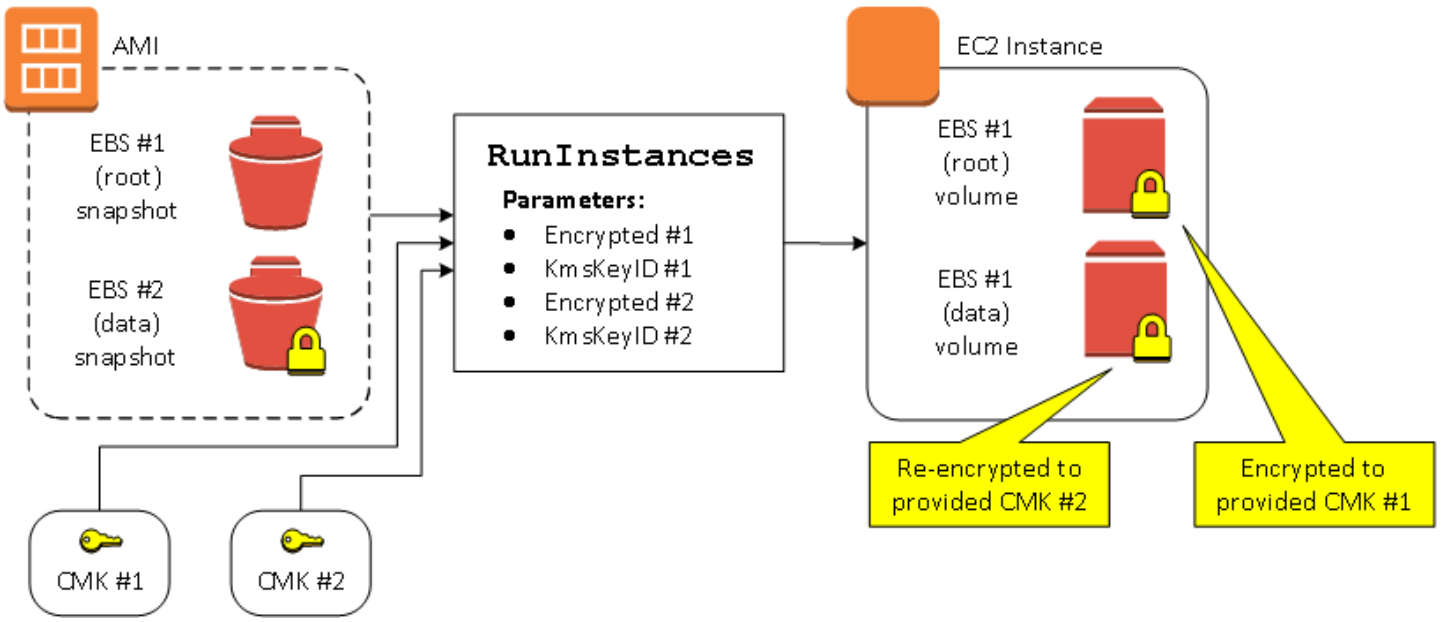


AMI を所有していて、暗号化パラメータを指定しない場合、作成されるインスタンスではスナップショットと同じ KMS キーでボリュームが暗号化されます。他のアカウントから共有された AMI に

対して暗号化パラメータを指定しない場合、ボリュームはデフォルト KMS キー により暗号化されます。図のように暗号化パラメータが指定されている場合、ボリュームは指定された KMS キー により暗号化されます。

起動時に複数のボリュームの暗号化状態を変更する

このより複雑な例では、複数のスナップショット (暗号化状態はそれぞれ異なります) でバックアップされた AMI を使用して、新しく暗号化されたボリュームと再暗号化されたボリュームがある EC2 インスタンスを起動します。



このシナリオでは、RunInstances アクションにソーススナップショットそれぞれに対する暗号化パラメータが指定されます。可能な暗号化パラメータがすべて指定されると、AMI を所有しているかどうかに関係なく、作成されるインスタンスは同じです。

イメージコピーのシナリオ

Amazon EC2 AMI をコピーするには、CopyImage または直接 Amazon EC2 API や CLI を使用して、AWS Management Console アクションを実行します。

デフォルトでは、明示的な暗号化パラメータがない場合、コピー中 CopyImage アクションは AMI のソーススナップショットの既存の暗号化状態を維持します。AMI をコピーし、同時に、暗号化パラメータを指定して、新しい暗号化状態を関連付けられている EBS スナップショットに適用することもできます。そのため、以下の動作が観察されます。

暗号化パラメータなしでのコピー

- デフォルトでの暗号化が有効にされている場合を除き、暗号化されていないスナップショットは、別の暗号化されていないスナップショットにコピーされます。デフォルトでの暗号化が有効にされている場合は、新しく作成されるすべてのスナップショットが暗号化されます。
- 所有する暗号化されたスナップショットは、同じ KMS キー で暗号化されたスナップショットにコピーされます。
- 所有していない (例えば、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているスナップショットにコピーされます。

これらすべてのデフォルトの動作は、暗号化パラメータを指定してオーバーライドできます。利用できるパラメータは、Encrypted と KmsKeyId です。Encrypted パラメータのみを設定すると、次のような結果になります。

Encrypted を設定し、**KmsKeyId** を指定しない場合のコピーイメージ動作

- 暗号化されていないスナップショットは、AWS アカウントのデフォルト KMS キーで暗号化されたスナップショットにコピーされます。
- 暗号化されたスナップショットは、同じ KMS キー により暗号化されたスナップショットにコピーされます。(つまり、Encrypted パラメータには効果がありません。)
- 所有していない (例えば、AMI が共有されている) 暗号化されたスナップショットは、ユーザーの AWS アカウントのデフォルト KMS キーで暗号化されているボリュームにコピーされます。(つまり、Encrypted パラメータには効果がありません。)

Encrypted と KmsKeyId 両方のパラメータを設定すると、暗号化オペレーションにカスタマー管理 KMS キー を指定できます。結果として次のように動作します。

Encrypted と **KmsKeyId** の両方を設定した場合のコピーイメージ動作

- 暗号化されていないスナップショットは、指定された KMS キー により暗号化されたスナップショットにコピーされます。
- 暗号化されたスナップショットは、元の KMS キー ではなく、指定された KMS キー に暗号化されたスナップショットにコピーされます。

Encrypted パラメータも設定せずに KmsKeyIdを送信すると エラーが発生します。

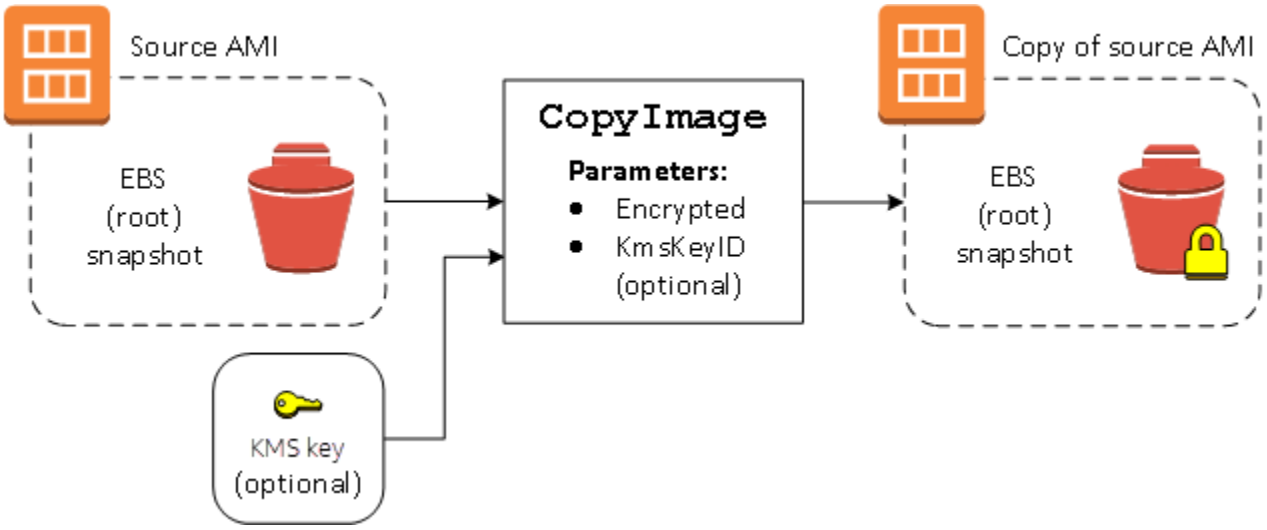
以下のセクションでは、デフォルトではない暗号化パラメータを使用して AMI をコピーし、結果として暗号化状態が変化する例を示します。

コンソールを使用する手順の詳細については、「AMI のコピー」を参照してください。

コピー時に暗号化されていないイメージを暗号化する

このシナリオでは、暗号化されていないルートスナップショットでバックアップされた AMI は、暗号化されたルートスナップショットを持つ AMI にコピーされます。CopyImage アクションは、カスタマーマネージド型キーなど、2 つの暗号化パラメータで呼び出されます。その結果、ルートスナップショットの暗号化ステータスが変更され、ターゲット AMI はソーススナップショットと同じデータを含むルートスナップショットにバックアップされますが、指定されたキーを使用して暗号化されます。両方の AMI でスナップショットのストレージコストと、いずれかの AMI から起動するインスタンスの料金が発生します。

> **Note**
>
> デフォルトで暗号化を有効にすると、AMI 内のすべてのスナップショットで Encrypted パラメータを true に設定したのと同じ効果があります。



Encrypted パラメータを設定すると、このインスタンスの単一のスナップショットが暗号化されます。KmsKeyId パラメータを指定しない場合は、デフォルトのカスタマーマネージド型キーを使用して、スナップショットのコピーが暗号化されます。

> **Note**
>
> 複数のスナップショットがあるイメージをコピーして、それぞれの暗号化状態を個々に設定することもできます。

Amazon EventBridge を使用して AMI イベントをモニタリングする

Amazon マシンイメージ (AMI) の状態に変更があった場合、Amazon EC2 はイベントを生成し、それを Amazon EventBridge (旧 Amazon CloudWatch Events) に送信します。Amazon EventBridge を使用することで、これらのイベントの検出と対応が行えるようになります。EventBridge では、イベントに応答してアクションをトリガーするためのルールを作成します。例えば、AMI 作成プロセスが完了したことを検出し、Amazon SNS トピックを呼び出して E メール通知をユーザーに送信する、EventBridge ルールを作成できます。

AMI が以下のいずれかの状態に遷移すると、Amazon EC2 はイベントを生成します。

- `available`
- `failed`
- `deregistered`
- `disabled`

次のテーブルは、AMI の操作と AMI が入力できる状態を示しています。テーブルの [はい] は、対応する操作が実行されたときに AMI が入力できる状態を示しています。

AMI オペレーション	available	failed	deregistered	disabled
CopyImage	はい	はい		
CreateImage	はい	はい		
CreateRestoreImageTask	はい	はい		
DeregisterImage			はい	

AMI オペレーション	available	failed	deregistered	disabled
DisableImage				はい
EnableImage	はい			
RegisterImage	はい	はい		

イベントは、ベストエフォートベースで生成されます。

トピック

- AMI イベント
- Amazon EventBridge ルールを作成する

AMI イベント

4 つの EC2 AMI State Change イベントがあります。

- available
- failed
- deregistered
- disabled

イベントは、EventBridge のデフォルトのイベントバスに、JSON 形式で送信されます。

イベント内の以下のフィールドは、アクションをトリガーするルールを作成するために使用します。

`"source": "aws.ec2"`

イベントが Amazon EC2 からのものであるかを特定します。

`"detail-type": "EC2 AMI State Change"`

イベント名を特定します。

`"detail": { "ImageId": "ami-0123456789example", "State": "available", }`

以下の情報を提供します。

- AMI ID – 特定の AMI を追跡する場合。
- AMI が (available、failed、deregistered、または disabled) の状態。

available

以下に、CreateImage、CopyImage、RegisterImage、CreateRestoreImageTask、または EnableImage が正常に処理された後、AMI が available 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。

"State": "available" は、このオペレーションが正常に処理されたことを示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0123456789example"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0123456789example",
        "State": "available",
        "ErrorMessage": ""
    }
}
```

failed

以下に、CreateImage、CopyImage、RegisterImage、または CreateRestoreImageTask が正常に処理された後、AMI が failed 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。

以下のフィールドにより、関連する情報が提供されます。

- "State": "failed" – オペレーションが失敗したことを示します。
- "ErrorMessage": "" – オペレーション失敗の理由を示します。

```
{
```

```
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0123456789example"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0123456789example",
        "State": "failed",
        "ErrorMessage": "Description of failure"
    }
}
```

deregistered

以下に、DeregisterImage が正常に処理された後、AMI が deregistered 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。オペレーションが失敗した場合、イベントの生成は行われません。DeregisterImage は同期オペレーションであるため、処理が失敗した場合は直ちに認識されます。

"State": "deregistered" は、DeregisterImage のオペレーションが正常に処理されたことを示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0123456789example"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0123456789example",
        "State": "deregistered",
        "ErrorMessage": ""
    }
}
```

disabled

以下に、DisableImage が正常に処理された後、AMI が disabled 状態に遷移する際に Amazon EC2 が生成するイベントの例を示します。オペレーションが失敗した場合、イベントの生成は行われません。DisableImage は同期オペレーションであるため、処理が失敗した場合は直ちに認識されます。

"State": "disabled" は、DisableImage のオペレーションが正常に処理されたことを示します。

```
{
    "version": "0",
    "id": "example-9f07-51db-246b-d8b8441bcdf0",
    "detail-type": "EC2 AMI State Change",
    "source": "aws.ec2",
    "account": "012345678901",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": ["arn:aws:ec2:us-east-1::image/ami-0123456789example"],
    "detail": {
        "RequestId": "example-9dcc-40a6-aa77-7ce457d5442b",
        "ImageId": "ami-0123456789example",
        "State": "disabled",
        "ErrorMessage": ""
    }
}
```

Amazon EventBridge ルールを作成する

Amazon EventBridge では、ルールを作成することで、そのルール内のイベントパターンに一致するイベントを受信した際に実行する、アクションを指定できます。イベントが一致すると、EventBridge は指定されたターゲットにイベントを送信し、ルールで定義されたアクションをトリガーします。

イベントパターンは、一致するイベントと同じ構造をしています。イベントパターンは、イベントに一致するか、一致しないかのいずれかになります。

AMI 状態変更イベントのルール作成時、イベントパターンに以下のフィールドを含めることができます。

```
"source": "aws.ec2“
```

イベントが Amazon EC2 からのものであるかを特定します。

```
"detail-type": "EC2 AMI State Change"
```

イベント名を特定します。

```
"detail": { "ImageId": "ami-0123456789example", "State": "available", }
```

以下の情報を提供します。

- AMI ID – 特定の AMI を追跡する場合。
- AMI が (`available`、`failed`、`deregistered`、または `disabled`) の状態。

例: 通知を送信する EventBridge ルールを作成する

以下の例では、`CreateImage` オペレーションが正常に完了した後、AMI が `available` 状態に遷移した際に、E メール、テキストメッセージ、あるいはモバイルのプッシュ通知を送信する、EventBridge ルールを作成します。

EventBridge ルールを作成する前に、E メール、テキストメッセージ、またはモバイルプッシュ通知用の Amazon SNS トピックを作成する必要があります。

AMI が作成され **`available`** 状態にある場合に通知を送信する EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. [Create rule] を選択します。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。

   b. [イベントバス] として、[デフォルト] を選択します。アカウント内の AWS のサービスで生成されたイベントは、常に、そのアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。
   d. [Next] を選択します。
4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。

a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択します。

b. この例では、AMI が `available` 状態に遷移した際に生成される `EC2 AMI State Change` イベントと一致するイベントパターンを [Event pattern] (イベントパターン) で指定します。

```
{
 "source": ["aws.ec2"],
 "detail-type": ["EC2 AMI State Change"],
 "detail": {"State": ["available"]}
}
```

イベントパターンを追加するには、以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには、以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択します。

B. [イベントパターンフォーム] では、AWS[サービス] を選択します。

C. [AWS Service] (のサービス) で [EC2] を選択します。

D. [Event type] (イベントタイプ) で [EC2 AMI State Change] (EC2 AMI の状態変更) を選択します。

E. テンプレートをカスタマイズするには、[Edit pattern] (パターンを編集) を選択した上で、この例のイベントパターンに合わせた変更を行います。

ii. カスタムイベントパターンを指定するには、以下の操作を行います。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択します。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択します。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. ターゲットタイプ] では、AWSサービス] を選択します。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択します。

c. [Topic (トピック)] で、既存のトピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択します。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

7. [Review and create] (確認して作成) で、以下の操作を行います。

a. ルールの詳細を確認し、必要な場合は変更を行います。

b. [ルールを作成] を選択します。

詳細については、「Amazon EventBridge ユーザーガイド」で以下のトピックを参照してください。

- Amazon EventBridge イベント
- Amazon EventBridge のイベントパターン
- Amazon EventBridge ルール

Lambda 関数を作成する方法と、その Lambda 関数を実行する EventBridge ルールのチュートリアルについては、「AWS Lambda デベロッパーガイド」の「チュートリアル: EventBridge を使用して Amazon EC2 インスタンスの状態を口グに記録する」を参照してください。

AMI の請求情報について

インスタンスの起動時に選択できる Amazon マシンイメージ (AMI) は多数あり、さまざまなオペレーティングシステムプラットフォームと機能をサポートしています。AWS からの最終的な請求金額に対し、インスタンスの起動時に選択した AMI がどのように影響するかは、関連するオペレーティングシステムプラットフォームと請求情報を調べることで知ることができます。オンデマンド

または スポットインスタンス を起動するか、リザーブドインスタンス を購入する前に、この操作を行ってください。

以下、どのように AMI を事前調査することで、ニーズに最も適した AMI を選択できるかを示した例を 2 つ紹介します。

- スポットインスタンス では、AMI プラットフォームの詳細を使用して、AMI が スポットインスタンス でサポートされていることを確認できます。
- リザーブドインスタンス を購入する際、AMI プラットフォームの詳細にマップするオペレーティングシステムプラットフォーム (プラットフォーム) を選択するようにできます。

インスタンスの料金の詳細については、「Amazon EC2 料金表」を参照してください。

コンテンツ

AMI 請求情報フィールド

次のフィールドは、AMI に関連付けられた請求情報を提供します。

プラットフォームの詳細

AMI の請求コードに関連付けられたプラットフォームの詳細。例えば、`Red Hat Enterprise Linux` と指定します。

使用オペレーション

Amazon EC2 インスタンスのオペレーション、および AMI に関連付けられている請求コード。例えば、`RunInstances:0010` と指定します。[使用オペレーション] は、AWS のコストと使用状況レポート (CUR) の 明細項目/オペレーション列と、AWS Price List API に対応しています。

これらのフィールドは、Amazon EC2 コンソールの 「インスタンス」ページまたは「AMI」ページ、あるいは [describe-images] コマンドまたは [Get-EC2Image] コマンドによって返されるレスポンスで表示できます。

サンプルデータ: プラットフォーム別の使用オペレーション

次の表は、Amazon EC2 コンソールの 「インスタンス」ページまたは「AMI」ページ、あるいは [describe-images] コマンドまたは [Get-EC2Image] コマンドによって返されるレスポンスに表示される、プラットフォームの詳細と使用オペレーションの値の一部を一覧表示しています。

プラットフォームの詳細	使用オペレーション**
Linux/UNIX	RunInstances
Red Hat BYOL Linux	RunInstances:00g0
Red Hat Enterprise Linux	RunInstances:0010
Red Hat Enterprise Linux with HA	RunInstances:1010
Red Hat Enterprise Linux with SQL Server Standard and HA	RunInstances:1014
Red Hat Enterprise Linux with SQL Server Enterprise and HA	RunInstances:1110
Red Hat Enterprise Linux with SQL Server Standard	RunInstances:0014
Red Hat Enterprise Linux with SQL Server Web	RunInstances:0210
Red Hat Enterprise Linux with SQL Server Enterprise	RunInstances:0110
SQL Server Enterprise	RunInstances:0100

プラットフォームの詳細	使用オペレーション**
SQL Server Standard	RunInstances:0004
SQL Server Web	RunInstances:0200
SUSE Linux	RunInstances:000g
Ubuntu Pro	RunInstances:0g00
Windows	RunInstances:0002
Windows BYOL	RunInstances:0800
Windows with SQL Server Enterprise *	RunInstances:0102
Windows with SQL Server Standard *	RunInstances:0006
Windows with SQL Server Web *	RunInstances:0202

* 2 つのソフトウェアライセンスが 1 つの AMI に関連付けられている場合、[Platform details] フィールドには両方が表示されます。

** スポットインスタンス を実行している場合、AWS のコストと使用状況レポートの `lineitem/Operation` は、ここに記載されている [使用オペレーション] の値と異なる場合があります。例えば、`lineitem/Operation` に `RunInstances:0010:SV006` が表示されている場合は、Amazon EC2 が VPC ゾーン #6 で米国東部 (バージニア) で Red Hat Enterprise Linux スポットインスタンス時間を実行していることを示します。

AMI の請求と使用状況の詳細の検索

Amazon EC2 コンソールでは、[AMI] ページまたは [インスタンス] ページから AMI 請求情報を表示できます。また、AWS CLI またはインスタンスメタデータサービスを使用して、請求情報を検索することもできます。

請求書の AMI 料金を確認するには、次のフィールドが役立ちます。

- プラットフォームの詳細
- 使用操作
- AMI ID

AMI の請求情報の検索 (コンソール)

Amazon EC2 コンソールで AMI 請求情報を確認するには、次の手順に従います。

[AMI] ページから AMI 請求情報を調べる

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [AMIs] を選択し、AMI を選択します。
3. [詳細] タブで、[プラットフォーム詳細] と [使用操作] の値を確認します。

[インスタンス] ページから AMI の請求情報を調べる

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択してから、インスタンスを選択します。
3. [詳細] タブ (以前のバージョンのコンソールを使用している場合は [説明] タブ) で、[プラットフォームの詳細] と [使用オペレーション] の値を確認します。

AMI 請求情報フィールドの検索 (AWS CLI)

AWS CLI を使用して AMI 請求情報を検索するには、AMI ID を確認する必要があります。AMI ID がわからない場合は、インスタンスに対し describe-instances コマンドを使用することで取得できます。

AMI ID を見つけるには

インスタンス ID がわかっている場合は、describe-instances コマンドを実行することで、インスタンスの AMI ID を取得できます。

```
aws ec2 describe-instances --instance-ids i-123456789abcde123
```

このコマンドの出力の `ImageId` フィールドに AMI ID が表示されます。

```
..."Instances": [
{
    "AmiLaunchIndex": 0,
    "ImageId": "ami-0123456789EXAMPLE",
    "InstanceId": "i-123456789abcde123",
    ...
}]
```

AMI の請求情報を見つけるには

AMI ID がわかっている場合は、describe-images コマンドを使用して AMI プラットフォームと使用オペレーションの詳細を取得できます。

```
$ aws ec2 describe-images --image-ids ami-0123456789EXAMPLE
```

次の出力例は、PlatformDetails フィールドと UsageOperation フィールドを示しています。この例では、ami-0123456789EXAMPLE プラットフォームは Red Hat Enterprise Linux であり、使用操作と請求コードは RunInstances:0010 です。

```
{
    "Images": [
        {
            "VirtualizationType": "hvm",
            "Description": "Provided by Red Hat, Inc.",
            "Hypervisor": "xen",
            "EnaSupport": true,
            "SriovNetSupport": "simple",
            "ImageId": "ami-0123456789EXAMPLE",
            "State": "available",
            "BlockDeviceMappings": [
                {
                    "DeviceName": "/dev/sda1",
                    "Ebs": {
                        "SnapshotId": "snap-111222333444aaabb",
                        "DeleteOnTermination": true,
                        "VolumeType": "gp2",
                        "VolumeSize": 10,
                        "Encrypted": false
                    }
                }
            ],
            "Architecture": "x86_64",
```

```
            "ImageLocation": "123456789012/RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-
GP2",
            "RootDeviceType": "ebs",
            "OwnerId": "123456789012",
            "PlatformDetails": "Red Hat Enterprise Linux",
            "UsageOperation": "RunInstances:0010",
            "RootDeviceName": "/dev/sda1",
            "CreationDate": "2019-05-10T13:17:12.000Z",
            "Public": true,
            "ImageType": "machine",
            "Name": "RHEL-8.0.0_HVM-20190618-x86_64-1-Hourly2-GP2"
        }
    ]
}
```

請求書に記載されている AMI の請求を確認する

AWS のコストと使用状況レポート (CUR) で示されたインスタンスの請求情報が、そのインスタンスの起動に使用した AMI に関連付けられた請求情報と一致していることを確認することで、予定外のコストの発生を防ぐことができます。

請求情報を確認するには、CUR でインスタンス ID を見つけ、`lineitem/Operation` 列で対応する値を確認します。値は、AMI に関連付けられた [使用オペレーション] の値と一致する必要があります。

例えば、AMI `ami-0123456789EXAMPLE` には次の請求情報があります。

- プラットフォームの詳細 = `Red Hat Enterprise Linux`
- 使用オペレーション = `RunInstances:0010`

この AMI を使用してインスタンスを起動した場合は、CUR でインスタンス ID を検索し、`lineitem/Operation` 列で対応する値を確認できます。この例では、値は `RunInstances:0010` であることが必要です。

Amazon Linux

Amazon Linux は Amazon Web Services (AWS) が提供します。これは Amazon EC2 上で実行するアプリケーションのために安定した安全で高性能な実行環境を提供できるよう設計されています。これには、起動設定ツールおよび多くの人気 AWS ライブラリやツールなど、AWS との統合を容易にするパッケージも含まれています。AWS は、Amazon Linux を実行しているすべてのインスタンスの

現行のセキュリティとメンテナンスの更新を提供します。CentOS (および同様のディストリビューション) で開発された多くのアプリケーションは、Amazon Linux で実行されます。

コンテンツ

Amazon Linux の入手可能性

AWS は AL2023、Amazon Linux 2、Amazon Linux AMI を提供します。別の Linux ディストリビューションから Amazon Linux に移行する場合、AL2023 に移行することをお勧めします。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

Amazon Linux の詳細については、「AL2023」、「Amazon Linux 2」、「Amazon Linux AMI」を参照してください。

Amazon Linux コンテナイメージについては、「Amazon Elastic コンテナレジストリユーザーガイド」の「Amazon Linux コンテナイメージ」を参照してください。

Amazon Linux インスタンスへの接続

デフォルトでは、Amazon Linux はリモートルート Secure Shell (SSH) を許可しません。また、パスワード認証は、パスワードのブルートフォース攻撃を防ぐために無効になっています。Amazon Linux インスタンスへの SSH ログインを有効にするには、起動時にキーペアをインスタンスに提供する必要があります。インスタンスを起動するときに使用するセキュリティグループで、SSH アクセスを許可するよう設定する必要もあります。デフォルトでは、SSH を使用してリモートログインできる唯一のアカウントは ec2-user; です。このアカウントには sudo 特権もあります。リモートルートログインを有効にする場合は、キーペアおよびセカンダリユーザーを使用する場合よりも安全性が低いことに注意してください。

Amazon Linux イメージの特定

各イメージには、そのイメージを特定する一意の `/etc/image-id` ファイルが含まれています。このファイルには、イメージに関する次の情報が含まれています。

- `image_name`、`image_version`、`image_arch` - 画像の構築時に Amazon が使用したビルド recipe からの値
- `image_stamp` – 画像の作成中に生成されたランダムで一意の 16 進値
- `image_date` - YYYYMMDDhhmmss 形式で画像を作成した UTC 時間。
- `recipe_name`、`recipe_id` – 画像を作成時に Amazon を使用したビルド recipe の名前と ID

Amazon Linux には、インストールされている現在のリリースを示す `/etc/system-release` ファイルが含まれています。このファイルは、yumを使用して更新され、`system-release`RPM パッケージマネージャー (RPM) の一部です。

Amazon Linux には、Common Platform Enumeration (CPE) 仕様に準拠した`/etc/system-release`の機械可読バージョンも含まれています。`/etc/system-release-cpe`を参照してください。

Amazon Linux 2

現在のバージョンの Amazon Linux 2 用の `/etc/image-id` の例を以下に示します。

```
[ec2-user ~]$ cat /etc/image-id
```

```
image_name="amzn2-ami-hvm"
image_version="2"
image_arch="x86_64"
image_file="amzn2-ami-hvm-2.0.20180810-x86_64.xfs.gpt"
image_stamp="8008-2abd"
image_date="20180811020321"
recipe_name="amzn2 ami"
recipe_id="c652686a-2415-9819-65fb-4dee-9792-289d-1e2846bd"
```

現在のバージョンの Amazon Linux 2 用の `/etc/system-release` の例を以下に示します。

```
[ec2-user ~]$ cat /etc/system-release
Amazon Linux 2
```

次は、Amazon Linux 2 の `/etc/os-release` の例です。

```
[ec2-user ~]$ cat /etc/os-release
NAME="Amazon Linux"
VERSION="2"
ID="amzn"
ID_LIKE="centos rhel fedora"
VERSION_ID="2"
PRETTY_NAME="Amazon Linux 2"
ANSI_COLOR="0;33"
CPE_NAME="cpe:2.3:o:amazon:amazon_linux:2"
HOME_URL="https://amazonlinux.com/"
```

Amazon Linux AMI

現在の Amazon Linux AMI 用の `/etc/image-id` の例を以下に示します。

```
[ec2-user ~]$ cat /etc/image-id
image_name="amzn-ami-hvm"
image_version="2018.03"
image_arch="x86_64"
image_file="amzn-ami-hvm-2018.03.0.20180811-x86_64.ext4.gpt"
image_stamp="cc81-f2f3"
image_date="20180811012746"
recipe_name="amzn ami"
recipe_id="5b283820-dc60-a7ea-d436-39fa-439f-02ea-5c802dbd"
```

現在の Amazon Linux AMI 用の `/etc/system-release` の例を以下に示します。

```
[ec2-user ~]$ cat /etc/system-release
Amazon Linux AMI release 2018.03
```

Amazon Linux 2 AMI ブートモード

Amazon Linux 2 AMI には、ブートモードパラメータが設定されていません。Amazon Linux 2 AMI から起動されたインスタンスは、インスタンスタイプのデフォルトのブートモード値に従います。詳細については、「ブートモード」を参照してください。

AWS コマンドラインツール

AWS Command Line Interface (AWS CLI) は、コマンドラインシェルでコマンドを使用して AWS のサービス とやり取りするための一貫性のあるインターフェイスを提供するオープンソースツールです。詳細については、AWS Command Line Interface ユーザーガイドの「AWS Command Line Interface とは」を参照してください。

Amazon Linux 2 および Amazon Linux AMI には、AWS CLI のバージョン 1 がプリインストールされています。Amazon Linux の現在のリリースである AL2023 には、AWS CLI のバージョン 2 がプリインストールされています。AL2023 で AWS CLI を使用する方法については、「AL2023 ユーザーガイド」の「AL2023 の使用を開始する」を参照してください。

パッケージリポジトリ

この情報は、Amazon Linux 2 と Amazon Linux AMI に適用されます。AL2023 の詳細については、「AL2023 ユーザーガイド」の「パッケージの管理とシステム更新の実行」を参照してください。

Amazon Linux 2 および Amazon Linux AMI は、各 Amazon EC2 AWS リージョンでホストされているオンラインパッケージリポジトリと一緒に使用するように設計されています。リポジトリはすべてのリージョンに存在し、yum 更新ツールを使用してアクセスできます。各リージョンでリポジトリをホストしているため、データ転送料金なしで、更新を迅速にデプロイできます。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

インスタンスでデータまたはカスタム設定を保存する必要がない場合は、現行の Amazon Linux 2 AMI を使用して新しいインスタンスを開始できます。インスタンスでデータまたはカスタム設定を保存する必要がある場合は、Amazon Linux パッケージリポジトリを介してこれらのインスタンスを維持できます。これらのリポジトリには、更新されたすべてのパッケージが含まれます。実行中のインスタンスにこれらの更新を適用するよう選択できます。新しいバージョンの AMI がリリースされても、古いバージョンの AMI と更新パッケージは引き続き利用できます。

> **Note**
>
> EC2 インスタンスで、インターネットにアクセスせずにパッケージを更新およびインストールするには、「Amazon Linux、Amazon Linux 2、または AL2023 を実行している Amazon EC2 インスタンスで、インターネットにアクセスせずに yum の更新や、パッケージのインストールを行う方法」を参照してください。

パッケージをインストールするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo yum install package
```

Amazon Linux に必要なアプリケーションが含まれていない場合は、Amazon Linux インスタンスにアプリケーションを直接インストールできます。Amazon Linux は RPM と yum をパッケージ管理に使用します。これは新しいアプリケーションをインストールする最も簡単な方法です。多くのアプリケーションが中央の Amazon Linux リポジトリで利用可能なので、最初にアプリケーションがそのリポジトリで利用できるかどうかを確認する必要があります。これらのアプリケーションは、Amazon Linux インスタンスに簡単に追加できます。

実行中の Amazon Linux インスタンスにアプリケーションをアップロードするには、scp または sftp を使用し、インスタンスにログインしてアプリケーションを設定します。組み込みの cloud-init パッケージから PACKAGE_SETUP アクションを使用して、インスタンスの起動時にアプリケーションをアップロードすることもできます。詳細については、cloud-init を参照してください。

セキュリティ更新プログラム

セキュリティの更新は、パッケージリポジトリと更新された AMI を使用して提供されます。セキュリティアラートは、Amazon Linux セキュリティセンターで公開されます。AWS セキュリティポリシーの詳細については、またはセキュリティの問題を報告するには、「AWS クラウドのセキュリティ」を参照してください。

Amazon Linux および Amazon Linux 2 は、起動時にクリティカルまたは重要なセキュリティ更新プログラムをダウンロードおよびインストールするよう設定されています。この設定にはカーネルの更新は含まれません。

AL2023 では、この設定が Amazon Linux および Amazon Linux 2 から変更されました。AL2023 のセキュリティアップデートの詳細については、「AL2023 ユーザーガイド」の「セキュリティ更新プログラムと機能」を参照してください。

起動後にユースケースに必要な更新を行うことをお勧めします。例えば、起動時にすべての更新 (セキュリティ更新だけでなく) を適用したり、各更新を評価してシステムに適用可能なもののみを適用することができます。これは、cloud-init 設定 `repo_upgrade` を使用して制御されます。次の cloud-init 設定のスニペットは、インスタンス初期化に渡すユーザーデータテキストで設定を変更する方法を示しています。

```
#cloud-config
repo_upgrade: security
```

`repo_upgrade` の有効な値は次のとおりです。

`critical`

まだ適用されていない緊急のセキュリティ更新プログラムを適用します。

`important`

まだ適用されていない緊急および重要なセキュリティ更新プログラムを適用します。

`medium`

まだ適用されていない緊急、重要、中レベルのセキュリティ更新プログラムを適用します。

`low`

低レベルのセキュリティ更新プログラムを含む、まだ適用されていないセキュリティ更新プログラムをすべて適用します。

`security`

Amazon によってセキュリティ更新としてマークされた保留中のクリティカルまたは重要な更新を適用します。

`bugfix`

Amazon によってバグフィックスとしてマークされた更新を適用します。バグフィックスは大きなサイズの更新セットで、セキュリティ更新および他のさまざまな小さなバグに対する修正が含まれます。

`all`

分類に関係なく、使用できる適切な更新すべてを適用します。

`none`

起動時に更新をインスタンスに適用しません。

`repo_upgrade` のデフォルトの設定は security です。つまり、ユーザーデータに異なる値を指定しない場合、デフォルトでは、Amazon Linux はその時点でインストールされているパッケージの起動時に、セキュリティ更新を実行します。Amazon Linux は、インストール済みのパッケージに更新がある場合も、`/etc/motd` ファイルを使用して、ログイン時に利用可能な更新の数を一覧表示して通知します。これらの更新をインストールするには、インスタンスで sudo yum upgrade を実行する必要があります。

リポジトリの設定

Amazon Linux および Amazon Linux 2 の場合、AMI は、セキュリティ更新プログラムを除き、AMI の作成時に利用可能なパッケージのスナップショットです。元の AMI にはないが実行時にインストールされるパッケージは、利用可能な最新バージョンです。Amazon Linux 2 で利用可能な最新のパッケージを入手するには、yum update -y を実行します。

> **トラブルシューティングのヒント**
>
> `t3.nano` などのナノインスタンスタイプで yum update の実行中に `cannot allocate memory` エラーが発生した場合は、更新を有効にするためにスワップ領域を割り当てる必要がある場合があります。

AL2023 では、リポジトリ設定が Amazon Linux および Amazon Linux 2 から変更されました。AL2023 リポジトリの詳細については、「パッケージおよびオペレーションシステムアップデートの管理」を参照してください。

AL2023 までのバージョンは、Amazon Linux の 1 つのマイナーバージョンから次のバージョンにローリングする更新の継続的なフロー (ローリングリリースとも呼ばれます) を提供するように設定

されていました。ベストプラクティスとして、現在ご利用の AMI を、利用可能な最新の AMI に更新することをお勧めします。2017.09 などの古いバージョンを使用する Amazon Linux AMI を起動しないでください。

インプレースアップグレードは、Amazon Linux から Amazon Linux 2、または Amazon Linux 2 から AL2023 など、Amazon Linux のメジャーバージョン間ではサポートされていません。詳細については、「Amazon Linux の入手可能性」を参照してください。

Amazon Linux での lock-on-launch の使用

ローリングリリースを無効にするには、lock-on-launch 機能を有効にします。Lock-on-launch 機能は、インスタンスが、指定したリリースの AMI からのみ更新を受け取れるようにロックします。例えば、2018.03 AMI に移行する準備ができるまでは、2017.09 AMI を起動したときに、2018.03 AMI より前にリリースされた更新のみを受け取るようにすることができます。

> **Important**
>
> lock-on-launch 機能を有効にし、最新ではないバージョンのリポジトリを選択すると、その後の更新は受信されません。ローリングリリースを受け取るには、最新の AMI を使用するか、必ず最新とされているリポジトリで AMI を更新する必要があります。

新しいインスタンスで lock-on-launch 機能を有効にするには、次のユーザーデータを cloud-init に渡してインスタンスを起動します。

```
#cloud-config
repo_releasever: 2017.09
```

既存のインスタンスを現在の AMI バージョンにロックするには

1. 編集 `/etc/yum.conf`。
2. `releasever=latest` をコメントアウトします。
3. キャッシュをクリアするには、yum clean all を実行します。

Extras library (Amazon Linux 2)

Amazon Linux 2 では、Extras Library を使用してアプリケーションおよびソフトウェア更新をインスタンスにインストールできます。このようなソフトウェア更新は、トピックと呼ばれます。特定の

バージョンのトピックをインストールしたり、最新バージョンを使用するためにバージョン情報を省略したりすることができます。

使用可能なトピックのリストを表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ amazon-linux-extras list
```

トピックを有効にし、パッケージの最新バージョンをインストールして最新の状態を維持するには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo amazon-linux-extras install topic
```

トピックを有効にし、パッケージの特定のバージョンをインストールして安定性を確保するには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo amazon-linux-extras install topic=version topic=version
```

トピックからインストールされたパッケージを削除するには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo yum remove $(yum list installed | grep amzn2extra-topic | awk
 '{ print $1 }')
```

> **Note**
>
> このコマンドでは、extra の依存関係としてインストールされたパッケージは削除されません。

トピックを無効にし、yum パッケージマネージャーからパッケージにアクセスできないようにするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo amazon-linux-extras disable topic
```

> **Important**
>
> このコマンドは、上級ユーザーを対象としています。このコマンドの使用が不適切な場合、パッケージ互換性の競合が発生する可能性があります。

Amazon Linux 2 でサポートされているカーネル

[Supported kernel versions] (サポートされているカーネルバージョン)

現在、Amazon Linux 2 (AL2) AMI はカーネルバージョン 5.10 がデフォルトで提供されていますが、バージョン 4.14 および 5.10 も利用可能です。また、エクストラリポジトリを使用して AL2 のカーネルをバージョン 5.15 にアップグレードするオプションもあります。5.15 へのアップグレードには、新しいカーネルを有効にするに再起動が必要であることに注意してください。ユースケースにアップグレードが必要かどうかを判断する前に、AL2 のカーネルバージョン 5.15 の新機能と制限を確認してください。ライブパッチのサポートが必要な場合、カーネル 5.10 を使用した AL2 AMI を使用することをお勧めします。

AL2023 AMI はカーネルバージョン 6.1 で使用できます。詳細については、「AL2023 ユーザーガイド」の「Amazon Linux 2 からの AL2023 のカーネルの変更点」を参照してください。

[New features in kernel 5.15] (カーネル 5.15 の新機能)

- [Kernel-based Virtual Machine] (カーネルベースの仮想マシン) (KVM) は新しい x86 TDP MMU にデフォルト設定され、AMD SVM 5 レベルのページングが追加されて、元の KVM x86 MMU コードと比較して並列性とスケーラビリティが向上します。
- OverlayFS はパフォーマンスが向上し、不変/追加/同期/noatime 属性のコピーも処理できるようになりました。
- EXT4 のための新しい最適化と改良が追加されました。例えば、大きな並列切り捨てやファイル削除のボトルネックを解消するための新しい orphan_file 機能の追加や、DISCARD の動作が遅いデバイスを支援するために JBD2 コミットスレッドから DISCARD 作業を移動し、JBD2 コミット KThread をブロックしないようにしました。
- XFS の新しい最適化と改善が追加されました。例えば、ディレクトリツリーの削除時間を改善する CPU ごとのバックグラウンドスレッドでのバッチ inode のアクティブ化や、大量のメタデータ更新の処理に関するパフォーマンスに役立つパイプライン化の有効化などがあります。
- DAMON は、プロアクティブなメモリ再利用とパフォーマンス分析のためのデータアクセス監視フレームワークとしてよりよくサポートされています。

[Limitations for kernel 5.15] (カーネル 5.15 の制限事項)

- LustreFSx はサポートされていません (サポートは準備中)。
- カーネルのライブパッチはサポートされていません。

カーネル 5.15 のインストール手順

次のコマンドを使用して、カーネル 4.14 で Amazon Linux 2 AMI とカーネル 5.10 の AL2 AMI の両方からカーネル 5.15 にアップグレードできます。

1. amazon-linux-extras の kernel-5.15 トピックをを有効にしてホストにカーネル 5.15 をインストールします。

```
sudo amazon-linux-extras install kernel-5.15
```

2. インストールされたカーネル 5.15 でホストを再起動します。

```
sudo reboot
```

3. システムのカーネルバージョンを確認します。

```
uname -r
```

[Support Timeframe] (サポート時間枠)

Amazon Linux 2 (4.14、5.10、5.15) で利用可能なすべての Linux カーネルは、Amazon Linux 2 AMI が標準サポートの終了に達するまでサポートされます。

[Live patching support] (ライブパッチサポート)

Amazon Linux 2 kernel version	Kernel live patching supported
4.14	Yes
5.10	Yes
5.15	No

参照のためのソースパッケージへのアクセス

Amazon Linux で提供されているツールを使用して、インスタンスにインストールしたパッケージソースを参照するために表示できます。ソースパッケージは、Amazon Linux およびオンラインパッケージリポジトリに含まれるすべてのパッケージで利用できます。インストールするソースパッケー

ジのパッケージ名を確認し、yumdownloader --source コマンドを使用して実行中のインスタンス内にソースを表示します。次に例を示します。

```
[ec2-user ~]$ yumdownloader --source bash
```

ソース RPM は解凍できます。そして、標準の RPM ツールを使用して、参照するためにソースツリーを表示できます。デバッグが完了したら、パッケージを利用できます。

cloud-init

cloud-init パッケージは、Canonical によって構築されたオープンソースアプリケーションであり、Amazon EC2 などのクラウドコンピューティング環境で Linux イメージをブートストラップするときに使用されます。Amazon Linux には、カスタマイズされたバージョンの cloud-init が含まれています。cloud-init のカスタマイズ版では、起動時のインスタンスに対するアクションを指定することができます。インスタンスの起動時に、ユーザーデータフィールドを使用して必要なアクションを cloud-init に渡すことができます。つまり、さまざまなユースケースに対して共通の AMI を使用し、起動時にその AMI を動的に設定できます。Amazon Linux は、ec2 ユーザーアカウントの初期設定を実行するためにも cloud-init を使用します。

詳細については、「cloud-init ドキュメント」を参照してください。

Amazon Linux は、`/etc/cloud/cloud.cfg.d` と `/etc/cloud/cloud.cfg` にある cloud-init アクションを使用します。独自の cloud-init アクションファイルを `/etc/cloud/cloud.cfg.d` に作成することができます。このディレクトリ内のすべてのファイルは、cloud-init で読み取られます。それらは辞書と同じ順序に読み取られ、後のファイルは以前のファイルの値を上書きします。

cloud-init パッケージは、起動時にインスタンスのこれらの (およびその他の) 共通の設定タスクを実行します。

- デフォルトのロケールを設定。
- ホスト名を設定。
- ユーザーデータの解析と処理。
- ホスト プライベート SSH キーの生成。
- 容易にログインおよび管理できるように、ユーザーのパブリック SSH キーを `.ssh/authorized_keys` に追加する。
- パッケージ管理のためにリポジトリを準備する
- ユーザーデータで定義されたパッケージアクションの処理。

- ユーザーデータにあるユーザースクリプトの実行。
- インスタンスストアボリュームをマウントする (該当する場合)
  - デフォルトでは、ephemeral0 インスタンスストアボリュームがある場合は /media/ephemeral0 にマウントされ、有効なファイルシステムが含まれます。それ以外の場合は、マウントされません。
  - デフォルトでは、インスタンスに関連付けられたスワップボリュームがマウントされます (m1.smallおよびc1.mediumインスタンスタイプの場合のみ)。
  - 次の cloud-init ディレクティブを使用して、デフォルトのインスタンスストアボリュームマウントを上書きすることができます。

    ```
    #cloud-config
    mounts:
    - [ ephemeral0 ]
    ```

    マウントをより詳細にコントロールするには、cloud-init ドキュメントの「マウント」を参照してください。
  - TRIM をサポートするインスタンスストアボリュームは、インスタンスの起動時にはフォーマットされないため、マウントする前にパーティション化してフォーマットする必要があります。詳細については、インスタンスストアボリュームの TRIM のサポート を参照してください。disk_setup モジュールを使用して、起動時にインスタンスストアボリュームをパーティションおよびフォーマットすることができます。詳細については、cloud-init ドキュメントの「Disk Setup」を参照してください。

サポートされているユーザーデータ形式

cloud-init パッケージは、さまざまな形式のユーザーデータを処理できます。

- Gzip
  - ユーザーデータが gzip で圧縮されている場合、cloud-init はデータを解凍し、適切に処理します。
- MIME マルチパート
  - MIME マルチパートファイルを使用して、複数のデータタイプを指定できます。例えば、ユーザーデータスクリプトとクラウド設定タイプの両方を指定できます。マルチパートファイルのパートの形式が、サポートされている形式のいずれかの場合、そのパートは cloud-init で処理できます。

- Base64 デコード
  - ユーザーデータが base64 でエンコードされている場合、cloud-init は、デコードされたデータをサポートされているタイプのいずれかとして認識できるか確認します。デコードされたデータを認識できる場合、データをデコードし、適切に処理します。認識できない場合、base64 データは変更されません。
- ユーザーデータスクリプト
  - 「#!」または「Content-Type: text/x-shellscript」で始まります。
  - このスクリプトは、初回の起動サイクル時に /etc/init.d/cloud-init-user-scripts によって実行されます。これは起動プロセスの後半 (初期設定アクションが実行された後) に実行されます。
- インクルードファイル
  - 「#include」または「Content-Type: text/x-include-url」で始まります。
  - このコンテンツはインクルードファイルです。ファイルには URL の一覧 (1行に1つの URL) が含まれます。各 URL が読み取られ、そのコンテンツが同じルールセットを使用して渡されます。URL から読み取られたコンテンツは gzip 圧縮され、MIME マルチパート、またはプレーンテキスト形式になります。
- クラウド Config データ
  - 「#cloud-config」または「Content-Type: text/cloud-config」で始まります。
  - このコンテンツはクラウド設定データです。サポートされている設定形式のコメント付きサンプルについては、例を参照してください。
- Upstart ジョブ (Amazon Linux 2 ではサポートされていません)
  - 「#upstart-job」または「Content-Type: text/upstart-job」で始まります。
  - このコンテンツは /etc/init のファイルに格納され、upstart は他の upstart ジョブごとにコンテンツを消費します。
- クラウドブートフック
  - 「#cloud-boothook」または「Content-Type: text/cloud-boothook」で始まります。
  - このコンテンツはブートフックデータです。このデータは /var/lib/cloud にあるファイルに保存され、すぐに実行されます。
  - これは最初に使用可能な [hook] (フック) です。1 回だけ実行するためのメカニズムはありません。ブートフックは自身でこの点に対処する必要があります。環境変数 INSTANCE_ID でインスタンス ID が指定されています。この変数を使用して、インスタンスあたり1つのブートフックデータのセットを提供します。

Amazon Linux 通知のサブスクライブ

新しい Amazon Linux AMI がリリースされた場合に通知を受取るには、Amazon SNS を使用して申し込みできます。

AL2023 の通知にサブスクライブする方法については、「AL2023 ユーザーガイド」の「新しい更新の通知を受け取る」を参照してください。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

Amazon Linux の通知をサブスクライブするには

1. Amazon SNS コンソール (https://console.aws.amazon.com/sns/v3/home) を開きます。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。購読する SNS 通知が作成されたリージョンを選択する必要があります。
3. ナビゲーションペインで、[Subscriptions]、[Create subscription] の順に選択します。
4. [サブスクリプションの作成] ダイアログボックスで、次の操作を行います。
   a. [Amazon Linux 2] [トピックの ARN] には、以下の Amazon リソースネーム (ARN) をコピーして貼り付けます。**`arn:aws:sns:us-east-1:137112412989:amazon-linux-2-ami-updates`**
   b. [Amazon Linux] [トピックの ARN] には、以下の Amazon リソースネーム (ARN) **`arn:aws:sns:us-east-1:137112412989:amazon-linux-ami-updates`** をコピーして貼り付けます。
   c. [Protocol] で [Email] を選択します。
   d. [エンドポイント] に、通知を受信するために使用できる E メールアドレスを入力します。
   e. [サブスクリプションを作成] を選択します。
5. 「AWS 通知 - サブスクリプションの確定」という件名の確認メールを受け取ります。メールを開いて [Confirm subscription] を選択して受信登録を完了します。

AMI がリリースされるごとに、対応するトピックの受信者に通知が送信されます。このような通知を停止するには、以下の手順を使用してサブスクリプション解除します。

Amazon Linux の通知の受信登録を解除するには

1. Amazon SNS コンソール (https://console.aws.amazon.com/sns/v3/home) を開きます。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。SNS 通知が作成されたリージョンを使用する必要があります。
3. ナビゲーションペインで、[サブスクリプション] を選択し、サブスクリプションを選択したら、[アクション]、[サブスクリプションの削除] の順に選択します。
4. 確認を求めるメッセージが表示されたら、[削除] を選択します。

Amazon Linux AMI の SNS メッセージ形式

SNS メッセージのスキーマは次のとおりです。

```
{
    "description": "Validates output from AMI Release SNS message",
    "type": "object",
    "properties": {
        "v1": {
            "type": "object",
            "properties": {
                "ReleaseVersion": {
                    "description": "Major release (ex. 2018.03)",
                    "type": "string"
                },
                "ImageVersion": {
                    "description": "Full release (ex. 2018.03.0.20180412)",
                    "type": "string"
                },
                "ReleaseNotes": {
                    "description": "Human-readable string with extra information",
                    "type": "string"
                },
                "Regions": {
                    "type": "object",
                    "description": "Each key will be a region name (ex. us-east-1)",
                    "additionalProperties": {
                        "type": "array",
                        "items": {
```

```
                            "type": "object",
                            "properties": {
                                "Name": {
                                    "description": "AMI Name (ex. amzn-ami-
hvm-2018.03.0.20180412-x86_64-gp2)",
                                    "type": "string"
                                },
                                "ImageId": {
                                    "description": "AMI Name (ex.ami-467ca739)",
                                    "type": "string"
                                }
                            },
                            "required": [
                                "Name",
                                "ImageId"
                            ]
                        }
                    }
                }
            },
            "required": [
                "ReleaseVersion",
                "ImageVersion",
                "ReleaseNotes",
                "Regions"
            ]
        }
    },
    "required": [
        "v1"
    ]
}
```

Amazon Linux 2 を仮想マシンとしたオンプレミスでの実行

オンプレミスの開発とテストには、Amazon Linux 2 仮想マシン (VM) イメージを使用します。サポートされている仮想化プラットフォームごとに異なる Amazon Linux 2 VM イメージをお使いいただけます。サポートされているプラットフォームのリストは、[Amazon Linux 2 virtual machine images] (Amazon Linux 2 仮想マシンイメージ) ページで確認できます。

サポートされているいずれかの仮想プラットフォームで Amazon Linux 2 仮想マシンイメージを使用するには、次の操作を行います。

- ステップ 1: seed.iso 起動イメージを準備する
- ステップ 2: Amazon Linux 2 VM イメージのダウンロード
- ステップ 3: 新しい VM を起動して接続する

ステップ 1: **seed.iso** 起動イメージを準備する

seed.iso 起動イメージには、新しい VM の起動に必要な初期設定情報 (例: ネットワーク設定、ホスト名、ユーザーデータ) が含まれます。

> **Note**
>
> seed.iso 起動イメージには、VM の起動に必要な設定情報のみ含まれています。Amazon Linux 2 オペレーティングシステムファイルは含まれていません。

seed.iso 起動イメージを生成するには、2 つの設定ファイルが必要です。

- meta-data – このファイルには、VM のホスト名と静的ネットワーク設定が含まれます。
- user-data – このファイルはユーザーアカウントを設定し、パスワード、キーペア、アクセスメカニズムを指定します。デフォルトでは、Amazon Linux 2 VM イメージでは、ec2-user のユーザーアカウントを作成します。デフォルトのユーザーアカウントのパスワードを設定するには、user-data 設定ファイルを使用します。

seed.iso 起動ディスクを作成するには

1. seedconfig という名前の新しいフォルダを作成し、そのフォルダに移動します。
2. meta-data 設定ファイルを作成します。
   a. meta-data という名前の新しいファイルを作成します。
   b. 任意のテキストエディタを使用して meta-data ファイルを開き、以下を追加します。

```
local-hostname: vm_hostname
# eth0 is the default network interface enabled in the image. You can configure
 static network settings with an entry like the following.
network-interfaces: |
  auto eth0
  iface eth0 inet static
  address 192.168.1.10
```

```
  network 192.168.1.0
  netmask 255.255.255.0
  broadcast 192.168.1.255
  gateway 192.168.1.254
```

vm_hostname を任意の VM ホスト名に置き換え、必要に応じてネットワーク設定を行います。

c. meta-data 設定ファイルを保存して閉じます。

VM ホスト名 (meta-data) を指定し、デフォルトのネットワークインターフェイス (amazonlinux.onprem) を構成し、必要なネットワークデバイスの静的 IP アドレスを指定する eth0 設定ファイルの例については、サンプルの Seed.iso ファイルを参照してください。

3. user-data 設定ファイルを作成します。

a. user-data という名前の新しいファイルを作成します。

b. 任意のテキストエディタを使用して user-data ファイルを開き、以下を追加します。

```
#cloud-config
#vim:syntax=yaml
users:
# A user by the name `ec2-user` is created in the image by default.
  - default
chpasswd:
  list: |
    ec2-user:plain_text_password
# In the above line, do not add any spaces after 'ec2-user:'.
```

plain_text_password を、デフォルトの ec2-user ユーザーアカウントの任意のパスワードに置き換えます。

c. (オプション) デフォルトでは、cloud-init は VM が起動される度にネットワーク設定に適用されます。ブート起動時の cloud-init によるネットワーク設定の適用を無効にし、最初の起動時のネットワーク設定を保持するには、以下を追加します。

```
# NOTE: Cloud-init applies network settings on every boot by default. To retain
 network settings
# from first boot, add the following 'write_files' section:
write_files:
  - path: /etc/cloud/cloud.cfg.d/80_disable_network_after_firstboot.cfg
```

```
    content: |
      # Disable network configuration after first boot
      network:
        config: disabled
```

d. user-data 設定ファイルを保存して閉じます。

また、他のユーザーアカウントを作成して、アクセスメカニズム、パスワード、およびキーペアを指定することもできます。サポートされるディレクティブについては、「モジュール参照」を確認してください。3 人のユーザーを追加で作成し、デフォルトの user-data ユーザーアカウントのカスタムパスワードを指定する ec2-user ファイルの例については、サンプル Seed.iso ファイルを参照してください。

4. seed.iso および meta-data 設定ファイルを使用して、user-data 起動イメージを作成します。

Linux の場合は、genisoimage などのツールを使用します。seedconfig フォルダに移動し、次のコマンドを実行します。

```
$ genisoimage -output seed.iso -volid cidata -joliet -rock user-data meta-data
```

macOS の場合は、hdiutil などのツールを使用します。seedconfig フォルダの 1 つ上に移動し、次のコマンドを実行します。

```
$ hdiutil makehybrid -o seed.iso -hfs -joliet -iso -default-volume-name cidata
 seedconfig/
```

ステップ 2: Amazon Linux 2 VM イメージのダウンロード

サポートされている仮想化プラットフォームごとに異なる Amazon Linux 2 VM イメージをお使いいただけます。サポートされているプラットフォームのリストを表示し、選択したプラットフォームに適した VM イメージを [Amazon Linux 2 virtual machine images] (Amazon Linux 2 仮想マシンイメージ) ページからダウンロードできます。

ステップ 3: 新しい VM を起動して接続する

新しい VM を起動して接続するには、seed.iso 起動イメージ (ステップ 1 で作成済み) と Amazon Linux 2 VM イメージ (ステップ 2 でダウンロード済み) が必要です。ステップは、選択した VM プラットフォームによって異なります。

VMware vSphere

VMware の VMイメージは、OVF 形式で提供されます。

VMware vSphere を使用して VM を起動するには

1. `seed.iso` ファイルの新しいデータストアを作成するか、既存のデータストアに追加します。
2. OVF テンプレートをデプロイしますが、VM はまだ起動しないでください。
3. ナビゲータパネルで、新しい仮想マシンを右クリックし、[設定の編集] を選択します。
4. [Virtual Hardware (仮想ハードウェア)] タブの [New device (新しいデバイス)] で、[CD/DVD Drive (CD/DVD ドライブ)] を選択し、[追加] を選択します。
5. [New CD/DVD Drive (新しい CD/DVD ドライブ)] で、[Datastore ISO File (データストア ISO ファイル)] を選択します。`seed.iso` ファイルを追加したデータストアを選択し、`seed.iso` ファイルを参照して選択し、[OK] を選択します。
6. [新しい CD/DVD ドライブ] で [接続]、[OK] の順にクリックします。

データストアを VM に関連付けると、起動できるようになります。

KVM virt-manager

KVM を使用して VM を起動するには

1. [Create new VM (新しい VM の作成)] ウィザードを開きます。
2. ステップ 1 で、[Import existing disk image (既存のディスクイメージのインポート)] を選択します。
3. ステップ 2 で、VM イメージを参照して選択します。[OS type] (OS タイプ) と [Version] (バージョン) では、[Linux] と [Red Hat Enterprise Linux 7.0] をそれぞれ選択します。
4. ステップ 3 では、RAM の容量と CPU の数を指定します。
5. ステップ 4 では、新しい VM の名前を入力し、[Customize configuration before install (インストール前に構成をカスタマイズする)] を選択し、[完了] を選択します。
6. 仮想マシンの [構成] ウィンドウで、[Add Hardware (ハードウェアの追加)] を選択します。
7. [Add New Virtual Hardware (新しい仮想ハードウェアの追加)] ウィンドウで、[ストレージ] を選択します。
8. [ストレージ] 構成で、[Select or create custom storage (カスタムストレージの選択または作成)] を選択します。[デバイスタイプ] で、[CDROM デバイス] を選択します。[管理]、

[Browse Local (ローカルを参照)] の順に選択し、`seed.iso` ファイルに移動して選択します。[Finish] を選択します。

9. [Begin Installation (インストールを開始)] を選択します。

Oracle VirtualBox

Oracle VirtualBox を使用して VM を起動するには

1. Oracle VirtualBoxを開き、[New (新規)] を選択します。
2. [Name] (名前) には、仮想マシン用のわかりやすい名前を入力します。[Type] (タイプ) と [Version] (バージョン) では、[Linux] と [Red Hat (64-bit)] をそれぞれ選択します。[続行] をクリックします。
3. [Memory size (メモリサイズ)] で、仮想マシンに割り当てるメモリの量を指定し、[Continue (続行)] を選択します。
4. [Hard disk (ハードディスク)] で、[Use an existing virtual hard disk file (既存の仮想ハードディスクファイルを使用する)] を選択し、VM イメージを参照して開き、[Create (作成)] を選択します。
5. VM を起動する前に、仮想マシンの仮想オプティカルドライブに `seed.iso` ファイルをロードする必要があります。

   a. 新しい VM を選択し、[設定]、[ストレージ] の順に選択します。
   b. [Storage Devices (ストレージデバイス)] リストの [Controller: IDE (コントローラー: IDE)] で、空のオプティカルドライブを選択します。
   c. オプティカルドライブの [属性] セクションで、参照ボタンを選択し、[Choose Virtual Optical Disk File (仮想オプティカルディスクファイルを選択する)] を選択し、`seed.iso` ファイルを選択します。[OK] を選択して変更を適用し、[Settings (設定)] を閉じます。

仮想オプティカルドライブに `seed.iso` ファイルを追加すると、VM を起動できます。

Microsoft Hyper-V

Microsoft Hyper-V 用の VM イメージは zip ファイルに圧縮されます。.zip ファイルの内容を展開する必要があります。

Microsoft Hyper-V を使用して VM を起動するには

1. [New Virtual Machine Wizard (新しい仮想マシンウィザード)] を開きます。
2. 世代を選択するよう求められたら、[Generation 1 (第 1 世代)] を選択します。
3. ネットワークアダプタの構成を求めるメッセージが表示されたら、[接続] に [外部] を選択します。
4. 仮想ハードディスクを接続するかどうかを確認するメッセージが表示されたら、[Use an existing virtual hard disk (既存の仮想ハードディスクを使用する)]、[参照] の順に選択し、VM イメージに移動して選択します。[完了] を選択し、VM を作成します。
5. 新しい VM を右クリックし、[設定] を選択します。[設定] ウィンドウの [IDE Controller 1 (IDE コントローラー 1)] で、[DVD Drive (DVD ドライブ)] を選択します。
6. DVD ドライブの場合は、[Image file (イメージファイル)] を選択し、`seed.iso` ファイルを参照して選択します。
7. 変更を適用し、VM を起動します。

VM を起動したら、`user-data` 設定ファイルで定義されているいずれかのユーザーアカウントを使用してログインします。初回ログイン後に、VM から `seed.iso` 起動イメージを切断できます。

Amazon Linux 2 のカーネルライブパッチ

Amazon Linux 2 のカーネルライブパッチを使用すると、実行中のアプリケーションを再起動や中断せずに、実行中の Linux カーネルにセキュリティの脆弱性や重大なバグのパッチを適用することができます。インフラストラクチャを安全かつ最新に保つとともに、サービスとアプリケーションの可用性を向上させることができます。

AL2023 用 Kernel Live Patching の情報については、「AL2023 ユーザーガイド」の「Amazon Linux 2023 の Kernel Live Patching」を参照してください。

AWS は、Amazon Linux 2 向けに 2 種類のカーネルライブパッチをリリースします。

- [Security updates] (セキュリティ更新) – Linux の共通脆弱性とエクスポージャー (CVE) の更新プログラムが含まれます。これらの更新プログラムは、通常、Amazon Linux Security Advisory の評価で Important または Critical と評価されます。これらは、通常、共通脆弱性評価システム (CVSS) の 7 以上のスコアに該当します。場合によっては、CVE が割り当てられる前に AWS から更新プログラムが提供されることがあります。そのような場合、パッチはバグ修正プログラムとして提供される場合があります。

- [Bug fixes] (バグ修正) – CVE に関連付けられていない重大なバグや安定性の問題の修正プログラムが含まれます。

AWS は、Amazon Linux 2 カーネルバージョンのリリースから 3 か月間、カーネルライブパッチを提供します。3 か月が過ぎた後にカーネルライブパッチを引き続き入手するには、新しいカーネルバージョンに更新する必要があります。

Amazon Linux 2 のカーネルライブパッチは、既存の Amazon Linux 2 リポジトリから署名付きの RPM パッケージとして入手できます。パッチを個別のインスタンスにインストールするには、既存の yum ワークフローを使用できます。パッチをマネージドインスタンスのグループにインストールするには、AWS Systems Manager を使用できます。

Amazon Linux 2 のカーネルライブパッチは、追加料金なしで提供されます。

トピック

サポートされている構成と前提条件

カーネルライブパッチは、Amazon EC2 インスタンスおよび Amazon Linux 2 が実行されているオンプレミスの仮想マシンでサポートされています。

Amazon Linux 2 でカーネルライブパッチを使用するには、以下を使用する必要があります。

- `x86_64` アーキテクチャのカーネルバージョン `4.14` または `5.10`
- `ARM64` アーキテクチャのカーネルバージョン `5.10`

ポリシーの要件

Amazon Linux リポジトリからパッケージをダウンロードするには、Amazon Elastic Compute Cloud がサービス所有の Amazon S3 バケットにアクセスする必要があります。環境で Amazon S3 用に Amazon Virtual Private Cloud (VPC) エンドポイントを使用している場合、VPC エンドポイントポリシーがそれらのパブリックバケットへのアクセスを許可しているようにする必要があります。

この表は、EC2 が Kernel Live Patching のためにアクセスする必要がある可能性がある各 Amazon S3 バケットを示しています。

S3 バケット ARN	説明
arn:aws:s3:::packages.*region*.amazonaws.com/*	Amazon Linux AMI パッケージを含む Amazon S3 バケット
arn:aws:s3:::repo.*region*.amazonaws.com/*	Amazon Linux AMI リポジトリを含む Amazon S3 バケット
arn:aws:s3:::amazonlinux.*region*.amazonaws.com/*	Amazon Linux 2 リポジトリを含む Amazon S3 バケット
arn:aws:s3:::amazonlinux-2-repos-*region*/*	Amazon Linux 2 リポジトリを含む Amazon S3 バケット

次のポリシーは、組織に属する ID とリソースへのアクセスを制限し、Kernel Live Patching に必要な Amazon S3 バケットへのアクセスを提供する方法を示しています。*region*、*principal-org-id*、*resource-org-id* を組織の値に置き換えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "AllowRequestsByOrgsIdentitiesToOrgsResources",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": "*",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:PrincipalOrgID": "principal-org-id",
          "aws:ResourceOrgID": "resource-org-id"
```

```
        }
      }
    },
    {
      "Sid": "AllowAccessToAmazonLinuxAMIRepositories",
      "Effect": "Allow",
      "Principal": {
        "AWS": "*"
      },
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::packages.region.amazonaws.com/*",
        "arn:aws:s3:::repo.region.amazonaws.com/*",
        "arn:aws:s3:::amazonlinux.region.amazonaws.com/*",
        "arn:aws:s3:::amazonlinux-2-repos-region/*"
      ]
    }
  ]
}
```

カーネルライブパッチの使用

個別のインスタンスでカーネルライブパッチを有効にして使用するには、インスタンス自体でコマンドラインを使用できます。マネージドインスタンスのグループでカーネルライブパッチを有効にして使用するには、AWS Systems Manager を使用できます。

以下のセクションでは、コマンドラインを使用して、カーネルライブパッチを有効にして個別のインスタンスで使用する方法について説明します。

マネージドインスタンスのグループでカーネルライブパッチを有効にして使用する方法の詳細については、AWS Systems Manager ユーザーガイドの「Amazon Linux 2 インスタンスでカーネルライブパッチを使用する」を参照してください。

トピック

カーネルライブパッチの有効化

Amazon Linux 2 では、カーネルライブパッチはデフォルトでは無効になっています。ライブパッチを使用するには、カーネルライブパッチの yum プラグインをインストールして、ライブパッチ機能を有効にする必要があります。

前提条件

カーネルライブパッチには binutils が必要です。binutils がインストールされていない場合は、以下のコマンドを使用してインストールします。

```
$ sudo yum install binutils
```

カーネルライブパッチを有効にするには

1. カーネルライブパッチは、次の Amazon Linux 2 カーネルバージョンで使用できます。

   - x86_64 アーキテクチャのカーネルバージョン 4.14 または 5.10
   - ARM64 アーキテクチャのカーネルバージョン 5.10

   カーネルバージョンを確認するには、次のコマンドを実行します。

   ```
   $ sudo yum list kernel
   ```
2. サポートされているカーネルバージョンが既にある場合は、この手順はスキップしてください。サポートされているカーネルバージョンがない場合は、次のコマンドを実行して、カーネルを最新バージョンに更新し、インスタンスを再起動します。

   ```
   $ sudo yum install -y kernel
   ```

   ```
   $ sudo reboot
   ```
3. カーネルライブパッチの yum プラグインをインストールします。

   ```
   $ sudo yum install -y yum-plugin-kernel-livepatch
   ```
4. カーネルライブパッチの yum プラグインを有効にします。

   ```
   $ sudo yum kernel-livepatch enable -y
   ```

このコマンドは、設定されているリポジトリから最新バージョンのカーネルライブパッチの RPM もインストールします。

5. カーネルライブパッチの yum プラグインが正常にインストールされたことを確認するには、次のコマンドを実行します。

```
$ rpm -qa | grep kernel-livepatch
```

カーネルライブパッチを有効にすると、空のカーネルライブパッチの RPM が自動的に適用されます。カーネルライブパッチが正常に有効になっていれば、このコマンドは最初の空のカーネルライブパッチの RPM を含むリストを返します。以下は出力例です。

```
yum-plugin-kernel-livepatch-1.0-0.11.amzn2.noarch
kernel-livepatch-5.10.102-99.473-1.0-0.amzn2.x86_64
```

6. kpatch パッケージをインストールします。

```
$ sudo yum install -y kpatch-runtime
```

7. 既にインストール済みの場合は、kpatch サービスを更新します。

```
$ sudo yum update kpatch-runtime
```

8. kpatch サービスを起動します。このサービスは、初期化時または起動時にすべてのカーネルライブパッチをロードします。

```
$ sudo systemctl enable kpatch.service
```

9. Amazon Linux 2 Extras Library のカーネルライブパッチというトピックを有効にします。このトピックには、カーネルライブパッチが含まれています。

```
$ sudo amazon-linux-extras enable livepatch
```

利用可能なカーネルライブパッチを表示する

Amazon Linux のセキュリティアラートは、Amazon Linux Security Center に公開されます。カーネルライブパッチのアラートを含む Amazon Linux 2 のセキュリティアラートの詳細については、Amazon Linux Security Center を参照してください。カーネルライブパッチに

は、ALASLIVEPATCH というプレフィクスが付きます。Amazon Linux Security Center では、バグに対応するカーネルライブパッチは一覧に表示されていない場合があります。

アドバイザリおよび CVE に対する利用可能なカーネルライブパッチは、コマンドラインを使用して見つけることもできます。

アドバイザリに対する利用可能なすべてのカーネルライブパッチを一覧表示するには

次のコマンドを使用します。

```
$ yum updateinfo list
```

出力例を次に示します。

```
Loaded plugins: extras_suggestions, kernel-livepatch, langpacks, priorities, update-
motd
ALAS2LIVEPATCH-2020-002 important/Sec. kernel-
livepatch-5.10.102-99.473-1.0-3.amzn2.x86_64
ALAS2LIVEPATCH-2020-005 medium/Sec. kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
updateinfo list done
```

CVE に対する利用可能なすべてのカーネルライブパッチを一覧表示するには

次のコマンドを使用します。

```
$ yum updateinfo list cves
```

出力例を次に示します。

```
Loaded plugins: extras_suggestions, kernel-livepatch, langpacks, priorities, update-
motdamzn2-core/2/x86_64 | 2.4 kB 00:00:00
CVE-2019-15918 important/Sec. kernel-livepatch-5.10.102-99.473-1.0-3.amzn2.x86_64
CVE-2019-20096 important/Sec. kernel-livepatch-5.10.102-99.473-1.0-3.amzn2.x86_64
CVE-2020-8648 medium/Sec. kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
updateinfo list done
```

カーネルライブパッチの適用

カーネルライブパッチは、yum パッケージマネージャーを使用して、通常の更新プログラムを適用するのと同じ方法で適用します。カーネルライブパッチの yum プラグインは、適用するカーネルライブパッチを管理し、再起動も必要ありません。

> **Tip**
>
> カーネルが安全かつ最新に保たれるよう、カーネルライブパッチを使用して定期的にカーネルを更新することをお勧めします。

特定のカーネルライブパッチを適用するか、利用可能なカーネルライブパッチを定期的なセキュリティ更新プログラムと一緒に適用するかを選択できます。

特定のカーネルライブパッチを適用するには

1. 「利用可能なカーネルライブパッチを表示する」で説明されているコマンドのいずれかを使用して、カーネルライブパッチのバージョンを取得します。
2. Amazon Linux 2 カーネルのカーネルライブパッチを適用します。

   ```
   $ sudo yum install kernel-livepatch-kernel_version.x86_64
   ```

   例えば、次のコマンドは、Amazon Linux 2 カーネルバージョン `5.10.102-99.473` のカーネルライブパッチを適用します。

   ```
   $ sudo yum install kernel-livepatch-5.10.102-99.473-1.0-4.amzn2.x86_64
   ```

利用可能なカーネルライブパッチを定期的なセキュリティ更新プログラムと一緒に適用するには

次のコマンドを使用します。

```
$ sudo yum update --security
```

バグ修正プログラムを含めるには、`--security` オプションを省略します。

> **Important**
>
> - カーネルライブパッチを適用しても、カーネルバージョンは更新されません。バージョンは、インスタンスを再起動した後でのみ新しいバージョンに更新されます。
> - Amazon Linux 2 カーネルは、カーネルライブパッチを 3 か月間入手できます。3 か月が過ぎると、そのカーネルバージョンの新しいカーネルライブパッチはリリースされなくなります。3 か月が過ぎた後にカーネルライブパッチを引き続き入手するには、インスタンス

を再起動して新しいカーネルバージョンに移行する必要があります。これにより、次の 3 か月も引き続きカーネルライブパッチを入手することができます。お使いのカーネルバージョンのサポート期間を確認するには、`yum kernel-livepatch supported` を実行します。

適用されたカーネルライブパッチの表示

適用されたカーネルライブパッチを表示するには

次のコマンドを使用します。

```
$ kpatch list
```

このコマンドは、ロードおよびインストールされたセキュリティ更新プログラムのカーネルライブパッチのリストを返します。出力例を次に示します。

```
Loaded patch modules:
livepatch_cifs_lease_buffer_len [enabled]
livepatch_CVE_2019_20096 [enabled]
livepatch_CVE_2020_8648 [enabled]

Installed patch modules:
livepatch_cifs_lease_buffer_len (5.10.102-99.473.amzn2.x86_64)
livepatch_CVE_2019_20096 (5.10.102-99.473.amzn2.x86_64)
livepatch_CVE_2020_8648 (5.10.102-99.473.amzn2.x86_64)
```

Note

1 つのカーネルライブパッチには、複数のライブパッチが含まれていてインストールされる場合があります。

カーネルライブパッチの無効化

カーネルライブパッチを使用する必要がなくなった場合は、いつでも無効にできます。

カーネルライブパッチを無効にするには

1. 適用されたカーネルライブパッチの RPM パッケージを削除します。

```
$ sudo yum kernel-livepatch disable
```

2. カーネルライブパッチの yum プラグインをアンインストールします。

```
$ sudo yum remove yum-plugin-kernel-livepatch
```

3. インスタンスを再起動します。

```
$ sudo reboot
```

制限事項

カーネルライブパッチには以下の制限があります。

- カーネルライブパッチの適用中は、休止を実行したり、高度なデバッグツール (SystemTap、kprobes、eBPF ベースのツールなど) を使用したり、カーネルライブパッチを適用したインフラストラクチャで使用されている ftrace の出力ファイルにアクセスしたりすることはできません。

よくある質問

Amazon Linux 2 のカーネルライブパッチに関するよくある質問については、「Amazon Linux 2 に関するよくある質問」を参照してください。

ユーザー提供カーネル

Amazon EC2 インスタンスでカスタムカーネルが必要な場合は、必要としているものに近い AMI を使用して開始し、インスタンスでカスタムカーネルをコンパイルして、新しいカーネルを参照するようにブートローダーを更新します。このプロセスは AMI が使用する仮想化タイプによって異なります。詳細については、Linux AMI 仮想化タイプ を参照してください。

コンテンツ

- HVM AMIs (GRUB)
- AMIs の準仮想化 (PV-GRUB)

HVM AMIs (GRUB)

HVM インスタンスボリュームは実際の物理ディスクのように扱われます。起動プロセスは、パーティション分割ディスクとブートローダーを備えるベアメタルオペレーティングシステムの起動プロセスに似ています。ブートローダーでは、現在サポートされているすべての Linux ディストリビューションを使用できます。最も一般的なブートローダーは GRUB または GRUB2 です。

起動が遅くなるため、デフォルトでは GRUB はインスタンスのコンソールに出力を送信しません。詳細については、インスタンスコンソール出力 を参照してください。カスタムカーネルをインストールする場合は、GRUB 出力を有効にすることを検討してください。

フォールバックカーネルを指定する必要はありません。ただし、新しいカーネルをテストする場合は、フォールバックを設定することをお勧めします。GRUB では、新しいカーネルにエラーがあった場合に別のカーネルにフォールバックできます。フォールバックカーネルを設定すると、新しいカーネルが見つからない場合でも、インスタンスを起動できます。

Amazon Linux 用のレガシー GRUB では `/boot/grub/menu.lst` を使用します。Amazon Linux 2 用の GRUB2 では `/etc/default/grub` を使用します。ブートローダーでデフォルトカーネルを更新する方法の詳細については、ご使用の Linux ディストリビューションのドキュメントを参照してください。

AMIs の準仮想化 (PV-GRUB)

準仮想化 (PV) を使用する Amazon マシンイメージ では、起動プロセスで PV-GRUB と呼ばれるシステムが利用されます。PV-GRUB は、パッチが適用されたバージョンの GNU GRUB 0.97 を実行する準仮想化ブートローダーです。インスタンスを起動すると、PV-GRUB では起動プロセスが開始され、お客様のイメージの `menu.lst` ファイルが指定するカーネルがチェーンロードされます。

PV-GRUB は標準の `grub.conf` または `menu.lst` コマンドを認識しますこれにより、現在サポートされているすべての Linux ディストリビューションとともに利用できます。Ubuntu 10.04 LTS、Oracle Enterprise Linux、CentOS 5.x など、古いディストリビューションでは特別な「ec2」や「xen」カーネルパッケージが必要です。新しいディストリビューションでは、デフォルトのカーネルパッケージに必要なドライバーが含まれています。

最新の準仮想 AMI では、デフォルトで PV-GRUB AKI を使用します (Amazon EC2 Launch Wizard Quick Start メニューで利用できるすべての準仮想 Linux AMI が含まれています)。そのため、使用するカーネルにディストリビューションとの互換性がある場合、インスタンスで別のカーネルを使用するために必要な追加の手順はありません。インスタンスでカスタムカーネルを実行する最適な方法と

しては、必要としているものに近い AMI を使用して開始し、インタンスでカスタムカーネルをコンパイルして、そのカーネルを使用して起動するように menu.lst ファイルを変更します。

AMI のカーネルイメージが PV-GRUB AKI であることを確認できます。次の describe-images コマンドを実行して (ご使用のカーネルイメージ ID に置き換えます)、Name フィールドが pv-grub で始まっているかどうかを確認します。

```
aws ec2 describe-images --filters Name=image-id,Values=aki-880531cd
```

コンテンツ

PV-GRUB の制約事項

PV-GRUB には次の制約事項があります。

- PV-GRUB の 64 ビットバージョンを使用して 32 ビットカーネルを起動したり、PV-GRUB の 32 ビットバージョンを使用して 64 ビットカーネルを起動したりすることはできません。
- PV-GRUB AKI の使用時には、Amazon ラムディスクイメージ (ARI) を指定できません。
- AWS は、PV-GRUB が EXT2、EXT3、EXT4、JFS、XFS、ReiserFS のファイルシステム形式で動作することをテストし、確認しています。その他のファイルシステム形式では動作しない場合があります。
- PV-GRUB は、gzip、bzip2、lzo、xz 圧縮形式を利用して圧縮されたカーネルを起動できます。
- Cluster AMI は PV-GRUB をサポートせず、また、必要としません。完全ハードウェア仮想化 (HVM) が使用されるためです。準仮想インスタンスは PV-GRUB を使用して起動します。一方、HVM インスタンスボリュームは実際のディスクのように扱われ、その起動プロセスはパーティション分割ディスクとブートローダーを備えるベアメタルオペレーティングシステムの起動プロセスに似ています。
- PV-GRUB バージョン 1.03 以前では、GPT パーティショニングをサポートしません。MBR パーティショニングがサポートされています。

- Amazon Elastic Block Store (Amazon EBS) で Logical Volume Manager (LVM) を使用する場合、LVM の外側に別の起動パーティションが必要です。その場合、LVM で論理ボリュームを作成できます。

準仮想化 AMIs 向けの GRUB の設定

PV-GRUB を起動するには、GRUB `menu.lst` ファイルがイメージに含まれている必要があります。このファイルの最も一般的な場所は `/boot/grub/menu.lst` です。

次の例は、PV-GRUB AKI を使用して AMI を起動する `menu.lst` 設定ファイルです。この例では、Amazon Linux 2018.03 (この AMI の元のカーネル) と Vanilla Linux 4.16.4 (https://www.kernel.org/ の Vanilla Linux カーネルの新しいバージョン) の 2 つのカーネルエントリが選択できます。Vanilla エントリは、この AMI の元々のエントリからコピーされました。`kernel` と `initrd` パスは新しい場所に更新されました。`default 0` パラメータは、ブートローダーをそれが検出した最初のエントリ (この場合、Vanilla エントリ) にポイントします。`fallback 1` パラメータは、最初のエントリの起動に問題が発生した場合、次のエントリにブートローダーをポイントします。

```
default 0
fallback 1
timeout 0
hiddenmenu

title Vanilla Linux 4.16.4
root (hd0)
kernel /boot/vmlinuz-4.16.4 root=LABEL=/ console=hvc0
initrd /boot/initrd.img-4.16.4

title Amazon Linux 2018.03 (4.14.26-46.32.amzn1.x86_64)
root (hd0)
kernel /boot/vmlinuz-4.14.26-46.32.amzn1.x86_64 root=LABEL=/ console=hvc0
initrd /boot/initramfs-4.14.26-46.32.amzn1.x86_64.img
```

`menu.lst` ファイルにフォールバックカーネルを指定する必要はありません。ただし、新しいカーネルをテストするときは、フォールバックを設定することをお勧めします。PV-GRUB では、新しいカーネルにエラーがあった場合に別のカーネルにフォールバックできます。フォールバックカーネルを設定すると、新しいカーネルが見つからない場合でもインスタンスを起動できます。

PV-GRUB は、次の場所で `menu.lst` をチェックします。その際、それが検出した最初の場所が利用されます。

- (hd0)/boot/grub
- (hd0,0)/boot/grub
- (hd0,0)/grub
- (hd0,1)/boot/grub
- (hd0,1)/grub
- (hd0,2)/boot/grub
- (hd0,2)/grub
- (hd0,3)/boot/grub
- (hd0,3)/grub

PV-GRUB 1.03 以前では、このリストの最初の 2 つの場所うちの 1 つのみがチェックされることに注意してください。

Amazon PV-GRUB カーネルイメージ ID

PV-GRUB AKI は、アジアパシフィック (大阪) を除くすべての Amazon EC2 リージョンで利用できます。32 ビットと 64 ビットの両方のアーキテクチャタイプに AKI があります。最新の AMI では、デフォルトで PV-GRUB AKI が使用されます。

すべてのバージョンの PV-GRUB AKI がすべてのインスタンスタイプと互換性があるとは限らないため、常に最新バージョンの PV-GRUB AKI を使用することをお勧めします。次の describe-images コマンドを使用し、現在のリージョンの PV-GRUB AKI のリストを取得します。

```
aws ec2 describe-images --owners amazon --filters Name=name,Values=pv-grub-*.gz
```

PV-GRUB は、ap-southeast-2 リージョンで利用できる唯一の AKI です。このリージョンにコピーする AMI が、このリージョンで利用できる PV-GRUB のバージョンを使用していることを確認してください。

各リージョンの現在の AKI ID は次のとおりです。新しい AMI は、hd0 AKI を使用して登録します。

> **Note**
>
> hd00 AKI は、以前に利用可能であったリージョンでの下位互換性のために引き続き提供されます。

ap-northeast-1、アジアパシフィック (東京)

イメージ ID	イメージ名
aki-f975a998	pv-grub-hd0_1.05-i386.gz
aki-7077ab11	pv-grub-hd0_1.05-x86_64.gz

ap-southeast-1、アジアパシフィック (シンガポール) リージョン

イメージ ID	イメージ名
aki-17a40074	pv-grub-hd0_1.05-i386.gz
aki-73a50110	pv-grub-hd0_1.05-x86_64.gz

ap-southeast-2、アジアパシフィック (シドニー)

イメージ ID	イメージ名
aki-ba5665d9	pv-grub-hd0_1.05-i386.gz
aki-66506305	pv-grub-hd0_1.05-x86_64.gz

eu-central-1、欧州 (フランクフルト)

イメージ ID	イメージ名
aki-1419e57b	pv-grub-hd0_1.05-i386.gz
aki-931fe3fc	pv-grub-hd0_1.05-x86_64.gz

eu-west-1、欧州 (アイルランド)

イメージ ID	イメージ名
aki-1c9fd86f	pv-grub-hd0_1.05-i386.gz

イメージ ID	イメージ名
aki-dc9ed9af	pv-grub-hd0_1.05-x86_64.gz

sa-east-1、南米 (サンパウロ)

イメージ ID	イメージ名
aki-7cd34110	pv-grub-hd0_1.05-i386.gz
aki-912fbcfd	pv-grub-hd0_1.05-x86_64.gz

us-east-1、US East (N. Virginia)

イメージ ID	イメージ名
aki-04206613	pv-grub-hd0_1.05-i386.gz
aki-5c21674b	pv-grub-hd0_1.05-x86_64.gz

us-gov-west-1、AWS GovCloud (米国 - 西部)

イメージ ID	イメージ名
aki-5ee9573f	pv-grub-hd0_1.05-i386.gz
aki-9ee55bff	pv-grub-hd0_1.05-x86_64.gz

us-west-1、米国西部 (北カリフォルニア)

イメージ ID	イメージ名
aki-43cf8123	pv-grub-hd0_1.05-i386.gz
aki-59cc8239	pv-grub-hd0_1.05-x86_64.gz

us-west-2、米国西部 (オレゴン)

イメージ ID	イメージ名
aki-7a69931a	pv-grub-hd0_1.05-i386.gz
aki-70cb0e10	pv-grub-hd0_1.05-x86_64.gz

PV-GRUB の更新

すべてのバージョンの PV-GRUB AKI がすべてのインスタンスタイプと互換性があるとは限らないため、常に最新バージョンの PV-GRUB AKI を使用することをお勧めします。また、古いバージョンの PV-GRUB はすべてのリージョンで使用できるわけではないため、旧バージョンを使用する AMI を、そのバージョンをサポートしないリージョンにコピーした場合、カーネルのイメージを更新するまで、その AMI から起動されたインスタンスを起動できなくなります。次の手順を使用してインスタンスの PV-GRUB のバージョンを確認し、必要に応じて更新します。

PV-GRUB のバージョンを確認するには

1. インスタンスのカーネル ID を見つけます。

```
aws ec2 describe-instance-attribute --instance-id instance_id --attribute kernel --
region region

{
    "InstanceId": "instance_id",
    "KernelId": "aki-70cb0e10"
}
```

   このインスタンスのカーネル ID は、aki-70cb0e10 です。

2. このカーネル ID のバージョン情報を表示します。

```
aws ec2 describe-images --image-ids aki-70cb0e10 --region region

{
    "Images": [
        {
            "VirtualizationType": "paravirtual",
            "Name": "pv-grub-hd0_1.05-x86_64.gz",
            ...
```

```
            "Description": "PV-GRUB release 1.05, 64-bit"
        }
    ]
}
```

このカーネルイメージは PV-GRUB 1.05 です。PV-GRUB のバージョンが最新バージョン (Amazon PV-GRUB カーネルイメージ ID を参照) でない場合、次の手順を使用して更新する必要があります。

PV-GRUB のバージョンを更新するには

インスタンスが古いバージョンの PV-GRUB を使用している場合は、最新バージョンに更新する必要があります。

1. Amazon PV-GRUB カーネルイメージ ID で、使用するリージョンとプロセッサアーキテクチャーの最新の PV-GRUB AKI を特定します。
2. インスタンスを停止します。使用されるカーネルイメージを変更するには、インスタンスを停止する必要があります。

   ```
   aws ec2 stop-instances --instance-ids instance_id --region region
   ```
3. インスタンスに使用するカーネルイメージを変更します。

   ```
   aws ec2 modify-instance-attribute --instance-id instance_id --kernel kernel_id --
   region region
   ```
4. インスタンスを再起動します。

   ```
   aws ec2 start-instances --instance-ids instance_id --region region
   ```

Amazon Linux 2 MATE デスクトップ接続を設定する

MATE デスクトップ環境は、AMI にあらかじめインストールおよび設定されています。説明は次のとおりです。

".NET Core *x.x*, Mono *x.xx*, PowerShell *x.x*, and MATE DE pre-installed to run your .NET applications on Amazon Linux 2 with Long Term Support (LTS)."

この環境は、コマンドラインを極力使用せずに Amazon Linux 2 インスタンスを管理できる、直感的なグラフィカルユーザーインターフェイスを提供します。このインタフェースでは、アイコン、ウィンドウ、ツールバー、フォルダ、壁紙、デスクトップウィジェットなどのグラフィカルな表現が使用されています。一般的なタスクを実行するために、組み込みの GUI ベースのツールを使用することができます。例えば、ソフトウェアの追加と削除、更新プログラムの適用、ファイルの整理、プログラムの起動、システムの状態のモニタリングのためのツールが用意されています。

Important

`xrdp` は、AMI にバンドルされているリモートデスクトップソフトウェアです。デフォルトでは、`xrdp` は、自己署名の TLS 証明書を使用してリモートデスクトップセッションを暗号化します。AWS も `xrdp` のメンテナンス担当者も、本番環境での自己署名証明書の使用を推奨していません。代わりに、適切な認証局 (CA) から証明書を取得し、インスタンスにインストールします。TLS の設定については、`xrdp` wiki の「TLS セキュリティレイヤー」を参照してください。

Note

xrdp の代わりに仮想ネットワークコンピューティング (VNC) サービスを使用する場合は、AWS ナレッジセンターの記事「Amazon Linux 2 を実行している Amazon EC2 インスタンスに GUI をインストールする方法を教えてください。」を参照してください。

前提条件

このトピックにあるコマンドを実行するには、AWS Command Line Interface (AWS CLI) または AWS Tools for Windows PowerShell をインストールし、AWS プロファイルを設定する必要があります。

オプション

1. AWS CLI のインストール - 詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS CLI のインストール」と「設定の基本」を参照してください。
2. Tools for Windows PowerShell のインストール — 詳細については、「AWS Tools for Windows PowerShell ユーザーガイド」の「AWS Tools for Windows PowerShell のインストール」と「共有認証情報」を参照してください。

RDP 接続の設定

次の手順に従って、ローカルマシンから MATE デスクトップ環境が稼働している Amazon Linux 2 インスタンスへの、リモートデスクトップ (RDP) 接続をセットアップします。

1. Amazon Linux 2 で AMI 名に MATE が含まれている AMI の ID を取得するには、ローカルのコマンドラインツールから describe-images コマンドを使用します。コマンドラインツールをインストールしていない場合は、次のクエリを AWS CloudShell セッションから直接実行します。CloudShell からシェルセッションを起動する方法の詳細については、「[Getting started with AWS CloudShel]」(CloudShell の使用方法) を参照してください。Amazon EC2 コンソールでは、インスタンスを起動し、AMI 検索バーに MATE と入力すると、MATE が含まれている AMI を見つけることができます。MATE が事前インストールされた Amazon Linux 2 クイックスタートが、検索結果に表示されます。

```
aws ec2 describe-images --filters "Name=name,Values=amzn2*MATE*" --query
 "Images[*].[ImageId,Name,Description]"
[
    [
        "ami-0123example0abc12",
        "amzn2-x86_64-MATEDE_DOTNET-2020.12.04",
        ".NET Core 5.0, Mono 6.12, PowerShell 7.1, and MATE DE pre-installed to run
 your .NET applications on Amazon Linux 2 with Long Term Support (LTS)."
    ],
    [
        "ami-0456example0def34",
        "amzn2-x86_64-MATEDE_DOTNET-2020.04.14",
        "Amazon Linux 2 with .Net Core, PowerShell, Mono, and MATE Desktop
 Environment"
    ]
]
```

   用途に合った AMI を選択します。

2. 前のステップで特定した AMI を使用して EC2 インスタンスを起動します。ポート 3389 へのインバウンド TCP トラフィックを許可するようにセキュリティグループを設定します。セキュリティグループの設定の詳細については、「VPC のセキュリティグループ」を参照してください。この設定により、RDP クライアントを使用してインスタンスに接続できます。
3. SSH を使用してインスタンスに接続します。
4. インスタンス上のソフトウェアとカーネルを更新します。

```
[ec2-user ~]$ sudo yum update
```

更新が完了したら、インスタンスを再起動し、更新から最新のパッケージとライブラリが使用されていることを確認します。カーネルの更新は、再起動するまでロードされません。

```
[ec2-user ~]$ sudo reboot
```

5. インスタンスに再接続し、Linux のインスタンスで次のコマンドを実行して、ec2-user のパスワードを設定します。

```
[ec2-user ~]$ sudo passwd ec2-user
```

6. 証明書ファイルとキーをインストールする

証明書とキーがすでにある場合は、証明書とキーを以下のように /etc/xrdp/ ディレクトリにコピーします。

- 証明書 - /etc/xrdp/cert.pem
- キー — /etc/xrdp/key.pem

証明書とキーがない場合は、次のコマンドを使用して/etc/xrdpディレクトリに証明書とキーを作成します。

```
$ sudo openssl req -x509 -sha384 -newkey rsa:3072 -nodes -keyout /etc/xrdp/key.pem
 -out /etc/xrdp/cert.pem -days 365
```

> **Note**
>
> このコマンドは 365 日間有効な証明書を生成します。

7. インスタンスに接続するコンピュータで RDP クライアントを開きます (Microsoft Windows を実行するコンピュータのリモートデスクトップ接続など)。ユーザー名として「ec2-user」と入力し、前のステップで設定したパスワードを入力します。

Amazon EC2 インスタンスで **xrdp** を無効にするには

Linux インスタンスで次のコマンドのいずれかを実行することにより、いつでも xrdp を無効にすることができます。次のコマンドは、X11 サーバーを使用して MATE を使用する能力に影響を及ぼすものではありません。

```
[ec2-user ~]$ sudo systemctl disable xrdp
```

```
[ec2-user ~]$ sudo systemctl stop xrdp
```

Amazon EC2 インスタンスで **xrdp** を有効にするには

MATE デスクトップ環境を実行している Amazon Linux 2 インスタンスに接続できるように xrdp を再度有効にするには、Linux インスタンスで次のコマンドのいずれかを実行します。

```
[ec2-user ~]$ sudo systemctl enable xrdp
```

```
[ec2-user ~]$ sudo systemctl start xrdp
```

AMI クォータ

AMI を作成および共有する際には、以下のクォータが適用されます。AWS リージョン ごとにクォータが適用されます。

クォータ名	説明	リージョンあたりのデフォルトのクォータ
AMI	リージョンごとに許可されているパブリック AMI およびプライベート AMI の最大数。これらには、利用可能な AMI と保留中の AMI、およびごみ箱にある AMI が含まれます。	50,000
パブリック AMI	リージョンごとに許可されているパブリック AMI の最大数 (ごみ箱内のパブリック AMI を含む)。	5

クォータ名	説明	リージョンあたりのデフォルトのクォータ
AMI 共有	リージョン内で AMI を共有できるエンティティ (組織、組織単位 (OU)、アカウント) の最大数。AMI を組織または OU と共有する場合、組織内のアカウント数や OU 数はクォータにカウントされません。	1,000

クォータを超えて AMI をさらに作成または共有したい場合は、以下を実行できます。

- AMI またはパブリック AMI のクォータの合計を超える場合は、未使用のイメージの登録を解除することを検討してください。
- パブリック AMI のクォータを超える場合は、1 つ以上のパブリック AMI をプライベートにすることを検討してください。
- AMI の共有クォータを超える場合は、個別のアカウントではなく、組織または OU と AMI を共有することを検討してください。
- AMI のクォータの引き上げをリクエストします。

AMI のクォータの引き上げをリクエストする

AMI のデフォルトクォータを超える容量が必要な場合は、クォータの引き上げをリクエストできます。

AMI のクォータの引き上げをリクエストするには

1. Service Quotas コンソール (https://console.aws.amazon.com/servicequotas/home) を開きます。
2. ナビゲーションペインで、[AWS のサービス] を選択します。
3. リストから [Amazon Elastic Compute Cloud (Amazon EC2)] を選択するか、検索ボックスにサービスの名前を入力します。
4. 引き上げをリクエストするには、AMI クォータを選択します。選択できる AMI クォータは次のとおりです:

- AMI
- パブリック AMI
- AMI 共有

5. [Request quota increase] (クォータの引き上げのリクエスト) を選択します。
6. [Change quota value] (クォータ値の変更) に新しいクォータ値を入力し、[Request] (リクエスト) を選択します。

保留中または最近解決されたリクエストを表示するには、ナビゲーションペインから [ダッシュボード] を選択します。保留中のリクエストの場合は、リクエストのステータスを選択してリクエストの受信をオープンします。リクエストの初期ステータスは [Pending] (保留中) です。ステータスが [Quota requested] (クォータをリクエスト済み) に変わると、[Support Center case number] (サポートセンターのケース番号) にケース番号が表示されます。リクエストのチケットを開くには、ケース番号を選択します。

リクエストが解決されると、クォータの [適用されたクォータ値] が新しい値に設定されます。

詳細については、Service Quotas ユーザーガイドを参照してください。

Amazon EC2 インスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

Amazon EC2 を初めて使用する場合は、次のトピックを参照して使用を開始してください。

- Amazon EC2 とは
- Amazon EC2 を使用するようにセットアップする
- チュートリアル: Amazon EC2 Linux インスタンスの開始方法
- インスタンスのライフサイクル

実稼働環境を起動する前に、以下の質問に答える必要があります。

Q. ニーズに最も合っているインスタンスタイプはどれか?

Amazon EC2 には、アプリケーションを実行するために必要な CPU、メモリ、ストレージ、ネットワークキャパシティーを選択できるようにするため、さまざまなインスタンスタイプが用意されています。詳細については、インスタンスタイプ を参照してください。

Q. ニーズに最も合っている購入オプションはどれか?

Amazon EC2 では、オンデマンドインスタンス (デフォルト)、スポットインスタンス、および リザーブドインスタンス をサポートします。詳細については、インスタンス購入オプション を参照してください。

Q. ニーズに合っているルートボリュームのタイプはどれか?

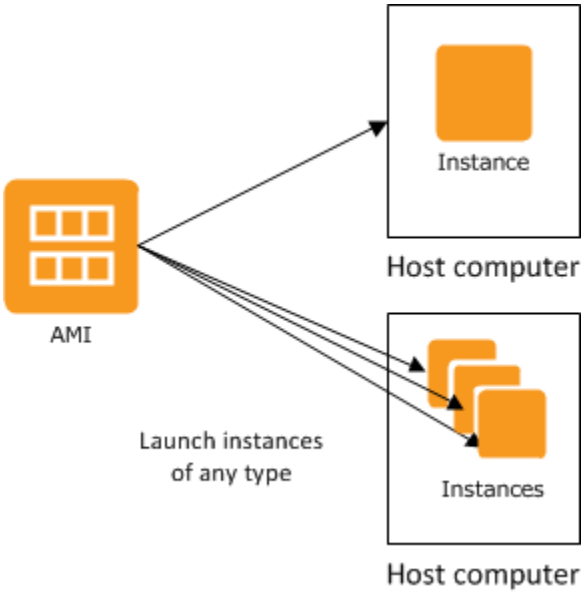
各インスタンスは Amazon EBS またはインスタンスストアによってサポートされています。必要なルートボリュームのタイプに基づいて AMI を選択します。詳細については、ルートデバイスのストレージ を参照してください。

Q. EC2 インスタンスフリートおよびハイブリッド環境にあるマシンを、遠隔から管理することは可能か?

AWS Systems Manager では、Amazon EC2 インスタンスの設定、オンプレミスのインスタンスの設定、および他のクラウドプロバイダーの仮想マシン (VM) などハイブリッド環境にある VM の設定を遠隔からセキュアに管理できます。詳細については、AWS Systems Manager ユーザーガイドを参照してください。

インスタンスと AMI

Amazon マシンイメージ (AMI) は、ソフトウェア構成 (オペレーティングシステム、アプリケーションサーバー、アプリケーションなど) を記録したテンプレートです。AMI から、クラウドで仮想サーバーとして実行される AMI のコピーであるインスタンスを起動します。以下の図に示すように、1 つの AMI の複数のインスタンスを起動することができます。



インスタンスは、停止、休止、または終了させるか、エラーが発生するまで実行を続けます。インスタンスがエラーで終了した場合は、元の AMI から新しいインスタンスを起動できます。

インスタンス

インスタンスとは、クラウドの仮想サーバーです。起動時の設定は、インスタンスを起動した際に指定した AMI のコピーです。

1 つの AMI から、複数の異なるタイプのインスタンスを起動することもできます。インスタンスタイプとは本質的に、インスタンスに使用されるホストコンピュータのハードウェアを決定するもの

です。インスタンスタイプごとに異なる処理内容やメモリの機能が提供されます。インスタンスタイプの選択は、そのインスタンス上で実行するアプリケーションやソフトウェアで必要となる、メモリ量や処理能力に基づき行います。インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

インスタンスの起動後は、通常のホストのように表示され、任意のコンピュータと同じように操作できます。インスタンスは完全に制御でき、sudo を使用して、ルート権限を必要とするコマンドを実行できます。

AWS アカウントでは、稼動できるインスタンスの数に制限があります。この制限の詳細、および増加を要求する方法については、Amazon EC2 の全般的なよくある質問の「Amazon EC2 ではいくつのインスタンスを稼動できますか」を参照してください。

インスタンスストレージ

インスタンスのルートデバイスには、インスタンスの起動に使用されるイメージが含まれています。ルートデバイスは、Amazon Elastic Block Store (Amazon EBS) ボリュームまたはインスタンスストアボリュームのいずれかです。詳細については、Amazon EC2 インスタンスのルートボリューム を参照してください。

インスタンスには、インスタンスストアボリュームと呼ばれるローカルストレージボリュームを含めることができます。これはブロックデバイスマッピングによって起動時に設定できます。詳細については、ブロックデバイスマッピング を参照してください。これらのボリュームがインスタンスに追加およびマッピングされたら、マウントして使用することができます。インスタンスが失敗、停止、または終了した場合、それらのボリュームのデータは失われます。したがって、これらのボリュームは一時データとして使用するのが最適です。重要なデータを安全に維持するには、複数のインスタンスにわたるレプリケーション方法を使用する必要があります。または、永続的なデータを Amazon S3 または Amazon EBS ボリュームに格納してください。詳細については、Amazon EC2 インスタンスのストレージオプション を参照してください。

セキュリティのベストプラクティス

- AWS Identity and Access Management (IAM) を使用して、インスタンスなど各 AWS リソースへのアクセスを制御します。詳細については、「Amazon EC2 の Identity and Access Management」を参照してください。
- 信頼されたホストまたはネットワークのみがインスタンスのポートにアクセスできるように制限します。例えば、ポート 22 の受信トラフィックを制限することで SSH アクセスを制限できます。

詳細については、Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ を参照してください。

- セキュリティグループのルールを定期的に確認し、最小権限 (— 必要なアクセス許可のみを開く) の原則を適用してください。また、さまざまなセキュリティグループを作成して、異なるセキュリティ要件を持つ各インスタンスに対応することもできます。外部ログインが許可された基本となるセキュリティグループの作成を検討し、外部ログインが許可されていないグループで残りのインスタンスを管理してください。
- AMI から起動されるインスタンスについてはパスワードベースのログインを無効にしてください。パスワードは検知または解読される恐れがあり、セキュリティ上のリスクです。詳細については、ルートユーザーのパスワードベースのリモートログインを無効にする を参照してください。AMI の安全な共有の詳細については、「共有 AMI」を参照してください。

インスタンスの停止と終了

実行中のインスタンスは、いつでも停止または終了できます。

インスタンスの停止

インスタンスが停止されると、インスタンスは通常のシャットダウンを実行してから、stopped 状態に移行します。そのすべての Amazon EBS ボリュームはアタッチされたままになり、後でインスタンスを再び開始することができます。

インスタンスが停止状態にあるとき、インスタンスの使用分が追加で課金されることはありません。最低 1 分間分の使用料が課金されます。インスタンスが停止状態にあるときにインスタンスタイプを変更した場合、インスタンスを起動すると同時に新しいインスタンスタイプの料金が課金されます。ルートデバイスの使用を含め、インスタンスに関連する Amazon EBS の使用はすべて、Amazon EBS 料金で課金されます。

インスタンスが停止状態の場合は、Amazon EBS ボリュームをアタッチおよびデタッチできます。インスタンスから AMI を作成し、カーネル、RAM ディスク、インスタンスタイプを変更することもできます。

インスタンスの終了

インスタンスを終了すると、インスタンスは正常なシャットダウンを実行します。ルートデバイスボリュームはデフォルトで削除されますが、アタッチされた Amazon EBS ボリュームはデフォルトでは保持されます (各ボリュームの `deleteOnTermination` 属性の設定によって決まります)。インスタンスそのものも削除され、後でインスタンスを再度起動することはできません。

間違って終了しないようにするため、インスタンスの削除を無効にすることができます。この場合、インスタンスの `disableApiTermination` 属性は必ず `true` にします。インスタンスのシャットダウン時の動作を制御するには (Linux の `shutdown -h` や Windows の `shutdown` など)、`instanceInitiatedShutdownBehavior` インスタンス属性を必要に応じて `stop` または `terminate` に設定します。Amazon EBS ボリュームをルートデバイスに持つインスタンスはデフォルトで `stop` に設定されます。インスタンスストアをルートデバイスに持つインスタンスはシャットダウンの結果として常に終了されます。

詳細については、インスタンスのライフサイクル を参照してください。

> **Note**
>
> Amazon EBS ボリュームや Elastic IP アドレスなど一部の AWS リソースでは、インスタンスの状態に関係なく利用料金が発生します。詳細については、AWS Billing ユーザーガイドの「予想外の料金の回避」を参照してください。Amazon EBS でのコストの詳細については、「Amazon EBS の価格」を参照してください。

AMI

Amazon Web Services (AWS) では、一般的な用途のソフトウェアに共通する構成を、多くの Amazon マシンイメージ (AMI) を通じて公開しています。加えて、AWS デベロッパーコミュニティのメンバーによって作成された、独自のカスタム AMI もあります。お客様自身でカスタム AMI を作成することもできます。必要なものがすべて含まれた新しいインスタンスを、すばやく簡単に起動できるようになります。例えば、ウェブサイトまたはウェブサービスに使用する場合は、AMI に含まれるものとして、ウェブサーバー、関連する静的コンテンツ、動的ページ用のコードが考えられます。この AMI からインスタンスを起動すると、ウェブサーバーが起動し、アプリケーションはリクエストを受け付け可能な状態になります。

すべての AMI は、Amazon EBS-backed (AMI からインスタンスを起動するときのルートデバイスは Amazon EBS ボリュームである) と Instance-store backed (AMI からインスタンスを起動するときのルートデバイスは、Amazon S3 に格納されているテンプレートから作成されたインスタンスストアボリュームである) のいずれかに分類されます。

AMI の説明に、ルートデバイスのタイプ (`ebs` または `instance store`) が明記されています。このことが重要であるのは、AMI のタイプによって、実行できる機能が大きく異なるからです。違いについての詳細は ルートデバイスのストレージ を参照してください。

AMI の利用が終わったら、その登録を解除できます。AMI の登録を解除すると、それを使用して新しいインスタンスを起動できなくなります。その AMI から起動された既存のインスタンスは影響を受けません。そのため、これらの AMI から起動されたインスタンスが終了した場合も、それらを削除する必要があります。

インスタンスタイプ

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

インスタンスを起動するときは、指定したインスタンスタイプによって、インスタンスに使用するホストコンピュータのハードウェアが決まります。インスタンスタイプごとに、コンピューティング、メモリ、およびストレージの機能が異なっており、これらの機能に基づいたインスタンスファミリーにグループ化されています。インスタンスタイプは、インスタンス上で実行するアプリケーションやソフトウェアの要件に基づいて選択します。

Amazon EC2 では、CPU、メモリ、インスタンスストレージなどホストコンピュータの一部のリソースを、特定のインスタンス専用に割り当てます。ネットワークやディスクサブシステムなどホストコンピュータでの他のリソースは、Amazon EC2 によりインスタンス間で共有されます。ホストコンピュータの各インスタンスが、これらの共有リソースの 1 つを可能な限り利用しようとする場合、それぞれのインスタンスは、そのリソースの共有分を等しく受け取ります。ただし、リソースの使用率が低い場合は、1 つのインスタンスがそのリソースのより多くの部分を利用できます。

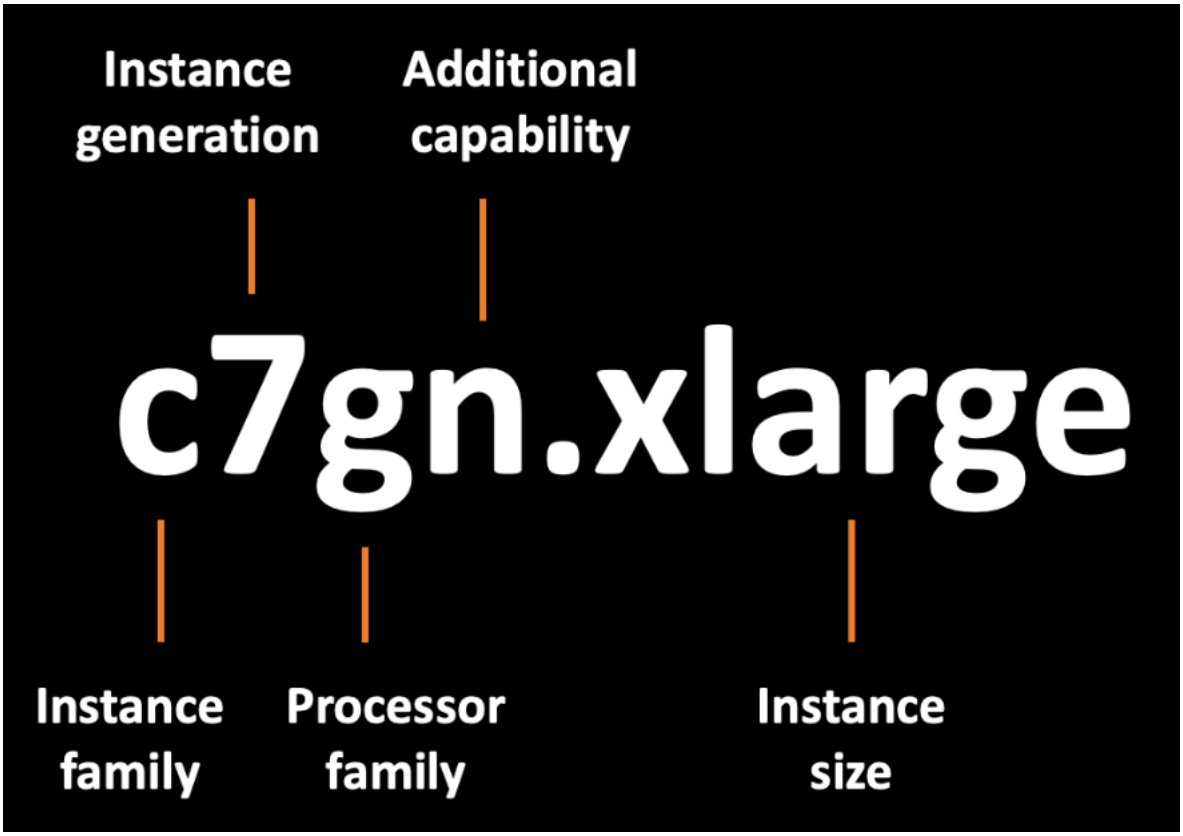
各インスタンスタイプは、共有リソースからからより高い、またはより低い最小性能を提供します。例えば、高速の I/O パフォーマンスを実行するインスタンスタイプは、共有リソースに対してより大きな割り当てを取得します。共有リソースをより大きく配分することによって、I/O 性能のばらつきを抑えることもできます。ほとんどのアプリケーションでは、中程度の I/O 性能があれば十分です。ただし、より高い、またはより一貫した I/O パフォーマンスを必要とするアプリケーションの場合は、より I/O パフォーマンスの高いインスタンスタイプを使用することを検討してください。

コンテンツ

- ハードウェア仕様
- AMI 仮想化タイプ
- Nitro System 上に構築されたインスタンス
- ネットワーキング機能とストレージ機能
- インスタンス制限
- 汎用インスタンス
- コンピュート最適化インスタンス
- メモリ最適化インスタンス
- ストレージ最適化インスタンス
- Linux 高速コンピューティングインスタンス
- ハイパフォーマンスコンピューティングインスタンス
- Amazon EC2 インスタンスタイプの検索
- インスタンスタイプに関する推奨事項の取得
- インスタンスタイプを変更する

インスタンスタイプの命名規則

Amazon EC2 はさまざまなインスタンスタイプを提供しており、要件に最適なタイプを選択できます。インスタンスタイプは、ファミリー、世代、プロセッサファミリー、追加機能、サイズに基づいて名前が付けられています。インスタンスタイプ名の先頭は、インスタンスファミリーを表しています (例: c)。2 番目のポジションでは、そのインスタンスの世代を示しています (例: 7)。g などの 3 番目の文字は、プロセッサファミリーを示します。ピリオドの前の残りの文字は、インスタンスストアボリュームなどの追加機能を示しています。ピリオド (.) の後はインスタンスサイズです (ベアメタルインスタンスの場合は `small`、`4xlarge`、`metal` など)。



インスタンスファミリー

- C — コンピューティング最適化
- D — 高密度ストレージ
- F — FPGA
- G — グラフィックを多用する
- Hpc – ハイパフォーマンスコンピューティング
- I — ストレージ最適化
- Im – vCPU とメモリの比率が 1 対 4 で最適化されたストレージ
- Is – vCPU とメモリの比率が 1 対 6 で最適化されたストレージ
- Inf – AWS 推論
- M — 汎用
- Mac — macOS
- P — GPU アクセラレーション
- R — メモリ最適化
- T — バースト可能パフォーマンス

- Trn – AWS Trainium
- U — ハイメモリ
- VT — ビデオトランスコーディング
- X — メモリ集約型

プロセッサファミリー

- a – AMD プロセッサ
- g – AWS Graviton プロセッサ
- i – Intel プロセッサ

その他の機能

- b – EBS 最適化
- d – インスタンスストアボリューム
- n — ネットワークと EBS の最適化
- e – 追加のストレージまたはメモリ
- z — 高パフォーマンス
- q – Qualcomm の推論アクセラレータ
- flex – Flex インスタンス

利用可能なインスタンスタイプ

Amazon EC2 では、幅広いインスタンスタイプの選択肢があり、さまざなユースケースに合わせて最適化できます。インスタンスタイプは、CPU、メモリ、ストレージ、およびネットワーク容量のさまざまな組み合わせで構成され、アプリケーションに適したリソースの組み合わせを柔軟に選択できます。各インスタンスタイプには 1 つ以上のインスタンスサイズが含まれているため、ターゲットワークロードの要件に合わせてリソースをスケーリングできます。

> **Note**
>
> 旧世代のインスタンスは引き続き完全にサポートされ、同じ特徴と機能が保持されます。最適なパフォーマンスを得るには、最新世代のインスタンスを使用してください。

サポートされるリージョン、コンピューティングリソース、ストレージリソースなど、要件を満たすインスタンスタイプを決定するには、「Amazon EC2 インスタンスタイプの検索」を参照してください。

トピック

現行世代のインスタンス

最適なパフォーマンスを得るために、新しいインスタンスを起動するときには、以下のインスタンスタイプを使用することをお勧めします。詳細については、「Amazon EC2 のインスタンスタイプ」を参照してください。

第 6 世代以降の Amazon EC2 インスタンスタイプ

- 汎用: M6a、M6g、M6gd、M6i、M6id、M6idn、M6in、M7a、M7g、M7gd、M7i、M7i-flex、T4g
- コンピューティング最適化:
  C6a、C6g、C6gd、C6gn、C6i、C6id、C6in、C7a、C7g、C7gd、C7gn、C7i
- メモリ最適化:
  R6a、R6g、R6gd、R6i、R6id、R6idn、R6in、R7a、R7g、R7gd、R7i、R7iz、X2gd、X2idn、X2iedn
- ストレージ最適化: I4g, I4i, Im4gn, Is4gen
- 高速コンピューティング: DL2q、G5g、Inf2、P5、Trn1、Trn1n
- ハイパフォーマンスコンピューティング: Hpc6a、Hpc6id、Hpc7a、Hpc7g

インスタンス

汎用

タイプ	Sizes
M5	m5.large|m5.xlarge |m5.2xlarge |m5.4xlarge | m5.8xlarge |m5.12xlarge |m5.16xlarge |m5.24xlarge | m5.metal
M5a	m5a.large |m5a.xlarge |m5a.2xlarge |m5a.4xlarge | m5a.8xlarge |m5a.12xlarge |m5a.16xlarge |m5a.24xlarge
M5ad	m5ad.large |m5ad.xlarge |m5ad.2xlarge |m5ad.4xlarge | m5ad.8xlarge |m5ad.12xlarge |m5ad.16xlarge |m5ad.24xlarge
M5d	m5d.large |m5d.xlarge |m5d.2xlarge |m5d.4xlarge | m5d.8xlarge |m5d.12xlarge |m5d.16xlarge |m5d.24xlarge |m5d.metal
M5dn	m5dn.large |m5dn.xlarge |m5dn.2xlarge |m5dn.4xlarge | m5dn.8xlarge |m5dn.12xlarge |m5dn.16xlarge |m5dn.24xlarge |m5dn.metal
M5n	m5n.large |m5n.xlarge |m5n.2xlarge |m5n.4xlarge | m5n.8xlarge |m5n.12xlarge |m5n.16xlarge |m5n.24xlarge |m5n.metal
M5zn	m5zn.large |m5zn.xlarge |m5zn.2xlarge |m5zn.3xlarge | m5zn.6xlarge |m5zn.12xlarge |m5zn.metal
M6a	m6a.large |m6a.xlarge |m6a.2xlarge |m6a.4xlarge | m6a.8xlarge |m6a.12xlarge |m6a.16xlarge |m6a.24xlarge |m6a.32xlarge |m6a.48xlarge |m6a.metal
M6g	m6g.medium |m6g.large |m6g.xlarge |m6g.2xlarge | m6g.4xlarge |m6g.8xlarge |m6g.12xlarge |m6g.16xlarge |m6g.metal

タイプ	Sizes
M6gd	m6gd.medium |m6gd.large |m6gd.xlarge |m6gd.2xlarge | m6gd.4xlarge |m6gd.8xlarge |m6gd.12xlarge |m6gd.16xl arge |m6gd.metal
M6i	m6i.large |m6i.xlarge |m6i.2xlarge |m6i.4xlarge | m6i.8xlarge |m6i.12xlarge |m6i.16xlarge |m6i.24xla rge |m6i.32xlarge |m6i.metal
M6id	m6id.large |m6id.xlarge |m6id.2xlarge |m6id.4xlarge | m6id.8xlarge |m6id.12xlarge |m6id.16xlarge |m6id.24xl arge |m6id.32xlarge |m6id.metal
M6idn	m6idn.large |m6idn.xlarge |m6idn.2xlarge |m6idn.4xl arge |m6idn.8xlarge |m6idn.12xlarge |m6idn.16xlarge | m6idn.24xlarge |m6idn.32xlarge |m6idn.metal
M6in	m6in.large |m6in.xlarge |m6in.2xlarge |m6in.4xlarge | m6in.8xlarge |m6in.12xlarge |m6in.16xlarge |m6in.24xl arge |m6in.32xlarge |m6in.metal
M7a	m7a.medium |m7a.large |m7a.xlarge |m7a.2xlarge | m7a.4xlarge |m7a.8xlarge |m7a.12xlarge |m7a.16xlarge |m7a.24xlarge |m7a.32xlarge |m7a.48xlarge |m7a.metal -48xl
M7g	m7g.medium |m7g.large |m7g.xlarge |m7g.2xlarge | m7g.4xlarge |m7g.8xlarge |m7g.12xlarge |m7g.16xlarge |m7g.metal
M7gd	m7gd.medium |m7gd.large |m7gd.xlarge |m7gd.2xlarge | m7gd.4xlarge |m7gd.8xlarge |m7gd.12xlarge |m7gd.16xl arge |m7gd.metal
M7i	m7i.large |m7i.xlarge |m7i.2xlarge |m7i.4xlarge | m7i.8xlarge |m7i.12xlarge |m7i.16xlarge |m7i.24xla rge |m7i.48xlarge |m7i.metal-24xl |m7i.metal-48xl

タイプ	Sizes
M7i-flex	m7i-flex.large |m7i-flex.xlarge |m7i-flex.2xlarge | m7i-flex.4xlarge |m7i-flex.8xlarge
Mac1	mac1.metal
Mac2	mac2.metal
Mac2-m2	mac2-m2.metal
Mac2-m2pro	mac2-m2pro.metal
T2	t2.nano|t2.micro|t2.small|t2.medium |t2.large| t2.xlarge |t2.2xlarge
T3	t3.nano|t3.micro|t3.small|t3.medium |t3.large| t3.xlarge |t3.2xlarge
T3a	t3a.nano|t3a.micro |t3a.small |t3a.medium |t3a.large |t3a.xlarge |t3a.2xlarge
T4g	t4g.nano|t4g.micro |t4g.small |t4g.medium |t4g.large |t4g.xlarge |t4g.2xlarge

コンピューティングの最適化

タイプ	Sizes
C5	c5.large|c5.xlarge |c5.2xlarge |c5.4xlarge | c5.9xlarge |c5.12xlarge |c5.18xlarge |c5.24xlarge | c5.metal
C5a	c5a.large |c5a.xlarge |c5a.2xlarge |c5a.4xlarge | c5a.8xlarge |c5a.12xlarge |c5a.16xlarge |c5a.24xla rge

タイプ	Sizes
C5ad	c5ad.large | c5ad.xlarge | c5ad.2xlarge | c5ad.4xlarge | c5ad.8xlarge | c5ad.12xlarge | c5ad.16xlarge | c5ad.24xlarge
C5d	c5d.large | c5d.xlarge | c5d.2xlarge | c5d.4xlarge | c5d.9xlarge | c5d.12xlarge | c5d.18xlarge | c5d.24xlarge | c5d.metal
C5n	c5n.large | c5n.xlarge | c5n.2xlarge | c5n.4xlarge | c5n.9xlarge | c5n.18xlarge | c5n.metal
C6a	c6a.large | c6a.xlarge | c6a.2xlarge | c6a.4xlarge | c6a.8xlarge | c6a.12xlarge | c6a.16xlarge | c6a.24xlarge | c6a.32xlarge | c6a.48xlarge | c6a.metal
C6g	c6g.medium | c6g.large | c6g.xlarge | c6g.2xlarge | c6g.4xlarge | c6g.8xlarge | c6g.12xlarge | c6g.16xlarge | c6g.metal
C6gd	c6gd.medium | c6gd.large | c6gd.xlarge | c6gd.2xlarge | c6gd.4xlarge | c6gd.8xlarge | c6gd.12xlarge | c6gd.16xlarge | c6gd.metal
C6gn	c6gn.medium | c6gn.large | c6gn.xlarge | c6gn.2xlarge | c6gn.4xlarge | c6gn.8xlarge | c6gn.12xlarge | c6gn.16xlarge
C6i	c6i.large | c6i.xlarge | c6i.2xlarge | c6i.4xlarge | c6i.8xlarge | c6i.12xlarge | c6i.16xlarge | c6i.24xlarge | c6i.32xlarge | c6i.metal
C6id	c6id.large | c6id.xlarge | c6id.2xlarge | c6id.4xlarge | c6id.8xlarge | c6id.12xlarge | c6id.16xlarge | c6id.24xlarge | c6id.32xlarge | c6id.metal

タイプ	Sizes
C6in	c6in.large |c6in.xlarge |c6in.2xlarge |c6in.4xlarge | c6in.8xlarge |c6in.12xlarge |c6in.16xlarge |c6in.24xlarge |c6in.32xlarge |c6in.metal
C7a	c7a.medium |c7a.large |c7a.xlarge |c7a.2xlarge | c7a.4xlarge |c7a.8xlarge |c7a.12xlarge |c7a.16xlarge |c7a.24xlarge |c7a.32xlarge |c7a.48xlarge |c7a.metal-48xl
C7g	c7g.medium |c7g.large |c7g.xlarge |c7g.2xlarge | c7g.4xlarge |c7g.8xlarge |c7g.12xlarge |c7g.16xlarge |c7g.metal
C7gd	c7gd.medium |c7gd.large |c7gd.xlarge |c7gd.2xlarge | c7gd.4xlarge |c7gd.8xlarge |c7gd.12xlarge |c7gd.16xlarge |c7gd.metal
C7gn	c7gn.medium |c7gn.large |c7gn.xlarge |c7gn.2xlarge | c7gn.4xlarge |c7gn.8xlarge |c7gn.12xlarge |c7gn.16xlarge
C7i	c7i.large |c7i.xlarge |c7i.2xlarge |c7i.4xlarge | c7i.8xlarge |c7i.12xlarge |c7i.16xlarge |c7i.24xlarge |c7i.48xlarge |c7i.metal-24xl |c7i.metal-48xl

メモリ最適化

タイプ	Sizes
R5	r5.large|r5.xlarge |r5.2xlarge |r5.4xlarge | r5.8xlarge |r5.12xlarge |r5.16xlarge |r5.24xlarge | r5.metal

タイプ	Sizes
R5a	r5a.large | r5a.xlarge | r5a.2xlarge | r5a.4xlarge | r5a.8xlarge | r5a.12xlarge | r5a.16xlarge | r5a.24xlarge
R5ad	r5ad.large | r5ad.xlarge | r5ad.2xlarge | r5ad.4xlarge | r5ad.8xlarge | r5ad.12xlarge | r5ad.16xlarge | r5ad.24xlarge
R5b	r5b.large | r5b.xlarge | r5b.2xlarge | r5b.4xlarge | r5b.8xlarge | r5b.12xlarge | r5b.16xlarge | r5b.24xlarge | r5b.metal
R5d	r5d.large | r5d.xlarge | r5d.2xlarge | r5d.4xlarge | r5d.8xlarge | r5d.12xlarge | r5d.16xlarge | r5d.24xlarge | r5d.metal
R5dn	r5dn.large | r5dn.xlarge | r5dn.2xlarge | r5dn.4xlarge | r5dn.8xlarge | r5dn.12xlarge | r5dn.16xlarge | r5dn.24xlarge | r5dn.metal
R5n	r5n.large | r5n.xlarge | r5n.2xlarge | r5n.4xlarge | r5n.8xlarge | r5n.12xlarge | r5n.16xlarge | r5n.24xlarge | r5n.metal
R6a	r6a.large | r6a.xlarge | r6a.2xlarge | r6a.4xlarge | r6a.8xlarge | r6a.12xlarge | r6a.16xlarge | r6a.24xlarge | r6a.32xlarge | r6a.48xlarge | r6a.metal
R6g	r6g.medium | r6g.large | r6g.xlarge | r6g.2xlarge | r6g.4xlarge | r6g.8xlarge | r6g.12xlarge | r6g.16xlarge | r6g.metal
R6gd	r6gd.medium | r6gd.large | r6gd.xlarge | r6gd.2xlarge | r6gd.4xlarge | r6gd.8xlarge | r6gd.12xlarge | r6gd.16xlarge | r6gd.metal

タイプ	Sizes
R6i	r6i.large | r6i.xlarge | r6i.2xlarge | r6i.4xlarge | r6i.8xlarge | r6i.12xlarge | r6i.16xlarge | r6i.24xlarge | r6i.32xlarge | r6i.metal
R6idn	r6idn.large | r6idn.xlarge | r6idn.2xlarge | r6idn.4xlarge | r6idn.8xlarge | r6idn.12xlarge | r6idn.16xlarge | r6idn.24xlarge | r6idn.32xlarge | r6idn.metal
R6in	r6in.large | r6in.xlarge | r6in.2xlarge | r6in.4xlarge | r6in.8xlarge | r6in.12xlarge | r6in.16xlarge | r6in.24xlarge | r6in.32xlarge | r6in.metal
R6id	r6id.large | r6id.xlarge | r6id.2xlarge | r6id.4xlarge | r6id.8xlarge | r6id.12xlarge | r6id.16xlarge | r6id.24xlarge | r6id.32xlarge | r6id.metal
R7a	r7a.medium | r7a.large | r7a.xlarge | r7a.2xlarge | r7a.4xlarge | r7a.8xlarge | r7a.12xlarge | r7a.16xlarge | r7a.24xlarge | r7a.32xlarge | r7a.48xlarge | r7a.metal-48xl
R7g	r7g.medium | r7g.large | r7g.xlarge | r7g.2xlarge | r7g.4xlarge | r7g.8xlarge | r7g.12xlarge | r7g.16xlarge | r7g.metal
R7gd	r7gd.medium | r7gd.large | r7gd.xlarge | r7gd.2xlarge | r7gd.4xlarge | r7gd.8xlarge | r7gd.12xlarge | r7gd.16xlarge | r7gd.metal
R7i	r7i.large | r7i.xlarge | r7i.2xlarge | r7i.4xlarge | r7i.8xlarge | r7i.12xlarge | r7i.16xlarge | r7i.24xlarge | r7i.48xlarge | r7i.metal-24xl | r7i.metal-48xl
R7iz	r7iz.large | r7iz.xlarge | r7iz.2xlarge | r7iz.4xlarge | r7iz.8xlarge | r7iz.12xlarge | r7iz.16xlarge | r7iz.32xlarge | r7iz.metal-16xl | r7iz.metal-32xl

タイプ	Sizes
U-3tb1	u-3tb1.56xlarge
U-6tb1	u-6tb1.56xlarge | u-6tb1.112xlarge | u-6tb1.metal
U-9tb1	u-9tb1.112xlarge | u-9tb1.metal
U-12tb1	u-12tb1.112xlarge | u-12tb1.metal
U-18tb1	u-18tb1.112xlarge | u-18tb1.metal
U-24tb1	u-24tb1.112xlarge | u-24tb1.metal
X1	x1.16xlarge | x1.32xlarge
X2gd	x2gd.medium | x2gd.large | x2gd.xlarge | x2gd.2xlarge | x2gd.4xlarge | x2gd.8xlarge | x2gd.12xlarge | x2gd.16xlarge | x2gd.metal
X2idn	x2idn.16xlarge | x2idn.24xlarge | x2idn.32xlarge | x2idn.metal
X2iedn	x2iedn.xlarge | x2iedn.2xlarge | x2iedn.4xlarge | x2iedn.8xlarge | x2iedn.16xlarge | x2iedn.24xlarge | x2iedn.32xlarge | x2iedn.metal
X2iezn	x2iezn.2xlarge | x2iezn.4xlarge | x2iezn.6xlarge | x2iezn.8xlarge | x2iezn.12xlarge | x2iezn.metal
X1e	x1e.xlarge | x1e.2xlarge | x1e.4xlarge | x1e.8xlarge | x1e.16xlarge | x1e.32xlarge
z1d	z1d.large | z1d.xlarge | z1d.2xlarge | z1d.3xlarge | z1d.6xlarge | z1d.12xlarge | z1d.metal

ストレージの最適化

タイプ	Sizes
D2	d2.xlarge | d2.2xlarge | d2.4xlarge | d2.8xlarge
D3	d3.xlarge | d3.2xlarge | d3.4xlarge | d3.8xlarge
D3en	d3en.xlarge | d3en.2xlarge | d3en.4xlarge | d3en.6xlarge | d3en.8xlarge | d3en.12xlarge
H1	h1.2xlarge | h1.4xlarge | h1.8xlarge | h1.16xlarge
I3	i3.large | i3.xlarge | i3.2xlarge | i3.4xlarge | i3.8xlarge | i3.16xlarge | i3.metal
I3en	i3en.large | i3en.xlarge | i3en.2xlarge | i3en.3xlarge | i3en.6xlarge | i3en.12xlarge | i3en.24xlarge | i3en.metal
I4g	i4g.large | i4g.xlarge | i4g.2xlarge | i4g.4xlarge | i4g.8xlarge | i4g.16xlarge
I4i	i4i.large | i4i.xlarge | i4i.2xlarge | i4i.4xlarge | i4i.8xlarge | i4i.12xlarge | i4i.16xlarge | i4i.24xlarge | i4i.32xlarge | i4i.metal
Im4gn	im4gn.large | im4gn.xlarge | im4gn.2xlarge | im4gn.4xlarge | im4gn.8xlarge | im4gn.16xlarge
Is4gen	is4gen.medium | is4gen.large | is4gen.xlarge | is4gen.2xlarge | is4gen.4xlarge | is4gen.8xlarge

高速コンピューティング

タイプ	Sizes
DL1	dl1.24xlarge

タイプ	Sizes
DL2q	dl2q.24xlarge
F1	f1.2xlarge |f1.4xlarge |f1.16xlarge
G3	g3.4xlarge |g3.8xlarge |g3.16xlarge
G4ad	g4ad.xlarge |g4ad.2xlarge |g4ad.4xlarge |g4ad.8xlarge |g4ad.16xlarge
G4dn	g4dn.xlarge |g4dn.2xlarge |g4dn.4xlarge |g4dn.8xlarge |g4dn.12xlarge |g4dn.16xlarge |g4dn.metal
G5	g5.xlarge |g5.2xlarge |g5.4xlarge |g5.8xlarge | g5.12xlarge |g5.16xlarge |g5.24xlarge |g5.48xlarge
G5g	g5g.xlarge |g5g.2xlarge |g5g.4xlarge |g5g.8xlarge | g5g.16xlarge |g5g.metal
Inf1	inf1.xlarge |inf1.2xlarge |inf1.6xlarge |inf1.24xlarge
Inf2	inf2.xlarge |inf2.8xlarge |inf2.24xlarge |inf2.48xlarge
P2	p2.xlarge |p2.8xlarge |p2.16xlarge
P3	p3.2xlarge |p3.8xlarge |p3.16xlarge
P3dn	p3dn.24xlarge
P4d	p4d.24xlarge
P4de	p4de.24xlarge
P5	p5.48xlarge
Trn1	trn1.2xlarge |trn1.32xlarge
Trn1n	trn1n.32xlarge

タイプ	Sizes
VT1	vt1.3xlarge | vt1.6xlarge | vt1.24xlarge

高性能コンピューティング

タイプ	Sizes
Hpc6a	hpc6a.48xlarge
Hpc6id	hpc6id.32xlarge
Hpc7a	hpc7a.12xlarge | hpc7a.24xlarge | hpc7a.48xlarge | hpc7a.96xlarge
Hpc7g	hpc7g.4xlarge | hpc7g.8xlarge | hpc7g.16xlarge

旧世代のインスタンス

アプリケーションが旧世代のインスタンスタイプ用に最適化されており、アップグレードはまだこれからというユーザー向けに、Amazon Web Services では、それらの旧世代のインスタンスを提供しています。最高のパフォーマンスを得るには、現世代のインスタンスタイプの使用をお勧めしますが、以下の旧世代のインスタンスタイプも引き続きサポートします。適切なアップグレードとなる現世代のインスタンスタイプの詳細については、「旧世代のインスタンス」を参照してください。

タイプ	Sizes
A1	a1.medium | a1.large | a1.xlarge | a1.2xlarge | a1.4xlarge | a1.metal
C1	c1.medium | c1.xlarge
C3	c3.large | c3.xlarge | c3.2xlarge | c3.4xlarge | c3.8xlarge
C4	c4.large | c4.xlarge | c4.2xlarge | c4.4xlarge | c4.8xlarge
G2	g2.2xlarge | g2.8xlarge

タイプ	Sizes
I2	i2.xlarge |i2.2xlarge |i2.4xlarge |i2.8xlarge
M1	m1.small|m1.medium |m1.large|m1.xlarge
M2	m2.xlarge |m2.2xlarge |m2.4xlarge
M3	m3.medium |m3.large|m3.xlarge |m3.2xlarge
M4	m4.large|m4.xlarge |m4.2xlarge |m4.4xlarge |m4.10xlarge | m4.16xlarge
R3	r3.large|r3.xlarge |r3.2xlarge |r3.4xlarge |r3.8xlarge
R4	r4.large|r4.xlarge |r4.2xlarge |r4.4xlarge |r4.8xlarge | r4.16xlarge
T1	t1.micro

ハードウェア仕様

インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

お客様のニーズに最適なインスタンスタイプを決定するには、インスタンスを起動し、独自のベンチマークアプリケーションを使用することをお勧めします。支払いはインスタンス秒単位であるため、決定する前に複数のインスタンスタイプをテストすると、便利なうえ、コストを抑えることができます。決定を行った後でも、ニーズが変化したときは、インスタンスタイプを変更できます。詳細については、「インスタンスタイプを変更する」を参照してください。

プロセッサの機能

Intel プロセッサの機能

Intel プロセッサで実行される Amazon EC2 インスタンスには、以下の機能が含まれる場合があります。次のプロセッサ機能のすべてが、すべてのインスタンスタイプでサポートされているわけではありません。各インスタンスタイプで使用できる機能の詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。

- インテルの AES New Instructions (AES-NI) — インテルの AES-NI 暗号化命令セットは、オリジナルの Advanced Encryption Standard (AES) アルゴリズムを改良し、より高速なデータ保護とより優れたセキュリティを提供します。現行世代の全 EC2 インスタンスがこのプロセッサ機能をサポートしています。
- Intel Advanced Vector Extensions (Intel AVX、Intel AVX2、および Intel AVX-512) — 浮動小数点 (FP) 集約型のアプリケーション用に設計された命令セット拡張で、Intel AVX および Intel AVX2 は 256 ビット、Intel AVX-512 は 512 ビットです。Intel AVX 命令は、画像およびオーディオ/ビデオ処理、科学的シミュレーション、財務分析、および 3D モデリングと分析などのアプリケーションに対するパフォーマンスを向上させます。これらの機能は、HVM AMI で起動されたインスタンスのみで利用できます。
- Intel Turbo Boost Technology — Intel Turbo Boost Technology プロセッサーは、定格の動作周波数よりも高速にコアを自動的に実行します。
- Intel Deep Learning Boost (Intel DL Boost) — AI の深層学習のユースケースを高速化します。第 2 世代インテル Xeon スケーラブルプロセッサでは、新しいベクトルニューラルネットワーク命令 (VNNI/INT8) を使ってインテル AVX-512 を拡張します。これにより、画像認識/セグメント化、物体検出、音声認識、言語翻訳、レコメンデーションシステム、強化学習などにおけるディープラーニングの推論パフォーマンスは、前世代のインテル Xeon スケーラブルプロセッサ (FP32) よりも大幅に向上します。VNNI はすべての Linux ディストリビューションと互換性があるわけではありません。

  `M5n`、`R5n`、`M5dn`、`M5zn`、`R5b`、`R5dn`、`D3`、`D3en` および `C6i` インスタンスでは、VNNI をサポートしています。`C5` および `C5d` インスタンスでは、`12xlarge`、`24xlarge`、`metal` インスタンスのみ VNNI をサポートしています。

これは、64 ビット CPU の命名に関する業界の慣習の影響で、ややわかりにくいものになっています。チップ製造元の Advanced Micro Devices (AMD) は、Intel x86 命令セットをベースとして商業的に初めて成功した 64 ビットアーキテクチャを導入しました。その結果、このアーキテクチャーはチップ製造元にかかわらず AMD64 と幅広く呼ばれています。Windows および複数の Linux ディストリビューションがこの慣習に従っています。インスタンスが Intel ハードウェアで実行されているにもかかわらず、Ubuntu または Windows を実行しているインスタンスの内部システム情報に CPU アーキテクチャが AMD64 と表示されるのはこのためです。

AMI 仮想化タイプ

インスタンスの仮想化タイプは、インスタンスの起動に使用する AMI によって決まります。現行世代のインスタンスタイプは、ハードウェア仮想マシン (HVM) のみをサポートしています。以前の

世代のインスタンスタイプの中には、準仮想化 (PV) をサポートするものがあり、一部の AWS リージョンは PV インスタンスをサポートしています。詳細については、Linux AMI 仮想化タイプ を参照してください。

最適なパフォーマンスを得るために、HVM AMI を使用することをお勧めします。さらに、拡張ネットワーキングのメリットを活用するには、HVM AMI が必要です。HVM 仮想化は、AWS プラットフォームによって提供されるハードウェアアシストテクノロジーを使用します。HVM 仮想化を使用すると、ゲスト VM はネイティブハードウェアプラットフォーム上で動作しているかのように動作します。ただし、パフォーマンスの向上のために PV ネットワークとストレージドライバは使用します。

Nitro System 上に構築されたインスタンス

Nitro System は、AWS が構築した、高パフォーマンス、高可用性、高度なセキュリティを実現するハードウェアとソフトウェアコンポーネントのコレクションです。詳細については、AWS Nitro System をご参照ください。

Nitro System は、ベアメタル機能を備えることで、仮想化オーバーヘッドを排除するとともに、ホストハードウェアへのフルアクセスを要求するワークロードをサポートします。ベアメタルインスタンスは、次の用途に適しています。

- 仮想環境で利用できない、または完全にサポートされていない低レベルのハードウェア機能 (例: Intel VT) へのアクセスを必要とするワークロード
- ライセンスやサポートを目的として非仮想化環境で実行する必要があるアプリケーション

Nitro コンポーネント

Nitro System には、以下のコンポーネントが含まれます。

- Nitro Card
  - ローカル NVMe ストレージボリューム
  - ネットワーキングハードウェアのサポート
  - 管理
  - モニタリング
  - セキュリティ
- Nitro Security Chip (マザーボードに統合)

- Nitro Hypervisor - メモリと CPU の割り当てを管理し、ほとんどのワークロードのベアメタルと見分けがつかないようなパフォーマンスを提供する軽量ハイパーバイザー。

仮想インスタンス

次の仮想インスタンスは Nitro System で構築されています。

- 汎用: A1 | M5 | M5a | M5ad | M5d | M5dn | M5n | M5zn | M6a | M6g | M6gd | M6i | M6id | M6idn | M6in | M7a | M7g | M7gd | M7i | M7i-flex | T3 | T3a | T4g
- コンピューティング最適化: C5 | C5a | C5ad | C5d | C5n | C6a | C6g | C6gd | C6gn | C6i | C6id | C6in | C7a | C7g | C7gd | C7gn | C7i
- メモリ最適化: R5 | R5a | R5ad | R5b | R5d | R5dn | R5n | R6a | R6g | R6gd | R6i | R6idn | R6in | R6id | R7a | R7g | R7gd | R7i | R7iz | U-3tb1 | U-6tb1 | U-9tb1 | U-12tb1 | U-18tb1 | U-24tb1 | X2gd | X2idn | X2iedn | X2iezn | z1d
- ストレージ最適化: D3 | D3en | I3en | I4g | I4i | Im4gn | Is4gen
- 高速コンピューティング: DL1 | DL2q | G4ad | G4dn | G5 | G5g | Inf1 | Inf2 | P3dn | P4d | P4de | P5 | Trn1 | Trn1n | VT1
- ハイパフォーマンスコンピューティング: Hpc6a | Hpc6id | Hpc7a | Hpc7g

ベアメタルインスタンス

次のベアメタルインスタンスは Nitro System で構築されています。

- 汎用: `a1.metal` | `m5.metal` | `m5d.metal` | `m5dn.metal` | `m5n.metal` | `m5zn.metal` | `m6a.metal` | `m6g.metal` | `m6gd.metal` | `m6i.metal` | `m6id.metal` | `m6idn.metal` | `m6in.metal` | `m7a.metal-48xl` | `m7g.metal` | `m7gd.metal` | `m7i.metal-24xl` | `m7i.metal-48xl` | `mac1.metal` | `mac2.metal` | `mac2-m2.metal` | `mac2-m2pro.metal`
- コンピューティング最適化: `c5.metal` | `c5d.metal` | `c5n.metal` | `c6a.metal` | `c6g.metal` | `c6gd.metal` | `c6i.metal` | `c6id.metal` | `c6in.metal` | `c7a.metal-48xl` | `c7g.metal` | `c7gd.metal` | `c7i.metal-24xl` | `c7i.metal-48xl`
- メモリ最適化: `r5.metal` | `r5b.metal` | `r5d.metal` | `r5dn.metal` | `r5n.metal` | `r6a.metal` | `r6g.metal` | `r6gd.metal` | `r6i.metal` | `r6idn.metal` | `r6in.metal` | `r6id.metal` | `r7a.metal-48xl` | `r7g.metal` | `r7gd.metal` | `r7i.metal-24xl` | `r7i.metal-48xl` | `r7iz.metal-16xl` | `r7iz.metal-32xl` | `u-6tb1.metal` | `u-9tb1.metal` | `u-12tb1.metal` | `u-18tb1.metal` | `u-24tb1.metal` | `x2gd.metal` | `x2idn.metal` | `x2iedn.metal` | `x2iezn.metal` | `z1d.metal`

- ストレージ最適化: `i3.metal` | `i3en.metal` | `i4i.metal`
- 高速コンピューティング `g4dn.metal` | `g5g.metal`

詳細はこちら

詳細については、以下の動画を参照してください。

- AWS re:Invent 2017: Amazon EC2 Nitro System アーキテクチャ
- AWS re:Invent 2017: Amazon EC2 ベアメタルインスタンス
- AWS re:Invent 2019: 次世代 Amazon EC2 の強化: Nitro System の詳細
- AWS re:Inforce 2019: Nitro アーキテクチャのセキュリティ上のメリット

ネットワーキング機能とストレージ機能

インスタンスタイプを選択すると、使用できるネットワーキング機能とストレージ機能が決まります。インスタンスタイプの情報を取得するには、describe-instance-types コマンドを使用します。

ネットワーキング機能

- IPv6 は C1、M1、M2、M3、および T1 を除くすべてのインスタンスタイプをサポートします。
- インスタンスタイプのネットワーキングと帯域幅のパフォーマンスを最大化するには、次のことを実行できます。
  - サポートされるインスタンスタイプをクラスタープレイスメントグループで起動し、ハイパフォーマンスコンピューティング (HPC) アプリケーション用にインスタンスを最適化します。共通のクラスタープレイスメントグループのインスタンスは、高帯域幅、低レイテンシーのネットワーキングから利点を得られます。詳細については、プレイスメントグループ を参照してください。
  - サポートされる現行世代のインスタンスタイプ用の拡張ネットワーキングを有効にして、パケット毎秒 (PPS) のパフォーマンスを大幅に高め、ネットワークのストレスとレイテンシーを低減することができます。詳細については、Linux での拡張ネットワーキング を参照してください。
- 拡張ネットワーキングに対して有効になっている現行世代のインスタンスタイプには、次のネットワーキングパフォーマンス属性があります。
  - 同じリージョン内でのプライベート IPv4 または IPv6 を介したトラフィックでは、シングルフロートラフィックで 5 Gbps、マルチフロートラフィックで最大 25 Gbps をサポートしています (インスタンスタイプによって異なります)。

  - 同じリージョン内でのインスタンスと Amazon S3 バケットとの間では、パブリック IP アドレス空間または VPC エンドポイントを介したトラフィックに、使用可能なすべてのインスタンスの集計帯域幅を使用できます。
- サポートされる最大送信単位 (MTU) は、インスタンスタイプごとに異なります。すべての Amazon EC2 インスタンスタイプは、標準イーサネット V2 1500 MTU フレームをサポートします。すべての現行世代のインスタンスは 9001 MTU、またはジャンボ フレームをサポートし、一部の旧世代のインスタンスも同様にそれらをサポートします。詳細については、EC2 インスタンスのネットワークの最大送信単位 (MTU) を参照してください。

ストレージ機能

- インスタンスタイプの中には、EBS ボリュームとインスタンスストアボリュームをサポートするものや、EBS ボリュームのみをサポートするものがあります。インスタンスストアボリュームをサポートする一部のインスタンスタイプは、ソリッドステートドライブ (SSD) を使用して非常に高いランダム I/O パフォーマンスを提供します。インスタンスタイプによっては、NVMe インスタンスストアボリュームをサポートしていないものがあります。インスタンスタイプによっては、NVMe EBS ボリュームをサポートしていないものがあります。詳細については、「Amazon EBS および NVMe」および「NVMe SSD ボリューム」を参照してください。
- 一部のインスタンスタイプを EBS 最適化インスタンスとして起動することで、Amazon EBS I/O 専用に追加のキャパシティーを取得できます。インスタンスタイプの中には、デフォルトで EBS に最適化されるものがあります。詳細については、Amazon EBS 最適化インスタンスを使用するを参照してください。

ネットワーキング機能とストレージ機能の概要

Amazon EC2 インスタンスでサポートされるネットワークおよびストレージ機能については、「Amazon EC2 インスタンスタイプガイド」の以下の項目を参照してください。

- 汎用インスタンス
- コンピュート最適化インスタンス
- メモリ最適化インスタンス
- ストレージ最適化インスタンス
- 高速コンピューティングインスタンス
- ハイパフォーマンスコンピューティングインスタンス
- 旧世代のインスタンス

インスタンス制限

リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。

デフォルトの制限の詳細については、「Amazon EC2 で実行できるインスタンスの数はいくつですか?」を参照してください。

現在の制限の表示、または現在の制限の引き上げリクエストについての詳細については、「Amazon EC2 の Service Quotas」を参照してください。

汎用インスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

汎用インスタンスは、コンピューティング、メモリ、およびネットワーキングリソースをバランスよく備えており、広範囲のワークロードに使用できます。

A1 インスタンス

これらのインスタンスは、Arm エコシステムがサポートするスケールアウト型ワークロードに最適です。これらのインスタンスは、次の用途に適しています。

- ウェブサーバー
- コンテナ化されたマイクロサービス

M5、M5a インスタンス

これらのインスタンスは、クラウドインフラストラクチャの実現に最適で、コンピューティング、メモリ、およびネットワーキングリソースをバランスよく備えており、クラウドにデプロイされる広範なアプリケーションに使用されます。このインスタンスは、以下に最適です。

- 小規模および中規模のデータベース
- 追加のメモリを必要とするデータ処理タスク

- キャッシュフリート
- SAP、Microsoft SharePoint、クラスターコンピューティング、その他のエンタープライズアプリケーションなどのバックエンドサーバー

詳細については、「Amazon EC2 M5 インスタンス」を参照してください。

m5.metal、m5n.metal、m5zn.metal などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスできます。

M5zn

このインスタンスは、非常に高いシングルスレッドパフォーマンス、高スループット、低レイテンシーネットワーキングの恩恵を受けるアプリケーションに最適です。このインスタンスは、以下に最適です。

- ゲーム
- ハイパフォーマンスコンピューティング
- シミュレーションモデリング

詳細については、「Amazon EC2 M5 インスタンス」を参照してください。

M6g インスタンスと M6gd インスタンス

これらのインスタンスは AWS Graviton2 プロセッサを搭載しており、幅広い汎用ワークロードに対応するバランスの取れたコンピューティング、メモリ、ネットワーキングを提供します。このインスタンスは、以下に最適です。

- アプリケーションサーバー
- マイクロサービス
- ゲームサーバー
- 中規模のデータストア
- キャッシュフリート

m6g.metal などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスできます。詳細については、「Amazon EC2 M6g インスタンス」を参照してください。

M6i および M6id インスタンス

これらのインスタンスは、次のような汎用ワークロードに適しています。

- アプリケーションサーバーやウェブサーバー
- マイクロサービス
- ハイパフォーマンスコンピューティング
- アプリケーション開発
- 小規模および中規模のデータベース
- キャッシュフリート

`m6i.metal` などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスできます。

詳細については、「Amazon EC2 M6i インスタンス」を参照してください。

M6in および M6idn インスタンス

これらのインスタンスは、次のようなネットワークを多用するワークロードに適しています。

- 高性能なファイルシステム
- 分散したウェブスケールインメモリキャッシュ
- キャッシュフリート
- リアルタイムビッグデータ分析
- 5G ユーザープレーン機能 (UPF) などの通信アプリケーション

詳細については、「Amazon EC2 M6i インスタンス」を参照してください。

M7i インスタンス

M7i インスタンスは、第 7 世代の Amazon EC2 インスタンスポートフォリオを拡張し、x86 ベースのオプションを含むようにしています。これらのインスタンスには、AWS 専用のカスタム第 4 世代インテル Xeon スケーラブルプロセッサー (Sapphire Rapids) が搭載されています。M6i インスタンスと比較して、コストパフォーマンスが最大 15% 向上します。M7i インスタンスは、ウェブサーバー、アプリケーションサーバー、マイクロサービス、小規模データストアなどの汎用ワークロードの実行に適しています。

詳細については、「Amazon EC2 M7i インスタンス」を参照してください。

M7i-flex インスタンス

汎用ワークロードの大部分は、最新世代のパフォーマンスの恩恵を受けていますが、コンピューティングリソースを完全には活用していません。Amazon EC2 flex インスタンスは、このようなワークロードを実行するための最初の選択肢として理想的です。Flex インスタンスは、コストが最適化された Amazon EC2 インスタンスのバリアントで、大半の一般的なワークロードでコストパフォーマンスのメリットと低料金を実現する最も簡単な方法です。

M7i-flex インスタンスは、第 4 世代インテル Xeon スケーラブルプロセッサー (Sapphire Rapids) をベースにした最初の Amazon EC2 Flex インスタンスであり、同等の M インスタンスと比較してコストを節約できます。M7i-flex インスタンスは、最大 32 個の vCPU と最大 128 GiB のメモリを搭載しており、バランスの取れたコンピューティング、メモリ、およびネットワークリソースを提供します。これらのインスタンスは、次のような汎用ワークロードに適しています。

- アプリケーションサーバーやウェブサーバー
- マイクロサービス
- 仮想デスクトップ
- アプリケーション開発
- データベース
- モバイルアプリケーション
- バッチワークロード

詳細については、「Amazon EC2 M7i and M7i-flex instances」を参照してください。

M7g と M7gd インスタンス

これらのインスタンスは AWS Graviton3 プロセッサを搭載しており、幅広い汎用ワークロードに対応するバランスの取れたコンピューティング、メモリ、ネットワーキングを提供します。このインスタンスは、以下に最適です。

- アプリケーションサーバー
- マイクロサービス
- ゲームサーバー
- 中規模のデータストア

- キャッシュフリート

詳細については、「Amazon EC2 M7g インスタンス」を参照してください。

M7a インスタンス

M7a インスタンスは、AWS Nitro System 上に構築され、アプリケーションサーバー、マイクロサービス、ゲームサーバー、中規模データストア、アプリケーション開発環境、キャッシュフリートなど、高パフォーマンス、高スループット、低レイテンシーのネットワーキングの恩恵を受けるアプリケーションに最適です。

詳細については、「Amazon EC2 M7a Instances」を参照してください。

x86 ベースおよび Apple シリコン (M1、M2、および M2 Pro) Mac インスタンス

EC2 Mac インスタンスは、iPhone、iPad、Mac、Vision Pro、Apple Watch、Apple TV、Safari などの Apple プラットフォーム用アプリケーションの開発、構築、テスト、署名に最適です。

詳細については、「Amazon EC2 Mac インスタンス」を参照してください。

T2、T3、T3a、T4g インスタンス

これらのインスタンスは、ベースラインレベルの CPU パフォーマンスを維持し、ワークロードの必要に応じてより高いレベルまでバーストすることもできます。無制限インスタンスは、必要なときに、任意の期間にわたって高い CPU パフォーマンスを保持できます。詳細については、バーストパフォーマンスインスタンス を参照してください。このインスタンスは、以下に最適です。

- ウェブサイトとウェブアプリケーション
- コードリポジトリ
- 開発、ビルド、テスト、およびステージング環境
- マイクロサービス

詳細については、「Amazon EC2 T2 インスタンス」、「Amazon EC2 T3 インスタンス」、および「Amazon EC2 T4g インスタンス」を参照してください。

コンテンツ

- Amazon EBS I/O パフォーマンス
- SSD ベースのインスタンスストアボリュームの I/O パフォーマンス
- リリースノート
- バーストパフォーマンスインスタンス

ハードウェア仕様

仮想中央処理ユニット (vCPU) は、仮想マシン (VM) に割り当てられた物理 CPU の一部を表します。x86 インスタンスの場合、コアごとに 2 つの vCPU があります。Graviton インスタンスの場合、コアごとに 1 つの vCPU があります。

ハードウェアの仕様については、「Amazon EC2 インスタンスタイプガイド」の「汎用インスタンス」を参照してください。

インスタンスのパフォーマンス

EBS 最適化インスタンスは、インスタンスからの Amazon EBS I/O とその他のネットワークトラフィックとの競合を排除することによって、EBS ボリュームの安定した高パフォーマンスを実現できます。一部の汎用インスタンスは、追加料金なしでデフォルトで EBS 最適化されています。詳細については、Amazon EBS 最適化インスタンスを使用する を参照してください。

一部の汎用インスタンスタイプでは、Linux でプロセッサの C ステートと P ステートを制御できます。C ステートは非アクティブ時のコアのスリープレベルを制御し、P ステートは希望するコアからのパフォーマンス (CPU 周波数) を制御します。詳細については、「EC2 インスタンスのプロセッサのステート制御」を参照してください。

Flex インスタンスのパフォーマンス

M7i-flex インスタンスでは、コンピューティング、メモリ、ネットワークリソースがバランスが良く構成されており、広範な汎用アプリケーションを実行するための費用対効果が最も高い方法が提供されています。M7i-flex インスタンスは、信頼性の高い CPU リソースを提供して、40% のベースラインの CPU パフォーマンスを実現します。これは、汎用ワークロードの大部分のコンピューティング要件を満たすように設計されています。ワークロードにさらなるパフォーマンスが必要な場合、M7i-flex インスタンスはベースライン CPU を超え、24 時間の 95 パーセントの時間で最大 100 パーセントの CPU 性能を実現する機能を提供します。

長期間にわたって一貫してベースラインを上回る高い CPU 使用率で稼働している M7i-flex インスタンスでは、最大バースト時の CPU スループットが徐々に低下する場合があります。

ネットワークパフォーマンス

サポートされているインスタンスタイプで拡張ネットワーキングを有効にすると、レイテンシーとネットワークジッターを低減し、パケット毎秒 (PPS) のパフォーマンスを高めることができます。ほとんどのアプリケーションでは、高いレベルのネットワークパフォーマンスが一貫して必要なわけではありませんが、データの送受信時にアクセスする帯域幅を増やすことでメリットを得られます。詳細については、「Linux での拡張ネットワーキング」を参照してください。

ネットワークの仕様については、「Amazon EC2 インスタンスタイプガイド」の「汎用インスタンス」を参照してください。

Amazon EBS I/O パフォーマンス

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、Amazon EBS ボリュームの最高のパフォーマンスを実現します。

詳細については、「Amazon EBS 最適化インスタンスを使用する」を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスストアボリュームは、インスタンスの存続中のみ使用できます。インスタンスを停止、休止、または終了すると、アプリケーションとそのインスタンスストアボリュームのデータは消去されます。インスタンスストアボリュームの重要なデータは、定期的にバックアップまたはレプリケートすることをお勧めします。詳細については、「Amazon EC2 インスタンスストア」および「SSD インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仕様については、「Amazon EC2 インスタンスタイプガイド」の「汎用インスタンス」を参照してください。

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これは、SSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは、利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み操作の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を

新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは、複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つには、SSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームには、オーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすには、ボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。

リリースノート

- Nitro System、M4、`t2.large` 以上のサイズ、 `t3.large` 以上のサイズ、および `t3a.large` 以上のサイズのインスタンスタイプ上に構築されたインスタンスには、64 ビットの HVM AMI が必要です。これらのインスタンスはハイメモリであるため、そのキャパシティーを活用するには 64 ビットのオペレーティングシステムが必要です。HVM AMI は、ハイメモリインスタンスタイプの準仮想化 (PV) AMI よりも優れたパフォーマンスを提供します。さらに、拡張ネットワーキングを利用するには、HVM AMI を使用する必要があります。
- Nitro System 上に構築されたインスタンスには、次の要件があります。
  - NVMe ドライバーがインストールされている必要があります。
  - Elastic Network Adapter (ENA) ドライバーがインストールされている必要があります。

  以下の Linux AMI はこれらの要件を満たしています。
  - AL2023
  - Amazon Linux 2
  - Amazon Linux AMI 2018.03 以降
  - `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

> **Note**
>
> AWS Graviton ベースのインスタンスタイプには、`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

  - Red Hat Enterprise Linux 7.4 以降
  - SUSE Linux Enterprise Server 12 SP2 以降
  - CentOS 7.4.1708 以降
  - FreeBSD 11.1 以降
  - Debian GNU/Linux 9 以降
- AWS Graviton プロセッサを使用するインスタンスには、以下の要件があります。
  - 64 ビット Arm アーキテクチャ用の AMI を使用する必要があります。
  - ACPI テーブルを含む UEFI による起動と、PCI デバイスの ACPI ホットプラグをサポートしている必要があります。

  以下の Linux と の AMI では、上記の要件を満たしています。
  - Amazon Linux 2 (64 ビット Arm)
  - Ubuntu 16.04 以降 (64 ビット Arm)
  - Red Hat Enterprise Linux 8.0 以降 (64 ビット Arm)
  - SUSE Linux Enterprise Server 15 以降 (64 ビット Arm)
  - Debian 10 以降 (64 ビット Arm)
- M6i インスタンスで最良のパフォーマンスを発揮させるには、ENA ドライバーのバージョン 2.2.9 以降を使用する必要があります。1.2 もしくは バージョンより前のENAドライバをこれらのインスタンスとともに使用すると、ネットワークインターフェイスのアタッチメントが失敗します。互換性のある ENA ドライバが利用できる AMI を以下に示します。
  - AL2023
  - Amazon Linux 2 (カーネル 4.14.186 以降)
  - Ubuntu 20.04 (カーネル 5.4.0-1025-aws 以降)
  - Red Hat Enterprise Linux 8.3 (カーネル 4.18.0-240.1.1.el8_3.ARCH 以降)
  - SUSE Linux Enterprise Server 15 SP2 (カーネル 5.3.18-24.15.1 以降)
- Amazon EC2 Mac インスタンスは macOS Mojave (バージョン 10.14) と macOS Catalina (バージョン 10.15) と macOS Big Sur (バージョン 11) をサポートしています。

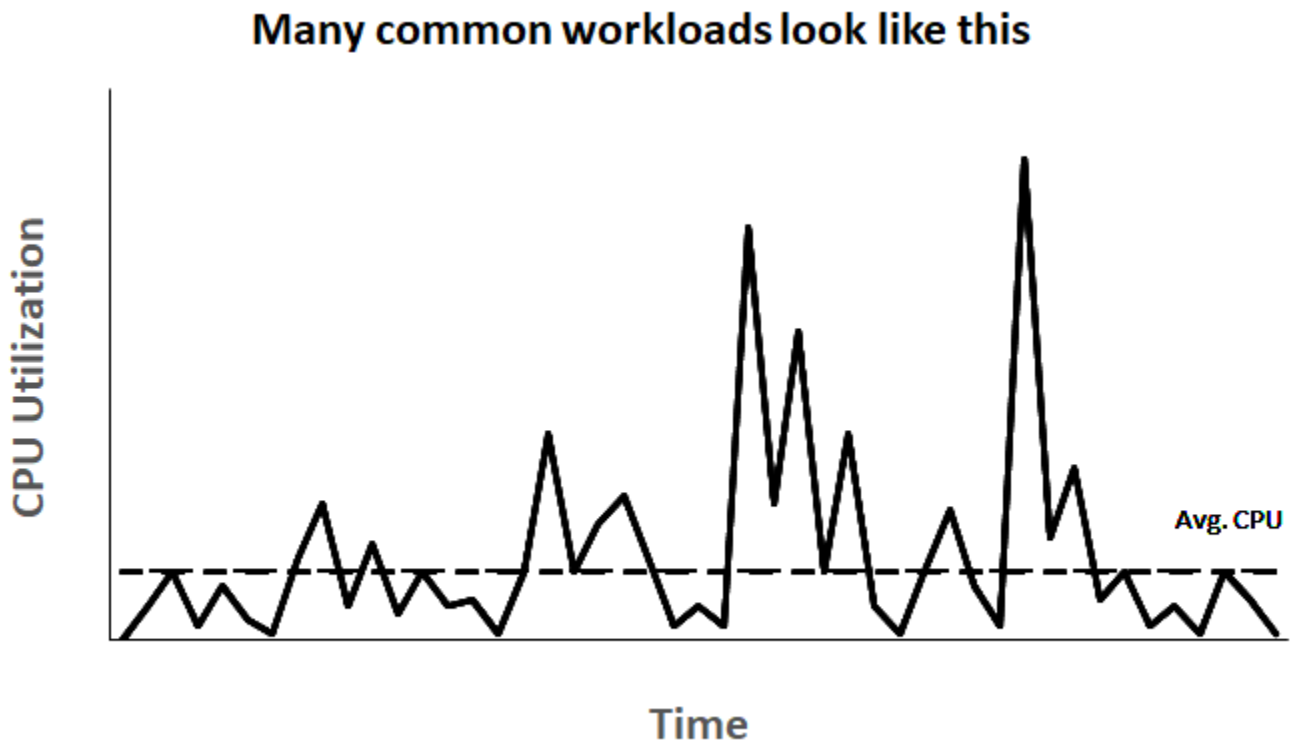
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。詳細については、「インスタンスボリューム数の制限」を参照してください。
- ベアメタルインスタンスを起動すると、基盤となるサーバーが起動します。これには、すべてのハードウェアやファームウェアコンポーネントの確認が含まれます。つまり、インスタンスが実行状態になってからネットワーク経由で使用できるようになるまでに 20 分かかることがあります。
- EBS ボリュームまたはセカンダリネットワークインターフェイスを、ベアメタルインスタンスにアタッチ (または、そこからデタッチ) するには、PCIe のネイティブホットプラグがサポートされている必要があります。PCIe のネイティブホットプラグは、Amazon Linux 2 および最新バージョンの Amazon Linux AMI でサポートされています。それ以前のバージョンではサポートされていません。次の Linux カーネル設定オプションを有効にする必要があります。

```
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_PCIEASPM=y
```

- ベアメタルインスタンスでは、I/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。アップストリームの Linux カーネルと最新の Amazon Linux AMI は、このデバイスをサポートしています。また、ベアメタルインスタンスでは、システムが PCI ベースのシリアルデバイスを自動的に使用できるようにする ACPI SPCR テーブルも使用できます。最新の Windows AMI では、自動的に PCI ベースのシリアルデバイスが使用されます。
- Nitro System 上に構築されたインスタンスには、API リクエストによるクリーンシャットダウンをサポートするための system-logind または acpid がインストールされている必要があります。
- リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。詳細については、Amazon EC2 の「よくある質問」の「Amazon EC2 で実行できるインスタンス数の上限は」を参照してください。

バーストパフォーマンスインスタンス

多くの汎用ワークロードでは、平均的な状態はビジーではないので、持続的で高いレベルの CPU パフォーマンスを必要としません。以下のグラフは、現在 AWS クラウドのお客様が実行している多くの一般的なワークロードにおける、CPU の使用率を示しています。



これらの CPU 使用率が低 ~ 中程度のワークロードが、CPU サイクルを浪費するような場合には、使用量以上の料金が発生することになります。こういった問題に対処するには、低コストのバースト汎用インスタンス (T インスタンス) を活用します。

T インスタンスファミリーは、ベースラインの CPU パフォーマンスを提供しながら、いつでも必要な時間だけ、能力をベースライン以上にバーストさせる機能を備えています。ベースライン CPU は汎用ワークロードにおける大部分のニーズを満たすように構成されており、大規模なマイクロサービス、ウェブサーバー、中小規模のデータベース、データのログ記録、コードリポジトリ、仮想デスクトップ、開発/テスト環境、ビジネスクリティカルなアプリケーションなどに対応できます。T インスタンスでは、コンピューティング、メモリ、ネットワークリソースがバランスしているので、CPU 使用率が低 ~ 中程度の広範な汎用アプリケーションを実行するための、費用対効果が最も高い方法が提供されます。コストの面では、M インスタンスと比較しても最大 15% の節約となります。また、2 個の vCPU と 0.5 GiB のメモリで動作する小型かつ経済的なインスタンスサイズを選択すれば、さらにコストの削減が可能です。nano、micro、Small、medium など小型の T インスタンスサイズは、必要なメモリが少なく、また想定される CPU 使用率も高くないワークロードに適しています。

> **Note**
>
> このトピックは、バースト可能な CPU について説明します。バースト可能なネットワークパフォーマンスの詳細については、「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。

EC2 バーストインスタンスタイプ

EC2 バーストインスタンスには、T4g、T3a、T3 インスタンスタイプ、および前世代の T2 インスタンスタイプが含まれます。

T4g インスタンスタイプは、最新世代のバーストインスタンスです。このタイプは、パフォーマンスに対して最良の価格設定が行われており、すべての EC2 インスタンスタイプ中でも、最も低いコストでの利用が可能です。T4g インスタンスタイプは、ARM ベースの AWSGraviton2 プロセッサーで動作します。また、オペレーティングシステムベンダー、独立系ソフトウェアベンダー、さらに一般的な AWS サービスとアプリケーションからの、広範なエコシステムサポートも利用できます。

次の表に、各バーストインスタンスタイプ間の主な違いをまとめます。

タイプ	説明	プロセッサファミリ
最新世代		
T4g	最低コストの EC2 インスタンスタイプで、T3 と比較した場合コストパフォーマンスは最大 40% 向上し、使用料金は 20% 低減	Arm Neoverse N1 コア搭載の AWS Graviton2 プロセッサ
T3a	T3 インスタンスと比較してコストを 10% 低減する、最低料金の x86 ベースインスタンス	AMD 第1世代 EPYC プロセッサー
T3	x86 ワークロードでピーク時に最良のコストパフォーマンスを発揮、前世代の T2 インスタンスと比較した場合コス	インテル Xeon スケーラブル (Skylake、Cascade Lake プロセッサー)

タイプ	説明	プロセッサファミリ
	トパフォーマンスが最大 30% 向上	
前の世代		
T2	前世代のバーストインスタンス	インテル Xeon プロセッサ

インスタンスの料金体系と、その他の仕様については、「Amazon EC2 の料金」および「Amazon EC2 インスタンスタイプ」を参照してください。バースト可能なネットワークパフォーマンスの詳細については、「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。

アカウントが 12 か月未満の場合は、特定の使用制限内で `t2.micro` インスタンスを無料で使用できます (`t3.micro` を利用できないリージョンでは `t2.micro` インスタンスを使用できます)。詳細については、「AWS 無料利用枠」を参照してください。

T インスタンスでサポートされる購入オプション

- On-Demand Instances
- Reserved Instances
- ハードウェア専有インスタンス (T3 のみ)
- Dedicated Hosts (T3 のみ、`standard`モードのみ)
- スポットインスタンス

詳細については、インスタンス購入オプション を参照してください。

目次

ベストプラクティス

これらのベストプラクティスに従って、バーストパフォーマンスインスタンスの利点を最大限に活用してください。

- 選択するインスタンスのサイズが、オペレーティングシステムおよびアプリケーションの最小メモリ要件を満たしていることを確認します。多量のメモリおよび CPU リソースを消費するグラフィカルユーザーインターフェースを使用するオペレーティングシステム (Windows など) は、多くのユースケースで `t3.micro` 以上のインスタンスサイズを必要とする場合があります。時間の経過とともに、メモリおよび CPU に対するワークロードからの要求が増大した場合のために、T インスタンスには、同じインスタンスタイプで大きなインスタンスサイズにスケールできる柔軟性が備わっています。あるいは、別のインスタンスタイプを選択することも可能です。
- アカウントの AWS Compute Optimizer を有効にし、ワークロードに関する Compute Optimizer 推奨事項を確認します。Compute Optimizer は、パフォーマンスを向上させるためにインスタンスをアップサイズする必要があるかや、コスト削減のためにダウンサイズする必要があるかを評価する際に役立ちます。Compute Optimizer は、シナリオに応じて異なるインスタンスタイプを推奨する場合もあります。詳細については、「AWS Compute Optimizer ユーザーガイド」の「EC2 インスタンスのレコメンデーションの表示」を参照してください。
- 追加の要件については、「リリースノート」を参照してください。

バーストパフォーマンスインスタンスに関する主要な概念と定義

従来の Amazon EC2 インスタンスタイプの場合は CPU リソースが固定されています。一方バーストパフォーマンスインスタンスの場合は、CPU 使用率にベースラインレベルを定義した上で、そのレベルを超えて CPU 使用率をバーストさせることが可能となっています。これにより料金は、ベースラインの CPU 使用率に加えて、バーストとして追加された分に対してのみ支払えば良いことになり、コンピューティングのコストを削減できます。ベースライン使用率とバースト機能は、CPU クレジットで管理します。バーストパフォーマンスインスタンスは、CPU 使用率にクレジットを使用する唯一のインスタンスタイプです。

各バーストパフォーマンスインスタンスは、CPU 使用率がベースラインを下回っている間は継続的にクレジットを獲得し、ベースラインを上回っている間は継続的にクレジットを消費します。獲得または消費されたクレジットの量は、インスタンスの CPU 使用率によって異なります。

- CPU 使用率がベースラインを下回っている場合、獲得するクレジットは消費するクレジットよりも大きくなります。
- CPU 使用率がベースラインと等しい場合、獲得するクレジットは消費するクレジットと等しくなります。

- CPU 使用率がベースラインよりも高い場合、消費するクレジットが獲得するクレジットよりも高くなります。

獲得したクレジットが消費したクレジットよりも大きい場合、その差額は蓄積されたクレジットと呼ばれ、後でベースラインを超えて CPU 使用率をバーストさせる際に使用できます。一方、消費したクレジットが獲得したクレジットよりも多い場合のインスタンスの動作は、クレジット設定モード (スタンダードモードまたは Unlimited モード) によって異なります。

スタンダードモードでは、消費したクレジットが獲得したクレジットよりも多い場合、インスタンスは、ベースラインを越えて CPU 使用率をバーストさせるために、蓄積されたクレジットを使用します。蓄積されたクレジットに残額がない場合、インスタンスは CPU 使用率を徐々にベースラインまで低下させ、より多くのクレジットが蓄積されるまで、ベースラインを超えてバーストすることはできなくなります。

Unlimited モードでは、CPU 使用率がベースラインを超えてバーストした場合、インスタンスは最初に蓄積されたクレジットを使用します。その後、蓄積されたクレジットの残額がなくなった場合には、インスタンスは余剰クレジットを消費してバーストを維持します。その CPU 利用率がベースラインを下回った場合、獲得した CPU クレジットを使用して、先に消費された余剰クレジットの支払いが行われます。CPU クレジットを獲得して余剰クレジットを支払う機能により、Amazon EC2 は 24 時間にわたるインスタンスの CPU 使用率を平均化できるようになります。24 時間の平均 CPU 使用率がベースラインを超えたインスタンスには、vCPU 時間あたりの超過の使用量に対して、均一追加料金が発生します。

コンテンツ

主要な概念と定義

バーストパフォーマンスインスタンスには、以下の主要な概念と定義が適用されます。

CPU 使用率

CPU 使用率とは、割り当てられた EC2 コンピューティングユニットのうち、現在インスタンス上で使用されているものが占める割合のことです。このメトリクスは、割り当てられた CPU サイクルの中で、インスタンスで使用されているサイクルの割合を測定します。CloudWatch の CPU 使用率に関するメトリクスでは、コアごとの CPU 使用率ではなく、インスタンスごとの CPU 使用率を示しています。インスタンスの CPU ベースラインに関する仕様は、インスタンスごとの CPU 使用率とも関連しています。AWS Management Console または AWS CLI を使用して CPU 使用率を測定する方法については、「特定のインスタンスの統計を取得する」を参照してください。

CPU クレジット

vCPU 時間の単位。

例:

1 CPU クレジット = 1 vCPU × 100% 使用率 × 1 分

1 CPU クレジット = 1 vCPU × 50% 使用率 × 2 分

1 CPU クレジット = 2 vCPU × 25% 使用率 × 2 分

ベースライン使用率

ベースライン使用率とは、CPU クレジットの獲得数と CPU クレジットの使用数が一致する場合に、正味のクレジット残高が 0 の状態で CPU を使用できるレベルのことです。ベースライン使用率はベースラインとも呼ばれます。ベースライン使用率は vCPU の使用率のパーセンテージとして表され、次のように計算されます。ベースライン使用率 % = (獲得したクレジットの数 ÷ vCPU の数) ÷ 60 分

各バーストパフォーマンスインスタンスタイプのベースライン使用率については、「クレジットの表」を参照してください。

獲得クレジット

実行中のインスタンスが継続的に獲得するクレジットです。

1 時間あたりの獲得クレジット数 =ベースライン使用率 (%) × vCPU 数 (個)× 60 (分)

例:

vCPU を 2 個使用し、ベースライン使用率が 5% に設定された T3.nano では次の計算のように、1 時間あたり 6 クレジットを獲得します。

2 個の vCPU × 5% のベースライン × 60 分 = 1 時間あたり 6 クレジット

消費または使用されたクレジット

実行中のインスタンスにより継続的に使用されるクレジットです。

1 分あたりに使用される CPU クレジット = vCPU 数 (個) × CPU 使用率 (%) × 1 分

蓄積されたクレジット

インスタンスの使用量がベースラインの使用率よりも少ないので、消費されなかった CPU クレジットです。つまり、蓄積されたクレジット = 獲得クレジット — 使用されたクレジット (ともにベースラインより低い場合)、となります。

例:

仮に t3.nano の CPU 使用率が 2% で、ベースラインである 5% を 1 時間の間下回っていた場合、蓄積されたクレジットは次のように計算されます。

蓄積された CPU クレジット = (1 時間あたりの獲得クレジット — 1 時間あたりの使用クレジット) = 6 — 2 (vCPU 個数) × 2 (CPU 使用率 %) × 60 (分) = 6 — 2.4 = 3.6 (1 時間あたりに蓄積されたクレジット)

クレジット蓄積制限

インスタンスのサイズによって異なりますが、通常は 24 時間以内に獲得できるクレジットの最大数と等しくなります。

例:

t3.nanoの場合、クレジット蓄積制限 = 24 × 6 = 144クレジット

起動クレジット

これは、スタンダードモードに設定された T2 インスタンスにのみ適用されます。起動クレジットは、新しい T2 インスタンスに割り当てられるもので、CPU クレジットの数に制限があります。スタンダードモードで起動することで、ベースラインを超えたバーストが可能になります。

余剰クレジット

蓄積されたクレジットの残高が枯渇したインスタンスが消費するクレジットです。余剰クレジットは、長期間高パフォーマンスを維持するバーストインスタンスのために設計されており、使用できるのは Unlimited モードでのみです。余剰クレジット残高は、Unlimited モードのインスタンスがバーストのために使用した、クレジットの数を判断するために使用されます。

スタンダードモード

クレジットの設定モードです。このモードのインスタンスでは、クレジット残高に蓄積されたクレジットを消費することで、そのベースラインを超えたバーストが可能です。

Unlimited モード

クレジットの設定モードです。必要な期間にわたって高い CPU 使用率を維持することで、インスタンスがベースラインを超えてバーストすることを可能にします。24 時間ごとのインスタンスの平均 CPU 使用率またはインスタンスの存続期間のいずれか短い方の時間で、インスタンスの平均 CPU 使用率がベースライン以下になった場合、1 時間ごとのインスタンス価格は自動的にすべての CPU 使用率スパイクをカバーします。長時間にわたって高い CPU 使用率でインスタンスを実行する場合には、vCPU 時間ごとに均一追加料金が発生します。

次の表は、バーストインスタンスタイプ間の主なクレジットの違いをまとめたものです。

タイプ	サポートされる CPU クレジットのタイプ	クレジットの設定モード	インスタンスの開始から停止までの間に蓄積された CPU クレジットのライフスパン
最新世代			
T4g	獲得クレジット、蓄積されたクレジット、消費されたクレジット、余剰クレジット (Unlimited モードのみ)	スタンダード、Unlimited (デフォルト)	7 日間 (クレジットは、インスタンスが停止した後、7 日間維持されます)
T3a	獲得クレジット、蓄積されたクレジット、消費されたクレジット、余剰クレジット (Unlimited モードのみ)	スタンダード、Unlimited (デフォルト)	7 日間 (クレジットは、インスタンスが停止した後、7 日間維持されます)

タイプ	サポートされる CPU クレジットのタイプ	クレジットの設定モード	インスタンスの開始から停止までの間に蓄積された CPU クレジットのライフスパン
T3	獲得クレジット、蓄積されたクレジット、消費されたクレジット、余剰クレジット (Unlimited モードのみ)	スタンダード、Unlimited (デフォルト)	7 日間 (クレジットは、インスタンスが停止した後、7 日間維持されます)
前の世代			
T2	獲得クレジット、蓄積されたクレジット、使用されたクレジット、起動クレジット (スタンダードモードのみ)、余剰クレジット (Unlimited モードのみ)	スタンダード (デフォルト)、Unlimited	0 日 (インスタンスが停止するとクレジットは失われます)

> **Note**
>
> Dedicated Host で起動される T3 インスタンスでは、Unlimited モードはサポートされていません。

CPU クレジットの獲得

各バーストパフォーマンスインスタンスは、インスタンスサイズに応じて、1 時間当たりの CPU クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。クレジットを蓄積または消費する会計処理もミリ秒レベルの細かさで実施されるため、CPU クレジットの浪費について心配する必要はありません。CPU の短期バーストでは CPU クレジットのごく一部しか使用されません。

バーストパフォーマンスインスタンスが使用する CPU リソースが、ベースライン使用率に必要な CPU リソースよりも少ない場合 (アイドル時など)、未使用の CPU クレジットが CPU クレジット残高に蓄積されます。バーストパフォーマンスインスタンスがベースライン使用率を超えてバーストする必要がある場合は、蓄積されたクレジットを消費します。CPU 使用率を増やす必要がある場合、バーストパフォーマンスインスタンスが蓄積したクレジットが多いほど、ベースラインを超えてバーストできる時間が増えます。

次の表は、バーストパフォーマンスインスタンスのタイプ、1 時間あたりに CPU クレジットを獲得するレート、インスタンスが蓄積できる獲得 CPU クレジットの最大数、インスタンスあたりの vCPU 数、およびコア全体に対する割合で表したベースライン使用率 (単一の vCPU を使用した場合) の一覧です。

インスタンスタイプ	1 時間あたりに受け取る CPU クレジット	蓄積可能な最大獲得クレジット*	vCPU 数	vCPU あたりのベースライン使用率
T2				
t2.nano	3	72	1	5%
t2.micro	6	144	1	10%
t2.small	12	288	1	20%
t2.medium	24	576	2	20%**
t2.large	36	864	2	30%**
t2.xlarge	54	1296	4	22.5%**
t2.2xlarge	81.6	1958.4	8	17%**
T3				
t3.nano	6	144	2	5%**
t3.micro	12	288	2	10%**
t3.small	24	576	2	20%**

インスタンスタイプ	1 時間あたりに受け取る CPU クレジット	蓄積可能な最大獲得クレジット*	vCPU 数	vCPU あたりのベースライン使用率
t3.medium	24	576	2	20%**
t3.large	36	864	2	30%**
t3.xlarge	96	2304	4	40%**
t3.2xlarge	192	4608	8	40%**
T3a				
t3a.nano	6	144	2	5%**
t3a.micro	12	288	2	10%**
t3a.small	24	576	2	20%**
t3a.medium	24	576	2	20%**
t3a.large	36	864	2	30%**
t3a.xlarge	96	2304	4	40%**
t3a.2xlarge	192	4608	8	40%**
T4g				
t4g.nano	6	144	2	5%**
t4g.micro	12	288	2	10%**
t4g.small	24	576	2	20%**
t4g.medium	24	576	2	20%**
t4g.large	36	864	2	30%**

インスタンスタイプ	1 時間あたりに受け取る CPU クレジット	蓄積可能な最大獲得クレジット*	vCPU 数	vCPU あたりのベースライン使用率
t4g.xlarge	96	2304	4	40%**
t4g.2xlarge	192	4608	8	40%**

* 蓄積できるクレジットの数は、24 時間で獲得できるクレジットの数と同じです。

** 表内のベースライン使用率は vCPU 別の割合です。CloudWatch では、CPU 使用率は vCPU 別に表示されます。例えば、ベースラインレベルで動作する t3.large インスタンスの CPU 使用率は、CloudWatch の CPU メトリクスに 30% として表示されます。ベースライン使用率の計算方法については、「ベースライン使用率」を参照してください。

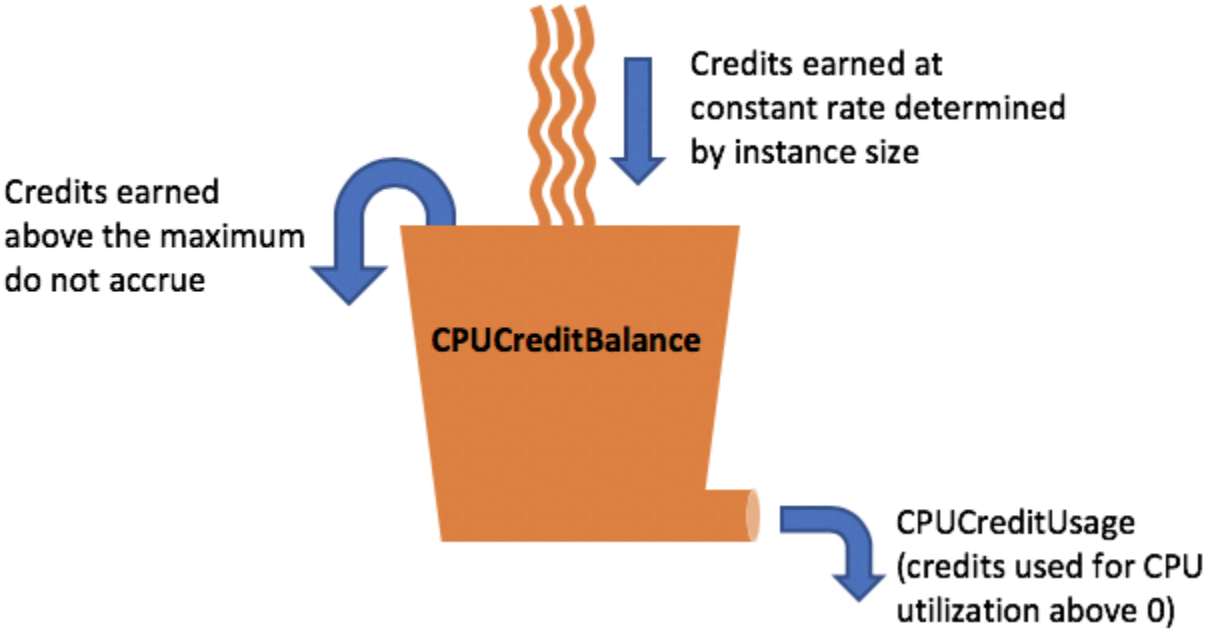
*** 各vCPUは、インテル Xeon コアまたは AMD EPYC コアのスレッドに対応します (T2 と T4g インスタンスを除く)。

CPU クレジットの獲得率

1 時間あたりに獲得する CPU クレジット数は、インスタンスのサイズによって決まります。例えば、t3.nano は 1 時間あたり 6 クレジットを獲得しますが、t3.small は 1 時間あたり 24 クレジットを獲得します。前記の表は、すべてのインスタンスのクレジット獲得率を示しています。

CPU クレジット蓄積制限

実行中のインスタンスで獲得されたクレジットが失効することはありませんが、インスタンスが蓄積できる獲得クレジットの数には制限があります。制限は、CPU クレジット残高により決まります。下記の図に示されているとおり、制限に到達すると、獲得された新しいクレジットはすべて破棄されます。フルバケットは CPU クレジット残高制限を示し、スピルオーバーは制限を超えた新しく獲得されたクレジットを示します。



CPU クレジット残高制限は、各 インスタンスのサイズによって異なります。例えば、t3.micro インスタンスは CPU クレジット残高で最大 288 の獲得 CPU クレジットを蓄積できます。前記の表は、各 インスタンスに累積できる獲得クレジットの最大数を示しています。

T2 スタンダードインスタンスは、起動クレジットも獲得します。起動クレジットは、CPU クレジット残高制限に対してカウントされません。T2 インスタンスがその起動クレジットを消費しておらず、獲得クレジットを蓄積しながら 24 時間以上アイドル状態が続いた場合、CPU クレジット残高は制限を上回って表示されます。詳細については、起動クレジット を参照してください。

T4g、T3a、および T3 インスタンスでは、起動クレジットを獲得させることはできません。これらのインスタンスはデフォルトで unlimited として起動するため、起動クレジットなしでも起動後すぐにバーストできます。Dedicated Host で起動された T3 インスタンスはstandard(デフォルト)unlimitedモードは Dedicated Host の T3 インスタンスではサポートされていません。

CPU 存続期間の蓄積

実行中のインスタンスの CPU クレジットは失効しません。

T2 では、CPU クレジット残高は、インスタンスが停止して起動すると引き継がれません。T2 インスタンスを停止した場合、蓄積されたすべてのクレジットが失われます。

T4g、T3a、および T3では、インスタンスが停止した後も CPU クレジット残高は 7 日間保持され、その後に失われます。7 日以内にインスタンスを起動する場合、クレジットは失われません。

詳細については、「CloudWatch メトリクスの表」の CPUCreditBalance を参照してください。

ベースライン使用率

ベースライン使用率とは、CPU クレジットの獲得数と CPU クレジットの使用数が一致する場合に、正味のクレジット残高が 0 の状態で CPU を使用できるレベルのことです。ベースライン使用率はベースラインとも呼ばれます。

ベースライン使用率は、vCPU の使用に対する割合として表され、次のように計算されます。

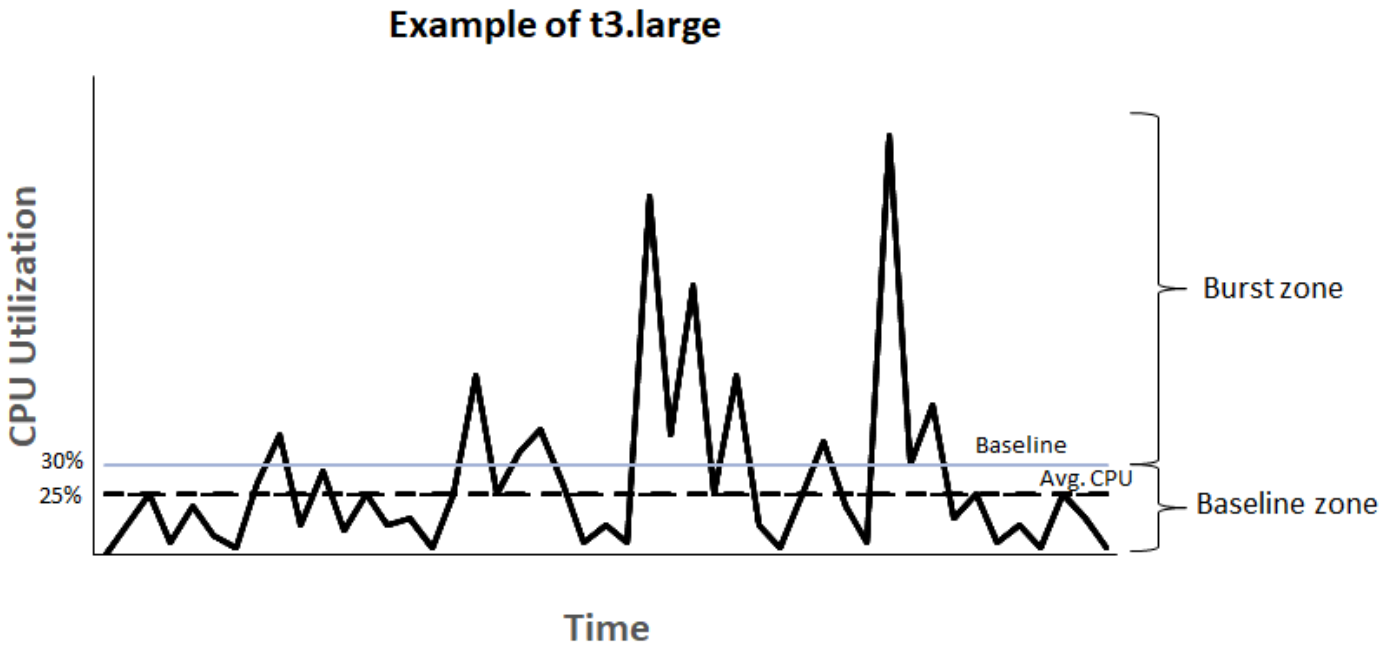
```
(number of credits earned/number of vCPUs)/60 minutes = % baseline
utilization
```

例えば、2 つの vCPU を持つ t3.nano インスタンスが 1 時間あたり 6 クレジットを獲得すると、ベースライン使用率は 5% になります。これは、次のように計算されます。

```
(6 credits earned/2 vCPUs)/60 minutes = 5% baseline utilization
```

2 つの vCPU を持つ t3.large インスタンスが 1 時間あたり 36 クレジットを獲得すると、ベースライン使用率は 30% になります ((36/2)/60)。

次のグラフは、平均 CPU 使用率がベースラインを下回っている t3.large の例を示しています。



バーストパフォーマンスインスタンスの Unlimited モード

unlimited として設定したバーストパフォーマンスインスタンスは、必要に応じた期間にわたり、高い CPU 使用率を保持できます。24 時間ごとのインスタンスの平均 CPU 使用率またはインスタンスの存続期間のいずれか短い方の時間で、インスタンスの平均 CPU 使用率がベースライン以下になった場合、1 時間ごとのインスタンス価格は自動的にすべての CPU 使用率スパイクをカバーします。

汎用のワークロードではほとんどの場合、unlimitedとして設定されたインスタンスは追加料金なしで十分なパフォーマンスを提供します。長時間にわたって高い CPU 使用率でインスタンスを実行する場合には、vCPU 時間ごとに均一追加料金が発生します。料金の詳細については、「Amazon EC2 の料金」および「T2/T3/T4g Unlimited モードの料金」、「」を参照してください。

t2.micro もしくは t3.micro インスタンスを AWS 無料利用枠の範囲で unlimited モードにより使用している場合、ローリング期間の 24 時間における平均使用率が、そのインスタンスのベースライン使用率を超過すると料金が発生することがあります。

T4g、T3a、および T3 インスタンスは (デフォルトを変更しない限り) デフォルトで unlimited として起動します。24 時間の平均 CPU 使用率がベースラインを超えた場合は、余剰クレジットに対して課金されます。スポットインスタンス を unlimited として起動し、CPU クレジットを計上するためのアイドル時間を待たず、すぐに短期間だけ使用する場合には、余剰クレジットの料金が発生します。コストの増加を抑えるには、スポットインスタンス を標準モードで起動することをお勧めします。詳細については、「余剰クレジットにより料金が発生することがある」および「バーストパフォーマンスインスタンス」を参照してください。

> **Note**
>
> Dedicated Host で起動された T3 インスタンスはstandard(デフォルト)unlimitedモードは Dedicated Host の T3 インスタンスではサポートされていません。

コンテンツ

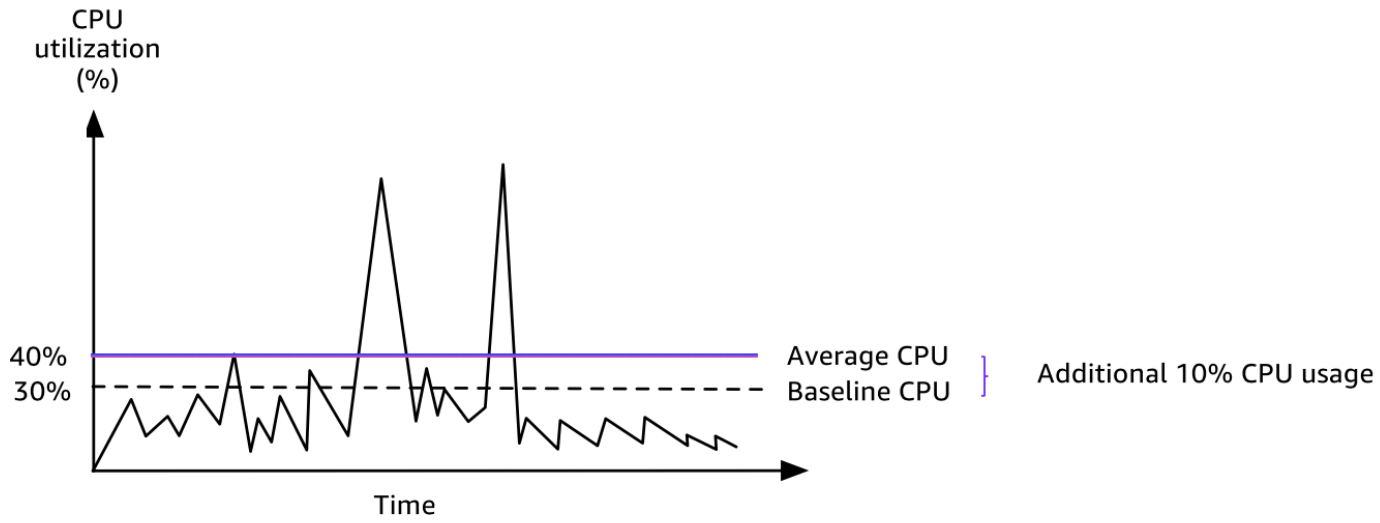
Unlimited モードの概念

unlimited モードはバーストパフォーマンスインスタンスのクレジットの設定オプションです。これにより、実行中または停止中のインスタンスをいつでも有効または無効にできます。各 AWS リージョンのアカウントレベルで、バーストパフォーマンスインスタンスファミリーごとに、デフォルトのクレジットオプションとして unlimited を設定できます。アカウント内のすべての新しいバーストパフォーマンスインスタンスは、このデフォルトのクレジットオプションを使用して起動されます。

無制限のバーストパフォーマンスインスタンスの仕組み

unlimited として設定したバーストパフォーマンスインスタンスが CPU クレジット残高を使い切った場合、ベースラインを超えてバーストするには余剰クレジットを使用できます。その CPU 利用率がベースラインを下回った場合、獲得した CPU クレジットを使用して、先に消費された余剰クレジットの支払いが行われます。CPU クレジットを獲得して余剰クレジットを支払う機能により、Amazon EC2 は 24 時間にわたるインスタンスの CPU 使用率を平均化できるようになります。24 時間の平均 CPU 使用率がベースラインを超えたインスタンスには、vCPU 時間あたりの超過の使用量に対して、均一追加料金が発生します。

以下のグラフは、t3.large の CPU 使用率を示します。t3.large のベースラインの CPU 使用率は 30% です。インスタンスが 24 時間にわたって平均 30% 以下の CPU 使用率で実行されている場合、コストはインスタンスの 1 時間あたりの料金ですでにカバーされているため、追加料金はかかりません。ただし、ここでのグラフに示されているように、24 時間の平均 CPU 使用率 40% でインスタンスが実行されている場合、そのインスタンスでの超過 CPU 使用量 10% に対しては、vCPU 時間あたりに均一追加料金が発生します。



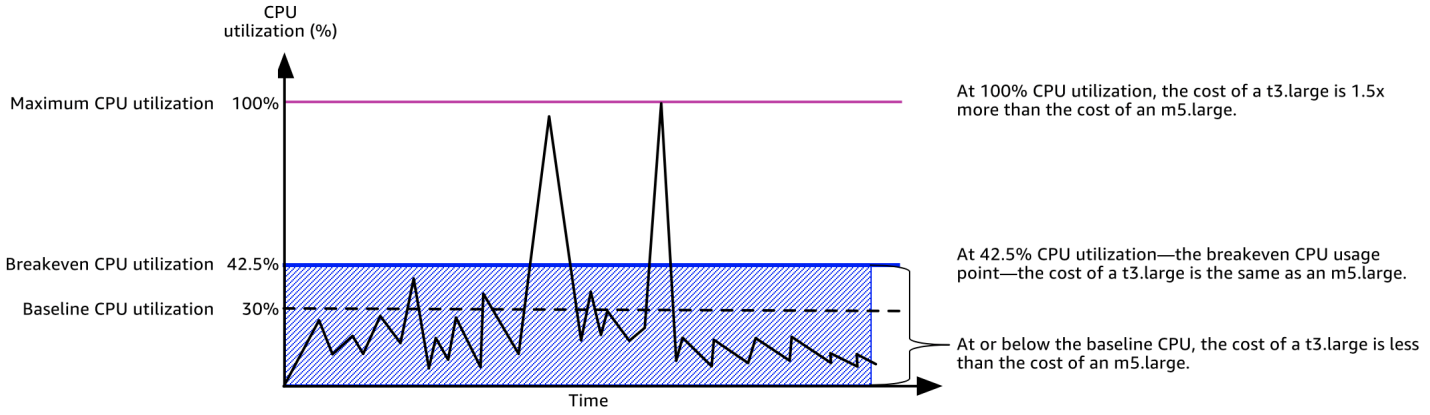
各インスタンスタイプの vCPU あたりのベースライン使用率、および各インスタンスタイプが獲得するクレジット数の詳細については、「クレジットの表」を参照してください。

Unlimited モードと固定 CPU を使用する場合

unlimited モード (T3 など) でバーストパフォーマンスインスタンスと、固定パフォーマンスインスタンス (M5 など) のどちらを使用するかを決める場合は、損益分岐点 CPU 使用率を判断する必要があります。バーストパフォーマンスインスタンスの損益分岐点 CPU 使用率は、バーストパフォーマンスインスタンスが固定パフォーマンスインスタンスと同じコストになるポイントです。損益分岐点 CPU 使用率は、次のことを判断するのに役立ちます。

- 24 時間の平均 CPU 使用率が損益分岐点の CPU 使用率以下である場合は、バーストパフォーマンスインスタンスを unlimited モードで使用すると、固定パフォーマンスインスタンスと同じパフォーマンスを維持しながら、バーストパフォーマンスインスタンスを低価格で使用できます。
- 24 時間の平均 CPU 使用率が損益分岐点 CPU 使用率を上回る場合、バーストパフォーマンスインスタンスは、同等サイズの固定パフォーマンスインスタンスよりもコストが高くなります。T3 インスタンスが 100% CPU で継続的にバーストする場合、同等サイズの M5 インスタンスの約 1.5 倍の価格を支払うことになります。

次のグラフは、t3.large のコストが m5.large と同じ場合の損益分岐点の CPU 使用率を示しています。t3.large の損益分岐点の CPU 使用率は 42.5% です。平均 CPU 使用率が 42.5% の場合、t3.large の実行コストは m5.large と同じです。平均 CPU 使用率が 42.5% を超える場合は、より高価になります。ワークロードの平均 CPU 使用率が 42.5% 未満であれば、t3.large と同じパフォーマンスを得ながら、m5.large を低価格で使用することができます。



次の表は、unlimited モードまたは固定パフォーマンスインスタンスでバーストパフォーマンスインスタンスを使用する方が安価な場合を判断できるように、損益分岐点 CPU 使用率のしきい値を計算する方法を示しています。表の列には A から K のラベルが付けられています。

インスタンスタイプ	vCPU	T3 の料金 */時間	M5 の料金 */時間	料金の違い	vCPU あたりの T3 ベースライン使用率 (%)	余剰クレジットに対する vCPU 時間あたりの料金	vCPU 時間 (分) あたりの料金	vCPU ごとに利用可能な追加のバースト (分)	利用可能な追加 CPU (%)	損益分岐 CPU %
A	B	C	D	E = D - C	F	G	H = G / 60	I = E / H	J = (I / 60) / B	K = F + J
t3.large	2	0.0835 USD	0.096 USD	0.0125 USD	30%	0.05 USD	0.000833 USD	15	12.5%	42.5%

* 料金は us-east-1 および Linux OS に基づいています。

テーブルは以下の情報を提供します。

- 列 A は、インスタンスタイプ t3.large を示します。
- 列 B は、t3.large の vCPU の数を示します。
- 列 C は、1 時間あたりの t3.large の料金を示します。
- 列 D は、1 時間あたりの m5.large の料金を示します。
- 列 E は、t3.large と m5.large の差額を示しています。
- 列 F は、t3.large の vCPU あたりのベースライン使用率 (30%) を示しています。ベースラインでは、インスタンスの 1 時間あたりのコストが CPU 使用率のコストになります。
- G 列は、獲得したクレジットを使い切った後のインスタンスが 100% の CPU 使用率でバーストした場合に請求される、vCPU 時間あたりの均一追加料金を示しています。
- H 列は、獲得したクレジットを使い切った後のインスタンスが 100% の CPU 使用率でバーストした場合に請求される、vCPU 分あたりの均一追加料金を示しています。
- 列 I は、t3.large と同じ 1 時間あたりの料金が発生している間に、100% CPU で m5.large が 1 時間あたりにバーストできる追加の時間 (分) を示しています。
- J 列は、インスタンスが m5.large と同じ 1 時間あたりの料金が発生している間にバーストする可能性がある、ベースラインを超えた追加の CPU 使用率 (%) を示しています。
- 列 K は、t3.large 以上支払わなくても m5.large がバーストする可能性がある損益分岐点 CPU 使用率 (%) を示しています。この使用率を超えた t3.large のコストは m5.large よりも高くなります。

次の表は、同様のサイズの M5 インスタンスタイプと比較した、T3 インスタンスタイプの損益分岐点 CPU 使用率 (%) を示しています。

T3 インスタンスタイプ	M5 と比較した T3 の損益分岐点 CPU 使用率 (%)
t3.large	42.5%
t3.xlarge	52.5%
t3.2xlarge	52.5%

余剰クレジットにより料金が発生することがある

インスタンスの平均 CPU 使用率がベースライン以下の場合、インスタンスに追加料金は発生しません。インスタンスは 24 時間でクレジット最大数を獲得 (例えば、t3.micro インスタンスは 24 時間で最大 288 クレジット獲得可能) するため、課金されることなく余剰クレジットを最大まで消費できます。

ただし、CPU 使用率がベースラインを上回ったままの場合、インスタンスは消費した余剰クレジットを支払うのに十分なクレジットを獲得できません。支払われない余剰クレジットに対して、vCPU 時間ごとに均一追加料金が発生します。料金の詳細については、「T2/T3/T4g Unlimited モードの料金」および「」を参照してください。

先に消費された余剰クレジットは、以下のいずれかの状況に当てはまると料金が発生します。

- 消費された余剰クレジットが、インスタンスが 24 時間に獲得できる最大クレジット数を超えている。最大数を越えて消費された余剰クレジットは、時間の最後に課金されます。
- インスタンスが停止または終了した。
- インスタンスは unlimited から standard に切り替わります。

消費された余剰クレジットは、CloudWatch メトリクス CPUSurplusCreditBalance により追跡されます。課金された余剰クレジットは、CloudWatch メトリクス CPUSurplusCreditsCharged で追跡できます 。詳細については、バーストパフォーマンスインスタンスの追加 CloudWatch メトリクス を参照してください。

T2 Unlimited インスタンスの起動クレジットはありません

T2 スタンダードインスタンスが起動クレジットを受け取っても、T2 Unlimited インスタンスは起動クレジットを受け取りません。T2 Unlimited インスタンスは、24 時間のローリング枠内または存続期間のどちらか短いほうで平均 CPU 使用率がベースラインを越えない限り、追加料金なしでいつでもベースラインを超えるバーストができます。したがって、T2 Unlimited インスタンスは、起動直後の高パフォーマンスを実現するために起動クレジットを必要としません。

T2 インスタンスが standard から unlimited に切り替えられた場合、残りの CPUCreditBalance が引き継がれる前に、蓄積された起動クレジットが CPUCreditBalance から削除されます。

T4g、T3a および T3 インスタンスは、Unlimited モードをサポートしているため、起動クレジットを受け取りません。T4g、T3a、および T3 インスタンスのクレジット設定を Unlimited モードにするこ

とで、ベースラインを超えてバーストさせるために必要な量の CPU リソースを、必要な期間だけ使用できるようになります。

無制限モードの有効化

実行中または停止中のインスタンスで、unlimited から standard、standard から unlimited へいつでも切り替えることができます。詳細については、「バーストパフォーマンスインスタンスを無制限またはスタンダードとして起動する」および「バーストパフォーマンスインスタンスのクレジット指定の変更」を参照してください。

各 unlimited リージョンのアカウントレベルで、バーストパフォーマンスインスタンスファミリーごとに、デフォルトのクレジットオプションとして AWS を設定できます。アカウント内のすべての新しいバーストパフォーマンスインスタンスは、このデフォルトのクレジットオプションを使用して起動されます。詳細については、アカウントのクレジット指定のデフォルト設定 を参照してください。

Amazon EC2 コンソールまたは unlimited を使用して、バーストパフォーマンスインスタンスが standard あるいは AWS CLI のどちらで設定されているを確認できます。詳細については、「バーストパフォーマンスインスタンスのクレジット指定の表示」および「デフォルトのクレジット指定の表示」を参照してください。

Unlimited とスタンダードを切り替えるとクレジットはどうなるか

CPUCreditBalance は、インスタンスが蓄積したクレジットの数を追跡する CloudWatch メトリクスです。CPUSurplusCreditBalance は、インスタンスが消費した余剰クレジットの数を追跡する CloudWatch メトリクスです。

unlimited として設定されたインスタンスを standard に変更すると、以下の状況が発生します

- CPUCreditBalance 値は変更されずに引き継がれます。
- CPUSurplusCreditBalance 値にはすぐに課金されます。

standard インスタンスが unlimited に切り替えられると、以下の状況が発生します。

- CPUCreditBalance 値に含まれる、蓄積された獲得クレジットが引き継がれます。
- T2 スタンダードインスタンスでは、起動クレジットがすべて CPUCreditBalance 値から削除され、蓄積された獲得クレジットを含む残りの CPUCreditBalance 値が引き継がれます。

クレジット使用状況のモニタリング

インスタンスが、ベースラインが提供するよりも多くクレジットを消費していないか確認するには、CloudWatch メトリクスを使用して使用率を追跡し、クレジット使用量を通知する時間ごとのアラームを設定できます。詳細については、バーストパフォーマンスインスタンスの CPU クレジットをモニタリングする を参照してください。

Unlimited モードの例

次の例では、`unlimited` として設定されているインスタンスのクレジットの使用について説明します。

例

- 例 1: T3 Unlimited でのクレジット使用についての説明
- 例 2: T2 Unlimited でのクレジット使用についての説明

例 1: T3 Unlimited でのクレジット使用についての説明

この例では、`t3.nano` として起動した `unlimited` インスタンスの CPU 使用率、獲得クレジットおよび余剰クレジットを使用して CPU 使用率を保持する方法を示します。

`t3.nano` インスタンスは、24 時間のローリング期間に渡って最大で 144 CPU クレジットを獲得し、それを 144 分の vCPU 使用と引き換えることができます。CPU クレジット残高 (CloudWatch メトリクス `CPUCreditBalance` で示される) が消耗すると、余剰 CPU クレジット — まだ獲得していない — を消費して必要なだけバーストします。`t3.nano` インスタンスは 24 時間あたり最大 144 クレジットを獲得するため、すぐに課金されることなく余剰クレジットを最大まで消費できます。CPU クレジットを 144 以上消費した場合、差分については時間の最後に課金されます。

以下のグラフにある例の目的は、`CPUCreditBalance` を使いきった後でも余剰クレジットを使用してインスタンスをバーストさせる方法を示すことです。以下のワークフローは、グラフの番号付きの点を参照します。

P1 – グラフの 0 時において、インスタンスは `unlimited` として起動され、すぐにクレジットを獲得します。このインスタンスは起動時からアイドル状態になり (CPU 使用率は 0%)、クレジットは消費されません。すべての未消費のクレジットはクレジット残高に蓄積されます。最初の 24 時間は、`CPUCreditUsage` は 0 で、`CPUCreditBalance` 値は、最大の 144 に達します。

P2 – 次の 12 時間では、CPU 使用率はベースラインの 5% を下回る 2.5% です。インスタンスは消費するよりも多くのクレジットを獲得しますが、`CPUCreditBalance` 値は、最大 144 クレジットを超えることはできません。

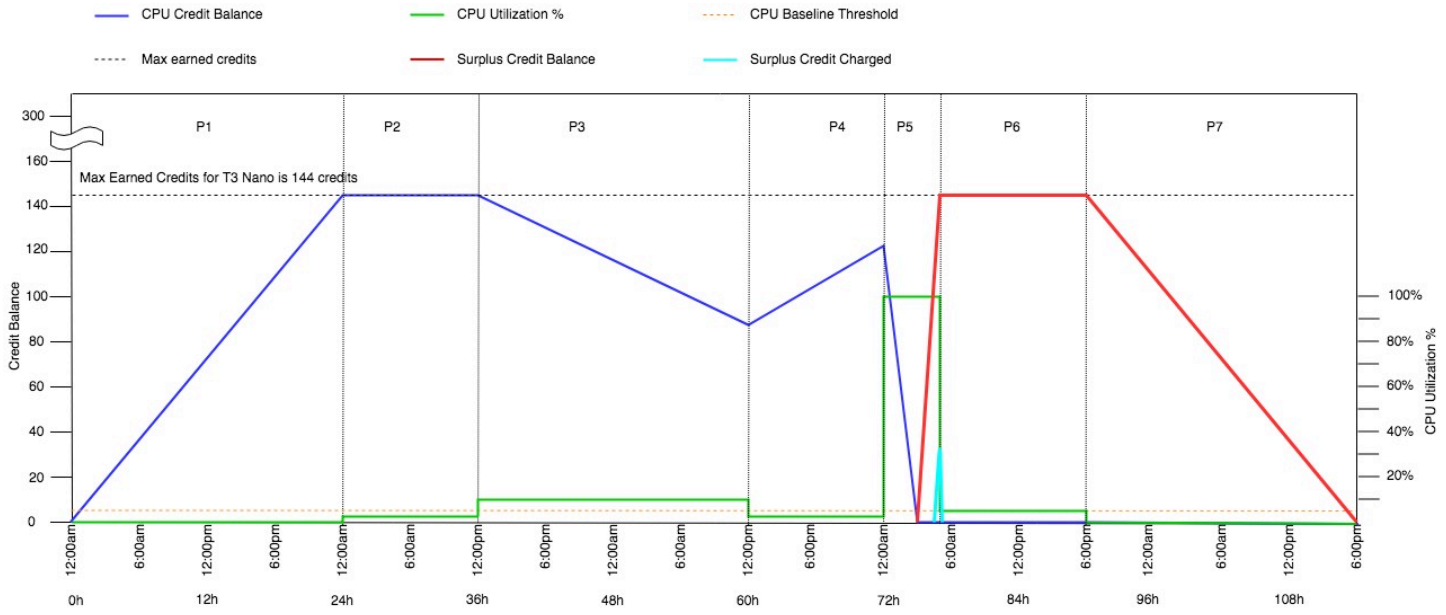
P3 – 次の 24 時間では、CPU 使用率は 7% (ベースラインを上回る) で 57.6 クレジットの消費を必要とします。インスタンスは獲得するよりも多くのクレジットを消費し、CPUCreditBalance 値は、86.4 クレジットに低減します。

P4 – 次の 12 時間では、CPU 使用率は 2.5% に減少し (ベースラインを下回る) で 36 クレジットの消費を必要とします。同時に、インスタンスは 72 クレジットを獲得します。インスタンスは消費するよりも多くのクレジットを獲得し、CPUCreditBalance 値は、122 クレジットに増加します。

P5 – 次の 5 時間で、インスタンスは 100% の CPU 使用率でバーストし、バーストを保持するために 570 クレジットを消費します。この期間内の約 1 時間で、インスタンスは CPUCreditBalance 全体の 122 クレジットを使い切り、高い CPU 使用率を維持するために余剰クレジットを使い始めます。この期間の余剰クレジット数は合計 448 (570-122=448) です。CPUSurplusCreditBalance 値が 144 CPU クレジット (t3.nano インスタンスが 24 時間に獲得できるクレジットの最大数) に達すると、その後に消費される余剰クレジットは獲得クレジットで相殺することはできません。その後に消費される余剰クレジットの量は 304 (448-144=304) クレジットで、時間の終了後に 304 クレジットに対して少額の追加料金が発生します。

P6 – 次の 13 時間では、CPU 使用率は 5% (ベースライン) です。インスタンスは消費したのと同量のクレジットを獲得するため、CPUSurplusCreditBalance の支払いを超過しません。CPUSurplusCreditBalance 値は、144 クレジットのままです。

P7 – この例の過去 24 時間では、インスタンスはアイドル状態で、CPU 使用率は 0% です。この間、インスタンスは、CPUSurplusCreditBalance の支払いに使用する 144 クレジットを獲得します。



例 2: T2 Unlimited でのクレジット使用についての説明

この例では、t2.nano として起動した unlimited インスタンスの CPU 使用率、獲得クレジットおよび余剰クレジットを使用して CPU 使用率を保持する方法を示します。

t2.nano インスタンスは、24 時間のローリング期間に渡って最大で 72 CPU クレジットを獲得し、それを 72 分の vCPU 使用と引き換えることができます。CPU クレジット残高 (CloudWatch メトリクス CPUCreditBalance で示される) が消耗すると、余剰 CPU クレジット — まだ獲得していない — を消費して必要なだけバーストします。t2.nano インスタンスは 24 時間あたり最大 72 クレジットを獲得するため、すぐに課金されることなく余剰クレジットを最大まで消費できます。CPU クレジットを 72 以上消費した場合、差分については時間の最後に課金されます。

以下のグラフにある例の目的は、CPUCreditBalance を使いきった後でも余剰クレジットを使用してインスタンスをバーストさせる方法を示すことです。グラフ中のタイムライン開始時点で、インスタンスは 24 時間に獲得可能なクレジットの最大数と同じクレジット残高を蓄積しているものとします。以下のワークフローは、グラフの番号付きの点を参照します。

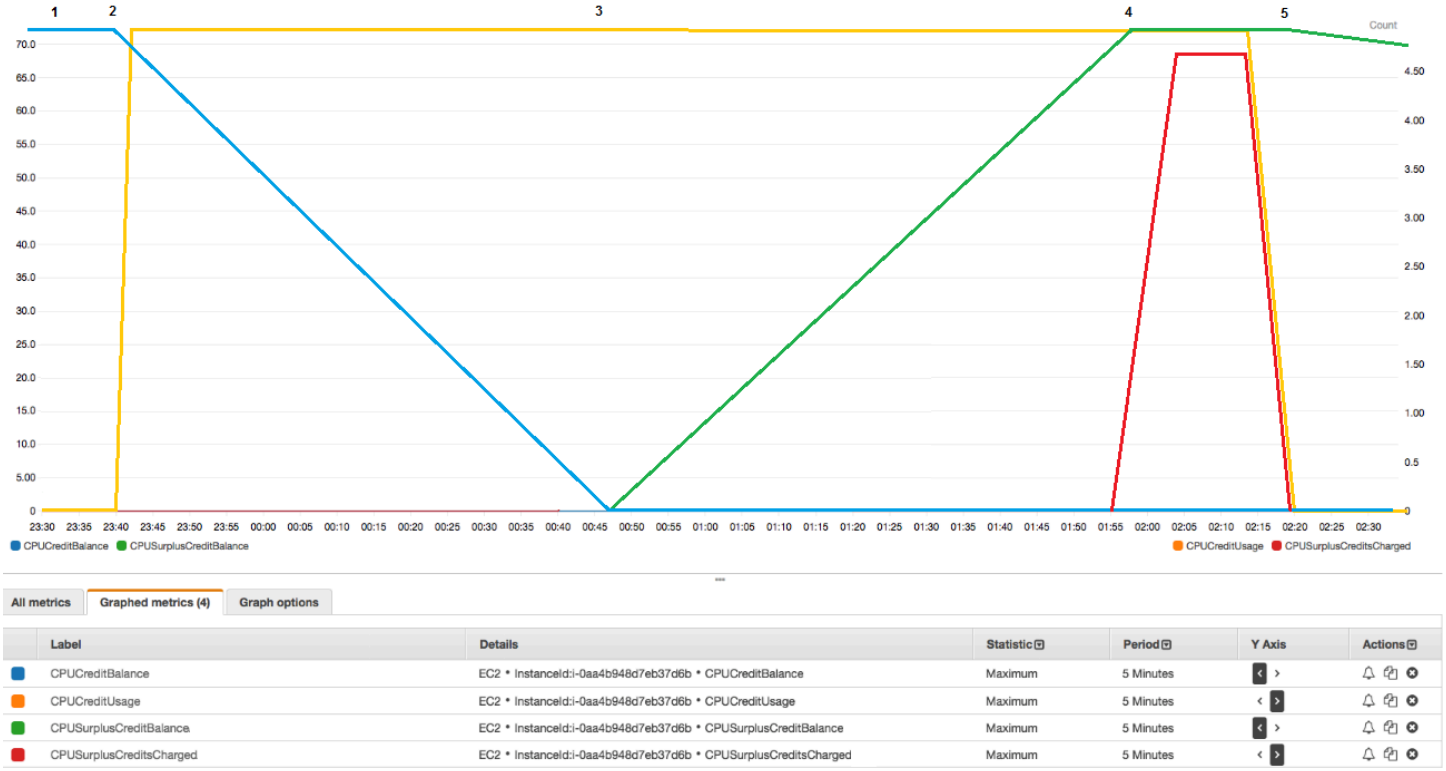
1 – 最初の 10 分間、CPUCreditUsage は 0 で、CPUCreditBalance 値は最大の 72 のままです。

2 – 23:40 に CPU 使用率が増加すると、インスタンスは CPU クレジットを消費し CPUCreditBalance 値が減少します。

3 – 00:47 頃、インスタンスは CPUCreditBalance 全体を使い切り、高い CPU 使用率を維持するために余剰クレジットを使い始めます。

4 – CPUSurplusCreditBalance 値が 72 CPU クレジットに達する 1:55 まで余剰クレジットが消費されます。これは、t2.nano インスタンスが 24 時間で獲得できる最大値と同じです。その後に消費される余剰クレジットは、24 時間以内の獲得クレジットで相殺することはできません。そのため、時間終了時に少額の追加料金が発生します。

5 – インスタンスは 2:20 頃まで余剰クレジットを消費し続けます。この時点で、CPU 使用率がベースラインを下回ると、インスタンスは 1 時間あたり 3 クレジット (または 5 分ごとに 0.25 クレジット) を獲得し始めます。これは、CPUSurplusCreditBalance の支払いに使用されます。CPUSurplusCreditBalance 値が 0 まで減った後、インスタンスは 5 分ごとに 0.25 クレジットの割合で CPUCreditBalance に獲得クレジットを蓄積し始めます。



All metrics | Graphed metrics (4) | Graph options

Label	Details	Statistic	Period	Y Axis	Actions
CPUCreditBalance	EC2 • InstanceId:i-0aa4b948d7eb37d6b • CPUCreditBalance	Maximum	5 Minutes		
CPUCreditUsage	EC2 • InstanceId:i-0aa4b948d7eb37d6b • CPUCreditUsage	Maximum	5 Minutes		
CPUSurplusCreditBalance	EC2 • InstanceId:i-0aa4b948d7eb37d6b • CPUSurplusCreditBalance	Maximum	5 Minutes		
CPUSurplusCreditsCharged	EC2 • InstanceId:i-0aa4b948d7eb37d6b • CPUSurplusCreditsCharged	Maximum	5 Minutes		

請求書の計算

余剰クレジットは vCPU 時間あたり 0.05 USD かかります。インスタンスは、1:55 から 2:20 の間におよそ 25 余剰クレジットを消費し、これは 0.42 vCPU 時間に相当します。

このインスタンスの追加料金は、0.42 vCPU 時間 x 0.05 USD/vCPU 時間 = 0.021 USD で、四捨五入すると 0.02 USD です。

これが、この T2 Unlimited インスタンスの月末請求書です。

Amazon Elastic Compute Cloud running Linux/UNIX		
$0.0058 per On Demand Linux t2.nano Instance Hour	720.000 Hrs	$4.18

Amazon Elastic Compute Cloud T2 CPU Credits		
$0.05 per vCPU-Hour of T2 CPU credits	0.420 vCPU-Hours	$0.02

1 時間ごとの料金の発生を通知する請求アラートを設定して、必要に応じてアクションを実行できます。

バーストパフォーマンスインスタンスのスタンダードモード

standard として設定したバーストパフォーマンスインスタンスは、平均 CPU 使用率がインスタンスのベースライン CPU 使用率を一貫して下回るワークロードに適しています。ベースラインより上にバーストする場合、インスタンスは CPU クレジット残高に蓄積されたクレジットを消費します。インスタンスの蓄積されたクレジットが少なくなると、CPU 使用率は徐々にベースラインパフォーマンスレベルまで下がるため、蓄積された CPU クレジット残高を使い切った場合でも、パフォーマンスが急激に低下することはありません。詳細については、バーストパフォーマンスインスタンスに関する主要な概念と定義 を参照してください。

コンテンツ

スタンダードモードの概念

standard モードはバーストパフォーマンスインスタンスの設定オプションです。これにより、実行中または停止中のインスタンスをいつでも有効または無効にできます。各 AWS リージョンのアカウントレベルで、バーストパフォーマンスインスタンスファミリーごとに、デフォルトのクレジットオプションとして standard を設定できます。アカウント内のすべての新しいバーストパフォーマンスインスタンスは、このデフォルトのクレジットオプションを使用して起動されます。

スタンダードのバーストパフォーマンスインスタンスの仕組み

standard に設定されているバーストパフォーマンスインスタンスが実行状態の場合、1 時間当たりの獲得クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。T2 スタンダードインスタンスが停止すると、蓄積されたクレジットがすべて失われ、クレジット残高はゼロにリセットされます。再起動されると、新しい起動クレジットのセットを受け取り、獲得したクレジットの蓄積を始めます。T4g、T3a、および T3 スタンダードインスタンスでは、CPU クレジット残高は、インスタンスが停止した後も 7 日間保持された後失われます。7 日以内にインスタンスを起動する場合、クレジットは失われません。

T2 スタンダードインスタンスは、獲得クレジットと起動クレジットの 2 種類の CPU クレジットを受け取ります。T2 スタンダードインスタンスが実行状態の場合、1 時間当たりの獲得クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。スタート時のインススタンスは、良いスタートアップエクスペリエンスのためのクレジットをまだ獲得していません。したがって、スタートアップエクスペリエンスを積み重ねるために、スタート時にクレジットを獲得します。インスタンスは、獲得クレジットを蓄積しながら最初にそのクレジットを消費します。

T4g、T3a、および T3 インスタンスは、Unlimited モードをサポートしているため、起動クレジットを受け取りません。T4g、T3a、および T3 インスタンスのクレジット設定を Unlimited モードにすることで、ベースラインを超えてバーストさせるために必要な量の CPU リソースを、必要な期間だけ使用できるようになります。

起動クレジット

T2 スタンダードインスタンスは、起動時またはスタート時に vCPU あたり 30 起動クレジットを獲得します。T1 スタンダードインスタンスは 15 起動クレジットを獲得します。例えば、t2.micro インスタンスは vCPU が 1 つのため 30 起動クレジット、t2.xlarge インスタンスには vCPU が 4 つあるため 120 起動クレジットを取得します。起動クレジットは、インスタンスが獲得クレジットを蓄積できるようになる前に、起動してすぐにバーストできるよう、最適な起動エクスペリエンスを提供するために設計されています。

起動クレジットは、獲得クレジットよりも先に消費されます。未使用の起動クレジットは CPU クレジット残高に蓄積されますが、CPU クレジット残高制限に対してカウントされません。例えば、t2.micro インスタンスの CPU クレジット残高制限は 144 獲得クレジットです。起動された後 24 時間アイドルのままであった場合、その CPU クレジット残高は 174 に到達し (30 起動クレジット + 144 獲得クレジット)、制限を上回ります。ただし、インスタンスが 30 起動クレジットを消費した後は、クレジット残高が 144 を超えることはありません。各インスタンスサイズの CPU クレジット残高制限の詳細については、「クレジットの表」を参照してください。

次の表は、起動または開始の際に受け取る初期 CPU クレジットの割り当てと vCPU の数を示しています。

インスタンスタイプ	起動クレジット	vCPU
t1.micro	15	1
t2.nano	30	1
t2.micro	30	1
t2.small	30	1
t2.medium	60	2
t2.large	60	2
t2.xlarge	120	4
t2.2xlarge	240	8

起動クレジット制限

T2 スタンダードインスタンスが起動クレジットを受け取る回数には制限があります。デフォルトの制限は、リージョンごとにローリング期間の 24 時間あたり各アカウントで合計で 100 回の T2 スタンダードインスタンスの起動または開始と設定されています。例えば、24 時間以内にインスタンスが 100 回停止および開始した場合、24 時間以内に 100 インスタンスが起動された場合、または他の組み合わせが 100 回の開始と同じである場合、制限に到達します。新しいアカウントでは、使用量に基づいて増える下限が設定される場合があります。

> **Tip**
>
> ワークロードに必要なパフォーマンスを常に確実に得るには、バーストパフォーマンスインスタンスの Unlimited モード に切り替えるか、またはより大きいインスタンスサイズの使用を検討してください。

起動クレジットと獲得クレジットの違い

次の表に、起動クレジットと獲得クレジットの違いを示します。

	起動クレジット	獲得クレジット
クレジットの獲得率	T2 スタンダードインスタンスは、起動時またはスタート時に vCPU あたり 30 起動クレジットを獲得します。 T2 インスタンスが `unlimited` から `standard` に切り替えられた場合、切り替えの時点では起動クレジットを取得しません。	各 T2 インスタンスは、インスタンスサイズに応じて、1 時間当たりの CPU クレジットを絶えず一定の割合で (ミリ秒レベルの細かさで) 獲得します。インスタンスサイズごとに獲得される CPU クレジット数の詳細については、「クレジットの表」を参照してください。
クレジットの獲得制限	起動クレジット受け取り制限は、リージョンごとにローリング期間の 24 時間あたり各アカウントで合計で 100 回の T2 スタンダードインスタンスの起動または開始と設定されています。新しいアカウントでは、使用量に基づいて増える下限が設定される場合があります。	T2 インスタンスは、CPU クレジット残高制限より多くのクレジットを蓄積することはできません。CPU クレジット残高がその制限に到達した場合、制限に到達した後に獲得されたクレジットはすべて破棄されます。起動クレジットは制限に対してはカウントされません。各 T2 インスタンスサイズの CPU クレジット残高制限の詳細については、「クレジットの表」を参照してください。
クレジットの使用	起動クレジットは、獲得クレジットよりも先に消費されます。	獲得クレジットは、すべての起動クレジットを消費した後にのみ消費されます。
クレジットの有効期限	T2 スタンダードインスタンスが実行中の場合、起動クレジットは期限切れになりません。T2 スタンダードインスタンスが停止し、T2 Unlimited に切り替えられた場合、すべての起動クレジットが失われます。	T2 インスタンスが実行中の場合、蓄積した獲得クレジットは期限切れになりません。T2 インスタンスが停止すると、蓄積された獲得クレジットはすべて失われます。

蓄積された起動クレジットと蓄積された獲得クレジットの数は、CloudWatch メトリクス CPUCreditBalance によって追跡されます 詳細については、「CloudWatch メトリクスの表」の CPUCreditBalance を参照してください。

スタンダードモードの例

次の例では、インスタンスが standard として設定された場合の、クレジットの使用について説明します。

例

- 例 1: T3 スタンダードでのクレジット使用についての説明
- 例 2: T2 スタンダードでのクレジット使用についての説明

例 1: T3 スタンダードでのクレジット使用についての説明

この例では、t3.nano として起動した standard インスタンスが、獲得クレジットを、獲得、蓄積、消費する方法について示します。クレジット残が、蓄積された獲得クレジットを反映するかについて示します。

実行中の t3.nano インスタンスは、24 時間ごとに 144 クレジットを獲得します。このクレジット残の制限は、144 の獲得クレジットです。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。獲得および蓄積できるクレジット数の詳細については、クレジットの表を参照してください。

T3 スタンダードインスタンスを起動し、すぐに使用することができます。または、T3 スタンダードインスタンスを起動し、何日間かアイドル状態にしてから、そこでアプリケーションを実行する場合があります。インスタンスを使用中か、アイドル状態であるかによって、クレジットが消費されるか、あるいは蓄積されるかが決まります。インスタンスが起動してから 24 時間アイドル状態のままの場合、蓄積できる獲得クレジットの最大数となり、クレジット残高が制限に達します。

この例では、起動後に 24 時間アイドル状態のままとなるインスタンスについて説明します。また、96 時間にわたる 7 つの期間で、クレジットが獲得、蓄積、消費、破棄される率と、各期間の終了時点でのクレジット残高の値を示します。

以下のワークフローは、グラフの番号付きの点を参照します。

P1 – グラフの 0 時において、インスタンスは standard として起動され、すぐにクレジットを獲得します。このインスタンスは起動時からアイドル状態になり (CPU 使用率は 0%)、クレジットは消費されません。すべての未消費のクレジットはクレジット残高に蓄積されます。最初の 24 時間は、CPUCreditUsage は 0 で、CPUCreditBalance 値は、最大の 144 に達します。

P2 – 次の 12 時間では、CPU 使用率はベースラインの 5% を下回る 2.5% です。インスタンスは消費するよりも多くのクレジットを獲得しますが、CPUCreditBalance 値は、最大 144 クレジットを超えることはできません。制限を超えて獲得されたクレジットはすべて破棄されます。

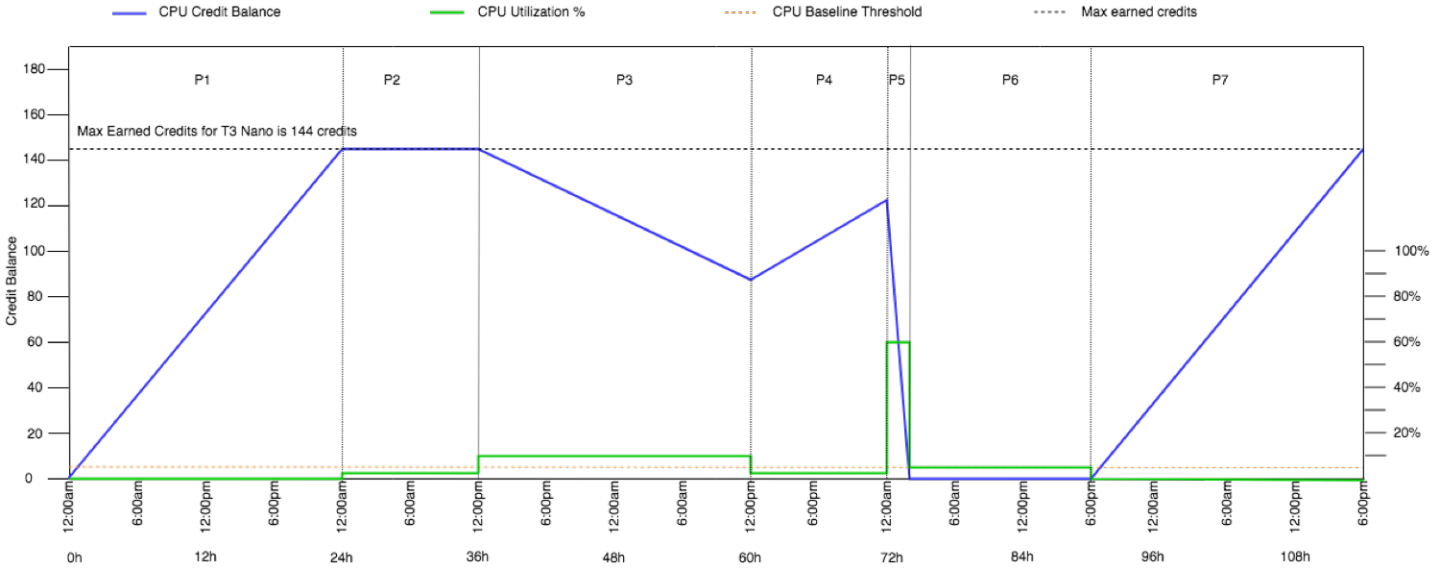
P3 – 次の 24 時間では、CPU 使用率は 7% (ベースラインを上回る) で 57.6 クレジットの消費を必要とします。インスタンスは獲得するよりも多くのクレジットを消費し、CPUCreditBalance 値は、86.4 クレジットに低減します。

P4 – 次の 12 時間では、CPU 使用率は 2.5% に減少し (ベースラインを下回る) で 36 クレジットの消費を必要とします。同時に、インスタンスは 72 クレジットを獲得します。インスタンスは消費するよりも多くのクレジットを獲得し、CPUCreditBalance 値は、122 クレジットに増加します。

P5 – 次の 2 時間で、インスタンスは 60% の CPU 使用率でバーストし、122 クレジットの全体 CPUCreditBalance 値を使い切ります。この期間の終わりに、CPUCreditBalance は 0 になり、CPU 使用率はベースライン使用率レベルの 5% まで強制的に落とされます。ベースラインで、インスタンスは消費した分のクレジットを獲得します。

P6 – 次の 14 時間では、CPU 使用率は 5% (ベースライン) です。インスタンスは消費した分のクレジットを獲得します。CPUCreditBalance 値は、0 のままです。

P7 – この例の過去 24 時間では、インスタンスはアイドル状態で、CPU 使用率は 0% です。この間、インスタンスは、CPUCreditBalance に蓄積する 144 クレジットを獲得します。



例 2: T2 スタンダードでのクレジット使用についての説明

この例では、t2.nano が起動クレジットおよび獲得クレジットを獲得、蓄積、消費する際に、standard インスタンスがどのように起動されるかについて示します。クレジット残高に、蓄

積された獲得クレジットだけでなく、蓄積された起動クレジットがどのように反映されるかについて示します。

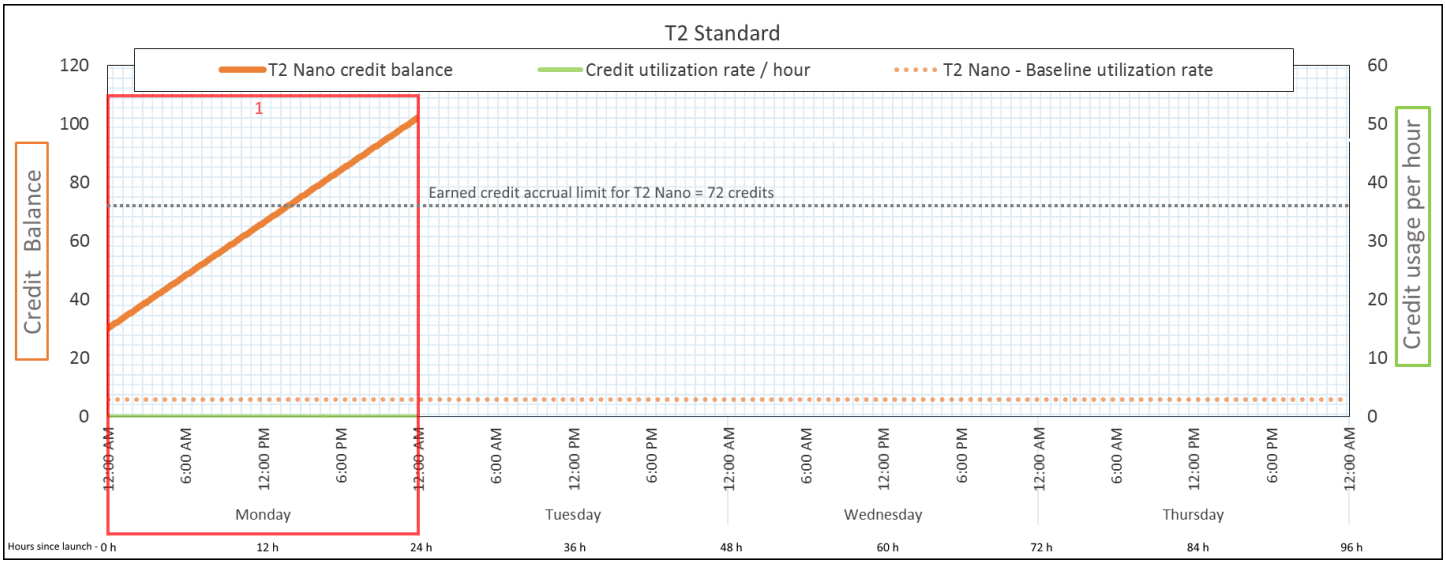
t2.nano インスタンスは、起動時に 30 起動クレジットを獲得し、24 時間ごとに 72 クレジットを獲得します。このクレジット残高の制限は 72 獲得クレジットです。起動クレジットはこの制限に対してカウントされません。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。獲得および蓄積できるクレジット数の詳細については、クレジットの表を参照してください。の制限事項の詳細については、「起動クレジット制限」を参照してください。

T2 スタンダードインスタンスを起動し、すぐに使用することができます。または、T2 スタンダードインスタンスを起動し、何日間かアイドル状態にしてから、そこでアプリケーションを実行する場合があります。インスタンスを使用中か、アイドル状態であるかによって、クレジットが消費されるか、あるいは蓄積されるかが決まります。インスタンスが起動後に 24 時間アイドル状態のままである場合、クレジット残高は制限を超えているように表示されます。これは、蓄積された獲得クレジットと蓄積された起動クレジットの両方が残高に反映されるためです。ただし、CPU を使用すると、起動クレジットが最初に使用されます。その後、この制限は、蓄積できる獲得クレジットの最大数を常に反映します。

この例では、起動後に 24 時間アイドル状態のままとなるインスタンスについて説明します。また、96 時間にわたる 7 つの期間で、クレジットが獲得、蓄積、消費、破棄される率と、各期間の終了時点でのクレジット残高の値を示します。

期間 1: 1 ~ 24 時間

グラフの 0 時において、T2 インスタンスは standard として起動され、すぐに 30 クレジットを獲得します。実行状態の間はクレジットを獲得します。このインスタンスは起動時からアイドル状態になり (CPU 使用率は 0%)、クレジットは消費されません。すべての未消費のクレジットはクレジット残高に蓄積されます。起動後約 14 時間で、クレジット残高は 72 (30 起動クレジット + 42 獲得クレジット) となり、これはインスタンスが 24 時間に獲得できる数と同等になります。起動後 24 時間で、クレジット残高は 72 クレジットを超えます。これは、未消費の起動クレジットがクレジット残高に蓄積されるためです (クレジット残高は— 102 クレジット: 30 起動クレジット + 72 獲得クレジット)。



クレジットの消費率	24 時間あたり 0 クレジット (0% の CPU 使用率)
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 0 クレジット
クレジット残高	102 クレジット (30 起動クレジット + 72 獲得クレジット)

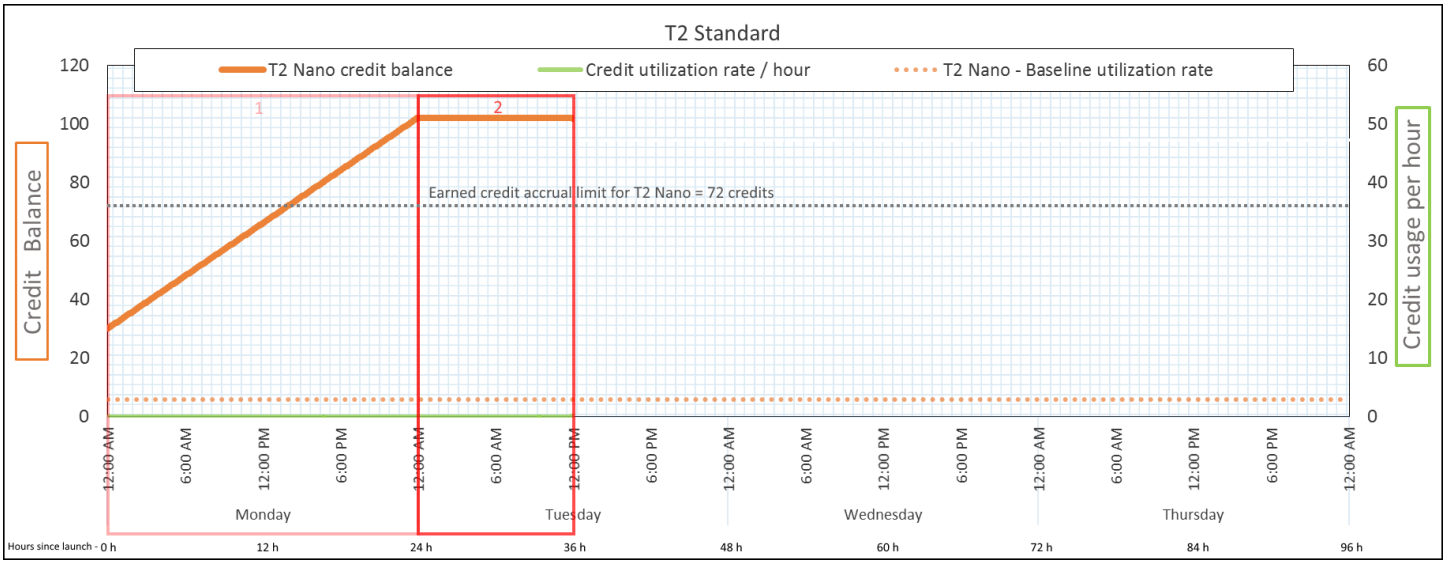
結論

起動後に CPU の使用がない場合、インスタンスは 24 時間に獲得できるよりも多くのクレジットを蓄積します (30 起動クレジット + 72 獲得クレジット = 102 クレジット)。

実際のシナリオでは、EC2 インスタンスは起動中および実行中にも少数のクレジットを消費します。それにより、残高がこの例の理論的な最大値に達することを防ぎます。

期間 2: 25 ~ 36– 時間

次の 12 時間中に、インスタンスは引き続きアイドル状態のままとなり、クレジットを獲得しますが、クレジット残高は増えません。102 クレジット (30 起動クレジット + 72 獲得クレジット) で頭打ちとなります。クレジット残高は制限である 72 の蓄積された獲得クレジットに達したため、新しく獲得されたクレジットは破棄されます。



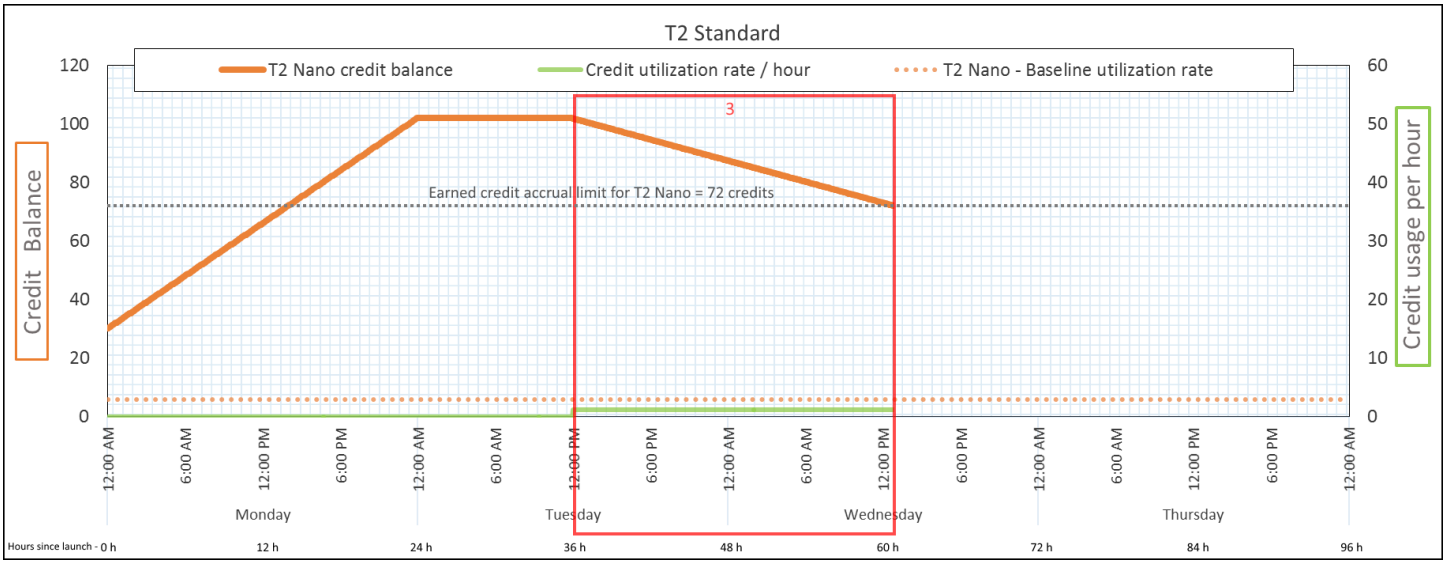
クレジットの消費率	24 時間あたり 0 クレジット (0% の CPU 使用率)
クレジットの獲得率	24 時間あたり 72 クレジット (1 時間で 3 クレジット)
クレジットの破棄率	24 時間あたり 72 クレジット (100% のクレジット獲得率)
クレジット残高	102 クレジット (30 起動クレジット + 72 獲得クレジット)— 残高は変更されません

結論

インスタンスはクレジットを継続して獲得しますが、クレジット残高が制限に達した場合、獲得クレジットはそれ以上蓄積されません。制限に到達すると、新しく獲得されたクレジットはすべて破棄されます。起動クレジットは、クレジット残高制限に対してカウントされません。残高に蓄積された起動クレジットが含まれている場合、残高は制限を超えているように表示されます。

期間 3: 37 ~ 61 時間

次の 25 時間で、インスタンスは 2% の CPU を使用します。これには 30 クレジットが必要です。同じ期間に 75 クレジットを取得しますが、クレジット残高は減ります。残高が減るのは、蓄積された起動クレジットが最初に消費されますが、クレジット残高が既に 72 獲得クレジットという制限に達しているため、新しく獲得されたクレジットは破棄されるためです。



クレジットの消費率	24 時間あたり 28.8 クレジット (1 時間ごとに 1.2 クレジット、2% の CPU 使用率、40% のクレジット獲得率)— 25 時間以上で 30 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 72 クレジット (100% のクレジット獲得率)
クレジット残高	72 クレジット (30 起動クレジットが消費され、72 獲得クレジットが未使用のまま)

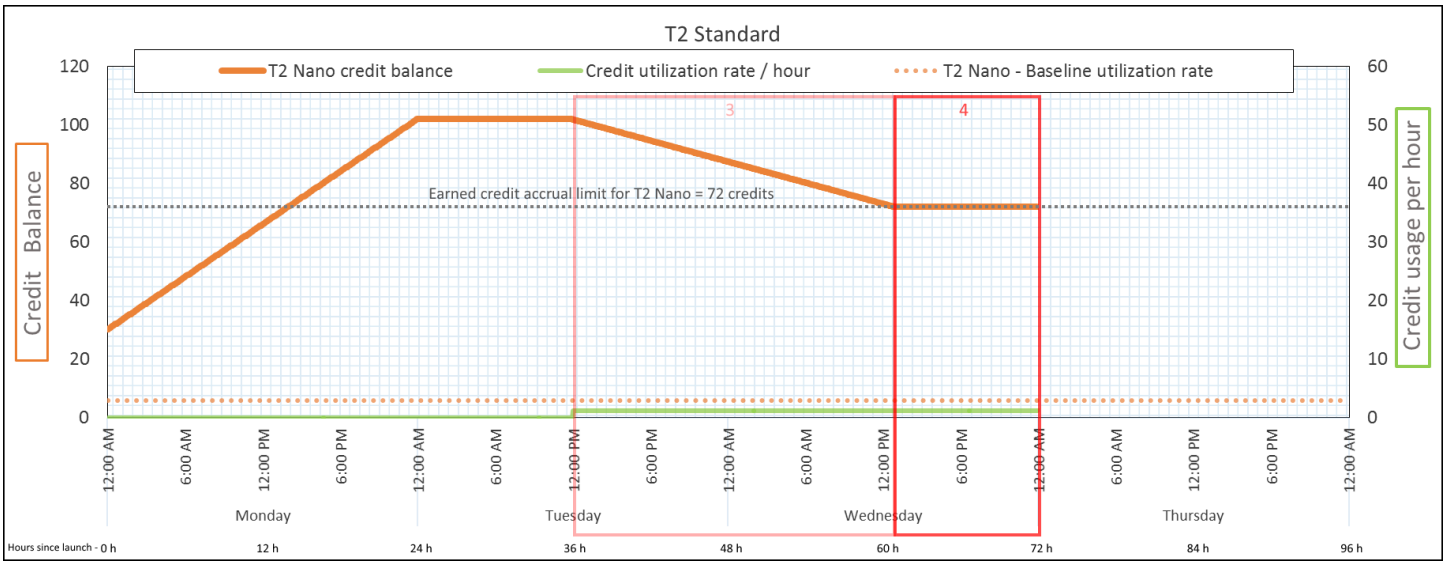
結論

インスタンスは、獲得クレジットを消費する前に、起動クレジットを最初に消費します。起動クレジットは、クレジット制限に対してカウントされません。起動クレジットが消費された後で、24 時間に獲得できる数よりも残高が高くなることはありません。さらに、インスタンスの実行中は、それ以上クレジットを獲得することはできません。

期間 4: 62 ~ 72 時間

次の 11 時間で、インスタンスは 2% の CPU を使用します。これには 13.2 クレジットが必要です。これは前の期間の CPU 使用率と同じですが、残高は減りません。72 クレジットのままです。

残高が減らないのは、クレジットの獲得率がクレジットの消費率よりも高いためです。また、インスタンスは 13.2 クレジットを消費する時間に、33 クレジットを獲得します。ただし、残高の制限は 72 クレジットであるため、制限を超えて獲得されたクレジットは破棄されます。残高は 72 で頭打ちとなります。これは期間 2 の 102 クレジットという頭打ちとは異なりますが、蓄積された起動クレジットがないためです。



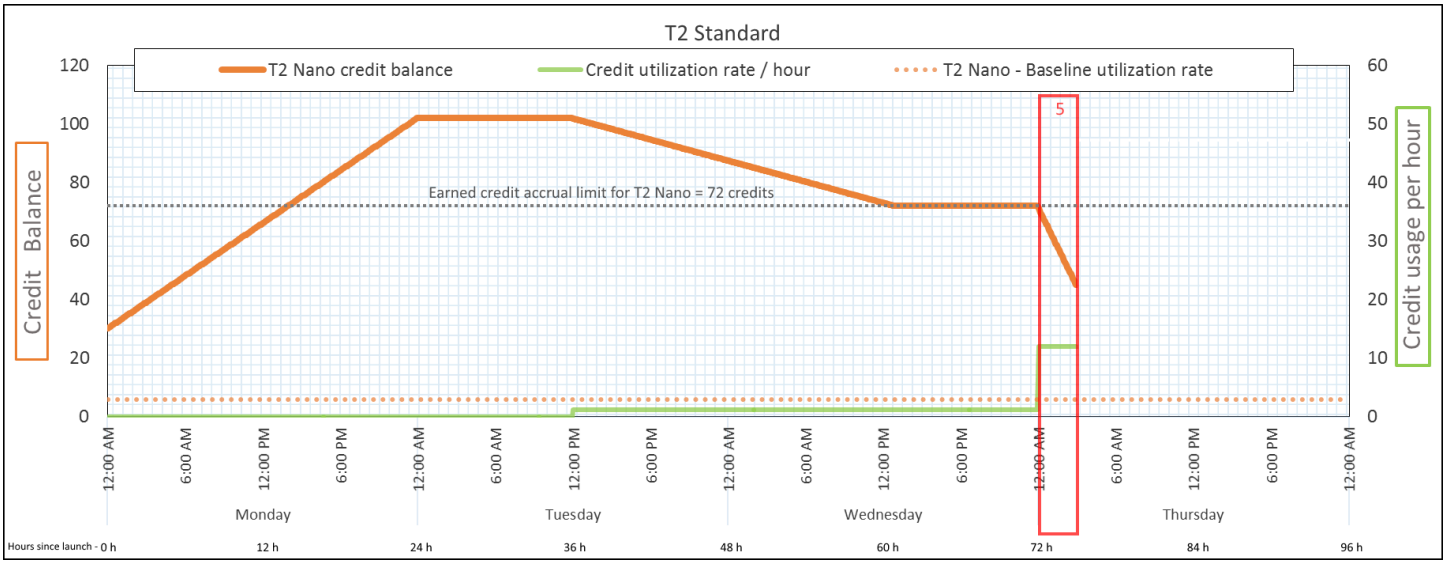
クレジットの消費率	24 時間あたり 28.8 クレジット (1 時間ごとに 1.2 クレジット、2% の CPU 使用率、40% のクレジット獲得率)— 11 時間以上で 13.2 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 43.2 クレジット (60% のクレジット獲得率)
クレジット残高	72 クレジット (0 起動クレジット、72 獲得クレジット)— 残高は上限

結論

起動クレジットの消費後、クレジット残高の制限はインスタンスが 24 時間に獲得できるクレジット数によって決まります。インスタンスが、消費するよりも多くのクレジットを獲得した場合、制限を超えて新しく獲得されたクレジットは破棄されます。

7500.0

次の 3 時間で、インスタンスは 20% の CPU 使用率でバーストします。これには 36 クレジットが必要です。インスタンスは同じ 3 時間で 9 クレジットを獲得します。これにより、実際の残高は 27 クレジット減ります。3 時間の最後に、クレジット残高は 45 の蓄積された獲得クレジットとなります。



クレジットの消費率	24 時間あたり 288 クレジット (1 時間ごとに 12 クレジット、20% の CPU 使用率、400% のクレジット獲得率)— 3 時間以上で 36 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット (3 時間で 9 クレジット)
クレジットの破棄率	24 時間あたり 0 クレジット
クレジット残高	45 クレジット (前の残高 (72) - 消費したクレジット (36) + 獲得したクレジット (9))— 残高は 24 時間あたり 216 クレジットの率で減少 (消費率 288/24 + 獲得率 72/24 = 残高減少率 216/24)

結論

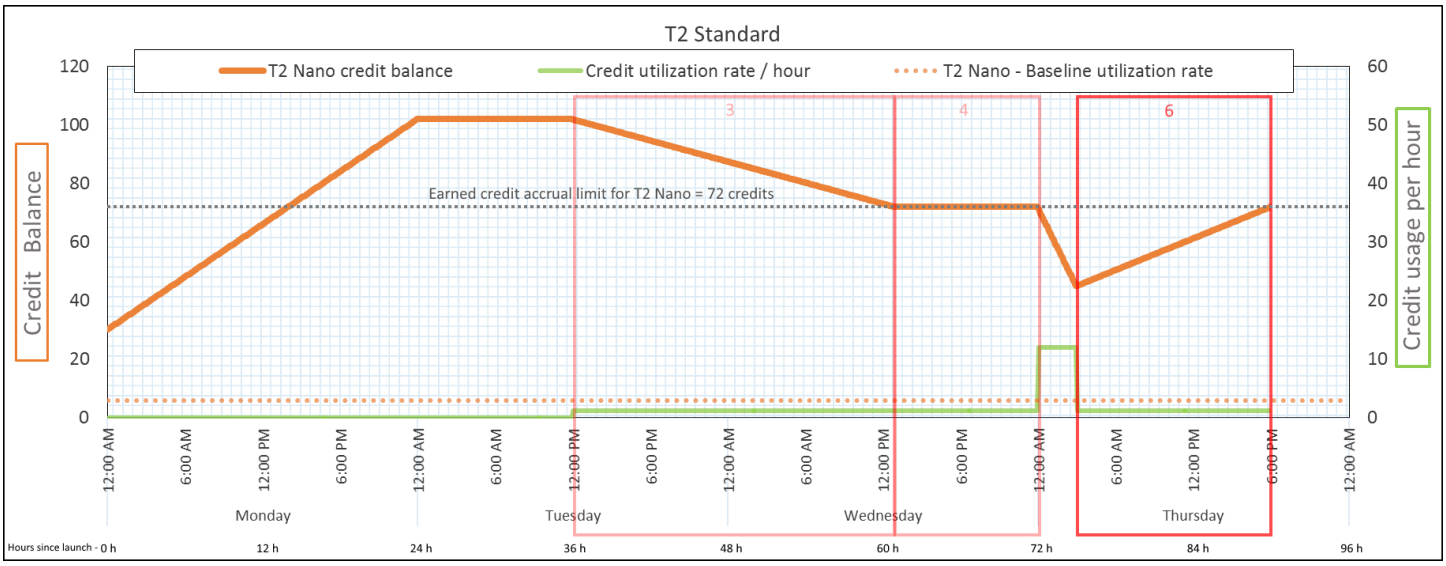
インスタンスが、獲得するよりも多くのクレジットを消費する場合、クレジット残高は減ります。

期間 6: 76 ~ 90 時間

次の 15 時間で、インスタンスは 2% の CPU を使用します。これには 18 クレジットが必要です。これは、期間 3 および 4 と同じ CPU 使用率です。ただし、期間 3 で残高が減り、期間 4 で頭打ちになりましたが、この期間の残高は増えます。

期間 3 で、蓄積された起動クレジットが消費されました。また、クレジットの制限を超えて獲得されたクレジットは破棄され、クレジット残高が減りました。期間 4 で、インスタンスが消費したクレジットは獲得したクレジットよりも少なくなりました。限度額を超えて獲得したクレジットはすべて破棄されたため、残高は最大 72 クレジットとなりました。

この期間に蓄積された起動クレジットはなく、残高の蓄積された獲得クレジットの数は制限を下回っています。獲得クレジットは破棄されません。さらに、インスタンスは消費するよりも多くのクレジットを獲得し、クレジット残高が増えます。



クレジットの消費率	24 時間あたり 28.8 クレジット (1 時間ごとに 1.2 クレジット、2% の CPU 使用率、40% のクレジット獲得率)— 15 時間以上で 18 クレジット
クレジットの獲得率	24 時間あたり 72 クレジット (15 時間で 45 クレジット)
クレジットの破棄率	24 時間あたり 0 クレジット

クレジット残高	72 クレジット (残高は 24 時間あたり 43.2 クレジットの率で増えます— 変更率 = 消費率 28.8/24 + 獲得率 72/24)

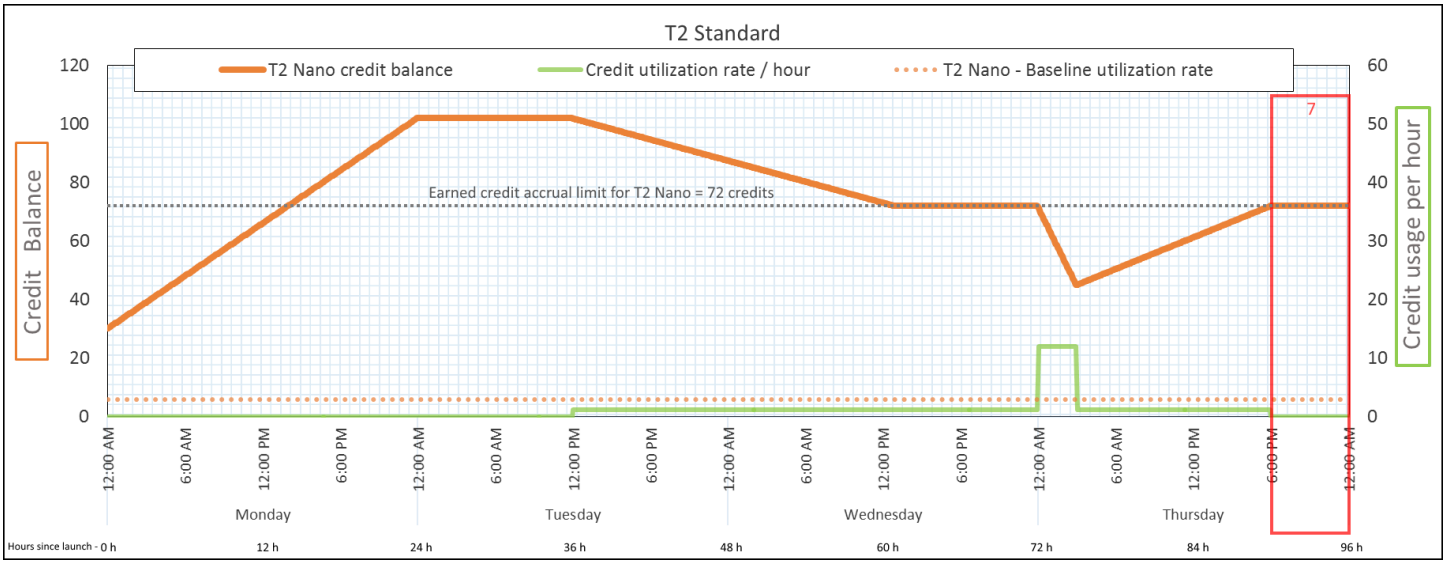
結論

インスタンスが、獲得するよりも少ないクレジットを消費する場合、クレジット残高は増えます。

期間 7: 91 ~ 96 時間

次の 6 時間は、インスタンスはアイドル状態になり (—CPU 使用率は 0%—)、クレジットは消費されません。これは、期間 2 の —CPU 使用率と同じですが、残高は 102 クレジットで頭打ちになりません。インスタンスのクレジット残高の制限である 72 クレジットで頭打ちになります。

期間 2 で、クレジット残高には蓄積された 30 起動クレジットが含まれます。起動クレジットは期間 3 で消費されました。実行中のインスタンスはそれ以上起動クレジットを取得できません。クレジット残高の制限に達すると、制限を超えて獲得されたクレジットは破棄されます。



クレジットの消費率	24 時間あたり 0 クレジット (0% の CPU 使用率)
クレジットの獲得率	24 時間あたり 72 クレジット
クレジットの破棄率	24 時間あたり 72 クレジット (100% のクレジット獲得率)

クレジット残高	72 クレジット (0 起動クレジット、72 獲得クレジット)

結論

インスタンスはクレジットを継続して獲得しますが、クレジット残高の制限に達した場合、獲得クレジットはそれ以上蓄積されません。制限に到達すると、新しく獲得されたクレジットはすべて破棄されます。クレジット残高の制限は、インスタンスが 24 時間に獲得できるクレジット数によって決まります。クレジット残高の制限の詳細については、クレジットの表を参照してください。

バーストパフォーマンスインスタンスの使用

バーストパフォーマンスインスタンス (T インスタンス) の起動、モニタリング、および変更の手順は似ています。主な違いは、起動時のデフォルトのクレジット指定です。

各 T インスタンスファミリーには、以下のデフォルトクレジット仕様が付属しています。

- T4g、T3a、および T3 インスタンスを `unlimited` として起動
- 専有ホストで `standard` として T3 インスタンスを起動のみ行えます。
- T2 インスタンスを `standard` として起動

アカウントのクレジット指定のデフォルト設定を変更できます。

コンテンツ

バーストパフォーマンスインスタンスを無制限またはスタンダードとして起動する

Amazon EC2 コンソール、AWS SDK、コマンドラインツール、または Auto Scaling グループを使用して、T インスタンスを `unlimited` または `standard` として起動できます。

次の手順では、EC2 コンソールまたは AWS CLI を使用する方法について説明します。Auto Scaling グループの使用の詳細については、「Auto Scaling グループを使用してバーストパフォーマンスインスタンスを無制限で起動する」を参照してください。

要件

- T インスタンスの起動には、Amazon EBS ボリュームをルートデバイスとして使用する必要があります。詳細については、「Amazon EC2 インスタンスのルートボリューム」を参照してください。
- T インスタンスの AMI とドライバーの要件の詳細については、「リリースノート」を参照してください。

Console

T インスタンスを Unlimited またはスタンダードとして起動するには

1. インスタンスを起動するための手順に従います。
2. [Instance type] (インスタンスタイプ) で、T インスタンスタイプを選択します。
3. [Advanced details] (高度な詳細) を展開し、[Credit specification] (クレジットの仕様) でクレジットの仕様を選択します。選択しない場合はデフォルトが使用されます。デフォルトは、T2 については `standard`、T4g、T3a、および T3 については `unlimited` です。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI

T インスタンスを Unlimited またはスタンダードとして起動するには

run-instances コマンドを使用して、インスタンスを起動します。`--credit-specification CpuCredits=` パラメータを使用してクレジット指定を指定します。有効なクレジット指定は `unlimited` と `standard` です。

- T4g、T3a、および T3 に `--credit-specification` パラメータを含めない場合、インスタンスはデフォルトで `unlimited` として起動します。
- T2 で、`--credit-specification` パラメータを含めない場合、インスタンスはデフォルトで `standard` として起動します。

```
aws ec2 run-instances \
    --image-id ami-abc12345 \
    --count 1 \
    --instance-type t3.micro \
    --key-name MyKeyPair \
    --credit-specification "CpuCredits=unlimited"
```

Auto Scaling グループを使用してバーストパフォーマンスインスタンスを無制限で起動する

T インスタンスが起動または開始する際、優れたブートストラップエクスペリエンスには CPU クレジットが必要です。Auto Scaling グループを使用してインスタンスを起動する場合は、インスタンスを unlimited として設定することをお勧めします。そうする場合、インスタンスは Auto Scaling グループによって自動的に起動または再開されたときに余剰クレジットを使用します。余剰クレジットを使用することで、パフォーマンスの制限を防ぐことができます。

起動テンプレートの作成

インスタンスを Auto Scaling グループで unlimited として起動するには、起動に起動テンプレートを使用する必要があります。起動設定では、インスタンスを unlimited として起動することはサポートされていません。

> **Note**
>
> unlimitedモードは、Dedicated Host で起動される T3 インスタンスではサポートされません。

Console

インスタンスを Unlimited として起動する起動テンプレートを作成するには

1. 「Amazon EC2 Auto Scaling ユーザーガイド」の「詳細設定を使用して起動テンプレートを作成する」を参照してください。
2. [Launch template contents] ((テンプレートコンテンツの起動) の [Instance type] (インスタンスタイプ) で、インスタンスサイズを選択します。
3. インスタンスを Auto Scaling グループで unlimited として起動するには、[Advanced details] (高度な詳細) の [Credit specification] (クレジット指定) で [Unlimited] (無制限) を選択します。

4. 起動テンプレートパラメータの定義が終了したら、[Create launch template] (起動テンプレートの作成) を選択します。

AWS CLI

インスタンスを Unlimited として起動する起動テンプレートを作成するには

create-launch-template コマンドを使用して、`unlimited` を CPU 使用率に関するクレジット指定として指定します。

- T4g、T3a、および T3 に `CreditSpecification={CpuCredits=unlimited}` 値を含めない場合、インスタンスはデフォルトで `unlimited` として起動します。
- T2 で、`CreditSpecification={CpuCredits=unlimited}` 値を含めない場合、インスタンスはデフォルトで `standard` として起動します。

```
aws ec2 create-launch-template \
    --launch-template-name MyLaunchTemplate \
    --version-description FirstVersion \
    --launch-template-data
 ImageId=ami-8c1be5f6,InstanceType=t3.medium,CreditSpecification={CpuCredits=unlimited}
```

起動テンプレートによる Auto Scaling グループの関連付け

起動テンプレートを Auto Scaling グループに関連付けるには、起動テンプレートを使用して Auto Scaling グループを作成するか、または既存の Auto Scaling グループに起動テンプレートを追加します。

Console

起動テンプレートを使用して Auto Scaling グループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面の上部のナビゲーションバーで、起動テンプレートを作成したときに使用したのと同じリージョンを選択します。
3. ナビゲーションペインで [Auto Scaling グループ]、[Auto Scaling グループの作成] の順に選択します。

4. [Launch Template (起動テンプレート)] で、起動テンプレートを選択し、[次のステップ] を選択します。
5. Auto Scaling グループ用のフィールドに入力します。[Review page (確認ページ)] で設定の確認を終えたら、[Create Auto Scaling group (Auto Scaling グループの作成)] を選択します。詳細については、『Amazon EC2 Auto Scaling ユーザーガイド』の「起動テンプレートを使用した Auto Scaling グループの作成 」を参照してください。

AWS CLI

起動テンプレートを使用して Auto Scaling グループを作成するには

create-auto-scaling-group AWS CLI コマンドを使用して、`--launch-template` パラメータを指定します。

Console

既存の Auto Scaling グループに起動テンプレートを追加するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面の上部のナビゲーションバーで、起動テンプレートを作成したときに使用したのと同じリージョンを選択します。
3. ナビゲーションペインで、[Auto Scaling Groups] をクリックします。
4. Auto Scaling グループの一覧から Auto Scaling グループを選択し、[アクション]、[編集] の順に選択します。
5. [Details (詳細)] タブの [Launch Template (起動テンプレート)] で起動テンプレートを選択して、[Save (保存)] を選択します。

AWS CLI

既存の Auto Scaling グループに起動テンプレートを追加するには

update-auto-scaling-group AWS CLI コマンドを使用して、`--launch-template` パラメータを指定します。

バーストパフォーマンスインスタンスのクレジット指定の表示

実行中または停止中の T インスタンスのクレジット指定 (unlimited または standard) を表示できます。

Console

T インスタンスのクレジット指定を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインの [インスタンス] を選択します。
3. インスタンスを選択します。
4. [Details (詳細)] を選択し、[Credit specification (クレジット指定)] フィールドを表示します。この値は unlimited または standard のどちらかです。

AWS CLI

T インスタンスのクレジット指定を記述するには

describe-instance-credit-specifications コマンドを使用します。1 つ以上のインスタンス ID を指定しない場合、以前に unlimited クレジット仕様で設定されていたインスタンスだけでなく、unlimited クレジット指定のすべてのインスタンスが返されます。例えば、T3 インスタンスを M4 インスタンスにサイズ変更し、unlimited に設定している場合、Amazon EC2 は M4 インスタンスを返します。

```
aws ec2 describe-instance-credit-specifications --instance-id i-1234567890abcdef0
```

出力例

```
{
  "InstanceCreditSpecifications": [
    {
        "InstanceId": "i-1234567890abcdef0",
        "CpuCredits": "unlimited"
    }
  ]
}
```

バーストパフォーマンスインスタンスのクレジット指定の変更

実行中または停止中の T インスタンスのクレジット指定は、unlimited と standard の間でいつでも切り替えることができます。

unlimited モードでは、インスタンスが余剰クレジットを使用することがあり、追加料金が発生する可能性があることに注意してください。詳細については、「余剰クレジットにより料金が発生することがある」を参照してください。

Console

T インスタンスのクレジット指定を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインの [インスタンス] を選択します。
3. インスタンスを選択します。複数のインスタンスのクレジット指定を一度に変更するには、適用可能なインスタンスをすべて選択します。
4. [Actions (アクション)]、[Instance settings (インスタンス設定)]、[Change credit specification (クレジット指定の変更)] の順に選択します。このオプションは、T インスタンスを選択した場合にのみ有効になります。
5. クレジット指定を unlimited に変更するには、インスタンス ID の横にあるチェックボックスをオンにします。クレジット指定を standard に変更するには、インスタンス ID の横にあるチェックボックスをオフにします。

AWS CLI

T インスタンスのクレジット指定を変更するには

modify-instance-credit-specification コマンドを使用します。--instance-credit-specification パラメータを使用して、インスタンスとクレジット指定を指定します。有効なクレジット指定は unlimited と standard です。

```
aws ec2 modify-instance-credit-specification \
    --region us-east-1 \
    --instance-credit-specification
 "InstanceId=i-1234567890abcdef0,CpuCredits=unlimited"
```

出力例

```
{
  "SuccessfulInstanceCreditSpecifications": [
    {
      "InstanceId": "i- 1234567890abcdef0"
    }
  ],
  "UnsuccessfulInstanceCreditSpecifications": []
}
```

アカウントのクレジット指定のデフォルト設定

各 T インスタンスファミリーには、デフォルトクレジット仕様が付属しています。各 AWS リージョンのアカウントレベルで、T インスタンスファミリーごとにデフォルトのクレジット仕様を変更できます。

EC2 コンソールのインスタンス起動ウィザードを使用してインスタンスを起動している場合、アカウントレベルのデフォルトのクレジット指定は、お客様により設定されたクレジット指定の値により上書きされます。AWS CLI を使用してインスタンスを起動する場合には、アカウント内のすべての新しい T インスタンスは、デフォルトのクレジット指定を使用して起動されます。既存の実行中または停止中のインスタンスのクレジット指定には影響しません。

考慮事項

インスタンスファミリーのデフォルトのクレジット指定は、継続した 5 分間に 1 回のみ変更でき、継続した 24 時間中に最大 4 回変更できます。

Console

リージョンごとにアカウントレベルでデフォルトのクレジット指定を設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. AWS リージョン を変更するには、ページの右上隅にあるリージョンセレクターを使用します。
3. 左側ナビゲーションペインで、[EC2 ダッシュボード] をクリックします。
4. [アカウントの属性] から、[デフォルトのクレジット指定] を選択します。
5. [管理] をクリックします。
6. インスタンスファミリーごとに、[無制限] または [標準] を選択した上で、[更新] をクリックします。

AWS CLI

アカウントレベルでデフォルトのクレジット指定を設定するには (AWS CLI)

modify-default-credit-specification コマンドを使用します。AWS パラメータを使用して、--cpu-credits リージョン、インスタンスファミリー、およびデフォルトのクレジット仕様を設定します。有効なデフォルトのクレジット指定は、unlimited および standard です。

```
aws ec2 modify-default-credit-specification \
    --region us-east-1 \
    --instance-family t2 \
    --cpu-credits unlimited
```

デフォルトのクレジット指定の表示

各 AWS リージョンのアカウントレベルで、T インスタンスファミリーのデフォルトのクレジット仕様を表示できます。

Console

アカウントレベルでデフォルトのクレジット指定を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. AWS リージョン を変更するには、ページの右上隅にあるリージョンセレクターを使用します。
3. 左側ナビゲーションペインで、[EC2 ダッシュボード] をクリックします。
4. [アカウントの属性] から、[デフォルトのクレジット指定] を選択します。

AWS CLI

アカウントレベルでデフォルトのクレジット指定を表示するには

get-default-credit-specification コマンドを使用します。AWS リージョンとインスタンスファミリーを指定します。

```
aws ec2 get-default-credit-specification --region us-east-1 --instance-family t2
```

バーストパフォーマンスインスタンスの CPU クレジットをモニタリングする

EC2 はメトリクスを Amazon CloudWatch に送信します。CPU クレジットのメトリクスは、CloudWatch コンソールの Amazon EC2 インスタンスごとのメトリクスで確認するか、AWS CLI を使用して各インスタンスのメトリクスを一覧表示することで確認できます。詳細については、「コンソールを使用したメトリクスの一覧表示AWS CLI を使用したメトリクスの一覧表示」および「」を参照してください。

コンテンツ

バーストパフォーマンスインスタンスの追加 CloudWatch メトリクス

バーストパフォーマンスインスタンスには、以下の追加の CloudWatch メトリクスがあり、5 分ごとに更新されます。

- `CPUCreditUsage` – 測定期間に消費された CPU クレジットの数。
- `CPUCreditBalance` – インスタンスが蓄積する CPU クレジット数。このバランスは CPU がバーストする際に枯渇し、CPU クレジットは獲得するよりも速い速度で使用されます。
- `CPUSurplusCreditBalance` – `CPUCreditBalance` 値がゼロになったときに CPU 使用率を保持するために使用される余剰 CPU クレジットの数。
- `CPUSurplusCreditsCharged` – 24 時間で獲得できる CPU クレジットの最大数を越えた、追加料金が発生する分の余剰 CPU クレジットの数。

最後の 2 つのメトリクスは `unlimited` として設定されたインスタンスにのみ適用されます。

バーストパフォーマンスインスタンスの CloudWatch メトリクスの説明を次の表に示します。詳細については、インスタンスの利用可能な CloudWatch メトリクスのリスト表示 を参照してください。

メトリクス	説明
`CPUCreditUsage`	CPU 使用率に関してインスタンスで消費される CPU クレジットの数。1 つの CPU クレジットは、1 個の vCPU が 100% の使用率で 1 分間実行されること、または、vCPU、使用率、時間の同等の組み合わせ (例えば、1 個の vCPU が 50% の使用率で

メトリクス	説明
	2 分間実行されるか、2 個の vCPU が 25% の使用率で 2 分間実行される) に相当します。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。5 分を超える期間を指定する場合は、Average 統計の代わりに Sum 統計を使用します。 単位: クレジット (vCPU 分)
CPUCreditBalance	インスタンスが起動または開始後に蓄積した獲得 CPU クレジットの数。T2 スタンダードの場合、CPUCreditBalance には蓄積された起動クレジットの数も含まれます。 クレジットは、獲得後にクレジット残高に蓄積され、消費されるとクレジット残高から削除されます。クレジット残高には、インスタンスサイズによって決まる上限があります。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。T2 スタンダードの場合、起動クレジットは制限に対してカウントされません。 CPUCreditBalance のクレジットは、インスタンスがそのベースライン CPU 使用率を超えてバーストするために消費できます。 インスタンスが実行中の場合、CPUCreditBalance のクレジットは期限切れになりません。T4g、T3a、もしくは T3 インスタンスの停止後、CPUCreditBalance 値は 7 日間保持されます。その後、蓄積されたすべてのクレジットが失われます。T2 インスタンスが停止すると、CPUCreditBalance 値は保持されず、蓄積されたすべてのクレジットが失われます。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。 単位: クレジット (vCPU 分)

メトリクス	説明
CPUSurplusCreditBalance	unlimited 値がゼロの場合に CPUCreditBalance インスタンスによって消費された余剰クレジットの数。 CPUSurplusCreditBalance 値は獲得した CPU クレジットによって支払われます。余剰クレジットの数が、24 時間にインスタンスが獲得できるクレジットの最大数を超えている場合、最大数を超えて消費された余剰クレジットに対しては料金が発生します。 単位: クレジット (vCPU 分)
CPUSurplusCreditsCharged	獲得 CPU クレジットにより支払われないために追加料金が発生した、消費された余剰クレジットの数。 消費された余剰クレジットは、以下のいずれかの状況に当てはまると料金が発生します。 • 消費された余剰クレジットが、インスタンスが 24 時間に獲得できる最大クレジット数を超えている。最大数を越えて消費された余剰クレジットは、時間の最後に課金されます。 • インスタンスが停止または終了した。 • インスタンスは unlimited から standard に切り替わります。 単位: クレジット (vCPU 分)

CPU クレジット使用状況の計算

インスタンスの CPU クレジット使用状況は、前述の表で説明したインスタンス CloudWatch メトリクスを使用して計算されます。

Amazon EC2 は、メトリクスを 5 分ごとに CloudWatch に送信します。前のメトリクス値の参照はいつでも、5 分前に送信された、直前のメトリクス値を意味します。

スタンダードインスタンスの CPU クレジット使用状況の計算

- CPU クレジット残高は、CPU 利用率がベースラインを下回り、前の 5 分間に消費したクレジットが獲得したクレジットより少なかった場合に増加します。
- CPU クレジット残高は、CPU 利用率がベースラインを上回り、前の 5 分間に消費したクレジットが獲得したクレジットよりも多かった場合に減少します。

数学的に、これは次の式で表されます。

Example

```
CPUCreditBalance = prior CPUCreditBalance + [Credits earned per hour * (5/60) -
 CPUCreditUsage]
```

インスタンスのサイズは、インスタンスが 1 時間あたりに獲得できるクレジットの数と、クレジット残高に蓄積できる獲得クレジットの数を決定します。1 時間あたりに獲得するクレジット数と、各インスタンスサイズのクレジット残高制限については、「クレジットの表」を参照してください。

例

この例では、t3.nano インスタンスを使用します。インスタンスの CPUCreditBalance 値を計算するには、前述の式を次のように使用します。

- CPUCreditBalance – 計算する現在のクレジット残高。
- prior CPUCreditBalance – 5 分前のクレジット残高。この例では、インスタンスは 2 クレジットを蓄積しています。
- Credits earned per hour – t3.nano インスタンスは 1 時間あたり 6 クレジット獲得します。
- 5/60 – CloudWatch メトリクスのパブリッシュ間の 5 分間隔を表します。1 時間あたりに獲得するクレジットに 60 分の 5 (5 分) を掛けて、インスタンスが過去 5 分間に獲得したクレジット数を求めます。t3.nano インスタンスは、5 分ごとに 0.5 クレジットを獲得します。
- CPUCreditUsage – インスタンスが過去 5 分間に消費したクレジット数。この例では、インスタンスは過去 5 分間に 1 クレジットを消費しました。

これらの値を使用して、CPUCreditBalance の値を計算できます。

Example

```
CPUCreditBalance = 2 + [0.5 - 1] = 1.5
```

無制限インスタンスの CPU クレジット使用状況の計算

バーストパフォーマンスインスタンスがベースラインを超えてバーストする必要がある場合、余剰クレジットを消費する前に、蓄積されたクレジットが常に消費されます。蓄積した CPU クレジット残高を使いきると、必要な期間だけ余剰クレジットを使用してバーストできます。CPU 利用率がベースラインを下回った場合、インスタンスが獲得クレジットを蓄積する前に常に余剰クレジットが支払われます。

この 5 分間に発生するアクティビティを反映するため、次の式では Adjusted balance という用語を使用します。CPUCreditBalance および CPUSurplusCreditBalance の CloudWatch メトリクスの値に達するため、この値を使用します。

Example

```
Adjusted balance = [prior CPUCreditBalance - prior CPUSurplusCreditBalance] + [Credits
 earned per hour * (5/60) - CPUCreditUsage]
```

0 に対する Adjusted balance の値は、インスタンスが獲得したすべてのクレジットがバーストに消費され、余剰クレジットは消費されなかったことを示します。結果として、CPUCreditBalance と CPUSurplusCreditBalance は 0 に設定されます。

Adjusted balance の正の値は、インスタンスが獲得クレジットを蓄積し、前の余剰クレジットが (ある場合) 支払われたことを示します。結果として、Adjusted balance の値は CPUCreditBalance に割り当てられ、CPUSurplusCreditBalance は 0に設定されます。インスタンスサイズは、蓄積可能な最大クレジット数を決定します。

Example

```
CPUCreditBalance = min [max earned credit balance, Adjusted balance]
CPUSurplusCreditBalance = 0
```

Adjusted balance の負の値は、インスタンスが蓄積したすべての獲得クレジットに加えて、余剰クレジットもバーストに消費されたことを示します。結果として、Adjusted balance の値は CPUSurplusCreditBalance と CPUCreditBalance に割り当てられ、0に設定されます。繰り返しになりますが、インスタンスサイズは、蓄積可能な最大クレジット数を決定します。

Example

```
CPUSurplusCreditBalance = min [max earned credit balance, -Adjusted balance]
CPUCreditBalance = 0
```

消費される余剰クレジットがインスタンスに蓄積可能な最大クレジットを越えた場合、余剰クレジット残高は前述の式に示すように最大に設定されます。残りの余剰クレジットは CPUSurplusCreditsCharged メトリクスで示すように課金されます。

Example

```
CPUSurplusCreditsCharged = max [-Adjusted balance - max earned credit balance, 0]
```

最後に、CPUSurplusCreditBalance により追跡された余剰クレジットもインスタンスの終了時に課金されます。インスタンスを unlimited から standard に切り替えると、残りの CPUSurplusCreditBalance も課金されます。

コンピュート最適化インスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

コンピューティング最適化インスタンスは、高パフォーマンスプロセッサから恩恵を受けるコンピューティングバウンドな用途に最適です。

C5 および C5n インスタンス

これらのインスタンスは、次の用途に適しています。

- 作業負荷の Batch 処理
- メディアの変換
- 高性能なウェブサーバー
- ハイパフォーマンスコンピューティング (HPC)
- 科学的なモデル

- 専用ゲームサーバーおよび広告エンジン
- 機械学習推論やその他の大量の演算を行うアプリケーション

`c5.metal` などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスすることができます。

詳細については、「Amazon EC2 C5 インスタンス」を参照してください。

C6G、C6GD、および C6gn インスタンス

これらのインスタンスは AWS Graviton2 プロセッサを搭載しており、次のような、高度なコンピューティング集約型ワークロードの実行に最適です。

- ハイパフォーマンスコンピューティング (HPC)
- Batch 処理
- 広告配信
- 動画エンコーディング
- ゲームサーバー
- 科学的なモデル
- 分散分析
- CPU ベースの機械学習推論

`c6g.metal` などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスすることができます。

詳細については、「Amazon EC2 C6g インスタンス」を参照してください。

C6i および C6id インスタンス

これらのインスタンスは、次のような高度で計算負荷の高いワークロードの実行に最適です。

- ハイパフォーマンスコンピューティング (HPC)
- Batch 処理
- 広告配信
- 動画エンコーディング
- 分散分析

- 拡張性の高いマルチプレイヤーゲーム

C6in インスタンス

これらのインスタンスは、次のような計算が重いワークロードに適しています。

- 分散コンピューティングアプリケーション
- ネットワーク仮想アプライアンス
- データ分析
- ハイパフォーマンスコンピューティング (HPC)
- CPU ベースの AI/ML

詳細については、「Amazon EC2 C6i インスタンス」を参照してください。

C7a インスタンス

これらのインスタンスは第 4 世代 AMD EPYC プロセッサを搭載しており、次のような大量の演算を行うワークロードの実行に最適です。

- ハイパフォーマンスコンピューティング (HPC)
- Batch 処理
- 広告配信
- 動画エンコーディング
- ゲームサーバー
- 科学的なモデル
- 分散分析

詳細については、「Amazon EC2 C7a instances」を参照してください。

C7g および C7gd インスタンス

これらのインスタンスは AWS Graviton3 プロセッサを搭載しており、次のような高度で計算負荷の高いワークロードの実行に最適です。

- ハイパフォーマンスコンピューティング (HPC)
- Batch 処理

- 広告配信
- 動画エンコーディング
- ゲームサーバー
- 科学的なモデル
- 分散分析

詳細については、「Amazon EC2 C7g インスタンス」を参照してください。

C7gn インスタンス

新しい AWS Nitro Card を搭載した C7gn インスタンスは、Graviton ベースの Amazon EC2 インスタンスに対して最高のネットワーク帯域幅およびパケット処理パフォーマンスを提供します。C7gn インスタンスは、前世代の C6gn インスタンスと比較して、最大 200 Gbps のネットワーク帯域幅と最大 50% 高いパケット処理パフォーマンスを提供します。C7gn インスタンスは、次のようなネットワーク集約型のワークロードに最適です。

- ネットワーク仮想アプライアンスのワークロード
- データ分析などのデータ負荷の高いワークロード
- CPU ベースの人工知能および機械学習 (AI/ML) 推論ワークロード

詳細については、「Amazon EC2 C7g インスタンス」を参照してください。

C7i インスタンス

C7i インスタンスは、バッチ処理、機械学習、ハイエンドゲーム、広告配信、動画エンコーディングなど、コンピューティング負荷の高いワークロードの実行に最適です。

詳細については、「Amazon EC2 C7i インスタンス」を参照してください。

コンテンツ

ハードウェア仕様

仮想中央処理ユニット (vCPU) は、仮想マシン (VM) に割り当てられた物理 CPU の一部を表します。x86 インスタンスの場合、コアごとに 2 つの vCPU があります。Graviton インスタンスの場合、コアごとに 1 つの vCPU があります。

ハードウェアの仕様については、「Amazon EC2 インスタンスタイプガイド」の「コンピューティング最適化インスタンス」を参照してください。

インスタンスのパフォーマンス

EBS 最適化インスタンスは、インスタンスからの Amazon EBS I/O とその他のネットワークトラフィックとの競合を排除することによって、EBS ボリュームの安定した高パフォーマンスを実現できます。一部のコンピューティングの最適化インスタンスは、追加料金なしでデフォルトで EBS 最適化されています。詳細については、Amazon EBS 最適化インスタンスを使用する を参照してください。

一部のコンピューティングの最適化インスタンスでは、Linux でプロセッサの C ステートと P ステートを制御できます。C ステートは非アクティブ時のコアのスリープレベルを制御し、P ステートは希望するコアからのパフォーマンス (CPU 周波数) を制御します。詳細については、EC2 インスタンスのプロセッサのステート制御 を参照してください。

ネットワークパフォーマンス

サポートされているインスタンスタイプで拡張ネットワーキングを有効にすると、レイテンシーとネットワークジッターを低減し、パケット毎秒 (PPS) のパフォーマンスを高めることができます。ほとんどのアプリケーションでは、高いレベルのネットワークパフォーマンスが一貫して必要なわけではありませんが、データの送受信時にアクセスする帯域幅を増やすことでメリットを得られます。詳細については、「Linux での拡張ネットワーキング」を参照してください。

ネットワークの仕様については、「Amazon EC2 インスタンスタイプガイド」の「コンピューティング最適化インスタンス」を参照してください。

Amazon EBS I/O パフォーマンス

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、Amazon EBS ボリュームの最高のパフォーマンスを実現します。

詳細については、「Amazon EBS 最適化インスタンスを使用する」を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスストアボリュームは、インスタンスの存続中のみ使用できます。インスタンスを停止、休止、または終了すると、アプリケーションとそのインスタンスストアボリュームのデータは消去されます。インスタンスストアボリュームの重要なデータは、定期的にバックアップまたはレプリケートすることをお勧めします。詳細については、「Amazon EC2 インスタンスストア」および「SSD インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仕様については、「Amazon EC2 インスタンスタイプガイド」の「コンピューティング最適化インスタンス」を参照してください。

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これは、SSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは、利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み操作の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは、複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つには、SSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームには、オーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすには、ボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。

リリースノート

- Nitro System で構築された C4 インスタンスおよびインスタンスには、64 ビットの EBS-backed HVM AMIs が必要です。これらのインスタンスはハイメモリであるため、そのキャパシティーを活用するには 64 ビットのオペレーティングシステムが必要です。HVM AMI は、ハイメモリインスタンスタイプの準仮想化 (PV) AMI よりも優れたパフォーマンスを提供します。さらに、拡張ネットワーキングを利用するには、HVM AMI を使用する必要があります。
- C7g インスタンスは、最新世代の AWS Graviton3 プロセッサによって駆動されます。第 6 世代の AWS Graviton2 ベースの C6G インスタンスに比べて、最大 25% パフォーマンスが向上します。C7g インスタンスはクラウドで初めて DDR5 メモリを搭載し、DDR4 メモリと比較して 50% 高いメモリ帯域幅を提供し、メモリ内のデータへの高速アクセスを可能にします。
  - C7g インスタンスは、インスタンス内のコア間で累積する浮動小数点乗算など、ハイパワー命令の総実行レートを制限します。他のワークロードセグメントに重点を置いた将来のインスタンスタイプには、この制限がない場合があります。
- Nitro System 上に構築されたインスタンスには、次の要件があります。
  - NVMe ドライバーがインストールされている必要があります。
  - Elastic Network Adapter (ENA) ドライバーがインストールされている必要があります。

  以下の Linux AMI はこれらの要件を満たしています。
  - AL2023
  - Amazon Linux 2
  - Amazon Linux AMI 2018.03 以降
  - `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

    > **Note**
    >
    > AWS Graviton ベースのインスタンスタイプには、`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

  - Red Hat Enterprise Linux 7.4 以降
  - SUSE Linux Enterprise Server 12 SP2 以降
  - CentOS 7.4.1708 以降
  - FreeBSD 11.1 以降
  - Debian GNU/Linux 9 以降
- AWS Graviton プロセッサを使用するインスタンスには、次の要件があります。

  - 64 ビット Arm アーキテクチャ用の AMI を使用する必要があります。
  - ACPI テーブルを含む UEFI による起動と、PCI デバイスの ACPI ホットプラグをサポートしている必要があります。

  以下の AMI はこれらの要件を満たしています。

  - Amazon Linux 2 (64 ビット Arm)
  - Ubuntu 16.04 以降 (64 ビット Arm)
  - Red Hat Enterprise Linux 8.0 以降 (64 ビット Arm)
  - SUSE Linux Enterprise Server 15 以降 (64 ビット Arm)
  - Debian 10 以降 (64 ビット Arm)
- C6i インスタンスで最良のパフォーマンスを発揮させるには、ENA ドライバーのバージョン 2.2.9 以降を使用する必要があります。1.2 もしくは バージョンより前のENAドライバをこれらのインスタンスとともに使用すると、ネットワークインターフェイスのアタッチメントが失敗します。互換性のある ENA ドライバが利用できる AMI を以下に示します。
  - AL2023
  - Amazon Linux 2 (カーネル 4.14.186 以降)
  - Ubuntu 20.04 (カーネル 5.4.0-1025-aws 以降)
  - Red Hat Enterprise Linux 8.3 (カーネル 4.18.0-240.1.1.el8_3.ARCH 以降)
  - SUSE Linux Enterprise Server 15 SP2 (カーネル 5.3.18-24.15.1 以降)
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。詳細については、「インスタンスボリューム数の制限」を参照してください。
- C6gn インスタンスから最高のパフォーマンスを得るには、ENA ドライバーバージョン 2.2.9 以降を使用していることを確認してください。1.2 バージョンより前のENAドライバをこれらのインスタンスとともに使用すると、ネットワークインターフェイスのアタッチメントが失敗します。互換性のある ENA ドライバが利用できる AMI を以下に示します。
  - AL2023
  - Amazon Linux 2 (カーネル 4.14.186 以降)
  - Ubuntu 20.04 (カーネル 5.4.0-1025-aws 以降)
  - Red Hat Enterprise Linux 8.3 (カーネル 4.18.0-240.1.1.el8_3.ARCH 以降)
  - SUSE Linux Enterprise Server 15 SP2 (カーネル 5.3.18-24.15.1 以降)

- C6gn インスタンスですべての Linux ディストリビューションの AMI を起動するには、最新バージョンの AMI を使用し、最新のドライバーに更新します。以前のバージョンをお使いの場合は、GitHub から最新のドライバーをダウンロードしてください。
- ベアメタルインスタンスを起動すると、基盤となるサーバーが起動します。これには、すべてのハードウェアやファームウェアコンポーネントの確認が含まれます。つまり、インスタンスが実行状態になってからネットワーク経由で使用できるようになるまでに 20 分かかることがあります。
- EBS ボリュームまたはセカンダリネットワークインターフェイスを、ベアメタルインスタンスにアタッチ (または、そこからデタッチ) するには、PCIe のネイティブホットプラグがサポートされている必要があります。PCIe のネイティブホットプラグは、Amazon Linux 2 および最新バージョンの Amazon Linux AMI でサポートされています。それ以前のバージョンではサポートされていません。次の Linux カーネル設定オプションを有効にする必要があります。

```
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_PCIEASPM=y
```

- ベアメタルインスタンスでは、I/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。アップストリームの Linux カーネルと最新の Amazon Linux AMI は、このデバイスをサポートしています。また、ベアメタルインスタンスでは、システムが PCI ベースのシリアルデバイスを自動的に使用できるようにする ACPI SPCR テーブルも使用できます。最新の Windows AMI では、自動的に PCI ベースのシリアルデバイスが使用されます。
- Nitro System 上に構築されたインスタンスには、API リクエストによるクリーンシャットダウンをサポートするために acpid がインストールされている必要があります。
- リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。詳細については、Amazon EC2 の「よくある質問」の「Amazon EC2 で実行できるインスタンス数の上限は」を参照してください。

メモリ最適化インスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

メモリ最適化インスタンスは、メモリ内の大きいデータセットを処理するワークロードに対して高速なパフォーマンスを実現するように設計されています。

R5、R5a、R5b、および R5n インスタンス

これらのインスタンスは、次の用途に適しています。

- リレーショナルな MySQL や NoSQL、例えば MongoDB や Cassandra データベースを含むハイパフォーマンスなもの。
- Memcached や Redis などのキー値タイプのデータのインメモリキャッシュを提供する分散型ウェブスケールキャッシュストア
- SAP や HANA などの、ビジネスインテリジェンス用に最適化されたデータストレージ形式と分析機能を使用するインメモリデータベース
- Hadoop や Spark クラスターを使用する巨大な非構造化データのリアルタイム処理を実行するアプリケーション
- ハイパフォーマンスコンピューティング (HPC) および Electronic Design Automation (EDA) アプリケーション。

`r5.metal` などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスすることができます。

詳細については、「Amazon EC2 R5 インスタンス」を参照してください。

R6a インスタンス

これらのインスタンスは、次のようなメモリを大量に使用するワークロードの実行に最適です。

- リレーショナルおよび NoSQL の両方の高性能データベース
- Memcached や Redis など、分散したウェブスケールインメモリキャッシュ
- Hadoop クラスターや Spark クラスターなど、リアルタイムのビッグデータ分析

R6g および R6gd インスタンス

これらのインスタンスは AWS Graviton2 プロセッサを搭載しており、次のようなメモリを大量に使用するワークロードの実行に最適です。

- MySQL、MariaDB、PostgreSQL などのオープンソースデータベース

- Memcached、Redis、KeyDB などのインメモリキャッシュ

`r6g.metal` などのベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスすることができます。

詳細については、「Amazon EC2 R6g インスタンス」を参照してください。

R6i および R6id インスタンス

これらのインスタンスは、次のようなメモリ集約型のワークロードの実行に最適です。

- 高性能データベース、リレーショナルおよび NoSQL
- SAP HANA などのインメモリデータベース
- Memcached や Redis など、分散したウェブスケールインメモリキャッシュ
- Hadoop クラスターや Spark クラスターを含む、リアルタイムのビッグデータ分析

R6in および R6idn インスタンス

これらのインスタンスは、次のようなネットワークを多用するワークロードに適しています。

- 高性能リレーショナル (MySQL および NoSQL) MongoDB データベースやカサンドラデータベースなど
- Memcached および Redis を含む、キー値タイプのデータのインメモリキャッシュを提供する分散型ウェブスケールキャッシュストア
- SAP や HANA などの、ビジネスインテリジェンス用に最適化されたデータストレージ形式と分析機能を使用するインメモリデータベース
- Hadoop クラスターや Spark クラスターなど、金融サービス向けのリアルタイムのビッグデータ分析

詳細については、「Amazon EC2 R6i インスタンス」を参照してください。

R7a インスタンス

これらのインスタンスは第 4 世代 AMD EPYC プロセッサを搭載しており、次のようなメモリを大量に使用するワークロードの実行に最適です。

- リレーショナルおよび NoSQL の両方の高性能データベース
- Memcached や Redis など、分散したウェブスケールインメモリキャッシュ

- Hadoop クラスターや Spark クラスターなど、リアルタイムのビッグデータ分析

R7i インスタンス

R7i インスタンスは第 4 世代 Intel Xeon プロセッサを搭載しており、次のようなメモリを大量に使用するワークロードの実行に最適です。

- 高性能データベース
- 分散したウェブスケールインメモリキャッシュ
- SAP HANA などのメモリ内データベース
- Hadoop クラスターや Spark クラスターなど、リアルタイムのビッグデータ分析

R7iz インスタンス

R7iz は第 4 世代の Intel Xeon プロセッサを搭載した、高周波数の大容量メモリインスタンスです。最大 3.9 GHz の持続的なオールコアターボ周波数、最大 1024 GB のシステムメモリ、最大 50 Gbps のネットワーク帯域幅、最大 40 Gbps の Amazon EBS 専用帯域幅を提供します。

R7iz インスタンスは、次のような高い処理能力と大容量メモリの組み合わせを必要とするワークロードに最適です。

- 電子設計自動化
- リレーショナルデータベース
- データ分析シミュレーション

詳細については、「Amazon EC2 R7iz インスタンス」を参照してください。

R7g および R7gd インスタンス

これらのインスタンスは AWS Graviton3 プロセッサを搭載しており、次のようなメモリを大量に使用するワークロードの実行に最適です。

- MySQL、MariaDB、PostgreSQL などのオープンソースデータベース
- Memcached、Redis、KeyDB などのインメモリキャッシュ

詳細については「Amazon EC2 R7g instances」を参照してくだい。

ハイメモリ (u-*) インスタンス

このインスタンスでは、インスタンスごとに 3 TiB、6 TiB、9 TiB、12 TiB、18 TiB、および 24 TiB のメモリが利用可能です。これらは、SAP HANA インメモリデータベースの実稼働向けデプロイを含む、大規模なインメモリデータベースを実行するように設計されています。

詳細については、「Amazon EC2ハイメモリインスタンス」と「SAP HANA のストレージ設定」を参照してください。サポートされるオペレーティングシステムの詳細については、「AWS での SAP HANA のEC2 ハイメモリインスタンスへの移行」を参照してください。

X1 インスタンス

これらのインスタンスは、次の用途に適しています。

- SAP HANA などのメモリ内データベース (Business Suite S/4HANA の SAP 認定サポート、Business Suite on HANA (SoH)、Business Warehouse on HANA (BW)、および Data Mart Solutions on HANA を含む)。詳細については、「AWS クラウドの SAP HANA」を参照してください。
- Apache Spark や Presto などのビッグデータ処理エンジン。
- ハイパフォーマンスコンピューティング (HPC) アプリケーション。

詳細については、「Amazon EC2 X1 インスタンス」を参照してください。

X1e インスタンス

これらのインスタンスは、次の用途に適しています。

- 高性能データベース。
- SAP HANA などのメモリ内データベース。詳細については、「AWS クラウドの SAP HANA」を参照してください。
- メモリを大量に消費するエンタープライズアプリケーション。

詳細については、「Amazon EC2 X1e インスタンス」を参照してください。

X2gd インスタンス

これらのインスタンスは、次の用途に適しています。

- Redis や Memcached などのインメモリデータベース。
- MySQL や PostgreSQL などのリレーショナルデータベース。
- 物理検証やレイアウトツールなどの Electronic Design Automation (EDA) ワークロード。

- リアルタイム分析やリアルタイムキャッシュサーバーなど、メモリに負担がかかるワークロード。

詳細については、「Amazon EC2 X2g Instances (Amazon EC2 X2g インスタンス)」を参照してください。

X2idn、X2iedn、および X2iezn インスタンス

これらのインスタンスは、次の用途に適しています。

- Redis や Memcached などのインメモリデータベース。
- MySQL や PostgreSQL などのリレーショナルデータベース。
- 物理検証やレイアウトツールなどの Electronic Design Automation (EDA) ワークロード。
- リアルタイム分析やリアルタイムキャッシュサーバーなど、メモリに負担がかかるワークロード。

詳細については、「Amazon EC2 X2i Instances (Amazon EC2 X2i インスタンス)」を参照してください。

z1d インスタンス

これらのインスタンスは、ハイコンピューティングとハイメモリの両方を提供し、以下の場合に最適です。

- Electronic Design Automation (EDA)
- リレーショナルデータベースワークロード

`z1d.metal` インスタンスは、プロセッサとメモリなどのホストサーバーの物理リソースにアプリケーションが直接アクセスできるようにします。

詳細については、「Amazon EC2 z1d インスタンス」を参照してください。

コンテンツ

- 個の vCPU のサポート
- リリースノート

ハードウェア仕様

仮想中央処理ユニット (vCPU) は、仮想マシン (VM) に割り当てられた物理 CPU の一部を表します。x86 インスタンスの場合、コアごとに 2 つの vCPU があります。Graviton インスタンスの場合、コアごとに 1 つの vCPU があります。

ハードウェアの仕様については、「Amazon EC2 インスタンスタイプガイド」の「メモリ最適化インスタンス」を参照してください。

メモリ性能

X1 インスタンスには、Intel Scalable Memory Buffer が備わっており、300 GiB/秒の持続可能なメモリ読み取り帯域幅と、140 GiB/秒の持続可能なメモリ書き込み帯域幅が利用可能です。

メモリ最適化インスタンスに対して有効にできる RAM の量の詳細については、「ハードウェア仕様」を参照してください。

メモリ最適化インスタンスにはハイメモリがあり、その処理能力を活用するためには 64 ビットの HVM AMI が必要です。HVM AMI は、メモリ最適化インスタンスの準仮想化 (PV) AMI よりも優れたパフォーマンスを提供します。詳細については、「Linux AMI 仮想化タイプ」を参照してください。

インスタンスのパフォーマンス

メモリ最適化インスタンスでは、最新の Intel AES-NI 機能を通じて暗号化のパフォーマンスが向上し、Advanced Vector Extensions 2 (Intel AVX2) プロセッサ命令のサポートにより、ほとんどの整数コマンドが 256 ビットに拡大されます。

一部のメモリ最適化インスタンスでは、Linux のプロセッサの C ステートと P ステートを制御できます。C ステートは非アクティブ時のコアのスリープレベルを制御し、P ステートは希望するコアからのパフォーマンス (CPU 周波数で測定) を制御します。詳細については、EC2 インスタンスのプロセッサのステート制御 を参照してください。

ネットワークパフォーマンス

サポートされているインスタンスタイプで拡張ネットワーキングを有効にすると、レイテンシーとネットワークジッターを低減し、パケット毎秒 (PPS) のパフォーマンスを高めることができます。ほとんどのアプリケーションでは、高いレベルのネットワークパフォーマンスが一貫して必要なわけ

ではありませんが、データの送受信時にアクセスする帯域幅を増やすことでメリットを得られます。詳細については、「Linux での拡張ネットワーキング」を参照してください。

ネットワークの仕様については、「Amazon EC2 インスタンスタイプガイド」の「メモリ最適化インスタンス」を参照してください。

Amazon EBS I/O パフォーマンス

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、Amazon EBS ボリュームの最高のパフォーマンスを実現します。

詳細については、「Amazon EBS 最適化インスタンスを使用する」を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスストアボリュームは、インスタンスの存続中のみ使用できます。インスタンスを停止、休止、または終了すると、アプリケーションとそのインスタンスストアボリュームのデータは消去されます。インスタンスストアボリュームの重要なデータは、定期的にバックアップまたはレプリケートすることをお勧めします。詳細については、「Amazon EC2 インスタンスストア」および「SSD インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仕様については、「Amazon EC2 インスタンスタイプガイド」の「メモリ最適化インスタンス」を参照してください。

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これは、SSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは、利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み操作の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは、複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つには、SSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビ

ジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームには、オーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすには、ボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。

個の vCPU のサポート

メモリ最適化インスタンスは多数の vCPU を提供するため、vCPU の制限が低いオペレーティングシステムで起動の問題が発生することがあります。メモリ最適化インスタンスを起動する場合は、最新の AMI を使用することをお勧めします。

以下の AMI では、メモリ最適化インスタンスの起動がサポートされています。

- Amazon Linux 2 (HVM)
- Amazon Linux AMI 2016.03 (HVM) 以降
- Ubuntu Server 14.04 LTS (HVM)
- Red Hat Enterprise Linux 7.1 (HVM)
- SUSE Linux Enterprise Server 12 SP1 (HVM)
- Windows Server 2019
- Windows Server 2016
- Windows Server 2012 R2
- Windows Server 2012
- Windows Server 2008 R2 64 ビット
- Windows Server 2008 SP2 64 ビット

リリースノート

- Nitro System 上に構築されたインスタンスには、次の要件があります。
  - NVMe ドライバーがインストールされている必要があります。

- Elastic Network Adapter (ENA) ドライバーがインストールされている必要があります。

以下の Linux AMI はこれらの要件を満たしています。

- AL2023
- Amazon Linux 2
- Amazon Linux AMI 2018.03 以降
- `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

> **Note**
>
> AWS Graviton ベースのインスタンスタイプには、`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

- Red Hat Enterprise Linux 7.4 以降
- SUSE Linux Enterprise Server 12 SP2 以降
- CentOS 7.4.1708 以降
- FreeBSD 11.1 以降
- Debian GNU/Linux 9 以降

- AWS Graviton プロセッサを使用するインスタンスには、次の要件があります。
  - 64 ビット Arm アーキテクチャ用の AMI を使用する必要があります。
  - ACPI テーブルを含む UEFI による起動と、PCI デバイスの ACPI ホットプラグをサポートしている必要があります。

  以下の AMI はこれらの要件を満たしています。

  - Amazon Linux 2 (64 ビット Arm)
  - Ubuntu 16.04 以降 (64 ビット Arm)
  - Red Hat Enterprise Linux 8.0 以降 (64 ビット Arm)
  - SUSE Linux Enterprise Server 15 以降 (64 ビット Arm)
  - Debian 10 以降 (64 ビット Arm)
- R6i インスタンスに最良のパフォーマンスを発揮させるには、ENA ドライバーのバージョン 2.2.9 以降を使用する必要があります。1.2 もしくは バージョンより前のENAドライバをこれらのインスタンスとともに使用すると、ネットワークインターフェイスのアタッチメントが失敗します。互換性のある ENA ドライバが利用できる AMI を以下に示します。
  - AL2023

  - Amazon Linux 2 (カーネル 4.14.186 以降)
  - Ubuntu 20.04 (カーネル 5.4.0-1025-aws 以降)
  - Red Hat Enterprise Linux 8.3 (カーネル 4.18.0-240.1.1.el8_3.ARCH 以降)
  - SUSE Linux Enterprise Server 15 SP2 (カーネル 5.3.18-24.15.1 以降)
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。詳細については、「インスタンスボリューム数の制限」を参照してください。
- ベアメタルインスタンスを起動すると、基盤となるサーバーが起動します。これには、すべてのハードウェアやファームウェアコンポーネントの確認が含まれます。つまり、インスタンスが実行状態になってからネットワーク経由で使用できるようになるまでに 20 分かかることがあります。
- EBS ボリュームまたはセカンダリネットワークインターフェイスを、ベアメタルインスタンスにアタッチ (または、そこからデタッチ) するには、PCIe のネイティブホットプラグがサポートされている必要があります。PCIe のネイティブホットプラグは、Amazon Linux 2 および最新バージョンの Amazon Linux AMI でサポートされています。それ以前のバージョンではサポートされていません。次の Linux カーネル設定オプションを有効にする必要があります。

```
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_PCIEASPM=y
```

- ベアメタルインスタンスでは、I/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。アップストリームの Linux カーネルと最新の Amazon Linux AMI は、このデバイスをサポートしています。また、ベアメタルインスタンスでは、システムが PCI ベースのシリアルデバイスを自動的に使用できるようにする ACPI SPCR テーブルも使用できます。最新の Windows AMI では、自動的に PCI ベースのシリアルデバイスが使用されます。
- Windows Server 2008 SP2 64 ビット AMI を使用して、X1 インスタンス (`x1.16xlarge`インスタンスは除く) を起動することはできません。
- Windows Server 2008 SP2 64 ビット AMI を使用して、X1e インスタンスを起動することはできません。
- Windows Server 2008 R2 64-bit AMI の旧バージョンでは、`r4.large` および `r4.4xlarge` インスタンスを起動できません。この問題が発生した場合は、この AMI の最新バージョンに更新してください。
- リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。詳細については、Amazon EC2 の「よくある質問」の「Amazon EC2 で実行できるインスタンス数の上限は」を参照してください。

ストレージ最適化インスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

ストレージ最適化インスタンスは、ローカルストレージの大規模データセットに対する高いシーケンシャル読み取りおよび書き込みアクセスを必要とするワークロード用に設計されています。ストレージ最適化インスタンスは、数万回の低レイテンシーとランダム I/O オペレーション/秒 (IOPS) をアプリケーションに提供するように最適化されています。使用されているテクノロジーなどの詳細については、「Amazon EC2 インスタンスタイプの詳細」ページを参照してください。

D2 インスタンス

これらのインスタンスは、次の用途に適しています。

- 超並列処理 (MPP) データウェアハウス
- MapReduce および Hadoop 分散コンピューティング
- ログまたはデータ処理アプリケーション

D3 および D3en インスタンス

これらのインスタンスは、インスタンスストレージのスケールアウトを提供し、以下に適しています。

- Hadoop ワークロード向けの分散ファイルシステム
- GPFC や BeeFS などのファイルストレージワークロード
- HPC ワークロード向けの大規模なデータレイク

H1 インスタンス

これらのインスタンスは、次の用途に適しています。

- MapReduce および分散ファイルシステムなどのデータ負荷の高いワークロード

- 直接アタッチされたインスタンスストレージにある大量データへのシーケンシャルアクセスを必要とするアプリケーション
- 大量のデータへの高スループットアクセスを必要とするアプリケーション

I3 および I3en インスタンス

これらのインスタンスは、次の用途に適しています。

- 高頻度オンライントランザクション処理 (OLTP) システム
- リレーショナルデータベース
- NoSQL データベース
- メモリ内データベース (Redis など) のキャッシュ
- データウェアハウスアプリケーション
- 分散されたファイルシステム

ベアメタルインスタンスを使用すると、アプリケーションから、プロセッサとメモリなどのホストサーバーの物理リソースに直接アクセスすることができます。

詳細については、「Amazon EC2 I3 インスタンス」を参照してください。

I4i インスタンス

これらのインスタンスは、トランザクションデータベースや NoSQL データベースなど、ローカルストレージで小規模から中規模のデータセットを必要とする I/O 負荷の高いワークロードに適しています。

詳細については、「Amazon EC2 I4i インスタンス」を参照してください。

I4g インスタンス

I4g インスタンスには、AWS Graviton 2 プロセッサと AWS Nitro SSD が搭載されています。I4g インスタンスは、ランダムな読み取り/書き込みが混在するオペレーションを多く実行し、I/O レイテンシーの影響を非常に受けやすく、高い CPU パフォーマンスを必要とするワークロードに最適です (SQL または NoSQL データベース、オンライントランザクション処理、リアルタイム分析など)。詳細については、「Amazon EC2 I4g インスタンス」を参照してください。

Im4gn インスタンス

これらのインスタンスは、レイテンシーが低くランダム I/O パフォーマンスが高い必要がある次のようなワークロードに適しています。

- リレーショナルデータベース
- NoSQL データベース
- 検索
- 分散されたファイルシステム

詳細については、「Amazon EC2 im4gn および is4Gen インスタンス」を参照してください。

is4Gen インスタンス

これらのインスタンスは、レイテンシーが低くランダム I/O パフォーマンスが高い必要がある次のようなワークロードに適しています。

- NoSQL データベース
- インデックス作成
- ストリーミング
- キャッシュ
- ウォームストレージ

詳細については、「Amazon EC2 im4gn および is4Gen インスタンス」を参照してください。

コンテンツ

ハードウェア仕様

仮想中央処理ユニット (vCPU) は、仮想マシン (VM) に割り当てられた物理 CPU の一部を表します。x86 インスタンスの場合、コアごとに 2 つの vCPU があります。Graviton インスタンスの場合、コアごとに 1 つの vCPU があります。

ハードウェアの仕様については、「Amazon EC2 インスタンスタイプガイド」の「ストレージ最適化インスタンス」を参照してください。

インスタンスのパフォーマンス

Linux のインスタンスから最善のディスクスループットパフォーマンスを得るには、最新バージョンの Amazon Linux 2 または Amazon Linux AMI を使用することをお勧めします。

NVMe インスタンスストアボリュームを持つインスタンスの場合は、カーネルバージョン 4.4 以降の Linux AMI を使用する必要があります。それ以外の場合、インスタンスは利用可能な最大の IOPS パフォーマンスを達成できません。

D2 インスタンスは、永続許可 (ディスクスループットと拡張性を大幅に向上させる Xen ブロックリングプロトコルの拡張機能) をサポートする Linux カーネルを使用するときに、最大のディスクパフォーマンスを提供します。永続許可の詳細については、Xen プロジェクトのブログのこちらの記事を参照してください。

EBS 最適化インスタンスは、インスタンスからの Amazon EBS I/O とその他のネットワークトラフィックとの競合を排除することによって、EBS ボリュームの安定した高パフォーマンスを実現できます。一部のストレージ最適化インスタンスは、追加料金なしでデフォルトで EBS 最適化されます。詳細については、Amazon EBS 最適化インスタンスを使用する を参照してください。

一部のストレージ最適化インスタンスでは、Linux でプロセッサの C ステートと P ステートを制御できます。C ステートは非アクティブ時のコアのスリープレベルを制御し、P ステートは希望するコアからのパフォーマンス (CPU 周波数) を制御します。詳細については、EC2 インスタンスのプロセッサのステート制御 を参照してください。

ネットワークパフォーマンス

サポートされているインスタンスタイプで拡張ネットワーキングを有効にすると、レイテンシーとネットワークジッターを低減し、パケット毎秒 (PPS) のパフォーマンスを高めることができます。ほとんどのアプリケーションでは、高いレベルのネットワークパフォーマンスが一貫して必要なわけではありませんが、データの送受信時にアクセスする帯域幅を増やすことでメリットを得られます。詳細については、「Linux での拡張ネットワーキング」を参照してください。

ネットワークの仕様については、「Amazon EC2 インスタンスタイプガイド」の「ストレージ最適化インスタンス」を参照してください。

Amazon EBS I/O パフォーマンス

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、Amazon EBS ボリュームの最高のパフォーマンスを実現します。

詳細については、「Amazon EBS 最適化インスタンスを使用する」を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスストアボリュームは、インスタンスの存続中のみ使用できます。インスタンスを停止、休止、または終了すると、アプリケーションとそのインスタンスストアボリュームのデータは消去されます。インスタンスストアボリュームの重要なデータは、定期的にバックアップまたはレプリケートすることをお勧めします。詳細については、「Amazon EC2 インスタンスストア」および「SSD インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仕様については、「Amazon EC2 インスタンスタイプガイド」の「ストレージ最適化インスタンス」を参照してください。

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これは、SSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは、利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み操作の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは、複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つには、SSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームには、オーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすには、

ボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。

個の vCPU のサポート

d2.8xlarge インスタンスでは 36 個の vCPU が提供されますが、これにより vCPU を 32 個に制限している一部の Linux オペレーティング システムで問題が生じる可能性があります。d2.8xlarge インスタンスを起動する場合は、最新の AMI を使用することをお勧めします。

次の Linux AMI は、36 個の vCPU を使用した d2.8xlarge インスタンスの起動をサポートしています。

- Amazon Linux 2 (HVM)
- Amazon Linux AMI 2018.03(HVM)
- Ubuntu Server 14.04 LTS (HVM) 以降
- Red Hat Enterprise Linux 7.1 (HVM)
- SUSE Linux Enterprise Server 12 (HVM)

アプリケーションに別の AMI を使用する必要がある場合に、d2.8xlarge インスタンスの起動が正常に完了しないとき (stopped 状態遷移に伴って起動中にインスタンスのステータスが Client.InstanceInitiatedShutdown に変更されるときなど) は、以下の手順に従って 32 個を超える vCPU をサポートするようにインスタンスを変更し、d2.8xlarge インスタンスタイプを使用できるようにします。

32 個の以上の vCPU をサポートするようにインスタンスを更新する

1. AMI を使用して D2 インスタンスを起動し、d2.8xlarge 以外の D2 インスタンスタイプを選択します。
2. 使用するオペレーティングシステム固有の手順に従って、カーネルを最新バージョンに更新します。例えば、RHEL 6 の場合は、次のコマンドを使用します。

```
sudo yum update -y kernel
```

3. インスタンスを停止します。
4. (オプション) 将来的に必要な追加の d2.8xlarge インスタンスを起動するために使用できるインスタンスから AMI を作成します。
5. 停止したインスタンスのインスタンスタイプを d2.8xlarge に変更します ([Actions] (アクション)、[Instance settings] (インスタンスの設定)、[Change instance type] (インスタンスタイプの変更) の順にクリックし、その後の指示に従います)。
6. インスタンスを起動します。インスタンスが正常に起動すれば、完了です。これでもインスタンスが正しく実施されない場合、以下のステップに進みます。
7. (オプショナル) インスタンスがまだ正しく実施されない場合、インスタンスのカーネルが 32 個の vCPU をサポートしていない可能性があります。ただし、vCPU を制限すると、インスタンスを起動できる場合があります。

   a. 停止したインスタンスのインスタンスタイプを、d2.8xlarge を除く D2 インスタンスタイプのいずれかに変更します ([Actions] (アクション)、[Instance settings] (インスタンスの設定)、[Change instance type] (インスタンスタイプの変更) の順にクリックし、その後の指示に従います)。

   b. 使用するオペレーティングシステム固有の手順に従って、カーネル起動パラメータに maxcpus=32 オプションを追加します。例えば、RHEL 6 の場合は、/boot/grub/menu.lst ファイルを編集し、最新のアクティブな kernel エントリに次のオプションを追加します。

```
default=0
timeout=1
splashimage=(hd0,0)/boot/grub/splash.xpm.gz
hiddenmenu
title Red Hat Enterprise Linux Server (2.6.32-504.3.3.el6.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-2.6.32-504.3.3.el6.x86_64 maxcpus=32 console=ttyS0 ro
 root=UUID=9996863e-b964-47d3-a33b-3920974fdbd9 rd_NO_LUKS  KEYBOARDTYPE=pc
 KEYTABLE=us LANG=en_US.UTF-8 xen_blkfront.sda_is_xvda=1 console=ttyS0,115200n8
 console=tty0 rd_NO_MD SYSFONT=latarcyrheb-sun16 crashkernel=auto rd_NO_LVM
 rd_NO_DM
initrd /boot/initramfs-2.6.32-504.3.3.el6.x86_64.img
```

   c. インスタンスを停止します。

d. (オプション) 将来的に必要な追加の d2.8xlarge インスタンスを起動するために使用できるインスタンスから AMI を作成します。

e. 停止したインスタンスのインスタンスタイプを d2.8xlarge に変更します ([Actions] (アクション)、[Instance Settings] (インスタンスの設定)、[Change Instance Type] (インスタンスタイプの変更) の順にクリックし、その後の指示に従います)。

f. インスタンスを起動します。

リリースノート

- r7a.metal-48xl インスタンスは 2023 年 7 月以前にリリースされた Windows Server 2019 AMI と Windows Server 2016 AMI をサポートしていません。
- Nitro System 上に構築されたインスタンスには、次の要件があります。
  - NVMe ドライバーがインストールされている必要があります。
  - Elastic Network Adapter (ENA) ドライバーがインストールされている必要があります。

  以下の Linux AMI はこれらの要件を満たしています。
  - AL2023
  - Amazon Linux 2
  - Amazon Linux AMI 2018.03 以降
  - linux-aws カーネルを搭載した Ubuntu 14.04 以降

    > **Note**
    >
    > AWS Graviton ベースのインスタンスタイプには、linux-aws カーネル搭載の Ubuntu 18.04 以降が必要です

  - Red Hat Enterprise Linux 7.4 以降
  - SUSE Linux Enterprise Server 12 SP2 以降
  - CentOS 7.4.1708 以降
  - FreeBSD 11.1 以降
  - Debian GNU/Linux 9 以降
- AWS Graviton プロセッサを使用するインスタンスには、次の要件があります。
  - 64 ビット Arm アーキテクチャ用の AMI を使用する必要があります。

  - ACPI テーブルを含む UEFI による起動と、PCI デバイスの ACPI ホットプラグをサポートしている必要があります。

  以下の AMI はこれらの要件を満たしています。

  - Amazon Linux 2 (64 ビット Arm)
  - Ubuntu 16.04 以降 (64 ビット Arm)
  - Red Hat Enterprise Linux 8.0 以降 (64 ビット Arm)
  - SUSE Linux Enterprise Server 15 以降 (64 ビット Arm)
  - Debian 10 以降 (64 ビット Arm)
- ベアメタルインスタンスを起動すると、基盤となるサーバーが起動します。これには、すべてのハードウェアやファームウェアコンポーネントの確認が含まれます。つまり、インスタンスが実行状態になってからネットワーク経由で使用できるようになるまでに 20 分かかることがあります。
- EBS ボリュームまたはセカンダリネットワークインターフェイスを、ベアメタルインスタンスにアタッチ (または、そこからデタッチ) するには、PCIe のネイティブホットプラグがサポートされている必要があります。PCIe のネイティブホットプラグは、Amazon Linux 2 および最新バージョンの Amazon Linux AMI でサポートされています。それ以前のバージョンではサポートされていません。次の Linux カーネル設定オプションを有効にする必要があります。

```
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_PCIEASPM=y
```

- ベアメタルインスタンスでは、I/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。アップストリームの Linux カーネルと最新の Amazon Linux AMI は、このデバイスをサポートしています。また、ベアメタルインスタンスでは、システムが PCI ベースのシリアルデバイスを自動的に使用できるようにする ACPI SPCR テーブルも使用できます。最新の Windows AMI では、自動的に PCI ベースのシリアルデバイスが使用されます。
- FreeBSD AMI では、ベアメタルインスタンスは、起動に約 1 時間かかり、ローカルの NVMe ストレージへの I/O は完了しません。回避策として、次の行を `/boot/loader.conf` に追加し、再起動します。

```
hw.nvme.per_cpu_io_queues="0"
```

- `d2.8xlarge` インスタンスタイプには 36 個の vCPU がありますが、これにより vCPU を 32 個に制限している一部の Linux オペレーティング システムで問題が生じる可能性があります。詳細については、個の vCPU のサポート を参照してください。

- d3.8xlarge および d3en.12xlarge インスタンスは、ルートボリュームを含め、最大 3 つのアタッチメントをサポートします。ネットワークインターフェイスまたは EBS ボリュームを追加するときにアタッチメントの制限を超えると、インスタンスでアタッチメントの問題が発生します。
- リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。詳細については、Amazon EC2 の「よくある質問」の「Amazon EC2 で実行できるインスタンス数の上限は」を参照してください。

Linux 高速コンピューティングインスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

高速コンピューティングインスタンスは、ハードウェアアクセラレーターやコプロセッサーを使用して、浮動小数点数計算、グラフィック処理、データパターンマッチングのような機能を CPU で実行されるソフトウェア以上に効率的に実行します。これらのインスタンスでは、大量の演算を行うワークロードでさらに多くの並列処理が可能となり、より高いスループットが得られます。

高度な処理機能が必要な場合は、高速コンピューティングインスタンスを使用すると、Graphics Processing Units (GPU)、Field Programmable Gate Arrays (FPGA)、AWS Inferentia などのハードウェアベースのコンピューティングアクセラレーターにアクセスできます。

コンテンツ

- ネットワークパフォーマンス
- Amazon EBS I/O パフォーマンス
- SSD ベースのインスタンスストアボリュームの I/O パフォーマンス
- リリースノート
- P5 インスタンスの使用を開始する
- Linux インスタンスへの NVIDIA ドライバーのインストール
- Linux インスタンスへの AMD ドライバーのインストール
- G4ad でのデュアル 4K ディスプレイの設定
- NVIDIA GRID 仮想アプリケーションの有効化
- GPU 設定の最適化

GPU インスタンス

GPU ベースのインスタンスでは、数千のコンピューティングコアを持つ NVIDIA GPU にアクセスできます。これらのインスタンスを使用すると、CUDA または Open Computing Language (OpenCL) パラレルコンピューティングフレームワークを活用することにより、サイエンス、エンジニアリング、およびレンダリングアプリケーションを高速化できます。また、ゲームストリーミング、3D アプリケーションストリーミング、およびその他のグラフィックスワークロードを含む、グラフィックアプリケーションにも使用できます。

G5 インスタンス

G5 インスタンスは NVIDIA A10G GPU を使用し、リモートワークステーション、ビデオレンダリング、クラウドゲームなどのグラフィックスを多用するアプリケーション、自然言語処理、コンピュータビジョン、レコメンデーションエンジンなどのアプリケーションの深層学習モデル向けに高いパフォーマンスを提供します。これらのインスタンスは、最大 8 つの NVIDIA A10G GPU、第 2 世代 AMD EPYC プロセッサ、最大 100 Gbps のネットワーク帯域幅、最大 7.6 TB のローカル NVMe SSD ストレージを備えています。

詳細については、「Amazon EC2 G5 インスタンス」を参照してください。

G5g インスタンス

G5G インスタンスは NVIDIA T4G GPU を使用し、OpenGL や Vulkan などの業界標準の API を活用したゲームストリーミングやレンダリングなど、グラフィックスを多用するアプリケーションに高い

パフォーマンスを提供します。これらのインスタンスは、自然言語処理やコンピュータビジョンなどのアプリケーションのディープラーニングモデルの実行にも適しています。これらのインスタンスは、最大 2 つの NVIDIA T4G テンソルコア GPU、AWS Graviton2 プロセッサ、最大 25 Gbps のネットワーク帯域幅を備えています。

詳細については、「Amazon EC2 G5g インスタンス」を参照してください。

G4ad インスタンスと G4dn インスタンス

G4ad インスタンスは AMD Radeon Pro V520 GPU と第 2 世代 AMD EPYC プロセッサを使用し、リモートグラフィックスワークステーションなどのグラフィックスアプリケーション、ゲームストリーミング、および OpenGL、DirectX、Vulkan などの業界標準の API を活用するレンダリングに最適です。最大 4 つの AMD Radeon Pro V520 GPU、64 の vCPU、25 Gbps ネットワーキング、および 2.4 TB のローカル NVMe ベースの SSD ストレージを提供します。

G4dn インスタンスは NVIDIA Tesla GPU を使用して、汎用 GPU コンピューティング用のコスト効率とパフォーマンスに優れたプラットフォームを CUDA を通じて提供するか、グラフィックアプリケーションを備えた機械学習フレームワークを DirectX または OpenGL を通じて提供します。このようなインスタンスは、高帯域幅ネットワーキング、強力な半精度浮動小数点機能、単精度浮動小数点機能、INT8 精度、および INT4 精度を提供します。各 GPU は 16 GiB の GDDR6 メモリを備えているため、G4dn インスタンスは機械学習推論、動画トランスコード、グラフィックアプリケーション (リモートグラフィックワークステーションやクラウド内のゲームストリーミングなど) に最適です。

詳細については、「Amazon EC2 G4 インスタンス」を参照してください。

G4dn インスタンスは、NVIDIA GRID 仮想ワークステーションをサポートしています。詳細については、NVIDIA Marketplace の提供サービスを参照してください。

G3 インスタンス

このインスタンスは NVIDIA Tesla M60 GPU を使用し、DirectX または OpenGL を使用してグラフィックアプリケーション向けに費用対効果の高パフォーマンスのプラットフォームを提供します。また、G3 インスタンスは、最大 4096x2160 の解像度を持つ 4 つのモニターと NVIDIA GRID 仮想アプリケーションのサポートなど、NVIDIA GRID 仮想ワークステーションの機能も提供します。G3 インスタンスは、アプリケーションの例としては、3D ビジュアライゼーション、グラフィックを多用するリモートワークステーション、3D レンダリング、動画エンコード、仮想リアリティやそのほかの大規模なパラレル処理を必要とするサーバー側のグラフィックワークロードなどのアプリケーションに最適です。

詳細については、「Amazon EC2 G3 インスタンス」を参照してください。

G3 インスタンスは、NVIDIA GRID 仮想ワークステーションと NVIDIA GRID 仮想アプリケーションをサポートします。これらの機能のいずれかを有効にするには、「NVIDIA GRID 仮想アプリケーションの有効化」を参照してください。

G2 インスタンス

このインスタンスは NVIDIA GRID K520 GPU を使用し、DirectX または OpenGL を使用してグラフィックアプリケーション向けに費用対効果の高パフォーマンスのプラットフォームを提供します。NVIDIA GRID GPU は、NVIDIA の高速キャプチャおよびエンコード API オペレーションもサポートします。アプリケーションのサンプルには、動画作成サービス、3D 仮想化、グラフィックを多用したストリーミングアプリケーションなどのサーバー側のグラフィックワークロードが含まれています。

P5 インスタンス

P5 インスタンスは、640 GB の高帯域幅 GPU メモリを搭載した 8 つの NVIDIA H100 GPU を提供します。これらは第 3 世代の AMD EPYC プロセッサを搭載し、2 TB のシステムメモリ、30 TB のローカル NVMe インスタンスストレージ、3,200 Gps の集約ネットワーク帯域幅、および GPUDirect RDMA サポートを提供します。P5 インスタンスは Amazon EC2 UltraCluster テクノロジーもサポートしているため、EFA を使用してレイテンシーを低減し、ネットワークパフォーマンスを向上させることができます。機械学習と HPC のワークロードでは、P5 インスタンスは前世代の GPU インスタンスの最大 6 倍のパフォーマンスを発揮します。

P5 インスタンスは、GPU 対応のさまざまなワークロードを加速でき、大規模な分散機械学習や高性能コンピューティングアプリケーションに適しています。

詳細については、「Amazon EC2 P5 インスタンス」を参照してください。

P4d インスタンス

このインスタンスは、NVIDIA A100 GPU を使用し、機械学習および HPC ワークロード用の高性能プラットフォームを提供します。また、P4d インスタンスは、400 Gbps の集約ネットワーク帯域幅スループットとサポート、Elastic Fabric Adapter (EFA) を提供します。これらは、複数のネットワークカードを提供する最初の EC2 インスタンスです。

詳細については、「Amazon EC2 P4d インスタンス」を参照してください。

P4d インスタンスは NVIDIA NVSwitch GPU 相互接続と NVIDIA GPUDirect RDMA をサポートします。

P4de インスタンスは NVIDIA 80GB-A100s GPU を提供します

P3 インスタンス

このインスタンスは NVIDIA Tesla V100 GPU を使用し、CUDA または OpenCL プログラミングモデルを使用するか、機械学習フレームワークを使用する汎用 GPU コンピューティング用に設計されています。P3 インスタンスは高帯域幅ネットワーキング、強力な半精度、単精度、および倍精度浮動小数点機能、および GPU ごとに最大 32 GiB メモリを提供し、深層学習、数値流体力学、金融工学、耐震解析、分子モデリング、ゲノム解析、レンダリング、その他サーバー側 GPU コンピューティングワークロードに最適です。Tesla V100 GPU はグラフィックモードをサポートしません。

詳細については、「Amazon EC2 P3 インスタンス」を参照してください。

P3 インスタンスは NVIDIA NVLink のピアツーピア転送をサポートします。詳細については、NVIDIA NVLink を参照してください。

P2 インスタンス

P2 インスタンスは NVIDIA Tesla K80 GPU を使用し、CUDA または OpenCL プログラミングモデルを使用する汎用 GPU コンピューティング用に設計されています。P2 インスタンスは高帯域幅ネットワーキング、強力な単精度および倍精度浮動小数点機能、および GPU ごとに 12 GiB メモリを提供し、ディープラーニング、グラフデータベース、高パフォーマンスデータベース、数値流体力学、金融工学、耐震解析、分子モデリング、ゲノム解析、レンダリング、その他サーバー側 GPU コンピューティングワークロードに最適です。

P2 インスタンスは NVIDIA GPUDirect のピアツーピア転送をサポートします。詳細については、NVIDIA GPUDirect を参照してください。

AWS Trainium を含むインスタンス

AWS Trainium を搭載した Amazon EC2 Trn1 および Trn1n インスタンスは、高性能で費用対効果の高い深層学習トレーニングを目的として構築されています。Trn1 および Trn1n インスタンスを使用すると、音声認識、推奨、不正検出、イメージや動画の分類など、幅広いアプリケーションで使用される自然言語処理、コンピュータビジョン、推奨モデルをトレーニングできます。PyTorch や TensorFlow などのよく使用される ML フレームワークで、既存のワークフローを使用できます。AWSNeuron SDK はこれらのフレームワークとシームレスに統合されるため、コードを数行変更するだけで開始できます。

詳細については、「Amazon EC2 Trn1 インスタンス」を参照してください。

AWS Inferentia を持つインスタンス

これらのインスタンスは、高性能で低レイテンシーの機械学習推論を提供する Amazon のカスタム AI/ML チップである AWS Inferentia を使用して機械学習を高速化するように設計されています。これらのインスタンスは、自然言語処理、オブジェクトの検出と分類、コンテンツのパーソナライズとフィルタリング、音声認識などのアプリケーション向け深層学習 (DL) モデルをデプロイするために最適化されています。

使用を開始するには、さまざまな方法があります。

- 機械学習モデルの使用を開始する最も簡単な方法であり、フルマネージド型のサービスである SageMaker を使用します。詳細については、「Amazon SageMaker 開発者ガイド」の「SageMaker の使用開始」を参照してください。
- 深層学習 AMI を使用して Inf1 または Inf2 インスタンスを起動します。詳細については、AWS デベロッパーガイドの 「DLAMI を使用した AWS Deep Learning AMI Inferentia」を参照してください。
- 独自の AMI を使用して Inf1 または Inf2 インスタンスを起動し、AWS Neuron SDK をインストールします。これにより、AWS Inferentia の深層学習モデルをコンパイル、実行、プロファイリングできます。
- Inf1 または Inf2 インスタンスと Amazon ECS 最適化 AMI を使用してコンテナインスタンスを起動します。詳細については、Amazon Elastic Container Service Developer Guideの「Amazon Linux 2 (Inferentia) AMI」を参照してください。
- Inf1 インスタンスを実行するノードを持つ Amazon EKS クラスターを作成します。詳細については、Amazon EKS ユーザーガイドの「Inferentia のサポート」を参照してください。

詳細については、AWS での Machine Learningをご参照ください。

Inf1 インスタンス

Inf1 インスタンスは、AWS Inferentia 機械学習推論チップを使用します。Inferentia は、コスト効率に優れた、低レイテンシーの推論性能をあらゆる規模で実現するために開発されました。

詳細については、「Amazon EC2 Inf1 インスタンス」を参照してください。

Inf2 インスタンス

Inf2 インスタンスは、AWS Inferentia2 機械学習推論チップを使用します。これらの第 2 世代インスタンスでは、Inf1 インスタンスと比較して推論単価が最大 25% 向上し、同等の Amazon EC2 インス

タンスよりも推論単価が最大 70% 向上します。これらのインスタンスは、深層学習モデルを使用する幅広いワークロードに最適です。

詳細については、「Amazon EC2 Inf2 インスタンス」を参照してください。

Habana アクセラレーターを使用したインスタンス

これらのインスタンスは、深層学習モデル (DL) のトレーニングワークロードを高速化するように設計されています。インテル社、Habana Labs のアクセラレーターを使用しています。これらのインスタンスは、画像認識、オブジェクトの検出と分類、レコメンデーションシステムなどのアプリケーションの DL モデルに最適化されています。

詳細については、AWS での Machine Learningをご参照ください。

DL1 インスタンス

DL1 インスタンスは、Habana Gaudi アクセラレーターを使用しています。最大 400 Gbps の集約ネットワーク帯域幅と、アクセラレーターあたり 32 GB の高帯域幅メモリ (HBM) を備えています。DL1 インスタンスは、深層学習モデルのトレーニングに高いパフォーマンスとコスト効率を提供するように設計されています。

使用を開始するには、さまざまな方法があります。

- Habana Deep Learning AMI を使用して DL1 インスタンスを起動します。
- 独自の AMI を使用して DL1 インスタンスを起動し、Habana ドライバーと Habana SynapseAI SDK をインストールします。
- DL1 インスタンスと Amazon ECS 最適化 AMI を使用してコンテナインスタンスを起動します。
- DL1 インスタンスを実行するノードを持つ Amazon EKS クラスターを作成します。

詳細については、「Amazon EC2 DL1 インスタンス」を参照してください。

Qualcomm アクセラレータを使用したインスタンス

DL2q

DL2q インスタンスは、第 7 世代の Qualcomm エッジ AI コアを搭載した Qualcomm AI100 推論アクセラレータを使用します。深層学習 (DL) ワークロードをコスト効率よくクラウドにデプロイしたり、Qualcomm のエッジデバイスにデプロイされる DL ワークロードのパフォーマンスと精度を検証したりするために使用できます。

DL2q インスタンスは、8 つの Qualcomm AI100 アクセラレータ、デュアル Intel Cascade Lake CPU で実現される 96 個の vCPU、768 GB のシステムメモリ、100 Gbps のネットワーク帯域幅により、最大 1.4 PFLOPS の機械学習パフォーマンスをサポートします。各 Qualcomm AI100 アクセラレータは、最大 175 TFLOP の FP16 パフォーマンスと 16 GB のアクセラレータメモリを備えています。

DL2q インスタンスは、スマートフォン、自動車、ロボット、拡張現実ヘッドセットにデプロイする前に、エッジ AI ワークロードを検証するのに最適です。また、コンテンツ生成、画像分析、テキスト要約、仮想アシスタントなど、一般的な DL アプリケーションを実行するためのクラウド推論もサポートしています。

DL2q インスタンスの使用を開始するには、Qualcomm のアプリケーションとプラットフォームの Software Development Kit (SDK) にあらかじめパッケージされている AWS 深層学習 AMI (DLAMI) と、PyTorch や TensorFlow などの一般的な機械学習フレームワークを使用することをお勧めします。

詳細については、「Amazon EC2 DL2q インスタンス」を参照してください。

ビデオトランスコードインスタンス

これらのインスタンスは、ライブブロードキャスト、ビデオ会議、ジャストインタイムトランスコードなどのビデオトランスコードのワークロードを高速化するように設計されています。

VT1 インスタンス

VT1 インスタンスは Xilinx Alveo U30 メディアアクセラレーターを備え、ライブビデオトランスコードのワークロード向けに設計されています。これらのインスタンスは、最大 8 つの Xilinx Alveo U30 アクセラレーションカードを提供し、最大 192 GB のシステムメモリ、最大 25 Gbps のネットワーク帯域幅を備えています。VT1 インスタンスは H.264/AVC および H.265/HEVC コーデックを備え、マルチストリーミングビデオトランスコーディング用に最大 4K UHD の解像度をサポートします。

使用を開始するには、さまざまな方法があります。

- AWS Marketplace で Xilinx U30 AMI を使用して VT1 インスタンスを起動します。
- 独自の AMI を使用して VT1 インスタンスを起動し、Xilinx U30 ドライバーと Xilinx ビデオ SDKをインストールします。
- VT1 インスタンスと Amazon ECS 最適化 AMI を使用してコンテナインスタンスを起動します。
- VT1 インスタンスを実行するノードを持つ Amazon EKS クラスターを作成します。

詳細については、「Amazon EC2 VT1 インスタンス」を参照してください。

FPGA インスタンス

FPGA ベースのインスタンスでは、数百万の並列システム論理セルを持つ大きな FPGA にアクセスできます。FPGA ベースの高速コンピューティングインスタンスを使用すると、カスタムハードウェアアクセラレーションを活用することにより、ゲノム解析、財務分析、リアルタイム動画処理、ビッグデータ解析、およびセキュリティワークロードなどのワークロードを高速化できます。Verilog や VHDL などのハードウェア記述言語を使用するか、または OpenCL パラレルコンピューティングフレームワークなどの高レベル言語を使用して、これらの加速度を開発できます。ハードウェアアクセラレーションコードを自身で作成することも、AWS Marketplace からハードウェアアクセラレーションを購入することもできます。

FPGA Developer AMI は、AFI を開発、テスト、および構築するためのツールを提供します。FPGA Developer AMI は、32 GB 以上のシステムメモリを備える任意の EC2 インスタンス (例: C5、M4、R4 インスタンス) で使用できます。

詳細については、「AWS FPGA Hardware Development Kit」のドキュメントを参照してください。

F1 インスタンス

F1 インスタンスは Xilinx UltraScale+ VU9P FPGA を使用し、汎用 CPU に適さないデータフローや高度な並列処理のような計算集約型のアルゴリズムを高速化するように設計されています。F1 インスタンスの各 FPGA には、約 250 万の論理要素と約 6,800 のデジタル信号処理 (DSP) エンジン、ローカルの 64 GiB DDR ECC 保護メモリが含まれ、専用の PCIe Gen3 x16 接続によってインスタンスに接続されています。F1 インスタンスは、ローカルの NVMe SSD ボリュームを提供します。

デベロッパーは FPGA Developer AMI および AWS Hardware Developer Kit を使用して、F1 インスタンスで使用するカスタムハードウェアアクセラレーションを作成できます。FPGA Developer AMI には、クラウド上の FPGA 完全サイクル開発用の開発ツールが含まれます。これらのツールを使用して、デベロッパーは F1 インスタンスの FPGA にロードできる Amazon FPGA Image (AFI) を作成し、共有できます。

詳細については、「Amazon EC2 F1 インスタンス」を参照してください。

ハードウェア仕様

仮想中央処理ユニット (vCPU) は、仮想マシン (VM) に割り当てられた物理 CPU の一部を表します。x86 インスタンスの場合、コアごとに 2 つの vCPU があります。Graviton インスタンスの場合、コアごとに 1 つの vCPU があります。

ハードウェアの仕様については、「Amazon EC2 インスタンスタイプガイド」の「高速コンピューティングインスタンス」を参照してください。

インスタンスのパフォーマンス

インスタンスで最大のパフォーマンスを実現するための GPU 設定の最適化には、さまざまなものがあります。詳細については、GPU 設定の最適化 を参照してください。

EBS 最適化インスタンスは、インスタンスからの Amazon EBS I/O とその他のネットワークトラフィックとの競合を排除することによって、EBS ボリュームの安定した高パフォーマンスを実現できます。一部の高速コンピューティングインスタンスは、追加料金なしでデフォルトで EBS 最適化されています。詳細については、Amazon EBS 最適化インスタンスを使用する を参照してください。

一部の高速コンピューティングインスタンスタイプでは、Linux でプロセッサの C ステートと P ステートを制御できます。C ステートは非アクティブ時のコアのスリープレベルを制御し、P ステートは希望するコアからのパフォーマンス (CPU 周波数) を制御します。詳細については、EC2 インスタンスのプロセッサのステート制御 を参照してください。

ネットワークパフォーマンス

サポートされているインスタンスタイプで拡張ネットワーキングを有効にすると、レイテンシーとネットワークジッターを低減し、パケット毎秒 (PPS) のパフォーマンスを高めることができます。ほとんどのアプリケーションでは、高いレベルのネットワークパフォーマンスが一貫して必要なわけではありませんが、データの送受信時にアクセスする帯域幅を増やすことでメリットを得られます。詳細については、「Linux での拡張ネットワーキング」を参照してください。

ネットワークの仕様については、「Amazon EC2 インスタンスタイプガイド」の「高速コンピューティングインスタンス」を参照してください。

Amazon EBS I/O パフォーマンス

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、Amazon EBS ボリュームの最高のパフォーマンスを実現します。

詳細については、「Amazon EBS 最適化インスタンスを使用する」を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスストアボリュームは、インスタンスの存続中のみ使用できます。インスタンスを停止、休止、または終了すると、アプリケーションとそのインスタンスストアボリュームのデータは消去されます。インスタンスストアボリュームの重要なデータは、定期的にバックアップまたはレプリケートすることをお勧めします。詳細については、「Amazon EC2 インスタンスストア」および「SSD インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仕様については、「Amazon EC2 インスタンスタイプガイド」の「高速コンピューティングインスタンス」を参照してください。

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これは、SSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは、利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み操作の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは、複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つには、SSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームには、オーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすには、ボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。

リリースノート

- P5 インスタンスでの最高のパフォーマンスを得るには、次のことを行うことをお勧めします。
  - Linux カーネルバージョン 5.10 以降の AMI を使用します。
  - より深い C ステートを使用しないようにシステムを設定します。詳細については、「より深い C ステートを制限することによる高パフォーマンスと低レイテンシー」を参照してください。
- インスタンスは、HVM AMI を使用して起動する必要があります。
- Nitro System 上に構築されたインスタンスには、次の要件があります。
  - NVMe ドライバーがインストールされている必要があります。
  - Elastic Network Adapter (ENA) ドライバーがインストールされている必要があります。

  以下の Linux AMI はこれらの要件を満たしています。
  - AL2023
  - Amazon Linux 2
  - Amazon Linux AMI 2018.03 以降
  - `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

    > **Note**
    >
    > AWS Graviton ベースのインスタンスタイプには、`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

  - Red Hat Enterprise Linux 7.4 以降
  - SUSE Linux Enterprise Server 12 SP2 以降
  - CentOS 7.4.1708 以降
  - FreeBSD 11.1 以降
  - Debian GNU/Linux 9 以降
- NVIDIA ドライバーがインストールされていない限り、GPU ベースのインスタンスは GPU にアクセスできません。詳細については、Linux インスタンスへの NVIDIA ドライバーのインストール を参照してください。
- ベアメタルインスタンスを起動すると、基盤となるサーバーが起動します。これには、すべてのハードウェアやファームウェアコンポーネントの確認が含まれます。つまり、インスタンスが実行状態になってからネットワーク経由で使用できるようになるまでに 20 分かかることがあります。

- EBS ボリュームまたはセカンダリネットワークインターフェイスを、ベアメタルインスタンスにアタッチ (または、そこからデタッチ) するには、PCIe のネイティブホットプラグがサポートされている必要があります。PCIe のネイティブホットプラグは、Amazon Linux 2 および最新バージョンの Amazon Linux AMI でサポートされています。それ以前のバージョンではサポートされていません。次の Linux カーネル設定オプションを有効にする必要があります。

```
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_PCIEASPM=y
```

- ベアメタルインスタンスでは、I/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。アップストリームの Linux カーネルと最新の Amazon Linux AMI は、このデバイスをサポートしています。また、ベアメタルインスタンスでは、システムが PCI ベースのシリアルデバイスを自動的に使用できるようにする ACPI SPCR テーブルも使用できます。最新の Windows AMI では、自動的に PCI ベースのシリアルデバイスが使用されます。
- リージョンごとに 100 AFI という制限があります。
- リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。詳細については、Amazon EC2 の「よくある質問」の「Amazon EC2 で実行できるインスタンス数の上限は」を参照してください。

P5 インスタンスの使用を開始する

P5 インスタンスは、640 GB の高帯域幅 GPU メモリを搭載した 8 つの NVIDIA H100 GPU を提供します。これらは第 3 世代の AMD EPYC プロセッサを搭載し、2 TB のシステムメモリ、30 TB のローカル NVMe インスタンスストレージ、3,200 Gbps の集約ネットワーク帯域幅、および GPUDirect RDMA サポートを提供します。P5 インスタンスは Amazon EC2 UltraCluster テクノロジーもサポートしているため、EFA を使用してレイテンシーを低減し、ネットワークパフォーマンスを向上させることができます。

次の表に、`p5.48xlarge` 仕様の概要を示します。

vCPUs	システムメモリ	GPU	GPU メモリ	ネットワーク帯域幅	GPUDirect RDMA	GPU ピアツーピア	インスタンスストレージ
192	2 TiB	8 NVIDIA H100 GPU	640 GB HBM3	EFAv2 を使用した 3200 Gbps	サポート	900 GB/秒 NVSw	8 x 3,800 GB NVMe SSD ボリューム

ソフトウェア設定

P5 インスタンスの使用を始める最も簡単な方法は、必要なすべてのソフトウェアが事前設定されている AWS Deep Learning AMI を使用してインスタンスを起動することです。P5 インスタンスで使用するための最新の AWS Deep Learning AMI については、「AWSDeep Learning Base GPU AMI (Ubuntu 20.04)」を参照してください。

P5 インスタンスで使用するカスタム AMI を構築する必要がある場合は、以下の最小ソフトウェアバージョンをインストールすることをお勧めします。

- NVIDIA ドライバー 535.54.03 以降
- CUDA 12.1 以降
- NVIDIA GDRCopy 2.3 以降
- EFA インストーラ 1.24.1 以降
- NCCL 2.18.3 以降
- aws-ofi-nccl プラグイン 1.7.2-aws 以降

また、より深い C ステートを使用しないようにインスタンスを設定することをお勧めします。詳細については、「深い C ステートの制限による高パフォーマンスと低レイテンシー」を参照してください。最新の AWS Deep Learning Base GPU AMI は、より深い C ステートを使用しないように事前設定されています。

Ubuntu 20.04 固有の推奨事項

Ubuntu 20.04 に関する以下の推奨事項は、起動時に想定外のインターフェイス名が付けられるのを防ぐのに役立ちます。

- 以下のコマンドを実行して、systemd 245.4-4ubuntu3.19 以降かを確認してください。

```
systemd --version
```

- GRUB を設定したことを確認します。
  - /etc/default/grub 設定ファイルをテキストエディタで開きます。
  - GRUB_CMDLINE_LINUX_DEFAULT エントリを編集して net.naming-scheme=v247 を含めます。
  - sudo update-grub を実行してインスタンスを再起動します。

ネットワークと EFA 設定

P5 インスタンスは、複数の EFA インターフェイスを使用して 3200 Gbps のネットワーク帯域幅を提供します。P5 インスタンスは 32 枚のネットワークカードをサポートします。ネットワークカードごとに 1 つの EFA ネットワークインターフェイスを定義することをお勧めします。起動時にこれらのインターフェイスを設定するには、以下の設定をお勧めします。

- ネットワークインターフェイス 0 の場合、デバイスインデックス 0 を指定する
- 31 を介したネットワークインターフェイス 1 の場合、デバイスインデックス 1 を指定する

P5 インスタンスを EFA 用に設定する方法の詳細については、「P5 インスタンスと EFA の使用を開始する」を参照してください。

Linux インスタンスへの NVIDIA ドライバーのインストール

NVIDIA GPU がアタッチされたインスタンス (P3 インスタンスや G4dn インスタンスなど) には、適切な NVIDIA ドライバーがインストールされている必要があります。インスタンスタイプに応じて、公開された NVIDIA ドライバーまたは AWS カスタマーのみが使用できる Amazon S3 のドライバーをダウンロードするか、ドライバーが事前インストールされた AMI を使用します。

AMD GPU がアタッチされた Linux インスタンス (G4ad インスタンスなど) に AMD ドライバーをインストールするには、代わりに「AMD ドライバーのインストール」を参照してください。Windows インスタンスに NVIDIA ドライバーをインストールするには、「Windows インスタンスへの NVIDIA ドライバーのインストール」を参照してください。

目次

NVIDIA ドライバーの種類

GPU ベースのインスタンスで使用できる主な種類の NVIDIA ドライバーを次に示します。

Tesla ドライバー

これらのドライバーは主に、機械学習用の並列浮動小数点計算、ハイパフォーマンスコンピューティングアプリケーション用の高速フーリエ変換などの計算タスクに GPU を使用するコンピューティングワークロードを対象としています。

GRID ドライバー

これらのドライバーは、3D モデルや高解像度動画などのコンテンツをレンダリングするプロフェッショナルな視覚化アプリケーションに最適なパフォーマンスを提供することが認定されています。GRID ドライバーを構成すると、2 つのモードをサポートできます。Quadro Virtual Workstation は、GPU あたり 4 台の 4K ディスプレイへのアクセスを提供します。GRID vApps は、RDSH アプリのホスティング機能を提供します。

ゲームドライバー

これらのドライバーはゲーム用に最適化されており、パフォーマンスを向上させるために頻繁に更新されます。これらは GPU あたり単一の 4K ディスプレイをサポートします。

NVIDIA コントロールパネル

NVIDIA コントロールパネルは、GRID およびゲームドライバーでサポートされています。Tesla ドライバーではサポートされていません。

Tesla、GRID、およびゲームとライバーに対してサポートされている API

- OpenCL、OpenGL、Vulkan

- NVIDIA CUDA および関連ライブラリ (cuDNN、TensorRT、nvJPEG、cuBLAS など)
- 動画エンコード用の NVENC と動画デコード用の NVDEC

インスタンスタイプ別の使用可能なドライバー

次の表は、GPU インスタンスタイプごとにサポートされている NVIDIA ドライバーをまとめたものです。

インスタンスタイプ	Tesla ドライバー	GRID ドライバー	ゲームドライバー
G2	はい	いいえ	いいえ
G3	はい	はい	いいえ
G4dn	はい	はい	はい
G5	はい	はい	はい
G5g	はい [1]	いいえ	いいえ
P2	はい	いいえ	いいえ
P3	はい	いいえ	いいえ
P4d	はい	いいえ	いいえ
P4de	はい	いいえ	いいえ

[1] この Tesla ドライバーは、ARM64 プラットフォーム固有の最適化されたグラフィックスアプリケーションもサポートしています。

[2] Marketplace AMI のみを使用

インストールオプション

次のいずれかのオプションを使用して、GPU インスタンスに必要な NVIDIA ドライバーを取得します。

Options

- オプション 1: NVIDIA ドライバーがインストールされた AMI

- オプション 2: パブリック NVIDIA ドライバー
- オプション 3: GRID ドライバー (G5、G4dn、および G3 インスタンス)
- オプション 4: NVIDIA ゲームドライバー (G5 および G4dn インスタンス)

オプション 1: NVIDIA ドライバーがインストールされた AMI

AWS と NVIDIA では、NVIDIA ドライバーがインストールされた、それぞれ異なる Amazon マシンイメージ (AMI) を提供しています。

- Tesla ドライバーを使用した Marketplace 製品
- GRID ドライバーを使用した Marketplace 製品
- ゲームドライバーを使用した Marketplace 製品

これらの AMI のいずれかを使用してインストールされたドライバーのバージョンを更新するには、バージョンの競合を避けるために、インスタンスから NVIDIA パッケージをアンインストールする必要があります。次のコマンドを使用して、NVIDIA パッケージをアンインストールします。

```
[ec2-user ~]$ sudo yum erase nvidia cuda
```

CUDA ツールキットパッケージは、NVIDIA ドライバーに依存します。NVIDIA パッケージをアンインストールすると、CUDA ツールキットが消去されます。NVIDIA ドライバーをインストールした後に、CUDA ツールキットを再インストールする必要があります。

オプション 2: パブリック NVIDIA ドライバー

AWS が提供するオプションには、ドライバーに必要なライセンスが付属しています。または、パブリックドライバーをインストールし、自分のライセンスを使用することもできます。パブリックドライバーをインストールするには、ここで説明するように NVIDIA サイトからドライバーをダウンロードします。

または、パブリックドライバーの代わりに AWS が提供するオプションを使用することもできます。P3 インスタンスで GRID ドライバーを使用するには、オプション 1 の説明に従って AWS Marketplace AMI を使用します。G5、G4dn、または G3 インスタンスで GRID ドライバーを使用するには、オプション 1 の説明に従って AWS Marketplace AMI を使用するか、オプション 3 の説明に従って AWS が提供する NVIDIA ドライバーをインストールします。

パブリック NVIDIA ドライバーをダウンロードするには

Linux もしくは インスタンスにログインし、http://www.nvidia.com/Download/Find.aspx から、使用するインスタンスタイプに適した 64 ビット NVIDIA ドライバーをダウンロードします。[製品タイプ]、[製品シリーズ]、[製品] の順にクリックし、次の表に示すオプションを使用します。

インスタンス	製品タイプ	製品シリーズ	製品
G2	GRID	GRID シリーズ	GRID K520
G3	Tesla	M-Class	M60
G4dn	Tesla	T シリーズ	T4
G5 [1]	Tesla	A シリーズ	A10
G5g [2]	Tesla	T シリーズ	NVIDIA T4G
P2	Tesla	K シリーズ	K80
P3	Tesla	V シリーズ	V100
P4d	Tesla	A シリーズ	A100
P4de	Tesla	A シリーズ	A100
P5 [3]	Tesla	H シリーズ	H100

[1] G5 インスタンスには、ドライバーバージョン 470.00 以降が必要です。

[2] G5g インスタンスには、ドライバーバージョン 470.82.01 以降が必要です。オペレーティングシステムは Linux aarch64 です。

[3] P5 インスタンスには、ドライバーバージョン 530 以降が必要です。

Linux で NVIDIA ドライバーをインストールするには

ドライバーのインストールと設定の詳細については、NVIDIA Driver Installation Quickstart Guide を参照してください。

オプション 3: GRID ドライバー (G5、G4dn、および G3 インスタンス)

これらのダウンロードは、AWS カスタマーのみが利用できます。ダウンロードすることにより、NVIDIA GRID Cloud エンドユーザーライセンス契約 (EULA) で言及されている AWS ソリューションの要件を遵守するため、お客様は、AMI の開発目的にのみダウンロードしたソフトウェアを NVIDIA A10G、NVIDIA Tesla T4、NVIDIA Tesla M60 のいずれかのハードウェアとともに使用することに同意します。このソフトウェアをインストールすることは、NVIDIA GRID Cloud End User License Agreement の規約の遵守に同意したものと見なされます。ご使用のオペレーティングシステムの NVIDIA GRID ドライバーのバージョンについては、NVIDIA ウェブサイトの「NVIDIA® 仮想 GPU (vGPU) ソフトウェアマニュアル」を参照してください。

考慮事項

- G5 インスタンスには GRID 13.1 以降 (または GRID 12.4 以降) が必要です。
- G3 インスタンスで GRID ライセンスを機能させるには、AWS が提供する DNS 解決が必要です。
- IMDSv2 は、NVIDIA ドライバーのバージョン 14.0 以降でのみサポートされています。

Amazon Linux および Amazon Linux 2

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. Linux インスタンスに接続します。
2. Linux インスタンスに AWS CLI をインストールし、デフォルトの認証情報を設定します。詳細については、AWS CLI ユーザーガイドの「AWS Command Line Interface のインストール」を参照してください。

   > **Important**
   >
   > ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「AWS マネージドポリシー: AmazonS3ReadOnlyAccess」を参照してください。

3. gcc および make をインストールします (まだインストールされていない場合)。

   ```
   [ec2-user ~]$ sudo yum install gcc make
   ```

4. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

5. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

6. 再起動後にインスタンスに再接続します。
7. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y gcc kernel-devel-$(uname -r)
```

8. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

9. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

10. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

> **Note**
>
> Amazon Linux 2 をカーネルバージョン 5.10 で使用している場合は、次のコマンドを使用して GRID ドライバーをインストールしてください。
>
> ```
> [ec2-user ~]$ sudo CC=/usr/bin/gcc10-cc ./NVIDIA-Linux-x86_64*.run
> ```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

11. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

12. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

14. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル NICE DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、「NVIDIA GRID 仮想アプリケーションの有効化」の GRID 仮想アプリケーションのアクティブ化手順を完了します。

CentOS 7 と Red Hat Enterprise Linux 7

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y gcc kernel-devel-$(uname -r)
```

6. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

```
[ec2-user ~]$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

   c. Grub 設定を再構築します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

7. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

   GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

8. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

9. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

11. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

12. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

13. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル NICE DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアク

ティブ化するには、「NVIDIA GRID 仮想アプリケーションの有効化」の GRID 仮想アプリケーションのアクティブ化手順を完了します。

c. GUI デスクトップ/ワークステーションパッケージをインストールします。

```
[ec2-user ~]$ sudo yum groupinstall -y "Server with GUI"
```

CentOS Stream 8 と Red Hat Enterprise Linux 8

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y make gcc elfutils-libelf-devel libglvnd-devel
 kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

7. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
[ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
```

8. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

9. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

10. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

11. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

12. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル NICE DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、「NVIDIA GRID 仮想アプリケーションの有効化」の GRID 仮想アプリケーションのアクティブ化手順を完了します。

    c. GUI ワークステーションパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf groupinstall -y workstation
```

Rocky Linux 8

Linux インスタンスに NVIDIA GRID ドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

   ```
   [ec2-user ~]$ sudo yum update -y
   ```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

   ```
   [ec2-user ~]$ sudo reboot
   ```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

   ```
   [ec2-user ~]$ sudo dnf install -y make gcc elfutils-libelf-devel libglvnd-devel
    kernel-devel-$(uname -r)
   ```

6. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

   ```
   [ec2-user ~]$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
   ```

   GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

   ```
   [ec2-user ~]$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
   ```

7. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

   ```
   [ec2-user ~]$ chmod +x NVIDIA-Linux-x86_64*.run
   ```

8. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
[ec2-user ~]$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

9. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
[ec2-user ~]$ nvidia-smi -q | head
```

10. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /
etc/modprobe.d/nvidia.conf
```

11. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

12. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル NICE DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、「NVIDIA GRID 仮想アプリケーションの有効化」の GRID 仮想アプリケーションのアクティブ化手順を完了します。

Ubuntu と Debia

インスタンスに NVIDIA GRID ドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
$ sudo apt-get update -y
```

3. (Ubuntu) 最新バージョンを受け取るには、`linux-aws` パッケージをアップグレードします。

```
$ sudo apt-get upgrade -y linux-aws
```

(Debian) 最新バージョンを受け取るには、パッケージをアップグレードします。

```
$ sudo apt-get upgrade -y
```

4. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
$ sudo reboot
```

5. 再起動後にインスタンスに再接続します。
6. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
$ sudo apt-get install -y gcc make linux-headers-$(uname -r)
```

7. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに `/etc/modprobe.d/blacklist.conf` を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

b. /etc/default/grub ファイルを編集して、次の行を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

c. Grub 設定を再構築します。

```
$ sudo update-grub
```

8. 次のコマンドを使用して、GRID ドライバーインストールユーティリティをダウンロードします。

```
$ aws s3 cp --recursive s3://ec2-linux-nvidia-drivers/latest/ .
```

GRID ドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
$ aws s3 ls --recursive s3://ec2-linux-nvidia-drivers/
```

9. 次のコマンドを使用して、ドライバーのインストールを実行するアクセス権限を追加します。

```
$ chmod +x NVIDIA-Linux-x86_64*.run
```

10. 次のようにセルフインストールスクリプトを実行して、ダウンロードした GRID ドライバーをインストールします。次に例を示します。

```
$ sudo /bin/sh ./NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

11. ドライバーが機能していることを確認します。次のコマンドのレスポンスに、インストールされた NVIDIA ドライバーバージョンおよび GPU に関する詳細が表示されます。

```
$ nvidia-smi -q | head
```

12. G4dn、G5 または G5g インスタンスで NVIDIA vGPU ソフトウェアバージョン 14.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/
modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
$ sudo reboot
```

14. (オプション) ユースケースによっては、以下のオプションのステップを実行できます。この機能が必要ない場合は、以下のステップを実行しないでください。

    a. 最大 4K の解像度のディスプレイを 4 台活用するには、高性能ディスプレイプロトコル NICE DCV を設定します。

    b. NVIDIA Quadro 仮想ワークステーションモードはデフォルトで有効になっています。RDSH アプリケーションホスティング機能用に GRID 仮想アプリケーションをアクティブ化するには、「NVIDIA GRID 仮想アプリケーションの有効化」の GRID 仮想アプリケーションのアクティブ化手順を完了します。

    c. GUI デスクトップ/ワークステーションパッケージをインストールします。

```
$ sudo apt-get install -y lightdm ubuntu-desktop
```

オプション 4: NVIDIA ゲームドライバー (G5 および G4dn インスタンス)

これらのドライバーは、AWS カスタマーのみが利用できます。これらをダウンロードすることで、ダウンロードしたソフトウェアは、NVIDIA A10G および NVIDIA Tesla T4 ハードウェアで、AMIs の開発目的のみに使用することに同意したことになります。このソフトウェアをインストールすることは、NVIDIA GRID Cloud End User License Agreement の規約の遵守に同意したものと見なされます。

考慮事項

- G3 インスタンスで GRID ライセンスを機能させるには、AWS が提供する DNS 解決が必要です。
- IMDSv2 は、NVIDIA ドライバーのバージョン 495.x 以降でのみサポートされています。

Amazon Linux および Amazon Linux 2

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。
2. Linux インスタンスに AWS CLI をインストールし、デフォルトの認証情報を設定します。詳細については、AWS CLI ユーザーガイドの「AWS Command Line Interface のインストール」を参照してください。

   > **Important**
   >
   > ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「AWS マネージドポリシー: AmazonS3ReadOnlyAccess」を参照してください。

3. gcc および make をインストールします (まだインストールされていない場合)。

   ```
   [ec2-user ~]$ sudo yum install gcc make
   ```

4. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

   ```
   [ec2-user ~]$ sudo yum update -y
   ```

5. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

   ```
   [ec2-user ~]$ sudo reboot
   ```

6. 再起動後にインスタンスに再接続します。
7. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

   ```
   [ec2-user ~]$ sudo yum install -y gcc kernel-devel-$(uname -r)
   ```

8. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

   ```
   [ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
   ```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

9. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip latest-driver-name.zip -d nvidia-drivers
```

10. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/NVIDIA-Linux-x86_64*-grid.run
```

11. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo ./nvidia-drivers/NVIDIA-Linux-x86_64*.run
```

> **Note**
>
> Amazon Linux 2 をカーネルバージョン 5.10 で使用している場合は、次のコマンドを使用して NVIDIA ゲームドライバーをインストールします。
>
> ```
> [ec2-user ~]$ sudo CC=/usr/bin/gcc10-cc ./NVIDIA-Linux-x86_64*.run
> ```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

12. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

13. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

    - バージョン 460.39 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2023_9_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

14. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/modprobe.d/nvidia.conf
```

15. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

16. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル、NICE DCV を設定します。

CentOS 7 と Red Hat Enterprise Linux 7

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y unzip gcc kernel-devel-$(uname -r)
```

6. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。

   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

```
[ec2-user ~]$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

   c. Grub 設定を再構築します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

7. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

   ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

8. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip vGPUSW-*vGaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

9. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/Linux/NVIDIA-Linux-x86_64*-grid.run
```

10. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo ./nvidia-drivers/Linux/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

11. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

12. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 460.39 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2023_9_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

13. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/modprobe.d/nvidia.conf
```

14. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

15. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル、NICE DCV を設定します。この機能が必要ない場合は、このステップを実行しないでください。

CentOS Stream 8 と Red Hat Enterprise Linux 8

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。
5. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y unzip gcc kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

7. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip vGPUSW-*vGaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

8. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/Linux/NVIDIA-Linux-x86_64*-grid.run
```

9. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo ./nvidia-drivers/Linux/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

11. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 460.39 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2023_9_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

12. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

14. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル、NICE DCV を設定します。

Rocky Linux 8

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
[ec2-user ~]$ sudo yum update -y
```

3. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
[ec2-user ~]$ sudo reboot
```

4. 再起動後にインスタンスに再接続します。

5. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
[ec2-user ~]$ sudo dnf install -y unzip gcc make elfutils-libelf-devel libglvnd-devel kernel-devel-$(uname -r)
```

6. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
[ec2-user ~]$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

7. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
[ec2-user ~]$ unzip vGPUSW-*vGaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

8. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
[ec2-user ~]$ chmod +x nvidia-drivers/Linux/NVIDIA-Linux-x86_64*-grid.run
```

9. 次のコマンドを使用してインストーラを実行します。

```
[ec2-user ~]$ sudo ./nvidia-drivers/Linux/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

10. 次のコマンドを使用して必要な設定ファイルを作成します。

```
[ec2-user ~]$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

11. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 460.39 以降:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2023_9_22.cert"
```

- バージョン 440.68 から 445.48:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
[ec2-user ~]$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

12. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
[ec2-user ~]$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
[ec2-user ~]$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/modprobe.d/nvidia.conf
```

13. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

14. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル、NICE DCV を設定します。

Ubuntu と Debia

インスタンスに NVIDIA ゲームドライバーをインストールするには

1. Linux インスタンスに接続します。gcc および make をインストールします (まだインストールされていない場合)。
2. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

```
$ sudo apt-get update -y
```

3. 最新バージョンにするには、linux-aws パッケージにアップグレードします。

```
$ sudo apt-get upgrade -y linux-aws
```

4. インスタンスを再起動して、最新のカーネルバージョンを読み込みます。

```
$ sudo reboot
```

5. 再起動後にインスタンスに再接続します。
6. 現在実行しているカーネルのバージョン用の gcc コンパイラおよびカーネルヘッダーパッケージをインストールします。

```
$ sudo apt-get install -y unzip gcc make linux-headers-$(uname -r)
```

7. NVIDIA グラフィックスカード用の nouveau オープンソースドライバーを無効にします。
   a. nouveau ブラックリストファイルに /etc/modprobe.d/blacklist.conf を追加します。次のコードブロックをコピーして、ターミナルに貼り付けます。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   b. /etc/default/grub ファイルを編集して、次の行を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

   c. Grub 設定を再構築します。

```
$ sudo update-grub
```

8. 次のコマンドを使用して、ゲームドライバーインストールユーティリティをダウンロードします。

```
$ aws s3 cp --recursive s3://nvidia-gaming/linux/latest/ .
```

ゲームドライバーの複数のバージョンがこのバケットに保存されます。使用可能なバージョンをすべて表示するには、次のコマンドを使用します。

```
$ aws s3 ls --recursive s3://nvidia-gaming/linux/
```

9. ダウンロードした .zip アーカイブから、ゲームドライバーのインストールユーティリティを抽出します。

```
$ unzip vGPUSW-*vGaming-Linux-Guest-Drivers.zip -d nvidia-drivers
```

10. 次のコマンドを使用して、ドライバーのインストールユーティリティを実行するためのアクセス許可を追加します。

```
$ chmod +x nvidia-drivers/Linux/NVIDIA-Linux-x86_64*-grid.run
```

11. 次のコマンドを使用してインストーラを実行します。

```
$ sudo ./nvidia-drivers/Linux/NVIDIA-Linux-x86_64*.run
```

プロンプトが表示されたら、ライセンス契約を受諾し、必要に応じてインストールオプションを指定します (デフォルトのオプションを使用できます)。

12. 次のコマンドを使用して必要な設定ファイルを作成します。

```
$ cat << EOF | sudo tee -a /etc/nvidia/gridd.conf
vGamingMarketplace=2
EOF
```

13. 次のコマンドを使用して認証ファイルをダウンロードし、名前を変更します。

- バージョン 460.39 以降:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCertLinux_2023_9_22.cert"
```

- バージョン 440.68 から 445.48:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2020_04.cert"
```

- それより前のバージョン:

```
$ sudo curl -o /etc/nvidia/GridSwCert.txt "https://nvidia-
gaming.s3.amazonaws.com/GridSwCert-Archive/GridSwCert-Linux_2019_09.cert"
```

14. G4dn、G5 または G5g インスタンスで NVIDIA ドライバーバージョン 510.x 以降を使用している場合は、次のコマンドで GSP を無効にします。これが必要な理由の詳細については、「NVIDIA のドキュメント」をご覧ください。

```
$ sudo touch /etc/modprobe.d/nvidia.conf
```

```
$ echo "options nvidia NVreg_EnableGpuFirmware=0" | sudo tee --append /etc/
modprobe.d/nvidia.conf
```

15. インスタンスを再起動します。

```
$ sudo reboot
```

16. (オプション) 最大 4K 解像度の 1 台のディスプレイを活用するには、高性能ディスプレイプロトコル、NICE DCV を設定します。この機能が必要ない場合は、このステップを実行しないでください。

CUDA の追加バージョンのインストール

インスタンスに NVIDIA グラフィックスドライバーをインストールした後、グラフィックスドライバーにバンドルされているバージョン以外の CUDA をインストールできます。以下の手順では、インスタンスで CUDA の複数のバージョンを設定する方法を示しています。

CUDA ツールキットをインストールするには

1. Linux インスタンスに接続します。
2. NVIDIA ウェブサイトを開き、必要な CUDA のバージョンを選択します。
3. インスタンスのオペレーティングシステムのアーキテクチャ、ディストリビューション、バージョンを選択します。[Installer Type (インストーラタイプ)] で、[runfile (local) (runfile (ローカル))] を選択します。

4. 手順に従ってインストールスクリプトをダウンロードします。
5. 以下のコマンドを使用してダウンロードしたインストールスクリプトに対する実行アクセス許可を追加します。

```
[ec2-user ~]$ chmod +x downloaded_installer_file
```

6. 以下のようにインストールスクリプトを実行して、CUDA ツールキットをインストールし、CUDA のバージョン番号をツールキットのパスに追加します。

```
[ec2-user ~]$ sudo sh downloaded_installer_file --silent --override --toolkit --
samples --toolkitpath=/usr/local/cuda-version --samplespath=/usr/local/cuda --no-
opengl-libs
```

7. (オプション) CUDA のデフォルトバージョンを以下のように設定します。

```
[ec2-user ~]$ sudo ln -s /usr/local/cuda-version /usr/local/cuda
```

Linux インスタンスへの AMD ドライバーのインストール

AMD GPU がアタッチされたインスタンス (G4ad インスタンスなど) には、適切な AMD ドライバーがインストールされている必要があります。要件に応じて、ドライバーをプリインストールした AMI を使用するか、Amazon S3 からドライバーをダウンロードできます。

NVIDIA GPU がアタッチされたインスタンス (G4dn インスタンスなど) に NVIDIA ドライバーをインストールするには、代わりに「NVIDIA ドライバーのインストール」を参照してください。Windows インスタンスに AMD ドライバーをインストールするには、「Windows インスタンスへの AMD ドライバーのインストール」を参照してください。

目次

エンタープライズドライバー向け AMD Radeon Pro ソフトウェア

エンタープライズドライバー向け AMD Radeon Pro ソフトウェアは、プロフェッショナルグレードのグラフィックスのユースケースをサポートするために構築されています。ドライバーを使用して、GPU ごとに 2 つの 4K ディスプレイでインスタンスを設定できます。

サポートされている API

- OpenGL、OpenCL
- Vulkan
- AMD Advanced Media Framework
- Video Acceleration API

AMDドライバーをインストールした AMI

AWS では、AMD ドライバーがインストールされた、それぞれ異なる Amazon マシンイメージ (AMI) を提供しています。AMD ドライバーで Marketplace 製品を開きます

AMD ドライバーのダウンロード

AMD ドライバーがインストールされた AMI を使用していない場合は、AMD ドライバーをダウンロードしてインスタンスにインストールできます。 AMD ドライバーは、カーネルバージョン 4.14 の Amazon Linux 2 でのみサポートされます。

> **Note**
>
> AMD ドライバーバージョン amdgpu-pro-20.20-1184451 以降のドライバーリリースには、カーネルバージョン 5.15 以降が必要です。

これらのダウンロードは、AWS カスタマーのみが利用できます。ダウンロードすることで、AMD Radeon Pro V520 ハードウェアの使用において、ダウンロードしたソフトウェアを AMIs の開発用のみで使用することに同意したことになります。このソフトウェアをインストールすることは、AMD Software End User License Agreement の規約の遵守に同意したものと見なされます。

Linux インスタンスに AMD ドライバーをインストールするには

1. Linux インスタンスに接続します。

2. Linux インスタンスに AWS CLI をインストールし、デフォルトの認証情報を設定します。詳細については、AWS CLI ユーザーガイドの「AWS Command Line Interface のインストール」を参照してください。

   > **Important**
   >
   > ユーザーまたはロールは、[AmazonS3ReadOnlyAccess] ポリシーを含む許可を持っている必要があります。詳細については、「Amazon Simple Storage Service ユーザーガイド」の「AWS マネージドポリシー: AmazonS3ReadOnlyAccess」を参照してください。

3. gcc および make をインストールします (まだインストールされていない場合)。

   ```
   $ sudo yum install gcc make
   ```

4. パッケージのキャッシュを更新し、インスタンスのためにパッケージを更新します。

   - 複数 Amazon Linux 2:

     ```
     $ sudo amazon-linux-extras install epel -y
     $ sudo yum update -y
     ```

   - Ubuntu 22.04 の場合:

     ```
     $ wget https://repo.radeon.com/.preview/a0e4ef1dffbc95b4abb54e891f265e61/amdgpu-install/5.5.02.05.2/ubuntu/jammy/amdgpu-install_5.5.02.05.50502-1_all.deb
     $ sudo apt install ./amdgpu-install_5.5.02.05.50502-1_all.deb
     $ sudo sed -i 's#repo.radeon.com#&/.preview/a0e4ef1dffbc95b4abb54e891f265e61#' /etc/apt/sources.list.d/{amdgpu.list,rocm.list,amdgpu-proprietary.list}
     ```

   - 他の Ubuntu バージョンの場合:

     ```
     $ sudo dpkg --add-architecture i386
     $ sudo apt-get update -y && sudo apt upgrade -y
     ```

   - CentOS の場合:

     ```
     $ sudo yum install epel-release -y
     $ sudo yum update -y
     ```

5. インスタンスを再起動します。

```
$ sudo reboot
```

6. 再起動後にインスタンスに再接続します。
7. 最新の AMD ドライバーをダウンロードします。

> **Note**
>
> Ubuntu 22.04 の場合は、このステップをスキップします。

```
$ aws s3 cp --recursive s3://ec2-amd-linux-drivers/latest/ .
```

8. ファイルを抽出します。

   - Amazon Linux 2 と CentOS の場合:

     ```
     $ tar -xf amdgpu-pro-*rhel*.tar.xz
     ```

   - Ubuntu の場合:

     > **Note**
     >
     > Ubuntu 22.04 の場合は、このステップをスキップします。

     ```
     $ tar -xf amdgpu-pro*ubuntu*.xz
     ```

9. 抽出されたドライバーのフォルダに変更します。
10. ドライバーをインストールする上で不足しているモジュールを追加します。

    - Amazon Linux 2 と CentOS の場合:

      この手順をスキップしてください。

    - Ubuntu の場合:

> **Note**
>
> Ubuntu 22.04 の場合は、このステップをスキップします。

```
$ sudo apt install linux-modules-extra-$(uname -r) -y
```

11. 自己インストールスクリプトを実行して、完全なグラフィックススタックをインストールします。

    - Ubuntu 22.04 の場合:

      ```
      $ sudo amdgpu-install --usecase=workstation --vulkan=pro --opencl=rocr,legacy -y
      ```

    - Amazon Linux 2 および CentOS およびその他の Ubuntu バージョンの場合:

      ```
      $ ./amdgpu-pro-install -y --opencl=pal,legacy
      ```

12. インスタンスを再起動します。

    ```
    $ sudo reboot
    ```

13. ドライバーが機能していることを確認します。

    ```
    $ dmesg | grep amdgpu
    ```

    レスポンスは次のようになります。

    ```
    Initialized amdgpu
    ```

対話型デスクトップのセットアップ

インスタンスに AMD GPU ドライバーがインストールされ、amdgpu が使用中であることを確認したら、対話型デスクトップマネージャーをインストールできます。最高の互換性とパフォーマンスを得るには、MATE デスクトップ環境をお勧めします。

前提条件

テキストエディタを開き、次のファイルを「xorg.conf」という名前のファイルとして保存します。このファイルはインスタンスで必要になります。

```
Section "ServerLayout"
    Identifier     "Layout0"
    Screen          0 "Screen0"
    InputDevice     "Keyboard0" "CoreKeyboard"
    InputDevice     "Mouse0" "CorePointer"
EndSection
Section "Files"
    ModulePath "/opt/amdgpu/lib64/xorg/modules/drivers"
    ModulePath "/opt/amdgpu/lib/xorg/modules"
    ModulePath "/opt/amdgpu-pro/lib/xorg/modules/extensions"
    ModulePath "/opt/amdgpu-pro/lib64/xorg/modules/extensions"
    ModulePath "/usr/lib64/xorg/modules"
    ModulePath "/usr/lib/xorg/modules"
EndSection
Section "InputDevice"
    # generated from default
    Identifier     "Mouse0"
    Driver         "mouse"
    Option         "Protocol" "auto"
    Option         "Device" "/dev/psaux"
    Option         "Emulate3Buttons" "no"
    Option         "ZAxisMapping" "4 5"
EndSection
Section "InputDevice"
    # generated from default
    Identifier     "Keyboard0"
    Driver         "kbd"
EndSection
Section "Monitor"
    Identifier     "Monitor0"
    VendorName     "Unknown"
    ModelName      "Unknown"
EndSection
Section "Device"
    Identifier     "Device0"
    Driver         "amdgpu"
    VendorName     "AMD"
    BoardName      "Radeon MxGPU V520"
    BusID          "PCI:0:30:0"
EndSection
```

```
Section "Extensions"
    Option         "DPMS" "Disable"
EndSection
Section "Screen"
    Identifier     "Screen0"
    Device         "Device0"
    Monitor        "Monitor0"
    DefaultDepth   24
    Option         "AllowEmptyInitialConfiguration" "True"
    SubSection "Display"
        Virtual    3840 2160
        Depth      32
    EndSubSection
EndSection
```

Amazon Linux 2 でインタラクティブデスクトップをセットアップするには

1. EPEL リポジトリをインストールします。

   ```
   $ sudo amazon-linux-extras install epel -y
   ```

2. MATE デスクトップをインストールします。

   ```
   $ sudo amazon-linux-extras install mate-desktop1.x -y
   $ sudo yum groupinstall "MATE Desktop" -y
   $ sudo systemctl disable firewalld
   ```

3. xorg.conf ファイルを /etc/X11/xorg.conf にコピーします。
4. インスタンスを再起動します。

   ```
   $ sudo reboot
   ```

5. (オプション) NICE DCV サーバーをインストールして、NICE DCV を高パフォーマンスの表示プロトコルとして使用してから、お好みのクライアントを使用して NICE DCV セッションに接続します。

Ubuntu でインタラクティブデスクトップをセットアップするには

1. MATE デスクトップをインストールします。

   ```
   $ sudo apt install xorg-dev ubuntu-mate-desktop -y
   ```

```
$ sudo apt purge ifupdown -y
```

2. xorg.conf ファイルを /etc/X11/xorg.conf にコピーします。
3. インスタンスを再起動します。

```
$ sudo reboot
```

4. 適切なバージョンの Ubuntu 用の AMF エンコーダーをインストールします。

```
$ sudo apt install ./amdgpu-pro-20.20-*/amf-amdgpu-pro_20.20-*_amd64.deb
```

5. (オプション) NICE DCV サーバーをインストールして、NICE DCV を高パフォーマンスの表示プロトコルとして使用してから、お好みのクライアントを使用して NICE DCV セッションに接続します。
6. DCV のインストール後、DCV ユーザーに動画のアクセス権限を付与します。

```
$ sudo usermod -aG video dcv
```

CentOS で対話型デスクトップをセットアップするには

1. EPEL リポジトリをインストールします。

```
$ sudo yum update -y
$ sudo yum install epel-release -y
```

2. MATE デスクトップをインストールします。

```
$ sudo yum groupinstall "MATE Desktop" -y
$ sudo systemctl disable firewalld
```

3. xorg.conf ファイルを /etc/X11/xorg.conf にコピーします。
4. インスタンスを再起動します。

```
$ sudo reboot
```

5. (オプション) NICE DCV サーバーをインストールして、NICE DCV を高パフォーマンスの表示プロトコルとして使用してから、お好みのクライアントを使用して NICE DCV セッションに接続します。

G4ad でのデュアル 4K ディスプレイの設定

G4ad インスタンスを起動する

1. Linux インスタンスに接続して、デュアル 4K (2x4k) 向けにターゲットとする GPU の PCI Bus アドレスを取得します。

```
lspci -vv | grep -i amd
```

以下のような出力結果が取得できます。

```
00:1e.0 Display controller: Advanced Micro Devices, Inc. [*AMD*/ATI] Device 7362 (rev
 c3)
Subsystem: Advanced Micro Devices, Inc. [AMD/ATI] Device 0a34
```

2. 上記の出力では、PCI バスアドレスは 00:1e.0 であることに注意してください。/etc/modprobe.d/amdgpu.conf という名前のファイルを作成して追加します。

```
options amdgpu virtual_display=0000:00:1e.0,2
```

3. Linux に AMD ドライバーをインストールするには、こちらの手順に従ってください。AMD GPU ドライバーが既にインストールされている場合は、dkms を通じて amdgpu カーネルモジュールを再構築する必要があります。
4. 以下の xorg.conf ファイルを使用して、デュアル (2x4K) スクリーントポロジを定義し、/etc/X11/xorg.conf: のファイルに保存します。

```
~$ cat /etc/X11/xorg.conf
Section "ServerLayout"
    Identifier     "Layout0"
    Screen          0 "Screen0"
    Screen        1 "Screen1"
    InputDevice     "Keyboard0" "CoreKeyboard"
    InputDevice     "Mouse0" "CorePointer"
    Option          "Xinerama" "1"
EndSection
Section "Files"
    ModulePath "/opt/amdgpu/lib64/xorg/modules/drivers"
    ModulePath "/opt/amdgpu/lib/xorg/modules"
    ModulePath "/opt/amdgpu-pro/lib/xorg/modules/extensions"
    ModulePath "/opt/amdgpu-pro/lib64/xorg/modules/extensions"
    ModulePath "/usr/lib64/xorg/modules"
```

```
    ModulePath "/usr/lib/xorg/modules"
EndSection
Section "InputDevice"
    # generated from default
    Identifier     "Mouse0"
    Driver         "mouse"
    Option         "Protocol" "auto"
    Option         "Device" "/dev/psaux"
    Option         "Emulate3Buttons" "no"
    Option         "ZAxisMapping" "4 5"
EndSection
Section "InputDevice"
    # generated from default
    Identifier     "Keyboard0"
    Driver         "kbd"
EndSection

Section "Monitor"
    Identifier     "Virtual"
    VendorName     "Unknown"
    ModelName      "Unknown"
    Option         "Primary" "true"
EndSection

Section "Monitor"
    Identifier     "Virtual-1"
    VendorName     "Unknown"
    ModelName      "Unknown"
    Option         "RightOf" "Virtual"
EndSection

Section "Device"
    Identifier     "Device0"
    Driver         "amdgpu"
    VendorName     "AMD"
    BoardName      "Radeon MxGPU V520"
    BusID          "PCI:0:30:0"
EndSection

Section "Device"
    Identifier     "Device1"
    Driver         "amdgpu"
    VendorName     "AMD"
    BoardName      "Radeon MxGPU V520"
```

```
    BusID          "PCI:0:30:0"
EndSection

Section "Extensions"
    Option         "DPMS" "Disable"
EndSection

Section "Screen"
    Identifier     "Screen0"
    Device         "Device0"
    Monitor        "Virtual"
    DefaultDepth   24
    Option         "AllowEmptyInitialConfiguration" "True"
    SubSection "Display"
        Virtual    3840 2160
        Depth      32
    EndSubSection
EndSection

Section "Screen"
    Identifier     "Screen1"
    Device         "Device1"
    Monitor        "Virtual"
    DefaultDepth   24
    Option         "AllowEmptyInitialConfiguration" "True"
    SubSection "Display"
        Virtual    3840 2160
        Depth      32
    EndSubSection
EndSection
```

5. インタラクティブデスクトップの設定手順に従って、DCV を設定します。
6. DCV の設定が完了したら、再起動します。
7. ドライバーが機能していることを確認します。

```
dmesg | grep amdgpu
```

レスポンスは次のようになります。

```
Initialized amdgpu
```

8. DISPLAY=:0 xrandr -q の出力で、2 つの仮想ディスプレイが接続されていることを確認できます。

```
~$ DISPLAY=:0 xrandr -q
Screen 0: minimum 320 x 200, current 3840 x 1080, maximum 16384 x 16384
Virtual connected primary 1920x1080+0+0 (normal left inverted right x axis y axis)
 0mm x 0mm
 4096x3112  60.00
 3656x2664  59.99
 4096x2160  60.00
 3840x2160  60.00
 1920x1200  59.95
 1920x1080  60.00
 1600x1200  59.95
 1680x1050  60.00
 1400x1050  60.00
 1280x1024  59.95
 1440x900 59.99
 1280x960 59.99
 1280x854 59.95
 1280x800 59.96
 1280x720 59.97
 1152x768 59.95
 1024x768 60.00 59.95
 800x600  60.32 59.96 56.25
 848x480  60.00 59.94
 720x480  59.94
 640x480  59.94 59.94
Virtual-1 connected 1920x1080+1920+0 (normal left inverted right x axis y axis) 0mm x
 0mm
 4096x3112  60.00
 3656x2664  59.99
 4096x2160  60.00
 3840x2160  60.00
 1920x1200  59.95
 1920x1080  60.00
 1600x1200  59.95
 1680x1050  60.00
 1400x1050  60.00
 1280x1024  59.95
 1440x900 59.99
 1280x960 59.99
 1280x854 59.95
```

```
 1280x800 59.96
 1280x720 59.97
 1152x768 59.95
 1024x768 60.00 59.95
 800x600  60.32 59.96 56.25
 848x480  60.00 59.94
 720x480  59.94
640x480  59.94 59.94
```

9. DCV に接続するときは、解像度を 2x4K に変更し、デュアルモニタのサポートが DCV によって登録されていることを確認します。

NVIDIA GRID 仮想アプリケーションの有効化

G3、G4dn、および G5 インスタンスで GRID 仮想アプリケーションをアクティブ化するには (NVIDIA GRID 仮想ワークステーションはデフォルトで有効)、/etc/nvidia/gridd.conf ファイルでドライバーの製品タイプを定義する必要があります。

Linux インスタンスで GRID 仮想アプリケーションを有効にするには

1. 提供されるテンプレートファイルから /etc/nvidia/gridd.conf ファイルを作成します。

```
[ec2-user ~]$ sudo cp /etc/nvidia/gridd.conf.template /etc/nvidia/gridd.conf
```

2. お好きなテキスト工ディタで /etc/nvidia/gridd.conf ファイルを開きます。
3. FeatureType 行を見つけ、それを 0 と等しくなるように設定します。次に、IgnoreSP=TRUE の行を追加します。

```
FeatureType=0
IgnoreSP=TRUE
```

4. ファイルを保存して終了します。
5. インスタンスを再起動し、新しい設定を取得します。

```
[ec2-user ~]$ sudo reboot
```

GPU 設定の最適化

NVIDIA GPU インスタンスで最大のパフォーマンスを実現するためには、いくつかの最適化方法の中から GPU 設定を選択できます。これらのインスタンスタイプの一部では、NVIDIA ドライバーは自動ブースト機能を使用しますが、これは GPU クロック速度に左右されます。自動ブーストを無効にし、GPU クロックを最大周波数に設定することで、GPU インスタンスのパフォーマンスを一貫して最大に維持することができます。次の手順では、GPU 設定を永続的に設定し、必要に応じて自動ブースト機能を無効化して、GPU クロック速度を最大周波数に設定します。

以下の手順は、 Linux インスタンスの GPU 設定を最適化するためのものです。 Windows インスタンスには、「」「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「https://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/optimize_gpu.html」を参照してください。

GPU 設定を最適化するには

1. GPU 設定を永続的になるように設定します。このコマンドの実行には数分かかることがあります。

   ```
   [ec2-user ~]$ sudo nvidia-persistenced
   ```

2. [G2、G3、および P2 インスタンスのみ] インスタンス上のすべての GPU の自動ブースト機能を無効にします。

   ```
   [ec2-user ~]$ sudo nvidia-smi --auto-boost-default=0
   ```

3. すべての GPU クロックを最大周波数に設定します。次のコマンドで指定されるメモリとグラフィッククロック速度を使用します。

   一部のバージョンの NVIDIA ドライバーでは、アプリケーションのクロック速度の設定をサポートしていないため、"Setting applications clocks is not supported for GPU..." エラーが表示されますが、無視できます。

   - G3 インスタンス:

     ```
     [ec2-user ~]$ sudo nvidia-smi -ac 2505,1177
     ```

   - G4dn インスタンス:

     ```
     [ec2-user ~]$ sudo nvidia-smi -ac 5001,1590
     ```

   - G5 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 6250,1710
```

- P2 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 2505,875
```

- P3 および P3dn インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 877,1530
```

- P4d インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 1215,1410
```

- P4de インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 1593,1410
```

- P5 インスタンス:

```
[ec2-user ~]$ sudo nvidia-smi -ac 2619,1980
```

ハイパフォーマンスコンピューティングインスタンス

> **Note**
>
> インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。

ハイパフォーマンスコンピューティングインスタンスは、AWS で HPC ワークロードを大規模に実行するのに最高のコストパフォーマンスを提供するよう設計されています。このインスタンスは、大規模で複雑なシミュレーションや深層学習ワークロードなど、ハイパフォーマンスプロセッサを活用できるアプリケーションに最適です。

Hpc6a インスタンス

これらのインスタンスは、次のようなハイパフォーマンスコンピューティング (HPC) ワークロードの実行に最適です。

- 分子動力学
- 計算化学
- 計算流体力学
- 気象予測
- マテリアルシミュレーション
- クラッシュシミュレーション
- 天体物理学

詳細については、「Amazon EC2 Hpc6a Instances (Amazon EC2 Hpc6a インスタンス)」を参照してください。

Hpc6id インスタンス

これらのインスタンスは、次のようなハイパフォーマンスコンピューティング (HPC) ワークロードの実行に最適です。

- 地震とリザーバ
- クラッシュシミュレーション
- 有限要素解析

Hpc7g インスタンス

HPC7g インスタンスは、AWS Graviton3E プロセッサと次世代の AWS Nitro Card を搭載しています。Elastic Fabric Adapter (EFA) による 200 Gbps の高度なネットワーキングを実現し、低レイテンシーと高いネットワークパフォーマンスを実現します。

Hpc7g インスタンスは、計算流体力学、分子動力学、気象シミュレーションなど、コンピューティングとメモリの帯域幅のパフォーマンス、および低いネットワークレイテンシーから恩恵を受ける計算負荷の高い HPC アプリケーションの実行に最適です。

詳細については、「Amazon EC2 Hpc7g インスタンス」を参照してください。

Hpc7a インスタンス

これらのインスタンスは、Elastic Fabric Adapter (EFA) による最大 300 Gbps のネットワーク帯域幅を備えており、Message Passing Interface (MPI) で低いレイテンシーと高いネットワークパフォーマンスを実現します。最大 768 GB のシステムメモリと最大 192 個の CPU コアを備えています。

Hpc7a インスタンスは、計算流体力学、分子動力学、気象シミュレーションなど、インスタンスごとの多数のコアおよび低いネットワークレイテンシーから恩恵を受ける計算負荷の高い HPC アプリケーションの実行に最適です。

詳細については、「Amazon EC2 Hpc7a instances」を参照してください。

コンテンツ

ハードウェア仕様

仮想中央処理ユニット (vCPU) は、仮想マシン (VM) に割り当てられた物理 CPU の一部を表します。x86 インスタンスの場合、コアごとに 2 つの vCPU があります。Graviton インスタンスの場合、コアごとに 1 つの vCPU があります。

ハードウェアの仕様については、「Amazon EC2 インスタンスタイプガイド」の「ハイパフォーマンスコンピューティングインスタンス」を参照してください。

インスタンスのパフォーマンス

EBS 最適化インスタンスは、インスタンスからの Amazon EBS I/O とその他のネットワークトラフィックとの競合を排除することによって、EBS ボリュームの安定した高パフォーマンスを実現できます。一部のコンピューティングの最適化インスタンスは、追加料金なしでデフォルトで EBS 最適化されています。詳細については、Amazon EBS 最適化インスタンスを使用する を参照してください。

一部のコンピューティングの最適化インスタンスでは、Linux でプロセッサの C ステートと P ステートを制御できます。C ステートは非アクティブ時のコアのスリープレベルを制御し、P ステート

は希望するコアからのパフォーマンス (CPU 周波数) を制御します。詳細については、EC2 インスタンスのプロセッサのステート制御 を参照してください。

ネットワークパフォーマンス

サポートされているインスタンスタイプで拡張ネットワーキングを有効にすると、レイテンシーとネットワークジッターを低減し、パケット毎秒 (PPS) のパフォーマンスを高めることができます。ほとんどのアプリケーションでは、高いレベルのネットワークパフォーマンスが一貫して必要なわけではありませんが、データの送受信時にアクセスする帯域幅を増やすことでメリットを得られます。詳細については、「Linux での拡張ネットワーキング」を参照してください。

ネットワークの仕様については、「Amazon EC2 インスタンスタイプガイド」の「ハイパフォーマンスコンピューティングインスタンス」を参照してください。

Amazon EBS I/O パフォーマンス

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、Amazon EBS ボリュームの最高のパフォーマンスを実現します。

詳細については、「Amazon EBS 最適化インスタンスを使用する」を参照してください。

SSD ベースのインスタンスストアボリュームの I/O パフォーマンス

インスタンスストアボリュームは、インスタンスの存続中のみ使用できます。インスタンスを停止、休止、または終了すると、アプリケーションとそのインスタンスストアボリュームのデータは消去されます。インスタンスストアボリュームの重要なデータは、定期的にバックアップまたはレプリケートすることをお勧めします。詳細については、「Amazon EC2 インスタンスストア」および「SSD インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仕様については、「Amazon EC2 インスタンスタイプガイド」の「ハイパフォーマンスコンピューティングインスタンス」を参照してください。

インスタンスに SSD ベースのインスタンスストアボリュームを使用するほど、アーカイブできる書き込み IOPS の数は減少します。これは、SSD コントローラーが実行する必要がある追加の作業が原因です。SSD コントローラーは、利用可能な領域を見つけ、既存のデータを再書き込みし、未使用の領域を消去して、再書き込みができるようにします。このガベージコレクションというプロセスにより、SSD への内部的な書き込み増幅が発生し、ユーザーの書き込み操作に対する SSD 書き込み

操作の割合として表示されます。書き込み操作が 4,096 バイトの倍数でないか、4,096 バイトの境界に整合していない場合、パフォーマンスの低下はさらに大きくなります。少量のバイト数または整合していないバイト数で書き込む場合、SSD コントローラーは周辺のデータを読み取り、その結果を新しい場所に保存する必要があります。このパターンにより、書き込み増幅が大幅に増え、レイテンシーが増加し、I/O パフォーマンスが大きく低下します。

SSD コントローラーは、複数の方法を利用すると、書き込み増幅の影響を減らすことができます。このような方法の 1 つには、SSD インスタンスストレージに領域を予約し、コントローラーが書き込み操作に利用できる領域をより効率的に管理できるようにすることです。これをオーバープロビジョニングと呼びます。インスタンスに提供された SSD ベースのインスタンスストアボリュームには、オーバープロビジョニングに対して予約された領域がありません。書き込み増幅を減らすには、ボリュームの 10% を未使用の状態のままにし、SSD コントローラーがこれをオーバープロビジョニングに使用できるようにすることをお勧めします。これにより、使用できるストレージは減りますが、ディスクが総容量に近づいた場合でもパフォーマンスを向上させることができます。

TRIM をサポートするインスタンスストアボリュームの場合、TRIM コマンドを使用して、書き込んだデータが不要になったときはいつでも SSD コントローラーに通知することができます。これにより、より多くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。

リリースノート

- Nitro System 上に構築されたインスタンスには、次の要件があります。
  - NVMe ドライバーがインストールされている必要があります。
  - Elastic Network Adapter (ENA) ドライバーがインストールされている必要があります。

  以下の Linux AMI はこれらの要件を満たしています。
  - AL2023
  - Amazon Linux 2
  - Amazon Linux AMI 2018.03 以降
  - `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

  > **Note**
  >
  > AWS Graviton ベースのインスタンスタイプには、`linux-aws` カーネル搭載の Ubuntu 18.04 以降が必要です

  - Red Hat Enterprise Linux 7.4 以降
  - SUSE Linux Enterprise Server 12 SP2 以降
  - CentOS 7.4.1708 以降
  - FreeBSD 11.1 以降
  - Debian GNU/Linux 9 以降
- AWS Graviton プロセッサを使用するインスタンスには、次の要件があります。
  - 64 ビット Arm アーキテクチャ用の AMI を使用する必要があります。
  - ACPI テーブルを含む UEFI による起動と、PCI デバイスの ACPI ホットプラグをサポートしている必要があります。

  以下の AMI はこれらの要件を満たしています。
  - Amazon Linux 2 (64 ビット Arm)
  - Ubuntu 16.04 以降 (64 ビット Arm)
  - Red Hat Enterprise Linux 8.0 以降 (64 ビット Arm)
  - SUSE Linux Enterprise Server 15 以降 (64 ビット Arm)
  - Debian 10 以降 (64 ビット Arm)
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。詳細については、「インスタンスボリューム数の制限」を参照してください。
- ベアメタルインスタンスを起動すると、基盤となるサーバーが起動します。これには、すべてのハードウェアやファームウェアコンポーネントの確認が含まれます。つまり、インスタンスが実行状態になってからネットワーク経由で使用できるようになるまでに 20 分かかることがあります。
- EBS ボリュームまたはセカンダリネットワークインターフェイスを、ベアメタルインスタンスにアタッチ (または、そこからデタッチ) するには、PCIe のネイティブホットプラグがサポートされている必要があります。PCIe のネイティブホットプラグは、Amazon Linux 2 および最新バージョンの Amazon Linux AMI でサポートされています。それ以前のバージョンではサポートされていません。次の Linux カーネル設定オプションを有効にする必要があります。

```
CONFIG_HOTPLUG_PCI_PCIE=y
CONFIG_PCIEASPM=y
```

- ベアメタルインスタンスでは、I/O ポートベースのシリアルデバイスではなく、PCI ベースのシリアルデバイスを使用しています。アップストリームの Linux カーネルと最新の Amazon Linux AMI は、このデバイスをサポートしています。また、ベアメタルインスタンスでは、システムが PCI

ベースのシリアルデバイスを自動的に使用できるようにする ACPI SPCR テーブルも使用できます。最新の Windows AMI では、自動的に PCI ベースのシリアルデバイスが使用されます。

- Nitro System 上に構築されたインスタンスには、API リクエストによるクリーンシャットダウンをサポートするために acpid がインストールされている必要があります。
- リージョンで起動できるインスタンスの合計数には制限があります。また、一部のインスタンスタイプにはその他の制限もあります。詳細については、Amazon EC2 の「よくある質問」の「Amazon EC2 で実行できるインスタンス数の上限は」を参照してください。

Amazon EC2 インスタンスタイプの検索

インスタンスの起動前には、使用するインスタンスタイプを選択しなければなりません。選択するインスタンスタイプは、コンピューティング、メモリ、ストレージリソースなど、ワークロードが必要とするリソースに応じて異なる場合があります。ワークロードに適したインスタンスタイプをいくつか特定し、テスト環境でそれらのパフォーマンスを評価することは有益な場合があります。負荷時のアプリケーションのパフォーマンスを測定するための、代替手段はありません。

EC2 インスタンスをすでに実行している場合は、AWS Compute Optimizer を使用すると、パフォーマンスの向上、コストの削減、またはその両方に使用するインスタンスタイプの、推奨事項を取得できます。詳細については、the section called “既存のワークロード用” を参照してください。

タスク

- コンソールを使用したインスタンスタイプの検索
- AWS CLI を使用したインスタンスタイプの検索

コンソールを使用したインスタンスタイプの検索

Amazon EC2 コンソールを使用して、ニーズに合ったインスタンスタイプを検索できます。

コンソールを使用したインスタンスタイプを検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、インスタンスを起動するリージョンを選択します。お客様は場所に関係なく、使用できるリージョンをどれでも選択できます。
3. ナビゲーションペインで、[インスタンスタイプ] を選択します。
4. (オプション) プリファレンス (歯車) アイコンを選択して オンデマンド Linux 料金などの表示するインスタンスタイプ属性を選択し、次に [確認] を選択します。または、インスタンスタイプ

の名前を選択して詳細ページを開き、コンソールで使用できるすべての属性を表示します。このコンソールには、API またはコマンドラインで使用できる属性のすべては表示されません。

5. インスタンスタイプ属性を使用して、ニーズを満たすインスタンスタイプのみがリスト表示されるようにフィルタリングします。例えば、以下の属性でフィルターをかけることができます。

   - [アベイラビリティーゾーン] — アベイラビリティーゾーン、ローカルゾーン、Wavelength Zone の名前。詳細については、the section called “リージョンとゾーン” を参照してください。
   - vCPU または コア — vCPUs またはコアの数。
   - [Memory (GiB)] (メモリ (GiB)) — メモリサイズ (GiB 単位)。
   - [Network performance] (ネットワークパフォーマンス) — ネットワークパフォーマンス (ギガビット)。
   - [Local instance storage] (ローカルインスタンスストレージ) — インスタンスタイプにローカルインスタンスストレージがあるかどうかを示します (`true|false`)。

6. (オプション) 並べて比較するには、複数のインスタンスタイプに対応するチェックボックスをオンにします。比較は、画面下部に表示されます。
7. (オプション) インスタンスタイプのリストをカンマ区切り値 (.csv) ファイルで保存してさらに見直せるようにするには、[Actions] (アクション)、[Download list] (リスト CSV をダウンロード) の順に選択します。このファイルには、設定したフィルターに一致するすべてのインスタンスタイプが含まれます。
8. (オプション) ニーズを満たすインスタンスタイプを使用してインスタンスを起動するには、インスタンスタイプのチェックボックスをオンにして [Actions] (アクション)、[Launch instance] (インスタンスを起動) の順に選択します。詳細については、新しいインスタンス起動ウィザードを使用してインスタンスを起動する を参照してください。

AWS CLI を使用したインスタンスタイプの検索

Amazon EC2 の AWS CLI コマンドを使用して、ニーズに合ったインスタンスタイプを見つけることができます。

AWS CLI を使用してインスタンスタイプを検索するには

1. まだ AWS CLI を用意していない場合はインストールします。詳細については、AWS Command Line Interface ユーザーガイドを参照してください。

2. インスタンス属性に基づいてインスタンスタイプをフィルターするには、ribe-instance-types コマンドを使用します。例えば、次のコマンドを使用すると、64 GiB (65536 MiB) のメモリを持つ現行世代のインスタンスタイプのみを表示できます。

```
aws ec2 describe-instance-types --filters "Name=current-generation,Values=true"
 "Name=memory-info.size-in-mib,Values=65536" --query "InstanceTypes[*].
[InstanceType]" --output text | sort
```

3. 場所 (リージョンまたはゾーン) によって提供されるインスタンスタイプをフィルタリングするには、describe-instance-type-offings コマンドを使用します。例えば、次のコマンドを使用して、指定されたゾーンで提供されるインスタンスタイプを表示できます。

```
aws ec2 describe-instance-type-offerings --location-type "availability-
zone" --filters Name=location,Values=us-east-2a --region us-east-2 --query
 "InstanceTypeOfferings[*].[InstanceType]" --output text | sort
```

4. ニーズを満たすインスタンスタイプを見つけたら、インスタンスを起動するときにこれらのインスタンスタイプを使用できるように、そのリストを保存しておきます。詳細については、AWS Command Line Interface ユーザーガイドの「インスタンスの起動」を参照してください。

インスタンスタイプに関する推奨事項の取得

以下のツールは、新規または既存のワークロードに最適なインスタンスタイプを選択するのに役立ちます。

- 新しいワークロード Amazon Q EC2 インスタンスタイプセレクターは、ユースケース、ワークロードタイプ、CPU メーカーの好みだけでなく、価格とパフォーマンスの優先順位も考慮します。次に、このデータを使用して、新しいワークロードへ最適な Amazon EC2 インスタンスタイプのガイダンスと提案を提供します。
- 既存のワークロード — AWS Compute Optimizer は既存インスタンスの仕様と使用率メトリクスを分析します。次に、コンパイルされたデータを使用して、既存のワークロードのコストまたはパフォーマンス、あるいはその両方で最適な Amazon EC2 インスタンスタイプを推奨します。

以下でインスタンスタイプの推奨事項を取得します。

- 新しいワークロードのインスタンスタイプに関する推奨事項の取得
- 既存のワークロード用インスタンスタイプに関する推奨事項の取得

新しいワークロードのインスタンスタイプに関する推奨事項の取得

> **Note**
>
> Amazon Bedrock を搭載: AWS により不正使用検知が自動化されています。Amazon Q EC2 インスタンスタイプセレクターは Amazon Bedrock 上に構築されているため、ユーザーは Amazon Bedrock に実装されている制御を最大限に活用し、安全性、セキュリティ、および人工知能 (AI) の責任ある使用を実現できます。

Amazon Q EC2 インスタンスタイプセレクターは Amazon EC2 のプレビュー版であり、変更される場合があります。

Amazon Q EC2 インスタンスタイプセレクターは、優先するユースケース、ワークロードタイプ、CPU メーカーだけではなく、価格とパフォーマンスの優先順位も考慮します。次に、このデータを使用して、新しいワークロードへ最適な Amazon EC2 インスタンスタイプのガイダンスと提案を提供します。

利用可能なインスタンスタイプの数が非常に多いため、ワークロードに適したインスタンスタイプを見つけるのは時間がかかり、困難である場合があります。Amazon Q EC2 インスタンスタイプセレクターを使用することで、最新のインスタンスタイプを常に把握し、ワークロードに最適なコストパフォーマンスを実現できます。

このトピックでは、Amazon EC2 コンソールで EC2 インスタンスタイプに関するガイダンスと推奨事項を表示する方法について説明します。Amazon Q に直接アクセスして、インスタンスタイプのアドバイスを求めることもできます。詳細については、「Amazon Q (AWS ビルダー用) ユーザーガイド」を参照してください。

既存のワークロードに適したインスタンスタイプをお探しの場合は、AWS Compute Optimizer を使用します。詳細については、「既存のワークロード用インスタンスタイプに関する推奨事項の取得」を参照してください。

サポートされている AWS リージョン

Amazon Q EC2 インスタンスタイプセレクターは Amazon Q を使用しているため、Amazon Q がサポートされているのと同じリージョンでサポートされています。詳細については、「Amazon Q (AWS ビルダー用) ユーザーガイド」の「Amazon Q のサポート対象リージョン」を参照してください。

Amazon Q EC2 インスタンスタイプセレクタを使用する

Amazon EC2 コンソールでは、[アドバイスの取得] リンクを選択して Amazon Q にインスタンスタイプに関するアドバイスを求めることができます。ユースケース、ワークロードタイプ、希望する CPU メーカー、価格とパフォーマンスの優先順位を指定すると、Amazon Q AI アシスタントが開き、インスタンスタイプのアドバイスが表示されます。Amazon Q に直接アクセスして、インスタンスタイプのアドバイスを求めることもできます。詳細については、「Amazon Q (AWS ビルダー用) ユーザーガイド」を参照してください。

次の EC2 コンソールページに [アドバイスを取得] リンクが表示されます。

- Amazon EC2 インスタンス起動ウィザード
- Amazon EC2 起動テンプレート

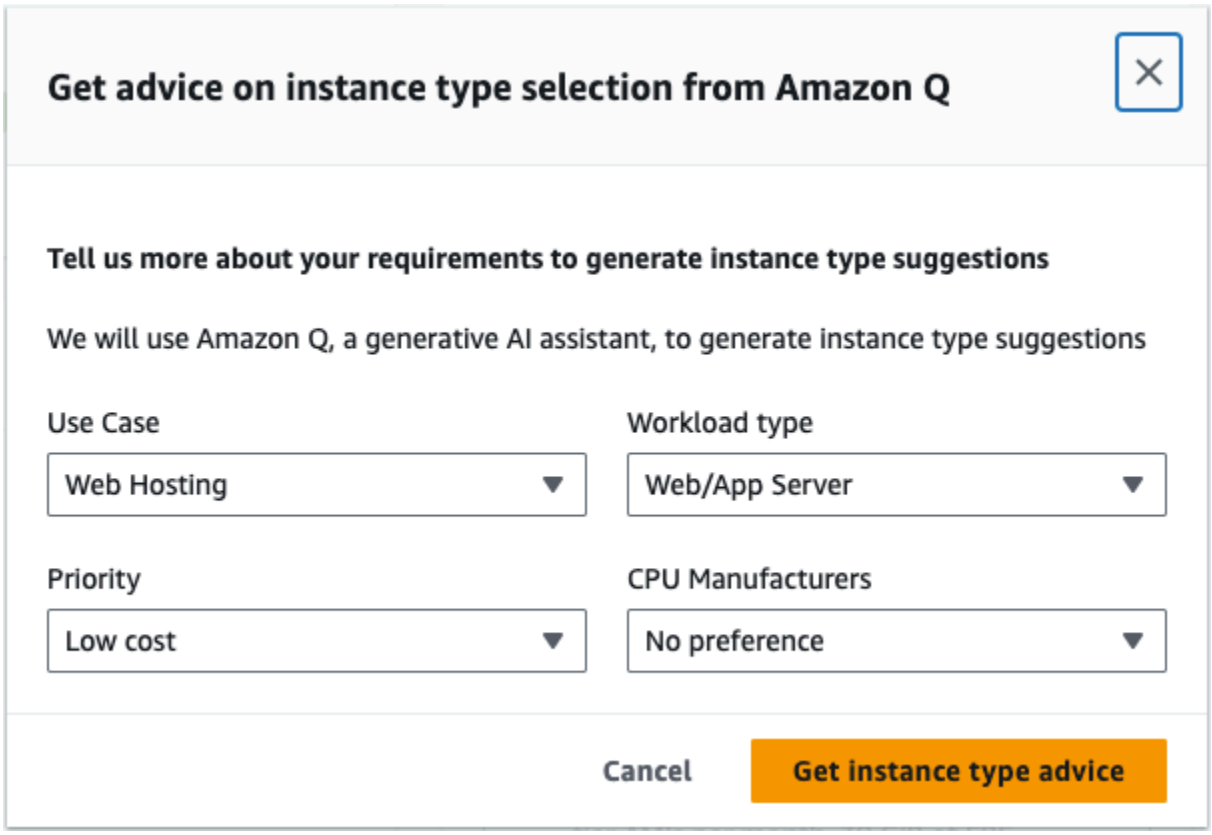
Amazon EC2 コンソールの Amazon Q EC2 インスタンスタイプセレクターを使用して EC2 インスタンスタイプのガイダンスとアドバイスを取得するには、次の手順に従います。

Amazon Q EC2 インスタンスタイプセレクターを使用してインスタンスタイプに関するアドバイスを取得するには

1. 手順に従ってインスタンスを起動するか、起動テンプレートを作成します。
2. Amazon Q EC2 インスタンスタイプセレクターを使用してインスタンスタイプに関するアドバイスを取得するには、以下を実行してください。

   a. [インスタンスタイプ] の横にある [アドバイスを取得] リンクを選択します。

   b. [Amazon Q からインスタンスタイプ選択に関するアドバイスを取得] ウィンドウで、ドロップダウンリストからオプションを選択してインスタンスタイプの要件を指定します。[ユースケース] と [ワークロードタイプ] では、[その他] を選択し、要件を入力できます。



c. [インスタンスタイプに関するアドバイスを取得] を選択します。

   Amazon Q AI アシスタントが開き、要件に合わせてカスタマイズされたインスタンスタイプの候補が表示されます。

d. その他のインスタンスタイプ要件については、引き続き Amazon Q と自然言語でチャットを続けることができます。

3. 使用するインスタンスタイプを決定したら、インスタンス起動ウィザードまたは起動テンプレートの [インスタンスタイプ] で、インスタンスタイプを選択します。
4. インスタンスを起動する手順を完了するか、起動テンプレートを作成します。

既存のワークロード用インスタンスタイプに関する推奨事項の取得

AWS Compute Optimizer から、パフォーマンスの向上、コストの削減、またはその両方に役立つ、Amazon EC2 インスタンスに関する推奨事項が得られます。これらの推奨事項を使用して、新しいインスタンスタイプに移行するかどうかを判断できます。

推奨事項を作成するために、Compute Optimizer は既存のインスタンスの仕様と使用率メトリクスを分析します。次に、コンパイルされたデータを使用して、既存のワークロードを処理するのに最適な

Amazon EC2 インスタンスタイプを推奨します。推奨事項は、時間あたりのインスタンス料金とともに返されます。

このトピックでは、Amazon EC2 コンソールで推奨事項を表示する方法について説明します。詳細については、AWS Compute Optimizer ユーザーガイドを参照してください。

> **Note**
>
> Compute Optimizer から推奨事項を取得するには、まず Compute Optimizer にオプトインする必要があります。詳細については、AWS Compute Optimizer ユーザーガイドの「AWS Compute Optimizer の使用開始」を参照してください。

新しいワークロード向けの推奨インスタンスタイプをお探しの場合は、Amazon Q EC2 インスタンスタイプセレクターを使用します。詳細については、「新しいワークロードのインスタンスタイプに関する推奨事項の取得」を参照してください。

コンテンツ

制限事項

現在、Compute Optimizer では、インスタンスタイプ C、D、H、I、M、R、T、X、および z の推奨事項を生成します。他のインスタンスタイプは、Compute Optimizer では考慮されません。他のインスタンスタイプを使用している場合は、Compute Optimizer の推奨事項ビューに表示されません。サポートされているインスタンスタイプおよびサポートされていないインスタンスタイプの詳細については、「AWS Compute Optimizer ユーザーガイド」の「Amazon EC2 インスタンスの要件」を参照してください。

結果

Compute Optimizer は、EC2 インスタンスの検出結果を以下のように分類します。

- プロビジョニング不足 – EC2 インスタンスは、インスタンス仕様の CPU、メモリ、ネットワークなどの 1 つ以上の要素がワークロードのパフォーマンス要件を満たしていない場合に、プロビジョニング不足と見なされます。EC2 インスタンスがプロビジョニング不足である場合、アプリケーションのパフォーマンスが低下することがあります。
- 過剰プロビジョニング – EC2 インスタンスは、インスタンス仕様の CPU、メモリ、ネットワークなどの 1 つ以上の要素をサイズダウンしてもワークロードのパフォーマンス要件を満たす場合や、どの仕様要素もプロビジョニング不足でない場合に、過剰プロビジョニングと見なされます。EC2 インスタンスの過剰プロビジョニングは、余分なインフラストラクチャコストを発生させる場合があります。
- 最適化 – EC2 インスタンスは、インスタンス仕様の CPU、メモリ、ネットワークなどのすべての要素がワークロードのパフォーマンス要件を満たし、インスタンスが過剰プロビジョニングされていない場合に、最適化されていると見なされます。最適化された EC2 インスタンスは、最適なパフォーマンスとインフラストラクチャコストでワークロードを実行します。最適化されたインスタンスとして、Compute Optimizer は新世代のインスタンスタイプを推奨する場合があります。
- なし – このインスタンスに対する推奨事項はありません。この結果になる可能性があるのは、Compute Optimizer にオプトインしてから 12 時間未満である場合、インスタンスの実行時間が 30 時間未満である場合、またはインスタンスタイプが Compute Optimizer でサポートされていない場合です。詳細については、前セクションの 制限事項 を参照してください。

推奨事項の表示

Compute Optimizer にオプトインすると、Compute Optimizer が EC2 インスタンスに関して生成した結果を EC2 コンソールで表示できます。次に、Compute Optimizer コンソールにアクセスして推奨事項を表示できます。最近オプトインした場合は、結果が EC2 コンソールに反映されるまで最大 12 時間かかることがあります。

EC2 コンソールを使用して EC2 インスタンスの推奨事項を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンス ID 。
3. インスタンスの概要ページのページ下部にある [AWS Compute Optimizer] バナーで、[詳細を表示] を選択します。

   インスタンスが Compute Optimizer で開きます。インスタンスは、[Current (最新)] インスタンスとしてラベル付けされます。最大 3 つの異なるインスタンスタイプが [Option 1 (オプション 1)]、[Option 2 (オプション 2)]、[Option 3 (オプション 3)] というラベル付きで推奨されます。

ウィンドウの下半分には、現在のインスタンスの最新の CloudWatch メトリクスデータとして、CPU 使用率、メモリ使用率、ネットワーク入力、ネットワーク出力が表示されます。

4. (オプション) Compute Optimizer コンソールで、設定 (⚙) を選択してテーブル内に表示されている列を変更するか、現在のインスタンスタイプと推奨されるインスタンスタイプの購入オプション別に公開されている料金情報を表示します。

> **Note**
>
> リザーブドインスタンスを購入した場合、オンデマンドインスタンスはリザーブドインスタンスとして請求される場合があります。現在のインスタンスタイプを変更する前に、まずリザーブドインスタンスの使用率と適用範囲に対する影響を評価します。

推奨事項の 1 つを使用するかどうかを決定します。最適化の目的を、パフォーマンスの向上、コストの削減、またはこの両方のいずれにするかを決定します。詳細については、AWS Compute Optimizer ユーザーガイド の「リソースの推奨事項の表示」を参照してください。

Compute Optimizer コンソールを使用して、すべてのリージョンにおけるすべての EC2 インスタンスに対する推奨情報を表示するには

1. https://console.aws.amazon.com/compute-optimizer/ で、Compute Optimizer コンソールを開きます。
2. [View recommendations for all EC2 instances (すべての EC2 インスタンスの推奨事項を表示)] を選択します。
3. 推奨事項ページでは、次のアクションを実行できます。
   a. 1 つ以上の AWS リージョンに対する推奨事項をフィルタリングするには、[Filter by one or more Regions] (1 つ以上のリージョンでフィルタリングする) のテキストボックスにリージョンの名前を入力するか、表示されるドロップダウンリストで 1 つ以上のリージョンを選択します。
   b. 別のアカウントのリソースに対する推奨情報を表示するには、[Account (アカウント)] を選択し、別のアカウント ID を選択します。

      このオプションは、組織の 管理アカウント にサインインしていて、組織内のすべてのメンバーアカウントをオプトインした場合にのみ使用できます。
   c. 選択したフィルタをクリアするには、[Clear filters (フィルターのクリア)] を選択します。

d. 現在のインスタンスタイプと推奨されるインスタンスタイプに表示される購入オプションを変更するには、設定 () を選択し、[オンデマンドインスタンス]、[リザーブドインスタンス、標準 1 年間前払いなし]、[リザーブドインスタンス、標準 3 年間前払いなし] のいずれかを選択します。

e. 追加の推奨事項や使用率メトリックスの比較などの詳細を表示するには、目的のインスタンスの横に表示される結果 ([Under-provisioned (プロビジョニング不足]、[Over-provisioned (過剰プロビジョニング)]、または [Optimized (最適化)]) を選択します。詳細については、AWS Compute Optimizer ユーザーガイドの「リソースの詳細の表示」を参照してください。

推奨事項の評価に関する考慮事項

インスタンスタイプを変更する前に、次の点を考慮してください。

- 推奨情報は使用状況を予測するものではありません。推奨事項は、直近の 14 日間の使用履歴に基づいています。将来のリソースニーズを満たすことが予想されるインスタンスタイプを必ず選択します。
- グラフ化されたメトリクスを参考にして、実際の使用量がインスタンスの容量よりも低いかどうかを判断します。メトリクスデータ (平均、ピーク、パーセンタイル) を CloudWatch で表示し、EC2 インスタンスの推奨事項をさらに評価することもできます。例えば、一日の CPU パーセンテージメトリクスがどのように変化するか、ピークに対応する必要があるかどうかに注目します。詳細については、Amazon CloudWatch ユーザーガイドの「使用可能なメトリクスの表示」を参照してください。
- Compute Optimizer は、バーストパフォーマンスインスタンス (T3、T3a、および T2 インスタンス) の推奨事項を提供する場合があります。定期的にベースラインを超えてバーストする場合は、新しいインスタンスタイプの vCPU に基づいて引き続きバーストを実行できることを確認します。詳細については、バーストパフォーマンスインスタンスに関する主要な概念と定義 を参照してください。
- リザーブドインスタンスを購入した場合、オンデマンドインスタンスはリザーブドインスタンスとして請求される場合があります。現在のインスタンスタイプを変更する前に、まずリザーブドインスタンスの使用率と適用範囲に対する影響を評価します。
- 可能であれば、新世代のインスタンスへの交換を検討します。

- 別のインスタンスファミリーに移行する場合は、仮想化、アーキテクチャー、ネットワークタイプなどの点で、現在のインスタンスタイプと新しいインスタンスタイプに互換性があることを確認してください。詳細については、インスタンスタイプ変更の互換性 を参照してください。
- 最後に、推奨事項ごとに提供されるパフォーマンスリスク評価を検討します。パフォーマンスリスクは、推奨されるインスタンスタイプがワークロードのパフォーマンス要件を満たすかどうかを検証するために費やす必要のある作業量を示します。また、変更前と変更後に厳格な負荷テストおよびパフォーマンステストを行うことをお勧めします。

EC2 インスタンスのサイズを変更する際には、他の考慮事項があります。詳細については、インスタンスタイプを変更する を参照してください。

追加リソース

詳細については:

- インスタンスタイプ
- AWS Compute Optimizer ユーザーガイド

インスタンスタイプを変更する

ニーズが変わるにつれて、インスタンスの利用率が高すぎたり (インスタンスタイプが小さすぎる)、低すぎたりする (インスタンスタイプが大きすぎる) ことに気付く場合があります。この場合は、インスタンスタイプを変更することでインスタンスのサイズを変更できます。例えば、ワークロードに対して `t2.micro` インスタンスが小さすぎる場合は、`t2.large` などのより大きな T2 インスタンスタイプに変更することで、サイズを大きくすることができます。また、`m5.large` などの別のインスタンスタイプに変更することもできます。また、IPv6 への対応など、いくつかの機能を利用するために、前世代のインスタンスタイプから現世代のインスタンスタイプに変更することもできます。

既存のワークロードを処理するのに最適なインスタンスタイプにレコメンデーションが必要な場合は、AWS Compute Optimizer を使用することができます。詳細については、「既存のワークロード用インスタンスタイプに関する推奨事項の取得」を参照してください。

インスタンスタイプを変更する場合、新しいインスタンスタイプのレートの課金が開始されます。すべてのインスタンスタイプのオンデマンド料金については、「Amazon EC2 オンデマンド料金」を参照してください。

インスタンスタイプを変更せずにインスタンスにストレージを追加するには、EBS ボリュームをインスタンスに追加します。詳細については、「Amazon EBS ユーザーガイド」の「インスタンスへの Amazon EBS ボリュームのアタッチ」を参照してください。

どの手順に従うべきですか?

インスタンスタイプを変更するには、さまざまな手順があります。使用する手順は、インスタンスのルートボリューム、およびインスタントタイプがインスタンスの現在の設定と互換性があるかどうかによって異なります。互換性が決定される方法については、「インスタンスタイプ変更の互換性」を参照してください。

次の表を使用して、従う手順を決定します。

ルートボリューム	互換性	以下の手順に従います
EBS	互換性あり	既存の EBS-backed インスタンスのインスタンスタイプの変更
EBS	互換性なし	新しいインスタンスを起動してインスタンスタイプを変更する
インスタンスストア	該当しない	instance store-backed インスタンスのインスタンスタイプを変更する

互換性のあるインスタンスタイプに関する考慮事項

既存のインスタンスのインスタンスタイプを変更する前に、次の点を考慮してください。

- Amazon EBS–backed インスタンスで、インスタンスタイプを変更するには、そのインスタンスを先に停止する必要があります。インスタンスが停止している間のダウンタイムを予定しておいてください。インスタンスを停止し、インスタンスタイプの変更を行うと、数分かかります。インスタンスを再起動すると、アプリケーションの起動スクリプトによってかかる時間が変動する場合があります。詳細については、インスタンスの停止と起動 を参照してください。
- インスタンスを停止して起動すると、インスタンスは新しいハードウェアに移動されます。インスタンスにパブリック IPv4 アドレスがある場合には、このアドレスはリリースされて、インスタンスは新しいパブリック IPv4 アドレスになります。変更されないパブリック IPv4 アドレスが必要な場合は、Elastic IP アドレスを使用します。
- [Spot Instance] (スポットインスタンス) のインスタンスタイプを変更することはできません。

- インスタンスが Auto Scaling グループにある場合、Amazon EC2 Auto Scaling サービスはインスタンスを異常と判断して停止し、場合によってはそれを終了して代わりのインスタンスを起動します。インスタンスタイプを変更するときに、そのグループのスケーリングプロセスを中断することで、これを防ぐことができます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「スケーリングプロセスの中断と再開」を参照してください。
- NVMe インスタンスストアボリュームを使用してインスタンスのインスタンスタイプを変更すると、AMI またはインスタンスブロックデバイスマッピングで指定されていない場合でもすべての NVMe インスタンスストアボリュームが利用可能であるため、追加のインスタンスストアボリュームが更新されたインスタンスに存在するようになる可能性があります。それ以外の場合、更新したインスタンスには、元のインスタンスの起動時に指定したのと同じ数のインスタンスストアボリュームが設定されます。
- インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。インスタンスにすでにアタッチされているボリュームの数をサポートしていないインスタンスタイプまたはインスタンスサイズに変更することはできません。詳細については、「インスタンスボリューム数の制限」を参照してください。

既存の EBS-backed インスタンスのインスタンスタイプの変更

必要なインスタンスタイプがインスタンスの現在の設定と互換性がある場合は、次の手順を使用して EBS-backed インスタンスのインスタンスタイプを変更します。

Amazon EBS-backed インスタンスのインスタンスタイプを変更するには

1. (オプション) 新しいインスタンスのタイプに既存のインスタンスにインストールされていないドライバーが必要な場合は、インスタンスに接続してドライバーをインストールする必要があります。詳細については、インスタンスタイプ変更の互換性 を参照してください。
2. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
3. ナビゲーションペインで、[インスタンス] を選択します。
4. インスタンスを選択し、[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。確認を求められたら、[Stop] を選択します。インスタンスが停止するまで、数分かかる場合があります。
5. インスタンスが選択された状態で、[Actions (アクション)]、[Instance settings (インスタンス設定)]、[Change instance type (インスタンスタイプの変更)] の順に選択します。状態が `stopped` ではないインスタンスの場合、このオプションはグレー表示されています。
6. [Change instance type] (インスタンスタイプの変更) ページで、次の操作を行います。

a. [Instance type] (インスタンスタイプ) で、使用するインスタンスタイプを選択します。

   インスタンスタイプがリストにない場合、そのインスタンスタイプはインスタンスの設定と互換性がありません。代わりに、以下の手順をすべて行います。新しいインスタンスを起動してインスタンスタイプを変更する。

b. (オプション) 選択したインスタンスタイプが EBS 最適化をサポートしている場合は、[EBS-optimized] (EBS 最適化) を選択して EBS 最適化を有効にするか、[EBS-optimized] (EBS 最適化) を選択解除して EBS 最適化を無効にします。選択したインスタンスタイプがデフォルトで EBS に最適化されている場合、[EBS-optimized] (EBS 最適化) は選択状態になっており、これを選択解除することはできません。

c. [Apply] を選択して、新しい設定を受け入れます。

7. インスタンスを起動するには、インスタンスを選択後、[Instance state] (インスタンスの状態)、[Start instance] (インスタンスの開始) の順に選択します。インスタンスが `running` 状態になるまで、数分かかる場合があります。インスタンスが起動しない場合は、「インスタンスタイプ変更のトラブルシューティング」を参照してください。

新しいインスタンスを起動してインスタンスタイプを変更する

EBS-backed インスタンスの現在の設定に、使用する新しいインスタンスタイプとの互換性がない場合、元のインスタンスのインスタンスタイプを変更することはできません。代わりに、使用する新しいインスタンスタイプと互換性がある設定で新しいインスタンスを起動し、アプリケーションを新しいインスタンスに移行する必要があります。例えば、PV AMI から元のインスタンスを起動し、HVM AMI でのみサポートされている現行世代のインスタンスタイプに変更する場合は、HVM AMI から新しいインスタンスを起動する必要があります。互換性が決定される方法については、「インスタンスタイプ変更の互換性」を参照してください。

アプリケーションを新しいインスタンスに移行するには、以下を実行します。

- 元のインスタンスのデータをバックアップします。
- 使用する新しいインスタンスタイプと互換性がある設定で新しいインスタンスを起動して、元のインスタンスにアタッチされた EBS ボリュームをアタッチします。
- アプリケーションとソフトウェアを新しいインスタンスにインストールします。
- データを復元します。
- 元のインスタンスに Elastic IP アドレスがあり、ユーザーが新しいインスタンス上のアプリケーションを中断することなく継続して使用できるようにするには、Elastic IP アドレスを新しいイン

スタンスに関連付ける必要があります。詳細については、「Elastic IP アドレス」を参照してください。

新しいインスタンス設定のインスタンスタイプを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 次のように、保持する必要があるデータをバックアップします。
   - インスタンスストアボリューム上のデータについては、永続的ストレージにバックアップしてください。
   - EBS ボリューム上のデータについては、ボリュームのスナップショットを作成するか、インスタンスからボリュームをデタッチして、後で新しいインスタンスにアタッチできるようにします。
3. ナビゲーションペインで、[インスタンス] を選択します。
4. [Launch Instances] (インスタンスの起動) を選択します。インスタンスを設定する場合は、次の操作を行います。
   a. 使用するインスタンスタイプをサポートする AMI を選択します。現行世代のインスタンスタイプには HVM AMI が必要であることに注意してください。
   b. 使用する新しいインスタンスタイプを選択します。使用するインスタンスタイプが使用できない場合は、選択した AMI の設定と互換性がありません。
   c. Elastic IP アドレスを使用している場合は、元のインスタンスを現在実行している VPC を選択します。
   d. 同じトラフィックが新しいインスタンスに到達できるようにする場合は、元のインスタンスと関連付けられるセキュリティグループを選択します。
   e. 新しいインスタンスの設定が完了したら、手順を完了してキーペアを選択し、インスタンスを起動します。インスタンスが `running` 状態になるまで、数分かかる場合があります。
5. 必要に応じて、作成したスナップショットに基づく新しい EBS ボリュームや、元のインスタンスからデタッチした EBS ボリュームを新しいインスタンスにアタッチします。
6. アプリケーションと必要なソフトウェアを新しいインスタンスにインストールします。
7. 元のインスタンスのインスタンスストアボリュームからバックアップしたデータを復元します。
8. Elastic IP アドレスを使用している場合、以下のように新しいインスタンスにそのアドレスを割り当てます。
   a. ナビゲーションペインで [Elastic IP] を選択します。

b. 元のインスタンスに関連付ける Elastic IP アドレスを選択して、[Actions (アクション)]、[Disassociate Elastic IP address (Elastic IP アドレスの関連付けの解除)] の順に選択します。確認を求めるメッセージが表示されたら、[Disassociate (関連付け解除)] を選択します。

c. Elastic IP アドレスがまだ選択された状態で、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択します。

d. [リソースタイプ] で、[Instance (インスタンス)] を選択します。

e. [Instance] (インスタンス) で、Elastic IP アドレスを関連付ける新しいインスタンスを選択します。

f. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。

g. [Associate] を選択します。

9. (オプション) 不要になった場合は、元のインスタンスを終了できます。インスタンスを選択し、新しいインスタンスではなく元のインスタンスを終了させようとしていることを確認し (名前や起動時間の確認など)、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) を選択します。

インスタンスタイプ変更の互換性

インスタンスの現在の設定が、使用するインスタンスタイプと互換性がある場合にのみ、インスタンスタイプを変更できます。使用するインスタンスタイプに、インスタンスの現在の設定との互換性がない場合、インスタンスタイプと互換性がある設定で新しいインスタンスを起動し、アプリケーションを新しいインスタンスに移行する必要があります。

Windows インスタンスタイプを変更するための互換性に関する情報については、「Windows インスタンスのユーザーガイド」の「インスタンスタイプ変更の互換性」を参照してください。

> **Tip**
>
> `AWSSupport-MigrateXenToNitroLinux` Runbook を使用して、互換性のあるインスタンスを Xen インスタンスタイプから Nitro インスタンスタイプに移行できます。詳細については、「AWS Systems Manager Automation ランブックリファレンス」の「`AWSSupport-MigrateXenToNitroLinux runbook`」を参照してください。

互換性は、次の方法で決定されます。

仮想化タイプ

Linux AMI では、2 つの仮想化タイプ (準仮想化 (PV) およびハードウェア仮想マシン (HVM)) のどちらかを使用します。PV AMI から起動したインスタンスの場合、HVM のみのインスタンスタイプに変更することはできません。詳細については、Linux AMI 仮想化タイプ を参照してください。インスタンスの仮想化タイプを確認するには、Amazon EC2 コンソールで [Instances] (インスタンス) 画面の詳細ペインの [Virtualization] (仮想化) の値を参照します。

アーキテクチャ

AMI はプロセッサのアーキテクチャに固有であるため、プロセッサアーキテクチャが現在のインスタンスタイプと同じインスタンスタイプを選択する必要があります。次に例を示します。

- 現在のインスタンスタイプが、Arm アーキテクチャに基づくプロセッサである場合、対象となるインスタンスタイプは、Arm アーキテクチャベースのプロセッサ (C6g や M6g など) をサポートするものに制限されます。
- 32 ビット AMIs をサポートするのは以下のインスタンスタイプのみです。`t2.nano`、`t2.micro`、`t2.small`、`t2.medium`、`c3.large`、`t1.micro`、`m1.small`、`m1.me` および `c1.medium`。32 ビットインスタンスのインスタンスタイプを変更する場合は、これらのインスタンスタイプに制限されます。

ネットワークカード

インスタンスタイプによっては、複数のネットワークカードがサポートされているものもあります。新たに選択するインスタンスタイプでも、現在のインスタンスタイプと同じ数のネットワークカードをサポートしている必要があります。

拡張ネットワーク

拡張ネットワークをサポートするインスタンスタイプでは、必要なドライバーがインストールされていなければなりません。例えば、Nitro System に基づくインスタンスには、Elastic Network Adapter (ENA) ドライバーがインストールされた EBS-backed AMI が必要です。拡張ネットワーキングをサポートしていないインスタンスタイプから、拡張ネットワーキングをサポートするインスタンスタイプに変更するには、インスタンスに ENA ドライバーまたは ixgbevf ドライバーを必要に応じてインストールする必要があります。

> **Note**
>
> ENA Express を有効にしてインスタンスのサイズを変更する場合、新しいインスタンスタイプも ENA Express をサポートしている必要があります。ENA Express をサポートしているインスタンスタイプのリストは、「ENA Express でサポートされるインスタンスタイプ」を参照してください。

ENA Express をサポートするインスタンスタイプから ENA Express をサポートしないインスタンスタイプに変更するには、インスタンスをサイズ変更する前に、ENA Express が現在有効になっていないことを確認します。

NVMe

EBS ボリュームは、Nitro System上に構築されたインスタンスで NVMe ブロックデバイスとして公開されます。NVMe をサポートしないインスタンスタイプから NVMe をサポートするインスタンスタイプに変更するには、まずインスタンスに NVMe ドライバーをインストールする必要があります。また、ブロックデバイスマッピングで指定したデバイスの名前は、NVMe デバイス名 (`/dev/nvme[0-26]n1`) を使用して名称変更されます。したがって、`/etc/fstab` を使用してブート時にファイルシステムをマウントするには、デバイス名の代わりに UUID/Label を使用する必要があります。

ボリュームの制限

インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。詳細については、「インスタンスボリューム数の制限」を参照してください。

インスタンスタイプまたはインスタンスサイズに変更できるのは、現在インスタンスにアタッチされているボリュームと同じ数またはそれ以上のボリュームをサポートするものに限られます。現在アタッチされているボリュームの数をサポートしていないインスタンスタイプまたはインスタンスサイズに変更すると、リクエストは失敗します。例えば、32 個のボリュームが接続されている `m7i.4xlarge` インスタンスから、最大 27 個のボリュームをサポートする `m6i.4xlarge` インスタンスに変更すると、リクエストは失敗します。

AMI

拡張ネットワークと NVMe をサポートするインスタンスタイプで必要な AMI については、次のマニュアルの「リリースノート」を参照してください。

- 汎用インスタンス
- コンピュート最適化インスタンス
- メモリ最適化インスタンス
- ストレージ最適化インスタンス

インスタンスタイプ変更のトラブルシューティング

以下の情報は、インスタンスタイプの変更時に発生する可能性のある問題の診断と修復に役立ちます。

インスタンスタイプを変更してもインスタンスが起動しない

考えられる原因: 新しいインスタンスタイプの要件が満たされていない

インスタンスが起動しない場合、新しいインスタンスタイプの要件の 1 つが満たされていない可能性があります。詳細については、「タイプを変更した後、Linuxインスタンスが起動しなくなったのはなぜですか?」を参照してください。

考えられる原因: AMI がインスタンスタイプをサポートしていない

EC2 コンソールを使用してインスタンスタイプを変更する場合、選択した AMI でサポートされているインスタンスタイプのみを使用できます。しかし、AWS CLI を使ってインスタンスを起動すると、互換性のない AMI とインスタンスタイプを指定してしまうことがあります。AMI とインスタンスタイプに互換性がない場合、インスタンスは起動できません。詳細については、インスタンスタイプ変更の互換性 を参照してください。

考えられる原因: インスタンスがクラスタープレイスメントグループに属している

インスタンスがクラスタープレイスメントグループに属しており、インスタンスタイプを変更した後にインスタンスの起動に失敗した場合、以下を試してください。

1. クラスタープレイスメントグループ内のすべてのインスタンスを停止します。
2. 影響を受けるインスタンスのインスタンスタイプを変更します。
3. クラスタープレイスメントグループ内のすべてのインスタンスを起動します。

インスタンスタイプを変更した後、インターネットからアプリケーションまたはウェブサイトにアクセスできない

考えられる原因: パブリック IPv4 アドレスがリリースされている

インスタンスタイプを変更する場合は、まずインスタンスを停止する必要があります。インスタンスを停止すると、パブリック IPv4 アドレスがリリースされ、インスタンスに新しいパブリック IPv4 アドレスが付与されます。

インスタンスの停止と開始の間、パブリック IPv4 アドレスを保持するには、Elastic IP アドレスを使用することをお勧めします。インスタンスが実行されていれば、追加コストなしです。詳細については、Elastic IP アドレス を参照してください。

instance store-backed インスタンスのインスタンスタイプを変更する

instance store-backed インスタンスは、インスタンスストアのルートボリュームを持つインスタンスです。インスタンスストアのルートボリュームを持つインスタンスのインスタンスタイプを変更することはできません。代わりに、インスタンスから AMI を作成し、この AMI から新しいインスタンスを起動して、必要なインスタンスタイプを選択し、アプリケーションを新しいインスタンスに移行する必要があります。使用するインスタンスタイプは、作成した AMI と互換性があることが必要なことに注意してください。互換性が決定される方法については、「インスタンスタイプ変更の互換性」を参照してください。

アプリケーションを新しいインスタンスに移行するには、以下を実行します。

- 元のインスタンスのデータをバックアップします。
- 元のインスタンスから AMI を作成します。
- この AMI から新しいインスタンスを起動し、使用するインスタンスタイプを選択します。
- アプリケーションを新しいインスタンスにインストールします。
- 元のインスタンスに Elastic IP アドレスがあり、ユーザーが新しいインスタンス上のアプリケーションを中断することなく継続して使用できるようにするには、Elastic IP アドレスを新しいインスタンスに関連付ける必要があります。詳細については、「Elastic IP アドレス」を参照してください。

instance store-backed インスタンスのインスタンスタイプを変更するには

1. 次のように、保持する必要があるデータをバックアップします。
   - インスタンスストアボリューム上のデータについては、永続的ストレージにバックアップしてください。
   - EBS ボリューム上のデータについては、ボリュームのスナップショットを作成するか、インスタンスからボリュームをデタッチして、後で新しいインスタンスにアタッチできるようにします。
2. インスタンスから AMI を作成するには、「instance store-backed Linux AMI を作成する」に記載された前提条件と手順に従います。インスタンスから AMI を作成したら、この手順に戻ります。
3. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

4. ナビゲーションペインで [AMI] を選択します。フィルターリストで [Owned by me] (自己所有) を選択し、ステップ 2 で作成したイメージを選択します。[AMI name] (AMI 名) は、イメージを登録したときに指定した名前であり、[Source] (送信元) は Amazon S3 バケットです。

   > **Note**
   >
   > ステップ 2 で作成した AMI が表示されない場合は、AMI を作成したリージョンを選択していることを確認します。

5. AMI を選択した状態で、[Launch instance from image] (イメージからインスタンスを起動) を選択します。インスタンスを設定する場合は、次の操作を行います。

   a. 使用する新しいインスタンスタイプを選択します。使用するインスタンスタイプが使用できない場合は、作成した AMI の設定と互換性がありません。詳細については、インスタンスタイプ変更の互換性 を参照してください。

   b. Elastic IP アドレスを使用している場合は、元のインスタンスを現在実行している VPC を選択します。

   c. 同じトラフィックが新しいインスタンスに到達できるようにする場合は、元のインスタンスと関連付けられるセキュリティグループを選択します。

   d. 新しいインスタンスの設定が完了したら、手順を完了してキーペアを選択し、インスタンスを起動します。インスタンスが `running` 状態になるまで、数分かかる場合があります。

6. 必要に応じて、作成したスナップショットに基づく新しい EBS ボリュームや、元のインスタンスからデタッチした EBS ボリュームを新しいインスタンスにアタッチします。

7. アプリケーションと必要なソフトウェアを新しいインスタンスにインストールします。

8. Elastic IP アドレスを使用している場合、以下のように新しいインスタンスにそのアドレスを割り当てます。

   a. ナビゲーションペインで [Elastic IP] を選択します。

   b. 元のインスタンスに関連付ける Elastic IP アドレスを選択して、[Actions (アクション)]、[Disassociate Elastic IP address (Elastic IP アドレスの関連付けの解除)] の順に選択します。確認を求めるメッセージが表示されたら、[Disassociate (関連付け解除)] を選択します。

   c. Elastic IP アドレスがまだ選択された状態で、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択します。

   d. [リソースタイプ] で、[Instance (インスタンス)] を選択します。

e. [Instance] (インスタンス) で、Elastic IP アドレスを関連付ける新しいインスタンスを選択します。

f. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。

g. [Associate] を選択します。

9. (オプション) 不要になった場合は、元のインスタンスを終了できます。インスタンスを選択し、新しいインスタンスではなく元のインスタンスを終了させようとしていることを確認し (名前や起動時間の確認など)、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) を選択します。

Amazon EC2 Mac インスタンス

Amazon EC2 Mac インスタンスは macOS オペレーティングシステムをネイティブでサポートしています。

- EC2 x86 Mac インスタンス (`mac1.metal`)は 2018 Mac mini ハードウェア上に構築され、3.2 GHz Intel 第 8 世代 (Coffee Lake) Core i7 プロセッサを搭載しています。
- EC2 M1 Mac インスタンス (`mac2.metal`) は Apple Silicon M1 プロセッサを搭載した 2020 Mac mini ハードウェアに構築されています。
- EC2 M2 Mac インスタンス (`mac2-m2.metal`) は Apple シリコン M2 プロセッサを搭載した 2023 Mac mini ハードウェアに構築されています。
- EC2 M2 Pro Mac インスタンス (`mac2-m2pro.metal`) は Apple Silicon M2 Pro プロセッサを搭載した 2023 Mac mini ハードウェアに構築されています。

EC2 Mac インスタンスは、iPhone、iPad、Mac、Vision Pro、Apple Watch、Apple TV、Safari などの Apple プラットフォーム用アプリケーションの開発、構築、テスト、署名に最適です。Mac インスタンスには、SSH または Apple Remote Desktop (ARD) を使用して接続できます。

> **Note**
>
> [unit of billing] (請求の単位) は [dedicated host] (専有ホスト) です。そのホストで実行されているインスタンスには追加料金はかかりません。

コンテンツ

考慮事項

Mac インスタンスには、次の考慮事項が適用されます。

- Mac インスタンスは、Dedicated Hosts のベアメタルインスタンスとしてのみ使用でき、Dedicated Host をリリースできる前の最低割り当て期間は 24 時間です。Dedicated Host ごとに 1 つの Mac インスタンスを起動できます。Dedicated Host は、AWS アカウント、 AWS 組織内の 組織単位、あるいは AWS 組織全体と共有することができます。
- Mac インスタンスは オンデマンドインスタンス としてのみ使用できます。スポットインスタンス または リザーブドインスタンス では使用できません。Savings Plan を購入すると、Mac インスタンスでコストを節約できます。
- Mac インスタンスでは、次のいずれかのオペレーティングシステムを実行できます。
  - macOS Mojave (バージョン 10.14) (x86 Mac インスタンスのみ)
  - macOS Catalina (バージョン 10.15) (x86 Mac インスタンスのみ)
  - macOS Big Sur (バージョン 11) (x86 および M1 Mac インスタンス)
  - macOS Monterey (バージョン 12) (x86 および M1 Mac インスタンス)

  - macOS Ventura (バージョン 13) (すべての Mac インスタンス、M2 および M2 Pro Mac インスタンスは macOS Ventura バージョン 13.2 以降をサポート)
  - macOS Sonoma (バージョン 14) (すべての Mac インスタンス)
- EBS ホットプラグはサポートされています。
- AWS は、Apple ハードウェアの内部 SSD を管理またはサポートしません。代わりに、Amazon EBS ボリュームを使用することを強くお勧めします。EBS ボリュームは、他の EC2 インスタンスと同じ伸縮自在性、可用性、耐久性の利点を Mac インスタンスでもたらします。
- Mac インスタンスで EBS のパフォーマンスを最適化するには、汎用 SSD (gp2 と gp3) およびプロビジョンド IOPS SSD (io1 と io2) の併用をお勧めします。
- Mac インスタンスは、Amazon EC2 Auto Scaling をサポートしています。
- x86 Mac インスタンスでは、自動ソフトウェア更新は無効化されています。インスタンスを本番環境に置く前に、更新を適用し、インスタンスでテストすることをお勧めします。詳細については、オペレーティングシステムとソフトウェアの更新 を参照してください。
- Mac インスタンスを停止または終了すると、Dedicated Host でスクラブワークフローが実行されます。詳細については、Mac インスタンスの停止と終了 を参照してください。

> **Warning**
>
> FileVault は使用しません。パーティションがロックされているので、FileVault を有効にするとホストの起動に失敗します。データ暗号化が必要な場合は、Amazon EBS 暗号化を使用して、ブートの問題やパフォーマンスへの影響を回避します。Amazon EBS の暗号化では、暗号化オペレーションはインスタンスをホストするサーバー上で実行されるため、インスタンスとそれにアタッチされた EBS ストレージ間に保管中のデータと転送中のデータの両方のセキュリティが確保されます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

インスタンスの準備状況

Mac インスタンスを起動したら、インスタンスに接続する前に、インスタンスの準備が整うまで待機する必要があります。x86 Mac インスタンスまたは Apple Silicon Mac インスタンスを含む AWS 提供の AMI の場合、起動時間は約 6 分から 20 分の範囲です。選択した Amazon EBS のボリュームサイズ、ユーザーデータへの追加スクリプトの導入、カスタム macOS AMI への追加ロードソフトウェアなどにより、起動時間が長くなる場合があります。

以下のような小さなシェルスクリプトを使用すれば、describe-instance-status API をポーリングし、インスタンスが接続できる状態になったかどうかを確認できます。次のコマンドで、インスタンス ID の例を独自の インスタンス ID に置き換えます。

```
for i in $(seq 1 200); do aws ec2 describe-instance-status --instance-
ids=i-0123456789example \
    --query='InstanceStatuses[0].InstanceStatus.Status'; sleep 5; done;
```

Mac インスタンスの作成

EC2 Mac インスタンスには専有ホストが必要です。まず、アカウントにホストを割り当ててから、そのホストにインスタンスを作成する必要があります。

AWS Management Console または AWS CLI を使用して Mac インスタンスを起動できます。

コンソールを使用した Mac インスタンスの起動

Mac インスタンスを Dedicated Host 上で起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 以下のように専有ホストを割り当てます。
    a. ナビゲーションペインで [Dedicated Hosts] を選択します。
    b. [Dedicated Host の割り当て] を選択し、次の操作を行います。
        i. [インスタンスファミリー] で、[mac1]、[mac2]、[mac2-m2]、[mac2-m2pro] のいずれかを選択します。リストにインスタンスファミリーが表示されない場合、それは、現在選択されているリージョンではサポートされていません。
        ii. [インスタンスタイプ] で、選択したインスタンスファミリーに基づき[mac1.metal]、[mac2.metal]、[mac2-m2.metal]、[mac2-m2pro.metal] のいずれかを選択します。
        iii. [アベイラビリティーゾーン] で、専有ホストのアベイラビリティーゾーンを選択します。
        iv. [Quantity] (数量) は [1] のままにします。
        v. [割り当て] を選択します。
3. 次のように、ホストにインスタンスを作成します。
    a. 作成した Dedicated Host を選択し、次の操作を実行します。

i. [Action] (アクション)、[Launch instance(s) onto host] (ホストにインスタンスを作成) の順に選択します。

ii. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、macOS AMI を選択します。

iii. [インスタンスタイプ] で、適切なインスタンスタイプ ([mac1.metal]、[mac2.metal]、[mac2-m2.metal]、[mac2-m2pro.metal] のいずれか) を選択します。

iv. [Advanced details] (詳細設定) で、[Tenancy] (テナンシー)、[Tenancy host by] (テナンシーホスト)、[Tenancy host ID] (テナンシーホスト ID) が、作成した専有ホストに基づいて事前設定されていることを確認します。必要に応じて [Tenancy affinity] (テナンシーのアフィニティ) を更新します。

v. ウィザードを完了し、必要に応じて EBS ボリューム、セキュリティグループ、およびキーペアを指定します。

vi. [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

b. 確認ページは、インスタンスが起動中であることを通知します。[View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。インスタンスの初期状態は `pending` です。インスタンスの状態が `running` に変わると、ステータスチェックに合格すると、インスタンスは準備完了になります。

AWS CLI を使用した Mac インスタンスの起動

専有ホストを割り当てる

次の allocate-hosts コマンドを使用して、お使いの Mac インスタンスに専有ホストを割り当て、`instance-type` を `mac1.metal`、`mac2.metal`、`mac2-m2.metal`、`mac2-m2pro.metal` のいずれかに置き換え、`region`と `availability-zone` をお使いの環境に適したものに置き換えます。

```
aws ec2 allocate-hosts --region us-east-1 --instance-type mac1.metal --availability-
zone us-east-1b --auto-placement "on" --quantity 1
```

ホスト上でインスタンスを起動する

次の run-instances コマンドを使用して Mac インスタンスを起動します。ここでも、instance-type を mac1.metal、mac2.metal、mac2-m2.metal、mac2-m2pro.metal のいずれかに置き換え、region と availability-zone を以前使用したものに置き換えます。

```
aws ec2 run-instances --region us-east-1 --instance-type mac1.metal --placement
 Tenancy=host --image-id ami_id --key-name my-key-pair
```

インスタンスの初期状態は pending です。インスタンスの状態が running に変わると、ステータスチェックに合格すると、インスタンスは準備完了になります。インスタンスのステータス情報を表示するには、次の describe-instance-status コマンドを使用します。

```
aws ec2 describe-instance-status --instance-ids i-017f8354e2dc69c4f
```

次に、ステータスチェックに合格した実行中のインスタンスの出力例を示します。

```
{
    "InstanceStatuses": [
        {
            "AvailabilityZone": "us-east-1b",
            "InstanceId": "i-017f8354e2dc69c4f",
            "InstanceState": {
                "Code": 16,
                "Name": "running"
            },
            "InstanceStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            },
            "SystemStatus": {
                "Details": [
                    {
                        "Name": "reachability",
                        "Status": "passed"
                    }
                ],
                "Status": "ok"
            }
```

```
        }
    ]
}
```

Mac インスタンスへ接続する

SSH またはグラフィカルユーザーインターフェイスを使用して Mac インスタンスに接続できます。

SSH を使用したインスタンスへの接続

> **Important**
>
> 複数のユーザーが同時に OS にアクセスできます。ポート 5900 の画面共有サービスが組み込まれているため、通常、1:1 user:GUI セッションがあります。macOS 内で SSH を使用すると、sshd_config ファイルの [最大セッション] クォータまで、複数のセッションがサポートされます。

Amazon EC2 Mac インスタンスは、デフォルトではリモートルート SSH を許可しません。パスワード認証は、パスワードのブルートフォース攻撃を防ぐために無効になっています。ec2-user アカウントは、SSH を使用してリモートでログインするように設定されています。ec2-user アカウントにも sudo 権限があります。インスタンスに接続したら、他のユーザーを追加できます。

SSH を使用したインスタンスへの接続をサポートするには、SSH アクセスを許可するキーペアとセキュリティグループを使用してインスタンスを起動し、インスタンスにインターネット接続があることを確認します。インスタンスに接続するときに、キーペアの .pem ファイルを指定します。

SSH クライアントを使用して Mac インスタンスに接続するには、次の手順に従います。インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

SSH を使用してインスタンスに接続するには

1. コマンドラインで「ssh」と入力して、ローカルコンピュータに SSH クライアントがインストールされていることを確認します。コンピュータがコマンドを認識しない場合は、お使いのオペレーティングシステム用の SSH クライアントを検索し、インストールします。
2. インスタンスのパブリック DNS 名を取得します。Amazon EC2 コンソールを使用して、[詳細] タブと [ネットワーク] タブの両方でパブリック DNS 名を確認できます。AWS CLI を使用して、describe-instances コマンドを使用してパブリック DNS 名を見つけることができます。

3. インスタンスの起動時に指定したキーペアの .pem ファイルを見つけます。
4. 次の ssh コマンドを使用してインスタンスに接続し、インスタンスのパブリック DNS 名と .pem ファイルを指定します。

```
ssh -i /path/key-pair-name.pem ec2-user@instance-public-dns-name
```

インスタンスのグラフィカルユーザーインターフェイス (GUI) に接続する

VNC、Apple Remote Desktop (ARD)、または Apple Screen Sharing アプリケーション (macOS に付属) を使用してインスタンスの GUI に接続するには、以下の手順に従います。

> **Note**
>
> macOS 10.14 以降では、画面共有がシステム環境設定で有効になっている場合のみ制御できます。

ARD クライアントまたは VNC クライアントを使用してインスタンスに接続するには

1. ローカルコンピュータに、ARD をサポートしている ARD クライアントまたは VNC クライアントが、インストールされていることを確認します。macOS では、組み込みの画面共有アプリケーションを利用できます。インストールされていない場合は、お使いのオペレーティングシステム向けの ARD を検索し、インストールします。
2. ローカルコンピュータから、SSH を使用してインスタンスに接続します。
3. 次のように passwd コマンドを使用して、ec2-user アカウントのパスワードを設定します。

```
[ec2-user ~]$ sudo passwd ec2-user
```

4. 次のコマンドを使用して macOS スクリーン共有をインストールして起動します。

```
[ec2-user ~]$ sudo launchctl enable system/com.apple.screensharing
sudo launchctl load -w /System/Library/LaunchDaemons/com.apple.screensharing.plist
```

5. exit と入力して Enter キーを押して、インスタンスとの接続を切断します。
6. コンピュータから、次の ssh コマンドを使用してインスタンスに接続します。前のセクションで示したオプションに加えて、ポート転送を有効にしローカルポート 5900 のすべてのトラフィックをインスタンスの ARD サーバーに転送する場合は、-L オプションを使用します。

```
ssh -L 5900:localhost:5900 -i /path/key-pair-name.pem ec2-user@instance-public-dns-name
```

7. ローカルコンピュータから、ARD をサポートしている ARD クライアントまたは VNC クライアントを使用して、`localhost:5900` に接続します。例えば、macOS で画面共有アプリケーションを次のように使用します。

   a. [検索] を開いて、[移動] を選択します。

   b. [サーバーに接続] を選択します。

   c. [サーバーアドレス] フィールドに、`vnc://localhost:5900` と入力します。

   d. プロンプトに従って、**ec2-user** を使用して ec2-user アカウント用に作成したユーザ名とパスワードでログインします。

Mac インスタンスで macOS の画面解像度を変更する

ARD または ARD をサポートする VNC クライアントを使用して EC2 Mac インスタンスに接続したら、displayplacer などの公開されている macOS ツールやユーティリティのいずれかを使用して macOS 環境の画面解像度を変更できます。

displayplacer を使用して画面解像度を変更するには

1. displayplacer をインストールします。

```
[ec2-user ~]$ brew tap jakehilborn/jakehilborn && brew install displayplacer
```

2. 現在の画面情報と可能な画面解像度を表示します。

```
[ec2-user ~]$ displayplacer list
```

3. 希望の画面解像度を適用します。

```
[ec2-user ~]$ displayplacer "id:<screenID> res:<width>x<height> origin:(0,0)
 degree:0"
```

   例:

```
RES="2560x1600"
```

```
displayplacer "id:69784AF1-CD7D-B79B-E5D4-60D937407F68 res:${RES} scaling:off
 origin:(0,0) degree:0"
```

EC2 macOS AMI

Amazon EC2 macOS は、Amazon EC2 Mac インスタンスで実行されるデベロッパーワークロードに対して、安定性、安全性、高パフォーマンスの環境を実現するように設計されています。EC2 macOS AMI には、起動設定ツールや一般的な AWS ライブラリやツールなど、AWS との統合を容易にするパッケージが含まれています。

EC2 macOS AMI には、デフォルトで以下が含まれます。

- ENA ドライバー
- EC2 macOS Init
- macOS 用の SSM Agent
- macOS の EC2 System Monitoring (x86 Mac インスタンスのみ)
- AWS Command Line Interface (AWS CLI) バージョン 2
- Xcode 用のコマンドラインツール
- Homebrew
- EC2 Instance Connect

AWS は、更新済みの EC2 macOS AMI を定期的に提供します。これには、AWS が所有するパッケージの更新と、完全にテストされた macOS の最新バージョンが含まれます。さらに、AWS は更新された AMI を提供し、さらに完全にテストおよび検証できるとすぐに最新のマイナーバージョンアップデートやメジャーバージョンアップデートを提供します。Mac インスタンスのデータやカスタマイズ情報を保持する必要がない場合は、現在の AMI を使用して新しいインスタンスを起動してから、前のインスタンスを終了することで、最新の更新プログラムを取得できます。また、Mac インスタンスに適用するアップデートを選択できます。

オペレーティングシステムとソフトウェアの更新

Warning

ベータ版またはプレビュー版の macOS バージョンのインストールは、Amazon EC2 M1 Mac インスタンスでのみ可能です。Amazon EC2 は macOS のベータ版やプレビュー版をサ

ポートしていないため、実稼働前の macOS バージョンに更新した後もインスタンスが機能し続けることは保証されません。
ベータ版またはプレビュー版の macOS バージョンを Amazon EC2 にインストールすると、インスタンスの停止または終了時に、x86 Mac インスタンスが Amazon EC2 Mac 専有ホストのパフォーマンスを低下させるため、そのホストで新しいインスタンスを開始または起動できなくなります。

x86 Mac インスタンスと Apple シリコン Mac インスタンスでソフトウェアを更新する手順

- x86 Mac インスタンスでのソフトウェアの更新
- Apple Silicon Mac インスタンスでソフトウェアを更新する

x86 Mac インスタンスでのソフトウェアの更新

x86 Mac インスタンスでは、`softwareupdate` コマンドを使用して、Apple からオペレーティングシステムの更新をインストールできます。

x86 Mac インスタンスで Apple からオペレーティングシステムの更新プログラムをインストールするには

1. 次のコマンドを使用して、利用可能な更新プログラムを含むパッケージを一覧表示します。

```
[ec2-user ~]$ softwareupdate --list
```

2. すべての更新プログラムをインストールするか、特定の更新プログラムのみをインストールします。特定の更新プログラムをインストールするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo softwareupdate --install label
```

すべての更新プログラムをインストールするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo softwareupdate --install --all --restart
```

システム管理者は、AWS Systems Manager を使用することで、事前に承認されたオペレーティングシステムの更新を x86 Mac インスタンスにロールアウトできます。詳細については、AWS Systems Manager ユーザーガイドを参照してください。

Homebrew を使用して、EC2 macOS AMI にパッケージへの更新プログラムをインストールします。これにより、インスタンスでこのパッケージの最新バージョンを使用できます。また、Homebrew を使用して Amazon EC2 macOS に共通の macOS アプリケーションをインストールして実行することもできます。詳細については、Homwbrew ドキュメントを参照してください。

Homebrew を使用して更新プログラムをインストールするには

1. 次のコマンドを使用して Homwbrew を更新します。

```
[ec2-user ~]$ brew update
```

2. 次のコマンドを使用して、利用可能な更新プログラムを含むパッケージを一覧表示します。

```
[ec2-user ~]$ brew outdated
```

3. すべての更新プログラムをインストールするか、特定の更新プログラムのみをインストールします。特定の更新プログラムをインストールするには、次のコマンドを使用します。

```
[ec2-user ~]$ brew upgrade package name
```

すべての更新プログラムをインストールするには、次のコマンドを使用します。

```
[ec2-user ~]$ brew upgrade
```

Apple Silicon Mac インスタンスでソフトウェアを更新する

考慮事項

Elastic Network Adapter (ENA) ドライバー

ネットワークドライバー設定が更新されたため、ENA ドライバーバージョン 1.0.2 は macOS 13.3 以降と互換性がありません。ベータ版、プレビュー版、または実稼働版の macOS バージョン 13.3 以降をインストールする必要があり、最新の ENA ドライバーをインストールしていない場合は、次の手順を使用して新しいバージョンのドライバーをインストールします。

ENA ドライバーの新しいバージョンをインストールするには

1. ターミナルウィンドウで、SSH を使用して Apple Silicon Mac インスタンスに接続します。

2. 次のコマンドを使用して、ENA Applications アプリケーションをファイルにダウンロードします。

```
[ec2-user ~]$ brew install amazon-ena-ethernet-dext
```

> **トラブルシューティングのヒント**
>
> 警告が表示されたら No available formula with the name amazon-ena-ethernet-dext、次のコマンドを実行します。
>
> ```
> [ec2-user ~]$ brew update
> ```

3. exit と入力して return キーを押して、インスタンスとの接続を切断します。
4. VNC クライアントを使用して ENA アプリケーションをアクティブ化します。
   a. インスタンスのグラフィカルユーザーインターフェイス (GUI) に接続する を使用して VNC クライアントを設定します。
   b. 画面共有アプリケーションを使用してインスタンスに接続したら、Applications フォルダーに移動して ENA アプリケーションを開きます。
   c. [Activate] を選択します。
   d. ドライバーが正しくアクティブ化されたことを確認するには、ターミナルウィンドウで次のコマンドを実行します。コマンドの出力は、古いドライバが終了状態で、新しいドライバがアクティブ状態であることを示しています。

```
systemextensionsctl list;
```

   e. インスタンスを再起動すると、新しいドライバーのみが表示されます。

Apple Silicon Mac インスタンスでのソフトウェアの更新

Apple Silicon Mac インスタンスでは、オペレーティングシステムのインプレースアップデートを実行するために数ステップの手順を実行する必要があります。最初に、VNC (仮想ネットワークコンピューティング) クライアントで GUI を使用してインスタンスの内部ディスクにアクセスします。この手順では、組み込みの VNC クライアントである macOS 画面共有を使用します。次に、Amazon EBS ボリュームに aws-managed-user としてサインインして、管理ユーザー (ec2-user) に所有権を委任します。

この手順を進めると、2 つのパスワードが作成されます。1 つのパスワードは管理ユーザー (ec2-user) 用で、もう 1 つのパスワードは特別な管理ユーザー (aws-managed-user) 用です。これらのパスワードは手順を進めるときに使用しますので、覚えておいてください。

> **Note**
>
> macOS Big Sur でこの手順を実行すると、macOS Big Sur 11.7.3 から macOS Big Sur 11.7.4 へのアップデートなど、マイナーアップデートのみしか実行できません。macOS Monterey 以降では、主要なソフトウェアアップデートを実行できます。

内部ディスクにアクセスするには

1. ローカルコンピュータのターミナルで、次のコマンドで SSH を使用して Apple Silicon Mac インスタンスに接続します。詳細については、「SSH を使用したインスタンスへの接続」を参照してください。

   ```
   ssh -i /path/key-pair-name.pem ec2-user@instance-public-dns-name
   ```

2. 次のコマンドを使用して macOS スクリーン共有をインストールして起動します。

   ```
   [ec2-user ~]$ sudo launchctl enable system/com.apple.screensharing
   sudo launchctl load -w /System/Library/LaunchDaemons/com.apple.screensharing.plist
   ```

3. 次のコマンドを実行して、ec2-user のパスワードを設定します。パスワードは後で使用するので覚えておいてください。

   ```
   [ec2-user ~]$ sudo /usr/bin/dscl . -passwd /Users/ec2-user
   ```

4. exit と入力して return キーを押して、インスタンスとの接続を切断します。
5. ローカルコンピュータのターミナルで、次のコマンドを使用して VNC ポートへの SSH トンネルを使用してインスタンスに再接続します。

   ```
   ssh -i /path/key-pair-name.pem -L 5900:localhost:5900 ec2-user@instance-public-dns-name
   ```

> **Note**
>
> 次の VNC 接続と GUI の手順が完了するまで、この SSH セッションを終了しないでください。インスタンスを再起動すると、接続は自動的に終了します。

6. ローカルコンピュータから、次の手順を使用して `localhost:5900` に接続します。

   a. [検索] を開いて、[移動] を選択します。

   b. [サーバーに接続] を選択します。

   c. [サーバーアドレス] フィールドに、`vnc://localhost:5900` と入力します。

7. macOS ウィンドウで、ステップ 3 で作成したパスワードを使用して、`ec2-user` として Apple Silicon Mac インスタンスのリモートセッションに接続します。

8. 次のいずれかのオプションを使用して、InternalDisk という名前の内部ディスクにアクセスします。

   a. macOS Ventura 以上の場合: [システム設定] を開き、左側のペインで [一般] を選択し、ペインの右下で [起動ディスク] を選択します。

   b. macOS Monterey 以下の場合: [システム環境設定] を開いて、[起動ディスク] を選択し、ウィンドウの左下にあるロックアイコンを選択してペインのロックを解除します。

> **トラブルシューティングのヒント**
>
> 内部ディスクをマウントする必要がある場合は、ターミナルで次のコマンドを実行します。
>
> ```
> APFSVolumeName="InternalDisk" ; SSDContainer=$(diskutil list | grep
>  "Physical Store disk0" -B 3 | grep "/dev/disk" | awk {'print $1'} ) ;
>  diskutil apfs addVolume $SSDContainer APFS $APFSVolumeName
> ```

9. InternalDisk という名前の内部ディスクを選択し、[再起動] を選択します。メッセージが表示されたら、もう一度 [再起動] を選択します。

> **Important**
>
> 内部ディスクの名前が InternalDisk ではなく Macintosh HD の場合は、専有ホストを更新できるようにインスタンスを停止して再起動する必要があります。詳細については、「Mac インスタンスの停止と終了」を参照してください。

管理ユーザーに所有権を委任するには、次の手順に従います。SSH でインスタンスに再接続すると、特別な管理ユーザー (aws-managed-user) を使用して内部ディスクから起動されます。aws-managed-user 用の初期パスワードは空白なため、最初の接続時に上書きする必要があります。その後、ブートボリュームが変更されたため、手順を繰り返して macOS の画面共有をインストールして起動する必要があります。

Amazon EBS ボリュームの管理者に所有権を委任するには

1. ローカルコンピュータのターミナルで、次のコマンドを使用して Apple Silicon Mac インスタンスに接続します。

   ```
   ssh -i /path/key-pair-name.pem aws-managed-user@instance-public-dns-name
   ```

2. WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! の警告が表示されたら、以下のいずれかのコマンドを使用してこの問題を解決します。

   a. 次のコマンドを使用して、既知のホストを削除します。次に、前の手順を繰り返します。

      ```
      rm ~/.ssh/known_hosts
      ```

   b. 前の手順の SSH コマンドに、次の形式を追加します。

      ```
      -o UserKnownHostsFile=/dev/null -o StrictHostKeyChecking=no
      ```

3. 次のコマンドを実行して、aws-managed-user のパスワードを設定します。aws-managed-user 初期パスワードは空白であるため、最初の接続時に上書きする必要があります。

   a.
      ```
      [aws-managed-user ~]$ sudo /usr/bin/dscl . -passwd /Users/aws-managed-
      user password
      ```

b. プロンプトが表示されたら、`Permission denied. Please enter user's old password:`、Enter キーを押します。

> **トラブルシューティングのヒント**
>
> `passwd: DS error: eDSAuthFailed` のエラーが発生した場合は、次のコマンドを使用します。
>
> ```
> [aws-managed-user ~]$ sudo passwd aws-managed-user
> ```

4. 次のコマンドを使用して macOS スクリーン共有をインストールして起動します。

```
[aws-managed-user ~]$ sudo launchctl enable system/com.apple.screensharing
sudo launchctl load -w /System/Library/LaunchDaemons/com.apple.screensharing.plist
```

5. exit と入力して return キーを押して、インスタンスとの接続を切断します。
6. ローカルコンピュータのターミナルで、次のコマンドを使用して VNC ポートへの SSH トンネルを使用してインスタンスに再接続します。

```
ssh -i /path/key-pair-name.pem -L 5900:localhost:5900 aws-managed-user@instance-public-dns-name
```

7. ローカルコンピュータから、次の手順を使用して `localhost:5900` に接続します。

   a. [検索] を開いて、[移動] を選択します。

   b. [サーバーに接続] を選択します。

   c. [サーバーアドレス] フィールドに、`vnc://localhost:5900` と入力します。

8. macOS ウィンドウで、ステップ 3 で作成したパスワードを使用して、`aws-managed-user` として Apple Silicon Mac インスタンスのリモートセッションに接続します。

> **Note**
>
> Apple ID でサインインするように求めるメッセージが表示されたら、[後でセットアップ] を選択します。

9. Amazon EBS ボリュームには、次のいずれかのオプションを使用してアクセスします。

a. macOS Ventura 以降の場合: [システム設定] を開き、左側のペインで [一般] を選択し、ペインの右下で [起動ディスク] を選択します。

b. macOS Monterey 以前の場合: [システム環境設定] を開き、[起動ディスク] を選択し、ウィンドウの左下にあるロックアイコンを使用してペインのロックを解除します。

> **Note**
>
> 再起動するまで、管理者パスワードの入力を求められたら、上記で設定した aws-managed-user 用のパスワードを使用してください。このパスワードは、ec2-user 用に設定したパスワードやインスタンスのデフォルトの管理者アカウントとは異なる場合があります。以下の手順では、インスタンスの管理者パスワードをいつ使用するかを指定します。

10. Amazon EBS ボリューム ([起動ディスク] ウィンドウの InternalDisk という名前が付いていないボリューム) を選択し、[再起動] を選択します。

> **Note**
>
> Apple Silicon Mac インスタンスに複数の起動可能な Amazon EBS ボリュームがアタッチされている場合は、必ず各ボリュームにそれぞれ固有の名前を使用してください。

11. 再起動を確認し、プロンプトが表示されたら [ユーザーを認証] を選択します。
12. [このボリュームペインのユーザーを認証] で、管理者ユーザー (デフォルトで ec2-user) が選択されていることを確認し、[承認] を選択します。
13. 前の手順の手順 3 で作成した ec2-user パスワードを入力し、[続行] を選択します。
14. プロンプトが表示されたら、特別管理ユーザー (aws-managed-user) のパスワードを入力します。
15. ローカルコンピュータからターミナルで、ec2-user ユーザー名を使用して SSH を使用してインスタンスに再接続します。

> **トラブルシューティングのヒント**
>
> WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED! の警告が表示されたら、次のコマンドを実行し、SSH を使用してインスタンスに再接続します。

```
rm ~/.ssh/known_hosts
```

16. ソフトウェアアップデートを実行するには、x86 Mac インスタンスでのソフトウェアの更新 の下にあるコマンドを使用します。

EC2 macOS Init

EC2 macOS Init は、起動時に EC2 Mac インスタンスを初期化するために使用します。優先順位グループを使用して、タスクの論理グループを同時に実行します。

launchd plist ファイルは `/Library/LaunchDaemons/com.amazon.ec2.macos-init.plist` です。EC2 macOS Init 用のファイルは、`/usr/local/aws/ec2-macos-init` にあります。

詳細については、https://github.com/aws/ec2-macos-init を参照してください。

macOS 用の EC2 System Monitoring

macOS 用の EC2 System Monitoring は、Amazon CloudWatch で CPU 使用率メトリクスを使用できるようにします。このメトリクスは、カスタムシリアルデバイス経由で 1 分間隔で CloudWatch に送信されます。このエージェントを有効または無効にするには、次の手順に従います。このエージェントは、デフォルトでは有効になっています。

```
sudo setup-ec2monitoring [enable | disable]
```

> **Note**
>
> macOS の EC2 System Monitoring は、現在、Apple Silicon Mac インスタンスではサポートされていません。

Mac インスタンスの EBS ボリュームのサイズを増やす

Mac インスタンスの Amazon EBS ボリュームのサイズを増やすことができます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS Elastic Volumes」を参照してください。

ボリュームのサイズを大きくした後、APFS コンテナのサイズを以下のように大きくする必要があります。

使用可能なディスク容量を増やす

1. 再起動が必要かどうかを判断します。実行中の Mac インスタンスで既存の EBS ボリュームのサイズを変更した場合、新たなサイズを使用可能にするには、そのインスタンスを再起動する必要があります。起動中にディスク領域の変更が行われた場合、再起動は必要ありません。

   ディスクサイズに関する現在のステータスを表示します。

   ```
   [ec2-user ~]$  diskutil list external physical
   /dev/disk0 (external, physical):
      #:                       TYPE NAME                    SIZE       IDENTIFIER
      0:                 GUID_partition_scheme            *322.1 GB     disk0
      1:                 EFI EFI                           209.7 MB     disk0s1
      2:                 Apple_APFS Container disk2        321.9 GB     disk0s2
   ```

2. 以下のコマンドをコピーして貼り付けます。

   ```
   [ec2-user ~]$ PDISK=$(diskutil list physical external | head -n1 | cut -d" " -f1)
   APFSCONT=$(diskutil list physical external | grep "Apple_APFS" | tr -s " " | cut -
   d" " -f8)
   yes | sudo diskutil repairDisk $PDISK
   ```

3. 以下のコマンドをコピーして貼り付けます。

   ```
   [ec2-user ~]$ sudo diskutil apfs resizeContainer $APFSCONT 0
   ```

Mac インスタンスの停止と終了

Mac インスタンスを停止すると、インスタンスは、stopping 状態に入るまでの約 15 分間は stopped 状態のままになります。

Mac インスタンスを停止または終了すると、Amazon EC2 は基盤となる専有ホスト上でスクラブワークフローを実行して、内部 SSD を消去し、永続的な NVRAM 変数をクリアし、最新のデバイスファームウェアに更新します。これにより、Mac インスタンスは、他の EC2 Nitro インスタンスと同じセキュリティとデータプライバシーを提供できます。また、最新の macOS AMI を実行することも可能です。スクラブワークフローの最中に、専有ホストは一時的に保留状態になります。x86 Mac インスタンスでは、スクラブワークフローが完了するまでに最大で 50 分かかる場合があります。Apple シリコン Mac インスタンスでは、スクラブワークフローが完了するまでに最大で 110 分

かかる場合があります。さらに、x86 Mac インスタンスでは、デバイスファームウェアの更新が必要な場合、スクラブワークフローが完了するまでに最大で 3 時間かかる場合があります。

スクラブワークフローが完了するまで、停止した Mac インスタンスを起動したり、新しい Mac インスタンスを起動したりすることはできません。完了の時点で、Dedicated Host は `available` 状態になります。

専有ホストが `pending` 状態になると、メータリングと課金が一時停止されます。スクラブワークフロー中は課金されません。

専有ホストでサポートされている macOS バージョン

Amazon EC2 Mac 専有ホストでサポートされている最新の macOS バージョンを表示できます。この機能を使用すると、専有ホストが任意の macOS バージョンでのインスタンスの起動をサポートできるかどうかを検証できます。

macOS の各バージョンで正常に起動するには、基盤となる Apple Mac でファームウェアの最小バージョンが必要です。割り当てられた Mac 専有ホストが長期間アイドル状態のままである場合、または長時間実行されているインスタンスがある場合、Apple Mac ファームウェアのバージョンが古くなる可能性があります。

最新の macOS バージョンを確実にサポートするために、割り当てられた Mac 専有ホストでインスタンスを停止または終了できます。これにより、ホストスクラブワークフローがトリガーされ、基盤となる Apple Mac のファームウェアが最新の macOS バージョンをサポートするように更新されます。インスタンスが長時間実行されている専有ホストは、実行中のインスタンスを停止または終了すると、自動的に更新されます。

スクラブワークフローの詳細については、「Mac インスタンスの停止と終了」を参照してください。

Mac インスタンスの起動方法の詳細については、「Mac インスタンスの作成」を参照してください。

Amazon EC2 コンソールまたは AWS CLI を使用して、割り当てられた専有ホストでサポートされている最新の macOS バージョンに関する情報を表示できます。

Console

コンソールを使用して専有ホストのファームウェア情報を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. [専有ホストの詳細] ページの [サポートされている最新の macOS バージョン] に、ホストがサポートできる最新の macOS バージョンが表示されます。

AWS CLI

AWS CLI を使用して専有ホストファームウェア情報を表示するには

`describe-mac-hosts` コマンドを使用して、`region` を適切な AWS リージョンに置き換えます。

```
$ aws ec2 describe-mac-hosts --region us-east-1
  {
      "MacHosts": [
          {
              "HostId": "h-07879acf49EXAMPLE",
              "MacOSLatestSupportedVersions": [
                  "14.3",
                  "13.6.4",
                  "12.7.3"
              ]
          }
      ]
  }
```

macOS AMI の通知へのサブスクライブ

新しい AMI のリリース時、あるいは BridgeOS の更新時に通知を受け取るには、Amazon SNS を使用して通知にサブスクライブします。

macOS AMI の通知にサブスクライブするには

1. Amazon SNS コンソール (https://console.aws.amazon.com/sns/v3/home) を開きます。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。購読する SNS 通知がこのリージョンで作成されているため、このリージョンを使用する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択します。
4. [Create subscription] を選択します。

5. [サブスクリプションの作成] ダイアログボックスで、次の操作を行います。

   a. [ARN のトピック] で、次の Amazon リソースネーム (ARN) のいずれかをコピーアンドペーストします。

      - **arn:aws:sns:us-east-1:898855652048:amazon-ec2-macos-ami-updates**
      - **arn:aws:sns:us-east-1:898855652048:amazon-ec2-bridgeos-updates**

      [プロトコル] の設定

   b. E メール:

      [エンドポイント] では、通知を受信するために使用できる E メールアドレスを入力します。サブスクリプションを作成した後、件名が「AWS Notification - Subscription Confirmation」とされた確認メッセージが送られてきます。このメールを開き、[サブスクリプションの確認] をクリックして受信登録を完了します。

   c. SMS:

      エンドポイントに、通知を受信するために使用する E メールアドレスを入力します。

   d. AWS Lambda、Amazon SQS、Amazon Data Firehose (通知は JSON 形式で送信されます):

      [Endpoint] (エンドポイント) に、通知を受信するために使用する Lambda 関数、SQS キュー、または Firehose ストリームの ARN を入力します。

   e. [Create subscription] を選択します。

macOS AMI がリリースされるたびに、amazon-ec2-macos-ami-updates トピックのサブスクライバーに対し通知が送信されます。BridgeOS が変更されるたびに、amazon-ec2-bridgeos-updates トピックのサブスクライバーに対し通知が送信されます。通知が不要になった場合は、次の手順で受信登録を解除します。

macOS AMI の通知のサブスクライブを解除するには

1. Amazon SNS コンソール (https://console.aws.amazon.com/sns/v3/home) を開きます。
2. ナビゲーションバーで、必要に応じて、リージョンを [米国東部 (バージニア北部)] に変更します。SNS 通知はこのリージョンで作成されたため、このリージョンを使用する必要があります。
3. ナビゲーションペインで [Subscriptions] を選択します。

4. サブスクリプションを選択し、[アクション]、[サブスクリプションの削除] を選択します。確認のプロンプトが表示されたら、[削除] を選択します。

Mac インスタンス用の専有ホストをリリースする

Mac インスタンスの使用が終了したら、専有ホストをリリースすることでクリーンアップできます。専有ホストをリリースする前に、Mac インスタンスを停止または終了する必要があります。割り当て期間が少なくとも 24 時間を超えるまで、ホストをリリースすることはできません。

専有ホストをリリースするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[インスタンスの状態] をクリックしてから、[インスタンスの停止] または [インスタンスの終了] を選択します。
4. ナビゲーションペインで [Dedicated Hosts] を選択します。
5. 専有ホストを選択し、[アクション]、[ホストのリリース] の順に選択します。
6. 確認を求めるメッセージが表示されたら、[リリース] を選択します。

関連リソース

料金については、「 料金表」を参照してください。

Mac インスタンスの詳細については、「Amazon EC2 Mac インスタンス」を参照してください。

Mac インスタンスのハードウェア仕様とネットワークパフォーマンスについて詳しくは、「汎用インスタンス」を参照してください。

Amazon EBS 最適化インスタンスを使用する

Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O 用に専用のキャパシティを追加で提供します。このように最適化することで、Amazon EBS I/O と、インスタンスからのその他のトラフィックとの間の競合を最小に抑え、EBS ボリュームの最高のパフォーマンスを実現します。

EBS 最適化インスタンスは、Amazon EBS 用に専用の帯域幅を用意します。汎用 SSD (gp2 および gp3) ボリュームを EBS 最適化インスタンスにアタッチすると、1 年で 99% の期間、プロビジョ

ンド IOPS パフォーマンスの少なくとも 90% のボリュームが提供されます。また、プロビジョンド IOPS SSD (io1 および io2) ボリュームでは、1 年で 99.9% の期間、プロビジョンド IOPS パフォーマンスの少なくとも 90% のボリュームが提供されます。スループット最適化 HDD (st1) および Cold HDD (sc1) のどちらでも、1 年で 99% の期間、想定されるスループットパフォーマンスの少なくとも 90% のボリュームが提供されます。毎時間、予測合計スループットの 99% 達成を目標に、準拠しない期間はほぼ均一に分散されています。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。

> **Important**
>
> インスタンスの EBS パフォーマンスは、インスタンスのパフォーマンス制限、またはアタッチされたボリュームの合計パフォーマンスのうち、どちらか小さい方によって制限されます。EBS のパフォーマンスを最大化するには、インスタンスにアタッチされたボリュームが合計でインスタンスの最大パフォーマンスと同等かそれ以上のパフォーマンスを発揮する必要があります。例えば、r6i.16xlarge の 80,000 IOPSを実現するには、インスタンスに少なくとも 16,000 IOPSがそれぞれプロビジョニングされた 5 gp3 ボリューム が必要です (5 ボリュームx 16,000 IOPS = 80,000IOPS)。

コンテンツ

サポートされるインスタンスタイプ

次の表は、EBS 最適化をサポートするインスタンスタイプを示しています。この表には、Amazon EBS の専用帯域幅、ストリーミング読み取りのワークロードと 128 KiB の I/O サイズでその接続において達成できる一般的な最大スループット、および 16 KiB の I/O を使用している場合にインスタンスがサポートできる IOPS の最大数などが含まれます。

アプリケーションのニーズよりも多い専用 Amazon EBS スループットを提供する EBS 最適化インスタンスを選択します。そうでないと、Amazon EBS と Amazon EC2 間の接続がパフォーマンスのボトルネックになる可能性があります。

コンテンツ

EBS 最適化 (デフォルト)

次の表は、EBS 最適化をサポートするインスタンスタイプを示します。EBS 最適化はデフォルトで有効になっています。EBS 最適化を有効にする必要はなく、EBS 最適化を無効にしてもその効果はなくなりません。

> **Note**
>
> この情報は、AWS CLI を使用してプログラムで表示することもできます。詳細については、「EBS 最適化をサポートするインスタンスタイプを表示する」を参照してください。

トピック

汎用

> **Important**
>
> [1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 30 分間維持することができます。その後、ベースラインのパフォーマンスに戻ります。
> [2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
a1.medium [1]	300	3500	37.50	437.50	2500	20000
a1.large [1]	525	3500	65.62	437.50	4000	20000
a1.xlarge [1]	800	3500	100.00	437.50	6000	20000
a1.2xlarge [1]	1750	3500	218.75	437.50	10000	20000
a1.4xlarge [2]	3500		437.5		20000	
a1.metal [2]	3500		437.5		20000	
m4.large [2]	450		56.25		3600	
m4.xlarge [2]	750		93.75		6000	
m4.2xlarge [2]	1000		125.0		8000	
m4.4xlarge [2]	2000		250.0		16000	
m4.10xlarge [2]	4000		500.0		32000	
m4.16xlarge [2]	10000		1250.0		65000	
m5.large [1]	650	4750	81.25	593.75	3600	18750

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5.4xlarge [2]	4750		593.75		18750	
m5.8xlarge [2]	6800		850.0		30000	
m5.12xlarge [2]	9500		1187.5		40000	
m5.16xlarge [2]	13600		1700.0		60000	
m5.24xlarge [2]	19000		2375.0		80000	
m5.metal [2]	19000		2375.0		80000	
m5a.large [1]	650	2880	81.25	360.00	3600	16000
m5a.xlarge [1]	1085	2880	135.62	360.00	6000	16000
m5a.2xlarge [1]	1580	2880	197.50	360.00	8333	16000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5a.4xlarge [2]	2880		360.0		16000	
m5a.8xlarge [2]	4750		593.75		20000	
m5a.12xlarge [2]	6780		847.5		30000	
m5a.16xlarge [2]	9500		1187.5		40000	
m5a.24xlarge [2]	13750		1718.75		60000	
m5ad.large [1]	650	2880	81.25	360.00	3600	16000
m5ad.xlarge [1]	1085	2880	135.62	360.00	6000	16000
m5ad.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
m5ad.4xlarge [2]	2880		360.0		16000	
m5ad.8xlarge [2]	4750		593.75		20000	
m5ad.12xlarge [2]	6780		847.5		30000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5ad.16xlarge [2]	9500		1187.5		40000	
m5ad.24xlarge [2]	13750		1718.75		60000	
m5d.large [1]	650	4750	81.25	593.75	3600	18750
m5d.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5d.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5d.4xlarge [2]	4750		593.75		18750	
m5d.8xlarge [2]	6800		850.0		30000	
m5d.12xlarge [2]	9500		1187.5		40000	
m5d.16xlarge [2]	13600		1700.0		60000	
m5d.24xlarge [2]	19000		2375.0		80000	
m5d.metal [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5dn.large [1]	650	4750	81.25	593.75	3600	18750
m5dn.xlarge [1]	1150	4750	143.75	593.75	6000	18750
m5dn.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5dn.4xlarge [2]	4750		593.75		18750	
m5dn.8xlarge [2]	6800		850.0		30000	
m5dn.12xlarge [2]	9500		1187.5		40000	
m5dn.16xlarge [2]	13600		1700.0		60000	
m5dn.24xlarge [2]	19000		2375.0		80000	
m5dn.metal [2]	19000		2375.0		80000	
m5n.large [1]	650	4750	81.25	593.75	3600	18750
m5n.xlarge [1]	1150	4750	143.75	593.75	6000	18750

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5n.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
m5n.4xlarge [2]	4750		593.75		18750	
m5n.8xlarge [2]	6800		850.0		30000	
m5n.12xlarge [2]	9500		1187.5		40000	
m5n.16xlarge [2]	13600		1700.0		60000	
m5n.24xlarge [2]	19000		2375.0		80000	
m5n.metal [2]	19000		2375.0		80000	
m5zn.large [1]	800	3170	100.00	396.25	3333	13333
m5zn.xlarge [1]	1564	3170	195.50	396.25	6667	13333
m5zn.2xlarge [2]	3170		396.25		13333	
m5zn.3xlarge [2]	4750		593.75		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m5zn.6xlarge [2]	9500		1187.5		40000	
m5zn.12xlarge [2]	19000		2375.0		80000	
m5zn.metal [2]	19000		2375.0		80000	
m6a.large [1]	650	10000	81.25	1250.00	3600	40000
m6a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m6a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m6a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m6a.8xlarge [2]	10000		1250.0		40000	
m6a.12xlarge [2]	15000		1875.0		60000	
m6a.16xlarge [2]	20000		2500.0		80000	
m6a.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6a.32xlarge [2]	40000		5000.0		160000	
m6a.48xlarge [2]	40000		5000.0		240000	
m6a.metal [2]	40000		5000.0		240000	
m6g.medium [1]	315	4750	39.38	593.75	2500	20000
m6g.large [1]	630	4750	78.75	593.75	3600	20000
m6g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
m6g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
m6g.4xlarge [2]	4750		593.75		20000	
m6g.8xlarge [2]	9500		1187.5		40000	
m6g.12xlarge [2]	14250		1781.25		50000	
m6g.16xlarge [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6g.metal [2]	19000		2375.0		80000	
m6gd.medium [1]	315	4750	39.38	593.75	2500	20000
m6gd.large [1]	630	4750	78.75	593.75	3600	20000
m6gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
m6gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
m6gd.4xlarge [2]	4750		593.75		20000	
m6gd.8xlarge [2]	9500		1187.5		40000	
m6gd.12xlarge [2]	14250		1781.25		50000	
m6gd.16xlarge [2]	19000		2375.0		80000	
m6gd.metal [2]	19000		2375.0		80000	
m6i.large [1]	650	10000	81.25	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m6i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m6i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m6i.8xlarge [2]	10000		1250.0		40000	
m6i.12xlarge [2]	15000		1875.0		60000	
m6i.16xlarge [2]	20000		2500.0		80000	
m6i.24xlarge [2]	30000		3750.0		120000	
m6i.32xlarge [2]	40000		5000.0		160000	
m6i.metal [2]	40000		5000.0		160000	
m6id.large [1]	650	10000	81.25	1250.00	3600	40000
m6id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m6id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m6id.8xlarge [2]	10000		1250.0		40000	
m6id.12xlarge [2]	15000		1875.0		60000	
m6id.16xlarge [2]	20000		2500.0		80000	
m6id.24xlarge [2]	30000		3750.0		120000	
m6id.32xlarge [2]	40000		5000.0		160000	
m6id.metal [2]	40000		5000.0		160000	
m6idn.large [1]	1562	25000	195.31	3125.00	6250	100000
m6idn.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
m6idn.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6idn.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
m6idn.8xlarge [2]	25000		3125.0		100000	
m6idn.12xlarge [2]	37500		4687.5		150000	
m6idn.16xlarge [2]	50000		6250.0		200000	
m6idn.24xlarge [2]	75000		9375.0		300000	
m6idn.32xlarge [2]	100000		12500.0		400000	
m6idn.metal [2]	100000		12500.0		400000	
m6in.large [1]	1562	25000	195.31	3125.00	6250	100000
m6in.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
m6in.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
m6in.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m6in.8xlarge [2]	25000		3125.0		100000	
m6in.12xlarge [2]	37500		4687.5		150000	
m6in.16xlarge [2]	50000		6250.0		200000	
m6in.24xlarge [2]	75000		9375.0		300000	
m6in.32xlarge [2]	100000		12500.0		400000	
m6in.metal [2]	100000		12500.0		400000	
m7a.medium [1]	325	10000	40.62	1250.00	2500	40000
m7a.large [1]	650	10000	81.25	1250.00	3600	40000
m7a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7a.8xlarge [2]	10000		1250.0		40000	
m7a.12xlarge [2]	15000		1875.0		60000	
m7a.16xlarge [2]	20000		2500.0		80000	
m7a.24xlarge [2]	30000		3750.0		120000	
m7a.32xlarge [2]	40000		5000.0		160000	
m7a.48xlarge [2]	40000		5000.0		240000	
m7a.metal-48xl [2]	40000		5000.0		240000	
m7g.medium [1]	315	10000	39.38	1250.00	2500	40000
m7g.large [1]	630	10000	78.75	1250.00	3600	40000
m7g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7g.8xlarge [2]	10000		1250.0		40000	
m7g.12xlarge [2]	15000		1875.0		60000	
m7g.16xlarge [2]	20000		2500.0		80000	
m7g.metal [2]	20000		2500.0		80000	
m7gd.medium [1]	315	10000	39.38	1250.00	2500	40000
m7gd.large [1]	630	10000	78.75	1250.00	3600	40000
m7gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7gd.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7gd.12xlarge [2]	15000		1875.0		60000	
m7gd.16xlarge [2]	20000		2500.0		80000	
m7gd.metal [2]	20000		2500.0		80000	
m7i.large [1]	650	10000	81.25	1250.00	3600	40000
m7i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
m7i.8xlarge [2]	10000		1250.0		40000	
m7i.12xlarge [2]	15000		1875.0		60000	
m7i.16xlarge [2]	20000		2500.0		80000	
m7i.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
m7i.48xlarge [2]	40000		5000.0		240000	
m7i.metal-24xl [2]	30000		3750.0		120000	
m7i.metal-48xl [2]	40000		5000.0		240000	
m7i-flex.large [1]	312	10000	39.06	1250.00	2500	40000
m7i-flex.xlarge [1]	625	10000	78.12	1250.00	3600	40000
m7i-flex.2xlarge [1]	1250	10000	156.25	1250.00	6000	40000
m7i-flex.4xlarge [1]	2500	10000	312.50	1250.00	12000	40000
m7i-flex.8xlarge [1]	5000	10000	625.00	1250.00	20000	40000
mac1.metal [2]	14000		1750.0		80000	
mac2.metal [2]	10000		1250.0		55000	
mac2-m2.metal [2]	8000		1000.0		55000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
mac2-m2pro.metal [2]	8000		1000.0		55000	
t3.nano [1]	43	2085	5.38	260.62	250	11800
t3.micro [1]	87	2085	10.88	260.62	500	11800
t3.small [1]	174	2085	21.75	260.62	1000	11800
t3.medium [1]	347	2085	43.38	260.62	2000	11800
t3.large [1]	695	2780	86.88	347.50	4000	15700
t3.xlarge [1]	695	2780	86.88	347.50	4000	15700
t3.2xlarge [1]	695	2780	86.88	347.50	4000	15700
t3a.nano [1]	45	2085	5.62	260.62	250	11800
t3a.micro [1]	90	2085	11.25	260.62	500	11800
t3a.small [1]	175	2085	21.88	260.62	1000	11800
t3a.medium [1]	350	2085	43.75	260.62	2000	11800
t3a.large [1]	695	2780	86.88	347.50	4000	15700
t3a.xlarge [1]	695	2780	86.88	347.50	4000	15700
t3a.2xlarge [1]	695	2780	86.88	347.50	4000	15700

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
t4g.nano [1]	43	2085	5.38	260.62	250	11800
t4g.micro [1]	87	2085	10.88	260.62	500	11800
t4g.small [1]	174	2085	21.75	260.62	1000	11800
t4g.medium [1]	347	2085	43.38	260.62	2000	11800
t4g.large [1]	695	2780	86.88	347.50	4000	15700
t4g.xlarge [1]	695	2780	86.88	347.50	4000	15700
t4g.2xlarge [1]	695	2780	86.88	347.50	4000	15700

コンピューティングの最適化

Important

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 30 分間維持することができます。その後、ベースラインのパフォーマンスに戻ります。
[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c4.large [2]	500		62.5		4000	
c4.xlarge [2]	750		93.75		6000	
c4.2xlarge [2]	1000		125.0		8000	
c4.4xlarge [2]	2000		250.0		16000	
c4.8xlarge [2]	4000		500.0		32000	
c5.large [1]	650	4750	81.25	593.75	4000	20000
c5.xlarge [1]	1150	4750	143.75	593.75	6000	20000
c5.2xlarge [1]	2300	4750	287.50	593.75	10000	20000
c5.4xlarge [2]	4750		593.75		20000	
c5.9xlarge [2]	9500		1187.5		40000	
c5.12xlarge [2]	9500		1187.5		40000	
c5.18xlarge [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c5.24xlarge [2]	19000		2375.0		80000	
c5.metal [2]	19000		2375.0		80000	
c5a.large [1]	200	3170	25.00	396.25	800	13300
c5a.xlarge [1]	400	3170	50.00	396.25	1600	13300
c5a.2xlarge [1]	800	3170	100.00	396.25	3200	13300
c5a.4xlarge [1]	1580	3170	197.50	396.25	6600	13300
c5a.8xlarge [2]	3170		396.25		13300	
c5a.12xlarge [2]	4750		593.75		20000	
c5a.16xlarge [2]	6300		787.5		26700	
c5a.24xlarge [2]	9500		1187.5		40000	
c5ad.large [1]	200	3170	25.00	396.25	800	13300

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c5ad.xlarge [1]	400	3170	50.00	396.25	1600	13300
c5ad.2xlarge [1]	800	3170	100.00	396.25	3200	13300
c5ad.4xlarge [1]	1580	3170	197.50	396.25	6600	13300
c5ad.8xlarge [2]	3170		396.25		13300	
c5ad.12xlarge [2]	4750		593.75		20000	
c5ad.16xlarge [2]	6300		787.5		26700	
c5ad.24xlarge [2]	9500		1187.5		40000	
c5d.large [1]	650	4750	81.25	593.75	4000	20000
c5d.xlarge [1]	1150	4750	143.75	593.75	6000	20000
c5d.2xlarge [1]	2300	4750	287.50	593.75	10000	20000
c5d.4xlarge [2]	4750		593.75		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c5d.9xlarge [2]	9500		1187.5		40000	
c5d.12xlarge [2]	9500		1187.5		40000	
c5d.18xlarge [2]	19000		2375.0		80000	
c5d.24xlarge [2]	19000		2375.0		80000	
c5d.metal [2]	19000		2375.0		80000	
c5n.large [1]	650	4750	81.25	593.75	4000	20000
c5n.xlarge [1]	1150	4750	143.75	593.75	6000	20000
c5n.2xlarge [1]	2300	4750	287.50	593.75	10000	20000
c5n.4xlarge [2]	4750		593.75		20000	
c5n.9xlarge [2]	9500		1187.5		40000	
c5n.18xlarge [2]	19000		2375.0		80000	
c5n.metal [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6a.large [1]	650	10000	81.25	1250.00	3600	40000
c6a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c6a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c6a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c6a.8xlarge [2]	10000		1250.0		40000	
c6a.12xlarge [2]	15000		1875.0		60000	
c6a.16xlarge [2]	20000		2500.0		80000	
c6a.24xlarge [2]	30000		3750.0		120000	
c6a.32xlarge [2]	40000		5000.0		160000	
c6a.48xlarge [2]	40000		5000.0		240000	
c6a.metal [2]	40000		5000.0		240000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6g.medium [1]	315	4750	39.38	593.75	2500	20000
c6g.large [1]	630	4750	78.75	593.75	3600	20000
c6g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
c6g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
c6g.4xlarge [2]	4750		593.75		20000	
c6g.8xlarge [2]	9500		1187.5		40000	
c6g.12xlarge [2]	14250		1781.25		50000	
c6g.16xlarge [2]	19000		2375.0		80000	
c6g.metal [2]	19000		2375.0		80000	
c6gd.medium [1]	315	4750	39.38	593.75	2500	20000
c6gd.large [1]	630	4750	78.75	593.75	3600	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
c6gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
c6gd.4xlarge [2]	4750		593.75		20000	
c6gd.8xlarge [2]	9500		1187.5		40000	
c6gd.12xlarge [2]	14250		1781.25		50000	
c6gd.16xlarge [2]	19000		2375.0		80000	
c6gd.metal [2]	19000		2375.0		80000	
c6gn.medium [1]	760	9500	95.00	1187.50	2500	40000
c6gn.large [1]	1235	9500	154.38	1187.50	5000	40000
c6gn.xlarge [1]	2375	9500	296.88	1187.50	10000	40000
c6gn.2xlarge [1]	4750	9500	593.75	1187.50	20000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6gn.4xlarge [2]	9500		1187.5		40000	
c6gn.8xlarge [2]	19000		2375.0		80000	
c6gn.12xlarge [2]	28500		3562.5		120000	
c6gn.16xlarge [2]	38000		4750.0		160000	
c6i.large [1]	650	10000	81.25	1250.00	3600	40000
c6i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c6i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c6i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c6i.8xlarge [2]	10000		1250.0		40000	
c6i.12xlarge [2]	15000		1875.0		60000	
c6i.16xlarge [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6i.24xlarge [2]	30000		3750.0		120000	
c6i.32xlarge [2]	40000		5000.0		160000	
c6i.metal [2]	40000		5000.0		160000	
c6id.large [1]	650	10000	81.25	1250.00	3600	40000
c6id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c6id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c6id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c6id.8xlarge [2]	10000		1250.0		40000	
c6id.12xlarge [2]	15000		1875.0		60000	
c6id.16xlarge [2]	20000		2500.0		80000	
c6id.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6id.32xlarge [2]	40000		5000.0		160000	
c6id.metal [2]	40000		5000.0		160000	
c6in.large [1]	1562	25000	195.31	3125.00	6250	100000
c6in.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
c6in.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
c6in.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
c6in.8xlarge [2]	25000		3125.0		100000	
c6in.12xlarge [2]	37500		4687.5		150000	
c6in.16xlarge [2]	50000		6250.0		200000	
c6in.24xlarge [2]	75000		9375.0		300000	
c6in.32xlarge [2]	100000		12500.0		400000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c6in.metal [2]	100000		12500.0		400000	
c7a.medium [1]	325	10000	40.62	1250.00	2500	40000
c7a.large [1]	650	10000	81.25	1250.00	3600	40000
c7a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7a.8xlarge [2]	10000		1250.0		40000	
c7a.12xlarge [2]	15000		1875.0		60000	
c7a.16xlarge [2]	20000		2500.0		80000	
c7a.24xlarge [2]	30000		3750.0		120000	
c7a.32xlarge [2]	40000		5000.0		160000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7a.48xlarge [2]	40000		5000.0		240000	
c7a.metal-48xl [2]	40000		5000.0		240000	
c7g.medium [1]	315	10000	39.38	1250.00	2500	40000
c7g.large [1]	630	10000	78.75	1250.00	3600	40000
c7g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7g.8xlarge [2]	10000		1250.0		40000	
c7g.12xlarge [2]	15000		1875.0		60000	
c7g.16xlarge [2]	20000		2500.0		80000	
c7g.metal [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7gd.medium [1]	315	10000	39.38	1250.00	2500	40000
c7gd.large [1]	630	10000	78.75	1250.00	3600	40000
c7gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7gd.8xlarge [2]	10000		1250.0		40000	
c7gd.12xlarge [2]	15000		1875.0		60000	
c7gd.16xlarge [2]	20000		2500.0		80000	
c7gd.metal [2]	20000		2500.0		80000	
c7gn.medium [1]	521	10000	65.12	1250.00	2083	40000
c7gn.large [1]	1042	10000	130.25	1250.00	4167	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7gn.xlarge [1]	2083	10000	260.38	1250.00	8333	40000
c7gn.2xlarge [1]	4167	10000	520.88	1250.00	16667	40000
c7gn.4xlarge [1]	8333	10000	1041.62	1250.00	33333	40000
c7gn.8xlarge [1]	16667	20000	2083.38	2500.00	66667	80000
c7gn.12xlarge [1]	25000	30000	3125.00	3750.00	100000	120000
c7gn.16xlarge [1]	33333	40000	4166.62	5000.00	133333	160000
c7i.large [1]	650	10000	81.25	1250.00	3600	40000
c7i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
c7i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
c7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
c7i.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
c7i.12xlarge [2]	15000		1875.0		60000	
c7i.16xlarge [2]	20000		2500.0		80000	
c7i.24xlarge [2]	30000		3750.0		120000	
c7i.48xlarge [2]	40000		5000.0		240000	
c7i.metal-24xl [2]	30000		3750.0		120000	
c7i.metal-48xl [2]	40000		5000.0		240000	

メモリ最適化

Important

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 30 分間維持することができます。その後、ベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r4.large [2]	425		53.125		3000	
r4.xlarge [2]	850		106.25		6000	
r4.2xlarge [2]	1700		212.5		12000	
r4.4xlarge [2]	3500		437.5		18750	
r4.8xlarge [2]	7000		875.0		37500	
r4.16xlarge [2]	14000		1750.0		75000	
r5.large [1]	650	4750	81.25	593.75	3600	18750
r5.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5.4xlarge [2]	4750		593.75		18750	
r5.8xlarge [2]	6800		850.0		30000	
r5.12xlarge [2]	9500		1187.5		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5.16xlarge [2]	13600		1700.0		60000	
r5.24xlarge [2]	19000		2375.0		80000	
r5.metal [2]	19000		2375.0		80000	
r5a.large [1]	650	2880	81.25	360.00	3600	16000
r5a.xlarge [1]	1085	2880	135.62	360.00	6000	16000
r5a.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
r5a.4xlarge [2]	2880		360.0		16000	
r5a.8xlarge [2]	4750		593.75		20000	
r5a.12xlarge [2]	6780		847.5		30000	
r5a.16xlarge [2]	9500		1187.5		40000	
r5a.24xlarge [2]	13570		1696.25		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5ad.large [1]	650	2880	81.25	360.00	3600	16000
r5ad.xlarge [1]	1085	2880	135.62	360.00	6000	16000
r5ad.2xlarge [1]	1580	2880	197.50	360.00	8333	16000
r5ad.4xlarge [2]	2880		360.0		16000	
r5ad.8xlarge [2]	4750		593.75		20000	
r5ad.12xlarge [2]	6780		847.5		30000	
r5ad.16xlarge [2]	9500		1187.5		40000	
r5ad.24xlarge [2]	13570		1696.25		60000	
r5b.large [1]	1250	10000	156.25	1250.00	5417	43333
r5b.xlarge [1]	2500	10000	312.50	1250.00	10833	43333
r5b.2xlarge [1]	5000	10000	625.00	1250.00	21667	43333

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5b.4xlarge [2]	10000		1250.0		43333	
r5b.8xlarge [2]	20000		2500.0		86667	
r5b.12xlarge [2]	30000		3750.0		130000	
r5b.16xlarge [2]	40000		5000.0		173333	
r5b.24xlarge [2]	60000		7500.0		260000	
r5b.metal [2]	60000		7500.0		260000	
r5d.large [1]	650	4750	81.25	593.75	3600	18750
r5d.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5d.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5d.4xlarge [2]	4750		593.75		18750	
r5d.8xlarge [2]	6800		850.0		30000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5d.12xlarge [2]	9500		1187.5		40000	
r5d.16xlarge [2]	13600		1700.0		60000	
r5d.24xlarge [2]	19000		2375.0		80000	
r5d.metal [2]	19000		2375.0		80000	
r5dn.large [1]	650	4750	81.25	593.75	3600	18750
r5dn.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5dn.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5dn.4xlarge [2]	4750		593.75		18750	
r5dn.8xlarge [2]	6800		850.0		30000	
r5dn.12xlarge [2]	9500		1187.5		40000	
r5dn.16xlarge [2]	13600		1700.0		60000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r5dn.24xlarge [2]	19000		2375.0		80000	
r5dn.metal [2]	19000		2375.0		80000	
r5n.large [1]	650	4750	81.25	593.75	3600	18750
r5n.xlarge [1]	1150	4750	143.75	593.75	6000	18750
r5n.2xlarge [1]	2300	4750	287.50	593.75	12000	18750
r5n.4xlarge [2]	4750		593.75		18750	
r5n.8xlarge [2]	6800		850.0		30000	
r5n.12xlarge [2]	9500		1187.5		40000	
r5n.16xlarge [2]	13600		1700.0		60000	
r5n.24xlarge [2]	19000		2375.0		80000	
r5n.metal [2]	19000		2375.0		80000	
r6a.large [1]	650	10000	81.25	1250.00	3600	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r6a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r6a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r6a.8xlarge [2]	10000		1250.0		40000	
r6a.12xlarge [2]	15000		1875.0		60000	
r6a.16xlarge [2]	20000		2500.0		80000	
r6a.24xlarge [2]	30000		3750.0		120000	
r6a.32xlarge [2]	40000		5000.0		160000	
r6a.48xlarge [2]	40000		5000.0		240000	
r6a.metal [2]	40000		5000.0		240000	
r6g.medium [1]	315	4750	39.38	593.75	2500	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6g.large [1]	630	4750	78.75	593.75	3600	20000
r6g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
r6g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
r6g.4xlarge [2]	4750		593.75		20000	
r6g.8xlarge [2]	9500		1187.5		40000	
r6g.12xlarge [2]	14250		1781.25		50000	
r6g.16xlarge [2]	19000		2375.0		80000	
r6g.metal [2]	19000		2375.0		80000	
r6gd.medium [1]	315	4750	39.38	593.75	2500	20000
r6gd.large [1]	630	4750	78.75	593.75	3600	20000
r6gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
r6gd.4xlarge [2]	4750		593.75		20000	
r6gd.8xlarge [2]	9500		1187.5		40000	
r6gd.12xlarge [2]	14250		1781.25		50000	
r6gd.16xlarge [2]	19000		2375.0		80000	
r6gd.metal [2]	19000		2375.0		80000	
r6i.large [1]	650	10000	81.25	1250.00	3600	40000
r6i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r6i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r6i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r6i.8xlarge [2]	10000		1250.0		40000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6i.12xlarge [2]	15000		1875.0		60000	
r6i.16xlarge [2]	20000		2500.0		80000	
r6i.24xlarge [2]	30000		3750.0		120000	
r6i.32xlarge [2]	40000		5000.0		160000	
r6i.metal [2]	40000		5000.0		160000	
r6idn.large [1]	1562	25000	195.31	3125.00	6250	100000
r6idn.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
r6idn.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
r6idn.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
r6idn.8xlarge [2]	25000		3125.0		100000	
r6idn.12xlarge [2]	37500		4687.5		150000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6idn.16xlarge [2]	50000		6250.0		200000	
r6idn.24xlarge [2]	75000		9375.0		300000	
r6idn.32xlarge [2]	100000		12500.0		400000	
r6idn.metal [2]	100000		12500.0		400000	
r6in.large [1]	1562	25000	195.31	3125.00	6250	100000
r6in.xlarge [1]	3125	25000	390.62	3125.00	12500	100000
r6in.2xlarge [1]	6250	25000	781.25	3125.00	25000	100000
r6in.4xlarge [1]	12500	25000	1562.50	3125.00	50000	100000
r6in.8xlarge [2]	25000		3125.0		100000	
r6in.12xlarge [2]	37500		4687.5		150000	
r6in.16xlarge [2]	50000		6250.0		200000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6in.24xlarge [2]	75000		9375.0		300000	
r6in.32xlarge [2]	100000		12500.0		400000	
r6in.metal [2]	100000		12500.0		400000	
r6id.large [1]	650	10000	81.25	1250.00	3600	40000
r6id.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r6id.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r6id.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r6id.8xlarge [2]	10000		1250.0		40000	
r6id.12xlarge [2]	15000		1875.0		60000	
r6id.16xlarge [2]	20000		2500.0		80000	
r6id.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r6id.32xlarge [2]	40000		5000.0		160000	
r6id.metal [2]	40000		5000.0		160000	
r7a.medium [1]	325	10000	40.62	1250.00	2500	40000
r7a.large [1]	650	10000	81.25	1250.00	3600	40000
r7a.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7a.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7a.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7a.8xlarge [2]	10000		1250.0		40000	
r7a.12xlarge [2]	15000		1875.0		60000	
r7a.16xlarge [2]	20000		2500.0		80000	
r7a.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7a.32xlarge [2]	40000		5000.0		160000	
r7a.48xlarge [2]	40000		5000.0		240000	
r7a.metal-48xl [2]	40000		5000.0		240000	
r7g.medium [1]	315	10000	39.38	1250.00	2500	40000
r7g.large [1]	630	10000	78.75	1250.00	3600	40000
r7g.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7g.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7g.8xlarge [2]	10000		1250.0		40000	
r7g.12xlarge [2]	15000		1875.0		60000	
r7g.16xlarge [2]	20000		2500.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7g.metal [2]	20000		2500.0		80000	
r7gd.medium [1]	315	10000	39.38	1250.00	2500	40000
r7gd.large [1]	630	10000	78.75	1250.00	3600	40000
r7gd.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
r7gd.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7gd.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7gd.8xlarge [2]	10000		1250.0		40000	
r7gd.12xlarge [2]	15000		1875.0		60000	
r7gd.16xlarge [2]	20000		2500.0		80000	
r7gd.metal [2]	20000		2500.0		80000	
r7i.large [1]	650	10000	81.25	1250.00	3600	40000
r7i.xlarge [1]	1250	10000	156.25	1250.00	6000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7i.2xlarge [1]	2500	10000	312.50	1250.00	12000	40000
r7i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7i.8xlarge [2]	10000		1250.0		40000	
r7i.12xlarge [2]	15000		1875.0		60000	
r7i.16xlarge [2]	20000		2500.0		80000	
r7i.24xlarge [2]	30000		3750.0		120000	
r7i.48xlarge [2]	40000		5000.0		240000	
r7i.metal-24xl [2]	30000		3750.0		120000	
r7i.metal-48xl [2]	40000		5000.0		240000	
r7iz.large [1]	792	10000	99.00	1250.00	3600	40000
r7iz.xlarge [1]	1584	10000	198.00	1250.00	6667	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
r7iz.2xlarge [1]	3168	10000	396.00	1250.00	13333	40000
r7iz.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
r7iz.8xlarge [2]	10000		1250.0		40000	
r7iz.12xlarge [2]	19000		2375.0		76000	
r7iz.16xlarge [2]	20000		2500.0		80000	
r7iz.32xlarge [2]	40000		5000.0		160000	
r7iz.metal-16xl [2]	20000		2500.0		80000	
r7iz.metal-32xl [2]	40000		5000.0		160000	
u-3tb1.56xlarge [2]	19000		2375.0		80000	
u-6tb1.56xlarge [2]	38000		4750.0		160000	
u-6tb1.112xlarge [2]	38000		4750.0		160000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
u-6tb1.metal [2]	38000		4750.0		160000	
u-9tb1.112xlarge [2]	38000		4750.0		160000	
u-9tb1.metal [2]	38000		4750.0		160000	
u-12tb1.112xlarge [2]	38000		4750.0		160000	
u-12tb1.metal [2]	38000		4750.0		160000	
u-18tb1.112xlarge [2]	38000		4750.0		160000	
u-18tb1.metal [2]	38000		4750.0		160000	
u-24tb1.112xlarge [2]	38000		4750.0		160000	
u-24tb1.metal [2]	38000		4750.0		160000	
x1.16xlarge [2]	7000		875.0		40000	
x1.32xlarge [2]	14000		1750.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x2gd.medium [1]	315	4750	39.38	593.75	2500	20000
x2gd.large [1]	630	4750	78.75	593.75	3600	20000
x2gd.xlarge [1]	1188	4750	148.50	593.75	6000	20000
x2gd.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
x2gd.4xlarge [2]	4750		593.75		20000	
x2gd.8xlarge [2]	9500		1187.5		40000	
x2gd.12xlarge [2]	14250		1781.25		60000	
x2gd.16xlarge [2]	19000		2375.0		80000	
x2gd.metal [2]	19000		2375.0		80000	
x2idn.16xlarge [2]	40000		5000.0		173333	
x2idn.24xlarge [2]	60000		7500.0		260000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x2idn.32xlarge [2]	80000		10000.0		260000	
x2idn.metal [2]	80000		10000.0		260000	
x2iedn.xlarge [1]	2500	20000	312.50	2500.00	8125	65000
x2iedn.2xlarge [1]	5000	20000	625.00	2500.00	16250	65000
x2iedn.4xlarge [1]	10000	20000	1250.00	2500.00	32500	65000
x2iedn.8xlarge [2]	20000		2500.0		65000	
x2iedn.16xlarge [2]	40000		5000.0		130000	
x2iedn.24xlarge [2]	60000		7500.0		195000	
x2iedn.32xlarge [2]	80000		10000.0		260000	
x2iedn.metal [2]	80000		10000.0		260000	
x2iezn.2xlarge [2]	3170		396.25		13333	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
x2iezn.4xlarge [2]	4750		593.75		20000	
x2iezn.6xlarge [2]	9500		1187.5		40000	
x2iezn.8xlarge [2]	12000		1500.0		55000	
x2iezn.12xlarge [2]	19000		2375.0		80000	
x2iezn.metal [2]	19000		2375.0		80000	
x1e.xlarge [2]	500		62.5		3700	
x1e.2xlarge [2]	1000		125.0		7400	
x1e.4xlarge [2]	1750		218.75		10000	
x1e.8xlarge [2]	3500		437.5		20000	
x1e.16xlarge [2]	7000		875.0		40000	
x1e.32xlarge [2]	14000		1750.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
z1d.large [1]	800	3170	100.00	396.25	3333	13333
z1d.xlarge [1]	1580	3170	197.50	396.25	6667	13333
z1d.2xlarge [2]	3170		396.25		13333	
z1d.3xlarge [2]	4750		593.75		20000	
z1d.6xlarge [2]	9500		1187.5		40000	
z1d.12xlarge [2]	19000		2375.0		80000	
z1d.metal [2]	19000		2375.0		80000	

ストレージの最適化

Important

[1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 30 分間維持することができます。その後、ベースラインのパフォーマンスに戻ります。

[2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
d2.xlarge [2]	750		93.75		6000	
d2.2xlarge [2]	1000		125.0		8000	
d2.4xlarge [2]	2000		250.0		16000	
d2.8xlarge [2]	4000		500.0		32000	
d3.xlarge [1]	850	2800	106.25	350.00	5000	15000
d3.2xlarge [1]	1700	2800	212.50	350.00	10000	15000
d3.4xlarge [2]	2800		350.0		15000	
d3.8xlarge [2]	5000		625.0		30000	
d3en.xlarge [1]	850	2800	106.25	350.00	5000	15000
d3en.2xlarge [1]	1700	2800	212.50	350.00	10000	15000
d3en.4xlarge [2]	2800		350.0		15000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
d3en.6xlarge [2]	4000		500.0		25000	
d3en.8xlarge [2]	5000		625.0		30000	
d3en.12xlarge [2]	7000		875.0		40000	
h1.2xlarge [2]	1750		218.75		12000	
h1.4xlarge [2]	3500		437.5		20000	
h1.8xlarge [2]	7000		875.0		40000	
h1.16xlarge [2]	14000		1750.0		80000	
i3.large [2]	425		53.125		3000	
i3.xlarge [2]	850		106.25		6000	
i3.2xlarge [2]	1700		212.5		12000	
i3.4xlarge [2]	3500		437.5		16000	
i3.8xlarge [2]	7000		875.0		32500	
i3.16xlarge [2]	14000		1750.0		65000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i3.metal [2]	19000		2375.0		80000	
i3en.large [1]	576	4750	72.10	593.75	3000	20000
i3en.xlarge [1]	1153	4750	144.20	593.75	6000	20000
i3en.2xlarge [1]	2307	4750	288.39	593.75	12000	20000
i3en.3xlarge [1]	3800	4750	475.00	593.75	15000	20000
i3en.6xlarge [2]	4750		593.75		20000	
i3en.12xlarge [2]	9500		1187.5		40000	
i3en.24xlarge [2]	19000		2375.0		80000	
i3en.metal [2]	19000		2375.0		80000	
i4g.large [1]	625	10000	78.12	1250.00	2500	40000
i4g.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i4g.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i4g.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
i4g.8xlarge [2]	10000		1250.0		40000	
i4g.16xlarge [2]	20000		2500.0		80000	
i4i.large [1]	625	10000	78.12	1250.00	2500	40000
i4i.xlarge [1]	1250	10000	156.25	1250.00	5000	40000
i4i.2xlarge [1]	2500	10000	312.50	1250.00	10000	40000
i4i.4xlarge [1]	5000	10000	625.00	1250.00	20000	40000
i4i.8xlarge [2]	10000		1250.0		40000	
i4i.12xlarge [2]	15000		1875.0		60000	
i4i.16xlarge [2]	20000		2500.0		80000	
i4i.24xlarge [2]	30000		3750.0		120000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
i4i.32xlarge [2]	40000		5000.0		160000	
i4i.metal [2]	40000		5000.0		160000	
im4gn.large [1]	1250	10000	156.25	1250.00	5000	40000
im4gn.xlarge [1]	2500	10000	312.50	1250.00	10000	40000
im4gn.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000
im4gn.4xlarge [2]	10000		1250.0		40000	
im4gn.8xlarge [2]	20000		2500.0		80000	
im4gn.16xlarge [2]	40000		5000.0		160000	
is4gen.medium [1]	625	10000	78.12	1250.00	2500	40000
is4gen.large [1]	1250	10000	156.25	1250.00	5000	40000
is4gen.xlarge [1]	2500	10000	312.50	1250.00	10000	40000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
is4gen.2xlarge [1]	5000	10000	625.00	1250.00	20000	40000
is4gen.4xlarge [2]	10000		1250.0		40000	
is4gen.8xlarge [2]	20000		2500.0		80000	

高速コンピューティング

> **Important**
>
> [1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 30 分間維持することができます。その後、ベースラインのパフォーマンスに戻ります。
> [2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
dl1.24xlarge [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
dl2q.24xlarge [2]	19000		2375.0		80000	
f1.2xlarge [2]	1700		212.5		12000	
f1.4xlarge [2]	3500		437.5		44000	
f1.16xlarge [2]	14000		1750.0		75000	
g3.4xlarge [2]	3500		437.5		20000	
g3.8xlarge [2]	7000		875.0		40000	
g3.16xlarge [2]	14000		1750.0		80000	
g4ad.xlarge [1]	400	3170	50.00	396.25	1700	13333
g4ad.2xlarge [1]	800	3170	100.00	396.25	3400	13333
g4ad.4xlarge [1]	1580	3170	197.50	396.25	6700	13333
g4ad.8xlarge [2]	3170		396.25		13333	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g4ad.16xlarge [2]	6300		787.5		26667	
g4dn.xlarge [1]	950	3500	118.75	437.50	3000	20000
g4dn.2xlarge [1]	1150	3500	143.75	437.50	6000	20000
g4dn.4xlarge [2]	4750		593.75		20000	
g4dn.8xlarge [2]	9500		1187.5		40000	
g4dn.12xlarge [2]	9500		1187.5		40000	
g4dn.16xlarge [2]	9500		1187.5		40000	
g4dn.metal [2]	19000		2375.0		80000	
g5.xlarge [1]	700	3500	87.50	437.50	3000	15000
g5.2xlarge [1]	850	3500	106.25	437.50	3500	15000
g5.4xlarge [2]	4750		593.75		20000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
g5.8xlarge [2]	16000		2000.0		65000	
g5.12xlarge [2]	16000		2000.0		65000	
g5.16xlarge [2]	16000		2000.0		65000	
g5.24xlarge [2]	19000		2375.0		80000	
g5.48xlarge [2]	19000		2375.0		80000	
g5g.xlarge [1]	1188	4750	148.50	593.75	6000	20000
g5g.2xlarge [1]	2375	4750	296.88	593.75	12000	20000
g5g.4xlarge [2]	4750		593.75		20000	
g5g.8xlarge [2]	9500		1187.5		40000	
g5g.16xlarge [2]	19000		2375.0		80000	
g5g.metal [2]	19000		2375.0		80000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
inf1.xlarge [1]	1190	4750	148.75	593.75	4000	20000
inf1.2xlarge [1]	1190	4750	148.75	593.75	6000	20000
inf1.6xlarge [2]	4750		593.75		20000	
inf1.24xlarge [2]	19000		2375.0		80000	
inf2.xlarge [1]	1250	10000	156.25	1250.00	6000	40000
inf2.8xlarge [2]	10000		1250.0		40000	
inf2.24xlarge [2]	30000		3750.0		120000	
inf2.48xlarge [2]	60000		7500.0		240000	
p2.xlarge [2]	750		93.75		6000	
p2.8xlarge [2]	5000		625.0		32500	
p2.16xlarge [2]	10000		1250.0		65000	

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
p3.2xlarge [2]	1750		218.75		10000	
p3.8xlarge [2]	7000		875.0		40000	
p3.16xlarge [2]	14000		1750.0		80000	
p3dn.24xlarge [2]	19000		2375.0		80000	
p4d.24xlarge [2]	19000		2375.0		80000	
p4de.24xlarge [2]	19000		2375.0		80000	
p5.48xlarge [2]	80000		10000.0		260000	
trn1.2xlarge [1]	5000	20000	625.00	2500.00	16250	65000
trn1.32xlarge [2]	80000		10000.0		260000	
trn1n.32xlarge [2]	80000		10000.0		260000	
vt1.3xlarge [1]	2375	4750	296.88	593.75	10000	20000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
vt1.6xlarge [2]	4750		593.75		20000	
vt1.24xlarge [2]	19000		2375.0		80000	

高性能コンピューティング

> **Important**
>
> [1] これらのインスタンスは、最大パフォーマンスを 24 時間ごとに少なくとも 30 分間維持することができます。その後、ベースラインのパフォーマンスに戻ります。
> [2] これらのインスタンスは、記載されているパフォーマンスを無期限に維持することができます。ワークロードで、最大パフォーマンスを 30 分以上維持する必要がある場合は、これらのインスタンスの中から 1 つ使用します。

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
hpc6a.48xlarge [1]	87	2085	10.88	260.62	500	11000
hpc6id.32xlarge [1]	87	2085	10.88	260.62	500	11000

インスタンスサイズ	ベースラインの帯域幅 (Mbps)	最大帯域幅 (Mbps)	ベースラインスループット (MB/秒、128 KiB I/O)	最大スループット (MB/秒、128 KiB I/O)	ベースライン IOPS (16 KiB I/O)	最大 IOPS (16 KiB I/O)
hpc7a.12xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7a.24xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7a.48xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7a.96xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7g.4xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7g.8xlarge [1]	87	2085	10.88	260.62	500	11000
hpc7g.16xlarge [1]	87	2085	10.88	260.62	500	11000

EBS 最適化をサポート

次の表は、EBS 最適化をサポートするインスタンスタイプを示します。EBS 最適化はデフォルトでは有効になっていません。EBS 最適化は、これらのインスタンスの起動時または実行後に有効にすることができます。前述のパフォーマンスレベルを達成するには、インスタンスで EBS 最適化を有効にする必要があります。デフォルトで EBS 最適化が行われないインスタンスに対して EBS 最適化を有効にするときは、専用の容量について安価な時間単位の料金を追加でお支払いいただきます。料金については、Amazon EC2 の料金、オンデマンド料金表ページで EBS 最適化インスタンスを参照してください。

> **Note**
>
> この情報は、AWS CLI を使用してプログラムで表示することもできます。詳細については、EBS 最適化をサポートするインスタンスタイプを表示するを参照してください。

インスタンスサイズ	最大帯域幅 (Mbps)	最大スループット (MB/秒、128 KiB I/O)	最大 IOPS (16 KiB I/O)
c1.xlarge	1000	125.0	8000
c3.xlarge	500	62.5	4000
c3.2xlarge	1000	125.0	8000
c3.4xlarge	2000	250.0	16000
g2.2xlarge	1000	125.0	8000
i2.xlarge	500	62.5	4000
i2.2xlarge	1000	125.0	8000
i2.4xlarge	2000	250.0	16000
m1.large	500	62.5	4000
m1.xlarge	1000	125.0	8000
m2.2xlarge	500	62.5	4000
m2.4xlarge	1000	125.0	8000
m3.xlarge	500	62.5	4000
m3.2xlarge	1000	125.0	8000
r3.xlarge	500	62.5	4000
r3.2xlarge	1000	125.0	8000

インスタンスサイズ	最大帯域幅 (Mbps)	最大スループット (MB/秒、128 KiB I/O)	最大 IOPS (16 KiB I/O)
r3.4xlarge	2000	250.0	16000

i2.8xlarge、c3.8xlarge、および r3.8xlarge インスタンスには専用の EBS 帯域幅がないため、EBS 最適化を提供しません。これらのインスタンスでは、ネットワークトラフィックと Amazon EBS トラフィックは同じ 10 ギガビットネットワークインターフェイスで共有されます。

最大のパフォーマンスの獲得

EBSIOBalance% および EBSByteBalance% メトリクスを使用して、インスタンスのサイズが正しく設定されているかどうかを判断できます。これらのメトリクスを CloudWatch コンソールで表示して、指定したしきい値に基づいてトリガーされるアラームを設定することができます。これらのメトリクスは、割合 (%) で表されます。バランスの割合が常に低いインスタンスは、拡大する必要があります。バランスの割合が 100% を下回ることのないインスタンスは縮小する必要があります。詳細については、CloudWatch を使用したインスタンスのモニタリングを参照してください。

ハイメモリインスタンスは大規模なインメモリデータベースを実行するよう設定されており、これにはクラウド内の SAP HANA インメモリデータベースの本番デプロイメントを含みます。EBS パフォーマンスを最大化するには、プロビジョニングされたパフォーマンスが同じで偶数の io1 または io2 ボリュームを持つハイメモリインスタンスを使用します。例えば、IOPS 負荷の高いワークロードの場合は、40,000 のプロビジョンド IOPS を持つ 4 つの io1 または io2 ボリュームを使用して、最大 160,000 のインスタンス IOPS を取得します。同様に、スループットの多いワークロードの場合は、48,000 のプロビジョンド IOPS を持つ 6 つの io1 または io2 ボリュームを使用して、最大 4,750 MB/秒のスループットを取得します。その他の推奨事項については、SAP HANA のストレージ構成を参照してください。

考慮事項

- 2020 年 2 月 26 日以降に起動された G4dn、I3en、 Inf1、M5a、M5ad、R5a、R5ad、T3、T3a、および Z1d インスタンスは、上記の表に記載されている最大のパフォーマンスを提供します。2020 年 2 月 26 日より前に起動されたインスタンスのパフォーマンスを最大化するには、インスタンスを停止してから起動します。
- 2019 年 12 月 3 日以降に起動された C5、C5d、C5n、M5、M5d、M5n、M5dn、R5、R5d、R5n、R5dn、P3dn の各インスタンスは、上記の表に一覧されて

いる最大のパフォーマンスを提供します。2019 年 12 月 3 日より前に起動されたインスタンスから最大のパフォーマンスを得るには、インスタンスを停止してから起動します。

- 2020 年 3 月 12 日以降に起動された u-6tb1.metal、u-9tb1.metal、および u-12tb1.metal インスタンスは、上記の表のパフォーマンスを提供します。2020 年 3 月 12 日より前に開始されたこれらのタイプのインスタンスのパフォーマンスはそれより低い可能性があります。2020 年 3 月 12 日より前に起動されたインスタンスから最大限のパフォーマンスを得るには、アカウントチームに連絡して、インスタンスをアップグレードしてください (追加料金なし)。

EBS 最適化をサポートするインスタンスタイプを表示する

AWS CLI を使用して、現在のリージョンで EBS 最適化をサポートするインスタンスタイプを表示します。

EBS 最適化をサポートするインスタンスタイプを表示して、EBS 最適化をデフォルトで有効に設定するには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instance-types \
--query 'InstanceTypes[].{InstanceType:InstanceType,"MaxBandwidth(Mb/
s)":EbsInfo.EbsOptimizedInfo.MaximumBandwidthInMbps,MaxIOPS:EbsInfo.EbsOptimizedInfo.MaximumIop
s)":EbsInfo.EbsOptimizedInfo.MaximumThroughputInMBps}' \
--filters Name=ebs-info.ebs-optimized-support,Values=default --output=table
```

eu-west-1 の出力例:

```
-------------------------------------------------------------------------------
|                            DescribeInstanceTypes                            |
+-------------------+---------------------+-----------+-----------------------+
|   InstanceType    | MaxBandwidth(Mb/s)  |  MaxIOPS  |  MaxThroughput(MB/s)  |
+-------------------+---------------------+-----------+-----------------------+
|  m5dn.8xlarge     |  6800               |  30000    |  850.0                |
|  m6gd.xlarge      |  4750               |  20000    |  593.75               |
|  c4.4xlarge       |  2000               |  16000    |  250.0                |
|  r4.16xlarge      |  14000              |  75000    |  1750.0               |
|  m5ad.large       |  2880               |  16000    |  360.0                |
...
```

EBS 最適化をサポートするインスタンスタイプを表示して、EBS 最適化がデフォルトで有効にしないようにするには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instance-types \
--query 'InstanceTypes[].{InstanceType:InstanceType,"MaxBandwidth(Mb/
s)":EbsInfo.EbsOptimizedInfo.MaximumBandwidthInMbps,MaxIOPS:EbsInfo.EbsOptimizedInfo.MaximumIop
s)":EbsInfo.EbsOptimizedInfo.MaximumThroughputInMBps}' \
--filters Name=ebs-info.ebs-optimized-support,Values=supported --output=table
```

eu-west-1 の出力例:

```
-------------------------------------------------------------------------
|                          DescribeInstanceTypes                        |
+--------------+----------------------+----------+----------------------+
| InstanceType | MaxBandwidth(Mb/s)   | MaxIOPS  |  MaxThroughput(MB/s)  |
+--------------+----------------------+----------+----------------------+
|  i2.2xlarge  |  1000                |  8000    |  125.0                |
|  m2.4xlarge  |  1000                |  8000    |  125.0                |
|  m2.2xlarge  |  500                 |  4000    |  62.5                 |
|  c1.xlarge   |  1000                |  8000    |  125.0                |
|  i2.xlarge   |  500                 |  4000    |  62.5                 |
|  m3.xlarge   |  500                 |  4000    |  62.5                 |
|  m1.xlarge   |  1000                |  8000    |  125.0                |
|  r3.4xlarge  |  2000                |  16000   |  250.0                |
|  r3.2xlarge  |  1000                |  8000    |  125.0                |
|  c3.xlarge   |  500                 |  4000    |  62.5                 |
|  m3.2xlarge  |  1000                |  8000    |  125.0                |
|  r3.xlarge   |  500                 |  4000    |  62.5                 |
|  i2.4xlarge  |  2000                |  16000   |  250.0                |
|  c3.4xlarge  |  2000                |  16000   |  250.0                |
|  c3.2xlarge  |  1000                |  8000    |  125.0                |
|  m1.large    |  500                 |  4000    |  62.5                 |
+--------------+----------------------+----------+----------------------+
```

起動時の EBS 最適化の有効化

インスタンスの最適化を有効にするには、EBS 最適化の属性を設定します。

コンソールを使用してインスタンスを起動するときに Amazon EBS 最適化を有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [インスタンスの作成] を選択します。
3. [Step 1: Choose an Amazon Machine Image (AMI)] で、AMI を選択します。

4. [Step 2: Choose an Instance Type] で、サポート対象の Amazon EBS 最適化として一覧表示されているインスタンスタイプを選択します。
5. [Step 3: Configure Instance Details] で必要なフィールドに入力し、[Launch as EBS-optimized instance] を選択します。前のステップで選択したインスタンスタイプが Amazon EBS 最適化をサポートしていない場合、このオプションは存在しません。選択したインスタンスタイプがデフォルトで Amazon EBS に最適化される場合、このオプションが選択されており、選択を解除することはできません。
6. 指示に従ってウィザードを完了し、インスタンスを起動します。

コマンドラインを使用してインスタンスを起動するときに EBS 最適化を有効にするには

次のいずれかのコマンドを対応するオプションで使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- `--ebs-optimized` (AWS CLI) を使用した run-instances
- `-EbsOptimized` (AWS Tools for Windows PowerShell) を使用した New-EC2Instance

既存のインスタンスの EBS 最適化の有効化

既存のインスタンスの最適化を有効または無効にするには、 Amazon EBS 最適化インスタンスの属性を変更します。インスタンスが実行中の場合は、まず停止する必要があります。

> **Warning**
>
> インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスストアボリュームのデータを保持するには、データを永続的ストレージに必ずバックアップします。

コンソールを使用して、既存のインスタンスで EBS 最適化を有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. インスタンスを停止し、[Actions (アクション)]、[Instance state (インスタンス状態)]、[Stop instance (インスタンスの停止)] の順に選択します。インスタンスが停止するまで、数分かかる場合があります。

4. インスタンスが選択された状態で、[Actions (アクション)]、[Instance settings (インスタンス設定)]、[Change instance type (インスタンスタイプの変更)] の順に選択します。
5. [Change Instance Type (インスタンスタイプの変更)] で、次のいずれかの操作を行います。
   - 目的のインスタンスのインスタンスタイプがデフォルトで Amazon EBS に最適化される場合、[EBS-optimized] が選択されており、変更できません。そのインスタンスでは Amazon EBS 最適化がすでに有効であるため、[Cancel] をクリックします。
   - 目的のインスタンスのインスタンスタイプが Amazon EBS 最適化をサポートしている場合は、[EBS-optimized (EBS 最適化)]、[Apply (適用)] の順に選択します。
   - 目的のインスタンスのインスタンスタイプが Amazon EBS 最適化をサポートしていない場合は、[EBS 最適化] を選択することはできません。[Instance Type (インスタンスタイプ)] から、Amazon EBS 最適化をサポートするインスタンスタイプを選択し、[EBS-optimized (EBS 最適化)]、[Apply (適用)] の順に選択します。
6. [Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択します。

コマンドラインを使用して、既存のインスタンスで EBS 最適化を有効にするには

1. インスタンスが実行中の場合は、次のいずれかのコマンドを使用して停止します。
   - stop-instances (AWS CLI)
   - Stop-EC2Instance (AWS Tools for Windows PowerShell)
2. EBS 最適化を有効にするには、次のいずれかのコマンドと対応するオプションを使用します。
   - `--ebs-optimized` (AWS CLI) を使用した modify-instance-attribute
   - `-EbsOptimized` (AWS Tools for Windows PowerShell) を使用した Edit-EC2InstanceAttribute

インスタンス購入オプション

Amazon EC2 には、ニーズに基づいてコストを最適化するための以下の購入オプションがあります。

- [オンデマンドインスタンス] – 起動するインスタンスに対して秒単位でお支払いいただきます。
- Savings Plans – 1〜3 年の期間、1 時間 あたり USD 単位で一定の使用量を契約することにより、Amazon EC2 にかかるコストを削減します。

- リザーブドインスタンス – 1 ~ 3 年の期間、インスタンスタイプとリージョンを含むインスタンス設定を維持する契約により、Amazon EC2 にかかるコストを削減します。
- スポットインスタンス – 未使用の EC2 インスタンスをリクエストすることで、Amazon EC2にかかるコストを大幅に削減できます。
- Dedicated Hosts – インスタンスの実行のみを目的とした物理ホストに対してお支払いいただきます。ソケット単位、コア単位、または VM 単位で、ご使用中のソフトウェアライセンスを持ち込むことでコストを削減できます。
- ハードウェア専有インスタンス – シングルテナントハードウェアで実行されるインスタンスに対して、時間単位でお支払いいただきます。
- キャパシティー予約 – 特定のアベイラビリティーゾーンの EC2 インスタンスのキャパシティを予約できます。

キャパシティ予約が必要な場合は、特定のアベイラビリティーゾーンのリザーブドインスタンスまたはキャパシティ予約を購入します。キャパシティブロックを使用すると GPU インスタンスのクラスターを予約できます。スポットインスタンス は、アプリケーションを実行するタイミングに柔軟性がある場合や、アプリケーションを中断できる場合に費用効率の高い選択肢です。Dedicated Hosts または Dedicated Instances (ハードウェア専有インスタンス) を使用すると、サーバーにバインドされた既存のソフトウェアライセンスを使用することにより、コンプライアンス要件に対応しながらコストを削減できます。詳細については、「Amazon EC2 の料金表」を参照してください。

Savings Plans の詳細については、Savings Plans ユーザーガイドを参照してください。

コンテンツ

インスタンスのライフサイクルの決定

インスタンスのライフサイクルは起動時に開始され、終了時に終了されます。選択する購入のオプションにより、インスタンスのライフサイクルに影響があります。例えば、起動時に オンデマンド

インスタンス が実行され、終了時に実行が終了されます。スポットインスタンス は、利用可能なキャパシティーがあり、スポット料金が上限価格以下である限り実行されます。

次のメソッドを使用して、インスタンスのライフサイクルを決定します。

コンソールを使用してインスタンスのライフサイクルを決定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択します。
4. [Details (詳細)] タブの [Instance details (インスタンスの詳細)] で、[Lifecycle (ライフサイクル)] を見つけます。値が `spot` の場合、そのインスタンスはスポットインスタンスです。値が `normal` の場合、インスタンスは オンデマンドインスタンス または リザーブドインスタンス です。
5. [Details (詳細)] タブの [Host and placement group (ホストとプレイスメントグループ)] で、[Tenancy (テナンシー)] を見つけます。値が `host` の場合、インスタンスは Dedicated Host で実行されています。値が `dedicated` の場合、インスタンスは ハードウェア専有インスタンス で実行されています。
6. (オプション) リザーブドインスタンス を購入し、適用されていることを確認するには、Amazon EC2 の使用状況レポートを確認できます。詳細については、Amazon EC2 使用状況レポート を参照してください。

AWS CLIを使用してインスタンスのライフサイクルを決定するには

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instances --instance-ids i-1234567890abcdef0
```

インスタンスが Dedicated Host で実行されている場合、出力には次の情報が含まれます。

```
"Tenancy": "host"
```

インスタンスが ハードウェア専有インスタンス の場合、出力には次の情報が含まれます。

```
"Tenancy": "dedicated"
```

インスタンスがスポットインスタンスの場合、出力には次の情報が含まれます。

```
"InstanceLifecycle": "spot"
```

それ以外の場合、出力には InstanceLifecycle は含まれません。

オンデマンドインスタンス

オンデマンドインスタンスでは長期契約は必要なく、料金はコンピューティング性能に対して秒単位で発生します。そのインスタンスライフサイクルを完全に制御でき、いつ起動、停止、休止、開始、再起動、または終了するかを決定できます。

オンデマンドインスタンス を購入するときに要求される長期的なコミットメントはありません。ご利用の オンデマンドインスタンス が running 状態になっている秒数 (最低 60 秒) に対してのみお支払いいただきます。オンデマンドインスタンス が実行される秒数あたりの料金は固定で、Amazon EC2 の料金、オンデマンド料金ページに記載されています。

短期間、不規則なワークロードがあり中断できないアプリケーションには、オンデマンドインスタンス の使用をお勧めします。

オンデマンドインスタンスで料金を大幅に削減するには、AWS Savings Plans、スポットインスタンス、または Reserved Instances を使用してください。

目次

オンデマンドインスタンス の操作

次の方法で オンデマンドインスタンス を使用できます。

- インスタンスの起動
- Linux インスタンスへの接続
- インスタンスの停止と起動
- Amazon EC2 インスタンスの休止
- インスタンスの再起動

- インスタンスのリタイア
- インスタンスの終了
- インスタンスの復旧
- Amazon Linux インスタンスの設定
- EC2 Linux インスタンスを特定する

Amazon EC2 を初めて使用する場合は、Amazon EC2 の使用を開始する を参照してください。

オンデマンドインスタンスクォータ

各リージョンごとに、AWS アカウント ごとに実行中のオンデマンド インスタンスの数に対するクォータがあります。オンデマンドインスタンスのクォータは、インスタンスタイプに関係なく、実行中のオンデマンドインスタンスで使用している仮想中央演算装置 (vCPU) の数で管理されます。

オンデマンドインスタンスには以下のクォータタイプがあります。

- オンデマンド DL インスタンスの実行
- オンデマンド F インスタンスの実行
- オンデマンド G および VT インスタンスの実行
- オンデマンドハイメモリインスタンスの実行
- オンデマンドオール HPC インスタンスの実行
- オンデマンド Inf インスタンスの実行
- オンデマンド P インスタンスの実行
- オンデマンド標準 (A、C、D、H、I、M、R、T、Z) インスタンスの実行
- オンデマンド Trn インスタンスの実行
- オンデマンド X インスタンスの実行

クォータは実行中のインスタンスにのみ適用されます。インスタンスが保留中、停止中、停止済み、または休止状態の場合、クォータにはカウントされません。

各クォータタイプは、1 つ以上のインスタンスファミリーに対し、最大の vCPU 数を指定しています。さまざまなインスタンスファミリー、世代、およびサイズの詳細については、Amazon EC2 インスタンスタイプを参照してください。

vCPU の数が自分のアカウントでのクォータを超えていない限り、変化するアプリケーションのニーズに合わせて、任意の組み合わせでインスタンスタイプを起動できます。例えば、256 vCPU の

クォータがあるスタンダードインスタンスでは、32 個の `m5.2xlarge` インスタンス (32 x 8 vCPU) または 16 個の `c5.4xlarge` インスタンス (16 x 16 vCPU) を起動できます。詳細については、「EC2 オンデマンドインスタンス の制限」を参照してください。

タスク

- オンデマンドインスタンス のクォータと使用量のモニタリング
- クォータ引き上げをリクエストする

オンデマンドインスタンス のクォータと使用量のモニタリング

次の方法を使用して、各リージョンのオンデマンド インスタンス クォータを表示および管理できます。

Service Quotas コンソールを使用して現在のクォータを表示するには

1. Service Quotas コンソール (https://console.aws.amazon.com/servicequotas/home/services/ec2/quotas/) を開きます。
2. ナビゲーションバーから、リージョンを選択します。
3. フィルターフィールドに、**`On-Demand`** と入力します。
4. [適用されたクォータ値] 列には、アカウントの各オンデマンドインスタンスのクォータタイプの vCPU の最大数が表示されます。

AWS Trusted Advisor コンソールを使用して現在のクォータを表示するには

AWS Trusted Advisor コンソールのサービスの制限ページを開きます。

CloudWatch アラームを設定するには

Amazon CloudWatch のメトリクス統合では、クォータに対して EC2 の使用量をモニタリングできます。クォータに近づいたときに警告を発するようにアラームを設定することもできます。詳細については、「Service Quotas」ユーザーガイドのサービスクォータと Amazon CloudWatch アラームを参照してください。

クォータ引き上げをリクエストする

オンデマンドインスタンスの上限は、使用量に基づき Amazon EC2 によって自動的に引き上げられますが、必要であればクォータの引き上げをリクエストすることも可能です。例えば、現在のクォータで許可されているよりも多くのインスタンスを起動する場合は、前のセクション「Amazon EC2

の Service Quotas」に記載されている Service Quotas コンソールで説明したように、 を使用してクォータの増加を要求できます。

オンデマンドインスタンスの料金を照会する

Price List Service API または AWS Price List API を使用して、オンデマンドインスタンス の料金を照会できます。詳細については、AWSユーザーガイドの「AWS Billing Price List API の使用」を参照してください。

Reserved Instances

オンデマンドインスタンスの料金と比較して、リザーブドインスタンスでは Amazon EC2 の料金を大幅に節約することができます。リザーブドインスタンスは物理インスタンスではありませんが、請求の割引はアカウントでのオンデマンドインスタンスの使用に適用されます。請求割引のメリットを得るには、これらのオンデマンドインスタンスは、インスタンスタイプやリージョンなどの特定の属性に一致する必要があります。

> **Note**
>
> Savings Plans でも、オンデマンドインスタンスの料金と比べて Amazon EC2 コストの大幅な割引を受けられます。Savings Plans では、1 時間 につき USD 単位で一定の使用量を守ります。これによって、特定のインスタンス設定を使用することにコミットせずに、ニーズに最も適したインスタンス設定を使用して、コストを削減し続ける柔軟性が得られます。詳細については、AWS Savings Plans ユーザーガイドをご参照ください。

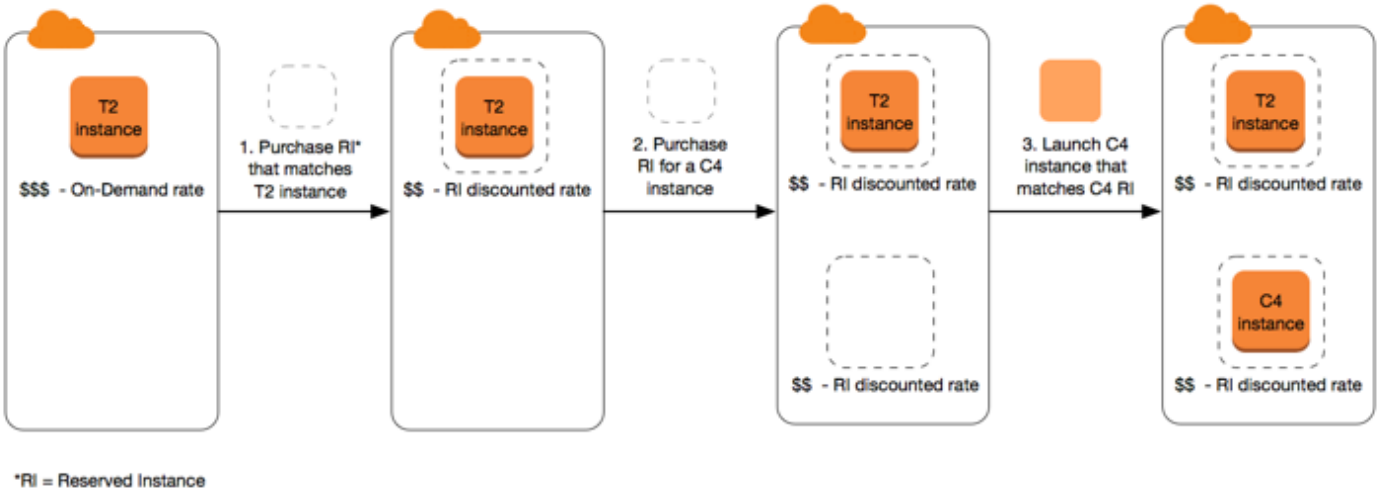
リザーブドインスタンスのトピック

- リザーブドインスタンス の概要
- リザーブドインスタンス 料金を決定するキー変数
- リージョンおよびゾーン リザーブドインスタンス (スコープ)
- リザーブドインスタンス のタイプ (提供しているクラス)
- リザーブドインスタンス がどのように適用されるか
- お使いの リザーブドインスタンス の使用
- 課金の仕組み
- リザーブドインスタンスの購入

- Reserved Instance Marketplace での販売
- リザーブドインスタンス の変更
- コンバーティブルリザーブドインスタンス の交換
- リザーブドインスタンスのクォータ

リザーブドインスタンス の概要

次の図では、リザーブドインスタンス の購入と使用の基本的な概要を示します。



このシナリオでは、現在オンデマンドレートを支払っているアカウントの オンデマンドインスタンス (T2) を実行しています。実行しているインスタンスの属性を一致する リザーブドインスタンス を購入すると、料金上の利点が即時適用されます。次に、C4 インスタンスに リザーブドインスタンス を購入します。この リザーブドインスタンス に属性が一致するアカウントで実行しているインスタンスはありません。この最終ステップでは、C4 リザーブドインスタンス の属性と一致するインスタンスを起動すると、料金上の利点が独自適用されます。

リザーブドインスタンス 料金を決定するキー変数

リザーブドインスタンス 料金は、次のキー変数によって決まります。

インスタンスの属性

リザーブドインスタンス には、その料金を決める 4 つのインスタンス属性があります。

- インスタンスタイプ: 例えば、`m4.large`。これは、インスタンスファミリー (`m4` など) とインスタンスサイズ (`large` など) で構成されます。

- リージョン: リザーブドインスタンスが購入されているリージョン。
- テナンシー: インスタンスが共有 (デフォルト) または単一のテナンシー (専用) のハードウェアで実行されるかについて。詳細については、Dedicated Instances を参照してください。
- プラットフォーム: オペレーティング システム。例えば、Windows や Linux/Unix。詳細については、プラットフォームの選択 を参照してください。

コミットメント期間

1 年あるいは 3 年のコミットメントで リザーブドインスタンス を購入することができます。3 年のコミットメントには大幅な割引が提供されます。

- 1 年: 1 年は 31536000 秒 (365 日) として定義されます。
- 3 年: 3 年は 94608000 秒 (1095 日) として定義されます。

リザーブドインスタンス は自動的に更新されません。有効期限が切れても、引き続き EC2 インスタンスを使用できますが、オンデマンド価格が課金されます。上記の例では、T2 および C4 インスタンスを対象とする リザーブドインスタンス の期限が切れた場合、インスタンスが終了するまでオンデマンドレートの支払いに戻るか、あるいはインスタンスの属性に一致する新しい リザーブドインスタンス を購入します。

> **Important**
>
> リザーブドインスタンス を購入した後で購入をキャンセルすることはできません。ただし、ユーザーのニーズが変更した場合、リザーブドインスタンス を変更、交換、売却できる場合もあります。

支払いオプション

リザーブドインスタンス では次の支払いオプションが用意されています。

- すべて前払い: 期間の開始時に全額が支払われ、使用時間数に関係なく、残りの期間にその他のコストや追加時間課金は生じません。
- 一部前払い: 料金の一部を前払いする必要があり、期間内の残りの時間は、リザーブドインスタンス が使用されたどうかにかかわらず、割引された時間料金で請求されます。
- 前払いなし: リザーブドインスタンス が使用されたどうかにかかわらず、期間内のすべての時間は割引時間料金での請求となります。前払い料金は必要ではありません。

> **Note**
>
> 前払いなしの リザーブドインスタンス は、予約の全期間について毎月支払いを行う契約義務に基づいています。そのため、前払いなしの リザーブドインスタンス を購入するには、請求履歴に問題がないことが必須となります。

一般的には、リザーブドインスタンス の前払い額を高く設定するほど、より多くの費用を節約できます。また、Reserved Instance Marketplace では、サードパーティーの販売者が提供する、より安価で期間の短いリザーブドインスタンス を見つけることもできます。詳細については、Reserved Instance Marketplace での販売 を参照してください。

提供クラス

コンピューティングに変更が必要な場合、提供クラスによって、リザーブドインスタンス を変更または交換することができます。

- スタンダード: 最大の割引を提供しますが、変更のみを行うことができます。スタンダード リザーブドインスタンス は交換できません。
- コンバーティブル: スタンダード リザーブドインスタンス より少ない割引ですが、異なるインスタンス属性を使用する別のコンバーティブル リザーブドインスタンス と交換できます。コンバーティブル リザーブドインスタンス は変更することもできます。

詳細については、リザーブドインスタンス のタイプ (提供しているクラス) を参照してください。

> **Important**
>
> リザーブドインスタンス を購入した後で購入をキャンセルすることはできません。ただし、ユーザーのニーズが変更した場合、リザーブドインスタンス を変更、交換、売却できる場合もあります。

詳細については、「Amazon EC2 リザーブドインスタンスの料金表」ページを参照してください。

リージョンおよびゾーン リザーブドインスタンス (スコープ)

リザーブドインスタンス の購入時に リザーブドインスタンス のスコープを決定します。スコープは、リージョンあるいはゾーンのいずれかになります。

- リージョナル: リージョン用に リザーブドインスタンス を購入する場合、これはリージョナル リザーブドインスタンス と呼ばれます。
- ゾーン: 特定のアベイラビリティーゾーン用に リザーブドインスタンス を購入する場合、これはゾーンリザーブドインスタンス と呼ばれます。

範囲は料金には影響しません。リージョンまたはゾーンごとの リザーブドインスタンス に同じ料金を支払います。リザーブドインスタンス の料金の詳細については、「リザーブドインスタンス 料金を決定するキー変数」と「Amazon EC2 リザーブドインスタンスの料金」を参照してください。

リザーブドインスタンスのスコープを指定する方法の詳細については、「RI の属性」、特に「アベイラビリティーゾーン」の箇条書きを参照してください。

リージョンとゾーンの リザーブドインスタンス の違い

次のテーブルでは、リージョン リザーブドインスタンス とゾーン リザーブドインスタンス の主な違いをいくつか示しています。

	リージョン リザーブドインスタンス	ゾーン リザーブドインスタンス
キャパシティーを予約する機能	リージョンの リザーブドインスタンス では、キャパシティーは予約されません。	ゾーンの リザーブドインスタンス では、指定されたアベイラビリティーゾーンでキャパシティーが予約されます。
アベイラビリティーゾーンの柔軟性	指定するリージョン内のすべてのアベイラビリティーゾーンにおけるインスタンスの使用に対して、リザーブドインスタンス 割引が適用されます。	アベイラビリティーゾーンの柔軟性なし — リザーブドインスタンス 割引は、指定したアベイラビリティー–ゾーン内のみのインスタンスの使用に対して適用されます。
インスタンスサイズの柔軟性	インスタンスファミリー内のインスタンスの使用に対して、サイズを問わず、リザー	インスタンスサイズの柔軟性なし — リザーブドインスタンス 割引は、指定されたインスタンスタイプとサイズにおけ

	リージョン リザーブドインスタンス	ゾーン リザーブドインスタンス
	ブドインスタンス 割引が適用されます。 Amazon Linux/Unix リザーブドインスタンス のデフォルトテナンシーのみでサポートされます。詳細については、正規化係数によって決定されたインスタンスサイズの柔軟性 を参照してください。	るインスタンスの使用に対してのみ適用されます。
購入をキューに入れる	リージョンリザーブドインスタンスの購入をキューに入れることができます。	ゾーンリザーブドインスタンスの購入をキューに入れることはできません。

詳細な説明と例については、「リザーブドインスタンス がどのように適用されるか」を参照してください。

リザーブドインスタンス のタイプ (提供しているクラス)

リザーブドインスタンス の提供クラスは、スタンダードまたはコンバーチブルのいずれかです。スタンダード リザーブドインスタンス は、コンバーチブル リザーブドインスタンス よりも大幅な割引が受けられますが、スタンダード リザーブドインスタンス を交換することはできません。コンバーチブル リザーブドインスタンス は交換できます。スタンダードおよびコンバーチブルのリザーブドインスタンスは変更可能です。

リザーブドインスタンス の設定は、期間内の 1 つのインスタンスタイプ、プラットフォーム、スコープ、およびテナンシーで構成されています。コンピューティングに変更が必要な場合は、リザーブドインスタンス を変更または交換できる可能性があります。

スタンダードとコンバーチブル リザーブドインスタンス の違い

以下に、スタンダードとコンバーティブル リザーブドインスタンス の違いを示します。

	スタンダード リザーブドインスタンス	Convertible Reserved Instance
リザーブドインスタンス の変更	一部の属性は変更できます。詳細については、リザーブドインスタンス の変更 を参照してください。	一部の属性は変更できます。詳細については、リザーブドインスタンス の変更 を参照してください。
リザーブドインスタンスの交換	交換できません。	期間内で、インスタンスファミリー、インスタンスタイプ、プラットフォーム、スコープやテナンシーなどの新しい属性の別の コンバーティブルリザーブドインスタンスに交換することができます。詳細については、コンバーティブルリザーブドインスタンス の交換 を参照してください。
Reserved Instance Marketplace での販売	Reserved Instance Marketplace で販売可能です。	Reserved Instance Marketplace では販売できません。
リザーブドインスタンスマーケットプレイスでの購入	Reserved Instance Marketplace で購入可能です。	Reserved Instance Marketplace では購入できません。

リザーブドインスタンス がどのように適用されるか

リザーブドインスタンスは物理インスタンスではありませんが、請求の割引はアカウントでのオンデマンドインスタンスの実行に適用されます。請求割引のメリットを得るには、オンデマンドインスタンスは、リザーブドインスタンスの特定の仕様に一致する必要があります。

リザーブドインスタンスを購入し、リザーブドインスタンスの仕様と一致するオンデマンドインスタンスを既に実行している場合、請求割引は直ちに自動的に適用されます。インスタンスを再起動する必要はありません。使用可能な実行中のオンデマンドインスタンスが存在しない場合は、リザーブドインスタンスと同じ仕様でオンデマンドインスタンスを起動します。詳細については、お使いの リザーブドインスタンス の使用 を参照してください。

リザーブドインスタンスの提供クラス (スタンダードまたはコンバーティブル) は、請求割引の適用方法には影響しません。

トピック

ゾーン リザーブドインスタンス がどのように適用されるか

特定のアベイラビリティーゾーンでキャパシティを予約するために購入されるリザーブドインスタンスは、ゾーンリザーブドインスタンスと呼ばれます。

- リザーブドインスタンスの割引は、そのアベイラビリティーゾーンにおける一致したインスタンスの使用に適用されます。
- 実行中のインスタンスの属性 (テナンシー、プラットフォーム、アベイラビリティーゾーン、インスタンスタイプ、およびインスタンスサイズ) は、リザーブドインスタンス の属性と一致する必要があります。

例えば、us-east-1a のアベイラビリティーゾーンで、デフォルトテナンシーの `c4.xlarge` Linux/Unix スタンダードリザーブドインスタンスを 2 つ購入すると、us-east-1a のアベイラビリティーゾーンで実行している 2 つまでのデフォルトテナンシーの `c4.xlarge` Linux/Unix インスタンスでリザーブドインスタンス割引を利用できます。

リージョンリザーブドインスタンスがどのように適用されるか

リージョン用に購入されるリザーブドインスタンスは、リージョンリザーブドインスタンスと呼ばれ、アベイラビリティーゾーンとインスタンスサイズの柔軟性を提供します。

- このリージョン内のすべてのアベイラビリティーゾーンにおけるインスタンスの使用に対して、リザーブドインスタンス 割引が適用されます。
- リザーブドインスタンスの割引は、サイズに関係なく、インスタンスファミリー内のインスタンスの使用に適用されます。これは「インスタンスサイズの柔軟性」と呼ばれます。

インスタンスサイズの柔軟性

インスタンスサイズの柔軟性により、ファミリー、世代、および属性が同一のインスタンスの使用に対してリザーブドインスタンス割引が適用されます。リザーブドインスタンスは、インスタンスファミリー内の最もサイズの小さいインスタンスからサイズの大きいインスタンスへ、正規化係数に基づいて適用されます。リザーブドインスタンス割引の適用例については、「シナリオ 2: 正規化係数を使用した 1 つのアカウントのリザーブドインスタンス」を参照してください。

制限事項

- サポート: インスタンスサイズの柔軟性は、リージョナルリザーブドインスタンスでのみサポートされています。
- 未サポート: 次のリザーブドインスタンスでは、インスタンスサイズの柔軟性はサポートされていません。
  - 特定のアベイラビリティーゾーン (ゾーンリザーブドインスタンス) 用に購入されたリザーブドインスタンス
  - G4ad、G4dn、G5、G5g、Inf1、Inf2 の各インスタンス用のリザーブドインスタンス
  - リザーブドインスタンス for Windows Server、Windows Server with SQL Standard、Windows Server with SQL Server Enterprise、Windows Server with SQL Server Web、RHEL、SUSE Linux Enterprise Server
  - 専有テナントを使用する リザーブドインスタンス

正規化係数によって決定されたインスタンスサイズの柔軟性

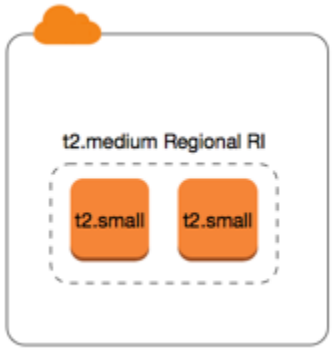
インスタンスサイズの柔軟性は、インスタンスサイズの正規化係数によって決定されます。割引は、リージョン内のアベイラビリティーゾーンで、予約したインスタンスサイズによって、同じインスタンスファミリーで実行中のインスタンスに完全または部分的に適用されます。一致する必要がある属性は、インスタンスファミリー、テナンシー、プラットフォームのみです。

次の表は、インスタンスファミリー内のさまざまなサイズおよび対応する正規化係数の一覧です。このスケールを使用して、リザーブドインスタンス の割引料金をインスタンスファミリーの正規化された使用に適用します。

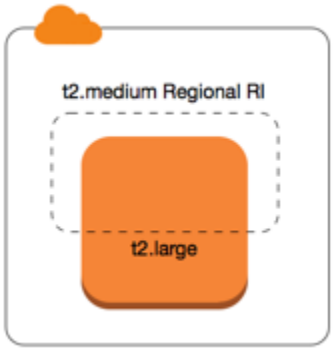
インスタンスサイズ	正規化係数
nano	0.25

インスタンスサイズ	正規化係数
micro	0.5
small	1
medium	2
large	4
xlarge	8
2xlarge	16
3xlarge	24
4xlarge	32
6xlarge	48
8xlarge	64
9xlarge	72
10xlarge	80
12xlarge	96
16xlarge	128
18xlarge	144
24xlarge	192
32xlarge	256
48xlarge	384
56xlarge	448
112xlarge	896

例えば、t2.medium インスタンスには 2 の正規化係数があります。US East (N. Virginia) で t2.medium デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入し、このリージョンのアカウントで 2 つの t2.small インスタンスを実行している場合、料金上の利点はどちらのインスタンスにも完全に適用されます。



または、US East (N. Virginia) リージョンのアカウントで実行している 1 つの t2.large インスタンスがある場合、料金上の利点はこのインスタンスの使用の 50% に適用されます。



正規化係数は、リザーブドインスタンス の変更時にも適用されます。詳細については、リザーブドインスタンス の変更 を参照してください。

ベアメタルインスタンスの正規化係数

また、インスタンスサイズの柔軟性は、インスタンスファミリー内のベアメタルインスタンスにも適用されます。ベアメタルインスタンスで共有するテナントを使用したリージョンの Amazon Linux/Unix リザーブドインスタンス がある場合、同じインスタンスファミリー内で リザーブドインスタンス 割引の特典を受けることができます。逆の場合も同様です。同じファミリー内のインスタンスでベアメタルインスタンスとしてテナントを共有している、リージョン Amazon Linux/Unix リザーブドインスタンス がある場合、ベアメタルインスタンスで リザーブドインスタンス 割引の特典を受けることができます。

metal インスタンスサイズには、単一の正規化係数がありません。ベアメタルインスタンスの正規化係数は、同じインスタンスファミリー内の同等な仮想インスタンスサイズと同じです。例えば、1 つの i3.metal インスタンスには i3.16xlarge インスタンスと同じ正規化係数があります。

インスタンスサイズ	正規化係数
a1.metal	32
m5zn.metal | z1d.metal	96
c6g.metal | c6gd.metal | i3.metal | m6g.metal | m6gd.metal | r6g.metal | r6gd.metal | x2gd.metal	128
c5n.metal	144
c5.metal | c5d.metal | i3en.metal | m5.metal | m5d.metal | m5dn.metal | m5n.metal | r5.metal | r5b.metal | r5d.metal | r5dn.metal | r5n.metal	192
u-*.metal	896

例えば、1 つの i3.metal インスタンスには 128 の正規化係数があります。US East (N. Virginia) で i3.metal デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入した場合、料金上のメリットは次のようになります。

- そのリージョン内のアカウントで実行している 1 つの i3.16xlarge がある場合、料金上のメリットは i3.16xlarge インスタンス全体に適用されます (i3.16xlarge 正規化係数 = 128)。
- あるいは、そのリージョン内のアカウントで実行している 2 つの i3.8xlarge がある場合、料金上のメリットは両方の i3.8xlarge インスタンス全体に適用されます (i3.8xlarge 正規化係数 = 64)。
- あるいは、そのリージョン内のアカウントで実行している 4 つの i3.4xlarge がある場合、料金上のメリットは 4 つの i3.4xlarge インスタンス全体に適用されます (i3.4xlarge 正規化係数 = 32)。

逆の場合も同様です。例えば、US East (N. Virginia) で 2 つの i3.8xlarge デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入し、そのリージョンで実行している 1 つの

i3.metal インスタンスがある場合、料金上のメリットは i3.metal インスタンス全体に適用されます。

リザーブドインスタンス の適用例

リザーブドインスタンス の適用方法を以下のシナリオで示します。

- シナリオ 1: 単一アカウントの リザーブドインスタンス
- シナリオ 2: 正規化係数を使用した 1 つのアカウントのリザーブドインスタンス
- シナリオ 3: 連結アカウントのリージョンリザーブドインスタンス
- シナリオ 4: 連結アカウントのゾーンリザーブドインスタンス

シナリオ 1: 単一アカウントの リザーブドインスタンス

アカウント A で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 4 つのデフォルトテナンシーの m3.large Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m4.xlarge Amazon Linux インスタンス
- us-east-1c アベイラビリティーゾーンで 1 つのデフォルトテナンシーの c4.xlarge Amazon Linux インスタンス

アカウント A で以下の リザーブドインスタンス を購入するとします。

- us-east-1a アベイラビリティーゾーンで 4 つのデフォルトテナンシーの m3.large Linux リザーブドインスタンス (キャパシティーの予約あり)
- us-east-1 リージョンで 4 つのデフォルトテナンシーの m4.large Amazon Linux リザーブドインスタンス
- us-east-1 リージョンで 1 つのデフォルトテナンシーの c4.large Amazon Linux リザーブドインスタンス

リザーブドインスタンス の利点は以下のように適用されます。

- 4 つの m3.large ゾーン リザーブドインスタンス の割引とキャパシティーの予約は、属性 (インスタンスサイズ、リージョン、プラットフォーム、テナンシー) が一致する 4 つの m3.large インスタンスによって使用されます。

- m4.large リージョン リザーブドインスタンス は、デフォルトテナンシーの Amazon Linux リザーブドインスタンス であるため、アベイラビリティーゾーンおよびインスタンスサイズの柔軟性を提供します。

  1 つの m4.large は、4 つの正規化された単位/時間に相当します。

  4 つの m4.large リージョン リザーブドインスタンス を購入したので、合計で 16 の正規化された単位/時間 (4x4) に相当します。アカウント A で実行している 2 つの m4.xlarge インスタンスは、16 の正規化された単位/時間 (2x8) に相当します。この場合、4 つの m4.large リージョナルリザーブドインスタンスによって、2 つの m4.xlarge インスタンスの使用に対する全面的な請求のメリットが提供されます。

- us-east-1 の c4.large リージョン リザーブドインスタンス は、デフォルトテナンシーのリージョンの Amazon Linux リザーブドインスタンス であるため、アベイラビリティーゾーンおよびインスタンスサイズの柔軟性を提供し、c4.xlarge インスタンスに適用されます。c4.large インスタンスは 4 つの正規化された単位/時間に相当し、c4.xlarge インスタンスは 8 つの正規化された単位/時間に相当します。

  この場合、c4.large リージョン リザーブドインスタンス は、c4.xlarge の使用の一部に対してメリットを提供します。これは、この c4.large リザーブドインスタンス は 4 つの正規化された単位/時間に相当しますが、c4.xlarge インスタンスは 8 つの正規化された単位/時間を必要とするためです。したがって、c4.large リザーブドインスタンス の請求割引は、c4.xlarge の使用の 50% に適用されます。c4.xlarge の残りの使用はオンデマンド価格で課金されます。

シナリオ 2: 正規化係数を使用した 1 つのアカウントのリザーブドインスタンス

アカウント A で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m3.xlarge Amazon Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m3.large Amazon Linux インスタンス

アカウント A で以下のリザーブドインスタンスを購入するとします。

- us-east-1 リージョンで 1 つのデフォルトテナンシーの m3.2xlarge Amazon Linux リザーブドインスタンス

リザーブドインスタンス の利点は以下のように適用されます。

- us-east-1 の m3.2xlarge リージョンリザーブドインスタンスは、デフォルトテナンシーのリージョンの Amazon Linux リザーブドインスタンスであるため、アベイラビリティーゾーンおよびインスタンスサイズの柔軟性を提供します。これは最初に m3.large インスタンスに適用され、次に m3.xlarge インスタンスに適用されます (インスタンスファミリー内の最もサイズの小さいインスタンスからサイズの大きいインスタンスへ、正規化係数に基づいて適用されるため)。

  1 つの m3.large インスタンスは、4 つの正規化された単位/時間に相当します。

  1 つの m3.xlarge インスタンスは、8 つの正規化された単位/時間に相当します。

  1 つの m3.2xlarge インスタンスは、16 の正規化された単位/時間に相当します。

  この利点は次のように適用されます。

  m3.2xlarge リージョナルリザーブドインスタンスは、2 つの m3.large の使用に最大の利点をもたらします。これらのインスタンスは、8 つの正規化された単位/時間に相当するためです。これにより、m3.xlarge インスタンスに適用される 8 つの正規化された単位/時間が残ります。

  残りの 8 つの正規化された単位/時間により、m3.2xlarge リージョナルリザーブドインスタンスは、1 つの m3.xlarge の使用に最大の利点をもたらします。各 m3.xlarge インスタンスは、8 つの正規化された単位/時間に相当するためです。m3.xlarge の残りの使用はオンデマンド価格で課金されます。

シナリオ 3: 連結アカウントのリージョンリザーブドインスタンス

リザーブドインスタンス は、最初に購入アカウント内の使用に適用され、次に組織内の他のアカウントの該当する使用に適用されます。詳細については、リザーブドインスタンス および一括請求 (コンソリデーティッドビリング) を参照してください。サイズの柔軟性を提供するリージョン リザーブドインスタンス の場合、インスタンスファミリー内のインスタンスサイズに関係なく、利点がインスタンスに適用されます。

アカウント A (購入しているアカウント) で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m4.xlarge Linux インスタンス

- us-east-1b アベイラビリティーゾーンで 1 つのデフォルトテナンシーの m4.2xlarge Linux インスタンス
- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの c4.xlarge Linux インスタンス
- us-east-1b アベイラビリティーゾーンで 1 つのデフォルトテナンシーの c4.2xlarge Linux インスタンス

別のお客様は、アカウント B— (連結アカウント) で以下の オンデマンドインスタンス を実行しています。

- us-east-1a アベイラビリティーゾーンで 2 つのデフォルトテナンシーの m4.xlarge Linux インスタンス

アカウント A で以下のリージョン リザーブドインスタンス を購入するとします。

- us-east-1 リージョンで 4 つのデフォルトテナンシーの m4.xlarge Linux リザーブドインスタンス
- us-east-1 リージョンで 2 つのデフォルトテナンシーの c4.xlarge Linux リザーブドインスタンス

リージョンリザーブドインスタンスの利点は以下のように適用されます。

- 4 つの m4.xlarge リザーブドインスタンス の割引は、アカウント A の 2 つの m4.xlarge インスタンスと 1 つの m4.2xlarge インスタンスによって使用されます。3 つのインスタンスすべてにおいて、属性 (インスタンスファミリー、リージョン、プラットフォーム、テナンシー) が一致しています。アカウント B (リンクされたアカウント) に リザーブドインスタンス にも一致する 2 つの m4.xlarge がある場合でも、この割引は購入したアカウント (アカウント A) 内のインスタンスにまず適用されます。この リザーブドインスタンス はリージョン リザーブドインスタンスであるため、キャパシティの予約はありません。
- c4.xlarge リザーブドインスタンス インスタンスよりもインスタンスサイズが小さいため、2 つの c4.xlarge リザーブドインスタンスの割引は、2 つの c4.2xlarge インスタンスに適用されます。この リザーブドインスタンス はリージョン リザーブドインスタンス であるため、キャパシティの予約はありません。

シナリオ 4: 連結アカウントのゾーンリザーブドインスタンス

通常、アカウントで所有されている リザーブドインスタンス が、そのアカウントでの使用に最初に適用されます。ただし、組織の他のアカウントに特定のアベイラビリティーゾーン (ゾーン リザーブドインスタンス) の未使用の リザーブドインスタンス がある場合は、これらがアカウントで所有されているリージョン リザーブドインスタンス より先に適用されます。これは、リザーブドインスタンス の使用率を最大限に高めて請求額を下げるための処置です。請求の目的では、組織内のすべてのアカウントが 1 つのアカウントとして扱われます。次の例が、この説明に役立つ場合があります。

アカウント A (購入しているアカウント) で以下の オンデマンドインスタンス を実行しているとします。

- us-east-1a アベイラビリティーゾーンで 1 つのデフォルトテナンシーの `m4.xlarge` Linux インスタンス

お客様は、別の連結アカウント B で以下の オンデマンドインスタンス を実行しています。

- us-east-1b アベイラビリティーゾーンで 1 つのデフォルトテナンシーの `m4.xlarge` Linux インスタンス

アカウント A で以下のリージョン リザーブドインスタンス を購入するとします。

- us-east-1 リージョンで 1 つのデフォルトテナンシーの `m4.xlarge` Linux リザーブドインスタンス

ユーザーが連結アカウント C で以下のゾーン リザーブドインスタンス も購入するとします。

- us-east-1a アベイラビリティーゾーンで 1 つのデフォルトテナンシーの `m4.xlarge` Linux リザーブドインスタンス

リザーブドインスタンス の利点は以下のように適用されます。

- アカウント C で所有されている `m4.xlarge` ゾーン リザーブドインスタンス の割引はアカウント A の `m4.xlarge` の使用に適用されます。
- アカウント A で所有されている `m4.xlarge` リージョン リザーブドインスタンス の割引はアカウント B の `m4.xlarge` の使用に適用されます。

- アカウント A で所有されているリージョン リザーブドインスタンス は、最初にアカウント A での使用に適用されます。アカウント C で所有されているゾーン リザーブドインスタンス は使用されず、アカウント B での使用はオンデマンド価格で課金されます。

詳細については、「Billing and Cost Management レポートの リザーブドインスタンス」を参照してください。

> **Note**
>
> ゾーンリザーブドインスタンスは所有アカウントにのみ容量を予約し、他の AWS アカウント と共有することはできません。他の AWS アカウント と容量を共有する必要がある場合は、On-Demand Capacity Reservations を使用してください。

お使いの リザーブドインスタンス の使用

リザーブドインスタンス は、仕様の一致する実行中の オンデマンドインスタンス に自動的に適用されます。リザーブドインスタンス の仕様と一致する実行中の オンデマンドインスタンス が存在しない場合、必要な仕様が搭載されるインスタンスを起動するまで、リザーブドインスタンス は未使用となります。

リザーブドインスタンスの料金上の利点を利用するためにオンデマンドインスタンスを起動する場合は、オンデマンドインスタンスの設定時に以下の情報を必ず指定してください。

プラットフォーム

リザーブドインスタンスのプラットフォーム (製品の説明) と一致する Amazon マシンイメージ (AMI) を指定する必要があります。例えば、Linux/UNIX をリザーブドインスタンスに指定する場合、Amazon Linux AMI または Ubuntu AMI からインスタンスを起動できます。

インスタンスタイプ

ゾーンリザーブドインスタンスを購入した場合は、リザーブドインスタンスと同じインスタンスタイプを指定する必要があります (例: `t3.large`)。詳細については、ゾーン リザーブドインスタンス がどのように適用されるか を参照してください。

リージョンリザーブドインスタンスを購入した場合は、リザーブドインスタンスのインスタンスタイプと同じインスタンスファミリーからインスタンスタイプを指定する必要があります。例えば、`t3.xlarge` をリザーブドインスタンスに指定する場合は、T3 ファミリーからインスタンス

を起動する必要がありますが、任意のサイズ (例: t3.medium) を指定できます。詳細については、リージョンリザーブドインスタンスがどのように適用されるか を参照してください。

アベイラビリティーゾーン

特定のアベイラビリティーゾーンにゾーンごとのリザーブドインスタンスを購入する場合、同じアベイラビリティーゾーンでインスタンスを起動する必要があります。

リージョンリザーブドインスタンスを購入した場合、リザーブドインスタンスに指定したリージョンの任意のアベイラビリティーゾーンにインスタンスを起動することができます。

テナンシー

インスタンスのテナンシー (dedicated や shared) は、リザーブドインスタンスのテナンシーが一致する必要があります。詳細については、Dedicated Instances を参照してください。

実行しているオンデマンドインスタンスにどのようにリザーブドインスタンスが適用されるかについての例は、「リザーブドインスタンス がどのように適用されるか」を参照してください。詳細については、「Amazon EC2 リザーブドインスタンスが想定通りに AWS 請求書に反映されないのはなぜですか?」を参照してください。

リザーブドインスタンス割引を使用するオンデマンドインスタンスを起動するには、さまざまな方法を使用できます。さまざまな起動方法の詳細については、「インスタンスの起動」を参照してください。Amazon EC2 Auto Scaling を使用してインスタンスを起動することもできます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」を参照してください。

課金の仕組み

すべての リザーブドインスタンス の料金は、オンデマンドインスタンスの料金から割引された額になります。リザーブドインスタンス では、実際の使用に関係なく、全期間の料金をお支払いいただきます。リザーブドインスタンス に特定された支払いオプションによって、リザーブドインスタンス の支払い方法を前払い、一部前払い、月ごとから選択できます。

リザーブドインスタンス 期間が終了すると、EC2 インスタンスの使用についてオンデマンド価格が課金されます。リザーブドインスタンス の購入を最大 3 年先までキューに入れることができます。これにより、サービスを切れ目なく利用できます。詳細については、購入をキューに入れる を参照してください。

AWS の無料利用枠は、新規の AWS アカウントで使用できます。AWS の無料利用枠を使用して Amazon EC2 インスタンスを実行している場合、購入したリザーブドインスタンスは標準の料金ガイドラインに基づいて課金されます。詳細については、「AWS 無料利用枠」を参照してください。

コンテンツ

使用料の請求

リザーブドインスタンス は、選択した期間内の 1 時間ごとに請求されます。インスタンスが実行中であるかどうかは関係しません。各時間は、標準の 24 時間の時計の正時 (毎時ゼロ分ゼロ秒) に開始します。例えば、1:00:00 ~ 1:59:59 が 1 時間です。インスタンスステータスの詳細については、「インスタンスのライフサイクル」を参照してください。

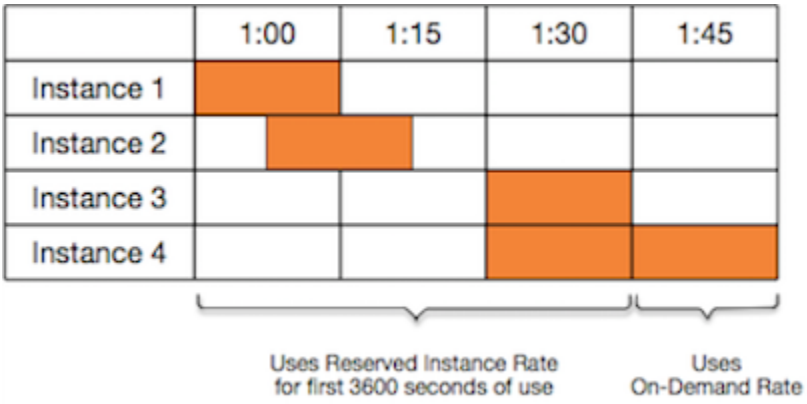
リザーブドインスタンス の料金上の特典は秒単位課金で実行中のインスタンスに適用されます。秒単位の請求は、オープンソースの Linux ディストリビューション (Amazon Linux、Ubuntu など) を使用するインスタンスで使用できます。時間単位の請求は、Linux の商用ディストリビューション (Red Hat Enterprise Linux、SUSE Linux Enterprise Server など) で使用できます。

リザーブドインスタンス の料金上の利点は、1 時間当たり最大 3600 秒 (1 時間) のインスタンス使用に適用できます。複数のインスタンスを同時に実行できますが、リザーブドインスタンス 割引の特典を受けることができるのは 1 時間あたり合計 3600 秒までです。インスタンスの使用が 1 時間あたり 3600 秒を超えると、オンデマンドレートで課金されます。

例えば、1 つの `m4.xlarge` リザーブドインスタンス を購入し、4 つの `m4.xlarge` インスタンスを 1 時間同時に実行する場合、1 つのインスタンスには リザーブドインスタンス の 1 時間分の使用料が、他の 3 つのインスタンスにはオンデマンドの 3 時間分の使用料が課金されます。

一方、1 つの `m4.xlarge` リザーブドインスタンス を購入し、4 つの `m4.xlarge` インスタンスを同じ 1 時間内にそれぞれ 15 分 (900 秒) ずつ実行した場合、インスタンスの合計実行時間は 1 時間となり、リザーブドインスタンス の使用料が 1 時間分課金されるだけで、オンデマンドの使用料は課金されません。

複数の対象インスタンスが同時に実行されている場合、リザーブドインスタンスの課金特典は、1 時間に最大 3600 秒まで、すべてのインスタンスに同時に適用され、その後、オンデマンド料金が適用されます。



Billing and Cost Management コンソール上の [Cost Explorer] は、オンデマンドインスタンス の実行に対する節約を分析することができます。リザーブドインスタンス についてのよくある質問には、表示価格の計算例があります。

AWS アカウントを解約すると、リソースのオンデマンド課金は停止します。ただし、アカウントにリザーブドインスタンス がある場合には、その期限が切れるまで続けて課金されます。

請求の表示

AWS Billing and Cost Management コンソールで、アカウントへの請求および料金を確認できます。

- [ダッシュボード] には、アカウント利用料の概要が表示されます。
- [請求書] ページの [明細] で、[Elastic Compute Cloud] セクションと リザーブドインスタンス の請求情報を取得するリージョンを展開します。

請求額をオンラインで表示することも、CSV ファイルとしてダウンロードすることもできます。

また、AWS のコストと使用状況レポートを使用して、リザーブドインスタンスの使用を追跡することも可能です。詳細については、AWS Billing ユーザーガイドで、コストと使用状況レポートの「リザーブドインスタンス」を参照してください。

リザーブドインスタンス および一括請求 (コンソリデーティッドビリング)

購入アカウントが、1 つの一括請求の支払いアカウントに請求される一連のアカウントの一部である場合、リザーブドインスタンス の料金面でのメリットを広範囲に利用できます。すべてのメンバーアカウントのインスタンス使用量が月次で支払人アカウントに集約されます。さまざまな役割を持つチームやグループがある企業にとっては特に便利です。したがって、請求書の計算には通常の リ

ザーブドインスタンス のロジックが適用されます。詳細については、「AWS Organizations の一括請求」を参照してください。

リザーブドインスタンス を購入したアカウントを解約した場合、そのリザーブドインスタンス が期限切れになるまで、支払いアカウントに対する請求が継続されます。アカウントは解約から 90 日後に完全に削除され、メンバーアカウントにはリザーブドインスタンスによる割引料金が適用されなくなります。

> **Note**
>
> ゾーンリザーブドインスタンスは所有アカウントにのみ容量を予約し、他の AWS アカウント と共有することはできません。他の AWS アカウント と容量を共有する必要がある場合は、On-Demand Capacity Reservations を使用してください。

リザーブドインスタンス 割引料金範囲

割引料金範囲が適用されると、そのアカウントは、以降、その範囲レベル内で行われる リザーブドインスタンス 購入の前払い料金およびインスタンス使用料に対して、自動的に割引を受けます。割引を受けるためには、リージョンの リザーブドインスタンス の表示価格が 500,000 USD 以上である必要があります。

以下のルールが適用されます。

- 料金範囲およびそれに関連する割引は、Amazon EC2 スタンダード リザーブドインスタンス の購入にのみ適用されます。
- 料金範囲は、SQL Server Standard、SQL Server Web、および SQL Server Enterprise を使用する Windows 用の リザーブドインスタンス には適用されません。
- 料金範囲は、SQL Server Standard、SQL Server Web、および SQL Server Enterprise を使用する Linux 用の リザーブドインスタンス には適用されません。
- 料金範囲の割引は、AWS での購入にのみ適用されます。これは、サードパーティーの リザーブドインスタンス の購入には適用されません。
- 料金範囲割引は現在、コンバーティブルリザーブドインスタンス の購入には適用されません。

トピック

- リザーブドインスタンス 料金割引の計算
- 割引範囲での購入

- 購入料金範囲
- 料金範囲の一括請求

リザーブドインスタンス 料金割引の計算

リージョンのすべての リザーブドインスタンス の合計表示価格を計算することによって、アカウントの料金範囲を決定できます。各予約の時間当たりの定期料金を期間の合計時間数に掛けて、購入時の割引されていない前払い料金 (固定料金とも呼ばれる) を加算します。表示価格は割引前料金 (一般料金) に基づいているため、従量制割引の適用を受けた場合または リザーブドインスタンス を購入した後の値下げ分は反映されません。

```
List value = fixed price + (undiscounted recurring hourly price * hours in term)
```

例えば、1 年間の一部前払い `t2.small` リザーブドインスタンス の場合、前払い料金は 60.00 ドル、時間料金は 0.007 ドルと仮定します。これにより、表示価格は 121.32 ドルとなります。

```
121.32 = 60.00 + (0.007 * 8760)
```

New console

Amazon EC2 コンソールを使用して リザーブドインスタンス の固定料金を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. [前払い料金] 列を表示するには、右上にある設定 (⚙) をクリックし、[前払い料金] をオンにして [確認] をクリックします。

Old console

Amazon EC2 コンソールを使用して リザーブドインスタンス の固定料金を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. [前払い料金] 列を表示するには、右上にある設定 (⚙) をクリックし、[前払い料金] をオンにして [閉じる] をクリックします。

リザーブドインスタンス の固定料金をコマンドラインを使用して表示するには

- describe-reserved-instances (AWS CLI)
- Get-EC2ReservedInstance (AWS Tools for Windows PowerShell)
- DescribeReservedInstances (Amazon EC2 API)

割引範囲での購入

リザーブドインスタンス を購入すると、割引料金範囲に該当する部分のリザーブドインスタンスに対応する割引があれば、Amazon EC2 によって自動的に適用されます。特に何かを行う必要はなく、どの Amazon EC2 ツールを使用しても リザーブドインスタンス を購入できます。詳細については、リザーブドインスタンスの購入 を参照してください。

リージョンでのアクティブな リザーブドインスタンス の表示価格が割引料金範囲に該当した場合、そのリージョンでの リザーブドインスタンス は、以降、すべての購入が割引料金で課金されます。リージョンの リザーブドインスタンス の 1 回の購入でしきい値を超える場合は、そのご購入分のうち、割引範囲のしきい値を超える部分が割引になります。購入プロセス中に作成された一時的な リザーブドインスタンス ID の詳細については「購入料金範囲」を参照してください。

表示価格が割引範囲の金額を下回った場合は (一部の リザーブドインスタンス の有効期限が切れた場合など)、以降、そのリージョンで購入される リザーブドインスタンス には割引が適用されません。ただし、もともと割引料金範囲で購入された リザーブドインスタンス に対しては、引き続き割引が適用されます。

リザーブドインスタンス を購入すると、次の 4 つのいずれかの状況になります。

- 割引なし — 1 つのリージョンでのリザーブドインスタンスの購入が、まだ割引しきい値より下である。
- 一部割引 — 1 つのリージョンでのリザーブドインスタンスの購入により、最初の割引範囲のしきい値を超える。割引なしが 1 つ以上の予約に適用され、割引料金が残りの予約に適用されます。
- 完全割引 — リージョン内の購入全体が 1 つの割引範囲に完全に含まれ、適切に割引されます。
- 2 つの割引料金 — 1 つのリージョンでのリザーブドインスタンスの購入が、下の割引範囲から上の割引範囲まで及ぶ。2 つの異なる料金が課金されます。1 つ以上の予約により低い割引率が適用され、残りの予約により高い割引率が適用されます。

購入料金範囲

割引料金範囲に到達した場合、その購入に対して複数のエントリが表示されます。通常の料金が課金される部分と、割引料金が適用されて課金される部分です。

リザーブドインスタンス サービスによって複数の リザーブドインスタンス ID が生成されます。これは、割引が適用されない範囲と割引範囲、または複数の割引範囲に購入がまたがるためです。範囲内の予約のセットにはそれぞれ ID があります。この結果、購入 CLI コマンドまたは API アクションによって返される ID は、新しい リザーブドインスタンス の実際の ID とは異なるものになります。

料金範囲の一括請求

一括請求アカウントはリージョン内のメンバーアカウントの表示価格を集計します。一括請求アカウントのすべてのアクティブな リザーブドインスタンス の表示価格が割引料金範囲に達すると、以降 (その一括請求アカウントの表示価格が割引価格範囲のしきい値を超えている限り)、一括請求アカウント内のアカウントで購入された リザーブドインスタンス には割引が適用されます。詳細については、リザーブドインスタンス および一括請求 (コンソリデーティッドビリング) を参照してください。

リザーブドインスタンスの購入

リザーブドインスタンス を購入するには、AWS とサードパーティー販売者からの リザーブドインスタンス 製品を、検索パラメータを調整しながら希望するものと完全に一致するものが見つかるまで検索します。

購入する リザーブドインスタンス を検索する際、提供タイプの費用の見積もりが表示されます。購入手続きに進むと、AWS は自動的に購入価格に上限価格を指定します。リザーブドインスタンス の合計コストが、指定した金額を超えることはありません。

何らかの理由により料金が上がったり変更された場合、購入は完了しません。購入時に、選択した内容と同じような内容で価格が安い製品があった場合、AWS はその製品をその安い価格で販売します。

購入を承認する前に、購入を検討している リザーブドインスタンス の詳細を点検して、すべてのパラメータが正しいことを確認してください。リザーブドインスタンス を購入した後は、 (販売者が Reserved Instance Marketplace のサードパーティーであっても、AWS であったとしても) その購入をキャンセルすることはできません。

> **Note**
>
> 購入後のリザーブドインスタンスを修正するには、ユーザーに、アベイラビリティーゾーンを記述する機能などの適切な許可が付与されていることを確認します。詳細については、「AWS CLI または AWS SDK で機能するサンプルポリシー」および「Amazon EC2 コンソールで機能するサンプルポリシー」を参照してください。

トピック

プラットフォームの選択

Amazon EC2 は、次の Linux プラットフォームを リザーブドインスタンス でサポートしています。

- Linux/UNIX
- Linux with SQL Server Standard
- Linux with SQL Server Web
- Linux with SQL Server Enterprise
- SUSE Linux
- Red Hat Enterprise Linux
- Red Hat Enterprise Linux with HA

リザーブドインスタンス を購入する際、インスタンスのオペレーティングシステムを表すプラットフォームに対するサービスを選択する必要があります。

- SUSE Linux および RHEL ディストリビューションでは、これらの特定のプラットフォーム (SUSE Linux または Red Hat Enterprise Linux プラットフォーム) 用のサービスを選択する必要があります。
- その他のすべての Linux ディストリビューション (Ubuntu を含む) の場合は、Linux/UNIX プラットフォームに対するサービスを選択します。
- 既存の RHEL サブスクリプションを持ち込む場合は、Red Hat Enterprise Linux プラットフォーム用のサービスではなく、Linux/UNIX プラットフォーム用のサービスを選択する必要があります。

> **Note**
>
> Ubuntu Pro はリザーブドインスタンスとしてはご利用いただけません。オンデマンドインスタンスの価格と比較して大幅に節約するには、Ubuntu Pro と Savings Plans の併用をお勧めします。詳細については、 Savings Plans ユーザーガイドを参照してください。

> **Important**
>
> AWS Marketplace AMI から作成されたオンデマンドインスタンスに適用するための、リザーブドインスタンスを購入する予定がある場合は、まず AMI の `PlatformDetails` フィールドを確認します。`PlatformDetails` フィールドには、どのリザーブドインスタンスを購入するかが示されます。AMI のプラットフォームの詳細とリザーブドインスタンスのプラットフォームが一致している必要があります。一致していない場合、リザーブドインスタンスはオンデマンドインスタンスに適用されません。AMI のプラットフォームの詳細を表示する方法については、「AMI の請求情報について」を参照してください。

Windows でサポートされているプラットフォームについては、『Windows インスタンスの Amazon EC2 ユーザーガイド』の「プラットフォームの選択」を参照してください。

購入をキューに入れる

デフォルトでは、リザーブドインスタンス を購入すると、すぐに購入が行われます。別の方法として、将来の日時の購入予約をキューに入れることができます。例えば、既存の リザーブドインスタンス が期限切れになる頃の購入予約をキューに入れることができます。これにより、サービスを切れ目なく利用できます。

リザーブドインスタンス の購入予約をキューに入れる場合、リージョンは指定できますが、ゾーンを指定した リザーブドインスタンス の購入予約や、他の販売者からの リザーブドインスタンス の購入予約を行うことはできません。購入予約は 3 年先までキューに入れることができます。予約した日時に、デフォルトの支払い方法を使用して購入が実行されます。支払いが正常に行われると、支払い特典が適用されます。

キューに入れた購入予約は Amazon EC2 コンソールで確認できます。キューに入れた購入予約のステータスは [queued] になります。キューに入れた購入予約は、予約日の前にいつでもキャンセルできます。詳細については、「キューに入れた購入のキャンセル」を参照してください。

スタンダード リザーブドインスタンス の購入

スタンダード リザーブドインスタンス を特定のアベイラビリティーゾーンで購入し、キャパシティーの予約ができます。または、キャパシティの予約を見送り、リージョンのスタンダード リザーブドインスタンス を購入することもできます。

New console

コンソールを使用してスタンダードリザーブドインスタンスを購入するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [リザーブドインスタンス] を選択し、[リザーブドインスタンス の購入] を選択します。
3. [提供クラス] で [標準] を選択し、標準 リザーブドインスタンス を表示します。
4. キャパシティーの予約を購入するには、購入画面の右上で、 [キャパシティ予約のある提供タイプのみ表示] をオンにします。この設定をオンにすると、[アベイラビリティーゾーン] フィールドが表示されます。

   リージョンの リザーブドインスタンス を購入するには、この設定をオフにします。この設定をオフにすると、[アベイラビリティーゾーン] フィールドが消えます。
5. 必要に応じて他の設定を選択してから、[検索] をクリックします。
6. 購入する リザーブドインスタンス ごとに、希望する数量を入力して、[カートに入れる] を選択します。

   Reserved Instance Marketplace からスタンダードリザーブドインスタンスを購入するには、検索結果の [販売者] 列から、[サードパーティ] を見つけます。[期間] 列には標準以外の期間が表示されます。詳細については、Reserved Instance Marketplace からの購入 を参照してください。

7. 選択した リザーブドインスタンス の概要を確認するには、[カートを見る] を選択します。
8. [注文日] が [Now (今すぐ注文)] になっている場合は、[すべて注文] をクリックすれば即時購入が完了します。購入予約をキューに入れるには、[Now] を選択して日付を選択します。カート内の有効なサービスごとに別の日付を選択できます。購入は、選択した日付の 00:00 UTC までキューに入れられます。
9. 注文を確定するには、[すべて注文] をクリックします。

   注文時に、選択したインスタンスと同等でより安価なインスタンスがある場合、AWS はより安価なインスタンスを販売します。
10. [閉じる] を選択します。

    注文のステータスは [State] 列に表示されます。注文が確定されると、[State] の値が [Payment-pending] から [Active] に変わります。リザーブドインスタンス が Active となっていれば、使用準備が完了しています。

> **Note**
>
> ステータスが Retired になると、AWS は支払いを受領していない場合があります。

Old console

コンソールを使用してスタンダードリザーブドインスタンス を購入するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [リザーブドインスタンス] を選択し、[リザーブドインスタンス の購入] を選択します。
3. [提供クラス] で [標準] を選択し、標準 リザーブドインスタンス を表示します。
4. キャパシティーの予約を購入するには、購入画面の右上で [Only show offerings that reserve capacity] を選択します。リージョン リザーブドインスタンス を購入するには、チェックボックスを選択しないままにします。
5. 必要に応じて他の設定を選択してから、[Search] を選択します。

   Reserved Instance Marketplace からスタンダードリザーブドインスタンスを購入するには、検索結果の [販売者] 列から [サードパーティ] を見つけます。[期間] 列には標準以外の期間が表示されます。

6. 購入する リザーブドインスタンス ごとに、数量を入力して、[カートに入れる] を選択します。
7. 選択した リザーブドインスタンス の概要を確認するには、[カートを見る] を選択します。
8. [Order On (注文日)] が [Now] の場合は、購入が即座に実行されます。購入予約をキューに入れるには、[Now] を選択して日付を選択します。カート内の有効なサービスごとに別の日付を選択できます。購入は、選択した日付の 00:00 UTC までキューに入れられます。
9. 注文を確定するには、[Order (注文)] を選択します。

   注文時に、選択したインスタンスと同等でより安価なインスタンスがある場合、AWS はより安価なインスタンスを販売します。
10. [閉じる] を選択します。

    注文のステータスは [State] 列に表示されます。注文が確定されると、[State] の値が [payment-pending] から [active] に変わります。リザーブドインスタンス が `active` となっていれば、使用準備が完了しています。

> **Note**
>
> ステータスが `retired` になると、AWS は支払いを受領していない場合があります。

AWS CLI を使用してスタンダードリザーブドインスタンス を購入するには

1. describe-reserved-instances-offerings コマンドを使用して、利用できる リザーブドインスタンス を見つけます。スタンダード リザーブドインスタンス のみを返すには、`standard` を `--offering-class` パラメータに指定します。追加のパラメータを適用して、結果を絞り込むことができます。例えば、`t2.large` のデフォルトテナンシーのリージョナル `Linux/UNIX` リザーブドインスタンス を 1 年間の期間だけで購入するには:

```
aws ec2 describe-reserved-instances-offerings \
    --instance-type t2.large \
    --offering-class standard \
    --product-description "Linux/UNIX" \
    --instance-tenancy default \
    --filters Name=duration,Values=31536000 Name=scope,Values=Region
```

Reserved Instance Marketplace からのみの リザーブドインスタンス を探すには、marketplace フィルターを使用します。期間が 1 年間 あるいは 3 年間より短い場合があるため、リクエストに期間は指定しません。

```
aws ec2 describe-reserved-instances-offerings \
    --instance-type t2.large \
    --offering-class standard \
    --product-description "Linux/UNIX" \
    --instance-tenancy default \
    --filters Name=marketplace,Values=true
```

ニーズに合う リザーブドインスタンス が見つかったら、提供 ID を書き留めます。次に例を示します。

```
"ReservedInstancesOfferingId": "bec624df-a8cc-4aad-a72f-4f8abc34caf2"
```

2. purchase-reserved-instances-offering コマンドを使用して、リザーブドインスタンス を購入します。前のステップで取得した リザーブドインスタンス 提供 ID を指定し、予約するインスタンスの数を指定する必要があります。

```
aws ec2 purchase-reserved-instances-offering \
    --reserved-instances-offering-id bec624df-a8cc-4aad-a72f-4f8abc34caf2 \
    --instance-count 1
```

デフォルトでは、購入は即座に実行されます。別の方法として、購入予約をキューに入れるには、次のパラメータを前の呼び出しに追加します。

```
--purchase-time "2020-12-01T00:00:00Z"
```

3. describe-reserved-instances コマンドを使用して、購入したリザーブドインスタンスのステータスを確認します。

```
aws ec2 describe-reserved-instances
```

または、以下の AWS Tools for Windows PowerShell コマンドを使用します。

- Get-EC2ReservedInstancesOffering

- New-EC2ReservedInstance
- Get-EC2ReservedInstance

購入の完了後、リザーブドインスタンス の仕様と一致するインスタンスをすでに実行している場合は、支払い特典が即座に適用されます。インスタンスを再起動する必要はありません。適切な実行中のインスタンスが存在しない場合、インスタンスを起動して、リザーブドインスタンス に指定した条件と一致していることを確認します。詳細については、お使いの リザーブドインスタンス の使用を参照してください。

実行しているインスタンスにどのように リザーブドインスタンス が適用されるかについての例は、「リザーブドインスタンス がどのように適用されるか」を参照します。

コンバーティブルリザーブドインスタンス の購入

コンバーティブルリザーブドインスタンス を特定のアベイラビリティーゾーンで購入し、キャパシティーの予約ができます。または、キャパシティの予約を見送り、リージョン コンバーティブルリザーブドインスタンス を購入することもできます。

New console

コンソールを使用して コンバーティブルリザーブドインスタンス を購入するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [リザーブドインスタンス] を選択し、[リザーブドインスタンス の購入] を選択します。
3. [提供クラス] で [コンバーティブル] を選択し、コンバーティブルリザーブドインスタンス を表示します。
4. キャパシティーの予約を購入するには、購入画面の右上で、 [キャパシティ予約のある提供タイプのみ表示] をオンにします。この設定をオンにすると、[アベイラビリティーゾーン] フィールドが表示されます。

   リージョンの リザーブドインスタンス を購入するには、この設定をオフにします。この設定をオフにすると、[アベイラビリティーゾーン] フィールドが消えます。
5. 必要に応じて他の設定を選択してから、[Search] を選択します。
6. 購入する コンバーティブルリザーブドインスタンス ごとに数量を入力して、[カートに入れる] を選択します。
7. 選択内容の概要を表示するには、[カートを見る] を選択します。

8. [注文日] が [Now (今すぐ注文)] になっている場合は、[すべて注文] をクリックすれば即時購入が完了します。購入予約をキューに入れるには、[Now] を選択して日付を選択します。カート内の有効なサービスごとに別の日付を選択できます。購入は、選択した日付の 00:00 UTC までキューに入れられます。
9. 注文を確定するには、[すべて注文] をクリックします。

   注文時に、選択したインスタンスと同等でより安価なインスタンスがある場合、AWS はより安価なインスタンスを販売します。
10. [閉じる] を選択します。

    注文のステータスは [State] 列に表示されます。注文が確定されると、[State] の値が [Payment-pending] から [Active] に変わります。リザーブドインスタンス が Active となっていれば、使用準備が完了しています。

> **Note**
>
> ステータスが Retired になると、AWS は支払いを受領していない場合があります。

Old console

コンソールを使用して コンバーティブルリザーブドインスタンス を購入するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [リザーブドインスタンス] を選択し、[リザーブドインスタンス の購入] を選択します。
3. [提供クラス] で [コンバーティブル] を選択し、コンバーティブルリザーブドインスタンス を表示します。
4. キャパシティーの予約を購入するには、購入画面の右上で [Only show offerings that reserve capacity] を選択します。リージョン リザーブドインスタンス を購入するには、チェックボックスを選択しないままにします。
5. 必要に応じて他の設定を選択してから、[Search] を選択します。
6. 購入する コンバーティブルリザーブドインスタンス ごとに、数量を入力して、[カートに入れる] を選択します。
7. 選択内容の概要を表示するには、[カートを見る] を選択します。

8. [Order On (注文日)] が [Now] の場合は、購入が即座に実行されます。購入予約をキューに入れるには、[Now] を選択して日付を選択します。カート内の有効なサービスごとに別の日付を選択できます。購入は、選択した日付の 00:00 UTC までキューに入れられます。
9. 注文を確定するには、[Order (注文)] を選択します。

   注文時に、選択したインスタンスと同等でより安価なインスタンスがある場合、AWS はより安価なインスタンスを販売します。
10. [閉じる] を選択します。

    注文のステータスは [State] 列に表示されます。注文が確定されると、[State] の値が [payment-pending] から [active] に変わります。リザーブドインスタンス が `active` となっていれば、使用準備が完了しています。

> **Note**
>
> ステータスが `retired` になると、AWS は支払いを受領していない場合があります。

AWS CLI を使用して コンバーティブルリザーブドインスタンス を購入するには

1. describe-reserved-instances-offerings コマンドを使用して、利用できる リザーブドインスタンス を見つけます。コンバーティブルリザーブドインスタンス だけを返すには、`convertible` を `--offering-class` パラメータに指定します。追加のパラメータを適用して結果を絞り込むことができます。例えば、`t2.large` のデフォルトテナンシーのリージョナル `Linux/UNIX` リザーブドインスタンス を購入するには:

   ```
   aws ec2 describe-reserved-instances-offerings \
       --instance-type t2.large \
       --offering-class convertible \
       --product-description "Linux/UNIX" \
       --instance-tenancy default \
       --filters Name=scope,Values=Region
   ```

   ニーズに合う リザーブドインスタンス が見つかったら、提供 ID を書き留めます。次に例を示します。

   ```
   "ReservedInstancesOfferingId": "bec624df-a8cc-4aad-a72f-4f8abc34caf2"
   ```

2. purchase-reserved-instances-offering コマンドを使用して、リザーブドインスタンス を購入します。前のステップで取得した リザーブドインスタンス 提供 ID を指定し、予約するインスタンスの数を指定する必要があります。

```
aws ec2 purchase-reserved-instances-offering \
    --reserved-instances-offering-id bec624df-a8cc-4aad-a72f-4f8abc34caf2 \
    --instance-count 1
```

デフォルトでは、購入は即座に実行されます。別の方法として、購入予約をキューに入れるには、次のパラメータを前の呼び出しに追加します。

```
--purchase-time "2020-12-01T00:00:00Z"
```

3. describe-reserved-instances コマンドを使用して、購入したリザーブドインスタンスのステータスを確認します。

```
aws ec2 describe-reserved-instances
```

または、以下の AWS Tools for Windows PowerShell コマンドを使用します。

- Get-EC2ReservedInstancesOffering
- New-EC2ReservedInstance
- Get-EC2ReservedInstance

リザーブドインスタンス の仕様と一致するインスタンスをすでに実行している場合、料金上の利点は即時適用されます。インスタンスを再起動する必要はありません。適切な実行中のインスタンスが存在しない場合、インスタンスを起動して、リザーブドインスタンス に指定した条件と一致していることを確認します。詳細については、お使いの リザーブドインスタンス の使用 を参照してください。

実行しているインスタンスにどのように リザーブドインスタンス が適用されるかについての例は、「リザーブドインスタンス がどのように適用されるか」を参照します。

Reserved Instance Marketplace からの購入

Reserved Instance Marketplace から、サードパーティーの販売者が所有しており、必要がなくなったリザーブドインスタンスを購入できます。これを行うには、Amazon EC2 コンソールまたはコマ

ンドラインツールを使用します。このプロセスは、AWS から リザーブドインスタンス を購入する場合と似ています。詳細については、スタンダード リザーブドインスタンス の購入 を参照してください。

Reserved Instance Marketplace で購入したリザーブドインスタンスと、AWS から直接購入したリザーブドインスタンスとの間には、以下のような違いがあります。

- 期間 – サードパーティー販売者から購入した リザーブドインスタンス は、残り期間が完全な標準期間よりも短くなります。AWS の完全な標準期間は 1 年または 3 年間です。
- 前払い料金 – サードパーティーの リザーブドインスタンス は、さまざまな前払い料金で販売されます。使用に対して、または定期的に支払う料金は、リザーブドインスタンスを最初に AWS から購入した際に設定された料金と同じ金額です。
- リザーブドインスタンスのタイプ – Reserved Instance Marketplace から購入できるのは、Amazon EC2 スタンダードリザーブドインスタンスのみです。コンバーティブルリザーブドインスタンス、Amazon RDS、および Amazon ElastiCache のリザーブドインスタンスは、Reserved Instance Marketplace では購入できません。

お客様に関する基本情報 (郵便番号や国情報など) は、販売者と共有されます。

この情報を使用して、販売者は、国に支払う必要な取引税 (売上税や付加価値税など) を計算し、支払いレポートとして提示します。まれに、AWS が販売者に E メールアドレスを提供する必要がある場合があります。これは、販売者が、販売に関する質問があり、それに関して連絡できるようにするためです (例えば税務上の質問など)。

同様の理由で、AWS は購入者の請求書に販売者の正式名を記載します。税金または関連する理由で販売者の情報が必要な場合は、AWS Support までお問い合わせください。

リザーブドインスタンス の表示

Amazon EC2 コンソールあるいはコマンドラインツールを使用して、購入した リザーブドインスタンス を表示できます。

リザーブドインスタンス をコンソールで表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. キューに登録された、アクティブな、およびリタイア済みの リザーブドインスタンス が一覧表示されます。[状態] 列には状態が表示されます。

4. Reserved Instance Marketplace 内の販売者に対しては、Reserved Instance Marketplaceに一覧表示されている予約の状態が、[出品] タブに表示されます。詳細については、リザーブドインスタンス の出品状態 を参照してください。

コマンドラインを使用して リザーブドインスタンス を表示するには

- describe-reserved-instances (AWS CLI)
- Get-EC2ReservedInstance (Tools for Windows PowerShell)

キューに入れた購入のキャンセル

購入予約は 3 年先までキューに入れることができます。キューに入れた購入予約は、予約日の前にいつでもキャンセルできます。

New console

キューに入れた購入をキャンセルするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. 1 つまたは複数の リザーブドインスタンス を選択します。
4. [アクション]、[キュー入りリザーブドインスタンスの削除] の順にクリックします。
5. 確認を求められたら、[削除] をクリックし、次に [閉じる] をクリックします。

Old console

キューに入れた購入をキャンセルするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. 1 つまたは複数の リザーブドインスタンス を選択します。
4. [アクション]、[キュー入りリザーブドインスタンスの削除] の順に選択します。
5. 確認を求めるメッセージが表示されたら、[Yes, Delete] を選択します。

コマンドラインを使用してキューに入れた購入予約をキャンセルするには

- delete-queued-reserved-instances (AWS CLI)
- Remove-EC2QueuedReservedInstance (Tools for Windows PowerShell)

リザーブドインスタンス の更新

リザーブドインスタンス は有効期限が切れる前に更新できます。リザーブドインスタンス を更新すると、現在の リザーブドインスタンス が期限切れになるまで、同じ設定の リザーブドインスタンス の購入予約がキューに入れられます。

New console

キューに入れた購入予約を使用してリザーブドインスタンスを更新するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. 更新するリザーブドインスタンスを選択します。
4. [Actions (アクション)]、[Renew Reserved Instances (リザーブドインスタンスの更新)] の順に選択します。
5. 注文を確定するには、[すべて注文]、[閉じる] の順にクリックします。

Old console

キューに入れた購入予約を使用してリザーブドインスタンスを更新するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. 更新するリザーブドインスタンスを選択します。
4. [Actions (アクション)]、[Renew Reserved Instances (リザーブドインスタンスの更新)] の順に選択します。
5. 注文を確定するには、[Order (注文)] を選択します。

Reserved Instance Marketplace での販売

Reserved Instance Marketplace は、サードパーティーや AWS のお客様が所有していながら使用していないスタンダードリザーブドインスタンスを、さまざまな期間と料金のオプションで販売できるようにするプラットフォームです。例えば、新しい AWS リージョンにインスタンスを移動した後や、新しいインスタンスタイプに変更した場合、また、期間が終了する前にプロジェクトが終了した場合や、ビジネスのニーズが変化した場合、そして不要なキャパシティーがある場合などに、 リザーブドインスタンスを販売することが考えられます。

リザーブドインスタンスは、Reserved Instance Marketplace への出品直後から、購入希望者に表示されます。すべての リザーブドインスタンス は、残り期間や時間料金別にグループ化されます。

AWS は購入者の希望を満たすため、指定されたグループで最も前払い料金が低いリザーブドインスタンスを最初に販売します。次に AWS は、購入者の注文全体が満たされるまで、料金が低い順にリザーブドインスタンスを販売します。AWS はその後、トランザクションを処理しリザーブドインスタンスの所有権を購入者に移転します。

出品した リザーブドインスタンス は、売れるまでお客様の所有です。販売後は、予約済みのキャパシティーと割引使用料金は使用できません。インスタンスの使用が継続している間、AWS はリザーブドインスタンスが販売された時間から起算したオンデマンド料金を課金します。

Reserved Instance Marketplace で、使用していないリザーブドインスタンスを販売するには、特定の要件基準を満たしている必要があります。

Reserved Instance Marketplace でのリザーブドインスタンスの購入の詳細については、「Reserved Instance Marketplace からの購入」を参照してください。

コンテンツ

- 支払いを受け取る
- 購入者と共有する情報

制約と制限

使用していないリザーブドインスタンスを販売できるようになる前に、Reserved Instance Marketplace に、自分を販売者として登録する必要があります。詳細については、販売者として登録する を参照してください。

リザーブドインスタンス の販売時に次の制約と制限が適用されます:

- Reserved Instance Marketplace で販売可能なのは、Amazon EC2 スタンダードのリージョンとゾーンのリザーブドインスタンスのみです。
- Amazon EC2 コンバーティブルリザーブドインスタンスは、Reserved Instance Marketplace では販売できません。
- 他の AWS サービス (Amazon RDS や Amazon ElastiCache など) 向けのリザーブドインスタンスは、Reserved Instance Marketplace では販売できません。
- スタンダード リザーブドインスタンス の有効期間が 1 か月以上残っている必要があります。
- デフォルトで無効になっているリージョンでは、スタンダード リザーブドインスタンス は販売できません。
- Reserved Instance Marketplace で許容される最低販売価格は、0.00 USD です。
- 前払いなし、一部前払い、または全額前払いのリザーブドインスタンスは、アカウント内で 30 日間以上アクティブになっている限り、リザーブドインスタンス Marketplace で販売できます。さらに、リザーブドインスタンスに前払いがある場合は、AWS が前払い料金を受領した後にのみ販売できます。
- Reserved Instance Marketplace に表示された出品内容は、直接変更することはできません。ただし、最初に出品をキャンセルしてから、新しいパラメータで別の出品を作成することはできます。詳細については、リザーブドインスタンス 料金設定 を参照してください。出品する前に リザーブドインスタンス を変更することもできます。詳細については、リザーブドインスタンス の変更 を参照してください。
- AWS は、リザーブドインスタンスマーケットプレイスで販売するスタンダードリザーブドインスタンスごとに、前払料金の総額の 12% をサービス料として課金します。前払い価格は、販売者がスタンダード リザーブドインスタンス に課金する価格です。
- 販売者として登録する場合、指定する銀行には米国の住所が必要です。詳細については、AWS Marketplace 販売者ガイドの「有料製品の販売者の追加要件」を参照してください。

- Amazon Internet Services Private Limited (AISPL) のお客様は、米国の銀行口座をお持ちの場合でも、Reserved Instance Marketplace でリザーブドインスタンスを販売することはできません。詳細については、AWS アカウントと AISPL アカウントの違いは何ですか?をご参照ください。

販売者として登録する

> **Note**
>
> アカウントを販売者として登録できるのは、AWS アカウントのルートユーザー のみです。

Reserved Instance Marketplace で販売するには、最初に自分を販売者として登録する必要がありますます。登録時には以下の情報を指定します。

- 銀行情報 – AWS は、予約の販売時に集金された金額をお支払いするために、お客様の銀行情報を必要とします。住所が米国内の銀行でなければなりません。詳細については、支払い用の銀行口座を参照してください。
- 税金情報 — 必要な税金報告義務を判断するために、販売者は必ず、税金情報の質問に回答する必要があります。詳細については、税金情報 を参照してください。

記入された販売者登録が AWS に受領されると、登録を確認する電子メールが送信され、Reserved Instance Marketplace での販売を開始できることが伝えられます。

支払い用の銀行口座

AWS は、リザーブドインスタンスの販売時に集金された金額をお支払いするために、お客様の銀行情報を必要とします。住所が米国内の銀行でなければなりません。詳細については、AWS Marketplace 販売者ガイドの「有料製品の販売者の追加要件」を参照してください。

支払い用のデフォルトの銀行口座を登録するには

1. [Reserved Instance Marketplace 販売者登録] ページを開き、AWS の認証情報を使用してサインインします。
2. [Manage Bank Account] ページで、支払いを受け取る銀行に関する以下の情報を提供します。
   - 銀行口座の名義
   - 支店コード
   - アカウント番号

- 銀行口座の種類

> **Note**
>
> 法人の銀行口座を使用する場合は、口座に関する情報を FAX (1-206-765-3424) で送信するように指示されます。

登録後、指定された銀行口座がデフォルトとして設定され、銀行の確認待ちとなります。新しい銀行口座を確認するには、最長で 2 週間かかります。この間は支払金を受け取ることができません。確立済みの口座の場合、支払い完了まで通常およそ 2 日かかります。

支払い用のデフォルトの銀行口座を変更するには

1. [Reserved Instance Marketplace 販売者登録] ページで、登録時に使用したアカウントでサインインします。
2. [Manage Bank Account] ページで、必要に応じて新規口座のアカウントを追加するか、デフォルトの銀行口座を変更します。

税金情報

リザーブドインスタンス の販売には、取引関連の税金 (消費税または付加価値税など) がかかることがあります。取引関連の税金が適用されるかどうかについては、税務部、法務部、財務部、または経理部に確認する必要があります。お客様は、取引関連の税金を収集し、該当する税務署に納める役割を担います。

販売者登録手続きの一環として、「販売者登録ポータル」の Tax Interview を完了させる必要があります。このインタビューでは、税金情報を収集し、IRS フォーム W-9、W-8BEN、または W-8BEN-E に入力します。このフォームは、必要な税金報告義務を明確にするために使用されます。

Tax Interview の一環として入力する税金情報は、個人または法人であるか、また、米国人または非米国人 (米国企業または非米国企業) かによって異なる場合があります。Tax interview の記入を行う際は、次に注意してください。

- AWS が提供する情報 (このトピックの情報を含む) は、税金、法律、またはその他の専門的なアドバイスではありません。IRS のレポート要件がビジネスに及ぼす影響について知りたい場合、またはご質問がある場合は、税金、法律、またはその他の専門家にお問い合わせください。

- IRS のレポート要件をできるだけ効率的に満たすには、Tax interview の中で要求されたすべての質問に答え、情報を入力します。
- 答えを確認します。綴りを間違ったり、誤った税金識別番号を入力したりしないようにします。これらのミスがあると、誤った税金フォームが生成されます。

税金に関する質問の回答と IRS 報告書のしきい値に基づいて、Amazon は Form 1099-K を提出する場合があります。 Amazon は、お客様の納税金額がしきい値レベルに達した年の翌年の 1 月 31 日までに、フォーム 1099-K のコピーを郵送します。例えば、税金口座が 2018 年にしきい値に達した場合は、2019 年の 1 月 31 日までにフォーム 1099-K が郵送されます。

IRS の要件とフォーム 1099-K の詳細については、「IRS」のウェブサイトを参照してください。

リザーブドインスタンス 料金設定

リザーブドインスタンスの料金を設定するときは、次の点を考慮してください。

- 前払い料金 – 前払い料金は、販売するリザーブドインスタンスについて指定できる唯一の料金です。前払い料金は、購入者がリザーブドインスタンスを購入する際に支払う一括払いの料金です。

  リザーブドインスタンス の価値は時間の経過に伴い低下するため、AWS ではデフォルトで、1 か月ごとに同じ割合で低下するような価格設定を行っています。ただし、予約を販売する時期に基づいて、異なる前払い価格を設定できます。例えば、リザーブドインスタンス の残りの期間が 9 か月の場合、顧客が残り 9 か月の リザーブドインスタンス を購入する場合、受領する額を指定できます。残りが 5 か月である別の価格を設定し、さらに残りが 1 か月の別の価格を設定することができます。

  Reserved Instance Marketplace で許容される最低販売価格は、0.00 USD です。
- 制限 – リザーブドインスタンスの販売に関する次の制限は、AWS アカウント の有効期間に適用されます。1 年ごとの制限ではありません。
  - 最高販売額は 50,000 USD (リザーブドインスタンス) です。
  - 最高販売額は 5,000 (リザーブドインスタンス) です。

  通常、これらの制限を引き上げることはできませんが、リクエストに応じてケースバイケースで評価されます。制限の引き上げをリクエストするには、サービス制限の引き上げフォームに記入してください。[制限タイプ] で、[EC2 リザーブドインスタンスの販売] を選択します。
- [変更不可] – 出品内容を直接変更することはできません。ただし、最初に出品をキャンセルしてから、新しいパラメータで別の出品を作成することはできます。

- [キャンセル可能] – 出品は、active 状態であれば、いつでもキャンセルできます。既にマッチングされていたり、販売処理が行われている出品はキャンセルできません。出品したインスタンスの一部がマッチングされている場合にその出品をキャンセルすると、マッチングされていない残りのインスタンスが出品から削除されます。

リザーブドインスタンス のリスト

登録済みの販売者の場合、販売する リザーブドインスタンス を 1 つまたは複数選択できます。1 件のリストにすべてまとめて販売することも、個別に販売することもできます。さらに、インスタンスタイプ、プラットフォーム、スコープのすべての設定で リザーブドインスタンス を出品できます。

コンソールによって、提示価格が決定します。お客様の リザーブドインスタンス に一致するサービスを確認し、最低価格のサービスに合わせます。それ以外の場合、残り時間の リザーブドインスタンス のコストに基づいて提示価格を計算します。計算後の値が 1.01 USD 未満の場合、提示価格は 1.01 USD です。

出品をキャンセルする場合、その一部が既に売れているとき、売却済みの部分についてのキャンセルは無効です。出品されたうち、まだ売れていない部分のみが Reserved Instance Marketplace から削除されます。

AWS Management Console を使用して、Reserved Instance Marketplace に、リザーブドインスタンスを出品するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. 出品する リザーブドインスタンス を選択し、[Actions (アクション)]、[リザーブドインスタンスの出品] の順に選択します。
4. [Configure Your リザーブドインスタンス Listing (リザーブドインスタンス出品の設定)] ページで、販売するインスタンス数および残り期間に対する前払い価格を該当列に設定します。[Months Remaining] 列の隣にある矢印を選択して、残りの有効期間における予約の価値の変化状況を確認します。
5. 上級ユーザーが価格をカスタマイズする場合は、今後の月に異なる価格を入力できます。デフォルトの直線形の価格減少に戻すには、[Reset] を選択します。
6. 出品の設定が終了したら、[Continue] を選択します。
7. [Confirm Your リザーブドインスタンス Listing (リザーブドインスタンス出品の確認)] ページに表示された出品詳細を確認し、問題がなければ [List Reserved Instance (リザーブドインスタンスの出品)] を選択します。

出品したインスタンスをコンソールで表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. 出品した リザーブドインスタンス を選択し、ページ下部にある [My Listings (自分の出品)] タブを選択します。

AWS CLI を使用して、Reserved Instance Marketplace 内のリザーブドインスタンスを管理するには

1. describe-reserved-instances コマンドを使用して、リザーブドインスタンス の一覧を取得します。
2. 出品する リザーブドインスタンス の ID を書き留めて、create-reserved-instances-listing を呼び出します。リザーブドインスタンス の ID、インスタンスの数、価格体系を指定する必要があります。
3. ユーザーの出品を表示するには、describe-reserved-instances-listings コマンドを使用します。
4. 出品をキャンセルするには、cancel-reserved-instances-listings コマンドを使用します。

リザーブドインスタンス の出品状態

リザーブドインスタンス ページの [出品] タブの [出品状態] には、出品の現在状況が表示されます。

[出品状態] には、Reserved Instance Marketplace へのお客様の出品に関する情報が表示されます。これは、[リザーブドインスタンス] ページの [State] 列に表示される状態情報とは異なります。この [State] 情報は、お客様の予約に関するものです。

- [アクティブ] — 購入できます。
- [キャンセル済み] – 出品がキャンセルされ、Reserved Instance Marketplace での購入ができません。
- [closed (クローズ)] — リザーブドインスタンス は出品されていません。リザーブドインスタンス は、出品が完了したため [closed] になっている可能性があります。

出品のライフサイクル

出品したすべてのインスタンスがマッチングされて売れると、[My Listings] タブに表示される [Total instance count] が [Sold] の下に表示された数と同じになります。出品で残っている [Available] インスタンスがなくなり、[Status] が [closed] になります。

出品の一部だけが売却された場合、AWS では、出品されている リザーブドインスタンス を取り下げ、売却されなかった残りの リザーブドインスタンスと同数のリザーブドインスタンス を作成します。したがって、出品 ID とその ID の出品は、販売中の予約が少なくなっていますがアクティブのままです。

この出品内の リザーブドインスタンス の売却は今後、この方法で行われます。出品されたすべての リザーブドインスタンス が売却されると、AWS はその出品を `closed` としてマークします。

例えば、出品数が 5 のリザーブドインスタンス ID 5ec28771-05ff-4b9b-aa31-9e57dexample の出品を作成するとします。

コンソールの [Reserved Instance] ページの [My Listings] タブに、次のように出品が表示されます。

リザーブドインスタンス の出品 ID 5ec28771-05ff-4b9b-aa31-9e57dexample

- Total reservation count = 5
- Sold = 0
- Available = 5
- Status = active

購入者が 2 つの予約を購入すると、3 つの予約が販売用に残ります。一部分が売れたため、AWS では引き続き販売される残りの予約に相当する、3 つの新しい予約が作成されます。

お客様の出品は、[My Listings (自分の出品)] タブで次のように表示されます。

リザーブドインスタンス の出品 ID 5ec28771-05ff-4b9b-aa31-9e57dexample

- Total reservation count = 5
- Sold = 2
- Available = 3
- Status = active

出品をキャンセルする場合、その一部が既に売れているとき、売却済みの部分についてのキャンセルは無効です。出品されたうち、まだ売れていない部分のみがリザーブドインスタンス Marketplace から削除されます。

リザーブドインスタンス が売却された後

自分のリザーブドインスタンス が売却されると、AWS から E メールの通知が届きます。何らかのアクティビティがあった日ごとに、毎日のすべてのアクティビティをキャプチャした 1 通の E メール通知が送信されます。アクティビティには、出品の作成または販売や、AWS によるアカウントへの資金の送金などがあります。

コンソールで リザーブドインスタンス の出品の状態を追跡するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションページで、[リザーブドインスタンス] を選択します。
3. [My Listings (自分の出品)] タブを選択します。

   [My Listings] タブには、[Listing State] の値が含まれています。また、期間、出品価格、および出品されているインスタンスの中で使用可能、売却済み、およびキャンセルされたものがいくつあるかの詳細といった情報が示されます。

また、describe-reserved-instances-listings コマンドを適切なフィルタで使用することで、リザーブドインスタンスの出品に関する情報を取得できます。

支払いを受け取る

AWS が購入者からの支払い金を受領するとすぐに、販売された リザーブドインスタンス に登録されている所有者アカウントの E メールアドレスに、メッセージが送信されます。

AWS は指定された銀行口座に、自動決済機関 (ACH) の電子送金を送信します。通常、この送金は、リザーブドインスタンス の売却後 1 ~ 3 日の間に行われます。支払いは、1 日に 1 回行われます。送金が行われると、支払い報告がメールで届きます。AWS が銀行からの検証結果を受領するまで、支払い金を受け取れないことに注意してください。これには最大 2 週間かかることがあります。

販売した リザーブドインスタンス は、ユーザーの リザーブドインスタンス の詳細に引き続き表示されます。

販売者は、リザーブドインスタンスに対する現金の支払いを、振込によって直接各自の銀行口座で受け取ります。 AWSは、Reserved Instance Marketplace で販売するリザーブドインスタンスごとに、前払い料金の総額の 12% をサービス料として課金します。

購入者と共有する情報

Reserved Instance Marketplace で販売する場合、AWS は米国の規制に従って、お客様の正式な会社名を購入者向けのステートメントに表示します。さらに、請求書またはその他の税金関連の理由で、購入者から販売者に連絡したいとの要望が AWS Support にあった場合、購入者から販売者に直接連絡できるよう、AWS は必要に応じて販売者の E メールアドレスを購入者に提供する場合があります。

同様の理由で、販売者には購入者の郵便番号および国情報が支払いレポートによって提供されます。販売者として、この情報を、国に支払わなければならない取引税 (売上税や付加価値税など) に添付する必要がある場合があります。

AWS は税金に関する助言を行うことはできませんが、お客様が追加情報を必要としているとお客様の税務専門家が判断した場合は、AWS Support にお問い合わせください。

リザーブドインスタンス の変更

ニーズが変化したときは、スタンダードまたは コンバーティブルリザーブドインスタンス を変更し、引き続き料金上の利点を得られます。アベイラビリティー–ゾーン、インスタンスサイズ (同じインスタンスファミリーや世代で) やリザーブドインスタンスのスコープなどの属性を変更できます。

> **Note**
>
> また、別の構成で、別の コンバーティブルリザーブドインスタンス の コンバーティブルリザーブドインスタンス に交換することもできます。詳細については、コンバーティブルリザーブドインスタンス の交換 を参照してください。

すべての リザーブドインスタンス、またはそのサブセットを変更できます。元の リザーブドインスタンス を 2 つ以上の新しい リザーブドインスタンス に分割できます。例えば、us-east-1a に 10 のインスタンスを予約があり、そのうち 5 つのインスタンスを us-east-1b に移動する場合、結果的にこの変更は 2 つの新しい予約をリクエストします。us-east-1a での 5 つのインスタンス用の予約と us-east-1b での 5 つのインスタンス用の予約です。

また、2 つ以上の リザーブドインスタンス を単一の リザーブドインスタンス にマージすることもできます。例えば、それぞれに 1 つのインスタンスがある 4 つの t2.small リザーブドインスタンスがある場合、これらをマージして 1 つの t2.large リザーブドインスタンス を作成できます。詳細については、インスタンスサイズの変更のサポート を参照してください。

変更後、リザーブドインスタンス の利点は、リザーブドインスタンスの新しいパラメータと一致するインスタンスのみに適用されます。例えば、予約のアベイラビリティーゾーンを変更する場合、キャパシティーの予約と料金上の利点は、新しいアベイラビリティーゾーン内のインスタンスの使用に対して自動的に適用されます。新しいパラメータに一致しないインスタンスは、他に適用可能な予約がない場合、オンデマンド価格で課金されます。

変更リクエストが成功した場合。

- 変更後の予約がすぐに有効になり、変更リクエストが完了した時刻から、割引料金が新しいインスタンスに適用されます。例えば、午後 9 時 15 分に予約の変更が成功した場合、割引料金は午後 9 時 00 分から新しいインスタンスに移ります。変更された リザーブドインスタンス の発行日は describe-reserved-instances コマンドを使用して取得できます。
- 元の予約は終了します。その終了日は新しい予約の開始日であり、新しい予約の終了日は元の リザーブドインスタンス の終了日と同じです。有効期限のうち 16 か月が残っている 3 年の予約を正常に変更した場合、変更後の予約は 16 か月の予約であり、終了日は変更前の予約と同じです。
- 変更後の予約の固定価格は 0 USD であり、元の予約の固定価格ではありません。
- 変更後の予約の固定価格はアカウントに適用される割引料金範囲の計算に影響を与えません。割引範囲の計算は元の予約の固定価格に基づきます。

変更リクエストに失敗した場合、リザーブドインスタンス は元の設定を維持し、別の変更リクエストをすぐに利用できます。

変更に手数料は必要なく、新しく課金されたり、請求書が届いたりすることはありません。

予約の変更は必要に応じて何度でも行うことができますが、変更を送信後に保留中の変更リクエストを変更またはキャンセルすることはできません。変更が完了した後は、必要に応じて別の変更リクエストを送信して、実行した変更をロールバックできます。

コンテンツ

変更の要件と制限

以下のように、これらの属性を変更できます。

変更可能な属性	サポートされているプラットフォーム	制約事項と考慮事項
同じリージョン内でアベイラビリティーゾーンを変更する	Linux と Windows	-
スコープをアベイラビリティーゾーンからリージョンに、またはその逆に変更する	Linux と Windows	ゾーンのリザーブドインスタンスは、アベイラビリティーゾーンとそのアベイラビリティーゾーンのリザーブドキャパシティにスコープされます。スコープをアベイラビリティーゾーンからリージョンに (つまり、ゾーンからリージョナルに) 変更すると、キャパシティ予約のメリットが失われます。 リージョンのリザーブドインスタンスのスコープは、リージョンに限定されます。リザーブドインスタンスの割引は、そのリージョンの任意のアベイラビリティーゾーンで実行されているインスタンスに適用できます。さらに、リザーブドインスタンスの割引は、選択したインスタンスファミリのすべてのサイズのインスタンスの使用に適用されます。スコープをリージョンからアベイラビリティーゾーンに (つまり、リージョナルからゾーンに) 変更すると、アベイラビリティーゾーンの柔

変更可能な属性	サポートされているプラットフォーム	制約事項と考慮事項
		軟性とインスタンスサイズの柔軟性 (該当する場合) が失われます。 詳細については、「リザーブドインスタンス がどのように適用されるか」を参照してください。
同じインスタンスファミリーや世代でインスタンスサイズを変更する	Linux/UNIX のみ リザーブドインスタンス のインスタンスサイズの柔軟性は、他のプラットフォーム (Linux with SQL Server Standard、Linux with SQL Server Web、Linux with SQL Server Enterprise、Red Hat Enterprise Linux、SUSE Linux、Windows、Windows with SQL Standard、Windows with SQL Server Enterprise、および Windows with SQL Server Web) では利用できません。	予約ではデフォルトのテナンシーを使用する必要があります。使用できる他のサイズがないため、一部のインスタンスファミリーはサポートされません。詳細については、「インスタンスサイズの変更のサポート」を参照してください。

要件

変更リクエストは、新しい設定 (該当する場合) に対して十分なリザーブドインスタンス容量があり、以下の条件が満たされている場合に Amazon EC2 で処理されます。

- 購入と同時期またはその前に リザーブドインスタンス を変更できないこと
- リザーブドインスタンス がアクティブであること
- 保留中の変更リクエストがないこと

- リザーブドインスタンスは Reserved Instance Marketplace に出品されていないこと
- 元の予約のインスタンスサイズのフットプリントと新しい設定が一致している必要があります。詳細については、インスタンスサイズの変更のサポート を参照してください。
- 元の リザーブドインスタンス はすべてスタンダード リザーブドインスタンス あるいはすべて コンバーティブルリザーブドインスタンス であり、両方のタイプが混ざっていないこと
- スタンダード リザーブドインスタンス の場合、元の リザーブドインスタンス は同じ時間内に期限切れとなること
- リザーブドインスタンスが G4、G4ad、G4dn、G5、G5g、Inf1、または Inf2 インスタンスではないこと

インスタンスサイズの変更のサポート

次の要件が満たされている場合は、リザーブドインスタンス のインスタンスサイズを変更できます。

要件

- プラットフォームが Linux/UNIX であること。
- 同じインスタンスファミリー (T などの文字で示される) と世代 (2 などの数字で示される) 内の別のインスタンスサイズを選択する必要があります。

  例えば、リザーブドインスタンスはどちらも同じ T2 ファミリーおよび世代に属しているため、リザーブドインスタンスを `t2.small` から `t2.large` に変更できます。ただし、どちらの例でも、ターゲットインスタンスのファミリーと世代が元のリザーブドインスタンスと同じではないため、リザーブドインスタンスを T2 から M2 に、または T2 から T3 に変更することはできません。
- 以下の各インスタンスはサイズが 1 つしかないため、リザーブドインスタンスのインスタンスサイズを変更することはできません。
  - `t1.micro`
- 以下のインスタンスファミリー、世代、属性の組み合わせのリザーブドインスタンスのインスタンスサイズを変更することはできません。
  - G4ad
  - G4dn
  - G5
  - G5g
  - Inf1

  - Inf2
- 元のと新しい リザーブドインスタンス は、インスタンスサイズのフットプリントが同じであること。

コンテンツ

インスタンスサイズのフットプリント

各 リザーブドインスタンス にはインスタンスサイズのフットプリントがあり、これはインスタンスサイズの正規化係数と予約に含まれるインスタンスの数によって決まります。リザーブドインスタンス のインスタンスサイズを変更する場合、新しい設定のフットプリントは元の設定のフットプリントと一致する必要があります。一致しないと、変更リクエストは処理されません。

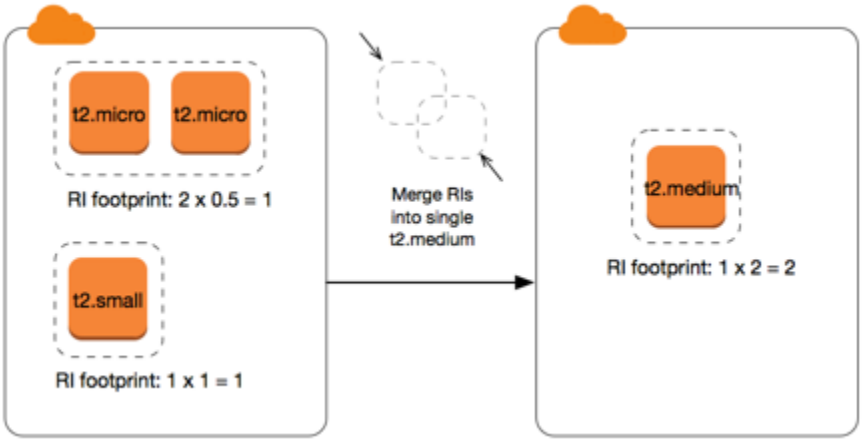
リザーブドインスタンス でインスタンスサイズのフットプリントを計算するには、インスタンスの数に正規化係数を掛けます。Amazon EC2 コンソールでは、正規化係数はユニットで測定されます。次の表に、インスタンスファミリーのインスタンスサイズの正規化係数を示します。例えば、`t2.medium` の正規化係数は 2 であるため、4 つの `t2.medium` インスタンスの予約は 8 ユニットのフットプリントになります。

インスタンスサイズ	正規化係数
nano	0.25
micro	0.5
small	1
medium	2
large	4
xlarge	8
2xlarge	16
3xlarge	24

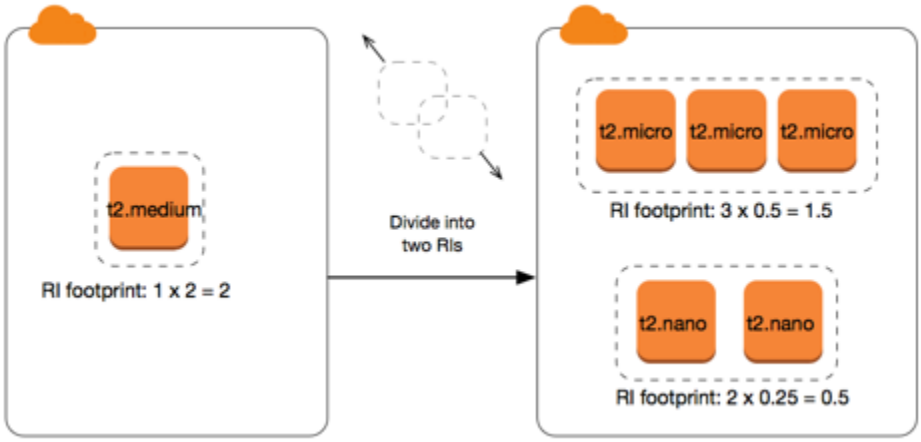
インスタンスサイズ	正規化係数
4xlarge	32
6xlarge	48
8xlarge	64
9xlarge	72
10xlarge	80
12xlarge	96
16xlarge	128
18xlarge	144
24xlarge	192
32xlarge	256
48xlarge	384
56xlarge	448
112xlarge	896

予約のインスタンスサイズのフットプリントが同じである場合は、同じインスタンスファミリー内で、予約を異なるインスタンスサイズとして割り当てることができます。例えば、1 つの t2.large (1 @ 4 ユニット) インスタンスの予約は 4 つの t2.small (4 @ 1 ユニット) インスタンスに分割できます。同様に、4 つの t2.small インスタンスの予約は 1 つの t2.large インスタンスにまとめることができます。ただし、2 つの t2.small インスタンスの予約を 1 つの t2.large インスタンスに変更することはできません。新しい予約のフットプリント (4 ユニット) が元の予約のフットプリント (2 ユニット) より大きいためです。

次の例では、2 つの t2.micro インスタンスの予約 (1 ユニット) と 1 つの t2.small インスタンスの予約 (1 ユニット) があります。両方の予約を 1 つの t2.medium インスタンスの予約 (2 ユニット) に結合すると、新しい予約のフットプリントと結合後の予約のフットプリントは等しくなります。



また、予約を変更して 2 つ以上の予約に分割することもできます。次の例では、1 つの t2.medium インスタンスの予約 (2 ユニット) があります。この予約を 2 つの予約に分割できます。t2.nano インスタンス 2 個の予約 (0.5 ユニット) と、t2.micro インスタンス 3 個の予約 (1.5 ユニット) です。



ベアメタルインスタンスの正規化係数

同じインスタンスファミリー内の他のサイズを使用して、metal インスタンスの予約を変更できます。同様に、同じインスタンスファミリー内の metal サイズを使用して、ベアメタルインスタンス以外のインスタンスの予約を変更できます。通常、ベアメタルインスタンスのサイズは、同じインスタンスファミリー内の最大のインスタンスサイズと同じです。例えば、i3.metal インスタンスは i3.16xlarge インスタンスと同じサイズであるため、正規化係数が同じになります。

次の表は、ベアメタルインスタンスを持つインスタンスファミリーのベアメタルインスタンスサイズの正規化係数を示しています。metal インスタンスの正規化係数は、他のインスタンスサイズとは異なり、インスタンスファミリーによって決まります。

インスタンスサイズ	正規化係数
a1.metal	32
m5zn.metal | z1d.metal	96
c6g.metal | c6gd.metal | i3.metal | m6g.metal | m6gd.metal | r6g.metal | r6gd.metal | x2gd.metal	128
c5n.metal	144
c5.metal | c5d.metal | i3en.metal | m5.metal | m5d.metal | m5dn.metal | m5n.metal | r5.metal | r5b.metal | r5d.metal | r5dn.metal | r5n.metal	192
u-*.metal	896

例えば、1 つの i3.metal インスタンスには 128 の正規化係数があります。i3.metal デフォルトテナンシー Amazon Linux/Unix リザーブドインスタンス を購入する場合、次のように予約を分割できます。

- i3.16xlarge は i3.metal インスタンスと同じサイズであるため、その正規化係数は 128 (128/1) です。1 つの i3.metal インスタンスの予約は、1 つの i3.16xlarge インスタンス内で変更できます。
- i3.8xlarge は i3.metal インスタンスの半分のサイズであるため、その正規化係数は 64 (128/2) です。1 つの i3.metal インスタンスの予約は、2 つの i3.8xlarge インスタンスに分割できます。
- i3.4xlarge は i3.metal インスタンスの 4 分の 1 のサイズであるため、その正規化係数は 32 (128/4) です。1 つの i3.metal インスタンスの予約は、4 つの i3.4xlarge インスタンスに分割できます。

変更リクエストの送信

リザーブドインスタンスを変更する前に、適用される制限を必ず確認してください。インスタンスのサイズを変更する前に、変更する元の予約のすべてのインスタンスサイズフットプリントを計算し、その結果が新しい構成の全インスタンスサイズフットプリントと一致することを確認してください。

New console

AWS Management Console を使用してリザーブドインスタンスを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [リザーブドインスタンス] ページで、変更する リザーブドインスタンス を 1 つ以上選択し、[アクション]、[リザーブドインスタンスの変更] の順に選択します。

   > **Note**
   >
   > リザーブドインスタンス がアクティブ状態ではない場合、または変更できない場合は、[リザーブドインスタンス の変更] が無効となります。

3. 変更テーブルの最初のエントリには、選択した リザーブドインスタンス の属性と、その下部に少なくとも 1 つのターゲット設定が表示されます。[単位] 列には全インスタンスサイズのフットプリントが表示されます。追加する各新規設定で 追加 を選択します。必要に応じて各構成の属性を変更します。

   - [スコープ]: 設定の適用先が 1 つのアベイラビリティーゾーンまたはリージョン全体のどちらであるかを選択します。
   - [アベイラビリティーゾーン]: 必要なアベイラビリティーゾーンを選択します。リージョン リザーブドインスタンス には適用されません。
   - [インスタンスタイプ]: 必要なインスタンスタイプを選択します。組み合わせた設定は、元の設定のインスタンスサイズのフットプリントと等しくなければなりません。
   - [カウント]: インスタンス数を指定します。リザーブドインスタンス を複数の設定に分割するには、カウントを減らし、[追加] を選択して、追加する設定のカウントを指定します。例えば、カウントが 10 の設定が 1 つある場合、そのカウントを 6 に変更し、カウントが 4 の設定を別に追加できます。このプロセスでは、新しい リザーブドインスタンス がアクティブになった後で、元の リザーブドインスタンス を終了させます。
4. [続行] をクリックします。
5. ターゲット設定の指定を完了し変更内容を確認したい場合は、[変更を送信] を選択します。
6. 変更リクエストのステータスは、リザーブドインスタンス 画面の [状態] 列で確認できます。有効な状態には以下のものがあります。

   - アクティブ (変更の保留) —元の リザーブドインスタンス の移行状態
   - リタイア (変更の保留) — 新しい リザーブドインスタンス を作成中の元の リザーブドインスタンス の移行状態

- リタイア — リザーブドインスタンス は正常に変更され、置き換えられました
- アクティブ — 次のいずれかを選択します。
  - 正常な変更リクエストにより新しい リザーブドインスタンス が作成されました
  - 変更リクエストが失敗したため、元の リザーブドインスタンス です

Old console

AWS Management Console を使用してリザーブドインスタンスを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [リザーブドインスタンス] ページで、変更する リザーブドインスタンス を 1 つ以上選択し、[アクション]、[リザーブドインスタンスの変更] の順に選択します。

   > **Note**
   >
   > リザーブドインスタンス がアクティブ状態ではない場合、または変更できない場合は、[リザーブドインスタンス の変更] が無効となります。

3. 変更テーブルの最初のエントリには、選択した リザーブドインスタンス の属性とその上部に少なくとも 1 つのターゲット設定が表示されます。[単位] 列には全インスタンスサイズのフットプリントが表示されます。追加する各新規設定で 追加 を選択します。各設定で必要に応じて属性を変更し、[続行] を選択します。

   - [スコープ]: 設定の適用先が 1 つのアベイラビリティーゾーンまたはリージョン全体のどちらであるかを選択します。
   - [アベイラビリティーゾーン]: 必要なアベイラビリティーゾーンを選択します。リージョン リザーブドインスタンス には適用されません。
   - [インスタンスタイプ]: 必要なインスタンスタイプを選択します。組み合わせた設定は、元の設定のインスタンスサイズのフットプリントと等しくなければなりません。
   - [カウント]: インスタンス数を指定します。リザーブドインスタンス を複数の設定に分割するには、カウントを減らし、[追加] を選択して、追加する設定のカウントを指定します。例えば、カウントが 10 の設定が 1 つある場合、そのカウントを 6 に変更し、カウントが 4 の設定を別に追加できます。このプロセスでは、新しい リザーブドインスタンス がアクティブになった後で、元の リザーブドインスタンス を終了させます。
4. ターゲット設定の指定を完了し変更内容を確認したい場合は、[変更を送信] を選択します。

5. 変更リクエストのステータスは、リザーブドインスタンス 画面の [状態] 列で確認できます。有効な状態には以下のものがあります。

   - アクティブ (変更の保留) —元の リザーブドインスタンス の移行状態
   - リタイア (変更の保留) — 新しい リザーブドインスタンス を作成中の元の リザーブドインスタンス の移行状態
   - リタイア — リザーブドインスタンス は正常に変更され、置き換えられました
   - アクティブ — 次のいずれかを選択します。
     - 正常な変更リクエストにより新しい リザーブドインスタンス が作成されました
     - 変更リクエストが失敗したため、元の リザーブドインスタンス です

コマンドラインを使用して リザーブドインスタンス を変更するには

1. リザーブドインスタンス を変更するには、次のコマンドの 1 つを使用できます。
   - modify-reserved-instances (AWS CLI)
   - Edit-EC2ReservedInstance (AWS Tools for Windows PowerShell)
2. 変更リクエスト (`processing`、`fulfilled`、または `failed`) のステータスを取得するには、以下のコマンドから 1 つを使用します。
   - describe-reserved-instances-modifications (AWS CLI)
   - Get-EC2ReservedInstancesModification (AWS Tools for Windows PowerShell)

変更リクエストのトラブルシューティング

リクエストしたターゲット設定が一意であれば、リクエストが処理されるメッセージを受信します。この時点では、Amazon EC2 は変更リクエストのパラメータが有効であることのみを確認しています。まだ、処理中に容量が利用できないために変更リクエストが失敗する可能性があります。

場合によって、確認の代わりに変更リクエストが不完全または失敗したことを示すメッセージが表示されることがあります。メッセージの情報を参考にして、別の変更リクエストを再送信します。リクエストを送信する前に、適用される制約を必ず確認してください。

選択された リザーブドインスタンス に変更できないものがあります

Amazon EC2 は変更できない リザーブドインスタンス を示します。このようなメッセージを受け取ったら、Amazon EC2 コンソールの [リザーブドインスタンス] ページ リザーブドインスタンス についての詳細情報を確認します。

変更リクエストの処理中にエラーが発生しました

送信した リザーブドインスタンス 変更リクエストをすべて処理できません。変更している予約の数によっては、メッセージが異なる場合があります。

Amazon EC2 は変更リクエストを処理できない理由を示します。例えば、変更している リザーブドインスタンス の 1 つ以上のサブセットに同じターゲット設定 (アベイラビリティーゾーンとプラットフォームの組み合わせ) を指定したような場合です。予約のインスタンス詳細が一致し、変更対象のすべてのサブセットのターゲット設定が一意であることを確認して、変更リクエストの再送信を試みます。

コンバーティブルリザーブドインスタンス の交換

また、インスタンスファミリー、オペレーティングシステム、およびテナンシーを含む別の構成で、1 つ以上の別の コンバーティブルリザーブドインスタンス の コンバーティブルリザーブドインスタンス に交換することもできます。新しいコンバーティブルリザーブドインスタンスが交換するコンバーティブルリザーブドインスタンスと同等あるいはそれ以上の値である限り、交換の実行回数に制限はありません。

コンバーティブルリザーブドインスタンスを交換する場合、現在の予約のインスタンスの数は、新しいコンバーティブルリザーブドインスタンスにおいて、同じ設定値かそれ以上のインスタンス数と交換されます。Amazon EC2 は、交換の結果として受け取ることができるリザーブドインスタンスの数を計算します。

スタンダード リザーブドインスタンス は交換できませんが、変更することはできます。詳細については、「リザーブドインスタンス の変更」を参照してください。

コンテンツ

コンバーティブルリザーブドインスタンス 交換の要件

以下の条件を満たしている場合に、Amazon EC2 では交換リクエストが処理されます。コンバーティブルリザーブドインスタンス は次のとおりである必要があります:

- アクティブ
- 保留中の以前の交換リクエストがないこと
- 有効期限が切れるまで、少なくとも 24 時間残っていること

以下のルールが適用されます。

- コンバーティブルリザーブドインスタンスは、その時点で AWS によって提供されている別のコンバーティブルリザーブドインスタンスにのみ、交換することができます。
- コンバーティブルリザーブドインスタンス は特定のリージョンと関連付けられ、予約の期間中は固定されます。コンバーティブルリザーブドインスタンス を別のリージョンの コンバーティブルリザーブドインスタンス と交換することはできません。
- 1 つの コンバーティブルリザーブドインスタンス で 1 つ以上の コンバーティブルリザーブドインスタンス を一度に交換することができます。
- コンバーティブルリザーブドインスタンス の一部を交換するには、2 つ以上の予約に変更して、1 つ以上の予約を新しい コンバーティブルリザーブドインスタンス に交換することができます。詳細については、コンバーティブルリザーブドインスタンス の一部の交換 を参照してください。リザーブドインスタンス の変更の詳細については、「リザーブドインスタンス の変更」を参照してください。
- 全額前払い コンバーティブルリザーブドインスタンス は一部前払い コンバーティブルリザーブドインスタンス に交換できます。その逆も可能です。

> **Note**
>
> 交換に必要な前払いの合計 (調整額) が 0.00 USD 未満の場合、その額が 0.00 USD 以上になるだけのコンバーティブルリザーブドインスタンス のインスタンス数が、AWS によって自動的に提供されます。

> **Note**
>
> 新しいコンバーティブルリザーブドインスタンス の合計料金 (前払い料金 + 時間料金 x 残り時間数) が交換前のコンバーティブルリザーブドインスタンスの合計料金未満の場合、交換前のコンバーティブルリザーブドインスタンスの合計料金以上になるように、AWS によって自動的にコンバーティブルリザーブドインスタンスのインスタンス数が提供されます。

- より有利な料金を利用するために、前払いなし コンバーティブルリザーブドインスタンス を全額前払いまたは一部前払い コンバーティブルリザーブドインスタンス に交換できます。
- 全額前払いまたは一部前払い コンバーティブルリザーブドインスタンス を前払いなし コンバーティブルリザーブドインスタンス に交換することはできません。
- 前払いなし コンバーティブルリザーブドインスタンス を別の前払いなし コンバーティブルリザーブドインスタンス に交換できます。ただし、新しい コンバーティブルリザーブドインスタンス の時間料金が交換元の コンバーティブルリザーブドインスタンス の時間料金以上である場合に限ります。

> **Note**
>
> 新しいコンバーティブルリザーブドインスタンスの合計料金 (時間料金 x 残り時間数) が交換前のコンバーティブルリザーブドインスタンスの合計料金未満の場合、交換前のコンバーティブルリザーブドインスタンスの合計料金以上になるように、AWS によって自動的にコンバーティブルリザーブドインスタンスのインスタンス数が提供されます。

- 有効期限の異なる複数の コンバーティブルリザーブドインスタンス を交換すると、新しい コンバーティブルリザーブドインスタンス の有効期限は、将来の最も遠い日付になります。
- 1 つの コンバーティブルリザーブドインスタンス を交換する場合は、新しい コンバーティブルリザーブドインスタンス と同じ期間 (1 年または 3 年) が必要です。異なる期間を持つ複数の コンバーティブルリザーブドインスタンス をマージすると、新しい コンバーティブルリザーブドインスタンス の期間は 3 年となります。詳細については、「コンバーティブルリザーブドインスタンス のマージ」を参照してください。
- Amazon EC2 がコンバーティブルリザーブドインスタンスを交換すると、関連する予約が取り消され、終了日が新しいリザベーションに転送されます。交換後、Amazon EC2 は古いリザベーションの終了日と新しいリザベーションの開始日の両方を交換日と同じ日付に設定します。例えば、有効期限のうち 16 か月が残っている 3 年の予約を交換する場合、新しい予約は 16 か月のリザベーションであり、終了日は交換したコンバーティブルリザーブドインスタンスのリザベーションと同じ日付です。

コンバーティブルリザーブドインスタンス の交換の計算

コンバーティブルリザーブドインスタンス の交換は無料です。ただし、前払い額を按分計算した結果、所有していたコンバーティブルリザーブドインスタンスと交換して受け取る新しいコンバーティブルリザーブドインスタンスに差額があれば、その清算額を支払う必要がある場合があります。

それぞれの コンバーティブルリザーブドインスタンス に定価があります。交換元と交換先の コンバーティブルリザーブドインスタンス で定価が比較され、交換の結果として得られるコンバーティブルリザーブドインスタンスの数が決まります。

例えば、定価 35 USD の 1 つの コンバーティブルリザーブドインスタンス を定価 10 USD の新しいインスタンスタイプに交換するとします。

```
$35/$10 = 3.5
```

コンバーティブルリザーブドインスタンス を 10 USD の 3 つの コンバーティブルリザーブドインスタンス に交換できます。半個単位で購入することはできないため、余り分を補うには、コンバーティブルリザーブドインスタンス を追加購入する必要があります。

```
3.5 = 3 whole Convertible Reserved Instances + 1 additional Convertible Reserved
 Instance
```

4 つ目の コンバーティブルリザーブドインスタンス で、終了日が他の 3 つのものと同じものだとすると、一部前払いまたは全額前払いの コンバーティブルリザーブドインスタンス を交換する場合、その 4 つ目のリザーブドインスタンスの料金を差額として支払います。交換元の コンバーティブルリザーブドインスタンス の前払い額のうち 500 USD が残っており、交換先のコンバーティブルリザーブドインスタンスの料金が按分計算で 600 USD になるとすると、100 USD が請求されます。

```
$600 prorated upfront cost of new reservations - $500 remaining upfront cost of old
 reservations = $100 difference
```

コンバーティブルリザーブドインスタンス のマージ

2 つ以上のコンバーティブルリザーブドインスタンスをマージする場合、新しいコンバーティブルリザーブドインスタンスの期限は、元のコンバーティブルリザーブドインスタンスの期限がすべて同じであればその期限、そうでなければ元のコンバーティブルリザーブドインスタンスの最も遅い期限になります。新しい コンバーティブルリザーブドインスタンス の有効期間は、将来の最も長い有効期間となります。

例えば、アカウントに以下の コンバーティブルリザーブドインスタンス があるとします。

Reserved Instance ID	用語	有効期限日
aaaa1111	1年	2018-12-31

Reserved Instance ID	用語	有効期限日
bbbb2222	1年	2018-07-31
cccc3333	3年	2018-06-30
dddd4444	3年	2019-12-31

- aaaa1111 と bbbb2222 をマージして、それらを 1 年の コンバーティブルリザーブドインスタンス と交換できます。それらを 3 年の コンバーティブルリザーブドインスタンス に交換することはできません。新しい コンバーティブルリザーブドインスタンス 有効期限は 2018-12-31 です。
- bbbb2222 と cccc3333 をマージして、それらを 3 年の コンバーティブルリザーブドインスタンス と交換できます。それらを 1 年の コンバーティブルリザーブドインスタンス に交換することはできません。新しい コンバーティブルリザーブドインスタンス 有効期限は 2018-07-31 です。
- cccc3333 と dddd4444 をマージして、それらを 3 年の コンバーティブルリザーブドインスタンス と交換できます。それらを 1 年の コンバーティブルリザーブドインスタンス に交換することはできません。新しい コンバーティブルリザーブドインスタンス 有効期限は 2019-12-31 です。

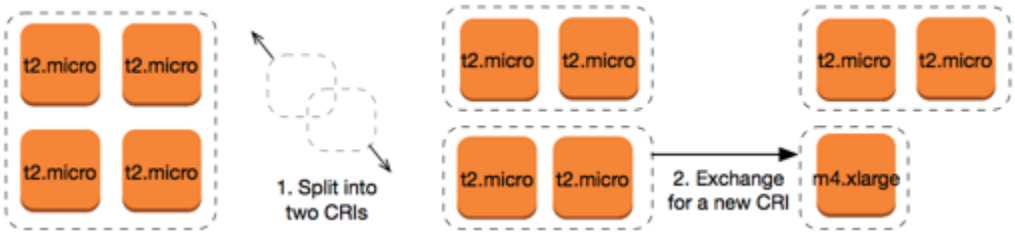
コンバーティブルリザーブドインスタンス の一部の交換

変更プロセスを使用して、コンバーティブルリザーブドインスタンス をより小さい予約に分割し、新しい予約のうちの 1 つ以上を新しい コンバーティブルリザーブドインスタンス と交換することができます。次の例はそれを行う方法を示しています。

Example 例: 複数のインスタンスを持つ コンバーティブルリザーブドインスタンス

この例では、この例では、予約に 4 つのインスタンスがある t2.micro コンバーティブルリザーブドインスタンス があります。t2.micro インスタンスに対して 2 つの m4.xlarge インスタンスを交換するには:

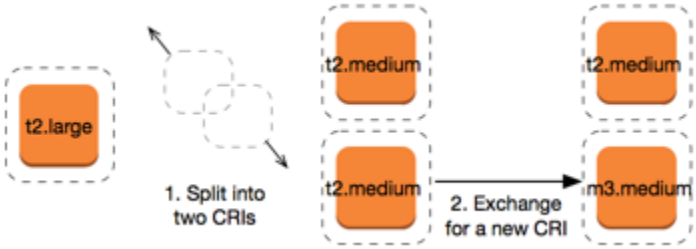
1. t2.micro コンバーティブルリザーブドインスタンス を変更するには、それぞれ 2 つの t2.micro コンバーティブルリザーブドインスタンス に分割します。
2. 新しい t2.micro コンバーティブルリザーブドインスタンス のいずれかを m4.xlarge コンバーティブルリザーブドインスタンス と交換します。



Example 例: 1 つのインスタンスを持つ コンバーティブルリザーブドインスタンス

この例では、t2.large コンバーティブルリザーブドインスタンス があります。小さな t2.medium インスタンスと m3.medium インスタンスに変更するには:

1. t2.large コンバーティブルリザーブドインスタンス を変更するには、2 つの t2.medium コンバーティブルリザーブドインスタンス に分割します。1 つの t2.large インスタンスに対して、2 つの t2.medium インスタンスと同じインスタンスサイズのフットプリントが含まれます。
2. 新しい t2.medium コンバーティブルリザーブドインスタンス のいずれかを m3.medium コンバーティブルリザーブドインスタンス と交換します。



詳細については、「インスタンスサイズの変更のサポート」および「交換リクエストの送信」を参照してください。

交換リクエストの送信

Amazon EC2 コンソールまたはコマンドラインツールを使って、コンバーティブルリザーブドインスタンス を交換できます。

コンソールを使用した コンバーティブルリザーブドインスタンス の交換

コンバーティブルリザーブドインスタンス 提供タイプを検索し、検索された選択肢の中から新しい設定を選択できます。

New console

Amazon EC2 コンソールを使用して コンバーティブルリザーブドインスタンス を交換するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [リザーブドインスタンス] を選択し、交換する コンバーティブルリザーブドインスタンス を選び、[アクション]、[リザーブドインスタンス の交換] の順に選択します。
3. 対象となる設定の属性を選択し、[提供タイプの検索] をクリックします。
4. 新しい コンバーティブルリザーブドインスタンス を選択します。画面の下部に、交換のために受け取った リザーブドインスタンスの番号と追加コストが表示されます。
5. 必要に応じて コンバーティブルリザーブドインスタンス を選択したら、[確認] をクリックします。
6. [交換]、[閉じる] の順にクリックします。

Old console

Amazon EC2 コンソールを使用して コンバーティブルリザーブドインスタンス を交換するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [リザーブドインスタンス] を選択し、交換する コンバーティブルリザーブドインスタンス を選び、[アクション]、[リザーブドインスタンス の交換] の順に選択します。
3. 対象となる構成の属性を選択し、「提供タイプの検索」をクリックします。
4. 新しい コンバーティブルリザーブドインスタンス を選択します。[インスタンス数] 列には、交換で受け取る リザーブドインスタンス の数が表示されます。ニーズに応じる コンバーティブルリザーブドインスタンス を選択したら、[交換] を選択します。

交換元の リザーブドインスタンス が消え、新しい リザーブドインスタンス が Amazon EC2 に表示されます。このプロセスが反映されるまでには数分かかることがあります。

コマンドラインインターフェイスを使用した コンバーティブルリザーブドインスタンス の交換

コンバーティブルリザーブドインスタンス を交換するには、まず自分のニーズに合った新しい コンバーティブルリザーブドインスタンス を見つけます。

- describe-reserved-instances-offerings (AWS CLI)
- Get-EC2ReservedInstancesOffering (Tools for Windows PowerShell)

交換で受け取る リザーブドインスタンス の数と交換時の差額の起案額を含む交換の見積りを取得します。

- get-reserved-instances-exchange-quote (AWS CLI)
- GetEC2-ReservedInstancesExchangeQuote (Tools for Windows PowerShell)

これで、交換を実行できます。

- accept-reserved-instances-exchange-quote (AWS CLI)
- Confirm-EC2ReservedInstancesExchangeQuote (Tools for Windows PowerShell)

リザーブドインスタンスのクォータ

毎月新しいリザーブドインスタンスを購入できます。毎月購入できる新しいリザーブドインスタンスの数は、次のように 1 か月ごとのクォータによって決まります。

クォータの説明	デフォルトのクォータ
新しいリージョンレベルのリザーブドインスタンス	リージョンあたり 20/月
新しいゾーンレベルのリザーブドインスタンス	アベイラビリティーゾーンあたり 20/月

例えば、3 つのアベイラビリティーゾーンがあるリージョンでは、デフォルトのクォータは 1 か月あたり 80 個の新規リザーブドインスタンスとなります。これは、次のように計算されます。

- リージョンの 20 個のリージョンレベルのリザーブドインスタンス
- さらに 60 個のゾーンレベルのリザーブドインスタンス (3 つのアベイラビリティーゾーン用に 20 個ずつ)

クォータは実行中のインスタンスにのみ適用されます。インスタンスが保留中、停止中、停止済み、または休止状態の場合、クォータにはカウントされません。

購入したリザーブドインスタンスの数を表示する

購入するリザーブドインスタンスの数は、[Instance count] (インスタンス数) フィールド (コンソール) または `InstanceCount` パラメータ (AWS CLI) によって示されます。新しいリザーブドインス

タンスを購入すると、クォータはインスタンスの総数に照らして測定されます。例えば、インスタンス数が 10 個のリザーブドインスタンス設定を 1 つ購入した場合、その購入はクォータに対して 1 ではなく 10 としてカウントされます。

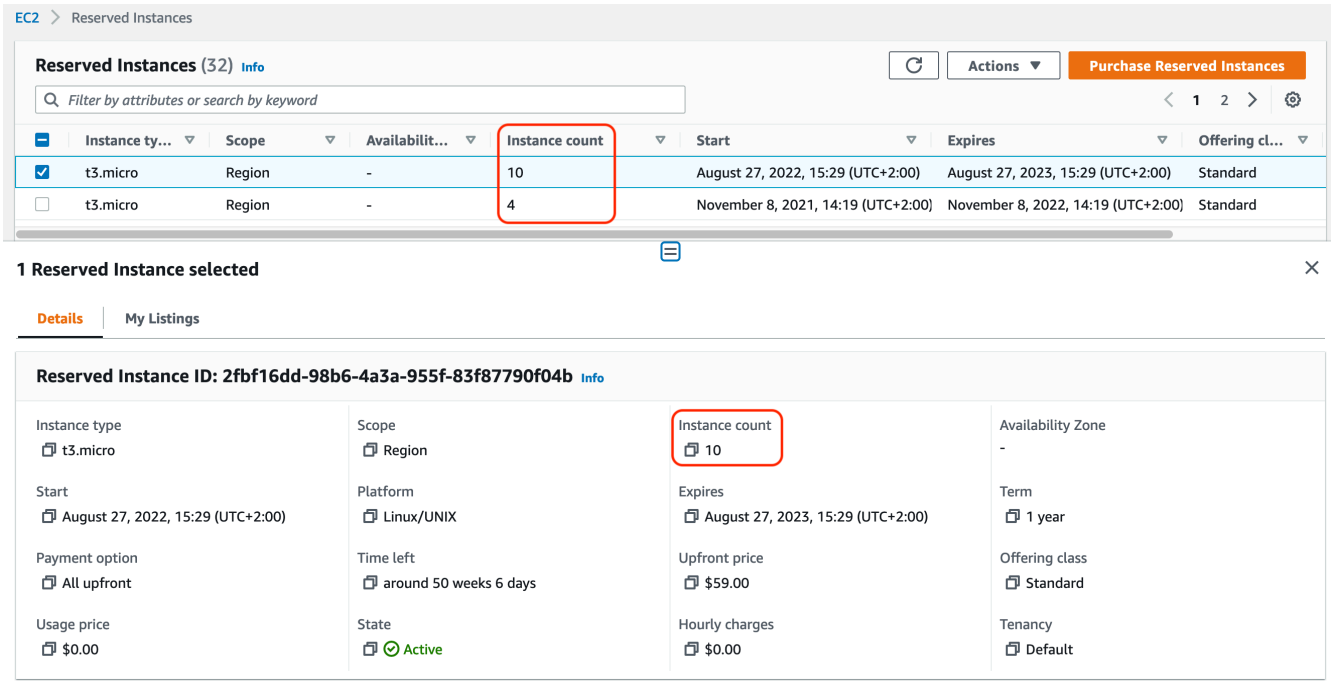
Amazon EC2 または AWS CLI を使用して、購入したリザーブドインスタンスの数を確認できます。

Console

購入したリザーブドインスタンスの数を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Reserved Instances] を選択します。
3. テーブルからリザーブドインスタンス設定を選択し、[Instance count] (インスタンス数) フィールドを確認します。

   次のスクリーンショット内の選択されている行は、`t3.micro` インスタンスタイプの単一のリザーブドインスタンス設定を表しています。テーブルビューの [Instance count] (インスタンス数) 列と詳細ビューの [Instance count] (インスタンス数) フィールド (スクリーンショット参照) は、この設定に 10 個のリザーブドインスタンスがあることを示しています。



AWS CLI

購入したリザーブドインスタンスの数を表示するには

describe-reserved-instances CLI コマンドを使用して、リザーブドインスタンス設定の ID を指定します。

```
aws ec2 describe-reserved-instances \
    --reserved-instances-ids 2fbf16dd-98b6-4a3a-955f-83f87790f04b \
    --output table
```

出力例 – [InstanceCount] フィールドは、この設定用に 10 個のリザーブドインスタンスがあることを示しています。

```
-------------------------------------------------------------------
|                    DescribeReservedInstances                    |
+-----------------------------------------------------------------+
||                       ReservedInstances                       ||
|+----------------------+----------------------------------------+|
||  CurrencyCode        |  USD                                   ||
||  Duration            |  31536000                              ||
||  End                 |  2023-08-27T13:29:44+00:00             ||
||  FixedPrice          |  59.0                                  ||
||  InstanceCount       |  10                                    ||
||  InstanceTenancy     |  default                               ||
||  InstanceType        |  t3.micro                              ||
||  OfferingClass       |  standard                              ||
||  OfferingType        |  All Upfront                           ||
||  ProductDescription  |  Linux/UNIX                            ||
||  ReservedInstancesId |  2fbf16dd-98b6-4a3a-955f-83f87790f04b  ||
||  Scope               |  Region                                ||
||  Start               |  2022-08-27T13:29:45.938000+00:00      ||
||  State               |  active                                ||
||  UsagePrice          |  0.0                                   ||
|+----------------------+----------------------------------------+|
|||                     RecurringCharges                        |||
||+---------------------------------+--------------------------+||
|||  Amount                         |  0.0                     |||
|||  Frequency                      |  Hourly                  |||
||+---------------------------------+--------------------------+||
```

考慮事項

リージョン リザーブドインスタンス では、オンデマンドインスタンス の実行に割引が適用されます。デフォルトの オンデマンドインスタンス の制限は 20 です。リージョン オンデマンドインスタ

ンス を購入すると、リザーブドインスタンス の実行制限を超えることはできません。例えば、すでに オンデマンドインスタンス を 20 回実行していて、20 のリージョン リザーブドインスタンス を購入した場合、20 のリージョン リザーブドインスタンス には 20 回の オンデマンドインスタンス の実行に割引が適用されます。さらに多くのリージョン リザーブドインスタンス を購入した場合は、オンデマンドインスタンス の制限に達しているため、さらにインスタンスを起動することはできません。

リージョン リザーブドインスタンス を購入する前に、オンデマンドインスタンス の制限数が所有する予定のリージョン リザーブドインスタンス の数に一致するかそれを超えることを確認してください。必要に応じて、さらにリージョン リザーブドインスタンス を購入する前に オンデマンドインスタンス の制限数の増加を依頼してください。

ゾーンレベルのリザーブドインスタンス (特定のアベイラビリティーゾーンで購入されたリザーブドインスタンス) は、キャパシティ予約と割引を提供します。ゾーン リザーブドインスタンス を購入することで、実行中の オンデマンドインスタンス の制限を超えることができます。例えば、すでに 20 の オンデマンドインスタンス を実行していて、20 のゾーン リザーブドインスタンス を購入した場合は、ゾーン リザーブドインスタンス の仕様に一致する 20 の オンデマンドインスタンス をさらに起動して、合計 40 実行インスタンスを実行できます。

リザーブドインスタンスのクォータを表示してクォータの引き上げをリクエストする

Amazon EC2 コンソールでクォータ情報を確認できます。クォータの引き上げをリクエストすることもできます。詳細については、現在の制限を表示するにはおよび引き上げのリクエストを参照してください。

スポットインスタンス

スポットインスタンスは、休止中の EC2 キャパシティーを使用するインスタンスで、オンデマンド価格より低料金で利用できます。スポットインスタンスでは未使用の EC2 インスタンスを大幅な割引価格でリクエストできるため、Amazon EC2 のコストを大幅に削減できます スポットインスタンスの時間料金は、スポット料金と呼ばれます。各アベイラビリティーゾーンにおける各インスタンスタイプのスポット料金は、Amazon EC2 によって設定され、スポットインスタンス の長期的な需給に基づいて徐々に調整されます。スポットインスタンスは、キャパシティが利用可能なときに、いつでも実行されます。

スポットインスタンス は、アプリケーションを実行する時間に柔軟性がある場合や、アプリケーションを中断できる場合に、費用効率の高い選択肢です。例えば、スポットインスタンス は、データ分析、バッチジョブ、バックグラウンド処理、およびオプションタスクに適しています。詳細については、「Amazon EC2スポットインスタンス」を参照してください。

EC2 インスタンスのさまざまな購入オプションの比較については、「インスタンス購入オプション」を参照してください。

トピック

- 概念
- 開始方法
- 関連サービス
- 料金と削減額

概念

スポットインスタンスを使用するときは、事前に以下の概念を理解しておく必要があります。

- スポットキャパシティープール – インスタンスタイプ (`m5.large` など) とアベイラビリティーゾーンが同一で、使用されていない EC2 インスタンスのセットです。
- スポット料金 – スポットインスタンスの現在の料金です (時間あたり)。
- スポットインスタンスリクエスト – スポットインスタンスに対するリクエストです。キャパシティーが利用可能になると、Amazon EC2 がリクエストを実行します。スポットインスタンスリクエストには、ワンタイムと永続の2種類があります。リクエストに関連付けられたスポットインスタンスが中断された後、Amazon EC2 は永続的スポットインスタンスリクエストを自動的に再送信します。

- EC2 インスタンスの再調整に関する推奨事項 - Amazon EC2は、インスタンスの再調整に関する推奨事項のシグナルを発し、スポットインスタンスにおいて中断のリスクが高まったことをユーザーに通知します。このシグナルにより、スポットインスタンスで中断 2 分前の通知が発信されていなくても、ユーザーは既存の、または新しいスポットインスタンスについて、前もってワークロードを再調整することができます。
- スポットインスタンスの中断 – Amazon EC2 が容量を戻してもらう必要がある場合には、Amazon EC2 はスポットインスタンスを終了、停止、または休止状態にします。Amazon EC2 は、スポットインスタンスが中断される 2 分前に、そのインスタンスに対し中断を警告するための通知を送信します。

スポットインスタンス と オンデマンドインスタンス の主な違い

次の表は、スポットインスタンスとオンデマンドインスタンスの主な違いをまとめたものです。

	Spot Instances	On-Demand Instances
作成時刻	スポットインスタンスリクエストがアクティブであり、利用可能なキャパシティーがある場合に限り即時に起動できます。	手動で起動リクエストを実行し、容量が利用可能である場合に限り、即時に起動できます。
使用可能な容量	利用可能なキャパシティ - がない場合、スポットインスタンスリクエストは、キャパシティ - が利用可能になるまで継続して自動的に起動リクエストを実行します。	起動リクエストを行うときに容量が利用可能でない場合は、容量不足エラー (ICE) が表示されます。
時間料金	スポットインスタンスの 1 時間単位の使用料金は、長期的な需要と供給に基づいて変化します。	オンデマンドインスタンス の時間単位の使用料金は固定です。
再調整に関する推奨事項	実行中のスポットインスタンスにおいて中断のリスクが高まった場合に、Amazon EC2 はそのインスタンスに対してシグナルを発します。	お客様は、いつオンデマンドインスタンスが中断 (停止、休止、または終了) されるかを決定します。

	Spot Instances	On-Demand Instances
インスタンスの中断	ユーザーは、Amazon EBS-backed スポットインスタンスを停止および開始することができます。さらに、キャパシティーが利用できなくなった場合、Amazon EC2 は個々のスポットインスタンスを中断することができます。	お客様は、いつオンデマンドインスタンスが中断 (停止、休止、または終了) されるかを決定します。

開始方法

最初に必要なことは、Amazon EC2 を使用するためのセットアップを行うことです。また、スポットインスタンス を起動する前に、オンデマンドインスタンス を起動した経験があると役立ちます。

起動と実行

- Amazon EC2 を使用するようにセットアップする
- チュートリアル: Amazon EC2 Linux インスタンスの開始方法

スポットの基本

- スポットインスタンス のしくみ

スポットインスタンス の操作

- スポットインスタンスリクエストを作成する
- リクエストステータス情報の取得
- スポットインスタンスの中断。

関連サービス

Amazon EC2 を使用して スポットインスタンス を直接プロビジョニングすることができます。また、他の AWS のサービスを使用して、スポットインスタンスをプロビジョニングすることもできます。詳細については、次のドキュメントを参照してください。

Amazon EC2 Auto Scaling および スポットインスタンス

Amazon EC2 Auto Scaling でスポットインスタンスを起動できるように、起動テンプレートまたは起動設定を作成できます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドのフォールトトレラントで柔軟性のあるアプリケーション用の スポットインスタンス のリクエストおよび複数のインスタンスタイプを持つ Auto Scalingグループとと購入オプションをご参照ください。

Amazon EMR および スポットインスタンス

シナリオによっては、Amazon EMR クラスターで スポットインスタンス を実行すると便利な場合があります。詳細については、『Amazon EMR 管理ガイド』の「スポットインスタンス」および「スポットインスタンス はどのような場合に使用しますか?」を参照してください。

AWS CloudFormation テンプレート

AWS CloudFormation を使用することで、JSON 形式のテンプレートを使用して、AWS リソースのコレクションを作成および管理できます。詳細については、「EC2 スポットインスタンスの更新 – Auto Scaling と CloudFormation の統合」を参照してください。

AWS SDK for Java

Java プログラミング言語を使用して、スポットインスタンス を管理できます。詳細については、「チュートリアル: Amazon EC2 スポットインスタンス」と「チュートリアル: Amazon EC2 スポットリクエストの高度な管理」を参照してください。

AWS SDK for .NET

.NET プログラミング環境を使用して、スポットインスタンス を管理できます。詳細については、「チュートリアル: Amazon EC2 スポットインスタンス」を参照してください。

料金と削減額

スポットインスタンス はスポット料金で課金されます。これは Amazon EC2 によって設定され、スポットインスタンス の長期供給と需要に基づいて徐々に調整されます。スポットインスタンス は、お客様が自らスポットインスタンスを終了するか、容量が使用できなくなるか、スケールイン時に Amazon EC2 Auto Scaling グループのインスタンスが削除されるまで実行されます。

ユーザー または Amazon EC2 が実行中のスポットインスタンスを中断した場合、使用しているオペレーティングシステムおよび中断したユーザーに応じて、使用した秒数または時間数の料金が請求されます (料金が発生しない場合もあります)。詳細については、中断された スポットインスタンス の請求 を参照してください。

料金の表示

AWS リージョン およびインスタンスタイプごとに、現在の (5 分ごとに更新される) 最低スポット料金を確認するには、Amazon EC2 スポットインスタンスの料金ページを参照してください。

過去 3 か月間のスポット価格の履歴を表示するには、Amazon EC2 コンソールを使用するか、describe-spot-price-history コマンド (AWS CLI) を使用します。詳細については、「スポットインスタンスの料金履歴」を参照してください。

AWS アカウント ごとに、個々のアベイラビリティーゾーンがコードにマッピングされます。したがって、アカウント間で同じアベイラビリティーゾーンコード (例えば、us-west-2a) に対して結果が異なる場合があります。

削減額の表示

スポットインスタンスを 1 つのスポットフリートまたはすべてのスポットインスタンスに対して使用することで得られる節約額を確認できます。過去 1 時間または過去 3 日間の削減状況を表示でき、vCPU 時間あたりの平均コストとメモリ (GiB) 時間あたりの平均コストも確認できます。削減額が予想されますが、使用状況に対する請求の調整が含まれていないため、実際の削減額と異なる場合があります。削減額情報の表示の詳細については、「スポットインスタンス 購入による削減額」を参照してください。

請求書の表示

請求書には、サービスの使用量に関する詳細が記載されています。詳細については、AWS Billing ユーザーガイドの「請求書の表示」を参照してください。

EC2 スポットを利用するうえでのベストプラクティス

Amazon EC2 スポットインスタンス は、AWS クラウド クラウドに用意された予備の EC2 コンピューティング性能であり、オンデマンド料金に比べて最大 90% の節約が可能です。オンデマンドインスタンス と スポットインスタンス の唯一の違いは、 Amazon EC2 が容量を必要とするときに、Amazon EC2 が スポットインスタンス を中断できることです。この中断の際には、2 分前に通知があります。

スポットインスタンス は、ステートレスかつフォールトトレラントで、柔軟性の高いアプリケーションに適しています。例えば、 スポットインスタンス はビッグデータ、コンテナ化されたワークロード、CI/CD、ステートレスウェブサーバー、ハイパフォーマンスコンピューティング (HPC)、レンダリングワークロードに適しています。

実行中、 スポットインスタンス は オンデマンドインスタンス とまったく同じ動作をします。ただし、スポットは、ワークロードが完了するまで十分な期間、実行中のインスタンスが動作し続けることを保証するものではありません。また、スポットは必要としているインスタンスをすぐに取得できること、またはリクエストした総容量がいつでも取得できることを保証するものではありません。さらに、スポットインスタンスの可用性は需要と供給によって変化し、将来のパフォーマンスが過去の実績により保証されるものではないため、スポットインスタンスの容量や発生する中断は、時間の経過とともに変化する可能性があります。

スポットインスタンス は、柔軟性がない、ステートフル、フォールトイントレラント、またはインスタンスノード間で緊密に結合されているワークロードには適していません。また、ターゲット容量が完全に利用できない場合がある期間に耐えられないワークロードにはお勧めしません。こうしたワークロードに スポットインスタンス を使用したり、中断を処理するために オンデマンドインスタンス へのフェールオーバーを試みたりしないよう強く警告します。

スポットの使用に慣れている場合でも、スポットインスタンスを使い始めたばかりの場合でも、スポットインスタンスの中断や可用性に関する問題が発生している場合には、スポットサービスを最大限に活用できるよう、これらのベストプラクティスに従うことをお勧めします。

スポットを利用するうえでのベストプラクティス

- 中断に備えて個々のインスタンスを準備する
- インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する
- EC2 Auto Scaling グループまたは EC2 フリートを使用して総容量を管理する
- 価格と容量を最適化する配分戦略を使用する
- 事前の容量再調整の利用
- 統合された AWS のサービスを使用して スポットインスタンス を管理する
- 使用すべき最適なスポットリクエスト方法はどれですか?

中断に備えて個々のインスタンスを準備する

スポットインスタンスの中断を適切に処理する最善の方法は、耐障害性のあるアプリケーションを設計することです。これを実現するためには、EC2 インスタンスの再調整に関する推奨事項、ならびにスポットインスタンスの中断通知を利用できます。

EC2 インスタンスの再調整に関するレコメンデーションは、スポットインスタンスで中断のリスクが高まった場合に通知するためのシグナルです。ユーザーは、このシグナルにより、スポットインスタンスの中断 2 分前の通知が届いていない段階で、事前にスポットインスタンスの管理を行えま

す。ワークロードを、中断のリスクが高くない新規または既存の スポットインスタンス に再調整することができます。このシグナルは、Auto Scaling グループと EC2 フリートの容量の再分散機能を使うことで簡単に利用できます。詳細については、「事前の容量再調整の利用」を参照してください。

スポットインスタンスの中断通知は、Amazon EC2 がスポットインスタンスを中断する 2 分前に発行される警告です。ワークロードに時間の面での柔軟性がある場合は、中断が発生した際にそれを終了するのではなく、停止または休止状態になるようにスポットインスタンスを設定することができます。Amazon EC2 は、中断が発生したスポットインスタンスを自動的に停止または休止状態にし、使用可能な容量が確保できた際にはそのインスタンスを自動的に再開します。

再調整に関する推奨事項と中断通知をキャプチャし、ワークロードの進行状況のチェックポイントをトリガーするか、中断を適切に処理するルールを Amazon EventBridge で作成することをお勧めします。詳細については、再調整に関する推奨事項シグナルのモニタリング を参照してください。イベントルールの作成および使用方法の詳細な例については、「Taking Advantage of Amazon EC2 スポットインスタンスInterruption Notices」を参照してください。

詳細については、「EC2 インスタンスの再調整に関する推奨事項」および「スポットインスタンスの中断。」を参照してください。

インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する

スポットキャパシティープールは、同じインスタンスタイプ (`m5.large` など) とアベイラビリティーゾーン (us-east-1a など) を持つ、未使用の EC2 インスタンスのセットです。どのインスタンスタイプをリクエストし、どのアベイラビリティーゾーンでワークロードをデプロイするか柔軟に対応することで、スポットが必要な量のコンピューティング性能を見つけ、割り当てられる可能性が高くなります。例えば、c4、m5、m4 ファミリのラージを使用してもよいのであれば、 `c5.large` を指定する必要はないということです。

具体的なニーズに応じて、コンピューティング要件を満たすためにどのインスタンスタイプを使用できるか評価できます。ワークロードを垂直にスケーリングできる場合は、より大きいインスタンスタイプ (vCPU とメモリが多い) をリクエストに含めてください。水平にしかスケーリングできない場合は、オンデマンドの顧客からの需要が少ない、旧世代のインスタンスタイプを含めることをお勧めします。

一般的に、ワークロードごとに少なくとも 10 種類のインスタンスタイプに柔軟に対応できれば十分です。さらに、すべてのアベイラビリティーゾーンが VPC で使用するように設定され、ワークロード用に選択されていることを確認してください。

EC2 Auto Scaling グループまたは EC2 フリートを使用して総容量を管理する

スポットを使用すると、個々のインスタンスの観点からではなく、総容量 (vCPUs、メモリ、ストレージ、またはネットワークスループットなどの単位) の観点から検討することが可能になります。Auto Scaling グループと EC2 フリートは、ターゲットキャパシティの起動および維持のために使用できます。これにより、中断されたり手動で終了されたりしたリソースを置き換えるリソースを自動的に要求できます。Auto Scaling グループまたは EC2 フリートを設定する際には、アプリケーションのニーズに基づいてインスタンスタイプとターゲットキャパシティを指定するだけで済みます。詳細については、Amazon EC2 Auto Scaling ユーザーガイド の Auto Scaling グループおよびこのユーザーガイドの EC2 フリートの作成 をご参照ください。

価格と容量を最適化する配分戦略を使用する

Auto Scaling グループの配分戦略を使えば、予備容量を持つスポットキャパシティープールを手動で探す必要なく、ターゲット容量をプロビジョニングできます。最も安い価格で最も利用性の高いスポットキャパシティープールからインスタンスが自動的にプロビジョニングされる、`price-capacity-optimized` 戦略を使用することをお勧めします。また、EC2 フリートの `price-capacity-optimized` 配分戦略も活用できます。最適な容量を持つプールからスポットインスタンス容量が供給されるため、使用しているスポットインスタンスが再要求される可能性は低くなります。配分戦略の詳細については、このユーザーガイドの「ワークロードの中断コストが高い場合」および Amazon EC2 Auto Scaling ユーザーガイドの「スポットインスタンス」を参照してください。

事前の容量再調整の利用

容量の再調整は、実行中のスポットインスタンスが 2 分間のスポットインスタンス中断通知を受け取る前に、新しいスポットインスタンスでフリートを事前に拡張することにより、ワークロードの可用性を維持するのに役立ちます。容量の再調整が有効の場合、Auto Scaling または EC2 フリートは、再調整の推奨事項を受け取ったスポットインスタンスを積極的に置き換えることを試みます。これにより、中断のリスクが高くない新しいスポットインスタンスにワークロードを再調整することができます。

容量の再調整は、price-capacity-optimized 配分戦略 (最適な予備容量を見つけるように設計) と混合インスタンスポリシー (複数のアベイラビリティーゾーンで実行されている複数のインスタンスタイプにインスタンスをデプロイすることで可用性を向上させるように設計) を補完します。

詳細については、「容量の再調整」を参照してください。

統合された AWS のサービスを使用して スポットインスタンス を管理する

他の AWS のサービスは、個々のインスタンスやフリートを管理する必要なく、全体的なコンピューティングコストを削減できるよう、スポットと統合されています。該当するワークロードの場

合、Amazon EMR、Amazon Elastic Container Service、AWS Batch、Amazon Elastic Kubernetes Service、Amazon SageMaker、AWS Elastic Beanstalk、Amazon GameLift の各ソリューションを検討することをお勧めします。これらのサービスでのスポットベストプラクティスの詳細については、Amazon EC2 スポットインスタンス Workshops Website を参照してください。

使用すべき最適なスポットリクエスト方法はどれですか?

次の表を使用して、スポットインスタンスをリクエストする際に使用する API を決定します。

API	どのようなときに使うか?	ユースケース	この API を使うべきか?
CreateAutoScalingGroup	• 単一の構成または混合の構成を持つ、複数のインスタンスが必要です。 • 構成可能な API を使用してライフサイクル管理を自動化するのがよいでしょう。	必要な数のインスタンスを維持しながら、インスタンスのライフサイクルを管理する Auto Scaling グループを作成します。指定した最小値と最大限度の間の水平スケーリング (インスタンスの追加) をサポートします。	はい
CreateFleet	• 単一の構成または混合の構成を持つ、複数のインスタンスが必要です。 • インスタンスのライフサイクルを自己管理するのがよいでしょう。 •	インスタンスタイプ別、AMI 別、アベイラビリティーゾーン別、またはサブネット別で異なる、複数の起動条件を指定し、オンデマンドインスタンスとスポットインスタンス両方のフリートを 1 回のリクエストで作成します。スポットインスタンスの割り当てス	はい - オートスケーリングを必要としない場合は `instant` モード。

API	どのようなときに使うか?	ユースケース	この API を使うべきか?
	オートスケーリングが必要ない場合は、instant タイプフリートの使用をお勧めします。	トラテジーのデフォルトは、ユニットあたりの lowest-price ですが、price-capacity-optimized 、capacity-optimized または diversified に変更可能です。	
RunInstances	• 既に RunInstances API を使用してオンデマンドインスタンスを起動しているので、単一のパラメータを変更することで、スポットインスタンスの起動に変更するとよいでしょう。 • 異なるインスタンスタイプを持つ複数のインスタンスは、必要ありません。	AMI と 1 つのインスタンスタイプを使用して、指定した数のインスタンスを起動します。	いいえ - RunInstances は、1 回のリクエストで複数のインスタンスタイプを許可しないため。

API	どのようなときに使うか?	ユースケース	この API を使うべきか?
RequestSpotFleet	• RequestSpotFleet API は計画投資のないレガシー API であるため、使用はお勧めしません。 • インスタンスのライフサイクルを管理する場合は、CreateFleet API を使用します。 • インスタンスのライフサイクルを管理したくない場合は、CreateAutoScalingGroup API を使用します。	使用しません。RequestSpotFleet は、計画投資のないレガシー API です。	いいえ
RequestSpotInstances	• RequestSpotInstances API は計画投資のないレガシー API であるため、使用はお勧めしません。	使用しません。RequestSpotInstances は、計画投資のないレガシー API です。	いいえ

スポットインスタンス のしくみ

スポットインスタンスを起動するには、ユーザーがスポットインスタンスリクエストを作成します。または、Amazon EC2 が自動的にスポットインスタンスリクエストを作成することもできます。スポットインスタンスは、スポットインスタンスリクエストが受理されると起動します。

スポットインスタンスは、いくつかの異なるサービスを使用して起動できます。詳細については、「Amazon EC2 スポットインスタンスの開始方法」を参照してください。このユーザーガイドでは、EC2 を使用してスポットインスタンスを起動する方法について説明します。

- スポットインスタンスリクエストは、Amazon EC2 コンソールの インスタンス起動ウィザード または run-instances AWS CLI コマンドを使用して作成できます。詳細については、「スポットインスタンスリクエストを作成する」を参照してください。
- EC2 フリートを作成して、必要な数のスポットインスタンスを指定することができます。Amazon EC2 は、EC2 フリートで指定されているすべてのスポットインスタンスについて、ユーザーに代わってスポットインスタンスリクエストを作成します。詳細については、EC2 フリートの作成 を参照してください。
- スポットフリートリクエストを作成し、必要な数のスポットインスタンスを指定することができます。Amazon EC2 は、スポットフリートリクエストで指定されたスポットインスタンスごとに、ユーザーに代わってスポットインスタンスリクエストを作成します。詳細については、「スポットフリートリクエストを作成します。」を参照してください。

空きキャパシティがある場合、スポットインスタンスが起動します。

スポットインスタンスは、ユーザーにより停止または終了されるか、Amazon EC2 により中断 (スポットインスタンスの中断と呼ばれます) されるまで実行されます。

スポットインスタンスを使用する場合には、中断に備えておく必要があります。スポットインスタンスの需要が増加した場合や、スポットインスタンスの供給が減少した場合、Amazon EC2 がスポットインスタンスを中断する可能性があります。Amazon EC2 によりスポットインスタンスが中断される際には、スポットインスタンスの中断通知が送信されます。それによりインスタンスに対して、Amazon EC2 による中断が発生する 2 分前の警告が提供されます。スポットインスタンス の削除保護を有効にすることはできません。詳細については、スポットインスタンスの中断。 を参照してください。

ユーザーは、Amazon EBS-backed スポットインスタンスを停止、起動、再起動、または終了することができます。スポットサービスは、スポットインスタンスを中断する際に、そのインスタンスを停止、終了、または休止状態にすることができます。

コンテンツ

起動グループでの スポットインスタンス の起動

スポットインスタンスリクエストで起動グループを指定することによって、一連のスポットインスタンスのすべてが起動可能な場合にのみ、それらを起動するよう、Amazon EC2 に指示することができます。また、スポットサービスで、起動グループ内のインスタンスのいずれかを終了する必要がある場合、すべてのインスタンスを終了することが必要となります。ただし、お客様が起動グループ内の 1 つ以上のインスタンスを終了する場合、Amazon EC2 は起動グループ内のその他のインスタンスを終了しません。

このオプションは便利な場合もありますが、この制約を追加することによって、スポットインスタンスリクエストが受理される可能性は低くなるので、スポットインスタンスが終了される可能性が高まります。例えば、起動グループに複数のアベイラビリティーゾーンのインスタンスが含まれるとします。これらのアベイラビリティーゾーンのいずれかのキャパシティーが減少して使用できなくなった場合、Amazon EC2 は起動グループのすべてのインスタンスを終了します。

以前に成功したリクエストと同じ (既存の) 起動グループを指定することで、新たに正常なスポットインスタンスリクエストを作成する場合には、新しいインスタンスがこの起動グループに追加されます。したがって、この起動グループ内のインスタンスが終了されると、起動グループ内のすべてのインスタンスが終了します。これには、最初のリクエストと 2 番目リクエストによって起動されたすべてのインスタンスが含まれます。

アベイラビリティーゾーングループでの スポットインスタンス の起動

スポットインスタンスリクエストでアベイラビリティーゾーングループを指定し、そのアベイラビリティーゾーン内で一連のスポットインスタンスを起動するように Amazon EC2 に指示します。Amazon EC2 は、アベイラビリティーゾーングループのすべてのインスタンスを同時に中断する必要はありません。Amazon EC2 がアベイラビリティーゾーングループ内のいずれかのインスタンスを中断する場合、他のインスタンスはそのまま実行されます。

このオプションは便利な場合もありますが、この制約を追加することによって、スポットインスタンスリクエストが受理される可能性は低くなります。

アベイラビリティーゾーングループを指定したものの、スポットインスタンスリクエストでアベイラビリティーゾーンを指定していない場合の結果は、使用するネットワークによって異なります。

デフォルト VPC

Amazon EC2 は、指定されたサブネットのアベイラビリティーゾーンを使用します。サブネットを指定しなかった場合は、アベイラビリティーゾーンとそのデフォルトのサブネットが選択されますが、最低価格のゾーンではない可能性があります。アベイラビリティーゾーンのデフォルトのサブネットを削除した場合は、別のサブネットを指定する必要があります。

デフォルトではない VPC

Amazon EC2 は、指定されたサブネットのアベイラビリティーゾーンを使用します。

VPC での スポットインスタンス の起動

スポットインスタンス のサブネットを指定するのと同じ方法で、オンデマンドインスタンス のサブネットを指定します。

- [デフォルトの VPC] 特定の低価格のアベイラビリティーゾーンでスポットインスタンスを起動したい場合には、対応するサブネットをスポットインスタンスリクエスト内で指定する必要があります。サブネットを指定しなかった場合、Amazon EC2 によってサブネットが選択されますが、このサブネットのアベイラビリティーゾーンのスポット料金は最低ではない可能性があります。
- [デフォルト以外の VPC] スポットインスタンスのサブネットを指定する必要があります。

スポットインスタンスの料金履歴

スポットインスタンス料金は Amazon EC2 で設定され、スポットインスタンス容量に対する長期の需給傾向に基づいて緩やかに調整されます。

スポットリクエストが受理されると、オンデマンド料金を超えない現在のスポット料金で、スポットインスタンス が起動されます。インスタンスタイプ、オペレーティングシステム、アベイラビリティーゾーンでフィルタリングして、過去 90 日間のスポット料金履歴を表示できます。

現在のスポット料金を表示するには

最新のスポットインスタンス料金については、「Amazon EC2 スポットインスタンスの料金」を参照してください。

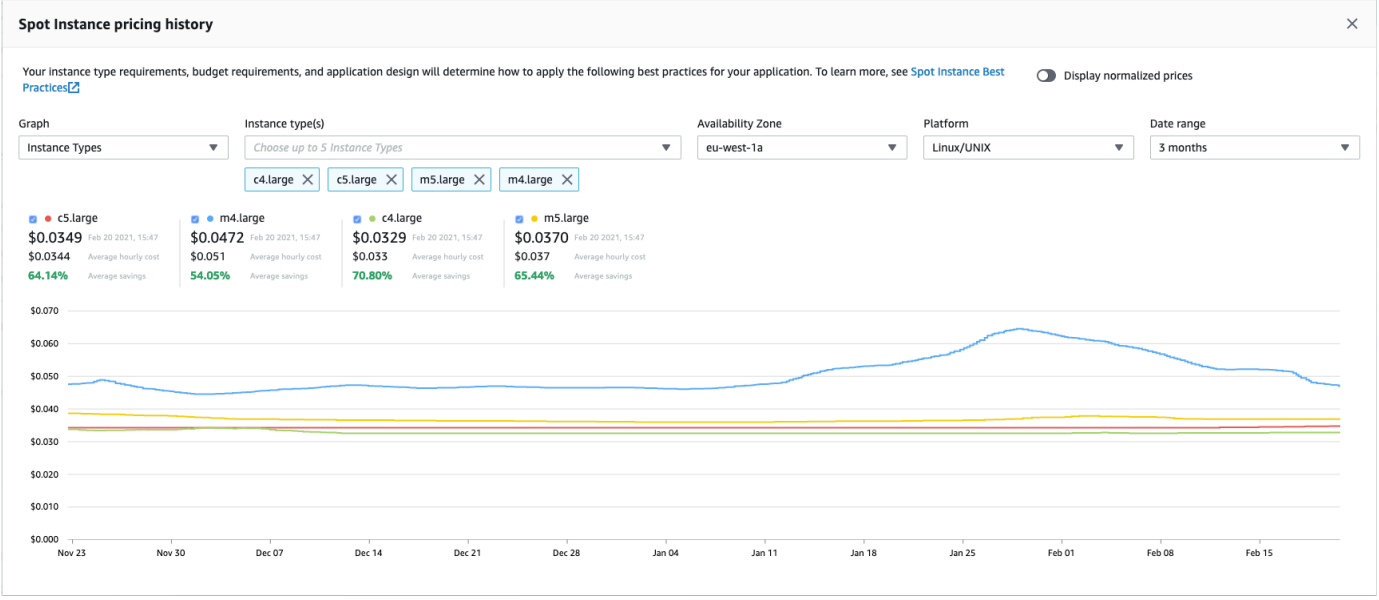
スポット料金履歴を表示するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Spot Requests] を選択します。
3. [料金設定履歴] を選択します。
4. [グラフ] で、料金履歴を[アベイラビリティーゾーン] 別に比較するか、または[インスタンスタイプ] 別に比較するかを選択します。

   - [アベイラビリティーゾーン] を選択した場合は、料金履歴を表示する [インスタンスタイプ]、オペレーティングシステム ([プラットフォーム])、および [日付範囲] を指定します。
   - [インスタンスタイプ] を選択した場合は、最大 5 つの [インスタンスタイプ] と、[アベイラビリティーゾーン]、オペレーティングシステム ([プラットフォーム])、および [日付範囲] を指定して料金履歴を表示します。

   次のスクリーンショットは、異なるインスタンスタイプでの料金比較を示しています。



5. マウスのカーソル (ポインタ) をグラフ上に移動させると、選択した日付範囲の特定の時刻の料金が表示されます。料金は、グラフの上にある情報ブロックに表示されます。一番上の行に表示される料金は、特定の日付の料金を示します。2 行目に表示される料金は、選択した日付範囲での平均料金です。
6. vCPU あたりの料金を表示するには、[正規化された料金を表示] をオンにします。インスタンスタイプの料金を表示するには、[正規化された料金を表示] をオフにします。

コマンドラインを使用してスポット料金履歴を表示するには

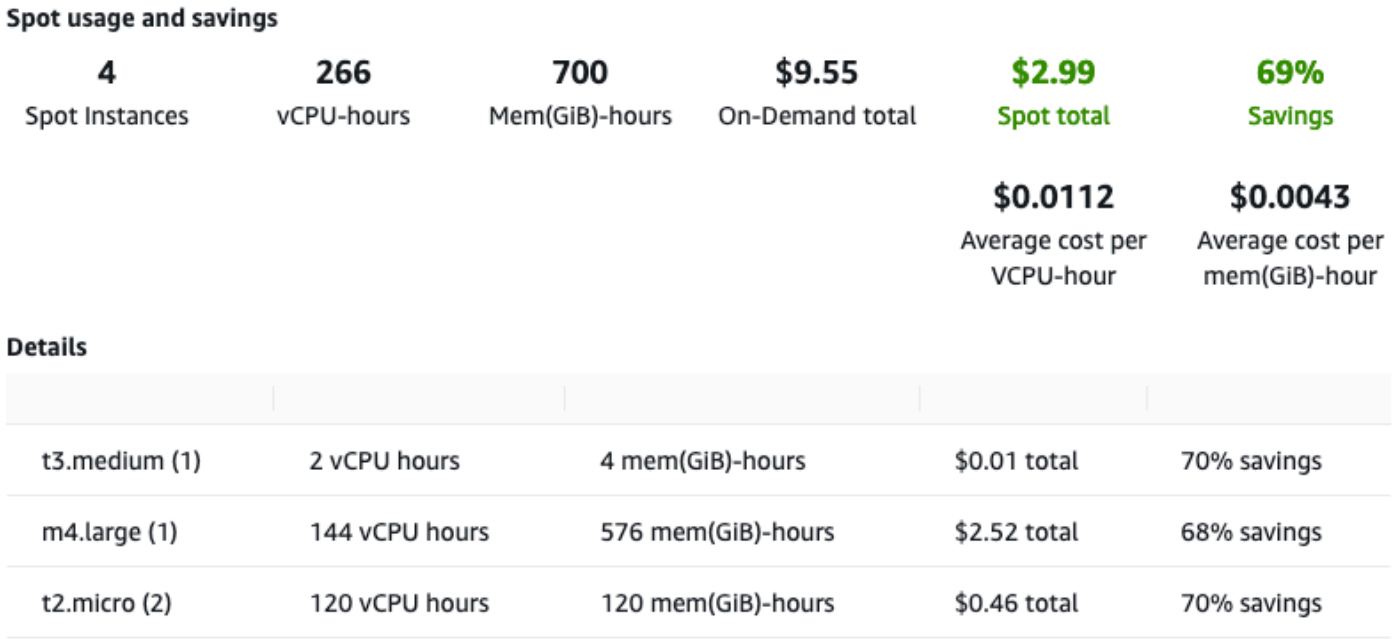
次のいずれかのコマンドを使用できます。詳細については、Amazon EC2 へのアクセス を参照してください。

- describe-spot-price-history (AWS CLI)
- Get-EC2SpotPriceHistory (AWS Tools for Windows PowerShell)

スポットインスタンス 購入による削減額

フリートあたりレベルの スポットインスタンス またはすべての実行中の スポットインスタンス に関する使用状況と削減額の情報を表示できます。フリートあたりのレベルでは、使用状況と削減額の情報にフリートが起動および終了するすべてのインスタンスが含まれます。この情報は、過去 1 時間または過去 3 日間から表示できます。

次の [削減額] セクションのスクリーンショットでは、スポットフリートでのスポットの使用状況、ならびに削減額の情報を示しています。



表示できる使用状況と削減額の情報は次のとおりです。

- スポットインスタンス – スポットフリートによって起動および終了されたスポットインスタンスの数。削減額の要約を表示した場合、その数字は実行中のすべての スポットインスタンス を表します。
- vCPU-hours – 選択した時間枠ですべての スポットインスタンス で使用される vCPU 時間数。
- Mem(GiB)-hours – 選択した時間枠ですべての スポットインスタンス で使用される GiB 時間数。

- On-Demand total – これらのインスタンスを オンデマンドインスタンス として起動した場合、選択した時間枠で支払った合計金額。
- Spot total – 選択した時間枠で支払う合計金額。
- Savings – オンデマンド価格を支払わないことで節約される割合。
- Average cost per vCPU-hour – 選択した時間枠ですべての スポットインスタンス で vCPU を使用する 1 時間あたりの平均コスト。次の式で計算されます: Average cost per vCPU-hour = Spot total / vCPU-hours
- Average cost per mem(GiB)-hour – 選択した時間枠ですべての スポットインスタンス で GiB を使用する 1 時間あたりの平均コスト。次の式で計算されます: Average cost per mem(GiB)-hour = Spot total / Mem(GiB)-hours
- 詳細 テーブル – スポットフリートを構成するさまざまなインスタンスタイプ (括弧内はインスタンスタイプあたりのインスタンス数です)。削減額の要約を表示した場合、その数字は実行中のすべての スポットインスタンス から成ります。

削減額情報は、Amazon EC2 コンソールからのみ表示できます。

スポットフリートの割引情報を表示するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストの ID を選択し、[削減額] セクションまでスクロールします。

   または、スポットフリートリクエスト ID の横にあるチェックボックスをオンにし、[削減額] タブを表示します。
4. デフォルトでは、過去 3 日間の使用状況と削減額の情報が表示されます。[last hour] または [last three days] を選択できます。1 時間未満前に起動された スポットフリート の場合は、その時間の削減見込み額が表示されます。

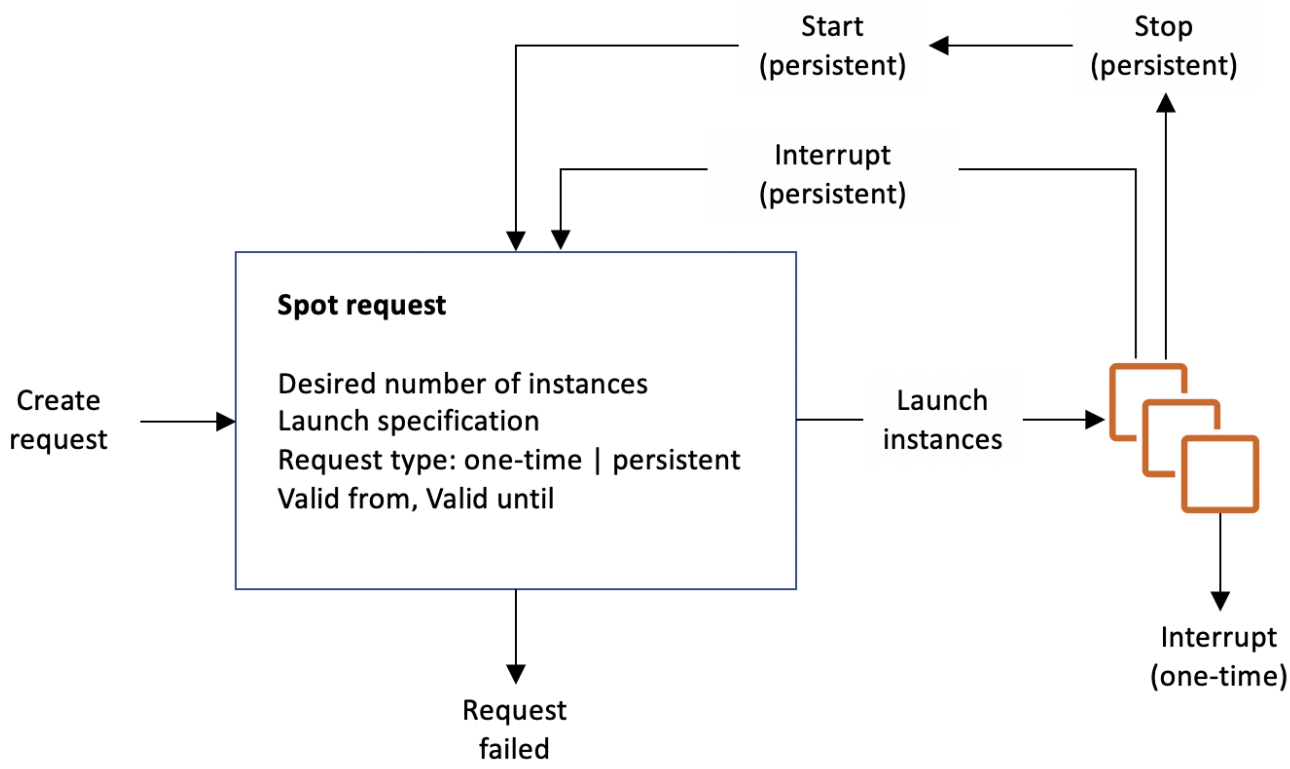
実行中のすべての スポットインスタンス の削減額情報を表示するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. [削減の概要] をクリックします。

スポットインスタンス の操作

スポットインスタンスを使用するには、希望するインスタンス数、インスタンスタイプ、アベイラビリティーゾーンを含む、スポットインスタンスリクエストを作成します。キャパシティが利用可能になると、Amazon EC2 がすぐにリクエストを受理します。それ以外の場合、Amazon EC2 は、リクエストが受理できるようになるか、お客様がリクエストをキャンセルするまで待機します。

次の図にスポットインスタンスリクエストが動作する様子を示します。Amazon EC2がスポットインスタンスを中断した場合、あるいはユーザーがスポットインスタンスを停止した場合に、リクエストが再度開かれるかどうかは、リクエストタイプ (ワンタイムまたは永続) によって決定されることに注意してください 。リクエストが永続リクエストの場合、スポットインスタンスの中断後、リクエストが再度開かれます。リクエストが永続的で、スポットインスタンスがユーザーにより停止された場合、リクエストはスポットインスタンスが開始されるまでは開かれません。



コンテンツ

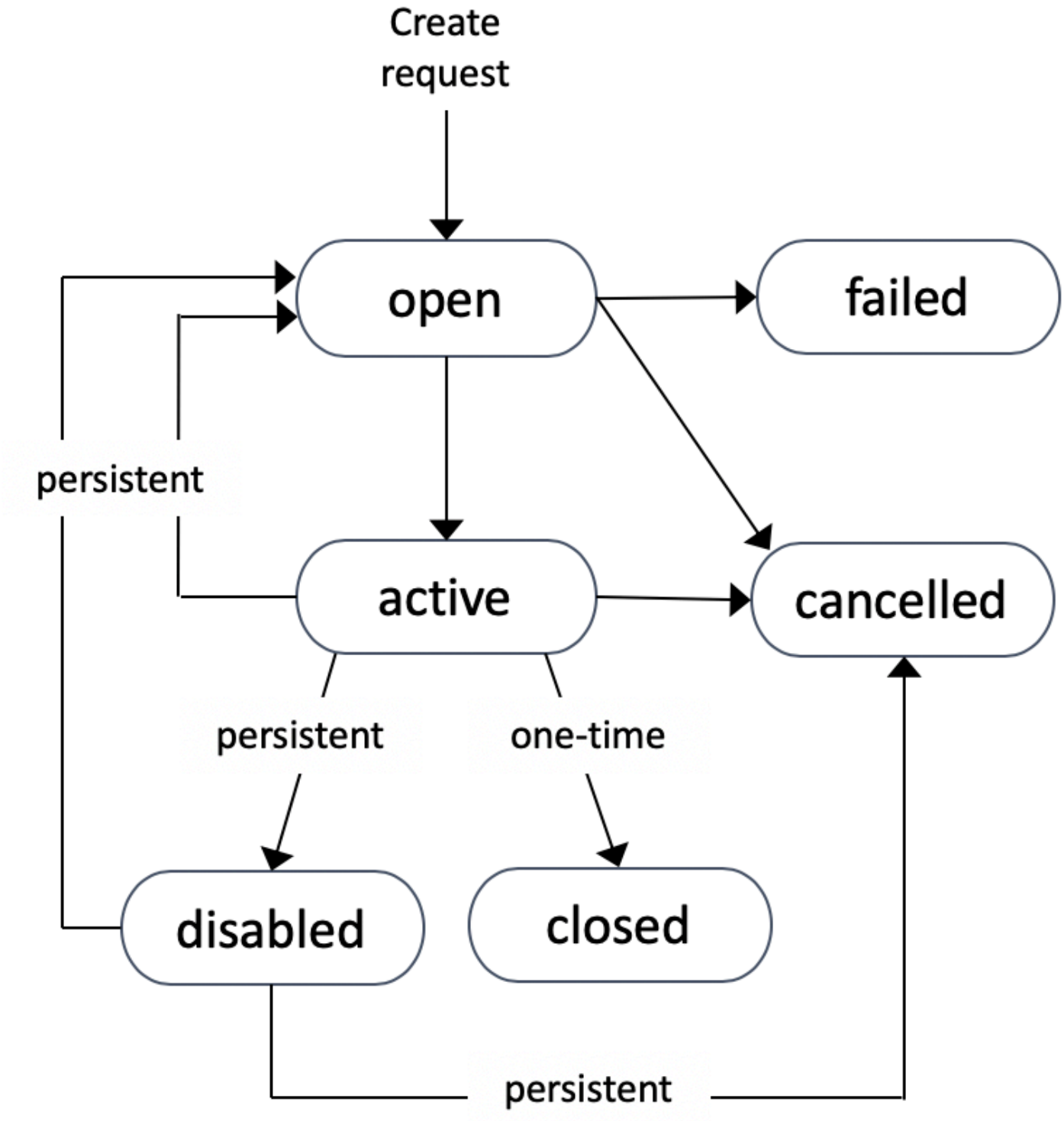
- スポットインスタンスリクエストを作成する
- 実行中の スポットインスタンス の検索
- スポットインスタンスリクエストをタグ付けする
- スポットインスタンスリクエストをキャンセルする
- スポットインスタンスを停止する
- スポットインスタンスを開始する
- スポットインスタンスを終了する
- スポットインスタンスリクエストでの起動仕様の例

スポットインスタンスリクエストの状態

スポットインスタンスリクエストは、次に示すいずれかの状態を取ります。

- open – リクエストは受理されるまで待機状態です。
- active – リクエストは受理されており、関連付けられたスポットインスタンスが存在します。
- failed – リクエストの 1 つ以上のパラメータが正しくありません。
- closed – スポットインスタンスは中断または終了されました。
- disabled – スポットインスタンスがユーザーにより停止されました。
- cancelled – このリクエストはユーザーによりキャンセルされたか、リクエストの有効期限が切れました。

次の図は、 リクエストの状態の遷移を示しています。遷移はリクエストのタイプ (ワンタイムまたは永続) によって異なります。



ワンタイムスポットインスタンスリクエストは、Amazon EC2 がスポットインスタンスを起動するか、リクエストの有効期限が切れるか、またはユーザーがリクエストをキャンセルするまでアクティ

ブ状態を維持します。利用できるキャパシティがない場合、スポットインスタンスは終了し、スポットインスタンスのリクエストは終了します。

永続スポットインスタンスリクエストは、リクエストが受理された後も、リクエストの有効期限が切れるかユーザーによりキャンセルされるまで、アクティブ状態を維持します。キャパシティを利用できない場合は、スポットインスタンスが中断されます。インスタンスが中断された後に、キャパシティが再び利用可能になると、スポットインスタンスが開始 (停止している場合)、あるいは再開 (休止状態の場合) されます。スポットインスタンスは、停止して、キャパシティを利用できるようになったとき再び開始することができます。スポットインスタンスが (停止状態にあるか実行状態にあるかに関係なく) 終了した場合には、スポットインスタンスリクエストが再び開かれ、Amazon EC2 により新しいスポットインスタンスが起動されます。詳細については、「スポットインスタンスを停止する」、「スポットインスタンスを開始する」、および「スポットインスタンスを終了する」を参照してください。

スポットインスタンスリクエストの状態と、起動済みのスポットインスタンスのステータスを追跡することができます。詳細については、「スポットリクエストステータス」を参照してください。

スポットインスタンス のテナンシーの指定

スポットインスタンスは、シングルテナントのハードウェア上で実行できます。ハードウェア専有スポットインスタンス は、他の AWS アカウントに属するインスタンスからは物理的に分離されます。詳細については、「Dedicated Instances」および「Amazon EC2 ハードウェア専有インスタンス」の製品ページを参照してください。

ハードウェア専有スポットインスタンスを使用するには、次のいずれかを実行します。

- スポットインスタンスリクエストを作成する際に、dedicated のテナンシーを指定します。詳細については、スポットインスタンスリクエストを作成する を参照してください。
- dedicated のインスタンステナンシーを持つ VPC 内で、スポットインスタンスをリクエストします。詳細については、専有インスタンスのテナンシーで VPC を作成します を参照してください。default のインスタンステナンシーを使用して、 VPC 内でインスタンスをリクエストした場合は、dedicated のテナンシーを使用しながらスポットインスタンスをリクエストすることはできません。

T インスタンスを除くすべてのインスタンスファミリーが、Dedicated スポットインスタンス をサポートしています。対象となるインスタンスファミリーにおいて、最大のインスタンスサイズまたはメタルサイズのみが、Dedicated スポットインスタンス をサポートします。

スポットインスタンスリクエスト向けのサービスにリンクされたロール

Amazon EC2 は、ユーザーに代わって AWS の他のサービスを呼び出すために必要なアクセス許可のために、サービスにリンクされたロールを使用します。サービスにリンクされたロールは、AWS のサービスに直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、AWS のサービスにアクセス許可を委任するためのセキュアな方法を提供します。これは、リンクされたサービスのみが、サービスにリンクされたロールを引き受けることができるためです。詳細については、IAM ユーザーガイドの「サービスにリンクされたロールの使用」を参照してください。

Amazon EC2 は、AWSServiceRoleForEC2Spot という、サービスにリンクされたロールを使用して、ユーザーの代わりに スポットインスタンス を起動して管理します。

AWSServiceRoleForEC2Spot によって付与されるアクセス許可

Amazon EC2 は、AWSServiceRoleForEC2Spot という、サービスにリンクされたロールを使用して、次のアクションを実行します。

- `ec2:DescribeInstances` – スポットインスタンスの記述
- `ec2:StopInstances` – スポットインスタンスの停止
- `ec2:StartInstances` – スポットインスタンスの開始

サービスにリンクされたロールの作成

ほとんどの状況では、サービスにリンクされたロールを手動で作成する必要はありません。Amazon EC2 は、ユーザーがコンソールを使用して初めてスポットインスタンスをリクエストした際に、サービスにリンクされたロール AWSServiceRoleForEC2Spot を作成します。

Amazon EC2 がこのサービスにリンクされたロールのサポートを開始した 2017 年 10 月よりも前に、ユーザーがアクティブなスポットインスタンスリクエストを行っている場合は、Amazon EC2 により AWSServiceRoleForEC2Spot ロールが AWS アカウントに作成されています。詳細については、IAM ユーザーガイドの「アカウントに新しいロールが表示される」を参照してください。

AWS CLI または API を使用してスポットインスタンスをリクエストするには、まずこのロールが存在していることを確認する必要があります。

コンソールを使用して AWSServiceRoleForEC2Spot を作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。

2. ナビゲーションペインで Roles (ロール) を選択します。
3. [ロールの作成] を選択します。
4. [Select type of trusted entity (信頼されたエンティティのタイプを選択)] ページで、[EC2]、[EC2 - Spot Instances (EC2 - スポットインスタンス)]、[Next: Permissions (次の手順: アクセス許可)] の順に選択します。
5. 次のページで、[次へ: 確認] を選択します。
6. [確認] ページで、[ロールの作成] を選択します。

AWS CLI を使用して AWSServiceRoleForEC2Spot を作成するには

次のように、create-service-linked-role コマンドを使用します。

```
aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

スポットインスタンス を使用する必要がなくなった場合は、[AWSServiceRoleForEC2Spot] ロールを削除することをお勧めします。このロールがアカウントから削除された後で、Amazon EC2 をリクエストすると、スポットインスタンス はロールを再度作成します。

暗号化された AMI および EBS スナップショット用のカスタマーマネージド型キーへのアクセス権限の付与

スポットインスタンスのために暗号化された AMI または暗号化された Amazon EBS スナップショットを指定しており、カスタマーマネージド型キーを暗号化に使用する場合は、Amazon EC2 がユーザーに代わってスポットインスタンスを起動できるようにするために、カスタマーマネージド型キーを使用する許可を AWSServiceRoleForEC2Spot ロールにより付与する必要があります。これを行うには、次の手順で示すように、カスタマーマネージド型キーに対し付与を追加する必要があります。

アクセス権限を設定するときは、付与がキーポリシーの代わりになります。詳細については、 デベロッパーガイドの「許可の使用」と「 でのキーポリシーの使用」を参照してください。https://docs.aws.amazon.com/kms/latest/developerguide/grants.htmlAWS KMSAWS Key Management Service

AWSServiceRoleForEC2Spot ロールにカスタマーマネージド型キーを使用する許可を付与するには

- create-grant コマンドを使用してカスタマーマネージド型キーに付与を追加し、プリンシパル (サービスにリンクされたロールの AWSServiceRoleForEC2Spot) を指定します。このプリンシパルには、付与が許可するオペレーションを実行するためのアクセス許可が含まれています。

カスタマーマネージド型キーは、key-id パラメータと、そのカスタマーマネージド型キーの ARN により指定します。プリンシパルを指定するには、grantee-principal パラメータとサービスにリンクされたロール AWSServiceRoleForEC2Spot の ARN を使用します。

```
aws kms create-grant \
    --region us-east-1 \
    --key-id arn:aws:kms:us-
east-1:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
spot.amazonaws.com/AWSServiceRoleForEC2Spot \
    --operations "Decrypt" "Encrypt" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "CreateGrant" "DescribeKey" "ReEncryptFrom"
 "ReEncryptTo"
```

スポットインスタンスリクエストを作成する

オンデマンドインスタンスを起動するのと同じ方法で、Amazon EC2 コンソールの インスタンス起動ウィザード または run-instances AWS CLI コマンドを使用してスポットインスタンスをリクエストできます。このメソッドは、以下の理由でのみ推奨されます。

- すでに インスタンスの起動ウィザード または run-instances コマンドを使用してオンデマンドインスタンスを起動しており、単一のパラメータを変更することでスポットインスタンスの起動に変更したいだけです。
- 異なるインスタンスタイプを持つ複数のインスタンスは、必要ありません。

複数のインスタンスタイプを指定することはできず、同じリクエストでスポットインスタンスとオンデマンドインスタンスを起動することはできないため、このメソッドは通常、スポットインスタンスの起動にはお勧めしません。複数のインスタンスタイプを持つスポットインスタンスとオンデマンドインスタンスを含むフリートの起動を含む、スポットインスタンスを起動するための推奨される方法については、「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

一度に複数のスポットインスタンスをリクエストした場合、 Amazon EC2 により個別のスポットインスタンスに対するリクエストが作成されるので、各リクエストのステータスを単独で追跡することが可能です。スポットインスタンスリクエストの追跡については、「スポットリクエストステータス」を参照してください。

New console

インスタンス起動ウィザードを使用してスポットインスタンスリクエストを作成するには

ステップ 1 ~ 9 は、オンデマンドインスタンスの起動に使用するステップと同じです。ステップ 10 で、スポットインスタンスリクエストを設定します。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、リージョンを選択します。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択します。
4. (オプション) [Name and tags] (名前とタグ) で、インスタンスに名前を付け、スポットインスタンス要求、インスタンス、ボリューム、および Elastic Graphics にタグを付けることができます。タグの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。

   a. [Name] (名前) に、インスタンスのわかりやすい名前を入力します。

      インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。名前を指定しない場合は、インスタンスをその ID で識別できます。ID は、インスタンスの起動時に自動的に生成されます。

   b. スポットインスタンスリクエスト、インスタンス、ボリューム、および Elastic Graphics にタグを付けするには、[Add additional tags] (タグを追加) を選択します。[Add tag] (タグを追加) を選択し、キーと値を入力し、タグ付けするリソースタイプを選択します。追加するタグごとに [Add tag] (タグを追加) を選択します。

5. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、インスタンスのオペレーティングシステム (OS) を選択してから、AMI を選択します。詳細については、「アプリケーションと OS イメージ (Amazon マシンイメージ)」を参照してください。
6. [Instance type] (インスタンスタイプ) で、インスタンスのハードウェア設定とサイズの要件を満たすインスタンスタイプを選択します。詳細については、「インスタンスタイプ」を参照してください。
7. [Key pair (login)] (キーペア (ログイン)) で、既存のキーペアを選択するか、[Create new key pair] (新しいキーペアを作成) を選択して新しいキーペアを作成します。詳細については、「Amazon EC2 のキーペアと Amazon EC2 インスタンス」を参照してください。

> **Important**
>
> [Proceed without key pair] (キーペアなしで進む) オプションを選択した場合 (非推奨)、ユーザーが別の方法でログインすることを許可するように設定された AMI を選択した場合でなければ、インスタンスに接続できなくなります。

8. [Network settings] (ネットワーク設定) で、デフォルト設定を使用するか、[Edit] (編集) を選択して必要に応じてネットワーク設定を構成します。

   セキュリティグループはネットワーク設定の一部を形成し、インスタンスのファイアウォールルールを定義します。このルールでは、どの着信ネットワークトラフィックをインスタンスに配信するかを指定します。

   詳細については、「ネットワーク設定」を参照してください。

9. 選択した AMI には、ルートデバイスボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。[Configure storage] (ストレージの設定) で、[Add new volume] (新しいボリュームの追加) を選択して、インスタンスに接続する追加のボリュームを指定できます。詳細については、「ストレージの設定」を参照してください。

10. [Advanced details] (高度な設定) で、スポットインスタンスリクエストを次のように設定します。

    a. [Purchasing option] (購入オプション) で、[Request Spot Instances] (スポットインスタンスのリクエスト) チェックボックスをオンにします。

    b. スポットインスタンスリクエストのデフォルト設定を維持するか、[Customize] (カスタマイズ) (右側) を選択して、スポットインスタンスリクエストのカスタム設定を指定できます。

       [Customize] (カスタマイズ) を選択すると、次のフィールドが表示されます。

       i. [Maximum price] (最大価格): スポット価格でスポットインスタンスをリクエストするか、オンデマンド価格を上限とするか、支払う金額の最大額を指定できます。

          > **Warning**
          >
          > 最大料金を指定すると、[No maximum price] (最大料金なし) を選択した場合よりもインスタンスが頻繁に中断されます。

- [No maximum price] (最大価格なし): スポットインスタンスは現在のスポット価格で起動します。価格はオンデマンド価格を超えることはありません。(推奨)
- [Set your maximum price (per instance/hour)] (最大価格を設定 (インスタンス / 時間あたり): 支払う意思のある最大金額を指定できます。
  - 現在のスポット価格よりも低い最大価格を指定すると、スポットインスタンスは起動しません。
  - 現在のスポット料金よりも高い最大料金を指定すると、スポットインスタンスが起動し、現在のスポット料金で請求されます。スポットインスタンスの実行後、スポット価格が最大価格を超えると、Amazon EC2 がスポットインスタンスを中断します。
  - 指定した上限料金にかかわらず、常に現在のスポット料金が請求されます。

  スポット料金の傾向を確認するには、「スポットインスタンスの料金履歴」を参照してください。

ii. [Request type] (リクエストタイプ): 選択したスポットインスタンスリクエストタイプによって、スポットインスタンスが中断された場合に何が発生するかが決まります。

- [One-time] (ワンタイム): Amazon EC2 は、スポットインスタンスに対して 1 回限りのリクエストを送信します。スポットインスタンスが中断された場合、リクエストは再送信されません。
- [Persistent request] (永続リクエスト): Amazon EC2 は、スポットインスタンスに対して永続リクエストを送信します。スポットインスタンスが中断された場合、要求は再送信され、中断されたスポットインスタンスを補充します。

値を指定しない場合、デフォルトは1回限りのリクエストです。

iii. [Valid to] (有効期限): 永続的な スポットインスタンスリクエストの有効期限日。

このフィールドは、1 回限りのリクエストではサポートされていません。ワンタイムリクエストは、リクエストのすべてのインスタンスが起動するか、またはユーザーがリクエストをキャンセルするまで有効です。

- [No request expiry date] (リクエストの有効期限なし): リクエストは、キャンセルされるまで有効です。

- [Set your request expiry date] (リクエストの有効期限を設定する): 永続的なリクエストは、指定した日付まで、またはキャンセルするまで有効です。

iv. [Interruption behavior] (中断動作): 選択した動作によって、スポットインスタンスが中断されたときに何が起こるかが決まります。

- 永続的なリクエストの場合、有効な値は [Stop] (停止) と [Hibernate] (休止) です。インスタンスが停止すると、EBS ボリュームストレージの料金が適用されます。

> **Note**
>
> スポットインスタンスはオンデマンドインスタンスと同じ休止機能を使用するようになりました。休止を有効にするには、ここで [休止] を選択するか、インスタンス起動ウィザードの下部に表示される [停止 – 休止動作] フィールドから [有効化] を選択します。休止の前提条件については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。

- ワンタイムリクエストの場合、[Terminate] (終了) のみが有効です。

値を指定していない場合、デフォルトは [Terminate] (終了) になり、これは、永続的なスポットインスタンスリクエストには無効です。デフォルトのままにして永続的なスポットインスタンスリクエストを起動しようとすると、エラーが発生します。

詳細については、「中断動作」を参照してください。

11. [Summary] (概要) パネルの [Number of instances] (インスタンス数) に、起動するインスタンス数を入力します。

> **Note**
>
> Amazon EC2 が、スポットインスタンスごとに個別のリクエストを作成します。

12. [Summary] (概要) パネルで、インスタンスの詳細を確認し、必要な変更を加えます。スポットインスタンスリクエストを送信した後は、リクエストのパラメータを変更することはできません。[Summary] (概要) パネルでリンクを選択すると、インスタンスの起動ウィザードのセクションに直接移動できます。詳細については、「[概要]」を参照してください。
13. インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択します。

インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

Old console

インスタンス起動ウィザードを使用してスポットインスタンスリクエストを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、リージョンを選択します。
3. Amazon EC2 コンソールダッシュボードで、[Launch Instance] を選択します。
4. [Amazon マシンイメージ (AMI)] ページで、AMI を選択します。詳細については、「ステップ 1: Amazon Machine Image (AMI) を選択する」を参照してください。
5. [Choose an Instance Type] (インスタンスタイプの選択) ページで、起動するインスタンスのハードウェア設定とサイズを選択し、[Next: Configure Instance Details] (次へ: インスタンスの詳細設定) をクリックします。詳細については、「ステップ 2: インスタンスタイプを選択する」を参照してください。
6. [インスタンスの詳細の設定] ページで、スポットインスタンスリクエストを次のように設定します。

   - [Number of instances]: 起動するインスタンスの数を入力します。

     > **Note**
     >
     > Amazon EC2 が、スポットインスタンスごとに個別のリクエストを作成します。

   - (オプション) アプリケーションで需要を処理するためにインスタンスの正しい数を確実に維持するには、[Launch into Auto Scaling Group (Auto Scaling グループに作成する)] を選択して起動設定と Auto Scaling グループを作成します。Auto Scaling によって、指定どおりにグループのインスタンス数がスケーリングされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」を参照してください。
   - [購入のオプション]: [スポットインスタンスのリクエスト] を選択してスポットインスタンスを起動します。このオプションを選択すると、次のフィールドが表示されます。
   - [現在の価格]: 選択したインスタンスタイプについて、各アベイラビリティーゾーンの現在のスポット料金が表示されます。

- (オプション) 最高料金: このフィールドは空のままにするか、支払う上限額を指定できます。

  > **Warning**
  > 上限料金を指定すると、フィールドを空にした場合よりも頻繁にインスタンスが中断されます。

  - スポット料金よりも低い最大料金を指定すると、スポットインスタンスは起動しません。
  - 現在のスポット料金よりも高い最大料金を指定すると、スポットインスタンスが起動し、現在のスポット料金で請求されます。スポットインスタンスの実行後、スポット価格が最大価格を超えると、Amazon EC2 がスポットインスタンスを中断します。
  - 指定した上限料金にかかわらず、常に現在のスポット料金が請求されます。
  - このフィールドを空のままにすると、現在のスポット料金を支払うことになります。
- [永続リクエスト]: スポットインスタンスが中断された場合に、スポットインスタンスリクエストを再送信するには、永続リクエストを選択します。
- [中断動作]: デフォルトでは、スポットサービスは中断されたスポットインスタンスを終了します。永続リクエストを選択している場合は、中断されたスポットインスタンスをスポットサービスが停止するか休止するかを指定できます。詳細については、中断動作 を参照してください。
- (オプション) リクエスト有効期間: スポットインスタンスリクエストの有効期限を指定するには、[編集] を選択します。

スポットインスタンスの設定の詳細については、「ステップ 3: インスタンスの詳細を設定する」を参照してください。

7. 選択した AMI には、ルートデバイスボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。[Add Storage] ページで、[Add New Volume] を選択することにより、インスタンスにアタッチする追加ボリュームを指定できます。詳細については、ステップ 4: ストレージを追加する を参照してください。
8. [Add Tags] ページで、キーと値の組み合わせをタグとして指定します。詳細については、ステップ 5: タグの追加 を参照してください。

9. [Configure Security Group] ページで、セキュリティグループを使用してインスタンスのファイアウォールルールを定義します このルールでは、どの着信ネットワークトラフィックをインスタンスに配信するかを指定します。他のトラフィックはすべて無視されます。(セキュリティグループの詳細については、「Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ」を参照してください)。セキュリティグループを選択または作成して、[確認して起動] をクリックします。詳細については、ステップ 6: セキュリティグループを設定する を参照してください。
10. [Review Instance Launch] ページで、インスタンスの詳細をチェックし、適切な [Edit] リンクを選択して必要な変更を加えます。準備ができたら、[Launch] を選択します。詳細については、ステップ 7: インスタンスの起動を確認し、キーペアを選択する を参照してください。
11. [Select an existing key pair or create a new key pair] ダイアログボックスで、既存のキーペアを選択するか、新しいキーペアを作成できます。例えば、[既存のキーペアの選択] をクリックし、セットアップ中に作成したキーペアを選択します。詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンス を参照してください。

    > **Important**
    >
    > [Proceed without key pair] オプションを選択した場合、ユーザーが別の方法でログインすることを許可するように設定された AMI を選択した場合でなければ、インスタンスに接続できなくなります。

12. インスタンスを起動するには、確認のチェックボックスをオンにし、続いて [Launch Instances] を選択します。

    インスタンスが起動しないか、状態が `terminated` ではなくすぐに `running` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

AWS CLI

run-instances を使用してスポットインスタンスリクエストを作成するには

run-instances コマンドを使用し、`--instance-market-options` パラメータでスポットインスタンスのオプションを指定します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type t2.micro \
```

```
    --count 5 \
    --subnet-id subnet-08fc749671b2d077c \
    --key-name MyKeyPair \
    --security-group-ids sg-0b0384b66d7d692f9 \
    --instance-market-options file://spot-options.json
```

--instance-market-options で JSON ファイルに指定するデータ構造は次のとおりです。ValidUntil、および InstanceInterruptionBehavior、を指定することもできます。データ構造でフィールドを指定しないと、デフォルト値が使用されます。

次のサンプルでは、persistent リクエストを作成します。

```
{
  "MarketType": "spot",
  "SpotOptions": {
    "SpotInstanceType": "persistent"
  }
}
```

request-spot-instances を使用してスポットインスタンスリクエストを作成するには

> **Note**
>
> request-spot-instances コマンドを使用してスポットインスタンスをリクエストすることは強くお勧めしません。これは、計画された投資がないレガシー API であるためです。詳細については、「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

ワンタイムリクエストを作成するには、request-spot-instances コマンドを使用します。

```
aws ec2 request-spot-instances \
    --instance-count 5 \
    --type "one-time" \
    --launch-specification file://specification.json
```

永続リクエストを作成するには、request-spot-instances を使用します。

```
aws ec2 request-spot-instances \
```

```
    --instance-count 5 \
    --type "persistent" \
    --launch-specification file://specification.json
```

以下のコマンドで使用する起動仕様ファイルの例については、「スポットインスタンスリクエストでの起動仕様の例」を参照してください。起動仕様ファイルをスポットリクエストコンソールからダウンロードする場合は、代わりに request-spot-fleet コマンドを使用する必要があります (スポットリクエストコンソールは、スポットフリートを使用してスポットインスタンスリクエストを指定します)。

実行中の スポットインスタンス の検索

Amazon EC2 は、キャパシティが利用可能であるときにスポットインスタンスを起動します。スポットインスタンスは中断されるか、ユーザーにより終了されるまで実行されます。

実行中の スポットインスタンス を検索するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。スポットインスタンスリクエストとスポットフリートリクエストの両方を表示できます。スポットインスタンスリクエストが受理された場合、[容量] がスポットインスタンスの ID となります。スポットフリートの場合、[容量] はリクエストされた容量のうち受理された量を示します。スポットフリートのインスタンスの ID を表示するには、拡張矢印を選択するか、フリートを選択した上で [インスタンス] を選択します。

   > **Note**
   >
   > スポットフリートによって作成されたスポットインスタンスリクエストの場合、そのリクエストが属するスポットフリートを示すシステムタグはリクエストに即座にタグ付けされません。また、一定期間、スポットフリートリクエストとは別に表示されることがあります。

   または、ナビゲーションペインで [Instances] を選択します。右上隅にある設定アイコン (⚙) を選択し、[Attribute (属性)] 列で [Instance lifecycle (インスタンスライフサイクル)] を選択します。各インスタンスの [Instance lifecycle (インスタンスライフサイクル)] は、`normal`、`spot`、または `scheduled` のいずれかです。

実行中のスポットインスタンスを検索するには (AWS CLI)

スポットインスタンス を一覧表示するには、--query オプションを指定して describe-spot-instance-requests コマンドを実行します。

```
aws ec2 describe-spot-instance-requests \
    --query "SpotInstanceRequests[*].{ID:InstanceId}"
```

出力例を次に示します。

```
[
    {
        "ID": "i-1234567890abcdef0"
    },
    {
        "ID": "i-0598c7d356eba48d7"
    }
]
```

または、--filters オプションを指定して describe-instances コマンドを実行しても、スポットインスタンス を一覧表示できます。

```
aws ec2 describe-instances \
    --filters "Name=instance-lifecycle,Values=spot"
```

単一のスポットインスタンスを表示するには、--spot-instance-request-ids オプションを指定しながら describe-spot-instance-requests コマンドを使用します。

```
aws ec2 describe-spot-instance-requests \
    --spot-instance-request-ids sir-08b93456
```

スポットインスタンスリクエストをタグ付けする

スポットインスタンスリクエストを分類および管理しやすくするため、カスタムメタデータでタグ付けすることができます。タグは、スポットインスタンスリクエストの作成時、またはその後に割り当てることができます。Amazon EC2 コンソールまたはコマンドラインツールを使用してタグを割り当てることができます。

スポットインスタンスリクエストにタグ付けを行っても、そのスポットインスタンスリクエストによって起動されたインスタンスやボリュームには、自動的なタグ付けは行われません。スポットイン

スタンスリクエストによって起動されたインスタンスやボリュームには、明示的にタグを付ける必要があります。スポットインスタンスおよびボリュームへのタグの割り当ては、起動時または起動後に行うことができます。

タグの仕組みの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。

コンテンツ

前提条件

リソースにタグ付けする許可をユーザーに付与します。IAM ポリシーとサンプルポリシーの詳細については、「例: リソースのタグ付け」を参照してください。

作成する IAM ポリシーは、スポットインスタンスリクエストの作成に使用する方法によって決まります。

- インスタンスの起動ウィザードまたは `run-instances` を使用して スポットインスタンス をリクエストする場合は、「To grant a user the permission to tag resources when using the launch instance wizard or run-instances」を参照してください。
- スポットインスタンスをリクエストするために `request-spot-instances`コマンドを使用する場合は、「To grant a user the permission to tag resources when using request-spot-instances」を参照してください。

インスタンス起動ウィザードまたは run-instances を使用する場合にリソースにタグを付けるための許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:RunInstances` アクション。これにより、インスタンスを起動するための許可がユーザーに付与されます。
- `Resource` で、`spot-instances-request` を指定します。これによりユーザーは、スポットインスタンスを要求するためのスポットインスタンスリクエストを作成できるようになります。
- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。

- Resource で、* を指定します。これにより、ユーザーはインスタンスの起動時に作成されるすべてのリソースにタグを付けることを許可されます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowLaunchInstances",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "TagSpotInstanceRequests",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

Note

RunInstances アクションを使用してスポットインスタンスリクエストを作成し、その際、リクエストにタグを付ける場合には、Amazon EC2 が RunInstances ステートメント内で spot-instances-request リソースをどのように評価するのかについて、注意を払う必要があります。

spot-instances-request リソースは、IAM ポリシーで次のように評価されます。

- スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 は RunInstances ステートメント内の `spot-instances-request` リソースを評価しません。
- スポットインスタンスリクエストの作成時にタグを付けると、 RunInstances ステートメント内の `spot-instances-request` リソースが、Amazon EC2 により評価されます。

したがって、`spot-instances-request` リソースの場合、次のルールが IAM ポリシーに適用されます。

- RunInstances を使用してスポットインスタンスリクエストを作成し、その際リクエストにタグを付けない場合は、`spot-instances-request` リソースを明示的に許可しなくても、その呼び出しは成功します。
- RunInstances を使用してスポットインスタンスリクエストを作成する際に、そのリクエストにタグを付ける場合には、RunInstances の許可ステートメントに `spot-instances-request` リソースを含める必要があります。これがない場合は呼び出しが失敗します。
- RunInstances を使用してスポットインスタンスリクエストを作成する際に、そのリクエストにタグを付ける場合は、許可ステートメント CreateTags で `spot-instances-request` リソースを指定するか、そこに `*` ワイルドカードを含める必要があります。これがない場合は呼び出しが失敗します。

IAM ポリシー (スポットインスタンスリクエストでサポートされていないポリシーを含む) の例については、「スポットインスタンス の操作」を参照してください。

request-spot-instances を使用する場合でリソースにタグを付けるための許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:RequestSpotInstances` アクション。これにより、スポットインスタンスリクエストを作成する許可がユーザーに付与されます。
- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。
- `Resource` で、`spot-instances-request` を指定します。これにより、ユーザーはスポットインスタンスリクエストにのみタグを付けることが許可されます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagSpotInstanceRequest",
            "Effect": "Allow",
            "Action": [
                "ec2:RequestSpotInstances",
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:spot-instances-request/*"
}
```

新しいスポットインスタンスリクエストにタグを付ける

コンソールを使用して新しいスポットインスタンスリクエストにタグ付けするには

1. 「スポットインスタンスリクエストを作成する」の手順に従います。
2. タグを追加するには、[タグの追加] ページで [タグの追加] をクリックし、タグのキーと値を入力します。追加するタグごとに [別のタグを追加] をクリックします。

   1 つのタグを、スポットインスタンスリクエスト、スポットインスタンス、およびボリュームに対し同時にタグ付けすることができます。3 つすべてにタグを付けるには、[インスタンス]、[ボリューム]、[スポットインスタンスリクエスト] をそれぞれ選択します。1 つまたは 2 つにのみタグを付けるには、タグを付けるリソースを選択し、他のリソースを選択していないことを確認します。
3. 必須フィールドにすべて入力してスポットインスタンスリクエストを作成した後、[起動] を選択します。詳細については、スポットインスタンスリクエストを作成する を参照してください。

AWS CLI を使用して新しいスポットインスタンスリクエストにタグ付けするには

スポットインスタンスリクエストの作成時にタグ付けするには、以下のようにスポットインスタンスリクエストを設定します。

- `--tag-specification` パラメータを使用してスポットインスタンスリクエストのタグを指定します。
- `ResourceType` で、`spot-instances-request` を指定します。別の値を指定すると、スポットインスタンスリクエストは失敗します。
- `Tags` で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

以下の例では、スポットインスタンスリクエストには 2 つのタグ (Environment キーと Production 値、ならびに Cost-Center キーと 123 値) が付けられています。

```
aws ec2 request-spot-instances \
    --instance-count 5 \
    --type "one-time" \
    --launch-specification file://specification.json \
    --tag-specification 'ResourceType=spot-instances-
request,Tags=[{Key=Environment,Value=Production},{Key=Cost-Center,Value=123}]'
```

既存のスポットインスタンスリクエストにタグ付けをする

コンソールを使用して既存のスポットインスタンスリクエストにタグ付けするには

スポットインスタンスリクエストの作成後に、コンソールを使用してそのリクエストにタグを追加できます。

Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

1. ナビゲーションペインで、[Spot Requests] を選択します。
2. スポットインスタンスリクエストを選択します。
3. [Tags (タグ)] タブを選択してから、[タグの作成] を選択します。

コンソールを使用して既存のスポットインスタンスにタグを付けるには

スポットインスタンスリクエストによってスポットインスタンスを起動した後で、コンソールを使用して、そのインスタンスにタグを追加できます。詳細については、個々のリソースのタグの追加および削除 を参照してください。

AWS CLI を使用して、既存のスポットインスタンスリクエストまたはスポットインスタンスにタグを付けるには

create-tags コマンドを使用して、既存のリソースにタグを付けます。次の例では、既存のスポットインスタンスリクエストとスポットインスタンスに、purpose キーと test 値のタグを付けています。

```
aws ec2 create-tags \
    --resources sir-08b93456 i-1234567890abcdef0 \
    --tags Key=purpose,Value=test
```

スポットインスタンスリクエストのタグを表示する

コンソールを使用してスポットインスタンスリクエストのタグを表示するには

Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

1. ナビゲーションペインで、[Spot Requests] を選択します。
2. スポットインスタンスリクエストを選択してから、[タグ] タブを選択します。

スポットインスタンスリクエストのタグを詳細表示するには

describe-tags コマンドを使用して、指定したリソースのタグを表示します。以下の例では、指定した リクエストのタグの情報を取得します。

```
aws ec2 describe-tags \
    --filters "Name=resource-id,Values=sir-11112222-3333-4444-5555-66666EXAMPLE"
```

```
{
    "Tags": [
        {
            "Key": "Environment",
            "ResourceId": "sir-11112222-3333-4444-5555-66666EXAMPLE",
            "ResourceType": "spot-instances-request",
            "Value": "Production"
        },
        {
            "Key": "Another key",
            "ResourceId": "sir-11112222-3333-4444-5555-66666EXAMPLE",
            "ResourceType": "spot-instances-request",
            "Value": "Another value"
        }
    ]
}
```

スポットインスタンスリクエストを詳細表示すると、そのスポットインスタンスリクエストのタグを確認することができます。

describe-spot-instance-requests コマンドを使用して、指定したスポットインスタンスリクエストの設定を表示します。この内容には、リクエストに指定されたタグがすべて含まれます。

```
aws ec2 describe-spot-instance-requests \
```

```
    --spot-instance-request-ids sir-11112222-3333-4444-5555-66666EXAMPLE
```


```
{
    "SpotInstanceRequests": [
        {
            "CreateTime": "2020-06-24T14:22:11+00:00",
            "InstanceId": "i-1234567890EXAMPLE",
            "LaunchSpecification": {
                "SecurityGroups": [
                    {
                        "GroupName": "launch-wizard-6",
                        "GroupId": "sg-1234567890EXAMPLE"
                    }
                ],
                "BlockDeviceMappings": [
                    {
                        "DeviceName": "/dev/xvda",
                        "Ebs": {
                            "DeleteOnTermination": true,
                            "VolumeSize": 8,
                            "VolumeType": "gp2"
                        }
                    }
                ],
                "ImageId": "ami-1234567890EXAMPLE",
                "InstanceType": "t2.micro",
                "KeyName": "my-key-pair",
                "NetworkInterfaces": [
                    {
                        "DeleteOnTermination": true,
                        "DeviceIndex": 0,
                        "SubnetId": "subnet-11122233"
                    }
                ],
                "Placement": {
                    "AvailabilityZone": "eu-west-1c",
                    "Tenancy": "default"
                },
                "Monitoring": {
                    "Enabled": false
                }
            },
            "LaunchedAvailabilityZone": "eu-west-1c",
```

```
            "ProductDescription": "Linux/UNIX",
            "SpotInstanceRequestId": "sir-1234567890EXAMPLE",
            "SpotPrice": "0.012600",
            "State": "active",
            "Status": {
                "Code": "fulfilled",
                "Message": "Your spot request is fulfilled.",
                "UpdateTime": "2020-06-25T18:30:21+00:00"
            },
            "Tags": [
                {
                    "Key": "Environment",
                    "Value": "Production"
                },
                {
                    "Key": "Another key",
                    "Value": "Another value"
                }
            ],
            "Type": "one-time",
            "InstanceInterruptionBehavior": "terminate"
        }
    ]
}
```

スポットインスタンスリクエストをキャンセルする

スポットインスタンスリクエストが不要になった場合には、それをキャンセルすることができます。open、active、または disabled のスポットインスタンスリクエストのみキャンセルできます。

- スポットインスタンスリクエストがまだ受理されておらず、インスタンスが起動されていない段階では、そのリクエストは open 状態にあります。
- スポットインスタンスリクエストが受理され、スポットインスタンスの起動が完了している場合、そのスポットインスタンスリクエストは active 状態になります。
- ユーザーがスポットインスタンスを停止した場合、スポットインスタンスリクエストは disabled 状態になります。

スポットインスタンスリクエストの状態が active で、関連付けられたスポットインスタンスが実行されている場合、そのリクエストをキャンセルしても、関連するインスタンスは終了しません。ス

ポットインスタンスの終了の詳細については、「スポットインスタンスを終了する」を参照してください。

スポットインスタンスリクエストをキャンセルするには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[スポットリクエスト] をクリックした後、スポットインスタンスリクエストを選択します。
3. [アクション]、[リクエストのキャンセル] の順にクリックします。
4. (オプション) 関連付けられたスポットインスタンスを使い終わったら、スポットインスタンスを終了できます。[スポットリクエストのキャンセル] ダイアログボックスで、[インスタンスの終了]、[確認] の順にクリックします。

スポットインスタンスリクエストをキャンセルするには (AWS CLI)

- 指定したスポットインスタンスリクエストをキャンセルするには、cancel-spot-instance-requests コマンドを使用します。

```
aws ec2 cancel-spot-instance-requests --spot-instance-request-ids sir-08b93456
```

スポットインスタンスを停止する

今すぐスポットインスタンスは必要ないが、Amazon EBS ボリューム内に保持されているデータを失わずに後で再起動する必要がある場合は、それらを停止できます。スポットインスタンスを停止する手順は、オンデマンドインスタンスを停止する手順と似ています。

> **Note**
>
> スポットインスタンスが停止している間、そのインスタンスの属性の一部は変更可能ですが、インスタンスタイプを変更することはできません。
> 停止しているスポットインスタンスの使用料またはデータ転送料は課金されませんが、Amazon EBS ボリュームのストレージに対しては課金されます。

制限事項

- スポットインスタンスを停止できるのは、そのインスタンスが、persistent なスポットインスタンスリクエストから起動された場合だけです。
- 関連するスポットインスタンスリクエストがキャンセルされている場合は、スポットインスタンスを停止することはできません。スポットインスタンスリクエストがキャンセルされた場合は、スポットインスタンスを終了することのみ可能です。
- フリート、起動グループ、またはアベイラビリティーゾーングループの一部であるスポットインスタンスは停止できません。

Console

スポットインスタンスを停止するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、スポットインスタンスを選択します。
3. [Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。
4. 確認を求められたら、[Stop] を選択します。

AWS CLI

スポットインスタンスを停止するには (AWS CLI)

- 1 つまたは複数のスポットインスタンスを手動で停止するには、stop-instances コマンドを使用します。

```
aws ec2 stop-instances --instance-ids i-1234567890abcdef0
```

スポットインスタンスを開始する

以前に停止したスポットインスタンスは開始することができます。スポットインスタンスを開始する手順は、オンデマンドインスタンスを開始する手順と似ています。

前提条件

スポットインスタンスは、次の場合にのみ開始できます。

- スポットインスタンスを手動で停止している。
- スポットインスタンスが EBS-backed インスタンスである。
- スポットインスタンスに使用可能な容量がある。
- スポット料金が上限価格より低くなっている。

制限事項

- フリート、起動グループ、またはアベイラビリティーゾーングループの一部であるスポットインスタンスを開始することはできません。

Console

スポットインスタンスを開始するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、スポットインスタンスを選択します。
3. [Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択します。

AWS CLI

スポットインスタンスを開始するには (AWS CLI)

- start-instances コマンドを使用して、1 つまたは複数のスポットインスタンスを手動で起動します。

```
aws ec2 start-instances --instance-ids i-1234567890abcdef0
```

スポットインスタンスを終了する

永続スポットインスタンスリクエストによって起動された実行中または停止中のスポットインスタンスを終了すると、そのスポットインスタンスリクエストの状態は open に遷移し、新たなスポットインスタンスを起動できるようになります。新しいスポットインスタンスが起動されないようにするには、まずスポットインスタンスリクエストをキャンセルする必要があります。

スポットインスタンスを実行させている `active` スポットインスタンスリクエストをキャンセルしても、実行中のスポットインスタンスは自動的に終了されません。スポットインスタンスは手動で終了する必要があります。

停止中のスポットインスタンスを持つ `disabled` スポットインスタンスリクエストをキャンセルした場合、この停止中のスポットインスタンスは、Amazon EC2 スポットサービスによって自動的に終了されます。スポットインスタンスリクエストをキャンセルしてから、スポットサービスがスポットインスタンスを終了するまでの間に、短い遅延が生じることがあります。

スポットインスタンスリクエストのキャンセルの詳細については、「スポットインスタンスリクエストをキャンセルする」を参照してください。

Console

コンソールを使用してスポットインスタンスを手動で終了するには

1. インスタンスを終了する前に、終了時に Amazon EBS ボリュームが削除されることと、必要なデータすべてをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認して、データが失われないことを確認します。
2. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
3. ナビゲーションペインで、[インスタンス] を選択します。
4. そのインスタンスがスポットインスタンスであることを確認するには、[インスタンスライフサイクル] 列に表示されている[スポット] のチェックをオンにします。
5. インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了] の順に選択します。
6. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択します。

AWS CLI

AWS CLI を使用してスポットインスタンスを手動で終了するには

- スポットインスタンス を手動で終了するには、terminate-instances コマンドを使用します。

```
aws ec2 terminate-instances --instance-
ids i-1234567890abcdef0 i-0598c7d356eba48d7
```

スポットインスタンスリクエストでの起動仕様の例

以下に、スポットインスタンスリクエストを作成するための request-spot-instances コマンドで使用できる起動設定の例を示します。詳細については、「スポットインスタンスリクエストを作成する」を参照してください。

> **Important**
>
> request-spot-instances コマンドを使用してスポットインスタンスをリクエストすることは強くお勧めしません。これは、計画された投資がないレガシー API であるためです。詳細については、「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

例

- 例 1: スポットインスタンス の起動
- 例 2: 指定したアベイラビリティーゾーンで スポットインスタンス を起動する
- 例 3: 指定したサブネットで スポットインスタンス を起動する
- 例 4: ハードウェア専有スポットインスタンスを起動する

例 1: スポットインスタンス の起動

以下の例にはアベイラビリティーゾーンやサブネットは指定していません。Amazon EC2 によって自動的にアベイラビリティーゾーンが選択されます。Amazon EC2 は、選択したアベイラビリティーゾーンのデフォルトのサブネットでインスタンスを起動します。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "m5.medium",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

例 2: 指定したアベイラビリティーゾーンで スポットインスタンス を起動する

以下の例には、アベイラビリティーゾーンが含まれています。Amazon EC2 は、指定したアベイラビリティーゾーンのデフォルトのサブネットでインスタンスを起動します。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "m5.medium",
  "Placement": {
    "AvailabilityZone": "us-west-2a"
  },
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

例 3: 指定したサブネットで スポットインスタンス を起動する

次の例には、サブネットが含まれます。Amazon EC2 は、指定されたサブネットでインスタンスを起動します。デフォルト以外の VPC である場合、インスタンスにはデフォルトでパブリック IPv4 アドレスは割り当てられません。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "m5.medium",
  "SubnetId": "subnet-1a2b3c4d",
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

デフォルト以外の VPC である場合、インスタンスにパブリック IPv4 アドレスを割り当てるには、以下の例に示しているように `AssociatePublicIpAddress` フィールドを指定します。ネットワークインターフェイスの指定時には、上記のコードブロックに示している `SubnetId` および `SecurityGroupIds` フィールドではなく、ネットワークインターフェイスを使用して、サブネット ID およびセキュリティグループ ID を含める必要があります。

```
{
```

```
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "InstanceType": "m5.medium",
  "NetworkInterfaces": [
    {
      "DeviceIndex": 0,
      "SubnetId": "subnet-1a2b3c4d5e6f7g8h9",
      "Groups": [ "sg-1a2b3c4d5e6f7g8h9" ],
      "AssociatePublicIpAddress": true
    }
  ],
  "IamInstanceProfile": {
      "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
  }
}
```

例 4: ハードウェア専有スポットインスタンスを起動する

次の例では、dedicated のテナンシーを使用するスポットインスタンスをリクエストしています。ハードウェア専用スポットインスタンスは、VPC 内で起動される必要があります。

```
{
  "ImageId": "ami-0abcdef1234567890",
  "KeyName": "my-key-pair",
  "SecurityGroupIds": [ "sg-1a2b3c4d5e6f7g8h9" ],
  "InstanceType": "c5.8xlarge",
  "SubnetId": "subnet-1a2b3c4d5e6f7g8h9",
  "Placement": {
    "Tenancy": "dedicated"
  }
}
```

スポットリクエストステータス

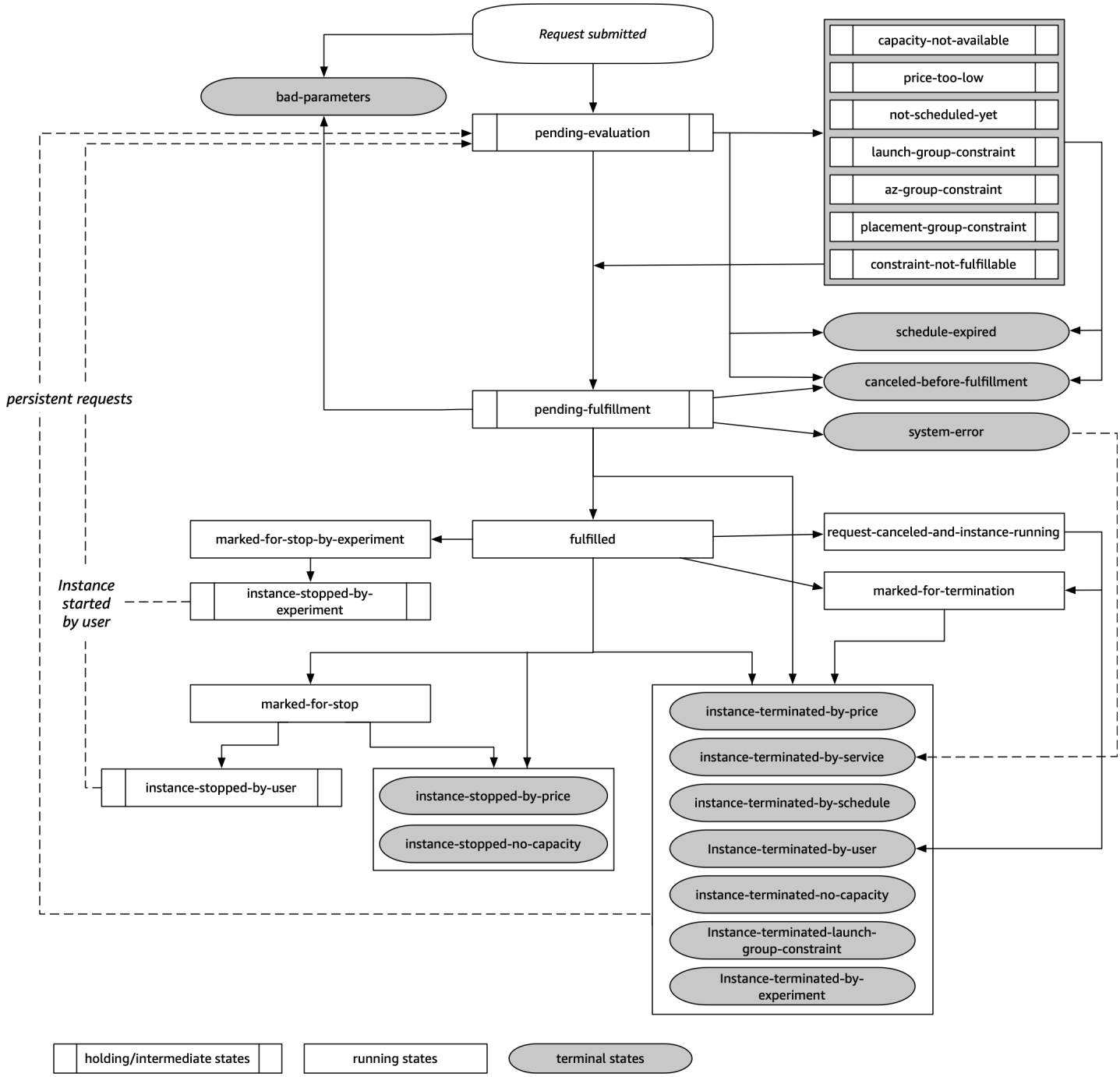
スポットインスタンスリクエストを追跡し、スポットインスタンスの使用を計画するには、Amazon EC2 によって提供されるリクエストステータスを使用します。例えば、リクエストステータスによって、スポットリクエストがまだ受理されていない理由や、スポットリクエストの受理を妨げている制約の一覧を確認できます。

このプロセスの各ステップ (スポットリクエストのライフサイクルとも呼ばれる) では、特定のイベントによって後続のリクエスト状態が決まります。

コンテンツ

スポットリクエストのライフサイクル

次の図は、申請から終了まで、スポットリクエストがライフサイクル全体を通してたどり得る経路を示しています。各ステップはノードとして表現され、各ノードのステータスコードはスポットリクエストおよびスポットインスタンスのステータスを示します。



評価保留

スポットインスタンスリクエストを作成すると、リクエストパラメータのいずれかが無効な (`bad-parameters`) 場合 を除き、そのリクエストは `pending-evaluation` 状態になります。

ステータスコード	リクエストの状態	インスタンスの状態
pending-evaluation	open	該当しない
bad-parameters	closed	該当しない

保持

1 つ以上のリクエストによる制約が有効であるが、まだ満足することができない場合や、容量が十分ではない場合、リクエストは制約が満たされるまで待機する保持状態になります。リクエストのオプションは、リクエストが受理される可能性に影響します。例えば、キャパシティがない場合、キャパシティが利用可能になるまでリクエストは保留状態になります。アベイラビリティーゾーングループを指定する場合、アベイラビリティーゾーンの制約が満たされるまで、リクエストは保持状態になります。

いずれかのアベイラビリティーゾーンが停止した場合、他のアベイラビリティーゾーンのスポットインスタンスリクエストで使用可能な予備の EC2 容量が、影響を受ける可能性があります。

ステータスコード	リクエストの状態	インスタンスの状態
capacity-not-available	open	該当しない
price-too-low	open	該当しない
not-scheduled-yet	open	該当しない
launch-group-constraint	open	該当しない
az-group-constraint	open	該当しない
placement-group-constraint	open	該当しない

ステータスコード	リクエストの状態	インスタンスの状態
constraint-not-fulfillable	open	該当しない

評価保留/受理終了

特定の期間のみ有効なスポットインスタンスリクエストを作成し、そのリクエストが受理保留段階に到達する前に有効期間が経過した場合、 そのリクエストは terminal 状態になることがあります。これは、お客様がリクエストをキャンセルした場合、またはシステムエラーが発生した場合にも発生する場合があります。

ステータスコード	リクエストの状態	インスタンスの状態
schedule-expired	cancelled	該当しない
canceled-before-fulfillment [1]	cancelled	該当しない
bad-parameters	failed	該当しない
system-error	closed	該当しない

[1] リクエストをキャンセルする場合。

受理保留

指定した制約条件 (もしあれば) が満たされると、スポットリクエストは pending-fulfillment ステータスになります。

この時点で、Amazon EC2 は要求されたインスタンスを提供するよう準備します。この段階でプロセスが停止した場合は、スポットインスタンスが起動される前に、ユーザーがリクエストをキャンセルしたことが原因である可能性があります。または、予期しないシステムエラーが発生したことが原因である可能性もあります。

ステータスコード	リクエストの状態	インスタンスの状態
pending-fulfillment	open	該当しない

受理済み

スポットインスタンスの仕様がすべて満たされると、スポットリクエストが受理されます。Amazon EC2 がスポットインスタンスを起動しますが、これには数分かかる場合があります。中断状態にあるスポットインスタンスが、休止または停止された場合、リクエストが再度受理できるようになるかキャンセルされるまで同じ状態が維持されます。

ステータスコード	リクエストの状態	インスタンスの状態
fulfilled	active	pending → running
fulfilled	active	stopped → running

スポットインスタンスを停止すると、そのインスタンスを再起動できるようになるか、リクエストがキャンセルされるまで、スポットリクエストは marked-for-stop または instance-stopped-by-user 状態になります。

ステータスコード	リクエストの状態	インスタンスの状態
marked-for-stop	active	stopping
instance-stopped-by-user [1]	disabled または cancelled [2]	stopped

[1] スポットインスタンスを停止するか、そのインスタンスからシャットダウンコマンドを実行すると、インスタンスは instance-stopped-by-user 状態になります。インスタンスを停止した後は、インスタンスを再起動できるようになります。再起動時に、スポットインスタンスリクエストは pending-evaluation 状態に戻り、制約事項が満たされると Amazon EC2 によって新しいスポットインスタンスが起動されます。

[2] スポットインスタンスを停止して、リクエストをキャンセルしていない場合には、スポットリクエストの状態は disabled になります。スポットインスタンスが停止しており、リクエストの有効期限が切れている場合、リクエストの状態は cancelled になります。

受理済み終了

インスタンスタイプで使用可能なキャパシティがあり、お客様がインスタンスを終了しない限り、スポットインスタンスの実行は続行されます。Amazon EC2 でスポットインスタンスを終了する必要がある場合、スポットリクエストは終了状態になります。リクエストは、お客様がスポットリクエストをキャンセルした場合や、スポットインスタンス を終了した場合も、終了状態になります。

ステータスコード	リクエストの状態	インスタンスの状態
request-canceled-and-instance-running	cancelled	running
marked-for-stop	active	running
marked-for-termination	active	running
instance-stopped-by-price	disabled	stopped
instance-stopped-by-user	disabled	stopped
instance-stopped-no-capacity	disabled	stopped
instance-terminated-by-price	closed (ワンタイム)、open (永続)	terminated
instance-terminated-by-schedule	closed	terminated
instance-terminated-by-service	cancelled	terminated

ステータスコード	リクエストの状態	インスタンスの状態
instance-terminated-by-user	closed または cancelled [1]	terminated
instance-terminated-no-capacity	closed (ワンタイム)、open (永続)	running †
instance-terminated-no-capacity	closed (ワンタイム)、open (永続)	terminated
instance-terminated-launch-group-constraint	closed (ワンタイム)、open (永続)	terminated

[1] インスタンスを終了したが、リクエストをキャンセルしていない場合、リクエストの状態は closed になります。インスタンスを終了し、リクエストをキャンセルする場合、リクエストの状態は cancelled になります。スポットリクエストをキャンセルする前にスポットインスタンスを終了した場合でも、そのスポットインスタンスの終了が Amazon EC2 によって検出されるまでに遅延が生じることがあります。この場合、リクエストの状態は closed または cancelled となります。

† Amazon EC2 が容量を戻す必要がある場合にスポットインスタンスに割り込み、かつ、インスタンスが割り込み時に終了するように設定されている場合、ステータスはすぐに instance-terminated-no-capacity に設定されます (marked-for-termination には設定されていません)。ただし、インスタンスは、インスタンスがスポットインスタンスの中断通知を受信した 2 分間を反映して、2 分間 running 状態のままになります。2 分後、インスタンスの状態は terminated に設定されます。

永続リクエスト

スポットリクエストが永続リクエストであり、関連するスポットインスタンスが (ユーザーまたは Amazon EC2 によって) 終了された場合には、そのリクエストは pending-evaluation 状態に戻るので、制約事項が満たされた後に Amazon EC2 は新しいスポットインスタンスを起動できます。

リクエストステータス情報の取得

AWS Management Consoleまたはコマンドラインツールを使用して、リクエストステータス情報を取得できます。

リクエストステータス情報を取得するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択し、スポットリクエストを選択します。
3. ステータスを確認するには、[説明] タブの [ステータス] フィールドをチェックします。

コマンドラインを使用してリクエストステータス情報を取得する

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセス を参照してください。

- describe-spot-instance-requests (AWS CLI)
- Get-EC2SpotInstanceRequest (AWS Tools for Windows PowerShell)

スポットリクエストコード

スポットリクエストステータス情報は、ステータスコード、更新時刻、およびステータスメッセージで構成されます。同時に、リクエスト入札ステータス情報は、スポットリクエストの処理を決定する場合にも役に立ちます。

スポットリクエストステータスコードは、次のとおりです。

`az-group-constraint`

Amazon EC2 は、同じアベイラビリティーゾーンでお客様が要求したインスタンスをすべて起動できるとは限りません。

`bad-parameters`

スポットリクエストの 1 つ以上のパラメータが有効ではありません (例えば、指定した AMI が存在していません)。ステータスメッセージによって、どのパラメータが無効かを確認できます。

`canceled-before-fulfillment`

スポットリクエストが受理される前にユーザーがスポットリクエストをキャンセルしました。

`capacity-not-available`

要求したインスタンスに使用できる十分な容量が存在しません。

`constraint-not-fulfillable`

1 つ以上の制約条件が有効ではないため、スポットリクエストを受理できません (例えば、アベイラビリティーゾーンが存在していません)。ステータスメッセージによって、どの制約条件が無効かを確認できます。

`fulfilled`

スポットリクエストは `active` で、Amazon EC2 は スポットインスタンス を起動しています。

`instance-stopped-by-price`

スポット料金が上限価格を超えたため、インスタンスは停止しました。

`instance-stopped-by-user`

ユーザーがインスタンスを停止したか、インスタンスからシャットダウンコマンドを実行したために、インスタンスが停止されました。

`instance-stopped-no-capacity`

EC2 の容量管理のニーズにより、インスタンスが停止されました。

`instance-terminated-by-price`

スポット料金が上限価格を超えたため、インスタンスは削除されました。リクエストが永続入札の場合、プロセスが再開され、リクエストが評価保留となります。

`instance-terminated-by-schedule`

スポットインスタンスは、スケジュールされた期間の最後に終了されました。

`instance-terminated-by-service`

インスタンスが停止状態から削除されました。

`instance-terminated-by-user`、または `spot-instance-terminated-by-user`

受理済みのスポットインスタンスを終了させたので、(永続リクエストでない限り) リクエストは `closed` 状態になり 、インスタンスは `terminated` 状態になます。

`instance-terminated-launch-group-constraint`

起動グループ内のインスタンスの 1 つ以上が終了したため、起動グループの制約条件が満たされなくなりました。

`instance-terminated-no-capacity`

標準的な容量管理プロセスにより、インスタンスは終了しました。

`launch-group-constraint`

Amazon EC2 は、お客様が同時に要求したインスタンスをすべて起動できるわけではありません。同じ起動グループ内のインスタンスはすべて、同時に起動されて同時に終了します。

`limit-exceeded`

EBS ボリューム数または合計ボリュームストレージの上限を超えました。これらの制限および増加を要求する方法の詳細については、「Amazon Web Services 全般のリファレンス」の「Amazon EBS の制限」を参照してください。

`marked-for-stop`

スポットインスタンスは停止中としてマーキングされます。

`marked-for-termination`

スポットインスタンスに終了のためのマークが付けられています。

`not-scheduled-yet`

スポットリクエストは、スケジュール設定された日付になるまで評価されません。

`pending-evaluation`

スポットインスタンスリクエストの作成後、システムがリクエストのパラメータを評価中は、そのリクエストは `pending-evaluation` 状態となります。

`pending-fulfillment`

Amazon EC2 は スポットインスタンス をプロビジョニングしようとしています。

`placement-group-constraint`

現時点でスポットインスタンスをプレイスメントグループに追加できないため、まだスポットリクエストを受理することができません。

`price-too-low`

上限料金がスポット料金を下回っているため、リクエストを受理できません。この場合、インスタンスは起動されず、リクエストは `open` のままになります。

`request-canceled-and-instance-running`

スポットインスタンス がまだ実行されている間に、リクエストをキャンセルしました。リクエストは `cancelled` ですが、インスタンスは `running` のままです。

`schedule-expired`

スポットリクエストは、指定された日付までに受理されなかったため、有効期限切れとなりました。

`system-error`

予期しないシステムエラーが発生しました。これが反復性の問題である場合は、AWS Support にお問い合わせください。

EC2 スポットインスタンスリクエストのフルフィルメントイベント

スポットインスタンスリクエストが受理されると、Amazon EC2 は EC2 スポットインスタンスリクエストのフルフィルメントイベントを Amazon EventBridge に送信します。Lambda 関数の呼び出しや Amazon SNS トピックへの通知など、このイベントが発生するたびにアクションを実行するルールを作成できます。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "01234567-1234-0123-1234-012345678901",
    "detail-type": "EC2 Spot Instance Request Fulfillment",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-1234567890abcdef0"],
    "detail": {
        "spot-instance-request-id": "sir-1a2b3c4d",
        "instance-id": "i-1234567890abcdef0"
    }
}
```

詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

EC2 インスタンスの再調整に関する推奨事項

EC2 インスタンスの再調整に関するレコメンデーションは、スポットインスタンスで中断のリスクが高まった場合に通知するためのシグナルです。シグナルは、スポットインスタンス中断 2 分前の通知よりも早く到着するので、スポットインスタンスを事前に管理するタイミングを知ることができ

ます。ワークロードを、中断のリスクが高くない新規または既存の スポットインスタンス に再調整することができます。

2 分間のスポットインスタンス中断通知の前に、Amazon EC2 が再調整に関する推奨事項シグナルを送信することは必ずしも可能ではありません。したがって、再調整に関する推奨事項シグナルは、2 分間の中断通知とともに到着する可能性があります。

再調整に関する推奨事項は、EventBridge イベントとして、およびスポットインスタンス上のインスタンスメタデータの項目として使用できます。イベントは、ベストエフォートベースで発生します。

> **Note**
>
> 再調整に関する推奨事項は、2020 年 11 月 5 日 00:00 UTC 以降に起動される スポットインスタンス のみをサポートしています。

トピック

- 実行できるアクションの再調整
- 再調整に関する推奨事項シグナルのモニタリング
- 再調整に関する推奨事項シグナルを使用するサービス

実行できるアクションの再調整

以下に、実行可能な再調整アクションをいくつか挙げます。

適切なシャットダウン

スポットインスタンスの再調整に関する推奨事項シグナルを受信すると、インスタンスのシャットダウン手順を開始できます。この際、インスタンスを停止する前に各プロセスの完了を確認する必要が生じることがあります。例えば、システムログまたはアプリケーションログを Amazon Simple Storage Service (Amazon S3) にアップロードしたり、Amazon SQS ワーカーをシャットダウンしたり、ドメインネームシステム (DNS) からの登録解除を完了したりできます。また、作業内容を外部ストレージに保存し、後で再開することもできます。

新しい作業がスケジュールされるのを防止

スポットインスタンスの再調整に関する推奨事項シグナルを受信すると、スケジュールされた作業が完了するまでインスタンスの使用を継続しながら、新しい作業がインスタンス上でスケジュールされることを回避できます。

新しい代替インスタンスを積極的に起動

Auto Scaling グループ、EC2 フリート、または スポットフリート を設定しておくと、再調整に関する推奨事項シグナルが送信された場合に、代替のスポットインスタンスを自動的に起動させることができます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「キャパシティーの再調整を使用して Amazon EC2 スポットの中断に対処する」、およびこのユーザーガイドの「EC2 フリートの 容量の再調整」、「スポットフリートの 容量の再調整」を参照してください。

再調整に関する推奨事項シグナルのモニタリング

再調整に関する推奨事項シグナルをモニタリングして、それが発されたときに、前のセクションで指定したアクションを実行できるようにすることができます。再調整に関する推奨事項シグナルは、Amazon EventBridge (旧称 Amazon CloudWatch Events) に送信されるイベントとして、およびスポットインスタンス上のインスタンスメタデータとしての使用が可能です。

再調整に関する推奨事項シグナルをモニタリングする:

- Amazon EventBridge の使用
- インスタンスメタデータの使用

Amazon EventBridge の使用

再調整に関する推奨事項シグナルがスポットインスタンスに対して発信されると、そのシグナルのイベントが Amazon EventBridge に送信されます。EventBridge がルールで定義されているパターンに一致するイベントパターンを検出すると、EventBridge はルールで指定されているターゲットを呼び出します。

次に、再調整に関する推奨事項シグナルのイベント例を示します。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "EC2 Instance Rebalance Recommendation",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2:123456789012:instance/i-1234567890abcdef0"],
    "detail": {
```

```
        "instance-id": "i-1234567890abcdef0"
    }
}
```

次のフィールドは、ルールで定義されているイベントパターンになります。

`"detail-type": "EC2 Instance Rebalance Recommendation"`

イベントが再調整に関する推奨事項イベントであることを特定します

`"source": "aws.ec2"`

イベントが Amazon EC2 からのものであることを特定します

EventBridge ルールを作成します

EventBridge ルールを作成し、イベントパターンがルールに一致したときに実行するアクションを自動化できます。

次の例では、Amazon EC2 が再調整に関する推奨事項シグナルを発するたびに、E メール、テキストメッセージ、またはモバイルプッシュ通知を送信する EventBridge ルールを作成します。シグナルは `EC2 Instance Rebalance Recommendation` イベントとして発され、ルールによって定義されたアクションがトリガーされます。

EventBridge ルールを作成する前に、E メール、テキストメッセージ、またはモバイルプッシュ通知用の Amazon SNS トピックを作成する必要があります。

再調整に関する推奨事項イベントの EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. [Create rule] を選択します。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択します。アカウント内の AWS のサービスで生成されたイベントは、常に、そのアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。

d. [Next] を選択します。

4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。

a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択します。

b. この例では [Event pattern] (イベントパターン) で、`EC2 Instance Rebalance Recommendation` イベントと一致するように次のイベントパターンを指定してから、[Save] (保存) を選択します。

```
{
 "source": ["aws.ec2"],
 "detail-type": ["EC2 Instance Rebalance Recommendation"]
}
```

イベントパターンを追加するには、以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには、以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択します。

B. [イベントパターンフォーム] では、AWS[サービス] を選択します。

C. [AWS Service] (サービス) で、[EC2 Spot Fleet] (EC2 スポットフリート) を選択します。

D. [Event type] (イベントタイプ) で、[EC2 Instance Rebalance Recommendation] (EC2 インスタンスのリバランスに関するレコメンデーション) を選択します。

E. テンプレートをカスタマイズするには、[Edit pattern] (パターンを編集) を選択した上で、この例のイベントパターンに合わせた変更を行います。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択します。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択します。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. ターゲットタイプ] では、AWSサービス] を選択します。
b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択します。
c. [Topic (トピック)] で、既存のトピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。
d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。
e. [Next] を選択します。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。
7. [Review and create] (確認して作成) で、以下の操作を行います。

a. ルールの詳細を確認し、必要な場合は変更を行います。
b. [ルールを作成] を選択します。

詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge ルール」と「Amazon EventBridge イベントパターン」を参照してください。

インスタンスメタデータの使用

インスタンスメタデータカテゴリ `events/recommendations/rebalance` は、スポットインスタンスに対して再調整に関する推奨事項シグナルが発されたおおよその時間 (UTC) を示します。

再調整に関する推奨事項シグナルを 5 秒ごとに確認し、再調整に関する推奨事項に基づいて行動する機会を見逃さないようにすることをお勧めします。

スポットインスタンスが受信した、再調整に関する推奨事項では、そのシグナルが発行された時刻がインスタンスメタデータに含まれています。信号が発された時間は以下のように取得できます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
```

```
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/events/recommendations/rebalance
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/events/recommendations/rebalance
```

次に、再調整に関する推奨事項シグナルがスポットインスタンスに対して送信された時刻を、UTC で表示する出力例を示します。

```
{"noticeTime": "2020-10-27T08:22:00Z"}
```

インスタンスに対してシグナルが送信されていない場合、`events/recommendations/rebalance` は存在せず、取得しようとすると HTTP 404 エラーが表示されます。

再調整に関する推奨事項シグナルを使用するサービス

Amazon EC2 Auto Scaling、EC2 フリート、およびスポットフリートにおいて再調整に関する推奨事項シグナルを使用することで、ワークロードの可用性を維持しやすくなります。実行中のインスタンスがスポットインスタンス中断 2 分前の通知を受信していない段階から、事前にフリートを新しいスポットインスタンスで強化することができます。このようなサービスでは、スポットインスタンスの可用性に影響する変更を事前にモニタリングし、対応させることができます。詳細については、次を参照してください:

- 「Amazon EC2 Auto Scaling ユーザーガイド」の「キャパシティーの再調整を使用して Amazon EC2 スポットの中断に対処する」
- このユーザーガイドの EC2 フリート トピックの 容量の再調整
- このユーザーガイドのスポットフリート トピックの 容量の再調整

スポットインスタンスの中断。

Amazon EC2 で再びキャパシティーが必要になったときは、予備の EC2 キャパシティーで スポットインスタンス を起動してキャパシティーを戻すのと引き換えに、大幅な割引を受けることができます。Amazon EC2 がスポットインスタンスを再要求した場合、このイベントをスポットインスタンスの中断と呼びます。

Amazon EC2 によりスポットインスタンスが中断される際には、スポットリクエストの作成時に指定した内容に応じて、インスタンスが終了、停止、または休止されます。

スポットインスタンス に対する需要は刻一刻と大幅に変化する可能性があります。また、スポットインスタンス の可用性も利用可能な未使用の EC2 インスタンスの数に応じて大きく変化する可能性があります。スポットインスタンスが中断される可能性は常に存在します。

EC2 フリートまたはスポットフリートで指定されたオンデマンドインスタンスは中断できません。

コンテンツ

中断の理由

Amazon EC2 が スポットインスタンス を中断する場合、次のような理由が考えられます。

容量

Amazon EC2 は、必要なときにスポットインスタンスを中断できます。EC2 は、主に容量を再利用するためにインスタンスを再利用しますが、ホストのメンテナンスやハードウェアの使用停止などの他の理由でも発生する可能性があります。

価格

スポット料金が、上限料金を超えています。

上限料金はスポットリクエストで指定できます。ただし、上限料金を指定すると、指定しなかった場合に比べ、インスタンスの中断が増えます。

制約

スポットリクエストに、起動グループやアベイラビリティーゾーングループなどの制約を含んでいて、その制約条件が満たされなくなった場合には、スポットインスタンスはグループとして終了されます。

インスタンスタイプの過去の中断率は、Spot Instance Advisorで見ることができます。

中断動作

Amazon EC2 がスポットインスタンスを中断させた際に、次のいずれかが実行されるように指定できます。

- スポットインスタンス の中断の停止
- 中断した スポットインスタンス の休止
- 中断したスポットインスタンスの終了 (これがデフォルトの動作です)

中断動作の指定

中断動作は、スポットリクエストの作成時に指定できます。中断動作を指定しない場合は、デフォルトで、中断されたときに Amazon EC2 がスポットインスタンスを終了させます。

中断動作の指定方法は、スポットインスタンス をリクエストする方法によって異なります。

- インスタンス起動ウィザードを使ってスポットインスタンスをリクエストする場合、次の方法で中断動作を指定できます。インスタンス起動ウィザードで [高度な詳細] を展開し、[スポットインスタンス のリクエスト] チェックボックスをオンにします。[Customize] (カスタマイズ) を選択します。[中断動作] から中断動作を選択します。中断動作が休止状態の場合は、[停止 - 休止動作] で [有効化] を選択することもできます。
- run-instances CLI を使用してスポットインスタンスをリクエストする場合、次の方法で中断動作を指定できます。リクエスト設定 (`--instance-market-options`) で `InstanceInterruptionBehavior` に中断動作を指定します。中断動作が `hibernate` である場合、代わりに `--hibernation-options Configured=true` パラメータを使用して休止状態を有効にできます。
- 起動テンプレートで スポットインスタンス を設定する場合、中断動作を指定するには、起動テンプレートで [高度な詳細] を展開し、[スポットインスタンス のリクエスト] チェックボックスをオンにします。[カスタマイズ] をクリックし、[中断動作] から中断動作を選択します。

- スポットコンソールを使用して スポットインスタンス をリクエストする場合、中断動作を指定するには、[ターゲット容量を維持する] チェックボックスをオンにし、[中断動作] から中断動作を選択します。
- create-fleet CLI の使用時にリクエスト設定でスポットインスタンスを設定する場合、中断動作を指定するには、`InstanceInterruptionBehavior` を使用します。
- request-spot-fleet CLI の使用時にリクエスト設定でスポットインスタンスを設定する場合、中断動作を指定するには、`InstanceInterruptionBehavior` を使用します。
- request-spot-instances CLI を使用して スポットインスタンス を設定する場合、中断動作を指定するには、`--instance-interruption-behavior` を使用します。

> **Note**
>
> request-spot-fleet コマンド、および request-spot-instances コマンドを使ってスポットインスタンスをリクエストすることは推奨されていません。これらはレガシー API で、投資が予定されていないためです。詳細については、「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

スポットインスタンス の中断の停止

中断時に Amazon EC2 がスポットインスタンスを停止するように指定できます。詳細については、中断動作の指定 を参照してください。

考慮事項

- 中断され停止したスポットインスタンスを再起動できるのは Amazon EC2 だけです。
- `persistent` スポットインスタンスリクエストで起動されたスポットインスタンスについては、停止したインスタンスと同じアベイラビリティーゾーンと同じインスタンスタイプで利用可能な容量がある場合に、Amazon EC2 がその停止したインスタンスを再起動することができます (同じ起動仕様を使用する必要があります)。
- タイプ `maintain` の EC2 フリートまたはスポットフリートによって起動されたスポットインスタンスについては、スポットインスタンスが中断されると、Amazon EC2 はターゲット容量を維持するために代替インスタンスを起動します。Amazon EC2 では、指定された配分戦略 (`lowestPrice`、`diversified`、`InstancePoolsToUseCount`) に基づき、最適なスポット容量プールを検索します。先に停止したインスタンスは、プールの優先順位に影響しません。後に、

配分戦略で、以前に停止したインスタンスを含むプールに導かれる場合、Amazon EC2 は、ターゲット容量に合うように停止したインスタンスを再起動します。

例えば、配分戦略が lowestPrice のスポットフリートが再起動の対象になります。初回起動時、c3.large プールは、起動仕様の lowestPrice 条件を満たしています。後に、c3.large インスタンスが中断されると、Amazon EC2 はそのインスタンスを停止し、lowestPrice 戦略に合う別のプールから容量を補充します。今回の場合、プールは c4.large プールになり、Amazon EC2 はターゲット容量を満たすように c4.large インスタンスを起動します。次のタイミングでも同様に、スポットフリートが c5.large プールに移動されます。これらの各遷移では、Amazon EC2 は、以前に停止したインスタンスを含むプールを優先せずに、指定された配分戦略を純粋に優先します。lowestPrice 戦略では、以前に停止したインスタンスを含むプールに戻る場合があります。例えば、インスタンスが c5.large プールで中断され、lowestPrice 戦略によって c3.large または c4.large プールに戻った場合、以前に停止したインスタンスはターゲット容量を満たすために再起動されます。

- スポットインスタンスが停止している間、そのインスタンスの属性の一部は変更可能ですが、インスタンスタイプを変更することはできません。デタッチまたは削除された EBS ボリュームは、スポットインスタンスが開始した際にアタッチされません。ユーザーがルートボリュームをデタッチし、Amazon EC2 がスポットインスタンスの開始を試みると、そのインスタンスは開始に失敗し、停止したインスタンスが Amazon EC2 により終了されます。
- ユーザーは、停止中のスポットインスタンスを終了できます。
- ユーザーがスポットインスタンスリクエスト、EC2 フリート、またはスポットフリートをキャンセルすると、Amazon EC2 は、それらに関連付けられていて停止中のスポットインスタンスを終了します。
- 中断されたスポットインスタンスの停止中は、維持されている EBS ボリュームに対してのみ課金されます。EC2 フリートおよびスポットフリートでは、停止中のインスタンスの数が多い場合、アカウント内の EBS ボリューム数の上限を超えることがあります。スポットインスタンスが中断されたときの料金の詳細については、「中断された スポットインスタンス の請求」を参照してください。
- インスタンスを停止することの影響について理解しておいてください。インスタンスが停止している場合に何が行われるかの詳細については、「再起動、停止、休止、削除の違い」を参照してください。

前提条件

中断されたスポットインスタンスを停止するには、以下の前提条件を設定する必要があります。

スポットリクエストタイプ

スポットインスタンスリクエストのタイプ – `persistent` である必要があります。スポットインスタンスリクエストで起動グループを指定することはできません。

EC2 フリートまたはスポットフリートのリクエストのタイプ – `maintain` である必要があります。

ルートボリュームタイプ

インスタンスストアボリュームではなく EBS ボリュームにする必要があります。

中断した スポットインスタンス の休止

中断時に Amazon EC2 がスポットインスタンスを休止するように指定できます。詳細については、「Amazon EC2 インスタンスの休止」を参照してください。

Amazon EC2 では、オンデマンドインスタンスで現在利用できるのと同じ休止状態をスポットインスタンスでも提供するようになりました。サポートの範囲が広がり、スポットインスタンスの休止では新たに以下がサポートされています。

- さらに多くの AMI をサポート
- さらに多くのインスタンスファミリーをサポート
- ユーザー起動の休止

中断したスポットインスタンスの終了

Amazon EC2 によりスポットインスタンスが中断される場合は、停止や休止などの別の中断動作を指定しない限り、デフォルトでインスタンスが終了します。詳細については、中断動作の指定 を参照してください。

中断に対する準備

スポットインスタンス に対する需要は刻一刻と大幅に変化する可能性があります。また、スポットインスタンス の可用性も利用可能な未使用の EC2 インスタンスの数に応じて大きく変化する可能性があります。スポットインスタンスが中断される可能性は常に存在します。したがって、アプリケーションでスポットインスタンスの中断に対して準備する必要があります。

スポットインスタンスの中断に備えて、以下のベストプラクティスに従うことをお勧めします。

- Auto Scaling グループを使用してスポットリクエストを作成します。スポットインスタンスが中断された場合、Auto Scaling グループは代替インスタンスを自動的に起動します。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
- 必要なソフトウェア設定を含む Amazon Machine Image (AMI) を使用することにより、リクエストが受理されたらすぐにインスタンスを実行できるように、準備が完了していることを確認します。また、ユーザーデータを使用して起動時にコマンドを実行することもできます。
- インスタンスストアボリュームのデータは、インスタンスの停止または終了に伴って失われます。インスタンスストアボリュームの重要なデータを、Amazon S3、Amazon EBS、または Amazon DynamoDB などのより永続的なストレージにバックアップします。
- スポットインスタンスの終了の影響を受けない場所に、定期的に重要なデータを保存します。例えば、Amazon S3、Amazon EBS、または DynamoDB を使用できます。
- 作業を頻繁に保存できるように、作業を (Grid、Hadoop、キューベースのアーキテクチャを使用して) 細かいタスクに分割するか、チェックポイントを使用します。
- Amazon EC2 は、スポットインスタンスでの中断のリスクが高まった場合に、再調整に関する推奨事項シグナルをそのインスタンスに対し送信します。再調整に関する推奨事項シグナルを利用すると、スポットインスタンス中断 2 分前の通知を待つことなく、スポットインスタンスの中断を事前に管理することができます。詳細については、EC2 インスタンスの再調整に関する推奨事項を参照してください。
- スポットインスタンスのステータスをモニタリングするには、スポットインスタンス中断 2 分前の通知を使用します。詳細については、スポットインスタンスの中断通知 を参照してください。
- この警告をできるだけ早く表示するよう努めていますが、警告が表示される前にスポットインスタンスが中断されることもあり得ます。再調整に関する推奨事項シグナルと中断通知をモニタリングしている場合でも、予期しないインスタンスの終了をアプリケーションが適切に処理できることを確認します。オンデマンドインスタンスを使用してアプリケーションを実行し、オンデマンドインスタンスを自分で終了することでこれを確認できます。
- AWS Fault Injection Service で制御された故障注入実験を実行し、スポットインスタンスが中断されたときのアプリケーションの応答をテストします。詳細については、「AWS Fault Injection Service ユーザーガイド」の「チュートリアル: AWS FIS を使用してスポットインスタンスの中断をテストする」を参照してください。

スポットインスタンスを中断させる

Amazon EC2 コンソールでスポットインスタンスリクエストまたはスポットフリートリクエストを選択してスポットインスタンスの中断を実行すると、スポットインスタンス上のアプリケーション

での中断に関する処理をテストできます。スポットインスタンスの中断を開始すると、最初にそのスポットインスタンスの中断が 2 分後に行われることが Amazon EC2 から通知され、2 分経過後にインスタンスが中断されます。

スポットインスタンスの中断を処理するための、基盤となるサービスは AWS Fault Injection Service (AWS FIS) です。AWS FIS の詳細については、「AWS Fault Injection Service」を参照してください。

> **Note**
>
> 中断動作は、`terminate`、`stop`、および `hibernate` です。中断動作に対し `hibernate` を設定してスポットインスタンスの中断を開始すると、休止プロセスがすぐに開始されます。

スポットインスタンスの中断は、すべての AWS リージョン (アジアパシフィック (ジャカルタ)、アジアパシフィック (大阪)、中国 (北京)、中国 (寧夏)、および中東 (UAE) を除く) で利用することができます。

トピック

- スポットインスタンスを中断させる
- スポットインスタンスの中断を検証する
- クォータ

スポットインスタンスを中断させる

EC2 コンソールを使用すると、スポットインスタンスの中断をすばやく開始できます。スポットインスタンスリクエストを選択すると、1 つのスポットインスタンスの中断を開始できます。スポットフリートリクエストを選択すると、複数のスポットインスタンスの中断を一度に開始できます。

より高度な実験によりスポットインスタンスの中断をテストするには、AWS FIS コンソールで独自の実験を作成します。

EC2 コンソールを使用してスポットインスタンスリクエストで 1 つのスポットインスタンスの中断を開始するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択します。

3. スポットインスタンスリクエストを選択した後、[Actions] (アクション)、[Initiate interruption] (中断を開始) の順に選択します。複数のスポットインスタンスリクエストを選択して中断を開始することはできません。
4. [Initiate Spot Instance interruption] (スポットインスタンスの中断を開始する) ダイアログボックスにある、[Service access] (サービスアクセス) で、デフォルトのロールか、既存のロールを選択します。既存のロールを選択するには、[既存のサービスロールを使用] を選択した後、[IAM ロール] で使用するロールを選択します。
5. スポットインスタンスの中断を開始する準備ができたら、[Initiate interruption] (中断を開始) を選択します。

EC2 コンソールを使用して、スポットフリートリクエストで 1 つまたは複数のスポットインスタンスの中断を開始するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択します。
3. スポットフリートリクエストを選択した後、[アクション]、[中断を開始] の順に選択します。複数のスポットフリートリクエストを選択して中断を開始することはできません。
4. [スポットインスタンスの数を指定] ダイアログボックスの [中断するインスタンスの数] に、中断するスポットインスタンスの数を入力し、[確認] を選択します。

> **Note**
>
> この数は、フリート内のスポットインスタンス数や、1 回の実験で AWS FIS が中断できるスポットインスタンス数のクォータを超えることはできません。

5. [Initiate Spot Instance interruption] (スポットインスタンスの中断を開始する) ダイアログボックスにある、[Service access] (サービスアクセス) で、デフォルトのロールか、既存のロールを選択します。既存のロールを選択するには、[既存のサービスロールを使用] を選択した後、[IAM ロール] で使用するロールを選択します。
6. スポットインスタンスの中断を開始する準備ができたら、[Initiate interruption] (中断を開始) を選択します。

AWS FIS コンソールを使用して、スポットインスタンスの中断をテストするためのより高度な実験を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択します。
3. [Actions] (アクション)、[Create advanced experiments] (高度な実験を作成) の順に選択します。

   AWS FIS コンソールが開きます。詳細については、「AWS Fault Injection Service ユーザーガイド」の「チュートリアル: AWS FIS を使用してスポットインスタンスの中断をテストする」を参照してください。

スポットインスタンスの中断を検証する

中断を開始すると、以下のことが発生します。

- 対象のスポットインスタンスに対し、インスタンスの再調整に関する推奨事項が送信されます。
- AWS FIS がインスタンスを中断する 2 分前に、スポットインスタンスの中断の通知が発行されます。
- 2 分経過後に、スポットインスタンスが中断されます。
- AWS FIS によって停止されたスポットインスタンスは、ユーザーにより再起動されるまで停止状態を維持します。

中断を開始した後に、インスタンスが中断されていることを検証するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[スポットリクエスト] を開いてから、別のブラウザタブまたはウィンドウで[インスタンス] を開きます。
3. [スポットリクエスト] で、スポットインスタンスリクエストまたはスポットフリートリクエストを選択します。初期ステータスは、`fulfilled` です。インスタンスが中断されると、その中断の動作に応じて、以下のようにステータスが変化します。
   - `terminate` – ステータスが `instance-terminated-by-experiment` に変わります。
   - `stop` – ステータスが `marked-for-stop-by-experiment` に変わり、その後 `instance-stopped-by-experiment` に変わります。
4. インスタンスで、スポットインスタンスを選択します。初期ステータスは、`Running` です。ユーザーがスポットインスタンスの中断通知を受け取り、2 分が経過すると、その中断の動作に応じて、以下のようにステータスが変化します。
   - `stop` – ステータスが `Stopping` に変わり、その後 `Stopped` に変わります。
   - `terminate` – ステータスが `Shutting-down` に変わり、その後 `Terminated` に変わります。

クォータ

AWS アカウント には、1 回の実験で AWS FIS が中断できるスポットインスタンス数について、以下のデフォルトクォータがあります。

名前	デフォルト	引き上げ可能	説明
aws:ec2:send-spot-instance-interruptions のターゲット SpotInstances	サポートされている各リージョン： 5	はい	タグを使用してターゲットを特定するときに、aws:ec2:send-spot-instance-interruptions がターゲットにできるスポットインスタンスの実験ごとの最大数。

クォータは、引き上げをリクエストすることができます。詳細については、「Service Quotas ユーザーガイド」の「クォータ引き上げのリクエスト」を参照してください。

AWS FIS のクォータをすべて表示するには、Service Quotas コンソールを開きます。ナビゲーションペインで、[AWS services] を選択し、AWS Fault Injection Service を選択します。「AWS Fault Injection Service ユーザーガイド」で「AWS Fault Injection Service のクォータ」をすべて確認することもできます。

スポットインスタンスの中断通知

スポットインスタンスの中断通知は、Amazon EC2 がスポットインスタンスを停止または終了する 2 分前に発行される警告です。中断動作として休止状態を指定した場合は、中断通知が表示されますが、休止状態プロセスはすぐに開始されるため、2 分間の警告は表示されません。

スポットインスタンスの中断を適切に処理する最善の方法は、耐障害性のあるアプリケーションを設計することです。これを実現するには、スポットインスタンスの中断通知を活用します。中断通知は 5 秒ごとに確認することをお勧めします。

この中断通知は、EventBridge イベントとして、またスポットインスタンス上のインスタンスメタデータの項目として使用できます。中断通知は、ベストエフォートベースで出力されます。

EC2 Spot Instance interruption notice

Amazon EC2 がスポットインスタンスを中断しようとすると、実際の中断が起こる 2 分前にイベントが発生します (休止の場合は、即時的にその状態に移行するため、中断通知は発行されますが 2 分前には提供されず、このイベントの対象にはなりません)。このイベントは Amazon EventBridge で検出できます。EventBridge イベントの詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。イベントルールの作成および使用方法の詳細な例については、「Taking Advantage of Amazon EC2 スポットインスタンスInterruption Notices」を参照してください。

以下に、スポットインスタンスでの中断イベントの例を示します。`instance-action` の可能な値は `hibernate`、`stop`、`terminate` です。

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "EC2 Spot Instance Interruption Warning",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-2",
    "resources": ["arn:aws:ec2:us-east-2a:instance/i-1234567890abcdef0"],
    "detail": {
        "instance-id": "i-1234567890abcdef0",
        "instance-action": "action"
    }
}
```

> **Note**
>
> スポットインスタンスでの中断イベントの ARN 形式は `arn:aws:ec2:`*availability-zone*`:instance/`*instance-id* です。この形式は、EC2 リソース ARN 形式とは異なります。

instance-action

Amazon EC2 が、スポットインスタンスを停止または終了のためにマークした場合、インスタンスメタデータ内に `instance-action` 項目が含まれるようになります。そうでない場合、これは存在しません。次のように、インスタンスメタデータサービスバージョン 2 (IMDSv2) を使用して、`instance-action` を取得できます。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-
ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
spot/instance-action
```

instance-action 項目は、アクションおよびアクションのおよその発生時刻 (UTC) を指定します。

次の出力例では、このインスタンスの停止時刻を示します。

```
{"action": "stop", "time": "2017-09-18T08:22:00Z"}
```

次の出力例では、このインスタンスの終了時刻を示します。

```
{"action": "terminate", "time": "2017-09-18T08:22:00Z"}
```

Amazon EC2 がインスタンスを停止または終了する準備をしていない場合や、お客様が自分でインスタンスを終了した場合、instance-action はインスタンスメタデータ内に存在せず、取得しようとした場合、HTTP 404 エラーが出力されます。

termination-time

この項目は下位互換性のために維持されています。代わりに instance-action を使用してください。

Amazon EC2 がスポットインスタンスを終了のためにマークした場合は、termination-time 項目がインスタンスメタデータ内に含まれるようになります。そうでない場合、これは存在しません。次のように、IMDSv2 を使用して termination-time を取得できます。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-
ec2-metadata-token-ttl-seconds: 21600"`
[ec2-user ~]$ if curl -H "X-aws-ec2-metadata-token: $TOKEN" -s http://169.254.169.254/
latest/meta-data/spot/termination-time | grep -q .*T.*Z; then echo terminated; fi
```

termination-time 項目は、インスタンスがシャットダウン信号を受信するおよその時刻 (UTC) を指定します。以下は出力例です。

```
2015-01-05T18:02:00Z
```

Amazon EC2 がインスタンスを終了する準備をしていない場合や、ユーザーがスポットインスタンスを終了した場合には、termination-time 項目はインスタンスメタデータ内に存在しない (この場合、HTTP 404 エラーが出力されます) か、時刻値以外の値が含まれます。

Amazon EC2 がインスタンスの終了に失敗した場合は、リクエストステータスが fulfilled に設定されます。termination-time 値は、元のおよその時刻のまま (過去の時刻になっていますが)、インスタンスのメタデータに残ります。

中断した スポットインスタンス の検索

コンソールの [Instances (インスタンス)] ペインには、スポットインスタンス を含むすべてのインスタンスが表示されます。スポットインスタンスのインスタンスライフサイクルは spot です。スポットインスタンスのインスタンス状態は、設定した中断動作に応じて stopped または terminated のいずれかになります。休止状態のスポットインスタンスの場合、インスタンスの状態は stopped です。

コンソールを使用して中断されたスポットインスタンスを検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 次のフィルターを適用してください:[インスタンスライフサイクル=スポット]。
4. 設定した中断動作に応じて、[インスタンス状態=停止] または [インスタンス状態=終了] フィルターを適用します。
5. スポットインスタンスごとに、[詳細] タブの [インスタンスの詳細] で、[状態遷移メッセージ] を探します。次のコードは、スポットインスタンスが中断されたことを示します。
   - Server.SpotInstanceShutdown
   - Server.SpotInstanceTermination
6. 中断の理由の詳細については、スポットリクエストのステータスコードを確認してください。詳細については、「the section called “スポットリクエストステータス”」を参照してください。

AWS CLI を使用して中断した スポットインスタンス を検索するには

--filters パラメータで describe-instances コマンドを使用すると、中断した スポットインスタンス を一覧表示できます。出力にインスタンス ID のみをリストするには、--query パラメータを含めます。

インスタンスの中断動作がスポットインスタンスを終了することである場合は、次のコマンドを使用してください:

```
aws ec2 describe-instances \
    --filters Name=instance-lifecycle,Values=spot Name=instance-state-
name,Values=terminated Name=state-reason-code,Values=Server.SpotInstanceTermination \
    --query "Reservations[*].Instances[*].InstanceId"
```

インスタンスの中断動作がスポットインスタンスを停止することである場合は、次のコマンドを使用してください:

```
aws ec2 describe-instances \
    --filters Name=instance-lifecycle,Values=spot Name=instance-state-
name,Values=stopped Name=state-reason-code,Values=Server.SpotInstanceShutdown \
    --query "Reservations[*].Instances[*].InstanceId"
```

Amazon EC2 がスポットインスタンスを終了しているかどうかを判別する

スポットインスタンスが終了している場合は、CloudTrail を使用して、Amazon EC2 がそのスポットインスタンスを終了しているかどうかを確認できます。AWS CloudTrail では、イベント名が `BidEvictedEvent` の場合、Amazon EC2 がスポットインスタンスを終了したことを示します。

CloudTrail で BidEvictedEvent イベントを表示するには

1. CloudTrail コンソール (https://console.aws.amazon.com/cloudtrail/) を開きます。
2. ナビゲーションペインで [Event history (イベント履歴)] を選択します。
3. フィルターのドロップダウンで、[イベント名] を選択し、右側のフィルターフィールドに [BidEvictedEvent] と入力します。
4. 結果のリストから [BidEvictedEvent] を選択し、その詳細を表示します。[イベントレコード] で、インスタンス ID を確認できます。

CloudTrail の使用の詳細については、「AWS CloudTrail による Amazon EC2 および Amazon EBS の API コールのログ記録」を参照してください。

中断された スポットインスタンス の請求

スポットインスタンスが中断された場合は、インスタンスおよび EBS ボリュームの使用状況に対して次のように料金が発生する場合があります。

インスタンスの使用状況

スポットインスタンスを中断するユーザー	オペレーティングシステム	最初の 1 時間で中断	最初の 1 時間後の任意の時間に中断
ユーザー自らスポットインスタンスを停止または終了した場合	Windows および Linux (RHEL と SUSE は除く)	使用された時間 (秒) の請求	使用された時間 (秒) の請求
	RHEL および SUSE	使用時間が 1 時間未満の場合でも、1 時間分の料金を請求	使用された 1 時間分 (中断された時間が 1 時間未満の場合も 1時間分) を請求
Amazon EC2 がスポットインスタンスを中断した場合	Windows および Linux (RHEL と SUSE は除く)	料金は発生しない	使用された時間 (秒) の請求
	RHEL および SUSE	料金は発生しない	使用された 1 時間分は請求されるが、中断された時間が 1 時間未満の場合は請求されない

EBS ボリュームの使用状況

中断されたスポットインスタンスの停止中は、維持されている EBS ボリュームに対してのみ課金されます。

EC2 フリートおよびスポットフリートでは、停止中のインスタンスの数が多い場合、アカウント内の EBS ボリューム数の上限を超えることがあります。

その他の料金

実行中のスポットインスタンスに、データ転送、Elastic IP アドレス、他の AWS マネージドサービスの使用など、他のサービスの料金が発生する場合、その使用分が請求されます。これは、誰がスポットインスタンスを中断したのか、いつ中断されたのかには関係ありません。Amazon EC2 が最

初の 1 時間以内にスポットインスタンスを中断したときにスポットインスタンスの使用料が請求されない場合でも、他の料金が発生する可能性があります。

他の料金の詳細については、「Amazon EC2 オンデマンド料金」を参照してください。

スポットプレイスメントスコア

スポットプレイスメントスコア機能では、スポットの容量要件に基づいて AWS リージョンやアベイラビリティーゾーンを推奨することができます。スポット容量は変動し、必要な容量が常に得られるかどうかはわかりません。スポットプレイスメントスコアは、リージョンまたはアベイラビリティーゾーンでスポットリクエストが成功する可能性を示します。

> **Note**
>
> スポットプレイスメントスコアは、利用可能な容量や中断のリスクに関して、いかなる保証も提供しません。スポットプレイスメントスコアは、レコメンデーションとしてのみ機能します。

利点

スポットプレイスメントスコア機能は、次の場合に使用できます。

- 現在のリージョンでの容量ニーズの増加または使用可能な容量の減少に応じて、必要に応じて別のリージョンでスポットコンピューティング性能を再配置してスケールします。
- 単一アベイラビリティーゾーンのワークロードを実行する最適なアベイラビリティーゾーンを特定します。
- 将来のスポット容量のニーズをシミュレートして、スポットベースのワークロードの拡張に最適なリージョンを選択できるようにします。
- スポットキャパシティのニーズを満たす、最適なインスタンスタイプの組み合わせを特定するには。

トピック

コスト

スポットプレイスメントスコア機能は追加料金なしで使用できます。

スポットプレイスメントスコアの仕組み

スポットプレイスメントスコア機能を使用する場合は、まずスポットインスタンスのコンピューティング要件を指定します。その後、Amazon EC2 は、スポットリクエストが成功する可能性が高い上位 10 リージョン、を返します。各リージョンまたはアベイラビリティーゾーンは、1 ~ 10 のスケールで採点されます。10 はスポットリクエストが成功する可能性が高いことを示し、1 はスポットリクエストが成功する可能性が低いことを示します。

スポットプレイスメントスコア機能を使用するには、次のステップに従います。

- ステップ 1: スポット要件を指定する
- ステップ 2: スポットプレイスメントスコアレスポンスをフィルターする
- ステップ 3: レコメンデーションを確認する
- ステップ 4: レコメンデーションを使用する

ステップ 1: スポット要件を指定する

まず、希望するターゲットスポット容量とコンピューティング要件を次のように指定します。

1. ターゲットスポット容量を指定し、オプションでターゲット容量の単位を指定します。

   目的のターゲットスポット容量は、インスタンスまたは vCPU の数、または MiB のメモリ量の観点から指定できます。vCPU 数またはメモリ量でターゲット容量を指定するには、ターゲット容量の単位を `vcpu` または `memory-mib` のように指定する必要があります。それ以外の場合、デフォルトはインスタンス数になります。

   vCPU の数またはメモリ量の観点からターゲット容量を指定することで、総容量をカウントするときにこれらの単位を使用できます。例えば、異なるサイズのインスタンスを組み合わせて使用する場合は、ターゲット容量を vCPU の総数として指定できます。スポットプレイスメントスコア機能は、vCPU の数でリクエスト内の各インスタンスタイプを考慮し、ターゲット容量を合計するときに、インスタンスの総数ではなく vCPU の総数をカウントします。

   例えば、合計ターゲット容量を 30 vCPU に指定し、インスタンスタイプリストが c5.xlarge (4 vCPU)、m5.2xlarge (8 vCPU)、および r5.large (2 vCPU) で構成されているとします。合計 30 個の vCPU を実現するには、2 個の c5.xlarge (2*4 vCPU)、2 個の m5.2xlarge (2*8 vCPU)、3 個の r5.large (3*2 vCPU) を混在させることができます。

2. インスタンスタイプまたはインスタンス属性を指定します。

   使用するインスタンスタイプを指定するか、コンピューティング要件に必要なインスタンス属性を指定して、それらの属性を持つインスタンスタイプを Amazon EC2 に識別させることができます。これは属性ベースのインスタンスタイプの選択と呼ばれます。

   同じスポットプレイスメントスコアリクエストで、インスタンスタイプとインスタンス属性の両方を指定することはできません。

   インスタンスタイプを指定する場合は、少なくとも 3 つの異なるインスタンスタイプを指定する必要があります。指定しないと、Amazon EC2 は低いスポットプレイスメントスコアを返します。同様に、インスタンス属性を指定する場合は、少なくとも 3 つの異なるインスタンスタイプを解決する必要があります。

スポット要件を指定するさまざまな方法の例については、「設定例」を参照してください。

ステップ 2: スポットプレイスメントスコアレスポンスをフィルターする

Amazon EC2 は、リージョンまたはアベイラビリティーゾーンごとにスポットプレイスメントスコアを計算し、スポットリクエストが成功する可能性のある上位 10 のリージョンまたは上位 10 のアベイラビリティーゾーンのいずれかを返します。デフォルトでは、スコアリングされたリージョンのリストが返されます。すべてのスポット容量を単一のアベイラビリティーゾーンに起動する場合は、スコアリングされたアベイラビリティーゾーンのリストをリクエストすると便利です。

リージョンフィルターを指定して、レスポンスで返されるリージョンを絞り込むことができます。

リージョンフィルターとスコアリングされたアベイラビリティーゾーンのリクエストを組み合わせることができます。このようにして、スコアリングされたアベイラビリティーゾーンは、フィルターしたリージョンに限定されます。リージョン内の最高スコアのアベイラビリティーゾーンを検索するには、そのリージョンのみを指定すると、そのリージョン内のすべてのアベイラビリティーゾーンのスコアリストが返されます。

ステップ 3: レコメンデーションを確認する

各リージョンまたはアベイラビリティーゾーンのスポットプレイスメントスコアは、ターゲット容量、インスタンスタイプの構成、過去および現在のスポット使用傾向、およびリクエストの時間に基づいて計算されます。スポット容量は絶えず変動するため、同じスポットプレイスメントスコアのリクエストは、異なる時間に計算されたときに異なるスコアを生成する可能性があります。

リージョンとアベイラビリティーゾーンは、1 ~ 10 のスケールで採点されます。スコアが 10 の場合は、スポットリクエストが成功する可能性が高いことを示します (ただし保証はされません)。スコアが 1 の場合は、スポットリクエストが成功する可能性がまったくないことを示します。異なるリージョンまたはアベイラビリティーゾーンで同じスコアが返される場合があります。

低スコアが返された場合は、コンピューティング要件を編集してスコアを再計算できます。また、同じコンピューティング要件についてスポットプレイスメントスコアのレコメンデーションを 1 日の異なる時間にリクエストすることもできます。

ステップ 4: レコメンデーションを使用する

スポットプレイスメントスコアは、スポットリクエストの構成がスポットプレイスメントスコアの構成とまったく同じであり (ターゲット容量、ターゲット容量の単位、インスタンスタイプまたはインスタンス属性)、`capacity-optimized` 配分戦略を使用するように構成されている場合にのみ意味を持ちます。それ以外の場合、使用可能なスポット容量が得られる可能性はスコアと一致しません。

スポットプレイスメントスコアはガイドラインとして機能し、スポットリクエストが完全にまたは部分的に満たされることを保証するスコアはありませんが、次の情報を使用して最良の結果を得ることができます。

- 同じ設定を使用する — スポットプレイスメントスコアは、Auto Scaling グループ、EC2 フリート、またはスポットフリートのスポットリクエスト設定 (ターゲット容量、ターゲット容量の単位、インスタンスタイプまたはインスタンス属性) がスポットプレイスメントスコアを取得するために入力した内容と同じである場合にのみ関連します。

  スポットプレイスメントスコアリクエストで属性ベースのインスタンスタイプの選択を使用した場合、属性ベースのインスタンスタイプの選択を使用して Auto Scaling グループ、EC2 フリート、またはスポットフリートを設定できます。詳細については、「使用するインスタンスタイプに関する一連の要件を持つ Auto Scaling グループの作成」、「EC2 フリートの属性ベースのインスタンスタイプの選択」、および「スポットフリートの属性ベースのインスタンスタイプの選択」を参照してください。

  > **Note**
  >
  > vCPU の数またはメモリ量の観点からターゲット容量を指定し、スポットプレイスメントスコア設定でインスタンスタイプを指定した場合は、現在 Auto Scaling グループ、EC2 フリート、またはスポットフリートでこの設定を作成できないことに注意してください。代わりに、インスタンスの重み付けは `WeightedCapacity` パラメータを使用して手動で設定する必要があります。

- **capacity-optimized** 配分戦略を使用する – いずれのスコアも、スポット容量のリクエストが成功するためには、フリートのリクエストがすべてのアベイラビリティーゾーン (リージョン間の容量をリクエストする場合) または単一のアベイラビリティーゾーン (1 つのアベイラビリティーゾーンで容量をリクエストする場合) と capacity-optimized スポット配分戦略を使用するように設定されていることを前提としています。lowest-price のような他の配分戦略を用いた場合、利用可能なスポット容量を得られる可能性はスコアと一致しません。
- すぐにスコアに基づいて行動する – スポットプレイスメントスコアのレコメンデーションは、リクエスト時の利用可能なスポット容量を反映したものであり、スポット容量の変動により、同じ構成でも異なる時期に計算すると異なるスコアになることがあります。スコアが 10 の場合、スポット容量リクエストが成功する可能性が高い (保証はされません) ことを意味しますが、最良の結果を得るには、すぐにスコアに基づいて行動することをお勧めします。また、容量リクエストを試行するたびに新しいスコアを取得することをお勧めします。

制限事項

- ターゲット容量制限 – スポットプレイスメントスコアのターゲット容量制限は、潜在的な使用量の増加を考慮しながら、最近のスポット使用量に基づきます。最近のスポット使用がない場合は、スポットリクエストの制限に合わせてデフォルトの低い制限が提供されます。
- リクエスト設定の制限 – スポットプレイスメントスコア機能の意図された使用に関連しないパターンを検出した場合、24 時間以内に新しいリクエスト設定の数を制限できます。上限に達した場合は、既に使用したリクエスト設定を再試行できますが、次の 24 時間まで新しいリクエスト設定を指定することはできません。
- インスタンスタイプの最小数 – インスタンスタイプを指定する場合は、少なくとも 3 つの異なるインスタンスタイプを指定する必要があります。指定しないと、Amazon EC2 は低いスポットプレイスメントスコアを返します。同様に、インスタンス属性を指定する場合は、少なくとも 3 つの異なるインスタンスタイプを解決する必要があります。インスタンスタイプは、異なる名前を持つ場合、異なると見なされます。例えば、m5.8xlarge、m5a.8xlarge、および m5.12xlarge はすべて異なると見なされます。

必要な IAM アクセス許可

デフォルトでは、IAM アイデンティティ (ユーザー、ロール、またはグループ) には、スポットプレイスメントスコア機能を使用するアクセス許可はありません。IAM アイデンティティにスポットプレイスメントスコア機能の使用を許可するには、ec2:GetSpotPlacementScores EC2 API アクションの使用許可を与える IAM ポリシーを作成する必要があります。次に、この許可を必要とする IAM アイデンティティにポリシーをアタッチします。

`ec2:GetSpotPlacementScores` EC2 API アクションの使用許可を与える IAM ポリシーの例を次に示します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:GetSpotPlacementScores",
            "Resource": "*"
        }
    ]
}
```

IAM ポリシーの編集の詳細については、「IAM ユーザーガイド」の「IAM ポリシーの編集」を参照してください。

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」の手順に従ってください。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。
- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

スポットプレイスメントスコアの計算

スポットプレイスメントスコアは、Amazon EC2 コンソールや AWS CLI を使って計算することができます。

トピック

インスタンス属性を指定してスポットプレイスメントスコアを計算する (コンソール)

インスタンス属性を指定してスポットプレイスメントスコアを計算するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. [Spot placement score] (スポットプレイスメントスコア) を選択します。
4. [Enter requirements] (要件の入力) を選択します。
5. [Target capacity] (ターゲット容量) には、インスタンス数や vCPU 数、またはメモリ量 (MiB) に関して希望する容量を入力します。
6. [Instance type requirements] (インスタンスタイプの要件) では、コンピューティング要件を指定し、Amazon EC2 にこれらの要件に最適なインスタンスタイプを識別させるために、[Specify instance attributes that match your compute requirements] (コンピューティング要件に一致するインスタンス属性を指定) を選択します。
7. [vCPUs] に、希望する vCPU の最小数と最大数を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。
8. [Memory (GiB)] (メモリ (GiB)) に、希望するメモリの最小値と最大値を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。
9. [CPU architecture] (CPU アーキテクチャ) では、必要なインスタンスアーキテクチャを選択します。
10. (オプション) [Additional instance attributes] (その他のインスタンス属性) では、オプションで 1 つ以上の属性を指定して、コンピューティング要件をより詳細に表現できます。追加の属性は、リクエストにさらに制約を追加します。追加の属性は省略できます。省略すると、デフォルト値が使用されます。各属性およびそのデフォルト値の説明については、「Amazon EC2 コマンドラインリファレンス」の「get-spot-placement-scores」を参照してください。
11. (オプション) 指定した属性を持つインスタンスタイプを表示するには、[Preview matching instance types] (一致するインスタンスタイプをプレビューする) を展開します。インスタンス

タイプをプレイスメント評価に使用しないようにするには、インスタンスを選択し、[Exclude selected instance types] (選択されたインスタンスタイプを除外する) を選択します。

12. [Load placement scores] (プレイスメントスコアのロード) を選択し、結果を確認します。
13. (オプション) 特定のリージョンのスポットプレイスメントスコアを表示するには、[Regions to evaluate] (評価するリージョン) で、評価するリージョンを選択し、[Calculate placement scores] (プレイスメントスコアの計算) を選択します。
14. (オプション) 表示されたリージョンの、アベイラビリティーゾーンのスポット配置スコアを表示するには、[アベイラビリティーゾーンあたりの配置スコアを表示] のチェックボックスをオンにします。スコアリングされたアベイラビリティーゾーンのリストは、すべてのスポット容量を 1 つのアベイラビリティーゾーンで起動する場合に便利です。
15. (オプション) コンピューティング要件を編集して新しいプレイスメントスコアを取得するには、[Edit] (編集) を選択し、必要な調整を行った後、[Calculate placement scores] (プレイスメントスコアの計算) を選択します。

インスタンスタイプを指定してスポットプレイスメントスコアを計算する (コンソール)

インスタンスタイプを指定してスポットプレイスメントスコアを計算するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. [Spot placement score] (スポットプレイスメントスコア) を選択します。
4. [Enter requirements] (要件の入力) を選択します。
5. [Target capacity] (ターゲット容量) には、インスタンス数や vCPU 数、またはメモリ量 (MiB) に関して希望する容量を入力します。
6. [Instance type requirements] (インスタンスタイプの要件) では、使用するインスタンスタイプを指定するため、[Manually select instance types] (手動でインスタンスタイプを選択する) を選択します。
7. [Select instance types] (インスタンスタイプを選択) を選択し、使用するインスタンスタイプを選択してから [Select] (選択) を選択します。インスタンスタイプをすばやく検索するには、フィルターバーを使用して、異なるプロパティでインスタンスタイプをフィルタリングできます。
8. [Load placement scores] (プレイスメントスコアのロード) を選択し、結果を確認します。
9. (オプション) 特定のリージョンのスポットプレイスメントスコアを表示するには、[Regions to evaluate] (評価するリージョン) で、評価するリージョンを選択し、[Calculate placement scores] (プレイスメントスコアの計算) を選択します。

10. (オプション) 表示されたリージョンの、アベイラビリティーゾーンのスポット配置スコアを表示するには、[アベイラビリティーゾーンあたりの配置スコアを表示] のチェックボックスをオンにします。スコアリングされたアベイラビリティーゾーンのリストは、すべてのスポット容量を 1 つのアベイラビリティーゾーンで起動する場合に便利です。
11. (オプション) インスタンスタイプのリストを編集して新しいプレイスメントスコアを取得するには、[Edit] (編集) を選択し、必要な調整を行ってから [Calculate placement scores] (プレイスメントスコアの計算) を選択します。

スポットプレイスメントスコアの計算 (AWS CLI)

スポットプレイスメントスコアを計算するには

1. (オプション) スポットプレイスメントスコアの設定で指定可能なすべてのパラメータを生成するには、get-spot-placement-scores コマンドと `--generate-cli-skeleton` パラメータを使用します。

```
aws ec2 get-spot-placement-scores \
    --region us-east-1 \
    --generate-cli-skeleton
```

正常な出力

```
{
    "InstanceTypes": [
        ""
    ],
    "TargetCapacity": 0,
    "TargetCapacityUnitType": "vcpu",
    "SingleAvailabilityZone": true,
    "RegionNames": [
        ""
    ],
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": [
            "x86_64_mac"
        ],
        "VirtualizationTypes": [
            "hvm"
        ],
        "InstanceRequirements": {
```

```
            "VCpuCount": {
                "Min": 0,
                "Max": 0
            },
            "MemoryMiB": {
                "Min": 0,
                "Max": 0
            },
            "CpuManufacturers": [
                "amd"
            ],
            "MemoryGiBPerVCpu": {
                "Min": 0.0,
                "Max": 0.0
            },
            "ExcludedInstanceTypes": [
                ""
            ],
            "InstanceGenerations": [
                "previous"
            ],
            "SpotMaxPricePercentageOverLowestPrice": 0,
            "OnDemandMaxPricePercentageOverLowestPrice": 0,
            "BareMetal": "excluded",
            "BurstablePerformance": "excluded",
            "RequireHibernateSupport": true,
            "NetworkInterfaceCount": {
                "Min": 0,
                "Max": 0
            },
            "LocalStorage": "included",
            "LocalStorageTypes": [
                "hdd"
            ],
            "TotalLocalStorageGB": {
                "Min": 0.0,
                "Max": 0.0
            },
            "BaselineEbsBandwidthMbps": {
                "Min": 0,
                "Max": 0
            },
            "AcceleratorTypes": [
                "fpga"
```

```
            ],
            "AcceleratorCount": {
                "Min": 0,
                "Max": 0
            },
            "AcceleratorManufacturers": [
                "amd"
            ],
            "AcceleratorNames": [
                "vu9p"
            ],
            "AcceleratorTotalMemoryMiB": {
                "Min": 0,
                "Max": 0
            }
        }
    },
    "DryRun": true,
    "MaxResults": 0,
    "NextToken": ""
}
```

2. 前のステップの出力を使用して JSON 設定ファイルを作成し、次のように設定します。

   a. TargetCapacity には、インスタンス数や vCPU 数、またはメモリ量 (MiB) に関して希望するスポット容量を入力します。

   b. TargetCapacityUnitType に、ターゲット容量の単位を入力します。このパラメータを省略すると、デフォルトで units になります。

      有効な値: units (インスタンス数に変換されます) | vcpu | memory-mib

   c. スコアリングされたアベイラビリティーゾーンのリストを返すレスポンスのため、SingleAvailabilityZone に true を指定します。スコアリングされたアベイラビリティーゾーンのリストは、すべてのスポット容量を 1 つのアベイラビリティーゾーンで起動する場合に便利です。このパラメータを省略すると、デフォルトで false となり、レスポンスは、スコアリングされたリージョンのリストを返します。

   d. (オプション) RegionNames で、フィルターとして使用するリージョンを指定します。リージョンコードを指定する必要があります (例: us-east-1)。

      リージョンフィルターを使用すると、レスポンスは指定したリージョンのみを返します。true で SingleAvailabilityZone を指定した場合は、指定したリージョンのアベイラビリティーゾーンのみを返します。

e. 同じ設定に InstanceTypes または InstanceRequirements を含めることができますが、両方を含めることはできません。

JSON 設定で、次のいずれかを指定します。

- インスタンスタイプのリストを指定するには、InstanceTypes パラメータでインスタンスタイプを指定します。少なくとも 3 つの異なるインスタンスタイプを指定します。1 つまたは 2 つのインスタンスタイプのみを指定した場合、スポットプレイスメントスコアは低スコアを返します。インスタンスタイプのリストについては、「Amazon EC2 インスタンスタイプ」を参照してください。
- Amazon EC2 がこれらの属性に一致するインスタンスタイプを識別するように、インスタンスの属性を指定するには、InstanceRequirements 構造内にある属性を指定します。

  VCpuCount、MemoryMiB および CpuManufacturers の値を指定する必要があります。その他の属性は省略できます。省略すると、デフォルト値が使用されます。各属性およびそのデフォルト値の説明については、「Amazon EC2 コマンドラインリファレンス」の「get-spot-placement-scores」を参照してください。

設定例については、「設定例」を参照してください。

3. JSON ファイルで指定した条件のスポットプレイスメントスコアを取得するには、get-spot-placement-scores コマンドを使用し、--cli-input-json パラメータで JSON ファイルの名前とパスを指定します。

```
aws ec2 get-spot-placement-scores \
    --region us-east-1 \
    --cli-input-json file://file_name.json
```

SingleAvailabilityZone が false に設定されているか、省略されている場合の出力例 (省略されている場合のデフォルトは false) – リージョンのスコアリングされたリストが返されます

```
"SpotPlacementScores": [
    {
        "Region": "us-east-1",
        "Score": 7
    },
    {
```

```
        "Region": "us-west-1",
        "Score": 5
    },
   ...
```

SingleAvailabilityZone が true に設定されている場合の出力例 – アベイラビリティーゾーンのスコアリストが返されます

```
"SpotPlacementScores": [
    {
        "Region": "us-east-1",
        "AvailabilityZoneId": "use1-az1"
        "Score": 8
    },
    {
        "Region": "us-east-1",
        "AvailabilityZoneId": "usw2-az3"
        "Score": 6
    },
   ...
```

設定例

AWS CLI を使用する場合、次の設定例を使用できます。

設定例

- 例: インスタンスタイプとターゲット容量の指定
- 例: メモリの観点からインスタンスタイプとターゲット容量を指定する
- 例: 属性ベースのインスタンスタイプ選択の属性を指定する
- 例: 属性ベースのインスタンスタイプ選択の属性を指定し、アベイラビリティーゾーンのスコアリストを返す

例: インスタンスタイプとターゲット容量の指定

次の設定例では、3 つの異なるインスタンスタイプと 500 スポットインスタンスのターゲットスポット容量を指定します。

```
{
    "InstanceTypes": [
```

```
        "m5.4xlarge",
        "r5.2xlarge",
        "m4.4xlarge"
    ],
    "TargetCapacity": 500
}
```

例: メモリの観点からインスタンスタイプとターゲット容量を指定する

次の設定例では、3 つの異なるインスタンスタイプと 500,000 MiB メモリのターゲットスポット容量を指定します。この場合、起動するスポットインスタンスの数で合計 500,000 MiB のメモリを提供する必要があります。

```
{
    "InstanceTypes": [
        "m5.4xlarge",
        "r5.2xlarge",
        "m4.4xlarge"
    ],
    "TargetCapacity": 500000,
    "TargetCapacityUnitType": "memory-mib"
}
```

例: 属性ベースのインスタンスタイプ選択の属性を指定する

次の設定例は、属性ベースのインスタンスタイプ選択用に設定され、その後に設定例の説明が記載されています。

```
{
    "TargetCapacity": 5000,
    "TargetCapacityUnitType": "vcpu",
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": ["arm64"],
        "VirtualizationTypes": ["hvm"],
        "InstanceRequirements": {
            "VCpuCount": {
                "Min": 1,
                "Max": 12
            },
            "MemoryMiB": {
                "Min": 512
            }
```

```
        }
    }
}
```

InstanceRequirementsWithMetadata

属性ベースのインスタンスタイプの選択を使用するには、設定に InstanceRequirementsWithMetadata 構造を含め、スポットインスタンスに必要な属性を指定します。

前の例で、次の必須インスタンス属性を指定しています。

- ArchitectureTypes – インスタンスタイプのアーキテクチャタイプは arm64 である必要があります。
- VirtualizationTypes – インスタンスタイプの仮想化タイプは hvm である必要があります。
- VCpuCount – インスタンスタイプには、1 個以上、最大 12 個の vCPU が必要です。
- MemoryMiB – インスタンスタイプには最低 512 MiB のメモリが必要です。Max パラメータを省略した場合、上限がないことを示します。

指定できるオプションの属性は他にもいくつかあります。属性のリストについては、「Amazon EC2 コマンドラインリファレンス」の「get-spot-placement-scores」を参照してください。

TargetCapacityUnitType

TargetCapacityUnitType パラメータは、ターゲット容量の単位を指定します。この例では、ターゲット容量が 5000、ターゲット容量単位のタイプが vcpu となっており、合わせて 5000 vCPU の希望ターゲット容量が指定されており、起動するスポットインスタンスの数で合計 5000 vCPU を提供する必要があります。

例: 属性ベースのインスタンスタイプ選択の属性を指定し、アベイラビリティーゾーンのスコアリストを返す

次の設定例は、属性ベースのインスタンスタイプ選択用に設定されています。"SingleAvailabilityZone": true を指定した場合、レスポンスはスコアリングされたアベイラビリティーゾーンのリストを返します。

```
{
    "TargetCapacity": 1000,
    "TargetCapacityUnitType": "vcpu",
    "SingleAvailabilityZone": true,
```

```
    "InstanceRequirementsWithMetadata": {
        "ArchitectureTypes": ["arm64"],
        "VirtualizationTypes": ["hvm"],
        "InstanceRequirements": {
            "VCpuCount": {
                "Min": 1,
                "Max": 12
            },
            "MemoryMiB": {
                "Min": 512
            }
        }
    }
}
```

スポットインスタンスのデータフィード

スポットインスタンスの料金について理解しやすくするため、Amazon EC2 では、スポットインスタンスの使用状況と料金の詳細を、データフィードにより提供しています。このデータフィードは、データフィードを購読するときに指定する Amazon S3 バケットに送信されます。

データフィードファイルは、通常、1 時間に 1 回バケットに届き、各使用時は、通常、単一のデータファイルでカバーされます。これらのファイルは、ユーザーのバケットに配信される前に圧縮 (gzip) されます。ファイルが大きい場合 (ある時間に関するファイルの内容が、圧縮前に 50 MB を超える場合など) は、Amazon EC2 は指定した時間の使用状況に関する情報を複数のファイルに書き込みます。

> **Note**
>
> 1 つの AWS アカウントにつき 1 つのスポットインスタンスデータフィードのみを作成できます。スポットインスタンス実行が一定の時間に満たない場合、その時間のデータフィードファイルは送信されません。

スポットインスタンスのデータフィードは、AWS中国 (北京)、中国 (寧夏)、AWSGovCloud (米国) 以外のすべてのリージョンおよびデフォルトでは無効になっているリージョンでサポートされています。

コンテンツ

- データフィードのファイル名と形式

- Amazon S3 バケットの要件
- スポットインスタンスのデータフィードの購読
- スポットインスタンスのデータフィードを詳細表示する
- データフィード内のデータを表示する
- スポットインスタンスのデータフィードを削除する

データフィードのファイル名と形式

スポットインスタンスのデータフィードのファイル名には、(UTC の日付と時刻を使用しながら)次のような形式が使用されます。

```
bucket-name.s3.amazonaws.com/optional-prefix/aws-account-id.YYYY-MM-DD-HH.n.unique-id.gz
```

例えば、バケット名が **my-bucket-name** で、プレフィクスが **my-prefix** である場合、ファイル名は次のようになります。

```
my-bucket-name.s3.amazonaws.com/my-prefix/111122223333.2023-12-09-07.001.b959dbc6.gz
```

バケット名の詳細については、「Amazon S3 ユーザーガイド」の「バケットの名前付け」を参照してください。

スポットインスタンスのデータフィードファイルはタブ区切りです。データファイルの各行は、1 個のインスタンス時間に対応し、次の表に示すフィールドが含まれています。

フィールド	説明
Timestamp	そのインスタンス使用量に対して請求される価格を決定するために使用されるタイムスタンプ。
UsageType	請求の対象となっている使用タイプおよびインスタンスタイプ。m1.small スポットインスタンス では、このフィールドは SpotUsage に設定されます。他のすべてのインスタンスタイプでは、このフィールドは SpotUsage:{instance-type} に設定されます。例えば、SpotUsage:c1.medium と指定します。

フィールド	説明
Operation	請求の対象となっている製品。Linux スポットインスタンス の場合、このフィールドは RunInstances に設定されます。Windows スポットインスタンス の場合、このフィールドは RunInstances:0002 に設定されます。スポット使用状況は、利用可能ゾーンに従ってグループ化されます。
InstanceID	このインスタンスの使用量情報を生成したスポットインスタンスの ID。
MyBidID	このインスタンスの使用量情報を生成したスポットインスタンスリクエストの ID。
MyMaxPrice	このスポットリクエストに指定された上限価格。
MarketPrice	Timestamp フィールドに指定された時刻のスポット料金。
Charge	このインスタンス使用量に請求される価格。
Version	データフィードバージョン。可能性のあるバージョンは 1.0 です。

Amazon S3 バケットの要件

データフィードの購読時に、データフィードファイルを格納する Amazon S3 バケットを指定する必要があります。

データフィード用の Amazon S3 バケットを選択する前に、以下の点を考慮します。

- バケットに対する FULL_CONTROL アクセス権限が必要です。バケット所有者には、デフォルトでこの権限があります。それ以外の場合、バケット所有者は AWS アカウントにこのアクセス権限を付与する必要があります。
- データフィードを購読すると、これらのアクセス権限を使用してバケット ACL が更新され、AWS データフィードアカウントに FULL_CONTROL アクセス権限が付与されます。AWS データフィードアカウントは、データフィードファイルをバケットに書き込みます。アカウントに必要なアクセス許可がない場合、データフィードファイルをバケットに書き込むことはできません。詳細につい

ては、「Amazon CloudWatch Logs ユーザーガイド」の「Amazon S3 に送信されたログ」を参照してください。

> **Note**
>
> ACL を更新して AWS データフィードアカウントのアクセス権限を削除すると、データフィードファイルをバケットに書き込むことができなくなります。データフィードファイルを受け取るには、データフィードを再購読する必要があります。

- 各データフィードファイルには、独自の ACL があります (バケットの ACL とは別です)。バケット所有者には、データファイルに対して FULL_CONTROL のアクセス許可があります。AWS データフィードアカウントには読み書きのアクセス権限があります。
- 無効化された ACL をバケットに適用した場合は、フルコントロール権限を持つユーザーにバケットへの書き込みを許可するバケットポリシーを追加してください。詳細については、バケットポリシーを確認および更新する方法を参照してください。
- データフィードの購読を削除しても、Amazon EC2 は、AWS データフィードアカウントでの、バケットまたはデータファイルに対する読み書きのアクセス許可を削除しません。これらのアクセス許可は自分で削除する必要があります。
- AWS Key Management Service (SSE-KMS) に保存されている AWS KMS キーによるサーバー側の暗号化を使用して Simple Storage Service (Amazon S3) バケットを暗号化する場合は、カスタマーマネージド型キーを使用する必要があります。詳細については、「Amazon CloudWatch Logs ユーザーガイド」の「Simple Storage Service (Amazon S3) バケットのサーバー側の暗号化」を参照してください。

> **Note**
>
> スポットインスタンスデータフィードの場合、S3 ファイルを生成するリソースは Amazon CloudWatch Logs ではなくなりました。したがって、`aws:SourceArn` セクションを S3 バケット許可ポリシーおよび KMS ポリシーから削除する必要があります。

スポットインスタンスのデータフィードの購読

データフィードを購読するには、create-spot-datafeed-subscription コマンドを使用します。

```
aws ec2 create-spot-datafeed-subscription \
    --bucket my-bucket-name \
```

```
    [--prefix my-prefix]
```

出力例

```
{
    "SpotDatafeedSubscription": {
        "OwnerId": "111122223333",
        "Bucket": "my-bucket-name",
        "Prefix": "my-prefix",
        "State": "Active"
    }
}
```

スポットインスタンスのデータフィードを詳細表示する

データフィードの購読の詳細を表示するには、describe-spot-datafeed-subscription コマンドを使用します。

```
aws ec2 describe-spot-datafeed-subscription
```

出力例

```
{
    "SpotDatafeedSubscription": {
        "OwnerId": "123456789012",
        "Prefix": "spotdata",
        "Bucket": "my-s3-bucket",
        "State": "Active"
    }
}
```

データフィード内のデータを表示する

AWS Management Consoleで AWS CloudShell を開きます。次の s3 sync コマンドを使用して、データフィードの S3 バケットから .gz ファイルを取得し、指定したフォルダに保存します。

```
aws s3 sync s3://my-s3-bucket ./data-feed
```

.gz ファイルの内容を表示するには、S3 バケットの内容を保存したフォルダに移動します。

```
cd data-feed
```

ls コマンドを使用してファイルの名前を表示します。zcat コマンドをファイルの名前と共に使用すると、圧縮ファイルの内容が表示されます。以下にサンプルコマンドを示します。

```
zcat  111122223333.2023-12-09-07.001.b959dbc6.gz
```

以下は出力例です。

```
#Version: 1.0
#Fields: Timestamp UsageType Operation InstanceID MyBidID MyMaxPrice MarketPrice Charge
 Version
2023-12-09 07:13:47 UTC USE2-SpotUsage:c7a.medium       RunInstances:SV050
 i-0c3e0c0b046e050df     sir-pwq6nmfp    0.0510000000 USD        0.0142000000 USD
  0.0142000000 USD        1
```

スポットインスタンスのデータフィードを削除する

データフィードを削除するには、delete-spot-datafeed-subscription コマンドを使用します。

```
aws ec2 delete-spot-datafeed-subscription
```

Spot Instance クォータ

実行中および要求されたスポットインスタンスの数、そして保留中のスポット インスタンスの数には、各リージョンにつき AWS アカウント ごとに割り当てがあります。保留中のスポットインスタンスリクエストが受理されると、実行中のインスタンスがクォータにカウントされるため、リクエストはクォータにカウントされなくなります。

スポットインスタンスのクォータは、仮想中央演算装置 (vCPU) の数について管理されます。この数は、実行中のスポットインスタンスが使用中であるか、未処理のスポットインスタンスリクエストの受理が保留中であるため、後に使用されるかにより決定されます。ユーザーがスポットインスタンスを終了しており、かつスポットインスタンスリクエストをキャンセルしていない場合、Amazon EC2 がスポットインスタンスの終了を検出してリクエストを閉じるまで、リクエストはスポットインスタンスでの vCPU のクォータ数についてカウントされます。

スポットインスタンスには以下のクォータタイプが用意されています。

- オール DL スポットインスタンスリクエスト
- オール F スポットインスタンスリクエスト
- オール G および VT スポットインスタンスリクエスト
- オール Inf スポットインスタンスリクエスト

- オール P スポットインスタンスリクエスト
- オールスタンダード (A、C、D、H、I、M、R、T、Z) スポットインスタンスリクエスト
- すべての Trn スポットインスタンスリクエスト
- オール X スポットインスタンスリクエスト

各クォータタイプは、1 つ以上のインスタンスファミリーに対し、最大の vCPU 数を指定しています。さまざまなインスタンスファミリー、世代、およびサイズの詳細については、Amazon EC2 インスタンスタイプを参照してください。

変化するアプリケーションのニーズに合わせて、任意の組み合わせのインスタンスタイプを起動できます。例えば、オールスタンダードスポットインスタンスリクエストのクォータが 256 vCPU の場合、32 `m5.2xlarge` 個のスポットインスタンス (32 x 8 vCPU) または、16 `c5.4xlarge` 個のスポットインスタンス (16 x 16 vCPU) をリクエストできます。

タスク

- スポットインスタンスのクォータと使用量のモニタリング
- クォータ引き上げをリクエストする

スポットインスタンスのクォータと使用量のモニタリング

以下を使用してスポットインスタンスのクォータを表示および管理できます。

- Service Quotas コンソールの Amazon EC2 サービスクォータページ
- get-service-quota AWS CLI

詳細については、Amazon EC2 Linux インスタンス用ユーザーガイドの「Amazon EC2 の Service Quotas」、および「Service Quotas User Guide」の「Viewing service quotas」を参照してください。

Amazon CloudWatch のメトリクス統合では、クォータに対して EC2 の使用量をモニタリングできます。クォータに近づいたときに警告を発するようにアラームを設定することもできます。詳細については、Service Quotas ユーザーガイドの「Service Quotas と Amazon CloudWatch アラーム」。

クォータ引き上げをリクエストする

スポットインスタンスの上限は、使用量に基づき Amazon EC2 によって自動的に引き上げられますが、必要であればクォータの引き上げをリクエストすることも可能です。例えば、現在のクォー

タで許可されているよりも多くの スポットインスタンス を起動する場合は、クォータの引き上げをリクエストできます。また、スポットインスタンスリクエストを送信した後にエラー `Max spot instance count exceeded` を受け取ったとしても、クォータの引き上げをリクエストできます。クォータの増加を要求するには、Amazon EC2 の Service Quotas で説明されている Service Quotas コンソールを使用します。

バーストパフォーマンスインスタンス

T インスタンスタイプはバーストパフォーマンスインスタンスです。バーストパフォーマンスインスタンスタイプを使用してスポットインスタンスを起動し、CPU クレジットを蓄積するアイドル時間なしでバーストパフォーマンススポットインスタンスをすぐに短時間使用する場合は、支払いコストが高くなるのを避けるために、インスタンスを標準モードで起動することをお勧めします。バーストパフォーマンス スポットインスタンス を Unlimited モード で起動し、すぐに CPU をバーストさせると、余分なクレジットがバーストに消費されます。インスタンスを短時間使用する場合、インスタンスは余分なクレジットに見合うだけの CPU クレジットを蓄積する時間がないため、インスタンスの終了時に余分なクレジットに対して課金されます。

Unlimited モードがバーストパフォーマンス スポットインスタンス に適しているのは、バースト用の CPU クレジットが蓄積されるまで、そのインスタンスが十分に長く実行される場合のみです。それ以外の場合は、余分なクレジットを支払う必要があるため、バーストパフォーマンス スポットインスタンス は他のインスタンスよりも、使用コストが高くなります。詳細については、「Unlimited モードと固定 CPU を使用する場合」を参照してください。

T2 インスタンスは、標準モードで設定すると、起動クレジットを取得します。T2 インスタンスは、起動クレジットを取得できる唯一のバーストパフォーマンスインスタンスです。起動クレジットは、インスタンスを構成するために十分なコンピューティングリソースを提供し、T2 インスタンスの初期起動を効率的に実現することを意図しています。T2 インスタンスの起動を繰り返して新しい起動クレジットにアクセスすることは許可されていません。CPU が持続的に必要な場合、(一定期間のアイドリングにより) クレジットを獲得して T2 スポットインスタンス の Unlimited モードを使用するか、専用 CPU を搭載したインスタンスタイプを使用します。

Dedicated Hosts

Amazon EC2 Dedicated Host は完全にお客様専用の物理サーバーです。オプションで、インスタンス容量を他の AWS アカウントと共有することもできます。詳細については、共有 Dedicated Hosts の操作 をご参照ください。

専有ホストは、インスタンスの配置を可視化および制御し、ホストアフィニティをサポートします。つまり、特定のホストでインスタンスを起動して実行でき、インスタンスが特定のホストでのみ実行

されるようにできます。詳細については、「自動配置とアフィニティについて」を参照してください。

専有ホストは、包括的な Bring-Your-Own-License (BYOL) サポートを提供します。これにより、Windows Server、SQL Server、SUSE Linux Enterprise Server、Red Hat Enterprise Linux、または VM、ソケット、または物理コアにバインドされているその他のソフトウェアライセンスを含む、既存のソケット単位、コア単位、または VM 単位のソフトウェアライセンスをライセンス条項に従って使用できます。

インスタンスを専用ハードウェアで実行する必要があるが、インスタンスの配置を可視化または制御する必要はなく、ソケット単位またはコア単位のソフトウェアライセンスを使用する必要がない場合は、代わりにハードウェア専有インスタンスを使用することを検討できます。ハードウェア専有インスタンスと専有ホストのどちらを使用しても、専用の物理サーバーに Amazon EC2 インスタンスを起動することができます。ハードウェア専有インスタンス と Dedicated Hosts のインスタンスの間に、パフォーマンス、セキュリティ、または物理的な違いはありません。ただし、これらにはいくつかの重要な違いがあります。次のテーブルでは、Dedicated Hosts とハードウェア専有インスタンスの主な違いをいくつか紹介します。

	Dedicated Host	Dedicated Instance
専用物理サーバー	お客様専用のインスタンス容量を持つ物理サーバー。	単一の顧客アカウント専用の物理サーバー。
インスタンス容量の共有	インスタンス容量を他のアカウントと共有できます。	サポートされていません
請求	ホストごとの請求	インスタンスごとの請求
ソケット、コア、ホスト ID の可視性	ソケットと物理コアの数が見える	可視性なし
ホストおよびインスタンスアフィニティ	インスタンスを同じ物理サーバーに徐々にデプロイし続けることができる	サポート外
ターゲットを絞ったインスタンスの配置	インスタンスを物理サーバーに配置する方法についての可視性と制御が高い	サポート外

	Dedicated Host	Dedicated Instance
インスタンスの自動復旧	サポート対象。詳細については、ホスト復旧 を参照してください。	サポート対象
Bring-Your-Own-License (BYOL)	サポート	部分的なサポート*
キャパシティ予約	サポート外	サポート

* ソフトウェアアシュアランスによるライセンスモビリティを使用する Microsoft SQL Server、および Windows Virtual Desktop Access (VDA) ライセンスを、ハードウェア専有インスタンスで使用することが可能です。

専有インスタンスの詳細については、「Dedicated Instances」を参照してください。

コンテンツ

インスタンスキャパシティの設定

Dedicated Hosts はさまざまな構成 (物理コア、ソケット、vCPU) をサポートしているため、さまざまなファミリーやサイズのインスタンスを実行できます。

アカウントに 専有ホスト を割り当てる場合、単一のインスタンスタイプ、または同じインスタンスファミリー内の複数のインスタンスタイプをサポートする構成を選択できます。ホストで実行できるインスタンスの数は、選択した設定によって異なります。

コンテンツ

単一インスタンスタイプのサポート

1 つのインスタンスタイプのみをサポートする専有ホストを割り当てることができます。この設定では、専有ホストで起動するすべてのインスタンスは、ホストを割り当てるときに指定する同じインスタンスタイプである必要があります。

例えば、m5.4xlarge インスタンスタイプのみをサポートするホストを割り当てることができます。この場合、そのホスト上で実行できるインスタンスは m5.4xlarge のみです。

ホスト上で起動できるインスタンスの数は、ホストが提供する物理コアの数と、指定されたインスタンスタイプによって消費されるコア数によって異なります。例えば、m5.4xlarge インスタンスにホストを割り当てると、ホストは 48 個の物理コアを提供し、各 m5.4xlarge インスタンスは 8 個の物理コアを消費します。つまり、そのホストでは最大 6 つのインスタンスを起動できます (48 物理コア/インスタンスあたり 8 コア = 6 インスタンス)。

複数のインスタンスタイプのサポート

同じインスタンスファミリー内の複数のインスタンスタイプをサポートする専有ホストを割り当てることができます。これにより、同じインスタンス ファミリー内にあり、ホストに十分なインスタンスキャパシティがある限り、同じホスト上で異なるインスタンスタイプを実行できます。

例えば、R5 インスタンス ファミリー内のさまざまなインスタンスタイプをサポートするホストを割り当てることができます。この場合は、そのホスト上で、ホストの物理コア容量まで、r5.large、r5.xlarge、r5.2xlarge、r5.4xlarge などの R5 インスタンスタイプの任意の組み合わせを起動できます。

次のインスタンスファミリーは、複数のインスタンスタイプをサポートする Dedicated Hosts をサポートしています。

- 一般的な用途:A1、M5、M5n、M6i、T3

- コンピューティングの最適化: C5、C5n、および C6i
- メモリ最適化: R5、R5n、R6i

ホストで実行できるインスタンスの数は、ホストが提供する物理コア数、およびホスト上で実行する各インスタンスタイプによって消費されるコア数によって異なります。例えば、48 個の物理コアを提供する R5 ホストを割り当て、2 つの r5.2xlarge インスタンス (4 コア x 2 インスタンス) と 3 つの r5.4xlarge インスタンス (8 コア x 3 インスタンス) を実行するとします。これらのインスタンスは合計 32 コアを消費します。残りの 16 コアを超えない限り、R5 インスタンスを任意に組み合わせて実行できます。

ただし、各インスタンスサイズで実行可能な上限のインスタンス数は、インスタンスファミリーごとに異なります。例えば、R5 専有ホスト は、32 個の物理コアを使用する r5.8xlarge インスタンスを最大 2 個までサポートします。さらに、ホストのコア容量を満たすために、より小さいサイズの R5 インスタンスを追加して使用できます。各インスタンスファミリーでサポートされているインスタンスサイズの数については、「Amazon EC2 Dedicated Hosts の料金」を参照してください。

次の表に、インスタンスタイプの組み合わせの例を示します。

インスタンスファミリー	インスタンスサイズの組合せ例	
R5	• 具体例 1: 4 x r5.4xlarge + 4 x r5.2xlarge • 具体例 2: 1 x r5.12xlarge + 1 x r5.4xlarge + 1 x r5.2xlarge + 5 x r5.xlarge + 2 x r5.large	
C5	• 具体例 1: 1 x c5.9xlarge + 2 x c5.4xlarge + 1 x c5.xlarge • 具体例 2: 4 x c5.4xlarge + 1 x c5.xlarge + 2 x c5.large	
M5	• 具体例 1: 4 x m5.4xlarge + 4 x m5.2xlarge •	

インスタンスファミリー	インスタンスサイズの組合せ例	
	具体例 2: 1 x m5.12xlarge + 1 x m5.4xlarge + 1 x m5.2xlarge + 5 x m5.xlarge + 2 x m5.large	

考慮事項

複数のインスタンスタイプをサポートする Dedicated Hosts を使用する場合は、次の点に注意してください。

- C5n、M5n、R5n などの N タイプの Dedicated Hosts では、小さいサイズのインスタンス (2xlarge 以下) と大きいインスタンスサイズ (4xlarge 以上、metal を含む) を混在させることはできません。N タイプの Dedicated Hosts で小さいインスタンスサイズと大きいインスタンスサイズを同時にホストする必要がある場合は、小さいインスタンスサイズと大きいインスタンスサイズに別々のホストを割り当てる必要があります。
- 大きいインスタンスサイズから起動し、必要に応じて残りのインスタンスキャパシティを小さいインスタンスサイズで埋めることをお勧めします。

Bring your own license

Dedicated Hosts を利用すると、ソフトウェアライセンスを、既存のソケット単位、コア単位、または VM 単位で使用できます。自分のライセンスを使用する場合、お客様は自分のライセンスを管理する責任があります。ただし、Amazon EC2 ではインスタンスアフィニティやターゲットを絞ったプレイスメントなど、ライセンスのコンプライアンスを維持するための機能を利用できます。

自分のボリュームライセンスマシンのイメージを Amazon EC2 で使用するための一般的な手順を以下に示します。

1. マシンイメージの使用を制御するライセンス条件が、仮想化クラウド環境での使用を許可していることを確認します。
2. マシンイメージを Amazon EC2 内で使用できることを確認したら、VM Import/Export を使用してインポートします。マシンイメージをインポートする方法については、VM Import/Export ユーザーガイドを参照してください。
3. マシンイメージをインポートしたら、自分のアカウント内のアクティブな Dedicated Hosts で、そのイメージからインスタンスを起動できます。

4. オペレーティングシステムによっては、これらのインスタンスを実行する際、自分の KMS サーバー 。

> **Note**
>
> イメージが AWS 内でどのように使用されているかを追跡するには、AWS Config でホストの記録を有効にします。AWS Config では、Dedicated Hosts での設定の変更を記録したり、その出力をライセンスレポートのデータソースとして使用したりすることができます。詳細については、「設定の変更の追跡」を参照してください。

料金と請求

Dedicated Host の料金は支払いオプションごとに異なります。

支払いオプション

- オンデマンド Dedicated Hosts
- Dedicated Host Reservations
- Savings Plans
- Dedicated Hosts での Windows サーバーの料金

オンデマンド Dedicated Hosts

アカウントに Dedicated Host を割り当てると、自動的にオンデマンド請求がアクティブになります。

Dedicated Host のオンデマンド価格は、インスタンスファミリーとリージョンによって異なります。起動するインスタンスの数量やサイズに関係なく、アクティブな Dedicated Host に対して 1 秒あたり (最低 60 秒) の料金が発生します。オンデマンド料金の詳細については、「Amazon EC2 Dedicated Hosts オンデマンド料金」を参照してください。

オンデマンド専有ホストはいつでもリリースして、料金の発生を止めることができます。Dedicated Host の解放の詳細については、「Dedicated Hosts のリリース」を参照してください。

Dedicated Host Reservations

Dedicated Host の予約 では、オンデマンド Dedicated Hosts の実行と比べて請求の割引が得られます。予約は、3 つの支払いオプションで利用できます。

- 前払いなし — 前払いなしの予約では、期間内の Dedicated Host の使用に対して割引があり、前払い料金は必要ありません。1 年および 3 年契約で利用できます。前払いなしの予約での 3 年契約は、一部のインスタンスファミリーのみでサポートされます。
- 一部前払い — 予約の一部を前払いする必要があり、期間内の残りの時間は割引された時間料金で請求されます。1 年および 3 年契約で利用できます。
- 全額前払い — 実質的に最低価格で利用できます。1 年および 3 年契約で利用でき、期間中のすべてのコストが含まれます。それ以外の料金は発生しません。

予約を購入するには、アカウントでアクティブな Dedicated Hosts が必要です。各予約では、単一のアベイラビリティーゾーンで同じインスタンスファミリーをサポートしている、複数のホストに対応できます。予約は、インスタンスサイズではなくホストのインスタンスファミリーに適用されます。インスタンスサイズが異なる 3 つの Dedicated Hosts (`m4.xlarge`、`m4.medium`、および `m4.large`) がある場合、1 つの `m4` 予約をこれらすべての Dedicated Hosts に関連付けることができます。予約のインスタンスファミリーとアベイラビリティーゾーンは、関連付ける Dedicated Hosts のインスタンスファミリーとアベイラビリティーゾーンに一致させる必要があります。

予約が専有ホストに関連付けられている場合、専有ホストは予約期間が終了するまでリリースできません。

予約の料金の詳細については、「Amazon EC2 Dedicated Hosts 料金表」を参照してください。

Savings Plans

Savings Plans は、オンデマンドインスタンスと比べて大幅に料金を節約できる柔軟な料金モデルです。Savings Plans は、1〜3 年間は時間あたりの USD 建て料金で一定量の使用を継続するという確約を条件とする割引プランです。これによって、特定のDedicated Hostを使用することをコミットせずに、ニーズに最も適したDedicated Hostsを使用して、コストを削減し続ける柔軟性が得られます。詳細については、AWS Savings Plans ユーザーガイドをご参照ください。

> **Note**
>
> Savings Plans は、u-6tb1.metal、u-9tb1.metal、u-12tb1.metal、u-18tb1.metal、および u-24tb1.metal の Dedicated Hosts ではサポートされていません。

Dedicated Hosts での Windows サーバーの料金

Microsoft のライセンス条項に抵触しなければ、お手元にすでにある Windows Server や SQL Server のライセンスを Dedicated Hosts に移すことができます。ご自分のライセンスを使用する場合、追加のソフトウェア使用料は発生しません。

さらに、Amazon が提供する Windows Server AMI を使用して、最新バージョンの Windows Server を Dedicated Hosts で実行できます。これは、Dedicated Hosts 上で実行できる既存の SQL Server のライセンスは持っているが、SQL Server ワークロードを実行するために Windows Server を必要としている場合に、広く見られることです。Amazon が提供する Windows Server AMI のサポートは、現行のインスタンスタイプにのみ適用されます。詳細については、Amazon EC2 Dedicated Hosts の料金を参照してください。

Dedicated Hosts 上のバースト可能な T3 インスタンス

Dedicated Hosts は、バースト可能なパフォーマンス T3 インスタンスをサポート T3 インスタンスは、適格な BYOL ライセンスソフトウェアを専用ハードウェアで使用するための費用対効果の高い方法を提供します。T3 インスタンスの vCPU フットプリントが小さいため、ワークロードを少数のホストに統合し、コアごとのライセンスの使用率を最大化できます。

T3 Dedicated Hosts は、CPU 使用率が低 ~ 中程度の BYOL ソフトウェアを実行するのに最適です。Windows Server、Windows デスクトップ、SQL Server、SUSE Enterprise Linux Server、Red Hat Enterprise Linux および Oracle データベースなどの、ソケット単位、コア単位または VM 単位が含まれます。T3 Dedicated Hosts に適したワークロードの例としては、小中規模のデータベース、仮想デスクトップ、開発/テスト環境、コードリポジトリ、製品プロトタイプなどがあります。T3 Dedicated Hosts は、CPU 使用率が高いワークロードや、相関する CPU バーストが同時に発生するワークロードには推奨されません。

Dedicated Hosts 上の T3 インスタンスは、共有テナンシーハードウェア上の T3 インスタンスと同じクレジットモデルを使用します。ただし、それらはstandardクレジットモードのみです。unlimitedクレジットモードはサポートしていません。standard モードの場合、Dedicated

Host 上の T3 インスタンスは、共有テナンシーハードウェア上のバースト可能なインスタンスと同じ方法で、クレジットの獲得、消費、および蓄積を行います。バーストパフォーマンスインスタンスは、ベースラインレベルの CPU パフォーマンスを提供しながら、必要に応じてバーストする機能を備えています。ベースラインより上にバーストする場合、インスタンスは CPU クレジット残高に蓄積されたクレジットを消費します。発生したクレジットが枯渇すると、CPU 使用率はベースラインレベルまで低下します。`standard`モードの詳細については、「スタンダードのバーストパフォーマンスインスタンスの仕組み」を参照してください。

T3 Dedicated Hosts は、Amazon EC2 Dedicated Hosts が提供するすべての機能をサポートします。これには、1 つのホスト上の複数のインスタンスサイズ、ホストリソースグループ、および BYOL が含まれます。

サポートされる T3 インスタンスのサイズと構成

T3 Dedicated Hosts は、ベースライン CPU パフォーマンスと、必要に応じてより高いレベルまでバーストする機能を提供することにより、ホストの CPU リソースを共有する汎用バースト可能な T3 インスタンスを実行します。これにより、48 個のコアを持つ T3 Dedicated Host は、ホストあたり最大 192 個のインスタンスをサポートできます。ホストのリソースを効率的に利用し、最高のインスタンスパフォーマンスを提供するために、Amazon EC2 インスタンス配置アルゴリズムは、ホスト上で起動できるサポートされているインスタンス数とインスタンスサイズの組み合わせを自動的に計算します。

T3 Dedicated Host は、同じホストで複数のインスタンスタイプをサポートします。Dedicated Hosts では、すべての T3 インスタンスがサポートされています。ホストの CPU 制限まで、T3 インスタンスのさまざまな組み合わせを実行できます。

次の表は、サポートされているインスタンスタイプのリストと、各インスタンスタイプのパフォーマンスのサマリー、および起動可能な各サイズのインスタンスの最大数を示しています。

インスタンスタイプ	vCPUs	メモリ (GiB)	vCPU あたりのベースライン CPU 使用率	ネットワークバースト帯域幅 (Gbps)	Amazon EBS バースト帯域幅 (Mbps)	専有ホストあたりの最大インスタンス数
t3.na	2	0.5	5%	5	最大 2,085	192
t3.m	2	1	10%	5	最大 2,085	192
t3.sn	2	2	20%	5	最大 2,085	192
t3.m	2	4	20%	5	最大 2,085	192
t3.la	2	8	30%	5	2,780	96
t3.xl	4	16	40%	5	2,780	48
t3.2x e	8	32	40%	5	2,780	24

T3 Dedicated Hosts の CPU 使用率の監視

`DedicatedHostCPUUtilization`Amazon CloudWatch メトリックスを使用して、専有ホストの vCPU 使用率を監視します。メトリクスは、`EC2`名前空間および`Per-Host-Metrics`ディメンションで利用可能です。詳細については、Dedicated Hostsメトリクス を参照してください。

Dedicated Hosts の制約事項

Dedicated Hosts を割り当てる際は、次の制限と制約に注意してください。

- Dedicated Hosts で RHEL、SUSE Linux、SQL Server を実行するには、独自の AMI を使用する必要があります。AWS が提供している、あるいは AWS Marketplace から入手が可能な RHEL、SUSE Linux、SQL Server の AMI は、Dedicated Hosts では使用できません。独自の AMI を作成する方法の詳細については、「Bring your own license」を参照してください。

この制限は、ハイメモリインスタンス (u-6tb1.metal、u-9tb1.metal、u-12tb1.metal、u-18tb1.metal、および u-24tb1.metal) に割り当てられたホストには適用されません。AWS によって提供される、または AWS Marketplace で利用できる RHEL および SUSE Linux AMI は、これらのホストで使用できます。

- インスタンスファミリーごとの Dedicated Hosts の実行数には、リージョンごとに AWS アカウントあたりの上限があります。クォータは実行中のインスタンスにのみ適用されます。インスタンスが保留中、停止処理中、停止済みの場合、クォータにはカウントされません。アカウントのクォータを確認する、または引き上げをリクエストするには、Service Quotas コンソールを使用してください。
- Dedicated Host で実行されるインスタンスは、VPC でのみ起動できます。
- Auto Scaling グループは、ホストリソースグループを指定する起動テンプレートを使用する場合にサポートされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「詳細設定を使用して起動テンプレートを作成する」を参照してください。
- Amazon RDS インスタンスはサポートされません。
- AWS 無料利用枠は Dedicated Hosts ではご使用になれません。
- インスタンスのプレイスメント制御は、Dedicated Hosts でのインスタンスの起動管理を表します。Dedicated Hosts をプレイスメントグループで起動することはできません。
- 仮想インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを .metal インスタンスタイプに変更することはできません。例えば、m5.large インスタンスタイプにホストを割り当てた場合、インスタンスタイプを m5.metal に変更することはできません。

  同様に、.metal インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを仮想インスタンスタイプに変更することはできません。例えば、m5.metal インスタンスタイプにホストを割り当てた場合、インスタンスタイプを m5.large に変更することはできません。

Dedicated Hosts の操作

Dedicated Host を使用するには、まずアカウントで使用するホストを割り当てます。次にインスタンスのホストテナンシーを指定して、ホストにインスタンスを起動します。インスタンスを起動する特定のホストを選択する必要があります。または、自動配置が有効になっており、そのインスタンスタイプが一致するどのホストでも起動できるようにすることもできます。インスタンスを停止して再起動する場合、同じホストで再起動されるか別のホストで再起動されるかは、ホストのアフィニティ設定によって決まります。

あるオンデマンドホストが不要になった場合は、そのホストで実行されているインスタンスを停止し、別のホストで起動するように指定してから、ホストをリリースすることができます。

Dedicated Hosts は AWS License Manager にも統合されています。License Manager では、ホストリソースグループを作成できます。ホストリソースグループは、1 つのエンティティとして管理される Dedicated Hosts コレクションです。ホストリソースグループを作成する場合は、Dedicated Hosts のホスト管理設定 (自動割り当てや自動リリースなど) を指定します。これにより、これらのホストを手動で割り当てて管理することなく、Dedicated Hosts にインスタンスを作成できます。詳細については、AWS License Manager ユーザーガイドの「 ホスト Resource Groups」を参照してください。

コンテンツ

Dedicated Hosts の割り当て

Dedicated Hostsの使用を始めるには、Amazon EC2コンソールまたはコマンドラインツールを使い、ご自身のアカウントにおいてDedicated Hostsを割り当てる必要があります。Dedicated Hostを割り当てると、ご自身のアカウントにおいてDedicated Hostの容量がすぐに使用できるようになり、Dedicated Hostにおけるインスタンスの起動を開始できます。

アカウントに 専有ホスト を割り当てる場合、単一のインスタンスタイプ、または同じインスタンスファミリー内の複数のインスタンスタイプをサポートする構成を選択できます。ホスト上で実行できるインスタンスの数は、選択した構成によって異なります。詳細については、「インスタンスキャパシティの設定」を参照してください。

Console

Dedicated Host を割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts]、[Dedicated Host の割り当て] の順に選択します。
3. [インスタンスファミリー] については、Dedicated Host向けのインスタンスファミリーを選びます。
4. Dedicated Host について、選択したインスタンスファミリー内にある複数のインスタンスをサポートするか、特定のインスタンスタイプのみサポートするかを指定します。次のいずれかを行ってください。

   - 選択したインスタンスファミリー内の複数のインスタンスタイプをサポートするように Dedicated Hostを設定するには、[複数のインスタンスタイプをサポートする]、[有効化] の順に選択します。この選択を行うと、Dedicated Host において、同一インスタンスファミリー内の異なるインスタンスサイズを起動できるようになります。例えば、m5インスタンスファミリーとこのオプションを選択すると、Dedicated Hostにおいて`m5.xlarge`インスタンスと`m5.4xlarge`インスタンスを起動できます。
   - 選択したインスタンスファミリーの 1 つのインスタンスタイプをサポートするように Dedicated Host を設定するには、[Support multiple instance types (複数のインスタンスをサポートする)] をオフにして、[インスタンスタイプ] でサポートするインスタンスタイプを選択します。これにより、Dedicated Host で 1 つのインスタンスタイプを起動できます。例えば、このオプションを選択し、`m5.4xlarge`をサポート対象インスタンスタイプとして指定すると、専有ホストにおいては`m5.4xlarge`インスタンスに限り起動できます。

5. [アベイラビリティーゾーン] については、専有ホストを割り当てるアベイラビリティーゾーンを選択します。
6. インスタンスタイプが一致する、ターゲットを絞らないインスタンスの起動を受け入れることを Dedicated Host に許可するには、[Instance auto-placement (インスタンスの自動プレイスメント)] で、[Enable (有効)] を選択します。自動配置の詳細については、「自動配置とアフィニティについて」を参照してください。

7. Dedicated Host のホスト復旧を有効にするには、[Host recovery (ホスト復旧)]、[有効化] の順に選択します。詳細については、ホスト復旧 を参照してください。
8. [数量] については、割り当てるDedicated Hostsの数を入力します。
9. (オプション) [新しいタグの追加] をクリックし、タグキーとタグ値を入力します。
10. [Allocate] を選択します。

AWS CLI

Dedicated Host を割り当てるには

allocate-hosts AWS CLI コマンドを使用します。次のコマンドでは、m5 アベイラビリティーゾーンで us-east-1a インスタンスファミリー内の複数のインスタンスタイプをサポートしている専有ホストを割り当てます。ホストでもホスト復旧が有効になっており、自動配置は無効になっています。

```
aws ec2 allocate-hosts --instance-family "m5" --availability-zone "us-east-1a" --
auto-placement "off" --host-recovery "on" --quantity 1
```

次のコマンドでは、eu-west-1a アベイラビリティーゾーンでターゲット未指定の m4.large インスタンス起動をサポートする専有ホストを割り当て、ホスト復旧を有効にして、purpose のキーと production の値を使用してタグを適用します。

```
aws ec2 allocate-hosts --instance-type "m4.large" --availability-zone "eu-west-1a"
 --auto-placement "on" --host-recovery "on" --quantity 1 --tag-specifications
 'ResourceType=dedicated-host,Tags=[{Key=purpose,Value=production}]'
```

PowerShell

Dedicated Host を割り当てるには

New-EC2Host AWS Tools for Windows PowerShell コマンドを使用します。次のコマンドでは、m5 アベイラビリティーゾーンで us-east-1a インスタンスファミリー内の複数のインスタンスタイプをサポートしている専有ホストを割り当てます。ホストでもホスト復旧が有効になっており、自動配置は無効になっています。

```
PS C:\> New-EC2Host -InstanceFamily m5 -AvailabilityZone us-east-1a -
AutoPlacement Off -HostRecovery On -Quantity 1
```

次のコマンドでは、eu-west-1a アベイラビリティーゾーンでターゲットを絞らないm4.large インスタンスの起動をサポートする専有ホストを割り当て、ホスト復旧を有効にして、purpose のキーと production の値を使用してタグを適用します。

作成時に Dedicated Host にタグを付けるために使用される TagSpecification パラメータには、タグ付けされるリソースのタイプ、タグキー、タグ値を指定するオブジェクトが必要です。次のコマンドは必要なオブジェクトを作成します。

```
PS C:\> $tag = @{ Key="purpose"; Value="production" }
PS C:\> $tagspec = new-object Amazon.EC2.Model.TagSpecification
PS C:\> $tagspec.ResourceType = "dedicated-host"
PS C:\> $tagspec.Tags.Add($tag)
```

次のコマンドは Dedicated Host を割り当て、$tagspec オブジェクトで指定されたタグを適用します。

```
PS C:\> New-EC2Host -InstanceType m4.large -AvailabilityZone eu-west-1a -
AutoPlacement On -HostRecovery On -Quantity 1 -TagSpecification $tagspec
```

Dedicated Host でのインスタンスの起動

Dedicated Host を割り当てたら、そのホストにインスタンスを起動できます。起動するインスタンスタイプに使用できる十分な容量を持つアクティブなDedicated Hostsがない場合には、hostテナンシーでインスタンスを起動できません。

> **Tip**
>
> 複数のインスタンスサイズをサポートする Dedicated Hosts については、大きいインスタンスサイズから始め、必要に応じて残りのインスタンスキャパシティを小さいインスタンスサイズで埋めることをお勧めします。

インスタンスを起動する前に、制限事項を確認してください。詳細については、Dedicated Hosts の制約事項 を参照してください。

次の方法を使用して Dedicated Host でインスタンスを起動できます。

Console

Dedicated Hosts ページから特定の Dedicated Host でインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Dedicated Hosts] を選択します。
3. [Dedicated Hosts] ページでホストを選択した後、[Actions] (アクション)、[Launch Instance(s) onto host] (インスタンスをホストで起動) の順に選択します。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。

   > **Note**
   >
   > Amazon EC2 によって提供されている SQL Server、SUSE、RHEL AMI を Dedicated Hosts で使用することはできません。

5. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択します。

   > **Note**
   >
   > Dedicated Hostが単一のインスタンスタイプのみサポートしている場合、デフォルトでは、サポートされているインスタンスタイプが選択され、変更できません。Dedicated Host が複数のインスタンスタイプをサポートしている場合は、Dedicated Host の使用可能なインスタンスキャパシティーに基づいて、サポートされているインスタンスファミリー内のインスタンスタイプを選択する必要があります。大きいインスタンスサイズから始め、必要に応じて残りのインスタンス容量を小さいインスタンスサイズで埋めることをお勧めします。

6. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択します。
7. [Advanced details] (高度な詳細) セクションにある [Tenancy affinity] (テナンシーのアフィニティ) で、以下のいずれかを実行します。

   - Off を選択– インスタンスは指定されたホストで起動されますが、停止された後に、以前と同一の Dedicated Host で再開される保証はありません。
   - Dedicated Host ID を選択 – インスタンスは停止後も、常に同じ特定のホストで再開されます。

アフィニティの詳細については、「自動配置とアフィニティについて」を参照してください。

> **Note**
>
> [Tenancy (テナンシー)] オプションと [Host (ホスト)] オプションは、選択したホストに基づき、事前に設定されています。

8. 必要に応じて、残りのインスタンスオプションを設定します。詳細については、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
9. [インスタンスを起動] を選択します。

インスタンス起動ウィザードを使用してインスタンスを Dedicated Host で起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスを起動) の順に選択します。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。

> **Note**
>
> Amazon EC2 によって提供されている SQL Server、SUSE、RHEL AMI を Dedicated Hosts で使用することはできません。

4. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択します。
5. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択します。
6. [Advanced] (高度な情報) セクションで以下を実行します。
   a. [Tenancy] (テナンシー) で、[Dedicated Host] (専有ホスト) を選択します。
   b. [Target host by] (ターゲットホスト) で、[Host ID] (ホスト ID) を選択します。
   c. [Target host ID] (ターゲットホスト ID) で、インスタンスを起動するホストを選択します。
   d. [Tenancy affinity] (テナンシーのアフィニティ) で、以下のいずれかを実行します。

- Off を選択– インスタンスは指定されたホストで起動されますが、停止された後に、以前と同一の Dedicated Host で再開される保証はありません。
- Dedicated Host ID を選択 – インスタンスは停止後も、常に同じ特定のホストで再開されます。

アフィニティの詳細については、「自動配置とアフィニティについて」を参照してください。

7. 必要に応じて、残りのインスタンスオプションを設定します。詳細については、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
8. [インスタンスを起動] を選択します。

AWS CLI

Dedicated Host でインスタンスを起動するには

run-instances AWS CLI コマンドを使用し、`Placement` リクエストパラメータでインスタンスのアフィニティ、テナンシー、およびホストを指定します。

PowerShell

Dedicated Host でインスタンスを起動するには

New-EC2Instance AWS Tools for Windows PowerShell コマンドを使用し、`Placement` リクエストパラメータでインスタンスのアフィニティ、テナンシー、およびホストを指定します。

ホストリソースグループへのインスタンスの作成

インスタンスの作成先のホストリソースグループ内のいずれかの Dedicated Host にインスタンス用の空き容量がある場合、Amazon EC2 はそのホストにインスタンスを作成します。ホストリソースグループ内のいずれのホストにもインスタンス用の空き容量がない場合、Amazon EC2 はホストリソースグループ内に新しいホストを自動的に割り当て、そのホストにインスタンスを作成します。詳細については、AWS License Manager ユーザーガイドの「 ホストリソースグループ」を参照してください。

要件と制限

- コアベースまたはソケットベースのライセンス設定を AMI に関連付ける必要があります。

- Dedicated Hosts の Amazon EC2 で提供されている SQL Server、SUSE、または RHEL AMI を使用することはできません。
- ホスト ID を選択して特定のホストをターゲットにすることはできません。また、ホストリソースグループにインスタンスを作成するときに、インスタンスのアフィニティを有効にすることはできません。

次の方法を使用して、ホストリソースグループにインスタンスを起動できます。

Console

ホストリソースグループにインスタンスを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスを起動) の順に選択します。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。

   > **Note**
   >
   > Amazon EC2 によって提供されている SQL Server、SUSE、RHEL AMI を Dedicated Hosts で使用することはできません。

4. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択します。
5. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択します。
6. [Advanced] (高度な情報) セクションで以下を実行します。
   a. [Tenancy] (テナンシー) で、[Dedicated Host] (専有ホスト) を選択します。
   b. [Target host by] (ターゲットホスト) で、[Host resource group] (ホストリソースグループ) を選択します。
   c. [Tenancy host resource group] (テナンシーのホストリソースグループ) で、インスタンスを起動するホストリソースグループを選択します。
   d. [Tenancy affinity] (テナンシーのアフィニティ) で、以下のいずれかを実行します。
      - Off を選択– インスタンスは指定されたホストで起動されますが、停止された後に、以前と同一の Dedicated Host で再開される保証はありません。

- Dedicated Host ID を選択 – インスタンスは停止後も、常に同じ特定のホストで再開されます。

アフィニティの詳細については、「自動配置とアフィニティについて」を参照してください。

7. 必要に応じて、残りのインスタンスオプションを設定します。詳細については、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
8. [インスタンスを起動] を選択します。

AWS CLI

ホストリソースグループにインスタンスを作成するには

run-instances AWS CLI コマンドを使用し、`Placement` リクエストパラメータでテナンシーオプションを省略してホストリソースグループ ARN を指定します。

PowerShell

ホストリソースグループにインスタンスを作成するには

New-EC2Instance AWS Tools for Windows PowerShell コマンドを使用し、`Placement` リクエストパラメータでテナンシーオプションを省略してホストリソースグループ ARN を指定します。

自動配置とアフィニティについて

Dedicated Hostsの配置制御は、インスタンスおよびホストの両レベルで行われます。

自動配置

自動配置はホストレベルで設定されます。自動配置を使用すると、起動するインスタンスについて、特定のホストで起動されるようにするか、設定が合致する任意のホストで起動されるようにするかを管理できます。

Dedicated Host の自動配置が無効になっている場合は、一意のホスト ID を指定するホストテナンシーインスタンス起動のみが受け入れられます。これは、新しい Dedicated Hosts に対する既定の設定です。

Dedicated Host の自動配置が有効になっている場合は、インスタンスタイプ設定が一致するすべてのターゲット未指定のインスタンス起動が受け入れられます。

インスタンスの起動時に、テナンシーを設定する必要があります。特定の HostId を指定せずに Dedicated Host でインスタンスを起動すると、自動配置が有効で、インスタンスタイプが一致するすべての Dedicated Host でインスタンスを起動できます。

ホストのアフィニティ

ホストのアフィニティは、インスタンスレベルで設定します。また、インスタンスと Dedicated Host の間に関係を作成します。

アフィニティが Host に設定されている場合は、特定のホストで起動したインスタンスが停止しても、常に同じホストで再開されます。これは、ターゲットを絞った起動にもターゲットを絞らない起動にも適用されます。

アフィニティがDefault に設定されているときにインスタンスを停止して再起動する場合は、使用可能な任意のホスト上で再起動できます。ただし、最後に実行した Dedicated Host 上でベストエフォートベースでの再起動を試みます。

Dedicated Host 自動配置の変更

Dedicated Host を AWS アカウントに割り当てると、その自動配置設定を変更することが可能になります。変更には、次のいずれかの方法を使用します。

Console

Dedicated Host の自動配置を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. ホストを選択し、[アクション]、[ホストの変更] の順に選択します。
4. [インスタンスの自動配置] で、[有効化] を選択して自動配置を有効にするか、[有効化] をオフにして自動配置を無効にします。詳細については、自動配置とアフィニティについて を参照してください。
5. [Save] を選択します。

AWS CLI

Dedicated Host の自動配置を変更するには

modify-hosts AWS CLI コマンドを使用します。次の例では、指定した Dedicated Host の自動配置を有効にします。

```
aws ec2 modify-hosts --auto-placement on --host-ids h-012a3456b7890cdef
```

PowerShell

Dedicated Host の自動配置を変更するには

Edit-EC2Host AWS Tools for Windows PowerShell コマンドを使用します。次の例では、指定した Dedicated Host の自動配置を有効にします。

```
PS C:\> Edit-EC2Host --AutoPlacement 1 --HostId h-012a3456b7890cdef
```

サポートされているインスタンスタイプの変更

同じ専有ホストにおける複数のインスタンスタイプのサポートは C5、M5、R5、C5n、R5n、M5n、および T3 の各インスタンスファミリーで利用できます。他のインスタンスファミリーは、同じ Dedicated Host における単一のインスタンスタイプのみをサポートしています。

Dedicated Host は、次の方法で割り当てることができます。

お客様は、Dedicated Hostを修正することで、このホストがサポートするインスタンスタイプを変更できます。単一のインスタンスタイプのみサポートしているホストの場合には、インスタンスファミリー内にある複数のインスタンスタイプをサポートするように修正できます。同様に、複数のインスタンスタイプをサポートしているホストの場合には、単一のインスタンスタイプのみをサポートするように修正できます。

複数のインスタンスタイプをサポートするようにDedicated Hostを修正するには、初めに、該当ホスト上で実行中のインスタンスをすべて停止する必要があります。修正は約 10 分で完了します。修正の実行中には、Dedicated Hostがpending状態に移行します。pending状態にあるDedicated Hostにおいて停止中のインスタンスを開始したり、新たなインスタンスを起動したりすることはできません。

複数のインスタンスタイプをサポートしているDedicated Hostを 1 つのインスタンスタイプのみをサポートするように変更するには、ホストで実行中のインスタンスがないか、実行中のインスタンスがホストでサポートするインスタンスタイプである必要があります。具体例を挙げると、m5インスタンスファミリー内にある複数のインスタンスタイプをサポートしているホストを、m5.largeインスタンスのみをサポートするように修正するには、Dedicated Hostが、いかなるインスタンスも実行していない状態であるか、m5.large実行中のインスタンスのみの状態でなければなりません。

仮想インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを .metal インスタンスタイプに変更することはできません。例えば、m5.large インスタンスタイプにホストを割り当てた場合、インスタンスタイプを m5.metal に変更することはできません。同様に、.metal インスタンスタイプにホストを割り当てる場合、ホストの割り当て後にインスタンスタイプを仮想インスタンスタイプに変更することはできません。例えば、m5.metal インスタンスタイプにホストを割り当てた場合、インスタンスタイプを m5.large に変更することはできません。

サポートされているインスタンスタイプは、次のいずれかの方法で変更できます。

Console

Dedicated Host のサポートされているインスタンスタイプを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] (専有ホスト) を選択します。
3. 変更する Dedicated Host を選択し、[アクション]、[ホストの変更] の順に選択します。
4. Dedicated Host の現在の設定に応じて、次のいずれかの操作を実行します。

   - Dedicated Host が特定のインスタンスタイプを現在サポートしている場合は、[複数のインスタンスタイプをサポートする] は有効にならず、現在サポートされているインスタンスタイプが [インスタンスタイプ] に表示されます。現在のインスタンスファミリー内にある複数のタイプをサポートするようにホストを変更するには、[複数のインスタンスタイプをサポートする] の [有効化] を選択します。

     複数のインスタンスタイプをサポートするようにホストを修正するには、初めに、該当ホスト上で実行されているすべてのインスタンスを停止する必要があります。
   - Dedicated Host がインスタンスファミリー内の複数のインスタンスタイプを現在サポートしている場合は、[複数のインスタンスタイプをサポートする] の [有効] が選択されています。特定のインスタンスタイプをサポートするようにホストを変更するには、[複数のインスタンスタイプをサポートする] で、[有効化] をオフにし、[インスタンスタイプ] で、サポートする特定のインスタンスタイプを選択します。

     Dedicated Host がサポートするインスタンスファミリーを変更することはできません。
5. [Save] を選択します。

AWS CLI

Dedicated Host のサポートされているインスタンスタイプを変更するには

modify-hosts AWS CLI コマンドを使用します。

以下のコマンドは、m5インスタンスファミリー内にある複数のインスタンスタイプをサポートするようにDedicated Hostを修正します。

```
aws ec2 modify-hosts --instance-family m5 --host-ids h-012a3456b7890cdef
```

以下のコマンドは、m5.xlargeインスタンスのみをサポートするように Dedicated Host を修正します。

```
aws ec2 modify-hosts --instance-type m5.xlarge --instance-family --host-
ids h-012a3456b7890cdef
```

PowerShell

Dedicated Host のサポートされているインスタンスタイプを変更するには

Edit-EC2Host AWS Tools for Windows PowerShell コマンドを使用します。

以下のコマンドは、m5インスタンスファミリー内にある複数のインスタンスタイプをサポートするようにDedicated Hostを修正します。

```
PS C:\> Edit-EC2Host --InstanceFamily m5 --HostId h-012a3456b7890cdef
```

以下のコマンドは、m5.xlargeインスタンスのみをサポートするように Dedicated Host を修正します。

```
PS C:\> Edit-EC2Host --InstanceType m5.xlarge --HostId h-012a3456b7890cdef
```

インスタンスのテナンシーとアフィニティの変更

インスタンスのテナンシーは、インスタンスの起動後に変更できます。インスタンスのアフィニティを変更して、特定のホストをターゲットにしたり、アカウント内の属性が一致する使用可能な専有ホストで起動できるようにしたりすることもできます。インスタンスのテナンシーまたはアフィニティを修正するには、そのインスタンスをstopped状態にする必要があります。

インスタンスのオペレーティングシステムの詳細、および SQL Server がインストールされているかどうかによって、サポートされる変換が影響されます。インスタンスで使用できるテナンシー変換パスの詳細については、「License Manager ユーザーガイド」の「テナンシー変換」を参照してください。

> **Note**
>
> T3 インスタンスの場合、host のテナンシーを使用するには専有ホストでインスタンスを起動する必要があります。T3 インスタンスの場合、テナンシーを host から dedicated または default に変更することはできません。これらのサポートされていないテナンシー変更のいずれかを試みると、エラーコード InvalidRequest が発生します。

インスタンスのテナンシーとアフィニティは、次の方法を使用して変更できます。

Console

インスタンスのテナンシーまたはアフィニティを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [Instances (インスタンス)] を選択し、変更するインスタンスを選択します。
3. [Instance state (インスタンスの状態)]、[Stop (停止)] の順に選択します。
4. 選択したインスタンスについて、[アクション]、[インスタンス設定]、[インスタンスの配置の変更] を選択します。
5. [インスタンスの配置の変更] ページで、次の設定を行います。

   - [Tenancy] — 次のいずれかを選択します。
     - [専用ハードウェアインスタンスの実行] — インスタンスを ハードウェア専有インスタンス として起動します。詳細については、Dedicated Instances を参照してください。
     - [Launch the instance on a Dedicated Host] — 設定可能なアフィニティを使用してインスタンスを Dedicated Host で起動します。
   - [Affinity] — 次のいずれかを選択します。
     - [This instance can run on any one of my hosts] — インスタンスは、そのインスタンスタイプをサポートするアカウントの利用可能な Dedicated Host で起動されます。
     - [This instance can only run on the selected host] — インスタンスは、[Target Host] (ターゲットホスト) として選択された Dedicated Host でのみ実行できます。
   - [Target Host] (ターゲットホスト) – インスタンスを実行させるための、Dedicated Host を選択します。ターゲットホストが表示されない場合は、アカウントに利用可能な、互換性のある Dedicated Hosts がない可能性があります。

   詳細については、自動配置とアフィニティについて を参照してください。

6. [Save] を選択します。

AWS CLI

インスタンスのテナンシーまたはアフィニティを変更するには

modify-instance-placement AWS CLI コマンドを使用します。次の例では、指定したインスタンスのアフィニティを `default` から `host` に変更し、インスタンスがアフィニティを持つ対象の Dedicated Host を指定します。

```
aws ec2 modify-instance-placement --instance-id i-1234567890abcdef0 --affinity host
 --tenancy host --host-id h-012a3456b7890cdef
```

PowerShell

インスタンスのテナンシーまたはアフィニティを変更するには

Edit-EC2InstancePlacement AWS Tools for Windows PowerShell コマンドを使用します。次の例では、指定したインスタンスのアフィニティを `default` から `host` に変更し、インスタンスがアフィニティを持つ対象の Dedicated Host を指定します。

```
PS C:\> Edit-EC2InstancePlacement -InstanceId i-1234567890abcdef0 -Affinity host -
Tenancy host -HostId h-012a3456b7890cdef
```

Dedicated Hosts の表示

Dedicated Host およびその各インスタンスの詳細を表示するには、次の方法を使用できます。

Console

Dedicated Host の詳細を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. [Dedicated Hosts] ページでホストを選択します。
4. ホストの情報を表示するには、[詳細] を選択します。

   [使用可能な vCPU] は、Dedicated Host における新たなインスタンスの起動に使用できる vCPU を示します。具体例を挙げると、c5インスタンスファミリー内にある複数のインスタ

ンスタイプをサポートしているDedicated Hostの場合、いかなるインスタンスも実行されていなければ 72 の vCPU を使用できます。これは、Dedicated Hostにおいては 72 の使用可能な vCPU を使って異なるインスタンスタイプの組合せを起動できることを意味します。

ホストで実行中のインスタンスの情報を表示するには、[実行中のインスタンス] を選択します。

AWS CLI

Dedicated Host の容量を表示するには

describe-hosts AWS CLI コマンドを使用します。

次の例では、c5 インスタンスファミリー内の複数のインスタンスタイプをサポートしている、Dedicated Host で使用可能なインスタンス容量を表示するために、describe-hosts (AWS CLI) コマンドを使用しています。このDedicated Hostにおいては、すでに 2 つのc5.4xlargeインスタンスと 4 つのc5.2xlargeインスタンスが実行されています。

```
$  aws ec2 describe-hosts --host-id h-012a3456b7890cdef
```

```
"AvailableInstanceCapacity": [
    { "AvailableCapacity": 2,
      "InstanceType": "c5.xlarge",
      "TotalCapacity": 18 },
    { "AvailableCapacity": 4,
      "InstanceType": "c5.large",
      "TotalCapacity": 36 }
  ],
"AvailableVCpus": 8
```

PowerShell

Dedicated Host のインスタンス容量を表示するには

Get-EC2Host AWS Tools for Windows PowerShell コマンドを使用します。

```
PS C:\> Get-EC2Host -HostId h-012a3456b7890cdef
```

Dedicated Hosts のタグ付け

既存の Dedicated Host にカスタムタグを割り当てて、目的、所有者、環境など、さまざまな方法で分類できます。これにより、割り当てたカスタムタグに基づいて特定の Dedicated Host をすばやく見つけることができます。Dedicated Host タグは、コスト割り当ての追跡にも使用できます。

作成時に Dedicated Hosts にタグを適用することもできます。詳細については、Dedicated Hosts の割り当て を参照してください。

Dedicated Host にタグを付けるには、次の方法を使用できます。

Console

Dedicated Host にタグを付けるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. タグを付ける対象の Dedicated Host を選択し、[アクション]、[タグの管理] の順に選択します。
4. [タグの管理] 画面で、[タグの追加] を選択し、タグのキーと値を指定します。
5. (オプション) [タグの追加] を選択して、Dedicated Host に付けるタグを追加します。
6. [Save changes] を選択します。

AWS CLI

Dedicated Host にタグを付けるには

create-tags AWS CLI コマンドを使用します。

次のコマンドでは、指定した Dedicated Host に `Owner=TeamA` のタグを付けます。

```
aws ec2 create-tags --resources h-abc12345678909876 --tags Key=Owner,Value=TeamA
```

PowerShell

Dedicated Host にタグを付けるには

New-EC2Tag AWS Tools for Windows PowerShell コマンドを使用します。

New-EC2Tag コマンドには、Dedicated Host のタグに使用するキーと値のペアを指定する Tag オブジェクトが必要です。下のコマンドでは、キーと値のペアとして Tag と $tag を使用し、Owner という名前の TeamA オブジェクトを作成します。

```
PS C:\> $tag = New-Object Amazon.EC2.Model.Tag
PS C:\> $tag.Key = "Owner"
PS C:\> $tag.Value = "TeamA"
```

次のコマンドでは、指定した Dedicated Host に $tag オブジェクトをタグ付けします。

```
PS C:\> New-EC2Tag -Resource h-abc12345678909876 -Tag $tag
```

Dedicated Hosts のモニタリング

Amazon EC2 は、Dedicated Hosts の状態を絶えずモニタリングします。更新は Amazon EC2 コンソールで伝達されます。Dedicated Host に関する情報は、次の方法を使用して表示できます。

Console

Dedicated Host の状態を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. リストで Dedicated Host を見つけ、[State] 列で値を確認します。

AWS CLI

Dedicated Host の状態を表示するには

describe-hosts AWS CLI コマンドを使用して、state レスポンス要素の hostSet プロパティを確認します。

```
aws ec2 describe-hosts --host-id h-012a3456b7890cdef
```

PowerShell

Dedicated Host の状態を表示するには

Get-EC2Host AWS Tools for Windows PowerShell コマンドを使用し、`state` レスポンス要素の `hostSet` プロパティを確認します。

```
PS C:\> Get-EC2Host -HostId h-012a3456b7890cdef
```

以下の表では、表示される可能性のある Dedicated Host の状態について説明します。

状態	説明
`available`	AWS は Dedicated Host で問題を検出しませんでした。予定されているメンテナンスまたは修復はありません。この専有ホストでインスタンスを起動できます。
`released`	専有ホストがリリースされました。ホスト ID は使用中ではありません。リリース済みのホストは再使用できません。
`under-assessment`	AWS は Dedicated Host の潜在的な問題を調査しています。アクションを実行する必要がある場合は、AWS Management Consoleまたは E メールで通知されます。この状態のDedicated Hostではインスタンスを起動できません。
`pending`	この状態のDedicated Hostは新たなインスタンスの起動に使用できません。この状態は、複数のインスタンスタイプをサポートするように修正されている状態か、ホスト復旧実行中の状態です。
`permanent-failure`	回復不可能な障害が検出されました。インスタンスおよび E メールで削除通知が届きます。インスタンスは引き続き実行する場合があります。この状態にある Dedicated Host 上のすべてのインスタンスを停止または終了すると、AWS はホストを使用停止にします。AWS はこの状態のインスタンスを再起動しません。この状態のDedicated Hostsではインスタンスを起動できません。
`released-permanent-failure`	AWS は、障害が発生してインスタンスが実行されていない Dedicated Hosts を完全にリリースします。Dedicated Host ID も使用できなくなります。

Dedicated Hosts のリリース

ホストをリリースする前に、専有ホストで実行中のインスタンスを停止する必要があります。これらのインスタンスはアカウントの他の Dedicated Hosts に移行し、引き続き使用することができます。これらのステップは、オンデマンド Dedicated Hosts にのみ適用されます。

専有ホストをリリースするには、次の方法を使用できます。

Console

専有ホストをリリースするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. [Dedicated Hosts] ページでリリースする専有ホストを選択します。
4. [アクション]、[ホストのリリース] の順に選択します。
5. [リリース] を選択して確定します。

AWS CLI

専有ホストをリリースするには

release-hosts AWS CLI コマンドを使用します。

```
aws ec2 release-hosts --host-ids h-012a3456b7890cdef
```

PowerShell

専有ホストをリリースするには

Remove-EC2Hosts AWS Tools for Windows PowerShell コマンドを使用します。

```
PS C:\> Remove-EC2Hosts -HostId h-012a3456b7890cdef
```

専有ホストをリリースすると、同じホストまたはホスト ID を再使用できなくなり、該当ホストのオンデマンド料金請求が停止します。Dedicated Host の状態は `released` に変わり、このホストではインスタンスを起動できなくなります。

> **Note**
>
> 最近 Dedicated Hosts をリリースした場合、制限に加算されなくなるまでに少し時間がかかることがあります。それまでは、新しい Dedicated Hosts を割り当てようとすると `LimitExceeded` エラーが発生する場合があります。このエラーが発生した場合は、数分後に新しいホストを再び割り当ててみてください。

停止したインスタンスはまだ使用可能であり、[Instances] ページに表示されます。その [host] テナンシー設定も維持されています。

Dedicated Host の予約 の購入

予約は、次の方法で購入できます。

Console

予約を購入するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [Dedicated Hosts]、[Dedicated Host の予約]、[Dedicated Host の予約 の購入] の順に選択します。
3. [サービスを検索] 画面で、次の操作を行います。
   a. [インスタンスファミリー] で、専有ホスト予約を購入しようとしている専有ホストの、インスタンスファミリーを選択します。
   b. [支払いオプション] で、希望の支払いオプションを選択し、設定します。
4. [Next] を選択します。
5. 専有ホスト予約に関連付ける専有ホストを選択し、[次へ] をクリックします。
6. (オプション) 専有ホスト予約にタグを割り当てます。
7. 注文を確認し、[購入] をクリックします。

AWS CLI

予約を購入するには

1. describe-host-reservation-offerings AWS CLI コマンドを使用して、ニーズに合った利用可能なオファリングを一覧表示します。次の例では、m4 インスタンスファミリー内のインスタンスをサポートし、契約期間が 1 年のオファリングを一覧表示します。

   > **Note**
   >
   > 期間は秒単位で指定されます。1 年契約は 31,536,000 秒で、3 年契約は 94,608,000 秒です。

   ```
   aws ec2 describe-host-reservation-offerings --filter Name=instance-family,Values=m4 --max-duration 31536000
   ```

   コマンドは、条件に合ったオファリングのリストを返します。購入するオファーの `offeringId` を書き留めます。

2. purchase-host-reservation AWS CLI コマンドを使用してオファリングを購入し、前のステップで書き留めた `offeringId` を指定します。次の例では、指定された予約を購入して、AWS アカウントに割り当て済みの特定の Dedicated Host に関連付けます。さらに、キーが `purpose` で値が `production` のタグを、購入した予約に対し適用します。

   ```
   aws ec2 purchase-host-reservation --offering-id hro-03f707bf363b6b324 --host-id-set h-013abcd2a00cbd123 --tag-specifications 'ResourceType=host-reservation,Tags={Key=purpose,Value=production}'
   ```

PowerShell

予約を購入するには

1. Get-EC2HostReservationOffering AWS Tools for Windows PowerShell コマンドを使用して、ニーズに合った利用可能なオファリングを一覧表示します。以下の例では、m4 インスタンスファミリーでインスタンスをサポートし、1 年契約を持っているオファーをリストします。

> **Note**
>
> 期間は秒単位で指定されます。1 年契約は 31,536,000 秒で、3 年契約は 94,608,000 秒です。

```
PS C:\> $filter = @{Name="instance-family"; Value="m4"}
```

```
PS C:\> Get-EC2HostReservationOffering -filter $filter -MaxDuration 31536000
```

コマンドは、条件に合ったオファリングのリストを返します。購入するオファーの `offeringId` を書き留めます。

2. New-EC2HostReservation AWS Tools for Windows PowerShell コマンドを使用してオファリングを購入し、前のステップで書き留めた `offeringId` を指定します。次の例では、指定した予約を購入し、それを AWS アカウントに割り当て済みの特定の Dedicated Host と関連付けます。

```
PS C:\> New-EC2HostReservation -OfferingId hro-03f707bf363b6b324 -
HostIdSet h-013abcd2a00cbd123
```

Dedicated Host 予約の表示

予約に関連する Dedicated Hosts の情報として以下を表示できます。

- 予約の期間
- 支払いオプション
- 開始日と終了日

Dedicated Host 予約の詳細は、次の方法で表示できます。

Console

Dedicated Host 予約の詳細を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Dedicated Hosts] を選択します。
3. [Dedicated Hosts] ページで、[Dedicated Host の予約] を選択し、表示されるリストから予約を選択します。
4. 予約の詳細については、[詳細] を選択します。
5. 予約が関連付けられている Dedicated Hosts に関する情報については、[Hosts (ホスト)] を選択します。

AWS CLI

Dedicated Host 予約の詳細を表示するには

describe-host-reservations AWS CLI コマンドを使用します。

```
aws ec2 describe-host-reservations
```

PowerShell

Dedicated Host 予約の詳細を表示するには

Get-EC2HostReservation AWS Tools for Windows PowerShell コマンドを使用します。

```
PS C:\> Get-EC2HostReservation
```

タグ Dedicated Host の予約

Dedicated Host の予約 にカスタムタグを割り当てて、目的、所有者、環境など、さまざまな方法で分類できます。これにより、割り当てたカスタムタグに基づいて特定の Dedicated Host の予約 をすばやく見つけることができます。

Dedicated Host の予約 にタグを付けるには、コマンドラインツールのみを使用できます。

AWS CLI

Dedicated Host の予約 にタグを付けるには

create-tags AWS CLI コマンドを使用します。

```
aws ec2 create-tags --resources hr-1234563a4ffc669ae --tags Key=Owner,Value=TeamA
```

PowerShell

Dedicated Host の予約 にタグを付けるには

New-EC2Tag AWS Tools for Windows PowerShell コマンドを使用します。

New-EC2Tag コマンドには、Dedicated Host の予約 のタグに使用するキーと値のペアを指定する Tag パラメータが必要です。以下のコマンドでは、Tag パラメータを作成します。

```
PS C:\> $tag = New-Object Amazon.EC2.Model.Tag
PS C:\> $tag.Key = "Owner"
PS C:\> $tag.Value = "TeamA"
```

```
PS C:\> New-EC2Tag -Resource hr-1234563a4ffc669ae -Tag $tag
```

共有 Dedicated Hosts の操作

Dedicated Host の共有を使用すると、Dedicated Host の所有者は Dedicated Host を他の AWS アカウントと共有したり、AWS 組織内で共有したりできます。これにより、Dedicated Hosts の作成と管理を一元的に行い、複数の AWS アカウント間や AWS 組織内で共有することが可能になります。

このモデルでは、Dedicated Host を所有する AWS アカウント (所有者) が、Dedicated Host を他の AWS アカウント (コンシューマー) と共有します。コンシューマーは、各自のアカウントに割り当てた Dedicated Hosts にインスタンスを作成する場合と同じように、共有している Dedicated Hosts にインスタンスを作成できます。所有者は、Dedicated Host およびそこに作成したインスタンスの管理に責任を負います。所有者は、コンシューマーが共有 Dedicated Hosts に作成したインスタンスを変更することはできません。コンシューマーは、自己が共有している Dedicated Hosts に作成したインスタンスの管理に責任を負います。コンシューマーは、他のコンシューマーまたは Dedicated Host 所有者が所有するインスタンスを表示または変更することはできません。また、自己が共有している Dedicated Hosts を変更することはできません。

Dedicated Host 所有者が Dedicated Host を共有できる相手は次のとおりです。

- AWS の組織内または組織外の特定の AWS アカウント
- AWS 組織内の組織単位
- AWS 組織全体

コンテンツ

Dedicated Hosts を共有するための前提条件

- Dedicated Host を共有するには、それを自分の AWS アカウント内で所有している必要があります。既に共有している Dedicated Host を共有することはできません。
- AWS 組織や AWS 組織内の組織単位との間で Dedicated Host を共有するには、AWS Organizations で共有を有効にする必要があります。詳細については、AWS Organizations ユーザーガイドの「AWS RAM で共有を有効化する」を参照してください。

Dedicated Host の共有に関する制限事項

`u-6tb1.metal`、`u-9tb1.metal`、`u-12tb1.metal`、`u-18tb1.metal`、および `u-24tb1.metal` のインスタンスタイプに割り当てられた Dedicated Hosts は共有できません

関連サービス

AWS Resource Access Manager

Dedicated Host の共有は AWS Resource Access Manager (AWS RAM) と統合されています。AWS RAM は、任意の AWS アカウントに対して、あるいは AWS 経由で AWS Organizations リソースを共有するためのサービスです。AWS RAM を使用すると、リソース共有を作成することで、自身が所有するリソースを共有できます。リソース共有では、共有対象のリソースと、共有先となるコン

シューマーを指定します。コンシューマーには、個人の AWS アカウントや、AWS Organizations 内の組織単位または組織全体を指定できます。

AWS RAM の詳細については、AWS RAM ユーザーガイドを参照してください。

アベイラビリティーゾーン間での共有

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 アカウントの名前に個別にマッピングされます。このため、アカウントが異なると、アベイラビリティーゾーンの命名方法が異なる場合があります。例えば、us-east-1a アカウントのアベイラビリティーゾーン AWS の場所は、別の us-east-1a アカウントのアベイラビリティーゾーン AWS の場所と異なる可能性があります。

自己のアカウントを基準にして Dedicated Hosts の場所を特定するには、アベイラビリティーゾーン ID (AZ ID) を使用する必要があります。アベイラビリティーゾーン ID は、すべての AWS アカウントにわたって各アベイラビリティーゾーンを一意に識別する ID です。例えば、use1-az1 は us-east-1 リージョンのアベイラビリティーゾーン ID であり、すべての AWS アカウントで同じ場所を示します。

アカウントのアベイラビリティーゾーンのアベイラビリティーゾーン ID を表示するには

1. AWS RAM コンソール (https://console.aws.amazon.com/ram) を開きます。
2. 現在のリージョンのアベイラビリティーゾーン ID は、画面の右側にある [お客様の AZ ID] パネルに表示されます。

Dedicated Host の共有

所有者が Dedicated Host を共有すると、コンシューマーはそのホストにインスタンスを作成できます。コンシューマーは、共有ホストに空き容量がある限り、そこに必要なだけのインスタンスを作成できます。

> **Important**
>
> Dedicated Hosts で BYOL ライセンスを共有するための適切なライセンス権限があることを確認する責任があります。

自動配置を有効にして Dedicated Host を共有する場合は、意図しない形で Dedicated Host が使用されないよう、次の点に注意してください。

- コンシューマーが Dedicated Host テナンシーを使用してインスタンスを作成する場合、自己のアカウントで所有している Dedicated Host に空き容量がないと、インスタンスは自動的に共有 Dedicated Host に作成されます。

Dedicated Host を共有するには、それをリソース共有に追加する必要があります。リソース共有とは、自身のリソースを AWS RAM アカウント間で共有するための AWS リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。Dedicated Host は、既存のリソースに追加することも、新しいリソース共有に追加することもできます。

自分が AWS Organizations 内の組織のメンバーであり、所属する組織で共有が有効化されている場合には、自分の組織内のコンシューマーに対し、共有された Dedicated Host に対するアクセス許可を自動的に付与することができます。それ以外の場合、コンシューマーはリソース共有への参加の招待を受け取り、その招待を受け入れた後で、共有 Dedicated Host へのアクセス許可が付与されます。

> **Note**
>
> Dedicated Host を共有した場合、コンシューマーがそれにアクセスできるまでに数分かかることがあります。

自己所有の Dedicated Host を共有するには、次のいずれかの方法を使用できます。

Amazon EC2 console

Amazon EC2 コンソールを使用して自己所有の Dedicated Host を共有するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. 共有する Dedicated Host を選択し、[アクション]、[ホストの共有] の順にクリックします。
4. Dedicated Host の追加先のリソース共有を選択し、[ホストの共有] をクリックします。

   コンシューマーから共有ホストにアクセスできるまでに、数分かかることがあります。

AWS RAM console

AWS RAM コンソールを使用して、自分が所有する Dedicated Hosts を共有するには

「AWS RAM ユーザーガイド」の「リソース共有の作成」を参照してください。

AWS CLI

AWS CLI を使用して、自分が所有する Dedicated Hosts を共有するには

create-resource-share コマンドを使用します。

共有 Dedicated Host の共有解除

Dedicated Host 所有者は、共有 Dedicated Host をいつでも共有解除できます。共有 Dedicated Host を共有解除する場合、以下のルールが適用されます。

- Dedicated Host を共有しているコンシューマーは、そこに新しいインスタンスを作成できなくなります。
- 共有解除時に Dedicated Host で実行されていたコンシューマー所有のインスタンスは、引き続き実行されますが、リタイアが予定されます。コンシューマーは、インスタンスのリタイア通知を受け取り、2 週間以内に通知に対処します。ただし、リタイア通知期間内に Dedicated Host がコンシューマーに再共有されると、インスタンスのリタイアはキャンセルされます。

自己所有の共有 Dedicated Host を共有解除するには、それをリソース共有から削除する必要があります。これを行うには、次のいずれかの方法を使用します。

Amazon EC2 console

Amazon EC2 コンソールを使用して、自己所有の共有 Dedicated Host を共有解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. 共有解除する Dedicated Host を選択し、[共有] タブをクリックします。
4. [共有] タブに、Dedicated Host の追加先のリソース共有が一覧表示されます。Dedicated Host を削除する対象のリソース共有を選択し、[リソース共有からホストを削除] をクリックします。

AWS RAM console

AWS RAM コンソールを使用して、自分が所有する共有済みの Dedicated Hosts で共有を解除するには

「AWS RAM ユーザーガイド」の「リソース共有の更新」を参照してください。

Command line

AWS CLI を使用して、自分が所有する共有済みの Dedicated Hosts で共有を解除するには

disassociate-resource-share コマンドを使用します。

共有 Dedicated Host の特定

所有者とコンシューマーは、次のいずれかの方法を使用して共有 Dedicated Hosts を特定できます。

Amazon EC2 console

Amazon EC2 コンソールを使用して共有 Dedicated Host を特定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。この画面には、自己が所有している Dedicated Hosts と共有している Dedicated Hosts が一覧表示されます。[所有者] 列には、Dedicated Host の所有者の AWS アカウント ID が表示されます。

Command line

AWS CLI を使用して共有済みの Dedicated Hosts を特定するには

describe-hosts コマンドを使用します。このコマンドは、自己が所有している Dedicated Hosts と共有している Dedicated Hosts を返します。

共有 Dedicated Host で実行されているインスタンスの表示

所有者とコンシューマーは、次のいずれかの方法を使用して、共有 Dedicated Host で実行されているインスタンスをいつでも表示できます。

Amazon EC2 console

Amazon EC2 コンソールを使用して共有 Dedicated Host で実行されているインスタンスを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. インスタンスを表示する対象の Dedicated Host を選択し、[インスタンス] を選択します。ホストで実行されているインスタンスがタブに一覧表示されます。所有者は、コンシューマー

によって作成されたインスタンスも含めて、ホストで実行されているすべてのインスタンスを表示できます。コンシューマーは、ホストで実行されている自己作成のインスタンスのみを表示できます。[所有者] 列には、インスタンスを起動した AWS アカウントのアカウント ID が表示されます。

Command line

AWS CLI を使用して共有済みの Dedicated Hosts で実行されているインスタンスを表示するには

describe-hosts コマンドを使用します。このコマンドは、各 Dedicated Host で実行されているインスタンスを返します。所有者は、ホストで実行されているすべてのインスタンスを表示できます。コンシューマーは、共有ホストで自身が起動し実行中のインスタンスのみを表示できます。`InstanceOwnerId` は、インスタンス所有者の AWS アカウント ID を示します。

共有 Dedicated Host のアクセス許可

所有者のアクセス許可

所有者は、共有 Dedicated Hosts およびそこに作成したインスタンスの管理に責任を負います。所有者は、コンシューマーによって作成されたインスタンスも含めて、共有 Dedicated Host で実行されているすべてのインスタンスを表示できます。ただし、所有者は、コンシューマーによって作成された実行中のインスタンスに対してアクションを実行することはできません。

コンシューマーのアクセス許可

コンシューマーは、共有 Dedicated Host に作成したインスタンスの管理に責任を負います。コンシューマーは、共有 Dedicated Host を一切変更できません。また、他のコンシューマーや Dedicated Host 所有者が作成したインスタンスを表示または変更することもできません。

請求と使用量測定

Dedicated Hosts の共有に追加料金はかかりません。

所有者は、自己が共有する Dedicated Hosts に対して課金されます。コンシューマーは、共有 Dedicated Hosts に作成したインスタンスに対して課金されません。

Dedicated Host の予約 は、共有 Dedicated Hosts に対して引き続き請求割引を提供します。Dedicated Host 所有者のみが、自己が所有する共有 Dedicated Hosts 用の Dedicated Host の予約 を購入できます。

Dedicated Host の制限

共有 Dedicated Hosts は、所有者の Dedicated Hosts 制限に対してのみカウントされます。共有 Dedicated Hosts は、コンシューマーの Dedicated Hosts 制限に対してはカウントされません。同様に、コンシューマーが共有 Dedicated Hosts に作成するインスタンスは、コンシューマーのインスタンス制限に対してカウントされません。

ホストの復旧と Dedicated Host の共有

ホストの復旧は、Dedicated Host の 所有者とその共有相手のコンシューマーによって作成されたインスタンスを復旧します。代替 Dedicated Host は所有者のアカウントに割り当てられます。元の Dedicated Host と同じリソース共有に追加され、同じコンシューマと共有されます。

詳細については、「ホスト復旧」を参照してください。

AWS Outposts での Dedicated Hosts

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをユーザーのオンプレミスまで拡張するフルマネージドサービスです。AWS Outposts は、AWS 管理インフラストラクチャへのローカルアクセスを提供することにより、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築および実行できると同時に、ローカルコンピューティングおよびストレージリソースを使用して、レイテンシとローカルデータ処理のニーズを低減します。

Outpost とは、お客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS は、AWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。

アカウントで所有している Outposts に Dedicated Hosts を割り当てることができます。これにより、専用の物理サーバーを必要とする既存のソフトウェアライセンスとワークロードを AWS Outposts に簡単に持ち込むことができます。Outpost の特定のハードウェアアセットをターゲットにして、ワークロード間のレイテンシーを最小限に抑えることもできます。

Dedicated Hosts を使用すると、Amazon EC2 で適格なソフトウェアライセンスを使用できるため、独自のライセンスを使用する場合の柔軟性と費用対効果が得られます。仮想マシン、ソケット、または物理コアにバインドされている他のソフトウェアライセンスも、ライセンス条項に従って、Dedicated Hosts で使用できます。Outpost は常に BYOL ワークロードに適格なシングルテナント環境でしたが、Dedicated Hosts を使用すると、Outpost の展開全体ではなく、必要なライセンスを単一のホストにデプロイできます。

さらに、Outpostで Dedicated Hosts を使用すると、インスタンスタイプのデプロイの柔軟性が高まり、インスタンスの配置をより細かく制御できます。インスタンスの起動に特定のホストをターゲットにして、ホストアフィニティを使用して、インスタンスが常にそのホストで実行されるようにするか、自動配置を使用して、設定と使用可能な容量が一致する使用可能なホストにインスタンスを起動できます。

目次

前提条件

Outpost は、自分のサイトにインストールする必要があります。詳細については、AWS Outposts ユーザーガイドの「Outpost を作成し、Outpost 容量を注文する」を参照してください。

サポートされている機能

- インスタンスファミリーC5、M5、R5、C5d、M5d、R5d、G4dn、i3enがサポートされています。
- Outposts での Dedicated Hosts は、複数のインスタンスサイズをサポートするように設定できます。複数のインスタンスサイズに対するサポートはインスタンスファミリーC5、M5、R5、C5d、M5d、R5dで利用できます。詳細については、インスタンスキャパシティの設定を参照してください。
- Outposts での Dedicated Hosts は、自動配置とターゲットインスタンスの起動をサポートします。詳細については、「自動配置とアフィニティについて」を参照してください。
- Outposts での Dedicated Hosts は、ホストアフィニティをサポートします。詳細については、「自動配置とアフィニティについて」を参照してください。
- Outposts での Dedicated Hosts は、AWS RAM との共有をサポートしています。詳細については、「共有 Dedicated Hosts の操作」を参照してください。

考慮事項

- 専有ホスト予約は Outpost ではサポートされていません。
- ホストリソースグループと AWS License Manager は、Outposts ではサポートされていません。

- Outposts 上の Dedicated Hosts は、バースト可能な T3 インスタンスをサポートしていません。
- Outposts の Dedicated Hosts は、ホストの回復をサポートしていません。
- Outposts の専有ホストテナンシーを使用するインスタンスでは、簡易自動復旧はサポートされていません。

Outpost で専有ホストを割り当てて使用する

AWS リージョンの Dedicated Hosts の場合と同じ方法で、Outpost に Dedicated Hosts を割り当てて使用します。

前提条件

Outpost にサブネットを作成します。詳細については、AWS Outposts ユーザーガイドの「サブネットの作成」を参照してください。

Outpost に専有ホストを割り当てるには、次のいずれかの方法を使用します。

AWS Outposts console

1. AWS Outposts コンソール (https://console.aws.amazon.com/outposts/) を開きます。
2. ナビゲーションペインで、[Outpost] を選択します。Outpost を選択し、[Actions] (アクション)、[Allocate Dedicated Host] (専有ホストの割り当て) を選択します。
3. 必要に応じて専有ホストを設定します。詳細については、Dedicated Hosts の割り当てを参照してください。

   > **Note**
   >
   > [アベイラビリティーゾーン] と [Outpost ARN]には、選択した Outpost のアベイラビリティーゾーンと ARN をあらかじめ組み込んでおく必要があります。

4. [割り当て] を選択します。

Amazon EC2 console

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインにおいて、[Dedicated Hosts] (専有ホスト) を選択してから、[Allocate Dedicated Host] (専有ホスト割り当て) を選択します。

3. [アベイラビリティーゾーン] には、Outpost に関連付けられているアベイラビリティーゾーンを選択します。
4. Outpost ARN には、Outpost の ARN を入力します。
5. Outpost の特定のハードウェアアセットをターゲットにするには、[Outpost の特定のハードウェアアセットをターゲットにする] で [有効] を選択します。ターゲットにする各ハードウェアアセットについて、[アセット ID を追加] を選択し、ハードウェアアセットの ID を入力します。

> **Note**
>
> [数量] に指定する値は、指定するアセット ID の数と等しくなければなりません。例えば、3 つのアセット ID を指定する場合、数量も 3 でなければなりません。

6. 必要に応じて、残りの専有ホストを設定します。詳細については、Dedicated Hosts の割り当てを参照してください。
7. [割り当て] を選択します。

AWS CLI

allocate-hostsAWS CLIコマンドを使用します。`--availability-zone` には、Outpost に関連付けられているアベイラビリティーゾーンを指定します。`--outpost-arn` には Outpost の ARN を指定します。オプションで、ターゲットにする Outpost ハードウェアアセットの ID を `--asset-ids` に指定します。

```
aws ec2 allocate-hosts --availability-zone "us-east-1a" --outpost-arn
 "arn:aws:outposts:us-east-1a:111122223333:outpost/op-4fe3dc21baEXAMPLE" --asset-
ids asset_id --instance-family "m5" --auto-placement "off" --quantity 1
```

Outpost 上の専有ホストでインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。前のステップで割り当てた専有ホストを選択し、[Actions] (アクション)、[Launch instance onto host] (ホストへのインスタンスの起動) を選択します。
3. 必要に応じてインスタンスを設定してから、インスタンスを起動します。詳細については、「Dedicated Host でのインスタンスの起動」を参照してください。

ホスト復旧

Dedicated Hosts 自動リカバリでは、Dedicated Hosts で特定の問題が検出されると、インスタンスが新しい代替ホストで再起動されます。ホスト復旧により、システム電源や Dedicated Hosts でネットワーク接続に関する予期せぬ障害が発生した場合に、手動による介入の必要性を減らし、運用の負担を軽減します。その他の Dedicated Hosts の問題は、手動での復旧が必要となります。

コンテンツ

ホスト復旧の基本

Dedicated Hosts とホスト Resource Groups のリカバリプロセスは、Dedicated Hosts の可用性を評価し、根本的なシステム障害を検出するために、ホストレベルのヘルスチェックを使用します。Dedicated Hosts の自動リカバリが可能かどうかは、Dedicated Hosts の障害の種類によって決まります。ホストレベルのヘルスチェックが失敗する場合、原因として以下のような問題が考えられます。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのハードウェアまたはソフトウェアの問題

> **Important**
>
> ホストのリタイアが予定されている場合、専有ホストの自動復旧は発生しません。

Dedicated Hosts 自動リカバリ

Dedicated Hosts でシステム電源やネットワーク接続の障害が検出されると、Dedicated Hosts の自動リカバリが開始され、Amazon EC2 は自動的に代替の Dedicated Hosts を割り当てます。代替 Dedicated Host は新しいホスト ID を受け取りますが、元の Dedicated Host と同じ以下の属性を保持します。

- アベイラビリティーゾーン
- インスタンスタイプ
- タグ
- 自動プレイスメントの設定
- 予約する

代替の Dedicated Hosts が割り当てられると、インスタンスは代替 Dedicated Hosts に復旧されます。復旧されたインスタンスは、元のインスタンスと同じ以下の属性を保持します。

- インスタンス ID
- プライベート IP アドレス
- Elastic IP アドレス
- EBS ボリュームアタッチメント
- すべてのインスタンスメタデータ

また、組み込まれている AWS License Manager との統合により、ライセンスの追跡と管理が自動的に行われます。

> **Note**
>
> AWS License Manager との統合は、AWS License Manager を利用できるリージョンでのみサポートされます。

インスタンスと障害が発生した Dedicated Host との間にホストのアフィニティがある場合、復旧したインスタンスは代替 Dedicated Host との間にホストのアフィニティを確立します。

すべてのインスタンスが代替専有ホストに復旧されると、障害が発生した専有ホストがリリースされて、代替専有ホストが使用可能になります。

ホスト復旧が開始されると、AWS アカウントの所有者に対してメールまたは AWS Health Dashboard イベントで通知されます。ホスト復旧が正常に完了すると、2 つ目の通知が送信されます。

AWS License Manager を使用してライセンスを追跡している場合は、その AWS License Manager により、ライセンス設定の制限に応じて新しいライセンスが代替 Dedicated Hosts に割り当てられます。ホスト復旧の結果としてライセンス設定のハード制限を超える場合、復旧プロセスは許可されず、Amazon SNS 通知を介してホスト復旧の失敗が通知されます (AWS License Manager の通知設定が構成されている場合)。ホスト復旧の結果としてライセンス設定のソフト制限を超える場合は、復旧の続行が許可され、Amazon SNS 通知を介して制限の超過が通知されます。詳細については、「AWS License Manager ユーザーガイド」の「ライセンス設定の使用」および「License Manager の設定」を参照してください。

Dedicated Hosts の自動リカバリがないシナリオ

ホストのリタイアが予定されている場合、専有ホストの自動復旧は発生しません Amazon CloudWatch Events に関する使用停止通知が AWS Health Dashboard に送信され、AWS アカウント所有者のメールアドレスには Dedicated Hosts の障害に関するメッセージが届きます。指定された期間内にリタイア通知に記載されている修復手順に従って、リタイアするホストのインスタンスを手動で復旧します。

停止したインスタンスは代替 Dedicated Host に復旧されません。障害が発生した Dedicated Host を対象とする、停止したインスタンスを起動しようとすると、インスタンスの起動は失敗します。停止したインスタンスを変更して別の Dedicated Host を対象とするか、停止したインスタンスと一致する設定を持ち、自動配置が有効になっている Dedicated Host で起動することをお勧めします。

インスタンスストレージのあるインスタンスは代替 Dedicated Host に復旧されません。是正措置として、障害が発生した Dedicated Host はリタイアとしてマークされ、ホスト復旧の完了後にリタイア通知が送信されます。リタイア通知に記載されている是正措置に従い、指定された期間内に障害が発生した Dedicated Host の残りのインスタンスを手動で復旧します。

サポートされるインスタンスタイプ

ホスト復旧は、インスタンスファミリー
A1、C3、C4、C5、C5n、C6a、C6g、C6i、Inf1、G2、G3、G5g、M3、M4、M5、M5n、M5zn、M6a、M6
に対してサポートされています。

サポートされていないインスタンスを復旧するには、「サポートされていないインスタンスの手動復旧」を参照してください。

> **Note**
>
> サポートされているメタルインスタンスタイプの Dedicated Hosts 自動リカバリは、非メタルインスタンスタイプよりも検出および復旧に時間がかかります。

ホスト復旧の設定

ホスト復旧は、Dedicated Hosts の割り当て時に設定することも、割り当て後に Amazon EC2 コンソールまたは AWS Command Line Interface (CLI) を使用して設定することもできます。

コンテンツ

- ホスト復旧の有効化
- ホスト復旧の無効化
- ホスト復旧の設定の表示

ホスト復旧の有効化

ホスト復旧は、Dedicated Host の割り当て時または割り当て後に有効にすることができます。

ホスト復旧を Dedicated Host の割り当て時に有効にする方法の詳細については、「Dedicated Hosts の割り当て」を参照してください。

ホスト復旧を割り当て後に有効にするには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. ホスト復旧を有効にする Dedicated Host を選択し、[Actions (アクション)]、[Modify Host Recovery (ホスト復旧の変更)] の順に選択します。
4. [Host recovery (ホスト復旧)] で、[Enable (有効化)]、[Save (保存)] の順に選択します。

ホスト復旧を割り当て後に有効にするには (AWS CLI)

modify-hosts コマンドを使用して `host-recovery` パラメータを指定します。

```
$ aws ec2 modify-hosts --host-recovery on --host-ids h-012a3456b7890cdef
```

ホスト復旧の無効化

ホスト復旧は Dedicated Host の割り当て後にいつでも無効にすることができます。

ホスト復旧を割り当て後に無効にするには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. ホスト復旧を無効にする Dedicated Host を選択し、[Actions (アクション)]、[Modify Host Recovery (ホスト復旧の変更)] の順に選択します。
4. [Host recovery (ホスト復旧)] で、[Disable (無効化)]、[Save (保存)] の順に選択します。

ホスト復旧を割り当て後に無効にするには (AWS CLI)

modify-hosts コマンドを使用して `host-recovery` パラメータを指定します。

```
$ aws ec2 modify-hosts --host-recovery off --host-ids h-012a3456b7890cdef
```

ホスト復旧の設定の表示

Dedicated Host のホスト復旧の設定はいつでも表示できます。

Dedicated Host のホスト復旧の設定を表示するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. Dedicated Host を選択し、[Description (説明)] タブの [Host Recovery (ホスト復旧)] フィールドを確認します。

AWS CLI を使用して Dedicated Hosts のホスト復旧の設定を表示するには

describe-hosts コマンドを使用します。

```
$ aws ec2 describe-hosts --host-ids h-012a3456b7890cdef
```

`HostRecovery` レスポンス要素に、ホスト復旧が有効であるか無効であるかが示されます。

ホスト復旧の状態

Dedicated Host の障害が検出されると、障害が発生した Dedicated Host は `under-assessment` 状態になり、すべてのインスタンスは `impaired` 状態になります。障害が起きているDedicated Hostが`under-assessment`状態の間は、このホストでインスタンスを起動できません。

代替 Dedicated Host が割り当てられると、この代替ホストは `pending` 状態になります。ホスト復旧プロセスが完了するまでは、この状態に留まります。代替Dedicated Hostが`pending`状態の間は、このホストでインスタンスを起動できません。代替 Dedicated Host に復旧されたインスタンスは、復旧プロセス中、`impaired` 状態に留まります。

ホスト復旧が完了すると、代替 Dedicated Host は `available` 状態になり、復旧されたインスタンスは `running` 状態に戻ります。代替 Dedicated Host が `available` 状態になると、このホストでインスタンスを起動できます。障害が発生した元の専有ホストは完全にリリースされ、`released-permanent-failure` 状態になります。

障害が発生した専有ホストにホスト復旧をサポートしていないインスタンス (instance store-backed ボリュームのインスタンスなど) がある場合、専有ホストはリリースされません。代わりに、そのホストはリタイアとしてマークされ、`permanent-failure` 状態になります。

サポートされていないインスタンスの手動復旧

ホスト復旧は、インスタンスストアボリュームを使用するインスタンスの復旧をサポートしていません。自動的に復旧されないインスタンスがある場合は、以下の手順に従って、これらのインスタンスを手動で復旧します。

> **Warning**
>
> インスタンスストアボリュームのデータは、インスタンスの停止、休止、または終了に伴って失われます。これには、EBS ボリュームをルートデバイスとするインスタンスにアタッチされたインスタンスストアボリュームも含まれます。インスタンスストアボリュームのデータを保護するには、インスタンスが停止または終了する前に、データを永続的ストレージにバックアップします。

EBS-backed インスタンスの手動復旧

自動的に復旧されない EBS-backed インスタンスの場合は、インスタンスを手動で停止または終了させて、新しい Dedicated Host に復旧することをお勧めします。インスタンスの停止や、インスタ

ンスの停止に伴うインスタンス設定の変更の詳細については、「インスタンスの停止と起動」を参照してください。

instance store-backed インスタンスの手動復旧

自動的に復旧されない instance store-backed インスタンスの場合は、以下の操作を行うことをお勧めします。

1. 新しい Dedicated Host で、最新の AMI から代替インスタンスを起動します。
2. すべての必要なデータを代替インスタンスに移行させます。
3. 障害が発生した Dedicated Host で元のインスタンスを終了します。

関連サービス

Dedicated Host は以下のサービスと統合します。

- AWS License Manager – Amazon EC2 Dedicated Hosts 全体でライセンスを追跡します (AWS License Manager が利用可能なリージョンでのみサポートされます)。詳細については、「AWS License Manager ユーザーガイド」を参照してください。

料金

ホスト復旧の使用に伴う追加の料金はありません。通常の Dedicated Host 料金が適用されます。詳細については、「Amazon EC2 Dedicated Hosts 料金」を参照してください。

ホスト復旧が開始されると同時に、障害が発生した Dedicated Host には課金されなくなります。代替の専有ホストに対する課金は、専有ホストが `available` 状態になった後でのみ開始されます。

障害が発生した Dedicated Host の課金にオンデマンド料金が使用されていた場合は、代替の Dedicated Host の課金にもオンデマンド料金が使用されます。障害が発生した Dedicated Host でアクティブになっていた Dedicated Host の予約 は、代替の Dedicated Host に転送されます。

ホストのメンテナンス

ホストのメンテナンスでは、スケジュールされたメンテナンスイベント中に、パフォーマンスが低下した専有ホスト上の Amazon EC2 インスタンスが、代替の専有ホストで自動的に再起動されます。これにより、アプリケーションのダウンタイムが減少し、AWS のメンテナンスという面倒な作業が軽減されます。ホストのメンテナンスは、Amazon EC2 の計画的かつ日常的なメンテナンスのためにも行われます。

ホストのメンテナンスは、Amazon EC2 コンソールから行われたすべての新しい専有ホスト割り当てでサポートされます。お客様の AWS アカウント の専有ホストまたは AllocateHosts API を介して割り当てられた新しい任意の専有ホストでは、サポートされている Dedicated Hosts に対し、ホストのメンテナンスを設定できます。詳細については、「the section called “ホストメンテナンスの設定”」を参照してください。

コンテンツ

ホストメンテナンスの基本

専有ホストでパフォーマンスの低下が検出されると、新しい専有ホストが割り当てられます。パフォーマンスの低下は、基盤となるハードウェアの劣化、または特定の問題のある状態の検出によって引き起こされる可能性があります。パフォーマンスの低下された専有ホストのインスタンスは、代替の専有ホストで自動的に再起動されるようにスケジュールされます。

代替の専有ホストは新しいホスト ID を受け取りますが、元の専有ホストと同じ属性を保持します。これらの属性には、次のようなものが含まれます。

- 自動プレイスメントの設定
- アベイラビリティーゾーン
- 予約する
- ホストのアフィニティ
- ホストのメンテナンス設定
- ホストの復旧設定
- インスタンスタイプ
- タグ

ホストのメンテナンスは、サポートされているすべての AWS リージョン の専有ホストで利用できます。ホストのメンテナンスがサポートされていない専有ホストの詳細については、「the section called “制限事項”」を参照してください。

パフォーマンスが低下した専有ホストは、すべてのインスタンスを新しい専有ホストで再起動するか、停止した後にリリースされます。予定されているメンテナンスイベントの前に、デグレードした専有ホストのインスタンスにアクセスできますが、デグレードした専有ホストでのインスタンスの起動はサポートされていません。

予定されているメンテナンスイベントの前に、代替の専有ホストを使用してホスト上で新しいインスタンスを起動できます。ただし、代替ホストの一部のインスタンス容量は、デグレードしたホストから移行する必要があるインスタンス用に予約されています。このリザーブドキャパシティで新しいインスタンスを起動することはできません。詳細については、「the section called “専有ホストでのインスタンス”」を参照してください。

制限事項

- ホストのメンテナンスは、AWS Outposts、AWS ローカルゾーンと AWS Wavelength ゾーンではサポートされていません。
- ホストリソースグループ内に既に含まれているホストについては、ホストメンテナンスをオンまたはオフにすることはできません。ホストリソースグループに追加されたホストは、そのホストメンテナンス設定を保持します。詳細については、「ホストリソースグループ」を参照してください。
- ホストのメンテナンスは特定のインスタンスタイプでのみサポートされます。詳細については、「the section called “サポートされるインスタンスタイプ”」を参照してください。

ホストメンテナンスとホスト復旧

次の表は、ホストメンテナンスとホスト復旧の主な違いを示しています。

	ホスト復旧	ホストのメンテナンス
アクセシビリティ	到達不能	到達可能
都道府県	`under-assessment`	`permanent-failure`
アクション	即時に復旧されます	メンテナンスが予定されています
スケジューリングの柔軟性	再スケジュール不可	再スケジュール可能

	ホスト復旧	ホストのメンテナンス
リソースグループをホストします	サポート	サポートされていません

ホスト復旧の詳細については、「ホスト復旧」を参照してください。

サポートされるインスタンスタイプ

ホストメンテナンスは以下のインスタンスファミリーでサポートされています。

- 汎用: A1 | M4 | M5 | M5a | M5n | M5zn | M6a | M6g | M6i | M6in | M7a | M7g | M7i | T3
- コンピューティングの最適化: C4 | C5 | C5a | C5n | C6a | C6g | C6gn | C6i | C6in | C7g | C7gn | C7i
- メモリ最適化: R4 | R5 | R5a | R5b | R5n | R6a | R6g | R6i | R6in | R7a | R7g | R7iz | u-12tb1 | u-18tb1 | u-24tb1 | u-3tb1 | u-6tb1 | u-9tb1 | X2iezn
- 高速コンピューティング: G3 | G5g | Inf1 | P2 | P3

専有ホストでのインスタンス

Amazon EC2 は、デグレードしたホストから自動的に移行されるインスタンスの代替ホストの容量を自動的に予約します。Amazon EC2 は、インスタンスストアのルートボリュームを持つインスタンスなど、自動的に移行できないインスタンスの代替ホストの容量は予約しません。リザーブドキャパシティは、新しいインスタンスの起動には使用できません。

> **Note**
>
> Amazon EC2 コンソールには、リザーブドキャパシティが使用済みキャパシティとして表示されます。インスタンスは、デグレードしたホストと代替ホストの両方で実行されているように見える場合があります。ただし、インスタンスは、停止するか、代替ホストのリザーブドキャパシティに移行するまで、デグレードしたホストでのみ引き続き実行されます。

自動的に移行できるデグレードしたホスト上のインスタンスを手動で停止すると、代替ホスト上のそのインスタンス用に予約された容量が解放され、使用できるようになります。

スケジュールされたメンテナンスイベント中に、デグレードしたホストのインスタンスは再起動され、代替の専有ホストのリザーブドキャパシティに移行されます。移行したインスタンスは、デグレードしたホスト上のものと同じ以下の属性を保持します。

- Amazon EBS ボリュームアタッチメント
- Elastic IP アドレス
- [インスタンス ID]
- インスタンスメタデータ
- プライベート IP アドレス

スケジュールされたメンテナンスイベントが開始される前であれば、いつでもデグレードしたホストで [インスタンスを停止および起動] できます。これを行うと、インスタンスが別のホストで再起動され、インスタンスは定期メンテナンス受けなくなります。インスタンスのホストアフィニティを、インスタンスを再起動する新しいホストに更新する必要があります。メンテナンスイベントが開始される前にデグレードしたホスト上のすべてのインスタンスを停止すると、デグレードしたホストは解放され、メンテナンスイベントはキャンセルされます。詳細については、「インスタンスの停止と起動」を参照してください。

> **Note**
>
> インスタンスを停止および再開しても、ローカルストアボリュームのデータは保持されません。

[インスタンスストアボリューム] をルートデバイスとするインスタンスは、指定された終了日が過ぎると終了します。インスタンスストアボリューム上のデータは、インスタンスが終了すると、削除されます。終了したインスタンスは完全に削除され、再び起動することはできません。インスタンスストアボリュームをルートデバイスとするインスタンスの場合は、最新の Amazon マシンイメージを使用して別の専有ホストで代替インスタンスを起動し、指定された終了日までに利用可能なすべてのデータを代替インスタンスに移行することをお勧めします。詳細については、「インスタンスの廃止」を参照してください。

自動的に [再起動できない] インスタンスは、指定された日付を過ぎると停止します。これらのインスタンスは、別のホストで再起動できます。Amazon EBS ボリュームをルートデバイスとして使用するインスタンスは、新しいホストで起動した後も同じ Amazon EBS ボリュームを引き続き使用します。

[インスタンスの再起動の順序] は、https://console.aws.amazon.com/ec2/ でインスタンスの再起動の開始時刻を再スケジュールすることで設定できます。

ホストメンテナンスの設定

AWS Management Console または AWS CLI を使用して、サポートされているすべての専有ホストのホストメンテナンスを設定できます。詳細については、以下の表をご参照ください。

AWS Management Console

AWS Management Console を使用して専有ホストのホストメンテナンスを有効にするには。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. [専有ホスト]、[アクション]、[ホストの変更] の順に選択します。
4. [ホストメンテナンス] フィールドで [オン] を選択します。

AWS Management Console を使用して専有ホストのホストメンテナンスを無効にするには。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. [専有ホスト]、[アクション]、[ホストの変更] の順に選択します。
4. [ホストメンテナンス] フィールドで [オフ] を選択します。

AWS Management Console を使用して専有ホストのホストメンテナンスの設定を表示するには。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Dedicated Hosts] を選択します。
3. 専有ホストを選択し、[説明] タブの [ホストメンテナンス] フィールドを確認します。

AWS CLI

AWS CLI を使用して割り当て中の新しい専有ホストについてホストメンテナンスを有効または無効するには。

allocate-hosts コマンドを使用します。

有効

```
aws ec2 allocate-hosts --region us-east-1 --quantity 1 --instance-type m3.large --
availability-zone us-east-1b --host-maintenance on
```

[無効]

```
aws ec2 allocate-hosts --region us-east-1 --quantity 1 --instance-type m3.large --
availability-zone us-east-1b --host-maintenance off
```

AWS CLI を使用して専有ホストのホストメンテナンスを有効または無効するには。

modify-hosts コマンドを使用します。

有効

```
aws ec2 modify-hosts --region us-east-1 --host-maintenance on --host-
ids h-0d123456bbf78910d
```

[無効]

```
aws ec2 modify-hosts --region us-east-1 --host-maintenance off --host-
ids h-0d123456bbf78910d
```

AWS CLI を使用して専有ホストのホストメンテナンスの設定を表示するには。

describe-hosts コマンドを使用します。

```
aws ec2 describe-hosts --region us-east-1 --host-ids h-0d123456bbf78910d
```

> **Note**
>
> ホストのメンテナンスを無効にすると、デグレードしたホストをエビクションし、28 日以内にインスタンスを別のホストに手動で移行するよう求めるメール通知が届きます。専有ホストを予約している場合は、代替ホストが割り当てられます。28 日後、デグレードしたホストで実行されていたインスタンスは終了し、そのホストは自動的にリリースされます。

メンテナンスイベント

デグレードが発生すると、14 日後にメンテナンスイベントがスケジュールされ、新しい専有ホストでインスタンスを再起動します。デグレードしたホスト、予定されているメンテナンスイベント、およびメンテナンスのタイムスロットに関する詳細が記載されたメール通知が届きます。詳細については、「スケジュールされたイベントの表示」を参照してください。

メンテナンスイベントは、予定されているイベントの日付から 7 日後までいつでも再スケジュールできます。再スケジュールの詳細については、「スケジュールされたイベントを再スケジュールする」を参照してください。

通常、メンテナンスイベントの完了までには数分かかります。まれにイベントが失敗した場合は、指定された時間内にデグレードしたホスト上のインスタンスを削除するように求めるメール通知が届きます。

ホストメンテナンスの状態

専有ホストは、デグレードが発生したときの `permanent-failure` の状態に設定されます。`permanent-failure` の状態の専有ホストではインスタンスを起動できません。メンテナンスイベントが完了すると、デグレードしたホストはリリースされ、`released, permanent-failure` の状態になります。

専有ホストのデグレードが検出され、メンテナンスイベントをスケジュールする前に、ホストメンテナンスはアカウントに代替の専有ホストを自動的に割り当てます。この代替ホストは、メンテナンスイベントが予定されるまで `pending` の状態のままになります。メンテナンスイベントがスケジュールされると、代替の専有ホストは `available` の状態に移行します。

予定されているメンテナンスイベントの前に、代替の専有ホストを使用してホスト上で新しいインスタンスを起動できます。ただし、代替ホストの一部のインスタンス容量は、デグレードしたホストから移行する必要があるインスタンス用に予約されています。このリザーブドキャパシティで新しいインスタンスを起動することはできません。詳細については、「the section called “専有ホストでのインスタンス”」を参照してください。

関連サービス

専有ホストと AWS License Manager との統合 - Amazon EC2 Dedicated Hosts 全体でライセンスを追跡します (AWS License Manager が利用可能なリージョンでのみサポートされます)。詳細については、「AWS License Manager ユーザーガイド」を参照してください。

新しい専有ホストには AWS アカウント で十分なライセンスが必要です。予定されているメンテナンスイベントの完了後にホストがリリースされると、デグレードしたホストに関連するライセンスはリリースされます。

料金

ホストメンテナンスの使用に伴う追加の料金はありません。通常の専有ホスト料金が適用されます。詳細については、「Amazon EC2 Dedicated Hosts 料金」を参照してください。

ホストメンテナンスが開始されると同時に、デグレードした専有ホストには課金されなくなります。代替の専有ホストに対する課金は、専有ホストが `available` 状態になった後でのみ開始されます。

デグレードした専有ホストの課金にオンデマンド料金が使用されていた場合は、代替の専有ホストの課金にもオンデマンド料金が使用されます。デグレードした専有ホストでアクティブになっていた専有ホストの予約は、新しい専有ホストに転送されます。

設定の変更の追跡

AWS Config を使用すると、Dedicated Hosts の設定変更や、Dedicated Hosts 上で起動、停止、終了されたインスタンスの設定変更を記録できます。そして、AWS Config でキャプチャされた情報をライセンスレポートのデータソースとして使用することができます。

AWS Config は、Dedicated Hosts やインスタンスの設定情報を個別に記録し、関係を利用してそれぞれの設定情報をペアにします。3 つのレポート条件があります。

- AWS Config の記録ステータス – [オン] のとき、AWS Config は 1 つ以上の AWS リソースタイプを記録中です。記録の対象には、Dedicated Hosts や ハードウェア専有インスタンス も含まれます。ライセンスレポートに必要な情報をキャプチャするには、次のフィールドによって Host とインスタンスが記録されていることを確認します。
- Host recording status — [Enabled] の場合は、Dedicated Hosts の設定情報が記録されます。
- インスタンスの記録ステータス — [Enabled (有効)] の場合は、ハードウェア専有インスタンス の設定情報が記録されます。

これら 3 つの条件のいずれかが無効になっている場合、[Config 記録の編集] ボタン内のアイコンは赤です。このツールのメリットをすべて引き出すために、3 つの記録方法すべてを有効にしてください。3 つすべてが有効なとき、アイコンは緑です。設定を編集するには、[Config 記録の編集] を選択します。AWS Config コンソールに [Set up AWS Config] ページが表示され、そこで AWS Config を設定し、ホスト、インスタンス、およびその他のサポートされるリソースタイプの記録を開始できま

す。詳細については、『AWS Config デベロッパーガイド』の「コンソールを使用した AWS Config のセットアップ」を参照してください。

> **Note**
>
> AWS Config はリソースを検出 (数分かかる場合があります) して、記録します。

AWS Config がホストおよびインスタンスへの設定変更の記録を開始した後、ユーザーが割り当てたかリリースしたホストと、起動、停止、または終了したインスタンスの設定履歴を取得できます。例えば、Dedicated Host の設定履歴の任意の時点で、そのホストのソケット数とコア数と共に、そのホストで起動されているインスタンスの数を調べることができます。それらのインスタンスについても、対応する Amazon マシンイメージ (AMI) の ID を調べることができます。これらの情報を使用して、ソケット単位またはコア単位でライセンスが与えられているサーバーバインドソフトウェアのライセンスに関するレポートを作成できます。

設定履歴は以下のいずれかの方法で閲覧できます。

- AWS Config コンソールを使用する。記録されたリソースごとに、設定の詳細の履歴を提供するタイムラインページを表示することができます。このページを表示するには、[Dedicated Hosts] ページの [設定タイムライン] 列にあるグレーのアイコンを選択します。詳細については、『AWS Config デベロッパーガイド』の「AWS Config コンソールでの設定詳細の表示」を参照してください。
- AWS CLI コマンドを実行する。まず、list-discovered-resources コマンドを使用して、すべてのホストとインスタンスのリストを取得できます。次に、get-resource-config-history コマンドを使用して、特定の時間間隔でホストまたはインスタンスの設定の詳細を取得できます。詳細については、『AWS Config デベロッパーガイド』の「CLI による設定詳細の表示」を参照してください。
- アプリケーションで AWS Config API を使用する。まず、ListDiscoveredResources アクションを使用して、すべてのホストとインスタンスのリストを取得できます。次に、GetResourceConfigHistory アクションを使用して、特定の時間間隔でホストまたはインスタンスの設定の詳細を取得できます。

例えば、AWS Config から Dedicated Hosts のリストを取得するには、次のような CLI コマンドを実行します。

```
aws configservice list-discovered-resources --resource-type AWS::EC2::Host
```

AWS Config から Dedicated Hosts の設定履歴を取得するには、次のような CLI コマンドを実行します。

```
aws configservice get-resource-config-history --resource-type AWS::EC2::Instance --
resource-id i-1234567890abcdef0
```

コンソールを使用して AWS Config の設定を管理するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [Dedicated Hosts] ページで、[Config 記録の編集] を選択します。
3. AWS Config コンソールで、次の手順に従って記録をオンにします。詳細については、「コンソールを使用した AWS Config のセットアップ」を参照してください。

詳細については、「AWS Config コンソールでの設定詳細の表示」を参照してください。

コマンドラインまたは API を使用して AWS Config をアクティブ化するには

- AWS CLI: AWS CLI デベロッパーガイドの「設定詳細の表示 (AWS Config)」
- Amazon EC2 API: 「GetResourceConfigHistory」

Dedicated Instances

デフォルトでは、EC2 インスタンスは共有テナンシーハードウェアで実行されます。つまり、複数の AWS アカウントが同じ物理ハードウェアを共有する可能性があります。

ハードウェア専有インスタンスとは、単一の AWS アカウント専用のハードウェア上で動作する EC2 インスタンスです。つまり、ハードウェア専有インスタンスは、それらのアカウントが単一の支払者アカウントにリンクされている場合でも、他の AWS アカウント に属するインスタンスからホストハードウェアレベルで物理的に分離されています。ただし、ハードウェア専有インスタンスは、同じ AWS アカウント に属する、ハードウェア専有インスタンスではない他のインスタンスとはハードウェアを共有できます。

ハードウェア専有インスタンスは、インスタンスの配置を可視化したり制御したりせず、ホストアフィニティもサポートしません。ハードウェア専有インスタンスを停止して起動すると、同じホストで実行されない場合があります。同様に、インスタンスを起動または実行する特定のホストをターゲットにすることはできません。さらに、ハードウェア専有インスタンスでは、Bring-Your-Own-License (BYOL) のサポートが制限されています。

インスタンス配置の可視性と制御、およびより包括的な BYOL サポートが必要な場合は、代わりに専有ホストの使用を検討してください。ハードウェア専有インスタンスと専有ホストのどちらを使用しても、専用の物理サーバーに Amazon EC2 インスタンスを起動することができます。ハードウェア専有インスタンス と Dedicated Hosts のインスタンスの間に、パフォーマンス、セキュリティ、または物理的な違いはありません。ただし、これらにはいくつかの重要な違いがあります。次のテーブルでは、Dedicated Hosts とハードウェア専有インスタンスの主な違いをいくつか紹介します。

	Dedicated Host	Dedicated Instance
専用物理サーバー	お客様専用のインスタンス容量を持つ物理サーバー。	単一の顧客アカウント専用の物理サーバー。
インスタンス容量の共有	インスタンス容量を他のアカウントと共有できます。	サポートされていません
請求	ホストごとの請求	インスタンスごとの請求
ソケット、コア、ホスト ID の可視性	ソケットと物理コアの数が見える	可視性なし
ホストおよびインスタンスアフィニティ	インスタンスを同じ物理サーバーに徐々にデプロイし続けることができる	サポート外
ターゲットを絞ったインスタンスの配置	インスタンスを物理サーバーに配置する方法についての可視性と制御が高い	サポート外
インスタンスの自動復旧	サポート対象。詳細については、ホスト復旧 を参照してください。	サポート対象
Bring-Your-Own-License (BYOL)	サポート	部分的なサポート*
キャパシティ予約	サポート外	サポート

* ソフトウェアアシュアランスによるライセンスモビリティを使用する Microsoft SQL Server、および Windows Virtual Desktop Access (VDA) ライセンスを、ハードウェア専有インスタンスで使用することが可能です。

専有インスタンスの詳細については、「Dedicated Hosts」を参照してください。

トピック

ハードウェア専有インスタンス の基本

VPC は、defaultまたはdedicatedのテナンシーを持つことができます。デフォルトでは、VPC default には default テナンシーがあり、テナンシー VPC default に起動されたインスタンスにはテナンシーがあります。ハードウェア専有インスタンス は次の手順を実行します。

- テナント属性が dedicated VPC を作成すると、VPC のすべてのインスタンスが専有インスタンスとして実行されます。詳細については、「専有インスタンスのテナンシーで VPC を作成します」を参照してください。
- テナンシー default の VPC を作成し、インスタンスを専用インスタンスとして実行するテナンシー dedicated を手動で指定します。詳細については、「VPC で ハードウェア専有インスタンス を起動する」を参照してください。

サポートされている機能

ハードウェア専有インスタンスは、以下の機能とAWSサービスの統合をサポートしています:

トピック

リザーブドインスタンス

ハードウェア専有インスタンスのキャパシティを予約するには、専有リザーブドインスタンスまたはキャパシティ予約を購入します。詳細については、Reserved InstancesおよびOn-Demand Capacity Reservationsを参照してください。

ハードウェア専有 リザーブドインスタンス を購入すると、VPC 内に ハードウェア専有インスタンス を起動するための容量を格安の料金で利用できます。使用料金引き下げは、専有テナントでインスタンスを起動した場合にのみ適用されます。デフォルトテナンシーで リザーブドインスタンス を購入する場合、これは `default` テナンシーがある実行中のインスタンスにのみ適用され、`dedicated` テナンシーがある実行中のインスタンスには適用されません。

さらに、リザーブドインスタンス の購入後に変更プロセスを使用してそのテナンシーを変更することはできません。ただし、新しい コンバーティブルリザーブドインスタンス の コンバーティブルリザーブドインスタンス を別のテナンシーと交換することはできます。

Auto Scaling

Amazon EC2 Auto Scaling を使用して ハードウェア専有インスタンス を起動できます。詳細については「VPC での Auto Scaling インスタンスの起動」(Amazon EC2 Auto Scaling ユーザーガイド) を参照してください。

自動復旧

基盤ハードウェアの障害、またはAWS による修復を必要とする問題によって正常に機能しなくなった場合のために、ハードウェア専有インスタンス に対し自動復旧を設定できます。詳細については、インスタンスの復旧 を参照してください。

ハードウェア専有スポットインスタンス

スポットインスタンスのリクエストを作成するとき、`dedicated` のテナントを指定することにより、ハードウェア専有スポットインスタンスを実行できます。詳細については、スポットインスタンス のテナンシーの指定 を参照してください。

バーストパフォーマンスインスタンス

the section called “バーストパフォーマンスインスタンス” では、専有テナントハードウェアで実行することの利点を活用できます。T3 ハードウェア専有インスタンスは、デフォルトで Unlimited モードで起動します。また、ベースラインレベルの CPU パフォーマンスを提供し、ワークロードの必要に応じてより高い CPU レベルにバーストできます。T3 ベースラインパフォーマンスとバースト機能は、CPU クレジットによって管理されます。T3 インスタンスタイプはバーストであるため、最適

なパフォーマンスを得るために T3 インスタンスで専用ハードウェアの CPU リソースをどのように使用しているかをモニタリングすることをお勧めします。T3 ハードウェア専有インスタンスは、お客様のワークロードが多様で CPU がランダムな動作を示すが、平均的な CPU 使用量が適切なベースライン使用量以下である場合に向いています。詳細については、the section called “主要なコンセプト” を参照してください。

Amazon EC2 には、パフォーマンスの変動を特定して修正するためのシステムが用意されています。ただし、CPU 使用パターンが相関する複数の T3 ハードウェア専有インスタンスを起動すると、依然として短期的な変動が発生する可能性があります。これらのより要求の厳しいワークロードや相関関係のあるワークロードについては、T3 ハードウェア専有インスタンスではなく、M5 または M5a ハードウェア専有インスタンスを使用することをお勧めします。

ハードウェア専有インスタンス の制限事項

ハードウェア専有インスタンスを使用するときは、以下の点を常に考慮する必要があります。

- 一部の AWS のサービスまたは機能は、インスタンスのテナント属性が `dedicated` に設定されている VPC ではサポートされていません。そのほかにも制限事項があるかどうかを確認するには、個別のサービスのドキュメントを参照してください。
- 一部の種類のインスタンスは、インスタンスのテナント属性が `dedicated` に設定されている VPC では起動できません。サポートされているインスタンスの種類の詳細については、「Amazon EC2 ハードウェア専有インスタンス」を参照してください。
- Amazon EBSでバックアップされたハードウェア専有インスタンスを起動する場合、シングルテナントのハードウェアで EBS ボリュームは実行できません。

ハードウェア専有インスタンス の料金表

ハードウェア専有インスタンス の料金表は、オンデマンドインスタンスの料金表と異なります。詳細については、「Amazon EC2 ハードウェア専有インスタンス 製品ページ」を参照してください。

ハードウェア専有インスタンス の操作

VPC の作成時にインスタンスのテナント属性として `dedicated` を指定すると、VPC 内に起動されるすべてのインスタンスを ハードウェア専有インスタンス にすることができます。インスタンスのテナント属性は起動時に指定することもできます。

トピック

- 専有インスタンスのテナンシーで VPC を作成します

- VPC で ハードウェア専有インスタンス を起動する
- テナント属性情報の表示
- インスタンスのテナンシーの変更
- VPC のテナント属性の変更

専有インスタンスのテナンシーで VPC を作成します

VPC を作成するときにインスタンスのテナント属性を指定できます。インスタンスのテナンシーが `dedicated` である VPC 内にインスタンスを起動すると、インスタンスは常に専用ハードウェアの専有インスタンスとして実行されます。

VPC の作成とテナンシーオプションの選択の詳細については、「Amazon VPC ユーザーガイド」の「VPC の作成」を参照してください。

VPC で ハードウェア専有インスタンス を起動する

ハードウェア専有インスタンス は、Amazon EC2 インスタンス起動ウィザードを使用して起動できます。

Console

コンソールを使用してデフォルトテナンシー VPC にハードウェア専有インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスを起動) の順に選択します。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションにあるリストから、使用する AMI を選択します。
4. [Instance type] (インスタンスタイプ) セクションで、起動するインスタンスタイプを選択します。

> **Note**
>
> ハードウェア専有インスタンスとしてサポートされているインスタンスタイプを必ず選択します。詳細については、「Amazon EC2 ハードウェア専有インスタンス」を参照してください。

5. [Key pair] (キーペア) セクションで、インスタンスに関連付けるキーペアを選択します。
6. [Advanced details] (高度な詳細) セクションにある [Tenancy] (テナンシー) で、[Dedicated] (ハードウェア専有) を選択します。
7. 必要に応じて、残りのインスタンスオプションを設定します。詳細については、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
8. [インスタンスを起動] を選択します。

Command line

コマンドラインを使用して起動中にインスタンスのテナント属性オプションを設定するには

- run-instances (AWS CLI)
- New-EC2Instance (AWS Tools for Windows PowerShell)

テナント属性として `host` を使用したインスタンスの作成の詳細については、「Dedicated Host でのインスタンスの起動」を参照してください。

テナント属性情報の表示

Console

コンソールを使用して VPC のテナント属性情報を表示するには

1. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
2. 画面左枠のナビゲーションペインで、[Your VPCs] を選択します。
3. [テナンシー] 列で、VPC のインスタンスのテナント属性を確認します。
4. [テナンシー] 列が表示されない場合は、右上の設定 () を選択し、[テナンシー] をオンにして [確認] をクリックします。

コンソールを使用してインスタンスのテナント属性情報を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. [テナンシー] 列でインスタンスのテナント属性を確認します。
4. [テナンシー] 列が表示されない場合は、次のいずれかを行います。

- 右上の設定
  ()
  を選択し、[テナンシー] をオンにして [確認] をクリックします。
- インスタンスを選択します。ページの下部近くにある [詳細] タブの [ホストとプレイスメントグループ] で、[テナンシー] の値を確認します。

Command line

コマンドラインを使用して VPC のテナンシーを記述するには

- describe-vpcs (AWS CLI)
- Get-EC2Vpc (AWS Tools for Windows PowerShell)

コマンドラインを使用してインスタンスのテナント属性を記述するには

- describe-instances (AWS CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

コマンドラインを使用してリザーブドインスタンスのテナント属性値を記述するには

- describe-reserved-instances (AWS CLI)
- Get-EC2ReservedInstance (AWS Tools for Windows PowerShell)

コマンドラインを使用してリザーブドインスタンス製品のテナント属性値を記述するには

- describe-reserved-instances-offerings (AWS CLI)
- Get-EC2ReservedInstancesOffering (AWS Tools for Windows PowerShell)

インスタンスのテナンシーの変更

インスタンスの起動後に停止されたテナンシーを変更できます。加えた変更は、次回のインスタンス起動時に有効になります。

インスタンスのオペレーティングシステムの詳細、および SQL Server がインストールされているかどうかによって、サポートされる変換が影響されます。インスタンスで使用できるテナンシー変換パ

スの詳細については、「License Manager ユーザーガイド」の「テナンシー変換」を参照してください。

> **Note**
>
> T3 インスタンスの場合、`host` のテナンシーを使用するには専有ホストでインスタンスを起動する必要があります。テナンシーを `host` から `dedicated` または `default` に変更することはできません。これらのサポートされていないテナンシー変更のいずれかを試みると、エラーコード `InvalidRequest` が発生します。

Console

コンソールを使用してインスタンスのテナント属性を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. [インスタンスの状態]、[インスタンスを停止]、[停止] の順に選択します。
4. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Modify instance placement (インスタンスの配置の変更)] の順に選択します。
5. [テナンシー] では、インスタンスを専有ハードウェアで実行するか、Dedicated Host で実行するかを選択します。[Save] を選択します。

Command line

コマンドラインを使用してインスタンスのテナント属性値を変更するには

- modify-instance-placement (AWS CLI)
- Edit-EC2InstancePlacement (AWS Tools for Windows PowerShell)

VPC のテナント属性の変更

VPC の作成後に VPC インスタンスのテナント属性を `dedicated` から `default` に変更することができます。VPC インスタンスのテナント属性を変更しても、VPC 内の既存のインスタンスのテナント属性には影響が及びません。次回 VPC でインスタンスを起動すると、起動時に指定していなければ、テナント属性が `default` になります。

> **Note**
>
> VPC の作成後に、VPC のインスタンステナンシーを `default` から `dedicated` に変更することはできません。

AWS CLIを使用しているVPC インスタンスのテナント属性を変更できるのは、 AWSSDK、あるいはAmazon EC2 API からのみです。

Command line

AWS CLI を使用して VPC インスタンスのテナント属性を変更するには

VPC の ID とインスタンスのテナント属性値を指定するには、modify-vpc-tenancy コマンドを使用します。`default` はサポートされる唯一の値です。

```
aws ec2 modify-vpc-tenancy --vpc-id vpc-1a2b3c4d --instance-tenancy default
```

キャパシティ予約

キャパシティ予約を使用すると、特定のアベイラビリティーゾーンで、Amazon EC2 インスタンスの計算能力を予約できます。キャパシティ予約には 2 種類あり、対応するユースケースが異なります。

キャパシティ予約の種類

- On-Demand Capacity Reservations
- 機械学習用のキャパシティブロック

オンデマンドキャパシティ予約の一般的なユースケースは以下のとおりです。

- スケーリングのイベント — ビジネスクリティカルなイベントの前にオンデマンドキャパシティ予約を作成しておくと、必要なときに確実にスケールすることができます。
- 規制要件とディザスタリカバリ — 高可用性に関する規制要件を満たしたり、ディザスタリカバリ用に別のアベイラビリティーゾーンまたはリージョンにキャパシティを予約したりするときは、オンデマンドキャパシティ予約を使用します。

ML 用のキャパシティブロックの一般的なユースケースは以下のとおりです。

- 機械学習 (ML) モデルトレーニングと微調整 — ML モデルトレーニングと微調整を完了するために予約した GPU インスタンスに、中断なしにアクセスできます。
- ML 実験とプロトタイプ — GPU インスタンスを必要とする実験の実行およびプロトタイプの構築を短期間で行えます。

オンデマンドキャパシティ予約の使用時期

キャパシティの要件が厳しく、キャパシティの保証を必要とするビジネスクリティカルなワークロードを実行している場合は、オンデマンドキャパシティ予約を使用します。オンデマンドキャパシティ予約を使用すると、予約した Amazon EC2 キャパシティに必要な限りアクセスすることができます。

機械学習用のキャパシティブロックを使用する時期

機械学習用のキャパシティブロックは、将来の一定期間、GPU インスタンスに中断なしにアクセスできるようにする必要がある場合に使用します。キャパシティブロックは、ML モデルのトレーニングや微調整、短期間の試験実行、将来、推論の需要が一時的に急増した場合の対応、に最適です。キャパシティブロックを使用すると、特定の日に GPU リソースにアクセスして確実に ML ワークロードを実行することができます。

On-Demand Capacity Reservations

オンデマンドキャパシティー予約 を使用すると、特定のアベイラビリティーゾーンで任意の所要時間だけ、Amazon EC2 インスタンスのコンピューティング能力を予約できます。キャパシティ予約は、キャパシティに制約がある場合にオンデマンドキャパシティを取得できないリスクを軽減します。キャパシティ要件が厳しく、一定レベルの長期または短期のキャパシティ保証を必要とするビジネスクリティカルなワークロードを実行している場合は、キャパシティ予約を作成して、必要なときに、必要な限り常に Amazon EC2 キャパシティにアクセスできるようにすることをお勧めします。

1 年間または 3 年間のコミットメント期間なしにいつでもキャパシティ予約を作成できます。アカウントでキャパシティ予約がプロビジョニングされるとすぐに、キャパシティが利用可能になり、課金が開始されます。キャパシティの保証が不要になった場合は、キャパシティ予約をキャンセルして、キャパシティをリリースし、料金の発生を停止します。また、Savings Plans やリージョナルリザーブドインスタンスが提供する請求割引を利用して、キャパシティ予約のコストを削減することもできます。

キャパシティーの予約を作成するときは、以下を指定します。

- キャパシティーが予約されているアベイラビリティーゾーン

- キャパシティーを予約するインスタンスの数
- インスタンスタイプ、テナンシー、プラットフォーム/OS を含む、インスタンスの属性

キャパシティーの予約を使用できるのは、属性が一致するインスタンスのみです。デフォルトでは、属性に一致する実行中のインスタンスによって自動的に使用されます。キャパシティーの予約 の属性と一致する実行中のインスタンスがない場合は、一致する属性を持つインスタンスを起動するまでは使用されません。

コンテンツ

キャパシティーの予約、リザーブドインスタンス、Savings Plans 間の違い

以下の表では、キャパシティーの予約、リザーブドインスタンス、Savings Plans 間の主な違いを示しています。

	Capacity Reservations	ゾーン リザーブドインスタンス	リージョン リザーブドインスタンス	Savings Plans
用語	コミットメントは不要です。必要に	固定の 1 年または 3 年のコミットメントが必要です。		

	Capacity Reservations	ゾーン リザーブドインスタンス	リージョン リザーブドインスタンス	Savings Plans
	応じて作成およびキャンセルすることができます。			
キャパシティーの利点	特定のアベイラビリティーゾーンで予約されるキャパシティー。		予約されたキャパシティーがありません。	
請求割引	請求割引がありません。	請求割引を提供します。		
インスタンスの制限	リージョンごとのオンデマンドインスタンス 制限が適用されます。	デフォルトは、アベイラビリティーゾーンごとに 20 です。制限の引き上げをリクエストできます。	デフォルトは、リージョンごとに 20 です。制限の引き上げをリクエストできます。	無制限。

† キャパシティー予約と、Savings Plans またはリージョンリでのザーブドインスタンスを組み合わせて、割引を受けることができます。

詳細については、以下を参照してください。

- Reserved Instances
- Savings Plans ユーザーガイド

サポートされているプラットフォーム

自分のインスタンスに適合するキャパシティー予約を、適切なプラットフォームで作成する必要があります。キャパシティー予約は、以下のプラットフォームをサポートしています。

- Linux/UNIX
- Linux with SQL Server Standard
- Linux with SQL Server Web

- Linux with SQL Server Enterprise
- SUSE Linux
- Red Hat Enterprise Linux
- RHEL with SQL Server Standard
- RHEL with SQL Server Enterprise
- RHEL with SQL Server Web
- RHEL with HA
- RHEL with HA および SQL Server Standard
- RHEL with HA および SQL Server Enterprise
- Ubuntu Pro

キャパシティーの予約 を購入する際、インスタンスのオペレーティングシステムを表すプラットフォームを指定する必要があります。

- BYOL を除く SUSE Linux および RHEL ディストリビューションの場合は、特定のプラットフォームを選択する必要があります。例えば、SUSE Linux や Red Hat Enterprise Linux プラットフォームなどです。
- その他のすべての Linux ディストリビューション (Ubuntu を含む) については、Linux/UNIX プラットフォームを選択します。
- 既存の RHEL サブスクリプション (BYOL) をお持ちの場合は、Linux/UNIX プラットフォームを選択する必要があります。

サポートされている Windows プラットフォームの詳細については、「Windows インスタンスの Amazon EC2 ユーザーガイド」の「サポートされるプラットフォーム」を参照してください。

クォータ

キャパシティの予約が許可されているインスタンスの数は、アカウントのオンデマンドインスタンスのクォータに基づいています。クォータに到達しない限り、既に実行されているインスタンスの数を差し引いた任意の数のインスタンスのキャパシティを予約できます。

クォータは実行中のインスタンスにのみ適用されます。インスタンスが保留中、停止中、停止済み、または休止状態の場合、クォータにはカウントされません。

制限事項

キャパシティーの予約 を作成する前に、次の制限と制約に注意してください。

- アクティブで未使用の キャパシティーの予約 は、オンデマンドインスタンス の制限の対象としてカウントされます。
- AWS アカウント間でキャパシティ予約を譲渡することはできません ただし、キャパシティー予約を他の AWS アカウントと共有することはできます。詳細については、「共有 キャパシティーの予約 の操作」を参照してください。
- ゾーン リザーブドインスタンス の請求割引は キャパシティーの予約 には適用されません。
- クラスタープレイスメントグループでキャパシティ予約を作成できます。スプレッドおよびパーティションプレイスメントグループはサポートされません。
- キャパシティーの予約 は Dedicated Hosts と供に使用することはできません キャパシティ予約は専有インスタンスで使用できます。
- キャパシティーの予約 では、休止状態のインスタンスを再開した場合でも、それが元の状態に復帰することを保証していません。

キャパシティーの予約 の料金と請求

トピック

- 料金
- 「請求」
- 請求割引
- 請求の表示

料金

インスタンスをリザーブドキャパシティで実行しているかどうかにかかわらず、オンデマンドの場合と同等の料金がキャパシティ予約に課金されます。予約を使用しない場合、この予約は Amazon EC2 請求書に未使用予約として記載されます。予約の属性に一致するインスタンスを実行するときは、そのインスタンスの料金のみを支払い、予約に料金はかかりません。前払い、または追加の料金はありません。

例えば、20 個の `m4.large` Linux インスタンスに対してキャパシティーの予約を作成し、同じアベイラビリティーゾーンで 15 個の `m4.large` Linux インスタンスを実行すると、15 個のアクティブインスタンスと予約されている 5 個の未使用のインスタンス分が課金されます。

Savings Plans とリージョンのリザーブドインスタンスの請求割引がキャパシティー予約に適用されます。詳細については、請求割引 を参照してください。

詳細については、「Amazon EC2 の料金表」を参照してください。

「請求」

アカウントでキャパシティ予約がプロビジョニングされるとすぐに課金が開始されます。以後、アカウントでキャパシティ予約がプロビジョニングされている間、継続して課金が発生します。

キャパシティーの予約 は、秒単位で課金されます。つまり、1 時間に満たない分に対して課金されます。例えば、24 時間 15 分の間、アカウント内でキャパシティ予約がプロビジョニングされたままである場合は、24.25 予約時間が課金されます。

次の例は、キャパシティーの予約 の請求方法を示しています。キャパシティーの予約は 1 つの m4.large Linux インスタンスに対して作成され、オンデマンド料金は 1 時間あたり 0.10 USD です。この例では、キャパシティ予約はアカウントで 5 時間プロビジョニングされます。キャパシティーの予約は最初の 1 時間は使用されないため、m4.large インスタンスタイプの標準オンデマンド料金で未使用の 1 時間分の料金が請求されます。2~5 時間目は、キャパシティーの予約は m4.large インスタンスによって占有されます。この間、キャパシティーの予約に料金は発生せず、代わりにそれを占有している m4.large インスタンスに対してアカウントが請求されます。6 時間目にはキャパシティーの予約がキャンセルされ、m4.large インスタンスはリザーブドキャパシティー外で通常どおりに実行されます。その時間は、m4.large インスタンスタイプのオンデマンド料金で請求されます。

Hour	1	2	3	4	5	6	Total cost
Unused Capacity Reservation	$0.10	$0.00	$0.00	$0.00	$0.00	$0.00	**$0.10**
On-demand Instance Usage	$0.00	$0.10	$0.10	$0.10	$0.10	$0.10	**$0.50**
Hourly cost	**$0.10**	**$0.10**	**$0.10**	**$0.10**	**$0.10**	**$0.10**	**$0.60**

請求割引

Savings Plans とリージョンのリザーブドインスタンスの請求割引がキャパシティー予約に適用されます。AWS は、属性が一致するキャパシティー予約に対しこれらの割引を自動的に適用します。キャパシティーの予約がインスタンスによって使用されると、割引がインスタンスに適用されます。割引は、未使用のキャパシティーの予約を対象とする前に、インスタンスの使用に優先的に適用されます。

ゾーンリザーブドインスタンスの請求割引はキャパシティーの予約には適用されません。

詳細については、以下を参照してください。

- Reserved Instances
- Savings Plans ユーザーガイド
- 請求と購入のオプション

請求の表示

アカウントの請求と料金は、AWS Billing and Cost Management コンソールで確認できます。

- [ダッシュボード] には、アカウント利用料の概要が表示されます。
- [請求書] ページの [明細] で、[Elastic Compute Cloud] セクションとリージョンを展開して、キャパシティーの予約 の請求情報を取得します。

請求額をオンラインで表示することも、CSV ファイルとしてダウンロードすることもできます。詳細については、AWS Billing and Cost Management ユーザーガイドの「キャパシティー予約の明細項目」を参照してください。

キャパシティーの予約 の操作

キャパシティーの予約の使用を開始するには、必要なアベイラビリティーゾーンにキャパシティーの予約を作成します。次に、インスタンスをリザーブドキャパシティーに起動し、そのキャパシティーの使用率をリアルタイムで表示して、必要に応じてキャパシティーを増減することができます。

デフォルトでは、キャパシティーの予約 は、新しいインスタンスと、一致する属性 (インスタンスタイプ、プラットフォーム、およびアベイラビリティーゾーン) を持つ実行中のインスタンスを自動的に一致させます。つまり、一致する属性を持つインスタンスがキャパシティーの予約で自動的に実行されます。ただし、特定のワークロードに対して キャパシティーの予約 を指定することもできます。これにより、リザーブドキャパシティーで実行できるインスタンスを明示的に制御できます。

予約が終了する方法を指定できます。キャパシティーの予約 をキャンセルするか、指定した時刻に自動的に終了させるかのどちらかから選択できます。終了時間を指定する場合、キャパシティーの予約 は指定した時刻の 1 時間以内にキャンセルされます。例えば、2019 年 5 月 31 日、13:30:55 を指定すると、キャパシティーの予約 は 2019 年 5 月 31 日の 13:30:55 と 14:30:55 の間に終了することが保証されます。予約が終了すると、インスタンスを キャパシティーの予約 のターゲットにすることはできなくなります。リザーブドキャパシティーで実行されているインスタンスは、中断されずに引き続き実行されます。キャパシティーの予約 をターゲットにしているインスタンスが停止してい

る場合は、キャパシティーの予約 ターゲット設定を削除するか、別の キャパシティーの予約 をターゲットに設定するまで再開できません。

目次

キャパシティーの予約 の作成

キャパシティ予約の作成リクエストが成功すると、そのキャパシティはすぐに利用可能になります。このキャパシティーは、キャパシティーの予約 がアクティブであれば、使用のために予約されており、いつでもインスタンスを起動することができます。キャパシティーの予約 がオープンの場合、新しいインスタンスと一致する属性を持つ既存のインスタンスは キャパシティーの予約 のキャパシティーで自動的に実行されます。キャパシティ予約 が `targeted` の場合、インスタンスはそれがリザーブドキャパシティで実行されるように具体的に設定する必要があります。

次のいずれかが当てはまる場合、キャパシティーの予約 を作成するリクエストは失敗する可能性があります。

- Amazon EC2 には、リクエストに対応する十分なキャパシティ - がありません。時間をおいてからもう一度試すか、別のアベイラビリティーゾーンを試すか、リクエストを小さくしてみてください。インスタンスタイプとサイズに応じてアプリケーションに柔軟性がある場合は、別のインスタンス属性を試してみてください。
- リクエストされた数量は、選択したインスタンスファミリーに対するオンデマンドインスタンスの上限を超えています。インスタンスファミリーに対するオンデマンドインスタンスの上限を上げて、もう一度試してください。詳細については、オンデマンドインスタンスクォータ を参照してください。

コンソールを使用して キャパシティーの予約 を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [キャパシティーの予約]、[作成キャパシティーの予約] の順に選択します。

3. キャパシティーの予約 の作成ページの、[Instance Details (インスタンスの詳細)] セクションで、以下の設定を指定します。起動するインスタンスのインスタンスタイプ、プラットフォーム、アベイラビリティーゾーンは、ここで指定するインスタンスタイプ、プラットフォーム、アベイラビリティーゾーンと一致する必要があります。一致しない場合、キャパシティーの予約は適用されません。例えば、開いているキャパシティーの予約が一致しない場合、このキャパシティーの予約を明示的に対象とするインスタンスの起動は失敗します。

   a. [Instance Type (インスタンスのタイプ)] — リザーブドキャパシティーに起動するインスタンスのタイプ。

   b. [Launch EBS-optimized instances (EBS 最適化インスタンスを起動する)] — EBS 最適化インスタンスのキャパシティーを予約するかどうかを指定します。このオプションは、一部のインスタンスタイプではデフォルトで選択されています。詳細については、「the section called “EBS 最適化”」を参照してください。

   c. [プラットフォーム] — インスタンスのオペレーティングシステム。詳細については、サポートされているプラットフォーム を参照してください。サポートされている Windows プラットフォームの詳細については、「Windows インスタンスの Amazon EC2 ユーザーガイド」の「サポートされるプラットフォーム」を参照してください。

   d. [アベイラビリティーゾーン] — キャパシティーを予約するアベイラビリティーゾーン。

   e. [テナンシー] — 共有ハードウェア (デフォルト) を実行するか専有インスタンスを実行するかを指定します。

   f. (オプション) [Placement group ARN] (プレイスメントグループ ARN) — キャパシティ予約が作成されるクラスタープレイスメントグループの ARN。

      詳細については、クラスタープレイスメントグループでのキャパシティ予約 を参照してください。

   g. [数量] — キャパシティーを予約するインスタンスの数。選択したインスタンスタイプの残りの オンデマンドインスタンス 制限を超える数量を指定すると、そのリクエストは拒否されます。

4. [Reservation details (予約の詳細)] セクションで次のように設定します。

   a. [Reservation Ends (予約終了)] — 次のいずれかのオプションを選択します。

      - [Manually (手動)] — 明示的にキャンセルするまで容量を予約してください。
      - [Specific time (特定の時間)] — 指定された日時にキャパシティーの予約を自動的に解除します。

   b. [Instance eligibility (インスタンスの利用資格)] — 次のいずれかのオプションを選択します。

- [開く] — (デフォルト) キャパシティーの予約 は、一致する属性 (インスタンスタイプ、プラットフォーム、およびアベイラビリティーゾーン) を持つインスタンスに一致します。一致する属性を持つインスタンスを起動すると、そのインスタンスはリザーブドキャパシティーに自動的に配置されます。
- [指定済み] — キャパシティーの予約は、一致する属性 (インスタンスタイプ、プラットフォーム、およびアベイラビリティーゾーン) を持つインスタンスのみを受け入れ、明示的に予約を行います。

5. [Request reservation (リクエスト予約)] を選択します。

AWS CLI を使用してキャパシティ予約を作成するには

create-capacity-reservation コマンドを使用します。詳細については、サポートされているプラットフォーム を参照してください。サポートされている Windows プラットフォームの詳細については、「Windows インスタンスの Amazon EC2 ユーザーガイド」の「サポートされるプラットフォーム」を参照してください。

例えば、次のコマンドで作成する キャパシティーの予約 では、`us-east-1a` アベイラビリティーゾーンで Red Hat Enterprise Linux AMI を実行する 3 つの `m5.2xlarge` インスタンスの容量を予約します。

```
aws ec2 create-capacity-reservation --instance-type m5.2xlarge --instance-platform Red
 Hat Enterprise Linux --availability-zone us-east-1a --instance-count 3
```

既存の キャパシティーの予約 へのインスタンスの起動

インスタンスを起動するときに、インスタンスを任意の open キャパシティーの予約 に起動するか、特定の キャパシティーの予約 に起動するか、または キャパシティーの予約 のグループに起動するかを指定できます。一致する属性 (インスタンスタイプ、プラットフォーム、およびアベイラビリティーゾーン) と十分なキャパシティーがある場合にのみ、インスタンスを キャパシティーの予約 に起動することができます。または、一致する属性と使用可能な容量を持つ open キャパシティーの予約 がある場合でも、キャパシティーの予約 でインスタンスを実行しないように設定できます。

キャパシティーの予約 にインスタンスを起動すると、起動されたインスタンスの数だけ使用可能なキャパシティーが減少します。例えば、3 つのインスタンスを起動すると、キャパシティーの予約 の使用可能なキャパシティーは 3 つ減少します。

コンソールを使用して既存の キャパシティーの予約 でインスタンスを起動するには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了してプレイスメントグループとキャパシティ予約の設定を指定するまでインスタンスを起動しないでください。
2. [高度な詳細] を展開し、以下の操作を行います。
   a. [プレイスメントグループ] で、インスタンスを起動するクラスタープレイスメントグループを選択します。
   b. [Capacity Reservation] (キャパシティ予約) で、キャパシティ予約の設定に応じて、次のいずれかのオプションを選択します。
      - [なし] — インスタンスがキャパシティ予約に起動しないようにします。インスタンスはオンデマンド型キャパシティーで実行されます。
      - [開く] — 選択したインスタンスの数に対して一致する属性と十分なキャパシティのあるキャパシティ予約にインスタンスを起動します。十分なキャパシティーを持つ、一致するキャパシティーの予約 がない場合は、インスタンスはオンデマンドのキャパシティーを使用します。
      - [ID 別のターゲット] — 選択したキャパシティ予約にインスタンスを起動します。選択されたこの キャパシティーの予約 に選択したインスタンスの数に対して十分なキャパシティーがない場合、インスタンスの起動に失敗します。
      - [グループ別のターゲット] — 選択したキャパシティ予約グループ内で一致する属性と使用可能なキャパシティを持つ任意のキャパシティ予約にインスタンスを起動します。選択したグループに、一致する属性と使用可能な容量を持つ キャパシティーの予約 がない場合、インスタンスはオンデマンド型キャパシティーに起動します。
3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI を使用して既存のキャパシティ予約の中でインスタンスを起動するには

run-instances コマンドを使用して `--capacity-reservation-specification` パラメータを指定します。

次の例では、属性と使用可能なキャパシティーが一致する任意の開いている キャパシティーの予約で `t2.micro` インスタンスを起動します。

```
aws ec2 run-instances --image-id ami-abc12345 --count 1 --instance-type t2.micro
 --key-name MyKeyPair --subnet-id subnet-1234567890abcdef1 --capacity-reservation-
specification CapacityReservationPreference=open
```

次の例では、t2.micro インスタンスをtargetedのキャパシティーの予約に起動します。

```
aws ec2 run-instances --image-id ami-abc12345 --count 1 --instance-type t2.micro
 --key-name MyKeyPair --subnet-id subnet-1234567890abcdef1 --capacity-reservation-
specification CapacityReservationTarget={CapacityReservationId=cr-a1234567}
```

次の例では、t2.micro インスタンスを キャパシティーの予約 グループに起動します。

```
aws ec2 run-instances --image-id ami-abc12345 --count 1
 --instance-type t2.micro --key-name MyKeyPair --subnet-
id subnet-1234567890abcdef1 --capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationResourceGroupArn=arn:aws:resource-
groups:us-west-1:123456789012:group/my-cr-group}
```

キャパシティーの予約 の変更

アクティブな キャパシティーの予約 の属性は、作成後に変更できます。期限が切れた後、または明示的にキャンセルした後で、キャパシティーの予約 を変更することはできません。

キャパシティーの予約 を変更する際は、数量を増減するだけで、リリースされる方法を変更することができます。キャパシティ予約のインスタンスタイプ、EBS 最適化、プラットフォーム、アベイラビリティーゾーン、またはインスタンス利用資格は変更できません。これらの属性を変更する必要がある場合は、予約をキャンセルし、必要な属性を持つ新しいものを作成することをお勧めします。

選択したインスタンスタイプの残りの オンデマンドインスタンス 制限を超える新しい数量を指定すると、その更新は失敗します。

コンソールを使用して キャパシティーの予約 を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [キャパシティーの予約] を選択し、キャパシティーの予約 を選択して、次に [Edit (編集)] を選択します。
3. 必要に応じて、[Quantity (数量)] または [Reservation ends (予約終了)] オプションを選択し、[Save changes (変更の保存)] を選択します。

AWS CLI を使用してキャパシティ予約を変更するには

modify-capacity-reservation コマンドを使用します。

例えば、次のコマンドは、8 つのインスタンスの容量を予約するために キャパシティーの予約 を変更します。

```
aws ec2 modify-capacity-reservation --capacity-reservation-id cr-1234567890abcdef0 --instance-count 8
```

インスタンスの キャパシティーの予約 設定の変更

停止したインスタンスの次のキャパシティーの予約設定は、いつでも変更できます。

- 一致する属性 (インスタンスタイプ、プラットフォーム、およびアベイラビリティーゾーン) と使用可能なキャパシティーを持つ任意の キャパシティーの予約 で起動します。
- 特定の キャパシティーの予約 でインスタンスを起動します。
- キャパシティー予約グループ内で、属性が一致し、キャパシティーが使用可能な、いずれかのキャパシティー予約を起動します。
- インスタンスが キャパシティーの予約 で起動しないようにします。

コンソールを使用して、インスタンスの キャパシティーの予約 設定を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [インスタンス] を選択し、変更するインスタンスを選択します。インスタンスをまだ停止していない場合は、停止します。
3. [アクション]、[キャパシティーの予約 設定を変更する] の順に選択します。
4. [キャパシティーの予約] で、以下のいずれかのオプションを選択します。
   - [Open (開く)] — 選択したインスタンスの数に対して一致する属性と十分なキャパシティ - のある キャパシティーの予約 にインスタンスを起動します。十分なキャパシティーを持つ、一致する キャパシティーの予約 がない場合は、インスタンスはオンデマンドのキャパシティーを使用します。
   - [なし] — インスタンスが キャパシティーの予約 に起動しないようにします。インスタンスはオンデマンド型キャパシティーで実行されます。

- [Specify Capacity Reservation (キャパシティー予約の指定)] — 選択した キャパシティーの予約 にインスタンスを起動します。選択されたこの キャパシティーの予約 に選択したインスタンスの数に対して十分なキャパシティーがない場合、インスタンスの起動に失敗します。
- [Specify Capacity Reservation group (キャパシティー予約グループの指定)] — 選択した キャパシティーの予約 グループ内で一致する属性と使用可能なキャパシティーを持つ キャパシティーの予約 にインスタンスを起動します。選択したグループに、一致する属性と使用可能な容量を持つ キャパシティーの予約 がない場合、インスタンスはオンデマンド型キャパシティーに起動します。

AWS CLI を使用してインスタンスのキャパシティ予約の設定を変更するには

modify-instance-capacity-reservation-attributes コマンドを使用します。

例えば、次のコマンドは、インスタンスの キャパシティーの予約 設定を open または none に変更します。

```
aws ec2 modify-instance-capacity-reservation-attributes --instance-
id i-1234567890abcdef0 --capacity-reservation-specification
 CapacityReservationPreference=none|open
```

例えば、次のコマンドは、特定の キャパシティーの予約 をターゲットにするようにインスタンスを変更します。

```
aws ec2 modify-instance-capacity-reservation-attributes --instance-
id i-1234567890abcdef0 --capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationId=cr-1234567890abcdef0}
```

例えば、次のコマンドは、特定の キャパシティーの予約 グループをターゲットにするようにインスタンスを変更します。

```
aws ec2 modify-instance-capacity-reservation-attributes --instance-
id i-1234567890abcdef0 --capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationResourceGroupArn=arn:aws:resource-
groups:us-west-1:123456789012:group/my-cr-group}
```

キャパシティーの予約 の表示

キャパシティーの予約には次の状態があります。

- `active`— キャパシティ - を使用できます。
- `expired` — キャパシティーの予約 は、予約リクエストで指定された日時に自動的に有効期限が切れました。リザーブドキャパシティー も使用できなくなります。
- `cancelled`— キャパシティーの予約 はキャンセルされました。リザーブドキャパシティー も使用できなくなります。
- `pending` — キャパシティーの予約 リクエストは成功しましたが、キャパシティーのプロビジョニングはまだ保留中です。
- `failed` — キャパシティーの予約 リクエストは失敗しました。有効でないリクエストパラメータ、キャパシティー制約、またはインスタンス制限の制約のため、リクエストが失敗する可能性があります。失敗したリクエストを 60 分間表示できます。

> **Note**
>
> 最終的な 整合性モデル とそれに続く Amazon EC2 API により、キャパシティ予約を作成した後、キャパシティ予約が `active` 状態であることを示すために、コンソールと describe-capacity-reservations 応答に最大 5 分かかる場合があります。この間、コンソールと `describe-capacity-reservations` レスポンスは、キャパシティ予約が `pending` 状態であることを示す場合があります。ただし、キャパシティ予約が既に使用可能になっている場合があり、その予約でインスタンスを起動してみることができます。

コンソールを使用して キャパシティーの予約 を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [キャパシティーの予約] を選択して、表示する キャパシティーの予約 を選択します。
3. [この予約の起動インスタンスを表示する]。

AWS CLI を使用してキャパシティ予約を表示するには

describe-capacity-reservations コマンドを使用します。

例えば、次のコマンドは、すべての キャパシティーの予約 について説明します。

```
aws ec2 describe-capacity-reservations
```

出力例。

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-1234abcd56EXAMPLE ",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-16T09:03:18.000Z",
            "AvailableInstanceCount": 1,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 1,
            "State": "active",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "a1.medium",
            "PlacementGroupArn": "arn:aws:ec2:us-east-1:123456789012:placement-group/
MyPG"
        },
        {
            "CapacityReservationId": "cr-abcdEXAMPLE9876ef ",
            "EndDateType": "unlimited",
            "AvailabilityZone": "eu-west-1a",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "EphemeralStorage": false,
            "CreateDate": "2019-08-07T11:34:19.000Z",
            "AvailableInstanceCount": 3,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 3,
            "State": "cancelled",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "m5.large"
        }
    ]
}
```

キャパシティーの予約 のキャンセル

リザーブドキャパシティーが不要になったら、いつでも キャパシティーの予約 をキャンセルできます。キャパシティーの予約 をキャンセルすると、キャパシティーがリリースされ、使用のために予約されなくなります。

空の キャパシティーの予約 と実行中のインスタンスがある キャパシティーの予約 をキャンセルすることができます。実行中のインスタンスがあるキャパシティー予約をキャンセルした場合、インスタンスは、標準のオンデマンドインスタンス料金または割引料金 (一致する Savings Plan またはリージョンのリザーブドインスタンスがある場合) で、キャパシティー予約外で正常に動作し続けます。

キャパシティーの予約 をキャンセルすると、それをターゲットとするインスタンスは起動できなくなります。これらのインスタンスを異なる キャパシティーの予約 をターゲットに設定するように変更し、一致する属性と十分なキャパシティーでオープンな キャパシティーの予約 に起動するか、キャパシティーの予約 への起動を回避します。詳細については、インスタンスの キャパシティーの予約 設定の変更 を参照してください。

コンソールを使用して キャパシティーの予約 をキャンセルするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [キャパシティーの予約] を選択し、キャンセルする キャパシティーの予約 を選択します。
3. [Cancel reservation (予約をキャンセル)] 選択し、[Cancel reservation (予約をキャンセル)] を選択します。

AWS CLI を使用してキャパシティ予約をキャンセルするには

cancel-capacity-reservation コマンドを使用します。

例えば、次のコマンドは、ID が `cr-1234567890abcdef0` である キャパシティーの予約 をキャンセルします。

```
aws ec2 cancel-capacity-reservation --capacity-reservation-id cr-1234567890abcdef0
```

キャパシティーの予約 グループの操作

AWS Resource Groups を使用して、Resource Groupsと呼ばれるキャパシティー予約の論理コレクションを作成できます。リソースグループは、すべて同じ AWS リージョンにある AWS リソースの論理的なグループです。リソースグループの詳細については、『AWS Resource Groups ユーザーガイド』の「 リソースグループとは」を参照してください。

自分のアカウントで所有するキャパシティ予約および他の AWS アカウントから共有を受けているキャパシティ予約は、1 つのリソースグループに含めることができます。また、異なる属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン) を持つキャパシティ予約も 1 つのリソースグループに含めることができます。

キャパシティ予約のリソースグループを作成すると、個別のキャパシティ予約ではなく、キャパシティ予約のグループをインスタンスのターゲットにできます。キャパシティーの予約 のグループをターゲットとするインスタンスは、属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン) と使用可能な容量を持つグループ内の キャパシティーの予約 と一致します。一致する属性と使用可能な容量を持つ キャパシティーの予約 がグループにない場合、インスタンスはオンデマンド型キャパシティーを使用して実行されます。一致する キャパシティーの予約 が後の段階でターゲットグループに追加されると、インスタンスは自動的にマッチングされ、リザーブドキャパシティーに移動されます。

グループで キャパシティーの予約 の意図しない使用を防ぐには、キャパシティー予約を明示的にターゲットとするインスタンスだけを受け入れるように、グループの キャパシティーの予約 を設定します。これを行うには、Amazon EC2 コンソールを使用して キャパシティーの予約 を作成するときに、[Instance eligibility (インスタンスの適格性)] を [targeted (ターゲット)] (古いコンソール) または [Only instances that specify this reservation (この予約を指定するインスタンスのみ)] (新しいコンソール) に設定します。AWS CLI を使用する場合は、キャパシティ予約の作成時に `--instance-match-criteria targeted` を指定します。これにより、グループまたはグループ内の キャパシティーの予約 を明示的にターゲットとするインスタンスのみが、グループ内で実行できるようになります。

実行中のインスタンスがある間にグループの キャパシティーの予約 がキャンセルまたは期限切れになった場合、インスタンスは、一致する属性と使用可能な容量を持つグループ内の別の キャパシティーの予約 に自動的に移動されます。一致する属性と使用可能な容量を持つ キャパシティーの予約 がグループに残っていない場合、インスタンスはオンデマンド型キャパシティーで実行されます。一致する キャパシティーの予約 が後の段階でターゲットグループに追加されると、インスタンスは自動的にリザーブドキャパシティーに移動されます。

トピック

- キャパシティ予約グループを作成する
- キャパシティ予約をグループに追加するには
- グループのキャパシティ予約を表示する
- キャパシティ予約が属するグループを表示する
- グループからキャパシティ予約を削除する

- キャパシティ予約グループを削除する

キャパシティ予約グループを作成する

キャパシティ予約のグループを作成するには

create-group AWS CLI コマンドを使用します。name で、グループのわかりやすい名前を指定し、configuration で、次の 2 つの Type リクエストパラメータを指定します。

- AWS::EC2::CapacityReservationPool を指定して、リソースグループがインスタンス起動の対象となるようにします。
- AWS::ResourceGroups::Generic で allowed-resource-types を AWS::EC2::CapacityReservation に設定して、リソースグループがキャパシティー予約のみを受け入れるようにします。

例えば、次のコマンドは、MyCRGroup という名前のグループを作成します。

```
$ aws resource-groups create-group --name MyCRGroup --configuration
 '{"Type":"AWS::EC2::CapacityReservationPool"}'
 '{"Type":"AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-resource-
types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

出力例を次に示します。

```
{
    "GroupConfiguration": {
        "Status": "UPDATE_COMPLETE",
        "Configuration": [
            {
                "Type": "AWS::EC2::CapacityReservationPool"
            },
            {
                "Type": "AWS::ResourceGroups::Generic",
                "Parameters": [
                    {
                        "Values": [
                            "AWS::EC2::CapacityReservation"
                        ],
                        "Name": "allowed-resource-types"
                    }
                ]
```

```
            }
        ]
    },
    "Group": {
        "GroupArn": "arn:aws:resource-groups:sa-east-1:123456789012:group/MyCRGroup",
        "Name": "MyCRGroup"
    }
}
```

キャパシティ予約をグループに追加するには

共有されているキャパシティ予約をグループに追加し、そのキャパシティ予約の共有が解除されると、そのキャパシティ予約はグループから自動的に削除されます。

キャパシティーの予約 をグループに追加するには

group-resources AWS CLI コマンドを使用します。group には、キャパシティーの予約 を追加するグループの名前を指定し、resources には、追加する キャパシティーの予約 の ARN を指定します。複数の キャパシティーの予約 を追加するには、ARN をスペースで区切ります。追加する キャパシティー予約 の ARN を取得するには、AWS CLI の describe-capacity-reservations コマンドを使用して、キャパシティー予約 の ID を指定します。

例えば、次のコマンドは、MyCRGroup という名前のグループに 2 つの キャパシティーの予約 を追加します。

```
$ aws resource-groups group-resources --group MyCRGroup --resource-arns arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1 arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-54321abcdef567890
```

出力例を次に示します。

```
{
    "Failed": [],
    "Succeeded": [
        "arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1",
        "arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-54321abcdef567890"
    ]
}
```

グループのキャパシティ予約を表示する

特定のグループの キャパシティーの予約 を表示するには

list-group-resources AWS CLI コマンドを使用します。group に、グループの名前を指定します。

例えば、次のコマンドは、MyCRGroup という名前のグループ内の キャパシティーの予約 をリストします。

```
$ aws resource-groups list-group-resources --group MyCRGroup
```

出力例を次に示します。

```
{
    "QueryErrors": [],
    "ResourceIdentifiers": [
        {
            "ResourceType": "AWS::EC2::CapacityReservation",
            "ResourceArn": "arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/
cr-1234567890abcdef1"
        },
        {
            "ResourceType": "AWS::EC2::CapacityReservation",
            "ResourceArn": "arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/
cr-54321abcdef567890"
        }
    ]
}
```

> **Note**
>
> コマンド出力には、自身が所有するキャパシティ予約と共有を受けているキャパシティ予約が含まれます。

キャパシティ予約が属するグループを表示する

AWS CLI

特定のキャパシティ予約が追加されたグループを表示するには

get-groups-for-capacity-reservation AWS CLI コマンドを使用します。

例えば、次のコマンドは、キャパシティーの予約 cr-1234567890abcdef1 が追加されたグループをリストします。

```
$ aws ec2 get-groups-for-capacity-reservation --capacity-reservation-
id cr-1234567890abcdef1
```

出力例を次に示します。

```
{
    "CapacityReservationGroups": [
        {
            "OwnerId": "123456789012",
            "GroupArn": "arn:aws:resource-groups:sa-east-1:123456789012:group/
MyCRGroup"
        }
    ]
}
```

> **Note**
>
> 共有しているキャパシティ予約を指定した場合、コマンドは所有しているキャパシティ予約グループのみを返します。

Amazon EC2 console

特定のキャパシティ予約が追加されたグループを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [キャパシティーの予約] を選択し、表示する キャパシティーの予約 を選択して、[表示] を選択します。

   キャパシティーの予約 が追加されたグループは、[グループ] カードにリストされます。

> **Note**
>
> 共有しているキャパシティ予約を選択した場合、コンソールには所有しているキャパシティ予約グループのみが表示されます。

グループからキャパシティ予約を削除する

グループから キャパシティーの予約 を削除するには

ungroup-resources AWS CLI コマンドを使用します。group には、キャパシティーの予約 を削除するグループの ARN を指定し、resources には、削除する キャパシティーの予約 の ARN を指定します。複数の キャパシティーの予約 を削除するには、ARN をスペースで区切ります。

次の例では、MyCRGroup という名前のグループから 2 つの キャパシティーの予約 を削除します。

```
$ aws resource-groups ungroup-resources --group MyCRGroup --resource-
arns arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/
cr-0e154d26a16094dd arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/
cr-54321abcdef567890
```

出力例を次に示します。

```
{
    "Failed": [],
    "Succeeded": [
        "arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-0e154d26a16094dd",
        "arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/cr-54321abcdef567890"
    ]
}
```

キャパシティ予約グループを削除する

グループを削除するには

delete-group AWS CLI コマンドを使用します。[group] で、削除するグループの名前を選択します。

例えば、次のコマンドは、MyCRGroup という名前のグループを削除します。

```
$ aws resource-groups delete-group --group MyCRGroup
```

出力例を次に示します。

```
{
    "Group": {
```

```
        "GroupArn": "arn:aws:resource-groups:sa-east-1:123456789012:group/MyCRGroup",
        "Name": "MyCRGroup"
    }
}
```

クラスタープレイスメントグループでのキャパシティ予約

クラスタープレイスメントグループでキャパシティ予約を作成して、ワークロードの Amazon EC2 コンピューティング性能を予約できます。クラスタープレイスメントグループは、ネットワークレイテンシーが低く、ネットワークスループットが高いという利点があります。

クラスタープレイスメントグループでキャパシティ予約を作成すると、必要なときに、必要な期間中、クラスタープレイスメントグループのコンピューティング性能に確実にアクセスできるようになります。これは、コンピューティングのスケーリングを必要とする高パフォーマンス (HPC) ワークロードのキャパシティを予約するのに最適です。これにより、キャパシティを使用できる状態を維持しながらクラスターをスケールダウンできるため、必要に応じて再びスケールアップすることができます。

トピック

- 制限事項
- クラスタープレイスメントグループでのキャパシティ予約の操作

制限事項

クラスタープレイスメントグループでキャパシティ予約を作成する場合は、以下の点を常に考慮します。

- 既存のキャパシティ予約がプレイスメントグループにない場合は、キャパシティ予約を変更してプレイスメントグループ内でキャパシティを予約することはできません。プレイスメントグループでキャパシティを予約するには、プレイスメントグループでキャパシティ予約を作成する必要があります。
- プレイスメントグループでキャパシティ予約を作成した後、プレイスメントグループ外のキャパシティを予約するように変更することはできません。
- プレイスメントグループの既存のキャパシティ予約を変更するか、プレイスメントグループに追加のキャパシティ予約を作成して、プレイスメントグループのリザーブドキャパシティを増やすことができます。ただし、容量不足エラーが発生する可能性が高くなります。
- クラスタープレイスメントグループで作成されたキャパシティ予約を共有することはできません。

- `active` 容量予約を持つクラスタープレイスメントグループは削除できません。クラスタープレイスメントグループ内のすべての容量予約を削除する前に、それらをキャンセルする必要があります。

クラスタープレイスメントグループでのキャパシティ予約の操作

クラスタープレイスメントグループでキャパシティ予約の使用を開始するには、次のステップを実行します。

> **Note**
>
> 既存のクラスタープレイスメントグループでキャパシティ予約を作成する場合は、ステップ 1 をスキップします。次に、ステップ 2 と 3 で、既存のクラスタープレイスメントグループの ARN を指定します。既存のクラスタープレイスメントグループの ARN を確認する方法については、「プレイスメントグループ情報を表示する」を参照してください。

トピック

- ステップ 1: (条件付き) キャパシティ予約で使用するクラスタープレイスメントグループを作成する
- ステップ 2: クラスタープレイスメントグループでキャパシティ予約を作成する
- ステップ 3: クラスタープレイスメントグループでインスタンスを起動する

ステップ 1: (条件付き) キャパシティ予約で使用するクラスタープレイスメントグループを作成する

このステップは、新しいクラスタープレイスメントグループを作成する必要がある場合にのみ実行します。既存のクラスタープレイスメントグループを使用する場合は、このステップをスキップし、ステップ 2 と 3 で、そのクラスタープレイスメントグループの ARN を使用します。既存のクラスタープレイスメントグループの ARN を確認する方法については、「プレイスメントグループ情報を表示する」を参照してください。

クラスタープレイスメントグループは、次のいずれかの方法で作成できます。

Console

コンソールを使用してクラスタープレイスメントグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Placement Groups] (プレイスメントグループ)、[Create placement group] (プレイスメントグループの作成) の順に選択します。
3. [Name] (名前) で、プレイスメントグループのわかりやすい名前を指定します。
4. [Placement strategy] (プレイスメント戦略) で、[Cluster] (クラスター) を選択します。
5. [グループを作成] を選択します。
6. [プレイスメントグループ] テーブルの [グループ ARN] 列で、作成したクラスタープレイスメントグループの ARN を書き留めます。これは次のステップで必要になります。

AWS CLI

AWS CLI を使用してクラスタープレイスメントグループを作成するには

create-placement-group コマンドを使用します。`--group-name` でプレイスメントグループのわかりやすい名前を指定し、`--strategy` で `cluster` を指定します。

次の例では、`cluster` プレイスメント戦略を使用する、`MyPG` という名前のプレイスメントグループを作成します。

```
$ aws ec2 create-placement-group \
    --group-name MyPG \
    --strategy cluster
```

コマンド出力で返されるプレイスメントグループ ARN は次のステップで必要となるので、メモしておいてください。

ステップ 2: クラスタープレイスメントグループでキャパシティ予約を作成する

キャパシティ予約を作成するのと同じ方法で、クラスタープレイスメントグループでキャパシティ予約を作成します。ただし、キャパシティ予約を作成するクラスタープレイスメントグループの ARN も指定する必要があります。詳細については、キャパシティーの予約 の作成 を参照してください。

考慮事項

- 指定したクラスタープレイスメントグループは `available` 状態になっている必要があります。クラスタープレイスメントグループが `pending`、`deleting`、または `deleted` 状態になっていると、リクエストは失敗します。
- キャパシティ予約とクラスタープレイスメントグループが同じアベイラビリティーゾーンに存在している必要があります。キャパシティ予約を作成するリクエストで、クラスタープレイスメントグ

ループのアベイラビリティーゾーンとは異なるアベイラビリティーゾーンが指定されている場合、リクエストは失敗します。

- キャパシティ予約は、クラスタープレイスメントグループでサポートされているインスタンスタイプに対してのみ作成できます。サポートされていないインスタンスタイプを指定すると、リクエストは失敗します。詳細については、クラスタープレイスメントグループのルールと制限 を参照してください。
- クラスタープレイスメントグループで open キャパシティ予約を作成し、一致する属性 (プレイスメントグループ ARN、インスタンスタイプ、アベイラビリティーゾーン、プラットフォーム、テナンシー) を持つ既存の実行中のインスタンスがある場合、それらのインスタンスはキャパシティ予約で自動的に実行されます。
- 次のいずれかが当てはまる場合、キャパシティーの予約 を作成するリクエストは失敗する可能性があります。
  - Amazon EC2 には、リクエストに対応する十分なキャパシティ - がありません。時間をおいてからもう一度試すか、別のアベイラビリティーゾーンを試すか、キャパシティ - を小さくしてみてください。インスタンスタイプとサイズに応じてワークロードに柔軟性がある場合は、別のインスタンス属性を試してみてください。
  - リクエストされた数量は、選択したインスタンスファミリーに対するオンデマンドインスタンスの上限を超えています。インスタンスファミリーに対するオンデマンドインスタンスの上限を上げて、もう一度試してください。詳細については、オンデマンドインスタンスクォータ を参照してください。

次のいずれかの方法で、クラスタープレイスメントグループでキャパシティ予約を作成できます。

Console

コンソールを使用して キャパシティーの予約 を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [キャパシティ予約]、[作成キャパシティ予約] の順に選択します。
3. [キャパシティ予約を作成] ページで、必要に応じてインスタンスタイプ、プラットフォーム、アベイラビリティーゾーン、テナンシー、数量、および終了日を指定します。
4. [プレイスメントグループ] で、キャパシティ予約が作成されるクラスタープレイスメントグループの ARN を選択します。
5. [Create] (作成) を選択します。

詳細については、キャパシティーの予約 の作成 を参照してください。

AWS CLI

AWS CLI を使用してキャパシティ予約を作成するには

create-capacity-reservation コマンドを使用します。--placement-group-arn で、キャパシティ予約が作成されるクラスタープレイスメントグループの ARN を指定します。

```
$ aws ec2 create-capacity-reservation \
    --instance-type instance_type \
    --instance-platform platform \
    --availability-zone az \
    --instance-count quantity \
    --placement-group-arn placement_group_ARN
```

詳細については、キャパシティーの予約 の作成 を参照してください。

ステップ 3: クラスタープレイスメントグループでインスタンスを起動する

キャパシティ予約でインスタンスを起動するのと同じ方法で、クラスタープレイスメントグループのキャパシティ予約でインスタンスを起動します。ただし、インスタンスを起動するクラスタープレイスメントグループの ARN も指定する必要があります。詳細については、キャパシティーの予約 の作成 を参照してください。

考慮事項

- キャパシティ予約が open の場合は、インスタンスの起動リクエストでキャパシティ予約を指定する必要はありません。インスタンスに、指定したプレイスメントグループのキャパシティ予約に一致する属性 (プレイスメントグループ ARN、インスタンスタイプ、アベイラビリティーゾーン、プラットフォーム、テナンシー) がある場合、インスタンスはキャパシティ予約で自動的に実行されます。
- キャパシティ予約がターゲットインスタンスの起動のみを受け入れる場合は、リクエストでクラスタープレイスメントグループに加えてターゲットキャパシティ予約を指定する必要があります。
- キャパシティ予約がキャパシティ予約グループに含まれる場合は、リクエストでクラスタープレイスメントグループに加えてターゲットキャパシティ予約グループを指定する必要があります。詳細については、キャパシティーの予約 グループの操作 を参照してください。

次のいずれかの方法で、クラスタープレイスメントグループのキャパシティ予約でインスタンスを起動できます。

Console

コンソールを使用して既存の キャパシティーの予約 でインスタンスを起動するには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了してプレイスメントグループとキャパシティ予約の設定を指定するまでインスタンスを起動しないでください。
2. [高度な詳細] を展開し、以下の操作を行います。
   a. [プレイスメントグループ] で、インスタンスを起動するクラスタープレイスメントグループを選択します。
   b. [Capacity Reservation] (キャパシティ予約) で、キャパシティ予約の設定に応じて、次のいずれかのオプションを選択します。
      - [開く] — 一致する属性と十分なキャパシティを持つ、クラスタープレイスメントグループの open キャパシティ予約でインスタンスを起動します。
      - [ID 別のターゲット] — ターゲットインスタンスの起動のみを受け入れるキャパシティ予約でインスタンスを起動します。
      - [グループ別のターゲット] — 選択したキャパシティ予約グループ内で一致する属性と使用可能なキャパシティを持つ任意のキャパシティ予約にインスタンスを起動します。
3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

詳細については、「既存の キャパシティーの予約 へのインスタンスの起動」を参照してください。

AWS CLI

AWS CLI を使用して既存のキャパシティ予約でインスタンスを起動するには

run-instances コマンドを使用します。特定のキャパシティ予約またはキャパシティ予約グループをターゲットにする必要がある場合は、`--capacity-reservation-specification` パラメータを指定します。`--placement` で、`GroupName` パラメータを指定し、前のステップで作成したプレイスメントグループの名前を指定します。

次のコマンドでは、クラスタープレイスメントグループの targeted キャパシティ予約でインスタンスが起動されます。

```
$ aws ec2 run-instances \
    --image-id ami_id \
    --count quantity \
    --instance-type instance_type \
    --key-name key_pair_name \
    --subnet-id subnetid \
    --capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationId=capacity_reservation_id} \
    --placement "GroupName=cluster_placement_group_name"
```

詳細については、既存の キャパシティーの予約 へのインスタンスの起動 を参照してください。

Local Zones での キャパシティーの予約

ローカルゾーンは、ユーザーに地理的に近い AWS リージョンを拡張したものです。ローカルゾーンで作成したリソースにより、非常に低いレイテンシーの通信がローカルユーザーに提供されます。詳細については、AWS Local Zonesをご参照ください。

VPC をその親 AWS リージョンからローカルゾーンに拡張するには、そのローカルゾーンに新しいサブネットを作成します。ローカルゾーンにサブネットを作成すると、VPC はそのローカルゾーンに拡張されます。ローカルゾーンのサブネットは、VPC 内の他のサブネットと同じように動作します。

Local Zones を使用すると、ユーザーに近い複数の場所に キャパシティーの予約 を配置できます。通常のアベイラビリティーゾーンで キャパシティーの予約 を作成して使用するのと同じ方法で、Local Zones で キャパシティーの予約 を作成して使用します。同じ機能とインスタンスマッチング動作が適用されます。Local Zones でサポートされている料金モデルの詳細については、「AWS Local Zones に関するよくある質問」を参照してください。

考慮事項

ローカルゾーンでキャパシティー予約グループを使用することはできません。

ローカルゾーンでキャパシティー予約を使用するには

1. AWS アカウントでローカルゾーンの使用を有効にします。詳細については、Local Zones へのオプトイン を参照してください。

2. ローカルゾーンにキャパシティー予約を作成します。アベイラビリティーゾーンには、 ローカルゾーンを選択します。ローカルゾーンは、AWS リージョンコードの末尾に場所を示す識別子を付加して表します (us-west-2-lax-1a など)。詳細については、キャパシティーの予約 の作成 を参照してください。
3. ローカルゾーン内にサブネットを作成します。アベイラビリティーゾーンには、 ローカルゾーンを選択します。詳細については、「Amazon VPC ユーザーガイド」の「VPC でサブネットを作成する」を参照してください。
4. インスタンスを起動します。サブネットには、ローカルゾーンのサブネット (subnet-123abc | us-west-2-lax-1a など) を選択し、キャパシティー予約 には、ローカルゾーンで作成したキャパシティー予約に必要な仕様 (open または ID で指定したターゲット) を選択します。詳細については、既存の キャパシティーの予約 へのインスタンスの起動 を参照してください。

Wavelength Zone 内の キャパシティ予約

AWS Wavelength を使用することで、デベロッパーは、モバイルデバイスおよびエンドユーザー向けに、非常にレイテンシーが低いアプリケーションを構築できます。Wavelength は、標準の AWS コンピューティングおよびストレージサービスを通信事業者の 5G ネットワークのエッジにデプロイします。Amazon Virtual Private Cloud (VPC) は、1 つまたは複数の Wavelength Zone に拡張できます。その後、Amazon EC2 インスタンスなどの AWS リソースを使用して、極めて低いレイテンシーやリージョンの AWS サービスへの接続を必要とするアプリケーションを実行できます。詳細については、「AWS Wavelength ゾーン」を参照してください。

オンデマンド キャパシティ予約を作成する場合は、Wavelength Zone を選択し、Wavelength Zone に関連付けられたサブネットを指定することで、Wavelength Zone 内の キャパシティ予約にインスタンスを起動できます。Wavelength Zone は、AWS リージョンコードの末尾に場所を示す識別子を付加して表します (us-east-1-wl1-bos-wlz-1 など)。

Wavelength Zone は、すべてのリージョンで利用できるわけではありません。Wavelength Zone をサポートするリージョンについては、AWS Wavelength デベロッパーガイドの利用可能な Wavelength Zoneを参照してください。

考慮事項

Wavelength Zone でキャパシティ予約グループを使用することはできません。

Wavelength Zone で キャパシティーの予約 を使用するには

1. AWS アカウントで Wavelength Zone の使用を有効にします。詳細については、Linux インスタンス用 Amazon EC2 ユーザーガイド の「Wavelength Zone の有効化」を参照してください。

2. Wavelength Zone にキャパシティ予約を作成します。[アベイラビリティーゾーン]で、Wavelength を選択します。Wavelength は、AWS リージョンコードの末尾に場所を示す識別子を付加して表します (us-east-1-wl1-bos-wlz-1 など)。詳細については、キャパシティーの予約 の作成 を参照してください。
3. Wavelength Zone にサブネットを作成します。[アベイラビリティーゾーン]で、Wavelength Zone を選択します。詳細については、「Amazon VPC ユーザーガイド」の「VPC でサブネットを作成する」を参照してください。
4. インスタンスを起動します。サブネットには、Wavelength Zone のサブネット (subnet-123abc | us-east-1-wl1-bos-wlz-1 など) を選択し、キャパシティーの予約 には、Wavelength で作成した キャパシティーの予約 に必要な仕様 (open または ID で指定したターゲット) を選択します。詳細については、既存の キャパシティーの予約 へのインスタンスの起動 を参照してください。

AWS Outposts でのキャパシティーの予約

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをお客様のオンプレミスまで拡張するフルマネージドサービスです。AWS は、AWS Outposts マネージドインフラストラクチャへのローカルアクセスを提供することで、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築して実行できるようにします。同時に、コンピューティングとストレージのローカルリソースを使用して、レイテンシーを短縮し、ローカルのデータ処理ニーズに対応します。

Outpost とは、お客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS は、AWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。

ユーザーは、自分のアカウントで作成した Outposts にキャパシティ予約を作成できます。これにより、自分のサイトにある Outpost で、コンピューティング性能を予約できるようになります。通常のアベイラビリティーゾーンでキャパシティー予約を作成して使用するのと同じ方法で、Outposts でもキャパシティー予約 を作成して使用できます。同じ機能とインスタンスマッチング動作が適用されます。

また、AWS を使用することで、Outposts にあるキャパシティ予約を自分の組織内の他の AWS Resource Access Manager アカウントと共有できます。キャパシティ予約の共有については、「共有 キャパシティーの予約 の操作」を参照してください。

前提条件

Outpost は、自分のサイトにインストールする必要があります。詳細については、AWS Outposts ユーザーガイドの「Outpost を作成し、Outpost 容量を注文する」を参照してください。

考慮事項

- Outpost では、キャパシティ予約グループを使用することはできません。

Outpost でキャパシティ予約を使用するには

1. Outpost にサブネットを作成します。詳細については、AWS Outposts ユーザーガイドの「サブネットの作成」を参照してください。
2. Outpost にキャパシティ予約を作成します。
   a. AWS Outposts コンソール (https://console.aws.amazon.com/outposts/) を開きます。
   b. ナビゲーションペインで [Outposts]、[アクション]、[キャパシティー予約の作成] の順に選択します。
   c. 必要に応じてキャパシティ予約を設定し、[作成] を選択します。詳細については、キャパシティーの予約 の作成 を参照してください。

   > **Note**
   > [インスタンスタイプ] ドロップダウンには、選択した Outpost でサポートされているインスタンスタイプのみが表示されます。また、[アベイラビリティーゾーン] ドロップダウンには、選択したアウトポストが関連付けられているアベイラビリティーゾーンのみが一覧表示されます。

3. キャパシティー予約 でのインスタンスの起動 [サブネット] では、ステップ 1 で作成したサブネットを選択します。また、[キャパシティー予約] では、ステップ 2 で作成したキャパシティ予約を選択します。詳細については、AWS Outposts ユーザーガイドの、「Outposts でインスタンスを起動する」を参照してください。

共有 キャパシティーの予約 の操作

キャパシティ予約の共有を使用すると、キャパシティ予約の所有者は、リザーブドキャパシティを他の AWS アカウントと共有することや、AWS 組織内で共有することが可能になります。これにより、キャパシティ予約の作成と管理を一元的に行い、リザーブドキャパシティを複数の AWS アカウント間や AWS 組織内で共有できるようになります。

このモデルでは、キャパシティ予約を所有する AWS アカウント (所有者) が、キャパシティ予約を他の AWS アカウント (コンシューマー) と共有します。コンシューマーは、自身のアカウントで所有しているキャパシティーの予約にインスタンスを起動する場合と同じように、共有を受けているキャパシティーの予約にインスタンスを起動できます。キャパシティーの予約 の所有者は、共有した キャパシティーの予約 と、そこで起動したインスタンスを管理します。所有者は、共有した キャパシティーの予約 でコンシューマーが起動したインスタンスを変更することはできません。コンシューマーは、共有を受けている キャパシティーの予約 で起動したインスタンスを管理します。コンシューマーが、キャパシティーの予約 の所有者や他のコンシューマーが所有するインスタンスを表示したり変更したりすることはできません。

キャパシティーの予約 の所有者が キャパシティーの予約 を共有できる相手は次のとおりです。

- AWS の組織内または組織外の特定の AWS アカウント
- AWS 組織内の組織単位
- AWS 組織全体

コンテンツ

キャパシティーの予約 を共有するための前提条件

- キャパシティ予約を共有するには、その予約を自分の AWS アカウント内で所有している必要があります。自身が共有を受けている キャパシティーの予約 を他者に共有することはできません。
- 共有テナンシーインスタンスの キャパシティーの予約 のみ共有できます。専用テナンシーインスタンスの キャパシティーの予約 は共有できません。

- 新規の AWS アカウントや、請求制限履歴のある AWS アカウントでは、キャパシティ予約の共有を使用できません。
- AWS 組織や AWS 組織内の組織単位とキャパシティ予約を共有するには、AWS Organizations で共有を有効にする必要があります。詳細については、AWS RAM ユーザーガイドの「AWS Organizations で共有を有効化する」を参照してください。

関連サービス

キャパシティ予約の共有は AWS Resource Access Manager (AWS RAM) と統合されます。AWS RAM は、AWS リソースを任意の AWS アカウントと共有したり、AWS Organizations 経由で共有したりするためのサービスです。AWS RAM を使用すると、リソース共有を作成することで、自身が所有するリソースを共有できます。リソース共有では、共有対象のリソースと、共有先となるコンシューマーを指定します。コンシューマーには、個人の AWS アカウントや、AWS Organizations 内の組織単位または組織全体を指定できます。

AWS RAM の詳細については、AWS RAM ユーザーガイドを参照してください。

アベイラビリティーゾーン間での共有

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 アカウントの名前に個別にマッピングされます。このため、アカウントが異なると、アベイラビリティーゾーンの命名方法が異なる場合があります。例えば、`us-east-1a` アカウントのアベイラビリティーゾーン AWS の場所は、別の `us-east-1a` アカウントのアベイラビリティーゾーン AWS の場所と異なる可能性があります。

自身のアカウントを基準にして キャパシティーの予約 の場所を特定するには、アベイラビリティーゾーン ID (AZ ID) を使用する必要があります。AZ ID は、すべての AWS アカウントで同じアベイラビリティーゾーンを一貫して示すための一意の識別子です。例えば、`use1-az1` は `us-east-1` リージョンの AZ ID であり、すべての AWS アカウントで同じ場所を示します。

アカウントのアベイラビリティーゾーンの AZ ID を表示するには

1. AWS RAM コンソール (https://console.aws.amazon.com/ram) を開きます。
2. 現在のリージョンの AZ ID は、画面の右側にある [お客様の AZ ID] パネルに表示されます。

キャパシティーの予約 の共有

自分が所有するキャパシティ予約を他の AWS アカウントと共有すると、そのアカウントに対して自分のリザーブドキャパシティ内でインスタンスを起動することを許可することになります。オープン

な キャパシティーの予約 を共有する場合は、意図しない形で キャパシティーの予約 が使用されないよう、次の点に注意してください。

- キャパシティーの予約 の属性に一致するインスタンスをコンシューマーが実行している場合に、CapacityReservationPreference パラメータが open に設定され、リザーブドキャパシティー内での実行がまだであれば、共有 キャパシティーの予約 が自動的に使用されます。
- 属性 (インスタンスタイプ、プラットフォーム、アベイラビリティーゾーン) が一致するインスタンスをコンシューマーが起動する場合、CapacityReservationPreference パラメータが open に設定されていれば、自動的に共有 キャパシティーの予約 で起動されます。

キャパシティーの予約 を共有するには、リソース共有に追加する必要があります。リソース共有とは、AWS RAM アカウント間で自身のリソースを共有するための AWS リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。Amazon EC2 コンソールを使用して キャパシティーの予約 を共有すると、既存のリソース共有に追加されます。キャパシティーの予約 を新しいリソース共有に追加するには、AWS RAM コンソールを使用してリソース共有を作成する必要があります。

自分が AWS Organizations 内の組織のメンバーであり、所属する組織内での共有が有効化されている場合、共有の前提条件が満たされていれば、組織内のコンシューマーに、共有されたキャパシティ予約へのアクセス許可を自動で付与することができます。キャパシティ予約が外部のアカウントと共有されている場合、コンシューマーは、リソースへの参加の招待を受け取り、その招待を受け入れた後で、共有されているキャパシティ予約に対するアクセス許可が付与されます。

> **Important**
>
> 共有されているキャパシティ予約でインスタンスを起動する前に、コンソールで共有キャパシティ予約を表示するか、describe-capacity-reservations AWS CLI コマンドを使用してこれを記述することで、共有キャパシティ予約へのアクセス権があることを確認します。コンソールで共有キャパシティ予約を表示できるか、AWS CLI を使ってそれを記述できる場合、ユーザーはこれを使って、そこでインスタンスを起動することができます。インスタンスをキャパシティ予約で起動しようとして、共有の障害によりアクセスできない場合、そのインスタンスは、オンデマンドキャパシティで起動できます。

Amazon EC2 コンソール、AWS RAM コンソール、または AWS CLI を使用して、自分が所有するキャパシティ予約を共有できます。

Amazon EC2 コンソールを使用して、自身が所有する キャパシティーの予約 を共有するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キャパシティーの予約] を選択します。
3. 共有する キャパシティーの予約 を選択し、[アクション]、[Share reservation] の順に選択します。
4. キャパシティーの予約 の追加先となるリソース共有を選択し、[Share キャパシティーの予約] を選択します。

   コンシューマーから共有 キャパシティーの予約 にアクセスできるようになるまでに、数分かかることがあります。

AWS RAM コンソールを使用して自分が所有するキャパシティ予約を共有するには

AWS RAM ユーザーガイドの「リソース共有の作成」を参照してください。

AWS CLI を使用して自分が所有するキャパシティ予約を共有するには

create-resource-share コマンドを使用します。

キャパシティーの予約 の共有を停止する

キャパシティーの予約 の所有者は キャパシティーの予約 の共有をいつでも停止できます。以下のルールが適用されます。

- コンシューマーが所有するインスタンスのうち、共有解除の時点で共有キャパシティー内で実行されていたインスタンスは、リザーブドキャパシティー外で正常に動作し続けます。キャパシティーは、Amazon EC2 キャパシティーの可用性に応じて キャパシティーの予約 に復元されます。
- キャパシティーの予約 の共有先コンシューマーが、このリザーブドキャパシティーで新たにインスタンスを起動することはできません。

所有している キャパシティーの予約 の共有を停止するには、リソース共有から削除する必要があります。この操作を行うには、Amazon EC2 コンソール、AWS RAM コンソール、または AWS CLI を使用します。

Amazon EC2 コンソールを使用して、所有している キャパシティーの予約 の共有を停止するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キャパシティーの予約] を選択します。

3. キャパシティーの予約 を選択して、[Sharing (共有)] タブを選択します。
4. [共有] タブに、キャパシティーの予約 の追加先のリソース共有が一覧表示されます。キャパシティーの予約 を削除する対象のリソース共有を選択し、[リソース共有から削除] を選択します。

AWS RAM コンソールを使用して自分が所有しているキャパシティ予約の共有を停止するには

AWS RAM ユーザーガイドの「リソース共有の更新」を参照してください。

AWS CLI を使用して自分が所有しているキャパシティ予約の共有を停止するには

disassociate-resource-share コマンドを使用します。

共有されているキャパシティ予約の特定と閲覧

> **Important**
>
> 共有されているキャパシティ予約でインスタンスを起動する前に、コンソールで共有キャパシティ予約を表示するか、AWS CLI を使用してこれを記述することで、共有キャパシティ予約へのアクセス権があることを確認します。コンソールで共有キャパシティ予約を表示できるか、AWS CLI を使ってそれを記述できる場合、ユーザーはこれを使って、そこでインスタンスを起動することができます。インスタンスをキャパシティ予約で起動しようとして、共有の障害によりアクセスできない場合、そのインスタンスは、オンデマンドキャパシティで起動できます。

所有者とコンシューマーは、Amazon EC2 コンソールおよび AWS CLI を使用して、共有されているキャパシティ予約を特定できます。

Amazon EC2 コンソールを使用して共有 キャパシティーの予約 を特定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キャパシティーの予約] を選択します。この画面には、自身が所有する キャパシティーの予約 と共有を受けている キャパシティーの予約 が一覧表示されます。[所有者] 列には、キャパシティー予約の所有者の AWS アカウント ID が示されます。AWS アカウント ID の横に (`me`) と表示されている場合は、自身が所有者であることを示します。

AWS CLI を使用して、共有されているキャパシティ予約を特定するには

describe-capacity-reservations コマンドを使用します。このコマンドでは、自身が所有するキャパシティ予約および他から共有を受けているキャパシティ予約が返されます。OwnerId は、キャパシティ予約の所有者の AWS アカウント ID を示します。

共有 キャパシティーの予約 の使用状況の表示

共有しているキャパシティ予約の所有者は、Amazon EC2 コンソールまたは AWS CLI を使用して、いつでも使用状況を表示できます。

Amazon EC2 コンソールを使用して キャパシティーの予約 の使用状況を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キャパシティーの予約] を選択します。
3. 使用状況を表示する キャパシティーの予約 を選択し、[使用状況] タブを選択します。

   [AWS アカウント ID] 列には、現在キャパシティー予約を使用しているコンシューマーのアカウント ID が表示されます。[起動したインスタンス] 列には、リザーブドキャパシティー内で各コンシューマーが現在実行しているインスタンスの数が表示されます。

AWS CLI を使用してキャパシティ予約の使用状況を表示するには

get-capacity-reservation-usage コマンドを使用します。AccountId では、キャパシティーの予約 を使用しているアカウントのアカウント ID が表示されます。UsedInstanceCount では、リザーブドキャパシティー内でコンシューマーが現在実行しているインスタンスの数が表示されます。

共有 キャパシティーの予約 のアクセス許可

所有者のアクセス許可

共有 キャパシティーの予約 の管理とキャンセルは、所有者が行います。所有者は、共有 キャパシティーの予約 内で実行されており他のアカウントが所有するインスタンスを変更することはできません。共有 キャパシティーの予約 で起動されされたインスタンスは、所有者が管理します。

コンシューマーのアクセス許可

コンシューマーは、共有 キャパシティーの予約 で実行している自身のインスタンスを管理します。コンシューマーは、共有 キャパシティーの予約 をどのような方法で変更することもできません。また、他のコンシューマーまたは キャパシティーの予約 の所有者が所有するインスタンスを表示または変更することもできません。

請求と使用量測定

キャパシティーの予約 の共有に追加料金はかかりません。

キャパシティーの予約 の所有者には、キャパシティーの予約 内で自身が実行するインスタンスと、使用されていないリザーブドキャパシティーに対する料金が請求されます。コンシューマーには、共有 キャパシティーの予約 内で自身が実行するインスタンスに対する料金が請求されます。

キャパシティ予約の所有者が別の支払いアカウントに属していて、キャパシティ予約がリージョンのリザーブドインスタンスまたは Savings Plans でカバーされている場合、キャパシティ予約の所有者には引き続きリージョンのリザーブドインスタンスまたは Savings Plans の料金が請求されます。この場合、キャパシティ予約の所有者はリージョンのリザーブドインスタンスまたは Savings Plans の料金を支払い、コンシューマーには共有キャパシティ予約で実行されるインスタンスに対して請求が行われます。

インスタンス制限

キャパシティーの予約 の使用量はすべて、キャパシティーの予約 の所有者の オンデマンドインスタンス 制限の対象としてカウントされます。ここでは次の点について説明します。

- 使用されていないリザーブドキャパシティ
- キャパシティーの予約 の所有者が所有するインスタンスによる使用量
- コンシューマーが所有するインスタンスによる使用量

共有キャパシティー内でコンシューマーによって起動されたインスタンスは、キャパシティーの予約の所有者の オンデマンドインスタンス 制限の対象としてカウントされます。コンシューマーのインスタンス制限は、コンシューマー自身が所有する オンデマンドインスタンス の制限と、コンシューマーがアクセスできる共有 キャパシティーの予約 内で使用可能なキャパシティーの合計です。

キャパシティー予約フリート

オンデマンドキャパシティー予約フリートとは、キャパシティー予約のグループです。

キャパシティー予約フリートリクエストには、キャパシティー予約フリートの起動に必要なすべての設定情報が含まれます。1 つのリクエストを使用して、指定したターゲット容量まで、複数のインスタンスタイプにわたって、ワークロードに大量の Amazon EC2 キャパシティーを予約できます。

キャパシティー予約フリートを作成した後は、キャパシティー予約フリートの変更やキャンセルなど、フリートのキャパシティー予約の管理を一括で行うことができます。

トピック

キャパシティー予約フリートの仕組み

キャパシティー予約フリートを作成すると、フリートリクエストで指定した合計のターゲット容量を満たすため、フリートはキャパシティーの予約を個別に作成しようとします。

フリートがキャパシティーを予約するインスタンス数は、指定したターゲットの総容量とインスタンスタイプの重みに依存します。キャパシティーを予約するインスタンスタイプは、使用する配分戦略とインスタンスタイプの優先順位により異なります。

フリートの作成時に十分なキャパシティーがなく、その総ターゲット容量をすぐに満たすことができない場合、フリートは、要求されたキャパシティーを予約が完了するまで、キャパシティー予約を非同期的に作成しようとします。

フリートが総ターゲット容量に達すると、そのキャパシティーを維持しようとします。フリートのキャパシティー予約がキャンセルされた場合、フリートの設定に応じて、フリートは自動的に 1 つ以上のキャパシティー予約を作成して、失われたキャパシティーを置き換え、その総ターゲット容量を維持します。

フリート内のキャパシティー予約を、個別に管理することはできません。フリートを変更することによって、まとめて管理する必要があります。フリートを変更すると、フリートのキャパシティー予約が自動的に更新され、変更が反映されます。

現在、キャパシティー予約フリートは open インスタンスの一致条件をサポートしています。フリートによって起動されるすべてのキャパシティー予約は、このインスタンス一致基準を自動的に使用します。この基準では、属性 (インスタンスタイプ、プラットフォーム、およびアベイラビリティーゾーン) が一致する新しいインスタンスと既存のインスタンスは、フリートにより作成されたキャパシティー予約内で自動的に実行されます。キャパシティー予約フリートでは、`target` インスタンス一致基準はサポートされていません。

考慮事項

キャパシティー予約フリートを使用する際には、次の点に注意してください。

- キャパシティー予約フリートは、AWS CLI および AWS API を使用して作成、変更、表示、キャンセルができます。
- フリート内のキャパシティー予約を、個別に管理することはできません。これらは、フリートを変更またはキャンセルすることで、まとめて管理する必要があります。
- キャパシティー予約フリートは、複数のリージョンにまたがることはできません。
- キャパシティー予約フリートは複数のアベイラビリティーゾーンにまたがることはできません。
- キャパシティー予約フリートによって作成されたキャパシティー予約には、AWS で作成された次のタグが自動的に付けられます。
  - キー — `aws:ec2-capacity-reservation-fleet`
  - 値 – *`fleet_id`*

  このタグを使用して、キャパシティー予約フリートによって作成されたキャパシティー予約を識別できます。

料金

キャパシティー予約フリートの使用に追加料金はかかりません。キャパシティー予約フリートによって作成された、個々のキャパシティー予約に対して料金が発生します。キャパシティー予約に対する課金については、「キャパシティーの予約 の料金と請求」を参照してください 。

キャパシティー予約フリートの概念

このトピックでは、キャパシティー予約フリートの概念のいくつかについて説明します。

トピック

- 総ターゲット容量
- 配分戦略
- インスタンスタイプの重み
- インスタンスタイプ優先順位

総ターゲット容量

総ターゲット容量の定義は、キャパシティー予約フリートで予約されるコンピューティング性能の総量です。キャパシティー予約フリートを作成するときは、総ターゲット容量を指定します。フリートが作成されると、Amazon EC2 は自動的にキャパシティー予約を作成し、総ターゲット容量までキャパシティーを予約します。

キャパシティー予約フリートでキャパシティーを予約する対象のインスタンスの数は、総ターゲット容量と、キャパシティー予約フリート内のインスタンスタイプごとに指定したインスタンスタイプの重みで決まります (`total target capacity/instance type weight=number of instances`)。

ワークロードにとって意味のあるユニット数に基づいて、総ターゲット容量を割り当てることができます。例えば、ワークロードで特定の数の vCPU が必要な場合、必要な vCPU の数に基づいて総ターゲット容量を割り当てることができます。ワークロードに 2048 個の vCPU が必要な場合、合計ターゲット容量として 2048 を指定します。次に、フリートのインスタンスタイプによって提供される vCPU の数に基づいて、インスタンスタイプの重みを割り当てます。例については、「インスタンスタイプの重み」を参照してください。

配分戦略

キャパシティー予約フリートの割り当て戦略により、キャパシティー予約フリート設定のインスタンスタイプの仕様を基に、リザーブドキャパシティーのリクエストを満たすための方法が決定されます。

現在は、`prioritized` の割り当て戦略のみがサポートされています。この戦略を使用するキャパシティー予約フリートは、キャパシティ–予約フリート設定で各インスタンスタイプ仕様に割り当てた優先順位に従い、キャパシティー予約を作成します。優先度の値が低いと、使用する優先順位が高くなります。例えば、次のインスタンスタイプと優先度を使用するキャパシティー予約フリートを作成するとします。

- `m4.16xlarge` – 優先度 = 1
- `m5.16xlarge` – 優先度 = 3
- `m5.24xlarge` – 優先度 = 2

フリートは、まず `m4.16xlarge` のキャパシティー予約の作成を試みます。Amazon EC2 に十分な `m4.16xlarge` キャパシティーがない場合、フリートは `m5.24xlarge` のキャパシティー予約の作成を試みます。Amazon EC2 の `m5.24xlarge` キャパシティーが不十分な場合には、フリートは、`m5.16xlarge` でキャパシティー予約を作成します。

インスタンスタイプの重み

インスタンスタイプの重みとは、キャパシティー予約フリート内の各インスタンスタイプに割り当てる分量のことです。重みによって、その特定のインスタンスタイプの各インスタンスがフリートの総ターゲット容量にカウントされるキャパシティーのユニット数が決まります。

ワークロードにとって意味のあるユニット数に基づいて重みを割り当てることができます。例えば、ワークロードに特定の数の vCPU が必要な場合、キャパシティー予約フリートの各インスタンスタイプごとに指定した vCPU の数に基づいて重みを割り当てることができます。この場合、m4.16xlarge および m5.24xlarge インスタンスを使用してキャパシティー予約フリートを作成したとすると、次のように各インスタンスの vCPU 数に対応する重みを割り当てます。

- m4.16xlarge – vCPU 数 64、重み = 64 ユニット
- m5.24xlarge – vCPU 数 96、重み = 96 ユニット

インスタンスタイプの重みによって、キャパシティーの予約フリートでキャパシティーを予約する対象となる、インスタンスの数が決定されます。例えば、総ターゲット容量が 384 ユニットのキャパシティー予約フリートが、前述の例のインスタンスタイプと重みを使用する場合、フリートのキャパシティー予約は、m4.16xlarge を 6 インスタンス (総ターゲット容量 384/インスタンスタイプの重み 64 = 6 インスタンス) になることも、m5.24xlarge を 4 インスタンス (384/96 = 4) になることもあります。

インスタンスタイプのウェイトを割り当てない場合、またはインスタンスタイプの重みに 1 を割り当てた場合には、合計ターゲット容量は純粋にインスタンス数に基づきます。例えば、総ターゲット容量が 384 ユニットのキャパシティー予約フリートが前の例のインスタンスタイプを使用する場合であっても、重みを省略するか、両方のインスタンスタイプに重み 1 を指定すると、フリートのキャパシティー予約は、m4.16xlarge を 384 インスタンスか、m5.24xlarge を 384 インスタンスかのいずれかになります。

インスタンスタイプ優先順位

インスタンスタイプの優先順位は、フリートのインスタンスタイプに割り当てる値です。優先順位は、フリートに指定されているインスタンスタイプのどれに対し、使用上の優先順位を付ける必要があるかを決定します。

優先度の値は、使用する優先順位が高いことを示します。

キャパシティー予約フリートの操作

トピック

開始する前に

キャパシティー予約フリートを作成する前に、以下を実行します。

1. ワークロードに必要なコンピューティング性能の量を決定します。
2. 使用するインスタンスタイプとアベイラビリティーゾーンを決定します。
3. 要件と設定内容に基づいて、各インスタンスタイプに優先度を割り当てます。詳細については、インスタンスタイプ優先順位 を参照してください。
4. ワークロードに適したキャパシティー重み付けシステムを作成します。各インスタンスタイプに重みを割り当て、総ターゲット容量を決定します。詳細については、インスタンスタイプの重みおよび総ターゲット容量を参照してください。
5. キャパシティー予約を無期限に必要とするか、特定の期間だけ必要かを決定します。

キャパシティー予約フリートのステータス

キャパシティー予約フリートは、以下のいずれかの状態を取ります。

- `submitted` – キャパシティー予約フリートへのリクエストが送信され、Amazon EC2 はキャパシティー予約を作成する準備をしています。
- `modifying` – キャパシティー予約フリートは変更中です。フリートは、変更が完了するまではこの状態のままになります。
- `active` – キャパシティー予約フリートが総ターゲット容量を満たしており、このキャパシティーを維持しようとしています。フリートは、変更または削除されるまで、この状態のままになります。
- `partially_fulfilled` – キャパシティー予約フリートは、その総ターゲット容量を部分的に満たしています。Amazon EC2 に、総ターゲット容量を満たすのに十分なキャパシティーがありません。フリートは、総ターゲット容量を非同期的に満たそうとしています。

- expiring – キャパシティー予約フリートが終了日に達し、期限切れになろうとしています。1 つ以上のキャパシティー予約がまだアクティブになっている可能性があります。
- expired – キャパシティー予約フリートの使用が終了日に達しました。フリートとそのキャパシティー予約は期限切れです。フリートは、キャパシティー予約を新たに作成することはできません。
- cancelling – キャパシティー予約フリートはキャンセルされようとしています。1 つ以上のキャパシティー予約がまだアクティブになっている可能性があります。
- cancelled – キャパシティー予約フリートが手動でキャンセルされました。フリートとそのキャパシティー予約はキャンセルされ、フリートはキャパシティー予約を新しく作成することはできません。
- failed – キャパシティー予約フリートは、指定されたインスタンスタイプのキャパシティーを予約できませんでした。

キャパシティー予約フリートを作成する

キャパシティー予約フリートを作成すると、フリートへのリクエスト内で指定されたインスタンスタイプのキャパシティー予約が、指定された合計ターゲット容量までフリートにより自動的に作成されます。キャパシティー予約フリートがキャパシティーを予約するインスタンスの数は、リクエストで指定する合計ターゲット容量とインスタンスタイプの重みによって異なります。詳細については、インスタンスタイプの重みおよび総ターゲット容量を参照してください。

フリートを作成する際には、使用するインスタンスタイプと、それらのインスタンスタイプごとに優先順位を指定する必要があります。詳細については、配分戦略およびインスタンスタイプ優先順位を参照してください。

> **Note**
>
> サービスにリンクされた AWSServiceRoleForEC2CapacityReservationFleet ロールは、キャパシティー予約フリートを初めて作成するときに、アカウントに自動的に作成されます。詳細については、キャパシティ予約フリートでのサービスにリンクされたロールの使用 を参照してください。

現在、キャパシティー予約フリートは open のインスタンス一致条件のみをサポートしています。

キャパシティー予約フリートは、コマンドラインのみを使用して作成できます。

キャパシティー予約フリートを作成するには

AWS CLI コマンドの create-capacity-reservation-fleet を使用します。

```
$ aws ec2 create-capacity-reservation-fleet \
--total-target-capacity capacity_units \
--allocation-strategy prioritized \
--instance-match-criteria open \
--tenancy dedicated|default \
--end-date yyyy-mm-ddThh:mm:ss.000Z \
--instance-type-specifications file://instanceTypeSpecification.json
```

instanceTypeSpecification.json の内容は次のとおりです。

```
[
    {
        "InstanceType": "instance_type",
        "InstancePlatform":"platform",
        "Weight": instance_type_weight,
        "AvailabilityZone":"availability_zone",
        "AvailabilityZoneId" : "az_id",
        "EbsOptimized": true|false,
        "Priority" : instance_type_priority
    }
]
```

正常な出力

```
{
    "Status": "status",
    "TotalFulfilledCapacity": fulfilled_capacity,
    "CapacityReservationFleetId": "cr_fleet_id",
    "TotalTargetCapacity": capacity_units
}
```

例

```
$ aws ec2 create-capacity-reservation-fleet \
--total-target-capacity 24 \
--allocation-strategy prioritized \
--instance-match-criteria open \
--tenancy default \
--end-date 2021-12-31T23:59:59.000Z \
--instance-type-specifications file://instanceTypeSpecification.json
```

`instanceTypeSpecification.json`

```
[
  {
    "InstanceType": "m5.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "Weight": 3.0,
    "AvailabilityZone":"us-east-1a",
    "EbsOptimized": true,
    "Priority" : 1
  }
]
```

出力例。

```
{
    "Status": "submitted",
    "TotalFulfilledCapacity": 0.0,
    "CapacityReservationFleetId": "crf-abcdef01234567890",
    "TotalTargetCapacity": 24
}
```

キャパシティー予約フリートを表示する

キャパシティー予約フリートの構成およびキャパシティーに関する情報は、任意のタイミングで表示できます。フリートを表示すると、フリート内の個々のキャパシティー予約に関する詳細も表示されます。

キャパシティー予約フリートは、コマンドラインのみを使用して表示できます。

キャパシティー予約フリートを表示するには

describe-capacity-reservation-fleets AWS CLI コマンドを使用します。

```
$ aws ec2 describe-capacity-reservation-fleets \
--capacity-reservation-fleet-ids cr_fleet_ids
```

正常な出力

```
{
    "CapacityReservationFleets": [
        {
```

```
            "Status": "status",
            "EndDate": "yyyy-mm-ddThh:mm:ss.000Z",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "CapacityReservationFleetId": "cr_fleet_id",
            "Tenancy": "dedicated|default",
            "InstanceTypeSpecifications": [
                {
                    "CapacityReservationId": "cr1_id",
                    "AvailabilityZone": "cr1_availability_zone",
                    "FulfilledCapacity": cr1_used_capacity,
                    "Weight": cr1_instance_type_weight,
                    "CreateDate": "yyyy-mm-ddThh:mm:ss.000Z",
                    "InstancePlatform": "cr1_platform",
                    "TotalInstanceCount": cr1_number of instances,
                    "Priority": cr1_instance_type_priority,
                    "EbsOptimized": true|false,
                    "InstanceType": "cr1_instance_type"
                },
    {
                    "CapacityReservationId": "cr2_id",
                    "AvailabilityZone": "cr2_availability_zone",
                    "FulfilledCapacity": cr2_used_capacity,
                    "Weight": cr2_instance_type_weight,
                    "CreateDate": "yyyy-mm-ddThh:mm:ss.000Z",
                    "InstancePlatform": "cr2_platform",
                    "TotalInstanceCount": cr2_number of instances,
                    "Priority": cr2_instance_type_priority,
                    "EbsOptimized": true|false,
                    "InstanceType": "cr2_instance_type"
                },
            ],
            "TotalTargetCapacity": total_target_capacity,
            "TotalFulfilledCapacity": total_target_capacity,
            "CreateTime": "yyyy-mm-ddThh:mm:ss.000Z",
            "AllocationStrategy": "prioritized"
        }
    ]
}
```

例

```
$ aws ec2 describe-capacity-reservation-fleets \
```

```
--capacity-reservation-fleet-ids crf-abcdef01234567890
```

出力例

```
{
    "CapacityReservationFleets": [
        {
            "Status": "active",
            "EndDate": "2021-12-31T23:59:59.000Z",
            "InstanceMatchCriteria": "open",
            "Tags": [],
            "CapacityReservationFleetId": "crf-abcdef01234567890",
            "Tenancy": "default",
            "InstanceTypeSpecifications": [
                {
                    "CapacityReservationId": "cr-1234567890abcdef0",
                    "AvailabilityZone": "us-east-1a",
                    "FulfilledCapacity": 5.0,
                    "Weight": 1.0,
                    "CreateDate": "2021-07-02T08:34:33.398Z",
                    "InstancePlatform": "Linux/UNIX",
                    "TotalInstanceCount": 5,
                    "Priority": 1,
                    "EbsOptimized": true,
                    "InstanceType": "m5.xlarge"
                }
            ],
            "TotalTargetCapacity": 5,
            "TotalFulfilledCapacity": 5.0,
            "CreateTime": "2021-07-02T08:34:33.397Z",
            "AllocationStrategy": "prioritized"
        }
    ]
}
```

キャパシティー予約フリートを変更する

キャパシティー予約フリートの合計ターゲット容量と日付は、任意のタイミングで変更できます。キャパシティー予約フリートの総ターゲット容量を変更すると、フリートは、新しい総ターゲット容量を満たすように、自動的に新しいキャパシティー予約を作成したり、フリート内の既存のキャパシティー予約を変更またはキャンセルしたりします。フリートの終了日を変更すると、個々のキャパシティー予約の終了日もそれに応じて更新されます。

フリートを変更すると、そのステータスは modifying に遷移します。フリートのステータスが modifying の間は、他の変更を試みることはできません。

キャパシティー予約フリートで使用されるテナンシー、アベイラビリティーゾーン、インスタンスタイプ、インスタンスプラットフォーム、優先順位、または重みを変更することはできません。これらのパラメータのいずれかを変更する必要がある場合は、既存のフリートをキャンセルし、必要なパラメータを持つ新しいフリートを作成する必要がある場合があります。

キャパシティー予約フリートは、コマンドラインのみを使用して変更できます。

キャパシティー予約フリートを変更するには

AWS CLI コマンドの modify-capacity-reservation-fleet を使用します。

> **Note**
>
> 同じコマンド内で、--end-date と --remove-end-date を指定することはできません。

```
$ aws ec2 modify-capacity-reservation-fleet \
--capacity-reservation-fleet-id cr_fleet_ids \
--total-target-capacity capacity_units \
--end-date yyyy-mm-ddThh:mm:ss.000Z \
--remove-end-date
```

正常な出力

```
{
    "Return": true
}
```

例: 総ターゲット容量の変更

```
$ aws ec2 modify-capacity-reservation-fleet \
--capacity-reservation-fleet-id crf-01234567890abcedf \
--total-target-capacity 160
```

例: 終了日の変更

```
$ aws ec2 modify-capacity-reservation-fleet \
```

```
--capacity-reservation-fleet-id crf-01234567890abcedf \
--end-date 2021-07-04T23:59:59.000Z
```

例: 終了日の削除

```
$ aws ec2 modify-capacity-reservation-fleet \
--capacity-reservation-fleet-id crf-01234567890abcedf \
--remove-end-date
```

出力例

```
{
    "Return": true
}
```

キャパシティー予約フリートをキャンセルする

キャパシティー予約フリートと予約しているキャパシティーが不要になった場合は、キャンセルできます。フリートをキャンセルすると、そのステータスが cancelled に変わり、キャパシティー予約を新たに作成することはできなくなります。さらに、フリート内の個々のキャパシティー予約はすべてキャンセルされます。また、以前にリザーブドキャパシティーで実行されていたインスタンスは、共有キャパシティーを使用して正常に実行が継続されます。

キャパシティー予約フリートは、コマンドラインのみを使用してキャンセルできます。

キャパシティー予約フリートをキャンセルするには

AWS CLI コマンドの cancel-capacity-reservation-fleet を使用します。

```
$ aws ec2 cancel-capacity-reservation-fleets \
--capacity-reservation-fleet-ids cr_fleet_ids
```

正常な出力

```
{
    "SuccessfulFleetCancellations": [
        {
            "CurrentFleetState": "state",
            "PreviousFleetState": "state",
```

```
            "CapacityReservationFleetId": "cr_fleet_id_1"
        },
        {
            "CurrentFleetState": "state",
            "PreviousFleetState": "state",
            "CapacityReservationFleetId": "cr_fleet_id_2"
        }
    ],
    "FailedFleetCancellations": [
        {
         "CapacityReservationFleetId": "cr_fleet_id_3",
         "CancelCapacityReservationFleetError": [
             {
              "Code": "code",
                    "Message": "message"
          }
         ]
     }
    ]
}
```

例: 正常なキャンセル処理

```
$ aws ec2 cancel-capacity-reservation-fleets \
--capacity-reservation-fleet-ids crf-abcdef01234567890
```

出力例

```
{
    "SuccessfulFleetCancellations": [
        {
            "CurrentFleetState": "cancelling",
            "PreviousFleetState": "active",
            "CapacityReservationFleetId": "crf-abcdef01234567890"
        }
    ],
    "FailedFleetCancellations": []
}
```

キャパシティー予約フリートでの設定例

トピック

例 1: vCPU の個数に基づいたキャパシティーの予約

次の例では、m5.4xlarge および m5.12xlarge という 2 つのインスタンスタイプを使用するキャパシティー予約フリートを作成します。

ここでは、指定されたインスタンスタイプによって提供される vCPU の数に基づく、重み付けシステムを使用しています。総ターゲット容量の vCPU 数は 480 です。m5.4xlarge により 16 個の vCPU が提供され、重みとして 16　が得られます。一方、m5.12xlarge では 48 個の vCPU が提供され 48 の重みが得られます。この重み付けシステムは、30 個の m5.4xlarge インスタンス (480/16=30)、または 10 個の m5.12xlarge インスタンス (480/48=10) でキャパシティーを予約するように、キャパシティー予約フリートを設定します。

フリートは、m5.12xlarge の容量を優先するように設定されおり優先順位として 1 を指定します。一方、m5.4xlarge には低い優先順位 2 を指定します。これは、フリートが最初にm5.12xlarge のキャパシティー予約を試みることを意味します。Amazon EC2 の m5.12xlarge キャパシティーが不十分な場合にのみ m5.4xlarge キャパシティーの予約を試みます

フリートは、Windows インスタンスでキャパシティーを予約します。この予約は、October 31, 2021 の 23:59:59 (UTC) に自動的に期限切れになります。

```
$ aws ec2 create-capacity-reservation-fleet \
--total-target-capacity 480 \
--allocation-strategy prioritized \
--instance-match-criteria open \
--tenancy default \
--end-date 2021-10-31T23:59:59.000Z \
--instance-type-specifications file://instanceTypeSpecification.json
```

instanceTypeSpecification.json の内容は次のとおりです。

```
[
    {
        "InstanceType": "m5.4xlarge",
        "InstancePlatform":"Windows",
        "Weight": 16,
        "AvailabilityZone":"us-east-1a",
        "EbsOptimized": true,
        "Priority" : 2
```

```
    },
    {
        "InstanceType": "m5.12xlarge",
        "InstancePlatform":"Windows",
        "Weight": 48,
        "AvailabilityZone":"us-east-1a",
        "EbsOptimized": true,
        "Priority" : 1
 }
]
```

キャパシティ予約フリートでのサービスにリンクされたロールの使用

オンデマンドキャパシティー予約フリートは AWS Identity and Access Management (IAM) サービスにリンクされたロール にリンクされたロールを使用します。サービスにリンクされたロールは、キャパシティー予約フリートに直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、キャパシティー予約フリートによって事前に定義され、ユーザーに代わってサービスが他の AWS のサービスを呼び出すために必要なアクセス許可がすべて含まれています。

サービスにリンクされたロールを使用することで、必要なアクセス許可を手動で追加する必要がなくなるため、キャパシティー予約フリートの設定が簡単になります。キャパシティー予約フリートは、サービスにリンクされたロールのアクセス許可を定義します。特に定義されている場合を除き、キャパシティー予約フリートのみが、そのロールを引き受けることができます。定義された権限には、信頼ポリシーと権限ポリシーに含まれており、その権限ポリシーを他の IAM エンティティにアタッチすることはできません。

サービスリンクロールは、まずその関連リソースを削除しなければ削除できません。これにより、リソースにアクセスするアクセス許可を不注意で削除する可能性がなくなるので、キャパシティー予約フリートリソースが保護されます。

キャパシティ予約フリートに対するサービスにリンクされたロールの許可

キャパシティー予約フリートでは、AWSServiceRoleForEC2CapacityReservationFleetという名前のサービスにリンクされたロールを使用します。これにより、ユーザーに代わってキャパシティー予約を作成、表示、変更でき、キャパシティー予約フリートによって以前に作成されたキャパシティー予約をキャンセルすることができます。

AWSServiceRoleForEC2CapacityReservationFleet というサービスにリンクされたロールは、以下のエンティティを信頼して `capacity-reservation-fleet.amazonaws.com` というロールを引き受けます。

このロールでは、AWSEC2CapacityReservationFleetRolePolicy というポリシーを使用します。このポリシーには次のアクセス許可が含まれています。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeCapacityReservations",
                "ec2:DescribeInstances"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateCapacityReservation",
                "ec2:CancelCapacityReservation",
                "ec2:ModifyCapacityReservation"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:capacity-reservation/*"
            ],
            "Condition": {
                "StringLike": {
                    "ec2:CapacityReservationFleet": "arn:aws:ec2:*:*:capacity-
reservation-fleet/crf-*"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:capacity-reservation/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateCapacityReservation"
                }
```

```
            }
        }
    ]
}
```

サービスリンクロールの作成、編集、削除を IAM エンティティ (ユーザー、グループ、ロールなど) に許可するには、アクセス許可を設定する必要があります。詳細については、「IAM ユーザーガイド」の「Service-Linked Role Permissions」(サービスにリンクされたロールのアクセス権限) を参照してください。

キャパシティ予約フリートでのサービスにリンクされたロールの作成

サービスリンクロールを手動で作成する必要はありません。AWS CLI コマンドの `create-capacity-reservation-fleet`、あるいは `CreateCapacityReservationFleet` API を使用してキャパシティー予約フリートを作成する場合、サービスにリンクされたロールが自動的に作成されます。

このサービスリンクロールを削除した後で再度作成する必要が生じた場合は、同じ方法でアカウントにロールを再作成できます。キャパシティー予約フリートを作成するたびに、キャパシティーの予約フリートによってサービスにリンクされたロールが自動的に作成されます。

キャパシティ予約フリートでのサービスにリンクされたロールの編集

キャパシティー予約フリートでは、サービスにリンクされたロール AWSServiceRoleForEC2CapacityReservationFleet を編集することはできません。サービスリンクロールを作成すると、多くのエンティティによってロールが参照される可能性があるため、ロール名を変更することはできません。ただし、IAM を使用したロールの説明の編集はできます。詳細については、IAM ユーザーガイドのサービスにリンクされたロールの編集を参照してください。

キャパシティ予約フリートでのサービスにリンクされたロールの削除

サービスリンクロールが必要な機能またはサービスが不要になった場合には、そのロールを削除することをお勧めします。そうすることで、モニタリングや保守が積極的に行われていない未使用のエンティティを排除できます。ただし、手動で削除する前に、サービスにリンクされたロールのリソースを削除する必要があります。

> **Note**
>
> このリソースを削除する際に、キャパシティー予約フリートのサービスでロールが使用されていると、削除処理が失敗する場合があります。失敗した場合は、数分待ってから操作を再試行してください。

サービスにリンクされたロールである AWSServiceRoleForEC2CapacityReservationFleet を削除するには

1. アカウント内のキャパシティー予約フリートを削除するには、AWS CLI コマンドの `delete-capacity-reservation-fleet` または `DeleteCapacityReservationFleet` API を使用します。
2. IAM コンソール、AWS CLI、または AWS API を使用して、サービスにリンクされたロールである AWSServiceRoleForEC2CapacityReservationFleet を削除します。詳細については、「IAM ユーザーガイド」の「サービスにリンクされたロールの削除」を参照してください。

キャパシティ予約フリートでのサービスにリンクされたロールでサポートされるリージョン

キャパシティー予約フリートでは、このサービスが利用可能なすべてのリージョンにおいて、サービスにリンクされたロールの使用をサポートしています。詳細については、「AWS リージョンとエンドポイント」を参照してください。

キャパシティ予約のモニタリング

キャパシティ予約をモニタリングするには、次の機能を使用できます。

トピック

- CloudWatch メトリクスを使用してキャパシティ予約をモニタリングする
- EventBridge を使用してキャパシティ予約をモニタリングする
- 使用率通知

CloudWatch メトリクスを使用してキャパシティ予約をモニタリングする

CloudWatch メトリクスでは、使用状況のしきい値に達したときに通知するように CloudWatch アラームを設定することにより、キャパシティーの予約 を効率的にモニタリングし、未使用の容量を特定できます。これは、一定量の キャパシティーの予約 ボリュームを維持し、ボリュームの使用効率を高めるのに役立ちます。

オンデマンドキャパシティー予約 は 5 分ごとにメトリクスデータを CloudWatch に送信します。キャパシティーの予約 は、アクティブな期間が 5 分未満のメトリクスをサポートしていません。

CloudWatch コンソールでのメトリクスの表示の詳細については、「Amazon CloudWatch メトリクスの使用」を参照してください。アラームの作成の詳細については、「Amazon CloudWatch アラームの作成」を参照してください。

コンテンツ

キャパシティーの予約 使用状況メトリクス

AWS/EC2CapacityReservations 名前空間には、以下の使用状況メトリクスが含まれています。それらのメトリクスを使用して、オンデマンド容量をモニタリングし、予約に指定したしきい値内に維持できます。

メトリクス	説明
UsedInstanceCount	現在使用中のインスタンスの数。 単位: 個
AvailableInstanceCount	使用可能なインスタンスの数。 単位: 個
TotalInstanceCount	予約済みのインスタンスの合計数。 単位: 個
InstanceUtilization	現在使用中のリザーブドキャパシティーインスタンスの割合。 単位: パーセント

キャパシティーの予約 メトリクスディメンション

以下のディメンションを使用して、前の表に示したメトリクスを絞り込むことができます。

ディメンション	説明
`CapacityReservationId`	このグローバルに一意のディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定した容量予約のものだけになります。

キャパシティーの予約 の CloudWatch メトリクスの表示

メトリクスはまずサービス名前空間ごとにグループ化され、次にサポートされているディメンションごとにグループ化されます。以下の手順を使用して キャパシティーの予約 メトリクスを表示できます。

CloudWatch コンソールを使用して キャパシティーの予約 メトリクスを表示するには

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. 必要に応じてリージョンを変更します。ナビゲーションバーから、キャパシティーの予約 があるリージョンを選択します。詳細については、「リージョンとエンドポイント」を参照してください。
3. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
4. [All metrics (すべてのメトリクス)] で、[EC2 Capacity Reservations (EC2 容量予約)] を選択します。
5. メトリクスディメンション [By Capacity Reservation (容量予約別)] を選択します。メトリクスが `CapacityReservationId` 別にグループ化されます。
6. メトリクスを並べ替えるには、列見出しを使用します。メトリクスをグラフ表示するには、メトリクスの横にあるチェックボックスを選択します。

キャパシティ予約に関するメトリクスを表示するには (AWS CLI)

次の list-metrics コマンドを使用します。

```
aws cloudwatch list-metrics --namespace "AWS/EC2CapacityReservations"
```

EventBridge を使用してキャパシティ予約をモニタリングする

アカウントのキャパシティ予約の使用率が特定の期間に 20% を下回ると、AWS Health は Amazon EventBridge にイベントを送信します。EventBridge を使用することで、このようなイベントに対応するプログラム的なアクションをトリガーするルールを設定できます。例えば、7 日間の利用率が 20% を下回った場合に、キャパシティ予約を自動的にキャンセルするルールを作成できます。

EventBridge でのイベントは、JSON オブジェクトとして表されます。イベント固有のフィールドは、JSON オブジェクトの「detail (詳細)」セクションに表示されます。「event」 フィールドにはイベント名が入ります。「result」 フィールドには、イベントをトリガーしたアクションの完了したステータスが入ります。詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントパターン」を参照してください。

詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

この機能は AWS GovCloud (US) ではサポートされていません。

コンテンツ

イベント

キャパシティ予約のキャパシティ使用率が 20% を下回ると、AWS Health は次のイベントを送信します。

イベント

- AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION
- AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY

AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION

次の例は、新しく作成されたキャパシティ予約の 24 時間のキャパシティ使用率が 20% を下回ったときに生成されるイベントです。

```
{
    "version": "0",
    "id": "b3e00086-f271-12a1-a36c-55e8ddaa130a",
    "detail-type": "AWS Health Event",
```

```
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-03-10T12:03:38Z",
    "region": "ap-south-1",
    "resources": [
        "cr-01234567890abcdef"
    ],
    "detail": {
        "eventArn": "arn:aws:health:ap-south-1::event/EC2/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_cr-01234567890abcdef-6211-4d50-9286-0c9fbc243f04",
        "service": "EC2",
        "eventTypeCode": "AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION",
        "eventTypeCategory": "accountNotification",
        "startTime": "Fri, 10 Mar 2023 12:03:38 GMT",
        "endTime": "Fri, 10 Mar 2023 12:03:38 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "A description of the event will be provided here"
            }
        ],
        "affectedEntities": [
            {
                "entityValue": "cr-01234567890abcdef"
            }
        ]
    }
    }
```

AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY

次の例は、1 つまたは複数のキャパシティ予約の 7 日間のキャパシティ使用率が 20% を下回ったときに生成されるイベントの例です。

```
{
    "version": "0", "id":"7439d42b-3c7f-ad50-6a88-25e2a70977e2",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-03-07T06:06:01Z",
    "region": "us-east-1",
    "resources": [
```

```
        "cr-01234567890abcdef | us-east-1b | t3.medium | Linux/UNIX | 0.0%",
        "cr-09876543210fedcba | us-east-1a | t3.medium | Linux/UNIX | 0.0%"
    ],
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/EC2/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY/
AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY_726c1732-d6f6-4037-b9b8-
bec3c2d3ba65",
        "service": "EC2",
        "eventTypeCode": "AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION_SUMMARY",
        "eventTypeCategory": "accountNotification",
        "startTime": "Tue, 7 Mar 2023 06:06:01 GMT",
        "endTime": "Tue, 7 Mar 2023 06:06:01 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "A description of the event will be provided
 here"
            }
        ],
        "affectedEntities": [
            {
                "entityValue": "cr-01234567890abcdef | us-east-1b | t3.medium | Linux/
UNIX | 0.0%"
            },
            {
                "entityValue": "cr-09876543210fedcba | us-east-1a | t3.medium | Linux/
UNIX | 0.0%"
            }
        ]
    }
}
```

EventBridge ルールを作成します

キャパシティ予約使用率が 20% を下回ったときに E メール通知を受け取るには、Amazon SNS トピックを作成してから、AWS_EC2_ODCR_UNDERUTILIZATION_NOTIFICATION イベントの EventBridge ルールを作成します。

Amazon SNS トピックを作成するには

1. Amazon SNS コンソール (https://console.aws.amazon.com/sns/v3/home) を開きます。

2. ナビゲーションペインで、[Topics (トピック)]、[Create topic (トピックの作成)] の順に選択します。
3. [Type (タイプ)] で、[Standard (標準)] を選択します。
4. [名前] に新しいトピックの名前を入力します。
5. [Create topic] (トピックの作成) を選択します。
6. [サブスクリプションを作成] を選択します。
7. [プロトコル] で [E メール] を選択し、次に [エンドポイント] に通知を受信する E メールアドレスを入力します。
8. [サブスクリプションを作成] を選択します。
9. 上記で入力した E メールアドレスには、「AWS Notification - Subscription Confirmation」という件名の E メールメッセージが届きます。指示に沿って操作し、登録を確認します。

EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. ナビゲーションペインで、[Rules (ルール)] を選択し、[Create rule (ルールの作成)] を選択します。
3. [名前] に新しいルールの名前を入力します。
4. ルールタイプ では、イベントパターンを持つルール] を選択します。
5. [Next] を選択します。
6. [イベントパターン] では、次のいずれかを実行します。
   a. [イベントパターンフォーム] では、AWS[サービス] を選択します。
   b. [AWS のサービス] で、[AWS Health] を選択します。
   c. [イベントタイプ] で、[EC2 ODCR 低使用率通知] を選択します。
7. [Next] を選択します。
8. [ターゲット 1] で、以下を実行します。
   a. ターゲットタイプ] では、AWSサービス] を選択します。
   b. [Select a target] (ターゲットの選択) には、[SNS topic] (SNS トピック) を選択します。
   c. [トピック] で、以前に作成したトピックを選択します。
9. [次へ] を選択し、もう一度 [次へ] を選択します。

10. [ルールを作成] を選択します。

使用率通知

アカウントのキャパシティ予約のキャパシティ使用率が 20% を下回ると、AWS Health は次の E メールと AWS Health Dashboard 通知を送信します。

- 新しく作成されたキャパシティ予約のうち、直近の 24 時間の使用率が 20% を下回ったものについての個別通知です。
- 直近の 7 日間の使用率が 20% を下回ったすべてのキャパシティ予約の概要通知です。

E メール通知および AWS Health Dashboard 通知は、キャパシティ予約を所有する AWS アカウントに関連付けられている E メールアドレスに送信されます。通知には、次の情報が含まれます。

- キャパシティーの予約の ID。
- キャパシティ予約のアベイラビリティーゾーン。
- キャパシティ予約の平均使用率。
- キャパシティ予約のインスタンスタイプとプラットフォーム (オペレーティングシステム)。

さらに、アカウントのキャパシティ予約の 24 時間のキャパシティ使用率と 7 日間のキャパシティ使用率が 20% を下回ると、AWS Health はイベントを EventBridge に送信します。EventBridge では、このようなイベントに応じて、E メール通知の送信や AWS Lambda 関数のトリガーなどの自動アクションをアクティブにするルールを作成できます。詳細については、「EventBridge を使用してキャパシティ予約をモニタリングする」を参照してください。

機械学習用のキャパシティブロック

ML 用のキャパシティブロックを使用すると、短期間の機械学習 (ML) ワークロードをサポートするために、非常に需要の高い GPU インスタンスを将来の日付で予約できます。キャパシティブロック内で実行されるインスタンスは、Amazon EC2 UltraClusters 内に自動的に互いに近く配置され、低レイテンシーでペタビットスケールのノンブロッキングネットワーキングを実現します。

キャパシティブロックを使用すると、GPU インスタンスのキャパシティを今後いつ使用できるかを確認でき、都合のよい時間にキャパシティブロックを開始するようにスケジュールできます。キャパシティブロックを予約すると、GPU インスタンスのキャパシティを予測して確保することができます。料金は必要な時間分しか発生しません。ML ワークロードを一度に数日間または数週間サポート

するために GPU が必要であり、GPU インスタンスを使用していない間は予約の料金を支払いたくないという場合は、キャパシティブロックをお勧めします。

キャパシティブロックの一般的なユースケースは以下のとおりです。

- ML モデルトレーニングと微調整 — ML モデルトレーニングと微調整を完了するために予約した GPU インスタンスに、中断なしにアクセスできます。
- ML 実験とプロトタイプ — GPU インスタンスを必要とする実験の実行およびプロトタイプの構築を短期間で行えます。

キャパシティブロックは現在、p5.48xlarge および p4d.24xlarge インスタンスで使用できます。p5.48xlarge インスタンスは、米国東部 (オハイオ) および米国東部 (バージニア北部) リージョンで使用できます。p4d.24xlarge インスタンスは、米国東部 (オハイオ) および米国西部 (オレゴン) リージョンで使用できます。キャパシティブロックは、最大 8 週間先を開始時刻に設定して予約することができます。

キャパシティブロックでは、p5 および p4d インスタンスを以下の予約期間とインスタンス数のオプションで予約できます。

- 予約期間は 1 日から 14 日間まで
- 予約インスタンスの数量は、1、2、4、8、16、32、64

キャパシティブロックを予約するには、インスタンスタイプ、必要なインスタンス数、日数、最も早い開始日、最も遅い終了日など、必要なキャパシティを最初に指定します。そうすると、その要件を満たす、利用可能なキャパシティブロックのサービスを確認できます。キャパシティブロックのサービスには、開始時刻、アベイラビリティーゾーン、予約料金などの詳細が記されています。キャパシティブロックサービスの料金は、サービスが提供される時点の需要と供給の状況によって異なります。キャパシティブロックの予約後に料金が変わることはありません。詳細については、「キャパシティブロックの料金と請求」を参照してください。

キャパシティブロックのサービスを購入すると、選択した日付とインスタンス数で予約が作成されます。キャパシティブロックの予約が開始されたら、起動リクエストで予約 ID を指定すると、インスタンスの起動をターゲットに設定できます。

予約したすべてのインスタンスを使用できるのは、キャパシティブロックの終了時刻の 30 分前までです。キャパシティブロックの予約が残り 30 分になると、キャパシティブロックで実行中のすべてのインスタンスの終了プロセスが開始されます。この時間を使ってインスタンスをクリーンアップしてから、キャパシティブロックを次の利用者に渡します。予約の最後の 30 分間は、キャパシティブ

ロックの料金に加算されません。当社は、終了プロセスが始まる 10 分前に EventBridge を通じてイベントを送信します。詳細については、「EventBridge を使用したキャパシティブロックのモニタリング」を参照してください。

トピック

サポートされているプラットフォーム

ML 用のキャパシティブロックは、現在、デフォルトテナンシーの `p5.48xlarge` および `p4d.24xlarge` インスタンスをサポートしています。AWS Management Console を使用してキャパシティブロックを購入する場合、デフォルトのプラットフォームは Linux/UNIX です。AWS Command Line Interface (AWS CLI) または AWS SDK を使用してキャパシティブロックを購入する場合、以下のプラットフォームオプションを使用できます。

- Linux/UNIX
- Red Hat Enterprise Linux
- RHEL with HA
- SUSE Linux
- Ubuntu Pro

考慮事項

キャパシティブロックを使用するときは、事前に以下の詳細と制限を念頭におきます。

- キャパシティブロックは、協定世界時 (UTC) の午前 11 時 30 分に開始および終了します。
- キャパシティブロック内で実行しているインスタンスの終了プロセスは、予約の最終日の協定世界時 (UTC) 午前 11 時に始まります。
- キャパシティブロックの開始時刻は最大 8 週間先を予約できます。
- キャパシティブロックは修正およびキャンセルはできません。

- キャパシティブロックは AWS アカウント間や AWS 組織内で共有することはできません。
- キャパシティブロックはキャパシティ予約グループでは使用できません。
- AWS 組織内の全アカウントのキャパシティブロックで予約できるインスタンスの合計数は、特定の日に 64 インスタンスを超えることはできません。
- キャパシティブロックを使用するには、インスタンスが予約 ID を明確にターゲットにしている必要があります。
- キャパシティブロック内のインスタンスは、オンデマンドインスタンスの制限にはカウントされません。

関連リソース

キャパシティブロックを作成したら、キャパシティブロックを使用して次の操作を実行できます。

- インスタンスをキャパシティブロックで起動します。「インスタンスをキャパシティブロックで起動します。」を参照してください。
- Amazon EC2 Auto Scaling グループを作成します。「Amazon EC2 Auto Scaling ユーザーガイド」の「Use Capacity Blocks for machine learning workloads」を参照してください。
- Amazon EKS セルフマネージド型ノードグループを作成します。「Amazon EKS ユーザーガイド」の「機械学習用のキャパシティブロック」を参照してください。

Amazon EC2 Auto Scaling または Amazon EKS を使用する場合は、キャパシティブロック予約の開始時にスケーリングを実行するようにスケジュールできます。スケジュールされたスケーリングでは、AWS が再試行を自動的に処理するため、一時的な障害を処理するための再試行ロジックの実装について心配する必要はありません。

キャパシティブロックの料金と請求

トピック

- 料金
- 「請求」

料金

ML 用の Amazon EC2 キャパシティブロックでは、料金は予約した分のみ発生します。キャパシティブロックの料金は、購入時のキャパシティブロックの需要と供給の状況に応じて異なります。キャパシティブロックサービスの料金は、予約前に確認できます。キャパシティブロックの料金は、

予約時に前払いで請求されます。特定の日付範囲でキャパシティブロックを検索すると、利用可能なキャパシティブロックの中で最も安価なものが表示されます。キャパシティブロックの予約後に料金が変わることはありません。

キャパシティブロックを使用する場合、インスタンスの実行時に使用した、オペレーティングシステムの料金が請求されます。Linux と Ubuntu Pro のオペレーティングシステムの料金は、秒単位で課金されます。Red Hat Enterprise Linux (RHEL)、RHEL with HA、SUSE Linux は、1 時間単位の定額料金 (最低 1 時間) で請求されます。

料金の詳細については、「Amazon EC2 Capacity Blocks for ML Pricing」を参照してください。

「請求」

キャパシティブロックサービスの料金は前払い制です。キャパシティブロックを購入すると、12 時間以内にご利用の AWS アカウントに料金が請求されます。支払いの処理中は、キャパシティブロックの予約リソースは `payment-pending` の状態となります。料金を 12 時間以内に処理できない場合、そのキャパシティブロックは解除され、予約状態は `payment-failed` に変わります。

料金の処理に成功すると、キャパシティブロックのリソース状態は `payment-pending` から `scheduled` に変わります。1 回限りの前払い料金が反映された請求書がお手元に届きます。請求書では、支払い金額をキャパシティブロックの予約 ID に関連付けることが可能です。

キャパシティブロックの予約が開始されると、その予約でインスタンスが実行されている間に使用した、オペレーティングシステムのみに基づいて料金が請求されます。ご自身の使用量および関連の料金は、AWS Cost and Usage Report で、使用した月の請求書で確認できます。

> **Note**
>
> Savings Plans とリザーブドインスタンス割引はキャパシティブロックには適用されません。

請求を表示する

請求書は AWS Billing and Cost Management コンソールで確認できます。キャパシティブロックの前払い料金は、予約の購入月に表示されます。

予約の開始後は、請求書にはブロック予約の使用時間と未使用の時間とが別々の行に表示されます。これらの行項目を使って、予約にどの程度の時間を使用したかを確認できます。プレミアムオペレーティングシステムを使用している場合は、使用時間の行には使用料金のみが表示されます。詳細については、「料金」を参照してください。未使用の時間には、追加料金は発生しません。

詳細については、AWS Billing and Cost Management ユーザーガイドの「請求書の表示」を参照してください。

キャパシティブロックの開始日が予約の購入月とは異なる場合、前払い料金と予約の使用量は、異なる請求月で表示されます。AWS Cost and Usage Report では、キャパシティブロックの予約 ID は前払い料金の Reservation/ReservationARN の行項目と、毎月の請求書の LineItem/ResourceID に記載されます。したがって、使用量を対応する前払い価格に関連付けることができます。

キャパシティブロックの操作

キャパシティブロックを使用するときは、まず、予約の規模、期間、タイミングの各ニーズに合った、利用可能なキャパシティブロックを見つけて購入します。次に、予約が始まったら、予約 ID をターゲットとするインスタンスを起動することでキャパシティブロックを使用できます。予約の有効期限が切れる 30 分前に、キャパシティブロック内でまだ実行しているインスタンスの終了プロセスを開始します。

キャパシティブロックは、単一のアベイラビリティーゾーンの `targeted` キャパシティの予約として提供されています。キャパシティブロックでインスタンスを実行するには、インスタンスの起動時に予約 ID を指定する必要があります。自分でインスタンスを停止してキャパシティブロックの有効期限が切れた場合、`active` 状態の別のキャパシティブロックをターゲットにするまで、再起動することはできません。

デフォルトでは、キャパシティブロックはキャパシティブロック内のインスタンス間に低レイテンシーで高スループットのネットワーク接続を提供するため、キャパシティブロックにクラスタープレイスメントグループを使用する必要はありません。

トピック

- 前提条件
- キャパシティブロックを見つけて購入する
- インスタンスをキャパシティブロックで起動します。
- キャパシティブロックを表示する

前提条件

使用するインスタンスタイプには、対応する AWS リージョンを使用する必要があります。詳細については、「リージョン」を参照してください。

`p5.48xlarge` インスタンスを含むキャパシティブロックは、次の AWS リージョン で入手できます。

リージョン名	リージョンコード
米国東部 (オハイオ)	us-east-2
米国東部 (バージニア北部)	us-east-1

`p4d.24xlarge` インスタンスを含むキャパシティブロックは、次の AWS リージョン で入手できます。

リージョン名	リージョンコード
米国東部 (オハイオ)	us-east-2
米国西部 (オレゴン)	us-west-2

> **Note**
>
> 64 インスタンスのキャパシティブロックサイズは、すべての AWS リージョン のすべてのインスタンスタイプでサポートされているわけではありません。

キャパシティブロックを見つけて購入する

キャパシティブロックを予約するには、まず、自分のニーズを満たすキャパシティを、利用できる時間帯を見つける必要があります。予約できるキャパシティブロックを見つけるには、以下を指定します。

- 必要なインスタンス数
- インスタンスを必要とする期間
- 予約が必要な日数の範囲

利用可能なキャパシティブロックサービスを見つけるには、予約期間とインスタンス数を指定します。次のいずれかのオプションを選択します。

- 予約期間 — 1 日単位で最大 14 日間
- インスタンス数 — 1、2、4、8、16、32、64 インスタンスのいずれか

自分の要件に合うキャパシティブロックを利用できる場合、1 件のキャパシティブロックサービスの詳細が返されます。サービスの詳細には、予約の開始時刻、予約のアベイラビリティーゾーン、予約の料金が記されています。詳細については、「料金」を参照してください。

表示されているキャパシティブロックを購入することもできますし、検索条件を変えて利用可能な他のサービスを探すこともできます。サービスの有効期限は事前に設定されていませんが、サービスの利用は申し込み順となります。

キャパシティブロックのサービスを購入すると、キャパシティブロックが予約されたことを確認する返信がすぐに届きます。その後、アカウントに新しいキャパシティ予約が、予約タイプ `capacity-block` と `start-date` が、購入したサービスの開始時刻に設定されて、表示されます。キャパシティブロックに予約は、`payment-pending` の状態で作成されます。前払い料金の処理が完了すると、予約状態は `scheduled` に変更されます。詳細については、「「請求」」を参照してください。

キャパシティブロックを探して購入するときは、次のいずれかの方法を使用します。

Console

コンソールを使ってキャパシティブロックを見つけ、購入するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、AWS リージョンを選択します。64 インスタンスのキャパシティブロックサイズは、すべてのリージョンのすべてのインスタンスタイプでサポートされているわけではないため、この選択は重要です。
3. ナビゲーションペインで [キャパシティ予約]、[キャパシティブロックの購入] を選択します。
4. [キャパシティの属性] ではキャパシティブロックの検索パラメータを定義できます。デフォルトでは、プラットフォームは Linux です。別のオペレーティングシステムを選択する場合は、AWS CLI を使用します。詳細については、「サポートされているプラットフォーム」を参照してください。
5. [合計キャパシティ] で予約するインスタンスの数を選択します。
6. [期間] に予約が必要な日数を入力します。
7. [キャパシティブロックを検索する日付範囲] で、予約の許容可能な最も早い開始日と、同じく許容可能な最も遅い終了日を入力します。
8. [キャパシティブロックを検索] を選択します。
9. 要件を満たすキャパシティブロックがある場合、[おすすめのキャパシティブロック] にそのサービスが表示されます。要件を満たすキャパシティブロックが複数ある場合、利用できる

中で最も料金の安いサービスが表示されます。他のキャパシティブロックサービスを表示するときは、検索条件を変更し、再度 [キャパシティブロックを検索] を選択します。

10. 購入したいキャパシティブロックサービスが見つかったら、[次へ] を選択します。
11. (オプション) [タグを追加] ページで、[新しいタグを追加] を選択します。
12. [確認と購入] ページに、開始日と終了日、期間、インスタンスの合計数、料金が表示されます。

> **Note**
>
> 予約後は、キャパシティブロックを変更したりキャンセルしたりすることはできません。

13. ポップアップウィンドウの [キャパシティブロックを購入] で [確認] を選択し、[購入] を選択します。

AWS CLI

AWS CLI を使ってキャパシティブロックを見つけるには

`describe-capacity-block-offerings`コマンドを実行します。

以下の例では、16 個の `p5.48xlarge` インスタンスを含み、日付範囲が `2023-08-14` から `2023-10-22` まで、期間が 48 時間のキャパシティブロックを検索します。インスタンス数は、整数で、事前定義された選択肢 (1、2、4、8、16、32、64) のいずれかである必要があります。キャパシティの期間は、整数で、24 から 336 までの24の倍数でなければならず、日数を時間単位で表記します。

```
aws ec2 describe-capacity-block-offerings --instance-type p5.48xlarge \
  --instance-count 16 --start-date-range 2023-08-14T00:00:00Z \
  --end-date-range 2023-10-22-T00:00:00Z --capacity-duration 48
```

AWS CLI を使ってキャパシティブロックを購入するには

`purchase-capacity-block` コマンドを使用して、購入するキャパシティブロックのサービス ID とインスタンスプラットフォームとを指定します。

```
aws ec2 purchase-capacity-block \
  --capacity-block-offering-id cbr-0123456789abcdefg \
```

```
  --instance-platform Linux/UNIX
```

インスタンスをキャパシティブロックで起動します。

キャパシティブロックを予約すると、AWS アカウントにキャパシティブロックの予約が表示されます。start-date と end-date を見ると、予約の開始日と終了日を確認できます。キャパシティブロック予約が始まるまで、利用可能なキャパシティは 0 と表示されます。キャパシティブロックで利用できるインスタンスの数は、タグキー aws:ec2capacityreservation:incrementalRequestedQuantity のタグ値を見ると確認できます。

キャパシティブロックの予約が始まると、予約状態は scheduled から active に変わります。Amazon EventBridge を通じて、キャパシティブロックが使用可能になったことを知らせるイベントが発行されます。詳細については、「キャパシティブロックのモニタリング」を参照してください。

キャパシティブロックを使用するには、インスタンスの起動時にキャパシティブロックの予約 ID を指定する必要があります。キャパシティブロックでインスタンスを起動すると、起動したインスタンスの数だけ、使用できるキャパシティの数が減ります。例えば、購入したインスタンスのキャパシティが 8 インスタンスで、4 つのインスタンスを起動した場合、使用できるキャパシティは 4 つ減ります。

予約が終了する前にキャパシティブロックで実行中のインスタンスを終了すると、新しいインスタンスを代わりに起動することができます。キャパシティブロック内のインスタンスを停止または終了すると、インスタンスのクリーンアップに数分かかります。置き換える別のインスタンスを起動できるのは、その後です。この間、インスタンスは停止または shutting-down 状態になります。このプロセスが完了すると、インスタンスの状態が stopped か terminated に変わります。その後、キャパシティブロックの利用可能な容量が更新され、使用できる別のインスタンスが表示されます。

以下のステップでは、active 状態のキャパシティブロックで AWS Management Console または AWS CLI を使用してインスタンスを起動する方法を説明します。

開始時に自動的にキャパシティブロックを使用するように EKS ノードグループを設定する方法については、「Amazon EKS ユーザーガイド」の「機械学習用のキャパシティブロック」を参照してください。

EC2 フリートを使用してキャパシティブロックでインスタンスを起動する方法については、「チュートリアル: キャパシティブロックでインスタンスを起動する」を参照してください。

キャパシティブロックをターゲットとする起動テンプレートの作成方法については、「起動テンプレートからのインスタンスの起動」を参照してください。

キャパシティブロックでインスタンスを起動するには、次のいずれかの方法に従います。

Console

コンソールを使用してキャパシティブロックでインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面上部のナビゲーションバーで、キャパシティブロックの予約の [リージョン] を選択します。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択します。
4. (オプション) [名前とタグ] では、インスタンスに名前を付けたりタグを付けたりできます。タグの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。
5. [アプリケーションと OS イメージ] で、Amazon マシンイメージ (AMI) を選択します。
6. [インスタンスタイプ] で、自分のキャパシティブロックの予約と一致するインスタンスタイプを選択します。
7. [キーペア (ログイン)] で既存のキーペアを選択するか、[新しいキーペアを作成] を選択して新しいキーペアを作成します。詳細については、「Amazon EC2 のキーペアと Amazon EC2 インスタンス」を参照してください。
8. [Network settings] (ネットワーク設定) で、デフォルト設定を使用するか、[Edit] (編集) を選択して必要に応じてネットワーク設定を構成します。

   > **Important**
   >
   > キャパシティブロックがあるアベイラビリティーゾーンとは異なるアベイラビリティーゾーンのサブネットでインスタンスを起動することはできません。

9. [高度な設定] で、インスタンスを次のように設定します。
   a. [購入オプション (マーケットタイプ)] で [キャパシティブロック] を選択します。
   b. [キャパシティ予約] で [ID 別のターゲット] を選択します。
   c. キャパシティブロック予約のキャパシティ予約 ID を選択します。
10. [Summary] (概要) パネルの [Number of instances] (インスタンス数) に、起動するインスタンス数を入力します。
11. [インスタンスを起動] を選択します。

AWS CLI

AWS CLI を使用してキャパシティブロックでインスタンスを起動するには

- run-instances コマンドを使用して、instance-market-options 構造の capacity-block の MarketType を指定します。また、capacity-reservation-specification パラメータを指定する必要があります。

  以下の例では、属性と使用可能なキャパシティとが一致するアクティブなキャパシティブロックで 1 つの p5.48xlarge インスタンスを起動します。

```
aws ec2 run-instances --image-id ami-abc12345 --count 1 \
  --instance-type p5.48xlarge --key-name MyKeyPair \
  --subnet-id subnet-1234567890abcdef1 \
  --instance-market-options MarketType='capacity-block'
  --capacity-reservation-specification
 CapacityReservationTarget={CapacityReservationId=cr-a1234567}
```

キャパシティブロックを表示する

キャパシティブロックには、以下のような状態があります。

- payment-pending — 前払い料金の処理が完了していない。
- payment-failed — 前払い料金の処理が 12 時間以内に完了しなかった。キャパシティブロックが解除された。
- scheduled — 料金の処理は完了したが、キャパシティブロックの予約はまだ始まっていない。
- active - リザーブドキャパシティを使用できる。
- expired — キャパシティブロックの予約の有効期限が予約リクエストで指定された日時に自動的に切れた。リザーブドキャパシティー も使用できなくなります。

キャパシティブロック予約は、次のいずれかの方法で表示できます。

Console

コンソールを使用してキャパシティブロックを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キャパシティーの予約] を選択します。

3. [キャパシティ予約の概要] ページに、すべてのキャパシティ予約リソースの詳細が記載されたリソーステーブルが表示されます。キャパシティブロック予約を検索するには、[キャパシティ予約 ID] の上にあるドロップダウンリストから[キャパシティブロック] を選択します。表には、開始日、終了日、期間、状態など、キャパシティブロックに関する情報が表示されています。
4. キャパシティブロックの詳細は、表示するキャパシティブロックの予約 ID を選択すると、表示されます。[キャパシティ予約の詳細] ページには、予約のすべてのプロパティと、キャパシティブロックで使用中かつ使用可能なインスタンスの数が表示されています。

> **Note**
>
> キャパシティブロック予約が始まるまで、利用可能なキャパシティは 0 と表示されます。キャパシティブロックが開始されたときに利用できるインスタンスの数は、タグキー `aws:ec2capacityreservation:incrementalRequestedQuantity` の以下のタグ値を使用して確認できます。

AWS CLI

AWS CLI を使用してキャパシティブロックを表示するには

デフォルトでは、describe-capacity-reservations コマンドを使用すると、オンデマンドキャパシティ予約とキャパシティブロック予約の両方が一覧表示されます。キャパシティブロック予約のみを表示するには、`capacity-reservation-type` パラメータに `capacity-block` を適用して絞り込みます。

例えば、次のコマンドは、現在の AWS リージョン にあるキャパシティブロック予約を 1 つ以上記述します。

```
aws ec2 describe-capacity-reservations -reservation-type capacity-block
```

出力例。

```
{
    "CapacityReservations": [
        {
            "CapacityReservationId": "cr-12345678",
            "EndDateType": "limited",
            "ReservationType": "capacity-block"
```

```
            "AvailabilityZone": "eu-east-2a",
            "InstanceMatchCriteria": "targeted",
            "EphemeralStorage": false,
            "CreateDate": "2023-11-29T14:22:45Z",
            "StartDate": "2023-12-15T12:00:00Z",
            "EndDate": "2023-08-19T12:00:00Z",
            "AvailableInstanceCount": 0,
            "InstancePlatform": "Linux/UNIX",
            "TotalInstanceCount": 16,
            "State": "payment-pending",
            "Tenancy": "default",
            "EbsOptimized": true,
            "InstanceType": "p5.48xlarge"
        },
        ...
```

キャパシティブロックのモニタリング

トピック

- EventBridge を使用したキャパシティブロックのモニタリング
- AWS CloudTrail を使用したキャパシティブロック API コールのロギング

EventBridge を使用したキャパシティブロックのモニタリング

キャパシティブロックの予約が始まると、Amazon EC2 は EventBridge を通じて、キャパシティが使用可能になったことを知らせるイベントを送信します。キャパシティブロック予約の終了 40 分前になると、予約で実行中のインスタンスが 10 分後に終了プロセスを開始することを知らせる、別のEventBridge イベントが手元に届きます。EventBridge イベントの詳細については、「Amazon EventBridge イベント」を参照してください。

キャパシティブロックに関して発生するイベントのイベント構造を以下に示します。

キャパシティブロックの配信

以下に示す例は、キャパシティブロックの配信のイベントです。

```
{
  "customer_event_id": "[Capacity Reservation Id]-delivered",
  "detail_type": "Capacity Block Reservation Delivered",
  "source": "aws.ec2",
  "account": "[Customer Account ID]",
```

```
  "time": "[Current time]",
  "resources": [
    "[ODCR ARN]"
  ],
  "detail": {
    "capacity-reservation-id": "[ODCR ID]",
    "end-date": "[ODCR End Date]"
  }
}
```

キャパシティブロックの有効期限切れの警告

以下に示す例は、キャパシティブロックの有効期限切れの警告のイベントです。

```
{
  "customer_event_id": "[Capacity Reservation Id]-approaching-expiry",
  "detail_type": "Capacity Block Reservation Expiration Warning",
  "source": "aws.ec2",
  "account": "[Customer Account ID]",
  "time": "[Current time]",
  "resources": [
    "[ODCR ARN]"
  ],
  "detail": {
    "capacity-reservation-id": "[ODCR ID]",
    "end-date": "[ODCR End Date]"
  }
}
```

AWS CloudTrail を使用したキャパシティブロック API コールのロギング

キャパシティブロックは、キャパシティブロックのユーザー、ロール、AWS サービスが実行したアクションを記録する AWS CloudTrail と連携しています。CloudTrail は、キャパシティブロックの API コールをイベントとしてキャプチャします。キャプチャされたコールには、キャパシティブロックコンソールからのコールと、キャパシティブロック API オペレーションへのコードコールが含まれています。証跡を作成する場合は、キャパシティブロックのイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [イベント履歴] で最新のイベントを表示できます。CloudTrail で収集された情報を使用して、キャパシティブロックに対するリクエスト、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

CloudTrail の詳細については、AWS CloudTrail ユーザーガイドを参照してください。

CloudTrail のキャパシティブロック情報

CloudTrail は、AWS アカウントを作成すると、その中で有効になります。キャパシティブロックでアクティビティが発生すると、そのアクティビティは [イベント履歴] で AWS のその他のサービスのイベントとともに CloudTrail イベントにレコードされます。最近のイベントは、AWS アカウント で表示、検索、ダウンロードできます。詳細については、 CloudTrail イベント履歴でのイベントの表示を参照してください。

イベント履歴のイベントなど、AWS アカウント のイベントの継続的な記録については、証跡を作成します。証跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョン に適用されます。証跡は、AWS パーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、CloudTrail ログで収集したイベントデータをより詳細に分析し、それに基づく対応するためにその他の AWS サービスを設定できます。詳細については、次を参照してください:

- 「追跡の作成の概要」
- CloudTrail がサポートされているサービスおよび統合
- CloudTrail の Amazon SNS 通知の設定
- 「複数のリージョンから CloudTrail ログファイルを受け取る」および「複数のアカウントから CloudTrail ログファイルを受け取る」

すべてのキャパシティブロックアクションは CloudTrail によって ログ記録され、Amazon EC2 API リファレンスに記録されます。例えば、`CapacityBlockScheduled` と `CapacityBlockActive` の各アクションを呼び出すと、CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するために役立ちます。

- リクエストが、ルート認証情報と AWS Identity and Access Management (IAM) ユーザー認証情報のどちらを使用して送信されたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが、別の AWS サービスによって送信されたかどうか。

詳細については、「CloudTrail userIdentity 要素」を参照してください。

キャパシティブロックのログファイルエントリについて

追跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルは、単一か複数のログエントリを含みます。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストのパラメータなどの情報が含まれます。CloudTrail ログファイルは、パブリック API 呼び出しの順序付けられたスタックトレースではないため、特定の順序では表示されません。

以下は、次の CloudTrail ログエントリの例です。

- TerminateCapacityBlocksInstances
- CapacityBlockPaymentFailed
- CapacityBlockScheduled
- CapacityBlockActive
- CapacityBlockFailed
- CapacityBlockExpired

> **Note**
>
> こちらの例では、データのプライバシーを保護するため一部のフィールドが削除されています。

TerminateCapacityBlocksInstances


```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "TerminateCapacityBlockInstances",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
```

```
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "accountId": "123456789012",
      "type": "AWS::EC2::Instance",
      "ARN": "arn:aws:ec2:US East (N. Virginia):123456789012:instance/
i-1234567890abcdef0"
    }
    {
      "accountId": "123456789012",
      "type": "AWS::EC2::Instance",
      "ARN": "arn:aws::ec2:US East (N. Virginia):123456789012:instance/
i-0598c7d356eba48d7"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      }
}
```

CapacityBlockPaymentFailed

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockPaymentFailed",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
```

```
      "ARN": "arn:aws:ec2:US East (N. Virginia):123456789012:capacity-reservation/
cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "payment-failed"
      }
}
```

CapacityBlockScheduled

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockScheduled",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:US East (N. Virginia):123456789012:capacity-reservation/
cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
```

```
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "scheduled"
      }
}
```

CapacityBlockActive

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockActive",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:US East (N. Virginia):123456789012:capacity-reservation/
cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "active"
      }
 }
```

CapacityBlockFailed

```
{
  "eventVersion": "1.05",
```

```
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockFailed",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:US East (N. Virginia):123456789012:capacity-reservation/
cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "failed"
      }
 }
```


CapacityBlockExpired

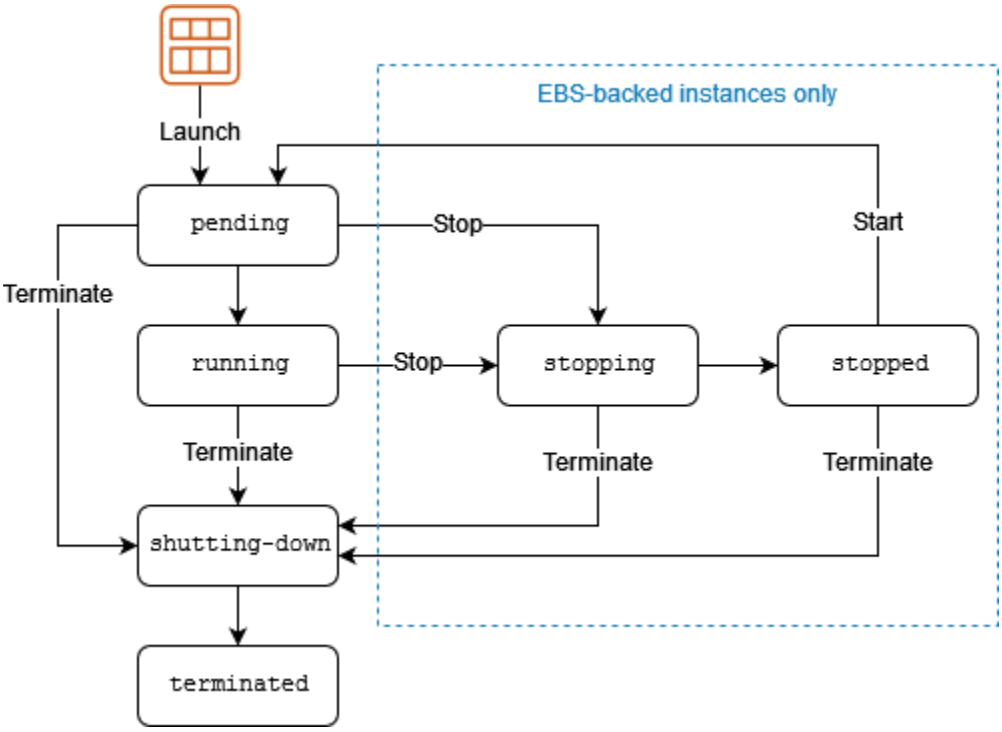

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "123456789012",
    "invokedBy": "AWS Internal;"
  },
  "eventTime": "2023-10-02T00:06:08Z",
  "eventSource": "ec2.amazonaws.com",
  "eventName": "CapacityBlockExpired",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "203.0.113.25",
```

```
  "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
  "requestParameters": null,
  "responseElements": null,
  "eventID": "a1b2c3d4-EXAMPLE",
  "readOnly": false,
  "resources": [
    {
      "ARN": "arn:aws:ec2:US East (N. Virginia):123456789012:capacity-reservation/
cr-12345678",
      "accountId": "123456789012",
      "type": "AWS::EC2::CapacityReservation"
    }
  ],
  "eventType": "AwsServiceEvent",
  "recipientAccountId": "123456789012",
  "serviceEventDetails": {
      "capacityReservationId": "cr-12345678",
      "capacityReservationState": "expired"
      }
 }
```


インスタンスのライフサイクル

Amazon EC2 インスタンスは、起動時から終了まで、さまざまな状態に移行します。

次の図は、インスタンス状態の遷移を示しています。instance store-backed インスタンスは停止および起動できないことに注意してください。instance store-backed インスタンスの詳細については、「ルートデバイスのストレージ」を参照してください。



各インスタンスの状態の概要と、請求有無を次の表に示します。

> **Note**
>
> この表では、インスタンス使用のみの請求を示します。Amazon EBS ボリュームや Elastic IP アドレスなど一部の AWS リソースでは、インスタンスの状態に関係なく利用料金が発生します。詳細については、AWS Billing ユーザーガイドの「予想外の料金の回避」を参照してください。

インスタンスの状態	説明	インスタンス使用の請求
pending	インスタンスは running 状態への移行準備中です。インスタンスは、起動時または stopped 状態になってから起動する	非請求対象

インスタンスの状態	説明	インスタンス使用の請求
	と、pending 状態になります。	
running	インスタンスは実行中で、使用できる状態です。	請求済み付
stoppir	インスタンスは停止の準備中です。	非請求対象
stopped	インスタンスはシャットダウンされているため、使用できません。インスタンスはいつでも起動できます。	課金されない
shuttir down	インスタンスは削除準備中です。	課金されない
termina d	インスタンスは完全に削除されているため、起動することはできません。	課金されない **Note** 終了したインスタンスに適用されるリザーブドインスタンスは、支払いオプションに従って、契約期間末まで請求が発生します。詳細については、「Reserved Instances」を参照してください。

Note

インスタンスは running 状態のため、インスタンスを再起動しても新しいインスタンスの請求期間が開始されることはありません。

トピック

インスタンスの作成

インスタンスを起動すると、インスタンスはpending状態に移行します。起動時に指定したインスタンスタイプによって、インスタンスのホストコンピュータのハードウェアが決定します。起動時に指定されたAmazon Machine Image (AMI) を使って、インスタンスを再作成します。インスタンスの準備ができると、running 状態へ移行します。実行中のインスタンスに接続して、自分の前にあるコンピュータと同じように使用することができます。

インスタンスが running 状態に遷移するとすぐに、インスタンスの実行時間に応じて (インスタンスがアイドル状態のままで、接続されていなくても) 最低 1 分以上の秒単位で使用料金が発生します。

詳細については、「インスタンスの起動Linux インスタンスへの接続」および「」を参照してください。

インスタンスの停止と起動 (Amazon EBS-Backed インスタンスのみ)

インスタンスのステータスチェックに失敗するか、インスタンスでアプリケーションが想定通りに動作しておらず、インスタンスのルートボリュームが Amazon EBS である場合、インスタンスの停止と起動を行い、問題が解決するか試してみることができます。

インスタンスを停止した場合、インスタンスはstopping状態に移行してから、stopped状態になります。インスタンスの使用料やデータ転送料金は、stopped 時点では請求されません。どのAmazon EBS ボリュームのストレージにも料金が発生します。インスタンスがstopped状態の間、インスタンスタイプなど、インスタンスの特定の属性を変更できます。

インスタンスを起動して pending 状態になると、インスタンスは新しいホストコンピュータに移動します (ただし、場合によっては、インスタンスは現在のホストに残ることもあります)。インスタンスの停止と起動を行うと、以前のホストコンピュータに接続されていたインスタンスストアボリュームのすべてのデータが失われます。

インスタンスはプライベートIPv 4アドレスを保持します。つまり、プライベートIPv 4アドレスまたはネットワークインターフェイスに関連付けられた Elastic IPアドレスは、インスタンスに関連付けられたままになります。インスタンスに IPv6 アドレスがある場合、IPv6 アドレスは保持されます。

stopped から running に移行したインスタンスについては、その実行中に秒単位の料金が発生します。また、インスタンスの起動時には、1 分間分の最低料金が課金されます。

インスタンスの停止と開始の詳細については、「インスタンスの停止と起動」を参照してください。

インスタンスの休止 (Amazon EBS Backed インスタンスのみ)

インスタンスを休止すると、オペレーティングシステムに休止を実行するように合図します (ディスクの停止)。これにより、内容がインスタンスのメモリ (RAM) から Amazon EBS ルートボリュームに保存されます。インスタンスの Amazon EBS ルートボリュームとアタッチされた Amazon EBS データボリュームは保持されます。インスタンスを起動すると、Amazon EBS ルートボリュームは以前の状態に復元され、RAM の内容が再ロードされます。以前にアタッチされたデータボリュームは再アタッチされ、インスタンスはそのインスタンス ID を保持します。

インスタンスを休止した場合、インスタンスは stopping 状態に移行してから、stopped 状態になります。休止状態にあるインスタンスが stopped 状態にある間は料金は課金されませんが、(休止せずにインスタンスを停止した場合とは異なり) stopping 状態にある間は料金が発生します。データ転送料金に対して使用料を課金しませんが、RAM データのストレージを含め、Amazon EBS ボリュームのストレージに対しては課金します。

休止状態のインスタンスを起動して pending 状態になると、インスタンスは新しいホストコンピュータに移動されます (ただし、場合によっては、インスタンスが現在のホストに残ることもあります)。

プライベート IPv4 アドレスは保持されます。つまり、プライベート IPv4 アドレスまたはネットワークインターフェイスに関連付けられていた Elastic IP アドレスは、インスタンスとの関連付けが

継続されるということです。インスタンスに IPv6 アドレスがある場合、IPv6 アドレスは保持されます。

詳細については、Amazon EC2 インスタンスの休止 を参照してください。

インスタンスの再起動

Amazon EC2 コンソール、コマンドラインツール、Amazon EC2 API を使って、インスタンスを再起動できます。インスタンスからオペレーティングシステムの再起動コマンドを実行する代わりに、Amazon EC2 を使ってインスタンスを再起動することをお勧めします。

インスタンスの再起動は、オペレーティングシステムの再起動と同等です。インスタンスは同じホストコンピュータに残り、そのパブリック DNS 名、プライベート IP アドレス、およびその他のデータをインスタンスストアボリュームに維持します。通常、再起動が完了するまでに数分かかりますが、再起動に必要な時間は、インスタンスの設定によって異なります。

インスタンスを再起動しても、新しいインスタンスの課金時間は開始されず、最低 1 分間分の料金はなしで秒単位の課金が継続します。

詳細については、インスタンスの再起動 を参照してください。

インスタンスのリタイア

インスタンスをホストしている基盤のハードウェアで回復不可能な障害が検出されると、AWS によってインスタンスのリタイアが予定されます。予定されたリタイア日になると、インスタンスは AWS によって停止または削除されます。インスタンスのルートデバイスが Amazon EBS ボリュームである場合、インスタンスは停止されますが、その後いつでも再び起動できます。インスタンスのルートデバイスがインスタンスストアボリュームである場合、インスタンスは削除し、再び使用することはできません。

詳細については、インスタンスのリタイア を参照してください。

インスタンスの削除

インスタンスが必要なくなったら、削除することができます。インスタンスのステータスが `shutting-down` または `terminated` に変わったら、そのインスタンスへの課金は停止します。

停止保護が有効な場合、コンソール、CLI、または API を使用してインスタンスを削除することはできません。

インスタンスの削除後、インスタンスはしばらくの間コンソールに表示されたままですが、エントリは自動的に削除されます。CLI および API を使って、削除したインスタンスを記述することもできま

す。(タグなどの) リソースは削除されたインスタンスから徐々に関連付けが解除されるため、しばらくすると、削除されたインスタンスで表示されなくなる可能性があります。削除したインスタンスへの接続や復旧はできません。

Amazon EBS-Backed インスタンスはそれぞれ、`InstanceInitiatedShutdownBehavior` 属性をサポートしています。この属性は、インスタンス自体からシャットダウンを開始した場合 (Linux で shutdown コマンドを使用した場合など) 、インスタンスを停止または終了するかを制御します。デフォルトの動作は、インスタンスの停止です。インスタンスの実行中または停止中に、この属性の設定を変更できます。

各 Amazon EBS ボリュームは `DeleteOnTermination` 属性をサポートします。この属性は、アタッチされたインスタンスを終了するときに、ボリュームの削除や保持を制御します。デフォルトでは、ルートデバイスボリュームを削除し、それ以外に EBS ボリュームがあれば保持します。

詳細については、インスタンスの終了 を参照してください。

再起動、停止、休止、削除の違い

次の表に、インスタンスの再起動、停止、休止、終了の主な違いをまとめました。

特徴	再起動	停止/開始 (Amazon EBS-Backed インスタンスのみ)	休止 (Amazon EBS Backed インスタンスのみ)	終了
ホストコンピュータ	インスタンスは、同じホストコンピュータで保持される	インスタンスは新しいホストコンピュータに移動されます (ただし、場合によっては、インスタンスが現在のホストに残ることもあります) 。	インスタンスは新しいホストコンピュータに移動されます (ただし、場合によっては、インスタンスが現在のホストに残ることもあります) 。	なし
プライベート IPv4 アドレスとパブリック	同一のまま保持される	インスタンスはプライベート IPv4 アドレスを保持します。インスタンスは、Elastic IP アドレス (停止/起動の際	インスタンスはプライベート IPv4 アドレスを保持します。インスタンスは、Elastic IP アドレス (停止/起動の際	なし

特徴	再起動	停止/開始 (Amazon EBS-Backed インスタンスのみ)	休止 (Amazon EBS Backed インスタンスのみ)	終了
IPv4 アドレス		に変更されない) を持っていない限り、新しいパブリック IPv4 アドレスを取得します。	に変更されない) を持っていない限り、新しいパブリック IPv4 アドレスを取得します。	
Elastic IP アドレス (IPv4)	Elastic IP アドレスは、インスタンスに関連付けられたまま維持される	Elastic IP アドレスは、インスタンスに関連付けられたまま維持される	Elastic IP アドレスは、インスタンスに関連付けられたまま維持される	Elastic IP アドレスはインスタンスの関連付けが解除される
IPv6 アドレス	インスタンスは、IPv6 アドレスを保持する	インスタンスは、IPv6 アドレスを保持する	インスタンスは、IPv6 アドレスを保持する	なし
インスタンスストアボリューム	データは保持される	データは消去される	データは消去される	データは消去される
ルートデバイスボリューム	ボリュームは保持される	ボリュームは保持される	ボリュームは保持される	ボリュームはデフォルトで削除される
RAM (メモリの内容)	RAM は消去される	RAM は消去される	RAM はルートボリュームにあるファイルに保存される	RAM は消去される

特徴	再起動	停止/開始 (Amazon EBS-Backed インスタンスのみ)	休止 (Amazon EBS Backed インスタンスのみ)	終了
「請求」	インスタンスの課金時間は変更されません。	インスタンスの状態が stopping に変わるとすぐに、そのインスタンスへの課金が停止されます。インスタンスが stopped 状態から running 状態に移行するたびに新しいインスタンスの課金時間が開始され、インスタンスの開始時には 1 分間の最低料金が発生します。	インスタンスが stopping 状態にある間は課金されますが、そのインスタンスが stopped 状態にある場合、課金は停止します。インスタンスが stopped 状態から running 状態に移行するたびに新しいインスタンスの課金時間が開始され、インスタンスの開始時には 1 分間の最低料金が発生します。	インスタンスの状態が shutting-down に変わるとすぐに、そのインスタンスに対して課金されなくなります。

オペレーティングシステムのシャットダウンコマンドを実行すると、instance store-backed インスタンスは必ず停止されます。オペレーティングシステムのシャットダウンコマンドによって Amazon EBS-backed インスタンスを停止または終了するかどうかを制御できます。詳細については、インスタンスによって起動されたシャットダウン動作の変更 を参照してください。

インスタンスの起動

インスタンスは、AWS クラウド内の仮想サーバーです。Amazon Machine Image (AMI) からインスタンスを起動します。AMI はインスタンスに対して、オペレーティングシステム、アプリケーションサーバー、およびアプリケーションを提供します。

AWS にサインアップすると、AWS 無料利用枠を利用して、Amazon EC2 を無料で使い始めることができます。無料利用枠を使用し、t2.micro インスタンスを 12 か月間無料で起動して利用できます (t2.micro が利用できないリージョンでは、無料利用枠で t3.micro インスタンスを使用できます)。無料利用枠に含まれないインスタンスを起動する場合は、そのインスタンスの通常の

Amazon EC2 使用料がかかります。詳細については、「Amazon EC2 料金表」を参照してください。

次の方法を使用してインスタンスを起動できます。

方法	ドキュメント
[Amazon EC2 コンソール] インスタンス起動ウィザードを使用して、起動パラメータを指定します。	古いインスタンス起動ウィザードを使用してインスタンスを起動する
[Amazon EC2 コンソール] 起動テンプレートを作成して、起動テンプレートからインスタンスを起動します。	起動テンプレートからのインスタンスの起動
[Amazon EC2 コンソール] 既存のインスタンスを基本として使用します。	既存のインスタンスのパラメータを使用したインスタンスの起動
[Amazon EC2 コンソール] AWS Marketplace から購入した AMI を使用します。	AWS Marketplace インスタンスの起動
[AWS CLI] 選択した AMI を使用します。	AWSCLI 経由で Amazon EC2 を使用する
[AWS Tools for Windows PowerShell] 選択した AMI を使用します。	Amazon EC2 の提供元AWS Tools for Windows PowerShell
[AWS CLI] EC2 フリートを使用すると、容量のプロビジョニングを異なる EC2 インスタンスタイプとアベイラビリティーゾーン間で行うことも、オンデマンドインスタンス、リザーブドインスタンス、スポットインスタンスの各購入モデル全体で行うこともできます。	EC2 Fleet
[AWS CloudFormation] AWS CloudFormation テンプレートを使用してインスタンスを指定します。	AWS::EC2::Instance()AWS CloudFormation ユーザーガイド
[AWS SDK] 言語固有の AWS SDK を使用してインスタンスを起動します。	AWS SDK for .NET AWS SDK for C++

方法	ドキュメント
	AWS SDK for Go AWSSDK for Java AWS SDK for JavaScript 「AWS SDK for PHP V3」 AWS SDK for Python AWS SDK for Ruby V3

> **Note**
>
> EC2 インスタンスを IPv6 専用サブネットで起動するには、Nitro System 上に構築されたインスタンス を使用する必要があります。

> **Note**
>
> IPv6 専用インスタンスを起動すると、DHCPv6 がインスタンスに IPv6 DNS ネームサーバーをすぐに提供しないことがあります。この初期遅延の間、インスタンスはパブリックドメインを解決できない場合があります。
> Amazon Linux 2 で実行されるインスタンスの場合、/etc/resolv.conf ファイルを IPv6 DNS ネームサーバーで直ちに更新するには、起動時に次の cloud-init directive コマンドを実行します:
>
> ```
> #cloud-config
> bootcmd:
> - /usr/bin/sed -i -E 's,^nameserver\s+[\.[:digit:]]+$,nameserver
>  fd00:ec2::253,' /etc/resolv.conf
> ```
>
> もう 1 つのオプションは、ブート時にファイルが IPv6 DNS ネームサーバーアドレスを直ちに持つように、設定ファイルを変更して AMI を再イメージ化することです。

インスタンスを起動する場合、次のいずれかのリソースに関連付けられているサブネットでインスタンスを起動できます。

- アベイラビリティーゾーン - このオプションはデフォルトです。
- ローカルゾーン - ローカルゾーンでインスタンスを起動するには、ローカルゾーンにオプトインし、このゾーンにサブネットを作成する必要があります。詳細については、「Local Zones」を参照してください。
- Wavelength Zone - Wavelength Zone でインスタンスを起動するには、Wavelength Zone にオプトインし、このゾーンにサブネットを作成する必要があります。Wavelength Zone でインスタンスを起動する方法については、AWS Wavelength デベロッパーガイドのAWS Wavelength の開始方法をご参照ください。
- アウトポスト - アウトポストでインスタンスを起動するには、アウトポストを作成する必要があります。Outpost の作成方法の詳細については、AWS Outposts ユーザーガイドの AWS Outposts の開始方法をご参照ください。

インスタンスを起動した後、インスタンスに接続して使用できます。最初、インスタンスの状態は `pending` です。インスタンスの状態が `running` の場合、インスタンスは起動を開始します。インスタンスに接続するまで、少し時間がかかることがあります。ベアメタルインスタンスタイプの起動には時間がかかることがあります。ベアメタルインスタンスの詳細については、「Nitro System 上に構築されたインスタンス」を参照してください。

インスタンスは、パブリック DNS 名を受信します。この DNS 名はインターネットからインスタンスに接続する場合に使用できます。また、インスタンスはプライベート DNS 名も受け取ります。これは、同じ VPC 内の他のインスタンスがインスタンスに接続するために使用できます。インスタンスへの接続の詳細については、Linux インスタンスへの接続を参照してください。

インスタンスを使い終わったら、必ずインスタンスを終了してください。詳細については、インスタンスの終了 を参照してください。

新しいインスタンス起動ウィザードを使用してインスタンスを起動する

新しいインスタンス起動ウィザードを使用してインスタンスを起動できます。インスタンス起動ウィザードでは、インスタンスの起動に必要な起動パラメータを指定します。インスタンスの起動ウィザードでデフォルト値が用意されている場合、デフォルト値を使用するか、独自の値を指定できます。デフォルト値をそのまま使用すると、キーペアだけを選択してインスタンスを起動できます。

インスタンスを起動する前に、セットアップが終了していることを確認してください。詳細については、Amazon EC2 を使用するようにセットアップする を参照してください。

> **Important**
>
> AWS 無料利用枠に含まれないインスタンスを起動すると、アイドル状態であっても、インスタンスの実行中は料金が発生します。

トピック

- インスタンスをすばやく起動する
- 定義済みのパラメータを使用したインスタンスの起動
- 古いインスタンス起動ウィザードを使用してインスタンスを起動する

インスタンスをすばやく起動する

テスト目的でインスタンスをすばやくセットアップするには、次のステップに従います。オペレーティングシステムとキーペアを選択し、デフォルト値を受け入れます。インスタンス起動ウィザードのすべてのパラメータについては、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。

インスタンスをすばやく起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面の上のナビゲーションバーで、現在の AWS リージョンが表示されます (例: 米国東部 (オハイオ))。インスタンスを起動するリージョンを選択します。一部の Amazon EC2 リソースはリージョン間で共有できるため、この選択は重要です。詳細については、リソースの場所 を参照してください。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択します。
4. (オプション) [Names and tags] (名前とタグ) における [Name] (名前) では、インスタンス用にわかりやすい名前を入力します。
5. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で [Quick Start] (クイックスタート) を選択し、インスタンスのオペレーティングシステム (OS) を選択します。
6. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。
7. [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

定義済みのパラメータを使用したインスタンスの起動

キーペアを除き、インスタンス起動ウィザードはすべてのパラメータのデフォルト値を提供します。デフォルトの一部またはすべてを受け入れるか、各パラメータに独自の値を指定してインスタンスを設定することができます。パラメータは、インスタンス起動ウィザードでグループ化されます。次の手順では、各パラメータグループについて説明します。

インスタンス設定のパラメータ

- インスタンスの起動開始
- 名前とタグ
- アプリケーションと OS イメージ (Amazon マシンイメージ)
- インスタンスタイプ
- キーペア (ログイン)
- ネットワーク設定
- ストレージの設定
- 高度な詳細
- [概要]

インスタンスの起動開始

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面の上のナビゲーションバーで、現在の AWS リージョンが表示されます (例: 米国東部 (オハイオ))。インスタンスを起動するリージョンを選択します。一部の Amazon EC2 リソースはリージョン間で共有できるため、この選択は重要です。詳細については、リソースの場所 を参照してください。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択します。

名前とタグ

インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。インスタンス、ボリューム、ネットワークインターフェイスにタグ付けできます。スポットインスタンスの場合、スポットインスタンスリクエストにのみタグを付けることができます。タグの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。

インスタンス名と追加のタグを指定することはオプションです。

- [Name] (名前) に、インスタンスのわかりやすい名前を入力します。名前を指定しない場合は、インスタンスをその ID で識別できます。ID は、インスタンスの起動時に自動的に生成されます。
- タグを追加するには、[Add additional tag] (追加のタグを追加) を選択します。[Add tag] (タグを追加) を選択し、キーと値を入力し、タグ付けするリソースタイプを選択します。追加するタグごとに [Add tag] (タグを追加) を選択します。

アプリケーションと OS イメージ (Amazon マシンイメージ)

Amazon マシンイメージ (AMI) には、インスタンスの作成に必要な情報が含まれています。例えば、ある AMI には、ウェブサーバーとして動作するために必要なソフトウェア (Linux、Apache、ウェブサイトなど) が含まれています。

適切な AMI は、次の手順で確認できます。AMI を検索する各オプションで、[Cancel] (キャンセル) (右上) を選択すれば、AMI を選択せずにインスタンス起動ウィザードに戻ることができます。

検索バー

利用可能なすべての AMI を検索するには、AMI 検索バーにキーワードを入力し、[Enter] キーを押します。AMI を選択するには、[Select] (選択) を選択します。

Recents (最新情報)

最近使用した AMI が表示されます。

[Recently launched] (最近の起動) または [Currently in use] (現在使用中) を選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から AMI を選択します。

マイ AMI

お客様が所有しているプライベート AMI、またはお客様が共有しているプライベート AMI。

[Owned by me] (ユーザーによる所有) または [Shared with me] (共有されている) を選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から AMI を選択します。

クイックスタート

AMI はオペレーティングシステム (OS) ごとにグループ化されているため、すぐに作業を開始できます。

まず、必要な OS を選択し、次に [Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) で、AMI を選択します。無料利用枠の対象となる AMI を選択するには、AMI が [Free tier eligible] (無料利用枠の対象) とマークされていることを確認してください。

Browse more AMIs (AMI をさらに表示する)

AMI カタログ全体を表示するには、[Browse more AMIs] (AMI をさらに表示する) を選択します。

- 利用可能な AMI すべてを検索するには、検索バーにキーワードを入力し、[Enter] キーを押します。
- Systems Manager パラメータを使用して AMI を検索するには、検索バーの右側にある矢印ボタンを選択し、[Search by Systems Manager parameter] (Systems Manager パラメータで検索) を選択します。詳細については、「Systems Manager パラメータを使用した AMI の検索」を参照してください。
- カテゴリで検索するには、[Quickstart AMIs] (AMI のクイックスタート)、[My AMIs] (私の AMI)、[AWS Marketplace AMIs]、または [Community AMIs] (コミュニティ AMI) を選択します。

  AWS Marketplace は、AMI を含む AWS 上で動作するソフトウェアを購入することができるオンラインストアです。AWS Marketplace からのインスタンスの起動の詳細については、AWS Marketplace インスタンスの起動 を参照してください。[Community AMIs] (コミュニティ AMI) では、AWS のコミュニティのメンバーが他の人が利用可能とした AMI を見つけることができます。Amazon または検証済みパートナーからの AMI は、[Verified provider] (検証済みプロバイダー) のマークが付されます。
- AMI のリストをフィルターするには、画面左の [Refine results] (結果を絞り込む) で 1 つまたは複数のチェックボックスをオンにします。フィルターオプションは、選択した検索カテゴリに応じて異なります。
- 各 AMI の [Root device type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは [ebs] (Amazon EBS でバックアップ) または [instance-store] (インスタンスストアでバックアップ) です。詳細については、ルートデバイスのストレージ を参照してください。
- 各 AMI の [Virtualization type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは [hvm] または [paravirtual] です。例えば、一部のインスタンスタイプには HVM が必要です。詳細については、「Linux AMI 仮想化タイプ」を参照してください。
- 各 AMI に記載された [Boot Mode] (起動モード) を確認します。必要なブートモードがどの AMI を使用するのか注意してください。必要なブートモードは [legacy-bios]、[uefi]、または [uefi-preferred] です。詳細については、「ブートモード」を参照してください。
- ニーズを満たす AMI を選択し、[Select] を選択します。

AMI を変更するときに警告します

選択した AMI に関連付けられているボリュームまたはセキュリティグループの設定を変更し、別の AMI を選択すると、現在の設定の一部が変更または削除されることを警告するウィンドウが開きます。セキュリティグループおよびボリュームに対する変更を確認できます。さらに、追加および削除されるボリュームを表示することも、追加されるボリュームのみを表示することもできます。

インスタンスタイプ

インスタンスタイプは、インスタンスのハードウェア設定とサイズを定義します。インスタンスタイプが大きくなると、CPU およびメモリも増えます。詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。

- [Instance type] (インスタンスタイプ) で、インスタンスのインスタンスタイプを選択します。

  無料利用枠 – AWS アカウントを作成してから 12 か月未満の場合は、[t2.micro] インスタンスタイプ (または [t2.micro] が利用できないリージョンでは [t3.micro] インスタンスタイプ) を選択すると、無料利用枠で Amazon EC2 を使用できます。インスタンスタイプが無料利用枠の下で適格である場合、それは [Free tier eligible] (無料利用枠適格) とラベル付けされます。t2.micro と t3.micro の詳細については、「バーストパフォーマンスインスタンス」を参照してください。
- [Compare instance types] (インスタンスタイプの比較): vCPU の数、アーキテクチャ、メモリ量 (GiB)、ストレージ量 (GB)、ストレージタイプ、ネットワークパフォーマンスなどの属性ごとにさまざまなインスタンスタイプを比較できます。
- [アドバイスの取得]: インスタンスタイプに関するガイダンスやアドバイスは、Amazon Q EC2 インスタンスタイプセレクターから入手できます。詳細については、「新しいワークロードのインスタンスタイプに関する推奨事項の取得」を参照してください。

キーペア (ログイン)

[Key pair name] (キーペア名) には、既存のキーペアを選択するか、[Create new key pair] (新しいキーペアを作成) を選択して新しいキーペアを作成します。詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンス を参照してください。

> **Important**
>
> [Proceed without key pair] (キーペアなしで進む) オプションを選択した場合 (非推奨)、ユーザーが別の方法でログインすることを許可するように設定された AMI を選択した場合でなければ、インスタンスに接続できなくなります。

ネットワーク設定

必要に応じて、ネットワーク設定を設定します。

- VPC: インスタンスに既存の VPC を選択します。デフォルト VPC を使用するか、作成した VPC を選択します。詳細については、「the section called “仮想プライベートクラウド”」を参照してください。
- [サブネット]: インスタンスは、アベイラビリティーゾーン、ローカルゾーン、Wavelength Zone、Outpost のいずれかに関連付けられたサブネットで起動できます。

  アベイラビリティーゾーンでインスタンスを起動するには、インスタンスを起動するサブネットを選択します。新しいサブネットを作成するには、[Create new subnet] を選択して Amazon VPC コンソールに移動します。終了したらインスタンス起動ウィザードに戻り、[Refresh] (更新) アイコンを選択して一覧にサブネットを読み込みます。

  IPv6 のみのサブネットでインスタンスを起動するには、Nitro System 上に構築されたインスタンスである必要があります。

  ローカルゾーンでインスタンスを起動するには、ローカルゾーン内に作成したサブネットを選択します。

  アウトポストでインスタンスを起動するには、アウトポストに関連付けられた VPC 内のサブネットを選択します。
- [Auto-assign Public IP]: インスタンスがパブリック IPv4 アドレスを受け取るかどうかを指定します。デフォルトでは、デフォルトのサブネットにあるインスタンスはパブリック IPv4 アドレスを受け取りますが、デフォルト以外のサブネットにあるインスタンスは受け取りません。[Enable] または [Disable] を選択すると、これがサブネットのデフォルト設定より優先されます。詳細については、パブリック IPv4 アドレス を参照してください。
- [Firewall (security groups)] (ファイアウォール (セキュリティグループ)): セキュリティグループを使用してインスタンスのファイアウォールルールを定義します。このルールでは、どの着信ネットワークトラフィックをインスタンスに配信するかを指定します。他のトラフィックはすべて無視されます。セキュリティグループの詳細については、Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ を参照してください。

  ネットワークインターフェイスを追加する場合、ネットワークインターフェイスに同じセキュリティグループを指定する必要があります。

  次のようにセキュリティグループを選択または作成します。

- VCP に既存のセキュリティグループを選択するには、[Select existing security group] (既存のセキュリティグループを選択) を選択し、[Common security groups] (共通セキュリティグループ) からセキュリティグループを選択します。
- VCP に新しいセキュリティグループを作成するには、[Create security group] (セキュリティグループの作成) を選択します。インスタンス起動ウィザードは、launch–wizard-x セキュリティグループを自動的に定義し、セキュリティグループルールをすばやく追加するために次のチェックボックスを提供します。

  [Allow SSH traffic from] (次のものから SSH トラフィックを許可) — [SSH (port 22)] (SSH (ポート 22)) (RDP (ポート3389)) にインスタンスへの接続を許可するインバウンドルールを作成します。トラフィックが[Anywhere] (どこでも)、[Custom] (カスタム)、または [My IP] (マイ IP) から来るのかどうかを指定します。

  [Allow HTTPs traffic from the internet] (インターネットから HTTPS トラフィックを許可) - 任意の場所からのインターネットトラフィックを許可するポート 443 (HTTPS) を開くインバウンドルールを作成します。インスタンスがウェブサーバーである場合、このルールが必要です。

  [Allow HTTP traffic from the internet] (インターネットから HTTPS トラフィックを許可) - 任意の場所からのインターネットトラフィックを許可するポート 80 (HTTP) を開くインバウンドルールを作成します。インスタンスがウェブサーバーである場合、このルールが必要です。

  ニーズに応じてこれらのルールを編集してルールを追加できます。

  ルールを編集または追加するには、[Edit] (編集) を選択します (右上)。ルールを追加するには、[Add security group rule] (セキュリティグループルールの追加) を選択します。[Type] (タイプ) で、ネットワークトラフィックタイプを選択します。[Protocol] (プロトコル) フィールドには、ネットワークトラフィックの送信を可能とするため、プロトコルが自動的に入力されます。[Source type] (送信元タイプ) で送信元のタイプを選択します。[My IP] (マイ IP) を選択し、インスタンス起動ウィザードでコンピュータのパブリック IP アドレスを追加します。ただし、ISP 経由で、またはファイアウォールの内側から静的な IP アドレスなしで接続している場合は、クライアントコンピュータで使用されている IP アドレスの範囲を見つける必要があります。

  > ⚠ Warning
  >
  > すべての IP アドレス (`0.0.0.0/0`) から SSH や RDP でインスタンスにアクセスできるようにするルールは、テスト用のインスタンスを短時間で立ち上げ、すぐに停止または

終了させる場合には許容されますが、本番環境では危険です。特定の IP アドレスまたは特定のアドレス範囲にのみ、インスタンスへのアクセスを限定してください。

- [Advanced network configuration] (アドバンストネットワーク設定) — サブネットを選択した場合のみ使用できます。

  ネットワークインターフェイス

  - [Device index] (デバイスインデックス): ネットワークカードのインデックス。プライマリネットワークインターフェイスは、ネットワークカードインデックス 0 に割り当てる必要があります。インスタンスタイプによっては、複数のネットワークカードがサポートされているものもあります。
  - [Network Interface] (ネットワークインターフェイス): [New interface] (新しいインターフェイス) を選択して Amazon EC2 によって新しいインターフェイスを作成するか、既存の使用できるネットワークインターフェイスを選択します。
  - [説明]: (オプション) 新しいネットワークインターフェイスの説明。
  - [Subnet] (サブネット): 新しいネットワークインターフェイスを作成するサブネット。プライマリネットワークインターフェイス (`eth0`) の場合、これはインスタンスが起動する先のサブネットです。`eth0` に既存のネットワークインターフェイスを入力すると、インスタンスはネットワークインターフェイスが存在するサブネット内で起動します。
  - [セキュリティグループ]: ネットワークインターフェイスを関連付ける VPC 内の 1 つ以上のセキュリティグループ。
  - [プライマリ IP]: サブネットの範囲からの プライマリプライベート IPv4 アドレス。Amazon EC2 によって自動的にプライベート IPv4 アドレスが選択されるようにするには、空白のままにします。
  - [Secondary IP] (セカンダリ IP): サブネットの範囲内にある 1 つまたは複数の追加のプライベート IPv4 アドレス。[Manually assign] (手動割り当て) を選択し、IP アドレスを入力します。別の IP アドレスを追加するには、[Add IP] (IP の追加) を選択します。または、Amazon EC2 により自動で割り当てるようにするには、[Automatically assign] (自動割り当て) を選択し、追加する IP アドレスの数を入力します。
  - (IPv6 のみ) [IPv6 IP]: サブネットの範囲の IPv6 アドレス。[Manually assign] (手動割り当て) を選択し、IP アドレスを入力します。別の IP アドレスを追加するには、[Add IP] (IP の追加) を選択します。または、Amazon EC2 により自動で割り当てるようにするには、[Automatically assign] (自動割り当て) を選択し、追加する IP アドレスの数を入力します。
  - [IPv4 Prefixes] (IPv4 プレフィクス): ネットワークインターフェイスの IPv4 プレフィクス。
  - [IPv6 Prefixes] (IPv6 プレフィクス): ネットワークインターフェイスの IPv6 プレフィクス。

- (デュアルスタックおよび IPv6 のみ) プライマリ IPv6 IP の割り当て: (オプション) インスタンスをデュアルスタックまたは IPv6 のみのサブネットで起動する場合、プライマリ IPv6 IP を割り当てるオプションがあります。プライマリ IPv6 アドレスを割り当てると、インスタンスまたは ENI へのトラフィックの中断を回避できます。このインスタンスが IPv6 アドレスが変更されないことに依存する場合、[有効化] を選択します。インスタンスを起動すると、AWS ではアタッチされている ENI に関連付けられた IPv6 アドレスがインスタンスにプライマリ IPv6 アドレスとして自動的に割り当てられます。IPv6 GUA アドレスをプライマリ IPv6 として有効にすると、無効にすることはできません。IPv6 GUA アドレスをプライマリ IPv6 にすることを有効にすると、インスタンスが終了するか、ネットワークインターフェースがデタッチされるまで、最初の IPv6 GUA がプライマリ IPv6 アドレスになります。インスタンスに複数の IPv6 アドレスがアタッチされていて、プライマリ IPv6 アドレスを有効にすると、ENI に関連付けられた最初の IPv6 GUA アドレスがプライマリ IPv6 アドレスになります。
- [終了時に削除]: インスタンス終了時にネットワークインターフェイスを削除するかどうか。
- Elastic Fabric Adapter: ネットワークインターフェイスが Elastic Fabric Adapter かどうかを示します。詳細については、「Elastic Fabric Adapter」を参照してください。
- ENA Express: ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD テクノロジーは、パケットスプレーメカニズムを使用して負荷を分散し、ネットワークの混雑を回避します。ENA Express を有効にすると、サポートされているインスタンスは、可能な場合は通常の TCP トラフィックに加えて SRD を使用して通信できるようになります。リストから [有効化] または [無効化] を選択しない限り、インスタンス起動ウィザードにはインスタンスの ENA Express 設定は含まれません。
- [ENA Express UDP]: ENA Express を有効にしている場合は、オプションで UDP トラフィックに使用できます。[有効化] または [無効化] を選択しない限り、インスタンス起動ウィザードにはインスタンスの ENA Express 設定は含まれません。

さらにネットワークインターフェイスを追加するには、[ネットワークインターフェイスの追加] を選択します。追加のネットワークインターフェイスは、同じ VPC の別のサブネット、または所有している別の VPC のサブネットに配置できます (サブネットがインスタンスと同じアベイラビリティーゾーンにある場合)。別の VPC サブネットに存在するネットワークインターフェイスを追加する場合は、サブネットを選択すると [マルチ VPC サブネット] オプションが表示されます。別の VPC でサブネットを選択すると、追加したネットワークインターフェイスの横に [マルチ VPC] ラベルが表示されます。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。別の VPC から追加の ENI をアタッチする場合は、その VPC から ENI のセキュリティグループを選択する必要があります。

詳細については、「Elastic Network Interface」を参照してください。複数のネットワークインターフェイスを指定した場合、インスタンスはパブリック IPv4 アドレスを受け取ることはできません。さらに、eth0 に既存のネットワークインターフェイスを指定した場合、[Auto-assign Public IP] を使用してサブネットのパブリック IPv4 設定をオーバーライドする操作は禁止されます。詳細については、インスタンス起動時のパブリック IPv4 アドレスの割り当て を参照してください。

ストレージの設定

選択した AMI には、ルートボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。インスタンスにアタッチする追加のボリュームを指定できます。

[Simple] (シンプル) または [Advanced] (アドバンスト) ビューを使用できます。[Simple] (シンプル) ビューでは、ボリュームのサイズとタイプを指定します。すべてのボリュームパラメータを指定するには、[Advanced] (アドバンスト) ビュー (カードの右上) を選択します。

[Advanced] (アドバンスト) ビューでは、各ボリュームを以下のように設定できます。

- [Storage type] (ストレージタイプ): インスタンスと関連付ける Amazon EBS またはインスタンスストアボリュームを選択します。一覧で利用できるボリュームタイプは、選択したインスタンスタイプに応じて異なります。詳細については、「Amazon EC2 インスタンスストア」および「Amazon EBS ボリューム」を参照してください。
- [Device name] (デバイス名): ボリュームで利用できるデバイス名の一覧から選択します。
- [Snapshot] (スナップショット): ボリュームを復元するスナップショットを選択します。[Snapshot] (スナップショット) フィールドにテキストを入力して、利用できる共有スナップショットとパブリックスナップショットを検索することもできます。
- [Size (GiB)] (サイズ (GiB)): EBS ボリュームの場合、ストレージサイズを指定できます。無料利用枠の対象となる AMI とインスタンスを選択した場合でも、無料利用枠内に収めるには、合計ストレージを 30 GiB 以下に維持する必要があることに注意してください。
- [Volume type] (ボリュームタイプ): EBS ボリュームの場合、ボリュームタイプを選択します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。
- [IOPS]: Provisioned IOPS SSD ボリュームタイプを選択した場合は、ボリュームがサポートできる I/O オペレーション/秒 (IOPS) を入力できます。
- [Delete on termination] (終了時に削除): Amazon EBS ボリュームで、インスタンスの終了時にボリュームを削除する場合は [Yes] (はい) を選択し、ボリュームを保持する場合は [No] (いいえ) を選択します。詳細については、インスタンスの終了時にデータを保持する を参照してください。

- [Encrypted] (暗号化): インスタンスタイプが EBS 暗号化をサポートしている場合、[Yes] (はい) を選択し、ボリュームの暗号化を有効にできます。このリージョンでデフォルトで暗号化を有効にした場合、暗号化は有効になります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。
- [KMS Key] (KMS キー): [Encrypted] (暗号化) で [Yes] (はい) を選択し、ボリュームで暗号化を使用する場合には、カスタマーマネージド型キーを選択する必要があります。このリージョンでデフォルトの暗号化を有効にした場合は、自動的にデフォルトのカスタマーマネージド型キーが選択されます。別のキーを選択するか、作成したカスタマーマネージド型キーの ARN を指定できます。
- [File systems] (ファイルシステム): Amazon EFS または Amazon FSx ファイルシステムをインスタンスにマウントします。Amazon EFS ファイルシステムのマウントの詳細については、「Amazon EC2 での Amazon EFS の使用」を参照してください。Amazon FSx ファイルシステムのマウントの詳細については、「Amazon EC2 での Amazon FSx の使用」を参照してください。

高度な詳細

[Advanced details] で、セクションを開いてフィールドを表示し、インスタンスの追加パラメータを指定します。

- [Purchasing option] (購入オプション): [Request Spot Instances] (スポットインスタンスのリクエスト) を選択して、オンデマンド価格を上限とするスポット料金でスポットインスタンスをリクエストし、[Customize] (カスタマイズ) を選択して、スポットインスタンスのデフォルト設定を変更します。上限料金を設定し (非推奨)、リクエストタイプ、リクエスト期間、中断動作を変更できます。スポットインスタンスをリクエストしない場合、Amazon EC2 はデフォルトでオンデマンドインスタンスを起動します。詳細については、「スポットインスタンスリクエストを作成する」を参照してください。
- [ドメイン結合ディレクトリ]: 起動後の Linux インスタンスを結合する先の AWS Directory Service ディレクトリ (ドメイン) を選択します。ドメインを選択する場合は、必要なアクセス許可を持つ IAM ロールを選択する必要があります。詳細については、「Linux EC2 インスタンスを AWS Managed Microsoft AD ディレクトリにシームレスに結合する」を参照してください。
- [IAM instance profile] (IAM インスタンスプロファイル): インスタンスに関連付ける AWS Identity and Access Management (IAM) インスタンスプロファイルを選択します。詳細については、Amazon EC2 の IAM ロール を参照してください。
- [Hostname type] (ホスト名タイプ): インスタンスのゲスト OS ホスト名をリソース名または IP 名に含めるかどうかを選択します。詳細については、Amazon EC2 インスタンスのホスト名タイプ を参照してください。

- [DNS Hostname] (DNS ホスト名): リソース名または IP 名への DNS クエリが、([Hostname type] (ホスト名タイプ) に何を選択したのかによって) IPv4 アドレス (A レコード)、IPv6 アドレス (AAAA レコード)、またはその両方で応答するかどうかを決定します。詳細については、Amazon EC2 インスタンスのホスト名タイプ を参照してください。
- [Shutdown behavior]: シャットダウン時にインスタンスを停止するか終了するかを選択します。詳細については、インスタンスによって起動されたシャットダウン動作の変更 を参照してください。
- [Stop - Hibernate behavior] (停止 – 休止動作): 休止を有効にするには、[Enable] (有効) を選択します。このフィールドは、インスタンスが休止の前提条件を満たしている場合にのみ使用できます。詳細については、Amazon EC2 インスタンスの休止 を参照してください。
- [Termination protection] (終了の保護): 偶発的な終了を防ぐには、[Enable] (有効) を選択します。詳細については、「終了保護を有効化する」を参照してください。
- 停止保護: 偶発的な停止を防ぐには、[Enable] (有効化) を選択します。詳細については、「停止保護を有効にします」を参照してください。
- [Detailed CloudWatch monitoring] (詳細な CloudWatch モニタリング): Amazon CloudWatch を使用したインスタンスの詳細なモニタリングをオンにする場合、[Enable] (有効) を選択します。別途料金がかかります。詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。
- [Elastic inference]: EC2 CPU インスタンスにアタッチする Elastic Inference アクセラレータ。詳細については、Amazon Elastic Inference デベロッパーガイドの「Working with Amazon Elastic Inference の使用」を参照してください。

> **Note**
>
> 2023 年 4 月 15 日以降、AWS では Amazon Elastic Inference (EI) への新規顧客のオンボーディングは行わず、既存の顧客がより価格とパフォーマンスの良いオプションにワークロードを移行できるよう支援します。2023 年 4 月 15 日以降、新規顧客は Amazon SageMaker、Amazon ECS、または Amazon EC2 の Amazon EI アクセラレータを使用してインスタンスを起動できなくなります。ただし、過去 30 日間に Amazon EI を少なくとも 1 回使用した顧客は、現在の顧客と見なされ、サービスを引き続き使用できます。

- [Credit specification] (クレジット指定): アプリケーションがベースラインを越えて必要なだけバーストできることを有効にするには、[Unlimited] (無制限) を選択します。このフィールドは、T インスタンスでのみ有効です。追加料金が適用される場合があります。詳細については、バーストパフォーマンスインスタンス を参照してください。

- [プレイスメントグループ名]: インスタンスを起動する先のプレイスメントグループを指定します。既存のプレイスメントグループを選択するか、新しいプレイスメントグループを作成することができます。すべてのインスタンスタイプが、プレイスメントグループでのインスタンスの起動をサポートしているわけではありません。詳細については、プレイスメントグループ を参照してください。
- [EBS-optimized instance] (EBS 最適化インスタンス): Amazon EBS に最適化されたインスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O に対して追加の専用容量を提供します。ご使用のインスタンスタイプでこの機能がサポートされている場合は、[Enable] (有効) を選択して有効にします。別途 料金がかかります。詳細については、「the section called “EBS 最適化”」を参照してください。
- [Capacity Reservation] (キャパシティ予約): インスタンスを起動するキャパシティ予約を指定します。任意のキャパシティ予約 ([Open] (オープン))、特定のキャパシティ予約 ([Target by ID] (ID を対象とする))、またはキャパシティ予約グループ ([Target by group] (グループを対象とする)) のいずれかから選択します。キャパシティ予約を使用しないように指定するには、[None] (なし) を選択します。詳細については、既存の キャパシティーの予約 へのインスタンスの起動 を参照してください。
- [テナンシー]: インスタンスを共有ハードウェア ([共有])、独立した専有ハードウェア ([専有])、あるいは Dedicated Host ([Dedicated host (専有ホスト)])で実行するかを選択します。Dedicated Host でインスタンスを起動する場合は、インスタンスをホストリソースグループ内で起動するかどうかを指定できます。または、特定の Dedicated Host をターゲットとして設定できます。追加料金が適用される場合があります。詳細については、Dedicated InstancesおよびDedicated Hostsを参照してください。
- [RAM disk ID] (RAM ディスク ID): (準仮想化 (PV) AMI に対してのみ有効) インスタンスの RAM ディスクを選択します。カーネルを選択した場合は、サポートするドライバーと共に特定の RAM ディスクを選択しなければならない可能性があります。
- [Kernel ID] (カーネル ID): (準仮想化 (PV) AMI に対してのみ有効) インスタンスのカーネルを選択します。
- [Nitro Enclaves]: Amazon EC2 インスタンスから、エンクレーブと呼ばれる分離された実行環境を作成することを許可します。AWS Nitro Enclaves のインスタンスを有効にするには、[Enable] (有効) を選択します。詳細については、「AWS Nitro Enclaves ユーザーガイド」の「AWS Nitro Enclaves とは」を参照してください。
- [ライセンス設定]: 指定したライセンス設定に対してインスタンスを起動して、ライセンスの使用状況を追跡できます。詳細については、AWS License Manager ユーザーガイドの「Create a license configuration」(ライセンス設定の作成) を参照してください。

- [Metadata accessible]: インスタンスメタデータへのアクセスを有効または無効にできます。詳細については、新規インスタンスのインスタンスメタデータオプションの設定 を参照してください。
- [Metadata transport] (メタデータの転送): インスタンスがリンクローカル IMDSv2 IPv6 アドレス (`fd00:ec2::254`) に到達できるようにして、インスタンスのメタデータを取得します。このオプションは、IPv6 専用サブネットで Nitro System 上に構築されたインスタンス を起動している場合にのみ使用できます。インスタンスメタデータの取得の詳細については、「インスタンスメタデータの取得」を参照してください。
- [Metadata version]: インスタンスメタデータへのアクセスを有効にする場合、インスタンスメタデータをリクエストするときに インスタンスメタデータサービスバージョン 2 の使用を必須にすることができます。詳細については、新規インスタンスのインスタンスメタデータオプションの設定 を参照してください。
- [メタデータレスポンスのホップ制限]: インスタンスメタデータを有効にする場合、メタデータトークンに許容されるネットワークホップ数を設定できます。詳細については、新規インスタンスのインスタンスメタデータオプションの設定 を参照してください。
- [Allow tags in metadata] (メタデータ内のタグを許可する): [Enable] (有効) を選択した場合、インスタンスはメタデータ内のすべてのタグへのアクセスを許可します。値を指定しない場合、インスタンスメタデータ内のタグへのアクセスはデフォルトで無効になります。詳細については、インスタンスメタデータのタグへのアクセスを許可する を参照してください。
- [ユーザーデータ]: 起動時にインスタンスを設定するユーザーデータ、または設定スクリプトを実行するユーザーデータを指定できます。詳細については、「起動時に Linux インスタンスでコマンドを実行する」を参照してください。

[概要]

[Summary] (サマリー) パネルを使用して、起動するインスタンスの数を指定し、インスタンス構成を確認し、インスタンスを起動します。

- [Number of instances]: 起動するインスタンスの数を入力します。すべてのインスタンスは、同じ設定で起動します。

> **Tip**
>
> インスタンスの起動を高速化するには、大きなリクエストをより小さなバッチに分割します。例えば、1 つの起動リクエストに 500 インスタンスが含まれている場合は、それを 5 つの起動リクエスト (各 100 インスタンス) に分割します。

- (オプション) 複数のインスタンスを指定した場合は、アプリケーションの要求に対処できるだけのインスタンス数が確保されるように、[consider EC2 Auto Scaling] (EC2 Auto Scaling を考慮) を選択して起動テンプレートと Auto Scaling グループを作成することができます。Auto Scaling によって、指定どおりにグループのインスタンス数がスケーリングされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」を参照してください。

  > **Note**
  >
  > Amazon EC2 Auto Scaling が Auto Scaling グループ内のインスタンスを異常とマークすると、そのインスタンスの置き換えが自動的にスケジュールされます。この場合、インスタンスは終了されて別のインスタンスが起動され、元のインスタンスのデータは失われます。インスタンスを停止または再起動するか、別のイベントがインスタンスを異常としてマークすると、インスタンスは異常としてマークされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インスタンスのヘルスチェック」を参照してください。

- インスタンスの詳細を確認し、必要な変更を加えます。[Summary] (サマリー) パネルのリンクを選択すると、セクションに直接移動することができます。
- インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択します。

  インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

  (オプション) インスタンスの請求アラートを作成できます。確認画面で、[Next Steps] (次のステップ) で [Create billing alerts] (請求アラートの作成) を選択し、指示に従います。請求アラートは、インスタンスの起動後に作成することもできます。詳細については、「Amazon CloudWatch ユーザーガイド」の「推定の AWS 料金をモニタリングする請求アラームを作成」を参照してください。

古いインスタンス起動ウィザードを使用してインスタンスを起動する

古いインスタンス起動ウィザードを使用してインスタンスを起動できるのは、リージョンが古い起動エクスペリエンスをサポートしている場合のみです。インスタンスの起動ウィザードでは、インスタンスの起動に必要なすべての起動パラメータを指定します。インスタンスの起動ウィザードでデフォルト値が用意されている場合、デフォルト値を使用するか、独自の値を指定できます。インスタンスを起動するには、AMI とキーペアを指定する必要があります。

新しいインスタンス起動ウィザードの使用については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

インスタンスを起動する前に、セットアップが終了していることを確認してください。詳細については、Amazon EC2 を使用するようにセットアップする を参照してください。

> **Important**
>
> AWS 無料利用枠に含まれないインスタンスを起動すると、アイドル状態であっても、インスタンスの実行中は料金が発生します。

インスタンスを起動するためのステップ:

- インスタンスの起動開始
- ステップ 1: Amazon Machine Image (AMI) を選択する
- ステップ 2: インスタンスタイプを選択する
- ステップ 3: インスタンスの詳細を設定する
- ステップ 4: ストレージを追加する
- ステップ 5: タグの追加
- ステップ 6: セキュリティグループを設定する
- ステップ 7: インスタンスの起動を確認し、キーペアを選択する

インスタンスの起動開始

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 画面の上のナビゲーションバーで、現在のリージョンが表示されます (例: US East (Ohio))。ニーズを満たすインスタンスのリージョンを選択します。一部の Amazon EC2 リソースはリージョン間で共有できるため、この選択は重要です。詳細については、リソースの場所 を参照してください。
3. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択します。

ステップ 1: Amazon Machine Image (AMI) を選択する

インスタンスを起動するときに、Amazon Machine Image (AMI) と呼ばれる設定を選択する必要があります。AMI には、新しいインスタンスの作成に必要な情報が含まれています。例えば、ある AMI

には、ウェブサーバーとして動作するために必要なソフトウェア (Linux、、Apache、ウェブサイトなど) が含まれています。

インスタンスを作成する場合は、リストから AMI を選択するか、AMI ID をポイントする Systems Manager パラメータを選択することができます。詳細については、「Systems Manager パラメータを使用した AMI の検索」を参照してください。

[Amazon マシンイメージ (AMI)] ページで、AMI を選択する 2 つの方法のいずれかを使用します。その方法は、AMI のリストから探すか、Systems Manager パラメータで探すかです。

AMI のリストから探す

1. 左ペインで、使用する AMI のタイプを選択します。

   クイックスタート

   すぐに作業を開始できるように、一般的な AMI を選択します。無料利用枠の対象となる AMI を選択するには、左ペインで [無料利用枠のみ] を選択します これらの AMI は [Free tier eligible] と表示されます。

   マイ AMI

   お客様が所有しているプライベート AMI、またはお客様が共有しているプライベート AMI。共有している AMI を表示するには、左ペインの [自分と共有] を選択します。

   AWS Marketplace

   AMI も含めて、AWS で実行するソフトウェアを購入できるオンラインストア。AWS Marketplace からのインスタンスの起動の詳細については、AWS Marketplace インスタンスの起動 を参照してください。

   コミュニティ AMI

   AWS コミュニティのメンバーが、メンバー以外でも使用できるようにした AMI。オペレーティングシステムを条件として AMI のリストをフィルタリングするには、[Operating system] の該当するチェックボックスをオンにします。アーキテクチャおよびルートデバイスタイプを条件としてフィルタリングすることもできます。

2. 各 AMI の [Root device type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは `ebs` (Amazon EBS でバックアップ) または `instance-store` (インスタンスストアでバックアップ) です。詳細については、ルートデバイスのストレージ を参照してください。

3. 各 AMI の [Virtualization type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは hvm または paravirtual です。例えば、一部のインスタンスタイプには HVM が必要です。詳細については、「Linux AMI 仮想化タイプ」を参照してください。
4. 各 AMI の [Boot Mode] を確認します。必要なブートモード (legacy-bios または uefi) をどの AMI が使用しているか注意を払ってください。必要なブートモードがどの AMI を使用するのか注意してください。詳細については、ブートモード を参照してください。
5. ニーズを満たす AMI を選択し、[Select] を選択します。

Systems Manager パラメータで探す

1. [Search by Systems Manager parameter (Systems Manager パラメータで検索)] (右上) を選択します。
2. [Systems Manager parameter (Systems Manager パラメータ)] でパラメータを選択します。対応する AMI ID が [Currently resolves to (現在対応するもの)] の横に表示されます。
3. [検索] を選択します。AMI ID に一致する AMI がリストに表示されます。
4. リストから AMI を選択し、[Select (選択)] を選択します。

ステップ 2: インスタンスタイプを選択する

[Choose an Instance Type] ページで、起動するインスタンスのハードウェア設定とサイズを選択します。インスタンスタイプが大きくなると、CPU およびメモリも増えます。詳細については、インスタンスタイプ を参照してください。

無料利用枠を利用し続けるには、t2.micro インスタンスタイプを選択します (t2.micro が利用できないリージョンでは t3.micro インスタンスタイプを選択します)。インスタンスタイプが無料利用枠の下で適格である場合、それは [Free tier eligible] (無料利用枠適格) とラベル付けされます。t2.micro と t3.micro の詳細については、「バーストパフォーマンスインスタンス」を参照してください。

デフォルトでは、ウィザードには現行世代のインスタンスタイプが表示され、お客様が選択した AMI に基づいて使用可能な最初のインスタンスタイプが選択されます。旧世代のインスタンスタイプを表示するには、フィルタリストから [All generations] を選択します。

> **Note**
>
> テスト目的でインスタンスをすばやくセットアップする必要がある場合は、[Review and Launch] を選択し、デフォルトの設定を受け入れてインスタンスを起動します。それ以外の

場合は、インスタンスをさらに設定するために、[Next: Configure Instance Details] を選択します。

ステップ 3: インスタンスの詳細を設定する

[Configure Instance Details] ページで、必要に応じて次の設定を変更し (すべての設定を表示するには [Advanced Details] を展開)、[Next: Add Storage] を選択します。

- [Number of instances]: 起動するインスタンスの数を入力します。

> **Tip**
>
> インスタンスの起動を高速化するには、大きなリクエストをより小さなバッチに分割します。例えば、1 つの起動リクエストに 500 インスタンスが含まれている場合は、それを 5 つの起動リクエスト (各 100 インスタンス) に分割します。

- (オプション) アプリケーションで需要を処理するためにインスタンスの正しい数を確実に維持するには、[Launch into Auto Scaling Group (Auto Scaling グループに作成する)] を選択して起動設定と Auto Scaling グループを作成します。Auto Scaling によって、指定どおりにグループのインスタンス数がスケーリングされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」を参照してください。

> **Note**
>
> Amazon EC2 Auto Scaling が Auto Scaling グループ内のインスタンスを異常とマークすると、そのインスタンスの置き換えが自動的にスケジュールされます。この場合、インスタンスは終了されて別のインスタンスが起動され、元のインスタンスのデータは失われます。インスタンスを停止または再起動するか、別のイベントがインスタンスを異常としてマークすると、インスタンスは異常としてマークされます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インスタンスのヘルスチェック」を参照してください。

- [購入のオプション]: [スポットインスタンスのリクエスト] を選択してスポットインスタンスを起動します。このページからオプションの追加と削除を行います。オプションで最大料金を設定でき (非推奨)、オプションでリクエストタイプ、中断動作、およびリクエストの有効性を変更できます。詳細については、「スポットインスタンスリクエストを作成する」を参照してください。

- [Network] (ネットワーク): VPC を選択します。または新しい VPC を作成するには、[Create new VPC] (新しい VPC の作成) を選択して Amazon VPC コンソールに移動します。終了したらインスタンス起動ウィザードに戻り、[Refresh] (更新) を選択して一覧に VPC を読み込みます。
- [サブネット]: インスタンスは、アベイラビリティーゾーン、ローカルゾーン、Wavelength Zone、Outpost のいずれかに関連付けられたサブネットで起動できます。

  アベイラビリティーゾーンでインスタンスを起動するには、インスタンスを起動するサブネットを選択します。[指定なし] を選択して、AWS で任意のアベイラビリティーゾーンのデフォルトサブネットを自動的に選択できます。新しいサブネットを作成するには、[Create new subnet] を選択して Amazon VPC コンソールに移動します。終了したらウィザードに戻り、[Refresh] を選択して一覧にサブネットを読み込みます。

  ローカルゾーンでインスタンスを起動するには、ローカルゾーン内に作成したサブネットを選択します。

  アウトポストでインスタンスを起動するには、アウトポストに関連付けられた VPC 内のサブネットを選択します。
- [Auto-assign Public IP]: インスタンスがパブリック IPv4 アドレスを受け取るかどうかを指定します。デフォルトでは、デフォルトのサブネットにあるインスタンスはパブリック IPv4 アドレスを受け取りますが、デフォルト以外のサブネットにあるインスタンスは受け取りません。[Enable] または [Disable] を選択すると、これがサブネットのデフォルト設定より優先されます。詳細については、パブリック IPv4 アドレス を参照してください。
- [Auto-assign IPv6 IP]: インスタンスがサブネットの範囲から IPv6 アドレスを受け取るかどうかを指定します。[Enable] または [Disable] を選択すると、これによりサブネットのデフォルト設定がオーバーライドされます。このオプションは IPv6 CIDR ブロックを VPC とサブネットに関連付けた場合にのみ使用できます。詳細については、「Amazon VPC ユーザーガイド」の「IPv6 CIDR ブロックの VPC への追加」を参照してください。
- [Hostname type] (ホスト名タイプ): インスタンスのゲスト OS ホスト名をリソース名または IP 名に含めるかどうかを選択します。詳細については、Amazon EC2 インスタンスのホスト名タイプ を参照してください。
- [DNS Hostname] (DNS ホスト名): リソース名または IP 名への DNS クエリが、([Hostname type] (ホスト名タイプ) に何を選択したのかによって) IPv4 アドレス (A レコード)、IPv6 アドレス (AAAA レコード)、またはその両方で応答するかどうかを決定します。詳細については、Amazon EC2 インスタンスのホスト名タイプ を参照してください。
- [ドメイン結合ディレクトリ]: 起動後の Linux インスタンスを結合する先の AWS Directory Service ディレクトリ (ドメイン) を選択します。ドメインを選択する場合は、必要なアクセス許可を持

つ IAM ロールを選択する必要があります。詳細については、「Linux EC2 インスタンスを AWS Managed Microsoft AD ディレクトリにシームレスに結合する」を参照してください。

- [プレイスメントグループ]: プレイスメントグループは、インスタンスの配置戦略を決定します。既存のプレイスメントグループを選択するか、新しいグループを作成します。このオプションは、プレイスメントグループをサポートするインスタンスタイプを選択した場合にのみ使用できます。詳細については、プレイスメントグループ を参照してください。
- キャパシティーの予約: インスタンスを共有キャパシティー、任意の open キャパシティーの予約、特定の キャパシティーの予約、または キャパシティーの予約 グループのどれに起動するかを指定します。詳細については、既存の キャパシティーの予約 へのインスタンスの起動 を参照してください。
- [IAM ロール]: インスタンスに関連付ける AWS Identity and Access Management (IAM) ロールを選択します。詳細については、Amazon EC2 の IAM ロール を参照してください。
- CPU オプション: 起動中に [CPU オプションを指定] を選択して、カスタム数の vCPU を指定します。CPU コアの数とコアごとのスレッド数を設定します。詳細については、CPU オプションの最適化 を参照してください。
- [Shutdown behavior]: シャットダウン時にインスタンスを停止するか終了するかを選択します。詳細については、インスタンスによって起動されたシャットダウン動作の変更 を参照してください。
- [Stop - Hibernate behavior]: 休止を有効にするには、このチェックボックスをオンにします。このオプションは、インスタンスが休止の前提条件を満たしている場合にのみ使用できます。詳細については、Amazon EC2 インスタンスの休止 を参照してください。
- [Enable termination protection]: 偶発的な終了を防ぐには、このチェックボックスをオンにします。詳細については、「終了保護を有効化する」を参照してください。
- 停止保護の有効化: 偶発的な停止を防ぐには、このチェックボックスをオンにします。詳細については、「停止保護を有効にします」を参照してください。
- [Monitoring] (モニタリング): Amazon CloudWatch を使用したインスタンスの詳細モニタリングを有効にするには、このチェックボックスをオンにします。別途 料金がかかります。詳細については、CloudWatch を使用したインスタンスのモニタリング を参照してください。
- [EBS 最適化インスタンス]: Amazon EBS 最適化インスタンスは、最適化された設定スタックを使用し、Amazon EBS I/O に対して追加の専用容量を提供します。ご使用のインスタンスタイプでこの機能がサポートされている場合は、このチェックボックスをオンにして有効化します。追加の変更が適用されます。詳細については、Amazon EBS 最適化インスタンスを使用する を参照してください。

- [Tenancy]: VPC でインスタンスを起動する場合、独立した専用のハードウェア ([Dedicated]) または Dedicated Host ([Dedicated host]) を選択できます。追加料金が適用される場合があります。詳細については、Dedicated InstancesおよびDedicated Hostsを参照してください。
- [T2/T3 Unlimited]: このチェックボックスをオンにすると、アプリケーションがベースラインを越えて必要なだけバーストできるようになります。追加料金が適用される場合があります。詳細については、バーストパフォーマンスインスタンス を参照してください。
- ファイルシステム: インスタンスにマウントする新しいファイルシステムを作成するには、[新しいファイルシステムの作成] を選択し、新しいファイルシステムの名前を入力して [作成] をクリックします。ファイルシステムは、サービスの推奨設定を適用する Amazon EFS Quick Create を使用して作成されます。ファイルシステムへのアクセスを有効にするために必要なセキュリティグループは自動的に作成され、ファイルシステムのインスタンスおよびマウントターゲットにアタッチされます。また、必要なセキュリティグループを手動で作成してアタッチすることもできます。既存の Amazon EFS ファイルシステムをインスタンスにマウントするには、[ファイルシステムの追加] を選択し、マウントするファイルシステムと使用するマウントポイントを選択します。詳細については、「Amazon EC2 での Amazon EFS の使用」を参照してください。
- [Network interfaces]: 特定のサブネットを選択すると、インスタンスに対して最大 2 つのネットワークインターフェイスを指定できます。
  - [Network Interface] で、[New network interface] を選択して AWS によって新しいインターフェイスを作成するか、既存の使用できるネットワークインターフェイスを選択します。
  - [Primary IP] で、サブネットの範囲からプライベート IPv4 アドレスを入力するか、[Auto-assign] をデフォルトのままにしてプライベート IPv4 アドレスが AWS によって自動的に選択されるようにします。
  - 選択したネットワークインターフェイスに対して複数のプライベート IPv4 アドレスを割り当てるには、[Secondary IP addresses] で [Add IP] を選択します。
  - (IPv6 のみ) [IPv6 IPs] で [Add IP] (IP の追加) をクリックした後、サブネット範囲内の IPv6 アドレスを入力します。あるいは、[Auto-assign] (自動的に割り当て) をそのまま受け入れて、IPv6 アドレスが AWS によって自動的に選択されるようにします。
  - ネットワークカードインデックス: ネットワークカードのインデックス。プライマリネットワークインターフェイスは、ネットワークカードインデックス 0 に割り当てる必要があります。インスタンスタイプによっては、複数のネットワークカードがサポートされているものもあります。
  - [Add Device] を選択して、セカンダリネットワークインターフェイスを追加します。セカンダリネットワークインターフェイスは、インスタンスと同じアベイラビリティーゾーンにある場合は、VPC の別のサブネットに存在できます。

詳細については、Elastic Network Interface を参照してください。複数のネットワークインターフェイスを指定した場合、インスタンスはパブリック IPv4 アドレスを受け取ることはできません。さらに、eth0 に既存のネットワークインターフェイスを指定した場合、[Auto-assign Public IP] を使用してサブネットのパブリック IPv4 設定をオーバーライドする操作は禁止されます。詳細については、インスタンス起動時のパブリック IPv4 アドレスの割り当て を参照してください。

- [カーネル ID]: (準仮想化 (PV) AMIs でのみ有効) 特定のカーネルを使用する場合を除き、[デフォルトを使用] を選択します。
- [RAM ディスク ID]: (準仮想化 (PV) AMIs でのみ有効) 特定の RAM ディスクを使用する場合を除き、[デフォルトを使用] を選択します。カーネルを選択した場合は、サポートするドライバーとともに特定の RAM ディスクを選択しなければならない可能性があります。
- エンクレーブ: AWS Nitro Enclaves のインスタンスを有効にするには、[有効] を選択します。詳細については、AWS Nitro Enclaves ユーザーガイドの「AWS Nitro Enclaves とは」を参照してください。
- [アクセス可能なメタデータ]: インスタンスメタデータサービス (IMDS) へのアクセスを有効または無効にできます。詳細については、「IMDSv2 の使用」を参照してください。
- [Metadata transport] (メタデータの転送): インスタンスがリンクローカル IMDSv2 IPv6 アドレス (`fd00:ec2::254`) に到達できるようにして、インスタンスのメタデータを取得します。このオプションは、IPv6 専用サブネットで Nitro System 上に構築されたインスタンス を起動している場合にのみ使用できます。インスタンスメタデータの取得の詳細については、「インスタンスメタデータの取得」を参照してください。
- [メタデータのバージョン]: インスタンスメタデータへのアクセスを有効にする場合、IMDS をリクエストするときにインスタンスメタデータサービスバージョン 2 の使用を必須にすることができます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [メタデータトークンの応答ホップ制限]: IMDS を有効にする場合、メタデータトークンに許容されるネットワークホップ数を設定できます。詳細については、「IMDSv2 の使用」を参照してください。
- [ユーザーデータ]: 起動時にインスタンスを設定するユーザーデータ、または設定スクリプトを実行するユーザーデータを指定できます。ファイルをアタッチするには、[As file] オプションを選択し、アタッチするファイルを参照します。

ステップ 4: ストレージを追加する

選択した AMI には、ルートデバイスボリュームを含む、1 つまたは複数のストレージボリュームが含まれます。[Add Storage] ページで、[Add New Volume] を選択することにより、インスタンスにアタッチする追加ボリュームを指定できます。各ボリュームを次のように設定し、[Next:Add Tags (次へ: タグの追加)] を選択します。

- [Type (タイプ)]: インスタンスと関連付けるインスタンスストアまたは Amazon EBS ボリュームを選択します。一覧で利用できるボリュームの種類は、選択したインスタンスタイプに応じて異なります。詳細については、「Amazon EC2 インスタンスストア」および「Amazon EBS ボリューム」を参照してください。
- [Device [デバイス]]: ボリュームで利用できるデバイス名の一覧から選択します。
- [Snapshot (スナップショット)]: ボリュームを復元するスナップショットの名前または ID を入力します。[Snapshot (スナップショット)] フィールドにテキストを入力して、利用できる共有スナップショットとパブリックスナップショットを検索することもできます。スナップショットの説明では大文字と小文字が区別されます。
- [Size (サイズ)]: EBS ボリュームの場合、ストレージサイズを指定できます。無料利用枠の対象となる AMI とインスタンスを選択した場合でも、無料利用枠内に収めるには、合計ストレージを 30 GiB 以下に維持する必要があります。
- [Volume Type (ボリュームタイプ)]: EBS ボリュームの場合、ボリュームタイプを選択します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。
- [IOPS]: Provisioned IOPS SSD ボリュームタイプを選択した場合は、ボリュームがサポートできる I/O オペレーション/秒 (IOPS) を入力できます。
- [Delete on Termination (終了時に削除)]: Amazon EBS ボリュームについては、インスタンスが終了したときにボリュームを削除するには、このチェックボックスをオンにします。詳細については、インスタンスの終了時にデータを保持する を参照してください。
- [Encrypted (暗号化)]: インスタンスタイプが EBS 暗号化をサポートしている場合、ボリュームの暗号化状態を指定できます。このリージョンでデフォルトの暗号化を有効にした場合は、自動的にデフォルトのカスタマーマネージド型キーが選択されます。別のキーを選択するか、暗号化を無効にすることができます。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

ステップ 5: タグの追加

[Add Tags] ページで、キーと値の組み合わせをタグとして指定します。インスタンス、ボリューム、またはその両方にタグ付けできます。スポットインスタンスの場合、スポットインスタンスリクエストにのみタグを付けることができます。リソースに複数のタグを追加するには、[Add another tag] を選択します。完了したら、[次の手順: セキュリティグループの設定] を選択します。

ステップ 6: セキュリティグループを設定する

[Configure Security Group] ページで、セキュリティグループを使用してインスタンスのファイアウォールルールを定義します このルールでは、どの着信ネットワークトラフィックをインスタンスに配信するかを指定します。他のトラフィックはすべて無視されます。(セキュリティグループの詳細については、「Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ」を参照してください)。以下のようにセキュリティグループを選択または作成して、[Review and Launch] を選択します。

- 既存のセキュリティグループを選択するには、[Select an existing security group (既存のセキュリティグループの選択)] を選択してから、セキュリティグループを選択します 既存のセキュリティグループのルールを編集することはできません。しかし、[Copy to new (コピーして新規作成)] を選択して、新しいグループにルールをコピーすることはできます。その後、次の手順で説明しているように、ルールを追加できます。
- 新しいセキュリティグループを作成するには、[Create a new security group (新しいセキュリティグループの作成)] を選択します。このウィザードでは、launch-wizard-x セキュリティグループが自動的に定義され、SSH (ポート 22) を介したインスタンスへの接続を許可するインバウンドルールが作成されます。
- ニーズに応じたルールを追加できます。例えば、インスタンスがウェブサーバーである場合は、ポート 80 (HTTP) とポート 443 (HTTPS) を開いて、インターネットトラフィックを許可します。

  ルールを追加するには、[Add Rule] を選択し、プロトコルを選択してネットワークトラフィックを開いてから、ソースを指定します。[Source] (送信元) リストから [My IP] (マイ IP) を選択し、ウィザードでコンピュータのパブリック IP アドレスを追加します。ただし、ISP 経由で、またはファイアウォールの内側から静的な IP アドレスなしで接続している場合は、クライアントコンピュータで使用されている IP アドレスの範囲を見つける必要があります。

  > ⚠ Warning
  >
  > すべての IP アドレス (`0.0.0.0/0`) に SSH または RDP を介したインスタンスへのアクセスを許可するルールは、この短期間の実習では許容されますが、本番稼働用環境では安全

ではありません。特定の IP アドレスまたは特定のアドレス範囲にのみ、インスタンスへのアクセスを限定してください。

ステップ 7: インスタンスの起動を確認し、キーペアを選択する

[Review Instance Launch] ページで、インスタンスの詳細をチェックし、適切な [Edit] リンクを選択して必要な変更を加えます。

準備ができたら、[Launch] を選択します。

[Select an existing key pair or create a new key pair] ダイアログボックスで、既存のキーペアを選択するか、新しいキーペアを作成できます。例えば、[Choose an existing key pair] を選択し、セットアップ中に作成したキーペアを選択します。詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンス を参照してください。

> **Important**
>
> [Proceed without key pair] オプションを選択した場合、ユーザーが別の方法でログインすることを許可するように設定された AMI を選択した場合でなければ、インスタンスに接続できなくなります。

インスタンスを起動するには、確認のチェックボックスをオンにし、続いて [Launch Instances] を選択します。

(オプション) インスタンスのステータスチェックアラームを作成することもできます (追加料金がかかります)。確認画面で、[Create status check alarms] を選択して、指示にしたがいます。ステータス確認アラームは、インスタンスの起動後に作成することもできます。詳細については、ステータスチェックアラームの作成と編集 を参照してください。

インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

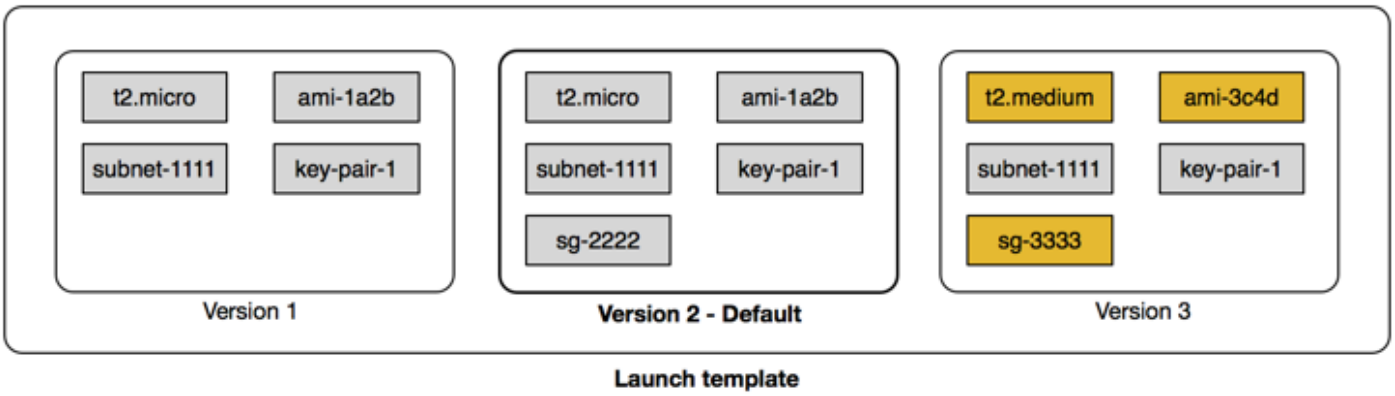
起動テンプレートからのインスタンスの起動

起動テンプレートを使用してインスタンス起動パラメータを保存すると、インスタンスを起動するたびにパラメータを指定する必要がなくなります。例えば、AMI ID やインスタンスタイプ、通常インスタンスの起動に使用しているネットワーク設定を使って、起動テンプレートを作成することができ

ます。Amazon EC2 コンソール、AWS SDK、コマンドラインツールのいずれかを使用してインスタンスを起動するときは、パラメータを再度入力する代わりに、起動テンプレートを指定できます。

各起動テンプレートについて、1 つ以上の番号付きの起動テンプレートのバージョンを作成できます。各バージョンに異なる起動パラメータを指定できます。起動テンプレートからインスタンスを起動する際、起動テンプレートのいずれかのバージョンを使用できます。バージョンを指定しない場合は、デフォルトバージョンが使用されます。いずれかの起動テンプレートをデフォルトバージョンとして設定できます — デフォルトでは、起動テンプレートの最初のバージョンです。

以下の図は、3 つのバージョンの起動テンプレートを示しています。最初のバージョンでは、インスタンスの起動に使用するインスタンスタイプ、AMI ID、サブネット、およびキーペアが指定されています。2 番目のバージョンは最初のバージョンに基づいており、インスタンスのセキュリティグループも指定しています。3 番目のバージョンは、パラメータの一部に異なる値を使用しています。バージョン 2 がデフォルトバージョンとして設定されています。この起動テンプレートからインスタンスを起動すると、他のバージョンを指定しない限りバージョン 2 の起動パラメータが使用されます。



コンテンツ

- 起動テンプレートの制限
- IAM アクセス許可を使用して起動テンプレートへのアクセスを制御する
- インスタンスの起動を制御する起動テンプレートを使用する
- 起動テンプレートの作成
- 起動テンプレートの変更 (起動テンプレートのバージョンの管理)
- 起動テンプレートの削除
- 起動テンプレートからのインスタンスの起動

起動テンプレートの制限

起動テンプレートおよび起動テンプレートのバージョンには次のルールが適用されます。

- クォータ – 起動テンプレートのクォータと起動テンプレートバージョンのクォータを確認するには、[Service Quotas] コンソールを開くか、AWS CLI コマンドの list-service-quotas を使用します。各 AWS アカウントでは、1 つのリージョンあたり最大で 5,000 の起動テンプレート、1 つの起動テンプレートあたり最大で 10,000 のバージョンを起動します。アカウントには、作成してからの期間や使用履歴に基づいて異なるクォータが設定されている場合があります。
- パラメータはオプション - 起動テンプレートのパラメータはオプションです。ただし、テンプレートには、インスタンス起動のためのリクエストに必要な、すべてのパラメータが含まれている必要があります。例えば、起動テンプレートに AMI ID が含まれていない場合、インスタンスの起動時に起動テンプレートと AMI ID の両方を指定する必要があります。
- パラメータは未検証 – 起動テンプレートパラメータは、起動テンプレート作成の際には完全には検証されていません。パラメータに誤った値を指定した場合、またはサポートされているパラメータの組み合わせを使用しない場合、この起動テンプレートを使用してインスタンスは起動できません。パラメータに正しい値を指定したか、およびサポートされているパラメータの組み合わせを使用しているかを確認します。例えば、プレイスメントグループ内でインスタンスを起動するには、サポートされているインスタンスタイプを指定する必要があります。
- タグ – 起動テンプレートにはタグ付けできますが、起動テンプレートのバージョンにはタグ付けできません。
- 変更不可能 – 起動テンプレートは変更不可能です。起動テンプレートを変更するには、起動テンプレートの新しいバージョンを作成する必要があります。
- バージョン番号 – 起動テンプレートのバージョンには、作成された順序で番号が付けられます。起動テンプレートのバージョンを作成する場合、自分でバージョン番号を指定することはできません。

IAM アクセス許可を使用して起動テンプレートへのアクセスを制御する

IAM アクセス許可を使用して、起動テンプレートの表示、作成、削除など、ユーザーが実行できる起動テンプレートのアクションを制御できます。

> **Note**
>
> 起動テンプレートを使用するユーザーには、起動テンプレートで指定されたリソースを使用および作成するためのアクセス許可が必要です。例:

- 起動テンプレートで指定された、共有されたプライベートの Amazon マシンイメージ (AMI) からインスタンスを起動するには、ユーザーはその AMI を起動するアクセス許可を持っている必要があります。
- 起動テンプレートで定義されている EBS ボリューム、タグ、その他リソースを作成するには、ユーザーはそれらのリソースを作成するためのアクセス許可を持っている必要があります。

起動テンプレートを作成および管理するためのアクセス許可は、次のように機能します。

コンテンツ

- ec2:CreateLaunchTemplate
- ec2:DescribeLaunchTemplates
- ec2:DescribeLaunchTemplateVersions
- ec2:DeleteLaunchTemplate
- バージョニングアクセス許可の制御
- 起動テンプレートのタグへのアクセスを制御する

ec2:CreateLaunchTemplate

コンソールで、または API を使用して起動テンプレートを作成するには、プリンシパルが IAM ポリシーで `ec2:CreateLaunchTemplate` アクセス許可を持っている必要があります。可能な限り、タグを使用してアカウントで起動テンプレートへのアクセスを制御できるようにします。

例えば、次の IAM ポリシーステートメントは、テンプレートが指定されたタグ (*purpose*=*testing*) を使用している場合にのみ、プリンシパルに起動テンプレートを作成する許可を付与します。

```
{
    "Sid": "IAMPolicyForCreatingTaggedLaunchTemplates",
    "Action": "ec2:CreateLaunchTemplate",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/purpose": "testing"
```

```
        }
    }
}
```

起動テンプレートを作成するプリンシパルには、次のような関連するアクセス許可が必要な場合があります。

- ec2:CreateTags — CreateLaunchTemplate 操作時に起動テンプレートにタグを追加するには、CreateLaunchTemplate の呼び出し元が IAM ポリシーで ec2:CreateTags アクセス許可を持っている必要があります。
- ec2:RunInstances – 作成した起動テンプレートから EC2 インスタンスを起動するには、プリンシパルは IAM ポリシーで ec2:RunInstances アクセス許可も持っている必要があります。

タグを適用するリソース作成アクションでは、ユーザーが ec2:CreateTags アクセス許可を持っている必要があります。次の IAM ポリシーステートメントは、ec2:CreateAction 条件キーを使用して、ユーザーが CreateLaunchTemplate のコンテキストでのみタグを使用できるようにしています。ユーザーは、既存の起動テンプレートにも他のリソースにもタグ付けできません。詳細については、「リソース作成時にタグ付けするアクセス許可の付与」を参照してください。

```
{
    "Sid": "IAMPolicyForTaggingLaunchTemplatesOnCreation",
    "Action": "ec2:CreateTags",
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:region:account-id:launch-template/*",
    "Condition": {
        "StringEquals": {
            "ec2:CreateAction": "CreateLaunchTemplate"
        }
    }
}
```

起動テンプレートを作成した IAM ユーザーに、作成した起動テンプレートを使用するアクセス許可が自動で付与されることはありません。他のプリンシパルと同様に、起動テンプレートの作成者は、IAM ポリシーを使用してアクセス許可を取得する必要があります。IAM ユーザーが起動テンプレートから EC2 インスタンスを起動する場合は、ec2:RunInstances アクセス許可が必要です。このアクセス許可を付与するときに、ユーザーが特定のタグまたは特定の ID を含む起動テンプレートのみを使用できるように指定できます。また、RunInstances 呼び出しに対するリソースレベルのアクセス許可を指定することで、起動テンプレートを使用するすべてのユーザーがインスタンスの

起動時に参照および使用できる AMI やその他のリソースを制御できます。エンドポイントポリシーの例については、「起動テンプレート」を参照してください。

ec2:DescribeLaunchTemplates

アカウントの起動テンプレートを一覧表示するには、プリンシパルが IAM ポリシーで ec2:DescribeLaunchTemplates アクセス許可を持っている必要があります。Describe アクションはリソースレベルのアクセス許可をサポートしていないため、条件なしで指定する必要があります。また、ポリシーのリソース要素の値は "*" である必要があります。

例えば、次の IAM ポリシーステートメントでは、アカウントのすべての起動テンプレートを一覧表示する許可をプリンシパルに付与します。

```
{
    "Sid": "IAMPolicyForDescribingLaunchTemplates",
    "Action": "ec2:DescribeLaunchTemplates",
    "Effect": "Allow",
    "Resource": "*"
}
```

ec2:DescribeLaunchTemplateVersions

起動テンプレートを表示するプリンシパルは、起動テンプレートを構成する属性セット全体を取得するための ec2:DescribeLaunchTemplateVersions アクセス許可も持つようにしてください。

アカウントの起動テンプレートのバージョンを一覧表示するには、プリンシパルが IAM ポリシーで ec2:DescribeLaunchTemplateVersions アクセス許可を持っている必要があります。Describe アクションはリソースレベルのアクセス許可をサポートしていないため、条件なしで指定する必要があります。また、ポリシーのリソース要素の値は "*" である必要があります。

例えば、次の IAM ポリシーステートメントでは、アカウントにおけるすべての起動テンプレートのバージョンを一覧表示する許可をプリンシパルに付与します。

```
{
    "Sid": "IAMPolicyForDescribingLaunchTemplateVersions",
    "Effect": "Allow",
    "Action": "ec2:DescribeLaunchTemplateVersions",
    "Resource": "*"
}
```

ec2:DeleteLaunchTemplate

> **Important**
>
> プリンシパルにリソースを削除する許可を与えるときは、注意する必要があります。起動テンプレートを削除すると、起動テンプレートに依存する AWS リソースに障害が発生する可能性があります。

起動テンプレートを削除するには、プリンシパルが IAM ポリシーで ec2:DeleteLaunchTemplate アクセス許可を持っている必要があります。可能な限り、タグベースの条件キーを使用してアクセス許可を制限します。

例えば、次の IAM ポリシーステートメントは、テンプレートが指定されたタグ (*purpose*=*testing*) を使用している場合にのみ、プリンシパルに起動テンプレートを削除する許可を付与します。

```
{
    "Sid": "IAMPolicyForDeletingLaunchTemplates",
    "Action": "ec2:DeleteLaunchTemplate",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/purpose": "testing"
        }
    }
}
```

または、ARN を使用して IAM ポリシーが適用される起動テンプレートを指定することもできます。

起動テンプレートには次の ARN が含まれます。

```
"Resource": "arn:aws:ec2:region:account-id:launch-template/lt-09477bcd97b0d310e"
```

複数の ARN を一覧で囲んで指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントの任意の起動テンプレートを削除できるようにすることもできます。

バージョニングアクセス許可の制御

信頼できる管理者には、以下の例のように IAM ポリシーを使用して、起動テンプレートのバージョンを作成および削除したり、起動テンプレートのデフォルトバージョンを変更したりするためのアクセス許可を付与できます。

Important

起動テンプレートのバージョンを作成したり、起動テンプレートを変更したりするアクセス許可をプリンシパルに付与する場合は、注意が必要です。

- 起動テンプレートのバージョンを作成すると、Amazon EC2 がユーザーに代わって `Latest` バージョンでインスタンスを起動できるようにするすべての AWS リソースに影響します。
- 起動テンプレートを変更すると、どのバージョンが `Default` になるかを変更できます。したがって、Amazon EC2 がこの変更済みバージョンを使用してユーザーに代わってインスタンスを起動できるようにするすべての AWS リソースに影響します。

また、EC2 フリートやスポットフリートなど、`Latest` または `Default` の起動テンプレートバージョンとやり取りする AWS リソースの取り扱い方にも注意が必要です。「`Latest`」または「`Default`」に別の起動テンプレートバージョンが使用されている場合、Amazon EC2 は、フリートの目標容量を満たすために新しいインスタンスを起動する際に、完了すべきアクションのユーザーアクセス許可を再確認することはありません。これは、AWS リソースとユーザーのやり取りがないためです。`CreateLaunchTemplateVersion` API と `ModifyLaunchTemplate` API を呼び出すアクセス許可をユーザーに付与すると、インスタンスプロファイル (IAM ロールのコンテナ) を含む別の起動テンプレートバージョンをフリートに指定する場合、ユーザーに `iam:PassRole` アクセス許可も効果的に付与できます。つまり、場合によっては `iam:PassRole` アクセス許可がなくても、起動テンプレートを更新して IAM ロールをインスタンスに渡すことができます。このリスクは、起動テンプレートバージョンを作成および管理できるユーザーにアクセス許可を付与する際に注意することで管理できます。

ec2:CreateLaunchTemplateVersion

起動テンプレートの新しいバージョンを作成するには、プリンシパルが IAM ポリシーで起動テンプレートに対する `ec2:CreateLaunchTemplateVersion` アクセス許可を持っている必要があります。

例えば、次の IAM ポリシーステートメントは、バージョンが指定されたタグ (*environment*=*production*) を使用している場合にのみ、プリンシパルに起動テンプレートのバージョンを作成する許可を付与します。あるいは、1 つまたは複数の起動テンプレートの ARN を指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントにおける任意の起動テンプレートのバージョンを作成できるようにすることもできます。

```
{
    "Sid": "IAMPolicyForCreatingLaunchTemplateVersions",
    "Action": "ec2:CreateLaunchTemplateVersion",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/environment": "production"
        }
    }
}
```

ec2:DeleteLaunchTemplateVersion

> **Important**
>
> プリンシパルにリソースを削除する権限を与えるときは、いつものように注意する必要があります。起動テンプレートのバージョンを削除すると、起動テンプレートのバージョンに依存する AWS リソースに障害が発生する可能性があります。

起動テンプレートのバージョンを削除するには、プリンシパルが IAM ポリシーで起動テンプレートに対する ec2:DeleteLaunchTemplateVersion アクセス許可を持っている必要があります。

例えば、次の IAM ポリシーステートメントは、バージョンが指定されたタグ (*environment*=*production*) を使用している場合にのみ、プリンシパルに起動テンプレートのバージョンを削除する許可を付与します。あるいは、1 つまたは複数の起動テンプレートの ARN を指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントにおける任意の起動テンプレートのバージョンを削除できるようにすることもできます。

```
{
    "Sid": "IAMPolicyForDeletingLaunchTemplateVersions",
    "Action": "ec2:DeleteLaunchTemplateVersion",
    "Effect": "Allow",
```

```
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/environment": "production"
        }
    }
}
```

ec2:ModifyLaunchTemplate

起動テンプレートに関連付けられている Default バージョンを変更するには、プリンシパルが IAM ポリシーで起動テンプレートに対する ec2:ModifyLaunchTemplate アクセス許可を持っている必要があります。

例えば、次の IAM ポリシーステートメントは、起動テンプレートが指定されたタグ (*environment*=*production*) を使用している場合にのみ、プリンシパルに起動テンプレートを変更する許可を付与します。あるいは、1 つまたは複数の起動テンプレートの ARN を指定することも、Condition 要素を使用せず Resource 値に "*" を指定して、プリンシパルがアカウントにおける任意の起動テンプレートのバージョンを変更できるようにすることもできます。

```
{
    "Sid": "IAMPolicyForModifyingLaunchTemplates",
    "Action": "ec2:ModifyLaunchTemplate",
    "Effect": "Allow",
    "Resource": "*",
    "Condition": {
        "StringEquals": {
            "aws:ResourceTag/environment": "production"
        }
    }
}
```

起動テンプレートのタグへのアクセスを制御する

リソースが起動テンプレートの場合、条件キーを使用してタグ付け許可を制限できます。例えば、次の IAM ポリシーでは、指定されたアカウントとリージョンの起動テンプレートから *temporary* キーを持つタグのみを削除できます。

```
{
    "Sid": "IAMPolicyForDeletingTagsOnLaunchTemplates",
    "Action": "ec2:DeleteTags",
```

```
    "Effect": "Allow",
    "Resource": "arn:aws:ec2:region:account-id:launch-template/*",
    "Condition": {
        "ForAllValues:StringEquals": {
            "aws:TagKeys": ["temporary"]
        }
    }
}
```

Amazon EC2 リソースに適用できるタグキーとタグ値を制御するのに使用できる条件キーの詳細については、「特定のタグへのアクセスの制御」を参照してください。

インスタンスの起動を制御する起動テンプレートを使用する

ユーザーが起動テンプレートを使用する場合、インスタンスのみを起動できるようにして、さらに特定の起動テンプレートのみを使用できるように指定することができます。また、起動テンプレートや起動テンプレートのバージョンを作成、変更、記述、削除できる人を制御することもできます。

起動テンプレートを使用した起動パラメータの制御

起動テンプレートには、インスタンスを起動するためのパラメータすべてまたは一部を含めることができます。起動テンプレートを使用してインスタンスを起動するときは、起動テンプレートで指定されたパラメータを上書きできます。または、起動テンプレートにない追加のパラメータを指定できます。

> **Note**
>
> 起動時に起動テンプレートパラメータを削除することはできません (例えば、パラメータに null 値を指定することはできません)。パラメータを削除するには、起動テンプレートの新しいバージョンをパラメータなしで作成し、そのバージョンを使用してインスタンスを起動します。

インスタンスを起動するには、ユーザーは ec2:RunInstances アクションを使用するための許可が必要です。また、ユーザーは、インスタンスに作成または関連付けられたリソースを作成または使用する許可が必要です。ec2:RunInstances アクションのリソースレベルのアクセス権限を使用して、ユーザーが指定できる起動パラメータを管理できます。または、起動テンプレートを使用してインスタンスを起動するアクセス権限をユーザーに付与することもできます。これにより、IAM ポリシーではなくむしろ起動テンプレートで起動パラメータを管理できるようになり、インスタンス起動の権限を付与するための手段として起動テンプレートを使用できます。例えば、ユーザーが起動テン

プレートを使用してのみインスタンスを起動できるようにして、さらに特定の起動テンプレートのみを使用できるように指定することができます。また、ユーザーが起動テンプレートで上書きする起動パラメータを制御することもできます。エンドポイントポリシーの例については、「起動テンプレート」を参照してください。

起動テンプレートの使用の管理

デフォルトでは、 ユーザーには起動テンプレートを使用するためのアクセス許可がありません。起動テンプレートと起動テンプレートバージョンの作成、変更、記述、および削除のアクセス許可をユーザーに付与するポリシーを作成できます。一部の起動テンプレートアクションにリソースレベルのアクセス許可を適用して、ユーザーがこれらのアクションに対する特定のリソースを使用する機能を制御することもできます。詳細については、「例: 起動テンプレートの使用」のポリシー例を参照してください。

ユーザーに ec2:CreateLaunchTemplate および ec2:CreateLaunchTemplateVersion のアクションを使用するアクセス許可を付与するには注意が必要です。リソースレベルのアクセス許可を使用して、ユーザーが起動テンプレートで指定できるリソースを制御することはできません。インスタンス起動のために使用されるリソースを制限するには、起動テンプレートと起動テンプレートのバージョンを作成するアクセス許可を、適切な管理者のみに付与していることを確認してください。

EC2 フリートまたはスポットフリートで起動テンプレートを使用する際の重要なセキュリティ上の懸念

起動テンプレートを使用するには、起動テンプレートおよび起動テンプレートのバージョンを作成、変更、記述、および削除するアクセス許可をユーザーに付与する必要があります。ec2:CreateLaunchTemplate および ec2:CreateLaunchTemplateVersion アクションへのアクセスを制御することで、起動テンプレートと起動テンプレートのバージョンをどのユーザーが作成できるかを制御できます。ec2:ModifyLaunchTemplate アクションへのアクセスを制御することで、起動テンプレートを変更できるユーザーを管理することもできます。

> **Important**
>
> EC2 フリートまたはスポットフリートが「最新」または「デフォルト」の起動テンプレートバージョンを使用するように設定されている場合、後でそれらが変更されて起動テンプレートの別のバージョンを指すようになっても、そのフリートでは認識されません。「最新」または「デフォルト」に別の起動テンプレートバージョンが使用されている場合、フリートの目標容量を満たすために新しいインスタンスを起動する際に、Amazon EC2 により完了すべきアクションのアクセス許可が再確認されることはありません。これは、起動テンプレートバージョン (特にユーザーがデフォルトの起動テンプレートバージョンを変更でき

る ec2:ModifyLaunchTemplate アクション) を作成および管理できるユーザーにアクセス許可を付与する際の重要な考慮事項です。

起動テンプレート API の EC2 アクションを使用するアクセス許可をユーザーに付与すると、インスタンスプロファイル (IAM ロールのコンテナ) を含む別の起動テンプレートバージョンを指すように EC2 フリートまたはスポットフリートを作成および更新する場合でも、ユーザーに iam:PassRole アクセス許可が効果的に付与されます。つまり、場合によっては iam:PassRole アクセス許可がなくても、起動テンプレートを更新して IAM ロールをインスタンスに渡すことができます。詳細および IAM ポリシーの例については、「IAM ユーザーガイド」の「Amazon EC2 インスタンスで実行されるアプリケーションに IAM ロールを使用してアクセス許可を付与する」を参照してください。

詳細については、起動テンプレートの使用の管理および例: 起動テンプレートの使用を参照してください。

起動テンプレートの作成

ユーザー定義のパラメータを使用して新しい起動テンプレートを作成するか、既存の起動テンプレートまたはインスタンスをベースにして新しい起動テンプレートを作成します。

タスク

- 定義したパラメータを使用した新しい起動テンプレートの作成
- 既存の起動テンプレートからの起動テンプレートの作成
- インスタンスからの起動テンプレートの作成
- AMI ID のかわりに Systems Manager パラメータを使用する

定義したパラメータを使用した新しい起動テンプレートの作成

起動テンプレートは、コンソールあるいは AWS CLI を使用して作成できます。

- コンソール
- AWS CLI

コンソール

起動テンプレートを作成する際には、そのテンプレートの名と、少なくとも 1 つのインスタンス設定パラメーターを指定する必要があります。

起動テンプレート内では、そのテンプレートのパラメーターがグループ化されています。次の手順では、各パラメータグループについて説明します。

起動テンプレート設定のパラメータ

- 起動テンプレートの作成を開始する
- 起動テンプレートの名前、説明およびタグ
- アプリケーションと OS イメージ (Amazon マシンイメージ)
- インスタンスタイプ
- キーペア (ログイン)
- ネットワーク設定
- ストレージの設定
- リソースタグ
- 高度な詳細
- [概要]

起動テンプレートの作成を開始する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択します。

起動テンプレートの名前、説明およびタグ

1. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力します。
2. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンの短い説明を入力します。
3. 作成時に起動テンプレートにタグを付けるには、[Template tags] (テンプレートタグ) を展開し、[Add tag] (タグの追加) を選択して、タグキーと値のペアを入力します。追加するタグごとに [Add tag] (タグを追加) を選択します。

> **Note**
>
> インスタンスの起動時に作成されるリソースにタグを付けるには、[Resource tags] (リソースタグ) のタグを指定する必要があります。詳細については、「リソースタグ」を参照してください。

アプリケーションと OS イメージ (Amazon マシンイメージ)

Amazon マシンイメージ (AMI) には、インスタンスの作成に必要な情報が含まれています。例えば、ある AMI には、ウェブサーバーとして動作するために必要なソフトウェア (Linux、Apache、ウェブサイトなど) が含まれています。

適切な AMI は、次の手順で確認できます。AMI を検索する各オプションで、右上にある [Cancel] (キャンセル) をクリックすると、AMI を選択せずに起動テンプレートに戻ることができます。

検索バー

利用可能なすべての AMI を検索するには、AMI 検索バーにキーワードを入力し、[Enter] キーを押します。AMI を選択するには、[Select] (選択) を選択します。

Recents (最新情報)

最近使用した AMI が表示されます。

[Recently launched] (最近の起動) または [Currently in use] (現在使用中) を選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から AMI を選択します。

マイ AMI

お客様が所有しているプライベート AMI、またはお客様が共有しているプライベート AMI。

[Owned by me] (ユーザーによる所有) または [Shared with me] (共有されている) を選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) から AMI を選択します。

クイックスタート

AMI はオペレーティングシステム (OS) ごとにグループ化されているため、すぐに作業を開始できます。

まず、必要な OS を選択し、次に [Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) で、AMI を選択します。無料利用枠の対象となる AMI を選択するには、AMI が [Free tier eligible] (無料利用枠の対象) とマークされていることを確認してください。

Browse more AMIs (AMI をさらに表示する)

AMI カタログ全体を表示するには、[Browse more AMIs] (AMI をさらに表示する) を選択します。

- 利用可能な AMI すべてを検索するには、検索バーにキーワードを入力し、[Enter] キーを押します。
- Systems Manager パラメータを使用して AMI を検索するには、検索バーの右側にある矢印ボタンを選択し、[Search by Systems Manager parameter] (Systems Manager パラメータで検索)

を選択します。詳細については、「Systems Manager パラメータを使用した AMI の検索」を参照してください。

- 起動テンプレートからインスタンスが起動されたときに AMI に変換される Systems Manager パラメータを指定するには、検索バーの右側にある矢印を選択し、[カスタム値/Systems Manager パラメータを指定する] を選択します。詳細については、「AMI ID のかわりに Systems Manager パラメータを使用する」を参照してください。
- カテゴリで検索するには、[Quickstart AMIs] (AMI のクイックスタート)、[My AMIs] (私の AMI)、[AWS Marketplace AMIs]、または [Community AMIs] (コミュニティ AMI) を選択します。

  AWS Marketplace は、AMI を含む AWS 上で動作するソフトウェアを購入することができるオンラインストアです。AWS Marketplace からのインスタンスの起動の詳細については、AWS Marketplace インスタンスの起動 を参照してください。[Community AMIs] (コミュニティ AMI) では、AWS のコミュニティのメンバーが他の人が利用可能とした AMI を見つけることができます。Amazon または検証済みパートナーからの AMI は、[Verified provider] (検証済みプロバイダー) のマークが付されます。
- AMI のリストをフィルターするには、画面左の [Refine results] (結果を絞り込む) で 1 つまたは複数のチェックボックスをオンにします。フィルターオプションは、選択した検索カテゴリに応じて異なります。
- 各 AMI の [Root device type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは [ebs] (Amazon EBS でバックアップ) または [instance-store] (インスタンスストアでバックアップ) です。詳細については、ルートデバイスのストレージ を参照してください。
- 各 AMI の [Virtualization type] を確認します。必要なタイプはどの AMI かに注意してください。タイプは [hvm] または [paravirtual] です。例えば、一部のインスタンスタイプには HVM が必要です。詳細については、「Linux AMI 仮想化タイプ」を参照してください。
- 各 AMI に記載された [Boot Mode] (起動モード) を確認します。必要なブートモードがどの AMI を使用するのか注意してください。必要なブートモードは [legacy-bios]、[uefi]、または [uefi-preferred] です。詳細については、「ブートモード」を参照してください。
- ニーズを満たす AMI を選択し、[Select] を選択します。

インスタンスタイプ

インスタンスタイプは、インスタンスのハードウェア設定とサイズを定義します。インスタンスタイプが大きくなると、CPU およびメモリも増えます。詳細については、「Amazon EC2 インスタンスタイプ」を参照してください。

[Instance type] (インスタンスタイプ) では、インスタンスタイプを選択します。あるいは、インスタンス属性を指定することで、それらの属性を持つインスタンスタイプを Amazon EC2 により識別させることも可能です。

> **Note**
>
> インスタンス属性の指定がサポートされるのは、Auto Scaling グループ、EC2 フリート、およびスポットフリートを使用してインスタンスを起動する場合のみです。詳細については、「属性ベースのインスタンスタイプの選択を使用して Auto Scaling グループを作成する」、「EC2 フリートの属性ベースのインスタンスタイプの選択」、および「スポットフリートの属性ベースのインスタンスタイプの選択」を参照してください。
> インスタンス起動ウィザードまたは RunInstances API で起動テンプレートを使用する予定がある場合は、インスタンスタイプを選択する必要があります。

- [インスタンスタイプ]: インスタンスタイプが、指定した AMI と互換性があることを確認します。詳細については、インスタンスタイプ を参照してください。
- [Compare instance types] (インスタンスタイプの比較): vCPU の数、アーキテクチャ、メモリ量 (GiB)、ストレージ量 (GB)、ストレージタイプ、ネットワークパフォーマンスなどの属性ごとにさまざまなインスタンスタイプを比較できます。
- [アドバイスの取得]: インスタンスタイプに関するガイダンスやアドバイスは、Amazon Q EC2 インスタンスタイプセレクターから入手できます。詳細については、「新しいワークロードのインスタンスタイプに関する推奨事項の取得」を参照してください。
- [Advanced] (アドバンスト): インスタンス属性を指定し、Amazon EC2 がそれらの属性を持つインスタンスタイプを識別できるようにするには、[Advanced] (アドバンスト) を選択してから、[Specify instance type attributes] (インスタンスタイプの属性を指定する) を選択します。
  - [Number of vCPUs] (vCPU の数): コンピューティングの要件に応じて、vCPU の最小数と最大数を入力します。制限がないことを示すには、最小値に **0** を入力し、最大値を空白のままにします。
  - [Amount of memory (MiB)] (メモリの量 (MiB)): コンピューティング要件に対応する最小メモリ量と最大メモリ量を MiB 単位で入力します。制限がないことを示すには、最小値に **0** を入力し、最大値を空白のままにします。
  - コンピューティング要件をより詳細に表現するには、[Optional instance type attributes] (オプションのインスタンスタイプ属性) を展開し、[Add attribute] (属性の追加) を選択します。各属性の詳細については、「Amazon EC2 API リファレンス」の「InstanceRequirementsRequest」を参照してください。

- [Resulting instance types] (インスタンスタイプの結果): 指定した属性に一致するインスタンスタイプをプレビューできます。インスタンスタイプを除外するには、[Add attribute] (属性の追加) を選択し、[Attribute] (属性) リストから [Excluded instance types] (インスタンスタイプの除外) を選択します。[Attribute value] (属性値) リストから、除外したいインスタンスタイプを選択します。

キーペア (ログイン)

インスタンスのキーペア。

[Key pair name] (キーペア名) には、既存のキーペアを選択するか、[Create new key pair] (新しいキーペアを作成) を選択して新しいキーペアを作成します。詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンス を参照してください。

ネットワーク設定

必要に応じて、ネットワーク設定を設定します。

- [サブネット]: インスタンスは、アベイラビリティーゾーン、ローカルゾーン、Wavelength Zone、Outpost のいずれかに関連付けられたサブネットで起動できます。

  アベイラビリティーゾーンでインスタンスを起動するには、インスタンスを起動するサブネットを選択します。新しいサブネットを作成するには、[Create new subnet] を選択して Amazon VPC コンソールに移動します。終了したらウィザードに戻り、[Refresh] (更新) アイコンを選択して一覧にサブネットを読み込みます。

  ローカルゾーンでインスタンスを起動するには、ローカルゾーン内に作成したサブネットを選択します。

  アウトポストでインスタンスを起動するには、アウトポストに関連付けられた VPC 内のサブネットを選択します。
- [Firewall (security groups)] (ファイアウォール (セキュリティグループ)): 1 つもしくは複数のセキュリティグループを使用して、インスタンスのファイアウォールルールを定義します。このルールでは、どの着信ネットワークトラフィックをインスタンスに配信するかを指定します。他のトラフィックはすべて無視されます。セキュリティグループの詳細については、Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ を参照してください。

  ネットワークインターフェイスを追加する場合、そのネットワークインターフェイスにも、同じセキュリティグループを指定する必要があります。

次のようにセキュリティグループを選択または作成します。

- 既存のセキュリティグループを選択するには、[Select existing security group] (既存のセキュリティグループを選択) を選択し、[Common security groups] (共通セキュリティグループ) からセキュリティグループを選択します。
- 新しいセキュリティグループを作成するには、[Create security group] (セキュリティグループの作成) を選択します。

  ニーズに応じたルールを追加できます。例えば、ウェブサーバーとしてインスタンスを使用する場合には、ポート 80 (HTTP) とポート 443 (HTTPS) を開いて、インターネットトラフィックを許可します。

  ルールを追加するには、[Add security group rule] (セキュリティグループルールの追加) を選択します。[Type] (タイプ) で、ネットワークトラフィックタイプを選択します。[Protocol] (プロトコル) フィールドには、ネットワークトラフィックの送信を可能とするため、プロトコルが自動的に入力されます。[Source type] (送信元タイプ) で送信元のタイプを選択します。[My IP] (マイ IP) を選択し、起動テンプレートによりコンピュータのパブリック IP アドレスを追加させます。ただし、ISP 経由で、またはファイアウォールの内側から静的な IP アドレスなしで接続している場合は、クライアントコンピュータで使用されている IP アドレスの範囲を見つける必要があります。

  > **Warning**
  >
  > すべての IP アドレス (`0.0.0.0/0`) から SSH や RDP でインスタンスにアクセスできるようにするルールは、テスト用のインスタンスを短時間で立ち上げ、すぐに停止または終了させる場合には許容されますが、本番環境では危険です。特定の IP アドレスまたは特定のアドレス範囲にのみ、インスタンスへのアクセスを限定してください。

- 高度なネットワーク設定

  ネットワークインターフェイス

  - [デバイスインデックス]: ネットワークインターフェイスのデバイス番号。例えば、プライマリネットワークインターフェイスなら `eth0` です。フィールドに何も指定しない場合、AWS がプライマリネットワークインターフェイスを作成します。
  - [Network Interface] (ネットワークインターフェイス): [New interface] (新しいインターフェイス) を選択して Amazon EC2 によって新しいインターフェイスを作成するか、既存の使用できるネットワークインターフェイスを選択します。

- [説明]: (オプション) 新しいネットワークインターフェイスの説明。
- [Subnet] (サブネット): 新しいネットワークインターフェイスを作成するサブネット。プライマリネットワークインターフェイス (eth0) の場合、これはインスタンスが起動する先のサブネットです。eth0 に既存のネットワークインターフェイスを入力すると、インスタンスはネットワークインターフェイスが存在するサブネット内で起動します。
- [セキュリティグループ]: ネットワークインターフェイスを関連付ける VPC 内の 1 つ以上のセキュリティグループ。
- [Auto-assign Public IP] (自動割り当てパブリック IP): インスタンスがパブリック IPv4 アドレスを受け取るかどうかを指定します。デフォルトで、デフォルトのサブネットにあるインスタンスはパブリック IPv4 アドレスを受け取り、デフォルト以外のサブネットにあるインスタンスは受け取りません。[Enable] または [Disable] を選択すると、これがサブネットのデフォルト設定より優先されます。詳細については、パブリック IPv4 アドレス を参照してください。
- [プライマリ IP]: サブネットの範囲からの プライマリプライベート IPv4 アドレス。Amazon EC2 によって自動的にプライベート IPv4 アドレスが選択されるようにするには、空白のままにします。
- [Secondary IP] (セカンダリ IP): サブネットの範囲内にある 1 つまたは複数の追加のプライベート IPv4 アドレス。[Manually assign] (手動割り当て) を選択し、IP アドレスを入力します。別の IP アドレスを追加するには、[Add IP] (IP の追加) を選択します。または、Amazon EC2 により自動で割り当てるようにするには、[Automatically assign] (自動割り当て) を選択し、追加する IP アドレスの数を入力します。
- (IPv6 のみ) [IPv6 IP]: サブネットの範囲の IPv6 アドレス。[Manually assign] (手動割り当て) を選択し、IP アドレスを入力します。別の IP アドレスを追加するには、[Add IP] (IP の追加) を選択します。または、Amazon EC2 により自動で割り当てるようにするには、[Automatically assign] (自動割り当て) を選択し、追加する IP アドレスの数を入力します。
- [IPv4 Prefixes] (IPv4 プレフィクス): ネットワークインターフェイスの IPv4 プレフィクス。
- [IPv6 Prefixes] (IPv6 プレフィクス): ネットワークインターフェイスの IPv6 プレフィクス。
- (オプション)プライマリ IPv6 IP の割り当て: デュアルスタックまたは IPv6 専用のサブネットでインスタンスを起動する場合、[プライマリ IPv6 IP を割り当て]を選択できます。　プライマリ IPv6 アドレスを割り当てると、インスタンスまたは ENI へのトラフィックの中断を回避できます。このインスタンスが IPv6 アドレスが変更されないことに依存する場合、[有効化] を選択します。インスタンスを起動すると、AWS ではアタッチされている ENI に関連付けられた IPv6 アドレスがインスタンスにプライマリ IPv6 アドレスとして自動的に割り当てられます。IPv6 GUA アドレスをプライマリ IPv6 として有効にすると、無効にすることはできません。IPv6 GUA アドレスをプライマリ IPv6 にすることを有効にすると、インスタンスが終了するか、ネッ

トワークインターフェースがデタッチされるまで、最初の IPv6 GUA がプライマリ IPv6 アドレスになります。インスタンスに複数の IPv6 アドレスがアタッチされていて、プライマリ IPv6 アドレスを有効にすると、ENI に関連付けられた最初の IPv6 GUA アドレスがプライマリ IPv6 アドレスになります。

- [終了時に削除]: インスタンス終了時にネットワークインターフェイスを削除するかどうか。
- Elastic Fabric Adapter: ネットワークインターフェイスが Elastic Fabric Adapter かどうかを示します。詳細については、「Elastic Fabric Adapter」を参照してください。
- ネットワークカードインデックス: ネットワークカードのインデックス。プライマリネットワークインターフェイスは、ネットワークカードインデックス 0 に割り当てる必要があります。インスタンスタイプによっては、複数のネットワークカードがサポートされているものもあります。
- ENA Express: ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD テクノロジーは、パケットスプレーメカニズムを使用して負荷を分散し、ネットワークの混雑を回避します。ENA Express を有効にすると、サポートされているインスタンスは、可能な場合は通常の TCP トラフィックに加えて SRD を使用して通信できるようになります。[有効化] または [無効化] を選択しない限り、起動テンプレートにはインスタンスの ENA Express 設定は含まれません。
- [ENA Express UDP]: ENA Express を有効にしている場合は、オプションで UDP トラフィックに使用できます。[有効化] または [無効化] を選択しない限り、起動テンプレートにはインスタンスの ENA Express 設定は含まれません。

さらにネットワークインターフェイスを追加するには、[Add network interface] (ネットワークインターフェイスを追加) を選択します。追加できるネットワークインターフェイスの数は、選択したインスタンスタイプでサポートされている数によって異なります。追加のネットワークインターフェイスは、同じ VPC の別のサブネット、または所有している別の VPC のサブネットに配置できます (サブネットがインスタンスと同じアベイラビリティーゾーンにある場合)。別の VPC でサブネットを選択すると、追加したネットワークインターフェイスの横に [マルチ VPC] ラベルが表示されます。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。別の VPC から追加の ENI をアタッチする場合は、その VPC から ENI のセキュリティグループを選択する必要があります。

詳細については、「Elastic Network Interface」を参照してください。複数のネットワークインターフェイスを指定した場合、インスタンスはパブリック IPv4 アドレスを受け取ることはできません。さらに、eth0 に既存のネットワークインターフェイスを指定した場合、[Auto-assign Public IP] を使用してサブネットのパブリック IPv4 設定をオーバーライドする操作は禁止されます。詳細については、インスタンス起動時のパブリック IPv4 アドレスの割り当て を参照してください。

ストレージの設定

起動テンプレートのために AMI を指定した場合、その AMI には、ルートボリュームである [Volume 1 (AMI Root)] (ボリューム 1 (AMI Root) を含む、1 つまたは複数のストレージボリュームが含まれます。インスタンスにアタッチする追加のボリュームを指定できます。

[Simple] (シンプル) または [Advanced] (アドバンスト) ビューを使用できます。[Simple] (シンプル) ビューでは、ボリュームのサイズとタイプを指定します。すべてのボリュームパラメータを指定するには、[Advanced] (アドバンスト) ビュー (カードの右上) を選択します。

新しいボリュームを追加するには、[Add new volume] を選択します。

[Advanced] (アドバンスト) ビューでは、各ボリュームを以下のように設定できます。

- [Storage type] (ストレージタイプ): インスタンスと関連付けるボリュームのタイプ (EBS または エフェメラル) です。インスタンスストア (エフェメラル) ボリュームタイプは、それをサポートするインスタンスタイプを選択した場合にのみ使用できます。詳細については、「Amazon EC2 インスタンスストア」および「Amazon EBS ボリューム」を参照してください。
- [Device name] (デバイス名): ボリュームで利用できるデバイス名の一覧から選択します。
- [Snapshot] (スナップショット): ボリュームの作成元となるスナップショットを選択します。[Snapshot] (スナップショット) フィールドにテキストを入力して、利用できる共有スナップショットとパブリックスナップショットを検索することもできます。
- [Size (GiB)] (サイズ (GiB)): EBS ボリュームの場合、ストレージサイズを指定できます。無料利用枠の対象となる AMI とインスタンスを選択した場合でも、無料利用枠内に収めるには、合計ストレージを 30 GiB 以下に維持する必要があることに注意してください。
- [Volume type] (ボリュームタイプ): EBS ボリュームの場合、ボリュームタイプを選択します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。
- [IOPS]: プロビジョンド IOPS SSD (io1 と io2) あるいは汎用 SSD (gp3) ボリュームタイプを選択した場合、そのボリュームでサポートが可能な 1 秒あたりの I/O オペレーション数 (IOPS) を入力します。これは、io1、io2、gp3 ボリュームに必要です。gp2、st1、sc1、またはスタンダードボリュームではサポートされていません。起動テンプレートでこのパラメータを省略した場合は、起動テンプレートからインスタンスを起動するときに、パラメータの値を指定する必要があります。
- [Delete on termination] (終了時に削除): Amazon EBS ボリュームで、インスタンスの終了時にボリュームを削除する場合は [Yes] (はい) を選択し、ボリュームを保持する場合は [No] (いいえ) を選択します。詳細については、インスタンスの終了時にデータを保持する を参照してください。

- [Encrypted] (暗号化): インスタンスタイプが EBS 暗号化をサポートしている場合、[Yes] (はい) を選択し、ボリュームの暗号化を有効にできます。このリージョンでデフォルトで暗号化を有効にした場合、暗号化は有効になります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。
- [KMS Key] (KMS キー): [Encrypted] (暗号化) で [Yes] (はい) を選択し、ボリュームで暗号化を使用する場合には、カスタマーマネージド型キーを選択する必要があります。このリージョンでデフォルトの暗号化を有効にした場合は、自動的にデフォルトのカスタマーマネージド型キーが選択されます。別のキーを選択するか、作成したカスタマーマネージド型キーの ARN を指定できます。

リソースタグ

インスタンスの起動時に作成されるリソースにタグを付けるには、[Resource tags] (リソースタグ) で、[Add tag] (タグの追加) を選択し、タグのキーと値のペアを入力します。[Resource types] (リソースタイプ) で、作成時にタグを付けるリソースを指定します。すべてのリソースに同じタグを指定することも、リソースごとに異なるタグを指定することもできます。追加するタグごとに [Add tag] (タグを追加) を選択します。

起動テンプレートの使用時に作成される次のリソースにタグを指定できます。

- インスタンス
- ボリューム
- スポットインスタンスリクエスト
- ネットワークインターフェイス

> **Note**
>
> 起動テンプレート自体にタグを付けるには、[Template tags] (テンプレートタグ) でタグを指定する必要があります。詳細については、「起動テンプレートの名前、説明およびタグ」を参照してください。

高度な詳細

[Advanced details] で、セクションを開いてフィールドを表示し、インスタンスの追加パラメータを指定します。

- [Purchasing option] (購入オプション): [Request Spot Instances] (スポットインスタンスのリクエスト) を選択して、オンデマンド価格を上限とするスポット料金でスポットインスタンスをリクエストし、[Customize] (カスタマイズ) を選択して、スポットインスタンスのデフォルト設定を変更します。上限料金を設定し (非推奨)、リクエストタイプ、リクエスト期間、中断動作を変更できます。スポットインスタンスをリクエストしない場合、EC2 はデフォルトでオンデマンドインスタンスを起動します。詳細については、スポットインスタンス を参照してください。
- [IAM instance profile] (IAM インスタンスプロファイル): インスタンスに関連付ける AWS Identity and Access Management (IAM) インスタンスプロファイルを選択します。詳細については、Amazon EC2 の IAM ロール を参照してください。
- [Hostname type] (ホスト名タイプ): インスタンスのゲスト OS ホスト名をリソース名または IP 名に含めるかどうかを選択します。詳細については、Amazon EC2 インスタンスのホスト名タイプ を参照してください。
- [DNS Hostname] (DNS ホスト名): リソース名または IP 名への DNS クエリが、([Hostname type] (ホスト名タイプ) に何を選択したのかによって) IPv4 アドレス (A レコード)、IPv6 アドレス (AAAA レコード)、またはその両方で応答するかどうかを決定します。詳細については、Amazon EC2 インスタンスのホスト名タイプ を参照してください。
- [Shutdown behavior]: シャットダウン時にインスタンスを停止するか終了するかを選択します。詳細については、インスタンスによって起動されたシャットダウン動作の変更 を参照してください。
- [Stop - Hibernate behavior] (停止 – 休止動作): 休止を有効にするには、[Enable] (有効) を選択します。このフィールドは、休止の前提条件を満たすインスタンスにのみ有効です。詳細については、Amazon EC2 インスタンスの休止 を参照してください。
- [Termination protection] (終了の保護): 偶発的な終了を防ぐには、[Enable] (有効) を選択します。詳細については、「終了保護を有効化する」を参照してください。
- 停止保護: 偶発的な停止を防ぐには、[Enable] (有効化) を選択します。詳細については、「停止保護を有効にします」を参照してください。
- [Detailed CloudWatch monitoring] (詳細な CloudWatch モニタリング): Amazon CloudWatch によるインスタンスの詳細モニタリングを許可する場合、[Enable] (有効) を選択します。別途 料金がかかります。詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。
- [Elastic inference]: EC2 CPU インスタンスにアタッチする Elastic Inference アクセラレータ。詳細については、Amazon Elastic Inference デベロッパーガイドの「Working with Amazon Elastic Inference の使用」を参照してください。

> **Note**
>
> 2023 年 4 月 15 日以降、AWS では Amazon Elastic Inference (EI) への新規顧客のオンボーディングは行わず、既存の顧客がより価格とパフォーマンスの良いオプションにワークロードを移行できるよう支援します。2023 年 4 月 15 日以降、新規顧客は Amazon SageMaker、Amazon ECS、または Amazon EC2 の Amazon EI アクセラレータを使用してインスタンスを起動できなくなります。ただし、過去 30 日間に Amazon EI を少なくとも 1 回使用した顧客は、現在の顧客と見なされ、サービスを引き続き使用できます。

- [Credit specification] (クレジット指定): アプリケーションがベースラインを越えて必要なだけバーストできることを有効にするには、[Unlimited] (無制限) を選択します。このフィールドは、T インスタンスでのみ有効です。追加料金が適用される場合があります。詳細については、バーストパフォーマンスインスタンス を参照してください。
- [プレイスメントグループ名]: インスタンスを起動する先のプレイスメントグループを指定します。既存のプレイスメントグループを選択するか、新しいプレイスメントグループを作成することができます。すべてのインスタンスタイプがプレイスメントグループ内で起動できるわけではありません。詳細については、プレイスメントグループ を参照してください。
- [EBS-optimized instance] (EBS 最適化インスタンス): Amazon EBS I/O 専用の追加キャパシティーを利用する場合は、[Enable] (有効) を選択します。すべてのインスタンスタイプがこの機能をサポートしているわけではありません。別途 料金がかかります。詳細については、「the section called “EBS 最適化”」を参照してください。
- [Capacity Reservation] (キャパシティ予約): インスタンスを起動するキャパシティ予約を指定します。任意のキャパシティ予約 ([Open] (オープン))、特定のキャパシティ予約 ([Target by ID] (ID を対象とする))、またはキャパシティ予約グループ ([Target by group] (グループを対象とする)) のいずれかから選択します。キャパシティ予約を使用しないように指定するには、[None] (なし) を選択します。詳細については、既存の キャパシティーの予約 へのインスタンスの起動 を参照してください。
- [テナンシー]: インスタンスを共有ハードウェア ([共有])、独立した専有ハードウェア ([専有])、あるいは Dedicated Host ([Dedicated host (専有ホスト)])で実行するかを選択します。Dedicated Host でインスタンスを起動する場合は、インスタンスをホストリソースグループ内で起動するかどうかを指定できます。または、特定の Dedicated Host をターゲットとして設定できます。追加料金が適用される場合があります。詳細については、Dedicated InstancesおよびDedicated Hostsを参照してください。

- [RAM disk ID] (RAM ディスク ID): (準仮想化 (PV) AMI に対してのみ有効) インスタンスの RAM ディスクを選択します。カーネルを選択した場合は、サポートするドライバーと共に特定の RAM ディスクを選択しなければならない可能性があります。
- [Kernel ID] (カーネル ID): (準仮想化 (PV) AMI に対してのみ有効) インスタンスのカーネルを選択します。
- [Nitro Enclaves]: Amazon EC2 インスタンスから、エンクレーブと呼ばれる分離された実行環境を作成することを許可します。AWS Nitro Enclaves のインスタンスを有効にするには、[Enable] (有効) を選択します。詳細については、「AWS Nitro Enclaves ユーザーガイド」の「AWS Nitro Enclaves とは」を参照してください。
- [ライセンス設定]: 指定したライセンス設定に対してインスタンスを起動して、ライセンスの使用状況を追跡できます。詳細については、AWS License Manager ユーザーガイドの「Create a license configuration」(ライセンス設定の作成) を参照してください。
- [Specify CPU options] (CPU オプションを指定): 起動中に [Specify CPU options] をクリックすることで、vCPU 数をカスタムで指定します。CPU コアの数とコアごとのスレッド数を設定します。詳細については、「CPU オプションの最適化」を参照してください。
- [メタデータのトランスポート]: インスタンスの IP アドレスのタイプ (IPv4、IPv6、または IPv4 と IPv6) に基づいて、この EC2 インスタンスで使用できるインスタンスメタデータサービス (IMDS) へのアクセス方法を有効または無効にすることができます。詳細については、「インスタンスメタデータの取得」を参照してください。
- [メタデータトークンの応答ホップ制限]: インスタンスメタデータへのアクセスを有効または無効にできます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [メタデータのバージョン]: インスタンスメタデータへのアクセスを有効にする場合、IMDS をリクエストするときにインスタンスメタデータサービスバージョン 2 の使用を必須にすることができます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [メタデータレスポンスのホップ制限]: IMDS を有効にする場合、メタデータトークンに許容されるネットワークホップ数を設定できます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。
- [Allow tags in metadata] (メタデータ内のタグを許可する): [Enable] (有効) を選択した場合、インスタンスはメタデータ内のすべてのタグへのアクセスを許可します。テンプレートにこの設定を含めない場合、インスタンスのメタデータに含まれるタグへのアクセスは、デフォルトで無効になります。詳細については、インスタンスメタデータのタグへのアクセスを許可する を参照してください。

- [ユーザーデータ]: 起動時にインスタンスを設定するユーザーデータ、または設定スクリプトを実行するユーザーデータを指定できます。詳細については、「起動時に Linux インスタンスでコマンドを実行する」を参照してください。

[概要]

Summary (概要) パネルから起動テンプレートの設定を確認した上で、起動テンプレートを作成します。

- 起動テンプレートの詳細を確認し、必要な変更を加えます。[Summary] (サマリー) パネルのリンクを選択すると、セクションに直接移動することができます。
- 起動テンプレートを作成する準備ができたら、[Create launch template] (起動テンプレートの作成) をクリックします。

AWS CLI

起動テンプレートを作成する際には、そのテンプレートの名と、少なくとも 1 つのインスタンス設定パラメーターを指定する必要があります。

AWS CLI を使用して起動テンプレートを作成するには

- create-launch-template コマンドを使用します。次の例では、以下を指定する起動テンプレートを作成します。
  - 起動テンプレートの名前 (*TemplateForWebServer*)
  - 起動テンプレートの説明 (*WebVersion1*)
  - 起動テンプレートのタグ (*purpose*=*production*)
  - JSON ファイルで指定された、インスタンス設定のデータ。
    - 起動するインスタンスタイプ (*r4.4xlarge*) と AMI (*ami-8c1be5f6*)
    - 合計 8 vCPU (4 コア x 2 スレッド) のコア数 (*4*) とコアごとのスレッド数 (*2*)
    - インスタンスを起動するサブネット (*subnet-7b16de0c*)
    - インスタンスに割り当てられる、パブリック IP アドレスおよび IPv6 アドレス
    - インスタンスに付けるタグ (*Name*=*webserver*)

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
```

```
    --version-description WebVersion1 \
    --tag-specifications 'ResourceType=launch-
template,Tags=[{Key=purpose,Value=production}]' \
    --launch-template-data file://template-data.json
```

インスタンスを設定するための起動テンプレートデータを含む、JSON ファイルの例を以下に示します。

```
{
    "NetworkInterfaces": [{
        "AssociatePublicIpAddress": true,
        "DeviceIndex": 0,
        "Ipv6AddressCount": 1,
        "SubnetId": "subnet-7b16de0c"
    }],
    "ImageId": "ami-8c1be5f6",
    "InstanceType": "r4.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }],
    "CpuOptions": {
        "CoreCount":4,
        "ThreadsPerCore":2
    }
}
```

以下は出力例です。

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-01238c059e3466abc",
        "LaunchTemplateName": "TemplateForWebServer",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:root",
        "CreateTime": "2017-11-27T09:13:24.000Z"
    }
}
```

既存の起動テンプレートからの起動テンプレートの作成

既存の起動テンプレートをクローンして、新しい起動テンプレートを作成するようにパラメータを調整できます。ただし、これを行えるのは、Amazon EC2 コンソールを使用している場合のみです。AWS CLI は、テンプレートのクローンをサポートしていません。

Console

既存の起動テンプレートから起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択します。
3. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力します。
4. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンの短い説明を入力します。
5. 作成時に起動テンプレートにタグを付けるには、[Template tags] を展開し、[タグの追加] を選択して、タグキーと値のペアを入力します。
6. [Source template] (ソーステンプレート) を展開し、[Launch template name] (起動テンプレート名) で、新しい起動テンプレートのベースとなる起動テンプレートを選択します。
7. [Source template version] (ソーステンプレートのバージョン) で、新しい起動テンプレートのベースとなる起動テンプレートのバージョンを選択します。
8. 必要に応じて起動パラメータを調整し、[起動テンプレートの作成] を選択します。

インスタンスからの起動テンプレートの作成

Console

インスタンスから起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選び、[Actions (アクション)]、[Create Template from Instance (インスタンスからテンプレートを作成)] の順に選択します。
4. 名前、説明、およびタグを入力し、必要に応じて起動パラメータを調整します。

> **Note**
>
> インスタンスから起動テンプレートを作成するとき、そのインスタンスのネットワークインターフェイス ID と IP アドレスはテンプレートに含まれません。

5. [起動テンプレートの作成] を選択します。

AWS CLI

AWS CLI を使用して既存のインスタンスから起動テンプレートを作成するには、まずインスタンスから起動テンプレートデータを取得し、次に起動テンプレートデータを使用して起動テンプレートを作成します。

インスタンスから起動テンプレートデータを取得するには

- get-launch-template-data コマンドを使用して、インスタンス ID を指定します。出力をベースとして使用して、新しい起動テンプレートや起動テンプレートのバージョンを作成できます。デフォルトでは、起動テンプレートデータで指定できない最上位レベルの LaunchTemplateData オブジェクトが出力に含まれています。このオブジェクトを除外するには、--query オプションを使用します。

```
aws ec2 get-launch-template-data \
    --instance-id i-0123d646e8048babc \
    --query "LaunchTemplateData"
```

出力例を次に示します。

```
    {
        "Monitoring": {},
        "ImageId": "ami-8c1be5f6",
        "BlockDeviceMappings": [
            {
                "DeviceName": "/dev/xvda",
                "Ebs": {
                    "DeleteOnTermination": true
                }
            }
        ],
        "EbsOptimized": false,
```

```
        "Placement": {
            "Tenancy": "default",
            "GroupName": "",
            "AvailabilityZone": "us-east-1a"
        },
        "InstanceType": "t2.micro",
        "NetworkInterfaces": [
            {
                "Description": "",
                "NetworkInterfaceId": "eni-35306abc",
                "PrivateIpAddresses": [
                    {
                        "Primary": true,
                        "PrivateIpAddress": "10.0.0.72"
                    }
                ],
                "SubnetId": "subnet-7b16de0c",
                "Groups": [
                    "sg-7c227019"
                ],
                "Ipv6Addresses": [
                    {
                        "Ipv6Address": "2001:db8:1234:1a00::123"
                    }
                ],
                "PrivateIpAddress": "10.0.0.72"
            }
        ]
    }
```

出力を次のようにファイルに直接書き込むことができます。

```
aws ec2 get-launch-template-data \
    --instance-id i-0123d646e8048babc \
    --query "LaunchTemplateData" >> instance-data.json
```

起動テンプレートデータを使用して起動テンプレートを作成するには

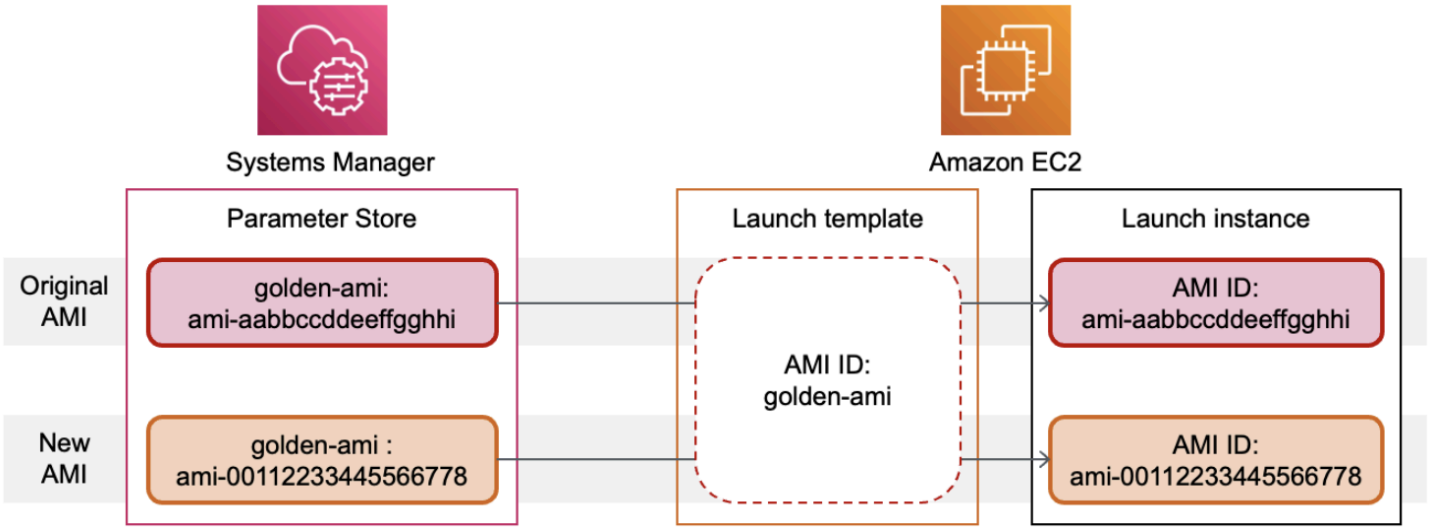
- create-launch-template コマンドを使用して、前の手順の出力を使用して起動テンプレートを作成します。AWS CLI を使用した起動テンプレートの作成の詳細については、「定義したパラメータを使用した新しい起動テンプレートの作成」を参照してください。

AMI ID のかわりに Systems Manager パラメータを使用する

起動テンプレートで AMI ID を指定する代わりに、AWS Systems Manager パラメータを指定できます。AMI ID が変更された場合は、Systems Manager パラメータストアの Systems Manager パラメータを更新することで、AMI ID を一箇所で更新できます。パラメータは、他の AWS アカウントと共有することもできます。AMI パラメータは、1 つのアカウントで一元的に保存および管理し、これを参照する必要がある他のすべてのアカウントに共有することができます。Systems Manager パラメータを使用すると、すべての起動テンプレートを 1 回のアクションで更新できます。

Systems Manager パラメータは、Systems Manager パラメータストアで作成できるユーザー定義のキーと値のペアです。パラメータストアは、アプリケーションの設定値を保存するための一元的な場所を提供します。詳細については、「AWS Systems Manager ユーザーガイド」の「AWS Systems Manager パラメータストア」を参照してください。

次の図では、`golden-ami` パラメータは最初にパラメータストア内の元の AMI `ami-aabbccddeeffgghhi` にマッピングされます。起動テンプレートの AMI ID の値は `golden-ami` です。この起動テンプレートを使用してインスタンスを起動するとき、AMI ID は `ami-aabbccddeeffgghhi` に変換されます。その後、AMI が更新され、新しい AMI ID が取得されます。パラメータストアでは、`golden-ami` パラメータが新しい `ami-00112233445566778` パラメータにマッピングされます。起動テンプレートは変更されずに引き継がれます。この起動テンプレートを使用してインスタンスを起動するとき、AMI ID は新しい `ami-00112233445566778` に変換されます。



AMI ID の Systems Manager パラメータ形式

起動テンプレートを AMI ID の代わりに使用する場合、ユーザー定義の Systems Manager パラメータを次の形式に従う必要があります。

- パラメータ型: `String`
- パラメータデータ型: `aws:ec2:image` - これにより、パラメータストアは入力した値が AMI ID の適切な形式であることを検証します。

AMI ID での有効なパラメータの作成方法については、「AWS Systems Manager ユーザーガイド」の「Systems Manager パラメータを作成する」を参照してください。

起動テンプレートの Systems Manager パラメータ形式

起動テンプレートの AMI ID の代わりに Systems Manager パラメータを使用するには、起動テンプレートでパラメータを指定するときに次のいずれかの形式を使用する必要があります。

パブリックパラメータを参照するには:

- `resolve:ssm:public-parameter`

同じアカウントに保存されているパラメータを参照するには:

- `resolve:ssm:parameter-name`
- `resolve:ssm:parameter-name:version-number` - バージョン番号自体がデフォルトのラベルです
- `resolve:ssm:parameter-name:label`

別の AWS アカウント から共有されたパラメータを参照するには:

- `resolve:ssm:parameter-ARN`
- `resolve:ssm:parameter-ARN:version-number`
- `resolve:ssm:parameter-ARN:label`

パラメータバージョン

Systems Manager パラメータはバージョニングされたリソースです。パラメータを更新すると、そのパラメータの新しいバージョンが連続して作成されます。Systems Manager は、特定のバージョンのパラメータにマッピングできるパラメータラベルをサポートしています。

例えば、golden-ami パラメータには、1、2 および 3 の 3 つのバージョンがあります。バージョン 2 にマッピングするパラメータラベル beta と、バージョン 3 にマップするパラメータラベル prod を作成できます。

起動テンプレートでは、次のいずれかの形式を使用して golden-ami パラメータのバージョン 3 を指定できます。

- resolve:ssm:golden-ami:3
- resolve:ssm:golden-ami:prod

バージョンまたはラベルの指定は任意です。バージョンやパラメータを指定しない場合は最新バージョンのパラメータが使用されます。

起動テンプレートの Systems Manager パラメータを指定する

起動テンプレートまたは起動テンプレートの新しいバージョンを作成するときに、起動テンプレートで AMI ID の代わりに Systems Manager パラメータを指定できます。

Console

起動テンプレートの Systems Manager パラメータを指定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択します。
3. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力します。
4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMIs] (その他の AMI を閲覧する) を選択します。
5. 検索バーの右側にある矢印ボタンを選択したら、[カスタム値を指定 / Systems Manager パラメータ] を選択します。
6. [カスタム値または Systems Manager のパラメータを指定] ダイアログボックスで、次の手順を実行します。

   a. [AMI ID または Systems Manager パラメータ文字列] の場合、次の形式のいずれかを使用して Systems Manager パラメータ名を入力します。

      パブリックパラメータを参照するには:

- **resolve:ssm:*public-parameter***

  同じアカウントに保存されているパラメータを参照するには:

- **resolve:ssm:*parameter-name***
- **resolve:ssm:*parameter-name*:*version-number***
- **resolve:ssm:*parameter-name*:*label***

  別の AWS アカウント から共有されたパラメータを参照するには:

- **resolve:ssm:*parameter-ARN***
- **resolve:ssm:*parameter-ARN*:*version-number***
- **resolve:ssm:*parameter-ARN*:*label***

b. [Save] を選択します。

7. 必要に応じて他の起動テンプレートパラメータを指定し、その後に [起動テンプレートの作成] を選択します。

詳細については、「定義したパラメータを使用した新しい起動テンプレートの作成」を参照してください。

AWS CLI

起動テンプレートの Systems Manager パラメータを指定するには

- create-launch-template コマンドを使用して、起動テンプレートを作成します。使用する AMI を指定するには、次のいずれかの形式を使用して Systems Manager パラメータ名を入力します。

  パブリックパラメータを参照するには:

- **resolve:ssm:*public-parameter***

  同じアカウントに保存されているパラメータを参照するには:

- **resolve:ssm:*parameter-name***
- **resolve:ssm:*parameter-name*:*version-number***

- **resolve:ssm:*parameter-name*:*label***

別の AWS アカウント から共有されたパラメータを参照するには:

- **resolve:ssm:*parameter-ARN***
- **resolve:ssm:*parameter-ARN*:*version-number***
- **resolve:ssm:*parameter-ARN*:*label***

次の例では、以下を指定する起動テンプレートを作成します。

- 起動テンプレートの名前 (*TemplateForWebServer*)
- 起動テンプレートのタグ (*purpose*=*production*)
- JSON ファイルで指定された、インスタンス設定のデータ。
  - 使用する AMI (resolve:ssm:*golden-ami*)
  - 起動するインスタンスタイプ (*m5.4xlarge*)
  - インスタンスに付けるタグ (*Name*=*webserver*)

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForWebServer \
    --tag-specifications 'ResourceType=launch-
template,Tags=[{Key=purpose,Value=production}]' \
    --launch-template-data file://template-data.json
```

インスタンスを設定するための起動テンプレートデータを含む、JSON ファイルの例を以下に示します。ImageId の値は Systems Manager パラメータ名で、必要な形式 resolve:ssm:*golden-ami* で入力されます。

```
{"LaunchTemplateData": {
    "ImageId": "resolve:ssm:golden-ami",
    "InstanceType": "m5.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
```

```
    }]
  }
}
```

関連リソース

Systems Manager パラメータの使用に関する詳細は、Systems Manager のドキュメントの次のリファレンス資料を参照してください。

- Amazon EC2 でサポートされている AMI パブリックパラメータを検索する方法については、「Calling AMI public parameters」を参照してください。
- パラメータを他の AWS アカウントと共有する方法、または AWS Organizations を介して共有する方法については、「Working with shared parameters」を参照してください。
- パラメータが正常に作成されたかどうかをモニタリングする方法については、「Native parameter support for Amazon Machine Image IDs」を参照してください。

制限事項

- 現在、EC2 フリートとスポットフリートは、AMI ID の代わりに Systems Manager パラメータが指定されている起動テンプレートの使用をサポートしていません。EC2 フリートとスポットフリートの場合、起動テンプレートに AMI を指定する場合、AMI ID を指定する必要があります。
- Amazon EC2 Auto Scaling には、その他の制限があります。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Use AWS Systems Manager parameters instead of AMI IDs in launch templates」を参照してください。

起動テンプレートの変更 (起動テンプレートのバージョンの管理)

起動テンプレートは変更不可能です。起動テンプレートを作成したら、それを変更することはできません。代わりに、必要な変更を含む新しいバージョンの起動テンプレートを作成できます。

起動テンプレートの別のバージョンの作成、デフォルトバージョンの設定、起動テンプレートバージョンの説明、不要になったバージョンの削除を行うことができます。

タスク

- 起動テンプレートのバージョンの作成
- デフォルトの起動テンプレートのバージョンの設定
- 起動テンプレートのバージョンの説明

- 起動テンプレートのバージョンの削除

起動テンプレートのバージョンの作成

起動テンプレートのバージョンを作成する際、新しいバージョンに新しい起動パラメータを指定するか、または既存のバージョンをベースとして使用できます。起動パラメーターの詳細については、「起動テンプレートの作成」を参照してください。

Console

起動テンプレートのバージョンを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Launch Templates] を選択します。
3. 起動テンプレートを選択し、[アクション]、[Modify template (Create new version)] の順に選択します。
4. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンについての説明を入力します。
5. (オプション) [Source template] (ソーステンプレート) を展開し、新しい起動テンプレートバージョンのベースとして使用する起動テンプレートのバージョンを選択します。新しい起動テンプレートバージョンは、この起動テンプレートバージョンから起動パラメータを継承します。
6. 必要に応じて起動パラメータを変更し、[起動テンプレートの作成] を選択します。

AWS CLI

起動テンプレートのバージョンを作成するには

- create-launch-template-version コマンドを使用します。新しいバージョンのベースとなるソースバージョンを指定できます。新しいバージョンはこのバージョンの起動パラメータを継承し、`--launch-template-data` を使用してパラメータを上書きできます。次の例では、起動テンプレートのバージョン 1 に基づいて新しいバージョンを作成し、異なる AMI ID を指定します。

```
aws ec2 create-launch-template-version \
    --launch-template-id lt-0abcd290751193123 \
    --version-description WebVersion2 \
    --source-version 1 \
```

```
    --launch-template-data "ImageId=ami-c998b6b2"
```

デフォルトの起動テンプレートのバージョンの設定

起動テンプレートにデフォルトバージョンを設定できます。起動テンプレートからインスタンスを起動し、バージョンを指定しない場合、インスタンスはデフォルトバージョンのパラメータを使用して起動されます。

Console

デフォルトの起動テンプレートのバージョンを設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Launch Templates] を選択します。
3. 起動テンプレートを選択し、[アクション]、[デフォルトバージョンの設定] を選択します。
4. [テンプレートバージョン] で、デフォルトバージョンとして設定するバージョン番号を選択し、[デフォルトバージョンとして設定] を選択します。

AWS CLI

デフォルトの起動テンプレートのバージョンを設定するには

- modify-launch-template コマンドを使用して、デフォルトとして設定するバージョンを指定します。

```
aws ec2 modify-launch-template \
    --launch-template-id lt-0abcd290751193123 \
    --default-version 2
```

起動テンプレートのバージョンの説明

コンソールを使用して、選択した起動テンプレートのすべてのバージョンを表示したり、特定のバージョン番号と一致する最新バージョンやデフォルトバージョンの起動テンプレートを一覧表示したりできます。AWS CLI を使用すると、指定した起動テンプレートのすべてのバージョン、各バージョン、特定範囲のバージョンを表示できます。また、アカウント内にあるすべての起動テンプレートについて、すべての最新バージョンを表示したり、すべてのデフォルトバージョンを表示したりすることもできます。

Console

起動テンプレートのバージョンを説明するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Launch Templates] を選択します。
3. 特定の起動テンプレートのバージョンを表示したり、特定のバージョン番号と一致する最新バージョンやデフォルトバージョンの起動テンプレートを一覧表示したりできます。
   - 起動テンプレートのバージョンを表示するには、起動テンプレートを選択します。[バージョン] タブの [バージョン] から、詳細を表示するバージョンを選択します。
   - 特定のバージョン番号と一致する最新バージョンの起動テンプレートを一覧表示するには、検索バーから [最新バージョン] を選択し、バージョン番号を選択します。
   - 特定のバージョン番号と一致するデフォルトバージョンの起動テンプレートを一覧表示するには、検索バーから [デフォルトバージョン] を選択し、バージョン番号を選択します。

AWS CLI

起動テンプレートのバージョンを説明するには

- delete-launch-template-versions コマンドを使用して、バージョン番号を指定します。次の例では、バージョン *1* と *3* を指定しています。

```
aws ec2 describe-launch-template-versions \
    --launch-template-id lt-0abcd290751193123 \
    --versions 1 3
```

アカウント内にある起動テンプレートのすべての最新バージョンやデフォルトバージョンを説明するには

- delete-launch-template-versions コマンドを使用し、$Latest または $Default を指定するか、両方を指定します。呼び出しでは、起動テンプレートの ID と名前を省略する必要があります。バージョン番号を指定することはできません。

```
aws ec2 describe-launch-template-versions \
    --versions "$Latest,$Default"
```

起動テンプレートのバージョンの削除

起動テンプレートのバージョンが不要になった場合には、それを削除することができます。

考慮事項

- 削除後にバージョン番号を置き換えることはできません。
- 起動テンプレートのデフォルトバージョンは削除できません。まず、デフォルトとして別のバージョンを割り当てる必要があります。デフォルトバージョンが起動テンプレートの唯一のバージョンである場合は、 起動テンプレート全体を削除する必要があります。
- コンソールを使用する場合、一度に削除できる起動テンプレートバージョンは 1 つです。AWS CLI を使用する場合は、一度のリクエストで最大 200個 の起動テンプレートバージョンを削除できます。1 回のリクエストで 200 を超えるバージョンを削除するには、起動テンプレートを削除します。その場合、バージョンもすべて削除されます。

Console

起動テンプレートのバージョンを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Launch Templates] を選択します。
3. 起動テンプレートを選択し、[アクション]、[テンプレートのバージョンの削除] を選択します。
4. 削除するバージョンを選択し、[削除] を選択します。

AWS CLI

起動テンプレートのバージョンを削除するには

- delete-launch-template-versions コマンドを使用して、削除するバージョン番号を指定します。1 回のリクエストで削除する起動テンプレートバージョンを最大 200 個指定できます。

```
aws ec2 delete-launch-template-versions \
    --launch-template-id lt-0abcd290751193123 \
    --versions 1
```

起動テンプレートの削除

起動テンプレートが不要になった場合には、それを削除することができます。起動テンプレートを削除すると、すべてのバージョンが削除されます。特定のバージョンの起動テンプレートのを削除するには、「起動テンプレートのバージョンの削除」を参照してください。

起動テンプレートを削除しても、起動テンプレートから起動したインスタンスには影響しません。

Console

起動テンプレートを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Launch Templates] を選択します。
3. 起動テンプレートを選択し、[アクション]、[テンプレートの削除] を選択します。
4. **`Delete`** と入力して削除を確認し、[Delete (削除)] を選択します。

AWS CLI

起動テンプレートを削除するには

- delete-launch-template (AWS CLI) コマンドを使用して、起動テンプレートを指定します。

```
aws ec2 delete-launch-template --launch-template-id lt-01238c059e3466abc
```

起動テンプレートからのインスタンスの起動

起動テンプレートは、いくつかのインスタンス起動サービスでサポートされています。このトピックでは、EC2 インスタンス起動ウィザード、Amazon EC2 Auto Scaling、EC2 フリート、スポットフリートを使用してインスタンスを起動するときに起動テンプレートを使用する方法について説明します。

トピック

- 「起動テンプレートからのインスタンスの起動」
- Amazon EC2 Auto Scaling での起動テンプレートの使用
- EC2 フリート での起動テンプレートの使用
- スポットフリートで起動テンプレートを使用する

「起動テンプレートからのインスタンスの起動」

起動テンプレートに含まれているパラメータを使用してインスタンスを起動できます。インスタンスを起動する前に、オプションで起動パラメータを上書きまたは追加できます。

起動テンプレートを使用して起動されたインスタンスには、aws:ec2launchtemplate:id と aws:ec2launchtemplate:version のキーを使用して自動的に 2 つのタグが割り当てられます。これらのタグを削除したり、編集することはできません。

Console

コンソールを使用して起動テンプレートからインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Launch Templates] を選択します。
3. 起動テンプレートを選択し、[アクション]、[テンプレートからインスタンスを起動する] を選択します。
4. [Source template version] (ソーステンプレートのバージョン) で、使用する起動テンプレートのバージョンを選択します。
5. [Number of instances] で、起動するインスタンスの数を指定します。
6. (オプション) [インスタンスの詳細] セクションでパラメータを変更または追加すると、起動テンプレートパラメータを上書きまたは追加することができます。
7. [テンプレートからインスタンスを起動する] を選択します。

AWS CLI

AWS CLIを使用して起動テンプレートからインスタンスを起動するには

- run-instances コマンドを使用して --launch-template パラメータを指定します。オプションで、使用する起動テンプレートのバージョンを指定します。バージョンを指定しない場合は、デフォルトバージョンが使用されます。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123,Version=1
```

- 起動テンプレートパラメータを上書きするには、run-instances コマンドでパラメータを指定します。次の例では、起動テンプレートで指定されたインスタンスタイプを上書きします (ある場合)。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123 \
    --instance-type t2.small
```

- 複雑な構造の一部である入れ子状のパラメータを指定した場合、インスタンスは、起動テンプレートで指定された複雑な構造および、指定した入れ子状の追加パラメータを使用して起動されます。

  次の例で、インスタンスは、タグ *Owner*=*TeamA* および起動テンプレートで指定された他のタグを使用して起動されます。起動テンプレートに既存の *Owner* のキーのタグがある場合、値は *TeamA* に置き換えられます。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123 \
    --tag-specifications "ResourceType=instance,Tags=[{Key=Owner,Value=TeamA}]"
```

  次の例で、インスタンスは、*/dev/xvdb* という名前のデバイス名を持つボリューム、および起動テンプレートで指定された他のブロックデバイスマッピングを使用して起動されます。起動テンプレートに */dev/xvdb* 用に定義された既存のボリュームがある場合、値は指定された値で置き換えられます。

```
aws ec2 run-instances \
    --launch-template LaunchTemplateId=lt-0abcd290751193123 \
    --block-device-mappings "DeviceName=/dev/
xvdb,Ebs={VolumeSize=20,VolumeType=gp2}"
```

インスタンスが起動しないか、状態が running ではなくすぐに terminated になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

PowerShell

AWS Tools for PowerShellを使用して起動テンプレートからインスタンスを起動するには

- [New-EC2Instance] コマンドを使用して、-LaunchTemplate パラメータを指定します。オプションで、使用する起動テンプレートのバージョンを指定します。バージョンを指定しない場合は、デフォルトバージョンが使用されます。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
```

```
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
    }
)
```

- 起動テンプレートパラメータを上書きするには、[New-EC2Instance] コマンドでパラメータを指定します。次の例では、起動テンプレートで指定されたインスタンスタイプを上書きします (ある場合)。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -InstanceType t4g.small `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
    }
)
```

- 複雑な構造の一部である入れ子状のパラメータを指定した場合、インスタンスは、起動テンプレートで指定された複雑な構造および、指定した入れ子状の追加パラメータを使用して起動されます。

  次の例で、インスタンスは、タグ *Owner*=*TeamA* および起動テンプレートで指定された他のタグを使用して起動されます。起動テンプレートに既存の *Owner* のキーのタグがある場合、値は *TeamA* に置き換えられます。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -InstanceType t4g.small  `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
        }
) `
    -TagSpecification (
```

```
        New-Object -TypeName Amazon.EC2.Model.TagSpecification -Property @{
            ResourceType = 'instance';
            Tags         = @(
                @{key = "Owner"; value = "TeamA" },
                @{key = "Department"; value = "Operations" }
            )
        }
)
```

次の例で、インスタンスは、*/dev/xvdb* という名前のデバイス名を持つボリューム、および起動テンプレートで指定された他のブロックデバイスマッピングを使用して起動されます。起動テンプレートに */dev/xvdb* 用に定義された既存のボリュームがある場合、値は指定された値で置き換えられます。

```
Import-Module AWS.Tools.EC2
New-EC2Instance `
    -InstanceType t4g.small  `
    -LaunchTemplate (
        New-Object -TypeName Amazon.EC2.Model.LaunchTemplateSpecification -
Property @{
            LaunchTemplateId = 'lt-0abcd290751193123';
            Version          = '4'
    }
) `
    -BlockDeviceMapping  (
        New-Object -TypeName Amazon.EC2.Model.BlockDeviceMapping -Property @{
            DeviceName = '/dev/xvdb';
            EBS        = (
                New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice -Property @{
                    VolumeSize = 25;
                    VolumeType = 'gp3'
                }
            )
        }
)
```

インスタンスが起動しないか、状態が running ではなくすぐに terminated になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

Amazon EC2 Auto Scaling での起動テンプレートの使用

Auto Scaling グループを作成して、グループに使用する起動テンプレートを指定できます。Auto Scalingグループ内で Amazon EC2 Auto Scaling がインスタンスを起動する際、関連する起動テンプレートで定義された起動パラメータが使用されます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループの起動テンプレートを作成する」および「詳細設定を使用して起動テンプレートを作成する」を参照してください。

起動テンプレートを使用して Auto Scaling グループを作成するには、Auto Scaling グループのインスタンスの起動に必要なパラメータを含む起動テンプレート (AMI の ID など) を作成する必要があります。コンソールには、Amazon EC2 Auto Scaling で使用できるテンプレートの、作成に役立つガイダンスが用意されています。

コンソールを使用して Auto Scaling で使用する起動テンプレートを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[起動テンプレート]、[起動テンプレートの作成] の順に選択します。
3. [起動テンプレート名] に、起動テンプレートのわかりやすい名前を入力します。
4. [Template version description] (テンプレートバージョンの説明) に、起動テンプレートバージョンの短い説明を入力します。
5. [Auto Scaling guidance] (Auto Scaling ガイダンス) でチェックボックスをオンにすると、Auto Scaling で使用するテンプレートの作成に役立つガイダンスが Amazon EC2 により表示されるようになります。
6. 必要に応じて起動パラメータを変更します。Auto Scaling ガイダンスを選択したため、一部のフィールドは必須で、一部のフィールドは使用できません。Amazon EC2 Auto Scaling の起動テンプレートの設定方法に関する詳細は、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling グループの起動テンプレートを作成する」および「詳細設定を使用して起動テンプレートを作成する」を参照してください。
7. [起動テンプレートの作成] を選択します。
8. (オプション) この起動テンプレートを使用して Auto Scaling グループを作成するには、[Next steps] (次のステップ) ページで [Create Auto Scaling group] (Auto Scaling グループの作成) を選択します。

AWS CLI を使って、さまざまなパラメータを組み合わせて起動テンプレートを作成する方法の例については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Examples for creating and managing launch templates with the AWS Command Line Interface (AWS CLI)」を参照してください。

AWS CLI を使用して、起動テンプレートを使って Auto Scaling グループを作成または更新するには

- create-auto-scaling-group または update-auto-scaling-group コマンドを使用して `--launch-template` パラメータを指定します。

起動テンプレートを使用した Auto Scaling グループの作成または更新に関する詳細は、「Amazon EC2 Auto Scaling ユーザーガイド」の以下のトピックを参照してください。

- 起動テンプレートを使用して Auto Scaling グループを作成する
- Auto Scaling グループを更新する

EC2 フリート での起動テンプレートの使用

EC2 フリート リクエストを作成して、インスタンス設定で起動テンプレートを指定できます。Amazon EC2 は、EC2 フリート リクエストを満たす際、関連する起動テンプレートで定義された起動パラメータを使用します。起動テンプレートで指定されたパラメータの一部を上書きすることができます。

詳細については、EC2 フリートの作成 を参照してください。

AWS CLI により起動テンプレートを使用して、EC2 フリートを作成するには

- create-fleet コマンドを使用します。`--launch-template-configs` パラメータを使用して、起動テンプレートと起動テンプレートの上書きを指定します。

スポットフリートで起動テンプレートを使用する

スポットフリートリクエストを作成して、インスタンス設定で起動テンプレートを指定できます。Amazon EC2 は、スポットフリートリクエストを処理する際、関連する起動テンプレートで定義された起動パラメータを使用します。起動テンプレートで指定されたパラメータの一部を上書きすることができます。

詳細については、「スポットフリートリクエストを作成します。」を参照してください。

コンソールで起動テンプレートを使用して、スポットフリートリクエストを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. [Request Spot Instances (スポットインスタンスのリクエスト)] を選択します。

4. [Launch parameters] (起動パラメータ) で、[Use a launch template] (起動テンプレートを使用する) を選択します。
5. [Launch tempate] (起動テンプレート) で、起動テンプレートを選択し、右側のフィールドから起動テンプレートのバージョンを選択します。
6. この画面で別のオプションを選択して、スポットフリートを設定します。オプションの詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)」を参照してください。
7. スポットフリートを作成する準備が整ったら、[Launch] (起動) を選択します。

AWS CLI により起動テンプレートを使用して、スポットフリートリクエストを作成するには

- request-spot-fleet コマンドを使用します。`LaunchTemplateConfigs` パラメータを使用して、起動テンプレートと起動テンプレートの上書きを指定します。

既存のインスタンスのパラメータを使用したインスタンスの起動

Amazon EC2 コンソールには、現在のインスタンスを他のインスタンスを起動するためのベースとして使用可能にする、[Launch More Like This] (同様のインスタンスをさらに起動) オプションが用意されています。このオプションでは、Amazon EC2 インスタンス起動ウィザードで、選択されたインスタンスから自動的に特定の設定が入力されます。

考慮事項

- インスタンスのクローニングは行わず、設定の詳細の一部だけを複製します。インスタンスのコピーを作成するには、最初にインスタンスから AMI を作成して、AMI からさらに多くのインスタンスを起動します。起動テンプレートを作成して、同じ起動詳細を使用してインスタンスを起動するようにしてください。
- 現在のインスタンスは `running` の状態である必要があります。

コピーされる詳細

次の設定詳細は、選択されたインスタンスからインスタンス起動ウィザードにコピーされます。

- AMI ID
- インスタンスタイプ
- アベイラビリティーゾーン、または選択されたインスタンスがある VPC とサブネット

- パブリック IPv4 アドレス。選択されたインスタンスの IPv4 アドレスが現在パブリック IPv4 アドレスの場合、選択されたインスタンスのパブリック IPv4 アドレスのデフォルト設定に関係なく、新しいインスタンスはパブリック IPv4 アドレスを受け取ります。パブリック IPv4 アドレスの詳細については、「パブリック IPv4 アドレス」を参照してください。
- プレイスメントグループ (該当する場合)
- 該当する場合は、インスタンスに関連付けられた IAM ロール
- シャットダウン動作の設定 (停止または終了)
- 終了保護設定 (true または false)
- CloudWatch モニタリング (有効または無効)
- Amazon EBS 最適化設定 (true または false)
- VPC (共有または専用) に起動する場合は、テナンシー設定
- 該当する場合は、カーネル ID および RAM ディスク ID
- ユーザーデータ (指定された場合)
- 該当する場合は、インスタンスに関連付けられたタグ
- インスタンスに関連付けられたセキュリティグループ

コピーされない詳細

次の設定の詳細は、選択したインスタンスからはコピーされません。代わりに、ウィザードはデフォルトの設定または動作を適用します。

- ネットワークインターフェイスの数 - デフォルトでは、1 つのネットワークインターフェイス、つまりプライマリネットワークインターフェイス (eth0) です。
- ストレージ - デフォルトのストレージ設定は AMI およびインスタンスタイプによって決まります。

既存のインスタンスと同様により多くのインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[イメージとテンプレート]、[同様のものを起動] を選択します。
4. インスタンス起動ウィザードが開きます。この画面で別のオプションを選択して、インスタンス設定に必要な変更を加えることができます。

インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択します。

5. インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

AWS Marketplace インスタンスの起動

AWS Marketplace 製品をサブスクライブすると、Amazon EC2 起動ウィザードを使用して、当該製品の AMI からインスタンスを起動できるようになります。有料の AMI の詳細については、有料 AMI を参照してください。起動後に受信登録をキャンセルするには、初めに受信登録から、実行されているすべてのインスタンスを削除する必要があります。詳細については、AWS Marketplace サブスクリプションを管理する を参照してください。

New console

launch wizardを使用して AWS Marketplace からインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. Amazon EC2 コンソールダッシュボードで、[インスタンスを起動] を選択します。
3. (オプション) [Names and tags] (名前とタグ) における [Name] (名前) では、インスタンス用にわかりやすい名前を入力します。
4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、[Browse more AMIs] (さらに AMI を参照) を選択し、[AWS Marketplace AMIs] タブを選択します。カテゴリを参照するか、検索機能を使用して適切な AMI を見つけます。製品を選択するには、[Select] (選択) を選択します。
5. 選択した製品の概要と共に、ウィンドウが開きます。価格情報と、ベンダーが提供したその他の情報を表示できます。準備ができたら、次のいずれかのボタンを選択します。

   - [インスタンス起動時に購読] – [インスタンスの起動] (ステップ 10) を選択すると、サブスクリプションが開始されます。
   - [今すぐ購読] – サブスクリプションはすぐに開始されます。サブスクリプションの進行中は、この手順のステップを続行してインスタンスを設定できます。クレジットカードの詳細に問題がある場合は、アカウントの詳細を更新するように求められます。

> **Note**
>
> AMI でインスタンスを起動するまで、製品の使用料は発生しません。インスタンスタイプを選択する際に、サポートされている各インスタンスタイプの料金を書き留めておいてください。追加の税金が製品に適用される場合があります。

6. [Instance type] (インスタンスタイプ) で、インスタンスのインスタンスタイプを選択します。インスタンスタイプは、起動するインスタンスのハードウェア設定とサイズを定義します。
7. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。
8. [Network settings] (ネットワーク設定) の [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) で、製品のベンダーの仕様に従って作成された新しいセキュリティグループを書き留めます。セキュリティグループには、Linux の SSH (ポート 22) または Windows の RDP (ポート 3389) ですべての IPv4 アドレス (`0.0.0.0/0`) を許可するルールが含まれる場合があります。これらのルールを調整して、特定のアドレスまたはアドレスの範囲のみが、これらのポート経由でインスタンスにアクセスできるようにすることをお勧めします。
9. 画面上の他のフィールドを使用して、インスタンスの設定、ストレージの追加、およびタグの追加を行うことができます。設定できるさまざまなオプションの詳細については、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
10. [Summary] (概要) パネルの [Software Image (AMI)] (ソフトウェアイメージ (AMI)) で、インスタンスを起動しようとしている AMI の詳細を確認します。また、指定した他の設定の詳細も確認します。インスタンスを起動する準備ができたら、[Launch instance] (インスタンスの起動) を選択します。
11. 受信登録した製品によっては、インスタンスの起動には数分以上かかります。ステップ 5 で [インスタンス起動時に購読] を選択した場合は、インスタンスを起動する前に、まず製品をサブスクライブします。クレジットカードの詳細に問題がある場合は、アカウントの詳細を更新するように求められます。起動確認ページが表示されたら、[View all instances] (すべてのインスタンスを表示) を選択して [Instances] (インスタンス) ページに移動します。

> **Note**
>
> インスタンスが `running` 状態である限り、アイドル状態であっても、受信登録費用が発生します。インスタンスが停止している場合でも、ストレージに対して課金されることがあります。

12. インスタンスの状態が [running] の場合、そのインスタンスに接続することができます。これを実行するには、リストでインスタンスを選択し、[Connect] (接続) を選択して、接続オプションを選択します。インスタンスへの接続の詳細については、Linux インスタンスへの接続を参照してください。

> **Important**
>
> インスタンスに接続するには、特定のユーザー名を使用しなければならない場合があるため、ベンダーの使用手順を慎重に確認してください。受信登録の詳細へのアクセスについては、AWS Marketplace サブスクリプションを管理する を参照してください。

13. インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

Old console

launch wizardを使用して AWS Marketplace からインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. Amazon EC2 ダッシュボードから、[インスタンスの作成] を選択します。
3. [Choose an Amazon Machine Image (AMI)] ページで、左の [AWS Marketplace] カテゴリを選択します。カテゴリを参照するか、検索機能を使用して適切な AMI を見つけます。[Select] を選択して製品を選択します。
4. ダイアログに、選択した製品の概要が表示されます。価格情報と、ベンダーが提供したその他の情報を表示できます。準備が完了したら、[Continue] を選択します。

> **Note**
>
> AMI でインスタンスを起動するまで、製品の使用料は発生しません。ウィザードの次のページでは、インスタンスタイプの選択が求められるため、サポートされているインスタンスタイプの料金をメモしておいてください。追加の税金が製品に適用される場合があります。

5. [Choose an Instance Type] ページで、起動するインスタンスのハードウェア設定とサイズを選択します。終了したら、[Next: Configure Instance Details] を選択します。
6. ウィザードの次のページでは、インスタンスの設定、ストレージの追加、およびタグの追加を行うことができます。設定できるさまざまなオプションの詳細については、古いインスタンス起動ウィザードを使用してインスタンスを起動する を参照してください。[Configure Security Group] ページが表示されるまで、[Next] を選択します。

   製品に関するベンダーの仕様にしたがって、新しいセキュリティグループが作成されます。セキュリティグループには、Linux の SSH (ポート 22) または Windows の RDP (ポート 3389) ですべての IPv4 アドレス (`0.0.0.0/0`) を許可するルールが含まれる場合があります。これらのルールを調整して、特定のアドレスまたはアドレスの範囲のみが、これらのポート経由でインスタンスにアクセスできるようにすることをお勧めします。

   準備ができたら、[Review and Launch] を選択します。
7. [Review Instance Launch] ページで、インスタンスを起動しようとしている AMI の詳細と、ウィザードでセットアップするその他の設定の詳細をチェックします。準備ができたら、[Launch] を選択してキーペアを選択または作成し、インスタンスを起動します。
8. 受信登録した製品によっては、インスタンスの起動には数分以上かかります。インスタンスが起動する前に、まず製品に登録されます。クレジットカードの詳細に問題がある場合は、アカウントの詳細を更新するように求められます。起動確認のページが表示されたら、[View Instances] を選択して [Instances] ページに移動します。

> **Note**
>
> インスタンスが実行されている限り、アイドル状態であっても、受信登録費用が発生します。インスタンスが停止している場合でも、ストレージに対して課金されることがあります。

9. インスタンスの状態が [running] の場合、そのインスタンスに接続することができます。そのためには、一覧でインスタンスを選択し、[Connect] (接続) を選択します。ダイアログの指示にしたがいます。インスタンスへの接続の詳細については、Linux インスタンスへの接続を参照してください。

   > **Important**
   >
   > インスタンスにログインするには、特定のユーザー名を使用しなければならない場合があるため、ベンダーの使用手順を慎重に確認してください。受信登録の詳細へのアクセスについては、AWS Marketplace サブスクリプションを管理する を参照してください。

10. インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

API と CLI を使用した AWS Marketplace AMI インスタンスの起動

API またはコマンドラインツールを使用して、AWS Marketplace 製品からインスタンスを起動するには、まず製品に登録していることを確認します。次の方法を使用して、製品の AMI ID でインスタンスを起動できます。

方法	ドキュメント
AWS CLI	run-instances コマンドを使用するか、詳細について「インスタンスの起動」を参照してください。
AWS Tools for Windows PowerShell	New-EC2Instance コマンドを使用するか、詳細について Windows PowerShell を使用した Amazon EC2 インスタンスの起動を参照してください。
クエリ API	RunInstancesリクエストを使用します。

インスタンスの停止と起動

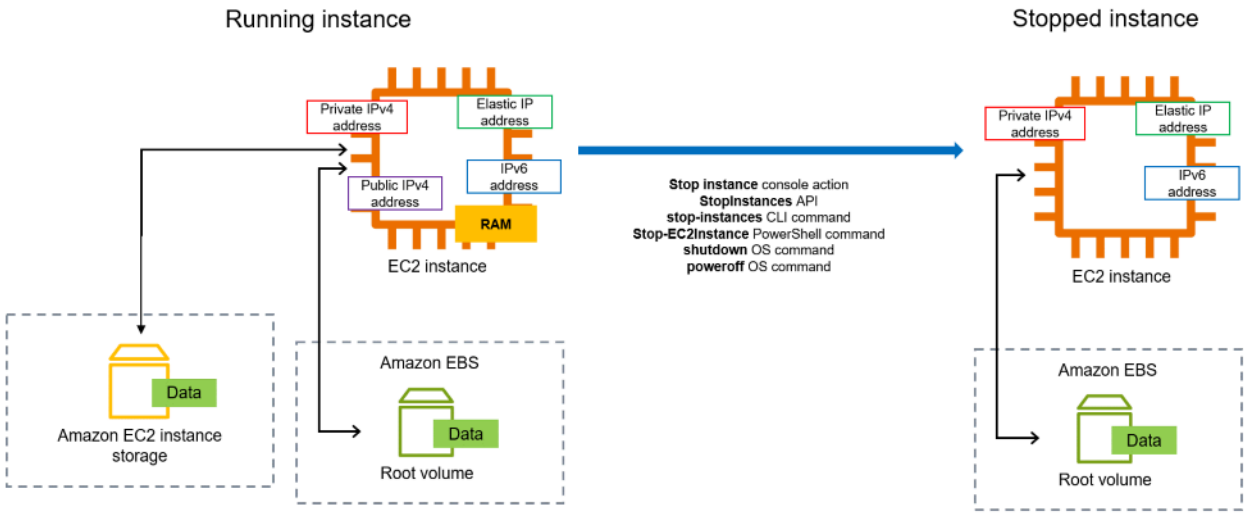
インスタンスにルートデバイスとして Amazon EBS ボリュームがある場合、そのインスタンスを停止して起動できます。インスタンスにはそのインスタンス ID が保持されますが、「停止したインスタンスの変更」セクションで述べられているように変更される可能性があります。インスタンスを停

止すると、インスタンスはシャットダウンします。インスタンスを起動すると、通常、インスタンスは基盤となる新しいホストコンピュータに移行され、新しいパブリック IPv4 アドレスが割り当てられます。

インスタンスを停止しても、そのインスタンスは削除されません。インスタンスが必要なくなったら、終了することができます。詳細については、「インスタンスの終了」を参照してください。インスタンスを休止状態にしてインスタンスメモリ (RAM) の内容を保存する場合は、を参照してください。Amazon EC2 インスタンスの休止インスタンスライフサイクルアクションの違いについては、を参照してください。再起動、停止、休止、削除の違い

AWS は、再起動、停止/開始、またはリタイアなど、インスタンスのイベントを予定できます。管理対象イベントの種類とAWS、予定されているイベントに関する通知を表示および受信する方法については、を参照してくださいインスタンスの予定されたイベント。

次の図は、Amazon EC2 インスタンスを停止したときに失われるものと残るものを示しています。インスタンスが停止すると、アタッチされたインスタンスストアボリュームと、それらのボリュームに保存されているデータ、インスタンス RAM に保存されているデータ、およびインスタンスに Elastic IP アドレスが関連付けられていない場合は割り当てられたパブリック IPv4 アドレスがすべて失われます。インスタンスには、割り当てられたプライベート IPv4 アドレス、インスタンスに関連付けられた Elastic IP アドレス、すべての IPv6 アドレス、アタッチされている Amazon EBS ボリューム、およびそれらのボリューム上のデータが保持されます。



トピック

- インスタンスの起動と停止に関連するコスト
- 実行中および停止中のインスタンスをすべて検索
- インスタンスを停止するための前提条件

- インスタンスを手動で停止して起動する
- インスタンスを自動的に停止して起動する
- インスタンスを停止するとどうなるか
- インスタンスを起動するとどうなるか
- 停止したインスタンスの変更
- 停止保護を有効にします
- アプリケーションの応答をテストして停止して起動する
- インスタンスの停止に関するトラブルシューティング

インスタンスの起動と停止に関連するコスト

インスタンスの停止と起動には以下のコストがかかります。

停止 — インスタンスの状態が `shutting-down` または `terminated` に変わると、そのインスタンスの料金は発生しなくなります。停止したインスタンスの使用料やデータ転送料は請求されません。Amazon EBS ストレージボリュームの保存には料金がかかります。

Starting — 停止したインスタンスを再起動するたびに、1 分間分の最低料金が課金されます。1 分経過した後は、使用した秒数のみ課金されます。例えば、インスタンスを 20 秒間実行して停止した場合は、1 分間分課金されます。インスタンスを 3 分 40 秒実行した場合は、ちょうど 3 分 40 秒間分課金されます。

実行中および停止中のインスタンスをすべて検索

Amazon EC2 グローバルビューでは、すべてにわたって、実行中と停止中のすべての AWS リージョン のインスタンスを 1 つのページで確認できます。これは、インベントリを取得し、忘れられたインスタンスを見つけるのに特に有用です。グローバルビューを使用する方法については、「Amazon EC2 Global View」を参照してください。

インスタンスを停止するための前提条件

停止できるのは Amazon EBS-Backed インスタンスだけです。インスタンスストアボリュームにバッキングされたインスタンスは、[停止] アクションをサポートしません。2 つのボリュームタイプの違いをよりよく理解するには、を参照してくださいAmazon EC2 インスタンスのストレージオプション。

インスタンスのルートデバイスタイプを確認するには、Amazon EC2 コンソールまたはを使用できます。AWS CLI

Amazon EC2 console

Amazon EC2 コンソールで [インスタンス] ペインを開き、インスタンスを選択します。[ルートデバイスタイプ] は、[ストレージ] タブの [ルートデバイスの詳細] に表示されます。

AWS CLI

インスタンスのルートデバイスタイプを確認するには、`describe-instances` AWS CLI コマンドを実行して `RootDeviceType` の出力を確認します: `ebs` または `instance-store`。詳細については、「AWS CLI コマンドリファレンス」の「describe-instances」を参照してください。

インスタンスを手動で停止して起動する

コンソールまたはコマンドラインを使用して、Amazon EBS-Backed インスタンスを起動および停止できます。

> **Warning**
>
> インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスを停止する前に、必要なデータをインスタンスストアボリュームから永続的ストレージ (Amazon EBS や Amazon S3 など) にコピーしていることを確認します。インスタンスストアボリュームにバッキングされたインスタンスは、[停止] アクションをサポートしません。

Console

Amazon EBS-Backed インスタンスを停止および起動するには

1. AWS Management Console にサインインし、Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. [Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。このオプションが無効になっている場合は、インスタンスが既に停止しているか、またはルートボリュームがインスタンスストアボリュームです。
4. 確認を求められたら、[Stop] を選択します。インスタンスが停止するまで、数分かかる場合があります。
5. (オプション) インスタンスが停止されている間、特定のインスタンス属性を変更できます。詳細については、「停止したインスタンスの変更」を参照してください。

6. 停止されているインスタンスを開始するには、インスタンスを選択後、[インスタンスの状態]、[インスタンスの開始] の順に選択します。
7. インスタンスが `running` 状態になるまで、数分かかる場合があります。

Command line

Amazon EBS-Backed インスタンスを停止および起動するには

以下のいずれかのコマンドを実行します:

- AWS CLI—stop-instances および start-instances。
- AWS Tools for PowerShell—Stop-EC2Instance および Start-EC2Instance。
- OS コマンド: shutdown または poweroff コマンドを使用してシャットダウンを開始できます。OS コマンドを使用すると、インスタンスはデフォルトで停止します。この動作を変更して、インスタンスの停止ではなく終了させることができます。詳細については、「インスタンスによって起動されたシャットダウン動作の変更」を参照してください。

インスタンスから halt コマンドを使用しても、シャットダウンは開始されません。haltコマンドを使用すると、インスタンスは終了せず、代わりに CPU をに配置して HLT CPU 操作を一時停止します。インスタンスは実行状態のままです。

インスタンスを自動的に停止して起動する

次のサービスを使用して、インスタンスの停止と起動を自動化できます。

AWS でインスタンススケジューラを使用する

インスタンススケジューラを AWS で使用して、EC2 インスタンスの開始と停止を自動化することができます。詳細については、「CloudFormation で Instance Scheduler を使用して EC2 インスタンスをスケジュールするにはどうすればよいですか?」を参照してください。追加料金が適用されることに注意してください。

AWS Lambda および Amazon EventBridge ルールを使用する

Lambda と EventBridge ルールを使用して、スケジュール上のインスタンスを停止および開始することができます。詳細については、「Lambda を使用して、Amazon EC2 インスタンスを一定の間隔で停止および起動するにはどうすればよいですか?」を参照してください。

Amazon EC2 Auto Scaling

アプリケーションの負荷を処理できる Amazon EC2 インスタンスの数が適切であることを確認するには、Auto Scaling グループを作成します。 Amazon EC2 Auto Scalingアプリケーションが常にトラフィック需要を処理する適切な容量を確保し、必要な場合にのみインスタンスを起動することでコストを節約できます。不要なインスタンスを停止するのではなく、Amazon EC2 Auto Scaling終了させることに注意してください。自動スケーリンググループを設定するには、「Amazon EC2 Auto Scalingをはじめる」を参照してください。

インスタンスを停止するとどうなるか

インスタンスを停止すると、変更はインスタンスの OS レベルで登録され、一部のシステムリソースは失われ、一部は存続します。

インスタンスを停止すると、OSレベルで以下のように登録されます：

- API リクエストは、ボタンのクリックイベントをゲストに送信します。
- ボタンのクリックイベントの結果として、さまざまなシステムサービスが停止します。適切なシャットダウンは、ハイパーバイザーからの ACPI シャットダウンボタンのクリックイベントによってトリガーされます。
- ACPI シャットダウンが開始されます。
- このインスタンスは、適切なシャットダウンプロセスが終了したときにシャットダウンされます。設定可能な OS シャットダウン時間はありません。
- インスタンス OS が数分以内に正常にシャットダウンされない場合は、ハードシャットダウンが実行されます。
- インスタンスが実行を停止します。
- インスタンスのステータスが `stopping` になり、その後 `stopped` になります。
- [自動スケーリング] インスタンスがAuto Scalingグループにある場合、インスタンスが `running` 以外のAmazon EC 2状態にある場合、またはステータスチェックのステータスが `impaired` になった場合、Amazon EC2 Auto Scaling はインスタンスを異常と見なして置き換えます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インスタンスのヘルスチェック」を参照してください。

インスタンスを停止すると、次のものが失われます。

- RAM に保存されているデータ。

- インスタンスストアボリュームに保存されているデータは失われます。
- 起動時または開始時に Amazon EC2 がインスタンスに自動的に割り当てられたパブリック IPv4 アドレス。パブリック IPv4 アドレスに変更を加えないようにするには、インスタンスに Elastic IP アドレスを関連付けます。

インスタンスを停止しても、次のものは保持されます。

- アタッチされた Amazon EBS ボリューム。
- Amazon EBS ボリュームに保存されているデータ。
- プライベート IPv4 アドレス。
- IPv6 アドレス。
- インスタンスに関連付けられた Elastic IP アドレス。インスタンスが停止すると、関連する Elastic IP アドレスに対する課金が始まります。

Mac インスタンスを停止するとどうなるかについては、「Mac インスタンスを停止して終了する」を参照してください。

インスタンスを起動するとどうなるか

インスタンスを起動すると、変更はインスタンスレベルで登録されます。

EC2 インスタンスを起動すると、次のことが起こります。

- ほとんどの場合、基盤となる新しいホストコンピュータにインスタンスが移行します (ただし、専有ホスト設定でインスタンスがホストに割り当てられた場合などは、現在のホストにインスタンスが残ります)。
- パブリック IPv4 アドレスを受信するようにインスタンスが設定されている場合、Amazon EC2 は新しいパブリック IPv4 アドレスをインスタンスに割り当てます。パブリック IPv4 アドレスに変更を加えないようにするには、インスタンスに Elastic IP アドレスを関連付けます。

停止したインスタンスの変更

インスタンスが停止している間、他のボリュームと同様にそのルートボリュームを扱い、変更することができます (ファイルシステムの問題を修復したり、ソフトウェアを更新したりするなど)。

以下のインスタンスの属性は停止されると、変更できます。

- インスタンスタイプ
- ユーザーデータ
- Kernel
- RAM ディスク

インスタンスの実行中にこれらの属性を変更しようとすると、Amazon EC2 が `IncorrectInstanceState` エラーを返します。

Amazon EC2 コンソールまたは AWS CLI を使用して、停止したインスタンスの次の属性を変更できます:

- インスタンスタイプ
- ユーザーデータ
- EBS 最適化

Amazon EC2 コンソールを使用した以下の属性の変更はサポートされていません。

- `DeleteOnTermination`
- Kernel
- RAM ディスク

インスタンス属性を変更

インスタンス属性は Amazon EC2 コンソールまたはコマンドラインを使用して更新できます。

Console

AWS Management Console で以下を変更するには	次のリソースを参照してください。
インスタンスタイプ	インスタンスタイプを変更する
ユーザーデータ	ユーザーデータおよびコンソール

AWS Management Console で以下を変更するには	次のリソースを参照してください。
EBS 最適化	EBS 最適化を有効にする
`DeleteOnTermination` ルートボリュームの属性	実行中のインスタンスのブロックデバイスマッピングの更新 。この属性を変更するためにインスタンスを停止する必要はありません。

Command line

コマンドラインを使用してインスタンス属性を変更するには

次のコマンドを使用してインスタンス属性を変更できます。これらのコマンドラインインターフェイスの詳細については、「Amazon EC2 へのアクセス」を参照してください。

- modify-instance-attribute (AWS CLI)
- Edit-EC2InstanceAttribute (AWS Tools for PowerShell)

インスタンスのルートボリュームを変更

次の手順を実行すると、インスタンスのルートボリュームを変更できます。

1. ボリュームをストップどインスタンスからデタッチします。
2. 実行中のインスタンスに EBS ボリュームを接続する
3. ボリュームタイプの変更
4. ボリュームを実行インスタンスからデタッチします。
5. 停止したインスタンスにボリュームを再接続します。

インスタンスのブロックデバイスマッピングにルートデバイスとして指定されたストレージデバイス名を使用して、ボリュームを接続解除していることを確認します。ブロックデバイスマッピングの指定の詳細については、「ブロックデバイスマッピング」を参照してください。

停止保護を有効にします

インスタンスが誤って停止するのを防ぐために、インスタンスに対する停止保護を有効にすることができます。停止保護は、インスタンスを偶発的な終了からも保護します。

Amazon EC2 `ModifyInstanceAttribute` API の `DisableApiStop` 属性は、Amazon EC2 コンソール、AWS CLI、Amazon EC2 API を使用してインスタンスを停止できるかどうかを制御します。この属性の値は、インスタンスの起動時、インスタンスの実行中、またはインスタンスの停止時に設定できます。

制限事項

- 停止保護を有効にしても、shutdown や poweroff などのオペレーティングシステムコマンドによりインスタンスからシャットダウンを開始してインスタンスを誤って停止することは、防げません。
- 停止保護を有効にしても、インスタンスにインスタンスを停止する予定されたイベントがある場合、AWS がインスタンスを停止するのを防ぐことはできません。
- 停止保護を有効にしても、インスタンスが異常な場合やスケールインイベント中にAmazon EC2 Auto Scaling がインスタンスを終了するのを防ぐことはできません。スケールイン時に Auto Scaling グループが特定のインスタンスを終了できるかどうかを制御するには、インスタンスのスケールイン保護を使用します。
- 停止保護は、インスタンスが誤って停止するのを防ぐだけでなく、コンソール、AWS CLI、または API を使用して誤って終了するのを防ぎます。ただし、`DisableApiTermination` 属性は自動的に変更されません。`DisableApiStop` 属性が `false` に設定されている場合、`DisableApiTermination` 属性の設定によって、コンソール、AWS CLI 、またはAPIを使用してインスタンスを終了できるかどうかが決まります。詳細については、「インスタンスの終了」を参照してください。
- インスタンスストアでバックアップされたインスタンスの停止保護は有効にできません。
- スポットインスタンスの停止保護は有効にできません。
- 停止保護を有効または無効にすると、Amazon EC2 API は最終的な整合性モデルに従います。つまり、停止保護属性を設定するコマンドを実行した結果が、それ以降に実行するすべてのコマンドにすぐには表示されない場合があります。詳細については、「Amazon EC2 API Referenc」(Amazon EC2 API レファレンス) の「Eventual consistency」(結果整合性) を参照してください。

停止保護タスク

- 起動時にインスタンスに対する停止保護を有効にします
- 実行中または停止したインスタンスに対する停止保護を有効にします

- 実行中または停止したインスタンスに対する停止保護を無効にします

起動時にインスタンスに対する停止保護を有効にします

次のいずれかの方法を使用して、インスタンスを起動するときにインスタンスに対する停止保護を有効にできます。

Console

起動時にインスタンスに対する停止保護を有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ダッシュボードで、[Launch Instance (インスタンスの起動)] を選択します。
3. [new launch instance wizard] (新しいインスタンス起動ウィザード) でインスタンスを設定します。
4. ウィザードで、[高度な詳細] の [保護停止] で [有効にする] を選択して、保護を停止します。

AWS CLI

起動時にインスタンスに対する停止保護を有効にするには

run-instancesAWS CLIコマンドを使用して、インスタンスを起動し、disable-api-stopパラメータを指定します。

```
aws ec2 run-instances \
    --image-id ami-a1b2c3d4e5example \
    --instance-type t3.micro \
    --key-name MyKeyPair \
    --disable-api-stop \
    ...
```

実行中または停止したインスタンスに対する停止保護を有効にします

次のいずれかの方法を使用して、インスタンスが実行中または停止したときにインスタンスに対する停止保護を有効にできます。

Console

実行中または停止中のインスタンスの停止保護を有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックします。
3. インスタンスを選択してから、[アクション] > [インスタンスの設定] > [保護停止を変更する] を選択します。
4. [Enable] (有効化) チェックボックスを選択し、[Save] (保存) を選択します。

AWS CLI

実行中または停止中のインスタンスの停止保護を有効にするには

modify-instance-attributeAWS CLIコマンドを使用して、`disable-api-stop`パラメターを指定します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --disable-api-stop
```

実行中または停止したインスタンスに対する停止保護を無効にします

次のいずれかの方法を使用して、実行中または停止したインスタンスに対する停止保護を無効にすることができます。

Console

実行中または停止中のインスタンスの停止保護を無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックします。
3. インスタンスを選択してから、[Actions] (アクション)、[Instance Settings] (インスタンスの設定)、[Change Stop Protection] (停止保護の変更) を選択します。
4. [Enable] (有効化) チェックボックスをオフにして、[Save] (保存) を選択します。

AWS CLI

実行中または停止中のインスタンスの停止保護を無効にするには

modify-instance-attributeAWS CLIコマンドを使用して、no-disable-api-stopパラメーターを指定します。

```
aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --no-disable-api-stop
```

アプリケーションの応答をテストして停止して起動する

インスタンスが停止後に起動された場合のアプリケーションの応答をテストするには、AWS Fault Injection Service を使用します。詳細については、AWS Fault Injection Service ユーザーガイドを参照してください。

インスタンスの停止に関するトラブルシューティング

Amazon EBS-Backed インスタンスを停止し、stopping 状態に 「stuck」 が表示されている場合、インスタンスを強制終了できます。詳細については、「インスタンスの停止に関するトラブルシューティング」を参照してください。

Amazon EC2 インスタンスの休止

インスタンスを休止すると、Amazon EC2 によってオペレーティングシステムに休止の実行 (suspend-to-disk) が指示されます。休止状態に入ると、インスタンスメモリ (RAM) に置かれていた内容が、Amazon Elastic Block Store (Amazon EBS) のルートボリュームに保存されます。インスタンスの EBS ルートボリュームとアタッチされた EBS データボリュームは、Amazon EC2 により保持されます。インスタンスが起動したとき、

- EBS ルートボリュームは前の状態に復元されます。
- RAM の内容が再ロードされます。
- インスタンスで以前に実行されていたプロセスが再開されます。
- 以前にアタッチされていたデータボリュームが再アタッチされ、インスタンスがそのインスタンス ID を保持します。

インスタンスは、休止が有効になっており、休止の前提条件を満たしている場合のみ、休止状態にすることができます。

インスタンスまたはアプリケーションが、ブートストラップし、メモリフットプリントを構築して完全に生産性を発揮するのに時間がかかる場合は、休止を使用してインスタンスを事前ウォーミングできます。インスタンスを事前ウォーミングするには、次の操作を行います。

1. 休止を有効にしてインスタンスを起動します。
2. インスタンスを必要な状態に移行させます。
3. 休止状態にして、必要なときにいつでも望ましい状態に回復されるようにします。

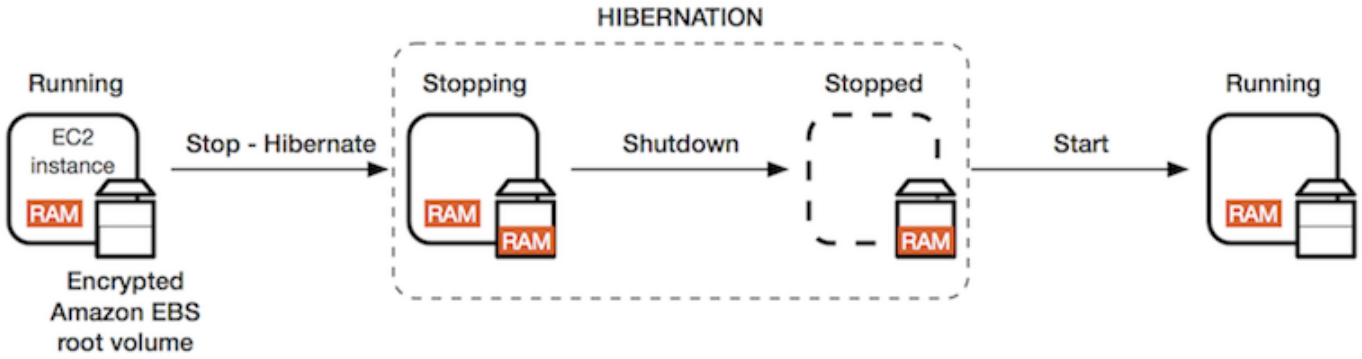
インスタンスが stopped 状態にある場合の休止状態のインスタンスにも、RAM の内容が EBS ルートボリュームに転送される場合のデータ転送にも、課金はされません。EBS ボリュームのストレージに対しては、RAM の内容のストレージも含めて、料金が発生します。

インスタンスが必要なくなった場合、stopped (休止) 状態にある場合を含め、いつでも終了することができます。詳細については、「インスタンスの終了」を参照してください。

コンテンツ

Amazon EC2 インスタンスの休止の仕組み

次の図は、EC2 インスタンスの休止処理の基本的な概要を示しています。



インスタンスを休止するとどうなるか

インスタンスを休止すると、次の処理が実行されます。

- インスタンスはstopping状態に移行します。Amazon EC2 が、オペレーティングシステムに対して休止処理 (suspend-to-disk) を指示します。休止に伴ってすべてのプロセスがフリーズされ、RAM の内容が EBS ルートボリュームに保存されます。その後に、通常のシャットダウンが実行されます。
- シャットダウンプロセスが完了した後、インスタンスは stopped 状態に移行します。
- EBS ボリュームはインスタンスにアタッチされたままとなり、保存された RAM の内容も含めて、データは保持されます。
- Amazon EC2 インスタンスストアボリュームはインスタンスにアタッチされたままになりますが、インスタンスストアボリューム上のデータは失われます。
- インスタンスが stopped 状態の間、インスタンスタイプやサイズなど、インスタンスの特定の属性を変更できます。
- 殆どの場合、インスタンスは基盤となる新しいホストコンピュータが起動したときに移行されます。これは、インスタンスを停止して起動した場合と同じです。
- インスタンスを起動すると、インスタンスのブートアッププロセスが実行され、オペレーティングシステムが EBS ルートボリュームから RAM の内容を読み取ります。次に、プロセスのフリーズが解除されて以前の状態が回復されます。
- インスタンスのプライベート IPv4 アドレスとすべての IPv6 アドレスは保持されます。インスタンスを起動すると、インスタンスは引き続きプライベート IPv4 アドレスとすべての IPv6 アドレスを保持します。
- Amazon EC2 はパブリック IPv4 アドレスをリリースします。インスタンスを起動すると、Amazon EC2 は新しいパブリック IPv4 アドレスをインスタンスに割り当てます。

- インスタンスには関連付けられた Elastic IP アドレスが保持されます。休止状態のインスタンスに関連付けられた Elastic IP アドレスに対して課金されます。

休止と再起動、停止、および終了の違いについては、「再起動、停止、休止、削除の違い」を参照してください。

制限事項

- インスタンスを休止すると、インスタンスストアボリューム上のデータは失われます。
- (Linux インスタンス) RAM が 150 GB を超える Linux インスタンスを休止することはできません。
- (Windows インスタンス) RAM が 16 GB を超える Windows インスタンスを休止することはできません。
- 休止状態になっている、または休止機能が有効になっているインスタンスからスナップショットまたは AMI を作成した場合、その AMI (あるいは、そのスナップショットから作成した AMI) から起動した新しいインスタンスに接続できないことがあります。
- (スポットインスタンスのみ) Amazon EC2 がスポットインスタンスを休止した場合、インスタンスを再開できるのは Amazon EC2 のみです。スポットインスタンスを休止状態 (ユーザー起動の休止) にする場合、ユーザーはインスタンスを再開できます。休止したスポットインスタンスは、容量が空いていて、スポット料金が指定した上限料金以下である場合、再開できます。
- Auto Scaling グループ内のインスタンス、または Amazon ECS が使用しているインスタンスを休止することはできません。インスタンスが Auto Scaling グループにあり、そのインスタンスを休止しようとしている場合、Amazon EC2 Auto Scaling サービスは停止したインスタンスを異常と判断し、そのインスタンスを終了して代わりのインスタンスを起動する場合があります。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インスタンスのヘルスチェック」を参照してください。
- UEFI Secure Boot を有効にした状態で、UEFI モードで起動するように設定されたインスタンスを休止することはできません。
- キャパシティーの予約では、キャパシティーの予約 で起動されたインスタンスを休止状態にする場合、そのインスタンスを再開しても、休止した時点の状態が維持されることを保証していません。
- 連邦情報処理標準 (FIPS) モードが有効になっている場合、5.10 未満のカーネルを使用するインスタンスを休止状態にすることはできません。
- 60 日間以上にわたる休止はサポートしていません。60 日より長くインスタンスを保持するには、休止したインスタンスを起動し、停止して、また起動する必要があります。

- 当社では、継続的にプラットフォームをアップグレードやセキュリティパッチで更新しており、休止されている既存のインスタンスと競合する可能性があります。シャットダウンまたは再起動を実行して必要なアップグレードとセキュリティパッチを適用できるように、休止されているインスタンスの起動が必要になる重要な更新については、通知を受け取ります。

スポットインスタンスを休止する場合の注意点

- ユーザーがスポットインスタンスを休止した場合、容量が空いていて、スポット料金が、指定した上限料金以下である場合、ユーザーがこれを再開できます。
- Amazon EC2 がスポットインスタンスを休止した場合は、
  - インスタンスを再開できるのは Amazon EC2 だけです。
  - Amazon EC2 は、容量が利用可能になり、スポット料金が、指定した上限料金以下である場合、休止したスポットインスタンスを再開します。
  - Amazon EC2 がスポットインスタンスを休止するときは、休止が始まる 2 分前にユーザーに中断通知が届きます。

  詳細については、「スポットインスタンスの中断。」を参照してください。
- スポットインスタンスの休止を有効にする方法はいくつかあります。詳細については、「中断動作の指定」を参照してください。

Amazon EC2 インスタンスの休止の前提条件

オンデマンドインスタンスまたはスポットインスタンスの休止のサポートは、起動時に有効にすることができます。実行中または停止状態の既存のインスタンスで休止を有効にすることはできません。詳細については、「インスタンスの休止の有効化」を参照してください。

インスタンスを休止するための要件

- AWS リージョン
- AMI
- インスタンスファミリー
- インスタンスの RAM サイズ
- ルートボリュームタイプ
- ルートボリュームサイズ
- ルートボリュームの暗号化
- EBS ボリュームタイプ

- スポットインスタンスリクエスト

AWS リージョン

すべての AWS リージョンのインスタンスで休止を使用できます。

AMI

休止をサポートする HVM AMI を使用する必要があります。

サポートされている AMI

- Linux AMI
- Windows AMI

Linux AMI

AMI	Xen -supported instance families only	Nitro-supported instance families only
AL2023 AMI (2023 年 9 月 20 日以降にリリース)	サポート	サポート
Amazon Linux 2 AMI (2019 年 8 月 29 日以降にリリース)	サポート	サポート
Amazon Linux AMI 2018.03 (2018 年 11 月 16 日以降にリリース)	サポート	サポート
CentOS バージョン 8 AMI [1] (追加設定が必要です)	サポート外	サポート
Fedora バージョン 34 以降の AMI [1] (追加設定が必要です)	サポート外	サポート
Red Hat Enterprise Linux (RHEL) 9 AMI [1] (追加設定が必要です)	サポート外	サポート

AMI	Xen -supported instance families only	Nitro-supported instance families only
Red Hat Enterprise Linux (RHEL) 8 AMI [1] (追加設定が必要です)	サポート外	サポート
20230303 以降のシリアル番号でリリースされた Ubuntu 22.04.2 LTS (Jammy Jellyfish) AMI [2]	サポート	サポート
20210820 以降のシリアル番号でリリースされた Ubuntu 20.04 LTS (Focal Fossa) AMI [2]	サポート	サポート
20190722.1 以降のシリアル番号でリリースされた Ubuntu 18.04 LTS (Bionic Beaver) AMI [2] [4]	サポート	サポート
Ubuntu 16.04 LTS (Xenial Xerus) AMI [2] [3] [4] (追加設定が必要です)	サポート	サポート

[1] CentOS、Fedora、および Red Hat Enterprise Linux の場合、休止状態は Nitro ベースのインスタンスでのみサポートされます。

[2] Ubuntu 22.04.2 LTS (Jammy Jellyfish)、Ubuntu 20.04 LTS (Focal Fossa)、Ubuntu 18.04 LTS (Bionic Beaver)、Ubuntu 16.04 LTS (Xenial Xerus) を使用するインスタンスでは、KASLR を無効にすることをお勧めします。詳細については、インスタンスでの KASLR の無効化 (Ubuntu のみ) をご参照ください。

[3] Ubuntu 16.04 LTS (Xenial Xerus) AMI の場合、ハイバネーションは `t3.nano` インスタンスタイプではサポートされません。Ubuntu (Xenial Xerus) が 2021 年 4 月にサポートを終了したため、パッチは利用できません。`t3.nano` インスタンスタイプを使用したい場合は、Ubuntu 22.04.2 LTS

(Jammy Jellyfish)、Ubuntu 20.04 LTS (Focal Fossa) AMI または Ubuntu 18.04 LTS (Bionic Beaver) AMI にアップグレードすることをお勧めします。

[4] Ubuntu 18.04 LTS (Bionic Beaver) および Ubuntu 16.04 LTS (Xenial Xerus) のサポートは終了しました。

独自の AMI が休止をサポートするように設定するには、「Linux AMI で休止がサポートされるように設定する」を参照してください。

他のバージョンの Ubuntu および他のオペレーティングシステムはまもなくサポートされる予定です。

Windows AMI

- Windows Server 2022 AMI (2023 年 9 月 13 日以降にリリース)
- Windows Server 2019 AMI (2019 年 9 月 11 日以降にリリース)
- Windows Server 2016 AMI (2019 年 9 月 11 日以降にリリース)
- Windows Server 2012 R2 AMI (2019 年 9 月 11 日以降にリリース)
- Windows Server 2012 AMI (2019 年 9 月 11 日以降にリリース)

インスタンスファミリー

休止をサポートするインスタンスファミリーを使用する必要があります。

- 汎用: M3、M4、M5、M5a、M5ad、M5d、M6i、M6id、M7i、M7i-flex、T2、T3、T3a
- コンピューティング最適化: C3、C4、C5、C5d、C6i、C6id、C7a、C7i
- メモリ最適化: R3、R4、R5、R5a、R5ad、R5d、R7a、R7i、R7iz
- ストレージの最適化: I3、I3en

Nitro インスタンス – ベアメタルインスタンスはサポートされていません。

特定のリージョンで休止状態をサポートする利用可能なインスタンスタイプを表示するには

利用可能なインスタンスタイプは、リージョンごとに異なります。リージョンで 休止状態 をサポートしている利用可能なインスタンスタイプを確認するには、describe-instance-types コマンドを `--region` パラメータとともに使用します。結果の範囲を、休止状態をサポートするインスタンスタイプに設定するために `--filters` パラメータをインクルードし、出力の範囲を `InstanceType` の値に設定するために `--query` パラメータをインクルードします。

```
aws ec2 describe-instance-types --filters Name=hibernation-supported,Values=true --
query "InstanceTypes[*].[InstanceType]" --output text | sort
```

出力例

```
c3.2xlarge
c3.4xlarge
c3.8xlarge
c3.large
c3.xlarge
c4.2xlarge
c4.4xlarge
c4.8xlarge
...
```

インスタンスの RAM サイズ

Linux インスタンス – 150 GB 未満である必要があります。

Windows インスタンス – 最大 16 GB です。T3 または T3a インスタンスの休止には、最低 1 GB の RAM をお勧めします。

ルートボリュームタイプ

ルートボリュームは、インスタンスストアボリュームではなく EBS ボリュームにする必要があります。

ルートボリュームサイズ

ルートボリュームは、RAM の内容を保存し、OS やアプリケーションなどの予想される使用量に対応できる大きさにする必要があります。休止を有効にすると、RAM を保存するために起動時にルートボリュームでスペースが割り当てられます。

ルートボリュームの暗号化

休止時にメモリ内にある機密性の高いコンテンツを保護するためにルートボリュームを暗号化する必要があります。RAM データを EBS ルートボリュームに移動する場合は、常に暗号化します。ルートボリュームの暗号化は、インスタンスの起動時に適用されます。

ルートボリュームが暗号化された EBS ボリュームであることを確認するには、次の 3 つのオプションのいずれかを使用します。

- デフォルトでの EBS 暗号化: EBS 暗号化をデフォルトで有効にして、AWS アカウントで作成されたすべての新しい EBS ボリュームを暗号化できます。この方法では、インスタンスの起動時に暗号化のインテントを指定することなく、インスタンスの休止を有効にすることができます。詳細については、「デフォルトの暗号化」を参照してください。
- EBS の「シングルステップ」暗号化: 暗号化されていない AMI から暗号化された EBS-Backed EC2 インスタンスを起動し、同時に休止状態を有効にすることができます。詳細については、EBS-backed AMI での暗号化の利用 を参照してください。
- 暗号化された AMI: 暗号化された AMI を使用してインスタンスを起動することで、EBS 暗号化を有効にすることができます。暗号化されたルートスナップショットが AMI にない場合は、それを新しい AMI にコピーして暗号化をリクエストできます。詳細については、「コピー時に暗号化されていないイメージを暗号化する」および「AMI のコピー」を参照してください。

EBS ボリュームタイプ

EBS ボリュームは、次のいずれかの EBS ボリュームタイプを使用する必要があります。

- 汎用 SSD (gp2 および gp3)
- プロビジョンド IOPS SSD (io1 および io2)

Provisioned IOPS SSD ボリュームタイプを選択した場合、休止状態の最適なパフォーマンスを実現するには、適切な IOPS で EBS ボリュームをプロビジョニングする必要があります。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの種類」を参照してください。

スポットインスタンスリクエスト

スポットインスタンスには、次の要件が適用されます。

- スポットインスタンスのリクエストのタイプは `persistent` である必要があります。
- スポットインスタンスリクエストで起動グループを指定することはできません。

Linux AMI で休止がサポートされるように設定する

次の Linux AMI は休止状態をサポートしていますが、これらの AMI のいずれかを使用して起動されたインスタンスを休止状態にするには、追加の設定が必要です。

その他の設定も必要です。

- Amazon Linux 2 minimal AMI (2019 年 8 月 29 日以降にリリース)

- 2019 年 8 月 29 日以前にリリースされた Amazon Linux 2
- 2018 年 11 月 16 日以前にリリースされた Amazon Linux
- CentOS バージョン 8 以降
- Fedora バージョン 34 以降
- Red Hat Enterprise Linux バージョン 8 または 9 以降
- シリアル番号 20210820 よりも前にリリースされた Ubuntu 20.04 LTS (Focal Fossa)
- シリアル番号 20190722.1 よりも前にリリースされた Ubuntu 18.04 (Bionic Beaver)
- Ubuntu 16.04 (Xenial Xerus)

詳細については、「Amazon Linux インスタンスでのインスタンスソフトウェアの更新」を参照してください。

次の AMI は、すでに休止状態をサポートするように設定されているため、追加の設定は必要ありません。

- AL2023 AMI (2023 年 9 月 20 日以降にリリース)
- Amazon Linux 2 full AMI (2019 年 8 月 29 日以降にリリース)
- Amazon Linux AMI 2018.03 (2018 年 11 月 16 日以降にリリース)
- 20230303 以降のシリアル番号でリリースされた Ubuntu 22.04.2 LTS (Jammy Jellyfish) AMI
- 20210820 以降のシリアル番号でリリースされた Ubuntu 20.04 LTS (Focal Fossa) AMI
- 20190722.1 以降のシリアル番号でリリースされた Ubuntu 18.04 LTS (Bionic Beaver) AMI

Amazon Linux 2 minimal AMI (2019 年 8 月 29 日以降にリリース)

2019 年 8 月 29 日以降にリリースされた Amazon Linux 2 minimal AMI で休止がサポートされるように設定するには

1. `ec2-hibinit-agent`パッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

2. サービスを再起動します。

```
[ec2-user ~]$ sudo systemctl start hibinit-agent
```

2019 年 8 月 29 日以前にリリースされた Amazon Linux 2

2019 年 8 月 29 日以前にリリースされた Amazon Linux 2 AMI で休止がサポートされるように設定するには

1. 4.14.138-114.102以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが 4.14.138-114.102 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

5. インスタンスを停止し、AMI を作成します。詳細については、「インスタンスからの Linux AMI の作成」を参照してください。

2018 年 11 月 16 日以前にリリースされた Amazon Linux

2018 年 11 月 16 日以前にリリースされた Amazon Linux AMI で休止がサポートされるように設定するには

1. 4.14.77-70.59以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが 4.14.77-70.59 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

5. インスタンスを停止し、AMI を作成します。詳細については、「インスタンスからの Linux AMI の作成」を参照してください。

CentOS バージョン 8 以降

休止状態をサポートするように CentOS バージョン 8 以降の AMI を設定するには

1. 4.18.0-305.7.1.el8_4.x86_64以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. このステップでは、Fedora Extra Packages for Enterprise Linux (EPEL) リポジトリをインストールします。

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

3. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

4. 起動時に休止状態エージェントを起動できるようにします。

```
[ec2-user ~]$ sudo systemctl enable hibinit-agent.service
```

5. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

6. 次のコマンドを実行して、カーネルバージョンが 4.18.0-305.7.1.el8_4.x86_64 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Fedora バージョン 34 以降

休止状態をサポートするために Fedora バージョン 34 以降の AMI を設定するには

1. 5.12.10-300.fc34.x86_64以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo dnf install ec2-hibinit-agent
```

3. 起動時に休止状態エージェントを起動できるようにします。

```
[ec2-user ~]$ sudo systemctl enable hibinit-agent.service
```

4. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

5. 次のコマンドを実行して、カーネルバージョンが 5.12.10-300.fc34.x86_64 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Red Hat Enterprise Linux バージョン 8 または 9 以降

休止状態をサポートするように Red Hat Enterprise Linux 8 または 9 AMI を設定するには

1. 4.18.0-305.7.1.el8_4.x86_64以降にカーネルを更新します。

```
[ec2-user ~]$ sudo yum update kernel
```

2. このステップでは、Fedora Extra Packages for Enterprise Linux (EPEL) リポジトリをインストールします。

RHEL バージョン 8:

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm
```

RHEL バージョン 9:

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-9.noarch.rpm
```

3. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo yum install ec2-hibinit-agent
```

4. 起動時に休止状態エージェントを起動できるようにします。

```
[ec2-user ~]$ sudo systemctl enable hibinit-agent.service
```

5. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

6. 次のコマンドを実行して、カーネルバージョンが 4.18.0-305.7.1.el8_4.x86_64 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

シリアル番号 20210820 よりも前にリリースされた Ubuntu 20.04 LTS (Focal Fossa)

シリアル番号 20210820 よりも前にリリースされた Ubuntu 20.04 LTS (Focal Fossa) AMI で休止がサポートされるように設定するには

1. linux-aws-kernel を 5.8.0-1038.40 以降に、grub2 を 2.04-1ubuntu26.13 以降に更新します。

```
[ec2-user ~]$ sudo apt update
[ec2-user ~]$ sudo apt dist-upgrade
```

2. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

3. 次のコマンドを実行して、カーネルバージョンが 5.8.0-1038.40 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

4. 次のコマンドを実行して、grub2 バージョンが 2.04-1ubuntu26.13 以降に更新されていることを確認します。

```
[ec2-user ~]$ dpkg --list | grep grub2-common
```

シリアル番号 20190722.1 よりも前にリリースされた Ubuntu 18.04 (Bionic Beaver)

シリアル番号 20190722.1 以前にリリースされた Ubuntu 18.04 LTS AMI で休止がサポートされるように設定するには

1. 4.15.0-1044以降にカーネルを更新します。

```
[ec2-user ~]$ sudo apt update
[ec2-user ~]$ sudo apt dist-upgrade
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo apt install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが 4.15.0-1044 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Ubuntu 16.04 (Xenial Xerus)

Ubuntu 16.04 LTS で休止がサポートされるように設定するには、linux-aws-hwe カーネルパッケージバージョン 4.15.0-1058-aws 以降および ec2-hibinit-agent をインストールする必要があります。

> **Important**
>
> linux-aws-hwe カーネルパッケージは、Canonical でサポートされています。Ubuntu 16.04 LTS の標準サポートは 2021 年 4 月に終了し、パッケージは定期的な更新を受信しなくなりました。ただし、拡張セキュリティメンテナンスのサポートが2024年に終了するまで、追加のセキュリティアップデートを受け取ります。詳細については、「Ubuntu 16.04 LTS 用 Amazon EC2 の休止機能が利用可能に」を参照してください。
> Ubuntu 20.04 LTS (Focal Fossa) AMI または Ubuntu 18.04 LTS (Bionic Beaver) AMI にアップグレードすることをお勧めします。

Ubuntu 16.04 LTS AMI で休止がサポートされるように設定するには

1. 4.15.0-1058-aws以降にカーネルを更新します。

```
[ec2-user ~]$ sudo apt update
[ec2-user ~]$ sudo apt install linux-aws-hwe
```

2. ec2-hibinit-agentパッケージをリポジトリからインストールします。

```
[ec2-user ~]$ sudo apt install ec2-hibinit-agent
```

3. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

4. 次のコマンドを実行して、カーネルバージョンが 4.15.0-1058-aws 以降に更新されていることを確認します。

```
[ec2-user ~]$ uname -a
```

Amazon EC2 インスタンスの休止の有効化

インスタンスを休止するには、まずインスタンスを起動するときに休止を有効にする必要があります。

> **Important**
>
> インスタンスの起動後に、そのインスタンスの休止を有効または無効にすることはできません。

トピック

オンデマンドインスタンスの休止を有効にする

オンデマンドインスタンスの休止を有効にするときは以下のいずれかの方法を使用します。

New console

オンデマンドインスタンスの休止を有効にするには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了して休止状態を有効にするまでインスタンスを起動しないでください。
2. 休止状態を有効にするには、インスタンス起動ウィザードで次のフィールドを設定します。
   a. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、休止状態をサポートする AMI を選択します。詳細については、「AMI」を参照してください。
   b. [Instance type] (インスタンスタイプ) で、サポートされているインスタンスタイプを選択します。詳細については、「インスタンスファミリー」を参照してください。
   c. [Configure storage] (ストレージを設定) で、[Advanced] (高度) (右側) を選択し、ルートボリュームに関する次の情報を指定します。

- [サイズ (GiB)] に、EBS ルートボリュームのサイズを入力します。ボリュームは、RAM の内容を格納して予想使用量に対応できるだけのサイズにする必要があります。
- [Volume type] (ボリュームタイプ) で、サポートされている EBS ボリュームタイプである汎用 SSD (gp2 および gp3) またはプロビジョンド IOPS SSD (io1 および io2) を選択します。
- [Encrypted] (暗号化) で、[Yes] (はい) を選択します。この AWS リージョンでデフォルトで暗号化を有効にした場合、[Yes] (はい) が選択されます。
- [KMS key] (KMS キー) で、ボリュームの暗号化キーを選択します。この AWS リージョンでデフォルトで暗号化を有効にした場合、デフォルトの暗号化キーが選択されます。

ルートボリュームの前提条件の詳細については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。

d. [Advanced details] (高度な詳細) を展開し、[Stop - Hibernate behavior] (停止 - 休止状態の動作) で [Enable] (有効にする) を選択します。

3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

Old console

オンデマンドインスタンスの休止を有効にするには

1. 「」の手順に従います。古いインスタンス起動ウィザードを使用してインスタンスを起動する
2. [Amazon マシンイメージ (AMI)] ページで、休止をサポートする AMI を選択します。サポート対象の AMI の詳細については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。
3. [インスタンスタイプの選択] ページで、サポート対象のインスタンスタイプを選択し、[次の手順: インスタンスの詳細の設定] を選択します。サポート対象のインスタンスタイプの詳細については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。

4. [インスタンスの詳細設定] ページの [Stop - Hibernate Behavior (停止 - 休止動作)] で、[Enable hibernation as an additional stop behavior (追加の停止動作として休止を有効にする)] チェックボックスをオンにします。
5. [ストレージの追加] ページで、ルートボリュームに関する次の情報を指定します。

   - [サイズ (GiB)] に、EBS ルートボリュームのサイズを入力します。ボリュームは、RAM の内容を格納して予想使用量に対応できるだけのサイズにする必要があります。
   - [ボリュームタイプ] で、サポートされている EBS ボリュームタイプである汎用 SSD (gp2 および gp3) またはプロビジョンド IOPS SSD (io1 および io2) を選択します。
   - [暗号化] で、ボリュームの暗号化キーを選択します。この AWS リージョンでデフォルトで暗号化を有効にした場合、デフォルトの暗号化キーが選択されます。

   ルートボリュームの前提条件の詳細については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。
6. ウィザードに従って続行します。[Review Instance Launch] (インスタンス作成の確認) ページでオプションの確認が終了したら、[Launch] (起動) を選択します。詳細については、「古いインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI

オンデマンドインスタンスの休止を有効にするには

run-instances コマンドを使用して、インスタンスを起動します。`--block-device-mappings file://mapping.json` パラメータを使用して EBS ルートボリュームのパラメータを指定し、`--hibernation-options Configured=true` パラメータを使用して休止状態を有効にします。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type m5.large \
    --block-device-mappings file://mapping.json \
    --hibernation-options Configured=true \
    --count 1 \
    --key-name MyKeyPair
```

mapping.json で、以下を指定します。

```
[
```

```
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
            "VolumeSize": 30,
            "VolumeType": "gp2",
            "Encrypted": true
        }
    }
]
```

> **Note**
>
> DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように describe-images コマンドを使用します。
>
> ```
> aws ec2 describe-images --image-id ami-0abcdef1234567890
> ```
>
> この AWS リージョンで暗号化をデフォルトで有効にした場合は、"Encrypted": true を省略できます。

PowerShell

AWS Tools for Windows PowerShell を使用してオンデマンドインスタンスの休止を有効にするには

New-EC2Instance コマンドを使用してインスタンスを起動します。EBS ルートボリュームを指定します。最初にブロックデバイスマッピングを定義し、次に -BlockDeviceMappings パラメータを使用してそれをコマンドに追加します。-HibernationOptions_Configured $true パラメータを使用して休止を有効にします。

```
PS C:\> $ebs_encrypt = New-Object Amazon.EC2.Model.BlockDeviceMapping
PS C:\> $ebs_encrypt.DeviceName = "/dev/xvda"
PS C:\> $ebs_encrypt.Ebs = New-Object Amazon.EC2.Model.EbsBlockDevice
PS C:\> $ebs_encrypt.Ebs.VolumeSize = 30
PS C:\> $ebs_encrypt.Ebs.VolumeType = "gp2"
PS C:\> $ebs_encrypt.Ebs.Encrypted = $true

PS C:\> New-EC2Instance `
            -ImageId ami-0abcdef1234567890 `
```

```
    -InstanceType m5.large `
    -BlockDeviceMappings $ebs_encrypt `
    -HibernationOptions_Configured $true `
    -MinCount 1 `
    -MaxCount 1 `
    -KeyName MyKeyPair
```

> **Note**
>
> DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように Get-EC2Image コマンドを使用します。
>
> ```
> Get-EC2Image -ImageId ami-0abcdef1234567890
> ```
>
> この AWS リージョンで暗号化をデフォルトで有効にした場合は、ブロックデバイスマッピングから Encrypted = $true を省略できます。

スポットインスタンスの休止を有効にする

スポットインスタンスの休止を有効にするときは以下のいずれかの方法を使用します。中断時のスポットインスタンスの休止に関する詳細は、「スポットインスタンスの中断。」を参照してください。

Console

スポットインスタンスの休止を有効にするには、Amazon EC2 コンソールのインスタンス起動ウィザードを使用します。

スポットインスタンスの休止を有効にするには

1. 次の手順に従って、インスタンス起動ウィザードを使ってスポットインスタンスをリクエストしますが、次のステップを完了して休止を有効にするまで、インスタンスを起動しないでください。
2. 休止状態を有効にするには、インスタンス起動ウィザードで次のフィールドを設定します。
   a. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、休止状態をサポートする AMI を選択します。詳細については、「AMI」を参照してください。

b. [Instance type] (インスタンスタイプ) で、サポートされているインスタンスタイプを選択します。詳細については、「インスタンスファミリー」を参照してください。

c. [Configure storage] (ストレージを設定) で、[Advanced] (高度) (右側) を選択し、ルートボリュームに関する次の情報を指定します。

- [サイズ (GiB)] に、EBS ルートボリュームのサイズを入力します。ボリュームは、RAM の内容を格納して予想使用量に対応できるだけのサイズにする必要があります。
- [Volume type] (ボリュームタイプ) で、サポートされている EBS ボリュームタイプである汎用 SSD (gp2 および gp3) またはプロビジョンド IOPS SSD (io1 および io2) を選択します。
- [Encrypted] (暗号化) で、[Yes] (はい) を選択します。この AWS リージョンでデフォルトで暗号化を有効にした場合、[Yes] (はい) が選択されます。
- [KMS key] (KMS キー) で、ボリュームの暗号化キーを選択します。この AWS リージョンでデフォルトで暗号化を有効にした場合、デフォルトの暗号化キーが選択されます。

ルートボリュームの前提条件の詳細については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。

d. [詳細設定] を展開し、スポットインスタンスを設定するフィールドに加えて次の操作を行います。

i. [リクエストタイプ] で [永続的] を選択します。

ii. [中断動作] で [休止] を選択します。または、[停止 - 休止動作] で [有効] を選択します。どちらのフィールドも、スポットインスタンスの休止を有効にします。いずれか 1 つ設定すれば済みます。

3. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI

スポットインスタンスの休止は、run-instances AWS CLI コマンドを使って有効にできます。

`hibernation-options` パラメータを使ってスポットインスタンスの休止を有効にするには

run-instances コマンドを使用してスポットインスタンスをリクエストします。--block-device-mappings file://mapping.json パラメータを使用して EBS ルートボリュームのパラメータを指定し、--hibernation-options Configured=true パラメータを使用して休止状態を有効にします。スポットのリクエストのタイプ (SpotInstanceType) は persistent である必要があります。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c4.xlarge \
    --block-device-mappings file://mapping.json \
    --hibernation-options Configured=true \
    --count 1 \
    --key-name MyKeyPair
    --instance-market-options
        {
           "MarketType":"spot",
           "SpotOptions":{
              "MaxPrice":"1",
              "SpotInstanceType":"persistent"
            }
        }
```

mapping.json の EBS ルートボリュームパラメータを次のとおり指定します。

```
[
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
            "VolumeSize": 30,
            "VolumeType": "gp2",
            "Encrypted": true
        }
    }
]
```

> **Note**
>
> DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように describe-images コマンドを使用します。

```
aws ec2 describe-images --image-id ami-0abcdef1234567890
```

この AWS リージョンで暗号化をデフォルトで有効にした場合は、"Encrypted": true を省略できます。

PowerShell

AWS Tools for Windows PowerShell を使ってスポットインスタンスの休止を有効にするには

New-EC2Instance コマンドを使用してスポットインスタンスをリクエストします。EBS ルートボリュームを指定します。最初にブロックデバイスマッピングを定義し、次に -BlockDeviceMappings パラメータを使用してそれをコマンドに追加します。-HibernationOptions_Configured $true パラメータを使用して休止を有効にします。

```
PS C:\> $ebs_encrypt = New-Object Amazon.EC2.Model.BlockDeviceMapping
PS C:\> $ebs_encrypt.DeviceName = "/dev/xvda"
PS C:\> $ebs_encrypt.Ebs = New-Object Amazon.EC2.Model.EbsBlockDevice
PS C:\> $ebs_encrypt.Ebs.VolumeSize = 30
PS C:\> $ebs_encrypt.Ebs.VolumeType = "gp2"
PS C:\> $ebs_encrypt.Ebs.Encrypted = $true

PS C:\> New-EC2Instance `
             -ImageId ami-0abcdef1234567890 `
             -InstanceType m5.large `
             -BlockDeviceMappings $ebs_encrypt `
             -HibernationOptions_Configured $true `
             -MinCount 1 `
             -MaxCount 1 `
             -KeyName MyKeyPair `
             -InstanceMarketOption @(
                   MarketType = spot;
                   SpotOptions @{
                   MaxPrice = 1;
                   SpotInstanceType = persistent}
               )
```

> **Note**
>
> DeviceName の値は、AMI に関連付けられているルートデバイス名と一致する必要があります。ルートデバイス名を確認するには、次のように Get-EC2Image コマンドを使用します。
>
> ```
> Get-EC2Image -ImageId ami-0abcdef1234567890
> ```
>
> この AWS リージョンで暗号化をデフォルトで有効にした場合は、ブロックデバイスマッピングから Encrypted = $true を省略できます。

スポットインスタンスの休止を有効にする方法はいくつかあります。詳細については、「中断動作の指定」を参照してください。

インスタンスで休止が有効かどうかを表示する

インスタンスで休止が有効になっているかどうかを確認するときは、次の手順に従います。

Console

インスタンスで休止が有効かどうかを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Details (詳細)] タブの [Instance details (インスタンスの詳細)] セクションで、[Stop-hibernate behavior (停止 - 休止動作)] を確認します。[有効] は、インスタンスが休止に対して有効であることを示します。

AWS CLI

インスタンスで休止が有効かどうかを表示するには

describe-instances コマンドを使用し、--filters "Name=hibernation-options.configured,Values=true" パラメータを指定して、休止が有効になっているインスタンスをフィルタリングします。

```
aws ec2 describe-instances \
    --filters "Name=hibernation-options.configured,Values=true"
```

次の出力フィールドは、インスタンスで休止が有効になっていることを示しています。

```
"HibernationOptions": {
    "Configured": true
}
```

PowerShell

AWS Tools for Windows PowerShell を使用して、インスタンスで休止が有効かどうかを表示するには

Get-EC2Instance コマンドを使用し、`-Filter @{ Name="hibernation-options.configured"; Value="true"}` パラメータを指定して、休止が有効になっているインスタンスをフィルタリングします。

```
(Get-EC2Instance -Filter @{Name="hibernation-options.configured";
 Value="true"}).Instances
```

休止が有効になっている EC2 インスタンスが出力に一覧表示されます。

インスタンスでの KASLR の無効化 (Ubuntu のみ)

Ubuntu 16.04 LTS (Xenial Xerus)、Ubuntu 18.04 LTS (Bionic Beaver) (シリアル番号 20190722.1 以降でリリース)、または Ubuntu 20.04 LTS (Focal Fossa) (シリアル番号 20210820 以降でリリース) で新しく起動されたインスタンスで休止を使用するには、KASLR (Kernel Address Space Layout Randomization) を無効にするようお勧めします。Ubuntu 16.04 LTS、Ubuntu 18.04 LTS、または Ubuntu 20.04 LTS では、デフォルトで KASLR が有効になっています。

KASLR は、Linux カーネルに対する標準的なセキュリティ機能であり、カーネルのベースアドレス値をランダム化することにより、未知のメモリアクセス脆弱性による露出と影響を軽減するために役立ちます。KASLR が有効になっている場合は、インスタンスを休止後に再開できないこともあります。

KASLR の詳細については、Ubuntu の機能に関する記述を参照してください。

Ubuntu で起動したインスタンスで KASLR を無効にするには

1. SSH を使用してインスタンスに接続します。詳細については、「SSH を使用して Linux または macOS から Linux インスタンスに接続します。」を参照してください。

2. 適切なエディタで、/etc/default/grub.d/50-cloudimg-settings.cfg ファイルを開きます。次の例のように、GRUB_CMDLINE_LINUX_DEFAULT 行を編集して、行末に nokaslr オプションを追加します。

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty1 console=ttyS0
 nvme_core.io_timeout=4294967295 nokaslr"
```

3. ファイルを保存し、エディタを終了します。
4. grub 設定を再構築するには、次のコマンドを実行します。

```
[ec2-user ~]$ sudo update-grub
```

5. インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

6. 次のコマンドを実行して、nokaslr が追加されたことを確認します。

```
[ec2-user ~]$ cat /proc/cmdline
```

コマンドの出力には、nokaslr オプションが含まれている必要があります。

Amazon EC2 インスタンスの休止

インスタンスが EBS ベースのインスタンスであり、休止が有効になっており、休止の前提条件を満たしている場合、オンデマンドインスタンスまたはスポットインスタンスで休止を開始できます。インスタンスを休止できない場合、通常のシャットダウンが実行されます。

Console

インスタンスを休止するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Instance state (インスタンスの状態)]、[Hibernate instance (インスタンスの休止)] の順に選択します。[Hibernate instance (インスタンスの休止)] が無効になっている場合は、インスタンスが既に休止または停止しているか、休止できません。詳細については、Amazon EC2 インスタンスの休止の前提条件 を参照してください。

4. 確認を求めるメッセージが表示されたら、[休止] を選択します。インスタンスが休止するまで、数分かかる場合があります。インスタンスの状態は、最初に停止中に変化し、インスタンスが休止状態になったときに停止に変化します。

AWS CLI

EBS-Backed インスタンスを休止するには

stop-instances コマンドを使用して `--hibernate` パラメータを指定します。

```
aws ec2 stop-instances \
    --instance-ids i-1234567890abcdef0 \
    --hibernate
```

PowerShell

AWS Tools for Windows PowerShell を使用してインスタンスを休止するには

Stop-EC2Instance コマンドを使用して、`-Hibernate $true` パラメータを指定します。

```
Stop-EC2Instance `
    -InstanceId i-1234567890abcdef0 `
    -Hibernate $true
```

Console

インスタンスで休止が開始されたかどうかを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[詳細] タブの [インスタンスの詳細] セクションで、[状態遷移メッセージ] の値を確認します。

   Client.UserInitiatedHibernate: User initiated hibernate というメッセージは、オンデマンドインスタンスまたはスポットインスタンスで休止が開始されたことを示しています。

AWS CLI

インスタンスで休止が開始されたかどうかを表示するには

describe-instances コマンドを使用して、state-reason-code フィルターを指定し、休止が開始されたインスタンスを確認します。

```
aws ec2 describe-instances \
    --filters "Name=state-reason-code,Values=Client.UserInitiatedHibernate"
```

以下の出力のフィールドは、そのオンデマンドインスタンスまたはスポットインスタンスで休止が開始されたことを示しています。

```
"StateReason": {
    "Code": "Client.UserInitiatedHibernate"
}
```

PowerShell

AWS Tools for Windows PowerShell を使用して、インスタンスで休止が開始されたかどうかを表示するには

Get-EC2Instance コマンドを使用し、state-reason-code フィルタを指定して休止が開始されたインスタンスを確認します。

```
Get-EC2Instance `
    -Filter @{Name="state-reason-code";Value="Client.UserInitiatedHibernate"}
```

休止が開始された EC2 インスタンスが出力に一覧表示されます。

休止した Amazon EC2 インスタンスの起動

休止したインスタンスは、停止したインスタンスを起動するのと同じ方法で起動します。

> **Note**
>
> スポットインスタンスの場合、Amazon EC2 がインスタンスを休止にした場合、それを再開できるのは Amazon EC2 のみです。ユーザーは、自分で休止した場合のみ、休止したスポッ

トインスタンスを再開できます。スポットインスタンスは、容量が空いていて、スポット料金が、指定した上限料金以下である場合、再開できます。

Console

休止したインスタンスの起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 休止したインスタンスを選択し、[Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択します。インスタンスが `running` 状態になるまで、数分かかる場合があります。この間、インスタンスの ステータスチェック では、インスタンスが起動するまで、インスタンスは失敗状態にあるように表示されます。

AWS CLI

休止したインスタンスの起動するには

start-instances コマンドを使用します。

```
aws ec2 start-instances \
    --instance-ids i-1234567890abcdef0
```

PowerShell

AWS Tools for Windows PowerShell を使用して、休止したインスタンスを起動するには

Start-EC2Instance コマンドを使用します。

```
Start-EC2Instance `
    -InstanceId i-1234567890abcdef0
```

Amazon EC2 インスタンスの休止のトラブルシューティング

次の情報を使用して、インスタンスを休止するときに発生する可能性がある問題の診断や修復を行います。

休止に関する問題

起動直後に休止できません

インスタンスの起動後にすぐ休止しようとすると、エラーが発生します。

起動後、Linux インスタンスの場合は約 2 分、Windows インスタンスの場合は約 5 分待ってから休止する必要があります。

stopping から **stopped** への移行に時間がかかりすぎ、起動後にメモリ状態が復元されません

休止しているインスタンスが stopping 状態から stopped に移行するのに時間がかかり過ぎ、メモリの状態が起動後に復元されない場合は、休止が正しく設定されていない可能性があります。

これらのプロセスで何もログが表示されない場合、AMI が休止をサポートしていない可能性があります。サポート対象の AMI の詳細については、「Amazon EC2 インスタンスの休止の前提条件」を参照してください。独自の Linux AMI を使用した場合は、Linux AMI で休止がサポートされるように設定する に関する指示に従っていることを確認します。

Linux インスタンス

インスタンスのシステムログをチェックして、休止に関連するメッセージを探します。システムログにアクセスするには、インスタンスに接続するか、get-console-output コマンドを使用します。hibinit-agent からログ行を見つけます。ログ行が失敗を示している場合、またはログ行がない場合、起動時に休止の設定に失敗している可能性が高いと思われます。

例えば、メッセージ「hibinit-agent: Insufficient disk space. Cannot create setup for hibernation. Please allocate a larger root device.」は、インスタンスのルートボリュームの大きさが十分ではないことを示しています。

hibinit-agent からの最後のログ行が hibinit-agent: Running: swapoff /swap である場合、休止は正常に設定されています。

Windows Server 2016 以降

EC2 起動ログをチェックして、休止に関連するメッセージを探します。EC2 起動ログにアクセスするには、インスタンスに接続し、テキストエディタで C:\ProgramData\Amazon\EC2-Windows\Launch\Log\Ec2Launch.log ファイルを開きます。EC2Launch v2 を使用している場合は、C:\ProgramData\Amazon\EC2Launch\log\agent.log を開きます。

> **Note**
>
> Windows では、デフォルトで C:\ProgramData 以下のファイルとフォルダは非表示になります。EC2 起動ディレクトリおよびファイルを表示するには、Windows エクスプローラーにパスを入力するか、フォルダのプロパティを変更して非表示のファイルおよびフォルダを表示します。

休止に関するログ行を見つけます。ログ行が失敗を示している場合、またはログ行がない場合、起動時に休止の設定に失敗している可能性が高いと思われます。

例えば、「Message: Failed to enable hibernation.」というメッセージは、休止の設定に失敗したことを示しています。エラーメッセージに 10 進数の ASCII 値が含まれている場合は、ASCII 値をプレーンテキストに変換すると、エラーメッセージ全体を読み取ることができます。

ログ行に HibernationEnabled: true が含まれている場合、休止は正常に設定されています。

Windows Server 2012 R2 以前

EC2 設定ログをチェックして、休止に関連するメッセージを探します。EC2 設定ログにアクセスするには、インスタンスに接続し、テキストエディタで C:\Program Files\Amazon\Ec2ConfigService\Logs\Ec2ConfigLog.txt ファイルを開きます。SetHibernateOnSleep のログ行を見つけます。ログ行が失敗を示している場合、またはログ行がない場合、起動時に休止の設定に失敗している可能性が高いと思われます。

例えば、メッセージ「SetHibernateOnSleep: Failed to enable hibernation: Hibernation failed with the following error: There is not enough space on the disk.」は、インスタンスのルートボリュームの大きさが十分ではないことを示しています。

ログ行が SetHibernateOnSleep: HibernationEnabled: true である場合、休止は正常に設定されています。

Windows インスタンスサイズ

1 GB 未満の RAM を持つ T3 または T3a Windows インスタンスを使用している場合は、インスタンスのサイズを少なくとも 1 GB の RAM に増加してみてください。

インスタンスの stopping 状態での停止

インスタンスを休止し、stopping 状態で止まったように見える場合は、インスタンスを強制終了できます。詳細については、「インスタンスの停止に関するトラブルシューティング」を参照してください。

休止の直後にスポットインスタンスを起動できない

休止にしてから 2 分以内にスポットインスタンスを起動しようとすると、次のエラーが発生する場合があります。

```
You failed to start the Spot Instance because the associated Spot Instance request is not in an appropriate state to support start.
```

Linux インスタンスの場合は約 2 分、Windows インスタンスの場合は約 5 分待ってから、インスタンスの起動を再試行してください。

スポットインスタンスを再開できない

スポットインスタンスが正常に休止されたが再開に失敗し、代わりに再起動 (休止状態を維持せずに新たに再起動) した場合、ユーザーデータに次のスクリプトが含まれていたことが原因である可能性があります。

```
/usr/bin/enable-ec2-spot-hibernation
```

起動テンプレートの [ユーザーデータ] フィールドからこのスクリプトを削除し、新しいスポットインスタンスをリクエストしてください。

休止状態を維持せずにインスタンスの再開に失敗した場合でも、インスタンスは stopped 状態から開始するのと同じ方法で起動できることに注意してください。

インスタンスの再起動

インスタンスの再起動は、オペレーティングシステムの再起動と同等です。ほとんどの場合、インスタンスの再起動には数分しかかかりません。

インスタンスを再起動しても、次の状態が維持されます。

- パブリック DNS 名 (IPv4)
- プライベート IPv4 アドレス
- パブリック IPv4 アドレス
- IPv6 アドレス (該当する場合)
- インスタンスストアボリューム上のすべてのデータ

インスタンスの停止および開始の場合とは異なり、インスタンスを再起動しても、インスタンスの (1 分間分の最低料金が発生する) 課金期間が新しく開始されることはありません。

再起動を必要とする更新の適用など、必要なメンテナンスのために、インスタンスの再起動を予定する場合があります。ユーザーが操作する必要はありません。予定されている時間帯に自動的に行われる再起動まで待つことをお勧めします。詳細については、インスタンスの予定されたイベント を参照してください。

インスタンスからオペレーティングシステムの再起動コマンドを実行する代わりに、Amazon EC2 コンソール、コマンドラインツール、または Amazon EC2 API を使用してインスタンスを再起動することをお勧めします。Amazon EC2 コンソール、コマンドラインツール、または Amazon EC2 API を使用してインスタンスを再起動する場合、インスタンスが数分以内に完全にシャットダウンしないと、ハードリブートが実行されます。AWS CloudTrail を使用しながら、Amazon EC2 によりインスタンスを再起動した場合は、インスタンスがいつ再起動されたかについての API レコードが作成されます。

Console

コンソールを使用してインスタンスを再起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択してから、[Instance state] (インスタンス状態)、[Reboot instance] (インスタンスの再起動) の順に選択します。

   または、インスタンスを選択して、[Actions] (アクション)、[Manage instance state] (インスタンス状態の管理) の順に選択します。表示される画面で、[Reboot] (再起動)、[Change state] (状態の変更) の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Yes, Reboot (再起動する)] を選択します。

   インスタンスは `running` 状態を維持します。

Command line

インスタンスを再起動するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- reboot-instances (AWS CLI)
- Restart-EC2Instance (AWS Tools for Windows PowerShell)

制御された故障注入実験を実行するには

AWS Fault Injection Service を使用して、インスタンスが再起動されたときのアプリケーションの応答をテストできます。詳細については、AWS Fault Injection Service ユーザーガイドを参照してください。

インスタンスのリタイア

インスタンスをホストしている基盤のハードウェアで回復不可能な障害が検出されると、AWS によってインスタンスのリタイアが予定されます。予定されたリタイア日になると、インスタンスは AWS によって停止または削除されます。

- インスタンスのルートデバイスが Amazon EBS ボリュームである場合、インスタンスは停止されますが、その後いつでも再び起動できます。停止したインスタンスを開始すると、新しいハードウェアに移行されます。
- インスタンスのルートデバイスがインスタンスストアボリュームである場合、インスタンスは削除し、再び使用することはできません。

インスタンスイベントのタイプの詳細については、「インスタンスの予定されたイベント」を参照してください。

コンテンツ

リタイアが予定されているインスタンスの特定

インスタンスのリタイアが予定された場合、イベントの前に、当該のインスタンス ID とリタイア日を記載したメールが送信されます。Amazon EC2 コンソールまたはコマンドラインを使用して、リタイアが予定されているインスタンスを確認することもできます。

> **Important**
>
> インスタンスのリタイアが予定されている場合は、インスタンスが到達不能になる可能性があるため、できるだけ早くアクションを実行することをお勧めします (受信した E メール通知では、「このパフォーマンスの低下により、インスタンスはすでに到達不能になっている可能性があります」と通知されます)。実行が推奨されるアクションの詳細については、「Check if your instance is reachable」を参照してください。

リタイアが予定されているインスタンスを特定する方法

- E メール通知
- コンソールの識別

E メール通知

インスタンスのリタイアが予定された場合、イベントの前に、当該のインスタンス ID とリタイア日を記載したメールが送信されます。

この電子メールは、プライマリアカウントの所有者とオペレーション担当の連絡先に送信されます。詳細については、 AWS Billing ユーザーガイドの、「代替連絡先の追加、変更、または削除」を参照してください。

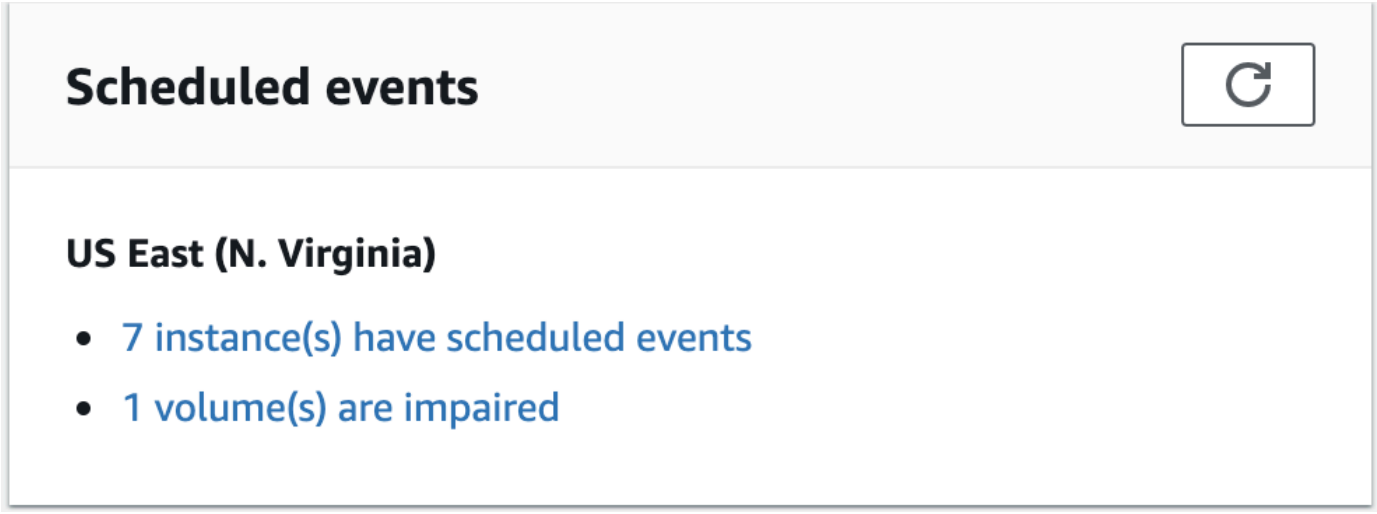
コンソールの識別

インスタンスのリタイア通知を定期的に確認しないメールアカウントを使用している場合は、Amazon EC2 コンソールまたはコマンドラインを使用して、いずれかのインスタンスにリタイアが予定されているかどうかを判断できます。

コンソールを使用してリタイアが予定されているインスタンスを特定するには

1. Amazon EC2 コンソールを開きます。

2. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。[スケジュールされたイベント] に、Amazon EC2 インスタンスおよびボリュームに関連付けられたイベントが、リージョン別に整理されて表示されます。



3. インスタンスに予定されたイベントが表示されている場合は、リージョン名の下のリンクを選択して [Events] ページにアクセスします。
4. [Events (イベント)] ページには、すべてのリソースとそれに関連付けられたイベントが一覧表示されます。リタイアが予定されているインスタンスを表示するには、1 つ目のフィルタリストから [Instance resources] を選択し、2 つ目のフィルタリストから [Instance stop or retirement] を選択します。
5. フィルタの結果にインスタンスのリタイアが予定されていることが表示されたら、当該のインスタンスを選択し、詳細ペインの [Start time] フィールドの日時を書き留めます。これがインスタンスのリタイア日です。

コマンドラインを使用してリタイアが予定されているインスタンスを特定するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- describe-instance-status (AWS CLI)
- Get-EC2InstanceStatus (AWS Tools for Windows PowerShell)

リタイアが予定されている EBS-backed インスタンスに対して実行するアクション

リタイアが予定されているインスタンスのデータを保持するには、次のいずれかのアクションを実行できます。予期しないダウンタイムやデータ消失を防ぐために、インスタンスのリタイア日より前にこのアクションを実行することが重要です。

インスタンスが EBS とインスタンスストアのどちらから提供されているかわからない場合は、「インスタンスのルートデバイスタイプの判別」を参照してください。

インスタンスが到達可能かどうかを確認する

インスタンスのリタイアが予定されていることが通知された場合は、できるだけ早く以下のアクションを実行することをお勧めします。

- インスタンスに接続するか、インスタンスに ping を実行して、インスタンスが到達可能かどうかを確認します。
- インスタンスに到達可能な場合は、スケジュールされたリタイア日の前の適切なタイミングで、その影響が最小限であるときにインスタンスを停止/開始するよう計画する必要があります。インスタンスの停止と開始、インスタンスを停止したときに予想される影響 (インスタンスに関連付けられたパブリック IP アドレス、プライベート IP アドレス、および Elastic IP アドレスへの影響など) の詳細については、「インスタンスの停止と起動」を参照してください。インスタンスを停止して起動すると、インスタンスストアボリュームのデータが失われることに注意してください。
- インスタンスに到達できない場合は、直ちにアクションを実行し、停止/開始を実行してインスタンスを復元する必要があります。
- または、インスタンスを削除する場合は、できるだけ早く削除するよう計画し、インスタンスの料金が発生しないようにしてください。

インスタンスのバックアップを作成する

バックアップが作成されるように、インスタンスから EBS-Backed AMI を作成します。データの整合性を確保するには、AMI を作成する前にインスタンスを停止します。予定されたリタイア日を待つことができます。その日になるとインスタンスが停止できます。または、リタイア日の前に自分でインスタンスを停止します。インスタンスはいつでも再開できます。詳細については、「Amazon EBS-backed Linux AMI を作成する」を参照してください。

代わりのインスタンスを起動する

インスタンスから AMI を作成した後、AMI を使用して代替インスタンスを起動できます。Amazon EC2 コンソールから、新しい AMI を選択して、[Actions (アクション)]、[Launch (起動)] の順に選択

します。ウィザードに従って、インスタンスを起動します。ウィザードの各ステップの詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

リタイアが予定されている instance-store backed インスタンスに対して実行するアクション

リタイアが予定されているインスタンスのデータを保持するには、次のいずれかのアクションを実行できます。予期しないダウンタイムやデータ消失を防ぐために、インスタンスのリタイア日より前にこのアクションを実行することが重要です。

> **Warning**
>
> Instance Store-Backed インスタンスの場合、リタイア日を過ぎるとインスタンスが終了し、インスタンスやインスタンスに格納されていたデータを復元できなくなります。インスタンスストアボリュームのデータは、インスタンスのルートデバイスにかかわらず、EBS-Backed インスタンスにボリュームがアタッチされている場合でも、インスタンスがリタイアされると失われます。

インスタンスが到達可能かどうかを確認する

インスタンスのリタイアが予定されていることが通知された場合は、できるだけ早く以下のアクションを実行することをお勧めします。

- インスタンスに接続するか、インスタンスに ping を実行して、インスタンスが到達可能かどうかを確認します。
- インスタンスに到達できない場合、インスタンスを復元するために実行できることはほとんどありません。詳細については、接続できないインスタンスのトラブルシューティング をご参照ください。AWS は、予定されたリタイア日にインスタンスを削除するため、到達不能なインスタンスについては、お客様自身ですぐにインスタンスを削除できます。

代わりのインスタンスを起動する

AMI ツールを使用してインスタンスから instance store-backed AMI を作成します (「instance store-backed Linux AMI を作成する」を参照)。Amazon EC2 コンソールから、新しい AMI を選択して、[Actions (アクション)]、[Launch (起動)] の順に選択します。ウィザードに従って、インスタンスを起

動します。ウィザードの各ステップの詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

インスタンスを EBS-backed インスタンスに変換する

データを EBS ボリュームに転送し、ボリュームのスナップショットを作成した後、スナップショットから AMI を作成します。新しい AMI から代替インスタンスを起動できます。詳細については、instance store-backed AMI を Amazon EBS-backed AMI への変換 を参照してください。

インスタンスの終了

不要になったインスタンスは削除できます。これは、インスタンスの終了 と呼ばれます。インスタンスの状態が `shutting-down` または `terminated` に変わったら、そのインスタンスへの課金は停止します。

インスタンスを削除した後に、接続または起動することはできません。ただし、同じ AMI から別のインスタンスを起動することができます。インスタンスを停止および起動するか、または休止する場合は、「インスタンスの停止と起動」または「Amazon EC2 インスタンスの休止」を参照してください。詳細については、「再起動、停止、休止、削除の違い」を参照してください。

インスタンスを終了した場合に何が行われるかの詳細については、「インスタンスを削除するとどうなるか」を参照してください。

トピック

- 終了に関する考慮事項
- インスタンスの終了

終了に関する考慮事項

インスタンスを終了する前に、次の点を考慮したほうがよいでしょう。

- 終了保護を有効化する – AWS Management Console、AWS Command Line Interface (AWS CLI)、API を使用している他のユーザーによって、誤ってインスタンスを終了されないようにします。
- インスタンスによって起動されたシャットダウン動作の変更 – システムをシャットダウンするオペレーティングシステムコマンドを使用して、インスタンスからシャットダウンが開始されたときに、インスタンスを停止または終了するかどうかを制御します。
- インスタンスの終了時にデータを保持する — インスタンスの終了時にデータが失われないようにします。

終了保護を有効化する

インスタンスを誤って終了できないようにするには、インスタンスの停止保護を有効にします。`DisableApiTermination` 属性は、インスタンスが AWS Management Console、AWS Command Line Interface (AWS CLI)、API を使用して終了できるかどうかを制御します。デフォルトでは、インスタンスの終了保護は無効になっています。つまり AWS Management Console、AWS CLI、API を使用してインスタンスを終了できます。インスタンスの実行中または停止中にインスタンスを起動する際に、この属性の値を設定できます (Amazon EBS backed インスタンスの場合)。

`DisableApiTermination` 属性が設定された場合、`InstanceInitiatedShutdownBehavior` 属性はインスタンスからシャットダウンを開始して (システムシャットダウン用のオペレーティングシステムコマンドを使用)、インスタンスを終了できます。詳細については、「インスタンスによって起動されたシャットダウン動作の変更」を参照してください。

制限事項

- 終了保護を有効にしても、インスタンスにインスタンスを終了する予定されたイベントがある場合、AWS によるインスタンスの終了は防げません。
- 終了保護を有効にしても、インスタンスが異常な場合やスケールインイベント中に Amazon EC2 Auto Scaling がインスタンスを終了することは防げません。スケールイン時に Auto Scaling グループが特定のインスタンスを終了できるかどうかを制御するには、インスタンスのスケールイン保護を使用します。Auto Scaling グループが異常なインスタンスを終了できるかどうかを制御するには、ReplaceUnhealthy スケーリングプロセスを中断します。
- スポットインスタンス の削除保護を有効にすることはできません。

起動時にインスタンスに対する終了保護を有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ダッシュボードで、[Launch Instance] を選択し、ウィザードの指示に従います。
3. [Configure Instance Details] ページで、[Enable termination protection] チェックボックスをオンにします。

実行中または停止中のインスタンスの削除保護を有効にするには

1. インスタンスを選択してから、[Actions (アクション)]、[インスタンスの設定]、[削除保護の変更] の順に選択します。
2. [はい、有効化する] を選択します。

実行中または停止中のインスタンスの削除保護を無効にするには

1. インスタンスを選択してから、[Actions (アクション)]、[インスタンスの設定]、[削除保護の変更] の順に選択します。
2. [Yes, Disable] を選択します。

コマンドラインを使用して終了保護を有効または無効にするには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- modify-instance-attribute (AWS CLI)
- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

終了保護を使用してアベイラビリティーゾーン全体で複数のインスタンスを終了する

複数のアベイラビリティーゾーンで複数のインスタンスを同時に終了する場合、指定した中に終了保護が有効になっているインスタンスが 1 つ以上存在すると、そのリクエストは失敗し次のような結果が返されます。

- 保護されたインスタンスと同じアベイラビリティーゾーンにあるインスタンスは終了されません。
- 保護されたインスタンスが他に存在しないアベイラビリティーゾーンでは、特定のインスタンスを正常に終了することができます。

例えば、以下のようなインスタンスを考えます。

インスタンス	アベイラビリティーゾーン	終了保護
インスタンス A	us-east-1a	Disabled
インスタンス B		Disabled
インスタンス C	us-east-1b	Enabled
インスタンス D		Disabled

これらのインスタンスすべてを、同じリクエストで終了しようとすると、そのリクエストは次のような結果とともに失敗します。

- us-east-1a 内で指定されたインスタンスのいずれも終了保護が有効化されていないので、インスタンス A およびインスタンス B は正常に終了します。
- インスタンス C およびインスタンス D は終了に失敗します。これは、us-east-1b 内で指定したインスタンスのうち少なくとも 1 つ(インスタンスC) で、終了保護が有効化されているためです。

インスタンスによって起動されたシャットダウン動作の変更

デフォルトで、Amazon EBS backed インスタンスからシャットダウンを開始すると (shutdown や poweroff などのコマンドを使用すると)、インスタンスは停止します。インスタンスの InstanceInitiatedShutdownBehavior 属性を変更すると、この動作を変更して、停止ではなく終了するようにできます。インスタンスの実行中または停止中に、この属性を変更できます。

halt コマンドはシャットダウンを開始しません。使用した場合、インスタンスは終了しません。代わりに、CPU が HLT 状態になり、インスタンスは実行されたままになります。

> **Note**
>
> InstanceInitiatedShutdownBehavior 属性は、インスタンス自体のオペレーティングシステムからシャットダウンを実行した場合にのみ適用されます。StopInstances API または Amazon EC2 コンソールを使用してインスタンスを停止した場合、適用されません。

InstanceInitiatedShutdownBehavior 属性は Amazon EC2 コンソールまたはコマンドラインを使用して変更できます。

Console

インスタンスによって開始されたシャットダウン動作を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択します。
4. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Change shutdown behavior (シャットダウン動作の変更)] の順に選択します

[シャットダウン動作] に現在の動作が表示されます。

5. 動作を変更するには、[シャットダウン動作] で [停止] または [終了] を選択します。
6. [Save] を選択します。

Command line

インスタンスによって開始されたシャットダウン動作を変更するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- modify-instance-attribute (AWS CLI)
- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

インスタンスの終了時にデータを保持する

ユースケースによっては、Amazon EC2 インスタンスが終了したときに、インスタンスストアボリュームまたは Amazon EBS ボリュームにデータを保存したい場合があるでしょう。インスタンスストアボリューム上のデータは、インスタンスが終了すると消滅します。インスタンスストアボリュームに保存されているデータを、インスタンスのライフタイムを超えて保持する必要がある場合は、そのデータを Amazon EBS ボリューム、Amazon S3 バケット、Amazon EFS ファイルシステムなどのより永続的なストレージに手動でコピーする必要があります。詳細については、「Amazon EC2 インスタンスのストレージオプション」を参照してください。

Amazon EBS ボリュームのデータについて、Amazon EC2 はアタッチされた各 Amazon EBS ボリュームの `DeleteOnTermination` 属性の値を使用して、ボリュームを保持するか削除するかを決定します。

`DeleteOnTermination` 属性のデフォルト値は、ボリュームがインスタンスのルートボリュームであるか、インスタンスにアタッチされているルート以外のボリュームであるかによって異なります。

ルートボリューム

デフォルトでは、インスタンスのルートボリュームの `DeleteOnTermination` 属性は `true` に設定されます。したがって、デフォルトではインスタンスの終了時に、インスタンスのルートボリュームが削除されます。

ルート以外のボリューム

デフォルトでは、インスタンスにルート以外の EBS ボリュームをアタッチするときは、DeleteOnTermination 属性が false に設定されます。したがって、デフォルトではこれらのボリュームが保持されます。

> **Note**
>
> インスタンスが終了したら、保持されたボリュームのスナップショットを作成するか、別のインスタンスにアタッチできます。不要な料金の発生を防ぐために、ボリュームを削除する必要があります。

DeleteOnTermination 属性は、AMI の作成者とインスタンスを起動するユーザーが設定できます。AMI の作成者またはインスタンスを起動したユーザーによって属性が変更された場合、元の AMI のデフォルト設定は新しい設定に上書きされます。AMI でインスタンスを起動したら、DeleteOnTermination 属性のデフォルト設定を確認することをお勧めします。

インスタンスの終了時に Amazon EBS ボリュームが削除されるかどうかを確認するには、インスタンスの詳細ペインでボリュームの詳細を表示します。ブロックデバイスの下にある、ストレージタブを右にスクロールしてボリュームの終了時に削除設定を確認します。

- [はい] の場合、ボリュームはインスタンスの終了時に削除されます。
- [いいえ] の場合、ボリュームはインスタンスの終了時に削除されません。インスタンスの終了時に削除されなかったボリュームには、引き続き料金がかかります。

コンソールまたはコマンドラインを使用して起動時に Amazon EBS ルートボリュームが存続するように変更する

コンソールを使用して、インスタンスの起動時に DeleteOnTermination 属性を変更できます。実行中のインスタンスのこの属性を変更するには、コマンドラインを使用する必要があります。

起動時にルートボリュームが存続するように変更するには、次のいずれかの方法を使用します。

Console

コンソールを使用して、起動時にインスタンスのルートボリュームが存続するように変更するには

1. 手順に従ってインスタンスを起動しますが、次のステップを完了してルートボリュームを存続するように変更するまでインスタンスを起動しないでください。
2. [ストレージ (ボリューム)] で、ルートボリュームの下の情報を展開します。
3. [終了時に削除] で [いいえ] を選択します。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

Command line

コマンドラインを使用して、起動時にインスタンスのルートボリュームが存続するように変更するには

EBS-backed インスタンスの起動時に、次のコマンドのいずれかを使用して、ルートデバイスボリュームが存続するように変更することができます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- run-instances (AWS CLI)
- New-EC2Instance (AWS Tools for Windows PowerShell)

永続化するボリュームのブロックデバイスマッピングで、`--DeleteOnTermination` を含め、`false` を指定します。

例えば、ボリュームを永続化するには、`run-instances` コマンドに次のオプションを追加します。

```
--block-device-mappings file://mapping.json
```

`mapping.json` では、デバイス名を指定し (例: `/dev/sda1` または `/dev/xvda`)、`--DeleteOnTermination` で `false` を指定します。

```
[
```

```
  {
    "DeviceName": "device_name",
    "Ebs": {
      "DeleteOnTermination": false
    }
  }
]
```

コマンドラインを使用して実行中のインスタンスの Amazon EBS ルートボリュームが存続するように変更する

次のいずれかのコマンドを使用して、実行中の EBS-backed インスタンスのルートデバイスボリュームを永続化するように変更できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- modify-instance-attribute (AWS CLI)
- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

例えば、以下のコマンドを使用します。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --block-device-mappings file://mapping.json
```

mapping.json では、デバイス名を指定し (例: /dev/sda1 または /dev/xvda)、--DeleteOnTermination で false を指定します。

```
[
  {
    "DeviceName": "device_name",
    "Ebs": {
      "DeleteOnTermination": false
    }
  }
]
```

インスタンスの終了

インスタンスを終了する前に、次の点を確認してください。

- インスタンスが終了すると、そのインスタンスに関連付けられたすべてのインスタンスストアボリュームのデータが削除されます。
- デフォルトでは、インスタンスの削除時に Amazon EBS のルートデバイスボリュームが自動的に削除されます。ただし、起動時にアタッチした追加の EBS ボリューム、または既存のインスタンスにアタッチした EBS ボリュームがある場合、インスタンスの削除後もそれらのボリュームは保持されます。詳細については、「インスタンスの終了時にデータを保持する」を参照してください。

> **Note**
>
> インスタンスの終了時に削除されなかったボリュームには、引き続き料金がかかります。

- インスタンスの終了時にスクリプトを実行した場合は、シャットダウンスクリプトの実行を保証する方法がないため、終了処理が正常に行われない可能性があります。Amazon EC2 は、必要なシステムシャットダウンスクリプトを実行し、インスタンスを正常にシャットダウンしようと試みます。ただし、特定のイベント (ハードウェア障害など) ではシステムシャットダウンスクリプトが実行されないことがあります。

トピック

インスタンスを削除するとどうなるか

インスタンスを終了すると、変更はインスタンスの OS レベルで登録され、一部のシステムリソースは失われ、一部は存続します。

インスタンスを終了すると、OSレベルで以下のように登録されます。

- API リクエストは、ボタンのクリックイベントをゲストに送信します。
- ボタンのクリックイベントの結果として、さまざまなシステムサービスが停止します。Linux では、systemdがシステムの正常なシャットダウンを処理します。適切なシャットダウンは、ハイパーバイザーからの ACPI シャットダウンボタンのクリックイベントによってトリガーされます。
- ACPI のシャットダウンが開始されます。

- このインスタンスは、適切なシャットダウンプロセスが終了したときにシャットダウンされます。設定可能な OS シャットダウン時間はありません。インスタンスはしばらくの間コンソールに表示されたままですが、エントリは自動的に削除されます。

インスタンスを終了すると、次のものが失われます。

- インスタンスストアボリュームに保存されているデータは失われます。
- `DeleteOnTermination` 属性が true に設定されている場合、データは Amazon EBS ルートデバイスボリュームに保存されます。

インスタンスを終了したとき、次のものが保持されます。

- インスタンスの起動時または起動後にアタッチされた追加の Amazon EBS ボリュームに保存されたデータ。

インスタンスの終了

インスタンスは AWS Management Console またはコマンドラインを使用して終了できます。

Console

コンソールを使用してインスタンスを終了するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[インスタンスの状態]、[インスタンスの終了] の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択します。

Command line

コマンドラインを使用してインスタンスを削除するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- terminate-instances (AWS CLI)
- Remove-EC2Instance (AWS Tools for Windows PowerShell)

インスタンスの終了に対するアプリケーションの応答をテスト

AWS Fault Injection Service を使用すると、インスタンスが終了した場合のアプリケーションの応答をテストできます。詳細については、AWS Fault Injection Service ユーザーガイドを参照してください。

インスタンスの終了のトラブルシューティング

インスタンスを終了して別のインスタンスを起動する場合、通常 EC2 フリート や Amazon EC2 Auto Scaling などの機能を通じて自動スケーリングを設定している可能性があります。

インスタンスが通常より長く `shutting-down` 状態になっている場合、Amazon EC2 サービス内の自動プロセスによってクリーンアップ (終了) されるはずです。詳細については、「インスタンスの削除の遅延」を参照してください。

関連リソース

Amazon EBS ボリュームの削除の詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームの削除」を参照してください。

インスタンスの復旧

システムステータスチェックが失敗した際にインスタンスを自動的に復旧するには、インスタンスのデフォルト設定を使用するか、Amazon CloudWatch アラームを作成できます。基盤となるハードウェアの障害、または修復に AWS の関与が必要である問題が原因でインスタンスが到達不能になった場合、インスタンスは自動的に復旧します。

復旧されたインスタンスは、インスタンス ID、プライベート IP アドレス、Elastic IP アドレス、すべてのインスタンスメタデータを含め、元のインスタンスと同じです。正常に機能していないインスタンスにパブリック IPv4 アドレスが割り当てられている場合は、そのインスタンスの復旧後も、パブリック IPv4 アドレスは保持されます。障害のあるインスタンスがプレイスメントグループ内にある場合、回復されたインスタンスはそのプレイスメントグループ内で実行されます。インスタンスの復元中、再起動処理の一部としてそのインスタンスの移行が行われ、メモリ内にあるデータは失われます。

インスタンスの復元が必要となる障害の例を以下に示します。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのソフトウェアの問題

- ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

トピック

- インスタンス構成に基づく簡易自動復旧
- Amazon CloudWatch アクションに基づく復旧
- インスタンスの復旧の失敗のトラブルシューティング

インスタンス構成に基づく簡易自動復旧

簡易自動復旧をサポートするインスタンスでは、障害が発生した場合のインスタンスの復元が、デフォルトで設定されています。デフォルト設定は、起動する新しいインスタンスおよび以前に起動した既存のインスタンスに適用されます。簡易自動復旧は、システムステータスチェックの失敗時に開始されます。簡易自動復旧は、Service Health Dashboard のイベントや、基盤となるハードウェアに影響するその他のイベント中は実行されません。詳細については、「the section called “インスタンスの復旧の失敗のトラブルシューティング”」を参照してください。

簡易自動復旧のイベントが成功すると、AWS Health Dashboard のイベントで通知されます。簡易自動復旧のイベントが失敗すると、AWS Health Dashboard のイベントと E メールで通知されます。Amazon EventBridge ルールを使用して、次のイベントコードを使って簡易自動復旧のイベントをモニタリングすることもできます。

- `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_SUCCESS` – 成功したイベント
- `AWS_EC2_SIMPLIFIED_AUTO_RECOVERY_FAILURE` – 失敗したイベント

詳細については、「Amazon EventBridge ルール」を参照してください。

トピック

- 要件
- 制限事項
- 復旧の動作を設定する

要件

以下の特性を持つインスタンスが、簡易自動復旧をサポートします。

- `default` または `dedicated` のインスタンステナンシーを使用している。

- Elastic Fabric Adaptor は使用しません。
- 以下のインスタンスタイプのいずれかを使用している。
  - 凡用: A1 | M3 | M4 | M5 | M5a | M5n | M5zn | M6a | M6g | M6i | M6in | M7a | M7g | M7i | M7i-flex | T1 | T2 | T3 | T3a | T4g
  - コンピューティング最適化: C3 | C4 | C5 | C5a | C5n | C6a | C6g | C6gn | C6i | C6in | C7a | C7g | C7gn | C7i
  - メモリ最適化: R3 | R4 | R5 | R5a | R5b | R5n | R6a | R6g | R6i | R6in | R7a | R7g | R7i | R7iz | u-3tb1 | u-6tb1 | u-9tb1 | u-12tb1 | u-18tb1 | u-24tb1 | X1 | X1e | X2iezn
  - 高速コンピューティング: G3 | G3s | G5g | Inf1 | P2 | P3 | VT1
  - ハイパフォーマンスコンピューティング: Hpc6a | Hpc7a | Hpc7g
- インスタンスストアボリュームがありません。Nitro インスタンスタイプにインスタンスストアボリュームがある場合、または Xen ベースのインスタンスで、使用されている AMI にインスタンスストアボリュームがマップされている場合、インスタンスは自動的に復旧できません。

> **Important**
>
> インスタンスにインスタンスストアボリュームがアタッチされている場合、インスタンスを停止して起動すると、インスタンスストアボリューム上のデータはすべて失われます。インスタンスストアのボリュームデータを、Amazon EBS、Amazon S3、Amazon EFS などのより永続的なストレージに定期的にバックアップする必要があります。システムステータスチェックに失敗した場合は、インスタンスストアボリュームを使用してインスタンスを停止および開始し、バックアップデータを使用してインスタンスストアボリュームを復元できます。

制限事項

- インスタンスストアボリュームを持ちメタルインスタンスタイプを使用するインスタンスでは、簡易自動復旧はサポートしていません。
- Auto Scaling グループ内のインスタンスの場合、簡易自動復旧は開始されません。ヘルスチェックを有効にした Auto Scaling グループに属するインスタンスの場合、障害が発生すると、そのインスタンスは別のインスタンスに置き換えられます。
- 簡易自動復旧は、計画外のイベントにのみ適用されます。スケジュールされたイベントには適用されません。
- 終了または停止したインスタンスは復旧できません。

復旧の動作を設定する

インスタンスの起動中または起動後に、自動復旧の動作を [disabled] または [default] に設定できます。簡易自動復旧をサポートしていないインスタンスタイプでは、デフォルトを設定してもこの機能は有効化されません。

Console

インスタンスの起動中に簡易自動復旧を無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス)、[Launch instance] (インスタンスの起動) の順に選択します。
3. [Advanced details] (高度な詳細) セクションの [Instance auto-recovery] (インスタンスの自動復旧) で、[Disabled] (無効) を選択します。
4. 必要に応じて残りのインスタンスの起動設定を設定し、インスタンスを起動します。

実行中または停止中のインスタンスの簡易自動復旧を無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Change shutdown behavior (自動復旧動作を変更)] の順に選択します 現在の動作が選択されます。
4. [Off] (オフ) を選択した上で、[Save] (保存) をクリックします。

実行中または停止中のインスタンスの自動復旧動作を `default` に設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. [Actions (アクション)]、[Instance settings (インスタンスの設定)]、[Change shutdown behavior (自動復旧動作を変更)] の順に選択します 現在の動作が選択されます。
4. [デフォルト] を選択した上で、[保存] をクリックします。

AWS CLI

起動時に簡易自動復旧を無効にするには

run-instances コマンドを使用します。

```
aws ec2 run-instances \
--image-id ami-1a2b3c4d \
--instance-type t2.micro \
--key-name MyKeyPair \
--maintenance-options AutoRecovery=Disabled \
  [...]
```

実行中または停止中のインスタンスの簡易自動復旧を無効にするには

modify-instance-maintenance-options コマンドを使用します。

```
aws ec2 modify-instance-maintenance-options \
--instance-id i-0abcdef1234567890 \
--auto-recovery disabled
```

実行中または停止中のインスタンスの自動復旧動作を **default** に設定するには

modify-instance-maintenance-options コマンドを使用します。

```
aws ec2 modify-instance-maintenance-options \
--instance-id i-0abcdef1234567890 \
--auto-recovery default
```

Amazon CloudWatch アクションに基づく復旧

インスタンスを復旧するタイミングをカスタマイズする場合は、Amazon CloudWatch アクションに基づく復旧を使用します。

StatusCheckFailed_System アラームがトリガーされ復旧アクションが開始されると、アラームの作成時に選択し復旧アクションに関連付けてある Amazon SNS トピックから、通知が送信されます。復旧アクションが完了すると、その情報が、アラームに設定された Amazon SNS トピックに対し発行されます。この Amazon SNS トピックをサブスクライブしているすべてのユーザーが、復旧処理のステータスと以降の手順が記載された通知を、E メールで受け取ります。復旧アクションの最後のステップとして、復旧されたインスタンスの再起動が行われます。

簡易自動復旧が無効になっていなくても、Amazon CloudWatch アラームを使用してインスタンスを復旧できます。Amazon CloudWatch アラームを作成してインスタンスを復旧する方法の詳細については、「Amazon CloudWatch アラームへの復旧アクションの追加」を参照してください。

サポートされるインスタンスタイプ

簡易自動復旧を利用できるすべてのインスタンスタイプは、Amazon CloudWatch アクションに基づく復旧もサポートします。さらに、CloudWatch アクションベースの復旧は、サポートされているインスタンスタイプのベアメタルバリアントをサポートします。簡易自動復旧でサポートされるインスタンスファミリーに加えて、以下のインスタンスファミリーもサポートされています。

- メモリ最適化: X2idn | X2iedn

> **Important**
>
> インスタンスストアボリュームを備えたサポートされているインスタンスタイプの場合、これらのボリューム上のデータは復旧中に失われます。インスタンスを停止して起動すると、インスタンスストアボリューム上のデータもすべて失われます。インスタンスストアのボリュームデータを、Amazon EBS、Amazon S3、Amazon EFS などのより永続的なストレージに定期的にバックアップする必要があります。システムステータスチェックに失敗した場合は、インスタンスストアボリュームを使用してインスタンスを停止および開始し、バックアップデータを使用してインスタンスストアボリュームを復元できます。

CloudWatch アクションに基づく復旧は、専有ホストテナンシーがあるインスタンスとメタルインスタンスの復旧をサポートしていません。Amazon EC2 Dedicated Host では、Dedicated Host Auto Recovery を使用して、異常のあるインスタンスを自動的に回復できます。

AWS Management Console または AWS CLI を使用して、CloudWatch アクションベースの復旧をサポートするインスタンスタイプを表示できます。

Console

Amazon CloudWatch アクションベースの復旧をサポートするインスタンスタイプを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで、[Instance Types] (インスタンスタイプ) を選択します。
3. フィルターバーに「Auto Recovery support: true」と入力します。あるいは、この文字列を入力していくと該当するフィルター名が表示されるので、そのフィルターを選択できます。

[インスタンスタイプ] テーブルには、Amazon CloudWatch アクションベースの復旧をサポートするすべてのインスタンスタイプが表示されます。

AWS CLI

Amazon CloudWatch アクションベースの復旧をサポートするインスタンスタイプを表示するには

describe-instance-types コマンドを使用します。

```
aws ec2 describe-instance-types --filters Name=auto-recovery-supported,Values=true
--query "InstanceTypes[*].[InstanceType]" --output text | sort
```

インスタンスの復旧の失敗のトラブルシューティング

インスタンスの復旧に失敗する場合、以下のような原因が考えられます。

- Service Health Dashboard イベント中、簡易的な自動リカバリではインスタンスが復旧されない場合があります。これらのイベントが原因で復旧が失敗しても、その通知を受信しない可能性があります。進行中の Service Health Dashboard イベントがある場合、CloudWatch アクションベースのリカバリで、インスタンスが正常に復旧されない可能性があります。サービスの可用性に関する最新情報については、http://status.aws.amazon.com/ を参照してください。
- 代替ハードウェアの一時的な容量不足。
- インスタンスが、1 日に許可されている 3 回の復旧試行回数に達しました。

自動復旧プロセスは、1 日で 3 回まで別個のエラーについてインスタンスの復旧を試みます。インスタンスのシステムステータスチェックの失敗が続く場合は、インスタンスを手動で停止および開始することをお勧めします。インスタンスストアボリュームのデータは、インスタンスの停止に伴って失われます。詳細については、「インスタンスの停止と起動」を参照してください。

自動復旧が失敗し、元のシステムステータスチェックの失敗の根本原因がハードウェアの機能低下であると判断された場合、対象のインスタンスが使用停止になることがあります。

接続

「Linux インスタンス用 Amazon EC2 ユーザーガイド」のこのセクションには、Linux インスタンスを起動した後に Linux インスタンスに接続するのに役立つ情報が記載されています。また、Linux インスタンスを別の AWS リソースに接続するのに役立つ情報も提供します。

Windows インスタンスに接続する方法の詳細については、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「Windows インスタンスに接続する」を参照してください。

トピック

- Linux インスタンスへの接続
- EC2 Instance Connect Endpoint を使用した、パブリック IPv4 アドレスを必要としないインスタンスへの接続
- EC2 インスタンスを AWS リソースに接続する

Linux インスタンスへの接続

Linux インスタンスに接続するには、さまざまな方法があります。一部は、接続元のローカルマシンのオペレーティングシステムによって異なります。EC2 Instance Connect、AWS Systems Manager Session Manager などのその他の機能は変わりません。このセクションでは、Linux インスタンスに接続し、ローカル コンピューターとインスタンス間でファイルを転送する方法について説明します。Windows インスタンスに接続する方法の詳細については、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「Windows インスタンスに接続する」を参照してください。

Linux インスタンスに接続する前に、以下の前提条件を満たしていることを確認してください。

- インスタンスに関する情報を取得する
- プライベートキーを見つけ、許可を設定する
- (オプション) インスタンスのフィンガープリントを取得する

次に、次のオプションのいずれかを選択して Linux インスタンスに接続します。

ローカルオペレーティングシステムに基づく接続オプション

- SSH を使用して Linux または macOS のローカルマシンから接続する
- Windows ローカルマシンから接続する

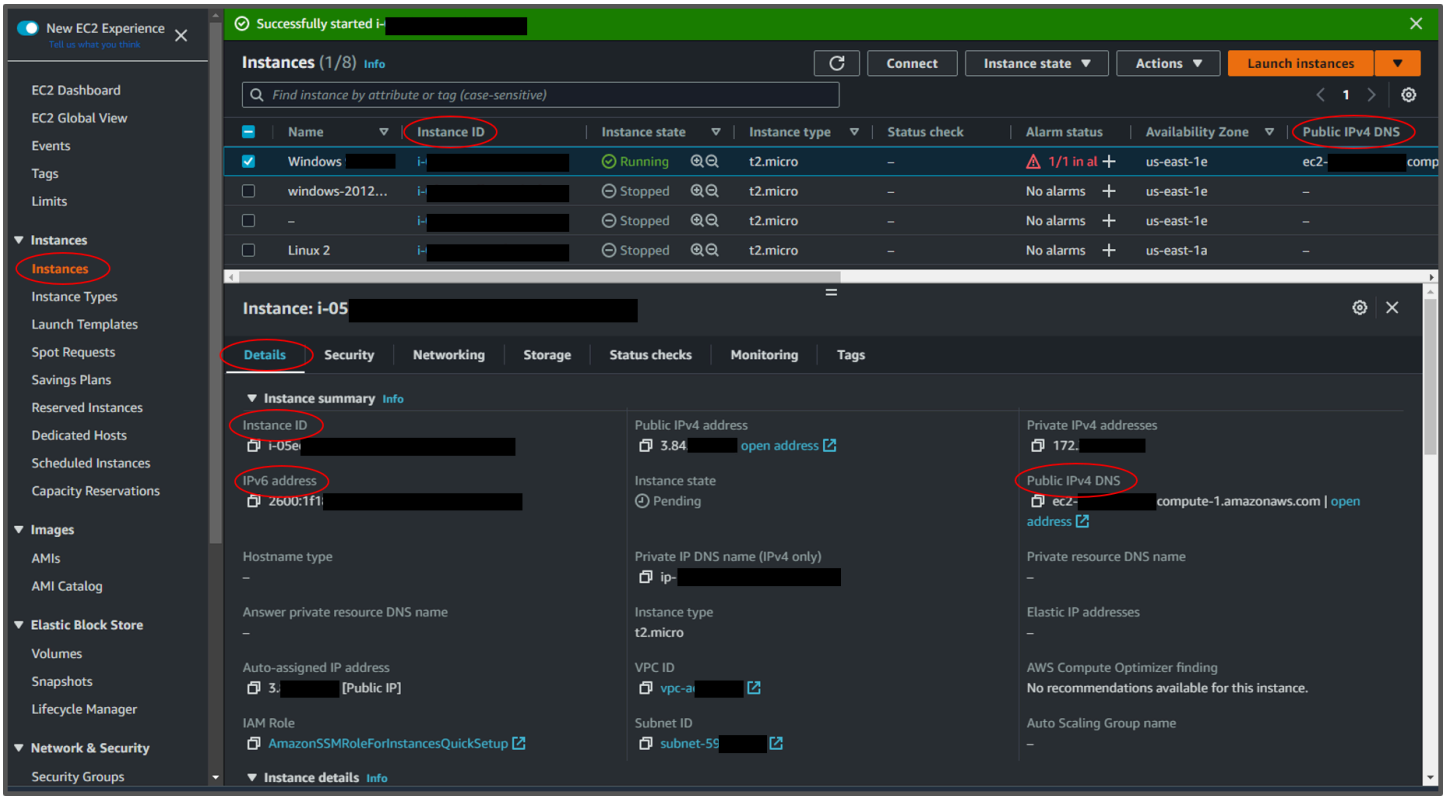
任意のローカルオペレーティングシステムから接続するオプション

- AWS Systems Manager Session Manager を使用して Linux インスタンスに接続する
- EC2 Instance Connect を使用して Linux インスタンスに接続する.

> **Note**
>
> 例えば、接続に関するトラブルシューティングのヒントについては、「インスタンスへの接続に関するトラブルシューティング」を参照してください。
> AWS Nitro System で構築されたインスタンスの起動、ネットワーク設定、および他の問題をトラブルシューティングするには、Linux インスタンス用 EC2 シリアルコンソール を使用できます。

インスタンスに関する情報を取得する

インスタンスに接続する準備をするには、Amazon EC2 コンソールまたは AWS CLI を使用して次の情報を取得します。



- インスタンスのパブリック DNS 名を取得します。

Amazon EC2 コンソールから、インスタンスのパブリック DNS を取得できます。[インスタンス] ペインの [パブリック IPv4 DNS] 列を確認します。この列が非表示になっている場合は、画面右上部にある設定アイコン () を選択し、[パブリック IPv4 DNS] を選択します。パブリック DNS は、[インスタンス] ペインのインスタンス情報セクションにもあります。Amazon EC2 コンソールの [インスタンス] ペインでインスタンスを選択すると、そのインスタンスに関する情報がページの下半分に表示されます。[詳細] タブで、[パブリック IPv4 DNS] を探します。

その代わりに、describe-instances (AWS CLI) または Get-EC2Instance (AWS Tools for Windows PowerShell) コマンドを使用することもできます。

[パブリック IPv4 DNS] が表示されない場合は、[インスタンスの状態] が [実行中] であり、プライベートサブネットでインスタンスを起動していないことを確認します。インスタンス起動ウィザードを使用してインスタンスを起動した場合、[ネットワーク設定] の [パブリック IP の自動割り当て] フィールドを編集して、値を [無効] に変更した可能性があります。[パブリック IP の自動割り当て] オプションを無効にすると、インスタンスは起動時にパブリック IP アドレスが割り当てられません。

- (IPv6 のみ) インスタンスの IPv6 アドレスを取得します。

  自分のインスタンスに IPv6 アドレスを割り当てている場合は、オプションで、パブリック IPv4 アドレスまたはパブリック IPv4 DNS のホスト名の代わりに、IPv6 アドレスを使用してインスタンスに接続することも可能です。ローカルコンピュータに IPv6 アドレスがあり、IPv6 を使用するように設定されている必要があります。Amazon EC2 コンソールから、インスタンスの IPv6 アドレスを取得できます。[インスタンス] ペインの [IPv6 IPs] 列を確認します。または、インスタンス情報セクションで IPv6 アドレスを確認できます。Amazon EC2 コンソールの [インスタンス] ペインでインスタンスを選択すると、そのインスタンスに関する情報がページの下半分に表示されます。[詳細] タブで、[IPv6 アドレス] を探します。

  その代わりに、describe-instances (AWS CLI) または Get-EC2Instance (AWS Tools for Windows PowerShell) コマンドを使用することもできます。IPv6 の詳細については、「IPv6 アドレス」を参照してください。

- インスタンスのユーザー名を取得します。

  インスタンスに接続するには、ユーザーアカウントのユーザー名、またはインスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。

  - ユーザーアカウントのユーザー名を取得します。

ユーザーアカウントの作成方法については、「Linux インスタンスのユーザーアカウントを管理する」を参照してください。

- インスタンスの起動に使用した AMI のデフォルトのユーザー名を取得します。

インスタンスの起動に使用される AMI	デフォルトユーザー名
AL2023 Amazon Linux 2 Amazon Linux	`ec2-user`
CentOS	`centos` または `ec2-user`
Debian	`admin`
Fedora	`fedora`、または `ec2-user`
RHEL	`ec2-user`、または `root`
SUSE	`ec2-user`、または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

プライベートキーを見つけ、許可を設定する

インスタンスに接続するには、プライベートキーファイルの場所を知っている必要があります。SSH 接続の場合、ユーザーのみがファイルを読み込むことができるように許可を設定する必要があります。

Amazon EC2 を使用する際のキーペアの仕組みについては、「Amazon EC2 のキーペアと Amazon EC2 インスタンス」を参照してください。

- [プライベートキーを見つける]

  インスタンスの起動時に指定したキーペアの .pem ファイルの、コンピュータ上の場所への完全修飾パスを取得します。詳細については、「起動時に指定されたパブリックキーの特定」、「」を参照してください。プライベートキーファイルが見つからない場合は、「I've lost my private key.」(プライベートキーを紛失しました) を参照してください。Linux インスタンスに接続するにはどうすればよいですか？

  PuTTY を使用してインスタンスに接続していて、.pem ファイルを .ppk に変換する必要がある場合は、このセクションの「PuTTY を使用して Windows から Linux インスタンスに接続する トピック」の「PuTTYgen を使用してプライベートキーを変換する」を参照してください。
- プライベートキーへの許可を設定し、お客様のみが読み込みできるようにする必要があります
  - macOS または Linux から接続する

    macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は、自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

    ```
    chmod 400 key-pair-name.pem
    ```

    これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については、「エラー: Unprotected Private Key File (保護されていないプライベートキーファイル)」を参照してください。
  - Windows から接続する

    ファイルエクスプローラーを開き、.pem ファイルを右クリックします。[プロパティ] を選択し、[セキュリティ] タブ、[詳細設定] の順に選択します。それから [継承の無効化] を選択します。現在のユーザーを除くすべてのユーザーのアクセスを削除します。

(オプション) インスタンスのフィンガープリントを取得する

中間者攻撃から自身を保護する場合、表示されるフィンガープリントを検証することで、接続しようとしているインスタンスの信頼性を検証できます。フィンガープリントの検証は、サードパーティが提供するパブリック AMI からインスタンスを起動した場合に役立ちます。

タスクの概要

まず、インスタンスのインスタンスフィンガープリントを取得します。次に、インスタンスに接続してフィンガープリントを確認するように求められたら、この手順で取得したフィンガープリントと表示されるフィンガープリントを比較します。これらのフィンガープリントが一致しない場合、何者かが中間者 (MITM) 攻撃を試みている可能性があります。一致する場合には、安心してインスタンスに接続できます。

インスタンスのフィンガープリントを取得するための前提条件

- インスタンスが pending の状態であってはいけません。フィンガープリントは、インスタンスの最初の起動が完了した後にのみ使用できます。
- コンソール出力を取得するには、インスタンス所有者である必要があります。
- インスタンスフィンガープリントを取得するには、さまざまな方法があります。AWS CLI を使用する場合は、ローカルコンピュータにインストールする必要があります。AWS CLI のインストールの詳細については、AWS Command Line Interface ユーザーガイドの「AWS Command Line Interface のインストール」を参照してください。

インスタンスのフィンガープリントを取得するには

ステップ 1 では、インスタンスフィンガープリントを含むコンソール出力を取得します。ステップ 2 では、コンソール出力でインスタンスフィンガープリントを見つけます。

1. コンソール出力を取得するには、以下のいずれかの方法を使用します。

   Console

   1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
   2. 左のナビゲーターの [インスタンス] を選択します。
   3. インスタンスを選択してから、[アクション]、[モニタリングとトラブルシューティング]、[システムログを取得] の順に選択します。

   AWS CLI

   使用するローカルコンピュータ (お客様が接続しているインスタンス上ではなく) で get-console-output (AWS CLI) コマンドを使用します。出力が大きい場合は、出力をテキストファイルにパイプして読み込みやすいようにすることができます。AWS CLI を使用する際は、明示的にまたはデフォルトリージョンを設定して、AWS リージョン を指定する必要が

あることに注意してください。リージョンを設定または指定する方法については、「AWS Command Line Interface ユーザーガイド」の「設定の基本」を参照してください。

```
aws ec2 get-console-output --instance-id instance_id --query Output --output
 text > temp.txt
```

2. コンソール出力で、BEGIN SSH HOST KEY FINGERPRINTS にあるインスタンス (ホスト) フィンガープリントを見つけます。インスタンスフィンガープリントが複数ある場合があります。インスタンスに接続すると、フィンガープリントの 1 つだけが表示されます。

   正確な出力は、オペレーティングシステム、AMI バージョン、AWS でキーペアを作成したかどうかによって異なります。以下は出力例です。

```
ec2:#############################################################
ec2: -----BEGIN SSH HOST KEY FINGERPRINTS-----
ec2: 256 SHA256:l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY no comment (ECDSA)
ec2: 256 SHA256:kpEa+rw/Uq3zxaYZN8KT501iBtJOIdHG52dFi66EEfQ no comment (ED25519)
ec2: 2048 SHA256:L8l6pepcA7iqW/jBecQjVZClUrKY+o2cHLI0iHerbVc no comment (RSA)
ec2: -----END SSH HOST KEY FINGERPRINTS-----
ec2: #############################################################
```

> **Note**
>
> このフィンガープリントは、インスタンスに接続するときに参照します。

SSH を使用して Linux または macOS から Linux インスタンスに接続します。

Secure Shell (SSH) を使用して、Linux または macOS オペレーティングシステムを実行するローカルマシンから Linux インスタンスに接続することも、EC2 Instance Connect や AWS Systems Manager セッションマネージャーなどのプラットフォームに依存しない接続ツールを使用することもできます。プラットフォームに依存しないツールの詳細については、「Linux インスタンスへの接続」を参照してください。

このページでは、SSH クライアントを使用してインスタンスに接続する方法について説明します。Windows から Linux インスタンスに接続するには、「Windows から接続する」を参照してください。

> **Note**
>
> インスタンスに接続しようとしているときにエラーが発生した場合は、インスタンスが SSH 接続の前提条件 のすべてを満たしていることを確認してください。前提条件をすべて満たしているにもかかわらず Linux インスタンスに接続できない場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

コンテンツ

SSH 接続の前提条件

Linux インスタンスに接続する前に、以下の前提条件を満たしていることを確認してください。

インスタンスのステータスの確認

インスタンスを起動してから接続できるようになるまでには、数分かかる場合があります。インスタンスのステータスチェックが成功していることを確認します。この情報は、[Instances (インスタンス)] ページの [Status check (ステータスチェック)] 列で確認できます。

インスタンスに接続するためのパブリック DNS 名とユーザー名の取得

インスタンスのパブリック DNS 名または IP アドレス、およびインスタンスへの接続に使用するユーザー名を確認するには、「インスタンスに関する情報を取得する」を参照してください。

プライベートキーを見つけ、アクセス許可を設定する

インスタンスへの接続に必要なプライベートキーを特定し、キーのアクセス許可を設定するには、「プライベートキーを見つけ、許可を設定する」を参照してください。

必要に応じてローカルコンピュータに SSH クライアントをインストールする

ローカルコンピュータには、デフォルトで SSH クライアントがインストールされている場合があります。これは、コマンドラインに「ssh」と入力することで確認できます。ご使用のコンピュータでこのコマンドが認識されない場合、SSH クライアントをインストールできます。

- インストール可能なコンポーネントとして、最新バージョンの Windows サーバー 2019 と Windows 10 - OpenSSH が含まれています。詳細については、「Windows での OpenSSH」を参照してください。

- 以前のバージョンの Windows - OpenSSH をダウンロードしてインストールします。詳細については、「Win32-OpenSSH」を参照してください。
- Linux および MacOS X - OpenSSH をダウンロードしてインストールします。詳細については、https://www.openssh.com を参照してください。

SSH クライアントを使用して Linux インスタンスに接続する

SSH クライアントを使用して Linux インスタンスに接続するには、次の手順に従います。インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

SSH を使用したインスタンスへの接続

1. ターミナルウィンドウで ssh コマンドを使用して、インスタンスに接続します。プライベートキー (.pem) のパスとファイル名、インスタンスのユーザー名、およびインスタンスのパブリック DNS 名や IPv6 アドレスを指定します。プライベートキー、インスタンスのユーザー名、およびインスタンスの DNS 名や IPv6 アドレスの検索方法の詳細については、「プライベートキーを見つけ、許可を設定する」および「インスタンスに関する情報を取得する」を参照してください。インスタンスに接続するには、次のいずれかのコマンドを使用します。
   - (パブリック DNS) インスタンスのパブリック DNS 名を使用して接続するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@instance-public-dns-name
     ```

   - (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用して接続するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@instance-IPv6-address
     ```

   以下のようなレスポンスが表示されます。

   ```
   The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (198-51-100-1)'
    can't be established.
   ECDSA key fingerprint is l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY.
   Are you sure you want to continue connecting (yes/no)?
   ```

2. (オプション) セキュリティアラートのフィンガープリントが、(オプション) インスタンスのフィンガープリントを取得する で事前に取得したフィンガープリントと一致することを確認しま

す。これらのフィンガープリントが一致しない場合、何者かが中間者 (MITM) 攻撃を試みている可能性があります。一致した場合は、次のステップに進んでください。

3. **yes** と入力します。

   以下のようなレスポンスが表示されます。

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (ECDSA) to
 the list of known hosts.
```

SCP クライアントを使用した Linux インスタンスへのファイルの転送

ローカルコンピュータと Linux インスタンスの間でファイルを転送する方法の 1 つとして、セキュアコピープロトコル (SCP) を使用します。このセクションでは、SCP でファイルを転送する方法について説明します。この手順は、SSH を使用してインスタンスに接続する手順と似ています。

前提条件

- インスタンスにファイルを転送するための一般的な前提条件の確認

  ローカルマシンとインスタンスの間でファイルを転送する前に、次のアクションを実行して、必要な情報がすべて揃っていることを確認してください。
  - インスタンスに関する情報を取得する
  - プライベートキーを見つけ、許可を設定する
  - (オプション) インスタンスのフィンガープリントを取得する
- SCP クライアントのインストール

  ほとんどの Linux、Unix、および Apple コンピュータには、デフォルトで SCP クライアントが含まれています。含まれていない場合は、OpenSSH プロジェクトから、SSH ツールの完全なスイートの無料実装が提供されており、これに SCP クライアントが含まれます。詳細については、https://www.openssh.com を参照してください。

以下では、インスタンスのパブリック DNS 名、またはインスタンスに IPv6 アドレスがある場合は IPv6 アドレスを使用し、SCP でファイルを転送する手順を示します。

SCP を使用してコンピュータとインスタンス間でファイルを転送するには

1. コンピュータ上のソースファイルの場所と、インスタンス上の送信先パスを決定します。以下の例では、プライベートキーファイルの名前が `key-pair-name.pem`、転送するファイルが

my-file.txt、インスタンスのユーザー名が ec2-user、インスタンスのパブリック DNS の名前が instance-public-dns-name で、インスタンスの IPv6 アドレスが instance-IPv6-address です。

- (パブリック DNS) インスタンスの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。

```
scp -i /path/key-pair-name.pem /path/my-file.txt ec2-user@instance-public-dns-name:path/
```

- (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。IPv6 アドレスは、(\) でエスケープした角かっこ ([ ]) で囲む必要があります。

```
scp -i /path/key-pair-name.pem /path/my-file.txt ec2-user@\[instance-IPv6-address\]:path/
```

2. SSH を使用してインスタンスに接続していない場合は、次のようなレスポンスが表示されます。

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (10.254.142.33)'
can't be established.
RSA key fingerprint is 1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```

(オプション) オプションで、セキュリティアラートのフィンガープリントがインスタンスのフィンガープリントと一致することを確認できます。詳細については、(オプション) インスタンスのフィンガープリントを取得する を参照してください。

yes と入力します。

3. 転送が成功した場合、レスポンスは以下のようになります。

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (RSA)
to the list of known hosts.
my-file.txt                                   100%   480     24.4KB/s   00:00
```

4. 逆の方向 (Amazon EC2 インスタンスからコンピュータ) にファイルを転送するには、ホストパラメータの順番を逆にします。例えば、次の例に示すように、EC2 インスタンスからローカルコンピュータの送信先に my-file.txt として my-file2.txt を転送できます。

- (パブリック DNS) コンピュータの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。

```
scp -i /path/key-pair-name.pem ec2-user@instance-public-dns-name:path/my-file.txt path/my-file2.txt
```

- (IPv6) インスタンスに IPv6 アドレスがある場合、コンピュータの送信先にファイルを転送するには、コンピュータから次のコマンドを入力します。IPv6 アドレスは、(\) でエスケープした角かっこ ([ ]) で囲む必要があります。

```
scp -i /path/key-pair-name.pem ec2-user@\[instance-IPv6-address\]:path/my-file.txt path/my-file2.txt
```

Windows から Linux インスタンスに接続する

次の方法を使用して、Windows オペレーティングシステムを搭載したローカルマシンから Linux インスタンスに接続できます。

- OpenSSH
- PuTTY
- Windows Subsystem for Linux

OpenSSH を使用して Windows から Linux インスタンスに接続する

次の手順は、SSH プロトコルを使用したリモートログイン用のオープンソース接続ツールである OpenSSH で、Windows から Linux インスタンスに接続する方法を示しています。OpenSSH は、Windows Server 2019 以降のオペレーティングシステムでサポートされています。

目次

前提条件

OpenSSH を使用して Windows から Linux インスタンスに接続する前に、次の前提条件を完了してください。

インスタンスの準備ができていることを確認する

インスタンスを起動してから接続できるようになるまでには、数分かかる場合があります。インスタンスのステータスチェックが成功していることを確認します。この情報は、[Instances (インスタンス)] ページの [Status check (ステータスチェック)] 列で確認できます。

インスタンスに接続するための一般的な前提条件を確認する

インスタンスのパブリック DNS 名または IP アドレス、およびインスタンスへの接続に使用するユーザー名を確認するには、「インスタンスに関する情報を取得する」を参照してください。

Windows のバージョンを確認する

OpenSSH を使用して Windows から Linux インスタンスに接続するには、Windows バージョンが Windows Server 2019 以降である必要があります。

PowerShell の前提条件を確認する

PowerShell を使用して Windows OS に OpenSSH をインストールするには、PowerShell バージョン 5.1 以降を実行していて、アカウントがビルトインの管理者グループのメンバーである必要があります。PowerShell から `$PSVersionTable.PSVersion` を実行して、PowerShell のバージョンを確認します。

自分がビルトインの管理者グループのメンバーかどうかを確認するには、次の PowerShell コマンドを実行します。

```
(New-Object
 Security.Principal.WindowsPrincipal([Security.Principal.WindowsIdentity]::GetCurrent())).Is
```

ビルトインの管理者グループのメンバーである場合、出力は `True` です。

PowerShell を使用して OpenSSH for Windows をインストールする

PowerShell を使用して OpenSSH for Windows をインストールするには、次の PowerShell コマンドを実行します。

```
Add-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0
```

正常な出力:

```
Path          :
Online        : True
RestartNeeded : False
```

OpenSSH を使用して Windows から Linux インスタンスに接続する

OpenSSH をインストールしたら、次の手順に従って、OpenSSH を使用して Windows から Linux インスタンスに接続します。インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

OpenSSH を使用してインスタンスに接続するには

1. PowerShell またはコマンドプロンプトで、ssh コマンドを使用してインスタンスに接続します。プライベートキー (.pem) のパスとファイル名、インスタンスのユーザー名、およびインスタンスのパブリック DNS 名または IPv6 アドレスを指定します。プライベートキー、インスタンスのユーザー名、およびインスタンスの DNS 名や IPv6 アドレスの検索方法の詳細については、「プライベートキーを見つけ、許可を設定する」および「インスタンスに関する情報を取得する」を参照してください。インスタンスに接続するには、次のいずれかのコマンドを使用します。
   - (パブリック DNS) インスタンスのパブリック DNS 名を使用して接続するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@instance-public-dns-name
     ```
   - (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用して接続するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@instance-IPv6-address
     ```

   以下のようなレスポンスが表示されます。

   ```
   The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (198-51-100-1)'
    can't be established.
   ECDSA key fingerprint is l4UB/neBad9tvkgJf1QZWxheQmR59WgrgzEimCG6kZY.
   Are you sure you want to continue connecting (yes/no/[fingerprint])?
   ```
2. (オプション) セキュリティアラートのフィンガープリントが、(オプション) インスタンスのフィンガープリントを取得する で事前に取得したフィンガープリントと一致することを確認しま

す。これらのフィンガープリントが一致しない場合、何者かが中間者 (MITM) 攻撃を試みている可能性があります。一致した場合は、次のステップに進んでください。

3. **yes** と入力します。

   以下のようなレスポンスが表示されます。

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (ECDSA) to
 the list of known hosts.
```

PowerShell を使用して Windows から OpenSSH をアンインストールする

PowerShell を使用して Windows から OpenSSH をアンインストールするには、次の PowerShell コマンドを実行します。

```
Remove-WindowsCapability -Online -Name OpenSSH.Client~~~~0.0.1.0
```

正常な出力:

```
Path          :
Online        : True
RestartNeeded : True
```

PuTTY を使用して Windows から Linux インスタンスに接続する

Windows Server 2019 以降を実行している場合は、SSH プロトコルを使用したリモートログイン用のオープンソース接続ツールである OpenSSH の使用をお勧めします。OpenSSH を使用して Windows から Linux インスタンスに接続するステップについては、「OpenSSH を使用して Windows から Linux インスタンスに接続する」を参照してください。

次の手順では、Windows 用の無料の SSH クライアントである PuTTY を使用して、インスタンスに接続する方法について説明します。インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

目次

- PuTTY Secure Copy Client を使用した Linux インスタンスへのファイルの転送
- WinSCP を使用した Linux インスタンスへのファイルの転送

前提条件

PuTTY を使用して Linux インスタンスに接続する前に、以下の前提条件を満たしていることを確認してください。

インスタンスの準備ができていることを確認する

インスタンスを起動してから接続できるようになるまでには、数分かかる場合があります。インスタンスのステータスチェックが成功していることを確認します。この情報は、[Instances (インスタンス)] ページの [Status check (ステータスチェック)] 列で確認できます。

インスタンスに接続するための一般的な前提条件を確認する

インスタンスのパブリック DNS 名または IP アドレス、およびインスタンスへの接続に使用するユーザー名を確認するには、「インスタンスに関する情報を取得する」を参照してください。

ローカルコンピュータに PuTTY をインストールする

PuTTY のダウンロードページから、PuTTY をダウンロードしてインストールします。既にインストールされている旧バージョンの PuTTY がある場合は、最新バージョンをダウンロードすることをお勧めします。必ずスイート全体をインストールします。

PuTTYgen を使用してプライベート .pem キーを .ppk に変換する

インスタンスの起動時に指定したキーペアの場合、.pem 形式でプライベートキーを作成する場合は、PuTTY で使用できるように .ppk ファイルに変換する必要があります。プライベート.pem ファイルを検索し、次のセクションのステップに従ってください。

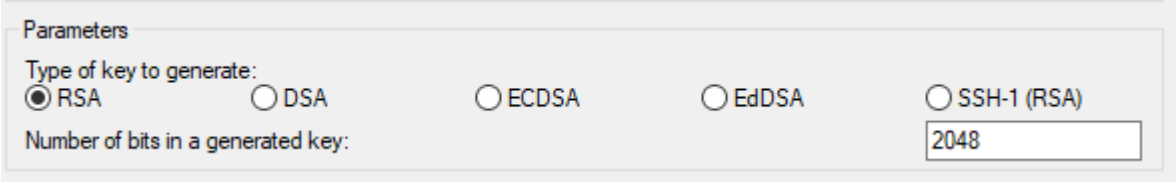
PuTTYgen を使用してプライベートキーを変換する

PuTTY は、SSH キーの PEM 形式をネイティブにサポートしていません。PuTTY には、PEM キーを PuTTY が必要とする PPK 形式に変換する PuTTYgen というツールが用意されています。PuTTY を使用してインスタンスに接続するには、プライベートキー (.pemファイル) を次の形式 (.ppkファイル) に変換する必要があります。

プライベート .pem キーを .ppk 形式に変換するには

1. [スタート] メニューで、[すべてのプログラム]、[PuTTY]、[PuTTYgen] の順に選択します。

2. [Type of key to generate (生成するキーのタイプ)] で、[RSA] を選択します。お使いの PuTTYgen のバージョンにこのオプションが含まれていない場合は、[SSH-2 RSA] を選択します。



3. ロード] を選択します。PuTTYgen では、デフォルトでは .ppk 拡張子を持つファイルだけが表示されます。.pem ファイルの場所を特定するには、すべてのタイプのファイルを表示するオプションを選択します。

4. インスタンスの起動時に指定したキーペアの .pem ファイルを選択し、[開く] を選択します。PuTTYgen により、.pem ファイルが正常にインポートされたことが表示されます。[OK] を選択します。

5. プライベートキーを PuTTY で使用できる形式で保存するには、[プライベートキーを保存] を選択します。PuTTYgen に、パスフレーズなしでキーを保存することに関する警告が表示されます。[Yes] を選択します。

> **Note**
>
> プライベートキーのパスフレーズは追加の保護レイヤーです。プライベートキーが検出されても、パスフレーズがなければ使用できません。パスフレーズを使用することの欠点は、インスタンスにログオンしたり、ファイルをインスタンスにコピーしたりするのに人間の介入が必要となるため、オートメーションが難しくなることです。

6. キーペアに使用した名前と同じ名前 (key-pair-name など) をキーに指定し、[保存] を選択します。PuTTY により、.ppk ファイルに拡張子が自動的に追加されます。

プライベートキーが PuTTY で使用するための正しい形式となりました。これで、PuTTY の SSH クライアントを使用してインスタンスに接続することができます。

Linux インスタンスへの接続

PuTTY を使用して Linux インスタンスに接続するには、次の手順に従います。秘密キーに作成した .ppk ファイルが必要になります。詳細については、前のセクションの「PuTTYgen を使用してプライベートキーを変換する」を参照してください。インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

PuTTY の最後にテストされたバージョン:.78

PuTTY を使用してインスタンスに接続するには

1. PuTTY を起動します ([スタート] メニューから [PuTTY] を検索し、[開く] を選択します)。
2. [Category (カテゴリ)] ペインで [Session (セッション)] を選択し、次のフィールドに入力します。

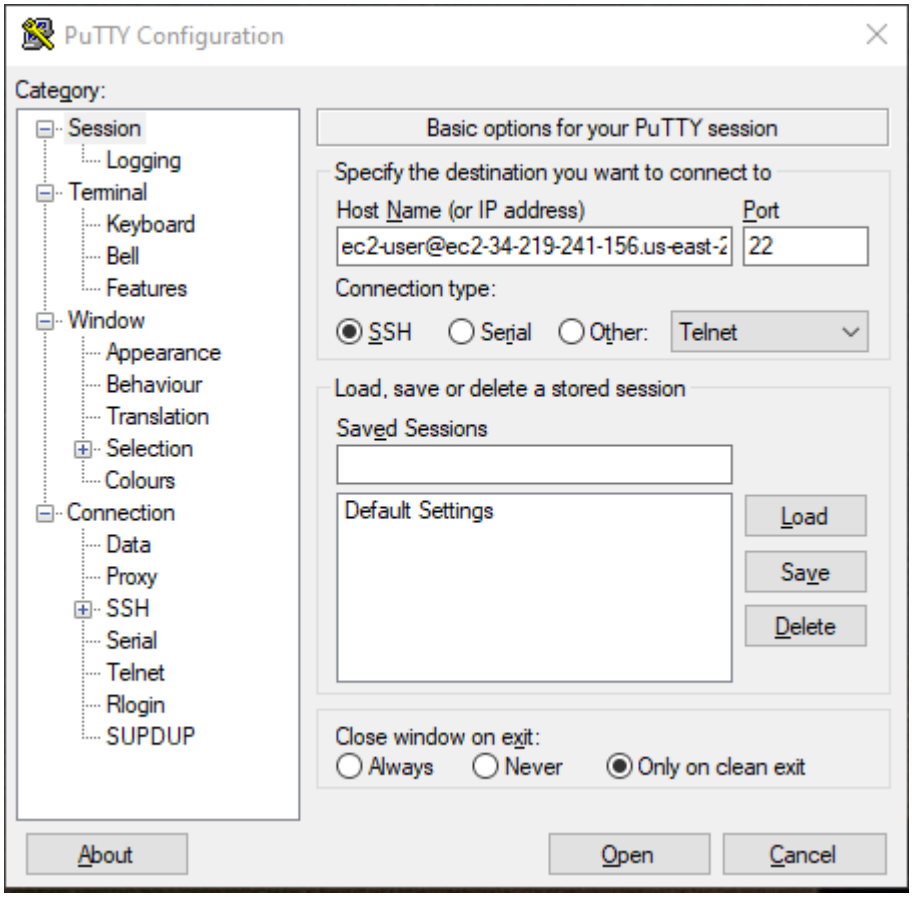
   a. [Host Name (ホスト名)] ボックスで、次のいずれかの操作を行います。

      - (パブリック DNS) インスタンスのパブリック DNS 名を使用して接続するには、*instance-user-name*@*instance-public-dns-name* と入力します。
      - (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用して接続するには、*instance-user-name*@*instance-IPv6-address* と入力します。

      インスタンスのユーザー名、およびインスタンスのパブリック DNS 名または IPv6 アドレスを取得する方法については、「インスタンスに関する情報を取得する」を参照してください。

   b. [Port (ポート)] の値が 22 であることを確認します。

   c. [Connection type (接続タイプ)] で [SSH] を選択します。



3. (オプション) セッションをアクティブに保つため、定期的に「キープアライブ」データを自動的に送信するように PuTTY を設定できます。これは、セッションがアイドル状態になった際にインスタンスから切断されないようにするのに便利です。[カテゴリー] ペインで [接続] を選択し、[キープアライブ間の秒数] フィールドで必要な間隔を入力します。例えば、10 分間アイドル状態が続いた後にセッションが切断される場合、180 と入力して PuTTY を設定し、キープアライブデータを 3 分ごとに送信するようにします。
4. [カテゴリー] ペインで、[接続]、[SSH] の順に展開し、[Auth] を選択します。[認証情報] を選択します。
5. [認証用プライベートキーファイル] の横にある [参照] を選択します。[プライベートキーファイルの選択] ダイアログで、.ppkキーペア用に生成したファイルを選択します。ファイルをダブルクリックするか、[プライベートキーファイルの選択] ダイアログで [開く] を選択します。
6. (オプション) このセッションの後にインスタンスに再度接続する場合は、今後使用するためにセッション情報を保存できます。[カテゴリー] ペインで、[セッション] を選択します。[保存されたセッション] にセッションの名前を入力し、[保存] を選択します。
7. インスタンスに接続するには、[開く] を選択します。

8. このインスタンスに接続するのが初めての場合、PuTTY は接続先のホストを信頼するかどうかを確認するセキュリティアラートダイアログボックスを表示します。

   a. (オプション) セキュリティアラートダイアログボックスのフィンガープリントが、(オプション) インスタンスのフィンガープリントを取得する で前に取得したフィンガープリントと一致することを確認します。これらのフィンガープリントが一致しない場合、「中間者 (MITM) 」攻撃を受けている可能性があります。一致した場合は、次のステップに進んでください。

   b. [Accept (承諾)] を選択します。ウィンドウが開き、インスタンスに接続した状態になります。

   > **Note**
   >
   > プライベートキーを PuTTY フォーマットに変換するときにパスフレーズを指定した場合は、インスタンスにログインする際にそのパスフレーズを指定する必要があります。

インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

PuTTY Secure Copy Client を使用した Linux インスタンスへのファイルの転送

PuTTY Secure Copy Client (PSCP) は、Windows コンピュータと Linux インスタンスの間でファイルを転送するために使用できるコマンドラインツールです。グラフィカルユーザーインターフェイス (GUI) を使用する場合は、WinSCP という名前のオープンソース GUI ツールを使用できます。詳細については、WinSCP を使用した Linux インスタンスへのファイルの転送 を参照してください。

PSCP を使用するには、「PuTTYgen を使用してプライベートキーを変換する」で生成したプライベートキーが必要です。Linux インスタンスのパブリック DNS 名、またはインスタンスに IPv6 アドレスがある場合は IPv6 アドレスも必要です。

次の例では、ファイル `Sample_file.txt` を Windows コンピュータの C:\ ドライブから Amazon Linux インスタンス上の `instance-user-name` ホームディレクトリに転送します。ファイルを転送するには、次のいずれかのコマンドを使用します。

- (パブリック DNS) インスタンスのパブリック DNS 名を使用してファイルを転送するには、次のコマンドを入力します。

```
pscp -i C:\path\my-key-pair.ppk C:\path\Sample_file.txt instance-user-name@instance-public-dns-name:/home/instance-user-name/Sample_file.txt
```

- (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用してファイルを転送するには、次のコマンドを入力します。IPv6 アドレスは角かっこ ([ ]) で囲む必要があります。

```
pscp -i C:\path\my-key-pair.ppk C:\path\Sample_file.txt instance-user-name@[instance-IPv6-address]:/home/instance-user-name/Sample_file.txt
```

WinSCP を使用した Linux インスタンスへのファイルの転送

WinSCP は Windows 用の GUI ベースのファイルマネージャで、SFTP、SCP、FTP、および FTPS プロトコルを使って、ファイルをリモートコンピュータにアップロードおよび転送することができます。WinSCP を使用すると、Windows コンピュータから Linux インスタンスにファイルをドラッグアンドドロップしたり、2 つのシステム間でディレクトリ構造全体を同期させることができます。

要件

- PuTTYgen を使用してプライベートキーを変換する で生成したプライベートキーが必要です。
- Linux インスタンスのパブリック DNS 名が必要です。
- Linux インスタンスに scp がインストールされている必要があります。一部のオペレーティングシステムでは、openssh-clients パッケージをインストールします。Amazon ECS に最適化された AMI など、その他の場合は、scp パッケージをインストールします。お使いの Linux ディストリビューションのドキュメントを確認してください。

WinSCP を使用してインスタンスに接続するには

1. http://winscp.net/eng/download.php から WinSCP をダウンロードしてインストールします。ほとんどの場合、デフォルトのインストールオプションでかまいません。
2. WinSCP を起動します。
3. [WinSCP login (WinSCP ログイン)] 画面の [ホスト名] に、次のいずれかを入力します。
   - (パブリック DNS または IPv4 アドレス) インスタンスのパブリック DNS 名またはパブリック IPv4 アドレスを使用してログインするには、インスタンスのパブリック DNS 名またはパブリック IPv4 アドレスを入力します。

- (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用してログインするには、インスタンスの IPv6 アドレスを入力します。

4. [ユーザー名] については、AMI のデフォルトユーザー名を入力します。

   - AL2023、Amazon Linux 2 または Amazon Linux AMI の場合、ユーザー名は `ec2-user` です。
   - Centos AMI の場合、ユーザー名は `centos` または `ec2-user`です。
   - Debian AMI の場合は、ユーザー名は `admin` です。
   - Fedora AMI の場合、ユーザー名は `fedora` または `ec2-user` です。
   - RHEL AMI の場合、ユーザー名は `ec2-user` または `root` です。
   - SUSE AMI の場合、ユーザー名は `ec2-user` または `root` です。
   - Ubuntu AMI の場合、ユーザー名は `ubuntu` です。
   - SUSE AMI の場合、ユーザー名は `ec2-user` です。
   - Bitnami AMI の場合は、ユーザー名は `bitnami` です。

   > **Note**
   >
   > 他の Linux ディストリビューションのデフォルトのユーザー名を確認するには、AMI プロバイダーに確認してください。

5. インスタンスのプライベートキーファイルを指定します。

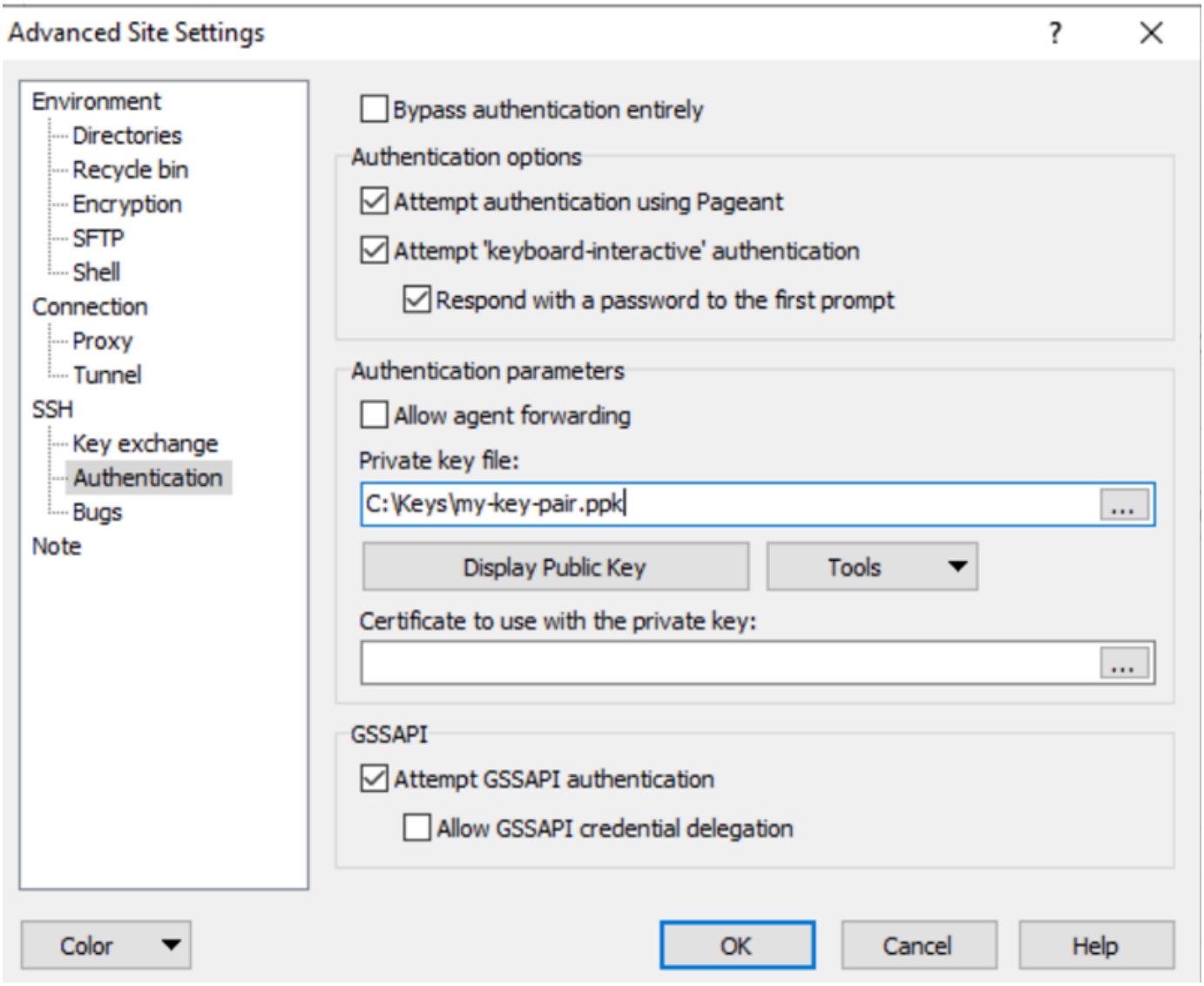
   a. [アドバンスト...] ボタンを選択します。

   b. SSH で、[認証] を選択します。

   c. プライベートキーファイルのパスを指定するか、[...] ボタンを選択して、キーペアファイルを参照します。

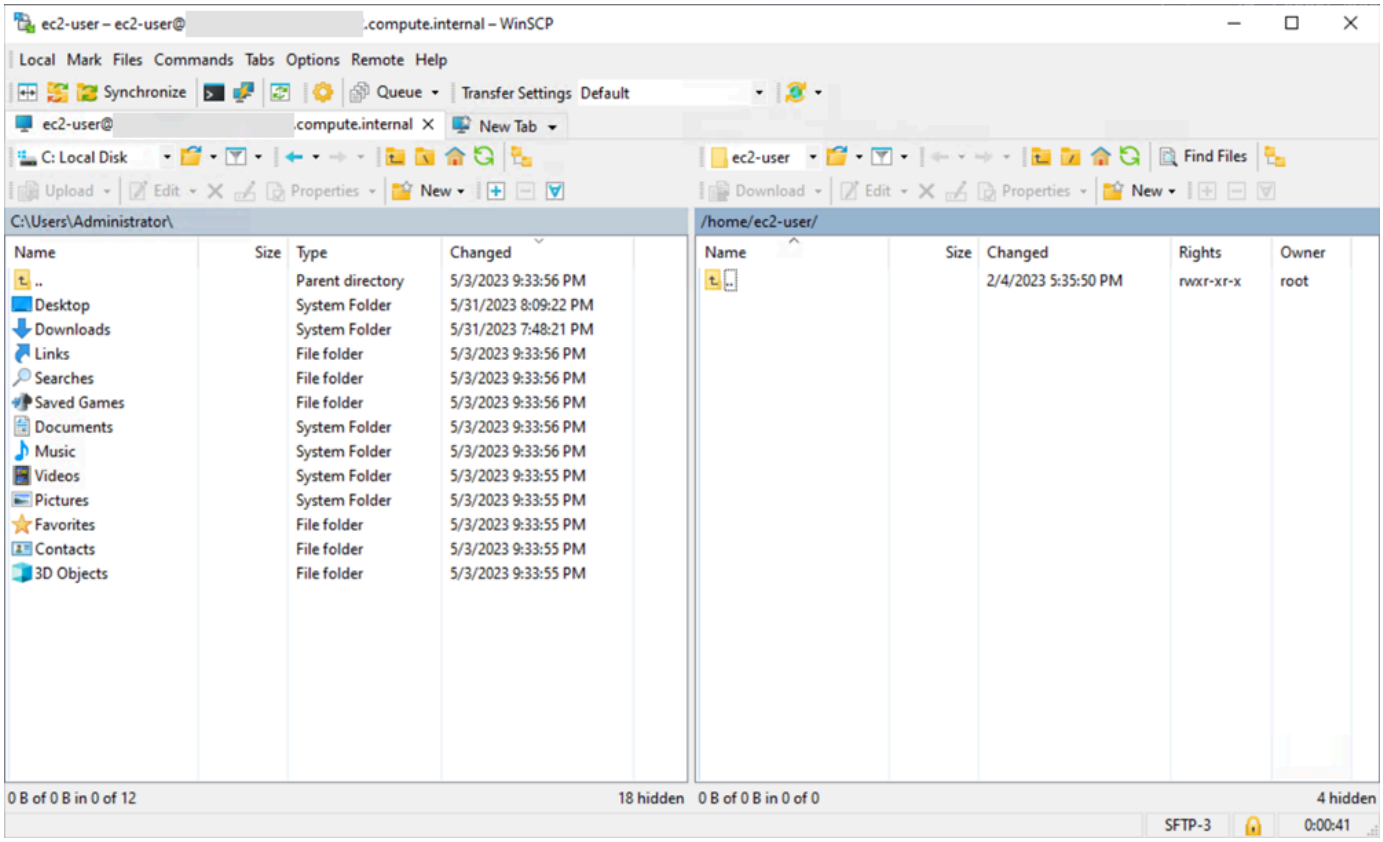
   d. [OK] を選択します。

   次に示すのは、WinSCP バージョン 6.1 のスクリーンショットです。



WinSCP は PuTTY プライベートキーファイル (.ppk) ファイルを必要とします。PuTTYgen を使用して、.pem セキュリティキーファイルを .ppk フォーマットに変換することができます。詳細については、PuTTYgen を使用してプライベートキーを変換する を参照してください。

6. (オプション) 左のパネルで、[ディレクトリ] を選択します。[リモートディレクトリ] に、ファイルを追加する先のディレクトリのパスを入力します。より新しいバージョンの WinSCP で高度なサイトの設定を開くには、[設定] を選択します。リモートディレクトリ設定を見つけるには、[環境] の [ディレクトリ] を選択します。
7. [ログイン] を選択します。ホストのフィンガープリントをホストのキャッシュに追加するには、[はい] を選択します。



8. 接続確立後、接続ウィンドウには Linux インスタンスが右側、ローカルマシンが左側に表示されます。リモートファイルシステムとローカルマシンの間でファイルをドラッグアンドドロップできます。WinSCP の詳細については、http://winscp.net/eng/docs/start のドキュメントを参照してください。

   SCP を実行して転送を開始できないというエラーが表示された場合は、Linux インスタンスに scp がインストールされていることを確認します。

Windows Subsystem for Linux (WSL) を使用して Windows から Linux インスタンスに接続する

インスタンスを起動したら、これに接続し、普通のコンピュータと同じように使用できます。

次の手順では、Windows Subsystem for Linux (WSL) で Linux ディストリビューションを使用してインスタンスに接続する方法について説明します。WSL は無料でダウンロードでき、Windows でネイティブ Linux コマンドラインツールを直接実行できます。それと同時に、仮想マシンのオーバーヘッドなしに、従来の Windows デスクトップも実行できます。

WSL をインストールすると、PuTTY または PuTTYgen を使用する代わりに、ネイティブ Linux 環境を使用して Linux EC2 インスタンスに接続できます。Linux 環境では、Linux インスタンスにより簡単に接続できます。これは、Linux インスタンスに接続し、.pem キーファイルのアクセス権限を

変更するために使用できるネイティブ SSH クライアントが付属しているためです。Amazon EC2 コンソールは、Linux インスタンスに接続するための SSH コマンドを提供します。この SSH コマンドから、トラブルシューティングのために詳細な出力を取得できます。詳細については、Windows Subsystem for Linux に関する情報を参照してください。

> **Note**
>
> WSL をインストールした後のすべての必須条件とステップは、「SSH を使用して Linux または macOS から Linux インスタンスに接続します。」で説明しているものと同じです。また、そのエクスペリエンスはネイティブ Linux の使用と同様です。

インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

目次

前提条件

Linux インスタンスに接続する前に、以下の前提条件を満たしていることを確認してください。

インスタンスの準備ができていることを確認する

インスタンスを起動してから接続できるようになるまでには、数分かかる場合があります。インスタンスのステータスチェックが成功していることを確認します。この情報は、[Instances (インスタンス)] ページの [Status check (ステータスチェック)] 列で確認できます。

インスタンスに接続するための一般的な前提条件を確認する

インスタンスのパブリック DNS 名または IP アドレス、およびインスタンスへの接続に使用するユーザー名を確認するには、「インスタンスに関する情報を取得する」を参照してください。

ローカルコンピュータに Windows Subsystem for Linux (WSL) と Linux ディストリビューションをインストールする

Windows 10 インストールガイドの手順を使用して、WSL と Linux ディストリビューションをインストールします。手順の例では、Linux の Ubuntu ディストリビューションをインストールしますが、任意のディストリビューションをインストールできます。コンピュータを再起動して変更を有効にすることが求められます。

プライベートキーを Windows から WSL にコピーする

WSL ターミナルウィンドウで、Windows から WSL に .pem ファイル (インスタンスの起動時に指定したキーペアの場合) をコピーします。インスタンスに接続する際に使用する、WSL の .pem ファイルへの完全修飾パスをメモします。Windows ハードドライブへのパスを指定する方法の詳細については、「C ドライブにアクセスする方法」を参照してください。キーペアと Windows インスタンスの詳細については、「Amazon EC2 キーペアと Windows インスタンス」を参照してください。

```
cp /mnt/<Windows drive letter>/path/my-key-pair.pem ~/WSL-path/my-key-pair.pem
```

WSL を使用して Linux インスタンスに接続します。

Windows Subsystem for Linux (WSL) を使用して Linux インスタンスに接続するには、次の手順に従います。インスタンスの接続でエラーが発生した場合は、「インスタンスへの接続に関するトラブルシューティング」を参照してください。

SSH を使用してインスタンスに接続するには

1. ターミナルウィンドウで ssh コマンドを使用して、インスタンスに接続します。プライベートキー (.pem) のパスとファイル名、インスタンスのユーザー名、およびインスタンスのパブリック DNS 名や IPv6 アドレスを指定します。プライベートキー、インスタンスのユーザー名、およびインスタンスの DNS 名や IPv6 アドレスの検索方法の詳細については、「プライベートキーを見つけ、許可を設定する」および「インスタンスに関する情報を取得する」を参照してください。インスタンスに接続するには、次のいずれかのコマンドを使用します。
   - (パブリック DNS) インスタンスのパブリック DNS 名を使用して接続するには、次のコマンドを入力します。

     ```
     ssh -i /path/key-pair-name.pem instance-user-name@my-instance-public-dns-name
     ```

- (IPv6) インスタンスに IPv6 アドレスがある場合は、その IPv6 アドレスを使用してインスタンスに接続できます。ssh コマンドで、プライベートキー (.pem) ファイルへのパス、適切なユーザー名、および IPv6 アドレスを指定します。

```
ssh -i /path/key-pair-name.pem instance-user-name@my-instance-IPv6-address
```

以下のようなレスポンスが表示されます。

```
The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (10.254.142.33)'
can't be established.
RSA key fingerprint is 1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
Are you sure you want to continue connecting (yes/no)?
```

2. (オプション) セキュリティアラートのフィンガープリントが、(オプション) インスタンスのフィンガープリントを取得する で事前に取得したフィンガープリントと一致することを確認します。これらのフィンガープリントが一致しない場合、「中間者 (MITM) 」攻撃を受けている可能性があります。一致した場合は、次のステップに進んでください。
3. yes と入力します。

   以下のようなレスポンスが表示されます。

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (RSA)
to the list of known hosts.
```

SCP を使用した Linux から Linux インスタンスへのファイルの転送

ローカルコンピュータと Linux インスタンスの間でファイルを転送する方法の 1 つとして、セキュアコピープロトコル (SCP) を使用します。このセクションでは、SCP でファイルを転送する方法について説明します。この手順は、SSH を使用してインスタンスに接続する手順と似ています。

前提条件

- インスタンスにファイルを転送するための一般的な前提条件の確認

  ローカルマシンとインスタンスの間でファイルを転送する前に、次のアクションを実行して、必要な情報がすべて揃っていることを確認してください。

  - インスタンスに関する情報を取得する
  - プライベートキーを見つけ、許可を設定する

  - (オプション) インスタンスのフィンガープリントを取得する
- SCP クライアントのインストール

  ほとんどの Linux、Unix、および Apple コンピュータには、デフォルトで SCP クライアントが含まれています。含まれていない場合は、OpenSSH プロジェクトから、SSH ツールの完全なスイートの無料実装が提供されており、これに SCP クライアントが含まれます。詳細については、https://www.openssh.com を参照してください。

SCP を使用してファイルを転送するステップを次に示します。既に SSH でインスタンスに接続し、フィンガープリントの確認が完了している場合は、SCP コマンドを実行するステップ (ステップ4) から開始できます。

SCP を使用してファイルを転送するには

1. インスタンスのパブリック DNS 名を使って、インスタンスにファイルを転送します。例えば、プライベートキーファイルの名前が `key-pair-name`、転送するファイルが `SampleFile.txt`、ユーザー名が `instance-user-name`、インスタンスのパブリック DNS の名前が `my-instance-public-dns-name`、または IPv6 アドレスが `my-instance-IPv6-address` の場合、次のコマンドを使ってファイルを `instance-user-name` ホームディレクトリにコピーします。
   - (パブリック DNS) インスタンスのパブリック DNS 名を使用してファイルを転送するには、次のコマンドを入力します。

     ```
     scp -i /path/key-pair-name.pem /path/SampleFile.txt instance-user-name@my-instance-public-dns-name:~
     ```
   - (IPv6) インスタンスに IPv6 アドレスがある場合は、インスタンスの IPv6 アドレスを使用してファイルを転送することができます。IPv6 アドレスは、(\) でエスケープした角かっこ ([ ]) で囲む必要があります。

     ```
     scp -i /path/key-pair-name.pem /path/SampleFile.txt instance-user-name@\[my-instance-IPv6-address\]:~
     ```

   以下のようなレスポンスが表示されます。

   ```
   The authenticity of host 'ec2-198-51-100-1.compute-1.amazonaws.com (10.254.142.33)' can't be established.
   RSA key fingerprint is 1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f.
   ```

```
Are you sure you want to continue connecting (yes/no)?
```

2. (オプション) セキュリティアラートのフィンガープリントが、(オプション) インスタンスのフィンガープリントを取得する で事前に取得したフィンガープリントと一致することを確認します。これらのフィンガープリントが一致しない場合、「中間者 (MITM) 」攻撃を受けている可能性があります。一致した場合は、次のステップに進んでください。
3. **yes** と入力します。

   以下のようなレスポンスが表示されます。

```
Warning: Permanently added 'ec2-198-51-100-1.compute-1.amazonaws.com' (RSA)
to the list of known hosts.
Sending file modes: C0644 20 SampleFile.txt
Sink: C0644 20 SampleFile.txt
SampleFile.txt                                100%   20     0.0KB/s   00:00
```

   [bash: scp: command not found] エラーを受け取った場合は、まず Linux インスタンスに scp をインストールする必要があります。一部のオペレーティングシステムでは、これは openssh-clientsパッケージに含まれます。Amazon Linux-optimized Amazon ECS などの AMI バリアントでは、以下のコマンドを使用して scp をインストールします。

```
[ec2-user ~]$ sudo yum install -y openssh-clients
```

4. 逆の方向 (Amazon EC2 インスタンスからローカルコンピュータに) にファイルを転送する場合は、ホストパラメータの順番を逆にします。例えば、SampleFile.txt ファイルを EC2 インスタンスからローカルコンピュータのホームディレクトリに SampleFile2.txt として転送するには、ローカルコンピュータで次のコマンドのうち 1 つを実行します。
   - (パブリック DNS) インスタンスのパブリック DNS 名を使用してファイルを転送するには、次のコマンドを入力します。

```
scp -i /path/key-pair-name.pem instance-user-
name@ec2-198-51-100-1.compute-1.amazonaws.com:~/SampleFile.txt ~/
SampleFile2.txt
```

   - (IPv6) インスタンスに IPv6 アドレスがある場合、インスタンスの IPv6 アドレスを使用して別の方向にファイルを転送するには、次のコマンドを入力します。

```
scp -i /path/key-pair-name.pem instance-user-name@
\[2001:db8:1234:1a00:9691:9503:25ad:1761\]:~/SampleFile.txt ~/SampleFile2.txt
```

WSL のアンインストール

Windows Subsystem for Linux のアンインストールの詳細については、「WSL ディストリビューションをアンインストールする方法」について参照してください。

EC2 Instance Connect を使用して Linux インスタンスに接続する

Amazon EC2 Instance Connect は、Secure Shell (SSH) を使用して Linux インスタンスに接続するシンプルで安全な方法を提供します。EC2 Instance Connect では、AWS Identity and Access Management (IAM) ポリシーおよびプリンシパルを使用して、インスタンスへの SSH によるアクセスをコントロールします。SSH キーを共有および管理する必要はありません。EC2 Instance Connect を使用したすべての接続リクエストは、AWS CloudTrail に口グとして記録されるため、接続リクエストを監査できます。

EC2 Instance Connect を使用して、Amazon EC2 コンソールまたは任意の SSH クライアントによりインスタンスに接続できます。

EC2 Instance Connect を使用してインスタンスに接続すると、Instance Connect API から SSH パブリックキーがインスタンスメタデータにプッシュされ、60 秒間保持されます。ユーザーにアタッチされた IAM ポリシーにより、ユーザーはパブリックキーをインスタンスメタデータにプッシュすることを許可されます。SSH デーモンは、Instance Connect のインストール時に設定される `AuthorizedKeysCommand` および `AuthorizedKeysCommandUser` により、インスタンスメタデータからパブリックキーを見つけて認証を行い、ユーザーをインスタンスに接続します。

EC2 Instance Connect を使用して、パブリック IP アドレスまたはプライベート IP アドレスを持つインスタンスに接続できます。詳細については、「EC2 Instance Connect を使用して接続」を参照してください。

EC2 Instance Connect を使用して拠点ホストのセキュリティを強化する方法に関して説明しているブログ投稿については、「Amazon EC2 Instance Connect を使用した拠点ホストの保護」を参照してください。

> **Tip**
>
> EC2 Instance Connect は Linux インスタンスに接続するためのオプションの 1 つです。他のオプションについては、「Linux インスタンスへの接続」を参照してください。Windows インスタンスに接続する必要がある場合は、 の「Windows インスタンスへの接続」を参照してください。

コンテンツ

前提条件

EC2 Instance Connect をインストールし、EC2 Instance Connect を使用してインスタンスに接続するための前提条件は次のとおりです。

- AWS リージョン
- ローカルゾーン
- AMI
- EC2 Instance Connect のアンインストール
- IPv4 アドレス
- ネットワークアクセス
- セキュリティグループルール
- 許可を付与する
- ローカルコンピュータのセットアップ
- ユーザーネーム

AWS リージョン

カナダ西部 (カルガリー) を除くすべての AWS リージョン でサポートされています。

ローカルゾーン

サポート外。

AMI

EC2 Instance Connect は以下の AMI にプリインストールされています。

- AL2023

- Amazon Linux 2 2.0.20190618 以降
- macOS Sonoma 14.2.1 以降
- macOS Ventura 13.6.3 以降
- macOS Monterey 12.7.2 以降
- Ubuntu 20.04 以降

以下の AMI を使用して起動されたインスタンスに EC2 Instance Connect をインストールできます。

- バージョン 2.0.20190618 より前の Amazon Linux 2
- CentOS Stream 8 および 9
- 14.2.1 より前の macOS Sonoma、13.6.3 より前の Ventura、12.7.2 より前の Monterey
- Red Hat Enterprise Linux (RHEL) 8 および 9
- Ubuntu 16.04 または 18.04

EC2 Instance Connect のアンインストール

EC2 Instance Connect を使用してインスタンスに接続するには、インスタンスに EC2 Instance Connect がインストールされている必要があります。EC2 Instance Connect にプリインストールされている AMI を使用してインスタンスを起動することも、サポートされている AMI で起動されたインスタンスに EC2 Instance Connect をインストールすることもできます。サポートされている AMI については、前のセクションを参照してください。インストール手順については、「EC2 インスタンスでの EC2 Instance Connect のインストール」を参照してください。

IPv4 アドレス

インスタンスには IPv4 アドレス (プライベートまたはパブリックのいずれか) が必要です。EC2 Instance Connect は IPv6 アドレスを使用した接続をサポートしていません。

ネットワークアクセス

インスタンスは、ユーザーがインターネットまたはインスタンスのプライベート IP アドレスを介してインスタンスに接続できるように設定できます。ユーザーが EC2 Instance Connect を使用してインスタンスに接続する方法によって、次のネットワークアクセスを設定する必要があります。

- ユーザーがインターネット経由でインスタンスに接続する場合、インスタンスにパブリック IP アドレスがあり、インスタンスがパブリックサブネット内にある必要があります。詳細については、Amazon VPC ユーザーガイドのインターネットアクセスを有効にするを参照してください。

- ユーザーがインスタンスのプライベート IP アドレスを介してインスタンスに接続する場合は、AWS Direct Connect、AWS Site-to-Site VPN、または VPC ピアリングを使用して VPC へのプライベートネットワーク接続を確立し、ユーザーがインスタンスのプライベート IP アドレスに到達できるようにする必要があります。

インスタンスにパブリック IPv4 アドレスがなく、上記のようにネットワークアクセスを設定したくない場合は、EC2 インスタンス接続エンドポイントを EC2 インスタンス接続の代わりとして検討できます。EC2 インスタンスConnect エンドポイントを使用すると、インスタンスにパブリック IPv4 アドレスがなくても、SSH または RDP 経由でインスタンスに接続できます。詳細については、「Amazon EC2 コンソールを使用した Linux インスタンスへの接続」を参照してください。

セキュリティグループルール

インスタンスに関連付けられているセキュリティグループで、IP アドレスまたはネットワークからのインバウンド SSH トラフィックがポート 22 で許可されることを確認します。VPC のデフォルトのセキュリティグループでは、着信 SSH トラフィックはデフォルトでは許可されません。インスタンス起動ウィザードで作成されたセキュリティグループは、デフォルトで受信 SSH トラフィックを許可します。詳細については、「Linux インスタンス用のインバウンドトラフィックの承認」を参照してください。

EC2 Instance Connect は、ブラウザベースの SSH 接続に対する特定の IP アドレス範囲をインスタンスに使用します (ユーザーが Amazon EC2 コンソールを使用してインスタンスに接続する場合)。ユーザーが Amazon EC2 コンソールを使用してインスタンスに接続する場合、インスタンスに関連付けられているセキュリティグループで、EC2_INSTANCE_CONNECT の IP アドレス範囲からのインバウンド SSH トラフィックが許可されていることを確認します。アドレス範囲を特定するには、AWS が提供する JSON ファイルをダウンロードした上で、EC2_INSTANCE_CONNECT をサービス値として使用しながら EC2 Instance Connect 用のサブセットをフィルタリングします。これらの IP アドレス範囲は、AWS リージョン間で異なります。JSON ファイルのダウンロードおよびサービスを使用したフィルタリングの詳細については、「Amazon VPC ユーザーガイド」の「AWS IP アドレスの範囲」を参照してください。

許可を付与する

EC2 Instance Connect を使用してインスタンスに接続するすべての IAM ユーザーに、必要なアクセス許可を付与する必要があります。詳細については、「IAM への EC2 Instance Connect のアクセス許可の付与」を参照してください。

ローカルコンピュータのセットアップ

ユーザーが SSH を使用して接続する場合、ローカルコンピュータに SSH クライアントがあることを確認する必要があります。

ほとんどの場合、ユーザーのローカルコンピュータにはデフォルトで SSH クライアントがインストールされています。SSH クライアントがあるかどうかを確認するには、コマンドラインで ssh と入力します。使用するローカルコンピュータでこのコマンドが認識されない場合、SSH クライアントをインストールできます。Linux または macOS X に SSH クライアントをインストールする詳細については、「http://www.openssh.com」を参照してください。Windows 10 に SSH クライアントをインストールする詳細については、「Windows の OpenSSH」を参照してください。

ユーザーが Amazon EC2 コンソールのみを使用してインスタンスに接続する場合、ローカルコンピュータに SSH クライアントをインストールする必要はありません。

ユーザーネーム

EC2 Instance Connect を使用してインスタンスに接続する場合、ユーザー名は次の前提条件を満たす必要があります。

- 1 字目はアルファベット (A-Z、a-z) または下線 (_) でなければなりません。
- 後続には、アルファベット (A-Z、a-z)、数字 (0-9)、または以下の文字を使用できます: @ . _ -
- 最小長: 1 文字
- 最大長: 31 文字
- 正規表現パターン ^[A-Za-z_][A-Za-z0-9\@\._-]{0,30}[A-Za-z0-9\$_-]?$ と一致する必要があります。

IAM への EC2 Instance Connect のアクセス許可の付与

EC2 Instance Connect を使用してインスタンスに接続するには、以下のアクションと条件に対するアクセス許可をユーザーに付与する IAM ポリシーを作成する必要があります。

- ec2-instance-connect:SendSSHPublicKey アクション – パブリックキーをインスタンスにプッシュするためのアクセス許可を付与します。
- ec2:osuser 条件 – パブリックキーをインスタンスにプッシュできる OS ユーザーの名前を指定します。インスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。デフォルトのユーザー名は AL2023 と Amazon Linux 2 では ec2-user、Ubuntu では ubuntu です。

- ec2:DescribeInstances アクション — ラッパーがこのアクションを呼び出すため、EC2 コンソールを使用するときに必要です。ユーザーには、別のポリシーからこのアクションを呼び出すためのアクセス許可が既に付与されている場合があります。

特定の EC2 インスタンスへのアクセスを制限することを検討してください。それ以外の場合、ec2-instance-connect:SendSSHPublicKey アクションのアクセス許可を持つすべての IAM プリンシパルは、すべての EC2 インスタンスに接続できます。リソース ARN を指定するか、リソースタグを条件キーとして使用して、アクセスを制限することができます。

詳細については、「Amazon EC2 Instance Connect のアクション、リソース、および条件キー」を参照してください。

IAM ポリシーの作成の詳細については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

ユーザーに特定のインスタンスへの接続を許可

次の IAM ポリシーは、リソース ARN で識別される特定のインスタンスに接続するアクセス許可を付与します。

次の IAM ポリシーの例では、以下のアクションと条件が指定されています。

- ec2-instance-connect:SendSSHPublicKey アクションは、リソース ARN で指定された 2 つのインスタンスに接続するためのアクセス許可をユーザーに付与します。すべての EC2 インスタンスに接続するためのアクセス許可をユーザーに付与するには、リソース ARN を * (ワイルドカード) に置き換えます。
- ec2:osuser 条件により、接続時に *ami-username* が指定されている場合にのみ、インスタンスに接続するためのアクセス許可が付与されます。
- ec2:DescribeInstances アクションは、コンソールを使用してインスタンスに接続するユーザーにアクセス許可を付与するように指定されます。ユーザーが SSH クライアントのみを使用してインスタンスに接続する場合は、ec2:DescribeInstances を省略できます。ec2:Describe* API アクションはリソースレベルのアクセス許可をサポートしないことに注意してください。したがって、Resource の要素には、* (ワイルドカード) が必要です。

```
{
 "Version": "2012-10-17",
 "Statement": [{
   "Effect": "Allow",
```

```
    "Action": "ec2-instance-connect:SendSSHPublicKey",
    "Resource": [
     "arn:aws:ec2:region:account-id:instance/i-1234567890abcdef0",
     "arn:aws:ec2:region:account-id:instance/i-0598c7d356eba48d7"
    ],
    "Condition": {
     "StringEquals": {
      "ec2:osuser": "ami-username"
     }
    }
   },
   {
    "Effect": "Allow",
    "Action": "ec2:DescribeInstances",
    "Resource": "*"
   }
  ]
}
```

ユーザーに特定のタグを持つインスタンスへの接続を許可

属性ベースのアクセス制御 (ABAC) は、ユーザーおよび AWS リソースにアタッチできるタグに基づいてアクセス許可を定義する認証戦略です。リソースタグを使用してインスタンスへのアクセスを制御することもできます。AWS リソースへのアクセスを制御するタグの使用の詳細については、IAM ユーザーガイドの「AWS リソースへのアクセス制御」を参照してください。

次の IAM ポリシーの例では、`ec2-instance-connect:SendSSHPublicKey` アクションは、インスタンスに key=`tag-key` と value=`tag-value` が付いたリソースタグが割り当てられていることを条件に、(リソース ARN の * (ワイルドカード) で示される) 任意のインスタンスに接続するためのアクセス許可をユーザーに付与します。

`ec2:DescribeInstances` アクションは、コンソールを使用してインスタンスに接続するユーザーにアクセス許可を付与するように指定されます。ユーザーが SSH クライアントのみを使用してインスタンスに接続する場合は、`ec2:DescribeInstances` を省略できます。`ec2:Describe*` API アクションはリソースレベルのアクセス許可をサポートしないことに注意してください。したがって、`Resource` の要素には、* (ワイルドカード) が必要です。

```
{
 "Version": "2012-10-17",
 "Statement": [{
   "Effect": "Allow",
```

```
    "Action": "ec2-instance-connect:SendSSHPublicKey",
    "Resource": "arn:aws:ec2:region:account-id:instance/*",
    "Condition": {
     "StringEquals": {
      "aws:ResourceTag/tag-key": "tag-value"
     }
    }
   },
   {
    "Effect": "Allow",
    "Action": "ec2:DescribeInstances",
    "Resource": "*"
   }
  ]
}
```

EC2 インスタンスでの EC2 Instance Connect のインストール

EC2 Instance Connect を使用してインスタンスに接続するには、EC2 Instance Connect がインストールされている必要があります。

EC2 Instance Connect には以下の AMI がプリインストールされています。

- AL2023
- Amazon Linux 2 2.0.20190618 以降
- macOS Sonoma 14.2.1 以降
- macOS Ventura 13.6.3 以降
- macOS Monterey 12.7.2 以降
- Ubuntu 20.04 以降

インスタンスが前述のリストの AMI のいずれかを使用して起動された場合は、この手順をスキップできます。

> **Note**
>
> SSH 認証の `AuthorizedKeysCommand` および `AuthorizedKeysCommandUser` 設定を構成した場合、EC2 Instance Connect をインストールしても更新されません。その結果、EC2 Instance Connect は使用できません。

EC2 Instance Connect インストールの前提条件

- 以下のサポートされている AMI のいずれかを使用してインスタンスを起動します。

  バージョン 2.0.20190618 より前の Amazon Linux 2

  CentOS Stream 8 および 9

  14.2.1 より前の macOS Sonoma、13.6.3 より前の Ventura、12.7.2 より前の Monterey

  Red Hat Enterprise Linux (RHEL) 8 および 9

  Ubuntu 16.04 および 18.04

  インスタンスが Amazon Linux 2、macOS Sonoma、Ventura、Monterey、または Ubuntu の新しいバージョンで起動されている場合は、EC2 Instance Connect があらかじめインストールされているため、この手順をスキップできます。
- EC2 Instance Connect の一般的な前提条件を確認します。

  詳細については、「前提条件」を参照してください。
- ローカルマシン上の SSH クライアントを使用してインスタンスに接続するための前提条件を確認します。

  ローカルマシンが Linux または macOS の場合は、「SSH を使用して Linux または macOS から Linux インスタンスに接続します。」を参照してください。ローカルマシンが Windows の場合は、「前提条件」を参照してください。

  詳細については、「SSH 接続の前提条件」を参照してください。
- インスタンスの ID を取得します。

  自分のインスタンスの ID は、Amazon EC2 コンソールを使用して ([インスタンス ID] 列から) 取得できます。その代わりに、describe-instances (AWS CLI) または Get-EC2Instance (AWS Tools for Windows PowerShell) コマンドを使用することもできます。
- 使用するローカルコンピュータに SSH クライアントをインストールします。

  ほとんどの場合、ローカルコンピュータにはデフォルトで SSH クライアントがインストールされています。SSH クライアントがあるかどうかを確認するには、コマンドラインで ssh と入力します。使用するローカルコンピュータでこのコマンドが認識されない場合、SSH クライアントをインストールできます。Linux または macOS X に SSH クライアントをインストールする詳細につ

いては、「http://www.openssh.com」を参照してください。Windows 10 に SSH クライアントをインストールする詳細については、「Windows の OpenSSH」を参照してください。

- (Ubuntu) AWS CLI をインスタンスにインストールします。

  EC2 Instance Connect を Ubuntu インスタンスにインストールするには、インスタンスで AWS CLI を使用する必要があります。AWS CLI のインストールの詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS CLI のインストール」を参照してください。

EC2 Instance Connect のインストール

EC2 Instance Connect をインストールすると、インスタンスに SSH デーモンが設定されます。

インスタンスのオペレーティングシステムに応じて、次のいずれかの手順を使用して EC2 Instance Connect をインストールします。

Amazon Linux 2

EC2 Instance Connect を Amazon Linux 2 で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。Amazon Linux 2 の場合、デフォルトのユーザー名は `ec2-user` です。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。
2. インスタンスに EC2 Instance Connect パッケージをインストールします。

   ```
   [ec2-user ~]$ sudo yum install ec2-instance-connect
   ```

   3 つの新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

   ```
   eic_curl_authorized_keys
   eic_parse_authorized_keys
   ```

```
eic_run_authorized_keys
```

3. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

```
[ec2-user ~]$ sudo less /etc/ssh/sshd_config
```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

CentOS

EC2 Instance Connect を CentOS で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。CentOS の場合、デフォルトユーザー名は centos または ec2-user です。

```
$ ssh -i my_ec2_private_key.pem centos@ec2-a-b-c-d.us-
west-2.compute.amazonaws.com
```

インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。

2. HTTP または HTTPS プロキシを使用する場合は、現在のシェルセッションで `http_proxy` または `https_proxy` の環境変数を設定する必要があります。

   プロキシを使用していない場合は、この手順を省略できます。

   - HTTP プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=http://hostname:port
     $ export https_proxy=http://hostname:port
     ```

   - HTTPS プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=https://hostname:port
     $ export https_proxy=https://hostname:port
     ```

3. 次のコマンドを実行して、インスタンスに EC2 Instance Connect パッケージをインストールします。

   CentOS 用の EC2 Instance Connect 設定ファイルは Red Hat Package Manager (RPM) パッケージで提供され、CentOS 8 と CentOS 9 用の異なる RPM パッケージ、および Intel/AMD (x86_64) または ARM (AArch64) で実行されるインスタンスタイプ用の異なる RPM パッケージが含まれています。

   オペレーティングシステムと CPU アーキテクチャに合ったコマンドブロックを使用してください。

   - CentOS 8

     Intel/AMD (x86_64)

     ```
     [ec2-user ~]$ mkdir /tmp/ec2-instance-connect
     [ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
     west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect.rhel8.rpm -o /
     tmp/ec2-instance-connect/ec2-instance-connect.rpm
     ```

```
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect.rhel8.rpm -o /
tmp/ec2-instance-connect/ec2-instance-connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

- CentOS 9

  Intel/AMD (x86_64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect.rpm -o /tmp/ec2-
instance-connect/ec2-instance-connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

  ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect.rpm -o /tmp/ec2-
instance-connect/ec2-instance-connect.rpm
```

```
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

次の新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

```
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

   - CentOS 8 の場合:

     ```
     [ec2-user ~]$ sudo less /lib/systemd/system/ssh.service.d/ec2-instance-
     connect.conf
     ```

   - CentOS 9 の場合:

     ```
     [ec2-user ~]$ sudo less /etc/ssh/sshd_config.d/60-ec2-instance-connect.conf
     ```

   AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

   ```
   AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
   AuthorizedKeysCommandUser ec2-instance-connect
   ```

   - AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
   - AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> `AuthorizedKeysCommand` や `AuthorizedKeysCommandUser` を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

macOS

EC2 Instance Connect を macOS で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。macOS インスタンスの場合、デフォルトのユーザー名は `ec2-user` です。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。

2. 次のコマンドを使用して Homwbrew を更新します。このアップデートでは、Homebrew が認識しているソフトウェアが一覧表示されます。EC2 Instance Connect パッケージは、macOS インスタンスでは Homebrew 経由で提供されます。詳細については、オペレーティングシステムとソフトウェアの更新 をご参照ください。

   ```
   [ec2-user ~]$ brew update
   ```

3. インスタンスに EC2 Instance Connect パッケージをインストールします。これによりソフトウェアがインストールされ、sshd がそれを使用するように設定されます。

   ```
   [ec2-user ~]$ brew install ec2-instance-connect
   ```

   次の新しいスクリプトが `/opt/aws/bin/` フォルダに表示されます。

```
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

```
[ec2-user ~]$ sudo less /etc/ssh/sshd_config.d/60-ec2-instance-connect.conf
```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

RHEL

EC2 Instance Connect を Red Hat Enterprise Linux (RHEL) で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。RHEL の場合、デフォルトのユーザー名は ec2-user または root です。

```
$ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
west-2.compute.amazonaws.com
```

インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。

2. HTTP または HTTPS プロキシを使用する場合は、現在のシェルセッションで `http_proxy` または `https_proxy` の環境変数を設定する必要があります。

   プロキシを使用していない場合は、この手順を省略できます。

   - HTTP プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=http://hostname:port
     $ export https_proxy=http://hostname:port
     ```

   - HTTPS プロキシサーバーの場合は、次のコマンドを実行します。

     ```
     $ export http_proxy=https://hostname:port
     $ export https_proxy=https://hostname:port
     ```

3. 次のコマンドを実行して、インスタンスに EC2 Instance Connect パッケージをインストールします。

   RHEL 用の EC2 Instance Connect 設定ファイルは Red Hat Package Manager (RPM) パッケージで提供され、RHEL 8 と RHEL 9 用の異なる RPM パッケージ、および Intel/AMD (x86_64) または ARM (AArch64) で実行されるインスタンスタイプ用の異なる RPM パッケージが含まれています。

   オペレーティングシステムと CPU アーキテクチャに合ったコマンドブロックを使用してください。

   - RHEL 8

     Intel/AMD (x86_64)

     ```
     [ec2-user ~]$ mkdir /tmp/ec2-instance-connect
     [ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
     west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect.rhel8.rpm -o /
     tmp/ec2-instance-connect/ec2-instance-connect.rpm
     ```

```
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect.rhel8.rpm -o /
tmp/ec2-instance-connect/ec2-instance-connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

- RHEL 9

  Intel/AMD (x86_64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect.rpm -o /tmp/ec2-
instance-connect/ec2-instance-connect.rpm
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

  ARM (AArch64)

```
[ec2-user ~]$ mkdir /tmp/ec2-instance-connect
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_arm64/ec2-instance-connect.rpm -o /tmp/ec2-
instance-connect/ec2-instance-connect.rpm
```

```
[ec2-user ~]$ curl https://amazon-ec2-instance-connect-us-west-2.s3.us-
west-2.amazonaws.com/latest/linux_amd64/ec2-instance-connect-
selinux.noarch.rpm -o /tmp/ec2-instance-connect/ec2-instance-connect-
selinux.rpm
[ec2-user ~]$ sudo yum install -y /tmp/ec2-instance-connect/ec2-instance-
connect.rpm /tmp/ec2-instance-connect/ec2-instance-connect-selinux.rpm
```

次の新しいスクリプトが /opt/aws/bin/ フォルダに表示されます。

```
eic_run_authorized_keys
```

4. (オプション) EC2 Instance Connect がインスタンスに正常にインストールされたことを確認します。

   - RHEL 8 の場合:

     ```
     [ec2-user ~]$ sudo less /lib/systemd/system/ssh.service.d/ec2-instance-
     connect.conf
     ```

   - RHEL 9 の場合:

     ```
     [ec2-user ~]$ sudo less /etc/ssh/sshd_config.d/60-ec2-instance-connect.conf
     ```

   AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

   ```
   AuthorizedKeysCommand /opt/aws/bin/eic_run_authorized_keys %u %f
   AuthorizedKeysCommandUser ec2-instance-connect
   ```

   - AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
   - AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

Ubuntu

EC2 Instance Connect を Ubuntu 16.04 で起動したインスタンスにインストールするには

1. SSH を使用してインスタンスに接続します。

   次のコマンド内のサンプル値を独自の値に置き換えます。インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスを起動するために使用した AMI のデフォルトのユーザー名を使用します。Ubuntu AMI の場合、ユーザー名は ubuntu です。

   ```
   $ ssh -i my_ec2_private_key.pem ubuntu@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。
2. (オプション) インスタンスに最新の Ubuntu AMI があることを確認します。

   次のコマンドを実行して、インスタンスのすべてのパッケージを更新します。

   ```
   ubuntu:~$ sudo apt-get update
   ```

   ```
   ubuntu:~$ sudo apt-get upgrade
   ```
3. インスタンスに EC2 Instance Connect パッケージをインストールします。

   ```
   ubuntu:~$ sudo apt-get install ec2-instance-connect
   ```

   3 つの新しいスクリプトが /usr/share/ec2-instance-connect/ フォルダに表示されます。

```
eic_curl_authorized_keys
eic_parse_authorized_keys
eic_run_authorized_keys
```

4. (オプション) Instance Connect がインスタンスに正常にインストールされたことを確認します。

```
ubuntu:~$ sudo less /lib/systemd/system/ssh.service.d/ec2-instance-connect.conf
```

AuthorizedKeysCommand 行と AuthorizedKeysCommandUser 行に以下の値が含まれていれば、EC2 Instance Connect は正常にインストールされています。

```
AuthorizedKeysCommand /usr/share/ec2-instance-connect/eic_run_authorized_keys %
%u %%f
AuthorizedKeysCommandUser ec2-instance-connect
```

- AuthorizedKeysCommand は、インスタンスメタデータからキーを探すように eic_run_authorized_keys スクリプトを設定します。
- AuthorizedKeysCommandUser は、システムユーザーを ec2-instance-connect として設定します。

> **Note**
>
> AuthorizedKeysCommand や AuthorizedKeysCommandUser を設定済みである場合は、EC2 Instance Connect をインストールしても値は変更されないため、EC2 Instance Connect は使用できません。

EC2 Instance Connect パッケージの詳細については、GitHub ウェブサイトの「aws/aws-ec2-instance-connect-config」を参照してください。

EC2 Instance Connect を使用して接続

次の手順では、EC2 Instance Connect を使用して Linux インスタンスに接続する方法について説明します。

どの接続オプションを使用するかを決めます。使用する接続オプションは、インスタンスにパブリック IPv4 アドレスがあるかどうかによって異なります。

- Amazon EC2 コンソール – Amazon EC2 コンソールを使用して接続するには、インスタンスにパブリック IPv4 アドレスが必要です。
- SSH クライアント – インスタンスにパブリック IP アドレスがない場合は、SSH クライアントを使用して、プライベートネットワーク経由でインスタンスに接続できます。例えば、同じ VPC 内からの接続や、VPN 接続、Transit Gateway、AWS Direct Connect を介した接続などがあります。

EC2 Instance Connect は IPv6 アドレスを使用した接続をサポートしていません。

> **Tip**
>
> EC2 Instance Connect は Linux インスタンスに接続するためのオプションの 1 つです。他のオプションについては、「Linux インスタンスへの接続」を参照してください。Windows インスタンスに接続する必要がある場合は、 の「Windows インスタンスへの接続」を参照してください。

EC2 Instance Connect の接続オプション

- Amazon EC2 コンソールを使用した接続
- 独自のキーと SSH クライアントを使用して接続する
- EC2 Instance Connect を使用して、AWS CLI で Linux インスタンスに接続します。
- トラブルシューティング

Amazon EC2 コンソールを使用した接続

コンソールからインスタンスを選択し、EC2 Instance Connect を使用した接続を選択することで、Amazon EC2 コンソールを使用してインスタンスに接続できます。Instance Connect はアクセス許可を処理し、正常な接続を提供します。

Amazon EC2 コンソールを使用して接続するには、インスタンスにパブリック IPv4 アドレスが必要です。接続する前に、すべての前提条件を確認してください。

Amazon EC2 コンソールからブラウザベースのクライアントを使用してインスタンスに接続するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。

3. インスタンスを選択し、[接続] を選択します。
4. [EC2 Instance Connect] を選択します。
5. ユーザー名を検証し、[Connect (接続)] を選択してターミナルウィンドウを開きます。

独自のキーと SSH クライアントを使用して接続する

EC2 Instance Connect API の使用中に、独自の SSH キーを使用して、選択した SSH クライアントからインスタンスに接続できます。これにより、インスタンスにパブリックキーをプッシュする Instance Connect 機能を活用できます。この接続方法は、パブリック IP アドレスとプライベート IP アドレスを持つインスタンスに対して機能します。

要件

- キーペアの要件
  - サポートされているタイプ: RSA (OpenSSH および SSH2) および ED25519
  - サポートされている長さ: 2048 および 4096
  - 詳細については、サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする を参照してください。
- プライベート IP アドレスのみを持つインスタンスに接続する場合、SSH セッションを開始するローカルコンピュータには、EC2 Instance Connect サービスエンドポイントへの接続 (SSH パブリックキーをインスタンスにプッシュするため) と、SSH セッションを確立するためのインスタンスのプライベート IP アドレスへのネットワーク接続が必要です。EC2 Instance Connect のサービスエンドポイントには、インターネットまたは AWS Direct Connect パブリック仮想インターフェイス経由で到達が可能です。インスタンスのプライベート IP アドレスに接続するには、AWS Direct Connect、AWS Site-to-Site VPN や VPC ピアリングなどのサービスを利用できます。

接続する前に、すべての前提条件を確認してください。

独自のキーと任意の SSH クライアントを使用してインスタンスに接続するには

1. (オプション) 新しい SSH プライベートキーとパブリックキーを生成する

   新しい SSH プライベートキーとパブリックキー (`my_key` および `my_key.pub`) は、次のコマンドを使用して生成できます。

   ```
   $ ssh-keygen -t rsa -f my_key
   ```

2. SSH パブリックキーをインスタンスにプッシュする

   send-ssh-public-key コマンドを使用して、SSH パブリックキーをインスタンスにプッシュします。AL2023 または Amazon Linux 2 を使用してインスタンスを起動した場合、AMI のデフォルトのユーザー名は ec2-user です。Ubuntu を使用してインスタンスを起動した場合、AMI のデフォルトのユーザー名は ubuntu です。

   以下に、ec2-user を認証するために、指定されたアベイラビリティーゾーンで指定されたインスタンスにパブリックキーをプッシュする例を示しています。

   ```
   $ aws ec2-instance-connect send-ssh-public-key \
       --region us-west-2 \
       --availability-zone us-west-2b \
       --instance-id i-001234a4bf70dec41EXAMPLE \
       --instance-os-user ec2-user \
       --ssh-public-key file://my_key.pub
   ```

3. プライベートキーを使用してインスタンスに接続する

   パブリックキーがインスタンスメタデータから削除される前に (削除されるまでの時間は 60 秒です)、プライベートキーを使用してインスタンスに接続するには、ssh コマンドを使用します。パブリックキーに対応するプライベートキー、インスタンスを起動するために使用した AMI のデフォルトのユーザー名、およびインスタンスのパブリック DNS 名を指定します (プライベートネットワーク経由で接続する場合は、プライベート DNS 名または IP アドレスを指定します)。IdentitiesOnly=yes オプションを追加し、ssh config 内のファイルと指定したキーのみが接続に使用されるようにします。

   ```
   $ ssh -o "IdentitiesOnly=yes" -i my_key ec2-
   user@ec2-198-51-100-1.compute-1.amazonaws.com
   ```

EC2 Instance Connect を使用して、AWS CLI で Linux インスタンスに接続します。

インスタンス ID がわかっている場合は、Amazon EC2 Instance Connect を使用すると、SSH クライアントを使用してインスタンスに接続できます。接続タイプを指定しない場合は、EC2 Instance Connect が自動的にインスタンスのパブリック IPv4 アドレスへの接続を試みます。インスタンスにパブリック IPv4 アドレスがない場合、EC2 Instance Connect は EC2 Instance Connect Endpoint を介してインスタンスのプライベート IPv4 アドレスへの接続を試みます。インスタンスにプライベート IPv4 アドレスがない場合、または VPC に EC2 Instance Connect Endpoint がない場合、EC2 Instance Connect はインスタンスの IPv6 アドレスへの接続を試みます。

この方法で接続する前に、使用する資格情報を含めて AWS CLI を設定し、AWS CLI の最新バージョンを使用していることを確認します。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS CLI の最新バージョンのインストールまたは更新」および「AWS CLI の設定」を参照してください。

接続タイプ

`auto` (デフォルト)

CLI は、次の順序でインスタンスの IP アドレスを使用し、対応する接続タイプを使用して接続を試みます。

- パブリック IPv4: `direct`
- プライベート IPv4: `eice`
- IPv6: `direct`

`direct`

CLI は、次の順序でインスタンスの IP アドレスを使用して接続を試みます (EC2 Instance Connect Endpoint 経由では接続しません)。

- パブリック IPv4
- IPv6
- プライベート IPv4

`eice`

CLI は常にインスタンスのプライベート IPv4 アドレスを使用します。

> **Note**
>
> 将来的には、`auto` 接続タイプの動作を変更する可能性があります。希望する接続タイプを確実に使用するには、`--connection-type` を `direct` または `eice` のいずれかに明示的に設定することをお勧めします。

EC2 Instance Connect を使用してインスタンスに接続すると、EC2 Instance Connect API から SSH パブリックキーがインスタンスメタデータにプッシュされ、60 秒間保持されます。ユーザーにアタッチされた IAM ポリシーにより、ユーザーはパブリックキーをインスタンスメタデータにプッ

シュすることを許可されます。EC2 Instance Connect の詳細については、「EC2 Instance Connect を使用して Linux インスタンスに接続する」を参照してください。

インスタンス ID を使用してインスタンスに接続するには

インスタンス ID のみがわかっていて、インスタンスへの接続時に使用する接続タイプを EC2 Instance Connect に決定する場合は、`ec2-instance-connect` CLI を使用して `ssh` コマンドとインスタンス ID を指定します。

```
aws ec2-instance-connect ssh --instance-id i-1234567890example
```

インスタンス ID と EC2 Instance Connect Endpoint を使用してインスタンスに接続するには

EC2 Instance Connect Endpoint を介してインスタンスに接続する場合は、前述のコマンドを使用し、`--connection-type` パラメータと `eice` 値も指定します。

```
aws ec2-instance-connect ssh --instance-id i-1234567890example --connection-type eice
```

インスタンス ID と独自のプライベートキーファイルを使用してインスタンスに接続するには

独自のプライベートキーを使用して EC2 Instance Connect Endpoint 経由でインスタンスに接続する場合は、インスタンス ID とプライベートキーファイルへのパスを指定します。パスに *file://*を含めないでください。次のようなパスは失敗します: *file:///path/to/key*。

```
aws ec2-instance-connect ssh --instance-id i-1234567890example --private-key-file /
path/to/key.pem
```

トラブルシューティング

インスタンスへの接続を試みた際にエラーが発生した場合は、以下を参照してください。

- インスタンスへの接続に関するトラブルシューティング
- EC2 Instance Connect を使用して EC2 インスタンスへ接続しようとしたときの問題をトラブルシューティングするにはどうすればよいですか?

EC2 Instance Connect のアンインストール

EC2 Instance Connect を無効にするには、インスタンスに接続し、OS にインストールした `ec2-instance-connect` パッケージをアンインストールします。`sshd` 設定が EC2 Instance Connect をインストールしたときのまま変更されていない場合、`ec2-instance-connect` をアンインス

トールすると、sshd 設定も削除されます。sshd 設定が EC2 Instance Connect のインストール後に変更されている場合は、それを手動で更新する必要があります。

Amazon Linux

EC2 Instance Connect が事前設定されている AL2023 および Amazon Linux 2 2.0.20190618 以降では EC2 Instance Connect をアンインストールできます。

Amazon Linux 2 で起動したインスタンスの EC2 Instance Connect をアンインストールするには

1. SSH を使用してインスタンスに接続します。インスタンスの起動時に使用した SSH キーペアと AL2023 または Amazon Linux 2 AMI のデフォルトのユーザー名 (ec2-user) を指定します。

   例えば、次の ssh コマンドは、キーペア ec2-a-b-c-d.us-west-2.compute.amazonaws.com を使用して、パブリック DNS 名 my_ec2_private_key.pem でインスタンスに接続します。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-west-2.compute.amazonaws.com
   ```

2. yum コマンドを使用して ec2-instance-connect パッケージをアンインストールします。

   ```
   [ec2-user ~]$ sudo yum remove ec2-instance-connect
   ```

Ubuntu

Ubuntu AMI で起動したインスタンスの EC2 Instance Connect をアンインストールするには

1. SSH を使用してインスタンスに接続します。インスタンスの起動時に使用した SSH キーペアと、Ubuntu AMI のデフォルトのユーザー名 (ubuntu) を指定します。

   例えば、次の ssh コマンドは、キーペア ec2-a-b-c-d.us-west-2.compute.amazonaws.com を使用して、パブリック DNS 名 my_ec2_private_key.pem でインスタンスに接続します。

   ```
   $ ssh -i my_ec2_private_key.pem ubuntu@ec2-a-b-c-d.us-west-2.compute.amazonaws.com
   ```

2. apt-get コマンドを使用して ec2-instance-connect パッケージをアンインストールします。

```
ubuntu:~$ sudo apt-get remove ec2-instance-connect
```

AWS Systems Manager Session Manager を使用して Linux インスタンスに接続する

Session Manager はフルマネージドの AWS Systems Manager 機能です。ブラウザベースのインタラクティブなワンクリックシェルまたは AWS CLI を介して Amazon EC2 インスタンスを管理できます。Session Manager を使用して、アカウント内のインスタンスとのセッションを開始できます。セッションの開始後、他の接続タイプと同様、bash コマンドを実行できます。Session Manager の詳細については、AWS Systems Manager ユーザーガイドの「AWS Systems Manager Session Manager」を参照してください。

前提条件

Session Manager を使用してインスタンスに接続する前に、必要な設定手順が完了していることを確認してください。詳細と手順については、「Session Manager のセットアップ」を参照してください。

Amazon EC2 コンソールで Session Manager を使用して Linux インスタンスに接続するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[接続] を選択します。
4. セッションマネージャーを選択します。
5. [接続]を選択します。

トラブルシューティング

1 つ以上の Systems Manager アクション (ssm:*command-name*) の実行を承認されていないというエラーが表示された場合は、Amazon EC2 コンソールからセッションを開始できるようにポリシーを更新する必要があります。詳細については、「AWS Systems Manager ユーザーガイド」の「Sample IAM policies for Session Manager」を参照してください。

EC2 Instance Connect Endpoint を使用した、パブリック IPv4 アドレスを必要としないインスタンスへの接続

EC2 インスタンスConnect エンドポイントを使用すると、インスタンスにパブリック IPv4 アドレスがなくても、SSH または RDP 経由でインスタンスに接続できます。

仕組み

まず、仮想プライベートクラウド (VPC) のサブネットに EC2 Instance Connect Endpoint を作成します。次に、インスタンスに接続するときに、インスタンスの ID を指定します。オプションで EC2 Instance Connect Endpoint を指定できます。エンドポイントはインスタンスへのプライベートトンネルとして機能します。

サブネットに EC2 Instance Connect Endpoint を作成すると、VPC がサブネットの通信を許可するように設定されていれば、そのエンドポイントを使用して VPC 内の任意のサブネットの任意のインスタンスに接続できます。

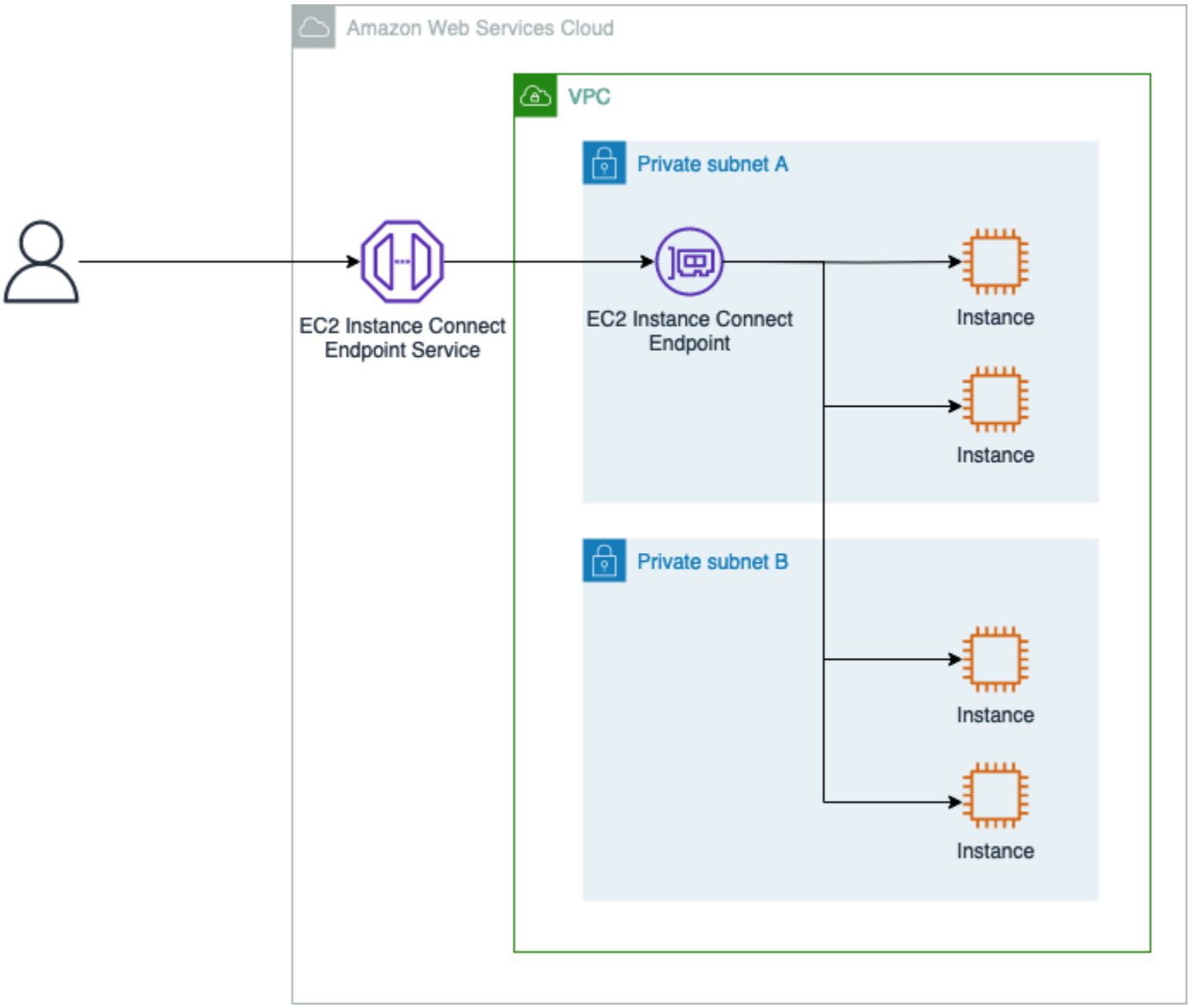
> **Note**
>
> あるサブネットの EC2 Instance Connect Endpoint を使用して、別のアベイラビリティーゾーンにある別のサブネットのインスタンスに接続する場合、アベイラビリティーゾーン間のデータ転送に追加料金がかかります。

次の図は、VPC のプライベートサブネットにあるインスタンスにインターネットから接続しているユーザーを示しています。図には、以下の主要コンポーネントがあります。

- EC2 Instance Connect Endpoint サービスは、ユーザーが EC2 Instance Connect Endpoint を使用して、プライベートサブネットにあるインスタンスにインターネットから接続できるようにする AWS のサービスです。
- プライベートサブネット A の EC2 Instance Connect Endpoint はプライベートトンネルとして機能し、ユーザーはプライベートサブネットにあるインスタンスに接続できます。

  EC2 Instance Connect Endpoint の作成と接続へのアクセスは、IAM アクセス許可によって制御されます。インスタンスに追加のセキュリティグループルールを設定して、インバウンドトラフィックを制限できます。例えば、インスタンスのインバウンドルールを使用して、EC2 Instance Connect Endpoint からの管理ポート上のトラフィックのみを許可することができます。
- プライベートサブネット A には EC2 Instance Connect Endpoint がありますが、プライベートサブネット B にはありません。VPC の設定に基づいて、プライベートサブネット A とプライベート

サブネット B が通信を許可されている場合、プライベートサブネット A の EC2 Instance Connect Endpoint を使用してプライベートサブネット B のインスタンスに接続できます。



利点

EC2 Instance Connect Endpoint には以下の利点があります。

- インスタンスにパブリック IPv4 アドレスを設定することなく、インスタンスに接続できます。AWS は、実行中のインスタンスと Elastic IP アドレスに関連付けられたパブリック IPv4 アドレスを含む、すべてのパブリック IPv4 アドレスに対して料金を請求します。詳細については、「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。

- VPC にインターネットゲートウェイがなくても、インターネットからインスタンスに接続できます。
- IAM ポリシーとアクセス許可を使用して、インスタンスに接続する EC2 Instance Connect Endpoint の作成と使用へのアクセスを制御できます。
- インスタンスへの接続を試みると、成功または失敗を問わず、すべて CloudTrail にログが作成されます。

コンテンツ

前提条件

EC2 インスタンス接続エンドポイントを使用してインスタンスに接続するための前提条件は次のとおりです。

- AWS リージョン
- AMI
- IPv4 アドレス
- セキュリティグループ
- 許可を付与する

AWS リージョン

カナダ西部 (カルガリー) を除くすべての AWS リージョン でサポートされています。

AMI

サポートされる AMI は、EC2 Instance Connect Endpoint を使用してインスタンスに接続する方法によって異なります。

インスタンスには、EC2 コンソール、SSH、または RDP のいずれかの方法を使用して接続できます。

Console

EC2 インスタンス接続エンドポイントを使用してインスタンスに接続する際に EC2 コンソールを使用している場合、インスタンスには EC2 Instance Connect がインストールされている必要があります。

EC2 Instance Connect は以下の AMI にプリインストールされています。

- AL2023
- Amazon Linux 2 2.0.20190618 以降
- macOS Sonoma 14.2.1 以降
- macOS Ventura 13.6.3 以降
- macOS Monterey 12.7.2 以降
- Ubuntu 20.04 以降

以下の AMI を使用して起動されたインスタンスに EC2 Instance Connect をインストールできます。

- バージョン 2.0.20190618 より前の Amazon Linux 2
- CentOS Stream 8 および 9
- 14.2.1 より前の macOS Sonoma、13.6.3 より前の Ventura、12.7.2 より前の Monterey
- Red Hat Enterprise Linux (RHEL) 8 および 9
- Ubuntu 16.04 または 18.04

インストール手順については、「EC2 インスタンスでの EC2 Instance Connect のインストール」を参照してください。

SSH

EC2 インスタンス接続エンドポイントを使用してインスタンスに接続する際に SSH を使用している場合、サポートされる AMI は、独自のキーペアを使用するか、EC2 Instance Connect にエフェメラルキーを処理させるかによって異なります。

独自のキーペアを使用する場合

任意の Linux AMI を使用できます。

EC2 Instance Connect にエフェメラルキーを処理させたい場合

インスタンスには EC2 Instance Connect がインストールされている必要があります。

EC2 Instance Connect は以下の AMI にプリインストールされています。

- AL2023
- Amazon Linux 2 2.0.20190618 以降
- macOS Sonoma 14.2.1 以降
- macOS Ventura 13.6.3 以降
- macOS Monterey 12.7.2 以降
- Ubuntu 20.04 以降

以下の AMI を使用して起動されたインスタンスに EC2 Instance Connect をインストールできます。

- バージョン 2.0.20190618 より前の Amazon Linux 2
- CentOS Stream 8 および 9
- 14.2.1 より前の macOS Sonoma、13.6.3 より前の Ventura、12.7.2 より前の Monterey
- Red Hat Enterprise Linux (RHEL) 8 および 9
- Ubuntu 16.04 または 18.04

インストール手順については、「EC2 インスタンスでの EC2 Instance Connect のインストール」を参照してください。

RDP

EC2 Instance Connect Endpoint を使用してインスタンスに接続する際に RDP を使用している場合は、Windows AMI を使用する必要があります。

IPv4 アドレス

インスタンスには IPv4 アドレス (プライベートまたはパブリックのいずれか) が必要です。EC2 Instance Connect Endpoint は IPv6 アドレスを使用した接続をサポートしていません。

セキュリティグループ

EC2 Instance Connect Endpoint と接続先のインスタンスには、それぞれセキュリティグループが割り当てられます。セキュリティグループのルールを設定する推奨方法については、EC2 Instance Connect Endpoint のセキュリティグループ を参照してください。

許可を付与する

EC2 インスタンス接続エンドポイントを使用してインスタンスに接続するすべての IAM ユーザーに、必要なアクセス許可を付与する必要があります。詳細については、「EC2 Instance Connect Endpoint を使用するための IAM アクセス許可の付与」を参照してください。

EC2 Instance Connect Endpoint を使用するための IAM アクセス許可の付与

EC2 Instance Connect Endpoint を作成または使用するには、ユーザーに以下に対するアクセス許可を付与する IAM ポリシーを作成する必要があります。

- EC2 Instance Connect Endpoint を作成、記述、削除する
- `ec2-instance-connect:OpenTunnel` アクションを使用して、EC2 Instance Connect Endpoint を使用してインスタンスに接続する

IAM ポリシーの作成の詳細については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。

EC2 Instance Connect Endpoint の IAM ポリシーの例

- ユーザーに EC2 Instance Connect Endpoint の作成、記述、削除を許可
- ユーザーに EC2 Instance Connect Endpoint を使用したインスタンスへの接続を許可
- ユーザーに指定した送信元 IP アドレス範囲からの接続のみを許可

ユーザーに EC2 Instance Connect Endpoint の作成、記述、削除を許可

EC2 Instance Connect Endpoint を作成するには、ユーザーには次のアクションのアクセス許可が必要です。

- ec2:CreateInstanceConnectEndpoint
- ec2:CreateNetworkInterface
- ec2:CreateTags
- iam:CreateServiceLinkedRole

EC2 Instance Connect Endpoint を記述して削除するには、ユーザーに次のアクションに対するアクセス許可が必要です。

- ec2:DescribeInstanceConnectEndpoints
- ec2:DeleteInstanceConnectEndpoint

すべてのサブネットで EC2 Instance Connect Endpoint を作成、記述、削除するためのアクセス許可を付与するポリシーを作成できます。または、サブネット ARN を許可済みの Resource として指定するか、ec2:SubnetID 条件キーを使用して、指定したサブネットのアクションのみを制限することもできます。aws:ResourceTag 条件キーを使用して、特定のタグによるエンドポイントの作成を明示的に許可または拒否することもできます。詳細については、「 IAM ユーザーガイド」の「IAM のポリシーとアクセス許可」を参照してください。

IAM ポリシーの例

次の IAM ポリシーの例では、[Resource] セクションはアスタリスク (*) で指定されたすべてのサブネットでエンドポイントを作成および削除するアクセス許可を付与しています。ec2:Describe* API アクションは、リソースレベルのアクセス許可をサポートしていません。したがって、Resource の要素には、* (ワイルドカード) が必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "GrantAllActionsInAllSubnets",
            "Action": [
                "ec2:CreateInstanceConnectEndpoint",
                "ec2:DeleteInstanceConnectEndpoint",
                "ec2:CreateNetworkInterface",
                "ec2:CreateTags",
                "iam:CreateServiceLinkedRole"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ec2:region:account-id:subnet/*"
        },
```

```
        {
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ec2:::security-group/*"
        },
        {
            "Sid": "DescribeInstanceConnectEndpoints",
            "Action": [
                "ec2:DescribeInstanceConnectEndpoints"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

ユーザーに EC2 Instance Connect Endpoint を使用したインスタンスへの接続を許可

`ec2-instance-connect:OpenTunnel` アクションは、EC2 Instance Connect Endpoint を介して接続するインスタンスへの TCP 接続を確立するためのアクセス許可を付与します。使用する EC2 Instance Connect Endpoint を指定できます。または、アスタリスク (*) が付いている `Resource` を使用すると、ユーザーは利用可能な任意の EC2 Instance Connect Endpoint を使用できます。条件キーとしてのリソースタグの有無に基づいて、インスタンスへのアクセスを制限することもできます。

条件

- `ec2-instance-connect:remotePort` — TCP 接続の確立に使用できるインスタンスのポートを指定します。この条件キーを使用した場合、ポリシーで指定されているポート以外のポート上のインスタンスに接続しようとすると失敗します。
- `ec2-instance-connect:privateIpAddress` — TCP 接続を確立するインスタンスに関連する宛先プライベート IP アドレスを指定します。例えば、1 つの IP アドレス (`10.0.0.1/32` など) を指定することも、CIDR を介して IP 範囲 (`10.0.1.0/28` など) を指定することもできます。この条件キーを使用した場合、別のプライベート IP アドレスまたは CIDR 範囲外のインスタンスに接続しようとすると失敗します。
- `ec2-instance-connect:maxTunnelDuration` — 確立された TCP 接続の最大期間を指定します。単位は秒で、期間の範囲は最小 1 秒から最大 3,600 秒 (1 時間) です。条件が指定されていない場合、デフォルトの時間は 3,600 秒 (1 時間) に設定されます。IAM ポリシーで指定された期間

よりも長く、またはデフォルトの最大期間よりも長くインスタンスに接続しようとすると、失敗します。指定した期間が経過すると、接続は切断されます。

maxTunnelDuration は IAM ポリシーで指定されていて、指定した値が 3,600 秒 (デフォルト) 未満の場合は、インスタンスに接続するときにコマンドで --max-tunnel-duration を指定する必要があります。インスタンスへの接続方法については、「EC2 Instance Connect Endpoint を使用して Linux インスタンスに接続する」を参照ください。

ユーザーには、EC2 Instance Connect Endpoint のリソースタグの有無に基づいて、インスタンスへの接続を確立するためのアクセス許可を付与することもできます。詳細については、「 IAM ユーザーガイド」の「IAM のポリシーとアクセス許可」を参照してください。

ec2-instance-connect:SendSSHPublicKey アクションは、パブリックキーをインスタンスにプッシュするアクセス許可を付与します。ec2:osuser 条件は、パブリックキーをインスタンスにプッシュできる OS (オペレーティングシステム) ユーザーの名前を指定します。インスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。詳細については、IAM への EC2 Instance Connect のアクセス許可の付与 をご参照ください。

IAM ポリシーの例

次の IAM ポリシー例では、IAM プリンシパルは、指定されたエンドポイント ID eice-123456789abcdef で識別される指定された EC2 Instance Connect Endpoint のみを使用してインスタンスに接続できます。接続は、すべての条件が満たされた場合にのみ正常に確立されます。例えば、SSH 接続がインスタンスのポート 22 で確立されている、インスタンスのプライベート IP アドレスが 10.0.1.0/31 の範囲 (10.0.1.0 ~ 10.0.1.1 の間) にある、maxTunnelDuration が 3600 秒以下である場合です。3600 秒(1 時間) 後 に接続が切断されます。

ec2:Describe* API アクションは、リソースレベルのアクセス許可をサポートしていません。したがって、Resource の要素には、* (ワイルドカード) が必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [{
            "Sid": "EC2InstanceConnect",
            "Action": "ec2-instance-connect:OpenTunnel",
            "Effect": "Allow",
            "Resource": "arn:aws:ec2:region:account-id:instance-connect-
endpoint/eice-123456789abcdef",
            "Condition": {
                "NumericEquals": {
```

```
                    "ec2-instance-connect:remotePort": "22"
                },
                "IpAddress": {
                    "ec2-instance-connect:privateIpAddress": "10.0.1.0/31"
                },
                "NumericLessThanEquals": {
                    "ec2-instance-connect:maxTunnelDuration": "3600"
                }
            }
        },
        {
            "Sid": "SSHPublicKey",
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:osuser": "ami-username"
                }
            }
        },
        {
            "Sid": "Describe",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceConnectEndpoints"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

ユーザーに指定した送信元 IP アドレス範囲からの接続のみを許可

次の IAM ポリシーの例では、ポリシーで指定された IP アドレス範囲内の IP アドレスから接続することを条件に、IAM プリンシパルがインスタンスに接続することを許可しています。IAM プリンシパルが 192.0.2.0/24 (このポリシーの IP アドレス範囲の例) 内ではない IP アドレスから OpenTunnel の呼び出しを行った場合、応答は Access Denied になります。詳細については、IAM ユーザーガイドaws:SourceIpの を参照してください。

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [{
            "Effect": "Allow",
            "Action": "ec2-instance-connect:OpenTunnel",
            "Resource": "arn:aws:ec2:region:account-id:instance-connect-
endpoint/eice-123456789abcdef",
            "Condition": {
                "IpAddress": {
                    "aws:SourceIp": "192.0.2.0/24"
                },
                "NumericEquals": {
                    "ec2-instance-connect:remotePort": "22"
                }
            }
        },
        {
            "Sid": "SSHPublicKey",
            "Effect": "Allow",
            "Action": "ec2-instance-connect:SendSSHPublicKey",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ec2:osuser": "ami-username"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceConnectEndpoints"
            ],
            "Resource": "*"
        }
    ]
}
```

EC2 Instance Connect Endpoint のセキュリティグループ

EC2 Instance Connect Endpoint と接続先のインスタンスには、それぞれセキュリティグループが割り当てられます。このトピックで説明されている方法でセキュリティグループルールを設定することをお勧めします。

トピック

- EC2 Instance Connect Endpoint セキュリティグループルール
- インスタンスセキュリティグループルール
- 例

EC2 Instance Connect Endpoint セキュリティグループルール

EC2 Instance Connect エンドポイントを作成するときに、セキュリティグループを指定しないと、VPC のデフォルトのセキュリティグループが割り当てられます。デフォルトのアウトバウンドルールでは、すべての宛先へのすべてのアウトバウンドトラフィックが許可されます。接続を VPC 内のインスタンスのみに制限するには、アウトバウンドルールで指定された宛先へのトラフィックのみを許可することをお勧めします。

推奨アウトバウンドルール

- 指定された宛先 (セキュリティのニーズに応じて、セキュリティグループまたは VPC CIDR) へのアウトバウンドトラフィックを許可します。

インスタンスセキュリティグループルール

インスタンスには、EC2 Instance Connect Endpoint からのトラフィックを許可するインバウンドルールが少なくとも 1 つ必要です。

推奨インバウンドルール

セキュリティのニーズと、クライアント IP 保護が有効になっているかどうかに応じて、次のルールを 1 つ以上指定します。

- EC2 Instance Connect Endpoint セキュリティグループからのインバウンドトラフィックを許可します。
- クライアント IP アドレスからのインバウンドトラフィックを許可します。
- VPC CIDR からのインバウンドトラフィックを許可して、VPC 内のすべてのインスタンスが宛先インスタンスにトラフィックを送信できるようにします。

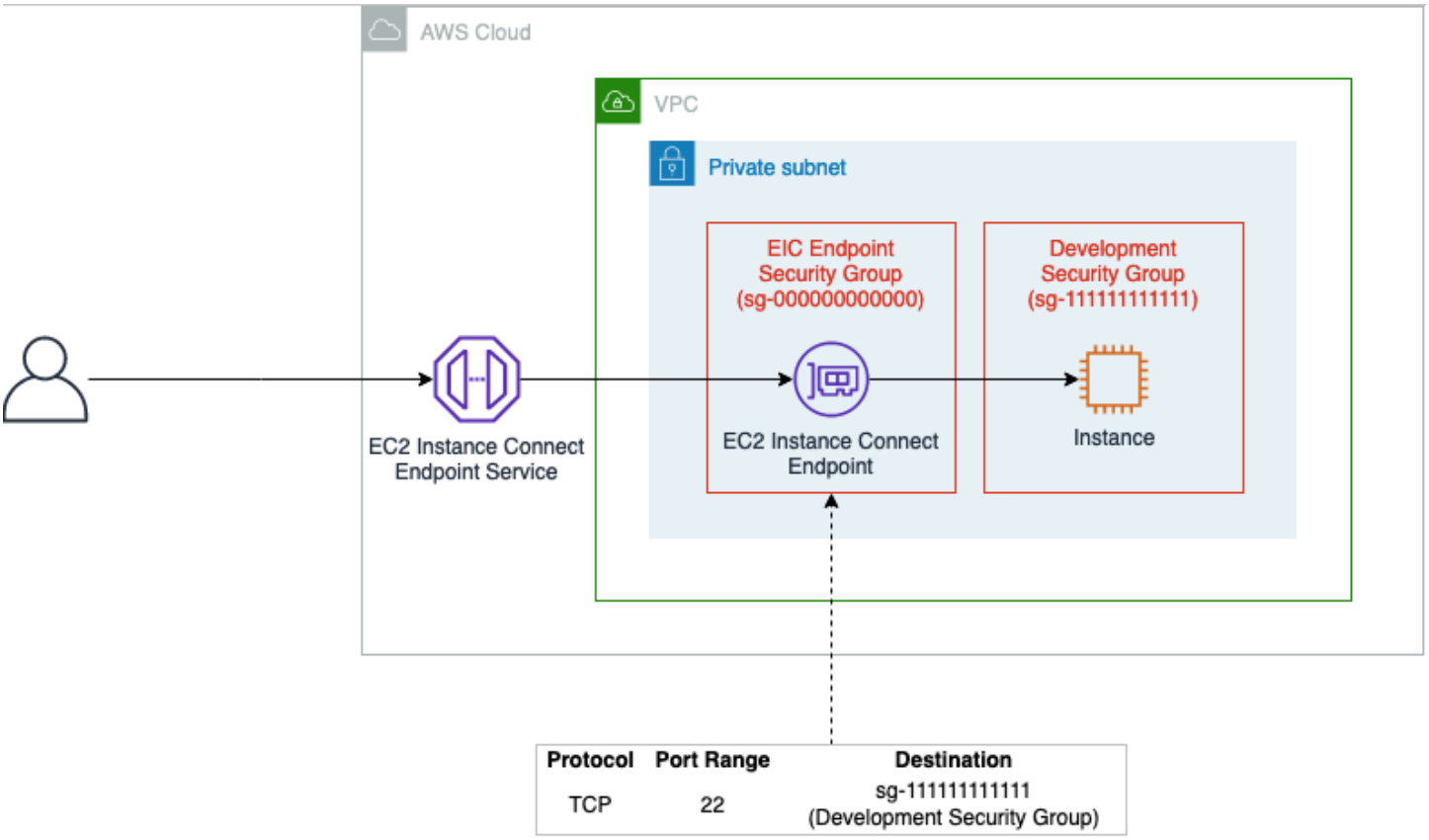
指定するインバウンドルールは、EC2 Instance Connect Endpoint がクライアント IP の保存を有効にするように設定されているかどうかによって異なります。すべてのインスタンスタイプでクライアント IP 保護がサポートされているわけではありません。詳細については、「制限事項」を参照してください。

次の表は、preserveClientIp に設定された値に応じて設定できるインスタンスのセキュリティグループルールを示しています。

クライアント IP の保存	インスタンスでサポートされているセキュリティグループルール
preserveClientIp=false	• EC2 Instance Connect Endpoint セキュリティグループからのインバウンドトラフィックを許可します。 • VPC CIDR からのインバウンドトラフィックを許可します。
preserveClientIp=true	• EC2 Instance Connect Endpoint セキュリティグループからのインバウンドトラフィックを許可します。 • クライアント IP アドレスからのインバウンドトラフィックを許可します。

例

以下のイメージでは、EC2 Instance Connect Endpoint にセキュリティグループ EIC エンドポイントセキュリティグループが割り当てられています。EIC エンドポイントセキュリティグループには、開発セキュリティグループへの TCP トラフィックを許可するアウトバウンドルールが 1 つあります。この設定は、EC2 Instance Connect Endpoint が開発セキュリティグループに割り当てられたインスタンスにのみトラフィックを送信できることを意味します。この図では、インスタンスに開発セキュリティグループが割り当てられています。つまり、この例では、EC2 Instance Connect Endpoint はインスタンスに TCP トラフィックを送信できます。



EC2 Instance Connect Endpoint の作成

VPC のサブネットに EC2 Instance Connect Endpoint を作成できます。その後、EC2 Instance Connect Endpoint を使用して、パブリック IPv4 アドレスを必要としない VPC 内のインスタンスに接続できます。

EC2 Instance Connect Endpoint はクライアント IP の保存をサポートしています。インスタンスに接続するときに、クライアントの IP アドレスをソース (`preserveClientIp` パラメータは `true`) として使用するように EC2 Instance Connect Endpoint を設定できます。

EC2 Instance Connect Endpoint を作成した場合、AWS Identity and Access Management (IAM) の Amazon EC2 サービスに対して、サービスにリンクされたロールが自動的に作成されます。Amazon EC2 は、サービスにリンクされたロールを使用してアカウントインターフェイスをプロビジョニングします。これは、EC2 Instance Connect Endpoint を作成するときに必要です。詳細については、「EC2 Instance Connect Endpoint のサービスにリンクされたロール」を参照してください。

> **Note**
>
> EC2 Instance Connect Endpoint は、作成後に変更することはできません。別のエンドポイント設定が必要な場合は、EC2 Instance Connect Endpoint を削除し、必要な設定で新しいエンドポイントを作成する必要があります。

前提条件

EC2 Instance Connect Endpoint を作成するには、必要な IAM アクセス許可が付与されている必要があります。詳細については、「ユーザーに EC2 Instance Connect Endpoint の作成、記述、削除を許可」を参照してください。

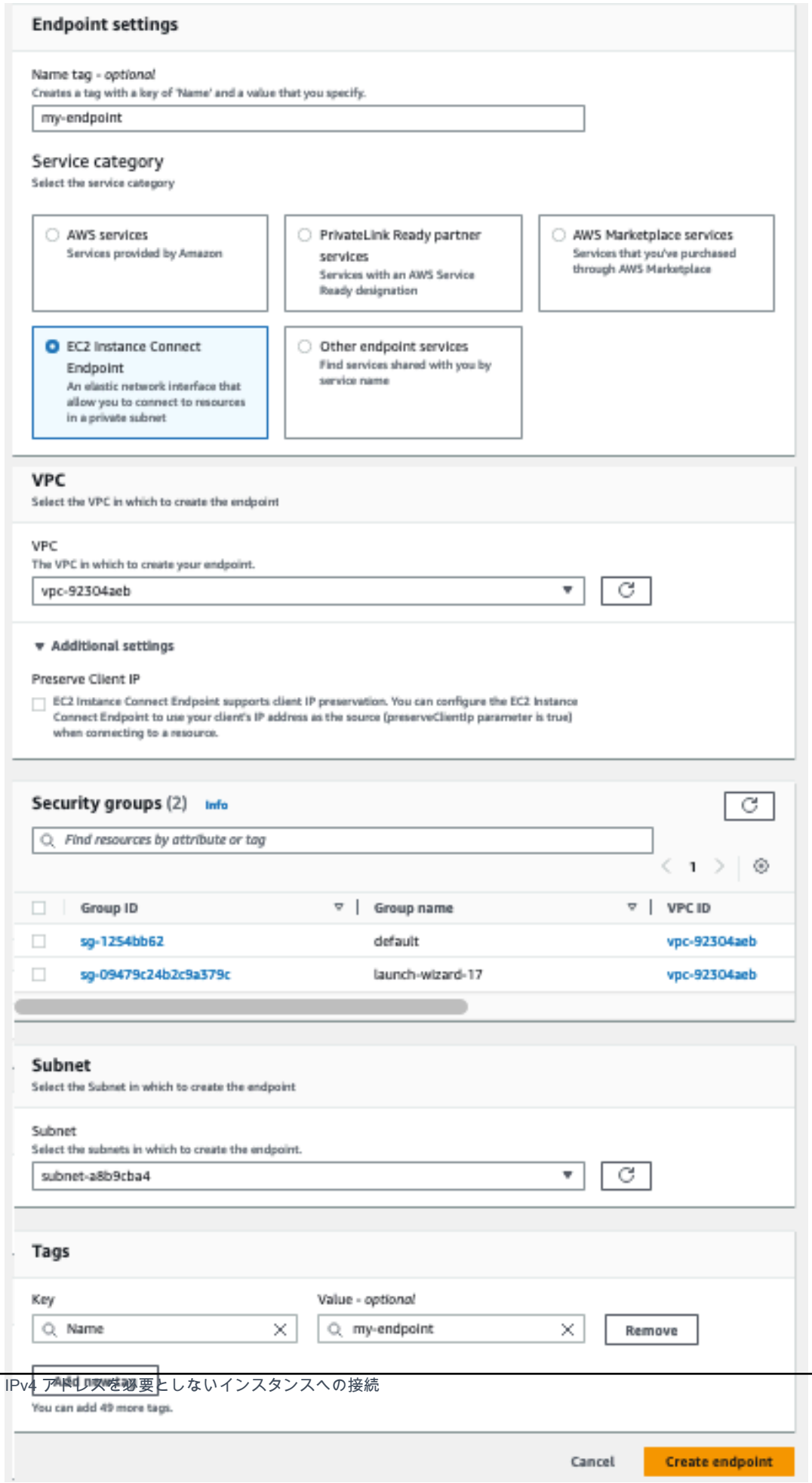
EC2 Instance Connect Endpoint の作成

EC2 Instance Connect Endpoint を作成するには、以下のメソッドのいずれかを使用します。

Console

EC2 Instance Connect Endpoint を作成するには

1. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
2. 左のナビゲーションペインで [エンドポイント] を選択します。
3. [エンドポイントの作成] を選択し、ダイアログボックスで次のように設定を完了します。


Endpoint settings
Name tag - optional
Creates a tag with a key of 'Name' and a value that you specify.
my-endpoint
Service category
Select the service category
AWS services
Services provided by Amazon
PrivateLink Ready partner services
Services with an AWS Service Ready designation
AWS Marketplace services
Services that you've purchased through AWS Marketplace
EC2 Instance Connect Endpoint
An elastic network interface that allow you to connect to resources in a private subnet
Other endpoint services
Find services shared with you by service name
VPC
Select the VPC in which to create the endpoint
VPC
The VPC in which to create your endpoint.
vpc-92304aeb
Additional settings
Preserve Client IP
EC2 Instance Connect Endpoint supports client IP preservation. You can configure the EC2 Instance Connect Endpoint to use your client's IP address as the source (preserveClientIp parameter is true) when connecting to a resource.
Security groups (2) Info
Find resources by attribute or tag
1
Group ID
Group name
VPC ID
sg-1254bb62
default
vpc-92304aeb
sg-09479c24b2c9a379c
launch-wizard-17
vpc-92304aeb
Subnet
Select the Subnet in which to create the endpoint
Subnet
Select the subnets in which to create the endpoint.
subnet-a8b9cba4
Tags
Key
Value - optional
Name
my-endpoint
Remove
Add new tag
You can add 49 more tags.
Cancel
Create endpoint

a. (オプション) [名前タグ] にエンドポイントの名前を入力します。
b. [サービスカテゴリ] で [EC2 Instance Connect Endpoint] を選択します。
c. [VPC] で、エンドポイントを作成する先の VPC を選択します。
d. [追加設定] を展開し、[クライアント IP の保持] で次のいずれかを実行します。

- チェックボックスをオンにすると、インスタンスに接続するときにクライアントの IP アドレスがソースとして使用されます。

  注: [クライアント IP アドレスの保持] がオンになっている場合、インスタンスのセキュリティグループはクライアント IP アドレスからのトラフィックを許可する必要があります。詳細については、「インスタンスセキュリティグループルール」を参照してください。
- チェックボックスをオフにすると、インスタンスに接続するときに Elastic Network Interface の IP アドレスがソースとして使用されます。[クライアント IP の保持] をオフにすると、VPC からルーティング可能な任意の IP アドレスに接続できます。

e. (オプション) [セキュリティグループ] で、エンドポイントに関連付けるセキュリティグループを選択します。セキュリティグループを選択しないと、VPC のデフォルトのセキュリティグループはエンドポイントに関連付けられます。詳細については、「EC2 Instance Connect Endpoint のセキュリティグループ」を参照してください。
f. [サブネット] で、エンドポイントを作成するサブネットを選択します。
g. (オプション) タグを追加するには、[新しいタグを追加] を選択し、そのタグのキーと値を入力します。
h. [エンドポイントの作成] を選択します。

   初期ステータスは、Pending です。このエンドポイントを使用してインスタンスに接続する前に、ステータスが [使用可能] になるまで待ってください。これは数分かかることがあります。エンドポイントの状態をモニタリングするには、「EC2 Instance Connect Endpoint の記述」を参照してください。

AWS CLI

EC2 Instance Connect Endpoint を作成するには

create-instance-connect-endpoint AWS CLIコマンドを使用して、EC2 Instance Connect Endpoint を作成するサブネットを指定します。AWS CLI の最新バージョンを利用していることを確認します。

```
aws ec2 create-instance-connect-endpoint --region us-east-1 --subnet-
id subnet-0123456789example
```

出力例

```
 {
        "VpcId": "vpc-0123abcd",
        "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
        "AvailabilityZone": "us-east-1a",
        "NetworkInterfaceIds": [
            "eni-0123abcd"
        ],
        "PreserveClientIp": true,
        "Tags": [],
        "FipsDnsName": "eice-0123456789example.0123abcd.fips.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
        "StateMessage": "",
        "State": "create-complete",
        "DnsName": "eice-0123456789example.0123abcd.ec2-instance-connect-
endpoint.us-east-1.amazonaws.com",
        "SubnetId": "subnet-0123abcd",
        "OwnerId": "111111111111",
        "SecurityGroupIds": [
            "sg-0123abcd"
        ],
        "InstanceConnectEndpointId": "eice-0123456789example",
        "CreatedAt": "2023-04-07T15:43:53.000Z"
}
```

State フィールドの初期値は create-in-progress です。このエンドポイントを使用してインスタンスに接続するには、状態が create-complete になるまで待ってください。これは数分かかることがあります。エンドポイントの状態をモニタリングするには、「EC2 Instance Connect Endpoint の記述」を参照してください。

EC2 Instance Connect Endpoint の記述

EC2 Instance Connect Endpoint を記述するには、以下のメソッドを使用します。

Console

EC2 Instance Connect Endpoint を表示するには

1. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
2. 左のナビゲーションペインで [エンドポイント] を選択します。
3. テーブルでエンドポイントを見つけて選択すると、詳細が表示されます。エンドポイントを使用してインスタンスに接続するには、ステータスフィールドは必ず表示してください利用可能。

AWS CLI

EC2 Instance Connect Endpoint を記述するには

describe-instance-connect-endpoints AWS CLIコマンドを使用して、EC2 Instance Connect Endpoint ID を指定します。

```
aws ec2 describe-instance-connect-endpoints --region us-east-1 --instance-connect-endpoint-ids eice-0123456789example
```

出力例 - インスタンスへの接続にエンドポイントを使用するには、[State] フィールドに [create-complete] が表示される必要があります。

```
{
    "InstanceConnectEndpoints": [
        {
            "OwnerId": "111111111111",
            "InstanceConnectEndpointId": "eice-0123456789example",
            "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-connect-endpoint/eice-0123456789example",
            "State": "create-complete",
            "StateMessage": "",
            "DnsName": "eice-0123456789example.b67b86ba.ec2-instance-connect-endpoint.us-east-1.amazonaws.com",
            "NetworkInterfaceIds": [
                "eni-0123456789example"
            ],
            "VpcId": "vpc-0123abcd",
            "AvailabilityZone": "us-east-1d",
            "CreatedAt": "2023-02-07T12:05:37+00:00",
            "SubnetId": "subnet-0123abcd",
```

```
            "Tags": []
        }
    ]
}
```

EC2 Instance Connect Endpoint を使用して Linux インスタンスに接続する

EC2 Instance Connect Endpoint を使用すると、パブリック IPv4 アドレスを必要としないインスタンスに接続できます。TCP をサポートしている任意のインスタンスに接続できます。

インスタンスに接続するには、インスタンス ID を指定します。オプションで EC2 Instance Connect Endpoint を指定できます。

Windows インスタンスに接続する方法については、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「EC2 Instance Connect Endpoint を使用したインスタンスの接続」を参照してください。

トピック

- 制限事項
- 前提条件
- Amazon EC2 コンソールを使用した Linux インスタンスへの接続
- SSH を使用した Linux インスタンスへの接続
- トラブルシューティング

制限事項

- ポート 22 および 3389 のみがサポートされます。
- EC2 Instance Connect Endpoint は IPv6 アドレスを使用する接続をサポートしていません。
- 各 EC2 インスタンス接続エンドポイントは、最大 20 の同時接続をサポートできます。
- EC2 Instance Connect Endpoint は、特に管理トラフィックのユースケースを対象としており、大量のデータ転送を想定していません。大量のデータ転送はスロットリングされます。
- TCP 接続が確立されている最大持続時間: 1 時間 (3,600 秒)。IAM ポリシーでは最大許容時間を 3,600 秒以下で指定できます。詳細については、「ユーザーに EC2 Instance Connect Endpoint を使用したインスタンスへの接続を許可」を参照してください。
- クライアント IP の保存が有効の場合、接続するインスタンスは EC2 Instance Connect Endpoint と同じ VPC にある必要があります。

- トラフィックが AWS Transit Gateway を経由してルーティングされる場合、クライアント IP の保存はサポートされません。
- インスタンスタイプが C1、CG1、CG2、G1、G2、HI1、M1、M2、M3、T1である場合、クライアント IP 保存をサポートしません。これらのインスタンスタイプを使用している場合は、`preserveClientIp` パラメータを `false` に設定します。そうでない場合、EC2 Instance Connect Endpoint を使用してこれらのインスタンスタイプに接続しようとすると失敗します。`preserveClientIp` パラメータの詳細については、EC2 Instance Connect Endpoint の作成 コンソールプロシージャのステップ 3.d を参照してください。

前提条件

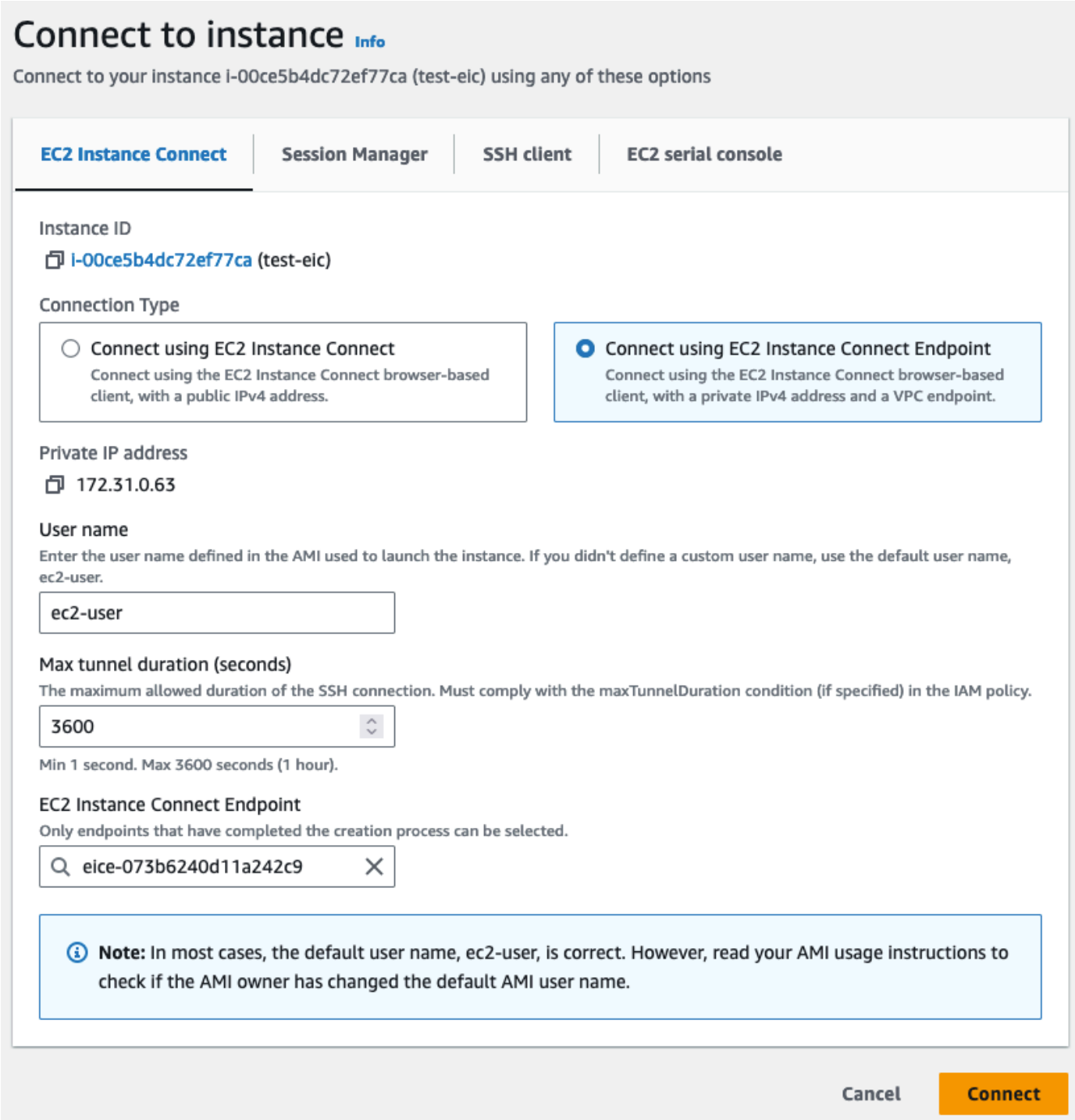
- EC2 Instance Connect Endpoint に接続するには、必要な IAM アクセス許可が必要です。詳細については、「ユーザーに EC2 Instance Connect Endpoint を使用したインスタンスへの接続を許可」を参照してください。
- EC2 Instance Connect Endpoint は [使用可能] (コンソール) または [`create-complete`] (AWS CLI) 状態である必要があります。エンドポイントの状態をモニタリングするには、「EC2 Instance Connect Endpoint の記述」を参照してください。
- EC2 コンソールを使用してインスタンスに接続する場合、または CLI を使用して接続し、EC2 Instance Connect にエフェメラルキーを処理させる場合は、インスタンスに EC2 Instance Connect がインストールされている必要があります。詳細については、「AMI」を参照してください。
- 接続するインスタンスのセキュリティグループがインバウンドトラフィックに対して正しく設定されていることを確認します。詳細については、「インスタンスセキュリティグループルール」を参照してください。
- AWS CLI を使用している場合は、AWS CLI を設定している (使用する認証情報を含む) ことと、AWS CLI の最新バージョンを使用していることを確認します。詳細については、「AWS Command Line Interface ユーザーガイド」の「AWS CLI の最新バージョンのインストールまたは更新」および「AWS CLI の設定」を参照してください。

Amazon EC2 コンソールを使用した Linux インスタンスへの接続

コンソールからインスタンスを選択し、EC2 Instance Connect を使用する選択をすることで、Amazon EC2 コンソールを使用してインスタンスに接続できます。これにより、アクセス許可を処理して正常な接続を供給します。

Amazon EC2 コンソールからブラウザベースのクライアントを使用してインスタンスに接続するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択して [接続] を選択し、次の操作を行います。



a. [EC2 Instance Connect] タブを選択します。
b. [接続タイプ] で、[EC2 Instance Connect Endpoint を使用して接続] を選択します。
c. [ユーザー名] で、ユーザー名を確認します。
d. [最大トンネル期間 (秒)] に、SSH 接続の最大許容期間を入力します。

   期間は IAM ポリシーで指定されている `maxTunnelDuration` 条件を満たしている必要があります。IAM ポリシーにアクセスできない場合は、管理者に問い合わせて、IAM ポリシーを確認してください。`maxTunnelDuration` が IAM ポリシーで指定されていない場合は、デフォルトの 3600 秒 (1 時間) を入力します。
e. EC2 Instance Connect Endpoint については、インスタンスの VPC にある EC2 Instance Connect Endpoint を選択します。
f. [接続] を選択してターミナルウィンドウを開きます。

SSH を使用した Linux インスタンスへの接続

SSH を使用して Linux インスタンスに接続し、`open-tunnel` コマンドを使用してプライベートトンネルを確立できます。`open-tunnel` はシングル接続またはマルチ接続モードで使用できます。

SSH を使用したインスタンスに接続するための AWS CLI の使用の詳細については、「EC2 Instance Connect を使用して、AWS CLI で Linux インスタンスに接続します。」を参照してください。

以下の例では OpenSSH を使用しています。プロキシモードをサポートする他の SSH クライアントを使用できます。

シングル接続

SSH と **`open-tunnel`** コマンドを使用してインスタンスに 1 つの接続のみを許可するには

`ssh` と open-tunnel AWS CLI コマンドを次のように使用します。`-o` プロキシコマンドには、インスタンスへのプライベートトンネルを作成する `open-tunnel` コマンドが含まれています。

```
ssh -i my-key-pair.pem ec2-user@i-0123456789example \
    -o ProxyCommand='aws ec2-instance-connect open-tunnel --instance-
id i-0123456789example'
```

内容:

- `-i` — インスタンスの起動に使用されたキーペアを指定します。

- *ec2-user*@*i-0123456789example* — インスタンスの起動に使用された AMI のユーザー名とインスタンス ID を指定します。
- --instance-id — 接続するインスタンスの ID を指定します。または、ユーザーからインスタンス ID を抽出するように %h を指定します。

マルチ接続

1 つのインスタンスに複数の接続を許可するには、最初に open-tunnel AWS CLI コマンドを実行して新しい TCP 接続のリスニングを開始し、次に ssh を使用して新しい TCP 接続とインスタンスへのプライベートトンネルを作成します。

SSH と **open-tunnel** コマンドを使用してインスタンスへの複数の接続を許可するには

1. ローカルマシンの特定のポートで新しい TCP 接続のリスニングを開始するには、次のコマンドを実行します。

```
aws ec2-instance-connect open-tunnel \
    --instance-id i-0123456789example \
    --local-port 8888
```

正常な出力

```
Listening for connections on port 8888.
```

2. 新しいターミナルウィンドウで、次の ssh コマンドを実行して、新しい TCP 接続とインスタンスへのプライベートトンネルを作成します。

```
ssh -i my-key-pair.pem ec2-user@localhost -p 8888
```

期待される出力 — 最初のターミナルウィンドウには、以下が表示されます。

```
[1] Accepted new tcp connection, opening websocket tunnel.
```

以下が表示される可能性があります。

```
[1] Closing tcp connection.
```

トラブルシューティング

以下の情報は、EC2 Instance Connect Endpoint を使用してインスタンスを接続するときに発生する可能性のある問題の診断と修復に役立ちます。

インスタンスに接続できない

インスタンスに接続できない一般的な理由は次のとおりです。

- セキュリティ グループ – EC2 Instance Connect Endpoint とインスタンスに割り当てられたセキュリティグループを確認してください。必要なセキュリティグループルールの詳細については、「EC2 Instance Connect Endpoint のセキュリティグループ」を参照してください。
- インスタンスの状態 – インスタンスの状態が [running] であることを確認します。
- キーペア — 接続に使用しているコマンドにプライベートキーが必要な場合は、インスタンスにパブリックキーと、対応するプライベートキーがあることを確認します。
- IAM アクセス許可 — 必要な IAM アクセス許可があることを確認します。詳細については、「EC2 Instance Connect Endpoint を使用するための IAM アクセス許可の付与」を参照してください。

トラブルシューティングのヒントについては、「インスタンスへの接続に関するトラブルシューティング」と「」を参照してください。

ErrorCode: AccessDeniedException

`AccessDeniedException` エラーが発生し、`maxTunnelDuration` 条件が IAM ポリシーで指定されている場合は、インスタンスに接続するときに必ず `--max-tunnel-duration` パラメータを指定してください。このパラメータの詳細については「AWS CLIコマンド リファレンス」の「open-tunnel」を参照してください。

EC2 Instance Connect Endpoint を介して確立された接続のログの作成

リソース操作のログを作成し、AWS CloudTrail ログを使用して EC2 Instance Connect Endpoint を介して確立された接続を監査できます。

Amazon EC2 での AWS CloudTrail の使用に関する詳細は、「AWS CloudTrail による Amazon EC2 および Amazon EBS の API コールのログ記録」を参照してください。

AWS CloudTrail を使用した EC2 Instance Connect Endpoint API コールのログの作成

EC2 Instance Connect Endpoint のリソース操作は、管理イベントとして CloudTrail にログが作成されます。次の API コールが行われると、アクティビティは CloudTrail イベントとして [イベント履歴] にログが作成されます。

- `CreateInstanceConnectEndpoint`
- `DescribeInstanceConnectEndpoints`
- `DeleteInstanceConnectEndpoint`

最近のイベントは、AWS アカウント で表示、検索、ダウンロードできます。詳細については、「AWS CloudTrail ユーザーガイド 」の「CloudTrail イベント履歴でのイベントの表示」を参照してください。

EC2 Instance Connect Endpoint を使用してインスタンスに接続するユーザーを監査するための AWS CloudTrail の使用

EC2 Instance Connect Endpoint を経由してインスタンスに接続を試みると、CloudTrail の [イベント履歴] にログが作成されます。インスタンスへの接続が EC2 Instance Connect Endpoint を経由して開始されると、その接続は `OpenTunnel` の `eventName` という CloudTrail 管理イベントとしてログが作成されます。

CloudTrail イベントをターゲットにルーティングする Amazon EventBridge ルールを作成できます。詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

CloudTrail にログが作成された `OpenTunnel` 管理イベントの例を以下に示します。

```
{
     "eventVersion": "1.08",
     "userIdentity": {
         "type": "IAMUser",
         "principalId": "ABCDEFGONGNOMOOCB6XYTQEXAMPLE",
         "arn": "arn:aws:iam::1234567890120:user/IAM-friendly-name",
         "accountId": "123456789012",
         "accessKeyId": "ABCDEFGUKZHNAW4OSN2AEXAMPLE",
         "userName": "IAM-friendly-name"
     },
     "eventTime": "2023-04-11T23:50:40Z",
     "eventSource": "ec2-instance-connect.amazonaws.com",
     "eventName": "OpenTunnel",
```

```
    "awsRegion": "us-east-1",
    "sourceIPAddress": "1.2.3.4",
    "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
    "requestParameters": {
        "instanceConnectEndpointId": "eici-0123456789EXAMPLE",
        "maxTunnelDuration": "3600",
        "remotePort": "22",
        "privateIpAddress": "10.0.1.1"
    },
    "responseElements": null,
    "requestID": "98deb2c6-3b3a-437c-a680-03c4207b6650",
    "eventID": "bbba272c-8777-43ad-91f6-c4ab1c7f96fd",
    "readOnly": false,
    "resources": [{
        "accountId": "123456789012",
        "type": "AWS::EC2::InstanceConnectEndpoint",
        "ARN": "arn:aws:ec2:us-east-1:123456789012:instance-connect-endpoint/
eici-0123456789EXAMPLE"
    }],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
    "eventCategory": "Management"
}
```

EC2 Instance Connect Endpoint の削除

VPC から EC2 Instance Connect Endpoint を削除するには、サブネットで作成されたエンドポイントを削除します。

EC2 Instance Connect Endpoint を削除すると、最初に [削除中] (コンソール) または [delete-in-progress] (AWS CLI) 状態になり、次に [delete-complete] (AWS CLI) 状態になります。削除されたエンドポイントはコンソールに表示されなくなります。削除アクションが失敗した場合、状態は [delete-failed] になり、[ステータスメッセージ] (コンソール) または [StateMessage] (AWS CLI) により失敗理由が示されます。

EC2 Instance Connect Endpoint を削除するには、以下のメソッドのいずれかを使用します。

Console

EC2 Instance Connect Endpoint を削除するには

1. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。

2. 左のナビゲーションペインで [エンドポイント] を選択します。
3. エンドポイントを選択します。
4. [Actions] (アクション)、[Delete VPC endpoints] (VPC エンドポイントを削除) の順に選択します。
5. 確認を求められたら、**delete** をクリックします。
6. [削除] を選択します。

AWS CLI

EC2 Instance Connect Endpoint を削除するには

delete-instance-connect-endpoints AWS CLI コマンドを使用し、削除する EC2 Instance Connect Endpoint の ID を指定します。

```
aws ec2 delete-instance-connect-endpoint --instance-connect-endpoint-
id eice-03f5e49b83924bbc7
```

出力例

```
{
    "InstanceConnectEndpoint": {
        "OwnerId": "111111111111",
        "InstanceConnectEndpointId": "eice-0123456789example",
        "InstanceConnectEndpointArn": "arn:aws:ec2:us-east-1:111111111111:instance-
connect-endpoint/eice-0123456789example",
        "State": "delete-in-progress",
        "StateMessage": "",
        "NetworkInterfaceIds": [],
        "VpcId": "vpc-0123abcd",
        "AvailabilityZone": "us-east-1d",
        "CreatedAt": "2023-02-07T12:05:37+00:00",
        "SubnetId": "subnet-0123abcd"
    }
}
```

EC2 Instance Connect Endpoint のサービスにリンクされたロール

Amazon EC2 は、AWS Identity and Access Management (IAM) サービスにリンクされたロールを使用します。サービスにリンクされたロールは、Amazon EC2 に直接リンクされた一意のタイプ

の IAM ロールです。サービスにリンクされたロールは Amazon EC2 によって事前に定義されており、Amazon EC2 がユーザーに代わって他の AWS のサービスを呼び出すために必要なすべてのアクセス許可がロールには含まれています。詳細については、「 IAM ユーザーガイド」の「サービスにリンクされたロールの使用」を参照してください。

EC2 Instance Connect Endpoint を作成すると、AWSServiceRoleForEC2InstanceConnect という名前のサービスにリンクされたロールと、EC2InstanceConnectEndpoint という名前の管理ポリシーが自動的に AWS アカウントに作成され、管理ポリシーがサービスにリンクされたロールに自動的にアタッチされます。

Amazon EC2 は、AWSServiceRoleForEC2InstanceConnect を使用して、EC2 Instance Connect Endpoint を作成する際に必要なアカウントのネットワークインターフェイスを管理します。

AWSServiceRoleForEC2InstanceConnect によって付与されるアクセス許可

Amazon EC2 は、AWSServiceRoleForEC2InstanceConnect を使用して、次のアクションを実行します。

- `ec2:CreateNetworkInterface` – ネットワークインターフェイスを作成する
- `ec2:DeleteNetworkInterface` – ネットワークインターフェイスを削除する
- `ec2:DescribeNetworkInterfaces` – ネットワークインターフェイスを記述する
- `ec2:DescribeAvailabilityZones` – アベイラビリティーゾーンを記述する
- `ec2:ModifyNetworkInterfaceAttribute` – 送信元/送信先チェックを無効にする

サービスにリンクされたロールの使用

EC2 Instance Connect Endpoint は、AWSServiceRoleForEC2InstanceConnect という名前のサービスにリンクされたロールを使用して、サービスの使用に必要なネットワークインターフェイスをアカウントにプロビジョニングします。

EC2 Instance Connect Endpoint を作成すると、EC2InstanceConnectEndpoint 管理ポリシーが AWS アカウントに自動的に作成され、AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールにアタッチされます。

EC2 Instance Connect Endpoint のサービスにリンクされたロール

AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールは、以下のサービスを信頼してロールを引き受けます。

- ec2-instance-connect.amazonaws.com

EC2InstanceConnectEndpoint という名前のロールのアクセス許可ポリシーは、EC2 Instance Connect Endpoint が、以下の特定のリソースでのアクションを実行できるようにします。

- アクション: `ec2:CreateNetworkInterface` — EC2 Instance Connect Endpoint のネットワークインターフェイスを作成するための NULL 以外のタグキー InstanceConnectEndpointId を持つすべてのサブネットとすべてのネットワークインターフェイス上で

- アクション: `ec2:CreateTags` — 作成時にタグキーの InstanceConnectEndpointId を使用して EC2 Instance Connect Endpoint 用に作成されたすべてのネットワークインターフェイス上で

- アクション: `ec2:DeleteNetworkInterface` — タグキーの InstanceConnectEndpointId を使用して EC2 Instance Connect Endpoint 用に作成されたネットワークインターフェイス上で

- アクション: `ec2:DescribeNetworkInterfaces` — Instance Connect Endpoint のネットワークインターフェイス上で

- アクション: `ec2:DescribeAvailabilityZones` — お客様のアベイラビリティーゾーンの内部マッピング用

- アクション: `ec2:ModifyNetworkInterfaceAttribute` — 送信元と宛先のチェックを無効にするためのすべてのネットワークインターフェイスで

信頼ポリシー

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
```

```
            "Principal": {
                "Service": "ec2-instance-connect.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

アクセス許可ポリシー

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAvailabilityZones"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": "arn:aws:ec2:*:*:subnet/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "InstanceConnectEndpointId"
                    ]
                },
                "Null": {
                    "aws:RequestTag/InstanceConnectEndpointId": "false"
                }
```

```
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ModifyNetworkInterfaceAttribute"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/InstanceConnectEndpointId": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateNetworkInterface"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": [
                        "InstanceConnectEndpointId"
                    ]
                },
                "Null": {
                    "aws:RequestTag/InstanceConnectEndpointId": "false"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteNetworkInterface"
            ],
            "Resource": "*",
            "Condition": {
                "StringLike": {
                    "aws:ResourceTag/InstanceConnectEndpointId": [
                        "eice-*"
```

```
                    ]
                }
            }
        }
    ]
}
```

EC2 Instance Connect Endpoint のサービスにリンクされたロールの作成

EC2 Instance Connect Endpoint を作成すると、サービスにリンクされたロール AWSServiceRoleForEC2InstanceConnect が自動的に作成されます。

> **Important**
>
> EC2 Instance Connect Endpoint の作成に使用された AWS アカウントに、`iam:CreateServiceLinkedRole` アクションを許可する IAM ポリシーがアタッチされていることを確認します。

EC2 Instance Connect Endpoint のサービスにリンクされたロールの編集

EC2 Instance Connect Endpoint では、AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールを編集することはできません。

EC2 Instance Connect Endpoint のサービスにリンクされたロールの削除

EC2 Instance Connect Endpoint を使用する必要がなくなった場合は、AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールを削除することをお勧めします。

> **Note**
>
> サービスにリンクされたロールを削除するには、EC2 Instance Connect Endpoint のリソースをすべて削除する必要があります。

AWS CLI を使用して、サービスにリンクされたロールを削除します。詳細については、 IAM ユーザーガイド の「サービスにリンクされたロールの削除」を参照してください。

AWS CLI を使用して、サービスにリンクされたロールを削除するには、次のステップを実行します。

1. `delete-instance-connect-endpoint` コマンドを使用してすべての EC2 Instance Connect Endpoint を削除します。これにより、関連するリソースも削除されます。
2. `delete-service-linked-role` コマンドを使用して、サービスにリンクされたロールを削除します。サービスにリンクされたロールを削除すると、関連する管理ポリシーも削除されます。

EC2 Instance Connect Endpoint は、サービスが利用可能なすべての AWS リージョンで AWSServiceRoleForEC2InstanceConnect サービスにリンクされたロールの使用をサポートしています。

EC2 Instance Connect Endpoint の AWS 管理ポリシー

AWS マネージドポリシー: EC2InstanceConnectEndpoint

このポリシーは、EC2 Instance Connect Endpoint がユーザーに代わってアクションを実行できるようにするサービスにリンクされたロールに関連付けられています。詳細については、「EC2InstanceConnectEndpoint」を参照してください。

このポリシーのアクセス許可を確認するには、「AWS Management Console」の「Ec2InstanceConnectEndpoint」を参照してください。

AWS 管理ポリシーに対する EC2 Instance Connect Endpoint の更新

このサービスがこれらの変更の追跡を開始してからの、EC2 Instance Connect Endpoint の AWS 管理ポリシーの更新に関する詳細を表示します。

変更	説明	日付
EC2 Instance Connect Endpoint が変更の追跡を開始しました	EC2 Instance Connect Endpoint は、AWS 管理ポリシーの変更の追跡を開始しました。	2023 年 6 月 13 日

クォータ

次のように AWS リージョン ごとに最大数の EC2 Instance Connect Endpoint を作成できます。

説明	クォータ
AWS アカウント ごとの AWS リージョン あたりの EC2 Instance Connect Endpoint の最大数	5
VPC あたりの EC2 Instance Connect Endpoint の最大数	1
サブネットあたりの EC2 Instance Connect Endpoint の最大数	1

各 EC2 Instance Connect Endpoint は、次のように同時接続の最大数をサポートできます。

説明	クォータ
EC2 Instance Connect Endpoint あたりの同時接続の最大数	20

EC2 インスタンスを AWS リソースに接続する

インスタンスを起動したら、そのインスタンスを 1 つまたは複数の AWS リソースに接続できます。

このセクションでは、Amazon EC2 インスタンスを Amazon RDS データベースに自動接続する方法を説明しています。

EC2 インスタンスを RDS データベースに自動接続する

Amazon EC2 コンソールの自動接続機能を使用すると、1 つ以上の EC2 インスタンスを RDS データベースにすばやく接続し、これらの間のトラフィックを許可することができます。

詳細については、「接続が自動的に構成される方法」を参照してください。EC2 インスタンスと RDS データベースを接続する他の方法を含む詳細な手順については、チュートリアル: Amazon RDS データベースに Amazon EC2 インスタンスを接続する を参照してください。

トピック

- コスト
- 前提条件

- インスタンスとデータベースを自動接続する
- 接続が自動的に構成される方法

コスト

EC2 インスタンスを RDS データベースに自動接続する際の料金はかかりませんが、基盤となるサービスには課金されます。EC2 インスタンスと RDS データベースが異なるアベイラビリティーゾーンにある場合は、データ転送料金がかかります。データ転送料金の詳細については、「Amazon EC2 オンデマンド料金」ページの「データ転送」を参照してください。

前提条件

EC2 インスタンスを RDS データベースに自動接続する前に、以下を確認してください。

- EC2 インスタンスは [Running] (実行中) 状態である必要があります。EC2 インスタンスが別の状態にある場合は、接続できません。
- EC2 インスタンスと RDS データベースは同じ仮想プライベートクラウド (VPC) 内に存在する必要があります。EC2 インスタンスと RDS データベースが異なる VPC にある場合、自動接続機能はサポートされません。

インスタンスとデータベースを自動接続する

インスタンスを起動した直後またはその後に、EC2 インスタンスを RDS データベースに自動接続できます。

起動直後に自動接続する

EC2 インスタンスを起動した直後に EC2 インスタンスを RDS データベースに自動接続するには、次のステップを使用します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: 新しく起動した EC2 インスタンスを RDS データベースに自動接続する」を参照してください。

EC2 コンソールを使用して、新しく起動した EC2 インスタンスを RDS データベースに自動接続する場合

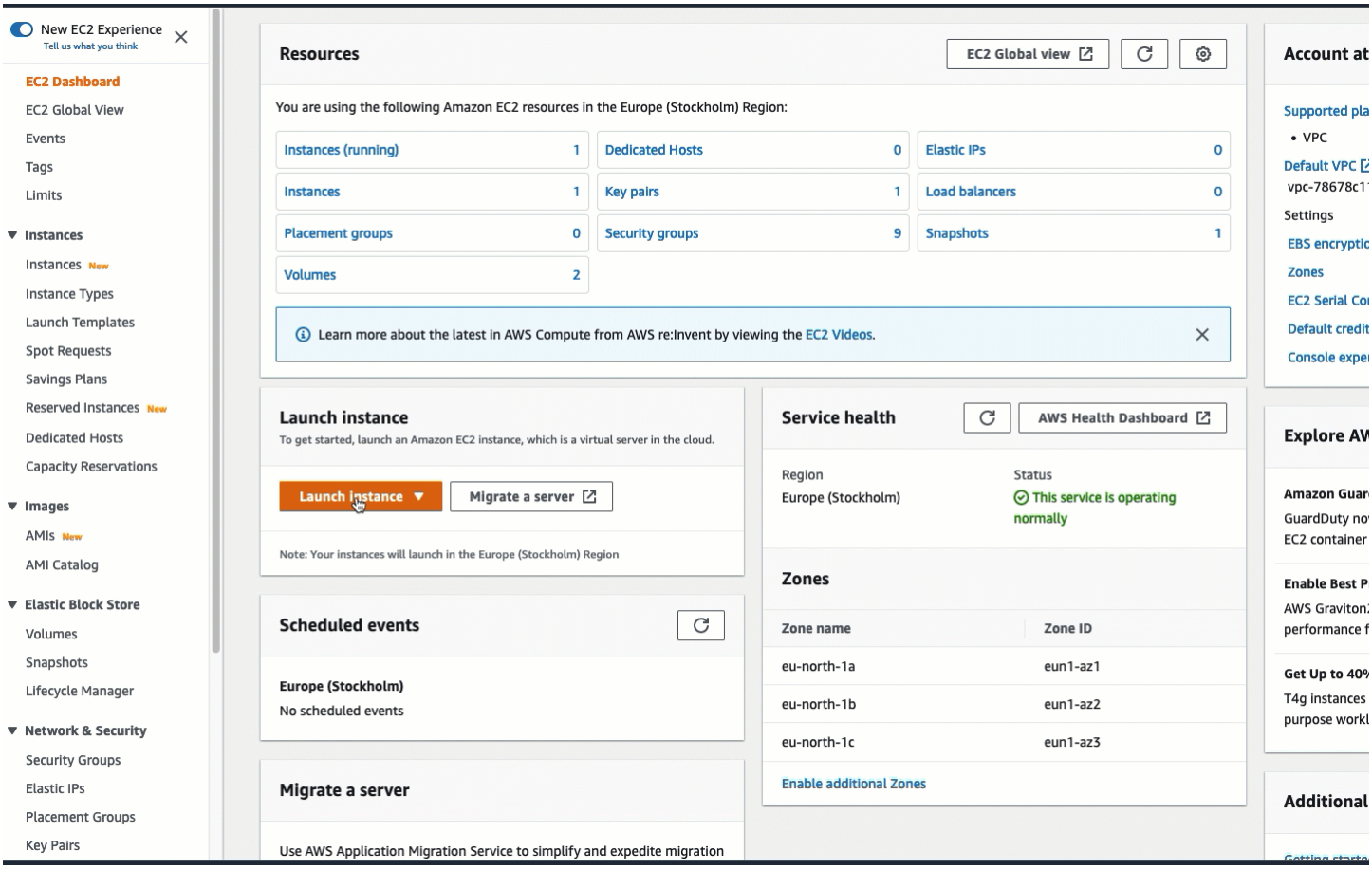
1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. コンソールダッシュボードで [Launch instance] (インスタンスを起動する) を選択し、その後ステップをに従ってインスタンスを起動します。

3. インスタンスの起動確認ページで、[Connect an RDS database] (RDS データベースに接続) を選択します。
4. [Connect RDS Database] (RDS データベースに接続) ダイアログボックスで、次を実行します。
   a. [Database role] (データベースロール) には、[Cluster] (クラスター) または [Instance] (インスタンス) のいずれかを選択します。
   b. [RDS database] (RDS データベース) の場合は、接続するデータベースを選択します。

> **Note**
>
> EC2 インスタンスと RDS データベースが相互に接続するには、同じ VPC 内にある必要があります。

   c. [接続]を選択します。

アニメーションを表示: 新しく起動した EC2 インスタンスを RDS データベースに自動接続する



既存のインスタンスに自動接続する

既存の EC2 インスタンスを RDS データベースに自動接続するには、次のステップを使用します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: 既存の EC2 インスタンスを RDS データベースに自動接続する」を参照してください。

EC2 コンソールを使用して、既存の EC2 インスタンスを RDS データベースに自動接続する場合

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. RDS データベースに接続する 1 つ以上の EC2 インスタンスを選択し、[Actions] (アクション)、[Networking] (ネットワーク)、[Connect RDS database] (RDS データベースに接続) を選択します。

   [Connect RDS database] (RDS データベースに接続) を選択できない場合は、EC2 インスタンスが [Running] (実行中) の状態であり、同じ VPC 内にあることを確認します。
4. [Connect RDS Database] (RDS データベースに接続) ダイアログボックスで、次を実行します。

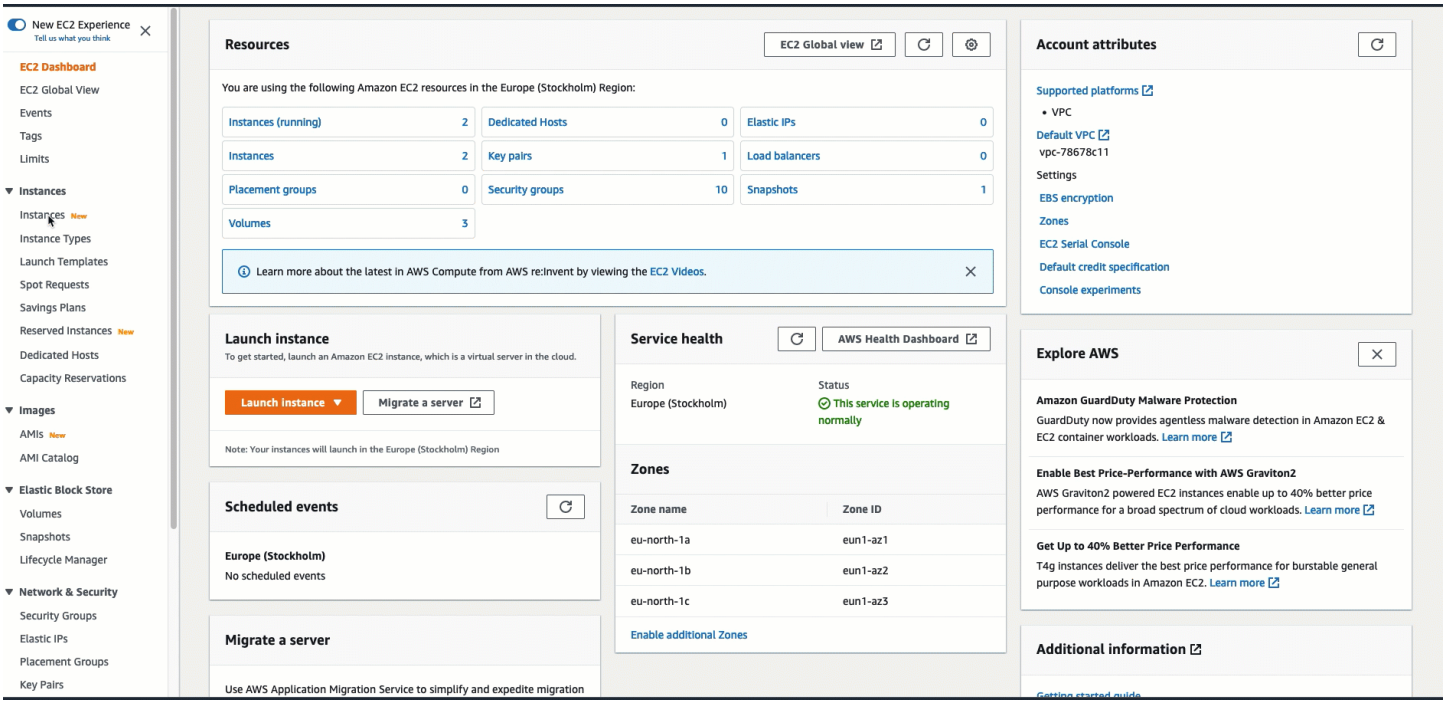
   a. [Database role] (データベースロール) には、[Cluster] (クラスター) または [Instance] (インスタンス) のいずれかを選択します。

   b. [RDS database] (RDS データベース) の場合は、接続するデータベースを選択します。

      > **Note**
      >
      > EC2 インスタンスと RDS データベースが相互に接続するには、同じ VPC 内にある必要があります。

   c. [接続]を選択します。

アニメーションを表示: 既存の EC2 インスタンスを RDS データベースに自動接続する



Amazon RDS コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する方法については、Amazon RDS ユーザーガイドにある「Configure automatic network connectivity with an EC2 instance」(EC2 インスタンスで自動ネットワーク接続を設定する) を参照してください。

接続が自動的に構成される方法

EC2 コンソールを使用して EC2 インスタンスと RDS データベース間のトラフィックを許可する接続を自動で設定する場合、接続はセキュリティグループによって設定されます。

セキュリティグループは、次のように自動作成され、EC2 インスタンスと RDS データベースに追加されます。

- Amazon EC2 は ec2-rds-*x* という名前のセキュリティグループを作成し、それを EC2 インスタンスに追加します。ここには、rds-ec2-*x* (データベースのセキュリティグループ) を送信先として指定することでデータベースへのトラフィックを許可するアウトバウンドルールが 1 つあります。
- Amazon RDS は rds-ec2-*x* という名前のセキュリティグループを作成し、それをデータベースに追加します。ここには、ec2-rds-*x* (EC2 インスタンスのセキュリティグループ) をソースとして指定することで EC2 インスタンスからのトラフィックを許可するインバウンドルールが 1 つあります。

セキュリティグループはお互いを送信先および送信元として参照し、データベースポート上でのみトラフィックを許可します。これらのセキュリティグループは再利用して、rds-ec2-*x* セキュリティグループが含まれる任意のデータベースが、ec2-rds-*x* セキュリティグループが含まれる任意の EC2 インスタンスと通信できるようにすることができます。

セキュリティグループ名はパターンに従って命名されます。Amazon EC2 によって作成されたセキュリティグループの場合、パターンは ec2-rds-*x* であり、Amazon RDS によって作成されたセキュリティグループの場合、パターンは rds-ec2-*x* になります。*x* は数値で、新しいセキュリティグループが自動的に作成されるたびに 1 ずつ増加します。

チュートリアル: Amazon RDS データベースに Amazon EC2 インスタンスを接続する

チュートリアルの目的

このチュートリアルでは、AWS Management Console を使用して、Amazon EC2 インスタンスと Amazon RDS データベース間のセキュア接続を設定する方法について説明します。

接続を設定する方法は複数あります。このチュートリアルでは、以下の 3 つのオプションについて説明します。

- オプション 1: EC2 コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する

  EC2 コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。
- オプション 2: RDS コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する

  RDS コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。
- オプション 3: 自動接続機能を模倣して EC2 インスタンスを RDS データベースに手動で接続する

  EC2 インスタンスと RDS データベース間の接続を設定するには、セキュリティグループを手動で設定してから、セキュリティグループを割り当てて、オプション 1 とオプション 2 の自動接続機能が自動作成する設定を再現します。

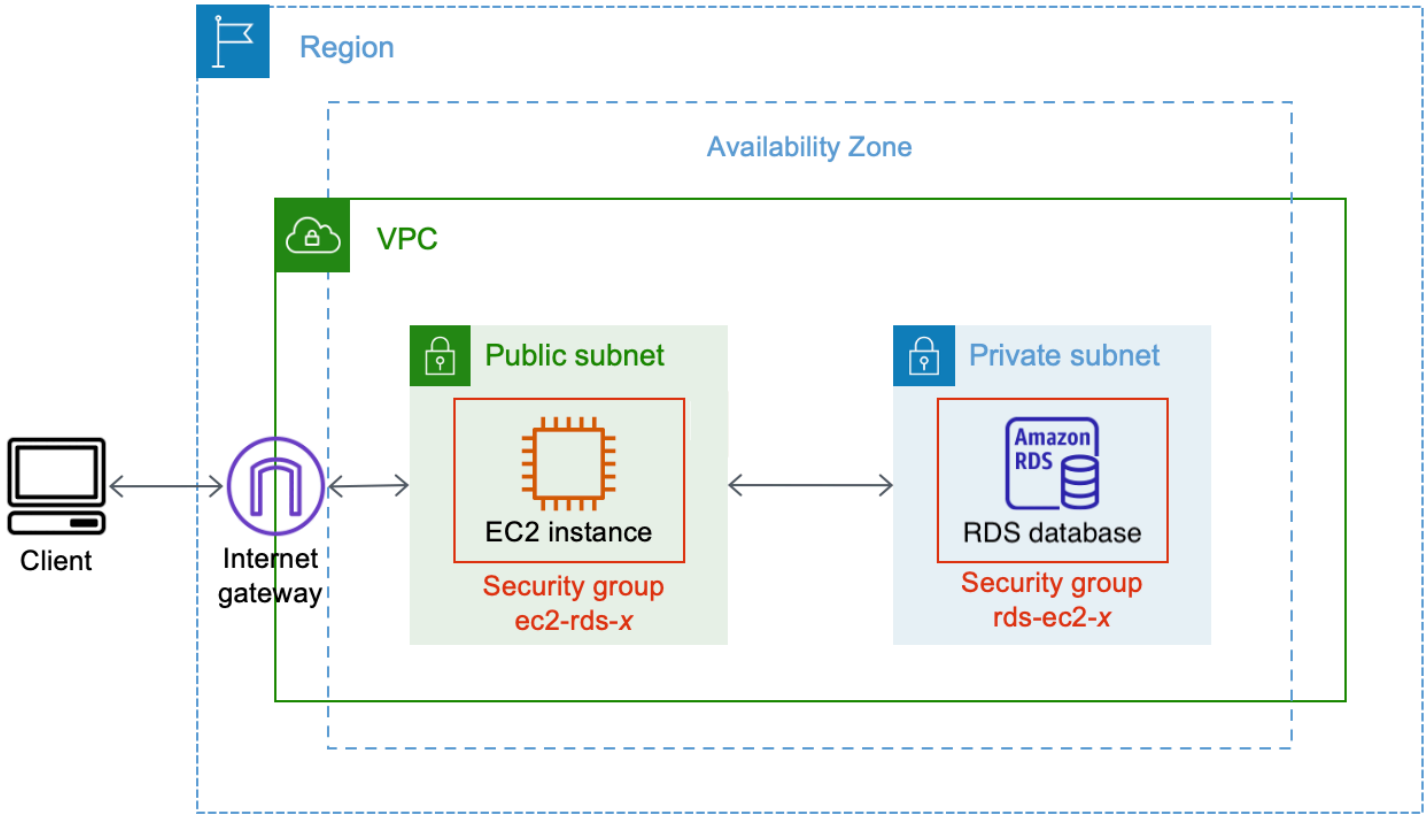
Context

EC2 インスタンスと RDS データベース間の接続を設定する理由の背景として、次のシナリオを考えてみましょう。ウェブサイトでユーザーに入力してもらうフォームがある場合、フォームデータ

をデータベースに取り込む必要があります。ウェブサーバーとして設定された EC2 インスタンスでウェブサイトをホストし、フォームデータを RDS データベースに取得できます。フォームデータを EC2 インスタンスから RDS データベースに送信するには、EC2 インスタンスと RDS データベースを相互接続する必要があります。このチュートリアルでは、その接続を構成する方法について説明します。なお、これは EC2 インスタンスと RDS データベースを接続するユースケースの一例にすぎません。

アーキテクチャ

次の図は、作成されるリソースと、このチュートリアルのすべてのステップを完了した結果のアーキテクチャ構成を示しています。



図は、作成する以下のリソースを示しています。

- 同じ AWS リージョン、VPC、およびアベイラビリティーゾーンに EC2 インスタンスと RDS データベースを作成します。
- パブリックサブネットに EC2 インスタンスを作成します。
- プライベートサブネットに RDS データベースを作成します。

RDS コンソールを使用して RDS データベースを作成し、EC2 インスタンスに自動接続すると、VPC、DB サブネットグループ、およびデータベースのパブリックアクセス設定が自動的に選択されます。RDS データベースは、EC2 インスタンスと同じ VPC 内のプライベートサブネットに自動的に作成されます。

- インターネットユーザーは、SSH または HTTP/HTTPS を使用して、インターネットゲートウェイ経由で EC2 インスタンスに接続できます。
- インターネットユーザーは RDS データベースに直接接続することはできません。EC2 インスタンスのみが RDS データベースに接続します。
- 自動接続機能を使用して EC2 インスタンスと RDS データベース間のトラフィックを許可すると、次のセキュリティグループが自動的に作成および追加されます。
  - セキュリティグループ ec2-rds-*x* が作成され、EC2 インスタンスに追加されます。ここには、rds-ec2-*x* セキュリティグループを送信先として参照するアウトバウンドルールが 1 つ存在します。このルールにより、EC2 インスタンスからのトラフィックが rds-ec2-*x* セキュリティグループのある RDS データベースに到達できるようになります。
  - セキュリティグループ rds-ec2-*x* が作成され、RDS データベースに追加されます。ec2-rds-*x* セキュリティグループを送信元として参照するインバウンドルールが 1 つ存在します。このルールにより、ec2-rds-*x* セキュリティグループを持つ EC2 インスタンスからのトラフィックが RDS データベースに到達できるようになります。

  個別のセキュリティグループ (EC2 インスタンス用と RDS データベース用にそれぞれ 1 つずつ) を使用することで、インスタンスとデータベースのセキュリティをより適切に管理できます。インスタンスとデータベースの両方で同じセキュリティグループを使用し、例えばデータベースのみと合うようにセキュリティグループを変更した場合、その変更はインスタンスとデータベースの両方に影響します。言い換えると、1 つのセキュリティグループを使用した場合、セキュリティグループがアタッチされていることを忘れてしまい、リソース (インスタンスまたはデータベース) のセキュリティを意図せず変更してしまう可能性があることになります。

  また、自動的に作成されるセキュリティグループは最小特権に従っており、ワークロード固有のセキュリティグループペアを作成することで、データベースポート上のこのワークロードに対する相互接続のみを許可します。

考慮事項

このチュートリアルのステップを実行する際には、次の点を考慮してください。

- 2 つのコンソール — このチュートリアルでは、次の 2 つのコンソールを使用します。

  - Amazon EC2 コンソール — EC2 コンソールを使用してインスタンスを起動したり、EC2 インスタンスを RDS データベースに自動接続したり、手動オプションのときにはセキュリティグループを作成して接続を設定します。
  - Amazon RDS コンソール — RDS コンソールを使用して RDS データベースを作成したり、EC2 インスタンスを RDS データベースに自動接続します。
- 1 つの VPC — 自動接続機能を使用するには、EC2 インスタンスと RDS データベースが同じ VPC 内にある必要があります。

  EC2 インスタンスと RDS データベース間の接続を手動で設定する場合、VPC で EC2 インスタンスを起動して、別の VPC で RDS データベースを起動することができますが、追加のルーティングと VPC 設定を設定する必要があります。このシナリオについては、このチュートリアルでは説明しません。
- 1 つの AWS リージョン — EC2 インスタンスと RDS データベースは同じリージョンにある必要があります。
- 2 つのセキュリティグループ — EC2 インスタンスと RDS データベース間の接続は、EC2 インスタンスのセキュリティグループと RDS データベースのセキュリティグループという 2 つのセキュリティグループが設定します。

  EC2 コンソールまたは RDS コンソールの自動接続機能を使用して接続を設定する場合 (このチュートリアルのオプション 1 とオプション 2)、セキュリティグループは自動的に作成され、EC2 インスタンスと RDS データベースに割り当てられます。

  自動接続機能を使用しない場合、セキュリティグループを手動で作成して割り当てる必要があります。これは、このチュートリアルのオプション 3 で行います。

チュートリアル完了までの時間

30 分

チュートリアルの全過程を 1 回で完了することも、1 つのタスクずつに分けて完了することもできます。

コスト

このチュートリアルを完了すると、作成する AWS リソースのコストが発生する可能性があります。

AWS アカウントの使用開始から 12 か月未満で、無料利用枠の要件に従ってリソースを設定している場合は、Amazon EC2 の無料利用枠を利用できます。

EC2 インスタンスと RDS データベースが異なるアベイラビリティーゾーンにある場合は、データ転送料金が発生します。これらの料金の発生を回避するには、EC2 インスタンスと RDS データベースが同じアベイラビリティーゾーンにある必要があります。データ転送料金の詳細については、「Amazon EC2 オンデマンド料金」ページの「データ転送」を参照してください。

チュートリアル完了後にコストが発生するのを避けるため、不要になったリソースは必ず削除してください。リソースを削除するステップについては、「クリーンアップ」を参照してください。

オプション 1: EC2 コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する

目的

オプション 1 では、EC2 コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。オプション 3 では、この接続を手動で設定する方法について説明します。

開始する前に

このチュートリアルを完了するには、以下が必要です。

- EC2 インスタンスと同じ VPC にある RDS データベース。既存の RDS データベースを使用するか、タスク 1 のステップに従って新しい RDS データベースを作成することができます。
- RDS データベースと同じ VPC にある EC2 インスタンス。既存の EC2 インスタンスを使用するか、タスク 2 のステップに従って新しい EC2 インスタンスを作成することができます。
- 次の操作を呼び出すアクセス許可:
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSecurityGroup`
  - `ec2:CreateSubnet`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

オプション 1 を完了するためのタスク

タスク 1: RDS データベースを作成する — オプション

> **Note**
>
> Amazon RDS データベースの作成については、このチュートリアルの対象外です。RDS データベースがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

タスクの目標

このタスクでは、RDS データベースを作成して、EC2 インスタンスと使用する RDS データベース間の接続を設定するタスク 3 を完了できるようにします。使用できる RDS データベースがある場合は、このタスクをスキップできます。

> **Important**
>
> 既存の RDS データベースを使用する場合は、自動接続機能を使用できるように、その RDS データベースが EC2 インスタンスと同じ VPC 内にあることを確認してください。

RDS データベースを作成するためのステップ

次のステップに従って RDS データベースを作成します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: RDS データベースの作成」を参照してください。

RDS データベースの設定

このタスクのステップでは、RDS データベースを次のように設定します。

- エンジンタイプ: MySQL

- テンプレート: 無料利用枠
- DB インスタンス識別子: **tutorial-database-1**
- DB インスタンスクラス: db.t3.micro

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、データベースを設定する必要があります。

MySQL RDS データベースを作成するには

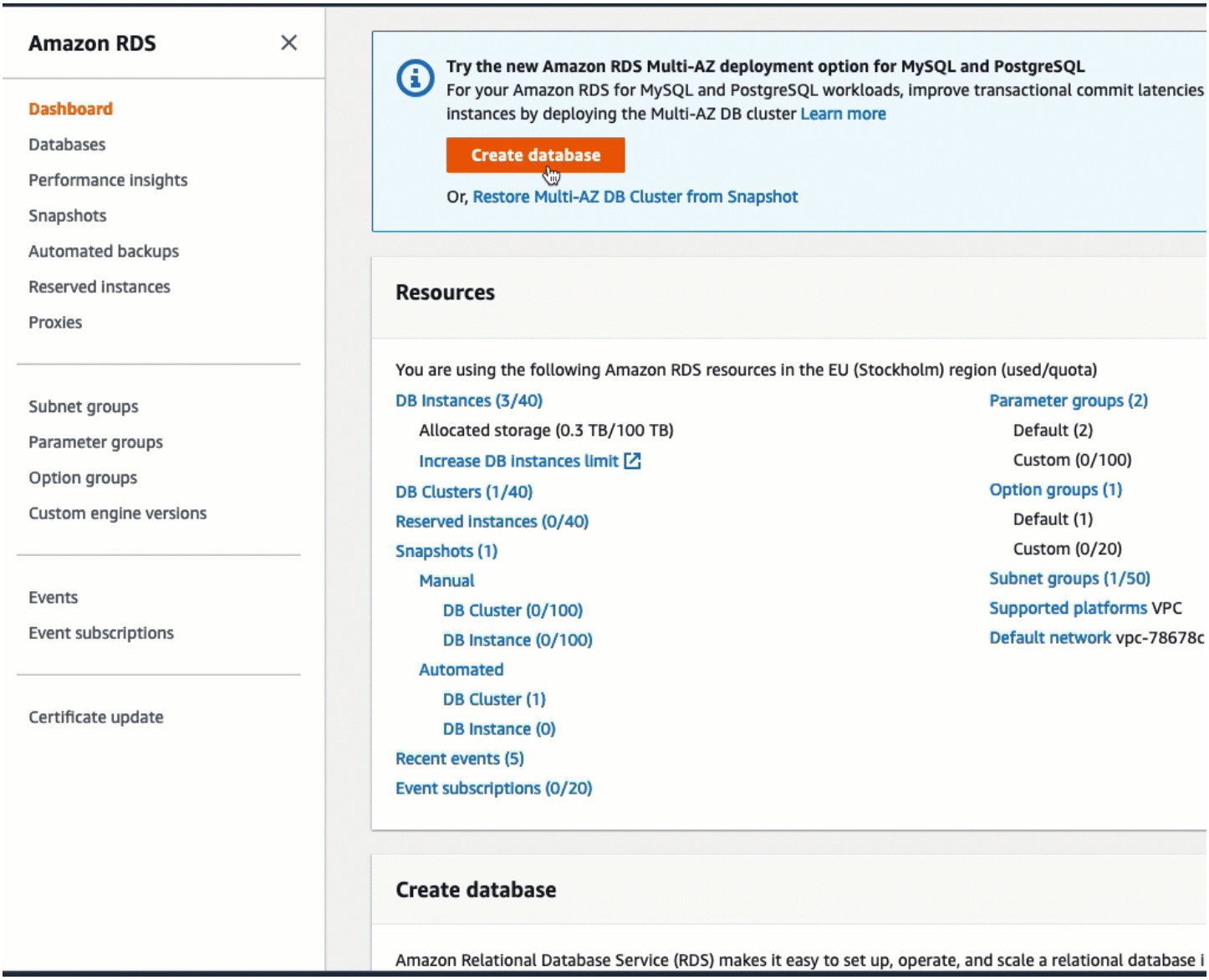
1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. リージョンセレクター (右上) から AWS リージョン を選択します。EC2 コンソールの自動接続機能を使用するには、データベースと EC2 インスタンスが同じリージョンにある必要があります。
3. ダッシュボード で、[Create database] (データベースの作成) を選択します。
4. [Choose a database creation method] (データベース作成方法を選択) で [Standard Create] (スタンダード作成) が選択されていることを確認します。[Easy create] (簡易作成) を選択した場合、VPC セレクターは利用できません。EC2 コンソールの自動接続機能を使用するには、データベースが EC2 インスタンスと同じ VPC 内にあることを必ず確認してください。
5. [Engine options] (エンジンオプション) にある [Engine type] (エンジンタイプ) で MySQL を選択します。
6. [Templates] (テンプレート) では、ニーズに合うサンプルテンプレートを選択します。このチュートリアルでは、[Free tier] (無料利用枠) を選択してデータベースを無料で作成します。ただし、無料利用枠は、アカウント作成から 12 か月未満の場合にのみご利用いただけるので注意してください。その他の制限が適用されます。詳細については、[Free tier] (無料利用枠) ボックスにある [Info] (情報) リンクを選択してください。
7. [設定] で、次のいずれかを実行します。
   a. [DB instance identifier] (DB インスタンス識別子) に、データベースの名前を入力します。このチュートリアルでは、**tutorial-database-1** と入力します。
   b. [Master username] (マスターユーザー名) は、デフォルトの名前である **admin** のままにします。
   c. [Master password] (マスターパスワード) に、このチュートリアルに使用するパスワードを入力し、[Confirm password] (パスワードの確認) にパスワードをもう一度入力します。

8. [Instance configuration] (インスタンス設定) の [DB instance class] (DB インスタンスクラス) は、デフォルトの db.t3.micro のままにします。アカウントが 12 か月未満の場合は、このデータベースクラスを無料で使用できます。その他の制限が適用されます。詳細については、AWS 無料利用枠を参照してください。
9. [Connectivity] (接続) では、[Compute resource] (コンピュートリソース) に [Don't connect to an EC2 compute resource] (EC2 コンピュートリソースに接続しない) を選択します。EC2 インスタンスと RDS データベースは、タスク 3 の後半で接続するからです。

   (後ほど、このチュートリアルのオプション 2 で、[Connect to an EC2 compute resource] (EC2 コンピュートリソースに接続) を選択して RDS コンソールの自動接続機能を試します。)
10. [Virtual private cloud (VPC)] (仮想プライベートクラウド (VPC)) には VPC を選択します。VPC には DB サブネットグループが必要です。自動接続機能を使用するには、EC2 インスタンスと RDS データベースが同じ VPC 内にある必要があります。
11. このページの他のフィールドについては、すべてのデフォルト値をそのまま使用します。
12. [データベースの作成] を選択します。

    [Databases] (データベース) 画面では、データベースが使用可能になるまで、新しいデータベースの [Status] (ステータス) は [Creating] (作成中) になります。ステータスが [Available] (使用可能) に変わったら、データベースに接続できます。データベースクラスとストレージ量によっては、新しいデータベースが使用可能になるまで最長 20 分かかることがあります。

アニメーションを表示: RDS データベースの作成



これで、「タスク 2: EC2 インスタンスを起動する – オプション」を行う準備ができました。

タスク 2: EC2 インスタンスを起動する – オプション

> **Note**
>
> インスタンスの起動は、このチュートリアルの対象外です。Amazon EC2 インスタンスがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

タスクの目標

このタスクでは、EC2 インスタンスを起動して、EC2 インスタンスと使用する Amazon RDS データベース間の接続を設定するタスク 3 を完了できるようにします。使用できる EC2 インスタンスがある場合は、このタスクをスキップできます。

> **Important**
>
> 既存の EC2 インスタンスを使用する場合は、自動接続機能を使用できるように、その EC2 インスタンスが RDS データベースと同じ VPC 内にあることを確認してください。

EC2 インスタンスを起動するためのステップ

以下のステップに従って、このチュートリアル用に EC2 インスタンスを起動します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: EC2 インスタンスを起動する」を参照してください。

EC2 インスタンスの設定

このタスクのステップでは、EC2 インスタンスを次のように設定します。

- インスタンス名: **`tutorial-instance-1`**
- AMI: Amazon Linux 2
- インスタンスタイプ: `t2.micro`
- パブリック IP の自動割り当て: 有効
- 次の 3 つのルールを持つセキュリティグループ:
  - IP アドレスからの SSH を許可
  - 任意の場所からの HTTPS トラフィックを許可
  - 任意の場所からの HTTP トラフィックを許可

> **Important**
>
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

EC2 インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. リージョンセレクター (右上) から AWS リージョン を選択します。EC2 コンソールの自動接続機能を使用するには、インスタンスと RDS データベースが同じリージョンにある必要があります。
3. [EC2 Dashboard] (EC2 ダッシュボード) で、[Launch instance] (インスタンスの作成) を選択します。
4. [Names and tags] (名前とタグ) にある [Name] (名前) には、インスタンスを識別するための名前を入力します。このチュートリアルでは、インスタンス名は「**tutorial-instance-1**」にします。インスタンス名は必須ではありませんが、EC2 コンソールでインスタンスを選択するときに、名前があると識別しやすくなります。
5. [Application and OS Images] (アプリケーションと OS イメージ) で、ウェブサーバーに必要な AMI を選択します。このチュートリアルでは Amazon Linux 2 を使用します。
6. [Instance type] (インスタンスタイプ) にある [Instance type] (インスタンスタイプ) には使用しているウェブサーバーに必要なインスタンスタイプを選択します。このチュートリアルでは、`t2.micro` を使用します。

   > **Note**
   >
   > AWS アカウントの作成から 12 か月未満で、`t2.micro` インスタンスタイプ (または `t2.micro` を利用できないリージョンでは `t3.micro`) を選択している場合は、Amazon EC2 の無料利用枠を利用できます。

7. [Key pair (login)] (キーペア (ログイン)) にある [Key pair name] (キーペア名) には、使用するキーペアを選択します。
8. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. デフォルトの VPC またはサブネットに変更を加えていない場合は、[Network] (ネットワーク) と [Subnet] (サブネット) でデフォルトの設定をそのまま使用できます。

      デフォルトの VPC またはサブネットに変更を加えた場合は、以下を確認してください。

      i. 自動接続機能を使用するには、インスタンスと RDS データベースが同じ VPC 内にある必要があります。デフォルトでは、VPC は 1 つのみです。

ii. インスタンスを起動する VPC には、インターネットからウェブサーバーにアクセスできるように、インターネットゲートウェイがアタッチされている必要があります。デフォルト VPC はインターネットゲートウェイで自動的に設定されます。

iii. インスタンスがパブリック IP アドレスを受け取れるように、[Auto-assign Public IP] (自動割り当てパブリック IP) で [Enable] (有効) が選択されていることを確認します。[Disable] (無効) が選択されている場合は、[Edit] (編集) ([Network Settings] (ネットワーク設定) の右側) を選択し、その後 [Auto-assign public IP]] (パブリック IP の自動割り当て) で [Enable] (有効) を選択します。

b. SSH を使用してインスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH (Linux) または RDP (Windows) トラフィックを承認するセキュリティグループのルールが必要です。デフォルトでは、インスタンスを起動すると、任意の場所からのインバウンド SSH トラフィックを許可するルールで新しいセキュリティグループが作成されます。

使用する IP アドレスのみがインスタンスに接続できるようにするには、[Firewall (security groups)] (ファイアウォール (セキュリティグループ)) にある [Allow SSH traffic from] (SSH トラフィックを許可する送信元) チェックボックスの横にあるドロップダウンリストから [My IP] (マイ IP) を選択します。

c. インターネットからインスタンスへのトラフィックを許可するには、次のチェックボックスを選択します。

- [Allow HTTPs traffic from the internet] (インターネットからの HTTPs トラフィックを許可する)
- [Allow HTTP traffic from the internet] (インターネットからの HTTP トラフィックを許可する)

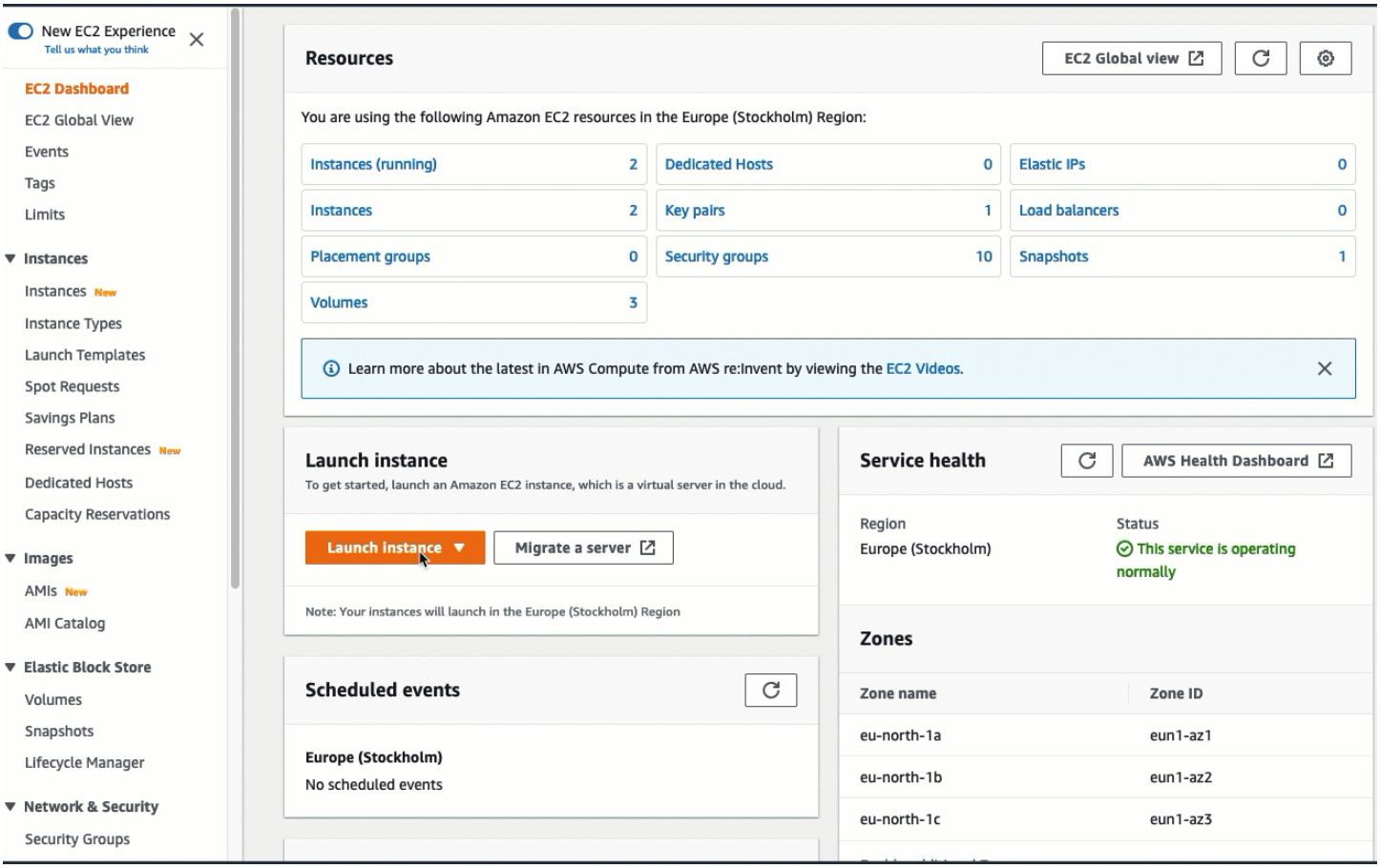
9. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動する) を選択します。

10. 確認ページは開いたままにします。次のタスクで、インスタンスをデータベースに自動接続するときに必要になります。

インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

インスタンスの起動方法の詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

アニメーションを表示: EC2 インスタンスを起動する



これで、「タスク 3: EC2 インスタンスを RDS データベースに自動接続する」を行う準備ができました。

タスク 3: EC2 インスタンスを RDS データベースに自動接続する

タスクの目標

このタスクでは、EC2 コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間の接続を自動的に設定します。

EC2 インスタンスと RDS データベースを接続するステップ

次のステップに従って、EC2 コンソールの自動機能を使用して EC2 インスタンスと RDS データベースを接続します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: 新しく起動した EC2 インスタンスを RDS データベースに自動接続する」を参照してください。

EC2 コンソールを使用して、EC2 インスタンスを RDS データベースに自動接続する場合

1. インスタンスの起動確認ページ (前述のタスク実行時に開いたままにしておく) で、[Connect an RDS database] (RDS データベースに接続) を選択します。

   確認ページを閉じてしまった場合は、次の手順に従ってください。

   a. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
   b. ナビゲーションペインで、[インスタンス] を選択します。
   c. 先ほど作成した EC2 インスタンスを選択し、[Actions] (アクション)、[Networking] (ネットワーク)、[Connect RDS database] (RDS データベースを接続) の順に選択します。

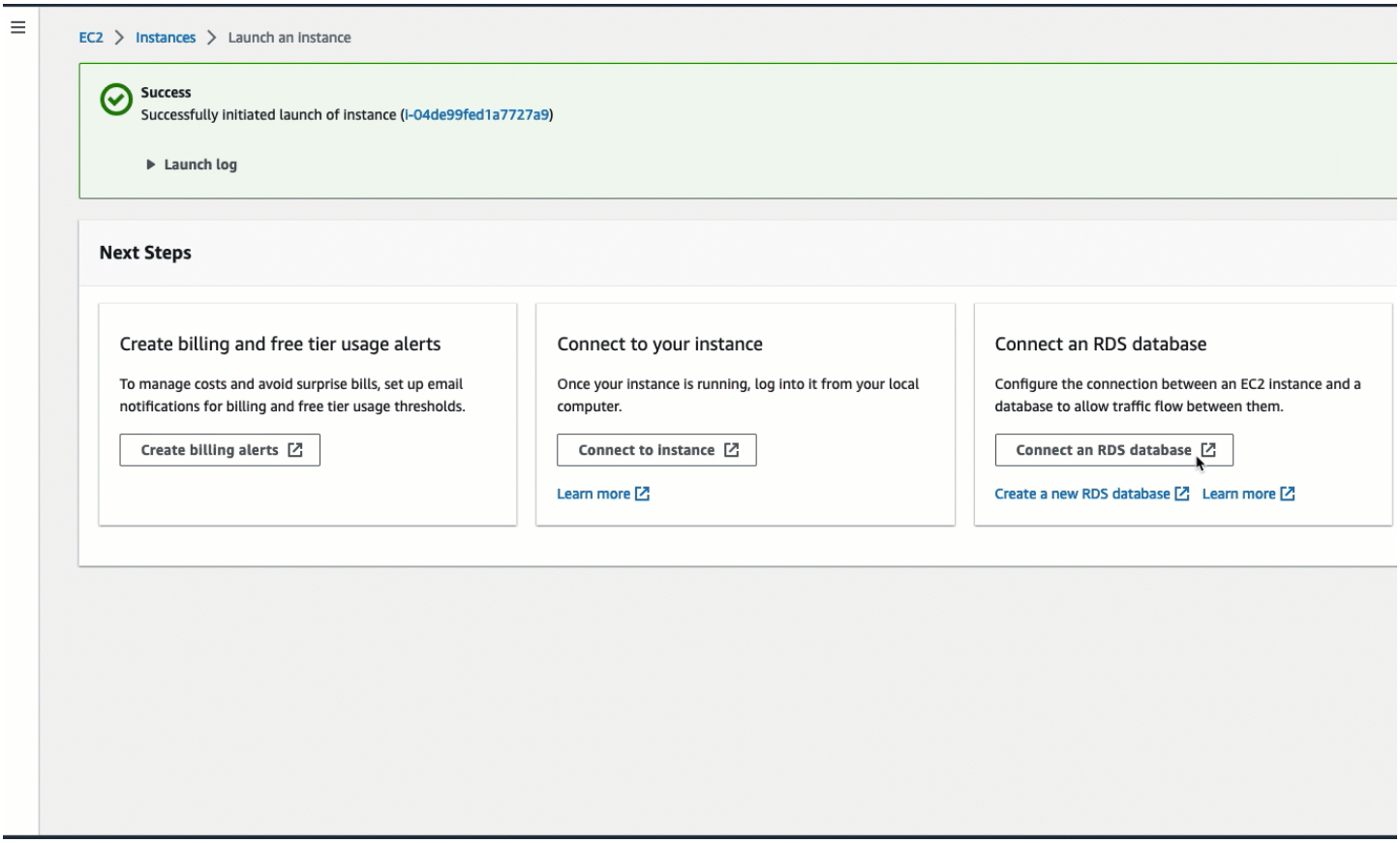
      [Connect RDS database] (RDS データベースに接続) を選択できない場合は、EC2 インスタンスが [Running] (実行中) の状態であることを確認します。

2. [Database role] (データベースロール) で [Instance] (インスタンス) を選択します。この場合のインスタンスは、データベースインスタンスを指しています。
3. [RDS database] (RDS データベース) の場合は、タスク 1 で作成した RDS データベースを選択します。

   > **Note**
   >
   > EC2 インスタンスと RDS データベースが相互接続するためは、同じ VPC 内にある必要があります。

4. [接続]を選択します。

アニメーションを表示: 新しく起動した EC2 インスタンスを RDS データベースに自動接続する



これで、「タスク 4: 接続設定を検証する」を行う準備ができました。

タスク 4: 接続設定を検証する

タスクの目標

このタスクでは、2 つのセキュリティグループが作成され、インスタンスとデータベースに割り当てられていることを確認します。

EC2 コンソールの自動接続機能を使用して接続を設定する場合、セキュリティグループは自動的に作成され、次のように、EC2 インスタンスと RDS データベースに割り当てられます。

- セキュリティグループ rds-ec2-*x* が作成され、RDS データベースに追加されます。ec2-rds-*x* セキュリティグループを送信元として参照するインバウンドルールが 1 つ存在します。このルールにより、ec2-rds-*x* セキュリティグループを持つ EC2 インスタンスからのトラフィックが RDS データベースに到達できるようになります。
- セキュリティグループ ec2-rds-*x* が作成され、EC2 インスタンスに追加されます。ここには、rds-ec2-*x* セキュリティグループを送信先として参照するアウトバウンドルールが 1 つ存在します。こ

のルールにより、EC2 インスタンスからのトラフィックが rds-ec2-*x* セキュリティグループのある RDS データベースに到達できるようになります。

接続構成を確認するためのステップ

接続構成を確認するには、次のステップを実行します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: 接続設定を確認する」を参照してください。

コンソールを使用して接続構成を確認する場合

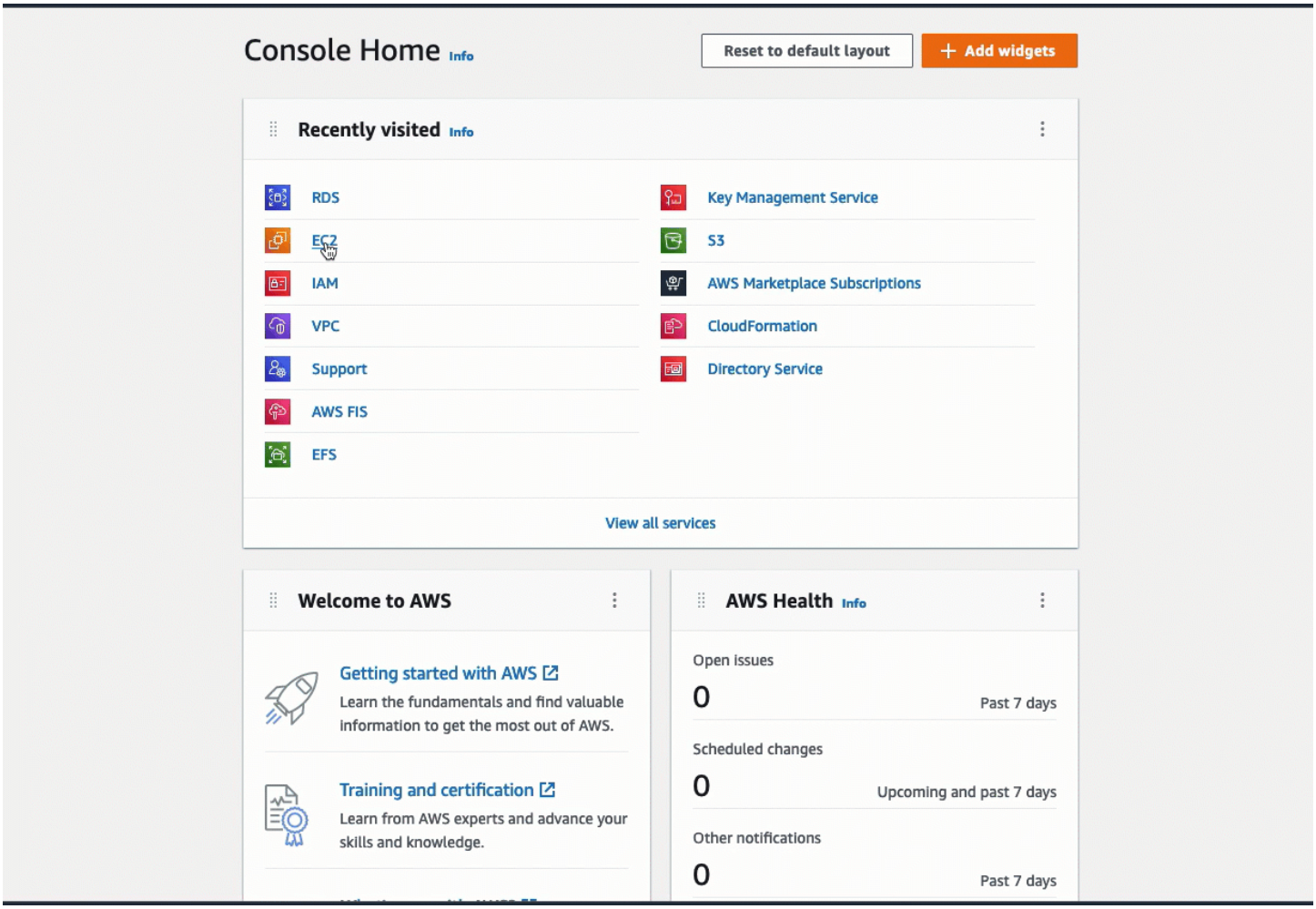
1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションページで、[Databases] (データベース) を選択します。
3. このチュートリアル用に作成した RDS データベースを選択します。
4. [Connectivity & security] (接続とセキュリティ) タブの [Security] (セキュリティ) と [VPC security groups] (VPC セキュリティグループ) に、rds-ec2-*x* という名前のセキュリティグループが表示されていることを確認します。
5. rds-ec2-*x* セキュリティグループを選択します。EC2 コンソールにある [Security Groups] (セキュリティグループ) 画面が開きます。
6. rds-ec2-*x* セキュリティグループを選択して開きます。
7. [Inbound rules] (インバウンドルール) タブを開きます。
8. 次のセキュリティグループルールが存在することを確認します。
   - タイプ: MYSQL/Aurora
   - ポート範囲: 3306
   - ソース: ***sg-0987654321example*** / ec2-rds-*x* – これは、前述の手順で検証した EC2 インスタンスに割り当てられているセキュリティグループです。
   - 説明: ***sg-1234567890example*** が添付された EC2 インスタンスからの接続を許可するルール
9. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
10. ナビゲーションペインで、[インスタンス] を選択します。
11. 前述のタスクで RDS データベースに接続するために選択した EC2 インスタンスを選択し、[Security] (セキュリティ) タブを選択します。

12. [Security details] (セキュリティの詳細) の [Security groups] (セキュリティグループ) にあるリストの中に、ec2-rds-*x* という名前のセキュリティグループが含まれていることを確認します。*x* は数字です。
13. ec2-rds-*x* セキュリティグループを選択して開きます。
14. [Outbound rules] (アウトバウンドルール) タブを選択します。
15. 次のセキュリティグループルールが存在することを確認します。

    - タイプ: MYSQL/Aurora
    - ポート範囲: 3306
    - 送信先: ***sg-1234567890example*** / rds-ec2-*x*
    - 説明: このセキュリティグループがアタッチされている任意のインスタンスからの **database-tutorial** への接続を許可するルール

これらのセキュリティグループとセキュリティグループルールが存在し、この手順の記述にあるように RDS データベースと EC2 インスタンスに割り当てられていることを確認することで、自動接続機能を使用して接続が自動的に設定されたことを確認できます。

アニメーションを表示: 接続設定を確認する



これで、このチュートリアルのオプション 1 が完了しました。これで、RDS コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する方法について説明するオプション 2 を完了するか、オプション 1 で自動的に作成されたセキュリティグループを手動で設定する方法について説明するオプション 3 を完了することができます。

オプション 2: RDS コンソールを使用して EC2 インスタンスを RDS データベースに自動接続する

目的

オプション 2 では、RDS コンソールの自動接続機能を使用して、EC2 インスタンスと RDS データベース間のトラフィックを許可するように EC2 インスタンスと RDS データベース間の接続を自動的に設定します。オプション 3 では、この接続を手動で設定する方法について説明します。

開始する前に

このチュートリアルを完了するには、以下が必要です。

- RDS データベースと同じ VPC にある EC2 インスタンス。既存の EC2 インスタンスを使用することも、タスク 1 のステップに従って新しいインスタンスを作成することもできます。
- 次の操作を呼び出すアクセス許可:
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSecurityGroup`
  - `ec2:CreateSubnet`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

オプション 2 を完了するためのタスク

- タスク 1: EC2 インスタンスを起動する – オプション
- タスク 2: RDS データベースを作成し、それを EC2 インスタンスに自動接続する
- タスク 3: 接続設定を検証する

タスク 1: EC2 インスタンスを起動する – オプション

> **Note**
>
> インスタンスの起動は、このチュートリアルの対象外です。Amazon EC2 インスタンスがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

タスクの目標

このタスクでは、EC2 インスタンスを起動して、EC2 インスタンスと使用する Amazon RDS データベース間の接続を設定するタスク 2 を完了できるようにします。使用できる EC2 インスタンスがある場合は、このタスクをスキップできます。

EC2 インスタンスを起動するためのステップ

以下のステップに従って、このチュートリアル用に EC2 インスタンスを起動します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: EC2 インスタンスを起動する」を参照してください。

EC2 インスタンスの設定

このタスクのステップでは、EC2 インスタンスを次のように設定します。

- インスタンス名: **`tutorial-instance-2`**
- AMI: Amazon Linux 2
- インスタンスタイプ: `t2.micro`
- パブリック IP の自動割り当て: 有効
- 次の 3 つのルールを持つセキュリティグループ:
  - IP アドレスからの SSH を許可
  - 任意の場所からの HTTPS トラフィックを許可
  - 任意の場所からの HTTP トラフィックを許可

> **Important**
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

EC2 インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [EC2 Dashboard] (EC2 ダッシュボード) で、[Launch instance] (インスタンスの作成) を選択します。
3. [Names and tags] (名前とタグ) にある [Name] (名前) には、インスタンスを識別するための名前を入力します。このチュートリアルでは、インスタンス名は「**`tutorial-instance-2`**」にし

ます。インスタンス名は必須ではありませんが、RDS コンソールでインスタンスを選択するときに、名前があると識別しやすくなります。

4. [Application and OS Images] (アプリケーションと OS イメージ) で、ウェブサーバーに必要な AMI を選択します。このチュートリアルでは Amazon Linux を使用します。
5. [Instance type] (インスタンスタイプ) にある [Instance type] (インスタンスタイプ) には使用しているウェブサーバーに必要なインスタンスタイプを選択します。このチュートリアルでは、`t2.micro` を使用します。

   > **Note**
   >
   > AWS アカウントの作成から 12 か月未満で、`t2.micro` インスタンスタイプ (または `t2.micro` を利用できないリージョンでは `t3.micro`) を選択している場合は、Amazon EC2 の無料利用枠を利用できます。

6. [Key pair (login)] (キーペア (ログイン)) にある [Key pair name] (キーペア名) には、使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. デフォルトの VPC またはサブネットに変更を加えていない場合は、[Network] (ネットワーク) と [Subnet] (サブネット) でデフォルトの設定をそのまま使用できます。

      デフォルトの VPC またはサブネットに変更を加えた場合は、以下を確認してください。

      i. 自動接続設定を使用するには、インスタンスが RDS データベースと同じ VPC 内に存在している必要があります。デフォルトでは、VPC は 1 つのみです。

      ii. インスタンスを起動する VPC には、インターネットからウェブサーバーにアクセスできるように、インターネットゲートウェイがアタッチされている必要があります。デフォルト VPC はインターネットゲートウェイで自動的に設定されます。

      iii. インスタンスがパブリック IP アドレスを受け取れるように、[Auto-assign Public IP] (自動割り当てパブリック IP) で [Enable] (有効) が選択されていることを確認します。[Disable] (無効) が選択されている場合は、[Edit] (編集) ([Network Settings] (ネットワーク設定) の右側) を選択し、その後 [Auto-assign public IP]] (パブリック IP の自動割り当て) で [Enable] (有効) を選択します。

   b. SSH を使用してインスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH (Linux) または RDP (Windows) トラフィックを承認するセキュリティグループのルールが必要です。デフォルトでは、インスタンスを起動すると、任意の場所からのイン

バウンド SSH トラフィックを許可するルールで新しいセキュリティグループが作成されます。

使用する IP アドレスのみがインスタンスに接続できるようにするには、[Firewall (security groups)] (ファイアウォール (セキュリティグループ)) にある [Allow SSH traffic from] (SSH トラフィックを許可する送信元) チェックボックスの横にあるドロップダウンリストから [My IP] (マイ IP) を選択します。

c. インターネットからインスタンスへのトラフィックを許可するには、次のチェックボックスを選択します。

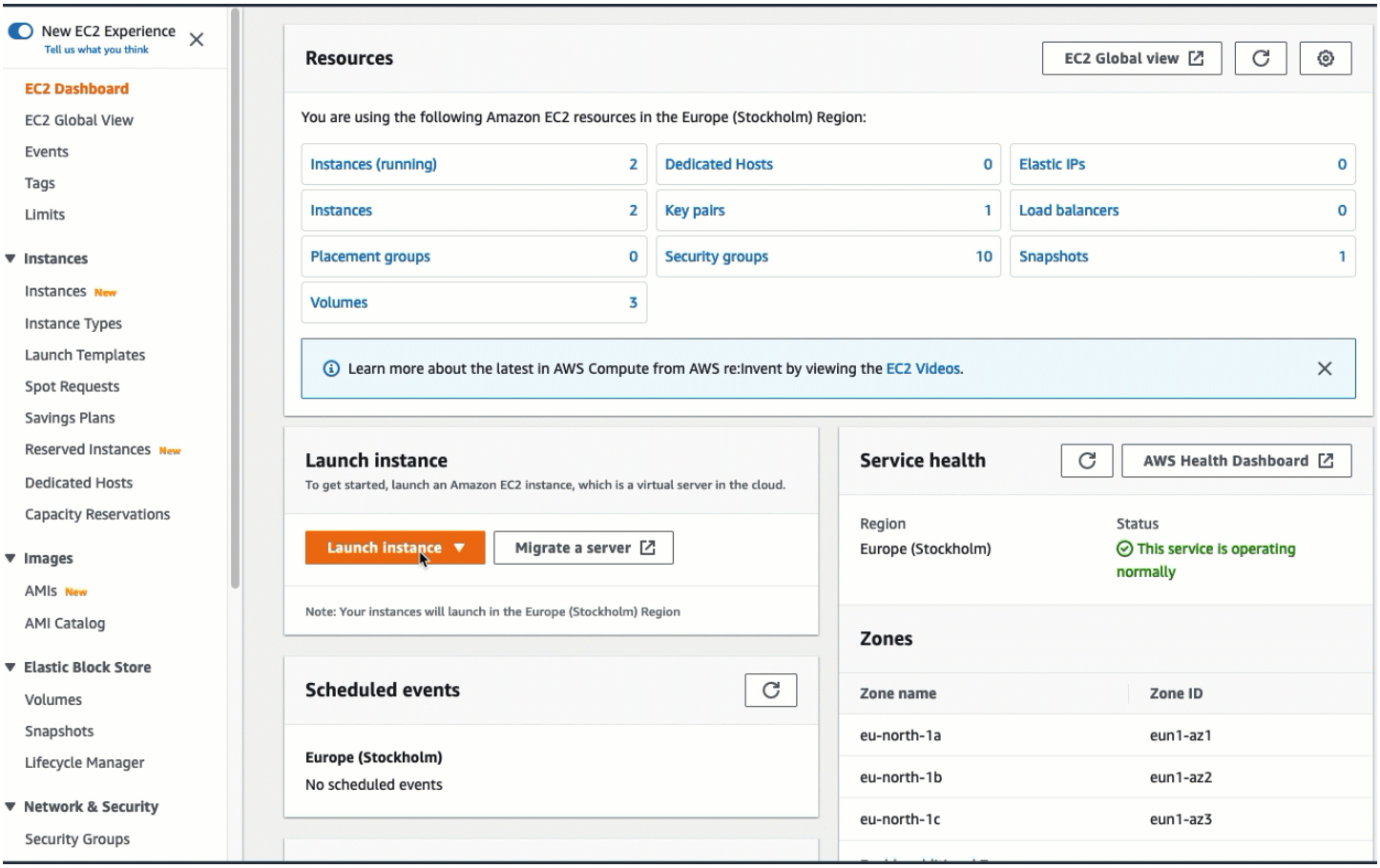
- [Allow HTTPs traffic from the internet] (インターネットからの HTTPs トラフィックを許可する)
- [Allow HTTP traffic from the internet] (インターネットからの HTTP トラフィックを許可する)

8. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動する) を選択します。
9. [View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。インスタンスは最初 `pending` 状態になり、その後 `running` 状態になります。

インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

インスタンスの起動方法の詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

アニメーションを表示: EC2 インスタンスを起動する



これで、「タスク 2: RDS データベースを作成し、それを EC2 インスタンスに自動接続する」を行う準備ができました。

タスク 2: RDS データベースを作成し、それを EC2 インスタンスに自動接続する

タスクの目標

このタスクでは、RDS データベースを作成して RDS コンソールの自動接続機能を使用し、EC2 インスタンスと RDS データベース間の接続を自動的に設定します。

RDS データベースを作成するためのステップ

次のステップで RDS データベースを作成し、RDS コンソールの自動機能を使用して EC2 インスタンスに接続します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: RDS データベースを作成して、EC2 インスタンスに自動接続する」を参照してください。

DB インスタンスの設定

このタスクのステップでは、DB インスタンスを次のように設定します。

- エンジンタイプ: MySQL
- テンプレート: 無料利用枠
- DB インスタンス識別子: **tutorial-database**
- DB インスタンスクラス: db.t3.micro

> **Important**
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

RDS データベースを作成して EC2 インスタンスに自動接続する

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. リージョンセレクター (右上) から、EC2 インスタンスを作成した AWS リージョン を選択します。EC2 インスタンスと RDS データベースは同じリージョンにある必要があります。
3. ダッシュボード で、[Create database] (データベースの作成) を選択します。
4. [Choose a database creation method] (データベース作成方法を選択) で [Standard Create] (スタンダード作成) が選択されていることを確認します。[Easy create] (簡易作成) を選択した場合、自動接続機能は使用できません。
5. [Engine options] (エンジンオプション) にある [Engine type] (エンジンタイプ) で MySQL を選択します。
6. [Templates] (テンプレート) では、ニーズに合うサンプルテンプレートを選択します。このチュートリアルでは、[Free tier] (無料利用枠) を選択して RDS データベースを無料で作成できます。ただし、無料利用枠は、アカウント作成から 12 か月未満の場合にのみご利用いただけるので注意してください。その他の制限が適用されます。詳細については、[Free tier] (無料利用枠) ボックスにある [Info] (情報) リンクを選択してください。
7. [設定] で、次のいずれかを実行します。
   a. [DB instance identifier] (DB インスタンス識別子) に、データベースの名前を入力します。このチュートリアルでは、**tutorial-database** と入力します。
   b. [Master username] (マスターユーザー名) は、デフォルトの名前である **admin** のままにします。

c. [Master password] (マスターパスワード) に、このチュートリアルに使用するパスワードを入力し、[Confirm password] (パスワードの確認) にパスワードをもう一度入力します。

8. [Instance configuration] (インスタンス設定) の [DB instance class] (DB インスタンスクラス) は、デフォルトの db.t3.micro のままにします。アカウントが 12 か月未満の場合は、このインスタンスを無料で使用できます。その他の制限が適用されます。詳細については、AWS 無料利用枠を参照してください。

9. [Connectivity] (接続) にある [Compute resource] (コンピューティングリソース) で、[Connect to an EC2 compute resource] (EC2 コンピューティングリソースに接続する) を選択します。これは RDS コンソールの自動接続機能です。

10. [EC2 instance] (EC2 インスタンス) で、接続先のインスタンスを選択します。このチュートリアルでは、前のタスクで作成して **`tutorial-instance`** と名付けたインスタンスを選択することも、別の既存のインスタンスを選択することもできます。リストにインスタンスが表示されない場合は、[Connectivity] (接続) の右にある更新アイコンを選択します。

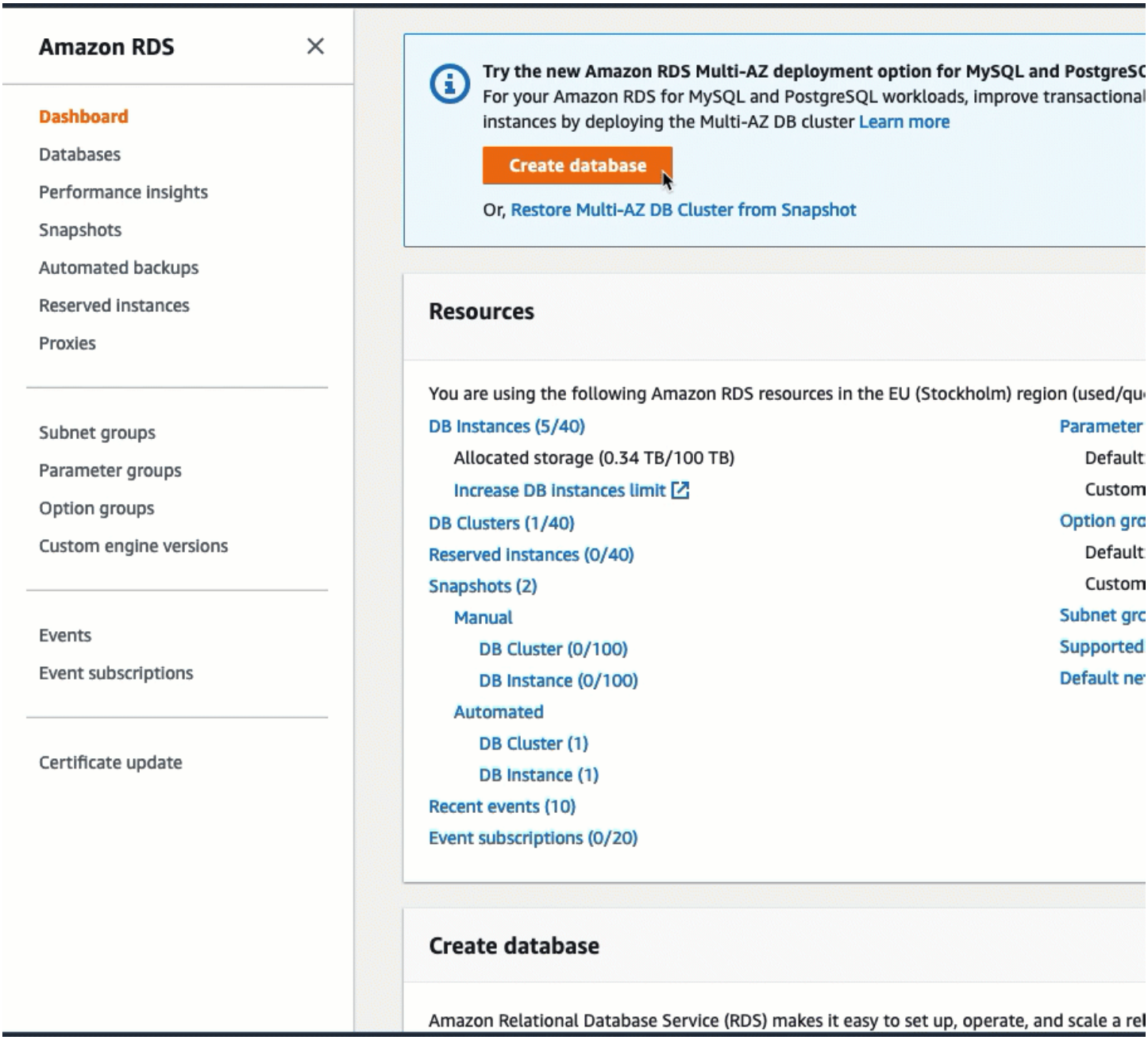
    自動接続機能を使用すると、この EC2 インスタンスにセキュリティグループが追加され、RDS データベースに別のセキュリティグループが追加されます。セキュリティグループは、EC2 インスタンスと RDS データベース間のトラフィックを許可するように自動的に設定されます。次のタスクでは、セキュリティグループが作成され、EC2 インスタンスと RDS データベースに割り当てられていることを確認します。

11. [データベースの作成] を選択します。

    [Databases] (データベース) 画面では、データベースが使用可能になるまで、新しいデータベースの [Status] (ステータス) は [Creating] (作成中) になります。ステータスが [Available] (使用可能) に変わったら、データベースに接続できます。データベースクラスとストレージ量によっては、新しいデータベースが使用可能になるまで最長 20 分かかることがあります。

詳細については、Amazon RDS ユーザーガイドの「EC2 インスタンスとの自動ネットワーク接続を設定する」を参照してください。

アニメーションを表示: RDS データベースを作成して、EC2 インスタンスに自動接続する



これで、「タスク 3: 接続設定を検証する」を行う準備ができました。

タスク 3: 接続設定を検証する

タスクの目標

このタスクでは、2 つのセキュリティグループが作成され、インスタンスとデータベースに割り当てられていることを確認します。

RDS コンソールの自動接続機能を使用して接続を設定する場合、セキュリティグループは自動的に作成され、次のように、インスタンスとデータベースに割り当てられます。

- セキュリティグループ rds-ec2-*x* が作成され、RDS データベースに追加されます。ec2-rds-*x* セキュリティグループを送信元として参照するインバウンドルールが 1 つ存在します。このルールにより、ec2-rds-*x* セキュリティグループを持つ EC2 インスタンスからのトラフィックが RDS データベースに到達できるようになります。
- セキュリティグループ ec2-rds-*x* が作成され、EC2 インスタンスに追加されます。ここには、rds-ec2-*x* セキュリティグループを送信先として参照するアウトバウンドルールが 1 つ存在します。このルールにより、EC2 インスタンスからのトラフィックが rds-ec2-*x* セキュリティグループのある RDS データベースに到達できるようになります。

接続構成を確認するためのステップ

接続構成を確認するには、次のステップを実行します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: 接続設定を確認する」を参照してください。

コンソールを使用して接続構成を確認する場合

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 前述のタスクで RDS データベースに接続するために選択した EC2 インスタンスを選択し、[Security] (セキュリティ) タブを選択します。
4. [Security details] (セキュリティの詳細) の [Security groups] (セキュリティグループ) にあるリストの中に、ec2-rds-*x* という名前のセキュリティグループが含まれていることを確認します。*x* は数字です。
5. ec2-rds-*x* セキュリティグループを選択して開きます。
6. [Outbound rules] (アウトバウンドルール) タブを選択します。
7. 次のセキュリティグループルールが存在することを確認します。
   - タイプ: MYSQL/Aurora
   - ポート範囲: 3306
   - 送信先: *sg-1234567890example* / rds-ec2-*x*

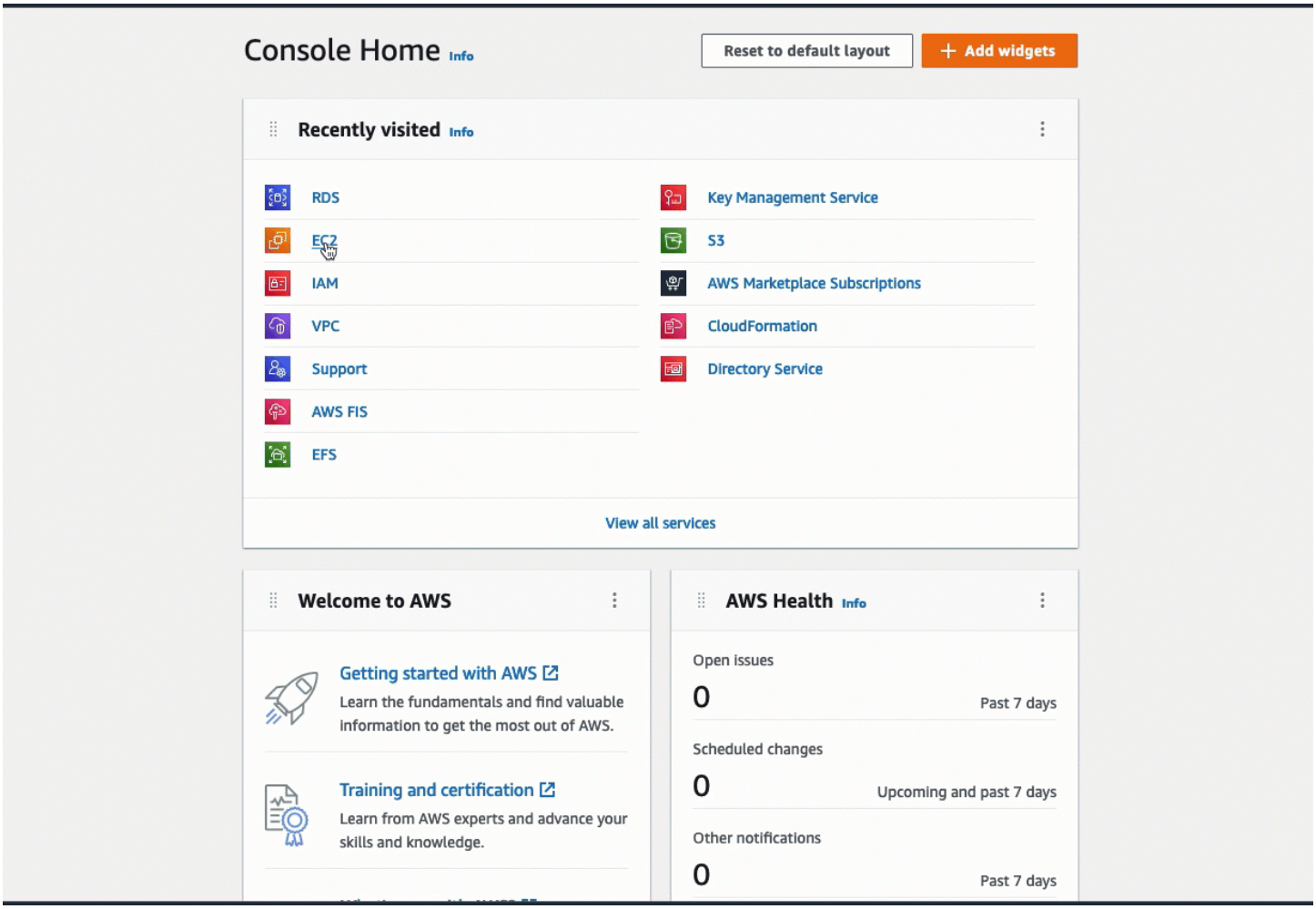
- 説明: このセキュリティグループがアタッチされている任意のインスタンスからの **database-tutorial** への接続を許可するルール

8. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
9. ナビゲーションページで、[Databases] (データベース) を選択します。
10. このチュートリアル用に作成した RDS データベースを選択します。
11. [Connectivity & security] (接続とセキュリティ) タブの [Security] (セキュリティ) と [VPC security groups] (VPC セキュリティグループ) に、rds-ec2-***x*** という名前のセキュリティグループが表示されていることを確認します。
12. rds-ec2-***x*** セキュリティグループを選択します。EC2 コンソールにある [Security Groups] (セキュリティグループ) 画面が開きます。
13. rds-ec2-***x*** セキュリティグループを選択して開きます。
14. [Inbound rules] (インバウンドルール) タブを開きます。
15. 次のセキュリティグループルールが存在することを確認します。

    - タイプ: MYSQL/Aurora
    - ポート範囲: 3306
    - ソース: ***sg-0987654321example*** / ec2-rds-***x*** – これは、前述の手順で検証した EC2 インスタンスに割り当てられているセキュリティグループです。
    - 説明: ***sg-1234567890example*** が添付された EC2 インスタンスからの接続を許可するルール

    これらのセキュリティグループとセキュリティグループルールが存在し、この手順の記述にあるように EC2 インスタンスと RDS データベースに割り当てられていることを確認することで、自動接続機能を使用して接続が自動的に設定されたことを確認できます。

アニメーションを表示: 接続設定を確認する



これで、このチュートリアルのオプション 2 が完了しました。これで、オプション 2 で自動的に作成されたセキュリティグループを手動で設定する方法を説明するオプション 3 を完了することもできます。

オプション 3: 自動接続機能を模倣して EC2 インスタンスを RDS データベースに手動で接続する

目的

オプション 3 では、自動接続機能の設定を手動で再現することで、EC2 インスタンスと RDS データベース間の接続を手動で設定する方法を説明します。

開始する前に

このチュートリアルを完了するには、以下が必要です。

- RDS データベースと同じ VPC にある EC2 インスタンス。既存の EC2 インスタンスを使用することも、タスク 1 のステップに従って新しいインスタンスを作成することもできます。
- EC2 インスタンスと同じ VPC にある RDS データベース。既存の RDS データベースを使用することも、タスク 2 のステップに従って新しいデータベースを作成することもできます。
- 次の操作を呼び出すアクセス許可。このチュートリアルのオプション 1 を完了している場合、すでにこれらのアクセス許可を持っています。
  - `ec2:AssociateRouteTable`
  - `ec2:AuthorizeSecurityGroupEgress`
  - `ec2:CreateRouteTable`
  - `ec2:CreateSecurityGroup`
  - `ec2:CreateSubnet`
  - `ec2:DescribeInstances`
  - `ec2:DescribeNetworkInterfaces`
  - `ec2:DescribeRouteTables`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `ec2:ModifyNetworkInterfaceAttribute`
  - `ec2:RevokeSecurityGroupEgress`

オプション 3 を完了するためのタスク

- タスク 1: EC2 インスタンスを起動する – オプション
- タスク 2: RDS データベースを作成する — オプション
- タスク 3: セキュリティグループを作成してインスタンスに割り当てることで、EC2 インスタンスを RDS データベースに手動で接続する

タスク 1: EC2 インスタンスを起動する – オプション

> **Note**
>
> インスタンスの起動は、このチュートリアルの対象外です。Amazon EC2 インスタンスがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

タスクの目標

このタスクでは、EC2 インスタンスを起動して、EC2 インスタンスと使用する Amazon RDS データベース間の接続を設定するタスク 3 を完了できるようにします。

EC2 インスタンスを起動するためのステップ

以下のステップに従って、このチュートリアル用に EC2 インスタンスを起動します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: EC2 インスタンスを起動する」を参照してください。

EC2 インスタンスの設定

このタスクのステップでは、EC2 インスタンスを次のように設定します。

- インスタンス名: **tutorial-instance**
- AMI: Amazon Linux 2
- インスタンスタイプ: t2.micro
- パブリック IP の自動割り当て: 有効
- 次の 3 つのルールを持つセキュリティグループ:
  - IP アドレスからの SSH を許可
  - 任意の場所からの HTTPS トラフィックを許可
  - 任意の場所からの HTTP トラフィックを許可

> **Important**
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

EC2 インスタンスを起動するには

1. AWS Management Console にサインインし、Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [EC2 Dashboard] (EC2 ダッシュボード) で、[Launch instance] (インスタンスの作成) を選択します。
3. [Names and tags] (名前とタグ) にある [Name] (名前) には、インスタンスを識別するための名前を入力します。このチュートリアルでは、インスタンス名は「**tutorial-instance-**

manual-1」にします。インスタンス名は必須ではありませんが、名前があると識別しやすくなります。

4. [Application and OS Images] (アプリケーションと OS イメージ) で、ウェブサーバーに必要な AMI を選択します。このチュートリアルでは Amazon Linux を使用します。
5. [Instance type] (インスタンスタイプ) にある [Instance type] (インスタンスタイプ) には使用しているウェブサーバーに必要なインスタンスタイプを選択します。このチュートリアルでは、`t2.micro` を使用します。

   > **Note**
   >
   > AWS アカウントの作成から 12 か月未満で、`t2.micro` インスタンスタイプ (または `t2.micro` を利用できないリージョンでは `t3.micro`) を選択している場合は、Amazon EC2 の無料利用枠を利用できます。

6. [Key pair (login)] (キーペア (ログイン)) にある [Key pair name] (キーペア名) には、使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) で、次の操作を行います：

   a. デフォルトの VPC またはサブネットに変更を加えていない場合は、[Network] (ネットワーク) と [Subnet] (サブネット) でデフォルトの設定をそのまま使用できます。

      デフォルトの VPC またはサブネットに変更を加えた場合は、以下を確認してください。

      i. インスタンスは、RDS データベースと同じ VPC 内に存在する必要があります。デフォルトでは、VPC は 1 つのみです。

      ii. インスタンスを起動する VPC には、インターネットからウェブサーバーにアクセスできるように、インターネットゲートウェイがアタッチされている必要があります。デフォルト VPC はインターネットゲートウェイで自動的に設定されます。

      iii. インスタンスがパブリック IP アドレスを受け取れるように、[Auto-assign Public IP] (自動割り当てパブリック IP) で [Enable] (有効) が選択されていることを確認します。[Disable] (無効) が選択されている場合は、[Edit] (編集) ([Network Settings] (ネットワーク設定) の右側) を選択し、その後 [Auto-assign public IP]] (パブリック IP の自動割り当て) で [Enable] (有効) を選択します。

   b. SSH を使用してインスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH (Linux) または RDP (Windows) トラフィックを承認するセキュリティグループのルールが必要です。デフォルトでは、インスタンスを起動すると、任意の場所からのイン

バウンド SSH トラフィックを許可するルールで新しいセキュリティグループが作成されます。

使用する IP アドレスのみがインスタンスに接続できるようにするには、[Firewall (security groups)] (ファイアウォール (セキュリティグループ)) にある [Allow SSH traffic from] (SSH トラフィックを許可する送信元) チェックボックスの横にあるドロップダウンリストから [My IP] (マイ IP) を選択します。

c. インターネットからインスタンスへのトラフィックを許可するには、次のチェックボックスを選択します。

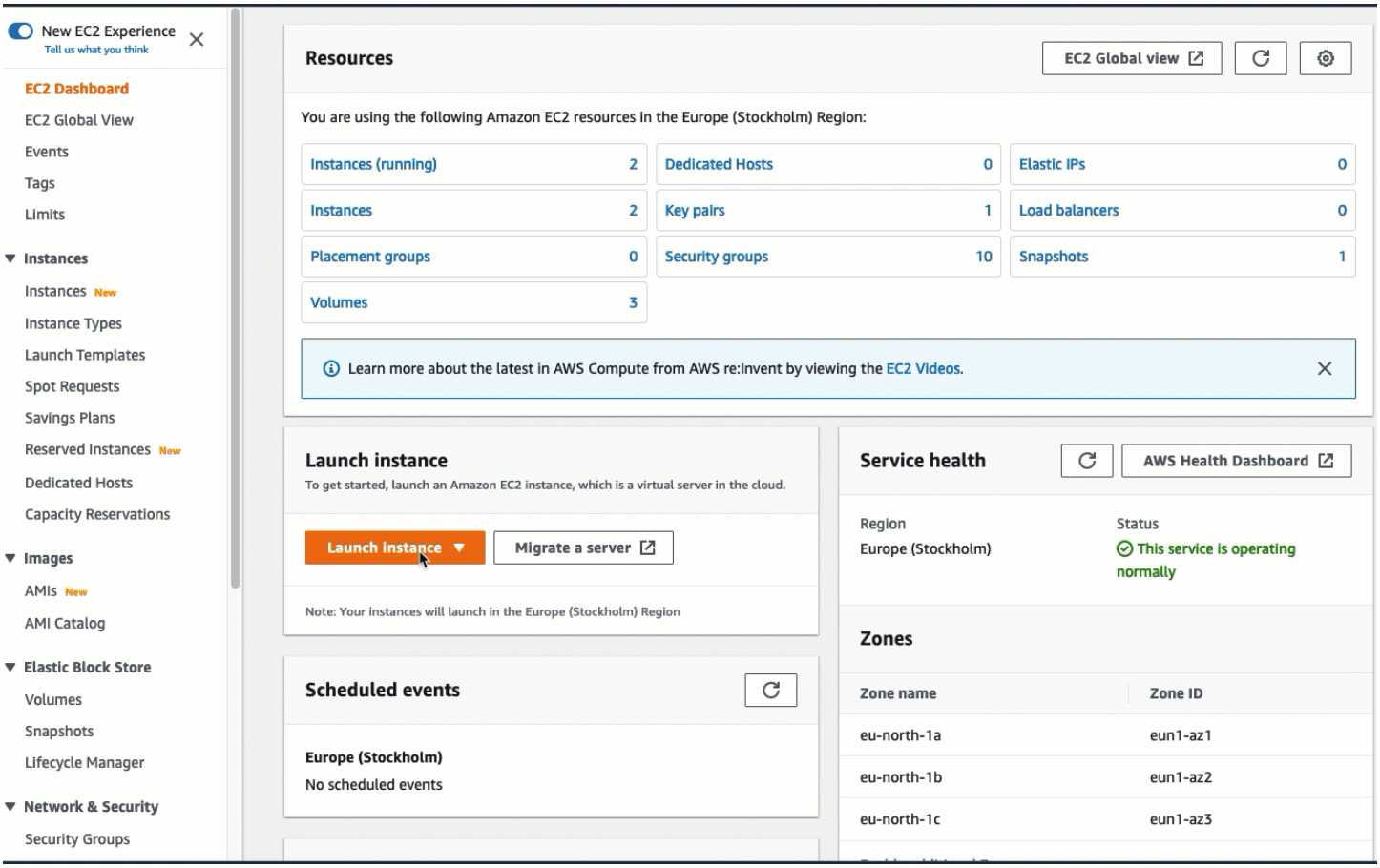
- [Allow HTTPs traffic from the internet] (インターネットからの HTTPs トラフィックを許可する)
- [Allow HTTP traffic from the internet] (インターネットからの HTTP トラフィックを許可する)

8. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動する) を選択します。
9. [View all instances] (すべてのインスタンスの表示) を選択して確認ページを閉じ、コンソールに戻ります。インスタンスは最初 `pending` 状態になり、その後 `running` 状態になります。

インスタンスが起動しないか、状態が `running` ではなくすぐに `terminated` になる場合は、「インスタンスの起動に関する問題のトラブルシューティング」を参照してください。

インスタンスの起動方法の詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

アニメーションを表示: EC2 インスタンスを起動する



これで、「タスク 2: RDS データベースを作成する — オプション」を行う準備ができました。

タスク 2: RDS データベースを作成する — オプション

> **Note**
>
> RDS データベースの作成については、このチュートリアルでカバーしません。RDS データベースがすでにあり、このチュートリアルで使用する場合は、このタスクをスキップできます。

タスクの目標

このタスクでは、RDS データベースを作成します。このインスタンスは、EC2 インスタンスに接続するタスク 3 で使用します。

RDS データベースを作成するためのステップ

次のステップを使用して、このチュートリアルのオプション 3 用に RDS データベースを作成します。

これらの手順のアニメーションを見る場合は、「アニメーションを表示: DB インスタンスの作成」を参照してください。

RDS データベースの設定

このタスクのステップでは、RDS データベースを次のように設定します。

- エンジンタイプ: MySQL
- テンプレート: 無料利用枠
- DB インスタンス識別子: **tutorial-database-manual**
- DB インスタンスクラス: db.t3.micro

> **Important**
> 本番環境では、具体的なニーズに合わせて、インスタンスを設定する必要があります。

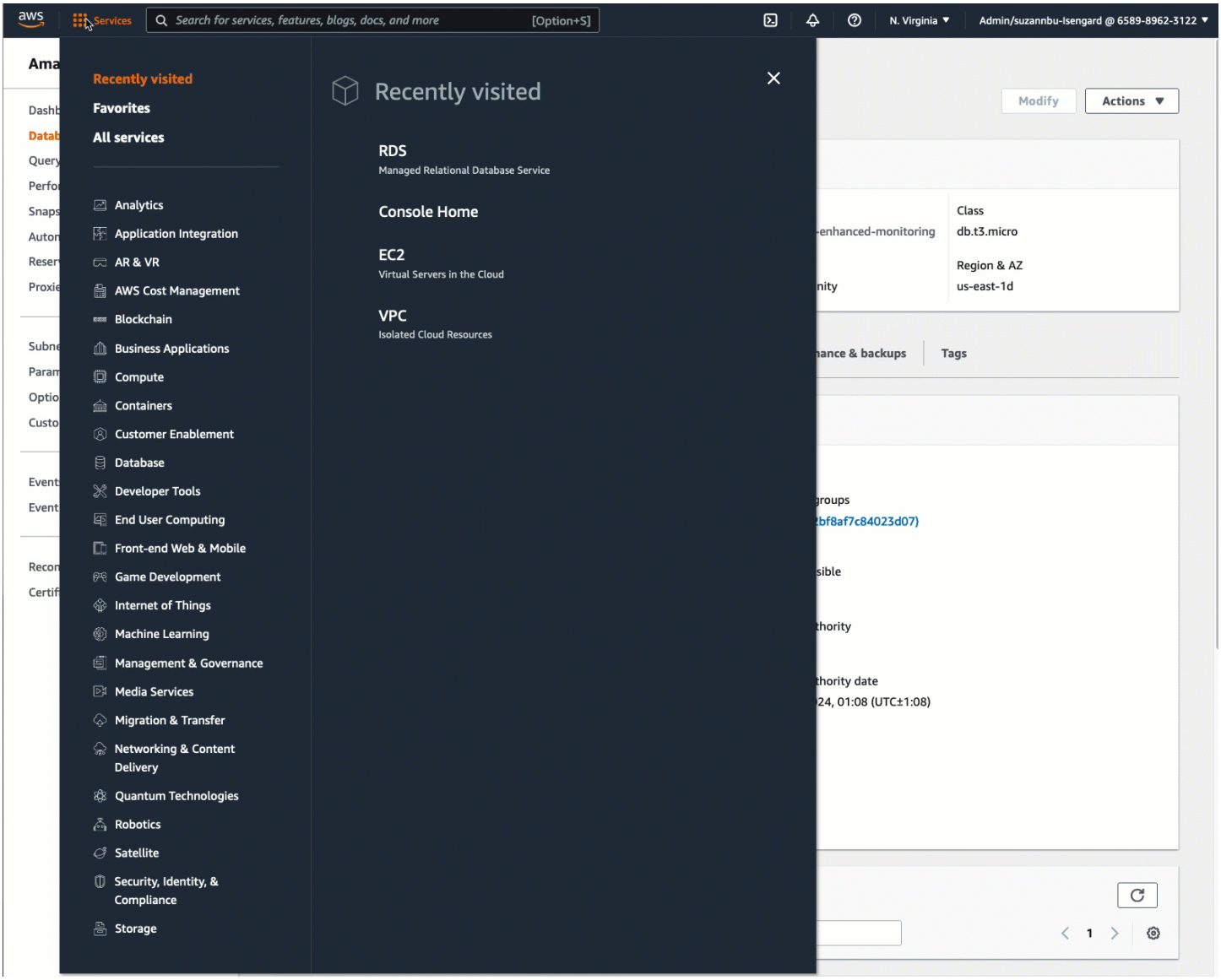
MySQL DB インスタンスを作成するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. リージョンセレクター (右上) から、EC2 インスタンスを作成した AWS リージョン を選択します。EC2 インスタンスと DB インスタンスは同じリージョンに存在する必要があります。
3. ダッシュボード で、[Create database] (データベースの作成) を選択します。
4. [Choose a database creation method] (データベース作成方法を選択) で [Easy Create] (簡易作成) を選択します。このオプションを選択すると、接続を自動的に構成する自動接続機能は使用できません。
5. [Engine options] (エンジンオプション) にある [Engine type] (エンジンタイプ) で MySQL を選択します。
6. [DB インスタンスサイズ] で、[無料利用枠] を選択します。
7. [DB instance identifier] (DB インスタンス識別子) に、RDS データベースの名前を入力します。このチュートリアルでは、**tutorial-database-manual** と入力します。

8. [Master username] (マスターユーザー名) は、デフォルトの名前である **admin** のままにします。
9. [Master password] (マスターパスワード) に、このチュートリアルに使用するパスワードを入力し、[Confirm password] (パスワードの確認) にパスワードをもう一度入力します。
10. [データベースの作成] を選択します。

    [Databases] (データベース) 画面では、DB インスタンスが使用可能な状態になるまで、新しい DB インスタンスの [Status] (ステータス) は [Creating] (作成中) になります。ステータスが [Available] (利用可能) に変わったら、DB インスタンスに接続できます。DB インスタンスクラスとストレージの合計によっては、新しいインスタンスを使用できるようになるまで最長 20 分かかることがあります。

アニメーションを表示: DB インスタンスの作成



これで、「タスク 3: セキュリティグループを作成してインスタンスに割り当てることで、EC2 インスタンスを RDS データベースに手動で接続する」を行う準備ができました。

タスク 3: セキュリティグループを作成してインスタンスに割り当てることで、EC2 インスタンスを RDS データベースに手動で接続する

タスクの目標

このタスクでは、次の手順を手動で実行して、自動接続機能の接続設定を再現します: 新しいセキュリティグループを 2 つ作成し、それぞれのセキュリティグループを EC2 インスタンスと RDS データベースに追加します。

新しいセキュリティグループを作成してインスタンスに追加するためのステップ

次のステップを使用して、2 つの新しいセキュリティグループを作成し、EC2 インスタンスを RDS データベースに接続します。次に、EC2 インスタンスと RDS データベースにそれぞれセキュリティグループを追加します。

2 つの新しいセキュリティグループを作成し、それぞれを EC2 インスタンスと RDS データベースに割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. まずは、EC2 インスタンスに追加するセキュリティグループを次のように作成します:
   a. ナビゲーションペインで、セキュリティグループ] を選択します。
   b. セキュリティグループの作成 を選択します。
   c. [Security group name] (セキュリティグループ名) には、分かりやすいセキュリティグループ名を入力します。このチュートリアルでは、**ec2-rds-manual-configuration** と入力します。
   d. [Description] (説明) に、簡単な説明を入力します。このチュートリアルでは、**EC2 instance security group to allow EC2 instance to securely connect to RDS database** と入力します。
   e. [Create Security Group] を選択します。RDS データベースのセキュリティグループを作成したら、このセキュリティグループに戻ってアウトバウンドルールを追加します。
3. 次に、以下の操作を行い、RDS データベースに追加するセキュリティグループを作成します。
   a. ナビゲーションペインで、セキュリティグループ] を選択します。
   b. セキュリティグループの作成 を選択します。
   c. [Security group name] (セキュリティグループ名) には、分かりやすいセキュリティグループ名を入力します。このチュートリアルでは、**rds-ec2-manual-configuration** と入力します。
   d. [Description] (説明) に、簡単な説明を入力します。このチュートリアルでは、**RDS database security group to allow EC2 instance to securely connect to RDS database** と入力します。
   e. [Inbound rules] (インバウンドルール) タブで [Add rule] (ルールの追加) を選択し、以下の操作を行います。
      i. [Type] (タイプ) では MySQL/Aurora を選択します。

ii. [Source] (ソース) には、この手順のステップ 2 で作成した EC2 インスタンスのセキュリティグループ ec2-rds-manual-configuration を選択します。

f. [Create Security Group] を選択します。

4. 次のように、EC2 インスタンスのセキュリティグループを編集して、アウトバウンドルールを追加します。

a. ナビゲーションペインで、セキュリティグループ] を選択します。

b. EC2 インスタンスのセキュリティグループ (**`ec2-rds-manual-configuration`** と名前を付けたもの) を選択し、[Outbound rules] (アウトバウンドルール) タブを選択します。

c. [Edit outbound rules] (アウトバウンドルールの編集) を選択します。

d. [Add rule] (ルールの追加) を選択し、次の操作を行います。

i. [Type] (タイプ) では MySQL/Aurora を選択します。

ii. [Source] (ソース) には、この手順のステップ 3 で作成した RDS データベースのセキュリティグループ rds-ec2-manual-configuration を選択します。

iii. [Save Rules] (ルールの保存) を選択します。

5. EC2 インスタンスのセキュリティグループを、次のように EC2 インスタンスに追加します:

a. ナビゲーションペインで、[インスタンス] を選択します。

b. EC2 インスタンスを選択し、[Actions] (アクション)、[Security] (セキュリティ)、[Change security groups] (セキュリティグループの変更) の順に選択します。

c. [Associated security groups] (関連するセキュリティグループ) で [Select security groups] (セキュリティグループの選択) フィールドを選択し、先程作成した ec2-rds-manual-configuration を選択して、[Add security group] (セキュリティグループの追加) を選択します。

d. [Save] を選択します。

6. 以下の操作を行い、RDS データベースのセキュリティグループを RDS データベースに追加します。

a. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。

b. ナビゲーションペインで [Databases] (データベース) を選択してから、使用するデータベースを選択します。

c. Modify を選択します。

d. [Connectivity] (接続) にある [Security group] (セキュリティグループ) では、先程作成した rds-ec2-manual-configuration を選択し、[Continue] (続行) を選択します。

e. [Scheduling of modifications] (変更のスケジュール) で [Apply immediately] (すぐに適用) を選択します。

f. [DB インスタンスを変更] を選択します。

これで、自動接続機能を使用した場合に発生する自動のステップを模倣した手動によるステップが完了しました。

これで、このチュートリアルのオプション 3 が完了しました。オプション 1、2、3 を完了し、このチュートリアルで作成したリソースが不要になった場合は、不要なコストが発生しないように、それらを削除する必要があります。詳細については、「クリーンアップ」を参照してください。

クリーンアップ

チュートリアルを完了したので、使用しなくなったリソースをすべてクリーンアップ (削除) することをおすすめします。AWS リソースをクリーンアップすることで、アカウントに追加料金が発生するのを防ぐことができます。

トピック

- EC2 インスタンスを終了する
- RDS データベースを削除するには

EC2 インスタンスを終了する

このチュートリアル専用に EC2 インスタンスを起動した場合、そのインスタンスを終了することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用してインスタンスを終了するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. このチュートリアルで作成したインスタンスを選択し、[Instance state] (インスタンスの状態)、[Terminate instance] (インスタンスの終了) の順に選択します。
4. 確認を求めるメッセージが表示されたら、[Terminate (終了)] を選択します。

RDS データベースを削除するには

このチュートリアル専用に RDS データベースを作成した場合、削除することで、それに関連した料金が発生するのを防ぐことができます。

コンソールを使用して RDS データベースを削除するには

1. Amazon RDS コンソール (https://console.aws.amazon.com/rds/) を開きます。
2. ナビゲーションペインで、[データベース] を選択します。
3. このチュートリアル用に作成した RDS データベースを選択し、[Actions] (アクション)、[Delete] (削除) を選択します。
4. ボックスに **`delete me`** を入力し、[Delete] (削除) をクリックします。

Amazon Linux インスタンスの設定

Amazon Linux インスタンスを正常に起動し、ログインしたら、変更できます。特定のアプリケーションのニーズを満たすためにインスタンスを設定するには、多くの方法があります。ここでは、初めて作業する場合の一般的なタスクについて説明します。

コンテンツ

一般的な設定シナリオ

Amazon Linux のベースのディストリビューションには、基本的なサーバー操作に必要なソフトウェアパッケージとユーティリティが含まれています。ただし、さまざまなソフトウェアリポジトリでさらに多くのソフトウェアパッケージを利用できます。また、ソースコードから、さらに多くのパッケージソースコードを作成できます。これらの場所からソフトウェアをインストールし、作成する方法についての詳細は、Amazon Linux インスタンスでのソフトウェアの管理 を参照してください。

Amazon Linux インスタンスには、`ec2-user` が事前設定されていますが、スーパーユーザー権限を持たない他のユーザーを追加することがあります。ユーザーの追加と削除についての詳細は、「Linux インスタンスのユーザーアカウントを管理する」を参照してください。

Amazon Linux インスタンスのデフォルトの時間設定では、Amazon Time Sync Service を利用し、システム時間をインスタンスに設定します。デフォルトの時間帯は UTC です。インスタンスの時間帯の設定または独自のタイムサーバーの利用についての詳細は、Linux インスタンスの時刻の設定 を参照してください。

お客様がネットワークを所有し、それにドメイン名を登録している場合、インスタンスのホスト名を変更して、そのドメインに含まれる一部としてインスタンスを識別できます。また、システムプロンプトを変更して、より意味のある名前を表示することもできます。ホスト名設定を変更する必要はありません。詳細については、Amazon Linux インスタンスのホスト名の変更 を参照してください。動的 DNS サービスプロバイダを使用するようにインスタンスを設定できます。詳細については、Amazon Linux インスタンスでの動的な DNS のセットアップ を参照してください。

Amazon EC2 でインスタンスを起動するとき、起動後にそのインスタンスにユーザーデータを渡し、一般的な設定タスクを実行したり、スクリプトを実行したりできます。2 つのタイプのユーザーデータを Amazon EC2 に渡すことができます。cloud-init ディレクティブとシェルスクリプトです。詳細については、起動時に Linux インスタンスでコマンドを実行する を参照してください。

Amazon Linux インスタンスでのソフトウェアの管理

Amazon Linux のベースのディストリビューションには、基本的なサーバー操作に必要なソフトウェアパッケージとユーティリティが含まれています。

この情報は、Amazon Linux 2 と Amazon Linux に適用されます。AL2023 の詳細については、「AL2023 ユーザーガイド」の「パッケージの管理とシステム更新の実行」を参照してください。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

コンテンツ

ソフトウェアは、最新の状態に維持することが重要です。Linux ディストリビューションの多くのパッケージは頻繁に更新されます。これにより、バグが修正され、機能が追加されて、セキュリティ上の弱点に対する防御措置が行われます。詳細については、Amazon Linux インスタンスでのインスタンスソフトウェアの更新 を参照してください。

デフォルトで、Amazon Linux インスタンスは、次のリポジトリを有効にして起動します。

- Amazon Linux 2: `amzn2-core`、および `amzn2extra-docker`
- Amazon Linux AMI: `amzn-main`、および `amzn-updates`

これらのリポジトリには、Amazon Web Services が更新するさまざまなパッケージが用意されていますが、別のリポジトリで、インストールしたいパッケージが見つかる可能性もあります。詳細については、「Amazon Linux インスタンスへのリポジトリの追加」を参照してください。有効なリポジトリでパッケージを検索してインストールする方法については、「Amazon Linux インスタンスでのソフトウェアパッケージの検索とインストール」を参照してください。

リポジトリに保管されているソフトウェアパッケージで、すべてのソフトウェアが利用できるわけではありません。一部のソフトウェアは、そのソースコードからインスタンスでコンパイルする必要があります。詳細については、Amazon Linux インスタンスでのソフトウェアのコンパイルの準備 を参照してください。

Amazon Linux インスタンスは、yum パッケージマネージャを利用してソフトウェアを管理します。yum パッケージマネージャはソフトウェアをインストール、削除、更新し、各パッケージのすべての依存関係を管理できます。Ubuntu のような Debian ベースの Linux ディストリビューションは、apt-get コマンドと dpkg パッケージマネージャを使用するため、次のセクションの yum の例は、このようなディストリビューションには該当しません。

Amazon Linux インスタンスでのインスタンスソフトウェアの更新

ソフトウェアは、最新の状態に維持することが重要です。Linux ディストリビューションのパッケージは頻繁に更新されます。これにより、バグが修正され、機能が追加されて、セキュリティ上の弱点に対する防御措置が行われます。最初に Amazon Linux インスタンスを起動して接続する際、セキュリティ上の目的から、ソフトウェアパッケージを更新するように促すメッセージが表示される場合があります。このセクションでは、システム全体またはパッケージを 1 つだけ更新する方法を紹介します。

この情報は、Amazon Linux 2 と Amazon Linux に適用されます。AL2023 の詳細については、「AL2023 ユーザーガイド」の「パッケージの管理とシステム更新の実行」を参照してください。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

Amazon Linux 2 の変更と更新については、「Amazon Linux 2 リリースノート」を参照してください。

AL2023 への変更と更新の詳細については、「AL2023 リリースノート」を参照してください。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

> **Important**
>
> Amazon Linux 2 AMI を使用する EC2 インスタンスを IPv6 専用サブネットに起動した場合は、インスタンスに接続して `sudo amazon-linux-https disable` を実行する必要があります。これにより、AL2 インスタンスが http パッチサービスを使用して、IPv6 経由で S3 の yum リポジトリに接続できるようになります。

Amazon Linux インスタンスのすべてのパッケージを更新するには

1. (オプション) シェルウィンドウで screen セッションを開始します。時折、ネットワークが遮断され、インスタンスへの SSH 接続が切断されることがあります。この状態が長時間におよぶソフトウェア更新中に発生した場合、インスタンスは混乱した状態になりますが、その復元は可能です。screen セッションを開始しておけば、接続が遮断された場合でも更新を続行でき、後で問題なくセッションに再接続できます。

   a. セッションを開始するには screen コマンドを実行します。

   ```
   [ec2-user ~]$ screen
   ```

   b. セッションが中断された場合、インスタンスにログインし直し、利用できる画面を表示します。

   ```
   [ec2-user ~]$ screen -ls
   There is a screen on:
    17793.pts-0.ip-12-34-56-78 (Detached)
   1 Socket in /var/run/screen/S-ec2-user.
   ```

   c. screen -r コマンドと前のコマンドのプロセス ID を使用して、画面に再接続します。

   ```
   [ec2-user ~]$ screen -r 17793
   ```

   d. screen の使用が終わったら、exit コマンドを使用してセッションを閉じます。

   ```
   [ec2-user ~]$ exit
   [screen is terminating]
   ```

2. yum update コマンドを実行します。オプションで、`--security` フラグを追加すれば、セキュリティ更新のみを適用できます。

```
[ec2-user ~]$ sudo yum update
```

3. 表示されたパッケージを確認したら、「**y**」と入力して Enter キーを押し、この更新を受け入れます。システムのパッケージをすべて更新するには数分かかります。実行中、yum 出力には更新のステータスが表示されます。
4. (オプション) インスタンスの再起動 により、更新によって最新のパッケージおよびライブラリが使用されていることを確認します。カーネル更新は再起動が発生するまでロードされません。glibc ライブラリを更新した後も再起動が必要です。サービスを制御する更新の場合は、更新を取得するにはサービスの再起動で十分かもしれませんが、システムを再起動することで、それ以前のすべてのパッケージとライブラリの更新を確実に完了できます。

Amazon Linux インスタンスの 1 つのパッケージを更新するには

システム全体ではなく、1 つのパッケージ (とその依存関係) を更新するには、この手順を使用します。

1. 更新するパッケージの名前を指定した yum update コマンドを実行します。

```
[ec2-user ~]$ sudo yum update openssl
```

2. 表示されたパッケージ情報を確認したら、「**y**」と入力して Enter キーを押し、この更新を受け入れます。時折、解決する必要があるパッケージ依存関係ある場合、リストには複数のパッケージがあります。実行中、yum 出力には更新のステータスが表示されます。
3. (オプション) インスタンスの再起動 により、更新によって最新のパッケージおよびライブラリが使用されていることを確認します。カーネル更新は再起動が発生するまでロードされません。glibc ライブラリを更新した後も再起動が必要です。サービスを制御する更新の場合は、更新を取得するにはサービスの再起動で十分かもしれませんが、システムを再起動することで、それ以前のすべてのパッケージとライブラリの更新を確実に完了できます。

Amazon Linux インスタンスへのリポジトリの追加

この情報は、Amazon Linux 2 と Amazon Linux に適用されます。AL2023 の詳細については、「AL2023 ユーザーガイド」の「バージョン管理されたリポジトリによる確定的なアップグレードの使用」を参照してください。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

デフォルトで、Amazon Linux インスタンスは、次のリポジトリを有効にして起動します。

- Amazon Linux 2: `amzn2-core`、および `amzn2extra-docker`
- Amazon Linux AMI: `amzn-main`、および `amzn-updates`

これらのリポジトリには、Amazon Web Servicesが更新するさまざまなパッケージが用意されていますが、別のリポジトリで、インストールしたいパッケージが見つかる可能性もあります。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

yum で異なるリポジトリからパッケージをインストールには、`/etc/yum.conf` ファイル、または *`repository`*`.repo` ディレクトリにあるお客様の `/etc/yum.repos.d` ファイルに、リポジトリ情報を追加する必要があります。これは手動で行えますが、ほとんどの yum リポジトリのリポジトリ URL で、独自の *`repository`*`.repo` ファイルが提供されています。

既にインストールされている yum レポジトリを調べるには

- 次のコマンドで、インストール済みの yum リポジトリを表示します。

  ```
  [ec2-user ~]$ yum repolist all
  ```

  生成される出力には、インストール済みのリポジトリが一覧表示され、それぞれのステータスが報告されます。有効なリポジトリには、そこに含まれているパッケージの数が表示されます。

yum リポジトリを **/etc/yum.repos.d** に追加するには

1. .repo ファイルの場所を検索します。場所は、追加しているリポジトリによって異なります。この例では、.repo ファイルは、https://www.*example*.com/*repository*.repo にあります。
2. yum-config-manager コマンドを使用してリポジトリを追加します。

```
[ec2-user ~]$ sudo yum-config-manager --add-repo https://
www.example.com/repository.repo
Loaded plugins: priorities, update-motd, upgrade-helper
adding repo from: https://www.example.com/repository.repo
grabbing file https://www.example.com/repository.repo to /etc/
yum.repos.d/repository.repo
repository.repo                                          | 4.0 kB     00:00
repo saved to /etc/yum.repos.d/repository.repo
```

リポジトリをインストールしたら、次の手順で説明するように有効にする必要があります。

yum リポジトリを **/etc/yum.repos.d** で有効にするには

- yum-config-manager フラグを付けて --enable *repository* コマンドを使用します。次のコマンドを使用すると、Fedora プロジェクトの Extra Packages for Enterprise Linux (EPEL) リポジトリが有効になります。デフォルトでは、このリポジトリは Amazon Linux AMI インスタンスの /etc/yum.repos.d にありますが、有効になっていません。

```
[ec2-user ~]$ sudo yum-config-manager --enable epel
```

> **Note**
>
> Amazon Linux 2 で EPEL レポジトリを有効にするには、次のコマンドを使用します。
>
> ```
> [ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
> ```
>
> 他のディストリビューションの EPEL リポジトリ (Red Hat や CentOS など) を有効にする方法については、EPEL ドキュメント (https://fedoraproject.org/wiki/EPEL) を参照してください。

Amazon Linux インスタンスでのソフトウェアパッケージの検索とインストール

パッケージ管理ツールを使用して、ソフトウェアパッケージを検索してインストールできます。Amazon Linux 2 および Amazon Linux では、デフォルトのソフトウェアパッケージ管理ツールは YUM です。AL2023 では、デフォルトのソフトウェアパッケージ管理ツールは DNF です。AL2023 の詳細については、「AL2023 ユーザーガイド」の「パッケージ管理ツール」を参照してください。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

Amazon Linux インスタンスでのソフトウェアパッケージの検索

yum search コマンドを使用すると、設定したリポジトリで利用できるパッケージの説明を検索できます。これは特に、インストールするパッケージの正確な名前がわからない場合に便利です。キーワード検索をコマンドに追加します。複数の単語を検索するには、引用符で検索クエリを囲みます。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

```
[ec2-user ~]$ yum search "find"
```

Amazon Linux 2 の出力例を次に示します。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
============================== N/S matched: find ===============================
findutils.x86_64 : The GNU versions of find utilities (find and xargs)
gedit-plugin-findinfiles.x86_64 : gedit findinfiles plugin
ocaml-findlib-devel.x86_64 : Development files for ocaml-findlib
```

```
perl-File-Find-Rule.noarch : Perl module implementing an alternative interface to
 File::Find
robotfindskitten.x86_64 : A game/zen simulation. You are robot. Your job is to find
 kitten.
mlocate.x86_64 : An utility for finding files by name
ocaml-findlib.x86_64 : Objective CAML package manager and build helper
perl-Devel-Cycle.noarch : Find memory cycles in objects
perl-Devel-EnforceEncapsulation.noarch : Find access violations to blessed objects
perl-File-Find-Rule-Perl.noarch : Common rules for searching for Perl things
perl-File-HomeDir.noarch : Find your home and other directories on any platform
perl-IPC-Cmd.noarch : Finding and running system commands made easy
perl-Perl-MinimumVersion.noarch : Find a minimum required version of perl for Perl code
texlive-xesearch.noarch : A string finder for XeTeX
valgrind.x86_64 : Tool for finding memory management bugs in programs
valgrind.i686 : Tool for finding memory management bugs in programs
```

Amazon Linux の出力例を次に示します。

```
Loaded plugins: priorities, security, update-motd, upgrade-helper
============================== N/S Matched: find ==============================
findutils.x86_64 : The GNU versions of find utilities (find and xargs)
perl-File-Find-Rule.noarch : Perl module implementing an alternative interface to
 File::Find
perl-Module-Find.noarch : Find and use installed modules in a (sub)category
libpuzzle.i686 : Library to quickly find visually similar images (gif, png, jpg)
libpuzzle.x86_64 : Library to quickly find visually similar images (gif, png, jpg)
mlocate.x86_64 : An utility for finding files by name
```

引用符で囲まれた複数の単語検索クエリは、正確なクエリに一致する結果のみを返します。予想されたパッケージが表示されない場合、キーワードを 1 つに絞って検索し、結果をスキャンします。キーワードの同義語を試して、検索の幅を広げることもできます。

Amazon Linux 2 および Amazon Linux のパッケージの詳細については、以下を参照してください。

- Extras library (Amazon Linux 2)
- パッケージリポジトリ

Amazon Linux インスタンスでのソフトウェアパッケージのインストール

Amazon Linux 2 および Amazon Linux では、yum パッケージ管理ツールが、異なるソフトウェアパッケージの有効になっているすべてのリポジトリを検索し、ソフトウェアのインストールプロセス

に伴う依存関係を処理します。AL2023 でのソフトウェアパッケージのインストールの詳細については、「AL2023 ユーザーガイド」の「パッケージの管理およびシステム更新の実行」を参照してください。

リポジトリからパッケージをインストールするには

yum install ***package*** コマンドを使用します。この際、*package* はインストールするソフトウェアの名前に置き換えます。例えば、links テキストベースウェブブラウザをインストールするには、次のコマンドを入力します。

```
[ec2-user ~]$ sudo yum install links
```

ダウンロードした RPM パッケージファイルをインストールするには

また、yum install を使用して、インターネットからダウンロードした RPM パッケージファイルをインストールすることもできます。その場合には、リポジトリのパッケージ名の代わりに、RPM ファイルのパス名をインストールコマンドに追加します。

```
[ec2-user ~]$ sudo yum install my-package.rpm
```

インストールされているパッケージを一覧表示するには

インスタンスにインストールされているパッケージを一覧表示するには、次のコマンドを使用します。

```
[ec2-user ~]$ yum list installed
```

Amazon Linux インスタンスでのソフトウェアのコンパイルの準備

オープンソースのソフトウェアは、事前コンパイルされていないインターネットで使用できます。これらは、パッケージリポジトリからダウンロードできます。入手したソフトウェアパッケージがソースコードであり、自分でコンパイルする必要があると判明することがあります。システムで Amazon Linux 2 のソフトウェアのコンパイルを可能にするには、make、gcc、autoconf など、いくつかの開発ツールをインストールする必要があります。

> **Note**
>
> Amazon Linux AMI は 2023 年 12 月 31 日にサポート終了となり、2024 年 1 月 1 日以降、セキュリティアップデートやバグ修正は一切行われません。Amazon Linux AMI のサポート終了およびメンテナンスサポートの詳細については、ブログ記事「Update on Amazon Linux

AMI end-of-life」を参照してください。アプリケーションを AL2023 にアップグレードすることをお勧めします。これには 2028 年までの長期サポートが含まれます。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

ソフトウェアのコンパイルはすべての Amazon EC2 インスタンスで必要なタスクではないため、そのようなツールはデフォルトでインストールされていません。ただし、「Development Tools」という名前のパッケージグループで利用でき、yum groupinstall コマンドでインスタンスに簡単に追加されます。

```
[ec2-user ~]$ sudo yum groupinstall "Development Tools"
```

ソフトウェアのソースコードパッケージは、多くの場合、tarball と呼ばれる圧縮アーカイブファイルの形で (https://github.com/ や http://sourceforge.net/ などのウェブサイトから) ダウンロードできます。通常、これらの tarball には `.tar.gz` というファイル拡張子が付いています。これらのアーカイブは tar コマンドで解凍できます。

```
[ec2-user ~]$ tar -xzf software.tar.gz
```

ソースコードパッケージを解凍したら、ソースコードディレクトリで `README` ファイルまたは `INSTALL` ファイルを探します。これらのファイルに、ソースコードのコンパイルとインストールに関する詳細な指示があります。

Amazon Linux パッケージのソースコードを取得するには

Amazon Web Services は、保守管理されているパッケージのソースコードを提供します。yumdownloader --source コマンドを使用して、インストールされているパッケージのソースコードをダウンロードできます。

- yumdownloader --source ***package*** コマンドを実行して、*package* のソースコードをダウンロードします。例えば、`htop` パッケージのソースコードをダウンロードするには、次のコマンドを入力します。

```
[ec2-user ~]$ yumdownloader --source htop

Loaded plugins: priorities, update-motd, upgrade-helper
Enabling amzn-updates-source repository
Enabling amzn-main-source repository
amzn-main-source
                           | 1.9 kB  00:00:00
amzn-updates-source
                           | 1.9 kB  00:00:00
(1/2): amzn-updates-source/latest/primary_db
                           |  52 kB  00:00:00
(2/2): amzn-main-source/latest/primary_db
                           | 734 kB  00:00:00
htop-1.0.1-2.3.amzn1.src.rpm
```

ソース RPM の場所は、コマンドを実行したディレクトリにあります。

Linux インスタンスのユーザーアカウントを管理する

各 Linux インスタンスは、デフォルトの Linux システムユーザーで起動されます。インスタンスには、ユーザーを追加することも、削除することもできます。

デフォルトのユーザーの場合、デフォルトのユーザー名は、インスタンスの起動時に指定した AMI によって決定されます。

> **Note**
>
> デフォルトでは、パスワード認証とルートログインは無効になっており、また、sudo は有効化されています。インスタンスにログインするには、キーペアを使用する必要があります。ログインの詳細については、「Linux インスタンスへの接続」を参照してください。
> ユーザーは、インスタンスのパスワード認証とルートログインを許可できます。詳細については、「インスタンスのオペレーティングシステムに関するドキュメント」を参照してください。

> **Note**
>
> Linux システムユーザーと IAM ユーザーを混同しないようにしてください。詳細については、『IAM ユーザーガイド』の「IAM ユーザー」を参照してください。

コンテンツ

デフォルトのユーザー名

EC2 インスタンスでのデフォルトのユーザー名は、そのインスタンスの起動時に指定した AMI によって決まります。

デフォルトのユーザー名は以下のとおりです。

- AL2023、Amazon Linux 2 または Amazon Linux AMI の場合、ユーザー名は `ec2-user` です。
- Centos AMI の場合、ユーザー名は `centos` または `ec2-user`です。
- Debian AMI の場合は、ユーザー名は `admin` です。
- Fedora AMI の場合、ユーザー名は `fedora` または `ec2-user` です。
- RHEL AMI の場合、ユーザー名は `ec2-user` または `root` です。
- SUSE AMI の場合、ユーザー名は `ec2-user` または `root` です。
- Ubuntu AMI の場合、ユーザー名は `ubuntu` です。
- SUSE AMI の場合、ユーザー名は `ec2-user` です。
- Bitnami AMI の場合は、ユーザー名は `bitnami` です。

> **Note**
>
> 他の Linux ディストリビューションのデフォルトのユーザー名を確認するには、AMI プロバイダーに確認してください。

考慮事項

デフォルトのユーザーを使用するのが多くのアプリケーションに適しています。ただし、個人が自分のファイルとワークスペースを持つことができるように、ユーザーを追加することを選択できます。さらに、新しいユーザー用にユーザーを作成することは、デフォルトユーザーへのアクセス権を複数のユーザーに (経験のないユーザーも含めて) 与えるよりも、はるかに安全です。これはデフォルトのユーザーが不適切に使用された場合、システムにさまざまな損害を与える可能性があるためです。詳細については、「EC2 インスタンスの保護のヒント」を参照してください。

Linux システムのユーザーを使用してユーザーが EC2 インスタンスに SSH アクセスできるようにするには、SSH キーをユーザーと共有する必要があります。または、EC2 Instance Connect を使用して、SSH キーを共有および管理せずにユーザーにアクセスを提供できます。詳細については、「EC2 Instance Connect を使用して Linux インスタンスに接続する」を参照してください。

ユーザーの作成

最初にユーザーを作成してから、ユーザーがインスタンスに接続してログインできるようにする SSH パブリックキーを追加します。

ユーザーを作成するには

1. 新しいキーペアを作成します。この .pem ファイルは、ユーザーを作成するユーザーに提供する必要があります。ユーザーがインスタンスに接続するには、このファイルを使用する必要があります。
2. 前のステップで作成したキーペアからパブリックキーを取得します。

   ```
   $  ssh-keygen -y -f /path_to_key_pair/key-pair-name.pem
   ```

   コマンドは、次の例に示すように、パブリックキーを返します。

   ```
   ssh-rsa
    AAAAB3NzaC1yc2EAAAADAQABAAABAQClKsfkNkuSevGj3eYhCe53pcjqP3maAhDFcvBS7O6Vhz2ItxCih
   +PnDSUaw+WNQn/mZphTk/a/gU8jEzoOWbkM4yxyb/wB96xbiFveSFJuOp/
   d6RJhJOI0iBXrlsLnBItntckiJ7FbtxJMXLvvwJryDUilBMTjYtwB+QhYXUMOzce5Pjz5/
   i8SeJtjnV3iAoG/cQk+0FzZqaeJAAHco
   +CY/5WrUBkrHmFJr6HcXkvJdWPkYQS3xqC0+FmUZofz221CBt5IMucxXPkX4rWi
   +z7wB3RbBQoQzd8v7yeb7OzlPnWOyN0qFU0XA246RA8QFYiCNYwI3f05p6KLxEXAMPLE
   ```

3. インスタンスに接続します。

4. adduser コマンドを使用して、ユーザーを作成し、システムに追加します (/etc/passwd ファイルにエントリが追加されます)。このコマンドでも、グループが作成され、ユーザーのホームディレクトリが作成されます。この例では、ユーザーは *newuser* という名前になります。

   - Amazon Linux および Amazon Linux 2

     Amazon Linux および Amazon Linux 2 では、パスワード認証が無効化されたデフォルトの状態で、ユーザーが作成されます。

     ```
     [ec2-user ~]$ sudo adduser newuser
     ```

   - Ubuntu

     パスワード認証が無効化されたユーザーを作成するには、--disabled-password パラメータを含めます。

     ```
     [ubuntu ~]$ sudo adduser newuser --disabled-password
     ```

5. 新しいユーザーに切り替えて、作成するディレクトリとファイルが適切な所有権を持つようにします。

   ```
   [ec2-user ~]$ sudo su - newuser
   ```

   シェルセッションが新しいユーザーに切り替わったことを示すために ec2-user から *newuser* に変更するように求められます。

6. ユーザーに SSH パブリックキーを追加します。以下のサブステップで説明しているように、最初に SSH キーファイル用のディレクトリをユーザーのホームディレクトリに作成し、次にキーファイルを作成して、最後に公開キーをキーファイルに貼り付けます。

   a. .ssh ホームディレクトリに *newuser* ディレクトリを作成し、そのファイルのアクセス許可を 700 (所有者のみ、読み取り、書き込み、削除が可能) に変更します。

      ```
      [newuser ~]$ mkdir .ssh
      ```

      ```
      [newuser ~]$ chmod 700 .ssh
      ```

> **Important**
> 厳密なファイル権限がなければ、ユーザーはログインできません。

b. authorized_keys という名前のファイルを .ssh ディレクトリに作成し、そのファイルのアクセス許可を 600 (所有者のみ、読み取りおよび書き込みが可能) に変更します。

```
[newuser ~]$ touch .ssh/authorized_keys
```

```
[newuser ~]$ chmod 600 .ssh/authorized_keys
```

> **Important**
> 厳密なファイル権限がなければ、ユーザーはログインできません。

c. お好みのテキストエディタ (例: vim や nano) で、authorized_keys ファイルを開きます。

```
[newuser ~]$ nano .ssh/authorized_keys
```

ステップ 2 で取得したパブリックキーをファイルに貼り付け、変更を保存します。

> **Important**
> パブリックキーは、必ず 1 つの連続した行に貼り付けてください。パブリックキーを複数行に分割することはできません。

これで、authorized_keys ファイルに追加したパブリックキーの対であるプライベートキーを使用して、インスタンスの *newuser* ユーザーにログインできるようになりました。Linux インスタンスに接続するさまざまな方法の詳細については、「Linux インスタンスへの接続」を参照してください。

ユーザーの削除

ユーザーが不要になった場合、今後使用されないようにそのユーザーを削除できます。

システムからユーザーアカウントを削除するには、userdel コマンドを使用します。-r パラメータを指定すると、ユーザーのホームディレクトリとメールスプールが削除されます。ユーザーのホームディレクトリとメールスプールを維持するには、-r パラメータを省略します。

```
[ec2-user ~]$ sudo userdel -r olduser
```

EC2 インスタンスのプロセッサのステート制御

C ステートはアイドル時のコアのスリープレベルを制御します。C ステートは、C0 (コアがアクティブで、命令を実行している最も浅い状態) から始まる番号が付けられ、C6 (コアの電源がオフになっている最も深いアイドル状態) まで移行します。

P ステートはコアに希望するパフォーマンス (CPU 周波数) を制御します。P ステートは、P0 (コアが Intel Turbo Boost Technology を使用して可能であれば周波数を上げることができる最高パフォーマンスの設定) から始まる番号が付けられ、P1 (最大限のベースライン周波数をリクエストする P ステート) から P15 (最小限の周波数) まで移行します。

C ステートと P ステート

次のインスタンスタイプにより、オペレーティングシステムがプロセッサの C ステートと P ステートを制御できるようになります。

- 汎用: m4.10xlarge | m4.16xlarge | m5.metal | m5d.metal | m5n.metal | m5zn.metal | m6i.metal | m6id.metal | m7i.metal-24xl | m7i.metal-48xl
- コンピューティングの最適化: c4.8xlarge | c5.metal | c5an.metal | c5adn.metal | c5n.metal | c6i.metal | c6id.metal | c7i.metal-24xl | c7i.metal-48xl
- メモリ最適化: r4.8xlarge | r4.16xlarge | r5.metal | r5b.metal | r5d.metal | r6i.metal | r7i.metal-24xl | r7i.metal-48xl | r7iz.metal-16xl | r7iz.metal-32xl | u-6tb1.metal | u-9tb1.metal | u-12tb1.metal | u-18tb1.metal | u-24tb1.metal | x1.16xlarge | x1.32xlarge | x1e.8xlarge | x1e.16xlarge | x1e.32xlarge | z1d.metal
- ストレージの最適化: d2.8xlarge | d3.metal | d3en.metal | i3.8xlarge | i3.16xlarge | i3.metal | i3en.metal | h1.8xlarge | h1.16xlarge
- 高速コンピューティング: f1.16xlarge | g3.16xlarge | g4dn.metal | p2.16xlarge | p3.16xlarge

C ステートのみ

次のインスタンスタイプにより、オペレーティングシステムがプロセッサの C ステートを制御できるようになります。

- 凡用: m5.12xlarge | m5.24xlarge | m5d.12xlarge | m5d.24xlarge | m5n.12xlarge | m5n.24xlarge | m5dn.12xlarge | m5dn.24xlarge | m6a.24xlarge | m6a.48xlarge | m6ad.metal | m6i.16xlarge | m6i.32xlarge | m7i.large | m7i.xlarge | m7i.2xlarge | m7i.4xlarge | m7i.8xlarge | m7i.12xlarge | m7i.16xlarge | m7i.24xlarge | m7i.48xlarge
- コンピューティング最適化: c5.9xlarge | c5.12xlarge | c5.18xlarge | c5.24xlarge | c5a.24xlarge | c5ad.24xlarge | c5d.9xlarge | c5d.12xlarge | c5d.18xlarge | c5d.24xlarge | c5n.9xlarge | c5n.18xlarge | c6a.24xlarge | c6a.32xlarge | c6a.48xlarge | c6i.16xlarge | c6i.32xlarge | c7i.large | c7i.xlarge | c7i.2xlarge | c7i.4xlarge | c7i.8xlarge | c7i.12xlarge | c7i.16xlarge | c7i.24xlarge | c7i.48xlarge
- メモリ最適化: r5.12xlarge | r5.24xlarge | r5d.12xlarge | r5d.24xlarge | r5n.12xlarge | r5n.24xlarge | r5dn.12xlarge | r5dn.24xlarge |r6a.24xlarge | r6a.48xlarge | r6i.16xlarge | r6i.32xlarge | r6id.32xlarge | r6in.32xlarge | r7i.large | r7i.xlarge | r7i.2xlarge | r7i.4xlarge | r7i.8xlarge | r7i.12xlarge | r7i.16xlarge | r7i.24xlarge | r7i.48xlarge | r7iz.large | r7iz.xlarge | r7iz.2xlarge | r7iz.4xlarge | r7iz.8xlarge | r7iz.12xlarge | r7iz.16xlarge | r7iz.32xlarge | u-6tb1.56xlarge | u-6tb1.112xlarge | u-9tb1.112xlarge | u-12tb1.112xlarge | u-18tb1.112xlarge | u-24tb1.112xlarge | z1d.6xlarge | z1d.12xlarge
- ストレージ最適化: d3en.12xlarge | dl1.24xlarge | i3en.12xlarge | i3en.24xlarge | i4i.metal | r5b.12xlarge | r5b.24xlarge | i4i.16xlarge
- 高速コンピューティング:dl1.24xlarge | g5.24xlarge | g5.48xlarge | inf1.24xlarge | p3dn.24xlarge | p4d.24xlarge | p4de.24xlarge | vt1.24xlarge

AWS Graviton プロセッサには組み込みの省電力モードがあり、固定周波数で動作します。そのため、オペレーティングシステムが C ステートと P ステートを制御する機能は提供されていません。

プロセッサのパフォーマンスの安定性を向上させたり、レイテンシーを減らしたり、インスタンスを特定のワークロード用に調整するために、C ステートまたは P ステートの設定を変更したいと思う場合があるかもしれません。デフォルトの C ステートおよび P ステートの設定は、ほとんどの作業負荷に対して最適なパフォーマンスを提供します。ただし、アプリケーションにおいて、より高いシングルコアまたはデュアルコアの周波数でレイテンシーを軽減したい場合、またはバースト的

な Turbo Boost 周波数よりも低い周波数でより安定したパフォーマンスを維持することを優先する場合、これらのインスタンスで利用可能な C ステートまたは P ステートを試みることを考慮してください。

以下のセクションでは、プロセッサのさまざまなステート設定と、設定の効果をモニタリングする方法について説明します。これらの手順は、Amazon Linux を対象としており、これに適用されますが、バージョン 3.9 以降の Linux カーネルを持つ他の Linux ディストリビューションでも使用できる可能性があります。他の Linux ディストリビューションやプロセッサのステート制御については、システム固有ドキュメントを参照してください。

> **Note**
>
> このページの例では、以下を使用しています。
>
> - プロセッサの周波数と C ステート情報を表示する turbostat ユーティリティ。turbostat ユーティリティは Amazon Linux でデフォルトで使用できます。
> - ワークロードをシミュレートする stress コマンド。stress をインストールするには、まず sudo amazon-linux-extras install epel を実行して EPEL リポジトリを有効にし、次に sudo yum install -y stress を実行します。
>
> 出力に C ステート情報が表示されない場合は、コマンド (--debug) に sudo turbostat --debug stress ***<options>*** オプションを含めます。

コンテンツ

最大 Turbo Boost 周波数による最高パフォーマンス

これは、Amazon Linux AMI のデフォルトのプロセッサのステート制御設定であり、ほとんどのワークロードにお勧めします。この設定では、変動性を抑え、最高パフォーマンスを実現します。非アクティブなコアは深いスリープ状態になることができるため、シングルまたはデュアルコアプロセスが Turbo Boost の潜在能力を最大限に引き出すために必要な発熱量の余裕を実現できます。

次の例は、2 個のコアでアクティブに処理を実行する c4.8xlarge インスタンスが Turbo Boost の最大周波数に到達した状況を示しています。

```
[ec2-user ~]$ sudo turbostat stress -c 2 -t 10
stress: info: [30680] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: info: [30680] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
            5.54 3.44 2.90   0   9.18   0.00  85.28   0.00   0.00   0.00   0.00   0.00
  94.04 32.70 54.18  0.00
 0   0   0   0.12 3.26 2.90   0   3.61   0.00  96.27   0.00   0.00   0.00   0.00   0.00
  48.12 18.88 26.02  0.00
 0   0  18   0.12 3.26 2.90   0   3.61
 0   1   1   0.12 3.26 2.90   0   4.11   0.00  95.77   0.00
 0   1  19   0.13 3.27 2.90   0   4.11
 0   2   2   0.13 3.28 2.90   0   4.45   0.00  95.42   0.00
 0   2  20   0.11 3.27 2.90   0   4.47
 0   3   3   0.05 3.42 2.90   0  99.91   0.00   0.05   0.00
 0   3  21  97.84 3.45 2.90   0   2.11
...
 1   1  10   0.06 3.33 2.90   0  99.88   0.01   0.06   0.00
 1   1  28  97.61 3.44 2.90   0   2.32
...
10.002556 sec
```

この例では、vCPU 21 と 28 が最大 Turbo Boost 周波数で実行され、他のコアは C6 スリープ状態になることで電力を節約し、実行中のコアに電力と発熱の余裕を持たせています。vCPU 3 と 10 (それぞれ vCPU 21 および 28 とプロセッサコアを共有する) は C1 ステートであり、命令を待っています。

以下の例では、18 個のコアはすべてアクティブに処理を実行しているため、Turbo Boost の最大周波数のための余裕はありませんが、すべてのコアが「all core Turbo Boost」の速度である 3.2 GHz で実行されています。

```
[ec2-user ~]$ sudo turbostat stress -c 36 -t 10
stress: info: [30685] dispatching hogs: 36 cpu, 0 io, 0 vm, 0 hdd
stress: info: [30685] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
           99.27 3.20 2.90   0   0.26   0.00   0.47   0.00   0.00   0.00   0.00   0.00
 228.59 31.33 199.26  0.00
```

```
 0   0   0  99.08 3.20 2.90   0   0.27   0.01   0.64   0.00   0.00   0.00   0.00   0.00
 114.69 18.55 99.32  0.00
 0   0  18  98.74 3.20 2.90   0   0.62
 0   1   1  99.14 3.20 2.90   0   0.09   0.00   0.76   0.00
 0   1  19  98.75 3.20 2.90   0   0.49
 0   2   2  99.07 3.20 2.90   0   0.10   0.02   0.81   0.00
 0   2  20  98.73 3.20 2.90   0   0.44
 0   3   3  99.02 3.20 2.90   0   0.24   0.00   0.74   0.00
 0   3  21  99.13 3.20 2.90   0   0.13
 0   4   4  99.26 3.20 2.90   0   0.09   0.00   0.65   0.00
 0   4  22  98.68 3.20 2.90   0   0.67
 0   5   5  99.19 3.20 2.90   0   0.08   0.00   0.73   0.00
 0   5  23  98.58 3.20 2.90   0   0.69
 0   6   6  99.01 3.20 2.90   0   0.11   0.00   0.89   0.00
 0   6  24  98.72 3.20 2.90   0   0.39
...
```

深い C ステートの制限による高パフォーマンスと低レイテンシー

C ステートは非アクティブ時のコアのスリープレベルを制御します。C ステートを制御して、システムのレイテンシーとパフォーマンスを調整することができます。コアをスリープ状態にするには時間がかかります。また、スリープ状態のコアによって、別のコアが高い周波数で動作するための余裕が生まれますが、そのスリープ状態にあるコアが再び稼働し処理を実行するのにも時間がかかります。例えば、ネットワークパケットの中断を処理するように割り当てられたコアがスリープ状態である場合、その中断の処理に遅延が生じる可能性があります。より深い C ステートを使用しないようにシステムを設定できます。これにより、プロセッサの応答のレイテンシーは減少しますが、他のコアの Turbo Boost 用の余裕も減少します。

深いスリープ状態を無効にする一般的なシナリオとして、Redis データベースアプリケーションがあります。このアプリケーションは、最速のクエリ応答時間を実現するために、データベースをシステムメモリ内に格納します。

Amazon Linux 2 で深いスリープ状態を制限するには

1. 適切なエディタで /etc/default/grub ファイルを開きます。

   ```
   [ec2-user ~]$ sudo vim /etc/default/grub
   ```

2. GRUB_CMDLINE_LINUX_DEFAULT 行を編集し intel_idle.max_cstate=1 と processor.max_cstate=1 のオプションを追加します。これにより、アイドル状態にあるコアの最も深い C ステートとして C1 を設定します。

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8 net.ifnames=0
 biosdevname=0 nvme_core.io_timeout=4294967295 intel_idle.max_cstate=1
 processor.max_cstate=1"
GRUB_TIMEOUT=0
```

intel_idle.max_cstate=1 オプションでは、インテルベースのインスタンスで C ステート制限が設定され、また、processor.max_cstate=1 オプションでは、AMD ベースのインスタンスで C ステート制限が設定されます。両方のオプションを設定に追加しておくと安心です。これにより、インテルと AMD のインスタンスに対し、共通の方法で目的の動作を設定することができます。

3. ファイルを保存し、エディタを終了します。
4. 起動設定を再構築するには、次のコマンドを実行します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

5. 新しい kernel オプションを有効にするためにインスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

Amazon Linux AMI で深いスリープ状態を制限するには

1. 適切なエディタで /boot/grub/grub.conf ファイルを開きます。

```
[ec2-user ~]$ sudo vim /boot/grub/grub.conf
```

2. 最初のエントリの kernel 行を編集して intel_idle.max_cstate=1 と processor.max_cstate=1 オプションを追加し、アイドル状態のコアの最も深い C ステートに C1 を設定します。

```
# created by imagebuilder
default=0
timeout=1
hiddenmenu

title Amazon Linux 2014.09 (3.14.26-24.46.amzn1.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-3.14.26-24.46.amzn1.x86_64 root=LABEL=/ console=ttyS0
 intel_idle.max_cstate=1  processor.max_cstate=1
```

```
initrd /boot/initramfs-3.14.26-24.46.amzn1.x86_64.img
```

intel_idle.max_cstate=1 オプションでは、インテルベースのインスタンスで C ステート制限が設定され、また、processor.max_cstate=1 オプションでは、AMD ベースのインスタンスで C ステート制限が設定されます。両方のオプションを設定に追加しておくと安心です。これにより、インテルと AMD のインスタンスに対し、共通の方法で目的の動作を設定することができます。

3. ファイルを保存し、エディタを終了します。
4. 新しい kernel オプションを有効にするためにインスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

次の例では、2 つのコアが「all core Turbo Boost」コア周波数でアクティブに処理を実行している c4.8xlarge インスタンスを示します。

```
[ec2-user ~]$ sudo turbostat stress -c 2 -t 10
stress: info: [5322] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: info: [5322] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
            5.56 3.20 2.90   0  94.44   0.00   0.00   0.00   0.00   0.00   0.00   0.00
 131.90 31.11 199.47  0.00
 0   0   0   0.03 2.08 2.90   0  99.97   0.00   0.00   0.00   0.00   0.00   0.00   0.00
  67.23 17.11 99.76  0.00
 0   0  18   0.01 1.93 2.90   0  99.99
 0   1   1   0.02 1.96 2.90   0  99.98   0.00   0.00   0.00
 0   1  19  99.70 3.20 2.90   0   0.30
...
 1   1  10   0.02 1.97 2.90   0  99.98   0.00   0.00   0.00
 1   1  28  99.67 3.20 2.90   0   0.33
 1   2  11   0.04 2.63 2.90   0  99.96   0.00   0.00   0.00
 1   2  29   0.02 2.11 2.90   0  99.98
...
```

この例では、vCPU 19 および 28 のコアは 3.2 GHz で実行中であり、その他のコアは C1 C ステートで、命令を待機しています。稼働中のコアは Turbo Boost の最大周波数には到達していませんが、非アクティブなコアはより深い C6 C ステートにある場合と比べて、新しいリクエストに迅速に応答します。

変動性が最も低いベースラインパフォーマンス

P ステートによってプロセッサの周波数の変動性を抑制することができます。P ステートはコアに希望するパフォーマンス (CPU 周波数) を制御します。ほとんどのワークロードでは、Turbo Boost をリクエストする、P0 でパフォーマンスが向上します。ただし、Turbo Boost 周波数が有効であるときに発生する可能性があるバースト的なパフォーマンスではなく、安定したパフォーマンスになるようにシステムを調整することもできます。

Intel Advanced Vector Extensions (AVX または AVX2) のワークロードは低い周波数でもパフォーマンスに優れ、AVX 命令はより多くの処理能力を使用できます。Turbo Boost を無効にして、プロセッサをより低い周波数で実行すると、使用される処理能力が抑えられ、より安定した速度が維持されます。AVX のインスタンス設定とワークロードの最適化の詳細については、インテルのウェブサイトを参照してください。

CPU がアイドル状態のドライバは P ステートを制御します。最近の世代の CPU には、以下のようにカーネルレベルでの対応が可能な、更新された CPU アイドルドライバが必要です。

- Linux カーネルバージョン 5.6 以上 (m6i など) – インテル IceLake をサポートします。
- Linux カーネルのバージョン 5.10 以上 (m6a など) – AMD Milan をサポートします。

実行中のシステムのカーネルが CPU を認識するかどうかを確認するには、次のコマンドを実行します。

```
if [ -d /sys/devices/system/cpu/cpu0/cpuidle ]; then echo "C-state control enabled";
 else echo "Kernel cpuidle driver does not recognize this CPU generation"; fi
```

このコマンドの出力により、サポートが不十分であることを確認した場合は、カーネルをアップグレードすることをお勧めします。

このセクションでは、深いスリープ状態を制限し、(P1 P ステートをリクエストすることにより) Turbo Boost を無効にすることで、これらのタイプのワークロードに対して、低レイテンシーを提供し、プロセッサ速度の変動性を最低限に抑える方法を説明します。

Amazon Linux 2 で深いスリープ状態を制限し、Turbo Boost を無効にするには

1. 適切なエディタで `/etc/default/grub` ファイルを開きます。

   ```
   [ec2-user ~]$ sudo vim /etc/default/grub
   ```

2. GRUB_CMDLINE_LINUX_DEFAULT 行を編集し intel_idle.max_cstate=1 と processor.max_cstate=1 のオプションを追加します。これにより、アイドル状態にあるコアの最も深い C ステートとして C1 を設定します。

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8 net.ifnames=0
 biosdevname=0 nvme_core.io_timeout=4294967295 intel_idle.max_cstate=1
 processor.max_cstate=1"
GRUB_TIMEOUT=0
```

   intel_idle.max_cstate=1 オプションでは、インテルベースのインスタンスで C ステート制限が設定され、また、processor.max_cstate=1 オプションでは、AMD ベースのインスタンスで C ステート制限が設定されます。両方のオプションを設定に追加しておくと安心です。これにより、インテルと AMD のインスタンスに対し、共通の方法で目的の動作を設定することができます。

3. ファイルを保存し、エディタを終了します。
4. 起動設定を再構築するには、次のコマンドを実行します。

```
[ec2-user ~]$ grub2-mkconfig -o /boot/grub2/grub.cfg
```

5. 新しい kernel オプションを有効にするためにインスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

6. P1 P ステートによってプロセッサ速度の変動性を抑える必要がある場合は、次のコマンドを実行して Turbo Boost を無効にします。

```
[ec2-user ~]$ sudo sh -c "echo 1 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

7. ワークロードが終了したら、次のコマンドで Turbo Boost を再度有効にすることができます。

```
[ec2-user ~]$ sudo sh -c "echo 0 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

Amazon Linux AMI で深いスリープ状態を制限し、Turbo Boost を無効にするには

1. 適切なエディタで /boot/grub/grub.conf ファイルを開きます。

```
[ec2-user ~]$ sudo vim /boot/grub/grub.conf
```

2. 最初のエントリの kernel 行を編集して intel_idle.max_cstate=1 と processor.max_cstate=1 オプションを追加し、アイドル状態のコアの最も深い C ステートに C1 を設定します。

```
# created by imagebuilder
default=0
timeout=1
hiddenmenu

title Amazon Linux 2014.09 (3.14.26-24.46.amzn1.x86_64)
root (hd0,0)
kernel /boot/vmlinuz-3.14.26-24.46.amzn1.x86_64 root=LABEL=/ console=ttyS0
 intel_idle.max_cstate=1 processor.max_cstate=1
initrd /boot/initramfs-3.14.26-24.46.amzn1.x86_64.img
```

   intel_idle.max_cstate=1 オプションでは、インテルベースのインスタンスで C ステート制限が設定され、また、processor.max_cstate=1 オプションでは、AMD ベースのインスタンスで C ステート制限が設定されます。両方のオプションを設定に追加しておくと安心です。これにより、インテルと AMD のインスタンスに対し、共通の方法で目的の動作を設定することができます。

3. ファイルを保存し、エディタを終了します。

4. 新しい kernel オプションを有効にするためにインスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

5. P1 P ステートによってプロセッサ速度の変動性を抑える必要がある場合は、次のコマンドを実行して Turbo Boost を無効にします。

```
[ec2-user ~]$ sudo sh -c "echo 1 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

6. ワークロードが終了したら、次のコマンドで Turbo Boost を再度有効にすることができます。

```
[ec2-user ~]$ sudo sh -c "echo 0 > /sys/devices/system/cpu/intel_pstate/no_turbo"
```

次の例では、2 つの vCPU が、Turbo Boost を使用せずに、ベースラインコア周波数でアクティブに処理を実行している c4.8xlarge インスタンスを示します。

```
[ec2-user ~]$ sudo turbostat stress -c 2 -t 10
```

```
stress: info: [5389] dispatching hogs: 2 cpu, 0 io, 0 vm, 0 hdd
stress: info: [5389] successful run completed in 10s
pk cor CPU    %c0  GHz  TSC SMI    %c1    %c3    %c6    %c7   %pc2   %pc3   %pc6   %pc7
  Pkg_W RAM_W PKG_% RAM_%
            5.59 2.90 2.90   0  94.41   0.00   0.00   0.00   0.00   0.00   0.00   0.00
 128.48 33.54 200.00  0.00
 0   0   0   0.04 2.90 2.90   0  99.96   0.00   0.00   0.00   0.00   0.00   0.00   0.00
  65.33 19.02 100.00  0.00
 0   0  18   0.04 2.90 2.90   0  99.96
 0   1   1   0.05 2.90 2.90   0  99.95   0.00   0.00   0.00
 0   1  19   0.04 2.90 2.90   0  99.96
 0   2   2   0.04 2.90 2.90   0  99.96   0.00   0.00   0.00
 0   2  20   0.04 2.90 2.90   0  99.96
 0   3   3   0.05 2.90 2.90   0  99.95   0.00   0.00   0.00
 0   3  21  99.95 2.90 2.90   0   0.05
...
 1   1  28  99.92 2.90 2.90   0   0.08
 1   2  11   0.06 2.90 2.90   0  99.94   0.00   0.00   0.00
 1   2  29   0.05 2.90 2.90   0  99.95
```

vCPU 21 および 28 のコアは、ベースラインプロセッサ速度の 2.9 GHz でアクティブに処理を実行し、すべての非アクティブなコアも、C1 C ステートのベースライン速度で実行され、すぐに命令を受け付けることができます。

I/O スケジューラ

I/O スケジューラは、I/O リクエストをソートおよびマージし、処理される順序を決定する Linux オペレーティングシステムの一部です。

I/O スケジューラは、シーク時間が長くなる可能性があり、コロケーションされた要求をマージするのが最適である磁気ハードドライブなどのデバイスにとって特に有益です。I/O スケジューラは、ソリッドステートデバイスや仮想化環境ではあまり効果がありません。これは、ソリッドステートデバイスの場合、シーケンシャルアクセスとランダムアクセスが異なることがなく、仮想化環境ではホストが独自のスケジューリング層を提供するためです。

このトピックでは、Amazon Linux I/O スケジューラについて説明します。他の Linux ディストリビューションで使用される I/O スケジューラの詳細については、それぞれのドキュメントを参照してください。

トピック

- サポートされているスケジューラ

サポートされているスケジューラ

Amazon Linux では、次の I/O スケジューラがサポートされています。

- deadline — 期限日 I/O スケジューラは I/O リクエストをソートし、最も効率的な順序で処理します。これにより、各 I/O リクエストの開始時間が保証されます。また、長い間保留中であった I/O リクエストの優先度も高くなります。
- cfq — 完全公平キュー (CFQ) I/O スケジューラは、プロセス間で I/O リソースを公平に割り当てようとします。I/O リクエストをソートし、プロセスごとのキューに挿入します。
- noop — オペレーションなし (noop) I/O スケジューラは、すべての I/O リクエストを FIFO キューに挿入し、それらを単一のリクエストにマージします。このスケジューラは、リクエストの並べ替えを行いません。

デフォルトスケジューラ

オペレーションなし (noop) は Amazon Linux のデフォルトの I/O スケジューラです。このスケジューラは、次の理由で使用されます。

- 多くのインスタンスタイプは、基盤となるホストがインスタンスのスケジュールを実行する仮想化デバイスを使用します。
- ソリッドステートデバイスは、I/O スケジューラの利点の影響が少ない多くのインスタンスタイプで使用されます。
- これは侵襲性の低い I/O スケジューラであり、必要に応じてカスタマイズできます。

スケジューラを変更する

I/O スケジューラを変更すると、スケジューラによって一定時間内に完了する I/O リクエストが増減するかどうかに基づいて、パフォーマンスが向上または低下することがあります。これは、ワークロード、使用されているインスタンスタイプの世代、アクセスされるデバイスのタイプに大きく依存します。使用する I/O スケジューラを変更する場合は、iotop などのツールを使用し、I/O パフォーマンスを測定し、その変更がユースケースにとって有益かどうかを判断することをお勧めします。

デバイスの I/O スケジューラは、nvme0n1 を例にした次のコマンドで確認できます。インスタンスの /sys/block に表示されているデバイスを使って、次のコマンドの nvme0n1 を置き換えます。

```
$  cat /sys/block/nvme0n1/queue/scheduler
```

デバイスの I/O スケジューラを設定するには、次のコマンドを使用します。

```
$  echo cfq|deadline|noop > /sys/block/nvme0n1/queue/scheduler
```

例えば、*xvda* デバイスの I/O スケジューラを noop から cfq に設定するには、次のコマンドを使用します。

```
$  echo cfq > /sys/block/xvda/queue/scheduler
```

Linux インスタンスの時刻の設定

多くのサーバータスクとプロセスにとって、Linux インスタンスでの整合性のある正確な時刻のリファレンスが不可欠です。システムログのタイムスタンプは、問題が発生した時期やイベントの時系列を特定する上で重要な役割を果たします。AWS CLI または AWS SDK を使用してインスタンスからリクエストを行う際に、これらのツールによって自動的にリクエストに署名されます。インスタンスの日時設定が不正確な場合、署名の日付とリクエストの日付が一致しないことがあり、その場合は AWS がリクエストを却下します。

これに対処することが重要なため、Amazon は Amazon Time Sync Service を提供しています。このサービスはすべての EC2 インスタンスからアクセスでき、さまざまな AWS のサービス で利用されます。このサービスは、各 AWS リージョン で衛星接続された基準となる原子時計のフリートを使用して、世界標準時 (UTC) の正確な現在時刻表示を配信します。

Amazon Time Sync Service は、Network Time Protocol (NTP) を使用するか、サポートされているインスタンスのローカル Precision Time Protocol (PTP) ハードウェアクロックを提供します。PTP ハードウェアクロックでは、NTP または直接 PTP 接続のいずれかがサポートされています。NTP 接続と直接 PTP 接続は非常に正確な同じ時刻を元にしていますが、直接 PTP 接続の方が NTP 接続より正確です。Amazon Time Sync Service への NTP 接続は Leap Smearing (うるう秒の調整) をサポートしていますが、PTP ハードウェアクロックへの PTP 接続は Leap Smearing を行いません。詳細については、「うるう秒」を参照してください。

インスタンスのローカル Amazon Time Sync Service へのバックアップとして、そして Amazon EC2 外部のリソースを Amazon Time Sync Service に接続するために、time.aws.com にあるパブリッ

ク Amazon Time Sync Service を使用できます。パブリック Amazon Time Sync Service は、ローカル Amazon Time Sync Service と同様に UTC に追加されたうるう秒の Leap Smearing を自動的に行います。パブリック Amazon Time Sync Service は、各 AWS リージョン で衛星接続された基準となる原子時計のフリートにより、世界中でサポートされています。

Windows インスタンスの場合は、「Windows インスタンスに時刻を設定する」を参照してください。

トピック

- ローカル Amazon Time Sync Service を使用してインスタンスを設定する
- インスタンスまたはインターネットに接続されたデバイスが、パブリック Amazon Time Sync Service を使用するように設定します。
- タイムスタンプの比較
- インスタンスのタイムゾーンを変更する
- うるう秒
- 関連リソース

ローカル Amazon Time Sync Service を使用してインスタンスを設定する

インスタンスは次のようにローカル Amazon Time Sync Service にアクセスできます。

- 以下の IP アドレスエンドポイントの NTP 経由。
  - IPv4:`169.254.169.123`
  - IPv6: `fd00:ec2::123` (Nitro System 上に構築されたインスタンスからのみアクセス可能)
- 直接 PTP 接続経由でローカル PTP ハードウェアクロックに接続:
  - `PHC0`

NTP 接続と PTP 接続では VPC の設定を変更する必要はなく、インスタンスはインターネットにアクセスする必要もありません。

PTP ハードウェアクロックは Nitro System の一部であるため、サポート対象のベアメタルインスタンスや仮想化 EC2 インスタンスでは、顧客のリソースを使用せずに直接アクセスできます。

PTP ハードウェアクロックへの NTP エンドポイントは、IPv4 または IPv6 での通常の Amazon Time Sync Service 接続と同じです。ソフトウェアが NTP エンドポイントに設定され、PTP ハード

ウェアクロックを備えたインスタンスで実行されている場合、そのソフトウェアは NTP 経由で PTP ハードウェアクロックに自動的に接続されます。

AL2023、および最新バージョンの Amazon Linux 2 と Amazon Linux AMI はデフォルトで Amazon Time Sync Service の IPv4 エンドポイントを使用するように設定されています。IPv6 エンドポイントを使用するか、PTP ハードウェアクロックに直接接続する場合を除いて、これらの AMI から起動するインスタンスにはこれ以上の設定は必要ありません。IPv6 の設定手順については、「Amazon Time Sync Service の IPv6 エンドポイントに接続する」を参照してください。PTP ハードウェアクロックの設定手順については、「PTP ハードウェアクロックに接続する」を参照してください。

Amazon Time Sync Service がデフォルトで設定されていない AMI を使用している場合は、次の手順のいずれかを使用して、chrony クライアントを使用してインスタンスに Amazon Time Sync Service を設定します。

コンテンツ

Amazon Time Sync Service の IPv4 エンドポイントに接続する

このセクションでは、IPv4 エンドポイントを通じてローカル Amazon Time Sync Service を使用するようにインスタンスを設定する方法について説明します。Amazon Time Sync Service のサーバーエントリを chrony 設定ファイルに追加する必要があります。これらの AMI の最新バージョンでは、chrony は既にインストールされ、Amazon Time Sync Service の IPv4 エンドポイントを使用するように設定されています。

インスタンスのオペレーティングシステムの説明を使用してください。

Amazon Linux

> **Note**
>
> AL2023、および最新バージョンの Amazon Linux 2 と Amazon Linux AMI では、chrony は既にインストールされ、Amazon Time Sync Service の IPv4 エンドポイントを使用するように設定されています。これらの最新の AMI のいずれかを使用している場合は、この手順をスキップできます。

chrony を使用して Amazon Linux で Amazon Time Sync Service の IPv4 エンドポイントに接続するには

1. インスタンスに接続し、NTP サービスをアンインストールします。

```
[ec2-user ~]$ sudo yum erase 'ntp*'
```

2. chrony パッケージをインストールします。

```
[ec2-user ~]$ sudo yum install chrony
```

3. 任意のテキストエディタ (例: /etc/chrony.conf または vim など) を使って nano ファイルを開きます。ファイルに次の行が含まれていることを確認します:

```
server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
```

この行が存在する場合は、Amazon Time Sync Service の IPv4 エンドポイントを使用するように Amazon Time Sync Service が既に設定されており、次のステップに進むことができます。そうでない場合は、すでにファイルに存在する他の server または pool ステートメントの後に行を追加し、変更を保存します。

4. chrony デーモン (chronyd) を再起動します。

```
[ec2-user ~]$ sudo service chronyd restart
```

```
Starting chronyd:                                          [  OK  ]
```

> **Note**
>
> RHEL と CentOS (バージョン 6 まで) では、サービス名は chrony ではなく chronyd です。

5. システムがブートするたびに起動するように chronyd を設定するには、chkconfig を使用します。

```
[ec2-user ~]$ sudo chkconfig chronyd on
```

6. chrony が IPv4 エンドポイント 169.254.169.123 を使用して時刻を同期させていることを確認します。

```
[ec2-user ~]$ chronyc sources -v
```

```
210 Number of sources = 7

  .-- Source mode  '^' = server, '=' = peer, '#' = local clock.
 / .- Source state '*' = current synced, '+' = combined , '-' = not
combined,
| /   '?' = unreachable, 'x' = time may be in error, '~' = time too
variable.
||                                                 .- xxxx [ yyyy ] +/-
zzzz
||      Reachability register (octal) -.           |  xxxx = adjusted
offset,
||      Log2(Polling interval) --.      |          |  yyyy = measured
offset,
||                                \     |          |  zzzz = estimated
error.
||                                 |    |           \
MS Name/IP address         Stratum Poll Reach LastRx Last sample


===============================================================================
^* 169.254.169.123               3   6    17    43    -30us[ -226us] +/-
 287us
^- ec2-12-34-231-12.eu-west>     2   6    17    43   -388us[ -388us] +/-
  11ms
^- tshirt.heanet.ie              1   6    17    44   +178us[  +25us] +/-
1959us
^? tbag.heanet.ie                0   6     0     -     +0ns[   +0ns] +/-
   0ns
^? bray.walcz.net                0   6     0     -     +0ns[   +0ns] +/-
   0ns
^? 2a05:d018:c43:e312:ce77:>     0   6     0     -     +0ns[   +0ns] +/-
   0ns
^? 2a05:d018:dab:2701:b70:b>     0   6     0     -     +0ns[   +0ns] +/-
   0ns
```

返される出力では、^* が優先時刻ソースを示します。

7. chrony で報告された時刻同期メトリクスを確認します。

```
[ec2-user ~]$ chronyc tracking
```

```
Reference ID    : A9FEA97B (169.254.169.123)
        Stratum         : 4
        Ref time (UTC)  : Wed Nov 22 13:18:34 2017
        System time     : 0.000000626 seconds slow of NTP time
        Last offset     : +0.002852759 seconds
        RMS offset      : 0.002852759 seconds
        Frequency       : 1.187 ppm fast
        Residual freq   : +0.020 ppm
        Skew            : 24.388 ppm
        Root delay      : 0.000504752 seconds
        Root dispersion : 0.001112565 seconds
        Update interval : 64.4 seconds
        Leap status     : Normal
```

Ubuntu

chrony を使用して Ubuntu で Amazon Time Sync Service の IPv4 エンドポイントに接続するには

1. インスタンスに接続し、apt を使用して chrony パッケージをインストールします。

```
ubuntu:~$ sudo apt install chrony
```

> **Note**
>
> 必要に応じて、sudo apt update を実行してインスタンスを最初に更新します。

2. 任意のテキストエディタ (例: /etc/chrony/chrony.conf または vim など) を使って nano ファイルを開きます。ファイルに既に存在する他の server ステートメントや pool ステートメントの前に次の行を追加し、変更を保存します。

```
server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
```

3. chrony サービスを再起動します。

```
ubuntu:~$ sudo /etc/init.d/chrony restart
```

```
Restarting chrony (via systemctl): chrony.service.
```

4. chrony が IPv4 エンドポイント 169.254.169.123 を使用して時刻を同期させていることを確認します。

```
ubuntu:~$ chronyc sources -v
```

```
210 Number of sources = 7

             .-- Source mode  '^' = server, '=' = peer, '#' = local clock.
            / .- Source state '*' = current synced, '+' = combined , '-' = not
combined,
           | /   '?' = unreachable, 'x' = time may be in error, '~' = time too
variable.
           ||                                                 .- xxxx [ yyyy ]
+/- zzzz
           ||      Reachability register (octal) -.           |  xxxx =
adjusted offset,
           ||      Log2(Polling interval) --.      |          |  yyyy =
measured offset,
           ||                                \     |          |  zzzz =
estimated error.
           ||                                 |    |           \
           MS Name/IP address         Stratum Poll Reach LastRx Last sample

===============================================================================
           ^* 169.254.169.123               3   6    17    12    +15us[  +57us]
+/-  320us
           ^- tbag.heanet.ie                1   6    17    13  -3488us[-3446us]
+/- 1779us
           ^- ec2-12-34-231-12.eu-west-     2   6    17    13   +893us[ +935us]
+/- 7710us
           ^? 2a05:d018:c43:e312:ce77:6     0   6     0   10y     +0ns[   +0ns]
+/-    0ns
           ^? 2a05:d018:d34:9000:d8c6:5     0   6     0   10y     +0ns[   +0ns]
+/-    0ns
           ^? tshirt.heanet.ie              0   6     0   10y     +0ns[   +0ns]
+/-    0ns
           ^? bray.walcz.net                0   6     0   10y     +0ns[   +0ns]
+/-    0ns
```

返される出力のうち、^* から始まる行は、優先時刻ソースが示されます。

5. chrony で報告された時刻同期メトリクスを確認します。

```
ubuntu:~$ chronyc tracking
```

```
Reference ID    : 169.254.169.123 (169.254.169.123)
            Stratum         : 4
            Ref time (UTC)  : Wed Nov 29 07:41:57 2017
            System time     : 0.000000011 seconds slow of NTP time
            Last offset     : +0.000041659 seconds
            RMS offset      : 0.000041659 seconds
            Frequency       : 10.141 ppm slow
            Residual freq   : +7.557 ppm
            Skew            : 2.329 ppm
            Root delay      : 0.000544 seconds
            Root dispersion : 0.000631 seconds
            Update interval : 2.0 seconds
            Leap status     : Normal
```

SUSE Linux

SUSE Linux Enterprise Server 15 以降、chrony は NTP にデフォルトで実装されています。

chrony を使用して SUSE Linux で Amazon Time Sync Service の IPv4 エンドポイントに接続するには

1. 任意のテキスト工ディタ (例: /etc/chrony.conf または vim など) を使って nano ファイルを開きます。
2. ファイルに次の行が含まれていることを確認します。

```
server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
```

この行が存在しない場合は追加します。

3. 他のサーバーまたはプールの行はすべてコメントアウトします。
4. yast を開き、chrony サービスを有効にします。

Amazon Time Sync Service の IPv6 エンドポイントに接続する

このセクションでは、IPv6 エンドポイントを通じてローカル Amazon Time Sync Service を使用するようにインスタンスを設定する場合、Amazon Time Sync Service の IPv4 エンドポイントに接続する で説明した手順と異なる点について説明します。Amazon Time Sync Service の設定プロセス全体について説明しているわけではありません。

IPv6 エンドポイントは、Nitro System 上に構築されたインスタンスでのみアクセス可能です。

> **Note**
>
> chrony.conf ファイルで IPv4 と IPv6 の両方のエンドポイントエントリを一緒に使用することはお勧めしません。IPv4 および IPv6 NTP パケットは、インスタンスの同じローカルサーバーから取得されます。IPv4 と IPv6 の両方のエンドポイントを設定する必要はなく、そうしてもインスタンスの時刻の精度は向上しません。

使用している Linux ディストリビューションに応じて、chrony.conf ファイルを編集する手順に到達すると、IPv4 エンドポイント (169.254.169.123)ではなく、Amazon Time Sync Service の IPv6 エンドポイント (fd00:ec2::123) を使用することになります。

```
server fd00:ec2::123 prefer iburst minpoll 4 maxpoll 4
```

ファイルを保存して chrony が fd00:ec2::123 IPv6 エンドポイントを使用して時刻を同期させていることを次のように確認します。

```
[ec2-user ~]$ chronyc sources -v
```

出力で、fd00:ec2::123 IPv6 エンドポイントが表示されているのを確認したら、設定は完了しています。

PTP ハードウェアクロックに接続する

このセクションでは、直接 PTP 接続を使用する PTP ハードウェアクロックを通じてローカル Amazon Time Sync Service を使用するようにインスタンスを設定する方法について説明します。PTP ハードウェアクロックのサーバーエントリを chrony 設定ファイルに追加する必要があります。

インスタンスに PTP ハードウェアクロックがあり、(IPv4 または IPv6 エンドポイントへの) NTP 接続を設定した場合、インスタンス時間は PTP ハードウェアクロックから自動的に取得されます。以下の手順で PTP 直接接続を設定します。これにより、NTP 接続よりも正確な時刻が得られます。

要件

PTP ハードウェアクロックは、以下の要件が満たされている場合にインスタンスで使用できます。

- サポート対象の AWS リージョン: アジアパシフィック (東京)
- サポート対象のインスタンスファミリー: R7g
- ENA ドライバーバージョン 2.10.0 以降がサポート対象のオペレーティングシステムにインストールされている。サポート対象のオペレーティングシステムの詳細については、GitHub でドライバーの「前提条件」を参照してください。

PTP ハードウェアクロックに接続するには

1. インスタンスに接続し、Elastic Network Adapter (ENA) バージョン 2.10.0 以降の Linux カーネルドライバーをインストールします。インストール手順については、GitHub で「Elastic Network Adapter (ENA) ファミリー用の Linux カーネルドライバー」を参照してください。
2. インスタンスに `/dev/ptp0` デバイスが表示されることを確認します。

   ```
   [ec2-user ~]$ ls /dev/ptp0
   ```

   予想される出力は次のようになります。出力に `/dev/ptp0` が表示されない場合は、ENA ドライバーが正しくインストールされていません。この手順のステップ 1 を確認して、ドライバーをインストールしてください。

   ```
   /dev/ptp0
   ```

3. テキストエディタを使用して `/etc/chrony.conf` を編集し、次の行をファイルの任意の場所に追加します。

   ```
   refclock PHC /dev/ptp0 poll 0 delay 0.000010 prefer
   ```

4. 次のコマンドを使用して chrony を再起動します。

   ```
   [ec2-user ~]$ sudo systemctl restart chronyd
   ```

5. chrony が PTP ハードウェアクロックを使用してこのインスタンスの時刻を同期していることを確認します。

```
[ec2-user ~]$ chronyc sources
```

正常な出力

```
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
#* PHC0                          0   0    377    1   +2ns[ +1ns] +/-   5031ns
```

返される出力で、* は優先される時刻の取得元を示します。PHC0 は PTP ハードウェアクロックに対応します。chrony を再起動した後、アスタリスクが表示されるまで数秒かかる場合があります。

インスタンスまたはインターネットに接続されたデバイスが、パブリック Amazon Time Sync Service を使用するように設定します。

インスタンス、またはローカルコンピュータやオンプレミスサーバーなどのインターネットに接続されたデバイスを、インターネット上の time.aws.com でアクセスできるパブリック Amazon Time Sync Service を使用するように設定できます。ローカル Amazon Time Sync Service のバックアップとして、そして AWS 外部のリソースを Amazon Time Sync Service に接続するために、パブリック Amazon Time Sync Service を使用できます。

インスタンスまたはデバイスのオペレーティングシステムに応じて、次のいずれかの手順を使用して、パブリック Amazon Time Sync Service を使用するようにインスタンスまたはデバイスを設定します。

Linux

chrony または ntpd を使用してパブリック Amazon Time Sync Service を使用するように Linux インスタンスまたはデバイスを設定するには

1. 次のように、テキストエディタを使用して /etc/chrony.conf (chrony を使用する場合) または /etc/ntp.conf (ntpd を使用する場合) を編集します。

a. インスタンスまたはデバイスが、Leap Smearing を行うサーバーと行わないサーバーを混在させようとしないように、ローカル Amazon Time Sync Service への既存の接続以外の `server` で始まる行を削除またはコメントアウトします。

> **Important**
>
> パブリック Amazon Time Sync Service に接続するように EC2 インスタンスを設定する場合は、次の行でローカル Amazon Time Sync Service に接続するようにインスタンスを設定しているので、この行を削除しないでください。ローカル Amazon Time Sync Service の方が、直接的な接続でクロックが正確です。パブリック Amazon Time Sync Service はバックアップとしてのみ使用してください。
>
> ```
> server 169.254.169.123 prefer iburst minpoll 4 maxpoll 4
> ```

b. パブリック Amazon Time Sync Service に接続するには、次の行を追加します。

```
pool time.aws.com iburst
```

2. 以下のコマンドのいずれかを使用してデーモンを再起動します。

- chrony

```
sudo service chronyd force-reload
```

- ntpd

```
sudo service ntp reload
```

macOS

パブリック Amazon Time Sync Service を使用するように macOS インスタンスまたはデバイスを設定するには

1. [システム環境設定] を開きます。
2. [Date & Time] (日付と時刻) を選択し、[Date & Time] (日付と時刻) タブを選択します。

3. 変更するには、ロックアイコンを選択し、プロンプトが表示されたらパスワードを入力します。
4. [Set date and time automatically] (日付と時刻を自動的に設定) に、**`time.aws.com`** と入力します。

Windows

パブリック Amazon Time Sync Service を使用するように Windows インスタンスまたはデバイスを設定するには

1. [Control Panel] (コントロールパネル) を開きます。
2. [Date and Time] (日付と時刻) アイコンを選択します。
3. [Internet Time] (インターネット時刻) タブを選択します。お使いの PC がドメインの一部である場合、このタブは使用できません。その場合は、ドメインコントローラと時刻が同期されます。Amazon Time Sync Service を使用するようにコントローラを設定できます。
4. [Change settings] (設定を変更) を選択します。
5. [Synchronize with an Internet time server] (インターネットタイムサーバーと同期) のチェックボックスを選択します。
6. [Server] (サーバー) の横に、**`time.aws.com`** と入力します。

パブリック Amazon Time Sync Service を使用するように Windows Server インスタンスまたはデバイスを設定するには

- Microsoft の手順に従ってレジストリを更新してください。

タイムスタンプの比較

Amazon Time Sync Service を使用している場合は、Amazon EC2 インスタンスのタイムスタンプと ClockBound を比較して、イベントの実際の時刻を判断できます。ClockBound は EC2 インスタンスのクロック精度を測定し、インスタンスの現在のクロックに関して、特定のタイムスタンプが過去または将来にあるかどうかを確認できます。この情報は、各インスタンスの地理的位置に関係なく、EC2 インスタンス間のイベントとトランザクションの順序と一貫性を判断するのに役立ちます。

ClockBound は、オープンソースのデーモンとライブラリです。インストール手順を含む ClockBound の詳細については、GitHub の「ClockBound」を参照してください。

PTP ハードウェアクロックへの直接 PTP 接続を使用している場合、chrony などのタイムデーモンはクロック誤差範囲を過小評価します。これは、PTP ハードウェアクロックが NTP と異なり誤差範囲の正しい情報を chrony に渡さないためです。その結果、クロック同期デーモンはクロックが UTC に対して正確であると想定しているため、0 という誤差範囲になります。誤差範囲全体を測定するために、Nitro System は PTP ハードウェアクロックの誤差範囲を計算し、ENA ドライバー sysfs ファイルシステムを介して EC2 インスタンスで使用できるようにします。以下のコマンドを使用して、これをナノ秒単位の値として直接読み取ることができます。

```
cat /sys/devices/pci0000:00/0000:00:05.0/phc_error_bound
```

出力は PTP ハードウェアクロックのクロック誤差範囲 (ナノ秒単位) です。

PTP ハードウェアクロックへの直接 PTP 接続を使用するときに、特定の時点における正しいクロック誤差範囲を計算するには、PTP ハードウェアクロックをポーリングする時点の「範囲からのクロックエラーバウンド」chrony または「ClockBound」を追加する必要があります。chrony クロック精度の測定とモニタリングの詳細については、「Amazon Time Sync Service と Amazon CloudWatch を使用して Amazon EC2 インスタンスのクロック精度を管理する — パート 1」を参照してください。

インスタンスのタイムゾーンを変更する

Amazon Linux インスタンスは、デフォルトで UTC (協定世界時) タイムゾーンに設定されています。インスタンスの時刻をローカルのタイムゾーンまたはネットワーク内の別のタイムゾーンに変更できます。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

AL2023 および Amazon Linux 2 インスタンスのタイムゾーンを変更するには

1. システムの現在のタイムゾーン設定を表示します。

   ```
   [ec2-user ~]$ timedatectl
   ```

2. 使用可能なタイムゾーンを一覧表示します。

```
[ec2-user ~]$ timedatectl list-timezones
```

3. 選択したタイムゾーンを設定します。

```
[ec2-user ~]$ sudo timedatectl set-timezone America/Vancouver
```

4. (オプション) timedatectl コマンドをもう一度実行して、現在のタイムゾーンが新しいタイムゾーンに更新されていることを確認します。

```
[ec2-user ~]$ timedatectl
```

Amazon Linux インスタンスのタイムゾーンを変更するには

1. インスタンスで使用する時間帯を特定します。/usr/share/zoneinfo ディレクトリには、タイムゾーンデータファイルの階層が含まれています。その場所でディレクトリ構造を閲覧し、お客様の時間帯のファイルを見つけます。

```
[ec2-user ~]$ ls /usr/share/zoneinfo
Africa      Chile    GB          Indian       Mideast   posixrules  US
America     CST6CDT  GB-Eire     Iran         MST       PRC         UTC
Antarctica  Cuba     GMT         iso3166.tab  MST7MDT   PST8PDT     WET
Arctic      EET      GMT0        Israel       Navajo    right       W-SU
...
```

この場所にある一部のエントリはディレクトリです (America など)。そのディレクトリには、特定の都市の時間帯ファイルが含まれています。インスタンスに使用する都市 (またはお客様の時間帯と同じ都市) を見つけます。

2. 新しいタイムゾーンを適用した /etc/sysconfig/clock ファイルを更新します。この例では、ロサンゼルスのタイムゾーンデータファイル /usr/share/zoneinfo/America/Los_Angeles を使用します。

   a. テキストエディタ (vim や nano など) で /etc/sysconfig/clock ファイルを開きます。エディタのコマンドで sudo を使用する必要があります。/etc/sysconfig/clock は root が所有するためです。

```
[ec2-user ~]$ sudo nano /etc/sysconfig/clock
```

b. ZONE エントリを特定し、タイムゾーンファイルに変更します (パスの /usr/share/zoneinfo セクションは省略します)。例えば、ロサンゼルスの時間帯に変更するには、ZONE エントリを次のように変更します。

```
ZONE="America/Los_Angeles"
```

> **Note**
>
> UTC=true エントリを別の値に変更しないでください。このエントリは、ハードウェアクロックに使用されるため、インスタンスで別のタイムゾーンを設定する場合は調整する必要はありません。

c. ファイルを保存し、テキストエディタを終了します。

3. インスタンスが現地時間情報を参照するとき、タイムゾーンファイルを見つけられるように、/etc/localtime とタイムゾーンファイルの間にシンボリックリンクを作成します。

```
[ec2-user ~]$ sudo ln -sf /usr/share/zoneinfo/America/Los_Angeles /etc/localtime
```

4. システムを再起動し、すべてのサーバーとアプリケーションで新しい時間帯情報を取得します。

```
[ec2-user ~]$ sudo reboot
```

5. (オプション) date コマンドを使用して、現在のタイムゾーンが新しいタイムゾーンに更新されていることを確認します。現在のタイムゾーンが出力に表示されます。以下の例では、現在のタイムゾーンは PDT であり、ロサンゼルスのタイムゾーンを参照しています。

```
[ec2-user ~]$ date
Sun Aug 16 05:45:16 PDT 2020
```

うるう秒

1972 年に導入されたうるう秒は、国際原子時 (TAI) と太陽時 (Ut1) の違いに対応するため、地球の自転の不規則性を考慮して UTC 時刻をときどき 1 秒調整するものです。お客様に代わってうるう秒を管理するために、当社は Amazon Time Sync Service 内での Leap Smearing を設計しました。詳細については、「うるう秒に備える — 迫り来るうるう秒と AWS」を参照してください。

うるう秒はなくなりつつあります。当社は、2035 年までにうるう秒を廃止するという第 27 回国際度量衡総会で採択された決議を全面的に支持しています。

この移行をサポートするために、ローカル NTP 接続または当社のパブリック NTP プール (`time.aws.com`) 経由で Amazon Time Sync Service にアクセスする場合、うるう秒の発生中に Leap Smearing を引き続き計画しています。ただし、PTP ハードウェアクロックには Leap Smearing のオプションはありません。うるう秒が発生した場合、PTP ハードウェアクロックは UTC 標準に従ってうるう秒を追加します。Leap Smearing を行う時刻の供給元とうるう秒を挿入する時刻の供給元は、ほとんどの場合同様です。ただし、うるう秒の発生中は両者が異なるため、うるう秒の発生中は、タイムクライアントの設定で Leap Smearing を行う時刻の供給元と行わない時刻の供給元の両方を使用することはお勧めしません。

関連リソース

- https://chrony-project.org/

CPU オプションの最適化

多くの Amazon EC2 インスタンスは、単一の Intel Xeon CPU コアで同時に複数のスレッドを実行できる同時マルチスレッドをサポートしています。各スレッドは、インスタンスの仮想 CPU (vCPU) として表されます。インスタンスには、インスタンスタイプによって異なるデフォルト数の CPU コアがあります。例えば、`m5.xlarge` インスタンスタイプには 2 つの CPU コアがあり、デフォルトでは各コアごとに 2 つのスレッドの合計で 4 つの vCPU があります。—

> **Note**
>
> 各 vCPU は、T2 インスタンス、M7a インスタンス、Apple シリコン Mac インスタンス、64 ビット ARM プラットフォーム (AWS Graviton プロセッサを搭載したインスタンスなど) を除いては、CPU コアのスレッドです。

ほとんどの場合、ワークロードに適したメモリと vCPU 数を組み合わせた Amazon EC2 インスタンスタイプがあります。ただし、特定のワークロードまたはビジネスのニーズに合わせて、インスタンスを最適化するために以下の CPU オプションを指定できます。

- CPU コア数: インスタンスの CPU コア数をカスタマイズできます。これによって、大量のメモリを使用するワークロード用に十分な RAM 量がありながら、少ない CPU コアのインスタンスのソフトウェアのライセンスコストを最適化することにつながります。

- コア別のスレッド: マルチスレッドを無効化するには、CPU コアごとに 1 つのスレッドを指定できます。高性能コンピューティング (HPC) のワークロードのような特定のワークロードでこれを使用できます。

この CPU オプションはインスタンスの起動時に指定できます。CPU オプションの指定には、追加あるいは割引課金はありません。デフォルト CPU オプションで起動したインスタンスと同じように課金されます。

コンテンツ

CPU オプションを指定するためのルール

インスタンスで CPU オプションを指定するには、次のルールに注意してください。

- ベアメタルインスタンスには CPU オプションを指定できません。
- CPU オプションはインスタンスの起動時のみ指定でき、起動後には変更できません。
- インスタンスを起動するときに、CPU コア数およびコアごとのスレッドの両方をリクエストで指定する必要があります。リクエスト例については、「インスタンスの CPU オプションの指定」を参照してください。
- インスタンスの vCPU の数は、コア別のスレッドで乗算した CPU コアの数です。vCPU のカスタム数を指定するには、インスタンスタイプで CPU およびコア別のスレッドの有効な数を指定する必要があります。インスタンスのデフォルト vCPU の数を超えることはできません。詳細については、インスタンスタイプ別の CPU コアごとの CPU コア数とスレッド数 を参照してください。
- マルチスレッドを無効にするには、コアごとに 1 つのスレッドを指定します。
- 既存のインスタンスのインスタンスタイプを変更する場合、CPU オプションは自動的に新しいインスタンスタイプのデフォルト CPU オプションに変更されます。
- 指定された CPU オプションは、インスタンスの停止、開始あるいは再起動後にも保持されます。

インスタンスタイプ別の CPU コアごとの CPU コア数とスレッド数

次の表では、CPU オプションの指定をサポートしているインスタンスタイプを一覧表示しています。

コンテンツ

汎用インスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m2.xlarge	2	2	1	1、2	1
m2.2xlarge	4	4	1	1、2、3、4	1
m2.4xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m3.large	2	1	2	1	1、2
m3.xlarge	4	2	2	1、2	1、2
m3.2xlarge	8	4	2	1、2、3、4	1、2
m4.large	2	1	2	1	1、2
m4.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m4.2xlarge	8	4	2	1、2、3、4	1、2
m4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m4.10xlarge	40	20	2	2、4、6、8、10、12、14、16、18、20	1、2
m4.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5.large	2	1	2	1	1、2
m5.xlarge	4	2	2	2	1、2
m5.2xlarge	8	4	2	2、4	1、2
m5.4xlarge	16	8	2	2、4、6、8	1、2
m5.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2
m5a.large	2	1	2	1	1、2
m5a.xlarge	4	2	2	2	1、2
m5a.2xlarge	8	4	2	2、4	1、2
m5a.4xlarge	16	8	2	2、4、6、8	1、2
m5a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
m5a.12xlarge	48	24	2	6, 12, 18, 24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5a.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5a.24xlarge	96	48	2	12, 18, 24, 36, 48	1、2
m5ad.large	2	1	2	1	1、2
m5ad.xlarge	4	2	2	2	1、2
m5ad.2xlarge	8	4	2	2、4	1、2
m5ad.4xlarge	16	8	2	2、4、6、8	1、2
m5ad.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
m5ad.12xlarge	48	24	2	6, 12, 18, 24	1、2
m5ad.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5ad.24xlarge	96	48	2	12, 18, 24, 36, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5d.large	2	1	2	1	1、2
m5d.xlarge	4	2	2	2	1、2
m5d.2xlarge	8	4	2	2、4	1、2
m5d.4xlarge	16	8	2	2、4、6、8	1、2
m5d.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5d.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m5d.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5dn.large	2	1	2	1	1、2
m5dn.xlarge	4	2	2	1、2	1、2
m5dn.2xlarge	8	4	2	2、4	1、2
m5dn.4xlarge	16	8	2	2、4、6、8	1、2
m5dn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5dn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5dn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5dn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m5n.large	2	1	2	1	1、2
m5n.xlarge	4	2	2	1、2	1、2
m5n.2xlarge	8	4	2	2、4	1、2
m5n.4xlarge	16	8	2	2、4、6、8	1、2
m5n.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m5n.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m5n.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m5n.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m5zn.large	2	1	2	1	1、2
m5zn.xlarge	4	2	2	1、2	1、2
m5zn.2xlarge	8	4	2	2、4	1、2
m5zn.3xlarge	12	6	2	2、4、6	1、2
m5zn.6xlarge	24	12	2	2、4、6、8、10、12	1、2
m5zn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m6a.large	2	1	2	1	1、2
m6a.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6a.2xlarge	8	4	2	1、2、3、4	1、2
m6a.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m6a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
m6a.12xlarge	48	24	2	1、2、3、4、5、6、7、8、16、24	1、2
m6a.16xlarge	64	32	2	4、6、8、10、12、14、16、32	1、2
m6a.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2
m6a.32xlarge	128	64	2	8、12、16、20、24、28、32、64	1、2
m6a.48xlarge	192	96	2	8、12、16、20、24、28、32、64、96	1、2
m6g.large	2	2	1	1、2	1
m6g.xlarge	4	4	1	1、2、3、4	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m6g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m6g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m6gd.large	2	2	1	1、2	1
m6gd.xlarge	4	4	1	1、2、3、4	1
m6gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m6gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m6gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m6i.large	2	1	2	1	1、2
m6i.xlarge	4	2	2	1、2	1、2
m6i.2xlarge	8	4	2	2、4	1、2
m6i.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
m6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m6id.large	2	1	2	1	1、2
m6id.xlarge	4	2	2	1、2	1、2
m6id.2xlarge	8	4	2	2、4	1、2
m6id.4xlarge	16	8	2	2、4、6、8	1、2
m6id.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
m6id.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6id.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
m6id.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
m6id.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m6idn.large	2	1	2	1	1、2
m6idn.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6idn.2xlarge	8	4	2	1、2、3、4	1、2
m6idn.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m6idn.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m6idn.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m6idn.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6idn.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2
m6idn.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m6in.large	2	1	2	1	1、2
m6in.xlarge	4	2	2	1、2	1、2
m6in.2xlarge	8	4	2	1、2、3、4	1、2
m6in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m6in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m6in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
m6in.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m6in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
m7a.large	2	2	1	1、2	1
m7a.xlarge	4	4	1	1、2、3、4	1
m7a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
m7a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
m7a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
m7a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
m7a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
m7a.32xlarge	128	128	1	4、6、8、10、12、14、16、32、48、64、80、96、112、128	1
m7a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
m7g.large	2	2	1	1、2	1
m7g.xlarge	4	4	1	1、2、3、4	1
m7g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m7g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m7g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m7gd.large	2	2	1	1、2	1
m7gd.xlarge	4	4	1	1、2、3、4	1
m7gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
m7gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
m7gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
m7i.large	2	1	2	1	1、2
m7i.xlarge	4	2	2	1、2	1、2
m7i.2xlarge	8	4	2	1、2、3、4	1、2
m7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
m7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
m7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
m7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
m7i-flex.large	2	1	2	1	1、2
m7i-flex.xlarge	4	2	2	1、2	1、2
m7i-flex.2xlarge	8	4	2	1、2、3、4	1、2
m7i-flex.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
m7i-flex.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
t3.nano	2	1	2	1	1、2
t3.micro	2	1	2	1	1、2
t3.small	2	1	2	1	1、2
t3.medium	2	1	2	1	1、2
t3.large	2	1	2	1	1、2
t3.xlarge	4	2	2	2	1、2
t3.2xlarge	8	4	2	2、4	1、2
t3a.nano	2	1	2	1	1、2
t3a.micro	2	1	2	1	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
t3a.small	2	1	2	1	1、2
t3a.medium	2	1	2	1	1、2
t3a.large	2	1	2	1	1、2
t3a.xlarge	4	2	2	2	1、2
t3a.2xlarge	8	4	2	2、4	1、2
t4g.nano	2	2	1	1、2	1
t4g.micro	2	2	1	1、2	1
t4g.small	2	2	1	1、2	1
t4g.medium	2	2	1	1、2	1
t4g.large	2	2	1	1、2	1
t4g.xlarge	4	4	1	1、2、3、4	1
t4g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

コンピュート最適化インスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c3.large	2	1	2	1	1、2
c3.xlarge	4	2	2	1、2	1、2
c3.2xlarge	8	4	2	1、2、3、4	1、2
c3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
c4.large	2	1	2	1	1、2
c4.xlarge	4	2	2	1、2	1、2
c4.2xlarge	8	4	2	1、2、3、4	1、2
c4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c4.8xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
c5.large	2	1	2	1	1、2
c5.xlarge	4	2	2	2	1、2
c5.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5.4xlarge	16	8	2	2、4、6、8	1、2
c5.9xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
c5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
c5.18xlarge	72	36	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36	1、2
c5.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c5a.large	2	1	2	1	1、2
c5a.xlarge	4	2	2	1、2	1、2
c5a.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5a.4xlarge	16	8	2	1、2、3、4、8	1、2
c5a.8xlarge	32	16	2	1、2、3、4、8、12、16	1、2
c5a.12xlarge	48	24	2	1、2、3、4、8、12、16、20、24	1、2
c5a.16xlarge	64	32	2	1、2、3、4、8、12、16、20、24、28、32	1、2
c5a.24xlarge	96	48	2	1、2、3、4、8、12、16、20、24、28、32、36、40、44、48	1、2
c5ad.large	2	1	2	1	1、2
c5ad.xlarge	4	2	2	1、2	1、2
c5ad.2xlarge	8	4	2	1、2、3、4	1、2
c5ad.4xlarge	16	8	2	1、2、3、4、8	1、2
c5ad.8xlarge	32	16	2	1、2、3、4、8、12、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5ad.12xlarge	48	24	2	1、2、3、4、8、12、16、20、24	1、2
c5ad.16xlarge	64	32	2	1、2、3、4、8、12、16、20、24、28、32	1、2
c5ad.24xlarge	96	48	2	1、2、3、4、8、12、16、20、24、28、32、36、40、44、48	1、2
c5d.large	2	1	2	1	1、2
c5d.xlarge	4	2	2	2	1、2
c5d.2xlarge	8	4	2	2、4	1、2
c5d.4xlarge	16	8	2	2、4、6、8	1、2
c5d.9xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
c5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5d.18xlarge	72	36	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36	1、2
c5d.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c5n.large	2	1	2	1	1、2
c5n.xlarge	4	2	2	2	1、2
c5n.2xlarge	8	4	2	2、4	1、2
c5n.4xlarge	16	8	2	2、4、6、8	1、2
c5n.9xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c5n.18xlarge	72	36	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36	1、2
c6a.large	2	1	2	1	1、2
c6a.xlarge	4	2	2	1、2	1、2
c6a.2xlarge	8	4	2	1、2、3、4	1、2
c6a.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c6a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
c6a.12xlarge	48	24	2	1、2、3、4、5、6、7、8、16、24	1、2
c6a.16xlarge	64	32	2	4、6、8、10、12、14、16、32	1、2
c6a.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2
c6a.32xlarge	128	64	2	8、12、16、20、24、28、32、64	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6a.48xlarge	192	96	2	8、12、16、20、24、28、32、64、96	1、2
c6g.large	2	2	1	1、2	1
c6g.xlarge	4	4	1	1、2、3、4	1
c6g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c6g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c6g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c6gd.large	2	2	1	1、2	1
c6gd.xlarge	4	4	1	1、2、3、4	1
c6gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c6gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c6gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c6gn.medium	1	1	1	1	1
c6gn.large	2	2	1	1、2	1
c6gn.xlarge	4	4	1	1、2、3、4	1
c6gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c6gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c6gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c6i.large	2	1	2	1	1、2
c6i.xlarge	4	2	2	1、2	1、2
c6i.2xlarge	8	4	2	2、4	1、2
c6i.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
c6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
c6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
c6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c6id.large	2	1	2	1	1、2
c6id.xlarge	4	2	2	1、2	1、2
c6id.2xlarge	8	4	2	2、4	1、2
c6id.4xlarge	16	8	2	2、4、6、8	1、2
c6id.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
c6id.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6id.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
c6id.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
c6id.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c6in.large	2	1	2	1	1、2
c6in.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6in.2xlarge	8	4	2	1、2、3、4	1、2
c6in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
c6in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c6in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c6in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c6in.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2
c6in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
c7a.large	2	2	1	1、2	1
c7a.xlarge	4	4	1	1、2、3、4	1
c7a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c7a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
c7a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1
c7a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
c7a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
c7a.32xlarge	128	128	1	4、6、8、10、12、14、16、32、48、64、80、96、112、128	1
c7a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
c7g.large	2	2	1	1、2	1
c7g.xlarge	4	4	1	1、2、3、4	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
c7g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c7g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c7gd.large	2	2	1	1、2	1
c7gd.xlarge	4	4	1	1、2、3、4	1
c7gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c7gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c7gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c7gn.large	2	2	1	1、2	1
c7gn.xlarge	4	4	1	1、2、3、4	1
c7gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
c7gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
c7gn.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
c7i.large	2	1	2	1	1、2
c7i.xlarge	4	2	2	1、2	1、2
c7i.2xlarge	8	4	2	1、2、3、4	1、2
c7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
c7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
c7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
c7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
c7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

メモリ最適化インスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r3.large	2	1	2	1	1、2
r3.xlarge	4	2	2	1、2	1、2
r3.2xlarge	8	4	2	1、2、3、4	1、2
r3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r4.large	2	1	2	1	1、2
r4.xlarge	4	2	2	1、2	1、2
r4.2xlarge	8	4	2	1、2、3、4	1、2
r4.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r4.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r4.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
				2、24、26、28、30、32	
r5.large	2	1	2	1	1、2
r5.xlarge	4	2	2	2	1、2
r5.2xlarge	8	4	2	2、4	1、2
r5.4xlarge	16	8	2	2、4、6、8	1、2
r5.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r5.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5a.large	2	1	2	1	1、2
r5a.xlarge	4	2	2	2	1、2
r5a.2xlarge	8	4	2	2、4	1、2
r5a.4xlarge	16	8	2	2、4、6、8	1、2
r5a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
r5a.12xlarge	48	24	2	6, 12, 18, 24	1、2
r5a.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5a.24xlarge	96	48	2	12, 18, 24, 36, 48	1、2
r5ad.large	2	1	2	1	1、2
r5ad.xlarge	4	2	2	2	1、2
r5ad.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5ad.4xlarge	16	8	2	2、4、6、8	1、2
r5ad.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
r5ad.12xlarge	48	24	2	6, 12, 18, 24	1、2
r5ad.16xlarge	64	32	2	8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5ad.24xlarge	96	48	2	12, 18, 24, 36, 48	1、2
r5b.large	2	1	2	1	1、2
r5b.xlarge	4	2	2	1、2	1、2
r5b.2xlarge	8	4	2	2、4	1、2
r5b.4xlarge	16	8	2	2、4、6、8	1、2
r5b.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5b.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5b.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5b.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r5d.large	2	1	2	1	1、2
r5d.xlarge	4	2	2	2	1、2
r5d.2xlarge	8	4	2	2、4	1、2
r5d.4xlarge	16	8	2	2、4、6、8	1、2
r5d.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5d.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5d.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5d.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2
r5dn.large	2	1	2	1	1、2
r5dn.xlarge	4	2	2	1、2	1、2
r5dn.2xlarge	8	4	2	2、4	1、2
r5dn.4xlarge	16	8	2	2、4、6、8	1、2
r5dn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5dn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5dn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5dn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r5n.large	2	1	2	1	1、2
r5n.xlarge	4	2	2	1、2	1、2
r5n.2xlarge	8	4	2	2、4	1、2
r5n.4xlarge	16	8	2	2、4、6、8	1、2
r5n.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r5n.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r5n.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r5n.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r6a.large	2	1	2	1	1、2
r6a.xlarge	4	2	2	1、2	1、2
r6a.2xlarge	8	4	2	1、2、3、4	1、2
r6a.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r6a.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6a.12xlarge	48	24	2	1、2、3、4、5、6、7、8、16、24	1、2
r6a.16xlarge	64	32	2	4、6、8、10、12、14、16、32	1、2
r6a.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2
r6a.32xlarge	128	64	2	8、12、16、20、24、28、32、64	1、2
r6a.48xlarge	192	96	2	8、12、16、20、24、28、32、64、96	1、2
r6g.large	2	2	1	1、2	1
r6g.xlarge	4	4	1	1、2、3、4	1
r6g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r6g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
r6g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r6gd.large	2	2	1	1、2	1
r6gd.xlarge	4	4	1	1、2、3、4	1
r6gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r6gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
r6gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r6i.large	2	1	2	1	1、2
r6i.xlarge	4	2	2	1、2	1、2
r6i.2xlarge	8	4	2	2、4	1、2
r6i.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6i.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r6i.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
r6i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r6i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r6idn.large	2	1	2	1	1、2
r6idn.xlarge	4	2	2	1、2	1、2
r6idn.2xlarge	8	4	2	1、2、3、4	1、2
r6idn.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r6idn.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6idn.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r6idn.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r6idn.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6idn.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r6in.large	2	1	2	1	1、2
r6in.xlarge	4	2	2	1、2	1、2
r6in.2xlarge	8	4	2	1、2、3、4	1、2
r6in.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r6in.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6in.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r6in.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r6in.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6in.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r6id.large	2	1	2	1	1、2
r6id.xlarge	4	2	2	1、2	1、2
r6id.2xlarge	8	4	2	2、4	1、2
r6id.4xlarge	16	8	2	2、4、6、8	1、2
r6id.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
r6id.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r6id.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
r6id.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
r6id.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
r7a.large	2	2	1	1、2	1
r7a.xlarge	4	4	1	1、2、3、4	1
r7a.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7a.4xlarge	16	16	1	1、2、4、6、8、10、12、14、16	1
r7a.8xlarge	32	32	1	1、2、3、4、8、12、16、20、24、28、32	1
r7a.12xlarge	48	48	1	1、2、3、4、5、6、12、18、24、30、36、42、48	1
r7a.16xlarge	64	64	1	1、2、3、4、5、6、7、8、16、24、32、40、48、56、64	1
r7a.24xlarge	96	96	1	1、2、3、4、5、6、7、8、9、10、11、12、24、36、48、60、72、84、96	1
r7a.32xlarge	128	128	1	4、6、8、10、12、14、16、32、48、64、80、96、112、128	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7a.48xlarge	192	192	1	4、6、8、10、12、14、16、18、20、22、24、48、72、96、120、144、168、192	1
r7g.large	2	2	1	1、2	1
r7g.xlarge	4	4	1	1、2、3、4	1
r7g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
r7g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r7g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7g.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r7gd.large	2	2	1	1、2	1
r7gd.xlarge	4	4	1	1、2、3、4	1
r7gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
r7gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
r7gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
r7i.large	2	1	2	1	1、2
r7i.xlarge	4	2	2	1、2	1、2
r7i.2xlarge	8	4	2	1、2、3、4	1、2
r7i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r7i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
r7i.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7i.24xlarge	96	48	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1、2
r7i.48xlarge	192	96	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64、66、68、70、72、74、76、78、80、82、84、86、88、90、92、94、96	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7iz.large	2	1	2	1	1、2
r7iz.xlarge	4	2	2	1、2	1、2
r7iz.2xlarge	8	4	2	1、2、3、4	1、2
r7iz.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
r7iz.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
r7iz.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
r7iz.16xlarge	64	32	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1、2
r7iz.32xlarge	128	64	2	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
u-3tb1.56xlarge	224	112	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64、68、72、76、80、84、88、92、96、100、104、108、112	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u-6tb1.56xlarge	224	224	1	8、16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1
u-6tb1.112xlarge	448	224	2	8、16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u-9tb1.112xlarge	448	224	2	8、16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2
u-12tb1.112xlarge	448	224	2	8、16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
u-18tb1.112xlarge	448	224	2	8、16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2
u-24tb1.112xlarge	448	224	2	8、16、24、32、40、48、56、64、72、80、88、96、104、112、120、128、136、144、152、160、168、176、184、192、200、208、216、224	1、2
x1.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x1.32xlarge	128	64	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64	1、2
x2gd.large	2	2	1	1、2	1
x2gd.xlarge	4	4	1	1、2、3、4	1
x2gd.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
x2gd.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
x2gd.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2gd.12xlarge	48	48	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2gd.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
x2idn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2idn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
x2idn.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
x2iedn.xlarge	4	2	2	1、2	1、2
x2iedn.2xlarge	8	4	2	2、4	1、2
x2iedn.4xlarge	16	8	2	2、4、6、8	1、2
x2iedn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2iedn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
x2iedn.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
x2iedn.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
x2iezn.2xlarge	8	4	2	2、4	1、2
x2iezn.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x2iezn.6xlarge	24	12	2	2、4、6、8、10、12	1、2
x2iezn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
x2iezn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
x1e.xlarge	4	2	2	1、2	1、2
x1e.2xlarge	8	4	2	1、2、3、4	1、2
x1e.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
x1e.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
x1e.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
x1e.32xlarge	128	64	2	4、8、12、16、20、24、28、32、36、40、44、48、52、56、60、64	1、2
z1d.large	2	1	2	1	1、2
z1d.xlarge	4	2	2	1、2	1、2
z1d.2xlarge	8	4	2	2、4	1、2
z1d.3xlarge	12	6	2	2、4、6	1、2
z1d.6xlarge	24	12	2	2、4、6、8、10、12	1、2
z1d.12xlarge	48	24	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24	1、2

ストレージ最適化インスタンス

インスタンスタイプ	デフォルトvCPU	デフォルトのCPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
d2.xlarge	4	2	2	1、2	1、2
d2.2xlarge	8	4	2	1、2、3、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
d2.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
d2.8xlarge	36	18	2	2、4、6、8、10、12、14、16、18	1、2
d3.xlarge	4	2	2	1、2	1、2
d3.2xlarge	8	4	2	2、4	1、2
d3.4xlarge	16	8	2	2、4、6、8	1、2
d3.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
d3en.xlarge	4	2	2	1、2	1、2
d3en.2xlarge	8	4	2	2、4	1、2
d3en.4xlarge	16	8	2	2、4、6、8	1、2
d3en.6xlarge	24	12	2	2、4、6、8、10、12	1、2
d3en.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
d3en.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
h1.2xlarge	8	4	2	1、2、3、4	1、2
h1.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
h1.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
h1.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
i2.xlarge	4	2	2	1、2	1、2
i2.2xlarge	8	4	2	1、2、3、4	1、2
i2.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
i2.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i3.large	2	1	2	1	1、2
i3.xlarge	4	2	2	1、2	1、2
i3.2xlarge	8	4	2	1、2、3、4	1、2
i3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
i3.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
i3.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
i3en.large	2	1	2	1	1、2
i3en.xlarge	4	2	2	1、2	1、2
i3en.2xlarge	8	4	2	2、4	1、2
i3en.3xlarge	12	6	2	2、4、6	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i3en.6xlarge	24	12	2	2、4、6、8、10、12	1、2
i3en.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
i3en.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
i4g.large	2	2	1	1、2	1
i4g.xlarge	4	4	1	1、2、3、4	1
i4g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
i4g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i4g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
i4g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i4i.large	2	1	2	1	1、2
i4i.xlarge	4	2	2	1、2	1、2
i4i.2xlarge	8	4	2	1、2、3、4	1、2
i4i.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
i4i.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
i4i.12xlarge	48	24	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24	1、2
i4i.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
i4i.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
i4i.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
im4gn.large	2	2	1	1、2	1
im4gn.xlarge	4	4	1	1、2、3、4	1
im4gn.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
im4gn.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
im4gn.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
im4gn.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1
is4gen.medium	1	1	1	1	1
is4gen.large	2	2	1	1、2	1
is4gen.xlarge	4	4	1	1、2、3、4	1
is4gen.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
is4gen.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1
is4gen.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1

高速コンピューティングインスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
dl1.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
dl2q.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
f1.2xlarge	8	4	2	1、2、3、4	1、2
f1.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2
f1.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
g2.2xlarge	8	4	2	1、2、3、4	1、2
g2.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
g3.4xlarge	16	8	2	1、2、3、4、5、6、7、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g3.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
g3.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
g4ad.xlarge	4	2	2	2	1、2
g4ad.2xlarge	8	4	2	2、4	1、2
g4ad.4xlarge	16	8	2	2、4、8	1、2
g4ad.8xlarge	32	16	2	2、4、8、16	1、2
g4ad.16xlarge	64	32	2	2、4、8、16、32	1、2
g4dn.xlarge	4	2	2	2	1、2
g4dn.2xlarge	8	4	2	2、4	1、2
g4dn.4xlarge	16	8	2	2、4、6、8	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g4dn.8xlarge	32	16	2	2、4、6、8、10、12、14、16	1、2
g4dn.12xlarge	48	24	2	2、4、6、8、10、12、14、16、18、20、22、24	1、2
g4dn.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
g5g.xlarge	4	4	1	1、2、3、4	1
g5g.2xlarge	8	8	1	1、2、3、4、5、6、7、8	1
g5g.4xlarge	16	16	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
g5g.8xlarge	32	32	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32	1
g5g.16xlarge	64	64	1	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16、17、18、19、20、21、22、23、24、25、26、27、28、29、30、31、32、33、34、35、36、37、38、39、40、41、42、43、44、45、46、47、48、49、50、51、52、53、54、55、56、57、58、59、60、61、62、63、64	1

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
inf1.xlarge	4	2	2	2	1、2
inf1.2xlarge	8	4	2	2、4	1、2
inf1.6xlarge	24	12	2	2、4、6、8、10、12	1、2
inf1.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
inf2.xlarge	4	2	2	1、2	1、2
inf2.8xlarge	32	16	2	4、6、8、10、12、14、16	1、2
inf2.24xlarge	96	48	2	4、6、8、10、12、14、16、32、48	1、2
inf2.48xlarge	192	96	2	4、8、12、16、20、24、28、32、64、96	1、2
p2.xlarge	4	2	2	1、2	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p2.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
p2.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
p3.2xlarge	8	4	2	1、2、3、4	1、2
p3.8xlarge	32	16	2	1、2、3、4、5、6、7、8、9、10、11、12、13、14、15、16	1、2
p3.16xlarge	64	32	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32	1、2
p3dn.24xlarge	96	48	2	4, 6, 8, 10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42, 44, 46, 48	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
p4d.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
p4de.24xlarge	96	48	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48	1、2
p5.48xlarge	192	96	2	12、24、36、48、60、72、84、96	1、2
trn1.2xlarge	8	4	2	2、4	1、2

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
trn1.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
trn1n.32xlarge	128	64	2	2、4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1、2
vt1.3xlarge	12	6	2	6	1、2
vt1.6xlarge	24	12	2	6、12	1、2
vt1.24xlarge	96	48	2	6、12、48	1、2

ハイパフォーマンスコンピューティングインスタンス

インスタンスタイプ	デフォルト vCPU	デフォルトの CPU コア	コアごとのデフォルトのスレッド	有効な CPU コア	コアあたりの有効なスレッド
hpc6id.32xlarge	64	64	1	4、6、8、10、12、14、16、18、20、22、24、26、28、30、32、34、36、38、40、42、44、46、48、50、52、54、56、58、60、62、64	1

インスタンスの CPU オプションの指定

インスタンスの起動時に CPU オプションを指定できます。

次の例は、EC2 コンソールのインスタンス起動ウィザードと run-instances AWS CLI コマンド、および EC2 コンソールの起動設定テンプレートページと create-launch-template AWS CLI コマンドを使用する際に CPU オプションを指定する方法を示しています。EC2 フリートまたはスポットフリートの場合は、起動テンプレートで CPU オプションを指定する必要があります。

以下の例は、次の デフォルト値がある r5.4xlarge インスタンスタイプの場合です。

- デフォルトの CPU コア: 8
- コアごとのデフォルトのスレッド: 2
- デフォルト vCPU: 16 (8 x 2)
- CPU コアの有効数: 2, 4, 6, 8
- コアごとのスレッドの有効数: 1、2

マルチスレッドの無効化

マルチスレッドを無効にするには、コアごとに 1 つのスレッドを指定します。

New console

インスタンス起動時にマルチスレッドを無効にするには

1. インスタンスをすばやく起動する の手順に従い、必要に応じてインスタンスを設定します。
2. [詳細設定] を展開し、[CPU オプションの指定] チェックボックスをオンにします。
3. [Core count (コア数)] では、必要な CPU コア数を選択します。この例では、`r5.4xlarge` インスタンスにデフォルトの CPU コア数を指定するには、8 を選択します。
4. マルチスレッドを無効にするには、[Threads per core (コアごとのスレッド)] で、[1] を選択します。
5. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

Old console

インスタンス起動時にマルチスレッドを無効にするには

1. 「」の手順に従います。古いインスタンス起動ウィザードを使用してインスタンスを起動する
2. [CPU オプション] の [インスタンスの詳細設定] ページで、[CPU オプションを指定] を指定します。
3. [Core count (コア数)] では、必要な CPU コア数を選択します。この例では、`r5.4xlarge` インスタンスにデフォルトの CPU コア数を指定するには、8 を選択します。
4. マルチスレッドを無効にするには、[Threads per core (コアごとのスレッド)] で、[1] を選択します。
5. ウィザードに従って続行します。[Review Instance Launch] (インスタンス作成の確認) ページでオプションの確認が終了したら、[Launch] (起動) を選択します。詳細については、「古いインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI

インスタンス起動時にマルチスレッドを無効にするには

run-instances AWS CLI コマンドを使用して、1 パラメータの `ThreadsPerCore` の `--cpu-options` の値を指定します。[`CoreCount`] では、CPU コア数を指定します。この例で

は、r5.4xlarge インスタンスにデフォルトの CPU コア数を指定するには、8 の値を選択します。

```
aws ec2 run-instances \
    --image-id ami-1a2b3c4d \
    --instance-type r5.4xlarge \
    --cpu-options "CoreCount=8,ThreadsPerCore=1" \
    --key-name MyKeyPair
```

起動時の vCPU のカスタム数の指定

インスタンスの CPU コア数とコアあたりのスレッドの数をカスタマイズできます。

次の例では、4 つの vCPU で r5.4xlarge インスタンスを起動します。

New console

インスタンス起動中に vCPU のカスタム数を指定するには

1. インスタンスをすばやく起動する の手順に従い、必要に応じてインスタンスを設定します。
2. [詳細設定] を展開し、[CPU オプションの指定] チェックボックスをオンにします。
3. 4 つの vCPU を取得するには、次のように、2 つの CPU コアおよびコアごとに 2 つのスレッドを指定します。
   - [コアカウント] には 2 を選択します。
   - [Threads per core (コアごとのスレッド)] には、[2] を選択します。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

Old console

インスタンス起動中に vCPU のカスタム数を指定するには

1. 「」の手順に従います。古いインスタンス起動ウィザードを使用してインスタンスを起動する
2. [CPU オプション] の [インスタンスの詳細設定] ページで、[CPU オプションを指定] を指定します。

3. 4 つの vCPU を取得するには、次のように、2 つの CPU コアおよびコアごとに 2 つのスレッドを指定します。

   - [コアカウント] には 2 を選択します。
   - [Threads per core (コアごとのスレッド)] には、[2] を選択します。

4. ウィザードに従って続行します。[Review Instance Launch] (インスタンス作成の確認) ページでオプションの確認が終了したら、[Launch] (起動) を選択します。詳細については、「古いインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI

インスタンス起動中に vCPU のカスタム数を指定するには

run-instances AWS CLI コマンドを使用して、`--cpu-options` パラメータの CPU コア数およびスレッドの数を指定します。2 つの CPU コアおよびコアごとに 2 つのスレッドを指定すると、4 つの vCPU を取得できます。

```
aws ec2 run-instances \
    --image-id ami-1a2b3c4d \
    --instance-type r5.4xlarge \
    --cpu-options "CoreCount=2,ThreadsPerCore=2" \
    --key-name MyKeyPair
```

また、4 つの CPU コアおよびコアごとに 1 つのスレッドを指定 (マルチスレッドを無効化) して、4 つの vCPU を取得することもできます。

```
aws ec2 run-instances \
    --image-id ami-1a2b3c4d \
    --instance-type r5.4xlarge \
    --cpu-options "CoreCount=4,ThreadsPerCore=1" \
    --key-name MyKeyPair
```

起動テンプレートでの vCPU のカスタム数の指定

起動テンプレートでインスタンスの CPU コア数とコアごとのスレッドの数をカスタマイズできます。

次の例では、vCPU が 4 つの `r5.4xlarge` インスタンスの設定を指定する起動テンプレートを作成します。

Console

起動テンプレートで vCPU のカスタム数を指定するには

1. 「定義したパラメータを使用した新しい起動テンプレートの作成」の手順に従い、必要に応じて起動テンプレートを設定します。
2. [詳細設定] を展開し、[CPU オプションの指定] チェックボックスをオンにします。
3. 4 つの vCPU を取得するには、次のように、2 つの CPU コアおよびコアごとに 2 つのスレッドを指定します。
   - [コアカウント] には 2 を選択します。
   - [Threads per core (コアごとのスレッド)] には、[2] を選択します。
4. [概要] パネルでインスタンスの設定を確認し、[起動テンプレートの作成] を選択します。詳細については、「起動テンプレートからのインスタンスの起動」を参照してください。

AWS CLI

起動テンプレートで vCPU のカスタム数を指定するには

create-launch-template AWS CLI コマンドを使用して、CpuOptions パラメータで CPU コア数およびスレッドの数を指定します。2 つの CPU コアおよびコアごとに 2 つのスレッドを指定すると、4 つの vCPU を取得できます。

```
aws ec2 create-launch-template \
    --launch-template-name TemplateForCPUOptions \
    --version-description CPUOptionsVersion1 \
    --launch-template-data file://template-data.json
```

インスタンスを設定するための起動テンプレートデータ (CPU オプションを含む) を含む、JSON ファイルの例を以下に示します。

```
{
    "NetworkInterfaces": [{
        "AssociatePublicIpAddress": true,
        "DeviceIndex": 0,
        "Ipv6AddressCount": 1,
        "SubnetId": "subnet-7b16de0c"
    }],
    "ImageId": "ami-8c1be5f6",
    "InstanceType": "r5.4xlarge",
```

```
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }],
    "CpuOptions": {
        "CoreCount":2,
        "ThreadsPerCore":2
    }
}
```

また、4 つの CPU コアおよびコアごとに 1 つのスレッドを指定 (マルチスレッドを無効化) して、4 つの vCPU を取得することもできます。

```
{
    "NetworkInterfaces": [{
        "AssociatePublicIpAddress": true,
        "DeviceIndex": 0,
        "Ipv6AddressCount": 1,
        "SubnetId": "subnet-7b16de0c"
    }],
    "ImageId": "ami-8c1be5f6",
    "InstanceType": "r5.4xlarge",
    "TagSpecifications": [{
        "ResourceType": "instance",
        "Tags": [{
            "Key":"Name",
            "Value":"webserver"
        }]
    }],
    "CpuOptions": {
        "CoreCount":4,
        "ThreadsPerCore":1
    }
}
```

インスタンスの CPU オプションの表示

既存のインスタンスの CPU オプションを表示するには、Amazon EC2 コンソールを使用するか、または AWS CLI を使用してインスタンスを表示します。

Console

コンソールを使用してインスタンスの CPU オプションを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. [Details (詳細)] タブの [Host and placement group (ホストとプレイスメントグループ)] で、[Number of vCPUs (vCPU の数)] を見つけます。

AWS CLI

インスタンスの CPU オプションを表示するには (AWS CLI)

describe-instances コマンドを使用します。

```
aws ec2 describe-instances --instance-ids i-123456789abcde123
```

```
...
    "Instances": [
        {
            "Monitoring": {
                "State": "disabled"
            },
            "PublicDnsName": "ec2-198-51-100-5.eu-central-1.compute.amazonaws.com",
            "State": {
                "Code": 16,
                "Name": "running"
            },
            "EbsOptimized": false,
            "LaunchTime": "2018-05-08T13:40:33.000Z",
            "PublicIpAddress": "198.51.100.5",
            "PrivateIpAddress": "172.31.2.206",
            "ProductCodes": [],
            "VpcId": "vpc-1a2b3c4d",
            "CpuOptions": {
                "CoreCount": 34,
                "ThreadsPerCore": 1
            },
            "StateTransitionReason": "",
            ...
        }
```

```
    ]
...
```

返される出力の CoreCount フィールドは、そのインスタンスのコア数を示しています。ThreadsPerCore フィールドは、コア別のスレッド数を示します。

また、インスタンスを接続し、のツール (lscpu など) を使用して、インスタンスの CPU 情報を表示することもできます。

インスタンスの終了を含む、インスタンスにおける設定変更の記録、判断、監査、評価のために、AWS Config を使用できます。詳細については、「AWS Config デベロッパーガイド」の「AWS Configの使用開始」を参照してください。

CPU の機能

Amazon EC2 インスタンスは以下の CPU 機能をサポートしています。サポートはインスタンスのタイプによって異なります。サポートされるインスタンスタイプの詳細については、各 CPU 機能のドキュメントを参照してください。

トピック

- AMD SEV-SNP

AMD SEV-SNP

AMD Secure Encrypted Virtualization-Secure Nested Paging (AMD SEV-SNP) は、以下の特性を持つ CPU 機能です。

- 認証 — AMD SEV-SNP を使用すると、インスタンスの状態と ID の検証に使用できる暗号化手段を含む署名済みの認証レポートを取得できます。また、正規の AMD ハードウェアで実行されていることも確認できます。詳細については、「AMD SEV-SNP による認証」を参照してください。
- メモリの暗号化 — AMD EPYC (Milan)、AWS Graviton2、Intel Xeon Scalable (Ice Lake) プロセッサ以降の CPU では、インスタンスメモリは常に暗号化されます。AMD SEV-SNP が有効になっているインスタンスは、メモリ暗号化にインスタンス固有のキーを使用します。

料金

AMD SEV-SNP を有効にして Amazon EC2 インスタンスを起動すると、選択したインスタンスタイプのオンデマンド時間料金の 10 パーセントに相当する追加の時間単位使用料が請求されます。

この AMD SEV-SNP 使用料は、Amazon EC2 インスタンスの使用料とは別に請求されます。リザーブドインスタンス、Savings Plans、およびオペレーティングシステムの使用量はこの料金に影響しません。

AMD SEV-SNP を有効にしてスポットインスタンスを起動するように設定すると、選択したインスタンスタイプのオンデマンド時間料金の 10% に相当する追加の時間単位使用料が請求されます。配分戦略で価格を入力として使用する場合、スポットフリートにはこの追加料金は含まれず、スポット料金のみが使用されます。

要件

AMD SEV-SNP を使用するには、以下の操作を行う必要があります。

- 以下のサポートされているインスタンスタイプのいずれかを使用します。
  - 汎用:: `m6a.large` | `m6a.xlarge` | `m6a.2xlarge` | `m6a.4xlarge` | `m6a.8xlarge`
  - コンピューティングの最適化:: `c6a.large` | `c6a.xlarge` | `c6a.2xlarge` | `c6a.4xlarge` | `c6a.8xlarge` | `c6a.12xlarge` | `c6a.16xlarge`
  - メモリの最適化:: `r6a.large` | `r6a.xlarge` | `r6a.2xlarge` | `r6a.4xlarge`
- サポートされている AWS リージョン でインスタンスを起動します。現在、米国東部 (オハイオ) と欧州 (アイルランド) のみがサポートされています。
- AMI は、`uefi` または `uefi-preferred` ブートモードおよび。AMD SEV-SNP をサポートするオペレーティングシステムで使用してください。ご使用のオペレーティングシステムでの AMD SEV-SNP サポートの詳細については、それぞれのオペレーティングシステムのマニュアルを参照してください。AWS については、AMD SEV-SNP は AL2023、RHEL 9.3、SLES 15 SP4、および Ubuntu 23.04 以降でサポートされています。

考慮事項

AMD SEV-SNP を使用する際には、次の点に注意してください。

- AMD SEV-SNP は、インスタンスの起動時にのみ有効になります。起動時に AMD SEV-SNP がオンになっても、インスタンスのライフサイクルを通して有効のままになります。
- AMD SEV-SNP がオンになっているインスタンスのインスタンスタイプは、AMD SEV-SNP をサポートする別のインスタンスタイプにのみ変更できます。
- AMD SEV-SNP がオンになっている場合、ハイバネーションと Nitro Enclave はサポートされません。
- 専用ホストはサポートされていません。

- インスタンスの基盤となるホストがメンテナンスの予定になっている場合は、イベントの 14 日前に予定されているイベント通知が届きます。インスタンスを新しいホストに移動するには、インスタンスを手動で停止または再起動する必要があります。

概念と用語

AMD SEV-SNP の使用を開始する前に、次の概念と用語を理解しておきます。

AMD SEV-SNP 認証レポート

AMD SEV-SNP 認証レポートは、インスタンスが CPU に要求できる文書です。AMD SEV-SNP 認証レポートを使用して、インスタンスの状態と ID を検証し、認可された AMD 環境で実行されていることを確認できます。レポートには起動測定値が含まれます。起動測定とは、インスタンスの初期起動状態の暗号化ハッシュであり、初期インスタンスのメモリ内容と vCPU の初期状態が含まれます。AMD SEV-SNP 認証レポートには、AMD のルートオブトラストに紐づく VLEK 署名が署名されています。

VLEK

バージョニングロードエンドースメントキー (VLEK) は、AMD が認定したバージョン付きの署名キーで、AMD CPU が AMD SEV-SNP 認証レポートに署名する際に使用します。VLEK 署名は、AMD が提供する証明書を使用して検証できます。

OVMF バイナリ

オープン仮想マシンファームウェア (OVMF) は、インスタンスに UEFI 環境を提供するために使用されるアーリーブートコードです。アーリーブートコードは、AMI のコードが起動する前に実行されます。また、OVMF は AMI で提供されるブートローダーを見つけて実行します。詳細については、「OVMF リポジトリ」を参照してください。

AMD SEV-SNP との連携

トピック

- サポートされているインスタンスタイプの検索
- 起動時に AMD SEV-SNP を有効にする
- AMD SEV-SNP のステータスをチェックしてください

サポートされているインスタンスタイプの検索

AWS CLI を使用して、AMD SEV-SNP をサポートするインスタンスタイプを検索できます。

AWS CLI

AWS CLI を使用して AMD SEV-SNP をサポートするインスタンスタイプを検索するには、「describe-instance-types」次のコマンドを使用します。

```
$ aws ec2 describe-instance-types \
--filters Name=processor-info.supported-features,Values=amd-sev-snp \
--query 'InstanceTypes[*].InstanceType'
```

出力例。

```
[
    "r6a.2xlarge",
    "m6a.large",
    "m6a.2xlarge",
    "r6a.xlarge",
    "c6a.16xlarge",
    "c6a.8xlarge",
    "m6a.4xlarge",
    "c6a.12xlarge",
    "r6a.4xlarge",
    "c6a.xlarge",
    "c6a.4xlarge",
    "c6a.2xlarge",
    "m6a.xlarge",
    "c6a.large",
    "r6a.large",
    "m6a.8xlarge"
]
```

起動時に AMD SEV-SNP を有効にする

AWS CLI を使用して、AMD SEV-SNP を有効にした状態でインスタンスを起動できます。

AWS CLI

AWS CLI を使用して AMD SEV-SNP「run-instances」を有効にした状態でインスタンスを起動するには、`--cpu-options AmdSevSnp=enabled` コマンドを使用してオプションを含めます。`--image-id` には、uefi uefi-prefered またはブートモードの AMI と AMD SEV-SNP をサポートするオペレーティングシステムを指定します。`--instance-type` には、サポートされているインスタンスタイプを指定してください。

```
$ aws ec2 run-instances \
--image-id supported_ami_id \
--instance-type supported_instance_type \
--key-name key_pair_name \
--subnet-id subnet_id \
--cpu-options AmdSevSnp=enabled
```

AMD SEV-SNP のステータスをチェックしてください

次のいずれかの方法を使用して、AMD SEV-SNP をサポートするインスタンスタイプを検索できます。

AWS CLI

AWS CLI を使用してインスタンスで AMD SEV-SNP がオンになっているかどうかを確認するには、「describe-instances」コマンドを使用します。`--instance-ids` には、チェックするインスタンスの ID を指定します。

```
$ aws ec2 describe-instances --instance-ids instance_id
```

コマンド出力の `AmdSevSnp` の `CpuOptions` の値は、AMD SEV-SNP がオンかオフかを示します。

AWS CloudTrail

インスタンス起動要求の AWS CloudTrail イベントでは、値 `"cpuOptions": {"AmdSevSnp": enabled}` は、インスタンスに対して AMD SEV-SNP がオンになっていることを示します。

AMD SEV-SNP による認証

認証は、インスタンスがその状態と身元を証明できるようにするプロセスです。インスタンスで AMD SEV-SNP を有効にすると、基盤となるプロセッサに AMD SEV-SNP 認証レポートをリクエストできます。AMD SEV-SNP 認証レポートには、初期ゲストメモリーの内容と vCPU の初期状態の起動測定と呼ばれる暗号化ハッシュが含まれています。認証レポートには VLEK 署名が付いており、AMD のルートオブトラストに紐づいています。認証レポートに含まれる起動測定値を使用して、インスタンスが正規の AMD 環境で実行されていることを確認し、インスタンスの起動に使用された初期ブートコードを検証できます。

AMD SEV-SNP で認証を実行するには、次のステップを完了します。

ステップ 1: 認証レポートを取得する

このステップでは、必要なツールをインストールし、プロセッサーからの AMD SEV-SNP 認証レポートとプロセッサーの VLEK 署名キーを要求します。

1. sev-guest ユーティリティを使用して CPU にアテステーションレポートをリクエストする必要があります。sev-guest からユーティリティをインストールするには「sev-guest repository」、次のコマンドを実行します。

```
$ git clone https://github.com/AMDESE/sev-guest.git
$ cd sev-guest
$ make sev-guest-get-report
$ make sev-guest-parse-report
```

2. sev-guest ユーティリティを使用して、認証レポートと、認証レポートの署名に使用された VLEK 証明書をリクエストします。

```
$ sudo ./sev-guest-get-report guest_report.bin -x
```

このコマンドは次の 2 つのファイルを作成します。

- guest_report.bin — 署名された認証レポート。
- a8074bc2-a25a-483e-aae6-39c045a0b8a1 — 認証レポートの署名に使用された識別符号化規則 (DER) 形式の VLEK 証明書。

ステップ 2: 認証レポートの署名を検証する

認証レポートには、AMD が AWS のために発行するバージョン対応認証キー (VLEK) と呼ばれる証明書が署名されています。このステップでは、VLEK 証明書が AMD によって発行されていること、および認証レポートが VLEK 証明書によって署名されていることを確認します。

1. sev-tool ユーティリティを使用して、認証レポートが VLEK 証明書によって署名されていることを確認する必要があります。このユーティリティでは、証明レポートと VLEK 証明書が、/certs という名前のフォルダに存在する必要があります。次のコマンドを実行し、/certs ディレクトリを作成します。

```
$ sudo mkdir certs
```

2. DER でエンコードされた VLEK 証明書 (a8074bc2-a25a-483e-aae6-39c045a0b8a1) を、sev-guest ユーティリティに必要な PEM 形式に変換します。

```
$ sudo openssl x509 -inform der -in a8074bc2-a25a-483e-aae6-39c045a0b8a1 -out
 certs/vcek.pem
```

VLEK 証明書は、/certs ディレクトリ内の新しいファイル (vcek.pem) に書き込まれます。

3. /certs VLEK のルートオブトラスト証明書を AMD の公式ウェブサイトからディレクトリにダウンロードします。

```
$ sudo curl --proto '=https' --tlsv1.2 -sSf https://kdsintf.amd.com/vlek/v1/Milan/
cert_chain -o certs/cert_chain.pem
```

4. openssl を使用して、VLEK 証明書が AMD の信頼証明書ルートによって署名されていることを確認します。

```
$ sudo openssl verify --CAfile certs/cert_chain.pem certs/vcek.pem
```

正常な出力

```
certs/vcek.pem: OK
```

5. /certs 認証レポートをディレクトリにコピーします。

```
$ sudo cp guest_report.bin certs/
```

6. sev-tool ユーティリティを使用して認証レポートの署名を検証する必要があります。次のコマンドを実行して、sev-tool ユーティリティをインストールします。

```
$ cd ..
$ git clone https://github.com/AMDESE/sev-tool.git
$ cd sev-tool
$ autoreconf -vif && ./configure && make
```

7. sev-tool ユーティリティを使用して、認証レポートが VLEK 証明書によって署名されていることを確認します。

```
$ sudo ./src/sevtool --ofolder ../sev-guest/certs --validate_guest_report
```

正常な出力

```
Guest report validated successfully!
```

Amazon Linux インスタンスのホスト名の変更

プライベート VPC 内でインスタンスを起動すると、Amazon EC2 によってゲスト OS ホスト名が割り当てられます。Amazon EC2 によって割り当てられるホスト名のタイプは、サブネット設定によって異なります。EC2 ホスト名の詳細については、「Amazon EC2 インスタンスのホスト名タイプ」を参照してください。

IPv4 アドレスで IP ベースの命名を使用するように構成された EC2 インスタンスの典型的な Amazon EC2 のプライベート DNS 名は、`ip-12-34-56-78.us-west-2.compute.internal` のような形式になります。この名前は内部ドメイン、サービス (この例では、`compute`)、リージョン、そしてプライベート IPv4 アドレスで構成されます。インスタンスにログインしたとき、このホスト名の一部がシェルプロンプトで表示されます (`ip-12-34-56-78` など)。Amazon EC2 インスタンスを停止し、再起動するたびに (Elastic IP アドレスを使用していない限り)、パブリック IPv4 アドレスが変わり、パブリック DNS 名、システムホスト名、シェルプロンプトも変わります。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

システムホスト名の変更

インスタンスの IP アドレスにパブリック DNS 名を登録している場合 (`webserver.mydomain.com` など)、インスタンスがそのドメインに含まれているものとして識別されるように、システムホスト名を設定できます。また、システムホスト名を変更すると、シェルプロンプトも変更され、AWS が提供するホスト名の代わりに、新しいシステムホスト名の最初の部分 (`ip-12-34-56-78` など) が表示されます。パブリック DNS 名を登録していない場合でもホスト名は変更できますが、プロセスが少し違います。

ホスト名の更新内容を維持するには、`preserve_hostname` cloud-init 設定が `true` に設定されていることを確認してください。この設定を編集または追加するには、次のコマンドを実行します。

```
sudo vi /etc/cloud/cloud.cfg
```

preserve_hostname 設定が一覧表示されない場合は、ファイルの末尾に次のテキスト行を追加します。

```
preserve_hostname: true
```

システムホスト名をパブリック DNS 名に変更するには

パブリック DNS 名を登録している場合、この手順を行います。

1. • Amazon Linux 2 の場合: hostnamectl コマンドを使用してホスト名を設定し、完全修飾ドメイン名 (**webserver.mydomain.com**) を反映させます。

   ```
   [ec2-user ~]$ sudo hostnamectl set-hostname webserver.mydomain.com
   ```

   • Amazon Linux AMI の場合: インスタンスで、お好みのテキストエディタを使用して /etc/sysconfig/network 設定ファイルを開き、HOSTNAME エントリを変更して、完全修飾ドメイン名 (**webserver.mydomain.com** など) を反映させます。

   ```
   HOSTNAME=webserver.mydomain.com
   ```

2. インスタンスを再起動し、新しいホスト名を取得します。

   ```
   [ec2-user ~]$ sudo reboot
   ```

   または、Amazon EC2 コンソールを使用して再起動することもできます ([Instances (インスタンス)] ページでインスタンスを選択し、[Instance state (インスタンスの状態)]、[Reboot instance (インスタンスの再起動)] の順に選択します)。

3. インスタンスにログインして、ホスト名が更新されていることを確認します。メッセージには、新しいホスト名が表示されるはずです (最初の「.」まで)。hostname コマンドで完全修飾ドメイン名が表示されます。

   ```
   [ec2-user@webserver ~]$ hostname
   webserver.mydomain.com
   ```

パブリック DNS 名なしでシステムホスト名を変更するには

1. • Amazon Linux 2 の場合: hostnamectl コマンドを使用してホスト名を設定し、必要なシステムホスト名 (**webserver** など) を反映させます。

```
[ec2-user ~]$ sudo hostnamectl set-hostname webserver.localdomain
```

   • Amazon Linux AMI の場合: インスタンスで、お好みのテキストエディタで /etc/sysconfig/network 設定ファイルを開き、HOSTNAME エントリを変更して、希望するシステムホスト名を反映させます (例: **webserver**)。

```
HOSTNAME=webserver.localdomain
```

2. お好みのテキストエディタで /etc/hosts ファイルを開き、下の例と一致する **127.0.0.1** で始まるエントリを変更します。ホスト名は自分のホスト名に置換します。

```
127.0.0.1 webserver.localdomain webserver localhost4 localhost4.localdomain4
```

3. インスタンスを再起動し、新しいホスト名を取得します。

```
[ec2-user ~]$ sudo reboot
```

   または、Amazon EC2 コンソールを使用して再起動することもできます ([Instances (インスタンス)] ページでインスタンスを選択し、[Instance state (インスタンスの状態)]、[Reboot instance (インスタンスの再起動)] の順に選択します)。

4. インスタンスにログインして、ホスト名が更新されていることを確認します。メッセージには、新しいホスト名が表示されるはずです (最初の「.」まで)。hostname コマンドで完全修飾ドメイン名が表示されます。

```
[ec2-user@webserver ~]$ hostname
webserver.localdomain
```

また、ユーザーデータを指定してインスタンスを設定するなど、よりプログラム的なソリューションを実装することもできます。インスタンスが Auto Scaling グループの一部である場合、ライフサイクルフックを使用してユーザーデータを定義できます。詳細については、「AWS CloudFormation ユーザーガイド」の「Run commands on your Linux instance at launch」(起動時に Linux インスタンスでコマンドを実行する) および「Lifecycle hook for instance launch」(インスタンス起動のライフサイクルフック) を参照してください。

ホスト名に影響を与えずにシェルプロンプトを変更する

インスタンスのホスト名を変更せずに、AWS が提供するプライベート名 (**webserver** など) よりも便利なシステム名 (ip-12-34-56-78 など) を表示させる場合、ホスト名の代わりにシステムニックネームを表示するようにシェルプロンプト設定ファイルを編集できます。

シェルプロンプトをホストニックネームに変更するには

1. /etc/profile.d で、NICKNAME と呼ばれる環境変数を設定するファイルを作成して、シェルプロンプトに表示する値を設定します。例えば、システムニックネームを **webserver** に設定するには、次のコマンドを実行します。

   ```
   [ec2-user ~]$ sudo sh -c 'echo "export NICKNAME=webserver" > /etc/profile.d/
   prompt.sh'
   ```

2. お好みのテキストエディタ (/etc/bashrc や /etc/bash.bashrc など) で、vim (Red Hat) または nano (Debian/Ubuntu) ファイルを開きます。エディタのコマンドで sudo を使用する必要があります。/etc/bashrc および /etc/bash.bashrc は root が所有するためです。
3. ファイルを編集し、ホスト名の代わりにニックネームを表示するようにシェルプロンプト変数 (PS1) を変更します。/etc/bashrc または /etc/bash.bashrc でシェルプロンプトを設定する次の行を見つけます (以下には、コンテキストを示すため前後の行も表示されています。["$PS1" で始まる行を探してください)。

   ```
   # Turn on checkwinsize
   shopt -s checkwinsize
   [ "$PS1" = "\\s-\\v\\$ " ] && PS1="[\u@\h \W]\\$ "
   # You might want to have e.g. tty in prompt (e.g. more virtual machines)
   # and console windows
   ```

   その行の \h (hostname を表す記号) を NICKNAME 変数の値に変更します。

   ```
   # Turn on checkwinsize
   shopt -s checkwinsize
   [ "$PS1" = "\\s-\\v\\$ " ] && PS1="[\u@$NICKNAME \W]\\$ "
   # You might want to have e.g. tty in prompt (e.g. more virtual machines)
   # and console windows
   ```

4. (オプション) シェルウィンドウのタイトルを新しいニックネームに設定するには、次の手順を完了します。

a. /etc/sysconfig/bash-prompt-xterm という名前のファイルを作成します。

```
[ec2-user ~]$ sudo touch /etc/sysconfig/bash-prompt-xterm
```

b. 次のコマンドを使用して、ファイルを実行可能にします。

```
[ec2-user ~]$ sudo chmod +x /etc/sysconfig/bash-prompt-xterm
```

c. お好みのテキストエディタ (/etc/sysconfig/bash-prompt-xterm や vim など) で、nano ファイルを開きます。エディタのコマンドで sudo を使用する必要があります。/etc/sysconfig/bash-prompt-xterm は root が所有するためです。

d. 次の行をファイルに追加します。

```
echo -ne "\033]0;${USER}@${NICKNAME}:${PWD/#$HOME/~}\007"
```

5. ログアウトしてから再度ログインし、新しいニックネーム値を取得します。

他の Linux ディストリビューションのホスト名の変更

このページの手順は、Amazon Linux のみで使用するためのものです。他の Linux ディストリビューションの詳細については、各ドキュメントおよび次の記事を参照してください。

- RHEL 7 または Centos 7 を実行するプライベート Amazon EC2 インスタンスに静的ホスト名を割り当てる方法を教えてください。

Amazon Linux インスタンスでの動的な DNS のセットアップ

EC2 インスタンスを起動すると、パブリック IP アドレスとパブリックドメインネームシステム (DNS) が割り当てられます。それらを使用してインターネットからインスタンスにアクセスできます。Amazon Web Services ドメインには数多くのホストが存在するため、これらのパブリック名はそれぞれの名前を一意にするために、かなり長くする必要があります。典型的な Amazon EC2 パブリック DNS 名は、ec2-12-34-56-78.us-west-2.compute.amazonaws.com のような形式になります。この名前は Amazon Web Services ドメイン、サービス (この例では、compute)、AWS リージョン、パブリック IP アドレスで構成されます。

動的 DNS サービスはそのドメイン領域内でカスタムの DNS ホスト名を提供します。この名前は覚えやすく、ホストのユースケースとの関連性が高くなっています。また、これらのサービスは一部は無料で提供されています。Amazon EC2 では動的 DNS プロバイダを利用できます。また、インスタ

ンスを起動するたびに、パブリック DNS 名に関連付けられている IP アドレスを更新するようにインスタンスを設定できます。プロバイダは数多く存在します。また、プロバイダを選択し、それぞれのプロバイダで名前を登録する方法については本ガイドの範囲外です。

> **Important**
>
> この情報は、Amazon Linux に適用されます。その他のディストリビューションの情報については、各ドキュメントを参照してください。

Amazon EC2 で動的 DNS を使用するには

1. 動的 DNS サービスプロバイダにサインアップし、そのサービスでパブリック DNS 名を登録します。この手順では、noip.com/free の無料サービスを例として使用します。
2. 動的 DNS 更新クライアントを設定します。動的 DNS サービスプロバイダを選び、そのサービスでパブリック DNS 名を登録したら、その DNS 名をインスタンスの IP アドレスにポイントします。多くのプロバイダ (noip.com を含む) では、この操作をウェブサイトのアカウントページから手動で実行できるようにしています。ただし、ソフトウェア更新クライアントもサポートしています。EC2 インスタンスで更新クライアントが動作している場合は、シャットダウン後に再起動したときに IP アドレスが変わるたびに動的 DNS レコードが更新されます。この例では、noip2 クライアントをインストールします。このクライアントは、「noip.com」が提供するサービスで動作します。

   a. noip2 クライアントにアクセスできるように、Extra Packages for Enterprise Linux (EPEL) リポジトリを有効にします。

      > **Note**
      >
      > Amazon Linux インスタンスは、デフォルトでインストールされる EPEL リポジトリに関する GPG キーとリポジトリ情報を保持しています。ただし、Red Hat インスタンスと CentOS インスタンスについては、`epel-release` パッケージを最初にインストールしてから、EPEL リポジトリを有効にする必要があります。このパッケージの最新バージョンの詳細およびダウンロードについては、「https://fedoraproject.org/wiki/EPEL」を参照してください。

      - Amazon Linux 2 用

```
[ec2-user ~]$ sudo yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
```

- Amazon Linux AMI 用

```
[ec2-user ~]$ sudo yum-config-manager --enable epel
```

b. noip パッケージをインストールします。

```
[ec2-user ~]$ sudo yum install -y noip
```

c. 設定ファイルを作成します。メッセージが表示されたら、ログインおよびパスワード情報を入力して、その後に続く質問に答え、クライアントを設定します。

```
[ec2-user ~]$ sudo noip2 -C
```

3. noip サービスを有効にします。

- Amazon Linux 2 用

```
[ec2-user ~]$ sudo systemctl enable noip.service
```

- Amazon Linux AMI 用

```
[ec2-user ~]$ sudo chkconfig noip on
```

4. noip サービスを開始します。

- Amazon Linux 2 用

```
[ec2-user ~]$ sudo systemctl start noip.service
```

- Amazon Linux AMI 用

```
[ec2-user ~]$ sudo service noip start
```

このコマンドを使用すると、クライアントが起動します。クライアントは前に作成した設定ファイル (/etc/no-ip2.conf) を読み、選択したパブリック DNS 名の IP アドレスを更新します。

5. 更新クライアントが動的 DNS 名に正しい IP アドレスを設定したことを確認します。DNS レコードの更新には数分かかります。その後、この手順で設定したパブリック DNS 名により、SSH を使用してインスタンスに接続します。

起動時に Linux インスタンスでコマンドを実行する

Amazon EC2 でインスタンスを起動するとき、起動後にそのインスタンスにユーザーデータを渡し、一般的な自動設定タスクを実行したり、スクリプトを実行したりできます。2 つのタイプのユーザーデータを Amazon EC2 に渡すことができます。シェルスクリプトと cloud-init ディレクティブです。また、このデータは、プレーンテキスト、ファイル (コマンドラインツールを使用してインスタンスを起動する場合に便利です)、または base64 でエンコードされたテキスト (API コール向け) として、インスタンス起動ウィザードに渡すこともできます。

より複雑なオートメーションのシナリオに興味がある場合、AWS CloudFormation や AWS OpsWorks のご利用を検討してください。詳細については、次を参照してください:

- 「AWS CloudFormation ユーザーガイド」の「AWS CloudFormation を使用して Amazon EC2 にアプリケーションをデプロイします」。
- AWS OpsWorks ユーザーガイド。

Windows インスタンスの起動時にコマンドを実行する方法については、Windows インスタンスの Amazon EC2 ユーザーガイドの「Windows インスタンスでの起動時のコマンドの実行」および「Windows インスタンス設定の管理」を参照してください。

次の例では、「Amazon Linux 2 に LAMP ウェブサーバーをインストールする」のコマンドが、シェルスクリプトと、インスタンスの起動時に実行される一連の cloud-init ディレクティブに変換されています。各例では、次のタスクがユーザーデータにより実行されます。

- ディストリビューションソフトウェアパッケージが更新されます。
- 必要なウェブサーバー、php、`mariadb` パッケージがインストールされます。
- systemctl を介して `httpd` サービスが開始され、オンになります。
- `ec2-user` が apache グループに追加されます。
- ウェブディレクトリとその中に含まれるファイルに対して、適切な所有権とファイル権限が設定されます。
- ウェブサーバーと PHP エンジンをテストするために、シンプルなウェブページが作成されます。

コンテンツ

前提条件

このトピックの例では、

- インスタンスには、インターネットからアクセス可能なパブリック DNS 名が設定されていることを前提にしています。詳細については、ネットワーク設定 セクションおよび セキュリティグループの作成 の [Auto-assign Public IP](パブリック IP の自動割り当て) を参照してください。
- インスタンスに関連付けられたセキュリティグループは、SSH (ポート 22) トラフィックを許可するように設定されているため、インスタンスに接続して出力ログファイルを表示できます。詳細については、「セキュリティグループの作成」を参照してください。
- インスタンスは Amazon Linux 2 AMI を使用して起動します。この説明は Amazon Linux 2 での使用を意図したものです。他の Linux ディストリビューションの場合、コマンドとディレクティブが動作しないことがあります。cloud-init のサポートなど、その他のディストリビューションについての詳細は、該当するディストリビューションの文書を参照してください。

ユーザーデータとシェルスクリプト

シェルスクリプトに慣れている場合は、この方法が最も簡単で完全に起動時に指示を送信する方法です。起動時にこれらのタスクを追加すると、インスタンスの起動にかかる時間が増えます。タスクが完了するまでさらに数分待ち、それからユーザースクリプトが正常に完了したことをテストしてください。

> **Important**
>
> デフォルトでは、ユーザーデータスクリプトと cloud-init ディレクティブは、インスタンスの最初の起動サイクル中にのみ実行されます。インスタンスを再起動するたびにユーザーデータスクリプトと cloud-init ディレクティブが実行されるように設定を更新できます。詳

細については、AWS ナレッジセンターの「Amazon EC2 Linux インスタンスを再起動する度にユーザーデータを実行して自動的にファイルを作成するにはどうすればよいですか?」を参照してください。

ユーザーデータシェルスクリプトは、#! 文字と、スクリプトの読み取り先であるインタープリタのパス (通常は /bin/bash)) で開始する必要があります。シェルスクリプティングに関する有用な紹介文は、Linux ドキュメントプロジェクト (tldp.org) の「BASHプログラミングのハウツー」で入手できます。

ユーザーデータとして入力されたスクリプトはルートユーザーとして実行されます。そのため、スクリプトでは sudo コマンドを使用しないでください。作成したファイルはすべてルートユーザーの所有になることを忘れないでください。ルート以外のユーザーにファイルアクセスを与える場合、スクリプトで許可を適宜変更する必要があります。また、スクリプトはインタラクティブに実行されないため、ユーザーフィードバックを必要とするコマンド (-y フラグのない yum update など) を含めることはできません。

ユーザーデータスクリプトで AWS API (AWS CLI など) を使用する場合は、インスタンスを起動するときにインスタンスプロファイルを使用する必要があります。インスタンスプロファイルは、API 呼び出しを発行するためにユーザーデータスクリプトが必要とする適切な AWS 認証情報を提供します。詳細については、IAM ユーザーガイド の「インスタンスプロファイルの使用」を参照してください。IAM ロールに割り当てるアクセス許可は、API で呼び出すサービスによって異なります。詳細については、「Amazon EC2 の IAM ロール」を参照してください。

cloud-init 出力ログファイルでコンソール出力がキャプチャされるため、インスタンスが意図したように動作しない場合でも、起動後、簡単にスクリプトをデバッグすることができます。ログファイルを表示するには、インスタンスに接続し、`/var/log/cloud-init-output.log` を開きます。

ユーザーデータスクリプトを処理すると、`/var/lib/cloud/instances/`*`instance-id`*`/` にコピーされ、実行されます。実行後にスクリプトを削除することはできません。必ず `/var/lib/cloud/instances/`*`instance-id`*`/` のユーザーデータスクリプトを削除してから、インスタンスに AMI を作成してください。それ以外の場合、スクリプトはこの AMI から起動されたインスタンスのこのディレクトリに存在します。

ユーザーデータおよびコンソール

インスタンスの起動時のインスタンスユーザーデータを指定できます。インスタンスのルートボリュームが EBS ボリュームの場合は、インスタンスを停止してユーザーデータを更新することもできます。

起動時にインスタンスユーザーデータを指定する

インスタンスを起動するための手順に従います。[User data] (ユーザーデータ) フィールドは、インスタンス起動ウィザードの 高度な詳細 セクションにあります。シェルスクリプトを [User data] (ユーザーデータ) フィールドに入力してから、インスタンスの起動手順を完了します。

下のスクリプト例では、スクリプトがウェブサーバーを作成し、設定します。

```
#!/bin/bash
yum update -y
amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2
yum install -y httpd mariadb-server
systemctl start httpd
systemctl enable httpd
usermod -a -G apache ec2-user
chown -R ec2-user:apache /var/www
chmod 2775 /var/www
find /var/www -type d -exec chmod 2775 {} \;
find /var/www -type f -exec chmod 0664 {} \;
echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php
```

インスタンスが起動し、スクリプトのコマンドを実行するまで十分待ち、それからスクリプトが意図したタスクを完了したことを確認します。

例では、ウェブブラウザにスクリプトが作成した PHP テストファイルの URL を入力します。この URL は、インスタンスのパブリック DNS アドレスにスラッシュとファイル名を追加したものです。

```
http://my.public.dns.amazonaws.com/phpinfo.php
```

PHP 情報ページが表示されるはずです。PHP 情報ページが表示されない場合、使用しているセキュリティグループに HTTP (ポート 80) トラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

(オプション) スクリプトが期待したタスクを達成しなかった場合、あるいは単にスクリプトがエラーなしで完了したことを確認する場合は、インスタンスに接続し、cloud-init 出力ログファイル (`/var/log/cloud-init-output.log`) を調べ、出力にエラーメッセージがないか探します。

デバッグの詳細情報を取得するには、次のディレクティブを指定して cloud-init データセクションを含む Mime マルチパートアーカイブを作成します。

```
output : { all : '| tee -a /var/log/cloud-init-output.log' }
```

このディレクティブにより、スクリプトから /var/log/cloud-init-output.log にコマンド出力が送信されます。cloud-init データ形式と MIME マルチパートアーカイブの作成方法の詳細については、「cloud-init Formats」を参照してください。

インスタンスユーザーデータの表示と更新

インスタンスのユーザーデータを更新するには、まずインスタンスを停止する必要があります。インスタンスが実行されている場合は、ユーザーデータを表示できますが、変更することはできません。

> **Warning**
>
> インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスストアボリュームのデータを保持するには、データを永続的ストレージに必ずバックアップします。

インスタンスユーザーデータを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。このオプションが無効になっている場合は、インスタンスが既に停止しているか、またはルートボリュームがインスタンスストアボリュームです。
4. 確認を求められたら、[Stop] を選択します。インスタンスが停止するまで、数分かかる場合があります。
5. インスタンスが選択された状態のまま、[Actions (アクション)]、[Instance settings (インスタンス設定)]、[Edit user data (ユーザーデータの編集)] の順に選択します。
6. 必要に応じてユーザーデータを変更し、[Save (保存)] を選択します。
7. インスタンスを起動します。新しいユーザーデータは、再起動後にインスタンス上に表示されますが、ユーザーデータスクリプトは実行されません。

ユーザーデータと cloud-init ディレクティブ

cloud-init パッケージは、新しい Amazon Linux インスタンスが起動したときに、特定の側面を設定します。具体的には、お客様のプライベートキーでログインできるように、ec2-user の .ssh/authorized_keys ファイルを設定します。Amazon Linux インスタンスに対して cloud-init パッケージが実行する設定タスクの詳細については、「cloud-init」を参照してください。

構文は異なりますが、渡されたスクリプトと同じ方法で cloud-init ユーザーディレクティブを起動時のインスタンスに渡すことができます。cloud-init の詳細については、http://cloudinit.readthedocs.org/en/latest/index.html を参照してください。

> **Important**
>
> デフォルトでは、ユーザーデータスクリプトと cloud-init ディレクティブは、インスタンスの最初の起動サイクル中にのみ実行されます。インスタンスを再起動するたびにユーザーデータスクリプトと cloud-init ディレクティブが実行されるように設定を更新できます。詳細については、AWS ナレッジセンターの「Amazon EC2 Linux インスタンスを再起動する度にユーザーデータを実行して自動的にファイルを作成するにはどうすればよいですか?」を参照してください。

起動時にこれらのタスクを追加すると、インスタンスの起動にかかる時間が増えます。タスクが完了するまでさらに数分待ち、それからユーザーデータディレクティブが完了したことをテストしてください。

ユーザーデータで cloud-init ディレクティブをインスタンスに渡すには

1. インスタンスを起動するための手順に従います。[User data] (ユーザーデータ) フィールドは、インスタンス起動ウィザードの 高度な詳細 セクションにあります。cloud-init ディレクティブテキストを[User data] (ユーザーデータ) フィールドに入力してから、インスタンスの起動手順を完了します。

   下の例では、ディレクティブが Amazon Linux 2 でウェブサーバーを作成し、設定します。一番上の `#cloud-config` 行は、cloud-init ディレクティブとしてコマンドを識別するために必要です。

   ```
   #cloud-config
   repo_update: true
   repo_upgrade: all

   packages:
    - httpd
    - mariadb-server

   runcmd:
    - [ sh, -c, "amazon-linux-extras install -y lamp-mariadb10.2-php7.2 php7.2" ]
    - systemctl start httpd
   ```

```
- sudo systemctl enable httpd
- [ sh, -c, "usermod -a -G apache ec2-user" ]
- [ sh, -c, "chown -R ec2-user:apache /var/www" ]
- chmod 2775 /var/www
- [ find, /var/www, -type, d, -exec, chmod, 2775, {}, \; ]
- [ find, /var/www, -type, f, -exec, chmod, 0664, {}, \; ]
- [ sh, -c, 'echo "<?php phpinfo(); ?>" > /var/www/html/phpinfo.php' ]
```

2. インスタンスが起動し、ユーザーデータのディレクティブを実行するまで十分待ち、それから意図したタスクをディレクティブが完了したことを確認します。

   この例では、ウェブブラウザで、ディレクティブが作成した PHP テストファイルの URL を入力します。この URL は、インスタンスのパブリック DNS アドレスにスラッシュとファイル名を追加したものです。

   ```
   http://my.public.dns.amazonaws.com/phpinfo.php
   ```

   PHP 情報ページが表示されるはずです。PHP 情報ページが表示されない場合、使用しているセキュリティグループに HTTP (ポート 80) トラフィックを許可するルールが含まれていることを確認します。詳細については、「セキュリティグループへのルールの追加」を参照してください。

3. (オプション) ディレクティブが期待したタスクを達成しなかった場合、あるいは単にディレクティブがエラーなしで完了したことを確認する場合は、インスタンスに接続し、出力ログファイル (/var/log/cloud-init-output.log) を調べ、出力にエラーメッセージがないか探します。デバッグの詳細情報を取得するには、ディレクティブに次の行を追加します:

   ```
   output : { all : '| tee -a /var/log/cloud-init-output.log' }
   ```

   このディレクティブにより、runcmd 出力が /var/log/cloud-init-output.log に送信されます。

ユーザーデータと AWS CLI

AWS CLI を使用して、インスタンスのユーザーデータを指定、変更、表示することができます。インスタンスのメタデータを使用して、インスタンスからユーザーデータを表示する方法については、「インスタンスからインスタンスユーザーデータを取得する」を参照してください。

Windows では、AWS CLI を使用する代わりに AWS Tools for Windows PowerShell を使用できます。詳細については、Windows インスタンスの Amazon EC2 ユーザーガイド の「ユーザーデータと Tools for Windows PowerShell」を参照してください。

例: ユーザーデータは、起動時に指定します。

インスタンスの起動時にユーザーデータを指定するには、run-instances コマンドと --user-data パラメータを使用します。run-instances で、AWS CLI はユーザーデータの base64 エンコードを実行します。

次の例は、コマンドラインで文字列としてスクリプトを指定する方法を示しています。

```
aws ec2 run-instances --image-id ami-abcd1234 --count 1 --instance-type m3.medium \
--key-name my-key-pair --subnet-id subnet-abcd1234 --security-group-ids sg-abcd1234 \
--user-data echo user data
```

次の例は、テキストファイルを使用してスクリプトを指定する方法を示しています。ファイルを指定するには、必ず file:// プレフィクスを使用してください。

```
aws ec2 run-instances --image-id ami-abcd1234 --count 1 --instance-type m3.medium \
--key-name my-key-pair --subnet-id subnet-abcd1234 --security-group-ids sg-abcd1234 \
--user-data file://my_script.txt
```

シェルスクリプトを使用したテキストファイルの例を次に示します。

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

例: 停止しているインスタンスのユーザーデータを変更する

停止したインスタンスのユーザーデータは、modify-instance-attribute コマンドを使用して変更できます。modify-instance-attribute では、AWS CLI はユーザーデータの base64 エンコードを実行しません。

- Linux コンピュータでは、base64 コマンドを使用してユーザーデータをエンコードします。

```
base64 my_script.txt >my_script_base64.txt
```

- Windows コンピュータでは、certutil コマンドを使用してユーザーデータをエンコードします。このファイルを AWS CLI で使用する前に、最初の (証明書の開始) 行と最後の (証明書の終了) 行を削除する必要があります。

```
certutil -encode my_script.txt my_script_base64.txt
notepad my_script_base64.txt
```

`--attribute` および `--value` パラメータを使用して、エンコードされたテキストファイルを使用してユーザーデータを指定します。ファイルを指定するには、必ず `file://` プレフィクスを使用してください。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 userData --value file://my_script_base64.txt
```

例: 停止しているインスタンスのユーザーデータをクリアする

既存のユーザーデータを削除するには、次の modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --user-data Value=
```

例: ユーザーデータの表示

インスタンスのユーザーデータを取得するには、describe-instance-attribute コマンドを使用します。describe-instance-attribute では、AWS CLI はユーザーデータの base64 デコードを実行しません。

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 userData
```

ユーザーデータが base64 でエンコードされた出力例を次に示します。

```
{
    "UserData": {
        "Value":
 "IyEvYmluL2Jhc2gKeXVtIHVwZGF0ZSAteQpzZXJ2aWNlIGh0dHBkIHN0YXJ0CmNoa2NvbmZpZyBodHRwZCBvbg=="
    },
    "InstanceId": "i-1234567890abcdef0"
}
```

- Linux コンピュータでは、--query オプションを使用してエンコードされたユーザーデータを取得し、base64 コマンドを使用してデコードします。

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 userData --output text --query "UserData.Value" | base64 --decode
```

- Windows コンピュータでは、--query オプションを使用してコード化されたユーザーデータを取得し、certutil コマンドを使用してコードをデコードします。エンコードされた出力はファイルに保存され、デコードされた出力は別のファイルに保存されることに注意してください。

```
aws ec2 describe-instance-attribute --instance-id i-1234567890abcdef0 --attribute
 userData --output text --query "UserData.Value" >my_output.txt
certutil -decode my_output.txt my_output_decoded.txt
type my_output_decoded.txt
```

以下は出力例です。

```
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

シェルスクリプトと cloud-init ディレクティブを組み合わせる

デフォルトでは、ユーザーデータに含めることができるコンテンツタイプは一度に 1 つだけです。ただし、MIME マルチパートファイルの中で text/cloud-config と text/x-shellscript のコンテンツタイプを使用して、ユーザーデータにシェルスクリプトと cloud-init ディレクティブの両方を含めることは可能です。

以下に、MIME マルチパートの形式を示します。

```
Content-Type: multipart/mixed; boundary="//"
MIME-Version: 1.0

--//
Content-Type: text/cloud-config; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="cloud-config.txt"
```

```
#cloud-config
cloud-init directives

--//
Content-Type: text/x-shellscript; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="userdata.txt"

#!/bin/bash
shell script commands
--//--
```

例えば、次のユーザーデータには cloud-init ディレクティブと bash シェルスクリプトが含まれています。cloud-init ディレクティブはファイル (`/test-cloudinit/cloud-init.txt`) を作成し、そのファイルに `Created by cloud-init` を書き込みます。bash シェルスクリプトはファイル (`/test-userscript/userscript.txt`) を作成し、そのファイルに `Created by bash shell script` を書き込みます。

```
Content-Type: multipart/mixed; boundary="//"
MIME-Version: 1.0

--//
Content-Type: text/cloud-config; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="cloud-config.txt"

#cloud-config
runcmd:
 - [ mkdir, /test-cloudinit ]
write_files:
 - path: /test-cloudinit/cloud-init.txt
   content: Created by cloud-init

--//
Content-Type: text/x-shellscript; charset="us-ascii"
MIME-Version: 1.0
Content-Transfer-Encoding: 7bit
Content-Disposition: attachment; filename="userdata.txt"

#!/bin/bash
  mkdir test-userscript
```

```
  touch /test-userscript/userscript.txt
  echo "Created by bash shell script" >> /test-userscript/userscript.txt
--//--
```

インスタンスメタデータとユーザーデータ

インスタンスメタデータは、インスタンスに関するデータで、実行中のインスタンスを設定または管理するために使用します。インスタンスメタデータは、ホスト名、イベント、およびセキュリティグループなどでカテゴリ分けされます。

インスタンスメタデータを使用して、インスタンスの起動時に指定したユーザーデータにアクセスすることもできます。例えば、インスタンスを設定するためにパラメータを指定したり、単純なスクリプトを含めたりすることができます。汎用 AMI をビルドし、ユーザーデータを使って起動時に提供された設定ファイルを変更することができます。例えば、さまざまな小規模ビジネスを対象としたウェブサーバーを実行する場合に、すべてのサーバーで同じ汎用 AMI を使用し、起動時にユーザーデータで指定したAmazon S3バケットからコンテンツを取得できます。随時新規顧客を追加するには、顧客のバケットを作成し、そのコンテンツを追加し、ユーザーデータのコードに提供された固有のバケット名を使って AMI を起動します。同じ RunInstances 呼び出しを使用して複数のインスタンスを起動する場合、ユーザーデータはその予約においてすべてのインスタンスで使用可能です。同じリザベーションの一部である各インスタンスには固有の ami-launch-index 番号があるため、インスタンスが実行する操作を制御するコードを書くことができます。例えば、最初のホストはクラスター内の最初のノードとしてそれ自体を選択する場合があります。詳しい AMI 起動例については、例: AMI 作成インデックス値を参照してください。

EC2 インスタンスには、インスタンスの起動時に生成されるインスタンスアイデンティティドキュメントなどの動的データも含まれます。詳細については、動的データのカテゴリ を参照してください。

> **Important**
>
> インスタンスメタデータおよびユーザーデータにはそのインスタンス自体内からのみアクセスできるものの、データは認証または暗号化手法によって保護されていません。インスタンス、そしてインスタンス上で実行される任意のソフトウェアに対して直接アクセス権がある可能性がある人は、メタデータを表示できます。そのため、パスワードまたは存続期間の長い暗号化キーなどの機密データは、ユーザーデータとして保管しないようにしてください。

> **Note**
>
> このトピックの例では、インスタンスメタデータサービス (IMDS) の IPv4 アドレス `169.254.169.254` を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6 アドレスを有効にして使用してください。`[fd00:ec2::254]`。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

コンテンツ

IMDSv2 の使用

次のいずれかのメソッドを使って、実行中のインスタンスからインスタンスメタデータにアクセスできます。

- インスタンスメタデータサービスバージョン 1 (IMDSv1) – リクエスト/レスポンスメソッド
- インスタンスメタデータサービスバージョン 2 (IMDSv2) – セッション志向メソッド

デフォルトでは、IMDSv1またはIMDSv2のいずれか、あるいは両方を使用できます。

ローカルコードまたはユーザーに IMDSv2 を使用させるように、各インスタンスのインスタンスメタデータサービス (IMDS) を設定することができます。IMDSv2を使用しなければならないように指定すると、IMDSv1はもう機能しなくなります。ユーザーに IMDSv2 を使用させるようにインスタンスを設定する方法については、「インスタンスメタデータオプションの設定」を参照してください。

PUT または GET ヘッダーは IMDSv2 に固有のものです。これらのヘッダーがリクエストに含まれている場合、そのリクエストは IMDSv2 を対象としています。ヘッダーが存在しない場合、そのリクエストは IMDSv1 を対象としているものとみなされます。

IMDSv2 の拡張のレビューの詳細については、「EC2 Instance Metadata Service の拡張により、オープンファイアウォール、リバースプロキシ、および SSRF の脆弱性に対して多層防御を追加」を参照してください。

インスタンスメタデータを取得するには、「インスタンスメタデータの取得」を参照してください。

トピック

- インスタンスメタデータサービスバージョン 2 の仕組み
- インスタンスメタデータサービスバージョン 2 の使用への移行
- サポートされる AWS SDK を使用する

インスタンスメタデータサービスバージョン 2 の仕組み

IMDSv2 は、セッション指向リクエストを使用します。セッション指向リクエストを使用して、セッション期間 (1 秒～6 時間) を定義するセッショントークンを作成します。指定した期間中、それ以降のリクエストに同じセッショントークンを使用できます。指定した期間が期限切れになった後、将来のリクエストに使用する新しいセッショントークンを作成する必要があります。

> **Note**
>
> このセクションの例では、インスタンスメタデータサービス (IMDS) の IPv4 アドレス `169.254.169.254` を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6 アドレスを有効にして使用してください。`[fd00:ec2::254]`。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

次の例では、LinuxシェルスクリプトとIMDSv2を使って、最上位インスタンスメタデータアイテムを取得しています。例:

- PUTリクエストを使って、6 時間 (21,600 秒) のセッショントークンを作成する
- セッショントークンヘッダーをTOKENという名前の変数に保管する
- トークンを使って最上位メタデータアイテムをリクエストする

2 つの個別のコマンドを実行することも、それらを組み合わせることもできます。

個別のコマンド

最初に、次のコマンドを使用してトークンを生成します。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-
ec2-metadata-token-ttl-seconds: 21600"`
```

その後、次のコマンドを使用して、トークンを使用して上位レベルのメタデータアイテムを生成します。

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/
```

組み合わされたコマンド

トークンを保存し、コマンドを組み合わせることができます。次の例では、上記の 2 つのコマンドを組み合わせて、セッショントークンヘッダーを TOKEN という名前の変数に格納します。

> **Note**
>
> トークンの作成時にエラーが発生した場合は、有効なトークンの代わりにエラーメッセージが変数に格納され、コマンドは機能しません。

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-
ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
```

トークンを作成した後、期限切れになるまで再使用することができます。次のコマンド例では、インスタンスの起動に AMI の ID が使用されていますが、前の例で $TOKENに保管されたトークンが再使用されています。

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/ami-id
```

IMDSv2を使ってインスタンスメタデータをリクエストする際は、リクエストに次の項目が含まれている必要があります。

1. PUT リクエストを使って、インスタンスメタデータサービスに対してセッションを開始します。PUT リクエストは、インスタンスメタデータサービスに対する後続の GET リクエストに含まれている必要のあるトークンを返します。このトークンは、IMDSv2を使ってメタデータにアクセスするのに必要です。
2. トークンを IMDS に対するすべての GET リクエストに含めます。トークンの使用が `required` に設定されている場合、有効なトークンがないリクエスト、または有効期限切れのトークンを持つリクエストで `401 - Unauthorized` HTTP エラーコードが発生します。
   - トークンはインスタンス固有のキーです。トークンは他の EC2 インスタンスで有効ではなく、生成されたインスタンスの外で使用しようとすると拒否されます。
   - PUTリクエストには、トークンの有効期限 (TTL) を最大 6 時間 (21,600 秒) まで秒単位で指定するヘッダーが含まれている必要があります。トークンは論理的セッションを表します。TTL は、トークンが有効な時間の長さ、つまりセッションの期間を指定します。
   - トークンの期限が切れた後、インスタンスメタデータにアクセスし続けるためには、別のPUTを使って新しいセッションを作成する必要があります。
   - 各リクエストについてトークンを再使用するか、あるいは新しいトークンを作成することを選択できます。少数のリクエストでは、IMDS にアクセスする必要があるたびに、トークンを生成してすぐに使用するほうが簡単である場合があります。ただし、効率を重視するなら、インスタンスメタデータをリクエストする必要があるたびにPUTリクエストを書くより、トークン期間を長く指定して再使用することができます。それぞれが独自のセッションを表すトークンを同時に使用できる数については、実際的に制限がありません。ただし、IMDSv2 では、通常の IMDS 接続とスロットリングの制限によって制約を受けます。詳細については、「クエリスロットル」を参照してください。

HTTP GETおよびHEADメソッドはIMDSv2インスタンスメタデータリクエストで許可されています。PUT リクエストは、X-Forwarded-For ヘッダーが含まれている場合、拒否されます。

デフォルトで、PUT リクエストに対するレスポンスには IP プロトコルレベルで 1 のレスポンスホップリミット (有効期限) があります。より大きなホップリミットが必要な場合は、modify-instance-metadata-options AWS CLI コマンドを使って調整できます。例えば、インスタンスで実行されているコンテナサービスとの下位互換性のために、ホップリミットを拡大する必要がある場合があります。詳細については、「既存インスタンスのインスタンスメタデータオプションの変更」を参照してください。

インスタンスメタデータサービスバージョン 2 の使用への移行

IMDSv2 の使用に移行する場合、次のツールと移行パスを使用することが推奨されます。

トピック

IMDSv2 への移行に役立つツール

お使いのソフトウェアで IMDSv1 が使用されている場合、次のツールを使用して IMDSv2 を使用するようソフトウェアを再構成することができます。

AWS ソフトウェア

最新バージョンの AWS CLI および AWS SDK では、IMDSv2 をサポートしています。IMDSv2 を使用するには、EC2 インスタンスで、最新バージョンの CLI および SDK を使用している必要があります。CLI の更新の詳細については、 AWS Command Line Interface ユーザーガイドの、「AWS CLI のインストール、更新、およびアンインストール」を参照してください。

すべての Amazon Linux 2 と Amazon Linux 2023 ソフトウェアパッケージが IMDSv2 をサポートしています。Amazon Linux 2023 では、IMDSv1 はデフォルトで無効になっています。

IMDSv2 をサポートする最低限の AWS SDK バージョンについては、「サポートされる AWS SDK を使用する」を参照してください。

IMDS パケットアナライザー

IMDS パケットアナライザーは、インスタンスのブートフェーズからの IMDSv1 呼び出しを特定して記録するオープンソースツールです。このツールは、EC2 インスタンスで IMDSv1 呼び出しを行うソフトウェアを特定するのに役立ち、インスタンスで IMDSv2 のみを使用できるようにするために何を更新する必要があるかを正確に特定できます。IMDS パケットアナライザーは、コマンドラインから実行することも、サービスとしてインストールすることもできます。詳細については、GitHub の「IMDS パケットアナライザー」を参照してください

CloudWatch

IMDSv2 では、IMDSv1 がサポートしていない、トークンベースのセッションが利用できます。`MetadataNoToken` CloudWatch メトリクスは、IMDSv1 を使用しているインスタンスメタデータサービス (IMDS) への呼び出しの数を追跡します。このメトリクスをゼロまでトラッキングすることにより、すべてのソフトウェアが IMDSv2 を使用するようアップグレードされたかどうか、およびいつアップデートが行われたかを測定できます。

IMDSv1 を無効にした後、`MetadataNoTokenRejected` CloudWatch メトリクスを使用して、IMDSv1 呼び出しが試行および拒否された回数を追跡できます。このメトリクスを追跡す

ることで、IMDSv2 を使用するようにソフトウェアを更新する必要があるかどうかを確認できます。

詳細については、「インスタンスメトリクス」を参照してください。

EC2 API および CLI への更新

新しいインスタンスについては、RunInstances API を使用して、IMDSv2 の使用を義務付ける新しいインスタンスを起動できます。詳細については、「新規インスタンスのインスタンスメタデータオプションの設定」を参照してください。

既存のインスタンスの場合、ModifyInstanceMetadataOptions API を使用して IMDSv2 の使用を要求できます。詳細については、「既存インスタンスのインスタンスメタデータオプションの変更」を参照してください。

Auto Scaling グループによって起動されたすべての新しいインスタンスで IMDSv2 の使用を必須にするために、Auto Scaling グループは起動テンプレートまたは起動設定を使用できます。起動テンプレートの作成時や起動設定の作成時に、IMDSv2 の使用が必須となるように `MetadataOptions` パラメータを設定する必要があります。Auto Scaling グループは、新しい起動テンプレートまたは起動設定を使用して新しいインスタンスは新しいインスタンスを起動しますが、既存のインスタンスは影響を受けません。Auto Scaling グループ内の既存のインスタンスの場合、ModifyInstanceMetadataOptions API を使用して、既存のインスタンスで IMDSv2 の使用を要求するか、インスタンスを終了すると、Auto Scaling グループは、新しい起動テンプレートまたは起動設定で定義されているインスタンスメタデータオプション設定で新しい置き換えインスタンスを起動します。

デフォルトで IMDSv2 を設定する AMI を使用する

`ImdsSupport` パラメータに `v2.0` を設定した AMI により、インスタンスを (`HttpTokens` パラメータに `required` を設定して) 起動する場合は、デフォルトで IMDSv2 を使用するように自動的に設定できます。register-image CLI コマンドを使用して AMI を登録するときに `ImdsSupport` パラメータを `v2.0` に設定することも、modify-image-attribute CLI コマンドを使用して既存の AMI を変更することもできます。詳細については、「AMI を設定する」を参照してください。

IAM ポリシーおよび SCP

以下に示すように、ユーザーの管理には、IAM ポリシーを使用することも、AWS Organizations サービスコントロールポリシー (SCP) を使用することもできます。

- インスタンスが IMDSv2 を使用するように設定されていない限り、RunInstances API を使用してそのインスタンスを起動することはできません。

- IMDSv1 を再度有効にするために、ModifyInstanceMetadataOptions API を使用して実行中のインスタンスを変更することはできません。

IAM ポリシーまたは SCP には、次の IAM 条件キーを含める必要があります。

- `ec2:MetadataHttpEndpoint`
- `ec2:MetadataHttpPutResponseHopLimit`
- `ec2:MetadataHttpTokens`

条件キーが含まれているポリシーで指定した状態と、API および CLI 呼び出し時のパラメータが一致しない場合、これらの API または CLI の呼び出しは失敗し `UnauthorizedOperation` レスポンスが返されます。

さらに、追加の保護レイヤーを選択して、IMDSv1からIMDSv2の変更を強制することもできます。EC2 ロールの認証情報経由でコールされた各 API に関するアクセス管理レイヤーでは、IAM ポリシーまたは AWS Organizations サービスコントロールポリシー (SCP) で新しい条件キーを使用できます。具体的には、IAM ポリシーで値 `2.0` を設定した条件キー `ec2:RoleDelivery` を使用していると、IMDSv1 から取得した EC2 ロールの認証情報を使用した API コールに対して、`UnauthorizedOperation` レスポンスが返されます。同じことは、SCP によって義務付けられる条件を使ってより広く達成できます。これにより、指定した条件と一致しない API コールに対しては `UnauthorizedOperation` エラーが返されるため、実際に IMDSv1 から取得した認証情報を使用して API を呼び出すことはできなくなります。

IAM ポリシーの例は、インスタンスメタデータの使用を参照してください。SCP の詳細については、「AWS Organizations ユーザーガイド」の「サービスコントロールポリシー (SCPs)」を参照してください。

IMDSv2 を必要とする推奨パス

上記のツールを使用する際、IMDSv2 への移行にこのパスに従うことを推奨します。

ステップ 1: 開始時

EC2 インスタンスのロール認証情報を使用する SDK、CLI、およびソフトウェアを、IMDSv2 対応のバージョンに更新します。CLI の更新に関する情報については、AWS Command Line Interface ユーザーガイドの「AWS CLI の最新バージョンへのアップグレード」を参照してください。

次に、IMDSv2 リクエストを使ってインスタンスメタデータに直接アクセスする (つまり、SDK を使用しない) ソフトウェアを変更します。IMDS パケットアナライザーを使用して、IMDSv2 リクエストを使用するために変更する必要があるソフトウェアを特定できます。

ステップ 2: 移行の進行状況を追跡する

CloudWatch の `MetadataNoToken` メトリクスを使用して、移行の進行状況を追跡します。このメトリクスは、インスタンスの IMDS に対する IMDSv1 呼び出しの数を示します。詳細については、「インスタンスメトリクス」を参照してください。

ステップ 3: IMDSv1 をまったく使用していない場合

CloudWatch メトリクス `MetadataNoToken` で記録される IMDSv1 の使用率がゼロであれば、そのインスタンスは IMDSv2 の使用に完全に移行するための準備が整っています。この段階で、次の操作を実行できます。

- アカウントのデフォルト

  IMDSv2 をアカウントのデフォルトとして必須に設定できます。インスタンスが起動すると、インスタンス設定は自動的にアカウントのデフォルトに設定されます。

  アカウントのデフォルトを設定するには、次の手順を実行します。

  - Amazon EC2 コンソール: EC2 ダッシュボードの [アカウントの属性]、[データ保護とセキュリティ] で、[IMDS のデフォルト] に対して、[インスタンスメタデータサービス] を [有効] に設定し、[メタデータのバージョン] を [V2 のみ (トークンは必須)] に設定します。詳細については、「IMDSv2 をアカウントのデフォルトとして設定する」を参照してください。
  - AWS CLI: modify-instance-metadata-defaults CLI コマンドを使用して、`--http-tokens required` と `--http-put-response-hop-limit 2` を指定します。
- 新規のインスタンス

  新しいインスタンスを起動する際には、以下のいずれかを実行できます。

  - Amazon EC2 コンソール: インスタンス起動ウィザードで、[Metadata accessible] (メタデータにアクセス可能) を [Enabled] (有効) に、[Metadata version] (メタデータバージョン) を [V2 only (token required)] (V2 のみ (トークンが必須)) に設定します。詳細については、「起動時にインスタンスを設定する」を参照してください。
  - AWS CLI: run-instances CLI コマンドを使用して、IMDSv2 が必須となるように指定します。
- 既存のインスタンス

  既存のインスタンスには、次の操作を実行できます。

  - Amazon EC2 コンソール: [インスタンス] ページでインスタンスを選択し、[アクション]、[インスタンス設定]、[インスタンスメタデータオプションの変更] を選択し、[IMDSv2] の場合は [必須] を選択します。詳細については、「IMDSv2 の使用を要求する」を参照してください。

- AWS CLI: IMDSv2 のみを使用するように指定するには、modify-instance-metadata-options CLI コマンドを使用します。

実行中のインスタンスで、インスタンスメタデータオプションを変更でき、インスタンスメタデータオプションを変更した後にインスタンスを再起動する必要はありません。

手順 4: すべてのインスタンスが IMDSv2 に移行されたかどうか確認する

IMDSv2 の使用を要求するようにまだ設定されていないインスタンスがないか、つまり、IMDSv2 がまだ optional として設定されているかどうかを確認できます。まだインスタンスが optional として設定されている場合は、前の手順 3 を繰り返し、インスタンスのメタデータオプションを変更して IMDSv2 required を作成できます。

インスタンスをフィルターするには

- Amazon EC2 コンソール: [インスタンス] ページで、[IMDSv2 = optional] フィルターを使用してインスタンスをフィルタリングします。のフィルタリングについての詳細は、「コンソールを使用したリソースのフィルタリング」を参照してください。また、各インスタンスで IMDSv2 が必須かオプションかを確認することもできます。[基本設定] ウィンドウで [IMDSv2] を切り替えて、[IMDSv2] 列を [インスタンス] テーブルに追加します。
- AWS CLI: describe-instances CLI コマンドを使用して、次のように metadata-options.http-tokens = optional よってフィルタリングします。

```
aws ec2 describe-instances --filters "Name=metadata-options.http-
tokens,Values=optional" --query "Reservations[*].Instances[*].[InstanceId]" --output
 text
```

手順 5: すべてのインスタンスが IMDSv2 に移行された時点

IAM 条件キーの ec2:MetadataHttpTokens、ec2:MetadataHttpPutResponseHopLimit、および ec2:MetadataHttpEndpoint により、RunInstances と ModifyInstanceMetadataOptions API、および対応する CLI の使用をコントロールできます。ポリシーを作成し、条件キーを使用してポリシーに指定した状態と API コールのパラメータが一致しない場合、API コールまたは CLI コールは失敗して UnauthorizedOperation レスポンスが返されます。IAM ポリシーの例は、インスタンスメタデータの使用を参照してください。

さらに、IMDSv1 を無効にした後、MetadataNoTokenRejected CloudWatch メトリクスを使用して、IMDSv1 呼び出しが試行および拒否された回数を追跡できます。IMDSv1 を無効にした後、正常

に動作していないソフトウェアがあり、MetadataNoTokenRejected メトリクスに IMDSv1 呼び出しが記録されている場合は、IMDSv2 を使用するようにこのソフトウェアを更新する必要がある可能性があります。

サポートされる AWS SDK を使用する

IMDSv2 を使用するには、EC2 インスタンスが IMDSv2 の使用をサポートする AWS SDK バージョンを使用する必要があります。最新バージョンの AWS SDK はすべて IMDSv2 の使用をサポートしています。

> **Important**
>
> 最新の機能、セキュリティアップデート、および基本的な依存関係を維持するために、SDK のリリースを常に更新することをお勧めします。サポート対象外の SDK バージョンを継続して使用することはお勧めできません。お客様の判断で行ってください。詳細については、「AWS SDK とツールのリファレンスガイド」の「AWS SDK とツールのメンテナンスポリシー」を参照してください。

IMDSv2 の使用をサポートする最小バージョンは次のとおりです。

- AWS CLI – 1.16.289
- AWS Tools for Windows PowerShell – 4.0.1.0
- AWS SDK for .NET – 3.3.634.1
- AWS SDK for C++ – 1.7.229
- AWS SDK for Go – 1.25.38
- AWS SDK for Go v2 – 0.19.0
- AWS SDK for Java – 1.11.678
- AWS SDK for Java 2.x – 2.10.21
- AWS Node.js 内の SDK for JavaScript – 2.722.0
- AWS SDK for PHP – 3.147.7
- AWS SDK for Python (Boto) – 1.13.25
- AWS SDK for Python (Boto3) – 1.12.6
- AWS SDK for Ruby – 3.79.0

インスタンスメタデータオプションの設定

インスタンスメタデータサービス (IMDS) は、すべての EC2 インスタンスでローカルに実行されます。インスタンスメタデータオプションは、EC2 インスタンス上の IMDS のアクセシビリティと動作を制御する一連の設定を参照します。

各インスタンスで、以下のインスタンスメタデータオプションを設定できます。

[インスタンスメタデータサービス (IMDS)]: `enabled` | `disabled`

インスタンスで IMDS を有効または無効にすることができます。無効にすると、ユーザーまたはコードはインスタンスのインスタンスメタデータにアクセスできなくなります。

IMDS のインスタンスには、IPv4 (`169.254.169.254`) と IPv6 (`[fd00:ec2::254]`) という 2 つのエンドポイントがあります。IMDS を有効にすると、IPv4 エンドポイントが自動的に有効になります。IPv6 エンドポイントを有効にする場合は、明示的に有効にする必要があります。

[IMDS IPv6 エンドポイント]: `enabled` | `disabled`

インスタンスで IPv6 IMDS エンドポイントを明示的に有効にできます。IPv6 エンドポイントが有効になっている場合、IPv4 エンドポイントは有効なままになります。IPv6 エンドポイントは、Nitro System 上に構築されたインスタンス でのみサポートされています。

[メタデータのバージョン]: `IMDSv1 or IMDSv2 (token optional)` | `IMDSv2 only (token required)`

インスタンスメタデータをリクエストするとき、IMDSv2 呼び出しはトークンを要求します。IMDSv1 呼び出はトークンを要求しません。IMDSv1 または IMDSv2 呼び出しを許可する (トークンがオプションの場合) か、IMDSv2 呼び出しのみを許可する (トークンが必須の場合) ように、インスタンスを設定できます。

[メタデータレスポンスのホップ制限]: `1 ~ 64`

ホップ制限は、PUT レスポンスが実行できるネットワークホップの数です。ホップ制限は、最小 1、最大 64 に設定できます。コンテナ環境では、ホップ制限を 2 に設定することをお勧めします。詳細については、「考慮事項」を参照してください。

[インスタンスメタデータ内のタグにアクセスする]: `enabled` | `disabled`

インスタンスのメタデータからインスタンスのタグへのアクセスを有効または無効にすることができます。詳細については、「インスタンスメタデータ内のインスタンスタグの使用」を参照してください。

インスタンスメタデータオプションを設定する場所

インスタンスメタデータオプションは、次のようにさまざまなレベルで設定できます。

- アカウント – 各 AWS リージョンのアカウントレベルで、インスタンスメタデータオプションのデフォルト値を設定できます。インスタンスが起動すると、インスタンスメタデータオプションは自動的にアカウントレベルの値に設定されます。これらの値は、起動時に変更できます。アカウントレベルのデフォルト値は、既存のインスタンスには影響しません。
- AMI – AMI を登録または変更するときに、`imds-support` パラメータを `v2.0` に設定できます。この AMI でインスタンスを起動すると、インスタンスメタデータバージョンは自動的に「IMDSv2」に設定され、ホップ制限は「2」に設定されます。
- インスタンス – 起動時にインスタンスのすべてのインスタンスメタデータオプションを変更し、デフォルト設定を上書きできます。実行中または停止中のインスタンスで起動した後に、インスタンスメタデータオプションを変更することもできます。変更は IAM または SCP ポリシーによって制限される可能性があることに注意してください。

詳細については、新規インスタンスのインスタンスメタデータオプションの設定および既存インスタンスのインスタンスメタデータオプションの変更を参照してください。

インスタンスメタデータオプションの優先順位

各インスタンスメタデータオプションの値は、インスタンスの起動時に優先順位に従って決定されます。最上位の優先順位を持つ階層は次のとおりです。

- 優先順位 1: 起動時のインスタンス設定 – 値は、起動テンプレートまたはインスタンス設定のいずれかで指定できます。ここで指定された値は、アカウントレベルまたは AMI で指定された値を上書きします。
- 優先順位 2: アカウント設定 – インスタンスの起動時に値が指定されていない場合は、アカウントレベルの設定 (AWS リージョンごとに設定) によって値が決まります。アカウントレベルの設定では、各メタデータオプションの値が含まれているか、まったく指定がないことが示されるかのどちらかです。
- 優先順位 3: AMI 設定 – インスタンスの起動時に、またはアカウントレベルで値が指定されていない場合は、AMI 設定によって値が決まります。これは、`HttpTokens` と `HttpPutResponseHopLimit` にのみ該当します。

各メタデータオプションは個別に評価されます。インスタンスは、直接インスタンス設定、アカウントレベルのデフォルト、および AMI からの設定を組み合わせて設定できます。

IAM または SCP ポリシーによって変更が制限されていない限り、実行中または停止中のインスタンスで起動した後に、任意のメタデータオプションの値を変更できます。

メタデータオプションの値を決定する – 例 1

この例では、EC2 インスタンスは、アカウントレベルで HttpPutResponseHopLimit が 1 に設定されているリージョンで起動されます。指定された AMI では、ImdsSupport が v2.0 に設定されています。起動時に、インスタンスでメタデータオプションが直接指定されることはありません。インスタンスは、次のメタデータオプションを使用して起動されます。

```
"MetadataOptions": {
    ...
    "HttpTokens": "required",
    "HttpPutResponseHopLimit": 2,
    ...
```

これらの値は次のように決定されました。

- 起動時にメタデータオプションが指定されていない: インスタンスの起動時に、メタデータオプションの特定の値が、インスタンス起動パラメータにも、起動テンプレートにも指定されていませんでした。
- アカウント設定が次に優先される: 起動時に特定の値が指定されていない場合は、リージョン内のアカウントレベルの設定が優先されます。つまり、アカウントレベルで設定されたデフォルト値が適用されます。この場合は、HttpPutResponseHopLimit が 1 に設定されました。
- AMI 設定が最後に優先される: 起動時に、またはアカウントレベル設定で特定の値が指定されていない場合は、AMI の設定が優先されます。この場合は、AMI 設定 ImdsSupport: v2.0 により、HttpTokens メタデータオプションが required に設定されました。

メタデータオプションの値を決定する – 例 2

この例では、EC2 インスタンスは前の例 1 と同じ設定で起動されますが、起動時にインスタンスで直接 HttpTokens が optional に設定されています。インスタンスは、次のメタデータオプションを使用して起動されます。

```
"MetadataOptions": {
    ...
    "HttpTokens": "optional",
    "HttpPutResponseHopLimit": 1,
    ...
```

HttpPutResponseHopLimit の値は、例 1 と同じ方法で決定されます。ただし、HttpTokens の値は次のように決定されます。起動時にインスタンスで設定されているメタデータオプションが最優先されます。AMI が ImdsSupport: v2.0 で設定されている (つまり、HttpTokens が required に設定されている) 場合でも、起動時にインスタンスで指定されている値 (HttpTokens が optional に設定されている) が優先されます。

インスタンスのメタデータバージョンを設定する

インスタンスが起動すると、インスタンスのメタデータバージョンの値は IMDSv1 or IMDSv2 (token optional) または IMDSv2 only (token required) になります。

インスタンスの起動時に、メタデータバージョンの値を手動で指定するか、デフォルト値を使用できます。値を手動で指定すると、すべてのデフォルト値が上書きされます。値を手動で指定しないことを選択した場合は、次の表に示すように、デフォルト設定の組み合わせによって値が決定されます。

この表は、起動時のインスタンスのメタデータバージョン (列 4 の [結果として生じるインスタンス設定] で示される) が、さまざまなレベルの設定によってどのように決定されるかを示しています。優先順位は左から右で、次のように最初の列が最も優先されます。

- 列 1: [起動パラメータ] - 起動時に手動で指定するインスタンスの設定を表します。
- 列 2: [アカウントレベルのデフォルト] - アカウントの設定を表します。
- 列 3: [AMI のデフォルト] - AMI の設定を表します。

起動パラメータ	アカウントレベルのデフォルト	AMI のデフォルト	結果として生じるインスタンス設定
V2 のみ (トークンが必要)	指定なし	V2 のみ	V2 のみ
V2 のみ (トークンが必要)	V2 のみ	V2 のみ	V2 のみ
V2 のみ (トークンが必要)	V1 または V2	V2 のみ	V2 のみ
V1 または V2 (トークンはオプション)	指定なし	V2 のみ	V1 または V2

起動パラメータ	アカウントレベルのデフォルト	AMI のデフォルト	結果として生じるインスタンス設定
V1 または V2 (トークンはオプション)	V2 のみ	V2 のみ	V1 または V2
V1 または V2 (トークンはオプション)	V1 または V2	V2 のみ	V1 または V2
未設定	指定なし	V2 のみ	V2 のみ
未設定	V2 のみ	V2 のみ	V2 のみ
未設定	V1 または V2	V2 のみ	V1 または V2
V2 のみ (トークンが必要)	指定なし	null	V2 のみ
V2 のみ (トークンが必要)	V2 のみ	null	V2 のみ
V2 のみ (トークンが必要)	V1 または V2	null	V2 のみ
V1 または V2 (トークンはオプション)	指定なし	null	V1 または V2
V1 または V2 (トークンはオプション)	V2 のみ	null	V1 または V2
V1 または V2 (トークンはオプション)	V1 または V2	null	V1 または V2
未設定	指定なし	null	V1 または V2
未設定	V2 のみ	null	V2 のみ
未設定	V1 または V2	null	V1 または V2

IAM 条件キーを使用してインスタンスメタデータオプションを制限する

IAM ポリシーまたは SCP で IAM 条件キーを次のように使用できます。

- IMDSv2 の使用を要求するようにインスタンスが設定されている場合にのみ、インスタンスの起動を許可する
- ホップの許可数を制限する
- インスタンスメタデータへのアクセスを無効にする

タスク

- 新規インスタンスのインスタンスメタデータオプションの設定
- 既存インスタンスのインスタンスメタデータオプションの変更

> **Note**
>
> 注意深く実行し、変更を行う前に慎重なテストを実施する必要があります。以下の情報を記録します。
>
> - IMDSv2の使用を強制する場合、インスタンスメタデータアクセスのためにIMDSv1を使用するアプリケーションまたはエージェントは休憩します。
> - インスタンスメタデータへのアクセスをすべてオフにする場合、インスタンスメタデータアクセスに依存して機能するアプリケーションまたはエージェントは休憩します。
> - IMDSv2 でトークンを取得する際には、`/latest/api/token` を使用する必要があります。

新規インスタンスのインスタンスメタデータオプションの設定

以下のインスタンスメタデータオプションを設定できます。

オプション

- IMDSv2 の使用を要求する
- IPv4 および IPv6 エンドポイントの設定
- インスタンスメタデータへのアクセスを無効にする

IMDSv2 の使用を要求する

次の方法を使用して、インスタンスで IMDSv2 の使用を必須にすることができます。

IMDSv2 を必須にするには

- IMDSv2 をアカウントのデフォルトとして設定する
- 起動時にインスタンスを設定する
- AMI を設定する
- IAM ポリシーを使用する

IMDSv2 をアカウントのデフォルトとして設定する

デフォルトのインスタンスメタデータバージョンは、各 AWS リージョンのアカウントレベルで設定できます。インスタンスが起動すると、インスタンスメタデータバージョンは自動的にアカウントレベルの値に設定されます。

アカウントレベルのデフォルトを変更したことがない場合は、設定がないことが示されます。

インスタンスメタデータバージョンのアカウントのデフォルトを IMDSv2 に設定して、アカウント内のすべての新しいインスタンスを IMDSv2 で起動できます (つまり、IMDSv1 は無効になっています)。このアカウントデフォルトでは、インスタンスを起動すると、インスタンスのデフォルト値は次のようになります。

- コンソール: [メタデータのバージョン] は [V2 のみ (トークンは必須)] に設定され、[メタデータレスポンスのホップ制限] は [2] に設定されます。
- AWS CLI: `HttpTokens` は `required` に設定され、`HttpPutResponseHopLimit` は `2` に設定されます。

> **Note**
>
> [メタデータのバージョン] のアカウントデフォルトを [V2 のみ (トークンは必須)] に設定する前に、どのインスタンスも IMDSv1 呼び出しを行っていないことを確認してください。`MetadataNoToken` CloudWatch メトリクスは IMDSv1 呼び出しを追跡します。`MetadataNoToken` で記録される IMDSv1 の使用率がゼロであれば、そのインスタンスは IMDSv2 の使用に完全に移行するための準備が整っています。

起動時に、インスタンス設定の値を変更できます。詳細については、「インスタンスのメタデータバージョンを設定する」を参照してください。

Console

指定したリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. AWS リージョン を変更するには、ページの右上隅にあるリージョンセレクターを使用します。
3. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
4. [アカウントの属性] で [データ保護とセキュリティ] を選択します。
5. [IMDS のデフォルト] の横にある [管理] を選択します。
6. [IMDS のデフォルトを管理] ページで、次の操作を実行します。
   a. [インスタンスメタデータサービス] で、[有効にする] を選択します。
   b. [Metadata version] (メタデータバージョン) には、[V2 only (token required)] (V2 のみ (トークンが必要)) を選択します。
   c. インスタンスがコンテナをホストする場合は、[メタデータレスポンスのホップ制限] で 2 を指定します。それ以外の場合は、[設定なし] を選択します。設定なしが指定されているとき、AMI が IMDSv2 を必要とする場合は、起動時の値がデフォルトで [2] になります。それ以外の場合は、デフォルトで [1] になります。
   d. [Update] (更新) を選択します。

AWS CLI

指定したリージョンのアカウントのデフォルトとして IMDSv2 を設定するには

modify-instance-metadata-defaults コマンドを使用して、IMDS アカウントレベルの設定を変更するリージョンを指定します。インスタンスがコンテナをホストする場合は、`--http-tokens` を `required` に、`--http-put-response-hop-limit` を 2 に設定します。それ以外の場合は、`-1` を指定して、設定がないことを示します。`-1` (設定なし) が指定されているとき、AMI が IMDSv2 を必要とする場合は、起動時の値がデフォルトで 2 になります。それ以外の場合は、デフォルトで 1 になります。

```
aws ec2 modify-instance-metadata-defaults \
    --region us-east-1 \
```

```
    --http-tokens required \
    --http-put-response-hop-limit 2
```

正常な出力

```
{
    "Return": true
}
```

指定したリージョンのインスタンスメタデータオプションのデフォルトのアカウント設定を表示するには

get-instance-metadata-defaults コマンドを使用して、リージョンを指定します。

```
aws ec2 get-instance-metadata-defaults --region us-east-1
```

出力例

```
{
    "AccountLevel": {
        "HttpTokens": "required",
        "HttpPutResponseHopLimit": 2
    }
}
```

起動時にインスタンスを設定する

インスタンスを起動する際に、以下のフィールドを設定しておくことで、IMDSv2 が使用されるようにそのインスタンスを構成できます。

- Amazon EC2 コンソール: [Metadata version] (メタデータバージョン) で、[V2 only (token required)] (V2 のみ (トークンが必須)) を設定します。
- AWS CLI: `HttpTokens` に `required` を設定します。

IMDSv2 が必須であることを指定する場合、[メタデータにアクセス可能] に [有効] (コンソールの場合) を設定するか、`HttpEndpoint` に `enabled` (AWS CLI の場合) を設定して、インスタンスメタデータサービス (IMDS) のエンドポイントも有効にする必要があります。

New console

新しいインスタンスで IMDSv2 の使用を要求するには

- Amazon EC2 コンソールで新しいインスタンスを起動するとき、[Advanced details] (高度な詳細) を展開し、次の操作を行います。
  - [Metadata accessible] (メタデータにアクセス可能) には、[Enabled] (有効) を選択します。
  - [Metadata version] (メタデータバージョン) には、[V2 only (token required)] (V2 のみ (トークンが必要)) を選択します。

詳細については、「高度な詳細」を参照してください。

Old console

新しいインスタンスで IMDSv2 の使用を要求するには

- Amazon EC2 コンソールで新しいインスタンスを起動するとき、[Configure Instance Details (インスタンスの詳細の設定)] ページで次のオプションを選択します。
  - [Advanced Details (高度な詳細)] の [Metadata accessible (メタデータにアクセス可能)] で、[Enabled (有効)] を選択します。
  - [Metadata version (メタデータバージョン)] で、[V2 (token required) (V2 (トークンが必要))] を選択します。

詳細については、「ステップ 3: インスタンスの詳細を設定する」を参照してください。

AWS CLI

新しいインスタンスで IMDSv2 の使用を要求するには

次の run-instances の例では、`c6i.large` を `--metadata-options` に設定して `HttpTokens=required` インスタンスを起動します。`HttpTokens` の値を指定する場合は、`HttpEndpoint` も `enabled` に設定する必要があります。メタデータの取得リクエストでは、セキュリティで保護されたトークンヘッダーは `required` に設定されるので、インスタンスメタデータのリクエストに際しては、そのインスタンスは必ず IMDSv2 を使用することになります。

```
aws ec2 run-instances \
```

```
    --image-id ami-0abcdef1234567890 \
    --instance-type c6i.large \
...
    --metadata-options "HttpEndpoint=enabled,HttpTokens=required"
```

PowerShell

新しいインスタンスで IMDSv2 の使用を要求するには

次の New-EC2Instance コマンドレットの例では、MetadataOptions_HttpEndpoint を enabled に、MetadataOptions_HttpTokens パラメータを required に設定して c6i.large インスタンスを起動します。HttpTokens の値を指定する場合は、HttpEndpoint も enabled に設定する必要があります。メタデータの取得リクエストでは、セキュリティで保護されたトークンヘッダーは required に設定されるので、インスタンスメタデータのリクエストに際しては、そのインスタンスは必ず IMDSv2 を使用することになります。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType c6i.large `
    -MetadataOptions_HttpEndpoint enabled `
    -MetadataOptions_HttpTokens required
```

AWS CloudFormation

AWS CloudFormation を使用してインスタンスのメタデータオプションを指定するには、「AWS CloudFormation ユーザーガイド」の「AWS::EC2::LaunchTemplate MetadataOptions」プロパティを参照してください。

AMI を設定する

新しい AMI を登録したり、既存の AMI を変更したりするときに、imds-support パラメータを v2.0 に設定できます。この AMI から起動されたインスタンスでは、[Metadata version] (メタデータバージョン) に V2 only (token required)] (V2 のみ (トークンが必要)) (コンソールの場合) が設定されるか、HttpTokens に required (AWS CLI の場合) が設定されます。この設定が行われている場合、インスタンスメタデータがリクエストされる際には IMDSv2 を使用することが、インスタンスでの必須になります。

この AMI から起動されるインスタンスでは、imds-support に v2.0 を設定している場合、[Metadata response hop limit] (メタデータレスポンスのホップ制限) (コンソールの場合)、または

`http-put-response-hop-limit` (AWS CLI の場合) が「2」に設定されることに注意してください。

> **Important**
>
> ご使用の AMI ソフトウェアが IMDSv2 をサポートしていない限りは、このパラメータを使用しないでください。値を `v2.0` に設定すると、元に戻すことはできません。AMI を「リセット」する唯一の方法は、基礎となるスナップショットから新しい AMI を作成することです。

IMDSv2 向けに AMI を新たに設定するには

新しい AMI IMDSv2 を設定するには、次のいずれかの方法を使用します。

AWS CLI

以下の register-image の例では、EBS ルートボリュームの指定されたスナップショットをデバイス `/dev/xvda` として使用して、AMI を登録しています。`imds-support` パラメータ用に `v2.0` を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
aws ec2 register-image \
    --name my-image \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/
xvda,Ebs={SnapshotId=snap-0123456789example} \
    --architecture x86_64 \
    --imds-support v2.0
```

PowerShell

次の Register-EC2Image コマンドレットの例では、EBS ルートボリュームの指定されたスナップショットをデバイス `/dev/xvda` として使用して、AMI を登録しています。`ImdsSupport` パラメータ用に `v2.0` を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
Import-Module AWS.Tools.EC2  # Required for Amazon.EC2.Model object creation.
Register-EC2Image `
    -Name 'my-image' `
```

```
    -RootDeviceName /dev/xvda `
    -BlockDeviceMapping  (
    New-Object `
        -TypeName Amazon.EC2.Model.BlockDeviceMapping `
        -Property @{
        DeviceName = '/dev/xvda';
        EBS        = (New-Object -TypeName Amazon.EC2.Model.EbsBlockDevice -Property
@{
                SnapshotId = 'snap-0123456789example';
                VolumeType = 'gp3'
                } )
        }  ) `
    -Architecture X86_64 `
    -ImdsSupport v2.0
```

IMDSv2 向けに既存の AMI を設定するには

IMDSv2 向けに既存の AMI を設定するには、次のいずれかの方法を使用します。

AWS CLI

次の modify-image-attribute の例では、IMDSv2 用の既存の AMI のみを変更します。imds-support パラメータ用に v2.0 を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
aws ec2 modify-image-attribute \
    --image-id ami-0123456789example \
    --imds-support v2.0
```

PowerShell

次の Edit-EC2ImageAttribute コマンドレットの例では、IMDSv2 用の既存の AMI のみを変更します。imds-support パラメータ用に v2.0 を指定し、この AMI から起動するインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 を使用することが、この AMI から起動されるインスタンスでの必須になります。

```
Edit-EC2ImageAttribute `
    -ImageId ami-0abcdef1234567890 `
    -ImdsSupport 'v2.0'
```

IAM ポリシーを使用する

IMDSv2 の使用が必須ではない新しいインスタンスをユーザーが起動できないように、IAM ポリシーを作成することもできます。

IAM ポリシーにより、すべての新しいインスタンスでの IMDSv2 の使用を必須にするには

ユーザーがインスタンスメタデータをリクエストする際に IMDSv2 の使用を義務付けるインスタンスみを起動できるようにするには、IMDSv2 を必要とする条件が満たされないとインスタンスを起動できないように指定することができます。IAM ポリシーの例については、「インスタンスメタデータの使用」を参照してください。

IPv4 および IPv6 エンドポイントの設定

デフォルトでは、IPv6 エンドポイントは無効です。これは、IPv6 専用サブネットでインスタンスを起動する場合にも当てはまります。インスタンスの起動時に IPv6 エンドポイントを有効にできます。

IMDS の IPv6 エンドポイントは、Nitro System 上に構築されたインスタンス 上でのみアクセスできます。

IMDS で IPv6 エンドポイントを有効にしてインスタンスを起動するには、以下のいずれかの方法を使用します。

New console

起動時に IMDS IPv6 エンドポイントを有効にするには

- [Advanced details] (高度な詳細) で以下のように指定して、Amazon EC2 コンソールでインスタンスを起動します。
  - [メタデータの転送] では [有効] を選択します。

詳細については、「高度な詳細」を参照してください。

AWS CLI

以下の run-instances の例では、IMDS 用に IPv6 エンドポイントが有効化された、`c6i.large` インスタンスを起動しています。IPv6 エンドポイントを有効にするには、`--metadata-options` パラメータに `HttpProtocolIpv6=enabled` を指定します。`HttpProtocolIpv6` の値を指定する場合は、`HttpEndpoint` も `enabled` に設定する必要があります。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c6i.large \
    ...
    --metadata-options "HttpEndpoint=enabled,HttpProtocolIpv6=enabled"
```

PowerShell

次の New-EC2Instance コマンドレットの例では、IMDS 用に IPv6 エンドポイントが有効化された、c6i.large インスタンスを起動しています。IPv6 エンドポイントを有効にするには、MetadataOptions_HttpProtocolIpv6 を enabled に指定します。MetadataOptions_HttpProtocolIpv6 の値を指定する場合は、MetadataOptions_HttpEndpoint も enabled に設定する必要があります。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType c6i.large `
    -MetadataOptions_HttpEndpoint enabled `
    -MetadataOptions_HttpProtocolIpv6 enabled
```

インスタンスメタデータへのアクセスを無効にする

インスタンスを起動するときに IMDS を無効にすることで、インスタンスのメタデータへのアクセスを無効にできます。IMDS を再度有効にすると、その後でアクセスを有効にできます。詳細については、「インスタンスメタデータへのアクセスを有効にする」を参照してください。

> **Important**
>
> IMDS は起動時または起動後に無効化できます。起動時に IMDS を無効にすると、以下が機能しなくなる可能性があります。
>
> - インスタンスへの SSH アクセスがない可能性があります。キーは通常 EC2 インスタンスのメタデータから提供され、アクセスされるため、インスタンスのパブリック SSH キーである public-keys/0/openssh-key にはアクセスできません。
> - EC2 ユーザーデータは利用できず、インスタンスの起動時には実行されません。EC2 ユーザーデータは IMDS でホストされます。IMDS を無効にすると、ユーザーデータへのアクセスは事実上無効になります。

この機能にアクセスするには、起動後に IMDS を再度有効にします。

New console

起動時にインスタンスメタデータへのアクセスを無効にするには

- [Advanced details] (高度な詳細) で以下のように指定して、Amazon EC2 コンソールでインスタンスを起動します。
  - [Metadata accessible] (メタデータにアクセス可能) には、[Disabled] (無効) を選択します。

詳細については、「高度な詳細」を参照してください。

Old console

起動時にインスタンスメタデータへのアクセスを無効にするには

- [Configure Instance Details] (インスタンスの詳細の設定) ページで次のオプションを選択して、Amazon EC2 コンソールでインスタンスを起動します。
  - [Advanced Details (高度な詳細)] の [Metadata accessible (メタデータにアクセス可能)] で、[Disabled (無効)] を選択します。

詳細については、「ステップ 3: インスタンスの詳細を設定する」を参照してください。

AWS CLI

起動時にインスタンスメタデータへのアクセスを無効にするには

`--metadata-options` に `HttpEndpoint=disabled` を設定し、インスタンスを起動します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c6i.large \
    ...
    --metadata-options "HttpEndpoint=disabled"
```

PowerShell

起動時にインスタンスメタデータへのアクセスを無効にするには

次の New-EC2Instance コマンドレットの例では、MetadataOptions_HttpEndpoint を disabled に設定してインスタンスを起動します。

```
New-EC2Instance `
    -ImageId ami-0abcdef1234567890 `
    -InstanceType c6i.large `
    -MetadataOptions_HttpEndpoint disabled
```

AWS CloudFormation

AWS CloudFormation を使用してインスタンスのメタデータオプションを指定するには、「AWS CloudFormation ユーザーガイド」の「AWS::EC2::LaunchTemplate MetadataOptions」プロパティを参照してください。

既存インスタンスのインスタンスメタデータオプションの変更

既存のインスタンスのインスタンスメタデータオプションを変更することが可能です。

また、既存のインスタンスでインスタンスメタデータオプションを変更することをユーザーに禁止する IAM ポリシーを作成することもできます。インスタンスメタデータオプションを変更できるユーザーをコントロールするには、指定したロールを持つユーザー以外のすべてのユーザーに ModifyInstanceMetadataOptions API の使用を禁止するポリシーを指定できます。IAM ポリシーの例については、「インスタンスメタデータの使用」を参照してください。

既存インスタンスのインスタンスメタデータオプションのクエリ

次のいずれかの方法を使用して、既存のインスタンスのインスタンスメタデータオプションをクエリできます。

Console

コンソールを使用して既存のインスタンスのインスタンスメタデータオプションをクエリするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。

3. インスタンスを選択します。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択します。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで現在のインスタンスメタデータオプションを確認します。

AWS CLI

AWS CLI を使用して既存のインスタンスのインスタンスメタデータオプションをクエリするには

describe-instances CLI コマンドを使用します。

```
aws ec2 describe-instances \
    --instance-id i-1234567898abcdef0 \
    --query 'Reservations[].Instances[].MetadataOptions'
```

PowerShell

Tools for PowerShell を使用して既存のインスタンスのインスタンスメタデータオプションをクエリするには

Get-EC2Instance コマンドレットを使用します。

```
(Get-EC2Instance `
    -InstanceId i-1234567898abcdef0).Instances.MetadataOptions
```

IMDSv2 の使用を要求する

既存のインスタンスに対して、インスタンスメタデータのリクエスト時に IMDSv2 が使用されるようにするため、既存のインスタンスメタデータオプションを変更します。IMDSv2 が必須である場合、IMDSv1 は使用できません。

Console

既存インスタンスでの IMDSv2 の使用を義務付けるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択します。

4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択します。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] で、[有効にする] を選択します。
   b. [IMDSv2] の場合は、[必須] を選択します。
   c. [Save] を選択します。

AWS CLI

既存インスタンスでの IMDSv2 の使用を義務付けるには

modify-instance-metadata-optionsCLI コマンドを使って、`http-tokens` パラメータを `required` に設定できます。`http-tokens` の値を指定する場合は、`http-endpoint` も `enabled` に設定する必要があります。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-tokens required \
    --http-endpoint enabled
```

PowerShell

既存インスタンスでの IMDSv2 の使用を義務付けるには

Edit-EC2InstanceMetadataOption コマンドレットを使って、`HttpTokens` パラメータを `required` に設定できます。`HttpTokens` の値を指定する場合は、`HttpEndpoint` も `enabled` に設定する必要があります。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567898abcdef0 `
    -HttpTokens required `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

IMDSv1 の使用を再開します

IMDSv2 が必須である場合、インスタンスメタデータのリクエスト時に IMDSv1 は機能しません。IMDSv2 がオプションである場合、IMDSv2 と IMDSv1 の両方が機能します。したがって、IMDSv1 を復元するには、次のいずれかの方法を使用して IMDSv2 をオプションとします。

Console

インスタンスで IMDSv1 の使用を復元するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択します。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択します。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] で、[有効にする] が選択されていることを確認します。
   b. [IMDSv2] の場合は、[オプション] を選択します。
   c. [Save] を選択します。

AWS CLI

インスタンスで IMDSv1 の使用を復元するには

modify-instance-metadata-options CLI コマンドを、`http-tokens` を `optional` に設定して実行すると、インスタンスメタデータのリクエスト時に IMDSv1 の使用を復元できます。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-tokens optional \
    --http-endpoint enabled
```

PowerShell

インスタンスで IMDSv1 の使用を復元するには

Edit-EC2InstanceMetadataOption コマンドレットを、`HttpTokens` を `optional` に設定して実行すると、インスタンスメタデータのリクエスト時に IMDSv1 の使用を復元できます。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567898abcdef0 `
    -HttpTokens optional `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

PUT レスポンスホップリミットを変更する

既存インスタンスについて、PUTリスポンスホップリミットの設定を変更することができます。

現在、PUT 応答ホップ制限の変更をサポートしているのは、AWS CLI と AWS SDK のみです。

AWS CLI

PUT レスポンスホップリミットを変更するには

modify-instance-metadata-options CLI コマンドを使って、`http-put-response-hop-limit` パラメータを必要なホップ数に設定できます。以下の例では、ホップリミットが3に設定されています。`http-put-response-hop-limit` の値を指定する場合は、`http-endpoint` を `enabled` に設定することも必要です。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-put-response-hop-limit 3 \
    --http-endpoint enabled
```

PowerShell

PUT レスポンスホップリミットを変更するには

Edit-EC2InstanceMetadataOption コマンドレットを使って、`HttpPutResponseHopLimit` パラメータを必要なホップ数に設定できます。以下の例では、ホップリミットが3に設定されています。`HttpPutResponseHopLimit` の値を指定する場合は、`HttpEndpoint` を `enabled` に設定することも必要です。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567898abcdef0 `
    -HttpPutResponseHopLimit 3 `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

インスタンスの IPv6 エンドポイントを有効にする

デフォルトでは、IPv6 エンドポイントは無効です。これは、インスタンスを IPv6 専用サブネットで起動した場合にも当てはまります。IMDS の IPv6 エンドポイントは、Nitro System 上に構築されたインスタンス 上でのみアクセスできます。

現在、インスタンスの IPv6 エンドポイントの有効化をサポートしているのは、AWS CLI と AWS SDK のみです。

AWS CLI

インスタンスの IPv6 エンドポイントを有効にするには

modify-instance-metadata-options CLI コマンドを使って、`http-protocol-ipv6` パラメータを `enabled` に設定できます。`http-protocol-ipv6` の値を指定する場合は、`http-endpoint` を `enabled` に設定することも必要です。

```
aws ec2 modify-instance-metadata-options \
 --instance-id i-1234567898abcdef0 \
 --http-protocol-ipv6 enabled \
 --http-endpoint enabled
```

PowerShell

インスタンスの IPv6 エンドポイントを有効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使って、`HttpProtocolIpv6` パラメータを `enabled` に設定できます。`HttpProtocolIpv6` の値を指定する場合は、`HttpEndpoint` を `enabled` に設定することも必要です。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567898abcdef0 `
    -HttpProtocolIpv6 enabled `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

インスタンスメタデータへのアクセスを有効にする

使用中の IMDS のバージョンに関係なく、IMDS の HTTP エンドポイントを無効にすることによりインスタンスメタデータへのアクセスをオフにすることができます。HTTP エンドポイントを無効化することにより、この変更はいつでも元に戻すことができます。

次のいずれかの方法を使用して、インスタンスのインスタンスメタデータへのアクセスを有効にします。

Console

インスタンスメタデータへのアクセスを有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択します。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択します。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] で、[有効にする] を選択します。
   b. [Save] を選択します。

AWS CLI

インスタンスメタデータへのアクセスを有効にするには

modify-instance-metadata-options CLI コマンドを使って、`http-endpoint` パラメータを `enabled` に設定できます。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-endpoint enabled
```

PowerShell

インスタンスメタデータへのアクセスを有効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使って、`HttpEndpoint` パラメータを `enabled` に設定できます。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567898abcdef0 `
    -HttpEndpoint enabled).InstanceMetadataOptions
```

インスタンスメタデータへのアクセスを無効にする

使用中のインスタンスメタデータサービスのバージョンに関係なく、IMDS の HTTP エンドポイントを無効にすることにより IMDS へのアクセスをオフにすることができます。HTTP エンドポイントを有効化することにより、この変更はいつでも元に戻すことができます。

インスタンスのインスタンスメタデータへのアクセスを無効にするには、次のいずれかの方法を使用します。

Console

インスタンスメタデータへのアクセスを無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択します。
4. [アクション]、[インスタンスの設定]、[インスタンスメタデータのオプションを変更] の順に選択します。
5. [インスタンスメタデータオプションの変更] ダイアログボックスで、次の操作を行います。
   a. [インスタンスメタデータサービス] では、[有効にする] をオフにします。
   b. [Save] を選択します。

AWS CLI

インスタンスメタデータへのアクセスを無効にするには

modify-instance-metadata-options CLI コマンドを使って、`http-endpoint` パラメータを `disabled` に設定できます。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-1234567898abcdef0 \
    --http-endpoint disabled
```

PowerShell

インスタンスメタデータへのアクセスを無効にするには

Edit-EC2InstanceMetadataOption コマンドレットを使って、`HttpEndpoint` パラメータを `disabled` に設定できます。

```
(Edit-EC2InstanceMetadataOption `
    -InstanceId i-1234567898abcdef0 `
    -HttpEndpoint disabled).InstanceMetadataOptions
```

インスタンスメタデータの取得

インスタンスメタデータは実行中のインスタンスから取得できるため、Amazon EC2 コンソールまたは AWS CLI を使用する必要はありません。これは、インスタンスから実行するスクリプトを記述しているときに便利です。例えば、インスタンスメタデータからインスタンスのローカル IP アドレスにアクセスして、外部アプリケーションへの接続を管理できます。

インスタンスメタデータはいくつかのカテゴリに分けられます。各インスタンスメタデータカテゴリの説明については、インスタンスメタデータのカテゴリを参照してください。

実行中のインスタンス内にあるインスタンスメタデータの、すべてのカテゴリを表示するには、以下の IPv4 または IPv6 URI を使用します。

IPv4

```
http://169.254.169.254/latest/meta-data/
```

IPv6

```
http://[fd00:ec2::254]/latest/meta-data/
```

これらの IP アドレス は、リンクローカルアドレスであり、インスタンスからのみ使用することが可能です。詳細については、このユーザーガイドの「リンクローカルアドレス」と Wikipedia の「Link-local address」を参照してください。

> **Note**
>
> このセクションの例では、IMDS の IPv4 アドレス 169.254.169.254 を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6 アドレスを有効にして使用してください。[fd00:ec2::254]。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

コマンドフォーマットは、IMDSv1とIMDSv2のどちらを使うかによって異なります。デフォルトでは、両方のバージョンの IMDS を使用できます。IMDSv2の使用を義務付けるには、IMDSv2 の使用を参照してください。

次の例のように、cURL などのツールを使用できます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/
```

Windows インスタンスからインスタンスメタデータを取得するコマンドについては、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「インスタンスメタデータの取得」を参照してください。

コスト

インスタンスメタデータおよびユーザーデータの取得に使用する HTTP リクエストに対しては課金されません。

考慮事項

インスタンスメタデータの取得に関する問題を回避するには、次の点を考慮してください。

- コンテナ環境では、ホップ制限を 2 に設定することをお勧めします。

  AWS SDK はデフォルトで IMDSv2 コールを使用します。IMDSv2 呼び出しに応答がない場合、SDK は呼び出しを再試行し、それでも失敗した場合は、IMDSv1 を使用します。これにより、特にコンテナ環境では、遅延が発生することがあります。コンテナ環境では、ホップ制限が 1 の場合、コンテナへの到達は余分なネットワークホップと見なされるため、IMDSv2 応答は返されません。IMDSv1 へのフォールバックプロセスとその結果として生じる遅延を回避するために、コンテナ環境でホップ制限を 2 に設定することをお勧めします。詳細については、「インスタンスメタデータオプションの設定」を参照してください。
- IMDSv2 でトークンを取得する際には、**/latest/api/token** を使用する必要があります。

バージョン固有の任意のパス (例: /2021-03-23/api/token) に PUT リクエストを発行した場合は、メタデータサービスから 403 Forbidden エラーが返されます。この応答は意図されたものです。

- IMDSv2 が必要な場合は、IMDSv1 は動作しません。

  インスタンスに IMDSv2 が必要かどうかは、次のように確認できます。インスタンスを選択して詳細を表示し、[IMDSv2] の値を確認します。値は、[必須] (IMDSv2 のみ使用可能) または [オプション] (IMDSv2 と IMDSv1 を使用できます) のいずれかです。

リスポンスおよびエラーメッセージ

すべてのインスタンスメタデータがテキスト (HTTP コンテンツタイプ text/plain) として返されます。

特定のメタデータリソースに対するリクエストは、適切な値または 404 - Not Found HTTP エラーコード (リソースを使用できない場合) を返します。

一般的なメタデータリソースに対するリクエスト (/ で終わる URI) は、使用可能なリソースのリストまたは 404 - Not Found HTTP エラーコード (使用可能なリソースがない場合) を返します。リスト項目は個別の行に表示され、各行の末尾には改行記号 (ASCII 10) が付いています。

インスタンスメタデータサービスバージョン 2を使って行われたリクエストについては、次の HTTP エラーコードが返されます。

- 400 - Missing or Invalid Parameters–PUTリクエストが無効である。
- 401 - Unauthorized–GETリクエストが無効なトークンを使用している。推奨されるアクションは新しいトークンを生成することです。
- 403 - Forbidden - リクエストが許可されていないか、あるいは IMDS がオフです。

インスタンスメタデータの取得の例

次の例は、Linux インスタンスで使用できるコマンドを示しています。Windows インスタンスからインスタンスメタデータを取得するコマンドについては、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「インスタンスメタデータの取得」を参照してください。

例

- 使用できるインスタンスメタデータのバージョンを取得する

- 上位レベルのメタデータ項目を取得する
- 使用可能なパブリックキーのリストを取得する
- パブリックキー 0 が使用できるフォーマットを示す
- パブリックキー 0 を取得する (OpenSSH キーフォーマット)
- インスタンスのサブネット ID を取得する
- インスタンスのインスタンスタグを取得する

使用できるインスタンスメタデータのバージョンを取得する

次の例では、使用できるインスタンスメタデータのバージョンを取得しています。各バージョンは、新しいインスタンスのメタデータカテゴリがリリースされたときのインスタンスメタデータビルドを参照します。インスタンスメタデータビルドのバージョンは、Amazon EC2 API のバージョンとは相関しません。以前のバージョンに存在する構造および情報に依存するスクリプトがある場合は、以前のバージョンを使用することができます。

> **Note**
>
> Amazon EC2 が新しいインスタンスメタデータビルドをリリースするたびにコードを更新する必要をなくすために、バージョン番号ではなく、パス内の `latest` を使用することが推奨されます。例えば、以下のように `latest` を使用します。
> `curl http://169.254.169.254/latest/meta-data/ami-id`

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-
aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
```

```
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
...
latest
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/
1.0
2007-01-19
2007-03-01
2007-08-29
2007-10-10
2007-12-15
2008-02-01
2008-09-01
2009-04-04
2011-01-01
2011-05-01
2012-01-12
2014-02-25
2014-11-05
2015-10-20
2016-04-19
...
latest
```

上位レベルのメタデータ項目を取得する

次の例では、上位レベルのメタデータ項目を取得しています。詳細については、インスタンスメタデータのカテゴリ を参照してください。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/
```

```
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
iam/
instance-action
instance-id
instance-life-cycle
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/
ami-id
ami-launch-index
ami-manifest-path
block-device-mapping/
events/
hostname
iam/
instance-action
instance-id
instance-type
local-hostname
local-ipv4
mac
metrics/
network/
```

```
placement/
profile
public-hostname
public-ipv4
public-keys/
reservation-id
security-groups
services/
```

次の例では、前の例で取得された最上位メタデータアイテムの値のいくつかを取得しています。IMDSv2 リクエストは、前の例のコマンドで作成された保管済みトークン (期限内であると仮定) を使用します。

IMDSv2

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/ami-id
ami-0abcdef1234567890
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/ami-id
ami-0abcdef1234567890
```

IMDSv2

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/reservation-id
r-0efghijk987654321
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/reservation-id
r-0efghijk987654321
```

IMDSv2

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/local-hostname
ip-10-251-50-12.ec2.internal
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/local-hostname
ip-10-251-50-12.ec2.internal
```

IMDSv2

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/
latest/meta-data/public-hostname
ec2-203-0-113-25.compute-1.amazonaws.com
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-hostname
ec2-203-0-113-25.compute-1.amazonaws.com
```

使用可能なパブリックキーのリストを取得する

次の例では、使用できるパブリックキーの一覧を取得しています。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-
aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-
data/public-keys/
0=my-public-key
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/
```

```
0=my-public-key
```

パブリックキー 0 が使用できるフォーマットを示す

次の例は、パブリックキー0のフォーマットを示しています。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-keys/0/
openssh-key
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/0/
openssh-key
```

パブリックキー 0 を取得する (OpenSSH キーフォーマット)

次の例では、パブリックキー0を取得しています (OpenSSH キーフォーマット)。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
ssh-rsa MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
```

```
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE my-public-key
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
ssh-rsa MIICiTCCAfICCQD6m7oRw0uXOjANBgkqhkiG9w0BAQUFADCBiDELMAkGA1UEBhMC
VVMxCzAJBgNVBAgTAldBMRAwDgYDVQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6
b24xFDASBgNVBAsTC0lBTSBDb25zb2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAd
BgkqhkiG9w0BCQEWEG5vb25lQGFtYXpvbi5jb20wHhcNMTEwNDI1MjA0NTIxWhcN
MTIwNDI0MjA0NTIxWjCBiDELMAkGA1UEBhMCVVMxCzAJBgNVBAgTAldBMRAwDgYD
VQQHEwdTZWF0dGxlMQ8wDQYDVQQKEwZBbWF6b24xFDASBgNVBAsTC0lBTSBDb25z
b2xlMRIwEAYDVQQDEwlUZXN0Q2lsYWMxHzAdBgkqhkiG9w0BCQEWEG5vb25lQGFt
YXpvbi5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAMaK0dn+a4GmWIWJ
21uUSfwfEvySWtC2XADZ4nB+BLYgVIk60CpiwsZ3G93vUEIO3IyNoH/f0wYK8m9T
rDHudUZg3qX4waLG5M43q7Wgc/MbQITxOUSQv7c7ugFFDzQGBzZswY6786m86gpE
Ibb3OhjZnzcvQAaRHhdlQWIMm2nrAgMBAAEwDQYJKoZIhvcNAQEFBQADgYEAtCu4
nUhVVxYUntneD9+h8Mg9q6q+auNKyExzyLwaxlAoo7TJHidbtS4J5iNmZgXL0Fkb
FFBjvSfpJIlJ00zbhNYS5f6GuoEDmFJl0ZxBHjJnyp378OD8uTs7fLvjx79LjSTb
NYiytVbZPQUQ5Yaxu2jXnimvw3rrszlaEXAMPLE my-public-key
```

インスタンスのサブネット ID を取得する

次の例では、インスタンスのサブネット ID を取得しています。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-
aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-
data/network/interfaces/macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/network/interfaces/
macs/02:29:96:8f:6a:2d/subnet-id
subnet-be9b61d7
```

インスタンスのインスタンスタグを取得する

次の例では、サンプルインスタンスでインスタンスメタデータのタグが有効になっており、インスタンスタグ Name=MyInstance および Environment=Dev が含まれています。

この例では、インスタンスのインスタンスタグキーをすべて取得しています。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/tags/instance
Name
Environment
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/tags/instance
Name
Environment
```

次の例では、前の例で取得した Name キーの値を取得しています。IMDSv2 リクエストは、前の例のコマンドで作成された保管済みトークン (期限内であると仮定) を使用します。

IMDSv2

```
[ec2-user ~]$ curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/tags/instance/Name
MyInstance
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/tags/instance/Name
MyInstance
```

クエリスロットル

クエリは IMDS でインスタンスごとにスロットリングし、インスタンスから IMDS への同時接続数を制限します。

AWS セキュリティ認証情報を取得するために IMDS を使用している場合、毎回のトランザクションで、または高頻度のスレッドやプロセスから同時に認証情報をクエリしないようにします。スロットリングの原因となる可能性があります。代わりに、認証情報をキャッシュに格納して有効期限が近づくまで待つことをお勧めします。IAM ロールとロールに関連付けられたセキュリティ認証情報の詳細については、「インスタンスメタデータからのセキュリティ認証情報の取得」を参照してください。

IMDS にアクセスする際にスロットリングした場合、エクスポネンシャルバックオフ戦略でクエリを再試行します。

IMDS アクセスの制限

ローカルファイアウォールルールを使って、プロセスの一部またはすべてから IMDS へのアクセスを無効化することを検討できます。

> **Note**
>
> Nitro System 上に構築されたインスタンス では、VPC 内のネットワークアプライアンス (仮想ルーターなど) がパケットを IMDS アドレスに転送し、かつ、インスタンス上のデフォルトの送信元/送信先チェックが無効な場合、IMDS がユーザー自身のネットワークから到達可能になります。VPC の外側にある送信元から IMDS に到達しないようにするには、送信先 IMDS の IPv4 アドレスが `169.254.169.254` (IPv6 エンドポイントを有効にしている場合は、IMDS の IPv6 アドレスが `[fd00:ec2::254]`) であるパケットをドロップするように、ネットワークアプライアンスの設定を変更することをお勧めします。

iptables を使ったアクセス制限

次の例では、Linux iptables およびその`owner`モジュールを使って、Apache ウェブサーバーが (デフォルトインストールユーザー ID `apache`に基づいて) 169.254.169.254 にアクセスするのを防ぐことができます。拒否ルールを使って、そのユーザーとして実行中のプロセスからのインスタンスメタデータリクエスト (IMDSv1またはIMDSv2) をすべて拒否します。

```
$ sudo iptables --append OUTPUT --proto tcp --destination 169.254.169.254 --match owner
 --uid-owner apache --jump REJECT
```

また、ルールの許可を使うことで、特定のユーザーまたはグループへのアクセスを許可することを検討できます。ルールの許可は、どのソフトウェアがインスタンスメタデータへのアクセスが必要かについてユーザーが決定しなければならないため、セキュリティ観点からみたときに管理しやすいかも

しれません。ルールの許可 を使用すると、後にインスタンスのソフトウェアまたは構成を変更した場合に、誤ってソフトウェアがメタデータサービス (アクセスする意図がなかった) にアクセスするのを許可する可能性が低くなります。また、ファイアウォールのルールを変更しなくても許可されたグループにユーザーを追加/削除できるよう、グループ使用をルールの許可と組み合わせることもできます。

次の例では、ユーザーアカウント `trustworthy-user` で実行中のプロセス以外のすべてのプロセスによる IMDS へのアクセスを禁止しています。

```
$ sudo iptables --append OUTPUT --proto tcp --destination 169.254.169.254 --match
 owner ! --uid-owner trustworthy-user --jump REJECT
```

> **Note**
>
> - ローカルファイアウォールルールを使用するには、前の例のコマンドをニーズに合わせて変更する必要があります。
> - デフォルトでは、iptables ルールはシステム再起動全体で永続しません。ここには説明されていない OS 機能を使って永続的にすることができます。
> - iptables ownerモジュールは、グループが所定のローカルユーザーのプライマリグループである場合にのみツールメンバーシップと一致します。他のグループは一致しません。

PF または IPFW を使ってアクセスを制限する

FreeBSD または OpenBSD を使用している場合、PF または IPFW の使用も検討できます。次の例では、IMDS へのアクセスをルートユーザーにのみ制限しています。

PF

```
$ block out inet proto tcp from any to 169.254.169.254
```

```
$ pass out inet proto tcp from any to 169.254.169.254 user root
```

IPFW

```
$ allow tcp from any to 169.254.169.254 uid root
```

```
$ deny tcp from any to 169.254.169.254
```

> **Note**
>
> PF および IPFW コマンドの順序は重要となります。PF のデフォルトは最後に一致したルールであり、IPFW のデフォルトは最初に一致したルールです。

インスタンスユーザーデータの使用

インスタンスユーザーデータを使用してインスタンスをカスタマイズできます。インスタンスを起動すると、パラメータやスクリプトをユーザーデータとして保存できます。ユーザーデータのスクリプトは、インスタンスを起動すると実行されます。ユーザーデータはインスタンス属性として表示できます。インスタンスメタデータサービス (IMDS) を使用して、インスタンスのユーザーデータを表示することもできます。

考慮事項

- ユーザーデータは非透過的なデータとして取り扱われ、指定したデータがそのまま返されます。インスタンスによって解釈が異なります。
- ユーザーデータは、base64 でエンコードされている必要があります。Amazon EC2コンソールは、base64 エンコードを実行したり、base64 エンコード入力を受け入れたりできます。
- ユーザーデータは raw 形式の 16 KB に制限されます (以前は base64 エンコード)。base64 エンコード後の 文字列の長さサイズ n は、ceil(n/3)*4 です。
- ユーザーデータを取得するときにユーザーデータを base64 デコードする必要があります。インスタンスのメタデータあるいはコンソールを使用してデータを取得する場合、自動的にデコードされます。
- インスタンスを停止してユーザーデータを変更した後に、インスタンスを起動した場合でも、更新されたユーザーデータはには実行されません。
- ユーザーデータはインスタンス属性です。インスタンスから AMI を作成する場合、インスタンスのユーザーデータは AMI に含まれません。

起動時にインスタンスユーザーデータを指定する

インスタンスの起動時のユーザーデータを指定できます。コンソールの使用説明については、「起動時にインスタンスユーザーデータを指定する」を参照してください。AWS CLI の使用例については、「the section called “ユーザーデータと AWS CLI”」を参照してください。

インスタンスユーザーデータを変更する

EBS ルートボリュームを持つインスタンスのユーザーデータを変更できます。インスタンスは停止状態である必要があります。コンソールの使用説明については、「インスタンスユーザーデータの表示と更新」を参照してください。AWS CLI の使用例については、「modify-instance-attribute」を参照してください。

インスタンスからインスタンスユーザーデータを取得する

> **Note**
>
> このセクションの例では、IMDS の IPv4 アドレス 169.254.169.254 を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6 アドレスを有効にして使用してください。[fd00:ec2::254]。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

インスタンスからユーザーデータを取得するには、次の URI を使用します。

```
http://169.254.169.254/latest/user-data
```

ユーザーデータのリクエストは、データをそのままの状態で返します (コンテンツタイプ application/octet-stream)。インスタンスにユーザーデータがない場合、リクエストは 404 - Not Found を返します。

この例は、カンマで区切られたテキストとして指定されたユーザーデータを返します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/user-data
1234,john,reboot,true | 4512,richard, | 173,,,
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/user-data
```

```
1234,john,reboot,true | 4512,richard, | 173,,,
```

この例では、スクリプトとして指定されたユーザーデータを返します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/user-data
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/user-data
#!/bin/bash
yum update -y
service httpd start
chkconfig httpd on
```

お使いのコンピュータからインスタンスのユーザーデータを取得する

自分のコンピュータからインスタンスのユーザーデータを取得できます。コンソールの使用説明については、「インスタンスユーザーデータの表示と更新」を参照してください。AWS CLI の使用例については、「ユーザーデータと AWS CLI」を参照してください。

動的データの取得

実行中のインスタンス内から動的データを取得するには、次の URI を使用します。

```
http://169.254.169.254/latest/dynamic/
```

> **Note**
>
> このセクションの例では、IMDS の IPv4 アドレス 169.254.169.254 を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6

アドレスを有効にして使用してください。[fd00:ec2::254]。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

この例では、高レベルのインスタンスアイデンティティカテゴリを取得する方法を表示しています。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/instance-identity/
rsa2048
pkcs7
document
signature
dsa2048
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/dynamic/instance-identity/
rsa2048
pkcs7
document
signature
dsa2048
```

動的データの詳細およびその取得方法の例については、「インスタンスアイデンティティドキュメント」を参照してください。

インスタンスメタデータのカテゴリ

インスタンスメタデータはいくつかのカテゴリに分けられます。インスタンスのメタデータを取得するには、リクエストでカテゴリを指定します。すると、メタデータがレスポンスで返されます。

新しいカテゴリがリリースされると、新しいインスタンスメタデータビルドが新しいバージョン番号で作成されます。次の表では、[Version when category was released] (カテゴリがリリースされたときのバージョン) 列が、インスタンスメタデータカテゴリがリリースされたときのビルドバー

ジョンを指定しています。Amazon EC2 が新しいインスタンスメタデータビルドをリリースするたびにコードを更新する必要をなくすために、メタデータリクエストのバージョン番号の代わりに、`latest` を使用することが推奨されます。詳細については、使用できるインスタンスメタデータのバージョンを取得する を参照してください。

Amazon EC2 が新しいインスタンスメタデータカテゴリをリリースすると、新しいカテゴリのインスタンスメタデータが既存のインスタンスで使用できなくなる場合があります。Nitro システム上に構築されたインスタンスでは、起動時に用意されたカテゴリのインスタンスメタデータのみが取得可能です。Xen ハイパーバイザーを使用するインスタンスの場合は、一度停止した後に開始することで、そのインスタンスで使用可能なカテゴリを更新できます。

次の表は、インスタンスメタデータのカテゴリをまとめたものです。カテゴリ名のいくつかには、インスタンスに固有のデータのためのプレースホルダーが含まれています。例えば、*mac* はネットワークインターフェイスの MAC アドレスを表します。インスタンスメタデータを取得する際には、プレースホルダーを実際の値に置き換える必要があります。

カテゴリ	説明	カテゴリがリリースされたときのバージョン
`ami-id`	インスタンスの起動に使用される AMI ID。	1.0
`ami-launch-index`	同じ `RunInstances` 呼び出しを使用して複数のインスタンスを起動する場合、この値は各インスタンスの起動順序を示します。最初に起動されたインスタンスの値は 0 です。Auto Scaling または EC2 フリートを使用してインスタンスを起動する場合、この値は常に 0 です。	1.0
`ami-manifest-path`	Amazon S3 での AMI のマニフェストファイルのパス。Amazon EBS-backed AMI を使用してインスタンスを起動した場合、返される結果は unknown です。	1.0

カテゴリ	説明	カテゴリがリリースされたときのバージョン
ancestor-ami-ids	この AMI を作成するために再バンドルされたあらゆるインスタンスの AMI ID。この値は、AMI マニフェストファイルが ancestor-amis キーを含む場合にのみ存在します。	2007-10-10
autoscaling/target-lifecycle-state	Auto Scaling インスタンスの移行先となる、ターゲットの Auto Scaling ライフサイクルの状態を示す値。2022 年 3 月 10 日以降、インスタンスがターゲットのライフサイクル状態の 1 つに移行したときに表示されます。使用できる値: Detached|InService |Standby|Terminated |Warmed:Hibernated |Warmed:Running |Warmed:Stopped |Warmed:Terminated 「Amazon EC2 Auto Scaling ユーザーガイド」の「インスタンスメタデータを使用してターゲットライフサイクル状態を取得する」を参照してください。	2021-07-15
block-device-mapping/ami	root/boot ファイルシステムを含む仮想デバイス。	2007-12-15

カテゴリ	説明	カテゴリがリリースされたときのバージョン
block-device-mapping/ebs N	任意の Amazon EBS ボリュームに関連付けられた仮想デバイス。Amazon EBS ボリュームは、起動の時点、またはインスタンスが最後に開始された時点で存在している場合にのみ、メタデータで使用できます。N は、Amazon EBS ボリュームのインデックス (ebs1 や ebs2 など) を示します。	2007-12-15
block-device-mapping/ephemeral N	非 NVMe インスタンスストアボリュームの仮想デバイス。N は、各ボリュームのインデックスを示します。ブロックデバイスマッピングのインスタンスストアボリュームの数は、インスタンスのインスタンスストアボリュームの実際の数に一致しない場合があります。インスタンスに使用可能なインスタンスストアボリュームの数は、インスタンスタイプによって決定されます。ブロックデバイスマッピングのインスタンスストアボリュームの数が、インスタンスに利用可能な数を超える場合、追加のインスタンスストアボリュームは無視されます。	2007-12-15

カテゴリ	説明	カテゴリがリリースされたときのバージョン
block-device-mapping/root	ルートデバイスに関連付けられた仮想デバイスまたはパーティション、あるいは仮想デバイス上のパーティション。ルート (/ または C:) ファイルシステムは、所定のインスタンスに関連付けられています。	2007-12-15
block-device-mapping/swap	swap に関連付けられた仮想デバイス。存在しない場合もあります。	2007-12-15
elastic-gpus/associations/ *elastic-gpu-id*	インスタンスにアタッチされている Elastic GPU がある場合、その ID と接続情報を含めた Elastic GPU に関する情報の JSON 文字列が含まれます。	2016-11-30
elastic-inference/associations/ *eia-id*	インスタンスにアタッチされた Elastic Inference アクセラレーターがある場合、その ID とタイプを含めた Elastic Inference アクセラレーターに関する情報の JSON 文字列が含まれます。	2018-11-29
events/maintenance/history	インスタンスの完了またはキャンセルされたメンテナンスイベントがある場合は、イベントに関する情報を含む JSON 文字列を含みます。詳細については、「完了またはキャンセルされたイベントのイベント履歴を表示するには」を参照してください。	2018-08-17

カテゴリ	説明	カテゴリがリリースされたときのバージョン
events/maintenance/scheduled	インスタンスがアクティブなメンテナンスイベントがある場合は、イベントに関する情報を含む JSON 文字列を含みます。詳細については、予定されたイベントの表示 を参照してください。	2018-08-17
events/recommendations/rebalance	EC2 インスタンスの再調整推奨通知がインスタンスに対して送信されるおおよその時間 (UTC)。このカテゴリのメタデータの例を次に示します。{"noticeTime": "2020-11-05T08:22:00Z"} このカテゴリは、通知が発された後にのみ使用できます。詳細については、「EC2 インスタンスの再調整に関する推奨事項」を参照してください。	2020-10-27
hostname	EC2 インスタンスが IP ベースの命名 (IPBN) を使用している場合、これはインスタンスのプライベート IPv4 DNS ホスト名です。EC2 インスタンスがリソースベースの命名 (RBN) を使用している場合、これは RBN です。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。IPBN と RBN の詳細については、「Amazon EC2 インスタンスのホスト名タイプ」を参照してください。	1.0

カテゴリ	説明	カテゴリがリリースされたときのバージョン
iam/info	インスタンスに関連付けられた IAM ロールがある場合、インスタンスの LastUpdated の日付、InstanceProfileArn、InstanceProfileId など、インスタンスプロファイルが更新された最終時刻に関する情報が格納されます。そうでない場合は、なしになります。	2012-01-12
iam/security-credentials/role-name	インスタンスに関連付けられた IAM ロールがある場合、*role-name* はロールの名前になり、*role-name* に、そのロールに関連付けられた一時的なセキュリティ認証情報が格納されます (詳細については、「インスタンスメタデータからのセキュリティ認証情報の取得」を参照してください)。そうでない場合は、なしになります。	2012-01-12
identity-credentials/ec2/info	identity-credentials/ec2/security-credentials/ec2-instance の認証情報に関する情報。	2018-05-23

カテゴリ	説明	カテゴリがリリースされたときのバージョン
identity-credentials/ec2/security-credentials/ec2-instance	インスタンス上のソフトウェアが自身を AWS に識別し、EC2 Instance Connect や AWS Systems Manager デフォルトのホスト管理設定などの機能をサポートできるようにするインスタンスアイデンティティロール用の認証情報。これらの認証情報にはポリシーがアタッチされていないため、AWS 機能に対してインスタンスを識別する以外に追加の AWS API 許可はありません。詳細については、「インスタンスアイデンティティロール」を参照してください。	2018-05-23
instance-action	バンドルの準備のために再起動する必要があることをインスタンスに伝えます。有効な値: none | shutdown | bundle-pending 。	2008-09-01
instance-id	このインスタンスの ID。	1.0
instance-life-cycle	このインスタンスの購入オプション。詳細については、インスタンス購入オプション を参照してください。	2019-10-01
instance-type	インスタンスの種類。詳細については、インスタンスタイプ を参照してください。	2007-08-29

カテゴリ	説明	カテゴリがリリースされたときのバージョン
ipv6	インスタンスの IPv6 アドレス。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) のネットワークインターフェイスおよび最初に割り当てられた IPv6 アドレスを示します。ネットワークインターフェイス [0] に IPv6 アドレスが存在しない場合、この項目は設定されず、HTTP 404 応答が返されます。	2021-01-03
kernel-id	このインスタンスで起動したカーネルの ID (ある場合)。	2008-02-01
local-hostname	複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。EC2 インスタンスが IP ベースの命名 (IPBN) を使用している場合、これはインスタンスのプライベート IPv4 DNS ホスト名です。EC2 インスタンスがリソースベースの命名 (RBN) を使用している場合、これは RBN です。IPBN、RBN、および EC2 インスタンスの命名の詳細については、「Amazon EC2 インスタンスのホスト名タイプ」を参照してください。	2007-01-19

カテゴリ	説明	カテゴリがリリースされたときのバージョン
local-ipv4	インスタンスのプライベート IPv4 アドレス。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。これが IPv6 専用インスタンスの場合、この項目は設定されず、HTTP 404 応答が返されます。	1.0
mac	インスタンスのメディアアクセスコントロール (MAC) アドレス。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。	2011-01-01
metrics/vhostmd	使用不可	2011-05-01
network/interfaces/macs/mac/device-number	そのインターフェイスに関連付けられた固有のデバイス番号。デバイス番号はデバイス名に対応します。例えば、2 という device-number は eth2 デバイスを指します。このカテゴリは、Amazon EC2 API で使用される DeviceIndex フィールドと device-index フィールド、および AWS CLI の EC2 コマンドに対応します。	2011-01-01
network/interfaces/macs/mac/interface-id	ネットワークインターフェイスの ID。	2011-01-01

カテゴリ	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/mac/ipv4-associations/public-ip	各パブリック IP アドレスに関連付けられ、そのインターフェイスに割り当てられたプライベート IPv4 アドレス。	2011-01-01
network/interfaces/macs/mac/ipv6s	インターフェイスに割り当てられた IPv6 アドレス。	2016-06-30
network/interfaces/macs/mac/ipv6-prefix	ネットワークインターフェイスに割り当てられた IPv6 プレフィクス。	
network/interfaces/macs/mac/local-hostname	インスタンスのプライベート IPv4 DNS ホスト名。複数のネットワークインターフェイスが存在する場合、これは eth0 デバイス (デバイス番号が 0 のデバイス) を示します。これが IPv6 専用インスタンスの場合、これはリソースベースの名前です。IPBN と RBN の詳細については、「Amazon EC2 インスタンスのホスト名タイプ」を参照してください。	2007-01-19
network/interfaces/macs/mac/local-ipv4s	インターフェイスに関連付けられたプライベート IPv4 アドレス。これが IPv6 専用のネットワークインターフェイスである場合、この項目は設定されず、HTTP 404 応答が返されます。	2011-01-01
network/interfaces/macs/mac/mac	インスタンスの MAC アドレス。	2011-01-01

カテゴリ	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/*mac*/network-card	ネットワークカードのインデックス。インスタンスタイプによっては、複数のネットワークカードがサポートされているものもあります。	2020 年 11 月 1 日
network/interfaces/macs/mac/owner-id	ネットワークインターフェイスの所有者の ID。複数インターフェイスの環境では、インターフェイスは Elastic Load Balancing などのサードパーティによってアタッチできます。インターフェイス上のトラフィックは、常にインターフェイス所有者に対して課金されます。	2011-01-01
network/interfaces/macs/mac/public-hostname	インターフェイスのパブリック DNS (IPv4)。このカテゴリは、enableDnsHostnames 属性が true に設定されている場合にのみ返されます。詳細については、Amazon VPC ユーザーガイドの「DNS attributes for your VPC」(VPC の DNS 属性) を参照してください。インスタンスにパブリック IPv6 アドレスのみがあり、パブリック IPv4 アドレスがない場合、この項目は設定されず、HTTP 404 応答が返されます。	2011-01-01

カテゴリ	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/mac/public-ipv4s	インターフェイスに関連付けられたパブリック IP アドレスまたは Elastic IP アドレス。インスタンスには複数の IPv4 アドレスが存在する場合があります。	2011-01-01
network/interfaces/macs/mac/security-groups	ネットワークインターフェイスが属するセキュリティグループ。	2011-01-01
network/interfaces/macs/mac/security-group-ids	ネットワークインターフェイスが属するセキュリティグループの ID。	2011-01-01
network/interfaces/macs/mac/subnet-id	インターフェイスが存在するサブネットの ID。	2011-01-01
network/interfaces/macs/mac/subnet-ipv4-cidr-block	インターフェイスが存在するサブネットの IPv4 CIDR ブロック。	2011-01-01
network/interfaces/macs/mac/subnet-ipv6-cidr-blocks	インターフェイスが存在するサブネットの IPv6 CIDR ブロック。	2016-06-30
network/interfaces/macs/mac/vpc-id	インターフェイスが存在する VPC の ID。	2011-01-01
network/interfaces/macs/mac/vpc-ipv4-cidr-block	VPC のプライマリ IPv4 CIDR ブロック。	2011-01-01
network/interfaces/macs/mac/vpc-ipv4-cidr-blocks	VPC の IPv4 CIDR ブロック。	2016-06-30

カテゴリ	説明	カテゴリがリリースされたときのバージョン
network/interfaces/macs/mac/vpc-ipv6-cidr-blocks	インターフェイスが存在する VPC の IPv6 CIDR ブロック。	2016-06-30
placement/availability-zone	インスタンスが起動した利用可能ゾーン。	2008-02-01
placement/availability-zone-id	インスタンスが起動される静的アベイラビリティーゾーンの ID。このアベイラビリティーゾーン ID は、アカウント間で一貫しています。ただし、アベイラビリティーゾーンとは異なる場合があります (アベイラビリティーゾーンはアカウントによって異なります)。	2019-10-01
placement/group-name	インスタンスが起動されるプレイスメントグループの名前。	2020-08-24
placement/host-id	インスタンスが起動されるホストの ID。Dedicated Hosts にのみ適用されます。	2020-08-24
placement/partition-number	インスタンスが起動されるパーティションの番号。	2020-08-24
placement/region	インスタンスが起動される AWS リージョン。	2020-08-24
product-codes	インスタンスに関連付けられた AWS Marketplace 製品コード (ある場合)。	2007-03-01

カテゴリ	説明	カテゴリがリリースされたときのバージョン
public-hostname	インスタンスのパブリック DNS (IPv4)。このカテゴリは、enableDnsHostnames 属性が true に設定されている場合にのみ返されます。詳細については、Amazon VPC ユーザーガイドの「DNS attributes for your VPC」(VPC の DNS 属性) を参照してください。インスタンスにパブリック IPv6 アドレスのみがあり、パブリック IPv4 アドレスがない場合、この項目は設定されず、HTTP 404 応答が返されます。	2007-01-19
public-ipv4	パブリック IPv4 アドレス。インスタンスに Elastic IP アドレスが関連付けられている場合、返される値は Elastic IP アドレスです。	2007-01-19
public-keys/0/openssh-key	パブリックキー。インスタンスの起動時に指定された場合のみ返されます。	1.0
ramdisk-id	起動時に指定された RAM ディスクの ID (該当する場合)。	2007-10-10
reservation-id	予約の ID。	1.0

カテゴリ	説明	カテゴリがリリースされたときのバージョン
security-groups	インスタンスに適用されるセキュリティグループの名前。 起動後、インスタンスのセキュリティグループを変更できます。これらの変更は、この場所と network/interfaces/macs/*mac*/security-groups に反映されます。	1.0
services/domain	リージョンのAWSリソースのドメイン。	2014-02-25
services/partition	リソースが置かれているパーティションです。標準の AWS リージョンの場合、パーティションは aws です。他のパーティションにリソースがある場合、パーティションは aws-*partitionname* です。例えば、中国 (北京) リージョンにあるリソースのパーティションは、aws-cn です。	2015-10-20
spot/instance-action	アクション (休止、停止、または終了) およびアクションのおよその発生時刻 (UTC)。この項目が存在するのは、スポットインスタンスが休止、停止、または終了のためにマークされた場合のみです。詳細については、instance-action を参照してください。	2016-11-15

カテゴリ	説明	カテゴリがリリースされたときのバージョン
spot/termination-time	スポットインスタンスで使用しているオペレーティングシステムが、シャットダウン信号を受信するおよその時刻 (UTC)。この項目は、スポットインスタンスに対し Amazon EC2 による終了のマークが付けられている場合にのみ存在し、時刻値 (例えば 2015-01-05T18:02:00Z) が含まれます。ユーザー自身がスポットインスタンスを終了した場合、termination-time 項目に時刻は記述されません。詳細については、termination-time を参照してください。	2014-11-05
tags/instance	インスタンスに関連付けられるインスタンスタグ。インスタンスメタデータのタグへのアクセスを明示的に許可した場合のみ使用できます。詳細については、インスタンスメタデータのタグへのアクセスを許可する を参照してください。	2021-03-23

動的データのカテゴリ

次の表は、動的データのカテゴリをまとめたものです。

カテゴリ	説明	カテゴリがリリースされたときのバージョン
fws/instance-monitoring	顧客が CloudWatch で詳細な 1 分間隔のモニタリングを有効にしているかどうかを示す値。有効な値: enabled | disabled	2009-04-04
instance-identity/document	インスタンス ID、プライベート IP アドレスなど、インスタンスの属性を含む JSON。「インスタンスアイデンティティドキュメント」を参照してください。	2009-04-04
instance-identity/pkcs7	署名に対してドキュメントの真正性およびコンテンツを確認するために使用されます。「インスタンスアイデンティティドキュメント」を参照してください。	2009-04-04
instance-identity/signature	オリジンおよび権限を確認するために使用できるデータ。「インスタンスアイデンティティドキュメント」を参照してください。	2009-04-04

例: AMI 作成インデックス値

この例は、ユーザーデータおよびインスタンスメタデータの両方を使用してインスタンスを設定する方法を示しています、

> **Note**
>
> このセクションの例では、IMDS の IPv4 アドレス 169.254.169.254 を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6 アドレスを有効にして使用してください。[fd00:ec2::254]。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

この例で、Alice はお気に入りのデータベース AMI の 4 つのインスタンスを起動します。最初のインスタンスを元のインスタンスとし、残りの 3 つをレプリカとします。これらのインスタンスの起動

時に、レプリケーション戦略に関するユーザーデータを各レプリカに追加します。このデータはすべての 4 つのインスタンスで使用可能となるので、どの部分が各インスタンスに該当するかをそれぞれが認識できるように、ユーザーデータを構築する必要があります。この構築は、各インスタンスに対して一意となる ami-launch-index インスタンスメタデータ値を使用して行うことができます。同時に複数のインスタンスを起動する場合、ami-launch-index はインスタンスが起動された順序を示します。最初に起動されたインスタンスの値は 0 で示されます。

Alice が構築したユーザーデータを次に示します。

```
replicate-every=1min | replicate-every=5min | replicate-every=10min
```

replicate-every=1min データは最初のレプリカの設定を定義し、replicate-every=5min は 2 番目のレプリカの設定を定義します。以下、同様に設定を定義します。Alice は、個別のインスタンスのデータをパイプシンボル (|) で区切って、このデータを ASCII 文字列として指定することにしました。

Alice は、run-instancesコマンドを使用して 4 つのインスタンスを起動します。このとき、次のユーザーデータを指定します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --count 4 \
    --instance-type t2.micro \
    --user-data "replicate-every=1min | replicate-every=5min | replicate-every=10min"
```

起動したすべてのインスタンスに、ユーザーデータのコピーと次に示す一般的なメタデータが含まれています。

- AMI ID: ami-0abcdef1234567890
- 予約 ID: r-1234567890abcabc0
- パブリックキー: none
- セキュリティグループ名: default
- インスタンスタイプ: t2.micro

ただし、各インスタンスには所定の一意のメタデータが含まれます。

インスタンス 1

メタデータ	値
instance-id	i-1234567890abcdef0
ami-launch-index	0
public-hostname	ec2-203-0-113-25.compute-1.amazonaws.com
public-ipv4	67.202.51.223
local-hostname	ip-10-251-50-12.ec2.internal
local-ipv4	10.251.50.35

インスタンス 2

メタデータ	値
instance-id	i-0598c7d356eba48d7
ami-launch-index	1
public-hostname	ec2-67-202-51-224.compute-1.amazonaws.com
public-ipv4	67.202.51.224
local-hostname	ip-10-251-50-36.ec2.internal
local-ipv4	10.251.50.36

インスタンス 3

メタデータ	値
instance-id	i-0ee992212549ce0e7
ami-launch-index	2
public-hostname	ec2-67-202-51-225.compute-1.amazonaws.com

メタデータ	値
public-ipv4	67.202.51.225
local-hostname	ip-10-251-50-37.ec2.internal
local-ipv4	10.251.50.37

インスタンス 4

メタデータ	値
instance-id	i-1234567890abcdef0
ami-launch-index	3
public-hostname	ec2-67-202-51-226.compute-1.amazonaws.com
public-ipv4	67.202.51.226
local-hostname	ip-10-251-50-38.ec2.internal
local-ipv4	10.251.50.38

Alice は `ami-launch-index` 値を使用して、ユーザーデータのどの部分が特定のインスタンスに該当するかを判断できます。

1. いずれかのインスタンスに接続し、そのインスタンスの `ami-launch-index` を取得して、それがレプリカの 1 つであることを確認します。

   IMDSv2

   ```
   [ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/meta-data/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/ami-launch-index
   2
   ```

   次のステップでは、IMDSv2が前のIMDSv2コマンドからの保管済みトークン (期限内であると仮定) を使用するようリクエストします。

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/ami-launch-index
2
```

2. 変数としてami-launch-indexを保存します。

   IMDSv2

   ```
   [ec2-user ~]$ ami_launch_index=`curl -H "X-aws-ec2-metadata-token: $TOKEN"
    http://169.254.169.254/latest/meta-data/ami-launch-index`
   ```

   IMDSv1

   ```
   [ec2-user ~]$ ami_launch_index=`curl http://169.254.169.254/latest/meta-data/ami-
   launch-index`
   ```

3. ユーザーデータを変数として保存します。

   IMDSv2

   ```
   [ec2-user ~]$ user_data=`curl -H "X-aws-ec2-metadata-token: $TOKEN"
    http://169.254.169.254/latest/user-data`
   ```

   IMDSv1

   ```
   [ec2-user ~]$ user_data=`curl http://169.254.169.254/latest/user-data`
   ```

4. 最後に、Alice はcutコマンドを使用して、そのインスタンスに該当するユーザーデータの部分を抽出します。

   IMDSv2

   ```
   [ec2-user ~]$ echo $user_data | cut -d"|" -f"$ami_launch_index"
   replicate-every=5min
   ```

   IMDSv1

   ```
   [ec2-user ~]$ echo $user_data | cut -d"|" -f"$ami_launch_index"
   replicate-every=5min
   ```

インスタンスアイデンティティドキュメント

作成する各インスタンスには、インスタンス自体に関する情報を提供するインスタンスアイデンティティドキュメントがあります。インスタンスアイデンティティドキュメントを使用して、インスタンスの属性を検証することができます。

インスタンスアイデンティティドキュメントは、インスタンスが停止して起動、再起動、起動するときに生成されます。インスタンスアイデンティティドキュメントは、インスタンスが作成されてインスタンスメタデータサービス (IMDS) によって (プレーンテキストの JSON 形式で) 公開されます。IPv4 アドレス `169.254.169.254` は、リンクローカルアドレスで、インスタンスからのみ有効です。詳細については、Wikipedia の「リンクローカルアドレス」を参照してください。IPv6 アドレス `[fd00:ec2::254]` は、リンクローカルアドレスで、インスタンスからのみ有効です。詳細については、Wikipedia の ユニークなローカルアドレスを参照してください。

> **Note**
>
> このセクションの例では、IMDS の IPv4 アドレス `169.254.169.254` を使用します。IPv6 アドレスを使用して EC2 インスタンスのインスタンスメタデータを取得する場合は、IPv6 アドレスを有効にして使用してください。`[fd00:ec2::254]`。IMDS の IPv6 アドレスは、IMDSv2 コマンドと互換性があります。IPv6 アドレスは、Nitro System 上に構築されたインスタンス上でのみアクセスできます。

インスタンスアイデンティティドキュメントは、実行中のインスタンスからいつでも取得できます。インスタンスアイデンティティドキュメントには、以下の情報が含まれています。

データ	説明
`accountId`	インスタンスを起動した AWS アカウントの ID。
`architecture`	インスタンスの作成に使用された AMI のアーキテクチャ (i386 | x86_64 | arm64)。
`availabilityZone`	インスタンスが実行されているアベイラビリティーゾーン。
`billingProducts`	インスタンスの請求製品。

データ	説明
devpayProductCodes	廃止済み。
imageId	インスタンスの起動に使用される AMI の ID。
instanceId	インスタンスの ID。
instanceType	インスタンスのインスタンスタイプ。
kernelId	インスタンスに関連付けられているカーネルの ID (ある場合)。
marketplaceProductCodes	インスタンスの作成に使用された AMI の AWS Marketplace 製品コード。
pendingTime	インスタンスが作成された日時。
privateIp	インスタンスのプライベート IPv4 アドレス。
ramdiskId	インスタンスに関連付けられている RAM ディスクの ID (ある場合)。
region	インスタンスが実行されているリージョン。
version	インスタンスアイデンティティドキュメント形式のバージョン。

プレーンテキストの インスタンスアイデンティティドキュメント を取得する

プレーンテキストのインスタンスアイデンティティドキュメントを取得するには

インスタンスに接続し、インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを実行します。

IMDSv2

```
$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/instance-identity/document
```

IMDSv1

```
$ curl http://169.254.169.254/latest/dynamic/instance-identity/document
```

以下は出力例です。

```
{
    "devpayProductCodes" : null,
    "marketplaceProductCodes" : [ "1abc2defghijklm3nopqrs4tu" ],
    "availabilityZone" : "us-west-2b",
    "privateIp" : "10.158.112.84",
    "version" : "2017-09-30",
    "instanceId" : "i-1234567890abcdef0",
    "billingProducts" : null,
    "instanceType" : "t2.micro",
    "accountId" : "123456789012",
    "imageId" : "ami-5fb8c835",
    "pendingTime" : "2016-11-19T16:32:11Z",
    "architecture" : "x86_64",
    "kernelId" : null,
    "ramdiskId" : null,
    "region" : "us-west-2"
}
```

インスタンスアイデンティティドキュメント の検証

インスタンスアイデンティティドキュメントの内容を重要な用途に使用する場合は、使用前にその内容と真正性を検証する必要があります。

プレーンテキストのインスタンスアイデンティティドキュメントには、ハッシュ化および暗号化された署名が 3 つあります。これらの署名を使用して、インスタンスアイデンティティドキュメントの作成元および真正性とそれに含まれている情報を検証できます。提供されている署名は次のとおりです。

- base64 でエンコードされた署名—RSA キーペアを使用して暗号化されたインスタンスアイデンティティドキュメントの base64 でエンコードされた SHA256 ハッシュです。
- PKCS7 署名—DSA キーペアを使用して暗号化されたインスタンスアイデンティティドキュメントの SHA1 ハッシュです。
- RSA-2048 署名—RSA-2048 キーペアを使用して暗号化されたインスタンスアイデンティティドキュメントの SHA256 ハッシュです。

それぞれの署名は、インスタンスメタデータの異なるエンドポイントで取得できます。ハッシュ化と暗号化の要件に応じて、これらの署名のいずれかを使用できます。署名を検証するには、対応する AWS パブリック証明書を使用する必要があります。

以下のトピックでは、それぞれの署名を使用してインスタンスアイデンティティドキュメントを検証するための詳細な手順について説明します。

- PKCS7 署名を使用した インスタンスアイデンティティドキュメント の検証
- base64 でエンコードされた署名を使用した インスタンスアイデンティティドキュメント の検証
- RSA-2048 署名を使用した インスタンスアイデンティティドキュメント の検証

PKCS7 署名を使用した インスタンスアイデンティティドキュメント の検証

このトピックでは、PKCS7 署名と AWS DSA パブリック証明書を使用して、インスタンスアイデンティティドキュメントを検証する方法について説明します。

PKCS7 署名と AWS DSA パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから PKCS7 署名を取得し、必要なヘッダーとフッターとともにpkcs7という名前の新しいファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   $ echo "-----BEGIN PKCS7-----" >> pkcs7 \
   && TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/dynamic/instance-identity/pkcs7 >> pkcs7 \
   && echo "" >> pkcs7 \
   && echo "-----END PKCS7-----" >> pkcs7
   ```

   IMDSv1

   ```
   $ echo "-----BEGIN PKCS7-----" >> pkcs7 \
   && curl -s http://169.254.169.254/latest/dynamic/instance-identity/pkcs7
    >> pkcs7 \
   ```

```
&& echo "" >> pkcs7 \
&& echo "-----END PKCS7-----" >> pkcs7
```

3. AWS パブリック証明書でリージョン用に DSA パブリック証明書を検索し、certificate という名前の新しいファイルに追加します。
4. OpenSSL の smime コマンドを使用して、署名を検証します。署名を検証する必要があることを示す -verify オプションと証明書を検証する必要がないことを示す -noverify オプションを含めます。

```
$ openssl smime -verify -in pkcs7 -inform PEM -certfile certificate -noverify | tee
 document
```

署名が有効な場合は、Verification successful メッセージが表示されます。

また、このコマンドでは、インスタンスアイデンティティドキュメントの内容を、document という名前の新しいファイルにも書き込みます。以下のコマンドにより、このファイルの内容を、インスタンスメタデータからのインスタンスアイデンティティドキュメントの内容と比較できます。

```
$ openssl dgst -sha256 < document
```

```
$ curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/document | openssl dgst -sha256
```

署名を検証できない場合は、AWS Supportにお問い合わせください。

base64 でエンコードされた署名を使用した インスタンスアイデンティティドキュメント の検証

このトピックでは、base64 でエンコードされた署名と AWS RSA パブリック証明書を使用して、インスタンスアイデンティティドキュメントを検証する方法について説明します。

base64 でエンコードされた署名と AWS RSA パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから base64 でエンコードされた署名を取得し、バイナリに変換して、signature という名前のファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

IMDSv2

```
$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/signature | base64 -d >> signature
```

IMDSv1

```
$ curl -s http://169.254.169.254/latest/dynamic/instance-identity/signature |
 base64 -d >> signature
```

3. インスタンスメタデータからプレーンテキストのインスタンスアイデンティティドキュメントを取得し、document という名前のファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   $ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
   metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
   dynamic/instance-identity/document >> document
   ```

   IMDSv1

   ```
   $ curl -s http://169.254.169.254/latest/dynamic/instance-identity/document
    >> document
   ```

4. AWS パブリック証明書でリージョン用に RSA パブリック証明書を検索し、certificate という名前の新しいファイルに追加します。

5. AWS RSA パブリック証明書からパブリックキーを抽出し、key という名前のファイルに保存します。

   ```
   $ openssl x509 -pubkey -noout -in certificate >> key
   ```

6. OpenSSL の dgst コマンドを使用して、インスタンスアイデンティティドキュメントを検証します。

   ```
   $ openssl dgst -sha256 -verify key -signature signature document
   ```

署名が有効な場合は、Verification successful メッセージが表示されます。

また、このコマンドでは、インスタンスアイデンティティドキュメントの内容を、document という名前の新しいファイルにも書き込みます。以下のコマンドにより、このファイルの内容を、インスタンスメタデータからのインスタンスアイデンティティドキュメントの内容と比較できます。

```
$ openssl dgst -sha256 < document
```

```
$ curl -s -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
dynamic/instance-identity/document | openssl dgst -sha256
```

署名を検証できない場合は、AWS Supportにお問い合わせください。

RSA-2048 署名を使用した インスタンスアイデンティティドキュメント の検証

このトピックでは、RSA-2048 署名と AWS RSA-2048 パブリック証明書を使用して、インスタンスアイデンティティドキュメントを検証する方法について説明します。

RSA-2048 署名と AWS RSA-2048 パブリック証明書を使用してインスタンスアイデンティティドキュメントを検証するには

1. インスタンスに接続します。
2. インスタンスメタデータから RSA-2048 署名を取得し、必要なヘッダーとフッターとともにrsa2048 という名前のファイルに追加します。インスタンスで使用されている IMDS のバージョンに応じて、次のいずれかのコマンドを使用します。

   IMDSv2

   ```
   $ echo "-----BEGIN PKCS7-----" >> rsa2048 \
   && TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
   metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
   dynamic/instance-identity/rsa2048 >> rsa2048 \
   && echo "" >> rsa2048 \
   && echo "-----END PKCS7-----" >> rsa2048
   ```

IMDSv1

```
$ echo "-----BEGIN PKCS7-----" >> rsa2048 \
&& curl -s http://169.254.169.254/latest/dynamic/instance-identity/rsa2048
 >> rsa2048 \
&& echo "" >> rsa2048 \
&& echo "-----END PKCS7-----" >> rsa2048
```

3. AWS パブリック証明書でリージョン用に RSA-2048 パブリック証明書を検索し、certificate という名前の新しいファイルに追加します。
4. OpenSSL の smime コマンドを使用して、署名を検証します。署名を検証する必要があることを示す -verify オプションと証明書を検証する必要がないことを示す -noverify オプションを含めます。

```
$ openssl smime -verify -in rsa2048 -inform PEM -certfile certificate -noverify |
 tee document
```

署名が有効な場合は、Verification successful メッセージが表示されます。署名を検証できない場合は、AWS Supportにお問い合わせください。

AWS パブリック証明書

以下のトピックで説明するように、AWS パブリック証明書を使用してインスタンスのインスタンス ID ドキュメントの内容を検証できます。

- PKCS7 署名を使用した検証
- base64 でエンコードされた署名を使用した検証
- RSA-2048 署名を使用した検証

リージョンと使用している検証手順に適した証明書を使用していることを確認してください。PKCS7 の署名を検証する場合は、DSA 証明書を使用してください。base6 でエンコードされた署名を検証する場合は、RSA 証明書を使用してください。RSA-2048 署名を検証する場合は、RSA-2048 証明書を使用してください。

以下の各リージョンを展開すると、リージョン固有の証明書が表示されます。

米国東部 (オハイオ) - us-east-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAM07oeX4xevdMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjA2MTAx
MjU4MThaGA8yMTk1MTExNDEyNTgxOFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA6v6kGMnRmFDLxBEqXzP4npnL65OO0kmQ7w8YXQygSdmNIoScGSU5wfh9
mZdcvCxCdxgALFsFqPvH8fqiE9ttI0fEfuZvHOs8wUsIdKr0Zz0MjSx3cik4tKET
ch0EKfMnzKOgDBavraCDeX1rUDU0Rg7HFqNAOry3uqDmnqtk00XC9GenS3z/7ebJ
fIBEPAam5oYMVFpX6M6St77WdNE8wEU8SuerQughiMVx9kMB07imeVHBiELbMQ0N
lwSWRL/61fA02keGSTfSp/0m3u+lesf2VwVFhqIJs+JbsEscPxOkIRlzy8mGd/JV
ONb/DQpTedzUKLgXbw7KtO3HTG9iXQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU2CTGYE5fTjx7gQXzdZSGPEWAJY4wgY4GA1UdIwSBhjCBg4AU2CTG
YE5fTjx7gQXzdZSGPEWAJY6hYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAM07oeX4xevdMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBANdqkIpVypr2PveqUsAKke1wKCOSuw1UmH9k
xX1/VRoHbrI/UznrXtPQOPMmHA2LKSTedwsJuorUn3cFH6qNs8ixBDrl8pZwfKOY
IBJcTFBbI1xBEFkZoO3wczzo5+8vPQ60RVqAaYb+iCa1HFJpccC3Ovajfa4GRdNb
n6FYnluIcDbmpcQePoVQwX7W3oOYLB1QLN7fE6H1j4TBIsFdO3OuKzmaifQlwLYt
DVxVCNDabpOr6Uozd5ASm4ihPPoEoKo7Ilp0fOT6fZ41U2xWA4+HF/89UoygZSo7
K+cQ90xGxJ+gmlYbLFR5rbJOLfjrgDAb2ogbFy8LzHo2ZtSe60M=
-----END CERTIFICATE-----
```

米国東部 (バージニア) – us-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
```

```
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALFpzEAVWaQZMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
ODU5MTJaGA8yMTk1MDExNzA4NTkxMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAjS2vqZu9mEOhOq+0bRpAbCUiapbZMFNQqRg7kTlr7Cf+gDqXKpHPjsng
SfNz+JHQd8WPI+pmNs+q0Z2aTe23klmf2U52KH9/j1k8RlIbap/yFibFTSedmegX
```

```
E5r447GbJRsHUmuIIfZTZ/oRlpuIIO5/Vz7SOj22tdkdY2ADp7caZkNxhSP915fk
2jJMTBUOzyXUS2rBU/ulNHbTTeePjcEkvzVYPahD30TeQ+/A+uWUu89bHSQOJR8h
Um4cFApzZgN3aD5j2LrSMu2pctkQwf9CaWyVznqrsGYjYOY66LuFzSCXwqSnFBfv
fFBAFsjCgY24G2DoMyYkF3MyZlu+rwIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUrynSPp4uqSECwy+PiO4qyJ8TWSkwgY4GA1UdIwSBhjCBg4AUrynS
Pp4uqSECwy+PiO4qyJ8TWSmhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALFpzEAVWaQZMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBADW/s8lXijwdP6NkEoH1m9XLrvK4YTqkNfR6
er/uRRgTx2QjFcMNrx+g87gAml11z+D0crAZ5LbEhDMs+JtZYR3ty0HkDk6SJM85
haoJNAFF7EQ/zCp1EJRIkLLsC7bcDL/Eriv1swt78/BB4RnC9W9kSp/sxd5svJMg
N9a6FAplpNRsWAnbP8JBlAP93oJzblX2LQXgykTghMkQO7NaY5hg/H5o4dMPclTK
lYGqlFUCH6A2vdrxmpKDLmTn5//5pujdD2MN0df6sZWtxwZ0osljV4rDjm9Q3VpA
NWIsDEcp3GUB4proOR+C7PNkY+VGODitBOw09qBGosCBstwyEqY=
-----END CERTIFICATE-----
```

米国西部 (北カリフォルニア) - us-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
```

```
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluY4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJANNPkIpcyEtIMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
OTAzMDdaGA8yMTk1MDQwMzA5MDMwN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEApHQGvHvq3SVCzDrC7575BW7GWLzcj8CLqYcL3YY7Jffupz7OjcftO57Z
4fo5Pj0CaS8DtPzh8+8vdwUSMbiJ6cDd3ooio3MnCq6DwzmsY+pY7CiI3UVG7KcH
4TriDqr1Iii7nB5MiPJ8wTeAqX89T3SYaf6Vo+4GCb3LCDGvnkZ9TrGcz2CHkJsj
AIGwgopFpwhIjVYm7obmuIxSIUv+oNH0wXgDL029Zd98SnIYQd/njiqkzE+lvXgk
4h4Tu17xZIKBgFcTtWPky+POGu81DYFqiWVEyR2JKKm2/iR1dL1YsT39kbNg47xY
aR129sS4nB5Vw3TRQA2jL0ToTIxzhQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUgepyiONs8j+q67dmcWu+mKKDa+gwgY4GA1UdIwSBhjCBg4AUgepy
iONs8j+q67dmcWu+mKKDa+ihYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJANNPkIpcyEtIMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAGLFWyutf1u0xcAc+kmnMPqtc/Q6b79VIX0E
tNoKMI2KR8lcV8ZElXDb0NC6v8UeLpe1WBKjaWQtEjL1ifKg9hdY9RJj4RXIDSK7
33qCQ8juF4vep2U5TTBd6hfWxt1Izi88xudjixmbpUU4YKr8UPbmixldYR+BEx0u
B1KJi9l1lxvuc/Igy/xeHOAZEjAXzVvHp8Bne33VVwMiMxWECZCiJxE4I7+Y6fqJ
pLLSFFJKbNaFyXlDiJ3kXyePEZSc1xiWeyRB2ZbTi5eu7vMG4i3AYWuFVLthaBgu
```

```
lPfHafJpj/JDcqt2vKUKfur5edQ6j1CGdxqqjawhOTEqcN8m7us=
-----END CERTIFICATE-----
```

米国西部 (オレゴン) - us-west-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
```

```
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALZL3lrQCSTMMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
OTAxMzJaGA8yMTk1MDExNzA5MDEzMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA02Y59qtAA0a6uzo7nEQcnJ26OKF+LRPwZfixBH+EbEN/Fx0gYy1jpjCP
s5+VRNg6/WbfqAsV6X2VSjUKN59ZMnMY9ALA/Ipz0n00Huxj38EBZmX/NdNqKm7C
qWu1q5kmIvYjKGiadfboU8wLwLcHo8ywvfgI6FiGGsEO9VMC56E/hL6Cohko11LW
dizyvRcvg/IidazVkJQCN/4zC9PUOVyKdhW33jXy8BTg/QH927QuNk+ZzD7HH//y
tIYxDhR6TIZsSnRjz3bOcEHxt1nsidc65mY0ejQty4hy7ioSiapw316mdbtE+RTN
fcH9FPIFKQNBpiqfAW5Ebp3Lal3/+wIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU7coQx8Qnd75qA9XotSWT3IhvJmowgY4GA1UdIwSBhjCBg4AU7coQ
x8Qnd75qA9XotSWT3IhvJmqhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALZL3lrQCSTMMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAFZ1e2MnzRaXCaLwEC1pW/f0oRG8nHrlPZ9W
OYZEWbh+QanRgaikBNDtVTwARQcZm3z+HWSkaIx3cyb6vM0DSkZuiwzm1LJ9rDPc
aBm03SEt5v8mcc7sXWvgFjCnUpzosmky6JheCD4O1Cf8k0olZ93FQnTrbg62OK0h
83mGCDeVKU3hLH97FYoUq+3N/IliWFDhvibAYYKFJydZLhIdlCiiB99AM6Sg53rm
oukS3csyUxZyTU2hQfdjyo1nqW9yhvFAKjnnggiwxNKTTPZzstKW8+cnYwiiTwJN
QpVoZdt0SfbuNnmwRUMi+QbuccXweav29QeQ3ADqjgB0CZdSRKk=
-----END CERTIFICATE-----
```

アフリカ (ケープタウン) – af-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7DCCAqwCCQCncbCtQbjuyzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTA2MDQxMjQ4MDVaFw00
NTA2MDQxMjQ4MDVaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
```

```
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbYwggErBgcqhkjOOAQBMIIBHgKBgQC12Nr1gMrHcFSZ7S/A
pQBSCMHWmn2qeoQTMVWqe50fnTd0zGFxDdIjKxUK58/8zjWG5uR4TXRzmZpGpmXB
bSufAR6BGqud2LnT/HIWGJAsnX2uOtSyNfCoJigqwhea5w+CqZ6I7iBDdnB4TtTw
qO6TlnExHFVj8LMkylZgiaE1CQIVAIhdobse4K0QnbAhCL6R2euQzloXAoGAV/21
WUuMz/79Ga0JvQcz1FNy1sT0pU9rU4TenqLQIt5iccn/7EIfNtvVO5TZKulIKq7J
gXZr0x/KIT8zsNweetLOaGehPIYRMPX0vunMMR7hN7qA7W17WZv/76adywIsnDKq
ekfe15jinaX8MsKUdyDK7Y+ifCG4PVhoM4+W2XwDgYQAAoGAIxOKbVgwLxbn6Pi2
6hBOihFv16jKxAQI0hHzXJLV0Vyv9QwnqjJJRfOCy3dB0zicLXiIxeIdYfvqJr+u
hlN8rGxEZYYJjEUKMGvsc0DW85jonXz0bNfcP0aaKH0lKKVjL+OZi5n2kn9wgdo5
F3CVnMl8BUra8A1Tr2yrrE6TVZ4wCQYHKoZIzjgEAwMvADAsAhQfa7MCJZ+/TEY5
AUr0J4wm8VzjoAIUSYZVu2NdRJ/ERPmDfhW5EsjHlCA=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAKumfZiRrNvHMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTExMjcw
NzE0MDVaGA8yMTk5MDUwMjA3MTQwNVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQDFd571nUzVtke3rPyRkYfvs3jh0C0EMzzG72boyUNjnfw1+m0TeFraTLKb9T6F
7TuB/ZEN+vmlYqr2+5Va8U8qLbPF0bRH+FdaKjhgWZdYXxGzQzU3ioy5W5ZM1VyB
7iUsxEAlxsybC3ziPYaHI42UiTkQNahmoroNeqVyHNnBpQIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAAJLylWyElEgOpW4B1XPyRVD4pAds8Guw2+krgqkY0HxLCdjosuH
RytGDGN+q75aAoXzW5a7SGpxLxk6Hfv0xp3RjDHsoeP0i1d8MD3hAC5ezxS4oukK
s5gbPOnokhKTMPXbTdRn5ZifCbWlx+bYN/mTYKvxho7b5SVg2o1La9aK
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAIFI+O5A6/ZIMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTA2MDQx
MjQ4MDRaGA8yMTk4MTEwNzEyNDgwNFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAy7/WHBBHOrk+20aumT07g8rxrSM0UXgki3eYgKauPCG4Xx//vwQbuZwI
oeVmR9nqnhfij2wOcQdbLandh0EGtbxerete3IoXzd1KXJb11PVmzrzyu5SPBPuP
iCeV4qdjjkXo2YWM6t9YQ911hcG96YSp89TBXFYUh3KLxfqAdTVhuC0NRGhXpyii
```

```
j/czo9njofHhqhTr7UEyPun8NVS2QWctLQ86N5zWR3Q0GRoVqqMrJs0cowHTrVw2
9Qr7QBjjBOVbyYmtYxm/DtiKprYV/e6bCAVok015X1sZDd3oCOQNoGlv5XbHJe2o
JFD8GRRy2rkWO/lNwVFDcwec6zC3QwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQCE
goqzjpCpmMgCpszFHwvRaSMbspKtK7wNImUjrSBOfBJsfFulyg1Zgn2nDCK7kQhx
jMJmNIvXbps3yMqQ2cHUkKcKf5t+WldfeT4Vk1Rz6HSA8sd0kgVcIesIaoy2aaXU
VEB/oQziRGyKdN1d4TGYVZXG44CkrzSDvlbmfiTq5tL+kAieznVF3bzHgPZW6hKP
EXC3G/IXrXicFEe6YyE1Rakl62VncYSXiGe/i2XvsiNH3Qlmnx5XS7W0SCN0oAxW
EH9twibauv82DVg1WOkQu8EwFw8hFde9X0Rkiu0qVcuU8lJgFEvPWMDFU5sGB6ZM
gkEKTzMvlZpPbBhg99Jl
-----END CERTIFICATE-----
```

アジアパシフィック (香港) – ap–east–1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq4CCQCO7MJe5Y3VLjAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTAyMDMwMjIxMjFaFw00
NTAyMDMwMjIxMjFaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbgwggEsBgcqhkjOOAQBMIIBHwKBgQDvQ9RzVvf4MAwGbqfX
blCvCoVb9957OkLGn/04CowHXJ+vTBR7eyIa6AoXltsQXBOmrJswToFKKxT4gbuw
jK7s9QQX4CmTRWcEgO2RXtZSVjOhsUQMh+yf7Ht4OVL97LWnNfGsX2cwjcRWHYgI
7lvnuBNBzLQHdSEwMNq0Bk76PwIVAMan6XIEEPnwr4e6u/RNnWBGKd9FAoGBAOCG
eSNmxpW4QFu4pIlAykm6EnTZKKHT87gdXkAkfoC5fAfOxxhnE2HezZHp9Ap2tMV5
8bWNvoPHvoKCQqwfm+OUBlAxC/3vqoVkKL2mG1KgUH9+hrtpMTkwO3RREnKe7I5O
x9qDimJpOihrL4I0dYvy9xUOoz+DzFAW8+ylWVYpA4GFAAKBgQDbnBAKSxWr9QHY
6Dt+EFdGz6lAZLedeBKpaP53Z1DTO34J0C55YbJTwBTFGqPtOLxnUVDlGiD6GbmC
80f3jvogPR1mSmGsydbNbZnbUEVWrRhe+y5zJ3g9qs/DWmDW0deEFvkhWVnLJkFJ
9pdOu/ibRPH1lE2nz6pK7GbOQtLyHTAJBgcqhkjOOAQDAzAAMC0CFQCoJlwGtJQC
cLoM4p/jtVFOj26xbgIUUS4pDKyHaG/eaygLTtFpFJqzWHc=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICSzCCAbQCCQDtQvkVxRvK9TANBgkqhkiG9w0BAQsFADBqMQswCQYDVQQGEwJV
UzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2VhdHRsZTEYMBYGA1UE
ChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1hem9uYXdzLmNvbTAe
Fw0xOTAyMDMwMzAwMDZaFw0yOTAyMDIwMzAwMDZaMGoxCzAJBgNVBAYTAlVTMRMw
EQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgwFgYDVQQKEw9B
```

```
bWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3MuY29tMIGfMA0G
CSqGSIb3DQEBAQUAA4GNADCBiQKBgQC1kkHXYTfc7gY5Q55JJhjTieHAgacaQkiR
Pity9QPDE3b+NXDh4UdP1xdIw73JcIIG3sG9RhWiXVCHh6KkuCTqJfPUknIKk8vs
M3RXflUpBe8Pf+P92pxqPMCz1Fr2NehS3JhhpkCZVGxxwLC5gaG0Lr4rFORubjYY
Rh84dK98VwIDAQABMA0GCSqGSIb3DQEBCwUAA4GBAA6xV9f0HMqXjPHuGILDyaNN
dKcvplNFwDTydVg32MNubAGnecoEBtUPtxBsLoVYXCOb+b5/ZMDubPF9tU/vSXuo
TpYM5Bq57gJzDRaBOntQbX9bgHiUxw6XZWaTS/6xjRJDT5p3S1E0mPI3lP/eJv4o
Ezk5zb3eIf10/sqt4756
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAMoxixvs3YssMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xODA3MjAw
ODQ0NDRaGA8yMTk3MTIyMzA4NDQ0NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA4T1PNsOg0FDrGlWePoHeOSmOJTA3HCRy5LSbYD33GFU2eBrOIxoU/+SM
rInKu3GghAMfH7WxPW3etIAZiyTDDU5RLcUq2Qwdr/ZpXAWpYocNc/CEmBFtfbxF
z4uwBIN3/drM0RSbe/wP9EcgmNUGQMMZWeAji8sMtwpOblNWAP9BniUG0Flcz6Dp
uPovwDTLdAYT3TyhzlohKL3f6O48TR5yTaV+3Ran2SGRhyJjfh3FRpP4VC+z5LnT
WPQHN74Kdq35UgrUxNhJraMGCzznolUuoR/tFMwR93401GsM9fVA7SW3jjCGF81z
PSzjy+ArKyQqIpLW1YGWDFk3sf08FQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDK
2/+C3nPMgtyOFX/I3Cyk+Pui44IgOwCsIdNGwuJysdqp5VIfnjegEu2zIMWJSKGO
lMZoQXjffkVZZ97J7RNDW06oB7kj3WVE8a7U4WEOfnO/CbMUf/x99CckNDwpjgW+
K8V8SzAsQDvYZs2KaE+18GFfLVF1TGUYK2rPSZMHyX+v/TIlc/qUceBycrIQ/kke
jDFsihUMLqgmOV2hXKUpIsmiWMGrFQV4AeV0iXP8L/ZhcepLf1t5SbsGdUA3AUY1
3If8s81uTheiQjwY5t9nM0SY/1Th/tL3+RaEI79VNEVfG1FQ8mgqCK0ar4m0oZJl
tmmEJM7xeURdpBBx36Di
-----END CERTIFICATE-----
```

アジアパシフィック (ハイデラバード) - ap-south-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCArCgAwIBAgIGAXjrQ4+XMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24g
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
```

```
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYUAAoGBAJCKGBBoxIUxqBk94JHhwZZbgvbP0DAOoHENQWxp/98lI7/
YOfYJOVMJS22aCnHDurofmo5rvNIkgXi7Rztbhu
+lko9rK6DgpmpUwBU0WZtf34aZ2IWNBwHaVhHvWAQf9/46u18dMa2YucK1Wi+Vc+M
+KldrvgXmhym6ErNlzhJyMAkGByqGSM44BAMDLwAwLAIUaaPKxaOHoYvwz709xXpsQueIq+UCFFa/
GpzoDOSokl1057NU/2hnsiW4
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXjwLj9CMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
+sFcobrjvcAYm0PNRD8f4R1jAzvoLt2+qGeOTAyO1Httj6cmsYN3AP1hN5iYuppFiYsl2eNPa/
CD0Vg0BAfDFlV5rzjpA0j7TJabVh4kj7JvtD+xYMi6wEQA4x6SPONY4OeZ2+8o/
HS8nucpWDVdPRO6ciWUlMhjmDmwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAAy6sgTdRkTqELHBeWj69q60xHyUmsWqHAQ
TGGbYP0yP2qfMl0cCImzRI5W0gn8gogdervfeT7nH5ih0TWEy/QDWfkQ601L4erm4yh4YQq8vcqAPSkf04N
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAIVWfPw/X82fMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MDQx
NDMwMjhaGA8yMjAxMTIwODE0MzAyOFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAg29QEFriG+qFEjYW/v62nN70lMJY/Hevx5TtmU/VIYBPQa3HUGTBAbbI
2Tmy8UMpa8kZeaYeI3RAfiQWt0Ws7wUrBuO2Pdp518WDPaJUH7RWEuu1BDDkyZRW
NAMNPCn3ph7Od243IFcLGku7HVekel5poqRpSfojrMasjlf+CvixUeAJbmFoxUHK
kh5unzG2sZyO4wHXcJPQkRf5a8zSTPe9YZP1kXPPEv4p/jTSggaYPxXyS6QVaTlV
zLeLFZ0fesLPMeil3KYQtV7IKLQiEA2F6dxWnxNWQlyMHtdq6PucfEmVx17i/Xza
yNBRo0azY8WUNVkEXrRhp/pU8Nh3GQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU9AO1aZk9RLXk2ZvRVoUxYvQy9uwwgY4GA1UdIwSBhjCBg4AU9AO1
aZk9RLXk2ZvRVoUxYvQy9uyhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAIVWfPw/X82fMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBADexluMRQRftqViahCnauEWGdMvLCBr8A+Yr
6hJq0guoxEk/lahxR137DnfMPuSbi1Rx5QKo7oBrWfG/zsgQUnF2IwHTzwD+i/2m
XCane6FiS5RpK3lGdILq8ZmlhQk+6iI8yoZLr0LCfTh+CLgIKHOknfR5lFzgzAiF
SI8/Q9mm+uvYtSTZECI6Zh57QZPoETAG/yl+9jiOy2lAelqa/k1i+Qo8gMfOc+Pm
```

```
dwY7o6fV+oucgRlsdey6VM45LeyILQqv0RXtVzjuowanzmCCFMjgqiO9oZAWu4Oh
+F3unijELo01vZJs8s2N3KGlo3/jtUFTX6RTKShZlAPLwBi5GMI=
-----END CERTIFICATE-----
```

アジアパシフィック (ジャカルタ) - ap-southeast-3

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCArCgAwIBAgIGAXbVDEikMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24g
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYUAAoGBAPjuiEx05N3JQ6cVwntJie67D8OuNo4jGRn
+crEtL7YO0jSVB9zGE1ga
+UgRPIaYETL293S8rTJTVgXAqdpBwfaHC6NUzre8U8iJ8FMNnlP9Gw1oUIlgQBjORyynVJexoB31TDZM
+/52g9O/bpq1QqNyKbeIgyBBlc1dAtr1QLnsMAkGByqGSM44BAMDLwAwLAIUK8E6RDIRtwK+9qnaTOBhvO/
njuQCFFocyT1OxK+UDR888oNsdgtif2Sf
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXbVDG2yMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
Vbt0gQ1ebWcur2hSO7PnJifE4OPxQ7RgSAlc4/spJp1sDP+ZrS0LO1ZJfKhXf1R9S3AUwLnsc7b
+IuVXdY5LK9RKqu64nyXP5dx170zoL8loEyCSuRR2fs+04i2QsWBVP+KFNAn7P5L1EHRjkgTO8kjNKviwRV
+OkP9ab5wIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAI4WUy6+DKh0JDSzQEZNyBgNlSoSuC2owtMxCwGB6nBfzzfcekWvs
+87w/g9lNwUnUt0ZHYyh2tuBG6hVJuUEwDJ/z3wDd6wQviLOTF3MITawt9P8siR1hXqLJNxpjRQFZrgHqi
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAMtdyRcH51j9MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA0MDgx
MjM5MTZaGA8yMjAxMDkxMjEyMzkxNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
```

```
CgKCAQEAvUsKCxoH6KXRYJLeYTWAQfaBQeCwhJaR56mfUeFHJE4g8aFjWkiN4uc1
TvOyYNnIZKTHWmzmulmdinWNbwP0GiROHb/i7ro0Hhvnptyyc Gt8ag8affiIbx5X
7ohdwSN2KJ6G0IKflIx7f2NEI0oAMM/9k+T1eVF+MVWzpZoiDp8frLNkqp8+RAgz
ScZsbRfwv3u/if5xJAvdg2nCkIWDMSHEVPoz0lJo7v0ZuDtWWsL1LHnL5ozvsKEk
+ZJyEi23r+U1hIT1NTBdp4yoigNQexedtwCSr7q36oOdDwvZpqYlkLi3uxZ4ta+a
01pzOSTwMLgQZSbKWQrpMvsIAPrxoQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU1GgnGdNpbnL3lLF30Jomg7Ji9hYwgY4GA1UdIwSBhjCBg4AU1Ggn
GdNpbnL3lLF30Jomg7Ji9hahYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAMtdyRcH51j9MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBACVl00qQlatBKVeiWMrhpczsJroxDxlZTOba
6wTMZk7c3akb6XMOSZFbGaifkebPZqTHEhDlrClM2j9AIlYcCx6YCrTf4cuhn2mD
gcJN33143eOWSaeRY3ee4j+V9ne98y3kO2wLz95VrRgclPFR8po2iWGzGhwUi+FG
q8dXeCH3N0DZgQsSgQWwmdNQXZZej6RHLU/8In5trHKLY0ppnLBjn/UZQbeTyW5q
RJB3GaveXjfgFUWj2qOcDuRGaikdS+dYaLsi5z9cA3FolHzWxx9MOs8io8vKqQzV
XUrLTNWwuhZy88cOlqGPxnoRbw7TmifwPw/cunNrsjUUOgs6ZTk=
-----END CERTIFICATE-----
```

アジアパシフィック (メルボルン) - ap-southeast-4

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAXjWF7P2MAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAPRXSsQP9E3dw8QXKlrgBgEVCprLHdK/bbrMas0XMu1EhOD
+q
+0PcTr8+iwbtoXlY5MCeatWIp1GrXQjVqsF8vQqxlEuRuYKbR3nq4mWwaeGlx9AG5EjQHRa3GQ44wWHOdof0M3NRI1MP
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXjSh40SMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
+qWTGAbGsPeMX4hBMjAJUKys2NIRcRZaLM/BCew2FIPVjNtlaj6Gwn9ipU4Mlz3zIwAMWi1AvGMSreppt
+wV6MRtfOjh0Dvj/veJe88aEZJMozNgkJFRS
```

```
+WFWsckQeL56tf6kY6QTlNo8V/0CsQIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAF7vpPghH0FRo5gu49EArRNPrIvW1egM
wcgkqIwwuXYj+1rhlL+/
iMpQWjdVGEqIZSeXn5fLmdx50eegFCwND837r9e8XYTiQS143Sxt9+Yi6BZ7U7YD8kK9NBWoJxFqUeHdpRCs0O7COjT3
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAN4GTQ64zVs8MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTMx
MzMzMDBaGA8yMjAxMTIxNzEzMzMwMFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA2BYgeCr+Rk/jIAEDOHS7wJql62vc83QEwjuzk0qOFEReIZzlN1fBRNXK
g0T178Kd3gLYcE59wEFbTe/X5yOA1Lo95x1anSAo7R+Cisf9C2HQuJp+gVb+zx71
lniPF7gHziGpm0M8DdAU/IW+wkZwGbP4z7Hq9+bJ0P2ltvPJ5yxSgkFuDsI9VBHa
CLoprHsCHh2VdP8KcMgQQMmHe1NmBpyTk0ul/aLmQkCQEX6ZIRGOeq228fwlh/t+
Ho+jv87duihVKic6MrL32SlD+maX0LSDUydWdaOLLTGkh7oV7+bFuH6msrXUu+Ur
ZEP1r/MidCWMhfgrFzeTBzOHA97qxQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUcHMd1cHqzmsQ5hpUK3EMLhHdsi4wgY4GA1UdIwSBhjCBg4AUcHMd
1cHqzmsQ5hpUK3EMLhHdsi6hYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAN4GTQ64zVs8MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAI4PFyVN+7EGSObioiPnv0LL0f70SSzUZJ8p
XO90d4rWea7jIbgZ2AKb+ErynkU9xVg7XQQ5k6KDWgp/4jYFL2dqnt/YAY4PS0un
RSrYElawxLT0BcLn4rcSDC79vQe1xGC5//wDdV6b399COAHRAK6axWYy5w32u9PL
uw0cIp3Ch8JoNwcgTHKRRGzePmBeR4PNqhHTArG4/dJk6/aUO4OpX0WzI6L67CGY
6Nex3dau+gkLCK93dTEkrXtyXHu4wB0J9zd1w+iQ0SEa9eKc78/NjEsF/FZdGrWC
t57lIMOOXJhQ1kRgSwNeZdQWV1dRakvO6sfcvVYkfj1wAvZvvAw=
-----END CERTIFICATE-----
```

アジアパシフィック (ムンバイ) - ap-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
```

```
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAPRYyD8TtmC0MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjAzMDcx
MDQ1MDFaGA8yMTk1MDgxMTEwNDUwMVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
```

```
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA0LSS5I/eCT2PM0+qusorBx67QL26BIWQHd/yF6ARtHBb/1DdFLRqE5Dj
07Xw7eENC+T79mOxOAbeWg91KaODOzw6i9I/2/HpK0+NDEdD6sPKDA1d45jRra+v
CqAjI+nV9Vw91wv7HjMk3RcjWGziM8/hw+3YNIutt7aQzZRwIWlBpcqx3/AFd8Eu
2UsRMSHgkGUW6UzUF+h/U8218XfrauKNGmNKDYUhtmyBrHT+k6J0hQ4pN7fe6h+Z
w9RVHm24BGhlLxLHLmsOIxvbrF277uX9Dxu1HfKfu5D2kimTY7xSZDNLR2dt+kNY
/+iWdIeEFpPT0PLSILt52wP6stF+3QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBI
E6w+WWC2gCfoJO6c9HMyGLMFEpqZmz1n5IcQt1h9iyO7Vkm1wkJiZsMhXpk73zXf
TPxuXEacTX3SOEa07OIMCFwkusO5f6leOyFTynHCzBgZ3U0UkRVZA3WcpbNB6Dwy
h7ysVlqyT9WZd7EOYm5j5oue2G2xdei+6etgn5UjyWm6liZGrcOF6WPTdmzqa6WG
ApEqanpkQd/HM+hUYex/ZS6zEhd4CCDLgYkIjlrFbFb3pJ1OVLztIfSN5J4Oolpu
JVCfIq5u1NkpzL7ys/Ub8eYipbzI6P+yxXiUSuF0v9b98ymczMYjrSQXIf1e8In3
OP2CclCHoZ8XDQcvvKAh
-----END CERTIFICATE-----
```

アジアパシフィック (大阪) – ap-northeast-3

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
```

```
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAMn1yPk22ditMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNzA3MTkx
MTEyNThaGA8yMTk2MTIyMjExMTI1OFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEArznEYef8IjhrJoazI0QGZkmlmHm/4rEbyQbMNifxjsDE8YWtHNwaM91z
zmyK6Sk/tKlWxcnl3g31iq305ziyFPEewe5Qbwf1iz2cMsvfNBcTh/E6u+mBPH3J
gvGanqUJt6c4IbipdEouIjjnynyVWd4D6erLl/ENijeR1OxVpaqSW5SBK7jms49E
pw3wtbchEl3qsE42Ip4IYmWxqjgaxB7vps91n4kfyzAjUmklcqTfMfPCkzmJCRgp
Vh1C79vRQhmriVKD6BXwfZ8tG3a7mijeDn7kTsQzgO07Z2SAE63PIO48JK8HcObH
tXORUQ/XF1jzi/SIaUJZT7kq3kWl8wIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBj
ThtO9dLvU2QmKuXAhxXjsIdlQgGG3ZGh/Vke4If1ymgLx95v2Vj9Moxk+gJuUSRL
BzFte3TT6b3jPolbECgmAorjj8NxjC17N8QAAI1d0S0gI8kqkG7V8iRyPIFekv+M
pcai1+cIv5IV5qAz8QOMGYfGdYkcoBjsgiyvMJu/2N2UbZJNGWvcEGkdjGJUYYOO
NaspCAFm+6HA/K7BD9zXB1IKsprLgqhiIUgEaW3UFEbThJT+z8UfHG9fQjzzfN/J
nT6vuY/0RRu1xAZPyh2gr5okN/s6rnmh2zmBHU1n8cbCc64MVfXe2g3EZ9Glq/9n
izPrI09hMypJDP04ugQc
-----END CERTIFICATE-----
```

アジアパシフィック (ソウル) - ap-northeast-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANuCgCcHtOJhMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA5MTQx
NTU3NDRaGA8yMTk1MDIxNzE1NTc0NFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA66iNv6pJPmGM20W8HbVYJSlKcAg2vUGx8xeAbzZIQdpGfkabVcUHGB6m
Gy59VXDMDlrJckDDk6dxUOhmcX9z785TtVZURq1fua9QosdbTzX4kAgHGdp4xQEs
mO6QZqg5qKjBP6xr3+PshfQ1rB8Bmwg0gXEm22CC7o77+7N7Mu2sWzWbiUR7vil4
9FjWS8XmMNwFTlShp4l1TDTevDWW/uYmC30RThM9S4QPvTZ0rAS18hHVam8BCTxa
LHaVCH/Yy52rsz0hM/FlghnSnK105ZKj+b+KIp3adBL8OMCjgc/Pxi0+j3HQLdYE
32+FaXWU84D2iP2gDT28evnstzuYTQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQC1
mA4q+12pxy7By6g3nBk1s34PmWikNRJBwOqhF8ucGRv8aiNhRRye9lokcXomwo8r
KHbbqvtK85l0xUZp/Cx4sm4aTgcMvfJP29jGLclDzeqADIvkWEJ4+xncxSYVlS9x
+78TvF/+8h9U2LnSl64PXaKdxHy2IsHIVRN4GtoaP2Xhpa1S0M328Jykq/571nfN
1WRD1c/fQf1edgzRjhQ4whcAhv7WRRF+qTbfQJ/vDxy8lkiOsvU9XzUaZ0fZSfXX
wXxZamQbONvFcxVHY/0PSiM8nQoUmkkBQuKleDwRWvkoJKYKyr3jvXK7HIWtMrO4
jmXe0aMy3thyK6g5sJVg
-----END CERTIFICATE-----
```

アジアパシフィック (シンガポール) - ap-southeast-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
```

```
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAJVMGw5SHkcvMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
ODU3MTlaGA8yMTk1MDQwMzA4NTcxOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAlaSSLfBl7OgmikjLReHuNhVuvM20dCsVzptUyRbut+KmIEEc24wd/xVy
2RMIrydGedkW4tUjkUyOyfET5OAyT43jTzDPHZTkRSVkYjBdcYbe9o/0Q4P7IVS3
XlvwrUu0qo9nSID0mxMnOoF1l8KAqnn10tQ0W+lNSTkasW7QVzcb+3okPEVhPAOq
MnlY3vkMQGI8zX4iOKbEcSVIzf6wuIffXMGHVC/JjwihJ2USQ8fq6oy686g54P4w
ROg415kLYcodjqThmGJPNUpAZ7MOc5Z4pymFuCHgNAZNvjhZDA842Ojecqm62zcm
Tzh/pNMNeGCRYq2EQX0aQtYOIj7bOQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
```

```
BgNVHQ4EFgQU6SSB+3qALorPMVNjToM1Bj3oJMswgY4GA1UdIwSBhjCBg4AU6SSB
+3qALorPMVNjToM1Bj3oJMuhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAJVMGw5SHkcvMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAF/0dWqkIEZKg5rca8o0P0VS+tolJJE/FRZO
atHOeaQbWzyac6NEwjYeeV2kY63skJ+QPuYbSuIBLM8p/uTRIvYM4LZYImLGUvoO
IdtJ8mAzq8CZ3ipdMs1hRqF5GRp8lg4w2QpX+PfhnW47iIOBiqSAUkIr3Y3BDaDn
EjeXF6qS4iPIvBaQQ0cvdddNh/pE33/ceghbkZNTYkrwMyBkQlRTTVKXFN7pCRUV
+L9FuQ9y8mP0BYZa5e1sdkwebydU+eqVzsil98ntkhpjvRkaJ5+Drs8TjGaJWlRw
5WuOr8unKj7YxdL1bv7//RtVYVVi296ldoRUYv4SCvJF11z0OdQ=
-----END CERTIFICATE-----
```

アジアパシフィック (シドニー) - ap-southeast-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
```

```
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAL2bOgb+dq9rMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
OTAwNTdaGA8yMTk1MDQwMzA5MDA1N1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAmRcyLWraysQS8yDC1b5Abs3TUaJabjqWu7d5gHik5Icd6dKl8EYpQSeS
vz6pLhkgO4xBbCRGlgE8LS/OijcZ5HwdrxBiKbicR1YvIPaIyEQQvF5sX6UWkGYw
Ma5IRGj4YbRmJkBybw+AAV9Icb5LJNOMWPi34OWM+2tMh+8L234v/JA6ogpdPuDr
sM6YFHMZ0NWo58MQ0FnEj2D7H58Ti//vFPl0TaaPWaAIRF85zBiJtKcFJ6vPidqK
f2/SDuAvZmyHC8ZBHg1moX9bR5FsU3QazfbW+c+JzAQWHj2AaQrGSCITxCMlS9sJ
l51DeoZBjnx8cnRe+HCaC4YoRBiqIQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU/wHIo+r5U31VIsPoWoRVsNXGxowwgY4GA1UdIwSBhjCBg4AU/wHI
o+r5U31VIsPoWoRVsNXGxoyhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAL2bOgb+dq9rMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBACobLvj8IxlQyORTz/9q7/VJL509/p4HAeve
92riHp6+Moi0/dSEYPeFTgdWB9W3YCNc34Ss9TJq2D7t/zLGGlbI4wYXU6VJjL0S
hCjWeIyBXUZOZKFCb0DSJeUElsTRSXSFuVrZ9EAwjLvHni3BaC9Ve34iP71ifr75
8Tpk6PEj0+JwiijFH8E4GhcV5chB0/iooU6ioQqJrMwFYnwo1cVZJD5v6D0mu9bS
TMIJLJKv4QQQqPsNdjiB7G9bfkB6trP8fUVYLHLsVlIy5lGx+tgwFEYkG1N8IOO/
2LCawwaWm8FYAFd3IZl04RImNs/IMG7VmH1bf4swHOBHgCN1uYo=
-----END CERTIFICATE-----
```

アジアパシフィック (東京) - ap-northeast-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAL9KIB7Fgvg/MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
OTAwMjVaGA8yMTk1MDExNzA5MDAyNVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAz0djWUcmRW85C5CiCKPFiTIvj6y2OuopFxNE5d3Wtab10bm06vnXVKXu
tz3AndG+Dg0zIL0gMlU+QmrSR0PH2PfV9iejfLak9iwdm1WbwRrCEAj5VxPe0Q+I
KeznOtxzqQ5Wo5NLE9bA61sziUAFNVsTFUzphEwRohcekYyd3bBC4v/RuAjCXHVx
40z6AIksnAOGN2VABMlTeMNvPItKOCIeRLlllSqXX1gbtL1gxSW40JWdF3WPB68E
e+/1U3F7OEr7XqmNODOL6yh92QqZ8fHjG+afOL9Y2Hc4g+P1nk4w4iohQOPABqzb
MPjK7B2Rze0f9OEc51GBQu13kxkWWQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU5DS5IFdU/QwYbikgtWvkU3fDwRgwgY4GA1UdIwSBhjCBg4AU5DS5
IFdU/QwYbikgtWvkU3fDwRihYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAL9KIB7Fgvg/MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAG/N7ua8IE9IMyno0n5T57erBvLTOQ79fIJN
Mf+mKRM7qRRsdg/eumFft0rLOKo54pJ+Kim2cngCWNhkzctRHBV567AJNt4+ZDG5
hDgV0IxWO1+eaLE4qzqWP/9VrO+p3reuumgFZLVpvVpwXBBeBFUf2drUR14aWfI2
L/6VGINXYs7uP8v/2VBS7r6XZRnPBUy/R4hv5efYXnjwA9gq8+a3stC2ur8m5ySl
faKSwE4H320yAyaZWH4gpwUdbUlYgPHtm/ohRtiWPrN7KEG5Wq/REzMIjZCnxOfS
6KR6PNjlhxBsImQhmBvz6j5PLQxOxBZIpDoiK278e/1Wqm9LrBc=
-----END CERTIFICATE-----
```

カナダ (中部) - ca-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
```

```
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAJNKhJhaJOuMMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjA3Mjkx
MTM3MTdaGA8yMTk2MDEwMjExMzcxN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAhDUh6j1ACSt057nSxAcwMaGr8Ez87VA2RW2HyY8l9XoHndnxmP50Cqld
+26AJtltlqHpI1YdtnZ6OrVgVhXcVtbvte0lZ3ldEzC3PMvmISBhHs6A3SWHA9ln
```

```
InHbToLX/SWqBHLOX78HkPRaG2k0COHpRy+fG9gvz8HCiQaXCbWNFDHZev9OToNI
xhXBVzIa3AgUnGMalCYZuh5AfVRCEeALG60kxMMC8IoAN7+HG+pMdqAhJxGUcMO0
LBvmTGGeWhi04MUZWfOkwn9JjQZuyLg6B1OD4Y6s0LB2P1MovmSJKGY4JcF8Qu3z
xxUbl7Bh9pvzFR5gJN1pjM2n3gJEPwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQAJ
UNKM+gIIHNk0G0tzv6vZBT+o/vt+tIp8lEoZwaPQh1121iw/I7ZvhMLAigx7eyvf
IxUt9/nf8pxWaeGzi98RbSmbap+uxYRynqe1p5rifTamOsguuPrhVpl12OgRWLcT
rjg/K60UMXRsmg2w/cxV45pUBcyVb5h6Op5uEVAVq+CVns13ExiQL6kk3guG4+Yq
LvP1p4DZfeC33a2Rfre2IHLsJH5D4SdWcYqBsfTpf3FQThH0l0KoacGrXtsedsxs
9aRd7OzuSEJ+mBxmzxSjSwM84Ooh78DjkdpQgv967p3d+8NiSLt3/n7MgnUy6WwB
KtDujDnB+ttEHwRRngX7
-----END CERTIFICATE-----
```

カナダ西部 (カルガリー) — ca-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAYPouptUMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAMITzTJUa6cBsIfdHN69zW/
aHjUB4r1ZfKb1FMhIp9EZtEf5n+06oXjUG2+dKRSlFQeEK333ehNZsPd6uqey6TYKtHpFb5XRLS8BpqB
+7gnbAd0CBZM5o4NWesSQ1GLnTdQcGZkYG/
QESkbadoCXQTifCujJE682hTDLIVt1d4ewwCQYHKoZIzjgEAwMvADAsAhRJc4gRS/HWTkCR2MESaQEe/
jOMNQIUNoTwLvuPrmGPupPlGiHe0veZiO8=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAYPou9weMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
v4XBVH13ZCMgqlRHMqV8AWI5iO6gFn2A9sN3AZXTMqwtZeiDdebq3k6Wt7ieYvpXTg0qvgsjQIovRZWaBDBJy9x8C2hW
+w9lMQjFHkJ7Jy/
PHCJ69EzebQIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAGe9Snkz1A6rHBH6/5kDtYvtPYwhx2sXNxztbhkXErFk40Nw5l4
gvDVtWG7qyb6fAqgoisyAbk8K9LzxSim2S1nmT9vD84B/t/VvwQBylc
+ej8kRxMH7fquZLp7IXfmtBzyUqu6Dpbne+chG2
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALyTn5IHrIZjMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMzEyMDcx
NTM3MDFaGA8yMjAzMDUxMzE1MzcwMVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA1GP5os424BJmGPCKOSgOclP7lzUiB85du03M4hfjzSOszsBpmBGFDLz1
owYHtIxlq3+Vi1Lt5Qlx3id/ov1QyaBPFWXVeklHVXy9vieCcI3TdjGjTl1W/8MM
m3X26QPcsnHM/Kk2wJ7s186MrqmdSsp3SCPpxv4vEG2Q9yR2bXY41hpc2rWlW8qU
D0JGXluvmmAdFnto20l1XWZ6xFen1h60DRugek/ufCbN+ljKy0xLqPoavH0Ybjsb
UpsAsBs7phaoN+X/5hIERfbp5LfVnqq54pNG5KNu4KynfW9+kA/WS4cJ6FTTN5t+
y0P1HvcL+BL2RuDy6T2bB21xw5WqtQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQURTvu/Dd4zDnmS5G5CfVLnmUBNOswgY4GA1UdIwSBhjCBg4AURTvu
/Dd4zDnmS5G5CfVLnmUBNOuhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALyTn5IHrIZjMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAFt523A3Aug6/F8xxyITgA8gkU0btFhlXNSP
U4U2OQ9n0tWI9WqnKNWH3KBxwY5EPitU6b3LM4xc9lDWpz7h2Pto+WhxP9LVKe6f
r8r7teTLCVZ7cfYZHzHg+flZjVpAgzE5BVfrRlj3QKpv0hYT3J1wMtI++Vorq5Nf
aPjzedehJLhmZVALwnfqfLrgv6/gmraP9Vmoa8U4D6AljNiQGYaLwyoPoRm3bUs2
v1Mh9GkEQ1b9+1pFXcqgzJJTGRuiPCyPbECI79FAnx5JM/CkGJV8HlOmjIW1qkK1
Y2qT7wzErrKLJyB53Pw15BdIM1onbDAQreZb0yZQLdoEl/tx7Uk=
-----END CERTIFICATE-----
```

欧州 (フランクフルト) - eu-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
```

```
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAKD+v6LeR/WrMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
OTA4MTlaGA8yMTk1MDExNzA5MDgxOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAka8FLhxs1cSJGK+Q+q/vTf8zVnDAPZ3U6oqppOW/cupCtpwMAQcky8DY
Yb62GF7+C6usniaq/9W6xPn/3o//wti0cNt6MLsiUeHqNl5H/4U/Q/fR+GA8pJ+L
npqZDG2tFi1WMvvGhGgIbScrjR4VO3TuKy+rZXMYvMRk1RXZ9gPhk6evFnviwHsE
jV5AEjxLz3duD+u/SjPp1vloxe2KuWnyC+EKInnka909sl4ZAUh+qIYfZK85DAjm
```

```
GJP4W036E9wTJQF2hZJrzsiB1MGyC1WI9veRISd30izZZL6VVXLXUtHwVHnVASrS
zZDVpzj+3yD5hRXsvFigGhY0FCVFnwIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUxC2l6pvJaRflgu3MUdN6zTuP6YcwgY4GA1UdIwSBhjCBg4AUxC2l
6pvJaRflgu3MUdN6zTuP6YehYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAKD+v6LeR/WrMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAIK+DtbUPppJXFqQMv1f2Gky5/82ZwgbbfXa
HBeGSii55b3tsyC3ZW5ZlMJ7Dtnr3vUkiWbV1EUaZGOUlndUFtXUMABCb/coDndw
CAr53XTv7UwGVNe/AFO/6pQDdPxXn3xBhF0mTKPrOGdvYmjZUtQMSVb9lbMWCFfs
w+SwDLnm5NF4yZchIcTs2fdpoyZpOHDXy0xgxO1gWhKTnYbaZOxkJvEvcckxVAwJ
obF8NyJla0/pWdjhlHafEXEN8lyxyTTyOa0BGTuYOBD2cTYYynauVKY4fqHUkr3v
Z6fboaHEd4RFamShM8uvSu6eEFD+qRmvqlcodbpsSOhuGNLzhOQ=
-----END CERTIFICATE-----
```

欧州 (アイルランド) - eu-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
```

```
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAOrmqHuaUt0vMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEwMjkw
OTA2MTlaGA8yMTk1MDQwMzA5MDYxOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAjE7nVu+aHLtzp9FYV25Qs1mvJ1JXD7J0iQ1Gs/RirW9a5ZECCtc4ssnf
zQHq2JRVr0GRchvDrbm1HaP/avtFQR/Thvfltwu9AROVT22dUOTvERdkNzveoFCy
hf52Rqf0DMrLXG8ZmQPPXPDFAv+sVMWCDftcChxRYZ6mP9O+TpgYNT1krD5PdvJU
7HcXrkNHDYqbsg8A+Mu2hzl0QkvUET83Csg1ibeK54HP9w+FsD6F5W+6ZSHGJ88l
FI+qYKs7xsjJQYgXWfEt6bbckWs1kZIaIOyMzYdPF6ClYzEec/UhIe/uJyUUNfpT
VIsI5OltBbcPF4c7Y20jOIwwI2SgOQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUF2DgPUZivKQR/Zl8mB/MxIkjZDUwgY4GA1UdIwSBhjCBg4AUF2Dg
PUZivKQR/Zl8mB/MxIkjZDWhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAOrmqHuaUt0vMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAGm6+57W5brzJ3+T8/XsIdLTuiBSe5ALgSqI
qnO5usUKAeQsa+kZIJPyEri5i8LEodh46DAF1RlXTMYgXXxl0YggX88XPmPtok17
l4hib/D9/lu4IaFIyLzYNSzsETYWKWoGVe7ZFz60MTRTwY2u8YgJ5dec7gQgPSGj
avB0vTIgoW41G58sfw5b+wjXCsh0nROon79RcQFFhGnvup0MZ+JbljyhZUYFzCli
31jPZiKzqWa87xh2DbAyvj2KZrZtTe2LQ48Z4G8wWytJzxEeZdREe4NoETf+Mu5G
4CqoaPR05KWkdNUdGNwXewydb3+agdCgfTs+uAjeXKNdSpbhMYg=
-----END CERTIFICATE-----
```

欧州 (ロンドン) - eu-west-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANBx0E2bOCEPMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNjA4MTEx
NDU2NDJaGA8yMTk2MDExNTE0NTY0MlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEArYS3mJLGaMrh2DmiPLbqr4Z+xWXTzBWCjOwpsuHE9H6dWUUyl2Bgnu+Z
d8QvW306Yleec45M4F2RA3J4hWHtShzsMlOJVRt+YulGeTf9OCPr26QmIFfs5nD4
fgsJQEry2MBSGA9Fxq3Cw6qkWcrOPsCR+bHOU0XykdKl0MnIbpBf0kTfciAupQEA
dEHnM2J1L2iI0NTLBgKxy5PXLH9weX20BFauNmHH9/J07OpwL20SN5f8TxcM9+pj
Lbk8h1V4KdIwVQpdWkbDL9BCGlYjyadQJxSxz1J343NzrnDM0M4h4HtVaKOS7bQo
Bqt2ruopLRCYgcuFHck/1348iAmbRQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBG
wujwU1Otpi3iBgmhjMClgZyMMn0aQIxMigoFNqXMUNx1Mq/e/Tx+SNaOEAu0n2FF
aiYjvY0/hXOx75ewzZvM7/zJWIdLdsgewpUqOBH4DXFhbSk2TxggSPb0WRqTBxq5
Ed7F7+7GRIeBbRzdLqmISDnfqey8ufW0ks51XcQNomDIRG5s9XZ5KHviDCar8FgL
HngBCdFI04CMagM+pwTO9XN1Ivt+NzUj208ca3oP1IwEAd5KhIhPLcihBQA5/Lpi
h1s3170z1JQ1HZbDrH1pgp+8hSI0DwwDVb3IIH8kPR/J0Qn+hvOl2HOpaUg2Ly0E
pt1RCZe+W7/dF4zsbqwK
-----END CERTIFICATE-----
```

欧州 (ミラノ) - eu-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAqwCCQCMElHPdwG37jAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTA0MjkyMDM1MjJaFw00
NTA0MjkyMDM1MjJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbYwggErBgcqhkjOOAQBMIIBHgKBgQDAkoL4YfdMI/MrQ0oL
NPfeEk94eiCQA5xNOnU7+2eVQtEqjFbDADFENh1p3sh9Q9OoheLFH8qpSfNDWn/0
ktCS909ApTY6Esx1ExjGSeQq/U+SC2JSuuTT4WFMKJ63a/czMtFkEPPnVIjJJJmT
HJSKSsVUgpdDIRvJXuyB0zdB+wIVALQ3OLaVGdlPMNfS1nD/Yyn+32wnAoGAPBQ3
7XHg5NLOS4326eFRUT+4ornQFjJjP6dp3pOBEzpImNmZTtkCNNUKE4Go9hv5T4lh
R0pODvWv0CBupMAZVBP9ObplXPCyEIZtuDqVa7ukPOUpQNgQhLLAqkigTyXVOSmt
ECBj9tu5WNP/x3iTZTHJ+g0rhIqpgh012UwJpKADgYQAAoGAV1OEQPYQUg5/M3xf
```

```
6vE7jKTxxyFWEyjKfJK7PZCzOIGrE/swgACy4PYQW+AwcUweSlK/Hx2OaZVUKzWo
wDUbeu65DcRdw2rSwCbBTU342sitFo/iGCV/Gjf+BaiAJtxniZze7J1ob8vOBeLv
uaMQmgOYeZ5e0fl04GtqPl+lhcQwCQYHKoZIzjgEAwMwADAtAhQdoeWLrkm0K49+
AeBK+j6m2h9SKQIVAIBNhS2a8cQVABDCQXVXrc0tOmO8
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICNjCCAZ+gAwIBAgIJAOZ3GEIaDcugMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTEwMjQx
NTE5MDlaGA8yMTk5MDMyOTE1MTkwOVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKB
gQCjiPgW3vsXRj4JoA16WQDyoPc/eh3QBARaApJEc4nPIGoUolpAXcjFhWplo2O+
ivgfCsc4AU9OpYdAPha3spLey/bhHPRi1JZHRNqScKP0hzsCNmKhfnZTIEQCFvsp
DRp4zr91/WS06/flJFBYJ6JHhp0KwM81XQG59lV6kkoW7QIDAQABMA0GCSqGSIb3
DQEBCwUAA4GBAGLLrY3P+HH6C57dYgtJkuGZGT2+rMkk2n81/abzTJvsqRqGRrWv
XRKRXlKdM/dfiuYGokDGxiC0Mg6TYy6wvsR2qRhtXW1OtZkiHWcQCnOttz+8vpew
wx8JGMvowtuKB1iMsbwyRqZkFYLcvH+Opfb/Aayi20/ChQLdI6M2R5VU
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAO/+DgYF78KwMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTA0Mjky
MDM1MjJaGA8yMTk4MTAwMjIwMzUyMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAv1ZLV+Z/P6INq+R1qLkzETBg7sFGKPiwHekbpuB6lrRxKHhj8V9vaReM
lnv1Ur5LAPpMPYDsuJ4WoUbPYAqVqyMAo7ikJHCCM1cXgZJefgN6z9bpS+uA3YVh
V/0ipHh/X2hc2S9wvxKWiSHu6Aq9GVpqL035tJQD+NJuqFd+nXrtcw4yGtmvA6wl
5Bjn8WdsP3xOTKjrByYY1BhXpP/f1ohU9jE9dstsRXLa+XTgTPWcWdCS2oRTWPGR
c5Aeh47nnDsyQfP9gLxHeYeQItV/BD9kU/2Hn6mnRg/B9/TYH8qzlRTzLapXp4/5
iNwusrTNexGl8BgvAPrfhjDpdgYuTwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQB7
5ya11K/hKgvaRTvZwVV8GlVZt0CGPtNvOi4AR/UN6TMm51BzUB5nurB4z0R2MoYO
Uts9sLGvSFALJ4otoB77hyNpH3drttU1CVVwal/yK/RQLSon/IoUkaGEbqalu+mH
nYad5IG4tEbmepX456XXcO58MKmnczNbPyw3FRzUZQtI/sf94qBwJ1Xo6XbzPKMy
xjL57LHIZCssD+XPifXay69OFlsCIgLim11HgPkRIHEOXLSf3dsW9r+4CjoZqB/Z
jj/P4TLCxbYCLkvglwaMjgEWF40Img0fhx7yT2X92MiSrs3oncv/IqfdVTiN8OXq
```

```
jgnq1bf+EZEZKvb6UCQV
-----END CERTIFICATE-----
```

欧州 (パリ) - eu-west-3

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
```

```
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALWSfgHuT/ARMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNzA1MzEx
MTE4MTZaGA8yMTk2MTEwMzExMTgxNlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAy5V7KDqnEvF3DrSProFcgu/oL+QYD62b1U+Naq8aPuljJe127Sm9WnWA
EBdOSASkOaQ9fzjCPoG5SGgWKxYoZjsevHpmzjVv9+Ci+F57bSuMbjgUbvbRIFUB
bxQojVoXQPHgK5v433ODxkQ4sjRyUbf4YV1AFdfU7zabC698YgPVOExGhXPlTvco
8mlc631ubw2g52j0lzaozUkHPSbknTomhQIvO6kUfX0e0TDMH4jLDG2ZIrUB1L4r
OWKG4KetduFrRZyDHF6ILZu+s6ywiMicUd+2UllDFC6oas+a8D11hmO/rpWU/ieV
jj4rWAFrsebpn+Nhgy96iiVUGS2LuQIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDE
iYv6FQ6knXCg+svlcaQG9q59xUC5z8HvJZ1+SxzPKKC4PKQdKvIIfE8GxVXqlZG1
cl5WKTFDMapnzb9RV/DTaVzWx3cMYT77vm1Hl1XGjhx611CGcENH1egI3lOTILsa
+KfopuJEQQ9TDMAIkGjhA+KieU/U5Ctv9fdej6d0GC6OEuwKkTNzPWue6UMq8d4H
2xqJboWsE1t4nybEosvZfQJcZ8jyIYcYBnsG13vCLM+ixjuU5MVVQNMY/gBJzqJB
V+U0QiGiuT5cYgY/QihxdHt99zwGaE0ZBC7213NKrlNuLSrqhDI2NLu8NsExqOFy
OmY0v/xVmQUQl26jJXaM
-----END CERTIFICATE-----
```

欧州 (スペイン) - eu-south-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC8DCCAq
+gAwIBAgIGAXjwLk46MAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAGG2m8EKmaf5qQqj3Z
```

```
+rzSaTaXE3B/R/4A2VuGqRYR7MljPtwdmU6/3CPjCACcZmTIcOAKbFiDHqadQgBZXfzGpzw8Zo
+eYmmk5fXycgnj57PYH1dIWU6I7mCbAah5MZMcmHaTmIsomGrhcnWB8d8qOU7oZ0UWK4lbiAQs1MihoUwCQYHKoZIzjg
WmbaU7YM5GwCFCvIJOesO5hZ8PHC52dAR8WWC6oe
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXjwLkiaMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
VvR1+45Aey5zn3vPk6xBm5o9grSDL6D2iAuprQnfVXn8CIbSDbWFhA3fi5ippjKkh3sl8VyCvCOUXKdOaNrYBrPRkrdH
+3m/
rxIUZ2IK1fDlC6sWAjddf6sBrV2w2a78H0H8EwuwiSgttURBjwJ7KPPJCqaqrQIDAQABMA0GCSqGSIb3DQEBBQUAA4GB
+FzqQDzun/
iMMzcFucmLMl5BxEblrFXOz7IIuOeiGkndmrqUeDCykztLku45s7hxdNy4ltTuVAaE5aNBdw5J8U1mRvsKvHLy2ThH6h
+hBgiphYp84DUbWVYeP8YqLEJSqscKscWC
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALWSmO6DvSpQMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTgx
MzU4NDNaGA8yMjAxMTIyMjEzNTg0M1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAuAAhuSpsHC00/fD2zNlBDpNLRndi9qbHsNeuz3WqN7Samj2aSrM2hS+i
hUxx0BspZj0tZCOsbpPZ+i74NOEQtFeqQoEGvKhB1nJiF4y5I81HDhs5qHvoIivm
7rbbik3zgm1PqS/DmDjVQaXPcD31Rd9ILwBmWEwJqHigyNVlxYtCzTQcrlBrvNZM
dnNgCDAdX/HBEFxx9O12xeu0bSt0s+PJWZ1RTbYrNe7LIH6ntUqHxP/ziQ5trXEZ
uqy7aWk1L8uK4jmyNph0lbaqBa3Y6pYmU1nC27UE4i3fnPB0LSiAr+SrwVvX1g4z
ilo8kr+tbIF+JmcgYLBvO8Jwp+EUqQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUwvGzKJL9A5LReJ4Fxo5K6I20xcowgY4GA1UdIwSBhjCBg4AUwvGz
KJL9A5LReJ4Fxo5K6I20xcqhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALWSmO6DvSpQMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAJAZd31jyoTGLawAD2+v/vQsaB9vZIx5EImi
G8YGkd61uFWeNhAmtrwyE/i6FDSIphDrMHBkvw/D3BsqK+Ev/JOK/VYuaYDx/8fp
H4cwp9jC57CXzdIDREWNf6M9PsHFg2WA9XNNtClOZL5WJiJwel8eDSg+sqJUxE0l
MW+QChq/20F6niyaRK4bXrZq14as7h+F9u3A9xHEOVP7Zk9C2ehrBXzCMLSDt3GV
fEuMea2RxMhozwz34Hkdb6j18qoCfygubulovRNQjKw/cEmgPR16KfZPP5caILVt
9qkYPvePmbiVswZDee73cDymJYxLqILpOZwyXvUH8StiH42FHZQ=
-----END CERTIFICATE-----
```

欧州 (ストックホルム) - eu-north-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALc/uRxg++EnMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xODA0MTAx
NDAwMTFaGA8yMTk3MDkxMzE0MDAxMVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAzwCGJEJIxqtr2PD2a1mA6LhRzKhTBa1AZsg3eYfpETXIVlrpojMfvVoN
qHvGshWLgrGTT6os/3gsaADheSaJKavxwX3X6tJA8fvEGqr3a1C1MffH9hBWbQqC
LbfUTAbkwis4GdTUwOwPjT1Cm3u9R/VzilCNwkj7iQ65AFAI8Enmsw3UGldEsop4
yChKB3KW3WI0FTh0+gD0YtjrqqYJxpGOYBpJp5vwdd3fZ4t1vidmDMs7liv4f9Bx
p0oSmUobU4GUlFhBchK1DukICVQdnOVzdMonYm7s+HtpFbVHR8yf6QoixBKGdSal
mBf7+y0ixjCn0pnC0VLVooGo4mi17QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQDG
4ONZiixgk2sjJctwbyD5WKLTH6+mxYcDw+3y/F0fWz561YORhP2FNnPOmEkf0Sl/
Jqk4svzJbCbQeMzRoyaya/46d7UioXMHRZam5IaGBhOdQbi97R4VsQjwQj0RmQsq
yDueDyuKTwWLK9KnvI+ZA6e6bRkdNGflK4N8GGKQ+fBhPwVELkbT9f16OJkezeeN
S+F/gDADGJgmPXfjogICb4Kvshq0H5Lm/xZlDULF2g/cYhyNY6EOI/eS5m1I7R8p
D/m6WoyZdpInxJfxW616OMkxQMRVsruLTNGtby3u1g6ScjmpFtvAMhYejBSdzKG4
FEyxIdEjoeO1jhTsck3R
-----END CERTIFICATE-----
```

欧州 (チューリッヒ) - eu-central-2

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAXjXiKJnMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAYNjaCNg/
cfgQOOl1BUj5ClUulqwZ9Q+SfDzPZh9D2C0VbiRANiZoxrV8RdgmzzC5T7VcriVwjwvta2Ch//
b+sZ86E5h0XWWr+BeEjD9cu3eDj12XB5sWEbNHNx49p5Tmtu5r2LDtlL8X/
Rpfalu2Z2OJgjFJWGf7hRwxe456n
+lowCQYHKoZIzjgEAwMvADAsAhRChsLcj4U5CVb2cp5M0RE1XbXmhAIUeGSnH+aiUQIWmPEFja+itWDufIk=
```

```
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXjSGFGiMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
opKZAUusJx2hpgU3pUHhlp9ATh/VeVD582jTd9IY
+8t5MDa6Z3fGliByEiXz0LEHdi8MBacLREu1TwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBAILlpoE3k9o7KdALAxsFJNit
+g3RMzdbiFM+7MA63Nv5fsf+0xgcjSNBElvPCDKFvTJl4QQhToy056llO5GvdS9RK
+H8xrP2mrqngApoKTApv93vHBixgFSn5KrczRO0YSm3OjkqbydU7DFlmkXXR7GYE+5jbHvQHYiT1J5sMu
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJALvT012pxTxNMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTgx
NTEyMDdaGA8yMjAxMTIyMjE1MTIwN1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAyn+Lsnq1ykrfYlZkk6aAAYNReNd9Iw8AUwCBkgOr2eBiBBepYxHwU85N
++moQ+j0EV2VaahBeTLShGZZS1HsyK8+cYT2QzpghIoamcYhrPXyIxlWiRQlaqSg
OFiE9bsqL3rCF5Vz+tOiTe5W/7ojfOFls6++g7ZpobwJlpMbuJepqyeHMPyjvO5A
age81lJewc4bxo2ntaW0HCqNksqfYB78j6X6kn3PFpX7FaYAwZA+Xx6C7UCY7rNi
UdQzfAo8htfJi4chz7frpUdQ9kl3IOQrsLshBB5fFUjl09NiFipCGBwi+8ZMeSn1
5qwBI01BWXPfG7WX6Owyjhmh6JtE1wIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQU8HN4vvJrsZgPQeksMBgJb9xR1yYwgY4GA1UdIwSBhjCBg4AU8HN4
vvJrsZgPQeksMBgJb9xR1yahYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJALvT012pxTxNMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAGlHYDtcHpfBvdHx9HeQE8HgNugJUPdEqxun
t9U33p8VFrs+uLPtrOd9HDJEGvvs5h84EUie/oGJxRt7V1Vlid1PvHf6cRmpjgqY
YdggAVkZtY/PnFVmzf2bMVlSQPrqCl7U0zaw2Kvnj4zgX0rZyCetgrRZSUSxotyp
978Wy9ccXwVSeYG/YAr5rJpS6ZH7eRQvUY0IzwFNeaOPgOTEVpcjWlV6+MQEvsEx
W85q+s6AVr49eppEx8SLJsl0C23yB+L+t32tAveQImRWtJMpzZ5cxh/sYgDVeoC0
85H1NK/7H9fAzT1cPu1oHSnB0xYzzHGOAmXmusMfwUk8fL1RQkE=
-----END CERTIFICATE-----
```

イスラエル (テルアビブ) - il-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq+gAwIBAgIGAX0QPi
+9MAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcMB1NlYX
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/
hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAbazCL5XXyPmcw3+oMYQUF5/9YogW6D0FZbYuyPgjOoUwWdl6fjlzWca
pq+l1ezuK2DFOzNTEyPEwwCQYHKoZIzjgEAwMvADAsAhRt1jkpXsvrS
+xTo2M9h2s2uLAhEQIUOZ2FcnTSrshF2EIdixZZwtNv66Q=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAX0QQGVLMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
+S8vOy5hpLoRe4RkOrY0cM3bN07GdEMlin5mU0y1t8y3ct4YewvmkgT42kTyMM
+t1K4S0xsqjXxxS716uGYh7eWtkxrCihj8AbXN/6pa095h
+7TZyl2n83keiNUzM2KoqQVMwIDAQABMA0GCSqGSIb3DQEBBQUAA4GBADwA6VVEIIZD2YL00F12po40xDLzIc9XvqFPS
FmU7H8s62/jD6cOR1A1cClIyZUe1yT1ZbPySCs43J+Thr8i8FSRxzDBSZZi5foW
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAOVp1h2I9wW7MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA3MTUx
MjQ0MTJaGA8yMjAxMTIxOTEyNDQxMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEA13PkyWvl61iV/SYfO1UFO76UpDfPm2SF/Rz/o33cm699X++EYPxTnoEc
vmWeS0I7eDXc4OCUiToG0sEx0klE0CX1Z1tK6qJ+zgWQLZ9SZEC9H0NsSA6LhrHu
Nq0dzeK3LjhdfcX46/4GqdiptpdTuM4m/hOQ5yx4JMQ/n1sdpv4M5VLRWwWW9Lem
ufb79Id7O9SispxgRsz1KXIjp7N9S4BY7itSXz97uSyzTqEjWZ6mDUhTu3t21GKC
6flALGTTrG2yghEhz53rkvLsvwzjPSSlT6LIf0mrRPzHaf+EdaKoasElE1SHh+ZH
9mI81HywpE+HZ+W+5hBCvjYp90Y1fwIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
```

```
BgNVHQ4EFgQU58tN2JO+yEGq5JbIXxGi4vRVPyIwgY4GA1UdIwSBhjCBg4AU58tN
2JO+yEGq5JbIXxGi4vRVPyKhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAOVp1h2I9wW7MBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBANBN0e1EqNy4+IU2yQzMJ+Wy5ZIOtTP6GSBR
7muVY1bDeAwtNTEOpwgrZVlC7/xq5Q0LC1y0Z70hHXEf8au7qStaAoUtxzvhTAZI
NC0lwoFU56UFw4N0vZII17iqEfoqRC4PpI3OxqEJHFy0VLlvAzJoKB4QLLqDAYVA
LXCiOLoVT+y9tRYsxw5MyO0Bi6fxQIIADl2bE9xkunTN1Jkkwqo3LxNy/ryz4QWR
8K7jHUItifv4h/hxBKpHEquN8CkdvM9oeGl7I8PFrSFEpGr1euDXY0euZzzYiDBV
m6GpTJgzpVsEuIX52dPcPemwQncoIfZyhWDW85MJUnby2WTEcFo=
-----END CERTIFICATE-----
```

中東 (バーレーン) – me-south-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7jCCAq4CCQCVWIgSmP8RhTAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xOTAyMDUxMzA2MjFaFw00
NTAyMDUxMzA2MjFaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbgwggEsBgcqhkjOOAQBMIIBHwKBgQDcwojQfgWdV1QliO0B
8n6cLZ38VE7ZmrjZ9OQV//Gst6S1h7euhC23YppKXi1zovefSDwFU54zi3/oJ++q
PHlP1WGL8IZ34BUgRTtG4TVolvp0smjkMvyRu5hIdKtzjV93Ccx15gVgyk+o1IEG
fZ2Kbw/Dd8JfoPS7KaSCmJKxXQIVAIZbIaDFRGa2qcMkW2HWASyNDl7bAoGBANtz
IdhfMq+l2I5iofY2oj3HI21Kj3LtZrWEg3W+/4rVhL3lTm0Nne1rl9yGujrjQwy5
Zp9V4A/w9w2O10Lx4K6hj34Eefy/aQnZwNdNhv/FQP7Az0fju+Yl6L13OOHQrL0z
Q+9cF7zEosekEnBQx3v6psNknKgD3Shgx+GO/LpCA4GFAAKBgQCVS7m77nuNAlZ8
wvUqcooxXMPkxJFl54NxAsAul9KP9KN4svm0O3Zrb7t2FOtXRM8zU3TqMpryq1o5
mpMPsZDg6RXo9BF7Hn0DoZ6PJTamkFA6md+NyTJWJKvXC7iJ8fGDBJqTciUHuCKr
12AztQ8bFWsrTgTzPE3p6U5ckcgV1TAJBgcqhkjOOAQDAy8AMCwCFB2NZGWm5EDl
86ayV3c1PEDukgQIAhQow38rQkN/VwHVeSW9DqEshXHjuQ==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDPDCCAqWgAwIBAgIJAMl6uIV/zqJFMA0GCSqGSIb3DQEBCwUAMHIxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIDApXYXNoaW5ndG9uMRAwDgYDVQQHDAdTZWF0dGxlMSAw
HgYDVQQKDBdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzEaMBgGA1UEAwwRZWMyLmFt
YXpvbmF3cy5jb20wIBcNMTkwNDI2MTQzMjQ3WhgPMjE5ODA5MjkxNDMyNDdaMHIx
```

```
CzAJBgNVBAYTAlVTMRMwEQYDVQQIDApXYXNoaW5ndG9uMRAwDgYDVQQHDAdTZWF0
dGxlMSAwHgYDVQQKDBdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzEaMBgGA1UEAwwR
ZWMyLmFtYXpvbmF3cy5jb20wgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALVN
CDTZEnIeoX1SEYqq6k1BV0ZlpY5y3KnoOreCAE589TwS4MX5+8Fzd6AmACmugeBP
Qk7Hm6b2+g/d4tWycyxLaQlcq81DB1GmXehRkZRgGeRge1ePWd1TUA0I8P/QBT7S
gUePm/kANSFU+P7s7u1NNl+vynyi0wUUrw7/wIZTAgMBAAGjgdcwgdQwHQYDVR0O
BBYEFILtMd+T4YgH1cgc+hVsVOV+480FMIGkBgNVHSMEgZwwgZmAFILtMd+T4YgH
1cgc+hVsVOV+480FoXakdDByMQswCQYDVQQGEwJVUzETMBEGA1UECAwKV2FzaGlu
Z3RvbjEQMA4GA1UEBwwHU2VhdHRsZTEgMB4GA1UECgwXQW1hem9uIFdlYiBTZXJ2
aWNlcyBMTEMxGjAYBgNVBAMMEWVjMi5hbWF6b25hd3MuY29tggkAyXq4hX/OokUw
DAYDVR0TBAUwAwEB/zANBgkqhkiG9w0BAQsFAAOBgQBhkNTBIFgWFd+ZhC/LhRUY
4OjEiykmbEp6hlzQ79T0Tfbn5A4NYDI2icBP0+hmf6qSnIhwJF6typyd1yPK5Fqt
NTpxxcXmUKquX+pHmIkK1LKDO8rNE84jqxrxRsfDi6by82fjVYf2pgjJW8R1FAw+
mL5WQRFexbfB5aXhcMo0AA==
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANZkFlQR2rKqMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xOTAyMDUx
MzA2MjBaGA8yMTk4MDcxMTEzMDYyMFowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAy4Vnit2eBpEjKgOKBmyupJzJAiT4fr74tuGJNwwa+Is2vH12jMZn9Il1
UpvvEUYTIboIgISpf6SJ5LmV5rCv4jT4a1Wm0kjfNbiIlkUi8SxZrPypcw24m6ke
BVuxQZrZDs+xDUYIZifTmdgD50u5YE+TLg+YmXKnVgxBU6WZjbuK2INohi71aPBw
2zWUR7Gr/ggIpf635JLU3KIBLNEmrkXCVSnDFlsK4eeCrB7+UNak+4BwgpuykSGG
Op9+2vsuNqFeU1l9daQeG9roHR+4rIWSPa0opmMxv5nctgypOrE6zKXx2dNXQldd
VULv+WH7s6Vm4+yBeG8ctPYH5GOo+QIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBs
ZcViiZdFdpcXESZP/KmZNDxB/kktlIEIhsQ+MNn29jayE5oLmtGjHj5dtA3XNKlr
f6PVygVTKbtQLQqunRT83e8+7iCZMKI5ev7pITUQVvTUwI+Fc01JkYZxRFlVBuFA
WGZO+98kxCS4n6tTwVt+nSuJr9BJRVC17apfHBgSS8c5OWna0VU/Cc9ka4eAfQR4
7pYSDU3wSRE01cs30q34lXZ629IyFirSJ5TTOIc0osNL7vwMQYj8HOn4OBYqxKy8
ZJyvfXsIPh0Na76PaBIs6ZlqAOflLrjGzxBPiwRM/XrGmF8ze4KzoUqJEnK13O6A
KHKgfiigQZ1+gv5FlyXH
-----END CERTIFICATE-----
```

中東 (UAE) - me-central-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7zCCAq
+gAwIBAgIGAXjXhqnnMAkGByqGSM44BAMwXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgMEFdhc2hpbmd0b24gU3RhdGUxED
U4EddRIpUt9KnC7s5Of2EbdSPO9EAMMeP4C2USZpRV1AIlH7WT2NWPq/
xfW6MPbLm1Vs14E7gB00b/JmYLdrmVClpJ+f6AR7ECLCT7up1/63xhv4O1fnxqimFQ8E
+4P208UewwI1VBNaFpEy9nXzrith1yrv8iIDGZ3RSAHHAhUAl2BQjxUjC8yykrmCouuEC/
BYHPUCgYEA9+GghdabPd7LvKtcNrhXuXmUr7v6OuqC+VdMCz0HgmdRWVeOutRZT
+ZxBxCBgLRJFnEj6EwoFhO3zwkyjMim4TwWeotUfI0o4KOuHiuzpnWRbqN/C/ohNWLx
+2J6ASQ7zKTxvqhRkImog9/hWuWfBpKLZl6Ae1UlZAFMO/7PSSoDgYQAAoGAW+csuHsWp/7/
pv8CTKFwxsYudxuR6rbWaHCykIeAydXL9AWnphK6yp1ODEMBFl68Xq8Hp23sOWyf8moOhqCom9+0+ovuUFdpvCie86bp
TOZU568Ty1ff3dDWbdRzeNQRHodRG+XEQSizMkAreeWt4kBa+PUwCQYHKoZIzjgEAwMvADAsAhQD3Z
+XGmzKmgaLgGcVX/Qf1+Tn4QIUH1cgksBSVKbWj81tovBMJeKgdYo=
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIICMzCCAZygAwIBAgIGAXjRrnDjMA0GCSqGSIb3DQEBBQUAMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIDBBXYXNoaW5n
KyA6zyruJQrYy00a6wqLA7eeUzk3bMiTkLsTeDQfrkaZMfBAjGaaOymRo1C3qzE4rIenmahvUplu9ZmLwL1idWXMRX2R
+d2SeoKOKQWoc2UOFZMHYxDue7zkyk1CIRaBukTeY13/
RIrlc6X61zJ5BBtZXlHwayjQIDAQABMA0GCSqGSIb3DQEBBQUAA4GBABTqTy3R6RXKPW45FA+cgo7YZEj/
Cnz5YaoUivRRdX2A83BHuBTvJE2+WX00FTEj4hRVjameE1nENoO8Z7fUVloAFDlDo69fhkJeSvn51D1WRrPnoWGgEfr1
B+Wqm3kVEz/QNcz6npmA6
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAM4h7b1CVhqqMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0yMjA0MTEx
MDE1MDNaGA8yMjAxMDkxNTEwMTUwM1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEApybTWfMOhSoMpqPo72eqAmnn1dXGZM+G8EoZXzwHwT/+IHEXNB4q5N6k
tudYLre1bJxuzEw+iProSHjmb9bB9YscRTofjVhBlt35Fc+i8BaMeH94SR/eE8QO
mll8gnLNW3d62lyuhzuyv1e5wVlRqzYw+X2zRH4/wRDOCOpzjKoHIgyPKsMgwsw5
aTZhNMsGxZN9dbkf0iCGeQLDytwU/JTh/HqvSr3VfUOapTJJiyAxoCtZWgp1/7wC
RvOCSMRJobpUqxZgl/VsttwNkikSFz1wGkcYeSQvk+odbnYQckA8tdddoVI56eD4
```

```
qtREQvfpMAX5v7fcqLexl5d5vH8uZQIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUOadrbTs+OhzwoAgUJ7RqQNdwufkwgY4GA1UdIwSBhjCBg4AUOadr
bTs+OhzwoAgUJ7RqQNdwufmhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAM4h7b1CVhqqMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBAICTdAOGEOnII8HaGCpCB8us/hGFaLptJaAf
D5SJAyVy66/mdfjGzE1BKkKxnbxemEVUIzbRidOnyilB+pKwN3edAjTZtWdpVA0V
R/G/qQPmcVljtycBz4VC6SuOUYflGzLH1GZ6GJWbuDtFzw8r7HGdRN1wrEPe3UF2
sMpuVezqnRUdvVRoVQP4jFgNsE7kNvtn2NiPhb/CtrxpcwIQ7r6YeoHcBSheuV1Z
xZDHynC3KUprQGx1+Z9QqPrDf180MaoqAlTl4+W6Pr2NJYrVUFGS/ivYshMg574l
CPU6r4wWZSKwEUXq4BInYX6z6iclp/p/J5QnJp2mAwyi6M+I13Y=
-----END CERTIFICATE-----
```

南米 (サンパウロ) - sa-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDIjCCAougAwIBAgIJAKnL4UEDMN/FMA0GCSqGSIb3DQEBBQUAMGoxCzAJBgNV
BAYTAlVTMRMwEQYDVQQIEwpXYXNoaW5ndG9uMRAwDgYDVQQHEwdTZWF0dGxlMRgw
FgYDVQQKEw9BbWF6b24uY29tIEluYy4xGjAYBgNVBAMTEWVjMi5hbWF6b25hd3Mu
```

```
Y29tMB4XDTE0MDYwNTE0MjgwMloXDTI0MDYwNTE0MjgwMlowajELMAkGA1UEBhMC
VVMxEzARBgNVBAgTCldhc2hpbmd0b24xEDAOBgNVBAcTB1NlYXR0bGUxGDAWBgNV
BAoTD0FtYXpvbi5jb20gSW5jLjEaMBgGA1UEAxMRZWMyLmFtYXpvbmF3cy5jb20w
gZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBAIe9GN//SRK2knbjySG0ho3yqQM3
e2TDhWO8D2e8+XZqck754gFSo99AbT2RmXClambI7xsYHZFapbELC4H91ycihvrD
jbST1ZjkLQgga0NE1q43eS68ZeTDccScXQSNivSlzJZS8HJZjgqzBlXjZftjtdJL
XeE4hwvo0sD4f3j9AgMBAAGjgc8wgcwwHQYDVR0OBBYEFCXWzAgVyrbwnFncFFIs
77VBdlE4MIGcBgNVHSMEgZQwgZGAFCXWzAgVyrbwnFncFFIs77VBdlE4oW6kbDBq
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKV2FzaGluZ3RvbjEQMA4GA1UEBxMHU2Vh
dHRsZTEYMBYGA1UEChMPQW1hem9uLmNvbSBJbmMuMRowGAYDVQQDExFlYzIuYW1h
em9uYXdzLmNvbYIJAKnL4UEDMN/FMAwGA1UdEwQFMAMBAf8wDQYJKoZIhvcNAQEF
BQADgYEAFYcz1OgEhQBXIwIdsgCOS8vEtiJYF+j9uO6jz7VOmJqO+pRlAbRlvY8T
C1haGgSI/A1uZUKs/Zfnph0oEI0/hu1IIJ/SKBDtN5lvmZ/IzbOPIJWirlsllQIQ
7zvWbGd9c9+Rm3p04oTvhup99la7kZqevJK0QRdD/6NpCKsqP/0=
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIEEjCCAvqgAwIBAgIJAMcyoxx4U0xxMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQw
ODU4MDJaGA8yMTk1MDExNzA4NTgwMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAw45IhGZVbQcy1fHBqzROhO8CsrDzxj/WP4cRbJo/2DAnimVrCCDs5O86
FA39Zo1xsDuJHDlwMKqeXYXkJXHYbcPWc6EYYAnR+PlLG+aNSOGUzsy202S03hT0
B20hWPCqpPp39itIRhG4id6nbNRJOzLm6evHuepMAHR4/OV7hyGOiGaV/v9zqiNA
pMCLhbh2xk0PO35HCVBuWt3HUjsgeks2eEsu9Ws6H3JXTCfiqp0TjyRWapM29OhA
cRJfJ/d/+wBTz1fkWOZ7TF+EWRIN5ITEadlDTPnF1r8kBRuDcS/lIGFwrOOHLo4C
cKoNgXkhTqDDBDu6oNBb2rS0K+sz3QIDAQABo4HUMIHRMAsGA1UdDwQEAwIHgDAd
BgNVHQ4EFgQUqBy7D847Ya/w321Dfr+rBJGsGTwwgY4GA1UdIwSBhjCBg4AUqBy7
D847Ya/w321Dfr+rBJGsGTyhYKReMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQ4IJAMcyoxx4U0xxMBIGA1UdEwEB/wQIMAYBAf8C
AQAwDQYJKoZIhvcNAQELBQADggEBACOoWSBf7b9AlcNrl4lr3QWWSc7k90/tUZal
PlT0G3Obl2x9T/ZiBsQpbUvs0lfotG0XqGVVHcIxF38EbVwbw9KJGXbGSCJSEJkW
vGCtc/jYMHXfhx67Szmftm/MTYNvnzsyQQ3v8y3Rdah+xe1NPdpFrwmfL6xe3pFF
cY33KdHA/3PNLdn9CaEsHmcmj3ctaaXLFIzZhQyyjtsrgGfTLvXeXRokktvsLDS/
YgKedQ+jFjzVJqgr4NjfY/Wt7/8kbbdhzaqlB5pCPjLLzv0zp/XmO6k+JvOePOGh
JzGk5t1QrSju+MqNPFk3+1O7o910Vrhqw1QRB0gr1ExrviLbyfU=
-----END CERTIFICATE-----
```

中国 (北京) - cn-north-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIDNjCCAh4CCQD3yZ1w1AVkTzANBgkqhkiG9w0BAQsFADBcMQswCQYDVQQGEwJV
UzEZMBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEg
MB4GA1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEMwIBcNMTUwNTEzMDk1OTE1
WhgPMjE5NDEwMTYwOTU5MTVaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNo
aW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24g
V2ViIFNlcnZpY2VzIExMQzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AMWk9vyppSmDU3AxZ2Cy2bvKeK3F1UqNpMuyeriizi+NTsZ8tQqtNloaQcqhto/l
gsw9+QSnEJeYWnmivJWOBdn9CyDpN7cpHVmeGgNJL2fvImWyWe2f2Kq/BL9l7N7C
P2ZT52/sH9orlck1n2zO8xPi7MItgPHQwu3OxsGQsAdWucdxjHGtdchulpo1uJ31
jsTAPKZ3p1/sxPXBBAgBMatPHhRBqhwHO/Twm4J3GmTLWN7oVDds4W3bPKQfnw3r
vtBj/SM4/IgQ3xJslFcl90TZbQbgxIi88R/gWTbs7GsyT2PzstU30yLdJhKfdZKz
/aIzraHvoDTWFaOdy0+OOaECAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAdSzN2+0E
V1BfR3DPWJHWRf1b7zl+1X/ZseW2hYE5r6YxrLv+1VPf/L5I6kB7GEtqhZUqteY7
zAceoLrVu/7OynRyfQetJVGichaaxLNM3lcr6kcxOowb+WQQ84cwrB3keykH4gRX
KHB2rlWSxta+2panSEO1JX2q5jhcFP90rDOtZjlpYv57N/Z9iQ+dvQPJnChdq3BK
5pZlnIDnVVxqRike7BFy8tKyPj7HzoPEF5mh9Kfnn1YoSVu+61lMVv/qRjnyKfS9
c96nE98sYFj0ZVBzXw8Sq4Gh8FiVmFHbQp1peGC19idOUqxPxWsasWxQXO0azYsP
9RyWLHKxH1dMuA==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJALSOMbOoU2svMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yMzA3MDQw
ODM1MzlaFw0yODA3MDIwODM1MzlaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
uhhUNlqAZdcWWB/OSDVDGk3OA99EFzOn/mJlmciQ/Xwu2dFJWmSCqEAE6gjufCjQ
q3voxAhC2CF+elKtJW/C0Sz/LYo60PUqd6iXF4h+upB9HkOOGuWHXsHBTsvgkgGA
1CGgel4U0Cdq+23eANr8N8m28UzljjSnTlrYCHtzN4sCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFBkZu3wT27NnYgrfH+xJz4HJaNJoMIGOBgNVHSME
gYYwgYOAFBkZu3wT27NnYgrfH+xJz4HJaNJooWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQC0jjGzqFNrLzASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBAECji43p+oPkYqmzll7e8Hgb
oADS0ph+YUz5P/bUCm61wFjlxaTfwKcuTR3ytj7bFLoW5Bm7Sa+TCl3lOGb2taon
```

```
2h+9NirRK6JYk87LMNvbS40HGPFumJL2NzEsGUeK+MRiWu+Oh5/lJGii3qw4YByx
SUDlRyNy1jJFstEZjOhs
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAOtrM5XLDSjCMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA4MTQx
MDAxNDJaGA8yMTk1MDExNzEwMDE0MlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAvVBz+WQNdPiM9S+aUULOQEriTmNDUrjLWLr7SfaOJScBzis5D5ju0jh1
+qJdkbuGKtFX5OTWTm8pWhInX+hIOoS3exC4BaANoa1A3o6quoG+Rsv72qQf8LLH
sgEi6+LMlCN9TwnRKOToEabmDKorss4zFl7VSsbQJwcBSfOcIwbdRRaW9Ab6uJHu
79L+mBR3Ea+G7vSDrVIA8goAPkae6jY9WGw9KxsOrcvNdQoEkqRVtHo4bs9fMRHU
Etphj2gh4ObXlFN92VtvzD6QBs3CcoFWgyWGvzg+dNG5VCbsiiuRdmii3kcijZ3H
Nv1wCcZoEAqH72etVhsuvNRC/xAP8wIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQA8
ezx5LRjzUU9EYWYhyYIEShFlP1qDHs7F4L46/5lc4pL8FPoQm5CZuAF31DJhYi/b
fcV7i3n++/ymQbCLC6kAg8DUB7NrcROll5ag8d/JXGzcTCnlDXLXx1905fPNa+jI
0q5quTmdmiSi0taeaKZmyUdhrB+a7ohWdSdlokEIOtbH1P+g5yll3bI2leYE6Tm8
LKbyfK/532xJPqO9abx4Ddn89ZEC6vvWVNDgTsxERg992Wi+/xoSw3XxkgAryIv1
zQ4dQ6irFmXwCWJqc6kHg/M5W+z60S/94+wGTXmp+19U6Rkq5jVMLh16XJXrXwHe
4KcgIS/aQGVgjM6wivVA
-----END CERTIFICATE-----
```

中国 (寧夏) – cn-northwest-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIDNjCCAh4CCQD3yZ1w1AVkTzANBgkqhkiG9w0BAQsFADBcMQswCQYDVQQGEwJV
UzEZMBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEg
MB4GA1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEMwIBcNMTUwNTEzMDk1OTE1
WhgPMjE5NDEwMTYwOTU5MTVaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNo
aW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24g
V2ViIFNlcnZpY2VzIExMQzCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEB
AMWk9vyppSmDU3AxZ2Cy2bvKeK3F1UqNpMuyeriizi+NTsZ8tQqtNloaQcqhto/l
gsw9+QSnEJeYWnmivJWOBdn9CyDpN7cpHVmeGgNJL2fvImWyWe2f2Kq/BL9l7N7C
P2ZT52/sH9orlck1n2zO8xPi7MItgPHQwu3OxsGQsAdWucdxjHGtdchulpo1uJ31
jsTAPKZ3p1/sxPXBBAgBMatPHhRBqhwHO/Twm4J3GmTLWN7oVDds4W3bPKQfnw3r
```

```
vtBj/SM4/IgQ3xJslFcl90TZbQbgxIi88R/gWTbs7GsyT2PzstU30yLdJhKfdZKz
/aIzraHvoDTWFaOdy0+OOaECAwEAATANBgkqhkiG9w0BAQsFAAOCAQEAdSzN2+0E
V1BfR3DPWJHWRf1b7zl+1X/ZseW2hYE5r6YxrLv+1VPf/L5I6kB7GEtqhZUqteY7
zAceoLrVu/7OynRyfQetJVGichaaxLNM3lcr6kcxOowb+WQQ84cwrB3keykH4gRX
KHB2rlWSxta+2panSEO1JX2q5jhcFP90rDOtZjlpYv57N/Z9iQ+dvQPJnChdq3BK
5pZlnIDnVVxqRike7BFy8tKyPj7HzoPEF5mh9Kfnn1YoSVu+61lMVv/qRjnyKfS9
c96nE98sYFj0ZVBzXw8Sq4Gh8FiVmFHbQp1peGC19idOUqxPxWsasWxQXO0azYsP
9RyWLHKxH1dMuA==
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJALSOMbOoU2svMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yMzA3MDQw
ODM1MzlaFw0yODA3MDIwODM1MzlaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
uhhUNlqAZdcWWB/OSDVDGk3OA99EFzOn/mJlmciQ/Xwu2dFJWmSCqEAE6gjufCjQ
q3voxAhC2CF+elKtJW/C0Sz/LYo60PUqd6iXF4h+upB9HkOOGuWHXsHBTsvgkgGA
1CGgel4U0Cdq+23eANr8N8m28UzljjSnTlrYCHtzN4sCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFBkZu3wT27NnYgrfH+xJz4HJaNJoMIGOBgNVHSME
gYYwgYOAFBkZu3wT27NnYgrfH+xJz4HJaNJooWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQC0jjGzqFNrLzASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBAECji43p+oPkYqmzll7e8Hgb
oADS0ph+YUz5P/bUCm61wFjlxaTfwKcuTR3ytj7bFLoW5Bm7Sa+TCl3lOGb2taon
2h+9NirRK6JYk87LMNvbS40HGPFumJL2NzEsGUeK+MRiWu+Oh5/lJGii3qw4YByx
SUDlRyNy1jJFstEZjOhs
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJAPu4ssY3BlzcMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTEyMDMy
MTI5MzJaGA8yMTk1MDUwODIxMjkzMlowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAsOiGi4A6+YTLzCdIyP8b8SCT2M/6PGKwzKJ5XbSBoL3gsnSWiFYqPg9c
uJPNbiy9wSA9vlyfWMd90qvTfiNrT6vewP813QdJ3EENZOx4ERcf/Wd22tV72kxD
```

```
yw1Q3I1OMH4bOItGQAxU5OtXCjBZEEUZooOkU8RoUQOU2Pql4NTiUpzWacNutAn5
HHS7MDc4lUlsJqbN+5QW6fFrcNG/0Mrib3JbwdFUNhrQ5j+Yq5h78HarnUivnX/3
Ap+oPbentv1qd7wvPJu556LZuhfqI0TohiIT1Ah+yUdN5osoaMxTHKKtf/CsSJ1F
w3qXqFJQA0VWsqjFyHXFI32I/GOupwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQCn
Um00QHvUsJSN6KATbghowLynHn3wZSQsuS8E0COpcFJFxP2SV0NYkERbXu0n/Vhi
yq5F8v4/bRA2/xpedLWmvFs7QWlomuXhSnYFkd33Z5gnXPb9vRkLwiMSw4uXls35
qQraczUJ9EXDhrv7VmngIk9H3YsxYrlDGEqh/oz4Ze4ULOgnfkauanHikk+BUEsg
/jsTD+7e+niEzJPihHdsvKFDlud5pakEzyxovHwNJ1GS2I//yxrJFIL91mehjqEk
RLPdNse7N6UvSnuXcOokwu6l6kfzigGkJBxkcq4gre3szZFdCQcUioj7Z4xtuTL8
YMqfiDtN5cbD8R8ojw9Y
-----END CERTIFICATE-----
```

AWS GovCloud (米国東部) - us-gov-east-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJAIe9Hnq82O7UMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yMTA3MTQx
NDI3NTdaFw0yNDA3MTMxNDI3NTdaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
```

```
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
qaIcGFFTx/SO1W5G91jHvyQdGP25n1Y91aXCuOOWAUTvSvNGpXrI4AXNrQF+CmIO
C4beBASnHCx082jYudWBBl9Wiza0psYc9flrczSzVLMmN8w/c78F/95NfiQdnUQP
pvgqcMeJo82cgHkLR7XoFWgMrZJqrcUK0gnsQcb6kakCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFNWV53gWJz72F5B1ZVY4O/dfFYBPMIGOBgNVHSME
gYYwgYOAFNWV53gWJz72F5B1ZVY4O/dfFYBPoWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQCHvR56vNju1DASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBACrKjWj460GUPZCGm3/z0dIz
M2BPuH769wcOsqfFZcMKEysSFK91tVtUb1soFwH4/Lb/T0PqNrvtEwD1Nva5k0h2
xZhNNRmDuhOhW1K9wCcnHGRBwY5t4lYL6hNV6hcrqYwGMjTjcAjBG2yMgznSNFle
Rwi/S3BFXISixNx9cILu
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJALPB6hxFhay8MA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xODA0MTAx
MjMyNDlaGA8yMTk3MDkxMzEyMzI0OVowXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAva9xsI9237KYb/SPWmeCVzi7giKNron8hoRDwlwwMC9+uHPd53UxzKLb
pTgtJWAPkZVxEdl2Gdhwr3SULoKcKmkqE6ltVFrVuPT33La1UufguT9k8ZDDuO9C
hQNHUdSVEuVrK3bLjaSsMOS7Uxmnn7lYT990IReowvnBNBsBlcabfQTBV04xfUG0
/m0XUiUFjOxDBqbNzkEIblW7vK7ydSJtFMSljga54UAVXibQt9EAIF7B8k9l2iLa
mu9yEjyQy+ZQICTuAvPUEWe6va2CHVY9gYQLA31/zU0VBKZPTNExjaqK4j8bKs1/
7dOV1so39sIGBz21cUBec1o+yCS5SwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQBt
hO2W/Lm+Nk0qsXW6mqQFsAou0cASc/vtGNCyBfoFNX6aKXsVCHxq2aq2TUKWENs+
mKmYu1lZVhBOmLshyllh3RRoL3Ohp3jCwXytkWQ7ElcGjDzNGc0FArzB8xFyQNdK
MNvXDi/ErzgrHGSpcvmGHiOhMf3UzChMWbIr6udoDlMbSIO7+8F+jUJkh4Xl1lKb
YeN5fsLZp7T/6YvbFSPpmbn1YoE2vKtuGKxObRrhU3h4JHdp1Zel1pZ6lh5iM0ec
SD11SximGIYCjfZpRqI3q50mbxCd7ckULz+UUPwLrfOds4VrVVSj+x0ZdY19Plv2
9shw5ez6Cn7E3IfzqNHO
-----END CERTIFICATE-----
```

AWS GovCloud (米国西部) - us-gov-west-1

DSA

```
-----BEGIN CERTIFICATE-----
MIIC7TCCAq0CCQCWukjZ5V4aZzAJBgcqhkjOOAQDMFwxCzAJBgNVBAYTAlVTMRkw
FwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYD
VQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0xMjAxMDUxMjU2MTJaFw0z
ODAxMDUxMjU2MTJaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9u
IFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNl
cnZpY2VzIExMQzCCAbcwggEsBgcqhkjOOAQBMIIBHwKBgQCjkvcS2bb1VQ4yt/5e
ih5OO6kK/n1Lzllr7D8ZwtQP8fOEpp5E2ng+D6Ud1Z1gYipr58Kj3nssSNpI6bX3
VyIQzK7wLclnd/YozqNNmgIyZecN7EglK9ITHJLP+x8FtUpt3QbyYXJdmVMegN6P
hviYt5JH/nYl4hh3Pa1HJdskgQIVALVJ3ER11+Ko4tP6nwvHwh6+ERYRAoGBAI1j
k+tkqMVHuAFcvAGKocTgsjJem6/5qomzJuKDmbJNu9Qxw3rAotXau8Qe+MBcJl/U
hhy1KHVpCGl9fueQ2s6IL0CaO/buycU1CiYQk40KNHCcHfNiZbdlx1E9rpUp7bnF
lRa2v1ntMX3caRVDdbtPEWmdxSCYsYFDk4mZrOLBA4GEAAKBgEbmeve5f8LIE/Gf
MNmP9CM5eovQOGx5ho8WqD+aTebs+k2tn92BBPqeZqpWRa5P/+jrdKml1qx4llHW
MXrs3IgIb6+hUIB+S8dz8/mmO0bpr76RoZVCXYab2CZedFut7qc3WUH9+EUAH5mw
vSeDCOUMYQR7R9LINYwouHIziqQYMAkGByqGSM44BAMDLwAwLAIUWXBlk40xTwSw
7HX32MxXYruse9ACFBNGmdX2ZBrVNGrN9N2f6ROk0k9K
-----END CERTIFICATE-----
```

RSA

```
-----BEGIN CERTIFICATE-----
MIIDCzCCAnSgAwIBAgIJAIe9Hnq82O7UMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAeFw0yMTA3MTQx
NDI3NTdaFw0yNDA3MTMxNDI3NTdaMFwxCzAJBgNVBAYTAlVTMRkwFwYDVQQIExBX
YXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0dGxlMSAwHgYDVQQKExdBbWF6
b24gV2ViIFNlcnZpY2VzIExMQzCBnzANBgkqhkiG9w0BAQEFAAOBjQAwgYkCgYEA
qaIcGFFTx/SO1W5G91jHvyQdGP25n1Y91aXCuOOWAUTvSvNGpXrI4AXNrQF+CmIO
C4beBASnHCx082jYudWBBl9Wiza0psYc9flrczSzVLMmN8w/c78F/95NfiQdnUQP
pvgqcMeJo82cgHkLR7XoFWgMrZJqrcUK0gnsQcb6kakCAwEAAaOB1DCB0TALBgNV
HQ8EBAMCB4AwHQYDVR0OBBYEFNWV53gWJz72F5B1ZVY4O/dfFYBPMIGOBgNVHSME
gYYwgYOAFNWV53gWJz72F5B1ZVY4O/dfFYBPoWCkXjBcMQswCQYDVQQGEwJVUzEZ
MBcGA1UECBMQV2FzaGluZ3RvbiBTdGF0ZTEQMA4GA1UEBxMHU2VhdHRsZTEgMB4G
A1UEChMXQW1hem9uIFdlYiBTZXJ2aWNlcyBMTEOCCQCHvR56vNju1DASBgNVHRMB
Af8ECDAGAQH/AgEAMA0GCSqGSIb3DQEBCwUAA4GBACrKjWj460GUPZCGm3/z0dIz
M2BPuH769wcOsqfFZcMKEysSFK91tVtUb1soFwH4/Lb/T0PqNrvtEwD1Nva5k0h2
xZhNNRmDuhOhW1K9wCcnHGRBwY5t4lYL6hNV6hcrqYwGMjTjcAjBG2yMgznSNFle
Rwi/S3BFXISixNx9cILu
```

```
-----END CERTIFICATE-----
```

RSA-2048

```
-----BEGIN CERTIFICATE-----
MIIDOzCCAiOgAwIBAgIJANCOF0Q6ohnuMA0GCSqGSIb3DQEBCwUAMFwxCzAJBgNV
BAYTAlVTMRkwFwYDVQQIExBXYXNoaW5ndG9uIFN0YXRlMRAwDgYDVQQHEwdTZWF0
dGxlMSAwHgYDVQQKExdBbWF6b24gV2ViIFNlcnZpY2VzIExMQzAgFw0xNTA5MTAx
OTQyNDdaGA8yMTk1MDIxMzE5NDI0N1owXDELMAkGA1UEBhMCVVMxGTAXBgNVBAgT
EFdhc2hpbmd0b24gU3RhdGUxEDAOBgNVBAcTB1NlYXR0bGUxIDAeBgNVBAoTF0Ft
YXpvbiBXZWIgU2VydmljZXMgTExDMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIB
CgKCAQEAzIcGTzNqie3f1olrrqcfzGfbymSM2QfbTzDIOG6xXXeFrCDAmOq0wUhi
3fRCuoeHlKOWAPu76B9os71+zgF22dIDEVkpqHCjBrGzDQZXXUwOzhm+PmBUI8Z1
qvbVD4ZYhjCujWWzrsX6Z4yEK7PEFjtf4M4W8euw0RmiNwjy+knIFa+VxK6aQv94
lW98URFP2fD84xedHp6ozZlr3+RZSIFZsOiyxYsgiwTbesRMI0Y7LnkKGCIHQ/XJ
OwSISWaCddbu59BZeADnyhl4f+pWaSQpQQ1DpXvZAVBYvCH97J1oAxLfH8xcwgSQ
/se3wtn095VBt5b7qTVjOvy6vKZazwIDAQABMA0GCSqGSIb3DQEBCwUAA4IBAQA/
S8+a9csfASkdtQUOLsBynAbsBCH9Gykq2m8JS7YE4TGvqlpnWehz78rFTzQwmz4D
fwq8byPkl6DjdF9utqZ0JUo/Fxelxom0h6oievtBlSkmZJNbgc2WYm1zi6ptViup
Y+4S2+vWZyg/X1PXD7wyRWuETmykk73uEyeWFBYKCHWsO9sI+62O4Vf8Jkuj/cie
1NSJX8fkervfLrZSHBYhxLbL+actVEo00tiyZz8GnhgWx5faCY38D/k4Y/j5Vz99
7lUX/+fWHT3+lTL8ZZK7fOQWh6NQpI0wTP9KtWqfOUwMIbgFQPoxkP00TWRmdmPz
WOwTObEf9ouTnjG9OZ20
-----END CERTIFICATE-----
```

インスタンスアイデンティティロール

作成する各インスタンスには、インスタンスアイデンティティを表すインスタンスアイデンティティロールがあります。インスタンスアイデンティティロールは IAM ロールの一種です。インスタンス ID ロールを使用するように統合されている AWS サービスと機能は、そのロールを使用してサービスのインスタンスを識別できます。

インスタンスアイデンティティロール認証情報は、`/identity-credentials/ec2/security-credentials/ec2-instance` のインスタンスメタデータサービス (IMDS) からアクセスできます。認証情報は、AWS 一時アクセスキー ID およびセッショントークンで構成されています。これらは、インスタンス ID ロールを使用する AWS サービスへの AWS Sigv4 リクエストに署名するために使用されます。認証情報は、インスタンスアイデンティティロールを使用するサービスまたは機能がインスタンスで有効になっているかどうかにかかわらず、インスタンスメタデータに存在します。

インスタンス ID ロールは、インスタンスの起動時に自動的に作成され、ロール信頼ポリシー文書はなく、ID ポリシーやリソースポリシーの対象にもなりません。

サポートされる サービス

AWS サービスはインスタンス ID ロールを使用します。

- Amazon EC2 – EC2 Instance Connect は、インスタンス ID ロールを使用して Linux インスタンスのホストキーを更新します。
- Amazon GuardDuty — EKS ランタイムモニタリングはインスタンスアイデンティティロールを使用して、ランタイムエージェントが GuardDuty VPC エンドポイントにセキュリティテレメトリを送信できるようにします。
- AWS Security Token Service (AWS STS) - インスタンス ID ロールの認証情報は AWS STS `GetCallerIdentity` アクションで使用できます。
- AWS Systems Manager - デフォルトのホスト管理設定を使用する場合、AWS Systems Manager はインスタンスアイデンティティロールによって提供された ID を使用して EC2 インスタンスを登録します。インスタンスを識別すると、システムマネージャーは `AWSSystemsManagerDefaultEC2InstanceManagementRole` IAM ロールをインスタンスに渡すことができます。

インスタンス ID ロールは、インスタンス ID ロールと統合されていないため、他の AWS サービスや機能では使用できません。

インスタンスアイデンティティロール ARN

インスタンスアイデンティティロール ARN は次の形式です。

```
arn:aws-partition:iam::account-number:assumed-role/aws:ec2-instance/instance-id
```

例:

```
arn:aws:iam::0123456789012:assumed-role/aws:ec2-instance/i-0123456789example
```

ARN の詳細については、「IAM ユーザーガイド」の「Amazon リソースネーム (ARN)」を参照してください。

Amazon Elastic Inference

> **Note**
>
> 2023 年 4 月 15 日以降、AWS では Amazon Elastic Inference (EI) への新規顧客のオンボーディングは行わず、既存の顧客がより価格とパフォーマンスの良いオプションにワークロードを移行できるよう支援します。2023 年 4 月 15 日以降、新規顧客は Amazon SageMaker、Amazon ECS、または Amazon EC2 の Amazon EI アクセラレータを使用してインスタンスを起動できなくなります。ただし、過去 30 日間に Amazon EI を少なくとも 1 回使用した顧客は、現在の顧客と見なされ、サービスを引き続き使用できます。

Amazon Elastic Inference (EI) は、Amazon EC2 の CPU インスタンスにアタッチして、深層学習 (DL) の推論ワークロードのリソースです。

Amazon EI の詳細については、「Amazon Elastic Inference デベロッパーガイド」を参照してください。

EC2 Linux インスタンスを特定する

場合によっては、アプリケーションにより EC2 インスタンスで実行されているかどうかを判断する必要があります。

Windows インスタンス特定の詳細については、Windows インスタンスの Amazon EC2 ユーザーガイド の「EC2 Windows インスタンスの特定」を参照してください。

インスタンスアイデンティティドキュメント の検査

EC2 インスタンスを識別する、暗号により確認された確実な方法については、その署名を含めて、インスタンスアイデンティティドキュメントを参照してください。これらのドキュメントは、ローカルのルーティングできないアドレス `http://169.254.169.254/latest/dynamic/instance-identity/` の各 EC2 インスタンスで入手できます。詳細については、インスタンスアイデンティティドキュメント を参照してください。

システム UUID の検査

システムの UUID を取得して、UUID の最初のオクテットに「ec2」または「EC2」という文字が存在するかどうかを検索することができます。システムが EC2 インスタンスであるかどうかを判断するこの方法は、EC2 インスタンスではないシステムがこれらの文字で始まる UUID を持つ可能性

が低いため、迅速でありながら不正確である可能性があります。で、EC2 インスタンスが Amazon Linux 2 を使用していない場合、SMBIOS のディストリビューションの実装については、リトルエンディアン形式で UUID を表すことがあるため、"EC2" の文字列は UUID の先頭には使用されません。

Example : DMI から UUID を取得 (HVM AMI のみ)

デスクトップ管理インターフェイス (DMI) を使用して UUID を取得するには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo dmidecode --string system-uuid
```

次の出力例では、UUID は「EC2」で始まりますが、これは多くの場合システムが EC2 インスタンスであることを示しています。

```
EC2E1916-9099-7CAF-FD21-012345ABCDEF
```

次の出力例では、UUID がリトルエンディアン形式で表されています。

```
45E12AEC-DCD1-B213-94ED-012345ABCDEF
```

別の方法として、Nitro システムに構築されたインスタンスの場合には、次のコマンドを使用できます。

```
[ec2-user ~]$ cat /sys/devices/virtual/dmi/id/board_asset_tag
```

次の例のように出力がインスタンス ID となる場合、そのシステムは EC2 インスタンスです。

```
i-0af01c0123456789a
```

Example : ハイパーバイザーから UUID を取得 (PV AMI のみ)

次のコマンドを使用して、ハイパーバイザーから UUID を取得します。

```
[ec2-user ~]$ cat /sys/hypervisor/uuid
```

次の出力例では、UUID は「ec2」で始まりますが、これは多くの場合システムが EC2 インスタンスであることを示しています。

```
ec2e1916-9099-7caf-fd21-012345abcdef
```

システムの仮想マシン生成識別子を調べる

仮想マシン生成識別子は、暗号化ランダム整数識別子として解釈される 128 ビットの一意のバッファで構成されます。仮想マシン生成識別子を取得すると、Amazon Elastic Compute Cloud インスタンスを識別できます。生成識別子は、ACPI テーブルエントリを介してインスタンスのゲストオペレーティングシステム内に公開されています。AWS にマシンをクローン、コピー、またはインポートすると、値は VM Import/Export などに変わります。

Example : Linux から仮想マシン生成識別子を取得する

次のコマンドを使用して、Linux を実行しているインスタンスから仮想マシン生成識別子を取得できます。

Amazon Linux 2

1. 必要に応じて、次のコマンドを使用して既存のソフトウェアパッケージを更新します。

   ```
   sudo yum update
   ```

2. 必要に応じて、次のコマンドを使用して busybox パッケージを入手します。

   ```
   sudo curl https://www.rpmfind.net/linux/epel/next/8/Everything/x86_64/Packages/
   b/busybox-1.35.0-2.el8.next.x86_64.rpm --output busybox.rpm
   ```

3. 必要に応じて、次のコマンドを使用して前提条件パッケージをインストールします。

   ```
   sudo yum install busybox.rpm iasl -y
   ```

4. `iasl` コマンドを実行して、ACPI テーブルから出力を生成します。

   ```
   sudo iasl -p ./SSDT2 -d /sys/firmware/acpi/tables/SSDT2
   ```

5. 次のコマンドを実行して、`iasl` コマンドの出力をレビューします。

   ```
   cat SSDT2.dsl
   ```

   仮想マシン生成識別子を取得するために必要なアドレス空間が生成されているはずです。

   ```
   Intel ACPI Component Architecture
   ASL+ Optimizing Compiler/Disassembler version 20190509
   Copyright (c) 2000 - 2019 Intel Corporation
   ```

```
File appears to be binary: found 32 non-ASCII characters, disassembling
Binary file appears to be a valid ACPI table, disassembling
Input file /sys/firmware/acpi/tables/SSDT2, Length 0x7B (123) bytes
ACPI: SSDT 0x0000000000000000 00007B (v01 AMAZON AMZNSSDT 00000001 AMZN
 00000001)
Pass 1 parse of [SSDT]
Pass 2 parse of [SSDT]
Parsing Deferred Opcodes (Methods/Buffers/Packages/Regions)

Parsing completed
Disassembly completed
ASL Output:    ./SSDT2.dsl - 1065 bytes
$
/*
 * Intel ACPI Component Architecture
 * AML/ASL+ Disassembler version 20190509 (64-bit version)
 * Copyright (c) 2000 - 2019 Intel Corporation
 *
 * Disassembling to symbolic ASL+ operators
 *
 * Disassembly of /sys/firmware/acpi/tables/SSDT2, Tue Mar 29 16:15:14 2022
 *
 * Original Table Header:
 *     Signature        "SSDT"
 *     Length           0x0000007B (123)
 *     Revision         0x01
 *     Checksum         0xB8
 *     OEM ID           "AMAZON"
 *     OEM Table ID     "AMZNSSDT"
 *     OEM Revision     0x00000001 (1)
 *     Compiler ID      "AMZN"
 *     Compiler Version 0x00000001 (1)
 */
DefinitionBlock ("", "SSDT", 1, "AMAZON", "AMZNSSDT", 0x00000001)
{
    Scope (\_SB)
    {
        Device (VMGN)
        {
            Name (_CID, "VM_Gen_Counter")  // _CID: Compatible ID
            Name (_DDN, "VM_Gen_Counter")  // _DDN: DOS Device Name
            Name (_HID, "AMZN0000")  // _HID: Hardware ID
            Name (ADDR, Package (0x02)
```

```
            {
                0xFED01000,
                Zero
            })
        }
    }
}
```

6. (オプション) 次のコマンドを使用して、残りのステップのターミナル許可を昇格させます。

```
sudo -s
```

7. 次のコマンドを使用して、以前に収集したアドレス空間を保存します。

```
VMGN_ADDR=0xFED01000
```

8. 次のコマンドを使用して、アドレス空間を介して反復処理し、仮想マシン生成識別子を作成します。

```
for offset in 0x0 0x4 0x8 0xc; do busybox devmem $(($VMGN_ADDR + $offset)) | sed
 's/0x//' | sed -z '$ s/\n$//' >> vmgenid; done
```

9. 次のコマンドを使用して、出力ファイルから仮想マシン生成識別子を取得します。

```
cat vmgenid ; echo
```

出力は次のようになります。

```
EC2F335D979132C4165896753E72BD1C
```

Ubuntu

1. 必要に応じて、次のコマンドを使用して既存のソフトウェアパッケージを更新します。

```
sudo apt update
```

2. 必要に応じて、次のコマンドを使用して前提条件パッケージをインストールします。

```
sudo apt install busybox iasl -y
```

3. iasl コマンドを実行して、ACPI テーブルから出力を生成します。

```
sudo iasl -p ./SSDT2 -d /sys/firmware/acpi/tables/SSDT2
```

4. 次のコマンドを実行して、iasl コマンドの出力をレビューします。

```
cat SSDT2.dsl
```

仮想マシン生成識別子を取得するために必要なアドレス空間が生成されているはずです。

```
Intel ACPI Component Architecture
ASL+ Optimizing Compiler/Disassembler version 20190509
Copyright (c) 2000 - 2019 Intel Corporation

File appears to be binary: found 32 non-ASCII characters, disassembling
Binary file appears to be a valid ACPI table, disassembling
Input file /sys/firmware/acpi/tables/SSDT2, Length 0x7B (123) bytes
ACPI: SSDT 0x0000000000000000 00007B (v01 AMAZON AMZNSSDT 00000001 AMZN
 00000001)
Pass 1 parse of [SSDT]
Pass 2 parse of [SSDT]
Parsing Deferred Opcodes (Methods/Buffers/Packages/Regions)

Parsing completed
Disassembly completed
ASL Output:    ./SSDT2.dsl - 1065 bytes
$
/*
 * Intel ACPI Component Architecture
 * AML/ASL+ Disassembler version 20190509 (64-bit version)
 * Copyright (c) 2000 - 2019 Intel Corporation
 *
 * Disassembling to symbolic ASL+ operators
 *
 * Disassembly of /sys/firmware/acpi/tables/SSDT2, Tue Mar 29 16:15:14 2022
 *
 * Original Table Header:
 *     Signature        "SSDT"
 *     Length           0x0000007B (123)
 *     Revision         0x01
 *     Checksum         0xB8
 *     OEM ID           "AMAZON"
```

```
 *     OEM Table ID     "AMZNSSDT"
 *     OEM Revision     0x00000001 (1)
 *     Compiler ID      "AMZN"
 *     Compiler Version 0x00000001 (1)
 */
DefinitionBlock ("", "SSDT", 1, "AMAZON", "AMZNSSDT", 0x00000001)
{
    Scope (\_SB)
    {
        Device (VMGN)
        {
            Name (_CID, "VM_Gen_Counter")  // _CID: Compatible ID
            Name (_DDN, "VM_Gen_Counter")  // _DDN: DOS Device Name
            Name (_HID, "AMZN0000")  // _HID: Hardware ID
            Name (ADDR, Package (0x02)
            {
                0xFED01000,
                Zero
            })
        }
    }
}
```

5. (オプション) 次のコマンドを使用して、残りのステップのターミナル許可を昇格させます。

```
sudo -s
```

6. 次のコマンドを使用して、以前に収集したアドレス空間を保存します。

```
VMGN_ADDR=0xFED01000
```

7. 次のコマンドを使用して、アドレス空間を介して反復処理し、仮想マシン生成識別子を作成します。

```
for offset in 0x0 0x4 0x8 0xc; do busybox devmem $(($VMGN_ADDR + $offset)) | sed
 's/0x//' | sed -z '$ s/\n$//' >> vmgenid; done
```

8. 次のコマンドを使用して、出力ファイルから仮想マシン生成識別子を取得します。

```
cat vmgenid ; echo
```

出力は次のようになります。

```
EC2F335D979132C4165896753E72BD1C
```

EC2 フリートとスポットフリート

EC2 フリートとスポットフリートは、AWS でインスタンスのフリート (グループ) を起動するのに便利な方法となるように設計されています。フリート内の各インスタンスは起動テンプレートに基づいています。

フリートは次の特徴と利点を有しています。これらの利点により、複数の EC2 インスタンスでアプリケーションを実行するときに、コスト削減を最大化し、可用性とパフォーマンスを最適化できます。

複数のインスタンスタイプと購入オプション

単一の API コールで、フリートは複数のインスタンスタイプと購入オプション (スポットインスタンスおよびオンデマンドインスタンス) を起動できるため、スポットインスタンスの使用によってコストを最適化できます。また、リザーブドインスタンスと Savings Plans の割引は、フリート内のオンデマンドインスタンスと併用することで活用できます。

複数のアベイラビリティーゾーンにインスタンスを分散する

フリートは、複数のアベイラビリティーゾーンに自動で均等にインスタンスを配分して、高可用性を得られます。これにより、アベイラビリティーゾーンが使用できなくなった場合にも回復性が得られます。追加のメリットとして、単一のアベイラビリティーゾーン内のフリートと比較して、より深い Amazon EC2 キャパシティプールにアクセスできます。

スポットインスタンスの自動交換

フリートにスポットインスタンスが含まれている場合、インスタンスの状態の変化によりスポットインスタンスが中断されたり、機能しなくなったりした場合に、スポット容量の交換を自動的にリクエストできます。キャパシティリバランスにより、フリートは、中断されるリスクが高いスポットインスタンスを監視し、積極的に交換できます。

一般的なベストプラクティスとして、Amazon EC2 Auto Scaling でスポットインスタンスとオンデマンドインスタンスのフリートを起動することをお勧めします。フリートの管理に使用できる追加機能が提供されるからです。追加機能には、スポットインスタンスとオンデマンドインスタンスの両方の自動ヘルスチェック交換、アプリケーションベースのヘルスチェック、アプリケーショントラフィックを正常なインスタンスに均等に分散するための Elastic Load Balancing との統合が含まれます。Amazon ECS、Amazon EKS (セルフマネージドノードグループ)、Amazon VPC Lattice などの AWS サービスを使用するときも、Auto Scaling グループを使用できます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドを参照してください。

EC2 フリートは、インスタンスのライフサイクルやスケーリングメカニズムの側面をより柔軟に管理する必要がある場合に適しています。スポットフリートを使用することも可能ですが、推奨されていません。計画的な投資がないレガシー API だからです。ただし、既にスポットフリートを使用している場合は、引き続き使用できます。スポットフリートと EC2 フリートは同じコア機能を提供します。

トピック

EC2 Fleet

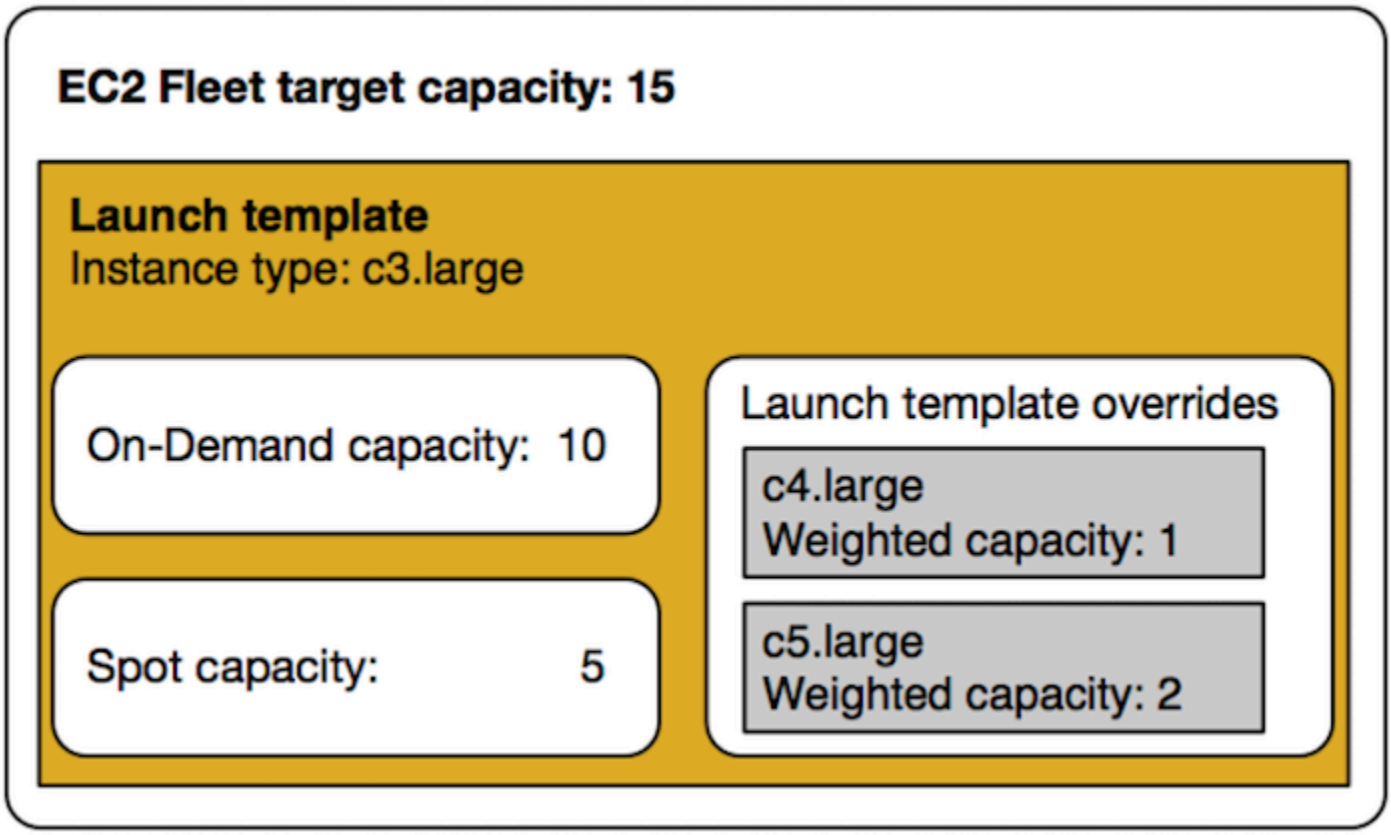
EC2 フリートには、インスタンスのフリートを起動するための設定情報が含まれています。単一の API コールで、フリートは、スポットインスタンス、オンデマンドインスタンス、リザーブドインスタンス、Savings Plans の購入オプションを一緒に使用して、複数のアベイラビリティーゾーンにまたがって複数のインスタンスタイプを起動できます。EC2 フリート を使用して、以下のことができます。

- スポットおよびオンデマンドのターゲット容量を別個に定義し、さらに 1 時間あたりの支払い上限料金を定義する
- アプリケーションに最適なインスタンスタイプを指定する
- 各購入オプション内でフリート容量を Amazon EC2 で分散する方法を指定する

フリートに対する 1 時間あたりの支払い上限容量を設定し、上限料金に達するまで EC2 フリート でインスタンスを起動することもできます。支払い上限料金に達すると、ターゲット容量に満たない場合でも、フリートはインスタンスの起動を停止します。

EC2 フリート は、リクエストで指定したターゲット容量を満たすために必要なインスタンス数の起動を試みます。1 時間あたりの上限の合計料金を指定すると、支払いの上限料金に達するまで、容量が満たされます。また、スポットインスタンス が中断した場合、フリートはスポットのターゲット

容量を維持しようとします。詳細については、「スポットインスタンス のしくみ」を参照してください。



EC2 フリート ごとにインスタンスタイプを無制限に指定できます。これらのインスタンスタイプは、スポットおよびオンデマンド購入オプションの両方を使用してプロビジョニングできます。複数のアベイラビリティーゾーンを指定し、インスタンスごとに異なる最大スポット料金を指定し、フリートごとに追加のスポットオプションを選択することもできます。Amazon EC2 は、フリートの起動時に指定されたオプションを使用して容量をプロビジョニングします。

フリートの実行中に、価格の値上げまたはインスタンスの障害のために Amazon EC2 が スポットインスタンス を再利用する場合、EC2 フリート は指定するインスタンスタイプのいずれかで、そのインスタンスを置き換えようとします。これにより、スポット料金の急激な増加中に容量を再取得することが容易になります。フリートごとに、柔軟で順応性に富むリソース戦略を作成できます。例えば、特定のフリート内で、プライマリ容量に、利用できる場合はより安価なスポット容量をオンデマンドで補足することができます。

リザーブドインスタンス があり、フリートで オンデマンドインスタンス を指定した場合、EC2 フリート は リザーブドインスタンス を使用します。例えば、フリートがオンデマンドインスタンスを `c4.large` として指定し、`c4.large` のリザーブドインスタンスがある場合、リザーブドインスタンスの価格を受け取ります。Savings Plans を使用する場合も同様です。

EC2 フリート は追加料金なしで使用できます。フリートが起動した EC2 インスタンスに対してのみお支払いいただきます。

コンテンツ

EC2 フリート の制限事項

以下の制限が EC2 フリート に適用されます。

- EC2 フリートは、Amazon EC2 API、AWS CLI、AWS SDK、およびAWS CloudFormation のみで利用可能です。
- EC2 フリート リクエストは、AWS リージョンにまたがることはできません。リージョンごとに別個の EC2 フリート を作成する必要があります。
- EC2 フリート リクエストは、同じアベイラビリティーゾーンから複数の異なるサブネットにまたがることはできません。

バーストパフォーマンスインスタンス

バーストパフォーマンスインスタンスタイプ を使用して スポットインスタンス を起動し、CPU クレジットを蓄積するアイドル時間なしでバーストパフォーマンス スポットインスタンス をすぐに短時間使用する場合は、支払いコストが高くなるのを避けるために、インスタンスを 標準モード で起動することをお勧めします。バーストパフォーマンス スポットインスタンス を Unlimited モード で起動し、すぐに CPU をバーストさせると、余分なクレジットがバーストに消費されます。インスタンスを短時間使用する場合、インスタンスは余分なクレジットに見合うだけの CPU クレジットを蓄積する時間がないため、インスタンスの終了時に余分なクレジットに対して課金されます。

Unlimited モードがバーストパフォーマンス スポットインスタンス に適しているのは、バースト用の CPU クレジットが蓄積されるまで、そのインスタンスが十分に長く実行される場合のみです。それ以外の場合は、余分なクレジットを支払う必要があるため、バーストパフォーマンス スポットインスタンス は他のインスタンスよりも、使用コストが高くなります。詳細については、「Unlimited モードと固定 CPU を使用する場合」を参照してください。

起動クレジットは、インスタンスを構成するために十分なコンピューティングリソースを提供し、T2 インスタンスの初期起動を効率的に実現することを意図しています。T2 インスタンスの起動を繰り返して新しい起動クレジットにアクセスすることは許可されていません。CPU が持続的に必要な場合、(一定期間のアイドリングにより) クレジットを獲得して T2 スポットインスタンス の Unlimited モード を使用するか、専用 CPU を搭載したインスタンスタイプを使用します。

EC2 フリートのリクエストタイプ

EC2 フリート リクエストには、次の 3 つの種類があります。

`instant`

リクエストタイプを `instant` として設定すると、EC2 フリートは必要な容量に対して同期ワンタイムリクエストを送信します。API レスポンスで、起動したインスタンスとともに起動できなかったインスタンスのエラーを返します。詳細については、「EC2フリート「インスタント」タイプの使用」を参照してください。

`request`

リクエストタイプを `request` として設定すると、EC2 フリートは、必要な容量に対して非同期の 1 回限りのリクエストを送信します。それ以降にスポットの中断のためにキャパシティーが減少した場合、フリートは スポットインスタンス の補充を試みません。また、キャパシティーが利用できない場合にも代替のスポットキャパシティープールへのリクエストを送信しません。

`maintain`

(デフォルト) リクエストタイプを `maintain` として設定すると、EC2 フリートは目的の容量に対して非同期リクエストを送信し、中断されたスポットインスタンスを自動的に補充することで容量を維持します。

3 つのタイプすべてのリクエストが、配分戦略の恩恵を受けます。詳細については、「スポットインスタンスの配分戦略」を参照してください。

EC2フリート「インスタント」タイプの使用

EC2 フリート インスタント タイプは、希望する容量を起動するために 1 回だけ試行する、同期ワンタイムリクエストです。API レスポンスは、起動したインスタンスとともに、起動できなかったインスタンスのエラーを一覧表で表示します。このガイドで述べている、EC2 フリートの インスタント タイプを使用することにはいくつかの利点があります。構成例については、ガイドの最後に記載しています。

EC2 インスタンスを起動するために起動専用 API が必要なワークロードの場合は、RunInstances API を使用できます。ただし、RunInstances では、オンデマンドインスタンスまたはスポットインスタンスのみを起動できますが、同じリクエストで両方を起動することはできません。さらに、RunInstances を使用してスポットインスタンスを起動する場合、スポットインスタンスリクエストは 1 つのインスタンスタイプと 1 つのアベイラビリティーゾーンに制限されます。これは、単一のスポットキャパシティープール (同じインスタンスタイプとアベイラビリティーゾーンを有する、未使用のインスタンスセット) をターゲットにしています。スポットキャパシティープールに、リクエストに対して十分なスポットインスタンス容量がない場合、RunInstances 呼び出しは失敗します。

RunInstances を使用してスポットインスタンスを起動する代わりに、CreateFleet API を `instant` に設定した `type` パラメータと使用すると、以下の利点があります。

- オンデマンドインスタンスとスポットインスタンスを 1 回のリクエストで起動します。EC2 フリートは、オンデマンドインスタンス、スポットインスタンス、またはその両方を起動できます。スポットインスタンスへのリクエストは、利用可能な容量があり、リクエストで指定した 1 時間あたりの上限料金がスポット料金を超えている場合に達成されます。
- スポットインスタンスの可用性を向上させます。EC2 フリートタイプ `instant` を使用してスポットインスタンスを起動でき、以下のような スポットベストプラクティス という利点があります:
  - ベストプラクティス: インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する。

    利点:複数のインスタンスタイプとアベイラビリティーゾーンを指定すると、スポットキャパシティープールの数が増加します。これにより、スポットサービスは、希望するスポットコンピューティング容量を見つけて割り当てる可能性が高くなります。経験則としては、ワークロードごとに少なくとも 10 種類のインスタンスタイプで柔軟に対応し、すべてのアベイラビリティーゾーンが VPC で使用するように設定されていることを確認します。
  - スポットベストプラクティス:容量を最適化する配分戦略を使用する。

    利点:最も利用性の高いスポットキャパシティプールから、`capacity-optimized` 配分戦略が自動的にインスタンスをプロビジョニングします。最適な容量を持つプールからスポットインスタンス容量が供給されるため、Amazon EC2 が容量を元に戻す必要があるときにスポットインスタンスが中断されます。
- 幅広い機能にアクセスする。起動専用 API が必要なワークロードで、EC2 フリートにインスタンスのライフサイクルを管理させるのではなく、インスタンスのライフサイクルを管理したい場合は、RunInstances API の代わりに EC2 フリートタイプ `instant` を使用します。EC2 フリートは、次の例で示すように、RunInstances よりも幅広い機能を提供します。その他のすべてのワー

クロードについては、Amazon EC2 Auto Scaling を使用する必要があります。これは、ELB ベースのアプリケーション、コンテナ化されたワークロード、キュー処理ジョブなど、さまざまなワークロードに対してより包括的な機能セットを提供するからです。

EC2 フリートインスタントタイプを使用して、キャパシティブロックにインスタンスを起動できます。詳細については、「チュートリアル: キャパシティブロックでインスタンスを起動する」を参照してください。

Amazon EC2 Auto Scaling や Amazon EMR などの AWS サービスでは、EC2 フリートの インスタント タイプを使用し、EC2 インスタンスを起動します。

EC2 フリートインスタントタイプ の前提条件

EC2 フリートを作成するための前提条件については、「EC2 フリートの前提条件」を参照してください。

瞬時に実行される EC2 フリート機能

EC2 フリートタイプ `instant` で作業する場合、イベントのシーケンスは以下のようになります。

1. `instant` のような CreateFleet リクエストタイプを設定してください。詳細については、「EC2 フリートの作成」を参照してください。API コールを行った後は、それを変更することはできません。
2. API コールを行うとき、EC2 フリート は、希望する容量に同期ワンタイムリクエストを配置します。
3. API レスポンスは、起動したインスタンスとともに、起動できなかったインスタンスのエラーを一覧表で表示します。
4. EC2 フリートの説明、EC2 フリートに関連付けられたインスタンスの一覧表示、EC2 フリートの履歴の表示を行うことができます。
5. インスタンスが起動したら、フリートリクエストを削除 できます。フリートリクエストを削除するときに、関連するインスタンスを終了するか、実行したままにすることもできます。
6. インスタンスはいつでも終了できます。

例

以下の例では、EC2 フリートタイプ `instant` の使用方法を示します。さまざまなユースケースで使用できます。EC2 CreateFleet API パラメータの使用方法の詳細については、Amazon EC2 API リファレンス 内の CreateFleet を参照してください。

例

例 1: 容量最適化配分戦略を使用してスポットインスタンスを起動する

次の例では、EC2 フリートのタイプで必要なパラメータ(起動テンプレート、ターゲット容量、デフォルト購入オプション、および起動テンプレートオーバーライド) を指定します。`instant`

- 起動テンプレートは、起動テンプレート名とバージョン番号によって識別されます。
- 12 の起動テンプレートオーバーライドでは、4 つの異なるインスタンスタイプと 3 つの異なるサブネットが指定され、それぞれ別のアベイラビリティーゾーンに配置されます。各インスタンスタイプとサブネットの組み合わせによってスポットキャパシティープールが定義され、12 個のスポットキャパシティープールが作成されます。
- フリートのターゲット容量は 20 インスタンスです。
- デフォルト購入オプションの `spot` では、フリートは、起動中のインスタンス数の最適な容量のスポットキャパシティプールに 20 個のスポットインスタンスを起動しようとします。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
```

```
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
```

```
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-fae8c380"
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-e7188bab"
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 2: 容量最適化割り当て戦略を使用して 1 つのスポットインスタンスを起動する

複数の EC2 フリート API コールタイプ `instant` を行い、TotalTargetCapacity を 1 に設定することで、一度に 1 つのスポットインスタンスを最適に起動できます。

次の例では、EC2 フリートインスタントタイプで必要なパラメータ (起動テンプレート、ターゲット容量、デフォルト購入オプション、および起動テンプレートオーバーライド) を指定します。起動テンプレートは、起動テンプレート名とバージョン番号によって識別されます。12 の起動テンプレートオーバーライドには、4 つの異なるインスタンスタイプと 3 つの異なるサブネットがあり、それぞれ別のアベイラビリティーゾーンにあります。フリートのターゲット容量は 1 インスタンスで、デフォルトの購入オプションはスポットです。これにより、フリートは、容量最適化の割り当て戦略に基づいて 12 のスポットキャパシティープールのいずれかからスポットインスタンスを起動し、最も利用可能な容量プールからスポットインスタンスを起動しようとします。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
```

```
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-fae8c380"
            },
```

```
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
   ],
   "TargetCapacitySpecification": {
       "TotalTargetCapacity": 1,
       "DefaultTargetCapacityType": "spot"
   },
   "Type": "instant"
}
```

例 3: インスタンスの重み付けを使用して、スポットインスタンスを起動する

次の例では、インスタンス分量指定を使っています。これは、料金が 1 インスタンス時間当たりではなく、1 ユニット時間当たりであることを意味します。各起動設定では、ワークロードのユニットに 15 GB のメモリと 4 vCPU が必要であると仮定して、インスタンスで実行できるワークロードのユニット数に基づいて、異なるインスタンスタイプと異なる重みが表示されます。たとえば、m5.xlarge (4 vCPU と 16 GB のメモリ) は 1 つのユニットを実行でき、重み付けは 1、m5.2xlarge (8 vCPU と 32 GB のメモリ) は 2 ユニットを実行でき、重み付けは 2 というように続きます。総目標容量は 40 ユニットに設定されています。デフォルトの購入オプションはスポットで、割り当て戦略は容量最適化です。これにより、40 m5.xlarge (40 を 1 で割ったもの) 、20 m5.2xlarge (40 を 2 で割ったもの) 、10 m5.4xlarge (40 を 4 で割ったもの) 、5 m5.8xlarge (40 を 8 で割ったもの) 、またはインスタンスタイプの組み合わせのいずれかになります。容量を最適化した割り当て戦略に基づきます。

詳細については、「EC2 フリートインスタンスの分量指定」を参照してください。

```
{
   "SpotOptions":{
      "AllocationStrategy":"capacity-optimized"
   },
   "LaunchTemplateConfigs":[
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
```

```
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"m5.xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":1
            },
            {
               "InstanceType":"m5.xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":1
            },
            {
               "InstanceType":"m5.xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":1
            },
            {
               "InstanceType":"m5.2xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":2
            },
            {
               "InstanceType":"m5.2xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":2
            },
            {
               "InstanceType":"m5.2xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":2
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":4
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":4
            },
            {
```

```
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":4
            },
            {
               "InstanceType":"m5.8xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":8
            },
            {
               "InstanceType":"m5.8xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":8
            },
            {
               "InstanceType":"m5.8xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":8
            }
         ]
      }
   ],
   "TargetCapacitySpecification":{
      "TotalTargetCapacity":40,
      "DefaultTargetCapacityType":"spot"
   },
   "Type":"instant"
}
```

例 4:1 つのアベイラビリティーゾーン内でスポットインスタンスを起動する

スポットオプション singleAvailabilityZone を true に設定することで、1 つのアベイラビリティーゾーンですべてのインスタンスを起動するようにフリートを設定できます。

12 の起動テンプレートオーバーライドでは、インスタンスタイプとサブネット (それぞれ別々のアベイラビリティーゾーン内) が異なりますが、重み容量は同じです。合計ターゲット容量は 20 インスタンスで、デフォルトの購入オプションは スポットで、スポット配分戦略は容量最適化です。EC2 フリートは、起動仕様を使用して最適な容量を持つスポットキャパシティープールから、1 つの AZ で 20 個のスポットインスタンスをすべて起動します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
```

```
        "SingleAvailabilityZone": true
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
```

```
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 5: 単一アベイラビリティーゾーン内で単一インスタンスタイプのスポットインスタンスを起動する

SpotOptions singleInstanceType を true、SingleAvailabilityZone を true に設定することで、同じインスタンスタイプのすべてのインスタンスを単一のアベイラビリティーゾーンで起動するようにフリートを設定できます。

12 の起動テンプレートオーバーライドでは、インスタンスタイプとサブネット (それぞれ別々のアベイラビリティーゾーン内) が異なりますが、重み容量は同じです。合計ターゲット容量は 20 インスタンスで、デフォルトの購入オプションは spot で、スポット配分戦略は容量最適化です。EC2 フリートは、起動仕様を使用して最適な容量でスポットインスタンスプールから、同じインスタンスタイプの 20 個のスポットインスタンスをすべて単一の AZ で起動します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true
    },
```

```
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
```

```
                "InstanceType":"m5d.4xlarge",
                "SubnetId":"subnet-fae8c380"
            },
            {
                "InstanceType":"m5d.4xlarge",
                "SubnetId":"subnet-e7188bab"
            },
            {
                "InstanceType":"m5d.4xlarge",
                "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 6: 最小ターゲット容量を起動できる場合にのみスポットインスタンスを起動する

スポットオプション MinTargetCapacity を一緒に起動する最小ターゲットキャパシティに設定することで、最小ターゲットキャパシティを起動できる場合にのみインスタンスを起動するようにフリートを設定できます。

12 の起動テンプレートオーバーライドでは、インスタンスタイプとサブネット (それぞれ別々のアベイラビリティーゾーン内) が異なりますが、重み容量は同じです。合計ターゲット容量と最小ターゲット容量は両方とも 20 インスタンスに設定され、デフォルトの購入オプションはスポット、スポット割り当て戦略は容量最適化です。EC2 フリートは、起動テンプレートのオーバーライドを使用して、最適な容量でスポットキャパシティープールから 20 個のスポットインスタンスを起動します。これは、20 個のインスタンスをすべて同時に起動できる場合のみです。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "MinTargetCapacity": 20
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
```

```
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
```

```
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 7: 単一のアベイラビリティーゾーンで同じインスタンスタイプで最小ターゲット容量を起動できる場合にのみスポットインスタンスを起動する

スポットオプション MinTargetCapacity を SingleInstanceType および SingleAvailabilityZone オプションとともに起動する最小ターゲットキャパシティに設定することで、単一のアベイラビリティーゾーン内の単一のインスタンスタイプで最小ターゲットキャパシティを起動できる場合にのみ、インスタンスを起動するようにフリートを設定できます。

起動テンプレートをオーバーライドする 12 の起動条件は、インスタンスタイプとサブネットが異なりますが(それぞれ異なるアベイラビリティーゾーン内で)、加重容量は同じです。合計ターゲット容量と最小ターゲット容量は両方とも 20 インスタンスに設定され、デフォルトの購入オプションはスポット、スポット割り当て戦略は容量最適化、SingleInstanceType は true、SingleAvailabilityZone は true です。EC2 フリートは、同じインスタンスタイプの 20 個のスポットインスタンスをすべて 1 つの AZ で起動し、起動条件を使用して最適な容量を持つスポットキャパシティープールから起動します。これは、20 個のインスタンスをすべて同時に起動できる場合に限ります。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true,
        "MinTargetCapacity": 20
    },
    "LaunchTemplateConfigs": [
```

```
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5.4xlarge",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.4xlarge",
```

```
                "SubnetId":"subnet-fae8c380"
            },
            {
                "InstanceType":"m5d.4xlarge",
                "SubnetId":"subnet-e7188bab"
            },
            {
                "InstanceType":"m5d.4xlarge",
                "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 8: 複数の起動テンプレートを使用したインスタンスの起動

複数の起動テンプレートを指定することで、異なるインスタンスタイプまたはインスタンスタイプのグループに対して異なる起動条件でインスタンスを起動するようにフリートを設定できます。この例では、インスタンスタイプごとに異なる EBS ボリュームサイズが必要で、起動テンプレート ec2-fleet-lt-4xl、ec2-fleet-lt-9xl、ec2-fleet-lt-18xl で設定されています。

この例では、サイズに基づいて 3 種類のインスタンスタイプに対して 3 種類の起動テンプレートを使用します。すべての起動テンプレートの起動条件オーバーライドでは、インスタンスタイプの vCPUs に基づくインスタンスの重みが使用されます。合計ターゲット容量は 144 ユニットで、デフォルトの購入オプションは spot で、スポット配分戦略は容量最適化です。EC2 フリートは、起動テンプレート ec2-fleet-4xl を使用して 9 c5n.4xlarge (144 を 16 で割った) を起動するか、起動テンプレート ec2-fleet-9xl を使用して 4 c5n.9xlarge (144 を 36 で割った) を起動するか、 または起動テンプレート ec2-fleet-18xl を使用して 2 c5n.18xlarge (144 を 72 で割った) を起動するか、もしくはインスタンスタイプ容量最適化の割り当て戦略に基づいて重みを追加したインスタンスタイプの混合を使って起動することができます。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
```

```
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt-18xl",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5n.18xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":72
            },
            {
               "InstanceType":"c5n.18xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":72
            },
            {
               "InstanceType":"c5n.18xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":72
            }
         ]
      },
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt-9xl",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5n.9xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":36
            },
            {
               "InstanceType":"c5n.9xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":36
            },
            {
               "InstanceType":"c5n.9xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":36
            }
```

```
         ]
      },
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt-4xl",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5n.4xlarge",
               "SubnetId":"subnet-fae8c380",
               "WeightedCapacity":16
            },
            {
               "InstanceType":"c5n.4xlarge",
               "SubnetId":"subnet-e7188bab",
               "WeightedCapacity":16
            },
            {
               "InstanceType":"c5n.4xlarge",
               "SubnetId":"subnet-49e41922",
               "WeightedCapacity":16
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 144,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 9: オンデマンドインスタンスのベースを使用してスポットインスタンスを起動する

次の例では、フリートの合計ターゲット容量を 20 インスタンス、ターゲット容量を 5 オンデマンドインスタンス として指定します。デフォルト購入オプションはスポットです。フリートは指定されたとおり 5 オンデマンドインスタンス を起動しますが、合計ターゲット容量を満たすために、さらに 15 以上のインスタンスを起動する必要があります。差額の購入オプションは、TotalTargetCapacity - OnDemandTargetCapacity = DefaultTargetCapacityType で計算されます。この結果、15 のスポットインスタンスを起動するフリートは容量最適化配分戦略に基づいて 12 スポットキャパシティープールのうちの 1 つを形成します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab"
            },
```

```
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-fae8c380"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-e7188bab"
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-49e41922"
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "OnDemandTargetCapacity": 5,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 10: キャパシティーの予約および優先順位配分戦略を使用したオンデマンドインスタンスをベースにして、キャパシティー最適化配分戦略を使用しスポットインスタンスを起動する

キャパシティーの予約の使用戦略を use-capacity-reservations-first に設定することで、デフォルトのターゲット容量タイプをスポットにしたオンデマンドインスタンス の起動時に、最初にオンデマンドキャパシティー予約 を使用するようにフリートを設定できます。また、複数のインスタンスプールに未使用のキャパシティーの予約がある場合、選択したオンデマンド配分戦略が適用されます。この例では、オンデマンド配分戦略は優先されています。

この例では、利用可能な未使用の予約予約 が 6 個あります。これは、フリートの目標オンデマンド容量である 10 オンデマンドインスタンス を下回っています。

アカウントには、2 つのプールに 6 個の未使用キャパシティーの予約 があります。各プールのキャパシティーの予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

以下のフリート設定は、この例に関連する設定のみを示しています。オンデマンド配分戦略は優先的であり、キャパシティーの予約の使用戦略予約は use-capacity-reservations-first です。スポット配分戦略を容量最適化する 合計ターゲット容量は 20 で、オンデマンドターゲット容量は 10 で、デフォルトのターゲット容量タイプはスポットです。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized"
    },
    "OnDemandOptions":{
       "CapacityReservationOptions": {
         "UsageStrategy": "use-capacity-reservations-first"
       },
       "AllocationStrategy":"prioritized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
```

```
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 1.0
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 2.0
            },
            {
               "InstanceType":"c5.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 3.0
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 4.0
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 5.0
            },
            {
               "InstanceType":"c5d.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 6.0
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 7.0
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 8.0
            },
            {
               "InstanceType":"m5.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 9.0
```

```
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-fae8c380",
               "Priority": 10.0
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-e7188bab",
               "Priority": 11.0
            },
            {
               "InstanceType":"m5d.large",
               "SubnetId":"subnet-49e41922",
               "Priority": 12.0
            }
         ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "OnDemandTargetCapacity": 10,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

上記の設定を使用してインスタントフリートを作成すると、目標容量を満たすために以下の 20 個のインスタンスが起動されます。

- 7 c5.large オンデマンドインスタンス (us-east-1a) – c5.large (us-east-1a) が最初に優先され、利用可能な未使用 c5.large キャパシティーの予約が 3 つあります。キャパシティーの予約は、3 つの オンデマンドインスタンス を起動するために最初に使用され、さらにオンデマンド配分戦略に従って、この例で優先されている追加の 4 つのオンデマンドインスタンス が起動されます。
- 3 m5.large オンデマンドインスタンス (us-east-1a) – m5.large (us-east-1a) が 2 番目に優先され、利用可能な未使用 c3.large キャパシティーの予約が 3 つあります。
- 容量最適化割り当て戦略に従って最適な容量を持つ 12 個のスポットキャパシティープールのうちの 1 つからの 10 個のスポットインスタンス。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、c5.large および m5.large のすべてのキャパシティーの予約が使用されていることを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.large",
    "AvailableInstanceCount": 0
}
```

例 11: 容量最適化優先順位配分戦略を使用してスポットインスタンスを起動する

次の例では、EC2 フリートインスタントタイプで必要なパラメータ (起動テンプレート、ターゲット容量、デフォルト購入オプション、および起動テンプレートオーバーライド) を指定します。起動テンプレートは、起動テンプレート名とバージョン番号によって識別されます。起動テンプレートを上書きする 12 の起動仕様には、優先順位が割り当てられた 4 つの異なるインスタンスタイプと、それぞれ別のアベイラビリティーゾーンに 3 つの異なるサブネットがあります。フリートのターゲット容量は 20 インスタンスで、デフォルトの購入オプションはスポットです。このため、フリートは、容量最適化優先順位付き割り当て戦略に基づいて 12 のスポットキャパシティープールのいずれかから 20 のスポットインスタンスを起動しようとします。これは、ベスト工フォート方式で優先順位を実装します。ですが、最初に容量を最適化します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized-prioritized"
    },
    "LaunchTemplateConfigs": [
      {
         "LaunchTemplateSpecification":{
            "LaunchTemplateName":"ec2-fleet-lt1",
            "Version":"$Latest"
         },
         "Overrides":[
            {
               "InstanceType":"c5.large",
```

```
            "SubnetId":"subnet-fae8c380",
            "Priority": 1.0
         },
         {
            "InstanceType":"c5.large",
            "SubnetId":"subnet-e7188bab",
            "Priority": 1.0
         },
         {
            "InstanceType":"c5.large",
            "SubnetId":"subnet-49e41922",
            "Priority": 1.0
         },
         {
            "InstanceType":"c5d.large",
            "SubnetId":"subnet-fae8c380",
            "Priority": 2.0
         },
         {
            "InstanceType":"c5d.large",
            "SubnetId":"subnet-e7188bab",
            "Priority": 2.0
         },
         {
            "InstanceType":"c5d.large",
            "SubnetId":"subnet-49e41922",
            "Priority": 2.0
         },
         {
            "InstanceType":"m5.large",
            "SubnetId":"subnet-fae8c380",
            "Priority": 3.0
         },
         {
            "InstanceType":"m5.large",
            "SubnetId":"subnet-e7188bab",
            "Priority": 3.0
         },
         {
            "InstanceType":"m5.large",
            "SubnetId":"subnet-49e41922",
            "Priority": 3.0
         },
         {
```

```
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-fae8c380",
                "Priority": 4.0
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-e7188bab",
                "Priority": 4.0
            },
            {
                "InstanceType":"m5d.large",
                "SubnetId":"subnet-49e41922",
                "Priority": 4.0
            }
        ]
      }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

EC2 フリートの設定戦略

EC2 フリート は、オンデマンドインスタンス と スポットインスタンス のグループです。EC2 フリートはキャパシティブロックインスタンスのグループでもあります。

オンデマンドインスタンスとスポットインスタンス

EC2 フリート は、フリートのリクエストで指定したターゲット容量を満たすために必要なインスタンス数の起動を試みます。フリートは、オンデマンドインスタンス のみ、または スポットインスタンス のみで構成するか、オンデマンドインスタンス と スポットインスタンス を組み合わせて構成できます。スポットインスタンス へのリクエストは、利用可能な容量があり、リクエストで指定した 1 時間あたりの上限料金がスポット料金を超えている場合に達成されます。また、スポットインスタンス が中断した場合、フリートはターゲット容量を維持しようとします。

フリートに対する 1 時間あたりの支払い上限容量を設定し、上限料金に達するまで EC2 フリート でインスタンスを起動することもできます。支払い上限料金に達すると、ターゲット容量に満たない場合でも、フリートはインスタンスの起動を停止します。

スポットキャパシティプール は、同じインスタンスタイプとアベイラビリティーゾーンを使用する、未使用の EC2 インスタンスのセットです。EC2 フリート を作成する場合に複数の起動条件を含めることができ、これにはインスタンスタイプ、アベイラビリティーゾーン、サブネット、上限価格があります。フリートは、リクエストに含まれる起動条件と、そのリクエストの設定に基づいてリクエストを処理するための、スポットキャパシティープールを選択します。スポットインスタンス は選択されたプールから取得されます。

EC2 フリート では、コアまたはインスタンスの数やメモリの量に基づいてアプリケーションにとって意味がある大量の EC2 容量をプロビジョニングできます。例えば、EC2 フリート が 200 インスタンス (そのうち 130 が オンデマンドインスタンス で、残りが スポットインスタンス) のターゲット容量を起動するように指定できます。

キャパシティブロックインスタンス

ML 用のキャパシティブロックを使用すると、短期間の機械学習 (ML) ワークロードをサポートするために、未来の日付で GPU インスタンスを予約できます。キャパシティブロックで実行されるインスタンスは、Amazon EC2 UltraClusters 内で自動的に近接して配置されます。キャパシティブロックの詳細については、「Capacity Blocks for ML」を参照してください。

ニーズを満たす EC2 フリート を作成するのに適切な設定戦略を使用してください。

コンテンツ

EC2 フリートの計画

EC2 フリートを計画するときは、次の操作を実行することをお勧めします。

- 目的のターゲット容量の同期または非同期のワンタイムリクエストを送信する EC2 フリート と、ターゲット容量の継続した維持を行うスポットフリートのどちらを作成するかを決定します。詳細については、「EC2 フリートのリクエストタイプ」を参照してください。
- アプリケーションの要件を満たすインスタンスタイプを決定します。
- EC2 フリート に スポットインスタンス を含める予定の場合、フリートを作成する前に「Spot Best Practices」を確認してください。フリートを計画するときにこれらのベストプラクティスを使用して、できるだけ低価格でインスタンスをプロビジョニングできるようにします。
- EC2 フリート のターゲット容量を決定します。インスタンスまたはカスタムユニットでターゲット容量を設定できます。詳細については、「EC2 フリートインスタンスの分量指定」を参照してください。
- EC2 フリート のターゲット容量のどの部分がオンデマンド容量およびスポット容量となるかを決定します。オンデマンド容量とスポット容量のいずれか、または両方に対して 0 を指定できます。
- インスタンス分量指定を使用している場合は、ユニット当りの料金を決定します。インスタンス時間当りの料金の計算は、インスタンス時間当たりの料金をそのインスタンスが表すユニット数 (または分量) で割って算出します インスタンス分量指定を使用する場合、ユニット当りのデフォルトの料金は 1 インスタンス時間当りの料金となります。
- フリートに支払う 1 時間あたりの上限料金を設定します。詳細については、「使用量の管理」を参照してください。
- EC2 フリートに対して可能なオプションを確認します。フリートパラメータの詳細については、「AWS CLI コマンドリファレンス」の「create-fleet」を参照してください。EC2 フリートの設定の例については、「EC2 フリートの設定例」を参照してください。

スポットインスタンスの配分戦略

起動設定によって、EC2 フリートがスポットインスタンスを起動できるすべてのスポットキャパシティプール (インスタンスタイプおよびアベイラビリティーゾーン) が決定されます。ただし、インスタンスを起動する際、EC2 フリートは指定された配分戦略を使用して、使用可能なすべてのプールから特定のプールを選択します。

> **Note**
>
> AMD SEV-SNP を有効にしてスポットインスタンスを起動するように設定すると、選択したインスタンスタイプのオンデマンド時間料金の 10% に相当する追加の時間単位使用料が請

求されます。配分戦略で価格を入力として使用する場合、EC2 フリートにはこの追加料金は含まれず、スポット料金のみが使用されます。

配分戦略

スポットインスタンスには次のいずれかの配分戦略を指定できます。

`price-capacity-optimized` (推奨)

EC2 フリートは、起動中のインスタンスの数に最適な容量の可用性を持つプールを識別します。つまり、短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。次に EC2 フリートは、これらのプールのうち最も価格の低いスポットインスタンスをリクエストします。

`price-capacity-optimized` 配分戦略は、ステートレスコンテナ化アプリケーション、マイクロサービス、ウェブアプリケーション、データおよび分析ジョブ、バッチ処理など、ほとんどのスポットワークロードに最適です。

`capacity-optimized`

EC2 フリートは、起動中のインスタンスの数に最適な容量の可用性を持つプールを識別します。つまり、短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。オプションで `capacity-optimized-prioritized` により、フリート内の各インスタンスタイプに優先順位を設定できます。EC2 フリートは最初に容量を最適化しますが、インスタンスタイプの優先順位をベストエフォートベースで尊重します。

スポットインスタンス では、価格は需要と供給の長期的な傾向に基づいて時間の経過とともに緩やかに変動しますが、容量はリアルタイムで変動します。`capacity-optimized` 戦略では、リアルタイムの容量データを調べ、可用性の最も高いプールを予測することで、そのプールから スポットインスタンス を自動的に起動します。この戦略は、作業の再開に関連する中断のコストが高くなる可能性のあるワークロード (長時間の継続的インテグレーション (CI)、画像とメディアのレンダリング、深層学習およびハイパフォーマンスコンピューティング (HPC) など) に対応します。中断の可能性を低くすることにより、`capacity-optimized` 戦略ではワークロードの全体的なコストを削減できます。

または、優先パラメータで `capacity-optimized-prioritized` 配分戦略を使用して、インスタンスタイプを優先順位の高い順から低い順へ指定できます。異なるインスタンスタイプに対し同じ優先順位を設定できます。EC2 フリートは最初に容量を最適化しますが、インスタンスタ

イプの優先順位をベストエフォートベースで決定します (例えば、優先順位を尊重しても、EC2 フリートの最適な容量をプロビジョニングする能力に大きな影響を与えない場合など) 。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。優先順位の使用は、フリートが起動テンプレートを使用する場合にのみサポートされます。capacity-optimized-prioritized の優先順位を設定するとき、オンデマンド AllocationStrategy が prioritized に設定されていると、同じ優先順位がオンデマンドインスタンスにも適用されますのでご注意ください。

diversified

スポットインスタンス は、すべてのスポットキャパシティープールに分散されます。

lowest-price

スポットインスタンスは、使用可能な容量を持つ最低価格のプールから取得されます。これはデフォルトの戦略です。ただし、price-capacity-optimized 配分戦略を指定してデフォルトを上書きすることをお勧めします。

最低価格のプールに使用可能な容量がない場合、スポットインスタンスは使用可能な容量のある 2 番目に低価格のプールから取得されます。

希望する容量を満たす前にプールの容量が不足した場合、EC2 フリートは 2 番目に低い価格のプールから容量を引き出し、引き続きリクエストを満たします。希望する容量を確実に満たすために、複数のプールからスポットインスタンスを受け取る場合があります。

この戦略では、インスタンスの価格のみが考慮され、容量の可用性は考慮されないため、中断率が高くなる可能性があります。

InstancePoolsToUseCount

ターゲットスポット容量を割り当てる先のスポットプールの数。配分戦略が lowest-price に設定されている場合にのみ有効です。EC2 フリートでは最低価格のスポットプールを選択し、指定した数のスポットプールにターゲットスポット容量を均等に割り当てます。

EC2 フリートは、指定したプール数内のスポットインスタンスを、ベストエフォート方式で利用しようとすることご注意ください。ターゲット容量を満たす前にプールにスポットキャパシティーの残量がなくなった場合、EC2 フリートは次に低い価格のプールの容量を利用してリクエストを満たします。ターゲット容量を確実に満たすために、スポットインスタンスが、指定した数を超えるプールから割り当てられることがあります。また、ほとんどのプールにスポット容量がない場合には、指定した数より少ないプールからターゲット容量のすべてが割り当てられることがあります。

適切な配分戦略の選択

適切なスポット割り当て戦略を選択することで、ユースケースに合わせてフリートを最適化できます。オンデマンドインスタンスのターゲット容量では、EC2 フリートはスポットインスタンス の配分戦略 (price-capacity-optimized、capacity-optimized diversified または lowest-price) を採用しながら、パブリックオンデマンド料金に基づいて、最低価格のインスタンスタイプを常に選択します。

最低価格と容量可用性のバランスをとる

最低価格のスポット容量プールと容量の可用性が最も高いスポットキャパシティプールとのトレードオフのバランスをとるには、price-capacity-optimized 配分戦略を使用することをお勧めします。この戦略では、プールの価格とプール内のスポットインスタンスの空き容量の両方に基づいて、どのプールからスポットインスタンスをリクエストするかを決定します。つまり、価格を考慮しながらも短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。

コンテナ化されたアプリケーション、マイクロサービス、ウェブアプリケーション、データおよび分析ジョブ、バッチ処理など、レジリエントでステートレスなワークロードをフリートが実行している場合は、最適なコスト削減とキャパシティアベイラビリティを実現する price-capacity-optimized 配分戦略を使用してください。

作業の再開に関連する中断に伴うコストが高くなる可能性があるワークロードをフリートで実行している場合は、中断があった場合にアプリケーションがそのポイントから再起動できるようにチェックポイントの設定を実装する必要があります。チェックポイントを使用すると、スポットインスタンスの中断率も低い最低価格のプールから容量が割り当てられるため、price-capacity-optimized 配分戦略がこれらのワークロードに適したものになります。

price-capacity-optimized 配分戦略を使用する設定例については、「例 11: price-capacity-optimized フリートでスポットインスタンスを起動する」を参照してください。

ワークロードの中断コストが高い場合

同様の価格のインスタンスタイプを使用するワークロードを実行する場合や、中断のコストが非常に高いため、中断のわずかな増加に比べてコスト削減が不十分な場合、オプションでこの capacity-optimized 戦略を使用できます。この戦略では、中断の可能性がより低く、最も可用性の高いスポットキャパシティプールから容量を割り当てることで、ワークロードの総コストを削減することができます。capacity-optimized 配分戦略を使用する設定例については、「例 9: 容量最適化フリートでスポットインスタンスを起動する」を参照してください。

中断の可能性を最小限に抑える必要があるが、特定のインスタンスタイプの優先順位が重要な場合は、capacity-optimized-prioritized の配分戦略を使用し、インスタンスタイプの順序を優先順位の高い順に表現することでプールの優先順位を設定することができます。設定の例については、「例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する」を参照してください。

優先順位の使用は、フリートが起動テンプレートを使用する場合にのみサポートされることに注意してください。capacity-optimized-prioritized の優先順位を設定する際に、オンデマンド AllocationStrategy が prioritized に設定されていると、同じ優先順位がオンデマンドインスタンスにも適用されるので注意してください。

ワークロードに時間的な柔軟性があり、キャパシティの可用性が問題にならない場合

フリートが小さい場合、または短時間の実行である場合、容量の可用性を考慮しながら、price-capacity-optimized を使用してコスト削減を最大化できます。

ワークロードに時間的な柔軟性があり、キャパシティの可用性が問題にならない場合は、オプションで lowest-price 配分戦略を使用してコスト削減を最大化できます。この lowest-price 配分戦略では、インスタンスの価格のみが考慮され、容量の可用性は考慮されないため、スポットインスタンス中断率が高くなる可能性があることをご注意ください。

フリートが大きい場合や長時間稼働している場合

フリートが大規模、または長期間実行される場合には、diversified 戦略を使用して複数のプールに スポットインスタンス を分散することで、フリートの可用性を改善できます。例えば、EC2 フリート の条件が 10 プールとして、ターゲット容量が 100 インスタンスとすると、フリートはプールごとに 10 個の スポットインスタンス を起動します。1 つのプールのスポット料金がこのプールの上限料金を超える場合、フリートの 10% のみに影響がおよびます。この戦略を使用すると、いずれのプールにおいても経時的にフリートが受けるスポット料金の上昇の影響を減少させます。diversified 戦略では、EC2 フリート は、オンデマンド価格 以上のスポット料金のいずれのプールにも スポットインスタンス を起動しません。

安価で分散型のフリートを作成するには、lowest-price 戦略を InstancePoolsToUseCount と組み合わせて使用します。例えば、ターゲットキャパシティが 10 のスポットインスタンスで、2 つのスポットキャパシティプールを (InstancePoolsToUseCount により) 指定した場合、EC2 フリートはスポットキャパシティを満たすために最も低い価格のプールを 2 つ利用します。

スポットインスタンス を配分するために、少数または多数のスポットキャパシティープールを選択して使用することができます。たとえば、バッチ処理を実行する場合は、少数のスポットキャパシ

ティープール (など) を指定することをお勧めします。これにより、キューのコンピューティング性能を常に確保しながら、コストを最大限削減できます。InstancePoolsToUseCount=2ウェブサービスを実行する場合は、スポットキャパシティプールが一時的に使用不可になった場合の影響を最小限に抑えるために、多数のスポットキャパシティプール (InstancePoolsToUseCount=10 など) を指定することをお勧めします。

EC2 フリートは、指定したプール数内のスポットインスタンスを、ベスト工フォート方式で利用しようとすることご注意ください。ターゲット容量を満たす前にプールにスポットキャパシティーの残量がなくなった場合、EC2 フリートは次に低い価格のプールの容量を利用してリク工ストを満たします。ターゲット容量を確実に満たすために、スポットインスタンスが、指定した数を超えるプールから割り当てられることがあります。また、ほとんどのプールにスポット容量がない場合には、指定した数より少ないプールからターゲット容量のすべてが割り当てられることがあります。

ターゲット容量の維持

スポット料金やスポットキャパシティプールで使用可能な容量の変動に伴って スポットインスタンス が終了すると、maintain 型の EC2 フリート によって代替の スポットインスタンス が起動されます。配分戦略によって、次のように置換先インスタンスを起動するプールが決まります。

- 割当戦略が price-capacity-optimized の場合、フリートは最もスポットインスタンスの容量が利用可能なプールで置換先インスタンスを起動します。また、価格も考慮し、容量利用率の高い価格の低いプールを特定します。
- 配分戦略が capacity-optimized の場合、フリートは、利用可能なスポットインスタンス容量が最大のプールで置換先インスタンスを起動します。
- 配分戦略が diversified である場合には、フリートは残りのプールに代替 スポットインスタンス を分散します。
- 配分戦略が lowest-price である場合、スポット群は、スポット料金が現在最低値のプールに代替インスタンスを起動します。
- 割り当て戦略が lowest-price と InstancePoolsToUseCount の組み合わせである場合、フリートは最低価格のスポット容量プールを選択し、指定した数のスポット容量プールにわたってスポットインスタンスを起動します。

EC2 フリートの属性ベースのインスタンスタイプの選択

EC2 フリートを作成するときは、フリートのオンデマンドインスタンスとスポットインスタンスを設定するため、1 つ以上のインスタンスタイプを指定する必要があります。インスタンスタイプを手動で指定する代わりに、インスタンスが持つ必要がある属性を指定でき、Amazon EC2 は、それ

らの属性を持つすべてのインスタンスタイプを識別します。これは 属性ベースのインスタンスタイプの選択 と呼ばれます。例えば、インスタンスに必要な vCPU の最小数および最大数を指定できます。EC2 フリートは、これらの vCPU 要件を満たす使用可能なインスタンスタイプを使用してインスタンスを起動します。

属性ベースのインスタンスタイプの選択は、コンテナやウェブフリートの実行、ビッグデータの処理、継続的インテグレーションおよびデプロイ (CI/CD) ツールの実装など、使用するインスタンスタイプについて柔軟に使用できるワークロードとフレームワークに最適です。

利点

属性ベースのインスタンスタイプを選択すると、次の利点があります。

- 適切なインスタンスタイプを簡単に使用 - 利用可能なインスタンスタイプの数が多いため、ワークロードに適したインスタンスタイプを見つけるには時間がかかることがあります。インスタンス属性を指定すると、インスタンスタイプにはワークロードに必要な属性が自動的に設定されます。
- 設定の簡素化 - EC2 フリートに複数のインスタンスタイプを手動で指定するには、インスタンスタイプごとに個別の起動テンプレートの上書きを作成する必要があります。ただし、属性ベースのインスタンスタイプを選択すると、複数のインスタンスタイプを提供するには、起動テンプレートまたは起動テンプレートの上書きでインスタンス属性を指定するだけで済みます。
- 新しいインスタンスタイプを自動的に使用 - インスタンスタイプではなくインスタンス属性を指定すると、フリートではリリース時に新しい世代のインスタンスタイプを使用できます。これにより、フリートの設定の将来の対応性も確保されます。
- インスタンスタイプの柔軟性 - インスタンスタイプではなくインスタンス属性を指定すると、EC2 フリートはスポットインスタンスを起動する際に幅広いインスタンスタイプから選択することができ、インスタンスタイプの柔軟性というスポットのベストプラクティスに準拠することができます。

トピック

属性ベースのインスタンスタイプ選択の仕組み

フリート設定で属性ベースのインスタンスタイプの選択を使用するには、インスタンスタイプのリストをインスタンスが必要とするインスタンス属性のリストに置き換えます。EC2 フリートは、指定されたインスタンス属性を持つ使用可能なインスタンスタイプでインスタンスを起動します。

トピック

- インスタンス属性のタイプ
- 属性ベースのインスタンスタイプの選択を設定する場所
- EC2 フリートがフリートをプロビジョニングするときに、属性ベースのインスタンスタイプ選択を使用する方法

インスタンス属性のタイプ

コンピューティング要件を表現するために指定できるインスタンス属性はいくつかあります。

- vCPU 数 – インスタンスあたりの vCPU の最小数と最大数。
- メモリ – インスタンスあたりのメモリの最小および最大 GiB。
- ローカルストレージ – EBS ボリュームとインスタンスストアボリュームのどちらをローカルストレージに使用するか。
- バースト可能なパフォーマンス – T4g、T3a、T3、および T2 タイプを含む T インスタンスファミリーを使用するかどうか。

各属性の説明およびデフォルト値については、「Amazon EC2 API リファレンス」の「InstanceRequirements」を参照してください。

属性ベースのインスタンスタイプの選択を設定する場所

コンソールと AWS CLI のどちらを使用するかによって、属性ベースのインスタンスタイプ選択のインスタンス属性を次のように指定できます。

コンソールでは、次のフリート設定コンポーネントでインスタンス属性を指定できます。

- 起動テンプレートでフリートリクエストの起動テンプレートを参照する

AWS CLI で、以下のフリート設定コンポーネントのいずれかまたはすべてでインスタンスの属性を指定することができます。

- 起動テンプレートでフリートリクエストの起動テンプレートを参照する
- 起動テンプレートの上書きで

  異なる AMI を使用するインスタンスを混在させたい場合は、複数の起動テンプレートの上書きでインスタンス属性を指定できます。例えば、異なるインスタンスタイプで x86 および ARM ベースのプロセッサを使用できます。
- 起動仕様で

EC2 フリートがフリートをプロビジョニングするときに、属性ベースのインスタンスタイプ選択を使用する方法

EC2 フリートは次の方法でフリートをプロビジョニングします。

- EC2 フリートは、指定された属性を持つインスタンスタイプを識別します。
- EC2 フリートは、料金保護を使用して、除外するインスタンスタイプを決定します。
- EC2 フリートは、インスタンスタイプが一致する AWS リージョンまたはアベイラビリティーゾーンに基づいて、インスタンスの起動を検討するキャパシティプールを決定します。
- EC2 フリートは、指定された割り当て戦略を適用して、インスタンスを起動するキャパシティプールを決定します。

  属性ベースのインスタンスタイプの選択では、フリートをプロビジョニングするキャパシティプールは選択されません。これが割り当て戦略のジョブです。指定された属性を持つインスタンスタイプが多数存在し、一部のインスタンスタイプにはコストがかかる場合があります。スポットとオンデマンドのデフォルトの割り当て戦略である `lowest-price` は、EC2 フリートが最も安価なキャパシティプールからインスタンスを起動することを保証します。

  割り当て戦略を指定すると、EC2 フリートは指定された割り当て戦略に従ってインスタンスを起動します。
  - スポットインスタンスでは、属性ベースのインスタンスタイプ選択により、`price-capacity-optimized`、`capacity-optimized`、`lowest-price` の配分戦略がサポートされます。
  - オンデマンドインスタンスでは、属性ベースのインスタンスタイプの選択は、`lowest-price` 配分戦略をサポートします。
- 指定されたインスタンス属性を持つインスタンスタイプの容量がない場合、インスタンスは起動できず、フリートはエラーを返します。

料金保護

料金保護は、EC2 フリートが指定した属性に適合した場合でも、高すぎると考えられるインスタンスタイプを使用できないようにする機能です。料金保護を使用するには、料金のしきい値を設定します。Amazon EC2 が属性を持つインスタンスタイプを選択すると、しきい値を超える料金が設定されたインスタンスタイプは除外されます。

Amazon EC2 が料金のしきい値を計算する方法は、次のとおりです。

- Amazon EC2 はまず、属性に一致するものから最低料金のインスタンスタイプを識別します。
- Amazon EC2 は、料金保護パラメータに指定した値 (パーセンテージで表される) を受け取り、識別されたインスタンスタイプの料金でそれを乗算します。その結果、料金しきい値として使用される料金になります。

オンデマンドインスタンスとスポットインスタンスには個別の料金しきい値があります。

属性ベースのインスタンスタイプを選択してフリートを作成すると、料金保護がデフォルトで有効になります。デフォルト値のままにすることも、独自の値を指定することもできます。

料金保護をオフにすることもできます。料金保護のしきい値を指定しない場合は、999999 などの高いパーセンテージ値を指定します。

トピック

- 最低料金のインスタンスタイプを特定する方法
- オンデマンドインスタンスの料金保護
- スポットインスタンスの料金保護
- 料金保護のしきい値を指定する

最低料金のインスタンスタイプを特定する方法

Amazon EC2 は、指定した属性に一致するものから最低料金のインスタンスタイプを特定することで、料金のしきい値に基づく料金を決定します。これは、次の方法で行います。

- まず、属性に一致する現行世代の C、M、または R インスタンスタイプを調べます。一致するものがある場合は、最低料金のインスタンスタイプを特定します。
- 一致するものがない場合は、属性に一致する現行世代のインスタンスタイプを調べます。一致するものがある場合は、最低料金のインスタンスタイプを特定します。

- 一致するものがない場合は、属性に一致する以前の世代のインスタンスタイプを調べ、最低料金のインスタンスタイプを特定します。

オンデマンドインスタンスの料金保護

オンデマンドインスタンスタイプの料金保護のしきい値は、特定された最低料金のオンデマンドインスタンスタイプ (OnDemandMaxPricePercentageOverLowestPrice) よりも高いパーセンテージで計算されます。支払い可能なパーセンテージを高く指定します。このパラメータを指定しない場合は、デフォルト値の 20 を使用して、識別された料金よりも 20% 高い料金保護しきい値が計算されます。

例えば、特定されたオンデマンドインスタンスの料金が 0.4271 で、25 を指定した場合、料金のしきい値は 0.4271 より 25% 高くなります。これは、次のように計算されます: 0.4271 * 1.25 = 0.533875。計算された料金は、オンデマンドインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は 0.533875 を超えるコストがかかるオンデマンドインスタンスタイプを除外します。

スポットインスタンスの料金保護

デフォルトでは、Amazon EC2 は最適なスポットインスタンス料金保護を自動的に適用し、幅広いインスタンスタイプから一貫して選択します。料金保護を手動で設定することもできます。ただし、Amazon EC2 に任せることで、スポット容量が満たされる可能性を高めることができます。

料金保護は、次のいずれかのオプションを使用して手動で指定できます。料金保護を手動で設定する場合は、最初のオプションを使用することをお勧めします。

- 特定された最低料金のオンデマンドインスタンスタイプ (**MaxSpotPriceAsPercentageOfOptimalOnDemandPrice**) のパーセンテージ

  例えば、特定されたオンデマンドインスタンスタイプの料金が 0.4271 で、60 を指定した場合、料金のしきい値は 0.4271 の 60% になります。これは、次のように計算されます: 0.4271 * 0.60 = 0.25626。計算された料金は、スポットインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は 0.25626 を超えるコストがかかるスポットインスタンスタイプを除外します。
- 特定された最低料金のスポットインスタンスタイプ (**SpotMaxPricePercentageOverLowestPrice**) よりも高いパーセンテージ

  例えば、特定されたスポットインスタンスタイプの料金が 0.1808 で、25 を指定した場合、料金のしきい値は 0.1808 より 25% 高くなります。これは、次のように計算されます: 0.1808 *

1.25 = 0.226。計算された料金は、スポットインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は 0.266 を超えるコストがかかるスポットインスタンスタイプを除外します。スポット料金が変動する可能性があり、料金保護のしきい値も変動する可能性があるため、このパラメータの使用はお勧めしません。

料金保護のしきい値を指定する

料金保護のしきい値を指定するには

EC2 フリートを作成するときに、属性ベースのインスタンスタイプを選択するようにフリートを設定してから、次の手順を実行します。

- オンデマンドインスタンスの料金保護のしきい値を指定するには、JSON 設定ファイルの InstanceRequirements 構造の OnDemandMaxPricePercentageOverLowestPrice で、料金保護のしきい値をパーセンテージ (%) で入力します。
- スポットインスタンスの料金保護のしきい値を指定するには、JSON 設定ファイルの InstanceRequirements 構造で、次のいずれかのパラメータを指定します。
  - MaxSpotPriceAsPercentageOfOptimalOnDemandPrice で、料金保護のしきい値をパーセンテージ (%) で入力します。
  - SpotMaxPricePercentageOverLowestPrice で、料金保護のしきい値をパーセンテージ (%) で入力します。

フリートの作成の詳細については、「属性ベースのインスタンスタイプを選択した EC2 フリートを作成する」を参照してください。

> **Note**
>
> EC2 フリートを作成するときに、TargetCapacityUnitType を vcpu または memory-mib に設定すると、インスタンスごとの料金ではなく、vCPU ごとまたはメモリごとの料金に基づいて料金保護のしきい値が適用されます。

考慮事項

- EC2 フリートのインスタンスタイプまたはインスタンス属性のいずれかを指定できますが、両方を同時に指定することはできません。

CLI を使用する場合、起動テンプレートの上書きによって起動テンプレートが上書きされます。例えば、起動テンプレートにインスタンスタイプが含まれ、起動テンプレートの上書きにインスタンス属性が含まれている場合、インスタンス属性によって識別されるインスタンスは、起動テンプレートのインスタンスタイプを上書きします。

- CLI を使用していて、インスタンス属性の上書きを指定する場合、重みまたは優先順位も指定できません。
- リクエスト設定では、最大 4 つの InstanceRequirements 構造を指定できます。

属性ベースのインスタンスタイプを選択した EC2 フリートを作成する

AWS CLI を使用して、属性ベースのインスタンスタイプの選択を使用するようにフリートを設定できます。

属性ベースのインスタンスタイプを選択した EC2 フリートを作成するには (AWS CLI)

create-fleet (AWS CLI) コマンドを使用して、EC2 フリートを作成します。JSON ファイルでフリート設定を指定します。

```
aws ec2 create-fleet \
    --region us-east-1 \
    --cli-input-json file://file_name.json
```

file_name.json ファイルの例

次の例には、属性ベースのインスタンスタイプ選択を使用するように EC2 フリートを設定するパラメータが含まれており、その後にテキストによる説明が続きます。

```
{
 "SpotOptions": {
  "AllocationStrategy": "price-capacity-optimized"
 },
 "LaunchTemplateConfigs": [{
  "LaunchTemplateSpecification": {
   "LaunchTemplateName": "my-launch-template",
   "Version": "1"
  },
  "Overrides": [{
   "InstanceRequirements": {
    "VCpuCount": {
     "Min": 2
```

```
    },
    "MemoryMiB": {
     "Min": 4
    }
   }
  }]
 }],
 "TargetCapacitySpecification": {
  "TotalTargetCapacity": 20,
  "DefaultTargetCapacityType": "spot"
 },
 "Type": "instant"
}
```

属性に基づくインスタンスタイプ選択のための属性は、InstanceRequirements 構造で指定されます。この例では、2 つのタグが指定されています。

- VCpuCount — 最低 2 つの vCPUs が指定されています。最大値は指定されていないため、上限はありません。
- MemoryMiB — 4 MiB 以上のメモリが指定されています。最大値は指定されていないため、上限はありません。

2 つ以上の vCPUs と 4 MiB 以上のメモリを持つすべてのインスタンスタイプが識別されます。ただし、EC2 フリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

指定できるすべての属性のリストと説明については、「Amazon EC2 API リファレンス」の「インスタンス要件」を参照してください。

> **Note**
>
> InstanceRequirements がフリート設定に含まれる場合、InstanceType と WeightedCapacity は除外しなければならず、インスタンス属性と同時にフリート設定を決定することはできません。

JSON には次のフリート設定も含まれています。

- "AllocationStrategy": "*price-capacity-optimized*" — フリート内のスポットインスタンスの割り当て戦略。

- "LaunchTemplateName": "*my-launch-template*", "Version": "*1*" — 起動テンプレートにはいくつかのインスタンス設定情報が含まれていますが、インスタンスタイプが指定されている場合は、InstanceRequirements で指定されている属性によってオーバーライドされます。
- "TotalTargetCapacity": *20* – ターゲット容量は 20 個のインスタンスです。
- "DefaultTargetCapacityType": "*spot*" — デフォルトの容量はスポットインスタンスです。
- "Type": "*instant*" — フリートのリクエストタイプは instant です。

有効な設定と無効な設定の例

AWS CLI を使用して EC2 フリートを作成する場合は、フリートの設定が有効であることを確認する必要があります。次の例は、有効な設定と無効な設定を示しています。

次のものが含まれている場合、設定は無効と見なされます。

- InstanceRequirements および InstanceType を持つ 1 つの Overrides 構造
- 一つは InstanceRequirements、もう一つは InstanceType を持つ 2 つの Overrides 構造
- 同じ LaunchTemplateSpecification 内で属性値が重複している 2 つの InstanceRequirements 構造

設定例

- 有効な設定: 上書きを含む単一の起動テンプレート
- 有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート
- 有効な設定: 2 つの起動テンプレート、それぞれに上書きがある
- 有効な設定: InstanceRequirements のみ指定され、重複する属性値なし
- 設定が無効です: Overrides が InstanceRequirements および InstanceType を含んでいる
- 設定が無効です: 2 つの Overrides に InstanceRequirements および InstanceType が含まれている
- 設定が無効です: 重複する属性値

有効な設定: 上書きを含む単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 1 つの Overrides が含まれています。以下に、構成例をテキストで説明します。

```
{
        "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "My-launch-template",
                "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 2,
                            "Max": 8
                        },
                        "MemoryMib": {
                            "Min": 0,
                            "Max": 10240
                        },
                        "MemoryGiBPerVCpu": {
                            "Max": 10000
                        },
                        "RequireHibernateSupport": true
                    }
                }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 5000,
        "DefaultTargetCapacityType": "spot",
        "TargetCapacityUnitType": "vcpu"
        }
    }
}
```

InstanceRequirements

属性ベースのインスタンス選択を使用するには、フリート設定に InstanceRequirements 構造を含め、フリート内のインスタンスに必要な属性を指定する必要があります。

前の例に、以下のインスタンス属性が指定されています。

- VCpuCount - インスタンスタイプには、2 個以上、最大 8 個の vCPU が必要です。

- MemoryMiB - インスタンスタイプには最大 10,240 MiB のメモリが必要です。最小数が 0 の場合、最小制限がないことを示します。
- MemoryGiBPerVCpu - インスタンスタイプには、vCPU あたり最大 10,000 GiB のメモリが必要です。Min パラメータはオプションです。省略すると、最小制限がないことを示します。

TargetCapacityUnitType

TargetCapacityUnitType パラメータは、ターゲット容量の単位を指定します。この例では、ターゲット容量は 5000 であり、ターゲット容量ユニットタイプは vcpu で、これを組み合わせて 5,000 vCPU の希望するターゲット容量を指定します。EC2 フリートは、フリート内の vCPU の総数が 5,000 vCPU になるように、十分なインスタンスを起動します。

有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 2 つの Overrides が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
```

```
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

有効な設定: 2 つの起動テンプレート、それぞれに上書きがある

以下の設定は有効です。これには 2 つの起動テンプレートが含まれ、各起動テンプレートには 1 つの InstanceRequirements 構造を含む Overrides 構造が 1 つ含まれています。この設定は、同じフリートで arm と x86 のアーキテクチャをサポートする場合に有効です。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "armLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
```

```
                {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "x86LaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
         "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

有効な設定: **InstanceRequirements** のみ指定され、重複する属性値なし

以下の設定は有効です。2 つの LaunchTemplateSpecification 構造が含まれ、各構造にそれぞれ起動テンプレートと、Overrides 構造を含む InstanceRequirements 構造が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
```

```
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

設定が無効です: **Overrides** が **InstanceRequirements** および **InstanceType** を含んでいる

以下は設定が有効ではありません。Overrides 構造体には InstanceRequirements および InstanceType が両方含まれています。Overrides では、InstanceRequirements または InstanceType のどちらかを指定できますが、両方は指定できません。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
        "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

設定が無効です: 2 つの **Overrides** に **InstanceRequirements** および **InstanceType** が含まれている

以下は設定が有効ではありません。Overrides 構造に InstanceRequirements および InstanceType が両方含まれています。異なる Overrides 構造にある場

合、InstanceRequirements または InstanceType を指定できますが、両方を指定することはできません。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
         "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
    }
}
```

設定が無効です: 重複する属性値

以下は設定が有効ではありません。2 つの InstanceRequirements 構造がそれぞれ "VCpuCount": {"Min": 0, "Max": 2} を含んでいます。これらの属性の値が重複するため、容量プールが重複します。

```
{
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    },
                    {
                      "InstanceRequirements": {
                          "VCpuCount": {
                              "Min": 0,
                              "Max": 2
                          },
                          "MemoryMiB": {
                              "Min": 0
                          }
                      }
                  }
                }
              ]
            }
        ],
         "TargetCapacitySpecification": {
            "TotalTargetCapacity": 1,
            "DefaultTargetCapacityType": "spot"
        }
```

```
    }
}
```

指定された属性でインスタンスタイプをプレビューする

get-instance-types-from-instance-requirements AWS CLI コマンドを使用して、指定した属性に一致するインスタンスタイプをプレビューします。これは、インスタンスを起動せずにリクエスト設定で指定する属性を調べる場合に特に便利です。このコマンドでは、使用可能な容量は考慮されません。

AWS CLI を使用して属性を指定してインスタンスタイプのリストをプレビューするには

1. (オプション) 指定可能なすべての属性を生成するには、get-instance-types-from-instance-requirements コマンドと --generate-cli-skeleton パラメータを使用します。オプションで、input > *attributes.json* を使用して出力を保存用ファイルに送ることができます。

```
aws ec2 get-instance-types-from-instance-requirements \
    --region us-east-1 \
    --generate-cli-skeleton input > attributes.json
```

正常な出力

```
{
    "DryRun": true,
    "ArchitectureTypes": [
        "i386"
    ],
    "VirtualizationTypes": [
        "hvm"
    ],
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 0,
            "Max": 0
        },
        "MemoryMiB": {
            "Min": 0,
            "Max": 0
        },
        "CpuManufacturers": [
            "intel"
        ],
        "MemoryGiBPerVCpu": {
```

```
            "Min": 0.0,
            "Max": 0.0
        },
        "ExcludedInstanceTypes": [
            ""
        ],
        "InstanceGenerations": [
            "current"
        ],
        "SpotMaxPricePercentageOverLowestPrice": 0,
        "OnDemandMaxPricePercentageOverLowestPrice": 0,
        "BareMetal": "included",
        "BurstablePerformance": "included",
        "RequireHibernateSupport": true,
        "NetworkInterfaceCount": {
            "Min": 0,
            "Max": 0
        },
        "LocalStorage": "included",
        "LocalStorageTypes": [
            "hdd"
        ],
        "TotalLocalStorageGB": {
            "Min": 0.0,
            "Max": 0.0
        },
        "BaselineEbsBandwidthMbps": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorTypes": [
            "gpu"
        ],
        "AcceleratorCount": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorManufacturers": [
            "nvidia"
        ],
        "AcceleratorNames": [
            "a100"
        ],
        "AcceleratorTotalMemoryMiB": {
```

```
            "Min": 0,
            "Max": 0
        },
        "NetworkBandwidthGbps": {
            "Min": 0.0,
            "Max": 0.0
        },
        "AllowedInstanceTypes": [
            ""
        ]
    },
    "MaxResults": 0,
    "NextToken": ""
}
```

2. 前のステップの出力を使用して JSON 設定ファイルを作成し、次のように設定します。

> **Note**
>
> ArchitectureTypes、VirtualizationTypes、VCpuCount、および MemoryMiB の値を指定する必要があります。その他の属性は省略できます。省略すると、デフォルト値が使用されます。
> 各属性およびそのデフォルト値の説明については、「Amazon EC2 コマンドラインリファレンス」の「get-instance-types-from-instance-requirements」を参照してください。

a. ArchitectureTypes に、1 つ以上のタイプのプロセッサアーキテクチャを指定します。

b. VirtualizationTypes に、1 つまたは複数のタイプの仮想化を指定します。

c. VCpuCount に、vCPU の最小数と最大数を指定します。最小制限を指定しない場合は、Min で、0 を指定します。最大制限を指定しない場合は、Max パラメータを省略します。

d. MemoryMiB に、最小値と最大値を MiB 単位で指定します。最小制限を指定しない場合は、Min で、0 を指定します。最大制限を指定しない場合は、Max パラメータを省略します。

e. オプションで、他の属性を 1 つ以上指定して、返されるインスタンスタイプのリストをさらに制約できます。

3. JSON ファイルで指定した属性を持つインスタンスタイプをプレビューするには、get-instance-types-from-instance-requirements コマンドを入力し、--cli-input-json パラメータを使用して、JSON ファイルの名前とパスを指定します。オプションで、出力が表形式で表示されるようにフォーマットできます。

```
aws ec2 get-instance-types-from-instance-requirements \
    --cli-input-json file://attributes.json \
    --output table
```

例: *attributes.json* ファイル

この例では、JSON ファイルに必須属性が含まれています。それらは、ArchitectureTypes、VirtualizationTypes、VCpuCount、および MemoryMiB です。さらに、オプションで InstanceGenerations 属性も含まれます。MemoryMiB では、Max の値を省略し、制限がないことを示すことができることを注意してください。

```
{

    "ArchitectureTypes": [
        "x86_64"
    ],
    "VirtualizationTypes": [
        "hvm"
    ],
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 4,
            "Max": 6
        },
        "MemoryMiB": {
            "Min": 2048
        },
        "InstanceGenerations": [
            "current"
        ]
    }
}
```

出力例

```
----------------------------------------
```

```
|GetInstanceTypesFromInstanceRequirements|
+----------------------------------------+
||             InstanceTypes            ||
|+--------------------------------------+|
||             InstanceType             ||
|+--------------------------------------+|
||  c4.xlarge                           ||
||  c5.xlarge                           ||
||  c5a.xlarge                          ||
||  c5ad.xlarge                         ||
||  c5d.xlarge                          ||
||  c5n.xlarge                          ||
||  d2.xlarge                           ||
...
```

4. ニーズに合ったインスタンスタイプを特定したら、フリートリクエストを設定するときにそれらを使用できるように、使用したインスタンスの属性をメモしておきます。

オンデマンドバックアップのための EC2 フリートの設定

重大なニュースイベントや試合の開始時にニュースウェブサイトをスケールする必要があるなど、予測できない緊急のスケーリングニーズが生じた場合、希望するオプションに十分な容量がないときは、オンデマンドインスタンスの代替インスタンスタイプを指定することをお勧めします。たとえば、c5.2xlarge オンデマンドインスタンス を希望するが使用可能な容量が十分でない場合、ピーク負荷時に c4.2xlarge インスタンスを使用できます。この場合、EC2 フリート は c5.2xlarge インスタンスを使用してすべてのターゲット容量を満たそうとしますが、容量が十分でない場合、c4.2xlarge インスタンスを自動的に起動してターゲット容量を満たします。

トピック

- オンデマンド容量に基づくインスタンスタイプの優先順位付け
- オンデマンドインスタンスのためのキャパシティ予約の使用

オンデマンド容量に基づくインスタンスタイプの優先順位付け

EC2 フリート でオンデマンド容量を達成する場合、デフォルトで最低価格のインスタンスタイプが最初に起動されます。AllocationStrategy を prioritized に設定すると、EC2 フリート は優先度に従って、オンデマンド容量を達成するために最初に使用するインスタンスタイプを決定します。優先度は起動テンプレートの上書きに割り当てられ、最も高い優先度が最初に起動されます。

例: インスタンスタイプの優先付け

例えば、3 つの起動テンプレートの上書きに、それぞれ異なるインスタンスタイプを設定したとします。

インスタンスタイプのオンデマンド料金は、幅があります。以下は、この例で使用しているインスタンスタイプで、料金の安いものから順に並んでいます。

- m4.large — 最も安価
- m5.large
- m5a.large

優先度を使って順番を決めない場合、フリートは、最も価格が低いインスタンスタイプから始めてオンデマンドの容量を満たします。

ただし、最初に使用する m5.large リザーブドインスタンスが未使用である場合、次のように、インスタンスタイプが優先度順に使われるように、起動テンプレートの、上書きの優先度を設定できます。

- m5.large – 優先度 1
- m4.large – 優先度 2
- m5a.large – 優先度 3

オンデマンドインスタンスのためのキャパシティ予約の使用

オンデマンド容量予約を使用すると、特定のアベイラビリティーゾーンで任意の所要時間だけ、オンデマンドインスタンスのコンピューティング性能を予約できます。オンデマンドインスタンスを起動するときに、容量予約を最初に使用するように EC2 フリートを設定できます。

容量予約は、open または targeted のいずれかで設定されます。EC2 フリートは、オンデマンドインスタンスを open または targeted キャパシティ予約で設定できます (次のとおり) :

- キャパシティ予約が open の場合、一致する属性を持つオンデマンドインスタンスは、リザーブドキャパシティで自動的に実行されます。
- キャパシティ予約 が targeted の場合、オンデマンドインスタンスはそれがリザーブドキャパシティで実行されるように具体的に設定する必要があります。これは、特定のキャパシティ予約を使い切ったり、特定のキャパシティ予約をいつ使用するかを制御する場合に便利です。

また、targeted を使用した場合、EC2 フリートのキャパシティ予約では、ターゲットのオンデマンドキャパシティを満たすのに十分なキャパシティ予約が必要です。そうしないと、起動に失敗します。起動が失敗するのを避けるには、targeted キャパシティ予約をリソースグループに加え、リソースグループをターゲットにします。リソースグループは十分なキャパシティ予約を持っている必要はありません。ターゲットのオンデマンドキャパシティが満たされる前にキャパシティ予約がなくなった場合、フリートは残りのターゲットキャパシティを通常のオンデマンドキャパシティに起動できます。

EC2 フリートでキャパシティ予約を使用するには

1. フリートをタイプ instant として設定する: その他のタイプのフリートには、キャパシティーの予約を使用することはできません。
2. キャパシティ予約の使用戦略として、use-capacity-reservations-first を設定する。
3. 起動テンプレートで、キャパシティ予約 には、オープン または グループ別のターゲット のいずれかを選択します。グループ別のターゲット を選択した場合、キャパシティ予約リソースグループ ID を指定します。

フリートがオンデマンド容量を満たそうとしたときに、複数のインスタンスプールで一致するキャパシティ予約が未使用であることがわかった場合、オンデマンド割り当て戦略に基づいてオンデマンドインスタンスを起動するプールを決定します (lowest-price または prioritized)。

オンデマンド容量を満たすために、キャパシティーの予約を使用するようフリートを設定する方法の例については、EC2 フリートの設定例 (特に例 5 から 7) を参照してください。

キャパシティ予約の設定の詳細については、On-Demand Capacity Reservations と オンデマンドキャパシティ予約のよくある質問 を参照してください。

容量の再調整

Amazon EC2 が再調整に関する推奨を発して、スポットインスタンスが中断リスクが高まっていることを通知したとき、代替スポットインスタンスを起動するように EC2 フリートを設定できます。容量の再調整は、実行中のインスタンスが Amazon EC2 により中断される前に、新しいスポットインスタンスでフリートを事前に拡張することにより、ワークロードの可用性を維持するのに役立ちます。詳細については、「EC2 インスタンスの再調整に関する推奨事項」を参照してください。

代替スポットインスタンスを起動するように EC2 フリート を設定するには、フリートの作成 (AWS CLI) コマンドおよび MaintenanceStrategies 構造内の関連するパラメータを使用します。詳細については、「起動設定の例」を参照してください。

制限事項

- 容量の再調整は、タイプ maintain のフリートでのみ使用可能です。
- フリートが実行されているときは、容量の再調整設定を変更できません。容量の再調整設定を変更するには、フリートを削除し、新しいフリートを作成する必要があります。

設定オプション

EC2 フリートの ReplacementStrategy では、次の 2 つの値がサポートされます。

`launch-before-terminate`

Amazon EC2 フリートは、新しい置換先スポットインスタンスが起動された後に、再調整通知を受信するスポットインスタンスを終了します。launch-before-terminate を指定する場合は、termination-delay の値も指定する必要があります。新しい置換先インスタンスが起動された後、Amazon EC2 フリートは termination-delay の時間だけ待って、古いインスタンスを終了させます。termination-delay では、最小値は 120 秒 (2 分)、最大値は 7200 秒 (2 時間) です。

launch-before-terminate は、インスタンスのシャットダウン手順が完了するまでの時間が予測できる場合にのみ使用することをお勧めします。これにより、シャットダウン手順が完了した後にのみ、古いインスタンスが確実に終了されます。Amazon EC2 は、termination-delay の前に 2 分間の警告を行い、その後古いインスタンスを中断する可能性があることに注意してください。

また、lowest-price 配分戦略を launch-before-terminate と組み合わせて使用することは、中断のリスクが高い代替スポットインスタンスを持つことになるため、強く推奨されません。

`launch`

Amazon EC2 フリートは、既存のスポットインスタンスに対して再調整通知が送信されると、置換先スポットインスタンスを起動します。Amazon EC2 フリートは、再調整通知を受け取るインスタンスを終了しません。古いインスタンスを終了することも、実行したままにすることもできます。実行中は、すべてのインスタンスに対して課金されます。

考慮事項

容量の再調整用に EC2 フリート を設定する場合は、次の点を考慮してください。

リクエストでは可能な限り多くのスポットキャパシティープールを指定する

複数のインスタンスタイプとアベイラビリティーゾーンを使用するように EC2 フリート を設定します。これにより、さまざまなスポットキャパシティープールでスポットインスタンスを起動するための柔軟性が得られます。詳細については、「インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する」を参照してください。

代替スポットインスタンスが中断されるリスクの増大を回避する

`lowest-price` 割り当て戦略を使用している場合、代替スポットインスタンスが中断するリスクが高くなることがあります。これは、置換先スポットインスタンスが起動後すぐに中断される可能性があっても、Amazon EC2 は、その時点で利用可能な容量を持つ最低価格のプールでインスタンスを常に起動するためです。中断のリスクが高くなるのを避けるため、`lowest-price` アロケーションストラテジー、代わりに推奨する `capacity-optimized` または `capacity-optimized-prioritized` 配分戦略。これらの戦略により、代替スポットインスタンスが最適なスポットキャパシティプールで起動されるため、近い将来中断される可能性が低くなります。詳細については、「価格と容量を最適化する配分戦略を使用する」を参照してください。

Amazon EC2 は、可用性が同じかそれ以上の場合にのみ、新しいインスタンスを起動します

容量の再調整の目的の 1 つは、スポットインスタンスの可用性を改善することです。既存のスポットインスタンスが再調整のレコメンデーションを受け取った場合、Amazon EC2 は、新しいインスタンスが既存のインスタンスと同等かそれ以上の可用性を提供する場合にのみ新しいインスタンスを起動します。新しいインスタンスの中断のリスクが既存のインスタンスよりもひどい場合、Amazon EC2 は新しいインスタンスを起動しません。ただし、Amazon EC2 は引き続きスポットキャパシティプールを評価し、可用性が向上したら新しいインスタンスを起動します。

Amazon EC2 が新しいインスタンスをプロアクティブに起動しないと、既存のインスタンスが中断する可能性があります。これが発生する場合、Amazon EC2 は、新しいインスタンスの中断リスクが高いかどうかに関らず、新しいインスタンスの起動を試みます。

キャパシティーの再調整は、スポットインスタンスの中断率を増加させるものではありません

キャパシティの再調整を有効にしても、スポットインスタンスの中断率 (Amazon EC2 がキャパシティを取り戻す必要があるときに再利用されるスポットインスタンスの数) は増加しません。ただし、インスタンスに中断のリスクがあることを容量の再調整が検出した場合、Amazon EC2 Auto Scaling は直ちに新しいインスタンスの起動を試みます。その結果、リスクのあるインスタンスが中断された後に Amazon EC2 が新しいインスタンスを起動するのを待つ場合よりも多くのインスタンスが置き換えられる可能性があります。

キャパシティーの再調整が有効になっているインスタンスをさらに置き換える可能性がありますが、インスタンスが中断される前にアクションを実行するための時間をより長く確保できるた

め、事後対応ではなくプロアクティブに対応できるというメリットがあります。スポットインスタンスの中断通知では、通常、インスタンスを正常にシャットダウンするための猶予期間が最大 2 分しかありません。キャパシティーの再調整で新しいインスタンスを事前に起動することで、既存のプロセスがリスクのあるインスタンスで完了する可能性が高くなり、インスタンスのシャットダウン手順を開始して、リスクのあるインスタンスで新しい作業がスケジュールされないようにできます。新しく起動したインスタンスの準備を開始して、アプリケーションを引き継ぐこともできます。キャパシティーの再調整のプロアクティブな置き換えにより、正常な継続性の恩恵を受けることができます。

キャパシティーの再調整を使用するリスクとメリットを示す理論的な例として、次のシナリオを検討してください。

- 午後 2 時 – インスタンス A の再調整の推奨が受信され、Amazon EC2 は直ちに置換先インスタンス B の起動の試行を開始するため、シャットダウン手順を開始する時間を確保できます。*
- 午後 2 時 30 分 – インスタンス B の再調整の推奨が受信され、インスタンス C に置き換えられるため、シャットダウン手順を開始する時間を確保できます。*
- 午後 2 時 32 分 – キャパシティーの再調整が有効になっておらず、インスタンス A のスポットインスタンスの中断通知が午後 2 時 32 分に受信されていたとすれば、アクションを実行するための猶予期間は最大でも 2 分だけでしたが、インスタンス A はこの時間まで稼働していたことでしょう。

* `launch-before-terminate` が指定されている場合、Amazon EC2 は、置換先インスタンスがオンラインになった後、リスクのあるインスタンスを終了します。

Amazon EC2 フリート は、満たされた容量がターゲット容量の 2 倍になるまで、新しい置換先スポットインスタンスを起動できます

EC2 フリートが容量の再調整用に設定されている場合、フリートは、再調整に関する推奨事項を受け取るすべてのスポットインスタンスに対して、新しい代替スポットインスタンスを起動しようとします。スポットインスタンスが再調整勧告を受け取った後は、満たされた容量の一部としてカウントされなくなります。交換戦略に応じて、Amazon EC2 は事前設定された終了遅延の後にインスタンスを終了するか、インスタンスを実行のままにします。これにより、インスタンスで 再調整アクション を実行できるようになります。

フリートがターゲットキャパシティの 2 倍に達すると、代替インスタンス自体が再調整に関する推奨事項を受け取った場合でも、新しい代替インスタンスの起動を停止します。

例えば、100 個のスポットインスタンスのターゲットキャパシティを持つ EC2 フリートを作成したとします。すべてのスポットインスタンスは、再調整に関するレコメンデーションを受け取り

ます。これにより、Amazon EC2 は 100 個の置換先スポットインスタンスを起動します。これにより、満たされたスポットインスタンスの数が 200 になり、ターゲットキャパシティの 2 倍になります。一部の代替インスタンスは再調整に関する推奨事項を受け取りますが、フリートがターゲット容量の 2 倍を超えることができないため、代替インスタンスはそれ以上起動されません。

インスタンスの実行中は、すべてのインスタンスに対して課金されることに注意してください。

再調整に関する推奨事項を受け取るスポットインスタンスを終了するため、EC2 フリートを設定することをお勧めします

EC2 フリートに容量の再調整を設定する場合は、インスタンスのシャットダウン手順の完了までの時間が予測できる場合に限り、適切な終了遅延を持つ `launch-before-terminate` を選択することをお勧めします。これにより、シャットダウン手順が完了した後にのみ、古いインスタンスが確実に終了されます。

再調整のために推奨されるインスタンスを終了する場合は、フリートのスポットインスタンスが受信する再調整レコメンデーションシグナルをモニタリングすることをお勧めします。シグナルをモニタリングすることで、Amazon EC2 が中断する前に、影響を受けるインスタンスで 再調整のアクション をすばやく実行し、手動で終了できます。インスタンスを終了しない場合、インスタンスの実行中、課金が継続します。Amazon EC2 は、再調整に関する推奨を受け取るインスタンスを自動的に終了しません。

Amazon EventBridge またはインスタンスメタデータを使用して通知を設定できます。詳細については、「再調整に関する推奨事項シグナルのモニタリング」を参照してください。

EC2 フリート は、スケールインまたはスケールアウト中に満たされた容量を計算する際、再調整に関する推奨事項を受け取るインスタンスはカウントされない

容量の再調整を行うように EC2 フリート が設定されていて、ターゲット容量をスケールインまたはスケールアウトするように変更した場合、次のように、フリートは、再調整の対象としてマークされたインスタンスを、満たされた容量の一部としてカウントしません。

- スケールイン – 希望するターゲット容量を減らすと、Amazon EC2 は目的の容量に達するまで、再調整の対象としてマークされていないインスタンスを終了します。再調整の対象としてマークされたインスタンスは、満たされた容量にはカウントされません。

  例えば、EC2 フリートを 100 個のスポットインスタンスのターゲット容量で作成します。10 個のインスタンスは再調整に関する推奨を受け取るため、Amazon EC2 は 10 個の新しい置換先インスタンスを起動し、その結果、110 個のインスタンスの容量が満たされます。その後、ターゲット容量を 50 個に減らしますが (スケールイン)、再調整の対象としてマークされた 10 個のインスタンスは Amazon EC2 によって終了されないため、満たされた容量は実際には 60

インスタンスになります。このようなインスタンスは手動で終了する必要があります。または、実行したままにしておくことができます。

- スケールアウト – 目的のターゲット容量を増やすと、目的の容量に達するまで Amazon EC2 は新しいインスタンスを起動します。再調整の対象としてマークされたインスタンスは、満たされた容量にはカウントされません。

  例えば、EC2 フリートを 100 個のスポットインスタンスのターゲット容量で作成します。10 個のインスタンスは再調整に関する推奨を受け取るため、フリートは 10 個の新しい代替インスタンスを起動し、その結果、110 個のインスタンスの容量が満たされます。その後、ターゲット容量を 200 個に増やし (スケールアウトし) ますが、再調整の対象としてマークされた 10 個のインスタンスは、フリートによってターゲット容量の一部としてカウントされないため、実際には 210 個のインスタンスになります。このようなインスタンスは手動で終了する必要があります。または、実行したままにしておくことができます。

上限価格の優先

各 EC2 フリート には、グローバルな上限料金を含めるか、デフォルト (オンデマンド価格) を使用できます。フリートは、これを起動条件のデフォルト上限料金として使用します。

任意で 1 つまたは複数の起動条件に上限料金を指定することができます。これは、起動条件に指定された料金です。起動条件に特定の料金が含まれる場合、EC2 フリート は起動条件の上限料金としてこの料金を使用し、全体の上限料金に優先することになります。特定の上限料金を含まないそのほかの起動条件は、全体の上限料金を引き続き使用することにご注意ください。

使用量の管理

EC2 フリート は、`TotalTargetCapacity` パラメータまたは `MaxTotalPrice` パラメータ (支払い上限料金) のいずれかに達すると、インスタンスの起動を停止します。フリートに支払う 1 時間あたりの料金を管理するには、`MaxTotalPrice` を指定します。上限の合計料金に達すると、ターゲット容量に満たない場合でも、EC2 フリート はインスタンスの起動を停止します。

以下の例は、2 つの異なるシナリオを示しています。最初の例では、ターゲット容量に達すると、EC2 フリート はインスタンスの起動を停止します。2 番目の例では、支払い上限料金 (`MaxTotalPrice`) に達すると、EC2 フリート はインスタンスの起動を停止します。

例: ターゲットキャパシティに達したときにインスタンスの起動を停止する

`m4.large` オンデマンドインスタンス に対するリクエストの内容が以下のとおりとします。

- オンデマンド料金: 1 時間あたり 0.10 USD
- OnDemandTargetCapacity: 10
- MaxTotalPrice: 1.50 USD

EC2 フリート は 10 オンデマンドインスタンス を起動します。合計料金 1.00 USD (10 インスタンス x 0.10 USD) は オンデマンドインスタンス の MaxTotalPrice (1.50 USD) を超えないためです。

例: 最大の合計料金に達したときにインスタンスの起動を停止する

m4.large オンデマンドインスタンス に対するリクエストの内容が以下のとおりとします。

- オンデマンド料金: 1 時間あたり 0.10 USD
- OnDemandTargetCapacity: 10
- MaxTotalPrice: 0.80 USD

EC2 フリート がオンデマンドターゲット容量 (10 オンデマンドインスタンス) を起動した場合、1 時間あたりの合計コストは 1.00 USD になります。これは オンデマンドインスタンス の MaxTotalPrice に指定した料金 (0.80 USD) を超えます。支払い可能な額を超えないように、EC2 フリート は 8 オンデマンドインスタンス (オンデマンドターゲット容量未満) を起動します。これを超えて起動すると、オンデマンドインスタンス の MaxTotalPrice を超えてしまいます。

EC2 フリートインスタンスの分量指定

EC2 フリートを作成する場合、各インスタンスタイプがアプリケーションのパフォーマンスに寄与する容量単位を定義できます。次に、インスタンスの分量 指定を使用して、起動仕様ごとの上限料金を調整できます。

デフォルトでは、指定する料金は 1 インスタンス時間あたり の料金となります。インスタンスの分量指定機能を使用すると、指定した料金は ユニット時間ごと の料金となります。ユニット時間あたりの使用料金はインスタンスタイプの料金を対応するユニット数で割って計算できます。EC2 フリートは、ターゲット容量をインスタンス分量で割ることで、起動するインスタンス数を計算します。その結果が整数でなければ、フリートはその数を次の整数に切り上げ、これによりフリートのサイズがターゲット容量以上になります。起動されたインスタンスの容量がリクエストされたターゲット容量を超えた場合でも、フリートは起動仕様で指定したどのプールでも選択できます。

次の表には、10 のターゲット容量の EC2 フリート のユニット当たり入札価格を特定するために計算の例が含まれています。

インスタンスタイプ	インスタンスの分量	ターゲット容量	起動されたインスタンスの数	インスタンス時間あたりのスポット料金	ユニット時間あたりの価格
r3.xlarge	2	10	5 (10 ÷ 2)	0.05 USD	0.025 USD (.05 ÷ 2)
r3.8xlarge	8	10	2 (10 ÷ 8、結果切り上げ)	0.10 USD	0.0125 USD (.10 ÷ 8)

次に示すように、EC2 フリート を使用して、受理時のユニットごとの最低価格のプールに指定するターゲット容量をプロビジョニングします。

1. EC2 フリート のターゲット容量を、インスタンス (デフォルト) あるいは仮想 CPU、メモリ、ストレージまたはスループットからご希望のユニットで設定します。
2. ユニットあたりの料金を設定します。
3. 各起動条件で、インスタンスタイプがターゲット容量に対して必要なユニット数である分量を指定します。

インスタンスの分量指定例

次の設定の EC2 フリート を検討します。

- ターゲット容量 24
- r3.2xlarge のインスタンスタイプの起動条件と分量 6
- c3.xlarge のインスタンスタイプの起動条件と分量 5

分量とは、インスタンスタイプがターゲット容量に対して必要なユニット数を表します。最初の起動条件がユニットあたりの料金を最低値で提供する場合 (インスタンス時間あたりの r3.2xlarge の

料金を 6 で割ったもの)、EC2 フリート はこれらのインスタンスから 4 つを起動します (24 を 6 で割ったもの)。

2 番目の起動条件がユニットあたりの料金を最低値で提供する場合 (インスタンス時間あたりの c3.xlarge の料金を 5 で割ったもの)、EC2 フリート はこれらのインスタンスから 5 つを起動します (24 を 5 で割ったもの、結果が切り上げられる)。

インスタンスの分量指定と配分戦略

次の設定の EC2 フリート を検討します。

- ターゲット容量 30 スポットインスタンス
- c3.2xlarge のインスタンスタイプの起動条件と分量 8
- m3.xlarge のインスタンスタイプの起動条件と分量 8
- r3.xlarge のインスタンスタイプの起動条件と分量 8

EC2 フリート は、4 つのインスタンスを起動します (30 を 8 出割ったもの、結果を切り上げ)。lowest-price 戦略では、すべての 4 つのインスタンスはユニットあたりの最低価格を提供するプールから取得されます。diversified 戦略では、フリートは 3 プールごとに 1 つのインスタンスを起動し、そしてこの 3 つのプールのいずれかから取得された 4 つ目のインスタンスがユニットあたりの最低価格を提供することになります。

EC2 フリートの操作

EC2 フリート を使用開始するには、合計ターゲット容量、オンデマンド容量、スポット容量、インスタンスの 1 つ以上の起動仕様、希望上限価格などを指定したリクエストを作成します。フリートリクエストには、フリートがインスタンスの起動に必要とする情報 (AMI、インスタンスタイプ、サブネットまたはアベイラビリティーゾーン、そして 1 つ以上のセキュリティグループ) を定義する起動テンプレートを含める必要があります。お客様は、インスタンスタイプ、サブネット、アベイラビリティーゾーン、支払い上限価格の起動条件オーバーライドを指定でき、各起動条件オーバーライドに加重容量を割り当てることができます。

EC2 フリート は、使用可能な容量があるときは オンデマンドインスタンス を起動し、上限価格がスポット料金を超えていて容量が利用可能なときは スポットインスタンス を起動します。

フリートに スポットインスタンス が含まれている場合、Amazon EC2 はスポット料金の変更に応じてフリートのターゲット容量を維持しようと試みることができます。

タイプ `maintain` または `request` の EC2 フリート リクエストは、期限切れになるかお客様によって削除されるまで、アクティブのままになります。タイプ `maintain` または `request` のフリートを削除するときは、削除によってそのフリートのインスタンスを終了するかどうかを指定できます。それ以外の場合、オンデマンドインスタンスは、終了されるまで実行され、スポットインスタンスは中断されるか終了されるまで実行されます。

コンテンツ

EC2 フリート リクエストの状態

EC2 フリート リクエストは、次に示す状態のいずれかになります。

`submitted`

EC2 フリート リクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。リクエストには オンデマンドインスタンス または スポットインスタンス、あるいはその両方を含めることができます。フリートの上限を超えたリクエストは、即時削除されます。

`active`

EC2 フリート リクエストは検証済みです。Amazon EC2 は実行中のインスタンスをターゲット数分、確保しようとしています。リクエストは、変更または削除されるまで、この状態のままになります。

`modifying`

EC2 フリート リクエストは変更中です。リクエストは、変更が完全に処理されるか、 リクエストが削除されるまで、この状態のままになります。`maintain` フリートタイプのみを変更できます。この状態はその他のリクエストタイプには適用されません。

`deleted_running`

EC2 フリート リクエストが削除され、追加のインスタンスは起動されません。その既存のインスタンスは、手動で中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。EC2 フリートリクエストが削除された後、タイプ `maintain` または `request` の EC2 フリート のみがインスタンスを実行できます。実行中のインスタンスを持つ削除した `instant` フリートはサポートされていません。この状態は `instant` フリートには適用されません。

`deleted_terminating`

EC2 フリートリクエストが削除され、そのインスタンスが終了します。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

`deleted`

EC2 フリートが削除され、実行中のインスタンスはありません。リクエストは、そのインスタンスが終了されてから 2 日後に削除されます。

EC2 フリートの前提条件

EC2 フリートを作成するには、以下の前提条件を設定する必要があります。

- 起動テンプレート
- EC2 フリート用のサービスにリンクされたロール
- 暗号化された AMI および EBS スナップショット用のカスタマーマネージド型キーへのアクセス権限の付与
- EC2 フリートユーザーのアクセス許可

起動テンプレート

起動テンプレートには、インスタンスタイプ、アベイラビリティーゾーン、支払い上限価格など、起動するインスタンスの情報が含まれています。詳細については、「起動テンプレートからのインスタンスの起動」を参照してください。

EC2 フリート用のサービスにリンクされたロール

AWSServiceRoleForEC2Fleet ロールは、インスタンスのリクエスト、起動、終了、タグ付けを行う許可を EC2 フリートに付与します。Amazon EC2 は、このサービスにリンクされたロールを使用して、以下のアクションを完了します。

- ec2:RunInstances – インスタンスを起動します。
- ec2:RequestSpotInstances – スポットインスタンスをリクエストします。
- ec2:TerminateInstances – インスタンスを終了します。
- ec2:DescribeImages – スポットインスタンス の Amazon マシンイメージ (AMI) の説明
- ec2:DescribeInstanceStatus – スポットインスタンス のステータスを表示します。
- ec2:DescribeSubnets – スポットインスタンス のサブネットを表示します。
- ec2:CreateTags – EC2 フリート、インスタンス、ボリュームにタグを追加します。

AWS CLI または API を使用して EC2 フリート を作成する前に、このロールが存在していることを確認します。

> **Note**
>
> instant EC2 フリート に、このロールは必要ありません。

ロールを作成するには、IAM コンソールを次のように使用します。

EC2 フリート の AWSServiceRoleForEC2Fleet ロールを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで ロール を選択してから、ロールを作成する を選択します。
3. [信頼されたエンティティのタイプを選択] ページで、以下の操作を実行します。
   a. [信頼できるエンティティタイプ] で、[AWS サービス] を選択します。
   b. [ユースケース] の [サービスまたはユースケース] で、[EC2 - フリート] を選択します。

> **Tip**
>
> 必ず [EC2 - フリート] を選択してください。[EC2] を選択した場合、[EC2 - フリート] ユースケースは [ユースケース] リストに表示されません。[EC2 - フリート] ユー

スケースでは、必要な IAM アクセス許可を持つポリシーが自動的に作成され、ロール名として AWSServiceRoleForEC2Fleet が提案されます。

c. [Next] を選択します。

4. [アクセス許可を追加] ページで [次へ] を選択します。
5. [名前、確認、および作成] ページで、[ロールの作成] をクリックします。

EC2 フリート を使用する必要がなくなった場合は、AWSServiceRoleForEC2Fleet ロールを削除することをお勧めします。このロールがアカウントから削除された後で、別のフリートを作成した場合はロールを再度作成できます。

詳細については、IAM ユーザーガイド の「サービスにリンクされたロールの使用」を参照してください。

暗号化された AMI および EBS スナップショット用のカスタマーマネージド型キーへのアクセス権限の付与

暗号化された AMI または暗号化された Amazon EBS スナップショットを EC2 フリートで指定し、暗号化の AWS KMS キーを使用する場合は、カスタマーマネージド型キーを使用して、Amazon EC2 がユーザーの代わりにインスタンスを起動する許可を、AWSServiceRoleForEC2Fleet ロールに付与する必要があります。これを行うには、次の手順で示すように、カスタマーマネージド型キーに許可を追加する必要があります。

アクセス権限を設定するときは、付与がキーポリシーの代わりになります。詳細については、「AWS Key Management Service デベロッパーガイド」で「許可の使用」と「AWS KMS でのキーポリシーの使用」を参照してください。

AWSServiceRoleForEC2Fleet ロールにカスタマーマネージド型キーを使用する許可を付与するには

- 許可の作成 コマンドを使用して、カスタマーマネージド型キーに許可を付与し、オペレーションを実行する許可を追加するプリンシパル (AWSServiceRoleForEC2Fleet サービスにリンクされたロール) を指定します。カスタマーマネージド型キーは、`key-id` パラメーターとカスタマーマネージド型キーの ARN を指定されます。プリンシパルを指定するには、`grantee-principal` パラメータと AWSServiceRoleForEC2Fleet サービスにリンクされたロールの ARN を使用します。

```
aws kms create-grant \
    --region us-east-1 \
```

```
    --key-id arn:aws:kms:us-
east-1:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/AWSServiceRoleForEC2Fleet \
    --operations "Decrypt" "Encrypt" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "CreateGrant" "DescribeKey" "ReEncryptFrom"
 "ReEncryptTo"
```

EC2 フリートユーザーのアクセス許可

ユーザーが EC2 フリートを作成または管理する場合、必ず必要な許可を付与してください。

EC2 フリートのポリシーを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで、ポリシー を選択します。
3. [Create policy] (ポリシーの作成) を選択します。
4. [Create policy] (ポリシーの作成) ページで、JSON タブを選択し、テキストを以下に置き換えて [Review policy] (ポリシーの確認) を選択します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
              "iam:ListRoles",
              "iam:PassRole",
              "iam:ListInstanceProfiles"
            ],
            "Resource":"arn:aws:iam::123456789012:role/DevTeam*"
        }
    ]
}
```

`ec2:*` は、ユーザーにすべての Amazon EC2 API アクションを呼び出す許可を付与します。特定の Amazon EC2 API アクションに制限するには、代わりにこれらのアクションを指定します。

IAM ユーザーは、既存の IAM ロールを列挙する `iam:ListInstanceProfiles` アクション、EC2 フリートロールを指定する `iam:PassRole` アクション、および既存のインスタンスプロファイルを列挙する `iam:ListRoles` アクションを呼び出すには、許可が必要です。

(オプション) ユーザーが IAM コンソールを使用してロールまたはインスタンスプロファイルを作成できるようにするには、次のアクションをポリシーに追加する必要もあります。

- `iam:AddRoleToInstanceProfile`
- `iam:AttachRolePolicy`
- `iam:CreateInstanceProfile`
- `iam:CreateRole`
- `iam:GetRole`
- `iam:ListPolicies`

5. [Review policy] (ポリシーの確認) ページでポリシー名と説明を入力し、[Create policy] (ポリシーの作成) を選択します。
6. アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

   - AWS IAM Identity Center のユーザーとグループ:

     アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」の手順に従ってください。

   - IAM 内で、ID プロバイダーによって管理されているユーザー:

     ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

   - IAM ユーザー:

     - ユーザーが担当できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。

- (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

EC2 フリート ヘルスチェック

EC2 フリート は、2 分ごとにフリートのインスタンスのヘルスステータスをチェックします。インスタンスのヘルスステータスは healthy または unhealthy です。

EC2 フリート は Amazon EC2 によって提供されるステータスチェックを使用して、インスタンスのヘルスステータスを判断します。インスタンスステータスチェックまたはシステムステータスチェックのいずれかのステータスが 3 回の連続したヘルスステータスチェックで impaired の場合、インスタンスは unhealthy と判断されます。詳細については、「インスタンスのステータスチェック」を参照してください。

フリートを設定して、異常のある スポットインスタンス を置き換えることができます。ReplaceUnhealthyInstances を true に設定した後、unhealthy として報告されたときにスポットインスタンスが置き換えられます。異常のあるスポットインスタンスを置き換えている間、最大数分間、フリートがターゲット容量を下回る場合があります。

要件

- ヘルスチェックによる置き換えは、タイプ request または instant のフリートではなく、ターゲットキャパシティを維持している EC2 フリート (タイプ maintain のフリート) でのみサポートされます。
- ヘルスチェックによる置き換えは、スポットインスタンス でのみサポートされます。この機能はオンデマンドインスタンス ではサポートされていません。
- 作成時のみ異常なインスタンスを置き換えるよう EC2 フリート を設定できます。
- ユーザーは、ec2:DescribeInstanceStatus アクションを呼び出す許可を持っている場合のみ、ヘルスチェックの置き換えを使用できます。

異常のある スポットインスタンス を置き換えるように EC2 フリート を設定するには

1. 表示されるステップに従って EC2 フリート を作成します。詳細については、「EC2 フリートの作成」を参照してください。
2. 異常のあるスポットインスタンスを置き換えるようにフリートを設定するには、JSON ファイルの ReplaceUnhealthyInstances に true と入力します。

EC2 フリート JSON 設定ファイルの生成

フリート設定パラメータの詳細なリストを見るには、JSON ファイルを次のように作成できます。各パラメータの説明については、AWS CLI コマンドリファレンスの「create-fleet」を参照してください。

コマンドラインを使用して使用可能なすべての EC2 フリートパラメータを含む JSON ファイルを生成するには

- create-fleet (AWS CLI) コマンドと `--generate-cli-skeleton` パラメータを使用して、EC2 フリート JSON ファイルを生成し、出力のファイルへの保存を指示します。

```
aws ec2 create-fleet \
    --generate-cli-skeleton input > ec2createfleet.json
```

出力例

```
{
    "DryRun": true,
    "ClientToken": "",
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "MaintenanceStrategies": {
            "CapacityRebalance": {
                "ReplacementStrategy": "launch"
            }
        },
        "InstanceInterruptionBehavior": "hibernate",
        "InstancePoolsToUseCount": 0,
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true,
        "MinTargetCapacity": 0,
        "MaxTotalPrice": ""
    },
    "OnDemandOptions": {
        "AllocationStrategy": "prioritized",
        "CapacityReservationOptions": {
            "UsageStrategy": "use-capacity-reservations-first"
        },
        "SingleInstanceType": true,
        "SingleAvailabilityZone": true,
        "MinTargetCapacity": 0,
```

```
        "MaxTotalPrice": ""
    },
    "ExcessCapacityTerminationPolicy": "termination",
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "",
                "LaunchTemplateName": "",
                "Version": ""
            },
            "Overrides": [
                {
                    "InstanceType": "r5.metal",
                    "MaxPrice": "",
                    "SubnetId": "",
                    "AvailabilityZone": "",
                    "WeightedCapacity": 0.0,
                    "Priority": 0.0,
                    "Placement": {
                        "AvailabilityZone": "",
                        "Affinity": "",
                        "GroupName": "",
                        "PartitionNumber": 0,
                        "HostId": "",
                        "Tenancy": "dedicated",
                        "SpreadDomain": "",
                        "HostResourceGroupArn": ""
                    },
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 0
                        },
                        "MemoryMiB": {
                            "Min": 0,
                            "Max": 0
                        },
                        "CpuManufacturers": [
                            "amd"
                        ],
                        "MemoryGiBPerVCpu": {
                            "Min": 0.0,
                            "Max": 0.0
                        },
```

```
                        "ExcludedInstanceTypes": [
                            ""
                        ],
                        "InstanceGenerations": [
                            "previous"
                        ],
                        "SpotMaxPricePercentageOverLowestPrice": 0,
                        "OnDemandMaxPricePercentageOverLowestPrice": 0,
                        "BareMetal": "included",
                        "BurstablePerformance": "required",
                        "RequireHibernateSupport": true,
                        "NetworkInterfaceCount": {
                            "Min": 0,
                            "Max": 0
                        },
                        "LocalStorage": "excluded",
                        "LocalStorageTypes": [
                            "ssd"
                        ],
                        "TotalLocalStorageGB": {
                            "Min": 0.0,
                            "Max": 0.0
                        },
                        "BaselineEbsBandwidthMbps": {
                            "Min": 0,
                            "Max": 0
                        },
                        "AcceleratorTypes": [
                            "inference"
                        ],
                        "AcceleratorCount": {
                            "Min": 0,
                            "Max": 0
                        },
                        "AcceleratorManufacturers": [
                            "amd"
                        ],
                        "AcceleratorNames": [
                            "a100"
                        ],
                        "AcceleratorTotalMemoryMiB": {
                            "Min": 0,
                            "Max": 0
                        }
```

```
                    }
                }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 0,
        "OnDemandTargetCapacity": 0,
        "SpotTargetCapacity": 0,
        "DefaultTargetCapacityType": "on-demand",
        "TargetCapacityUnitType": "memory-mib"
    },
    "TerminateInstancesWithExpiration": true,
    "Type": "instant",
    "ValidFrom": "1970-01-01T00:00:00",
    "ValidUntil": "1970-01-01T00:00:00",
    "ReplaceUnhealthyInstances": true,
    "TagSpecifications": [
        {
            "ResourceType": "fleet",
            "Tags": [
                {
                    "Key": "",
                    "Value": ""
                }
            ]
        }
    ],
    "Context": ""
}
```

EC2 フリートの作成

EC2 フリートを作成するには、次のパラメータを指定するだけです。

- LaunchTemplateId または LaunchTemplateName — 使用する起動テンプレートを指定します (インスタンスタイプ、アベイラビリティーゾーン、支払い上限価格など、起動するインスタンスのパラメータが含まれています)。
- TotalTargetCapacity — フリートの合計ターゲット容量を指定します。
- DefaultTargetCapacityType — デフォルトの購入オプションをオンデマンドにするかスポットにするかを指定します。

起動テンプレートをオーバーライドする複数の起動条件を指定できます。起動条件は、インスタンスタイプ、アベイラビリティーゾーン、サブネット、上限価格によって異なり、異なる加重容量が含まれていることがあります。または、インスタンスが持つ必要がある属性を指定すると、Amazon EC2 はそれらの属性を持つすべてのインスタンスタイプを識別します。詳細については、EC2 フリートの属性ベースのインスタンスタイプの選択 をご参照ください。

パラメータを指定しない場合、フリートはデフォルト値を使用します。

JSON ファイルのフリートパラメータを指定します。詳細については、「EC2 フリート JSON 設定ファイルの生成」を参照してください。

EC2 フリートを作成するためのコンソールのサポートは現在ありません。

EC2 フリート (AWS CLI) を作成するには

- create-fleet (AWS CLI) コマンドを使用して EC2 フリートを作成し、フリート設定パラメータを含む JSON ファイルを指定します。

```
aws ec2 create-fleet --cli-input-json file://file_name.json
```

設定ファイルの例については、「EC2 フリートの設定例」を参照してください。

タイプ `request` またはタイプ `maintain` のフリートの出力例を次に示します。

```
{
    "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE"
}
```

ターゲット容量を起動したタイプ `instant` のフリートの出力例を次に示します。

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
  "Errors": [],
  "Instances": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
```

```
        "Overrides": {
          "InstanceType": "c5.large",
          "AvailabilityZone": "us-east-1a"
        }
      },
      "Lifecycle": "on-demand",
      "InstanceIds": [
        "i-1234567890abcdef0",
        "i-9876543210abcdef9"
      ],
      "InstanceType": "c5.large",
      "Platform": null
    },
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c4.large",
          "AvailabilityZone": "us-east-1a"
        }
      },
      "Lifecycle": "on-demand",
      "InstanceIds": [
        "i-5678901234abcdef0",
        "i-5432109876abcdef9"
      ]
  ]
}
```

ターゲット容量の一部を起動し、起動されなかったインスタンスをエラーとするタイプ instant のフリートの出力例を次に示します。

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
  "Errors": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
```

```
        },
        "Overrides": {
          "InstanceType": "c4.xlarge",
          "AvailabilityZone": "us-east-1a",
        }
      },
      "Lifecycle": "on-demand",
      "ErrorCode": "InsufficientInstanceCapacity",
      "ErrorMessage": ""
    },
  ],
  "Instances": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c5.large",
          "AvailabilityZone": "us-east-1a"
        }
      },
      "Lifecycle": "on-demand",
      "InstanceIds": [
        "i-1234567890abcdef0",
        "i-9876543210abcdef9"
      ]
  ]
}
```

インスタンスを起動しなかったタイプ instant のフリートの出力例を次に示します。

```
{
  "FleetId": "fleet-12a34b55-67cd-8ef9-ba9b-9208dEXAMPLE",
  "Errors": [
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
```

```
          "InstanceType": "c4.xlarge",
          "AvailabilityZone": "us-east-1a",
        }
      },
      "Lifecycle": "on-demand",
      "ErrorCode": "InsufficientCapacity",
      "ErrorMessage": ""
    },
    {
      "LaunchTemplateAndOverrides": {
        "LaunchTemplateSpecification": {
          "LaunchTemplateId": "lt-01234a567b8910abcEXAMPLE",
          "Version": "1"
        },
        "Overrides": {
          "InstanceType": "c5.large",
          "AvailabilityZone": "us-east-1a",
        }
      },
      "Lifecycle": "on-demand",
      "ErrorCode": "InsufficientCapacity",
      "ErrorMessage": ""
    },
  ],
  "Instances": []
}
```

EC2 フリート のタグ付け

EC2 フリート リクエストを分類および管理しやすくするため、カスタムメタデータでタグ付けすることができます。EC2 フリート タグは、作成時または作成後にリクエストに割り当てることができます。

フリートリクエストにタグを付けると、フリートによって起動されるインスタンスとボリュームには自動的にタグ付けされません。フリートによって起動されるインスタンスとボリュームには、明示的にタグを付ける必要があります。タグは、フリートリクエストのみに割り当てるか、フリートによって起動されたインスタンスのみに割り当てるか、フリートによって起動されたインスタンスにアタッチされたボリュームのみに割り当てるか、または上記 3 つすべてに割り当てるかを選択できます。

> **Note**
>
> instant フリートタイプでは、 オンデマンドインスタンス および スポットインスタンス にアタッチされているボリュームにタグ付けできます。request または maintain フリートタイプでは、オンデマンドインスタンス にアタッチされているボリュームにのみタグ付けできます。

タグの仕組みの詳細については、Amazon EC2 リソースのタグ付けを参照してください。

前提条件

リソースにタグ付けする許可をユーザーに付与します。詳細については、「例: リソースのタグ付け」を参照してください。

リソースにタグ付けする許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- ec2:CreateTags アクション。これにより、タグを作成する許可がユーザーに付与されます。
- ec2:CreateFleet アクション。これにより、EC2 フリートリクエストを作成する許可がユーザーに付与されます。
- Resource に対しては、"*" を指定することをお勧めします。これにより、ユーザーはすべてのリソースタイプにタグ付けできます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagEC2FleetRequest",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:CreateFleet"
            ],
            "Resource": "*"
}
```

Important

現在、`create-fleet` リソースに対するリソースレベルのアクセス許可はサポートされていません。リソースとして `create-fleet` を指定した場合、フリートにタグ付けしようとすると、不正な例外が発生します。以下の例は、ポリシーを設定 しない 方法を示しています。

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:CreateTags",
        "ec2:CreateFleet"
    ],
    "Resource": "arn:aws:ec2:us-east-1:111122223333:create-fleet/*"
}
```

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」の手順に従ってください。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。
- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

新しい EC2 フリート リクエストにタグ付けするには

作成時に EC2 フリート リクエストをタグ付けするには、フリートを作成するために使用した JSON ファイル でキーと値のペアを指定します。ResourceType の値は fleet にする必要があります。別の値で指定すると、フリートリクエストは失敗します。

EC2 フリート によって起動されたインスタンスおよびボリュームにタグ付けするには

フリートが起動したインスタンスおよびボリュームにタグ付けするには、EC2 フリート リクエストで参照される 起動テンプレート でタグを指定します。

> **Note**
>
> request または maintain フリートタイプによって起動される スポットインスタンス にアタッチされたボリュームにタグを付けることはできません。

既存の EC2 フリート リクエスト、インスタンス、ボリュームにタグを付けるには (AWS CLI)

create-tags コマンドを使用して、既存のリソースにタグを付けます。

```
aws ec2 create-tags \
    --resources fleet-12a34b55-67cd-8ef9-
ba9b-9208dEXAMPLE i-1234567890abcdef0 vol-1234567890EXAMPLE \
    --tags Key=purpose,Value=test
```

EC2 フリートを記述する

EC2 フリートの設定、EC2 フリートのインスタンス、EC2 フリートのイベント履歴を記述できます。

EC2 フリートを記述するには (AWS CLI)

EC2 フリート の詳細を表示するには、describe-fleets コマンドを使用します。

```
aws ec2 describe-fleets
```

> **Important**
>
> フリートがタイプ instant の場合は、フリート ID を指定する必要があります。指定しない場合、レスポンスに表示されません。--fleet-ids を次のように含めます。

```
aws ec2 describe-fleets --fleet-ids fleet-8a22eee4-f489-ab02-06b8-832a7EXAMPLE
```

出力例

```
{
    "Fleets": [
        {
            "ActivityStatus": "fulfilled",
            "CreateTime": "2022-02-09T03:35:52+00:00",
            "FleetId": "fleet-364457cd-3a7a-4ed9-83d0-7b63e51bb1b7",
            "FleetState": "active",
            "ExcessCapacityTerminationPolicy": "termination",
            "FulfilledCapacity": 2.0,
            "FulfilledOnDemandCapacity": 0.0,
            "LaunchTemplateConfigs": [
                {
                    "LaunchTemplateSpecification": {
                        "LaunchTemplateName": "my-launch-template",
                        "Version": "$Latest"
                    }
                }
            ],
            "TargetCapacitySpecification": {
                "TotalTargetCapacity": 2,
                "OnDemandTargetCapacity": 0,
                "SpotTargetCapacity": 2,
                "DefaultTargetCapacityType": "spot"
            },
            "TerminateInstancesWithExpiration": false,
            "Type": "maintain",
            "ReplaceUnhealthyInstances": false,
            "SpotOptions": {
                "AllocationStrategy": "capacity-optimized",
                "InstanceInterruptionBehavior": "terminate"
            },
            "OnDemandOptions": {
                "AllocationStrategy": "lowestPrice"
            }
        }
    ]
```

```
}
```

指定した EC2 フリート のインスタンスの詳細を表示するには、describe-fleet-instances コマンドを使用します。実行中のインスタンスの返されるリストは定期的に更新されますが、古い可能性もあります。

```
aws ec2 describe-fleet-instances  --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

出力例

```
{
    "ActiveInstances": [
        {
            "InstanceId": "i-09cd595998cb3765e",
            "InstanceHealth": "healthy",
            "InstanceType": "m4.large",
            "SpotInstanceRequestId": "sir-86k84j6p"
        },
        {
            "InstanceId": "i-09cf95167ca219f17",
            "InstanceHealth": "healthy",
            "InstanceType": "m4.large",
            "SpotInstanceRequestId": "sir-dvxi7fsm"
        }
    ],
    "FleetId": "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

指定した EC2 フリート の指定期間の履歴を表示するには、describe-fleet-history コマンドを使用します。

```
aws ec2 describe-fleet-history --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --
start-time 2018-04-10T00:00:00Z
```

出力例

```
{
    "HistoryRecords": [
        {
            "EventInformation": {
                "EventSubType": "submitted"
```

```
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:05.000Z"
        },
        {
            "EventInformation": {
                "EventSubType": "active"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:15.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "t2.small, ami-07c8bc5c1ce9598c3, ...",
                "EventSubType": "progress"
            },
            "EventType": "fleetRequestChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "{\"instanceType\":\"t2.small\", ...}",
                "EventSubType": "launched",
                "InstanceId": "i-083a1c446e66085d2"
            },
            "EventType": "instanceChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        },
        {
            "EventInformation": {
                "EventDescription": "{\"instanceType\":\"t2.small\", ...}",
                "EventSubType": "launched",
                "InstanceId": "i-090db02406cc3c2d6"
            },
            "EventType": "instanceChange",
            "Timestamp": "2020-09-01T18:26:17.000Z"
        }
    ],
    "FleetId": "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
    "LastEvaluatedTime": "1970-01-01T00:00:00.000Z",
    "StartTime": "2018-04-09T23:53:20.000Z"
}
```

EC2 フリート の変更

状態が submitted または active の EC2 フリート を変更することができます。フリートを変更すると、そのフリートは modifying 状態に移行します。

変更できるのは、maintain タイプの EC2 フリート だけです。request または instant タイプの EC2 フリート は変更できません。

EC2 フリート の以下のパラメータを変更できます。

- target-capacity-specification – TotalTargetCapacity、OnDemandTargetCapacity、および SpotTargetCapacity のターゲット容量を増やすか減らします。
- excess-capacity-termination-policy – EC2 フリート の合計ターゲット容量がフリートの現在のサイズより小さくなった場合、実行中のインスタンスが終了されるかどうか。有効な値は、no-termination および termination です。

ターゲット容量を増やすと、EC2 フリート は DefaultTargetCapacityType で指定されたインスタンス購入オプション (オンデマンドインスタンス または スポットインスタンス) に従って追加のインスタンスを起動します。

DefaultTargetCapacityType が spot の場合、EC2 フリート はその配分戦略に従って追加のスポットインスタンス を起動します。配分戦略が lowest-price の場合、フリートは、リクエスト内にある最低価格のスポットキャパシティプールからインスタンスを起動します。配分戦略が diversified の場合、フリートは、リクエストのプールにインスタンスを分散します。

ターゲット容量を減らす場合、EC2 フリート は新しいターゲット容量を超えるすべてのオープンリクエストをキャンセルします。フリートのサイズが新しいターゲット容量に達するとフリートのスポットインスタンスが終了されるようにリクエストできます。配分戦略が lowest-price である場合は、フリートの最低単価のインスタンスが終了されます。配分戦略が diversified である場合は、フリートのプール全体でインスタンスが終了されます。あるいは、EC2 フリート の現在のサイズを保持するようにリクエストすることもできますが、中断された スポットインスタンス や手動終了されたインスタンスへの置き換えはできません。

ターゲット容量が減ったために EC2 フリートがスポットインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

EC2 フリートを変更するには (AWS CLI)

modify-fleet コマンドを使用して、指定された EC2 フリート のターゲット容量を更新します。

```
aws ec2 modify-fleet \
    --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity-specification TotalTargetCapacity=20
```

ターゲット容量を小さくしてもフリートの現在のサイズを保持する場合は、前のコマンドを以下のように変更できます。

```
aws ec2 modify-fleet \
    --fleet-id fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity-specification TotalTargetCapacity=10 \
    --excess-capacity-termination-policy no-termination
```

EC2 フリート の削除

EC2 フリート が不要になった場合には、それを削除することができます。フリートを削除すると、フリートに関連付けられているすべてのスポットリクエストがキャンセルされるため、新しいスポットインスタンスは起動されません。

EC2 フリートを削除するときは、そのインスタンスをすべて終了させるかどうかを指定する必要があります。これには、オンデマンドインスタンスとスポットインスタンスの両方が含まれます。instant フリートの場合、EC2 フリートはフリートの削除時にインスタンスを終了する必要があります。実行中のインスタンスを持つ削除した instant フリートはサポートされていません。

フリートを削除するときにインスタンスを終了する必要があることを指定した場合、フリートは deleted_terminating 状態へ移行します。それ以外の場合は deleted_running 状態になり、インスタンスは中断または手動終了されるまで、引き続き実行されます。

制限事項

- 1 回のリクエストで最大 25 個の instant タイプのフリートを削除できます。
- 1 回のリクエストで最大 100 個の maintain または request タイプのフリートを削除できます。
- 上記のように、各フリートタイプのクォータを超えない場合は、1 回のリクエストで最大 125 個のフリートを削除できます。
- 削除するフリートの指定された数を超えると、フリートは削除されません。
- instant フリートを削除するのに、1 回のリクエストで最大 1000 インスタンスを終了できます。

EC2 フリート を削除してインスタンスを終了するには (AWS CLI)

delete-fleets コマンドと --terminate-instances パラメータを使用し、指定された EC2 フリートを削除して関連するインスタンスを終了します。

```
aws ec2 delete-fleets \
    --fleet-ids fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --terminate-instances
```

出力例

```
{
    "UnsuccessfulFleetDeletions": [],
    "SuccessfulFleetDeletions": [
        {
            "CurrentFleetState": "deleted_terminating",
            "PreviousFleetState": "active",
            "FleetId": "fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
        }
    ]
}
```

インスタンスを終了せずに EC2 フリートを削除するには (AWS CLI)

--no-terminate-instances パラメータを使用して前のコマンドを変更することで、関連するインスタンスを終了せずに、指定された EC2 フリートを削除できます。

> **Note**
>
> --no-terminate-instances は instant フリートではサポートされていません。

```
aws ec2 delete-fleets \
    --fleet-ids fleet-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

出力例

```
{
    "UnsuccessfulFleetDeletions": [],
    "SuccessfulFleetDeletions": [
```

```
        {
            "CurrentFleetState": "deleted_running",
            "PreviousFleetState": "active",
            "FleetId": "fleet-4b8aaae8-dfb5-436d-a4c6-3dafa4c6b7dcEXAMPLE"
        }
    ]
}
```

フリートの削除に失敗した場合のトラブルシューティング

EC2 フリート の削除に失敗すると、出力中の UnsuccessfulFleetDeletions は EC2 フリートの ID、エラーコード、エラーメッセージを返します。

エラーコードは次のとおりです。

- ExceededInstantFleetNumForDeletion
- fleetIdDoesNotExist
- fleetIdMalformed
- fleetNotInDeletableState
- NoTerminateInstancesNotSupported
- UnauthorizedOperation
- unexpectedError

ExceededInstantFleetNumForDeletion のトラブルシューティング

1 回のリクエストで 25 個 を超える instant フリートを削除しようとすると、ExceededInstantFleetNumForDeletion エラーが返されます。このエラーの出力例を次に示します。

```
{
    "UnsuccessfulFleetDeletions": [
     {
          "FleetId": " fleet-5d130460-0c26-bfd9-2c32-0100a098f625",
          "Error": {
                  "Message": "Can’t delete more than 25 instant fleets in a single
 request.",
                  "Code": "ExceededInstantFleetNumForDeletion"
           }
     },
```

```
    {
          "FleetId": "fleet-9a941b23-0286-5bf4-2430-03a029a07e31",
          "Error": {
                "Message": "Can’t delete more than 25 instant fleets in a single
 request.",
                "Code": "ExceededInstantFleetNumForDeletion"
           }
    }
    .
    .
    .
    ],
    "SuccessfulFleetDeletions": []
}
```

NoTerminateInstancesNotSupported のトラブルシューティング

フリートを削除するときに instant フリート内のインスタンスを終了しないように指定した場合、NoTerminateInstancesNotSupported エラーが返されます。--no-terminate-instances は instant フリートではサポートされていません。このエラーの出力例を次に示します。

```
{
     "UnsuccessfulFleetDeletions": [
           {
                "FleetId": "fleet-5d130460-0c26-bfd9-2c32-0100a098f625",
                "Error": {
                      "Message": "NoTerminateInstances option is not supported for
 instant fleet",
                      "Code": "NoTerminateInstancesNotSupported"
                 }
           }
      ],
      "SuccessfulFleetDeletions": []
```

UnauthorizedOperation のトラブルシューティング

インスタンスを終了するアクセス許可がない場合、インスタンスを終了する必要があるフリートを削除するときに UnauthorizedOperation エラーが発生します。以下はエラーレスポンスです。

```
<Response><Errors><Error><Code>UnauthorizedOperation</Code><Message>You are not
 authorized to perform this
```

```
operation. Encoded authorization failure message: VvuncIxj7Z_CPGNYXWqnuFV-
YjByeAU66Q9752NtQ-I3-qnDLWs6JLFd
KnSMMiq5s6cGqjjPtEDpsnGHzzyHasFHOaRYJpaDVravoW25azn6KNkUQQlFwhJyujt2dtNCdduJfrqcFYAjlEiRMkfDHt7
BHturzDK6A560Y2nDSUiMmAB1y9UNtqaZJ9SNe5sNxKMqZaqKtjRbk02RZu5V2vn9VMk6fm2aMVHbY9JhLvGypLcMUjtJ76
VPiU5v2s-
UgZ7h0p2yth6ysUdhlONg6dBYu8_y_HtEI54invCj4CoK0qawqzMNe6rcmCQHvtCxtXsbkgyaEbcwmrm2m01-
EMhekLFZeJLr
DtYOpYcEl4_nWFX1wtQDCnNNCmxnJZAoJvb3VMDYpDTsxjQv1PxODZuqWHs23YXWVywzgnLtHeRf2o4lUhGBw17mXsS07k7
PT9vrHtQiILor5VVTsjSPWg7edj__1rsnXhwPSu8gI48ZLRGrPQqFq0RmKO_QIE8N8s6NWzCK4yoX-9gDcheurOGpkprPIC
</Message></Error></Errors><RequestID>89b1215c-7814-40ae-a8db-41761f43f2b0</
RequestID></Response>
```

エラーを解決するには、次の例に示すように、ec2:TerminateInstances アクションを IAM ポリシーに追加する必要があります。

```
{
       "Version": "2012-10-17",
       "Statement": [
            {
                "Sid": "DeleteFleetsAndTerminateInstances",
                "Effect": "Allow",
                "Action": [
                       "ec2:DeleteFleets"
                       "ec2:TerminateInstances"
                ],
                "Resource": "*"
            }
       ]
   }
```

スポットフリート

スポットフリートは、スポットインスタンスのセットであり、オプションで、指定した条件に基づいて起動されるオンデマンドインスタンスでもあります。スポットフリートは、ニーズに合うスポットキャパシティープールを選択して、フリートのターゲット容量を満たすまでスポットインスタンスを起動します。デフォルトでは、スポットフリート は、フリートの スポットインスタンス が削除された後に代替インスタンスを作成することによってターゲット容量が 維持 されるように設定されています。インスタンスの終了後に保持されないワンタイム リクエスト としてスポットフリートを送信できます。オンデマンドインスタンスリクエストをスポットフリートリクエストに含めることができます。

> **Note**
>
> コンソールを使用してスポットインスタンスを含むフリートを作成する場合は、スポットフリートではなく Auto Scaling グループを使用することをお勧めします。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「複数のインスタンスタイプと購入オプションを使用する Auto Scaling グループ」を参照してください。
>
> AWS CLI を使用してスポットインスタンスを含むフリートを作成する場合は、スポットフリートではなく Auto Scaling グループまたは EC2 フリートのいずれかを使用することをお勧めします。スポットフリートのベースとなる RequestSpotFleet API は、投資が計画されていないレガシー API です。
>
> 使用が推奨される API の詳細については、「使用すべき最適なスポットリクエスト方法はどれですか?」を参照してください。

トピック

- スポットフリートのリクエストタイプ
- スポットフリートの設定戦略
- スポットフリートの操作
- スポットフリートの CloudWatch メトリクス
- スポットフリートの自動スケーリング

スポットフリートのリクエストタイプ

スポットフリートリクエストには2 つのタイプがあります。

`request`

リクエストタイプを `request` として設定した場合、スポットフリートは希望する容量に非同期ワンタイムリクエストを配置します。それ以降にスポットの中断のためにキャパシティーが減少した場合、フリートは スポットインスタンス の補充を試みません。また、キャパシティーが利用できない場合にも代替のスポットキャパシティープールへのリクエストを送信しません。

`maintain`

(デフォルト) リクエストタイプを `maintain` として設定した場合、スポットフリートは希望する容量に非同期リクエストを配置し、中断された スポットインスタンス を自動的に補充して、容量を維持します。

Amazon EC2 コンソールでリクエストのタイプを指定するには、スポットフリートリクエストを作成するときに次の操作を行います。

- タイプ `request` のスポットフリートを作成するには [Maintain target capacity] (ターゲット容量を維持する) チェックボックスをクリアにします。
- タイプ `maintain` のスポットフリートを作成するには [Maintain target capacity] (ターゲット容量を維持する) チェックボックスを選択します。

詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)」を参照してください。

どちらのリクエストタイプも分散戦略の恩恵を受けます。詳細については、「スポットインスタンスの配分戦略」を参照してください。

スポットフリートの設定戦略

スポットフリート は、スポットインスタンスのコレクションまたはフリートであり、オプションで、オンデマンドインスタンスでもあります。

スポットフリートは、スポットフリートリクエストで指定したターゲット容量を満たすため、スポットインスタンスとオンデマンドインスタンスの数を起動しようとします。スポットインスタンス へのリクエストは、利用可能な容量があり、リクエストで指定した上限料金がスポット料金を超えている場合に達成されます。スポットインスタンス が中断した場合、スポットフリートはターゲット容量フリートを維持しようともします。

フリートに支払う 1 時間あたりの支払い上限額を設定し、上限額に達するまで スポットフリートでインスタンスを起動することもできます。支払い上限料金に達すると、ターゲット容量に満たない場合でも、フリートはインスタンスの起動を停止します。

[Spot capacity pool] (スポットキャパシティプール) は、同様のインスタンスタイプ (`m5.large` など)、オペレーティングシステム、アベイラビリティーゾーン、ネットワークプラットフォームの一連の使われていない EC2 インスタンスです。スポットフリートのリクエストを行う場合、複数の起動条件を含めることができ、これにはインスタンスタイプ、AMI、アベイラビリティーゾーン、またはサブネットがあります。スポットフリートは、スポットフリートリクエストに含まれる起動条件と、そのスポットフリートリクエストの設定に基づいてリクエストを満たすために使用されるスポットキャパシティープールを選択します。スポットインスタンスは選択されたプールから取得されます。

コンテンツ

スポットフリートリクエストの計画

スポットフリートリクエストを作成する前に、「スポットのベストプラクティス」を確認してください。スポットフリートリクエストを計画するときにこれらのベストプラクティスを使用して、できるだけ低価格で希望するインスタンスのタイプをプロビジョニングできます。また、次のことをお勧めします。

- 希望するターゲット容量のワンタイムリクエストを送信するスポットフリート、またはターゲット容量の継続した維持を行うスポットフリートのどちらを作成するか決定します。
- アプリケーションの要件を満たすインスタンスタイプを決定します。
- スポットフリートリクエストのターゲット容量を決定します。インスタンスまたはカスタムユニットでターゲット容量を設定できます。詳細については、「スポットフリートインスタンスの分量指定」を参照してください。

- スポットフリートのターゲット容量のどの部分がオンデマンド容量となるか決定します。オンデマンドキャパシティーに対して 0 を指定できます。
- インスタンス分量指定を使用している場合は、ユニット当りの料金を決定します。インスタンス時間当りの料金の計算は、インスタンス時間当たりの料金をそのインスタンスが表すユニット数 (または分量) で割って算出します インスタンス分量指定を使用する場合、ユニット当りのデフォルトの料金は 1 インスタンス時間当りの料金となります。
- スポットフリートリクエストの可能なオプションを確認します。詳細については、「AWS CLI コマンドリファレンス」の「request-spot-fleet」コマンドを参照してください。その他の例については、「スポットフリートの設定例」を参照してください。

スポットインスタンスの配分戦略

起動設定によって、スポットフリートがスポットインスタンスを起動できるすべてのスポットキャパシティプール (インスタンスタイプおよびアベイラビリティーゾーン) が決定されます。ただし、インスタンスを起動する際、スポットフリートは指定された配分戦略を使用して、使用可能なすべてのプールから特定のプールを選択します。

> **Note**
>
> AMD SEV-SNP を有効にしてスポットインスタンスを起動するように設定すると、選択したインスタンスタイプのオンデマンド時間料金の 10% に相当する追加の時間単位使用料が請求されます。配分戦略で価格を入力として使用する場合、スポットフリートにはこの追加料金は含まれず、スポット料金のみが使用されます。

配分戦略

スポットインスタンスには次のいずれかの配分戦略を指定できます。

`priceCapacityOptimized` (推奨)

スポットフリートは、起動中のインスタンスの数に容量の可用性が最も高いプールを識別します。つまり、短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。次にスポットフリートは、これらのプールのうち最も価格の低いスポットインスタンスをリクエストします。

priceCapacityOptimized 配分戦略は、ステートレスコンテナ化アプリケーション、マイクロサービス、ウェブアプリケーション、データおよび分析ジョブ、バッチ処理など、ほとんどのスポットワークロードに最適です。

capacityOptimized

スポットフリートは、起動中のインスタンスの数に容量の可用性が最も高いプールを識別します。つまり、短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。オプションで capacityOptimizedPrioritized により、フリート内の各インスタンスタイプに優先順位を設定できます。スポットフリートは、最初に容量を最適化しますが、ベストエフォートベースでインスタンスタイプの優先順位を重視します。

スポットインスタンス では、価格は需要と供給の長期的な傾向に基づいて時間の経過とともに緩やかに変動しますが、容量はリアルタイムで変動します。capacityOptimized 戦略では、リアルタイムの容量データを調べ、可用性の最も高いプールを予測することで、そのプールから スポットインスタンス を自動的に起動します。この戦略は、作業の再開に関連する中断のコストが高くなる可能性のあるワークロード (長時間の継続的インテグレーション (CI)、画像とメディアのレンダリング、深層学習およびハイパフォーマンスコンピューティング (HPC) など) に対応します。中断の可能性を低くすることにより、capacityOptimized 戦略ではワークロードの全体的なコストを削減できます。

または、優先パラメータで capacityOptimizedPrioritized 配分戦略を使用して、インスタンスタイプを優先順位の高い順から低い順へ指定できます。異なるインスタンスタイプに対し同じ優先順位を設定できます。スポットフリートは最初に容量を最適化しますが、インスタンスタイプの優先順位をベストエフォートベースで決定します (例えば、優先順位を尊重しても、EC2 フリートの最適な容量をプロビジョニングする能力に大きな影響を与えない場合など) 。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。優先順位の使用は、フリートが起動テンプレートを使用する場合にのみサポートされます。capacityOptimizedPrioritized の優先順位を設定するとき、オンデマンド AllocationStrategy が prioritized に設定されていると、同じ優先順位がオンデマンドインスタンスにも適用されますのでご注意ください。

diversified

スポットインスタンス はすべてのプールに分散されます。

lowestPrice

スポットインスタンスは、使用可能な容量を持つ最低価格のプールから取得されます。これはデフォルトの戦略です。ただし、priceCapacityOptimized 配分戦略を指定してデフォルトを上書きすることをお勧めします。

最低価格のプールに使用可能な容量がない場合、スポットインスタンスは使用可能な容量のある 2 番目に低価格のプールから取得されます。

希望する容量を満たす前にプールの容量が不足した場合、スポットフリートは 2 番目に低い価格のプールから容量を引き出し、引き続きリクエストを満たします。希望する容量を確実に満たすために、複数のプールからスポットインスタンスを受け取る場合があります。

この戦略では、インスタンスの価格のみが考慮され、容量の可用性は考慮されないため、中断率が高くなる可能性があります。

`InstancePoolsToUseCount`

ターゲットスポット容量を割り当てる先のスポットプールの数。配分戦略が `lowestPrice` に設定されている場合にのみ有効です。スポットフリートでは最低価格のスポットプールを選択し、指定した数のスポットプールにターゲットスポット容量を均等に割り当てます。

スポットフリートは、指定したプール数内のスポットインスタンスを、ベストエフォート方式で利用しようとすることにご注意ください。ターゲット容量を満たす前にプールにスポットキャパシティーの残量がなくなった場合、スポットフリートは次に低い価格のプールの容量を利用してリクエストを満たします。ターゲット容量を確実に満たすために、スポットインスタンスが、指定した数を超えるプールから割り当てられることがあります。また、ほとんどのプールにスポット容量がない場合には、指定した数より少ないプールからターゲット容量のすべてが割り当てられることがあります。

適切な配分戦略の選択

適切なスポット割り当て戦略を選択することで、ユースケースに合わせてフリートを最適化できます。オンデマンドインスタンスのターゲット容量では、スポットフリートはスポットインスタンス の配分戦略 (`priceCapacityOptimized`、`capacityOptimized` `diversified` または `lowestPrice`) を採用しながら、パブリックオンデマンド料金に基づいて、最低価格のインスタンスタイプを常に選択します。

最低価格と容量可用性のバランスをとる

最低価格のスポット容量プールと容量の可用性が最も高いスポットキャパシティプールとのトレードオフのバランスをとるには、`priceCapacityOptimized` 配分戦略を使用することをお勧めします。この戦略では、プールの価格とプール内のスポットインスタンスの空き容量の両方に基づいて、どのプールからスポットインスタンスをリクエストするかを決定します。つまり、価格を考慮しながらも短期的に中断の可能性が最も低いと思われるプールからスポットインスタンスをリクエストすることになります。

コンテナ化されたアプリケーション、マイクロサービス、ウェブアプリケーション、データおよび分析ジョブ、バッチ処理など、レジリエントでステートレスなワークロードをフリートが実行している場合は、最適なコスト削減とキャパシティアベイラビリティを実現する priceCapacityOptimized 配分戦略を使用してください。

作業の再開に関連する中断に伴うコストが高くなる可能性があるワークロードをフリートで実行している場合は、中断があった場合にアプリケーションがそのポイントから再起動できるようにチェックポイントの設定を実装する必要があります。チェックポイントを使用すると、スポットインスタンスの中断率も低い最低価格のプールから容量が割り当てられるため、priceCapacityOptimized 配分戦略がこれらのワークロードに適したものになります。

priceCapacityOptimized 配分戦略を使用する設定例については、「例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する」を参照してください。

ワークロードの中断コストが高い場合

同様の価格のインスタンスタイプを使用するワークロードを実行する場合や、中断のコストが非常に高いため、中断のわずかな増加に比べてコスト削減が不十分な場合、オプションでこの capacityOptimized 戦略を使用できます。この戦略では、中断の可能性がより低く、最も可用性の高いスポットキャパシティプールから容量を割り当てることで、ワークロードの総コストを削減することができます。capacityOptimized 配分戦略を使用する設定例については、「例 8: 容量の再調整を設定して代替スポットインスタンスを起動する」を参照してください。

中断の可能性を最小限に抑える必要があるが、特定のインスタンスタイプの優先順位が重要な場合は、capacityOptimizedPrioritized の配分戦略を使用し、インスタンスタイプの順序を優先順位の高い順に表現することでプールの優先順位を設定することができます。設定の例については、「例 9: 容量最適化フリートでスポットインスタンスを起動する」を参照してください。

優先順位の使用は、フリートが起動テンプレートを使用する場合にのみサポートされることに注意してください。capacityOptimizedPrioritized の優先順位を設定する際に、オンデマンド AllocationStrategy が prioritized に設定されていると、同じ優先順位がオンデマンドインスタンスにも適用されるので注意してください。

ワークロードに時間的な柔軟性があり、キャパシティの可用性が問題にならない場合

フリートが小さい場合、または短時間の実行である場合、容量の可用性を考慮しながら、priceCapacityOptimized を使用してコスト削減を最大化できます。

ワークロードに時間的な柔軟性があり、キャパシティの可用性が問題にならない場合は、オプションで lowestPrice 配分戦略を使用してコスト削減を最大化できます。この lowestPrice 配分戦略

では、インスタンスの価格のみが考慮され、容量の可用性は考慮されないため、スポットインスタンス中断率が高くなる可能性があることをご注意ください。

フリートが大きい場合や長時間稼働している場合

フリートが大規模、または長期間実行される場合には、diversified 戦略を使用して複数のプールに スポットインスタンス を分散することで、フリートの可用性を改善できます。例えば、スポットフリートが 10 プールとして、ターゲット容量が 100 インスタンスと指定すると、フリートはプールごとに 10 個の スポットインスタンス を起動します。1 つのプールのスポット料金がこのプールの上限料金を超える場合、フリートの 10% のみに影響がおよびます。この戦略を使用すると、いずれのプールにおいても経時的にフリートが受けるスポット料金の上昇の影響を減少させます。diversified 戦略では、スポットフリートは、オンデマンド価格 以上のスポット料金のいずれのプールにもスポットインスタンスを起動しません。

安価で分散型のフリートを作成するには、lowestPrice 戦略を InstancePoolsToUseCount と組み合わせて使用します。例えば、ターゲットのキャパシティが 10 のスポットインスタンスで、2 つのスポットキャパシティプールを (InstancePoolsToUseCount により) 指定した場合、スポットフリートはスポットキャパシティを満たすために最も安価プールを 2 つ利用します。

スポットインスタンス を配分するために、少数または多数のスポットキャパシティープールを選択して使用することができます。たとえば、バッチ処理を実行する場合は、少数のスポットキャパシティープール (など) を指定することをお勧めします。これにより、キューのコンピューティング性能を常に確保しながら、コストを最大限削減できます。InstancePoolsToUseCount=2ウェブサービスを実行する場合は、スポットキャパシティプールが一時的に使用不可になった場合の影響を最小限に抑えるために、多数のスポットキャパシティプール (InstancePoolsToUseCount=10 など) を指定することをお勧めします。

スポットフリートは、指定したプール数内のスポットインスタンスを、ベストエフォート方式で利用しようとすることにご注意ください。ターゲット容量を満たす前にプールにスポットキャパシティーの残量がなくなった場合、スポットフリートは次に低い価格のプールの容量を利用してリクエストを満たします。ターゲット容量を確実に満たすために、スポットインスタンスが、指定した数を超えるプールから割り当てられることがあります。また、ほとんどのプールにスポット容量がない場合には、指定した数より少ないプールからターゲット容量のすべてが割り当てられることがあります。

ターゲット容量の維持

スポット料金やスポットキャパシティプールで使用可能な容量の変動に伴ってスポットインスタンスが終了すると、タイプ maintain のスポットフリートは代替スポットインスタンスを起動します。配分戦略によって、次のように置換先インスタンスを起動するプールが決まります。

- 割当戦略が priceCapacityOptimized の場合、フリートは最もスポットインスタンスの容量が利用可能なプールで置換先インスタンスを起動します。また、価格も考慮し、容量利用率の高い価格の低いプールを特定します。
- 配分戦略が capacityOptimized の場合、フリートは、利用可能なスポットインスタンス容量が最大のプールで置換先インスタンスを起動します。
- 配分戦略が diversified である場合には、フリートは残りのプールに代替 スポットインスタンス を分散します。
- 配分戦略が lowestPrice である場合、スポット群は、スポット料金が現在最低値のプールに代替インスタンスを起動します。
- 割り当て戦略が lowestPrice と InstancePoolsToUseCount の組み合わせである場合、フリートは最低価格のスポット容量プールを選択し、指定した数のスポット容量プールにわたってスポットインスタンスを起動します。

スポットフリートの属性ベースのインスタンスタイプの選択

スポットフリートを作成するときは、フリートのオンデマンドインスタンスとスポットインスタンスを設定するための 1 つ以上のインスタンスタイプを指定する必要があります。インスタンスタイプを手動で指定する代わりに、インスタンスが持つ必要がある属性を指定でき、Amazon EC2 は、それらの属性を持つすべてのインスタンスタイプを識別します。これは 属性ベースのインスタンスタイプの選択 と呼ばれます。例えば、インスタンスに必要な vCPU の最小数と最大数を指定でき、スポットフリートはこれらの vCPU 要件を満たす使用可能なインスタンスタイプを使用してインスタンスを起動します。

属性ベースのインスタンスタイプの選択は、コンテナやウェブフリートの実行、ビッグデータの処理、継続的インテグレーションおよびデプロイ (CI/CD) ツールの実装など、使用するインスタンスタイプについて柔軟に使用できるワークロードとフレームワークに最適です。

利点

属性ベースのインスタンスタイプを選択すると、次の利点があります。

- 適切なインスタンスタイプを簡単に使用 - 利用可能なインスタンスタイプの数が多いため、ワークロードに適したインスタンスタイプを見つけるには時間がかかることがあります。インスタンス属性を指定すると、インスタンスタイプにはワークロードに必要な属性が自動的に設定されます。
- 設定の簡素化 - スポットフリートに複数のインスタンスタイプを手動で指定するには、インスタンスタイプごとに個別の起動テンプレートの上書きを作成する必要があります。ただし、属性ベース

のインスタンスタイプを選択すると、複数のインスタンスタイプを提供するには、起動テンプレートまたは起動テンプレートの上書きでインスタンス属性を指定するだけで済みます。

- 新しいインスタンスタイプを自動的に使用 - インスタンスタイプではなくインスタンス属性を指定すると、フリートではリリース時に新しい世代のインスタンスタイプを使用できます。これにより、フリートの設定の将来の対応性も確保されます。
- インスタンスタイプの柔軟性 - インスタンスタイプではなくインスタンス属性を指定すると、スポットフリートは、スポットインスタンスを起動するために幅広いインスタンスタイプから選択することができ、インスタンスタイプの柔軟性というスポットのベストプラクティスに準拠することができます。

トピック

- 属性ベースのインスタンスタイプ選択の仕組み
- 料金保護
- 考慮事項
- 属性ベースのインスタンスタイプを選択してスポットフリートを作成する
- 有効な設定と無効な設定の例
- 指定された属性でインスタンスタイプをプレビューする

属性ベースのインスタンスタイプ選択の仕組み

フリート設定で属性ベースのインスタンスタイプの選択を使用するには、インスタンスタイプのリストをインスタンスが必要とするインスタンス属性のリストに置き換えます。スポットフリートは、指定されたインスタンス属性を持つ使用可能なインスタンスタイプでインスタンスを起動します。

トピック

- インスタンス属性のタイプ
- 属性ベースのインスタンスタイプの選択を設定する場所
- フリートをプロビジョニングするときに、スポットフリートが属性ベースのインスタンスタイプの選択をどのように使用するかについて

インスタンス属性のタイプ

コンピューティング要件を表現するために指定できるインスタンス属性はいくつかあります。

- vCPU 数 – インスタンスあたりの vCPU の最小数と最大数。

- メモリ – インスタンスあたりのメモリの最小および最大 GiB。
- ローカルストレージ – EBS ボリュームとインスタンスストアボリュームのどちらをローカルストレージに使用するか。
- バースト可能なパフォーマンス – T4g、T3a、T3、および T2 タイプを含む T インスタンスファミリーを使用するかどうか。

各属性の説明およびデフォルト値については、「Amazon EC2 API リファレンス」の「InstanceRequirements」を参照してください。

属性ベースのインスタンスタイプの選択を設定する場所

コンソールと AWS CLI のどちらを使用するかによって、属性ベースのインスタンスタイプ選択のインスタンス属性を次のように指定できます。

コンソールでは、次のフリート設定コンポーネントの 1 つまたは両方でインスタンス属性を指定できます。

- 起動テンプレートでフリートリクエストの起動テンプレートを参照する
- フリートリクエストで

AWS CLI で、以下のフリート設定コンポーネントのいずれかまたはすべてでインスタンスの属性を指定することができます。

- 起動テンプレートでフリートリクエストの起動テンプレートを参照します
- 起動テンプレートの上書きで

  異なる AMI を使用するインスタンスを混在させたい場合は、複数の起動テンプレートの上書きでインスタンス属性を指定できます。例えば、異なるインスタンスタイプで x86 および ARM ベースのプロセッサを使用できます。
- 起動仕様で

フリートをプロビジョニングするときに、スポットフリートが属性ベースのインスタンスタイプの選択をどのように使用するかについて

スポットフリートは、以下の方法でフリートをプロビジョニングします。

- スポットフリートは、指定された属性を持つインスタンスタイプを識別します。

- スポットフリートは、料金保護を使用して、除外するインスタンスタイプを決定します。
- スポットフリートは、インスタンスタイプが一致する AWS リージョンまたはアベイラビリティーゾーンに基づいて、インスタンスの起動を検討する容量プールを決定します。
- スポットフリートは、指定された配分戦略を適用して、インスタンスを起動する容量プールを決定します。

  属性ベースのインスタンスタイプの選択では、フリートをプロビジョニングするキャパシティプールは選択されません。これが割り当て戦略のジョブです。指定された属性を持つインスタンスタイプが多数存在し、一部のインスタンスタイプにはコストがかかる場合があります。スポットとオンデマンドのデフォルトの割り当て戦略である `lowestPrice` は、スポットフリートが最も安価なキャパシティプールからインスタンスを起動することを保証します。

  配分戦略を指定すると、スポットフリートは指定された配分戦略に従ってインスタンスを起動します。

  - スポットインスタンスでは、属性ベースのインスタンスタイプ選択により、`capacityOptimizedPrioritized`、`capacityOptimized` および `lowestPrice` の配分戦略がサポートされます。
  - オンデマンドインスタンスでは、属性ベースのインスタンスタイプの選択は、`lowestPrice` 配分戦略をサポートします。
- 指定されたインスタンス属性を持つインスタンスタイプの容量がない場合、インスタンスは起動できず、フリートはエラーを返します。

料金保護

料金保護は、スポットフリートが指定した属性に適合した場合でも、コストが高すぎると考えるインスタンスタイプを使用できないようにする機能です。料金保護を使用するには、料金のしきい値を設定します。Amazon EC2 が属性を持つインスタンスタイプを選択すると、しきい値を超える料金が設定されたインスタンスタイプは除外されます。

Amazon EC2 が料金のしきい値を計算する方法は、次のとおりです。

- Amazon EC2 はまず、属性に一致するものから最低料金のインスタンスタイプを識別します。
- Amazon EC2 は、料金保護パラメータに指定した値 (パーセンテージで表される) を受け取り、識別されたインスタンスタイプの料金でそれを乗算します。その結果、料金しきい値として使用される料金になります。

オンデマンドインスタンスとスポットインスタンスには個別の料金しきい値があります。

属性ベースのインスタンスタイプを選択してフリートを作成すると、料金保護がデフォルトで有効になります。デフォルト値のままにすることも、独自の値を指定することもできます。

料金保護をオフにすることもできます。料金保護のしきい値を指定しない場合は、999999 などの高いパーセンテージ値を指定します。

トピック

最低料金のインスタンスタイプを特定する方法

Amazon EC2 は、指定した属性に一致するものから最低料金のインスタンスタイプを特定することで、料金のしきい値に基づく料金を決定します。これは、次の方法で行います。

- まず、属性に一致する現行世代の C、M、または R インスタンスタイプを調べます。一致するものがある場合は、最低料金のインスタンスタイプを特定します。
- 一致するものがない場合は、属性に一致する現行世代のインスタンスタイプを調べます。一致するものがある場合は、最低料金のインスタンスタイプを特定します。
- 一致するものがない場合は、属性に一致する以前の世代のインスタンスタイプを調べ、最低料金のインスタンスタイプを特定します。

オンデマンドインスタンスの料金保護

オンデマンドインスタンスタイプの料金保護のしきい値は、特定された最低料金のオンデマンドインスタンスタイプ (`OnDemandMaxPricePercentageOverLowestPrice`) よりも高いパーセンテージで計算されます。支払い可能なパーセンテージを高く指定します。このパラメータを指定しない場合は、デフォルト値の `20` を使用して、識別された料金よりも 20% 高い料金保護しきい値が計算されます。

例えば、特定されたオンデマンドインスタンスの料金が `0.4271` で、25 を指定した場合、料金のしきい値は `0.4271` より 25% 高くなります。これは、次のように計算されます: `0.4271 * 1.25 = 0.533875`。計算された料金は、オンデマンドインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は `0.533875` を超えるコストがかかるオンデマンドインスタンスタイプを除外します。

スポットインスタンスの料金保護

デフォルトでは、Amazon EC2 は最適なスポットインスタンス料金保護を自動的に適用し、幅広いインスタンスタイプから一貫して選択します。料金保護を手動で設定することもできます。ただし、Amazon EC2 に任せることで、スポット容量が満たされる可能性を高めることができます。

料金保護は、次のいずれかのオプションを使用して手動で指定できます。料金保護を手動で設定する場合は、最初のオプションを使用することをお勧めします。

- 特定された最低料金のオンデマンドインスタンスタイプ (**MaxSpotPriceAsPercentageOfOptimalOnDemandPrice**) のパーセンテージ

  例えば、特定されたオンデマンドインスタンスタイプの料金が `0.4271` で、`60` を指定した場合、料金のしきい値は `0.4271` の 60% になります。これは、次のように計算されます: `0.4271 * 0.60 = 0.25626`。計算された料金は、スポットインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は `0.25626` を超えるコストがかかるスポットインスタンスタイプを除外します。
- 特定された最低料金のスポットインスタンスタイプ (**SpotMaxPricePercentageOverLowestPrice**) よりも高いパーセンテージ

  例えば、特定されたスポットインスタンスタイプの料金が `0.1808` で、`25` を指定した場合、料金のしきい値は `0.1808` より 25% 高くなります。これは、次のように計算されます: `0.1808 * 1.25 = 0.226`。計算された料金は、スポットインスタンスに対して支払うことができる最大額であり、この例では、Amazon EC2 は `0.266` を超えるコストがかかるスポットインスタンスタイプを除外します。スポット料金が変動する可能性があり、料金保護のしきい値も変動する可能性があるため、このパラメータの使用はお勧めしません。

料金保護のしきい値を指定する

料金保護のしきい値を指定するには

スポットフリートを作成するときに、属性ベースのインスタンスタイプを選択するようにフリートを設定してから、次の手順を実行します。

- コンソール

  オンデマンドインスタンスの料金保護のしきい値を指定するには、[Additional instance attribute] (追加のインスタンス属性) で、[On-demand price protection] (オンデマンドの料金保護) を選択し

てから、[Add attribute] (属性を追加) を選択します。[On-Demand price protection percentage] (オンデマンドの料金保護 (%)) で、料金保護のしきい値をパーセンテージ (%) で入力します。

スポットインスタンスの料金保護のしきい値を指定するには、[Additional instance attribute] (追加のインスタンス属性) で、[Spot price protection] (スポットの料金保護) を選択してから、[Add attribute] (属性を追加) を選択します。パラメータを選択し、料金保護のしきい値をパーセンテージ (%) で入力します。

- AWS CLI

  オンデマンドインスタンスの料金保護のしきい値を指定するには、JSON 設定ファイルの `InstanceRequirements` 構造の `OnDemandMaxPricePercentageOverLowestPrice` で、料金保護のしきい値をパーセンテージ (%) で入力します。

  スポットインスタンスの料金保護のしきい値を指定するには、JSON 設定ファイルの `InstanceRequirements` 構造で、次のいずれかのパラメータを指定します。

  - `MaxSpotPriceAsPercentageOfOptimalOnDemandPrice` で、料金保護のしきい値をパーセンテージ (%) で入力します。
  - `SpotMaxPricePercentageOverLowestPrice` で、料金保護のしきい値をパーセンテージ (%) で入力します。

フリートの作成の詳細については、「属性ベースのインスタンスタイプを選択してスポットフリートを作成する」を参照してください。

> **Note**
>
> スポットフリートを作成するときに、[Total target capacity] (合計ターゲット容量) タイプを [vCPUs] もしくは [Memory (MiB)] (メモリ (MiB)) (コンソール) に、または `TargetCapacityUnitType` を `vcpu`、もしくは `memory-mib` (AWS CLI) に設定すると、料金保護のしきい値は、インスタンスごとの料金ではなく、vCPU ごとまたはメモリごとの料金に基づいて適用されます。

考慮事項

- スポットフリートでは、インスタンスタイプまたはインスタンス属性のいずれかを指定できますが、両方を同時に指定することはできません。

CLI を使用する場合、起動テンプレートの上書きによって起動テンプレートが上書きされます。例えば、起動テンプレートにインスタンスタイプが含まれ、起動テンプレートの上書きにインスタンス属性が含まれている場合、インスタンス属性によって識別されるインスタンスは、起動テンプレートのインスタンスタイプを上書きします。

- CLI を使用していて、インスタンス属性の上書きを指定する場合、重みまたは優先順位も指定できません。
- リクエスト設定では、最大 4 つの `InstanceRequirements` 構造を指定できます。

属性ベースのインスタンスタイプを選択してスポットフリートを作成する

属性ベースのインスタンスタイプ選択を使用するようにフリートを設定するには、Amazon EC2 コンソールまたは AWS CLI を使用します。

トピック

- コンソールを使用してスポットフリートを作成するには
- AWS CLI を使用したスポットフリートの作成

コンソールを使用してスポットフリートを作成するには

属性ベースのインスタンスタイプの選択にスポットフリートを設定するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] (スポットリクエスト) を選択した後、[Request Spot Instances] (スポットインスタンスのリクエスト) を選択します。
3. 手順に従ってスポットフリートを作成します。詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)」を参照してください。

   スポットフリートを作成するときに、属性ベースのインスタンスタイプ選択にフリートを次のように設定します。

   a. [Instance type requirements] (インスタンスタイプの要件) では、[Specify instance attributes that match your compute requirements] (コンピューティング要件に一致するインスタンス属性を指定する) を選択します。

   b. [vCPUs] に、希望する vCPU の最小数と最大数を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。

c. [Memory (GiB)] (メモリ (GiB)) に、希望するメモリの最小値と最大値を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。

d. (オプション) [Additional instance attributes] (その他のインスタンス属性) では、オプションで 1 つ以上の属性を指定して、コンピューティング要件をより詳細に表現できます。追加の属性は、リクエストにさらに制約を追加します。

e. (オプション) [Preview matching instance types] (一致するインスタンスタイプをプレビューする) を展開して、指定した属性を持つインスタンスタイプを表示します。

AWS CLI を使用したスポットフリートの作成

属性ベースのインスタンスタイプの選択にスポットフリートを設定するには (AWS CLI)

スポットフリートリクエストを作成するには、request-spot-fleet (AWS CLI) コマンドを使用します。JSON ファイルでフリート設定を指定します。

```
aws ec2 request-spot-fleet \
    --region us-east-1 \
    --spot-fleet-request-config file://file_name.json
```

file_name.json ファイルの例

次の例には、属性ベースのインスタンスタイプ選択を使用するようにスポットフリートを設定するパラメータが含まれており、その後にテキストによる説明が続きます。

```
{
 "AllocationStrategy": "priceCapacityOptimized",
 "TargetCapacity": 20,
 "Type": "request",
 "LaunchTemplateConfigs": [{
  "LaunchTemplateSpecification": {
   "LaunchTemplateName": "my-launch-template",
   "Version": "1"
  },
  "Overrides": [{
   "InstanceRequirements": {
    "VCpuCount": {
     "Min": 2
    },
    "MemoryMiB": {
```

```
      "Min": 4
     }
    }
   }]
  }]
}
```

属性に基づくインスタンスタイプ選択のための属性は、InstanceRequirements 構造で指定されます。この例では、2 つのタグが指定されています。

- VCpuCount — 最低 2 つの vCPUs が指定されています。最大値は指定されていないため、上限はありません。
- MemoryMiB — 4 MiB 以上のメモリが指定されています。最大値は指定されていないため、上限はありません。

2 つ以上の vCPUs と 4 MiB 以上のメモリを持つすべてのインスタンスタイプが識別されます。ただし、スポットフリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

指定できるすべての属性のリストと説明については、「Amazon EC2 API リファレンス」の「インスタンス要件」を参照してください。

> **Note**
>
> InstanceRequirements がフリート設定に含まれる場合、InstanceType と WeightedCapacity は除外しなければならず、インスタンス属性と同時にフリート設定を決定することはできません。

JSON には次のフリート設定も含まれています。

- "AllocationStrategy": "*priceCapacityOptimized*" — フリート内のスポットインスタンスの割り当て戦略。
- "LaunchTemplateName": "*my-launch-template*", "Version": "*1*" — 起動テンプレートにはいくつかのインスタンス設定情報が含まれていますが、インスタンスタイプが指定されている場合は、InstanceRequirements で指定されている属性によってオーバーライドされます。
- "TargetCapacity": *20* – ターゲット容量は 20 個のインスタンスです。
- "Type": "*request*" — フリートのリクエストタイプは request です。

有効な設定と無効な設定の例

AWS CLI を使用してスポットフリートを作成する場合は、フリート設定が有効であることを確認する必要があります。次の例は、有効な設定と無効な設定を示しています。

次のものが含まれている場合、設定は無効と見なされます。

- InstanceRequirements および InstanceType を持つ 1 つの Overrides 構造
- 一つは InstanceRequirements、もう一つは InstanceType を持つ 2 つの Overrides 構造
- 同じ LaunchTemplateSpecification 内で属性値が重複している 2 つの InstanceRequirements 構造

設定例

- 有効な設定: 上書きを含む単一の起動テンプレート
- 有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート
- 有効な設定: 2 つの起動テンプレート、それぞれに上書きがある
- 有効な設定: InstanceRequirements のみ指定され、重複する属性値なし
- 設定が無効です: Overrides が InstanceRequirements および InstanceType を含んでいる
- 設定が無効です: 2 つの Overrides に InstanceRequirements および InstanceType が含まれている
- 設定が無効です: 重複する属性値

有効な設定: 上書きを含む単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 1 つの Overrides が含まれています。以下に、構成例をテキストで説明します。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "My-launch-template",
                "Version": "1"
```

```
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 2,
                            "Max": 8
                        },
                        "MemoryMib": {
                            "Min": 0,
                            "Max": 10240
                        },
                        "MemoryGiBPerVCpu": {
                            "Max": 10000
                        },
                        "RequireHibernateSupport": true
                    }
                }
            ]
        }
    ],
        "TargetCapacity": 5000,
            "OnDemandTargetCapacity": 0,
            "TargetCapacityUnitType": "vcpu"
    }
}
```

InstanceRequirements

属性ベースのインスタンス選択を使用するには、フリート設定に InstanceRequirements 構造を含め、フリート内のインスタンスに必要な属性を指定する必要があります。

前の例に、以下のインスタンス属性が指定されています。

- VCpuCount - インスタンスタイプには、2 個以上、最大 8 個の vCPU が必要です。
- MemoryMiB - インスタンスタイプには最大 10,240 MiB のメモリが必要です。最小数が 0 の場合、最小制限がないことを示します。
- MemoryGiBPerVCpu - インスタンスタイプには、vCPU あたり最大 10,000 GiB のメモリが必要です。Min パラメータはオプションです。省略すると、最小制限がないことを示します。

TargetCapacityUnitType

TargetCapacityUnitType パラメータは、ターゲット容量の単位を指定します。この例では、ターゲット容量は 5000 であり、ターゲット容量ユニットタイプは vcpu で、これを組み合わせて 5,000 vCPU の希望するターゲット容量を指定します。スポットフリートは、フリート内の vCPU の総数が 5,000 vCPU になるように、十分なインスタンスを起動します。

有効な設定: 複数のインスタンス要件を持つ単一の起動テンプレート

以下の設定は有効です。これには、1 つの起動テンプレートと、InstanceRequirements 構造を含む 2 つの Overrides が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
```

```
                            "Min": 0
                        }
                    }
                }
            ]
        }
    ],
    "TargetCapacity": 1,
    "OnDemandTargetCapacity": 0,
    "Type": "maintain"
  }
}
```

有効な設定: 2 つの起動テンプレート、それぞれに上書きがある

以下の設定は有効です。これには 2 つの起動テンプレートが含まれ、各起動テンプレートには 1 つの InstanceRequirements 構造を含む Overrides 構造が 1 つ含まれています。この設定は、同じフリートで arm と x86 のアーキテクチャをサポートする場合に有効です。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "armLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
```

```
                {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "x86LaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

有効な設定: **InstanceRequirements** のみ指定され、重複する属性値なし

以下の設定は有効です。2 つの LaunchTemplateSpecification 構造が含まれ、各構造にぞれぞれ起動テンプレートと、Overrides 構造を含む InstanceRequirements 構造が含まれています。InstanceRequirements で指定された属性は、値が重複していないため有効です。最初の InstanceRequirements 構造は VCpuCount の 0～2 vCPU を指定し、2 つ目の InstanceRequirements 構造は 4～8 vCPU を指定しています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
```

```
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 4,
                            "Max": 8
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

設定が無効です: Overrides が InstanceRequirements および InstanceType を含んでいる

以下は設定が有効ではありません。Overrides 構造体には InstanceRequirements および InstanceType が両方含まれています。Overrides では、InstanceRequirements または InstanceType のどちらかを指定できますが、両方は指定できません。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                },
                {
                    "InstanceType": "m5.large"
                }
              ]
            }
        ],
        "TargetCapacity": 1,
        "OnDemandTargetCapacity": 0,
        "Type": "maintain"
    }
}
```

設定が無効です: 2 つの Overrides に InstanceRequirements および InstanceType が含まれている

以下は設定が有効ではありません。Overrides 構造に InstanceRequirements および InstanceType が両方含まれています。異なる Overrides 構造にある場合、InstanceRequirements または InstanceType を指定できますが、両方を指定することはできません。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
              ]
            },
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyOtherLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceType": "m5.large"
                }
```

```
            ]
        }
    ],
    "TargetCapacity": 1,
    "OnDemandTargetCapacity": 0,
    "Type": "maintain"
  }
}
```

設定が無効です: 重複する属性値

以下は設定が有効ではありません。2 つの InstanceRequirements 構造がそれぞれ "VCpuCount": {"Min": 0, "Max": 2} を含んでいます。これらの属性の値が重複するため、容量プールが重複します。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "MyLaunchTemplate",
                    "Version": "1"
                },
                "Overrides": [
                {
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 0,
                            "Max": 2
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    },
                    {
                      "InstanceRequirements": {
                          "VCpuCount": {
                              "Min": 0,
                              "Max": 2
```

```
                        },
                        "MemoryMiB": {
                            "Min": 0
                        }
                    }
                }
            }
        ]
    }
],
"TargetCapacity": 1,
"OnDemandTargetCapacity": 0,
"Type": "maintain"
}
}
```

指定された属性でインスタンスタイプをプレビューする

get-instance-types-from-instance-requirements AWS CLI コマンドを使用して、指定した属性に一致するインスタンスタイプをプレビューします。これは、インスタンスを起動せずにリクエスト設定で指定する属性を調べる場合に特に便利です。このコマンドでは、使用可能な容量は考慮されません。

AWS CLI を使用して属性を指定してインスタンスタイプのリストをプレビューするには

1. (オプション) 指定可能なすべての属性を生成するには、get-instance-types-from-instance-requirements コマンドと --generate-cli-skeleton パラメータを使用します。オプションで、input > *attributes.json* を使用して出力を保存用ファイルに送ることができます。

   ```
   aws ec2 get-instance-types-from-instance-requirements \
       --region us-east-1 \
       --generate-cli-skeleton input > attributes.json
   ```

   正常な出力

   ```
   {
       "DryRun": true,
       "ArchitectureTypes": [
           "i386"
       ],
       "VirtualizationTypes": [
           "hvm"
       ],
   ```

```
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 0,
            "Max": 0
        },
        "MemoryMiB": {
            "Min": 0,
            "Max": 0
        },
        "CpuManufacturers": [
            "intel"
        ],
        "MemoryGiBPerVCpu": {
            "Min": 0.0,
            "Max": 0.0
        },
        "ExcludedInstanceTypes": [
            ""
        ],
        "InstanceGenerations": [
            "current"
        ],
        "SpotMaxPricePercentageOverLowestPrice": 0,
        "OnDemandMaxPricePercentageOverLowestPrice": 0,
        "BareMetal": "included",
        "BurstablePerformance": "included",
        "RequireHibernateSupport": true,
        "NetworkInterfaceCount": {
            "Min": 0,
            "Max": 0
        },
        "LocalStorage": "included",
        "LocalStorageTypes": [
            "hdd"
        ],
        "TotalLocalStorageGB": {
            "Min": 0.0,
            "Max": 0.0
        },
        "BaselineEbsBandwidthMbps": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorTypes": [
```

```
            "gpu"
        ],
        "AcceleratorCount": {
            "Min": 0,
            "Max": 0
        },
        "AcceleratorManufacturers": [
            "nvidia"
        ],
        "AcceleratorNames": [
            "a100"
        ],
        "AcceleratorTotalMemoryMiB": {
            "Min": 0,
            "Max": 0
        },
        "NetworkBandwidthGbps": {
            "Min": 0.0,
            "Max": 0.0
        },
        "AllowedInstanceTypes": [
            ""
        ]
    },
    "MaxResults": 0,
    "NextToken": ""
}
```

2. 前のステップの出力を使用して JSON 設定ファイルを作成し、次のように設定します。

> **Note**
>
> `ArchitectureTypes`、`VirtualizationTypes`、`VCpuCount`、および `MemoryMiB` の値を指定する必要があります。その他の属性は省略できます。省略すると、デフォルト値が使用されます。
> 各属性およびそのデフォルト値の説明については、「Amazon EC2 コマンドラインリファレンス」の「get-instance-types-from-instance-requirements」を参照してください。

a. `ArchitectureTypes` に、1 つ以上のタイプのプロセッサアーキテクチャを指定します。

b. VirtualizationTypes に、1 つまたは複数のタイプの仮想化を指定します。

c. VCpuCount に、vCPU の最小数と最大数を指定します。最小制限を指定しない場合は、Min で、0 を指定します。最大制限を指定しない場合は、Max パラメータを省略します。

d. MemoryMiB に、最小値と最大値を MiB 単位で指定します。最小制限を指定しない場合は、Min で、0 を指定します。最大制限を指定しない場合は、Max パラメータを省略します。

e. オプションで、他の属性を 1 つ以上指定して、返されるインスタンスタイプのリストをさらに制約できます。

3. JSON ファイルで指定した属性を持つインスタンスタイプをプレビューするには、get-instance-types-from-instance-requirements コマンドを入力し、--cli-input-json パラメータを使用して、JSON ファイルの名前とパスを指定します。オプションで、出力が表形式で表示されるようにフォーマットできます。

```
aws ec2 get-instance-types-from-instance-requirements \
    --cli-input-json file://attributes.json \
    --output table
```

例: *attributes.json* ファイル

この例では、JSON ファイルに必須属性が含まれています。それらは、ArchitectureTypes、VirtualizationTypes、VCpuCount、および MemoryMiB です。さらに、オプションで InstanceGenerations 属性も含まれます。MemoryMiB では、Max の値を省略し、制限がないことを示すことができることを注意してください。

```
{
    "ArchitectureTypes": [
        "x86_64"
    ],
    "VirtualizationTypes": [
        "hvm"
    ],
    "InstanceRequirements": {
        "VCpuCount": {
            "Min": 4,
            "Max": 6
        },
        "MemoryMiB": {
```

```
            "Min": 2048
        },
        "InstanceGenerations": [
            "current"
        ]
    }
}
```

出力例

```
-----------------------------------------
|GetInstanceTypesFromInstanceRequirements|
+---------------------------------------+
||             InstanceTypes             ||
|+-------------------------------------+|
||             InstanceType              ||
|+-------------------------------------+|
||  c4.xlarge                          ||
||  c5.xlarge                          ||
||  c5a.xlarge                         ||
||  c5ad.xlarge                        ||
||  c5d.xlarge                         ||
||  c5n.xlarge                         ||
||  c6a.xlarge                          ||
...
```

4. ニーズに合ったインスタンスタイプを特定したら、フリートリクエストを設定するときにそれらを使用できるように、使用したインスタンスの属性をメモしておきます。

スポットフリートでのオンデマンド

インスタンス容量を常に確保するには、オンデマンド容量のリクエストをスポットフリートリクエストに含めることができます。スポットフリートリクエストでは、希望するターゲット容量と、その容量のうちオンデマンドである必要がある量を指定します。このバランスは、利用可能な Amazon EC2 キャパシティーと可用性がある場合に起動されるスポットキャパシティーで構成されます。例えば、スポットフリートのリクエストで、ターゲット容量を 10、オンデマンド容量を 8 と指定した場合、Amazon EC2 は 8 容量ユニットをオンデマンドとして、2 容量ユニット (10-8=2) をスポットとして起動します。

オンデマンド容量に基づくインスタンスタイプの優先順位付け

スポットフリートがオンデマンド容量を満たそうとする場合、デフォルトで、最低価格のインスタンスタイプを最初に起動します。OnDemandAllocationStrategy を prioritized に設定すると、スポットフリートは優先順位に従って、オンデマンド容量を満たすために最初に使用するインスタンスタイプを決定します。

優先度は起動テンプレートの上書きに割り当てられ、最も高い優先度が最初に起動されます。

例: インスタンスタイプの優先付け

例えば、3 つの起動テンプレートの上書きに、それぞれ異なるインスタンスタイプを設定したとします。

インスタンスタイプのオンデマンド料金は、幅があります。以下は、この例で使用しているインスタンスタイプで、料金の安いものから順に並んでいます。

- m4.large — 最も安い
- m5.large
- m5a.large

優先度を使って順番を決めない場合、フリートは、最も安いインスタンスタイプから始めてオンデマンドの容量を満たします。

ただし、最初に使用する m5.large リザーブドインスタンスが未使用である場合、次のように、インスタンスタイプが優先度順に使われるように、起動テンプレートの、上書きの優先度を設定できます。

- m5.large – 優先度 1
- m4.large – 優先度 2
- m5a.large – 優先度 3

容量の再調整

Amazon EC2 が再調整に関する推奨を発して、スポットインスタンスが中断リスクが高まっていることを通知したとき、代替スポットインスタンスを起動するようにスポットフリートを設定できます。容量の再調整は、実行中のインスタンスが Amazon EC2 により中断される前に、新しいスポットインスタンスでフリートを事前に拡張することにより、ワークロードの可用性を維持するのに役立ちます。詳細については、「EC2 インスタンスの再調整に関する推奨事項」を参照してください。

代替スポットインスタンスを起動するようにスポットフリートを設定するには、Amazon EC2 コンソールまたは AWS CLI を使用できます。

- Amazon EC2 コンソール: スポットフリートを作成するときは、[容量の再調整] チェックボックスをオンにする必要があります。詳細については、「」のステップ 6。d を参照してください。定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)
- AWS CLI: request-spot-fleet コマンドと SpotMaintenanceStrategies 構造内の関連するパラメーターを使用します。詳細については、「起動設定の例」を参照してください。

制限事項

- 容量の再調整は、タイプ maintain のフリートでのみ使用可能です。
- フリートが実行されているときは、容量の再調整設定を変更できません。容量の再調整設定を変更するには、フリートを削除し、新しいフリートを作成する必要があります。

設定オプション

スポットフリートの ReplacementStrategy では、次の 2 つの値がサポートされています。

`launch-before-terminate`

Amazon EC2 フリートは、新しい置換先スポットインスタンスが起動された後に、再調整通知を受信するスポットインスタンスを終了します。launch-before-terminate を指定する場合は、termination-delay の値も指定する必要があります。新しい置換先インスタンスが起動された後、Amazon EC2 フリートは termination-delay の時間だけ待って、古いインスタンスを終了させます。termination-delay では、最小値は 120 秒 (2 分)、最大値は 7200 秒 (2 時間) です。

launch-before-terminate は、インスタンスのシャットダウン手順が完了するまでの時間が予測できる場合にのみ使用することをお勧めします。これにより、シャットダウン手順が完了した後にのみ、古いインスタンスが確実に終了されます。Amazon EC2 は、termination-delay の前に 2 分間の警告を行い、その後古いインスタンスを中断する可能性があることに注意してください。

また、lowestPrice 配分戦略を launch-before-terminate と組み合わせて使用することは、中断のリスクが高い代替スポットインスタンスを持つことになるため、強く推奨されません。

`launch`

Amazon EC2 フリートは、既存のスポットインスタンスに対して再調整通知が送信されると、置換先スポットインスタンスを起動します。Amazon EC2 フリートは、再調整通知を受け取るインスタンスを終了しません。古いインスタンスを終了することも、実行したままにすることもできます。実行中は、すべてのインスタンスに対して課金されます。

考慮事項

容量の再調整用にスポットフリートを設定する場合は、次の点を考慮してください。

リクエストでは可能な限り多くのスポットキャパシティープールを指定する

複数のインスタンスタイプとアベイラビリティーゾーンを使用するように、スポットフリートを設定します。これにより、さまざまなスポットキャパシティープールでスポットインスタンスを起動するための柔軟性が得られます。詳細については、「インスタンスタイプとアベイラビリティーゾーンについて柔軟に対応する」を参照してください。

代替スポットインスタンスが中断されるリスクの増大を回避する

`lowestPrice` 割り当て戦略を使用している場合、代替スポットインスタンスが中断するリスクが高くなることがあります。これは、置換先スポットインスタンスが起動後すぐに中断される可能性があっても、Amazon EC2 は、その時点で利用可能な容量を持つ最低価格のプールでインスタンスを常に起動するためです。中断のリスクが高くなるのを避けるため、`lowestPrice` アロケーションストラテジー、代わりに推奨する `capacityOptimized` または `capacityOptimizedPrioritized` 配分戦略。これらの戦略により、代替スポットインスタンスが最適なスポットキャパシティプールで起動されるため、近い将来中断される可能性が低くなります。詳細については、「価格と容量を最適化する配分戦略を使用する」を参照してください。

Amazon EC2 は、可用性が同じかそれ以上の場合にのみ、新しいインスタンスを起動します

容量の再調整の目的の 1 つは、スポットインスタンスの可用性を改善することです。既存のスポットインスタンスが再調整のレコメンデーションを受け取った場合、Amazon EC2 は、新しいインスタンスが既存のインスタンスと同等かそれ以上の可用性を提供する場合にのみ新しいインスタンスを起動します。新しいインスタンスの中断のリスクが既存のインスタンスよりもひどい場合、Amazon EC2 は新しいインスタンスを起動しません。ただし、Amazon EC2 は引き続きスポットキャパシティプールを評価し、可用性が向上したら新しいインスタンスを起動します。

Amazon EC2 が新しいインスタンスをプロアクティブに起動しないと、既存のインスタンスが中断する可能性があります。これが発生する場合、Amazon EC2 は、新しいインスタンスの中断リスクが高いかどうかに関らず、新しいインスタンスの起動を試みます。

キャパシティーの再調整は、スポットインスタンスの中断率を増加させるものではありません

キャパシティの再調整を有効にしても、スポットインスタンスの中断率 (Amazon EC2 がキャパシティを取り戻す必要があるときに再利用されるスポットインスタンスの数) は増加しません。ただし、インスタンスに中断のリスクがあることを容量の再調整が検出した場合、Amazon EC2 Auto Scaling は直ちに新しいインスタンスの起動を試みます。その結果、リスクのあるインスタンスが中断された後に Amazon EC2 が新しいインスタンスを起動するのを待つ場合よりも多くのインスタンスが置き換えられる可能性があります。

キャパシティーの再調整が有効になっているインスタンスをさらに置き換える可能性がありますが、インスタンスが中断される前にアクションを実行するための時間をより長く確保できるため、事後対応ではなくプロアクティブに対応できるというメリットがあります。スポットインスタンスの中断通知では、通常、インスタンスを正常にシャットダウンするための猶予期間が最大 2 分しかありません。キャパシティーの再調整で新しいインスタンスを事前に起動することで、既存のプロセスがリスクのあるインスタンスで完了する可能性が高くなり、インスタンスのシャットダウン手順を開始して、リスクのあるインスタンスで新しい作業がスケジュールされないようにできます。新しく起動したインスタンスの準備を開始して、アプリケーションを引き継ぐこともできます。キャパシティーの再調整のプロアクティブな置き換えにより、正常な継続性の恩恵を受けることができます。

キャパシティーの再調整を使用するリスクとメリットを示す理論的な例として、次のシナリオを検討してください。

- 午後 2 時 – インスタンス A の再調整の推奨が受信され、Amazon EC2 は直ちに置換先インスタンス B の起動の試行を開始するため、シャットダウン手順を開始する時間を確保できます。*
- 午後 2 時 30 分 – インスタンス B の再調整の推奨が受信され、インスタンス C に置き換えられるため、シャットダウン手順を開始する時間を確保できます。*
- 午後 2 時 32 分 – キャパシティーの再調整が有効になっておらず、インスタンス A のスポットインスタンスの中断通知が午後 2 時 32 分に受信されていたとすれば、アクションを実行するための猶予期間は最大でも 2 分だけでしたが、インスタンス A はこの時間まで稼働していたことでしょう。

* `launch-before-terminate` が指定されている場合、Amazon EC2 は、置換先インスタンスがオンラインになった後、リスクのあるインスタンスを終了します。

Amazon EC2 フリート は、満たされた容量がターゲット容量の 2 倍になるまで、新しい置換先スポットインスタンスを起動できます

スポットフリートが容量の再調整用に設定されている場合、Amazon EC2 は、再調整に関する推奨を受け取るすべてのスポットインスタンスに対して、新しい置換先スポットインスタンスを起動しようとします。スポットインスタンスが再調整勧告を受け取った後は、満たされた容量の一部としてカウントされなくなります。交換戦略に応じて、Amazon EC2 は事前設定された終了遅延の後にインスタンスを終了するか、インスタンスを実行のままにします。これにより、インスタンスで 再調整アクション を実行できるようになります。

フリートがターゲットキャパシティの 2 倍に達すると、代替インスタンス自体が再調整に関する推奨事項を受け取った場合でも、新しい代替インスタンスの起動を停止します。

例えば、100 個のスポットインスタンスのターゲット容量を持つスポットフリートを作成するとします。すべてのスポットインスタンスは、再調整に関するレコメンデーションを受け取ります。これにより、Amazon EC2 は 100 個の置換先スポットインスタンスを起動します。これにより、満たされたスポットインスタンスの数が 200 になり、ターゲットキャパシティの 2 倍になります。一部の代替インスタンスは再調整に関する推奨事項を受け取りますが、フリートがターゲット容量の 2 倍を超えることができないため、代替インスタンスはそれ以上起動されません。

インスタンスの実行中は、すべてのインスタンスに対して課金されることに注意してください。

再調整通知を受け取ったスポットインスタンスを終了させるようにスポットフリートを設定することをお勧めします

容量再調整のためにスポットフリートを設定する場合、インスタンスのシャットダウン手順が完了するまでの時間を予測できる場合に限り、適切な終了遅延を持つ `launch-before-terminate` を選択することをお勧めします。これにより、シャットダウン手順が完了した後にのみ、古いインスタンスが確実に終了されます。

再調整のために推奨されるインスタンスを終了する場合は、フリートのスポットインスタンスが受信する再調整レコメンデーションシグナルをモニタリングすることをお勧めします。シグナルをモニタリングすることで、Amazon EC2 が中断する前に、影響を受けるインスタンスで 再調整のアクション をすばやく実行し、手動で終了できます。インスタンスを終了しない場合、インスタンスの実行中、課金が継続します。Amazon EC2 は、再調整に関する推奨を受け取るインスタンスを自動的に終了しません。

Amazon EventBridge またはインスタンスメタデータを使用して通知を設定できます。詳細については、「再調整に関する推奨事項シグナルのモニタリング」を参照してください。

スポットフリートは、スケールインまたはスケールアウト中に満たされた容量を計算するとき、再調整に関する推奨を受け取るインスタンスはカウントしません

容量の再調整のためにスポットフリートを設定し、ターゲット容量をスケールインまたはスケールアウトするように変更した場合、フリートは次のように、再調整の対象としてマークされたインスタンスを、満たされた容量の一部としてカウントしません。

- スケールイン – 希望するターゲット容量を減らすと、Amazon EC2 は目的の容量に達するまで、再調整の対象としてマークされていないインスタンスを終了します。再調整の対象としてマークされたインスタンスは、満たされた容量にはカウントされません。

  例えば、100 個のスポットインスタンスをターゲット容量を持つスポットフリートを作成するとします。10 個のインスタンスは再調整に関する推奨事項を受け取ります。そのため、Amazon EC2 は 10 個の新しい代替インスタンスを起動し、その結果、110 個のインスタンスの容量が満たされます。その後、ターゲット容量を 50 個に減らしますが (スケールイン)、再調整の対象としてマークされた 10 個のインスタンスは Amazon EC2 によって終了されないため、満たされた容量は実際には 60 インスタンスになります。このようなインスタンスは手動で終了する必要があります。または、実行したままにしておくことができます。

- スケールアウト – 目的のターゲット容量を増やすと、目的の容量に達するまで Amazon EC2 は新しいインスタンスを起動します。再調整の対象としてマークされたインスタンスは、満たされた容量にはカウントされません。

  例えば、100 個のスポットインスタンスをターゲット容量を持つスポットフリートを作成するとします。10 個のインスタンスは再調整に関する推奨事項を受け取ります。そのため、Amazon EC2 は 10 個の新しい代替インスタンスを起動し、その結果、110 個のインスタンスの容量が満たされます。その後、ターゲット容量を 200 個に増やし (スケールアウトし) ますが、再調整の対象としてマークされた 10 個のインスタンスは、フリートによってターゲット容量の一部としてカウントされないため、実際には 210 個のインスタンスになります。このようなインスタンスは手動で終了する必要があります。または、実行したままにしておくことができます。

スポット料金の優先

各スポットフリートは、グローバルな上限料金を含めるか、デフォルト (オンデマンド価格) を使用できます。スポットフリートは、これを起動仕様のデフォルト上限料金として使用します。

任意で 1 つまたは複数の起動条件に上限料金を指定することができます。これは、起動条件に指定された料金です。起動仕様に特定の料金が含まれる場合、スポットフリートはこの起動仕様の上限料

金を使用し、グローバル上限料金に優先することになります。特定の上限料金を含まないそのほかの起動条件は、全体の上限料金を引き続き使用することにご注意ください。

使用量の管理

ターゲット容量または支払い上限料金に達すると、スポットフリートはインスタンスの起動を停止します。フリートに支払う 1 時間あたりの料金を管理するには、スポット インスタンスの場合は SpotMaxTotalPrice を、オンデマンド インスタンスの場合は OnDemandMaxTotalPrice を指定できます。上限の合計料金に達すると、ターゲット容量に満たない場合でも、スポットフリートはインスタンスの起動を停止します。

以下の例は、2 つの異なるシナリオを示しています。最初の例では、ターゲット容量に達すると、スポットフリートはインスタンスの起動を停止します。2 番目の例では、支払い上限料金に達すると、スポットフリートはインスタンスの起動を停止します。

例: ターゲット容量に達したときにインスタンスの起動を停止する

m4.large オンデマンドインスタンス に対するリクエストの内容が以下のとおりとします。

- オンデマンド料金: 1 時間あたり 0.10 USD
- OnDemandTargetCapacity: 10
- OnDemandMaxTotalPrice: 1.50 USD

スポットフリートは 10 個のオンデマンドインスタンスを起動します。合計料金 1.00 USD (10 インスタンス x 0.10 USD) は、 の 1.50 USD を超えないためです。OnDemandMaxTotalPrice

例: 最大の合計料金に達したときにインスタンスの起動を停止する

m4.large オンデマンドインスタンス に対するリクエストの内容が以下のとおりとします。

- オンデマンド料金: 1 時間あたり 0.10 USD
- OnDemandTargetCapacity: 10
- OnDemandMaxTotalPrice: 0.80 USD

スポットフリートがオンデマンドターゲット容量 (10 オンデマンドインスタンス) を起動した場合、1 時間あたりの合計料金は 1.00 USD になります。これは OnDemandMaxTotalPrice に指定した料金 (0.80 USD) を超えます。支払い可能な額を超えないように、スポットフリートは 8 オンデ

マンドインスタンス (オンデマンドターゲット容量未満) だけを起動します。これ以上起動すると、OnDemandMaxTotalPrice を超えるためです。

スポットフリートインスタンスの分量指定

スポットインスタンス のフリートをリクエストする際に、それぞれのインスタンスタイプがアプリケーションのパフォーマンスに貢献するように容量ユニットを定義し、また、インスタンスの分量指定を利用してスポットキャパシティープールごとに上限価格を調整できます。

デフォルトでは、指定する料金は 1 インスタンス時間あたりの料金となります。インスタンスの分量指定機能を使用すると、指定した料金は ユニット時間ごと の料金となります。ユニット時間あたりの使用料金はインスタンスタイプの料金を対応するユニット数で割って計算できます。スポットフリートは、ターゲット容量をインスタンス分量で割ることで、起動するインスタンス数を計算します。結果が整数でない場合、スポットフリートはその数を次の整数に切り上げ、そのためフリートのサイズがターゲット容量以上になります。起動されたインスタンスの容量がリクエストされたターゲット容量を超えた場合でも、スポットフリートは起動仕様で指定したどのプールでも選択できます。

以下の表では、スポットフリートリクエストのターゲット容量が 10 の場合の、単位あたりの料金を計算する例を示します。

インスタンスタイプ	インスタンスの分量	インスタンス時間あたりのスポット料金	ユニット時間あたりの価格	起動されたインスタンスの数
r3.xlarge	2	0.05 USD	.025 (.05 ÷ 2)	5 (10 ÷ 2)

インスタンスタイプ	インスタンスの分量	インスタンス時間あたりのスポット料金	ユニット時間あたりの価格	起動されたインスタンスの数
r3.8xlarge	8	0.10 USD	.0125 (.10 ÷ 8)	2 (10 ÷ 8、結果切り上げ)

次のように、スポットフリートのインスタンスの分量指定を使用して、受理時に単位ごとの最低価格のプールに指定するターゲット容量をプロビジョニングします。

1. スポットフリートのターゲット容量を、インスタンス (デフォルト) または仮想 CPU、メモリ、ストレージまたはスループットなどご希望のユニットで設定します。
2. ユニットあたりの料金を設定します。
3. 各起動設定で、インスタンスタイプがターゲット容量に対して必要なユニット数である分量を指定します。

インスタンスの分量指定例

次の設定のスポットフリートリクエストの場合を考えます。

- ターゲット容量 24
- `r3.2xlarge` のインスタンスタイプの起動条件と分量 6
- `c3.xlarge` のインスタンスタイプの起動条件と分量 5

分量とは、インスタンスタイプがターゲット容量に対して必要なユニット数を表します。最初の起動条件がユニットあたりの料金を最低値で提供する場合 (インスタンス時間あたりの `r3.2xlarge` の料金を 6 で割ったもの)、スポットフリートはこれらのインスタンスから 4 つを起動します (24 を 6 で割ったもの)。

2 番目の起動仕様がユニットあたりの最低料金を提供する場合 (インスタンス時間あたりの `c3.xlarge` の料金を 5 で割ったもの)、スポットフリートはこれらのインスタンスから 5 個を起動します (24 を 5 で割ったもの、結果は切り上げられる)。

インスタンスの分量指定と配分戦略

次の設定のスポットフリートリクエストの場合を考えます。

- ターゲット容量 30
- `c3.2xlarge` のインスタンスタイプの起動条件と分量 8
- `m3.xlarge` のインスタンスタイプの起動条件と分量 8
- `r3.xlarge` のインスタンスタイプの起動条件と分量 8

スポットフリートは、4 個のインスタンスを起動します (30 を 8 で割ったもの、結果を切り上げ)。`lowestPrice` 戦略では、すべての 4 つのインスタンスはユニットあたりの最低価格を提供す

るプールから取得されます。diversified 戦略では、スポットフリート は 3 プールごとに 1 インスタンスを起動し、4 つ目のインスタンスはいずれかのプールで、単位あたりの最低価格を提供します。

スポットフリートの操作

スポットフリートを使用するには、ターゲット容量、オプションでオンデマンド部分、インスタンスの 1 つ以上の起動仕様、希望上限価格を含めたスポットフリートリクエストを作成します。フリートリクエストには、フリートがインスタンスの起動に必要とする情報 (AMI、インスタンスタイプ、サブネットまたはアベイラビリティーゾーン、そして 1 つ以上のセキュリティグループ) を定義する起動仕様を含める必要があります。

フリートに スポットインスタンス が含まれている場合、Amazon EC2 はスポット料金の変更に応じてフリートのターゲット容量を維持しようと試みることができます。

送信後にワンタイムリクエストのターゲット容量を変更することはできません。ターゲット容量を変更するには、リクエストを変更し、新しいリクエストを送信します。

スポットフリートリクエストは、期限切れになるかキャンセルされるまで、アクティブな状態を維持します。フリートリクエストをキャンセルするとき、フリートのスポットインスタンスをキャンセルするか、終了するか、どちらかを指定できます。

コンテンツ

スポットフリートリクエストの状態

スポットフリートリクエストは、次の状態のいずれかになります。

- submitted - スポットフリートリクエストは評価中です。Amazon EC2 はターゲット数のインスタンスの起動を準備中です。スポットフリートの上限を超えたリクエストは、即時キャンセルされます。
- active - スポットフリートは検証済みです。Amazon EC2 はターゲット数の実行中のスポットインスタンスを維持しようとしています。リクエストは、変更またはキャンセルされるまで、この状態のままになります。
- modifying - スポットフリートリクエストは変更中です。リクエストは、変更が完全に処理されるか、スポットフリートがキャンセルされるまで、この状態を維持します。ワンタイム request を変更することはできません。この状態は、そのようなスポットリクエストには適用されません。
- cancelled_running - スポットフリートはキャンセルされ、追加のスポットインスタンスを起動しません。その既存の スポットインスタンス は、中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。
- cancelled_terminating - スポットフリートはキャンセルされ、スポットインスタンスは終了します。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。
- cancelled - スポットフリートはキャンセルされ、実行中のスポットインスタンスはありません。スポットリクエストは、インスタンスが終了して 2 日後に削除されます。

スポットフリートのヘルスチェック

スポットフリートは、2 分ごとにフリートのスポットインスタンスのヘルスステータスをチェックします。インスタンスのヘルスステータスは healthy または unhealthy です。

スポットフリートは、Amazon EC2 が提供するステータスチェックを使用して、インスタンスのヘルスステータスを判断します。インスタンスステータスとシステムステータスのいずれかのチェック結果において、ステータスが 3 回連続して impaired を示した場合、そのインスタンスは unhealthy と判断されます。詳細については、「インスタンスのステータスチェック」を参照してください。

フリートを設定して、異常のある スポットインスタンス を置き換えることができます。ヘルスチェックによる置き換えを有効化すると、 と報告されたスポットインスタンスが置き換えられます。unhealthy異常なスポットインスタンスの置き換え中、最大数分間フリートがターゲット容量を下回る場合があります。

要件

- ヘルスチェックによる置き換えは、1 回限りの スポットフリート (maintain のフリート) ではなく、ターゲットキャパシティを維持しているスポットフリート (タイプ request のフリート) でのみサポートされます。
- ヘルスチェックによる置き換えは、スポットインスタンス でのみサポートされます。この機能はオンデマンドインスタンス ではサポートされていません。
- 作成時のみ、異常なインスタンスを置き換えるようスポットフリートを設定できます。
- ユーザーは、ec2:DescribeInstanceStatus アクションを呼び出す許可を持っている場合のみ、ヘルスチェックの置き換えを使用できます。

Console

コンソールを使用して、異常なスポットインスタンスを置き換えるようにスポットフリートを設定するには

1. 手順に従ってスポットフリートを作成します。詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)」を参照してください。
2. 異常のある スポットインスタンス を置き換えるようにフリートを設定するには、[ヘルスチェック] で [異常のあるインスタンスの置き換え] を選択します。このオプションを有効にするには、まず [Maintain target capacity](ターゲット容量の維持) を選択する必要があります。

AWS CLI

AWS CLI を使用して、異常なスポットインスタンスを置き換えるようにスポットフリートを設定するには

1. 手順に従ってスポットフリートを作成します。詳細については、「スポットフリートを作成するにはAWS CLI」を参照してください。
2. 異常のあるスポットインスタンスを置き換えるようにフリートを設定するには、ReplaceUnhealthyInstances に true と入力します。

スポットフリートアクセス許可

ユーザーがスポットフリートを作成または管理する場合、必要な許可を付与する必要があります。

Amazon EC2 コンソールを使用してスポットフリートを作成した場合、AWSServiceRoleForEC2SpotFleet および AWSServiceRoleForEC2Spot というサービスにリンクされた 2 つのロールと、aws-ec2-spot-fleet-tagging-role というロールが作成されます。ユーザーの代わりに、リソースのリクエスト、起動、終了、タグ付けを行うアクセス許可をスポットフリートに与えます。AWS CLI または API を使用する場合は、これらのロールが存在することを確認する必要があります。

次の手順に従って、必要なアクセス許可を付与し、ロールを作成します。

アクセス許可とロール

- ユーザーにスポットフリートの許可を付与する
- スポットフリート用のサービスにリンクされたロール
- スポットインスタンス用のサービスにリンクされたロール
- スポットフリートにタグ付けするための IAM ロール

ユーザーにスポットフリートの許可を付与する

ユーザーがスポットフリートを作成または管理する場合、必ず必要な許可を付与してください。

スポットフリートのポリシーを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで、[Policies]、[Create policy] の順に選択します。
3. [ポリシーの作成] ページで、 [JSON] を選択し、テキストを以下に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances",
                "ec2:CreateTags",
                "ec2:RequestSpotFleet",
                "ec2:ModifySpotFleetRequest",
                "ec2:CancelSpotFleetRequests",
                "ec2:DescribeSpotFleetRequests",
                "ec2:DescribeSpotFleetInstances",
                "ec2:DescribeSpotFleetRequestHistory"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::*:role/aws-ec2-spot-fleet-tagging-role"
        },
        {
            "Effect": "Allow",
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:ListRoles",
                "iam:ListInstanceProfiles"
            ],
            "Resource": "*"
        }
    ]
}
```

前述したポリシーの例では、ほとんどのスポットフリートのユースケースで必要な許可をユーザーに付与します。特定の API アクションに制限するには、代わりにこれらの API アクションのみを指定します。

必要な EC2 および IAM の API

ポリシーには、次の API を含める必要があります。

- `ec2:RunInstances` - スポットフリートでインスタンスを起動するために必要
- `ec2:CreateTags` - スポットフリートのリクエスト、インスタンス、またはボリュームのタグ付けに必要
- `iam:PassRole` - スポットフリートロールを指定するために必要
- `iam:CreateServiceLinkedRole` - サービスにリンクされたロールの作成に必要
- `iam:ListRoles` - 既存の IAM ロールを列挙するために必要
- `iam:ListInstanceProfiles` - 既存のインスタンスプロファイルを列挙するために必要

> **Important**
>
> 起動仕様または起動テンプレートで IAM インスタンスプロファイルのロールを指定する場合は、そのロールをサービスに渡す許可をユーザーに付与する必要があります。これを行うには、IAM ポリシーで `iam:PassRole` アクションのリソースとして `"arn:aws:iam::*:role/IamInstanceProfile-role"` を含めます。詳細については、「IAM ユーザーガイド」の「AWS サービスにロールを渡すアクセス権限をユーザーに付与する」を参照してください。

スポットフリートの API

必要に応じて、次のスポットフリート API アクションをポリシーに追加します。

- `ec2:RequestSpotFleet`
- `ec2:ModifySpotFleetRequest`
- `ec2:CancelSpotFleetRequests`
- `ec2:DescribeSpotFleetRequests`
- `ec2:DescribeSpotFleetInstances`
- `ec2:DescribeSpotFleetRequestHistory`

オプションの IAM API

(オプション) ユーザーが IAM コンソールを使用してロールまたはインスタンスプロファイルを作成できるようにするには、次のアクションをポリシーに追加する必要があります。

- `iam:AddRoleToInstanceProfile`
- `iam:AttachRolePolicy`
- `iam:CreateInstanceProfile`
- `iam:CreateRole`
- `iam:GetRole`
- `iam:ListPolicies`

4. [ポリシーの確認] を選択します。
5. [ポリシーの確認] ページでポリシー名と説明を入力し、[ポリシーの作成] を選択します。

6. アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

   - AWS IAM Identity Center のユーザーとグループ:

     アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」の手順に従ってください。

   - IAM 内で、ID プロバイダーによって管理されているユーザー:

     ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

   - IAM ユーザー:

     - ユーザーが担当できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
     - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

スポットフリート用のサービスにリンクされたロール

Amazon EC2 は、ユーザーに代わって AWS の他のサービスを呼び出すために必要なアクセス許可のために、サービスにリンクされたロールを使用します。サービスにリンクされたロールは、AWS のサービスに直接リンクされた一意のタイプの IAM ロールです。サービスにリンクされたロールは、AWS のサービスにアクセス許可を委任するためのセキュアな方法を提供します。これは、リンクされたサービスのみが、サービスにリンクされたロールを引き受けることができるためです。詳細については、「 IAM ユーザーガイド」の「サービスにリンクされたロールの使用」を参照してください。

Amazon EC2 は、AWSServiceRoleForEC2SpotFleet という、サービスにリンクされたロールを使用して、ユーザーの代わりにインスタンスを起動して管理します。

> **Important**
>
> 暗号化された AMI または暗号化された Amazon EBS スナップショットをスポットフリートで指定した場合は、CMK を使用して Amazon EC2 がユーザーの代わりにインスタンスを起動する許可を AWSServiceRoleForEC2SpotFleet ロールに付与する必要があります。詳細に

ついては、「暗号化された AMI および EBS スナップショット用の CMK へのアクセス権の付与」を参照してください。

AWSServiceRoleForEC2SpotFleet によって付与されるアクセス許可

Amazon EC2 は、AWSServiceRoleForEC2SpotFleet という、サービスにリンクされたロールを使用して、次のアクションを実行します。

- `ec2:RequestSpotInstances` - スポットインスタンスをリクエスト
- `ec2:RunInstances` - インスタンスを起動
- `ec2:TerminateInstances` - インスタンスを終了
- `ec2:DescribeImages` - インスタンスの Amazon マシンイメージ (AMI) を表示
- `ec2:DescribeInstanceStatus` - インスタンスのステータスを表示
- `ec2:DescribeSubnets` - インスタンスのサブネットを記述
- `ec2:CreateTags` - スポットフリートリクエスト、インスタンス、ボリュームにタグを追加
- `elasticloadbalancing:RegisterInstancesWithLoadBalancer` - 指定されたインスタンスを指定されたロードバランサーに追加
- `elasticloadbalancing:RegisterTargets` - 指定されたターゲットを指定されたターゲットグループに登録

サービスにリンクされたロールの作成

ほとんどの状況では、サービスにリンクされたロールを手動で作成する必要はありません。Amazon EC2 は、コンソールを使用してスポットフリートを初めて作成するときに、AWSServiceRoleForEC2SpotFleet サービスにリンクされたロールを作成します。

Amazon EC2 がこのサービスにリンクされたロールのサポートを開始した 2017 年 10 月よりも前にアクティブなスポットフリートリクエストを行った場合、Amazon EC2は AWS アカウントで AWSServiceRoleForEC2SpotFleet ロールを作成します。詳細については、IAM ユーザーガイドの「 アカウントに新しいロールが表示される」を参照してください。AWS

AWS CLI または API を使用してスポットフリートを作成する場合、最初にこのロールが存在しているか確認する必要があります。

コンソールを使用して AWSServiceRoleForEC2SpotFleet を作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで Roles (ロール) を選択します。
3. [Create role] を選択します。
4. [信頼されたエンティティを選択] ページで、以下の操作を実行します。
   a. [信頼できるエンティティタイプ] で、[AWS サービス] を選択します。
   b. [ユースケース] の [サービスまたはユースケース] で、[EC2] を選択します。
   c. [ユースケース] で、[EC2 - スポットフリート] を選択します。
   d. [Next] を選択します。
5. [アクセス許可を追加] ページで [次へ] を選択します。
6. [名前、確認、および作成] ページで、[ロールの作成] をクリックします。

AWS CLI を使用して AWSServiceRoleForEC2SpotFleet を作成するには

次のように、create-service-linked-role コマンドを使用します。https://docs.aws.amazon.com/cli/latest/reference/iam/create-service-linked-role.html

```
aws iam create-service-linked-role --aws-service-name spotfleet.amazonaws.com
```

スポットフリートを使用する必要がなくなった場合は、[AWSServiceRoleForEC2Fleet] ロールを削除することをお勧めします。このロールがアカウントから削除された後、コンソールを使用してスポットフリートをリクエストすると、Amazon EC2 はロールを再作成します。詳細については、 IAM ユーザーガイド の「サービスにリンクされたロールの削除」を参照してください。

暗号化された AMI および EBS スナップショット用の CMK へのアクセス権の付与

暗号化された AMI または暗号化された Amazon EBS スナップショットをスポットフリートリクエストで指定し、カスタマーマネージド型キーを暗号化に使用する場合は、CMK を使用して、Amazon EC2 がユーザーの代わりにインスタンスを起動する許可を、AWSServiceRoleForEC2SpotFleet ロールに付与する必要があります。これを行うには、次の手順で示すように、CMK に付与を追加する必要があります。

アクセス権限を設定するときは、付与がキーポリシーの代わりになります。詳細については、 デベロッパーガイドの「許可の使用」と「 でのキーポリシーの使用」を参照してください。https://

docs.aws.amazon.com/kms/latest/developerguide/grants.htmlAWS KMSAWS Key Management Service

CMK を使用するアクセス許可を AWSServiceRoleForEC2SpotFleet ロールに付与するには

- create-grant コマンドを使用して CMK に付与を追加し、プリンシパル (サービスにリンクされたロール AWSServiceRoleForEC2SpotFleet) を指定します。このプリンシパルには、付与が許可するオペレーションを実行するためのアクセス許可が提供されます。https://docs.aws.amazon.com/cli/latest/reference/kms/create-grant.htmlCMK を指定するには、 パラメータと CMK の ARN を使用します。`key-id`プリンシパルを指定するには、 パラメータとサービスにリンクされたロール AWSServiceRoleForEC2SpotFleet の ARN を使用します。`grantee-principal`

```
aws kms create-grant \
    --region us-east-1 \
    --key-id arn:aws:kms:us-
east-1:444455556666:key/1234abcd-12ab-34cd-56ef-1234567890ab \
    --grantee-principal arn:aws:iam::111122223333:role/
AWSServiceRoleForEC2SpotFleet \
    --operations "Decrypt" "Encrypt" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "CreateGrant" "DescribeKey" "ReEncryptFrom"
 "ReEncryptTo"
```

スポットインスタンス用のサービスにリンクされたロール

Amazon EC2 は、AWSServiceRoleForEC2Spot という、サービスにリンクされたロールを使用して、ユーザーの代わりに スポットインスタンス を起動して管理します。詳細については、「スポットインスタンスリクエスト向けのサービスにリンクされたロール」を参照してください。

スポットフリートにタグ付けするための IAM ロール

`aws-ec2-spot-fleet-tagging-role` IAM ロールは、スポットフリートリクエスト、インスタンス、ボリュームにタグ付けするアクセス権限をスポットフリートに付与します。詳細については、「スポットフリートにタグ付けします。」を参照してください。

> **Important**
>
> フリートのインスタンスにタグ付けすることを選択し、ターゲット容量を維持することを選択した場合 (スポットフリートリクエストのタイプは `maintain`)、ユーザーと `IamFleetRole` の許可の違いにより、フリートのインスタンスのタグ付け動作に整合性が

なくなる可能性があります。IamFleetRole に CreateTags アクセス許可が含まれていない場合、フリートによって起動されたインスタンスの一部がタグ付けされていない可能性があります。当社はこの不整合の修正に取り組んでいますが、フリートによって起動されたすべてのインスタンスがタグ付けされるようにするために、IamFleetRoleにはaws-ec2-spot-fleet-tagging-roleロールを使用することをお勧めします。または、既存のロールを使用するには、AmazonEC2SpotFleetTaggingRole の AWS 管理ポリシーを既存のロールにアタッチします。それ以外の場合は、既存のポリシーに CreateTags アクセス許可を手動で追加する必要があります。

スポットフリートにタグ付けする IAM ロールを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで Roles (ロール) を選択します。
3. [Create role] を選択します。
4. [Select trusted entity] (信頼できるエンティティの選択) ページの [Trusted entity type] (信頼できるエンティティタイプ) で、[AWS service] (のサービス) を選択します。
5. [Use case] (ユースケース) で、[Use cases for other AWS services] (他の サービスでのユースケース) から [EC2] を選択し、[EC2 - Spot Fleet Tagging] (EC2 - スポットフリートのタグ付け) を選択します。
6. [Next] を選択します。
7. [アクセス許可を追加] ページで [次へ] を選択します。
8. [Name, review, and create] (名前、レビュー、および作成) ページで、[Role name] (ロール名) にロールの名前 (例えば、**aws-ec2-spot-fleet-tagging-role**) を入力します。
9. ページ内の情報を確認し、[Create role] (ロールを作成) をクリックします。

サービス間の混乱した代理の防止

「混乱した代理」問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。aws-ec2-spot-fleet-tagging-role 信頼ポリシー内のグローバル条件コンテキストキー aws:SourceArn と aws:SourceAccount を使用して、リソースについてスポットフリートが別のサービスに付与するアクセス許可を、制限することをお勧めします。

aws:SourceArn および aws:SourceAccount 条件キーを **aws-ec2-spot-fleet-tagging-role** 信頼ポリシーに追加するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。
2. ナビゲーションペインで Roles (ロール) を選択します。
3. 前に作成してある aws-ec2-spot-fleet-tagging-role を見つけ、リンク (チェックボックスではありません) をクリックします。
4. [Summary] (概要) にある [Trust relationships] (信頼関係) タブを開き、[Edit trust policy] (信頼ポリシーの編集) をクリックします。
5. 「混乱した代理」問題を防止するために、JSON ステートメント内で、以下のようにグローバル条件コンテキストキー aws:SourceAccount および aws:SourceArn を含む Condition 要素を追加します。

```
"Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ec2:us-east-1:account_id:spot-fleet-request/sfr-
*"
      },
      "StringEquals": {
        "aws:SourceAccount": "account_id"
      }
```

> **Note**
>
> aws:SourceArn の値にアカウント ID が含まれており、上記のグローバル条件コンテキストキーの両方を同じポリシーステートメント内で使用する場合、aws:SourceAccount 値と aws:SourceArn 値の中のアカウントには、同じアカウント ID を使用する必要があります。

最終的な信頼ポリシーは次のようになります。

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
```

```
      "Service": "spotfleet.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": "arn:aws:ec2:us-east-1:account_id:spot-fleet-request/sfr-
*"
      },
      "StringEquals": {
        "aws:SourceAccount": "account_id"
      }
    }
  }
}
```

6. [ポリシーの更新] を選択します。

次の表に、aws-ec2-spot-fleet-tagging-role の範囲を制限するために想定される aws:SourceArn の値を、その特異性の様々なレベルについてまとめました。

API オペレーション	呼び出されたサービス	スコープ	**aws:SourceArn**
RequestSpotFleet	AWS STS (AssumeRole)	aws-ec2-spot-fleet-tagging-role が持つ AssumeRole の機能を、指定されたアカウントの spot-fleet-requests に制限します。	arn:aws:ec2:*: *123456789012* :spot-fleet-request/sfr-*
RequestSpotFleet	AWS STS (AssumeRole)	aws-ec2-spot-fleet-tagging-role が持つ AssumeRole の機能を、指定されたアカウントおよび指定されたリージョンの	arn:aws:ec2: *us-east-1* :*123456789012* :spot-fleet-request/sfr-*

API オペレーション	呼び出されたサービス	スコープ	**aws:SourceArn**
		spot-fleet-requests に制限します。このロールは他のリージョンでは使用できないことに注意してください。	
RequestSpotFleet	AWS STS (AssumeRole)	aws-ec2-spot-fleet-tagging-role が持つ AssumeRole の機能を、フリート sfr-11111111-1111-1111-1111-111111111111 に影響を与えるアクションのみに制限します。このロールは、他のスポットフリートでは使用できない場合があることに注意してください。また、このロールを使用して request-spot-fleet により新しいスポットフリートを起動することはできません。	arn:aws:ec2: *us-east-1* :*123456789012* :spot-fleet-request/sfr-*11111111-1111-1111-1111-111111111111*

スポットフリートリクエストを作成します。

AWS Management Console を使用して、アプリケーションまたはタスクのニーズと最低限のコンピューティング仕様のみを選択して、スポットフリートを迅速に作成します。Amazon EC2 は、ユーザーのニーズに最適なフリートを設定し、スポットベストプラクティスに従います。詳細については、「スポットフリートリクエストを迅速に作成します (コンソール)」を参照してください。それ

以外の場合は、デフォルト設定のいれかを変更できます。詳細については、定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)およびスポットフリートを作成するにはAWS CLIを参照してください。

スポットフリートを作成するためのオプション

- スポットフリートリクエストを迅速に作成します (コンソール)
- 定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)
- スポットフリートを作成するにはAWS CLI

スポットフリートリクエストを迅速に作成します (コンソール)

以下の手順に従って、スポットフリートリクエストを迅速に作成します。

推奨設定を使用してスポットフリートリクエストを作成するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットを初めて使用する場合は、ウェルカムページが表示されるので、そこで [Get started] を選択します。それ以外の場合は、[Request Spot Instances] (スポットインスタンスのリクエスト) を選択します。
4. [Launch parameters] (起動パラメータ) で、[Manually configure launch parameters] (起動パラメータを手動で構成する) を選択します。
5. AMI で、AMI を選択します。
6. [Target capacity] (ターゲット容量) の下の [Total target capacity] (総ターゲット容量) で、リクエストする単位数を指定します。ユニットのタイプには、[Instances] (ユニット)、[vCPU]、または [Memory (MiB)] (メモリ (MiB)) を選択できます。
7. [Your fleet request at a glance] (フリートリクエストの概要) で、フリートの設定を確認し、[Launch] (起動) を選択します。

定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)

定義済みパラメータを使用して、スポットフリートを作成できます。

定義済みパラメータを使用してスポットフリートリクエストを作成するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットを初めて使用する場合は、ウェルカムページが表示されるので、そこで [Get started] を選択します。それ以外の場合は、[Request Spot Instances] (スポットインスタンスのリクエスト) を選択します。
4. [Launch parameters] (起動パラメータ) では、以下の操作を行います。

   a. スポットコンソールで起動パラメータを定義するには、[Manually configure launch parameters] (起動パラメータを手動で構成する) を選択します。

   b. [AMI] で、AWS が提供する基本 AMI のいずれかを選択します。あるいは、[Search for AMI] (AMI を検索) をクリックして、ユーザーコミュニティの AMI、AWS Marketplace、または独自の AMI を選択することも可能です。

      > **Note**
      >
      > 起動パラメータで指定された AMI が登録解除または無効になっている場合、AMI から新しいインスタンスを起動することはできません。ターゲット容量を維持するように設定されたフリートの場合、ターゲット容量は維持されません。

   c. (オプション) [Key pair name] で、既存のキーペアを使用するか、新しいキーペアを作成するかを選択します。

      [既存のキーペア] キーペアを選択します。

      [新しいキーペア] [Create new key pair] (新しいキーペアの作成) を選択して [Key Pairs] (キーペア) ページに進みます。完了したら、[Spot Requests] (スポットリクエスト) ページに戻ってリストを更新します。

   d. (オプション) [Additional launch parameters] (追加の起動パラメータ) を展開し、次の操作を実行します。

      i. (オプション) Amazon EBS 最適化を有効にするには、[EBS-optimized] (EBS に最適化された) で [Launch EBS-optimized instances] (EBS に最適化されたインスタンスの起動) を選択します。

      ii. (オプション) インスタンス用の一時ブロックレベルストレージを追加するには、[Instance store] (インスタンスストア) で [Attach at launch] (起動時にアタッチ) を選択します。

iii. (オプション) ストレージを追加するには、[Add new volume] (新しいボリュームの追加) を選択し、インスタンスタイプに応じて追加のインスタンスストアボリュームまたは Amazon EBS ボリュームを指定します。

iv. (オプション) デフォルトでは、インスタンスに対して基本モニタリングが有効になります。詳細モニタリングを有効にするには、[Monitoring] (モニタリング) で [Enable CloudWatch detailed monitoring] (CloudWatch 詳細モニタリングの有効化) を選択します。

v. (オプション) 専有スポットインスタンスを実行するには、[Tenancy] (テナンシー) で [Dedicated - run a dedicated instance] (専有 - 専有インスタンスの実行) を選択します。

vi. (オプション) [Security groups] で、1 つ以上のセキュリティグループを選択するか、新しいセキュリティグループを作成します。

[既存のセキュリティグループ] 1 つ以上のセキュリティグループを選択します。

[新しいセキュリティグループ] [Create new security group] (新しいセキュリティグループの作成) を選択し、[Security Groups] (セキュリティグループ) ページに移動します。完了したら、[Spot Requests] (スポットリクエスト) に戻ってリストを更新します。

vii. (オプション) インスタンスにインターネットからアクセスできるようにするには、[Auto-assign IPv4 Public IP] (IPv4 パブリック IP の自動割り当て) で [Enable] (有効化) を選択します。

viii. (オプション) IAM ロールを指定して スポットインスタンス を起動するには、[IAM instance profile] でロールを選択します。

ix. (オプション) 起動スクリプトを実行するには、スクリプトを [User data] (ユーザーデータ) にコピーします。

x. (オプション) タグを追加するには、[Create tag] (タグの作成) を選択し、タグのキーと値を入力してから [Create] (作成) を選択します。各タグについて、これを繰り返します。

タグごとに、インスタンスとスポットフリートリクエストに同じタグを付けるには、[Instances] (インスタンス) と [Fleet] (フリート) の両方が選択されていることを確認します。フリートによって起動されたインスタンスのみにタグ付けするには、[Fleet] (フリート) をクリアします。スポットフリートリクエストのみにタグ付けするには、[Istances] (インスタンス) をクリアします。

5. [Additional request details] (追加のリクエスト詳細) で、以下を実行します。

a. 追加リクエストの詳細を確認します。変更するには、[Apply defaults] をオフにします。

b. (オプション) [IAM fleet role] で、デフォルトのロールを使用するか、または別のロールを選択できます。ロールの変更後にデフォルトのロールを使用するには、[Use default role] を選択します。

c. (オプション) [Maximum price] では、デフォルトの上限料金 (オンデマンド料金) を使用するか、支払う予定の上限料金を指定することができます。上限価格が選択したインスタンスタイプのスポット料金より低い場合、スポットインスタンス は起動されません。

d. (オプション) 特定の期間中のみ有効なリクエストを作成するには、[Request valid from] (リクエスト有効期間開始日) および [Request valid until] (リクエスト有効期間終了日) を編集します。

e. (オプション) デフォルトでは、リクエストの有効期限が切れるとスポットインスタンスは終了します。リクエストの有効期限が切れた後も実行し続ける場合、[Terminate the instances when the request expires] (リクエストの期限後にインスタンスを終了) をオフにします。

f. (オプション) ロードバランサーを使用する スポットインスタンスを登録するには、[Receive traffic from one or more load balancers] (1 つ以上のロードバランサーからトラフィックを受信) を選択して、1 つ以上のクラシックロードバランサーまたはターゲットグループを選択します。

6. [Minimum compute unit] (最小コンピューティングユニット) で、アプリケーションまたはタスクで必要な最低限のハードウェア仕様 (vCPU、メモリ、ストレージ) を選択して、[as specs] (仕様として) または [as an instance type] (インスタンスタイプとして) を指定します。

   - [as specs] (仕様として) については、必要な vCPU 数とメモリ量を指定します。
   - [as an instance type] (インスタンスタイプとして) では、デフォルトのインスタンスタイプをそのまま使用するか、[Change instance type] (インスタンスインスタンスタイプを変更) を選択して別のインスタンスタイプを選択します。

7. [Target capacity] (ターゲット容量) で、以下の操作を実行します。

   a. [Total target capacity] (総ターゲット容量) で、ターゲット容量にリクエストする単位数を指定します。ユニットのタイプには、[Instances] (ユニット)、[vCPU]、または [Memory (MiB)] (メモリ (MiB)) を選択できます。ターゲット容量を 0 に指定して後で容量を追加できるようにするには、[Maintain target capacity] を選択します。

   b. (オプション) [Include On-Demand base capacity] (オンデマンドベースの容量を含める) で、リクエストするオンデマンド単位数を指定します。数値は [Total target capacity] (ターゲットキャパシティの合計) 未満にする必要があります。Amazon EC2 は差分を計算し、この差をリクエストするスポット単位に割り当てます。

> **Important**
>
> オプションのオンデマンド容量を指定する場合、最初に起動テンプレートを選択する必要があります。

c. (オプション) デフォルトでは、Amazon EC2 は中断されるとスポットインスタンスを削除します。ターゲット容量を維持するには、[ターゲット容量を維持する] を選択します。これで、中断時に Amazon EC2 がスポットインスタンスを終了、停止、または休止するように指定できます。これを行うには、[Interruption behavior] から対応するオプションを選択します。

> **Note**
>
> 起動パラメータで指定された AMI が登録解除または無効になっている場合、AMI から新しいインスタンスを起動することはできません。ターゲット容量を維持するように設定されたフリートの場合、ターゲット容量は維持されません。

d. (オプション) フリートの既存スポットインスタンスにインスタンスの再調整の通知が発行されたときに、スポットフリートが代替スポットインスタンスを起動できるようにするには、[Capacity rebalance] (容量の再調整) を選択し、インスタンス置換戦略を選択します。[Launch before terminate] (終了前に起動する) を選択した場合、スポットフリートが古いインスタンスを終了させるまでの遅延時間 (秒単位) を指定します。詳細については、「容量の再調整」を参照してください。

e. (オプション) フリートのすべての スポットインスタンスに対して 1 時間あたりに支払う金額を制御するには、[Set maximum cost for Spot Instances] (スポットインスタンスの上限価格を設定する) を選択し、1 時間あたりに支払うことができる上限の合計金額を入力します。上限の合計金額に達すると、ターゲット容量に満たない場合でも、スポットフリートはスポットインスタンスの起動を停止します。詳細については、「使用量の管理」を参照してください。

8. [Network] (ネットワーク) で、以下の操作を実行します。

a. [Network] (ネットワーク) で既存の VPC を選択するか、新しい VPC を作成します。

[既存の VPC] VPC を選択します。

[新しい VPC] [新しい VPC の作成] を選択して Amazon VPC コンソールにアクセスします。完了したら、ウィザードに戻ってリストを更新します。

b. (オプション) [アベイラビリティーゾーン] では、 でスポットインスタンス のアベイラビリティーゾーンを選択するか、1 つ以上のアベイラビリティーゾーンを指定します。AWS

アベイラビリティーゾーンに複数のサブネットがある場合、[Subnet] から適切なサブネットを選択します。サブネットを追加するには、[Create new subnet] を選択して Amazon VPC にアクセスします。完了したら、ウィザードに戻ってリストを更新します。

9. [Instance type requirements] (インスタンスタイプの要件) では、インスタンス属性を指定して、Amazon EC2 にこれらの属性を持つ最適なインスタンスタイプを識別させるか、またはインスタンスのリストを指定することができます。詳細については、「スポットフリートの属性ベースのインスタンスタイプの選択」を参照してください。

a. [Specify instance attributes that match your compute requirements] (コンピューティング要件に一致するインスタンス属性を指定する) を選択した場合、インスタンス属性を次のように指定します。

i. [vCPUs] に、希望する vCPU の最小数と最大数を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。

ii. [Memory (GiB)] (メモリ (GiB)) に、希望するメモリの最小値と最大値を入力します。制限なしを指定するには、[No minimum] (最小値なし)、[No maximum] (最大値なし)、または両方を選択します。

iii. (オプション) [Additional instance attributes] (その他のインスタンス属性) では、オプションで 1 つ以上の属性を指定して、コンピューティング要件をより詳細に表現できます。追加の属性は、リクエストにさらに制約を追加します。追加の属性は省略できます。省略すると、デフォルト値が使用されます。各属性およびそのデフォルト値の説明については、「Amazon EC2 コマンドラインリファレンス」の「get-spot-placement-scores」を参照してください。

iv. (オプション) 指定した属性を持つインスタンスタイプを表示するには、[Preview matching instance types] (一致するインスタンスタイプをプレビューする) を展開します。インスタンスタイプがリクエストで使用されないようにするには、インスタンスを選択し、[Exclude selected instance types] (選択したインスタンスタイプを除外する) を選択します。

b. [Manually select instance types] (インスタンスタイプを手動で選択する) を選択すると、スポットフリートはインスタンスタイプのデフォルトのリストを提供します。さらにインスタンスタイプを選択するには、[Add instance types] (インスタンスタイプの追加) を選択し、リクエストで使用するインスタンスタイプを選択してから [Select] (選択) を選択します。イ

ンスタンスタイプを削除するには、インスタンスタイプを選択し、[Delete] (削除) を選択します。

10. [Allocation strategy] (配分戦略) で、ニーズに合った戦略を選択します。詳細については、「スポットインスタンスの配分戦略」を参照してください。
11. [Your fleet request at a glance] (フリートリクエストの概要) で、フリートの設定を確認し、必要な調整を行います。
12. (オプション) AWS CLI で使用される起動設定のコピーをダウンロードするには、[JSON config] (JSON 設定) を選択します。
13. [Launch] を選択します。

    スポットフリートリクエストタイプは `fleet` です。リクエストが実行されると、タイプ `instance` のリクエストが追加されます。このとき、状態は `active` になり、ステータスは `fulfilled` になります。

スポットフリートを作成するにはAWS CLI

AWS CLI を使用して、スポットフリートリクエストを作成するには

- スポットフリートリクエストを作成するには、request-spot-fleet コマンドを使用します。https://docs.aws.amazon.com/cli/latest/reference/ec2/request-spot-fleet.html

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

設定ファイルの例については、「スポットフリートの設定例」を参照してください。

出力例を次に示します。

```
{
    "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE"
}
```

スポットフリートにタグ付けします。

スポットフリートリクエストを分類および管理しやすくするため、カスタムメタデータでタグ付けできます。スポットフリートリクエストへのタグの割り当ては、リクエストの作成時または作成後に行うことができます。Amazon EC2 コンソールまたはコマンドラインツールを使用してタグを割り当てることができます。

フリートリクエストにタグを付けると、スポットフリートが起動するインスタンスとボリュームには自動的にタグ付けされません。スポットフリートが起動するインスタンスとボリュームには、明示的にタグを付ける必要があります。タグは、フリートリクエストのみに割り当てるか、スポットフリートが起動したインスタンスのみに割り当てるか、フリートが起動したインスタンスにアタッチされたボリュームのみに割り当てるか、または 3 つすべてに割り当てるかを選択できます。

> **Note**
>
> ボリュームタグは、オンデマンドインスタンス にアタッチされたボリュームでのみサポートされます。スポットインスタンス にアタッチされているボリュームにタグを付けることはできません。

タグの仕組みの詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。

コンテンツ

前提条件

リソースにタグ付けする許可をユーザーに付与します。詳細については、「例: リソースのタグ付け」を参照してください。

リソースにタグ付けする許可をユーザーに付与するには

以下を含む IAM ポリシーを作成します。

- `ec2:CreateTags` アクション。これにより、タグを作成する許可がユーザーに付与されます。
- `ec2:RequestSpotFleet` アクション。これにより、スポットフリートリクエストを作成する許可がユーザーに付与されます。
- `Resource` で、`"*"` を指定する必要があります。これにより、ユーザーはすべてのリソースタイプにタグ付けできます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "TagSpotFleetRequest",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags",
                "ec2:RequestSpotFleet"
            ],
            "Resource": "*"
        }
    ]
}
```

Important

現在、spot-fleet-request リソースに対するリソースレベルのアクセス許可はサポートされていません。リソースとして spot-fleet-request を指定した場合、フリートにタグ付けしようとすると、不正な例外が発生します。以下の例は、ポリシーを設定 しない 方法を示しています。

```
{
    "Effect": "Allow",
    "Action": [
        "ec2:CreateTags",
        "ec2:RequestSpotFleet"
    ],
    "Resource": "arn:aws:ec2:us-east-1:111122223333:spot-fleet-request/*"
}
```

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」の手順に従ってください。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

- IAM ユーザー:
  - ユーザーが担当できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

新しいスポットフリートにタグを付けます。

コンソールを使用して、新しいスポットフリートリクエストにタグ付けするには

1. 「」の手順に従います。定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)
2. タグを追加するには、[追加設定] を展開し、[新規タグの追加] を選択して、タグのキーと値を入力します。各タグについて、これを繰り返します。

   タグごとに、スポットフリートリクエストとインスタンスに同じタグを付けることができます。両方にタグ付けするには、[Instance tags (インスタンスタグ)] と [Fleet tags (フリートタグ)] の両方が選択されていることを確認します。スポットフリートリクエストのみにタグ付けするには、[インスタンスタグ] をクリアします。フリートによって起動されたインスタンスのみにタグ付けするには、[Fleet tags (フリートタグ)] をクリアします。
3. 必須フィールドに入力してスポットフリートリクエストを作成し、[起動] を選択します。詳細については、「定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)」を参照してください。

AWS CLI を使用して、新しいスポットフリートリクエストにタグ付けするには

作成時にスポットフリートリクエストにタグ付けするには、スポットフリートリクエスト設定を以下のようにします。

- スポットフリートリクエストのタグを `SpotFleetRequestConfig` で指定します。
- `ResourceType` の場合、`spot-fleet-request` を指定します。別の値を指定すると、フリートリクエストは失敗します。
- `Tags` で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

以下の例では、スポットフリートリクエストに 2 つのタグ (Key=Environment、Value=Production、および Key=Cost-Center、Value=123) が付けられています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchSpecifications": [
            {
                "ImageId": "ami-0123456789EXAMPLE",
                "InstanceType": "c4.large"
            }
        ],
        "SpotPrice": "5",
        "TargetCapacity": 2,
        "TerminateInstancesWithExpiration": true,
        "Type": "maintain",
        "ReplaceUnhealthyInstances": true,
        "InstanceInterruptionBehavior": "terminate",
        "InstancePoolsToUseCount": 1,
        "TagSpecifications": [
            {
                "ResourceType": "spot-fleet-request",
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value":"Production"
                    },
                    {
                        "Key": "Cost-Center",
                        "Value":"123"
                    }
                ]
            }
        ]
    }
}
```

新しいスポットフリート、およびそれが起動するインスタンスおよびボリュームにタグ付けします。

新しいスポットフリートリクエストと、 を使用して起動するインスタンスおよびボリュームにタグ付けするにはAWS CLI

作成時にスポットフリートリクエストにタグ付けし、フリートがインスタンスを起動するときにインスタンスおよびボリュームにタグ付けするには、スポットフリートリクエスト設定を次のようにします。

スポットフリートリクエストのタグ

- スポットフリートリクエストのタグを SpotFleetRequestConfig で指定します。
- ResourceType の場合、spot-fleet-request を指定します。別の値を指定すると、フリートリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

インスタンスタグ:

- LaunchSpecifications で、インスタンスのタグを指定します。
- ResourceType の場合、instance を指定します。別の値を指定すると、フリートリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

  または、スポットフリートリクエストで参照される起動テンプレートで、インスタンスのタグを指定できます。???

ボリュームタグ:

- スポットフリートリクエストで参照される起動テンプレートのボリュームのタグを指定します。???LaunchSpecifications でのボリュームのタグ付けはサポートされていません。

以下の例では、スポットフリートリクエストに 2 つのタグ (Key=Environment、Value=Production、および Key=Cost-Center、Value=123) が付けられています。フリートが起動するインスタンスには、1 つのタグ (スポットフリートリクエストのタグの 1 つと同じ) Key=Cost-Center and Value=123 が付けられます。

```
{
    "SpotFleetRequestConfig": {
```

```
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchSpecifications": [
            {
                "ImageId": "ami-0123456789EXAMPLE",
                "InstanceType": "c4.large",
                "TagSpecifications": [
                    {
                        "ResourceType": "instance",
                        "Tags": [
                            {
                                "Key": "Cost-Center",
                                "Value": "123"
                            }
                        ]
                    }
                ]
            }
        ],
        "SpotPrice": "5",
        "TargetCapacity": 2,
        "TerminateInstancesWithExpiration": true,
        "Type": "maintain",
        "ReplaceUnhealthyInstances": true,
        "InstanceInterruptionBehavior": "terminate",
        "InstancePoolsToUseCount": 1,
        "TagSpecifications": [
            {
                "ResourceType": "spot-fleet-request",
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value":"Production"
                    },
                    {
                        "Key": "Cost-Center",
                        "Value":"123"
                    }
                ]
            }
        ]
    }
```

```
}
```

AWS CLI を使用して、スポットフリートが起動したインスタンスにタグ付けするには

フリートがインスタンスを起動するときにインスタンスにタグ付けするには、スポットフリートリクエストで参照される起動テンプレートでタグを指定するか、以下のようにスポットフリートリクエスト設定でタグを指定できます。???

- LaunchSpecifications で、インスタンスのタグを指定します。
- ResourceType の場合、instance を指定します。別の値を指定すると、フリートリクエストは失敗します。
- Tags で、キーと値のペアを指定します。キーと値のペアは複数指定できます。

以下の例では、フリートによって起動されるインスタンスに 1 つのタグ (Key=Cost-Center and Value=123) が付けられています。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchSpecifications": [
            {
                "ImageId": "ami-0123456789EXAMPLE",
                "InstanceType": "c4.large",
                "TagSpecifications": [
                    {
                        "ResourceType": "instance",
                        "Tags": [
                            {
                                "Key": "Cost-Center",
                                "Value": "123"
                            }
                        ]
                    }
                ]
            }
        ],
        "SpotPrice": "5",
        "TargetCapacity": 2,
```

```
        "TerminateInstancesWithExpiration": true,
        "Type": "maintain",
        "ReplaceUnhealthyInstances": true,
        "InstanceInterruptionBehavior": "terminate",
        "InstancePoolsToUseCount": 1
    }
}
```

AWS CLI を使用して、スポットフリートが起動するオンデマンドインスタンスにアタッチされたボリュームにタグ付けするには

フリートが作成したときにボリュームにタグ付けするには、スポットフリートリクエストで参照される起動テンプレートでタグを指定する必要があります。???

> **Note**
>
> ボリュームタグは、オンデマンドインスタンス にアタッチされたボリュームでのみサポートされます。スポットインスタンス にアタッチされているボリュームにタグを付けることはできません。
> LaunchSpecifications でのボリュームのタグ付けはサポートされていません。

既存のスポットフリートにタグを付けます。

コンソールを使用して、既存のスポットフリートリクエストにタグ付けするには

スポットフリートリクエストを作成した後、コンソールを使用してフリートリクエストにタグを追加できます。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択します。
4. [Tags (タグ)] タブを選択してから、[タグの作成] を選択します。

AWS CLI を使用して、既存のスポットフリートリクエストにタグ付けするには

create-tags コマンドを使用して、既存のリソースにタグ付けできます。https://docs.aws.amazon.com/cli/latest/reference/ec2/create-tags.html以下の例では、既存のスポットフリートリクエストにタグ Key=purpose and Value=test が付けられています。

```
aws ec2 create-tags \
    --resources sfr-11112222-3333-4444-5555-66666EXAMPLE \
    --tags Key=purpose,Value=test
```

スポットフリートリクエストタグを表示する

コンソールを使用して、スポットフリートリクエストタグを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、[タグ] タブを選択します。

スポットフリートリクエストタグを記述するには

describe-tags コマンドを使用して、指定したリソースのタグを表示します。https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-tags.html以下の例では、指定したスポットフリートリクエストのタグを記述します。

```
aws ec2 describe-tags \
    --filters "Name=resource-id,Values=sfr-11112222-3333-4444-5555-66666EXAMPLE"
```

```
{
    "Tags": [
        {
            "Key": "Environment",
            "ResourceId": "sfr-11112222-3333-4444-5555-66666EXAMPLE",
            "ResourceType": "spot-fleet-request",
            "Value": "Production"
        },
        {
            "Key": "Another key",
            "ResourceId": "sfr-11112222-3333-4444-5555-66666EXAMPLE",
            "ResourceType": "spot-fleet-request",
            "Value": "Another value"
        }
    ]
}
```

スポットフリートリクエストを記述することで、スポットフリートリクエストのタグを表示することもできます。

describe-spot-fleet-requests コマンドを使用して、指定したスポットフリートリクエストの設定を表示します。これには、フリートリクエストに指定されたタグが含まれます。https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-spot-fleet-requests.html

```
aws ec2 describe-spot-fleet-requests \
    --spot-fleet-request-ids sfr-11112222-3333-4444-5555-66666EXAMPLE
```

```
{
    "SpotFleetRequestConfigs": [
        {
            "ActivityStatus": "fulfilled",
            "CreateTime": "2020-02-13T02:49:19.709Z",
            "SpotFleetRequestConfig": {
                "AllocationStrategy": "capacityOptimized",
                "OnDemandAllocationStrategy": "lowestPrice",
                "ExcessCapacityTerminationPolicy": "Default",
                "FulfilledCapacity": 2.0,
                "OnDemandFulfilledCapacity": 0.0,
                "IamFleetRole": "arn:aws:iam::111122223333:role/aws-ec2-spot-fleet-
tagging-role",
                "LaunchSpecifications": [
                    {
                        "ImageId": "ami-0123456789EXAMPLE",
                        "InstanceType": "c4.large"
                    }
                ],
                "TargetCapacity": 2,
                "OnDemandTargetCapacity": 0,
                "Type": "maintain",
                "ReplaceUnhealthyInstances": false,
                "InstanceInterruptionBehavior": "terminate"
            },
            "SpotFleetRequestId": "sfr-11112222-3333-4444-5555-66666EXAMPLE",
            "SpotFleetRequestState": "active",
            "Tags": [
                {
                    "Key": "Environment",
                    "Value": "Production"
                },
                {
                    "Key": "Another key",
                    "Value": "Another value"
                }
```

```
            ]
        }
    ]
}
```

スポットフリートを記述する

上限料金がスポット料金を超え、容量が利用可能な場合、スポットフリートはスポットインスタンスを起動します。スポットインスタンスは、中断されるか終了されるまで実行されます。

スポットフリートを記述するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択します。設定の詳細を表示するには、[Description] を選択します。
4. スポットフリートのスポットインスタンスを一覧表示するには、インスタンスを選択します。
5. スポットフリートの履歴を表示するには、[履歴] を選択します。

スポットフリートを記述するには (AWS CLI)

スポットフリートリクエストの詳細を表示するには、describe-spot-fleet-requests コマンドを使用します。https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-spot-fleet-requests.html

```
aws ec2 describe-spot-fleet-requests
```

指定したスポットフリートのスポットインスタンスの詳細を表示するには、describe-spot-fleet-instances コマンドを使用します。https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-spot-fleet-instances.html

```
aws ec2 describe-spot-fleet-instances \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE
```

指定したスポットフリートリクエストの履歴を表示するには、describe-spot-fleet-request-history コマンドを使用します。https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-spot-fleet-request-history.html

```
aws ec2 describe-spot-fleet-request-history \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
```

```
    --start-time 2015-05-18T00:00:00Z
```

スポットフリートリクエストを変更します。

以下のタスクを完了するように、アクティブスポットフリートリクエストを変更できます。

- ターゲット容量とオンデマンド部分を増やす
- ターゲット容量とオンデマンド部分を減らす

> **Note**
>
> ワンタイムスポットフリートリクエストは変更できません。スポットフリートリクエストの作成時に [ターゲット容量の維持] を選択した場合にのみ、スポットフリートリクエストを変更することができます。

ターゲット容量を増やすと、スポットフリートは追加のスポットインスタンスを起動します。オンデマンド部分を増やすと、スポットフリートは追加のオンデマンドインスタンスを起動します。

ターゲット容量を増やすと、スポットフリートは、スポットフリートリクエストの配分戦略に従って、追加のスポットインスタンスを起動します。配分戦略が `lowestPrice` の場合、スポットフリートは、スポットフリートリクエストの最低価格のスポットキャパシティープールからインスタンスを起動します。配分戦略が `diversified` の場合、スポットフリートは、スポットフリートリクエストのプール全体にインスタンスを分散します。

ターゲット容量を減らすと、スポットフリートは新しいターゲット容量を超えるすべてのオープンリクエストをキャンセルします。フリートのサイズが新しいターゲット容量に達するまで、スポットフリートがスポットインスタンスを終了させるようにリクエストできます。配分戦略が `lowestPrice` の場合、スポットフリートは最高単価のインスタンスを終了させます。配分戦略が `diversified` の場合、スポットフリートはプール全体でインスタンスを終了させます。または、スポットフリートが現在のサイズを維持するようにリクエストすることもできますが、中断されたり手動で終了されたスポットインスタンスを置き換えることはできません。

ターゲット容量が減ったためにスポットフリートがインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

スポットフリートリクエストを変更するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択します。
4. [Actions]、[Modify target capacity] の順に選択します。
5. [Modify target capacity] で、以下の操作を実行します。
   a. 新しいターゲット容量とオンデマンド部分を入力します。
   b. (オプション) ターゲット容量を小さくしてもスポット群の現在のサイズを保持する場合は、[Terminate instances] をオフにします。
   c. [Submit] を選択します。

AWS CLI を使用して、スポットフリートリクエストを変更するには

modify-spot-fleet-requestコマンドを使用して、指定するスポットフリートリクエストのターゲット容量を更新します。https://docs.aws.amazon.com/cli/latest/reference/ec2/modify-spot-fleet-request.html

```
aws ec2 modify-spot-fleet-request \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity 20
```

前のコマンドを以下のように変更して、結果的にいずれのスポットインスタンスも終了せずに、指定したスポットフリートのターゲット容量を減らすことができます。

```
aws ec2 modify-spot-fleet-request \
    --spot-fleet-request-id sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --target-capacity 10 \
    --excess-capacity-termination-policy NoTermination
```

スポットフリートリクエストをキャンセルします。

スポットフリートが不要になった場合は、スポットフリートリクエストをキャンセルできます。フリートリクエストをキャンセルすると、フリートに関連付けられているすべてのスポットリクエストがキャンセルされるため、新しいスポットインスタンスは起動されません。

スポットフリートをキャンセルするときは、そのインスタンスをすべて終了させるかどうかも指定する必要があります。これには、オンデマンドインスタンスとスポットインスタンスの両方が含まれます。

フリートリクエストをキャンセルするときにインスタンスを終了する必要があることを指定した場合、フリートリクエストは cancelled_terminating 状態へ移行します。それ以外の場合、フリートリクエストは cancelled_running 状態になり、インスタンスは中断または手動終了されるまで、引き続き実行されます。

制限事項

- 1 回のリクエストで最大 100 個のフリートを削除できます。指定した数を超えると、フリートは削除されません。

スポットフリートリクエストをキャンセルするには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択します。
4. [アクション]、[リクエストのキャンセル] の順にクリックします。
5. [スポットリクエストのキャンセル] ダイアログボックスで、次の操作を行います。
   a. スポットフリートリクエストのキャンセルと同時に関連インスタンスを終了するには、[インスタンスの終了] チェックボックスをオンのままにします。関連インスタンスを終了せずにスポットフリートリクエストをキャンセルするには、[インスタンスの終了] チェックボックスを選択解除します。
   b. [確認] を選択します。

AWS CLI を使用して、スポットフリートリクエストをキャンセルし、そのインスタンスをキャンセルするには

cancel-spot-fleet-requests コマンドを使用し、指定したスポットフリートリクエストをキャンセルし、オンデマンドインスタンスとスポットインスタンスを終了します。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --terminate-instances
```

出力例

```
{
```

```
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_terminating",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

AWS CLI を使用して、そのインスタンスを終了せずにスポットフリートリクエストをキャンセルするには

`--no-terminate-instances` パラメータを使用して前のコマンドを変更することで、オンデマンドインスタンスとスポットインスタンスを終了せずに、指定されたスポットフリートをキャンセルできます。

```
aws ec2 cancel-spot-fleet-requests \
    --spot-fleet-request-ids sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
    --no-terminate-instances
```

出力例

```
{
    "SuccessfulFleetRequests": [
        {
            "SpotFleetRequestId": "sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
            "CurrentSpotFleetRequestState": "cancelled_running",
            "PreviousSpotFleetRequestState": "active"
        }
    ],
    "UnsuccessfulFleetRequests": []
}
```

スポットフリートの CloudWatch メトリクス

Amazon EC2 は、スポットフリートをモニタリングするために使用できる Amazon CloudWatch メトリクスを提供します。

> **Important**
>
> 正確性を確実にするため、これらのメトリクスを使用する際は詳細モニタリングを有効にすることをお勧めします。詳細については、「インスタンスの詳細モニタリングを有効または無効にする」を参照してください。

Amazon EC2 によって提供される CloudWatch メトリクスの詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。

スポットフリートのメトリクス

AWS/EC2Spot 名前空間には、次のメトリクスに加えて、スポット群の スポットインスタンス 用の CloudWatch メトリクスが含まれます。詳細については、「インスタンスメトリクス」を参照してください。

メトリクス	説明
AvailableInstancePoolsCount	スポットフリートリクエストで指定されているスポットキャパシティープール 単位: カウント
BidsSubmittedForCapacity	Amazon EC2 がスポットフリートリクエストを送信した容量 単位: カウント
EligibleInstancePoolCount	Amazon EC2 がリクエストを受理できるスポットフリートリクエストで指定されたスポットキャパシティープール スポットインスタンスに支払う上限価格がスポット料金よりも低いか、スポット料金がオンデマンドインスタンス料金よりも高いプールでは、Amazon EC2 はリクエストを受理しません。 単位: カウント

メトリクス	説明
FulfilledCapacity	Amazon EC2 が落札した容量。 単位: カウント
MaxPercentCapacityAllocation	スポットフリートリクエストで指定されたすべてのスポットフリートプールでの PercentCapacityAllocation の最大値。 単位: パーセント
PendingCapacity	TargetCapacity と FulfilledCapacity の違い。 単位: カウント
PercentCapacityAllocation	指定されたディメンションのスポットキャパシティープールに配分された容量。すべてのスポットキャパシティープールにおける最大値を取得するには、 を使用します。MaxPercentCapacityAllocation 単位: パーセント
TargetCapacity	スポットフリートリクエストのターゲット容量 単位: カウント
TerminatingCapacity	プロビジョニングされた容量が目標の容量より大きいために終了した容量。 単位: カウント

メトリクスの測定単位が Count である場合、最も有用な統計は Average です。

スポットフリートディメンション

スポットフリートのデータをフィルタリングするには、次のディメンションを使用します。

ディメンション	説明
AvailabilityZone	アベイラビリティーゾーン別にデータをフィルタリングします。
FleetRequestId	スポットフリートのリクエスト別にデータをフィルタリングします。
InstanceType	インスタンスタイプ別にデータをフィルタリングします。

スポットフリートの CloudWatch メトリクスを表示します。

Amazon CloudWatch コンソールを使用して、スポットフリートの CloudWatch メトリクスを表示できます。これらのメトリクスは、モニタリング用のグラフのように表示されます。これらのグラフは、スポットフリートがアクティブの場合にデータポイントを表示します。

メトリクスはまず名前空間ごとにグループ化され、次に各名前空間内の種々のディメンションの組み合わせごとにグループ化されます。例えば、すべてのスポットフリートメトリクスまたはスポットフリートメトリクスグループは、スポットフリートリクエスト ID、インスタンスタイプ、またはアベイラビリティーゾーン別に表示できます。

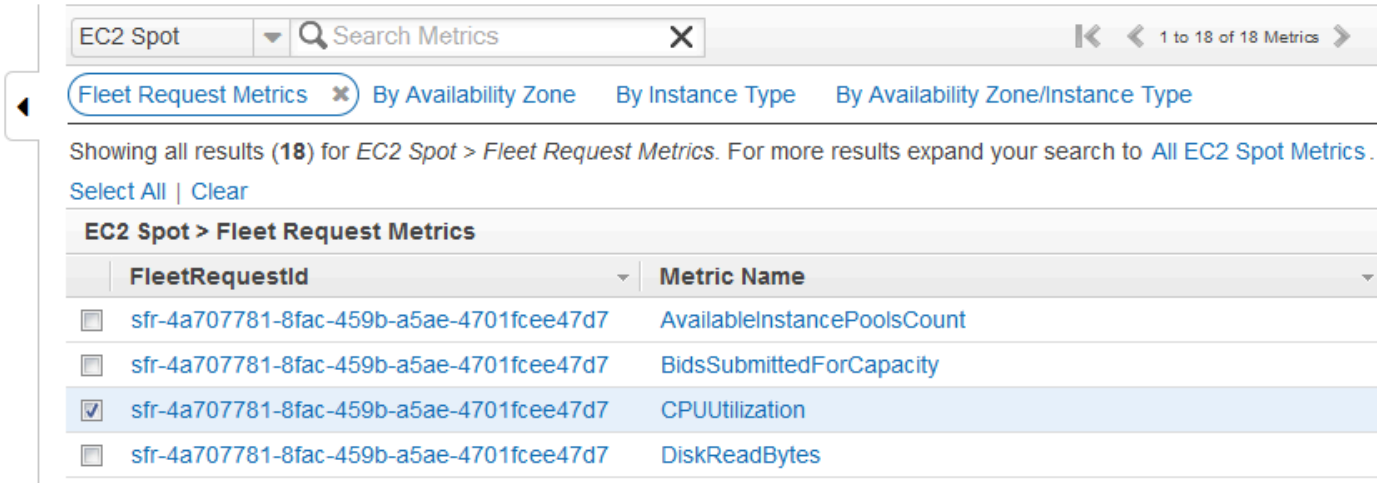
スポットフリートメトリクスを表示するには

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
3. EC2 スポットの名前空間を選択します。

> **Note**
>
> EC2 スポットの名前空間が表示されない場合、これには 2 つの原因があります。スポットフリートをまだ使用していません。使用中の AWS サービスのみメトリクスを

Amazon CloudWatch に送信します。または、過去 2 週間にスポットフリートを使用していない場合は、名前空間は表示されません。

4. (オプション) ディメンション別にメトリクスをフィルタするには、次のいずれかを選択します。
   - [フリートリクエストメトリクス] - スポットフリートリクエスト別にグループ化
   - [アベイラビリティーゾーン別] - スポットフリートリクエストおよびアベイラビリティーゾーン別にグループ化
   - [インスタンスタイプ別] - スポットフリートリクエストおよびインスタンスタイプ別にグループ化
   - [アベイラビリティーゾーン/インスタンスタイプ別] - スポットフリートリクエスト、アベイラビリティーゾーン、インスタンスタイプ別にグループ化
5. メトリクスのデータを表示するには、メトリクスの横にあるチェックボックスをオンにします。



スポットフリートの自動スケーリング

自動スケーリングは、需要に応じてスポットフリートのターゲット容量を自動的に増減する機能です。スポットフリートは、1 つ以上のスケーリングポリシーに応答して、選択する範囲内でインスタンスを起動 (スケールアウト) するか、インスタンスを終了 (スケールイン) できます。

スポットフリートは、以下のタイプの自動スケーリングをサポートします。

- ターゲット追跡スケーリング - 特定のメトリクスのターゲット値に基づいて、フリートの現在の容量を増減させます。???これはサーモスタットで家の温度を管理する方法と似ています (温度を選択すれば、後はサーモスタットがすべてを実行する)。

- ステップスケーリング - アラーム違反のサイズに応じて変動する一連のスケーリング調整値 (ステップ調整値) に基づいて、フリートの現在の容量を増減させます。???
- スケジュールに基づくスケーリング - 日付と時刻に基づいて、フリートの現在の容量を増減させます。???

インスタンスの分量指定を使用している場合は、必要に応じてスポットフリートがターゲット容量を超える場合があることに注意してください。???取得済みの容量が浮動小数点数となってもターゲット容量は整数でなければならないため、スポットフリートはその数を次の整数に切り上げます。アラームがトリガーされたときにスケーリングポリシーの結果を確認する際は、このような動作を考慮に入れる必要があります。例えば、ターゲット容量が 30、取得済み容量が 30.1 で、スケーリングポリシーが 1 を減算するとします。アラームがトリガーされると、自動スケーリングプロセスは 30.1 から 1 を減算して 29.1 を得るため、この数は 30 に切り上げられることになり、スケーリングアクションは実行されません。別の例として、選択したインスタンスの分量が 2、4、8 であり、ターゲット容量が 10 であるとします。分量 2 のインスタンスが利用できなかったために、スポットフリートは分量 4 と 8 のインスタンスをプロビジョニングして取得済みの容量が 12 になったとします。スケーリングポリシーがターゲット容量を 20% 減らしてアラームがトリガーされた場合、自動スケーリングプロセスは 12 から 12×0.2 を減算して 9.6 を得るため、この数は 10 に切り上げられることになり、スケーリングアクションは実行されません。

スポットフリート用に作成したスケーリングポリシーは、クールダウン期間をサポートしています。クールダウン期間は、以前のトリガー関連のスケーリングアクティビティが以後のスケーリングイベントに影響を及ぼすことができる期限であり、スケーリングアクティビティが終了した時点からの秒数として指定します。スケールアウトポリシーにクールダウン期間を設定すると、その期間中にクールダウンを開始したスケールアウトイベントによって追加された容量は、次のスケールアウトに予定される容量の一部として繰り入れられます。これにより、スケールアウトが継続的に (ただし過剰になることなく) 行われます。スケールインポリシーにクールダウン期間を設定すると、その期間が過ぎるまでは以後のスケールインリクエストがブロックされます。これにより、スケールインが抑制されてアプリケーションの可用性が確保されます。ただし、スケールイン後のクールダウン期間中に別のアラームによってスケールアウトポリシーがトリガーされると、自動スケーリングによってスケーラブルなターゲットが即座にスケールアウトされます。

使用率の変化に迅速に対応できるように、1 分間隔でインスタンスのメトリクスをスケーリングすることをお勧めします。5 分間隔でメトリクスをスケールすると、応答時間が低速になり、古いメトリクスデータに基づいてスケールすることになる可能性があります。1 分ごとにインスタンスのメトリクスデータを CloudWatch に送信するには、インスタンスで詳細モニタリングを有効にできます。詳細については、インスタンスの詳細モニタリングを有効または無効にするおよび定義済みパラメータを使用してスポットフリートリクエストを作成する (コンソール)を参照してください。

スポットフリートのスケーリングの設定の詳細については、次のリソースを参照してください。

- AWS CLI コマンドリファレンスの application-autoscaling セクション
- Application Auto Scaling API リファレンスhttps://docs.aws.amazon.com/autoscaling/application/APIReference/
- アプリケーション Auto Scaling ユーザーガイドhttps://docs.aws.amazon.com/autoscaling/application/userguide/

スポットフリートの自動スケーリングに必要な IAM のアクセス許可

スポットフリートの自動スケーリングは、Amazon EC2、Amazon CloudWatch、および Application Auto Scaling API の組み合わせによって可能になります。スポットフリートは Amazon EC2 で作成され、アラームは CloudWatch で作成され、スケーリングポリシーは Application Auto Scaling で作成されます。

スポットフリートの IAM 許可と Amazon EC2 に加えて、フリートスケーリング設定にアクセスするユーザーは、動的スケーリングをサポートするサービスに対する適切な許可が必要です。ユーザーには、次のポリシー例に示すアクションを使用するための許可が必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "iam:CreateServiceLinkedRole",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
```

```
                "sns:List*"
            ],
            "Resource": "*"
        }
    ]
}
```

独自の IAM ポリシーを作成し、アプリケーションの Auto Scaling API を呼び出すためのよりきめ細かなアクセス許可を付与することもできます。詳細については、アプリケーションの Auto Scaling ユーザーガイドの「認証とアクセスコントロール」を参照してください。https://docs.aws.amazon.com/autoscaling/application/userguide/auth-and-access-control.html

Application Auto Scaling サービスには、スポットフリートおよび CloudWatch アラームを記述するアクセス許可、およびユーザーの代わりにスポットフリートターゲット容量を変更するアクセス許可も必要です。スポットフリートの自動スケーリングを有効にすると、サービスにリンクされた `AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest` というロールが作成されます。このサービスにリンクされたロールは、アプリケーションの Auto Scaling に対して、ポリシーのアラームの記述、フリートの現容量のモニタリング、およびフリートの容量の変更を行うためのアクセス許可を付与します。Application Auto Scaling の元のマネージド型のスポットフリートロールは `aws-ec2-spot-fleet-autoscale-role` ですが、これは不要になりました。サービスにリンクされたロールは、アプリケーションの Auto Scaling のデフォルトロールです。詳細については、アプリケーションの Auto Scaling ユーザーガイドの「サービスにリンクされたロール」を参照してください。https://docs.aws.amazon.com/autoscaling/application/userguide/application-auto-scaling-service-linked-roles.html

ターゲット追跡ポリシーを使用して、スポットフリートをスケーリングします。

ターゲット追跡スケーリングポリシーで、メトリクスを選択してターゲット値を設定します。スポットフリートは、スケーリングポリシーをトリガーする CloudWatch アラームを作成および管理し、メトリクスとターゲット値に基づいてスケーリング調整値を計算します。スケーリングポリシーは、指定されたターゲット値、またはそれに近い値にメトリクスを維持するため、必要に応じて容量を追加または削除します。ターゲットの追跡スケーリングポリシーは、メトリクスをターゲット値近くに維持することに加えて、負荷パターンの変動によるメトリクスの変動に合わせて調整し、フリートの容量の急速な変動を最小化します。

それぞれが異なるメトリクスを使用していれば、スポットフリートに複数のターゲットの追跡スケーリングポリシーを作成できます。フリートは、最大のフリート容量を提供する方針に基づいてスケーリングされます。これにより、複数のシナリオに対応して、アプリケーションワークロードを処理するのに十分な容量が常に確保されます。

アプリケーションの可用性を高めるために、フリートのスケールアウトはメトリクスに比例して可能な限り高速に行われますが、スケールインはより緩やかです。

ターゲット容量が減ったためにスポットフリートがインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

ターゲットの追跡スケーリングポリシーのためにスポットフリートが管理する CloudWatch アラームを編集または削除しないでください。ターゲット追跡スケーリングポリシーを削除すると、スポットフリートが自動的にアラームを削除します。

制限

スポットフリートリクエストには、タイプが `maintain` のリクエストが必要です。自動スケーリングはタイプ `request` のリクエストではサポートされていません。

ターゲットの追跡スケーリングポリシーを設定するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、[Auto Scaling] を選択します。
4. 自動スケーリングが設定されていない場合は、[Configure] を選択します。
5. スポットフリートの最小容量および最大容量を設定するには、[Scale capacity between] を使用します。自動スケーリングにより、最小容量以下または最大容量以上にスポットフリートがスケールされることはありません。
6. [Policy Name] にこのポリシーの名前を入力します。
7. [Target metric] を選択します。
8. メトリクスの [Target value] を入力します。
9. [クールダウン期間] には、新しい値 (秒単位) を指定するか、デフォルトのままにします。
10. (オプション) 現在の構成に基づいてスケールインポリシーの作成を省略するには、[スケールインの無効化] を選択します。別の構成を使用してスケールインポリシーを作成できます。
11. [Save] を選択します。

AWS CLI を使用して、ターゲットの追跡スケーリングポリシーを設定します。

1. register-scalable-target コマンドを使用して、スケーラブルなターゲットとしてスポットフリートリクエストを登録します。
2. put-scaling-policy コマンドを使用して、スケーリングポリシーを作成します。

ステップスケーリングポリシーを使用して、スポットフリートをスケーリングします。

ステップスケーリングポリシーでは、CloudWatch アラームを指定してスケーリングプロセスをトリガーします。例えば、CPU 利用率が一定のレベルに達したときにスケールアウトする場合、Amazon EC2 によって提供される `CPUUtilization` メトリクスを使用してアラームを作成します。

ステップスケーリングポリシーを作成したら、次のいずれかのスケーリング調整タイプを指定する必要があります。

- [追加] - 指定した数の容量ユニットまたは指定した割合の現在の容量で、スポットフリートのターゲット容量を増やします。
- [削除] - 指定した数の容量ユニットまたは指定した割合の現在の容量で、スポットフリートのターゲット容量を減らします。
- [設定] - 指定した数の容量ユニットに、スポットフリートのターゲット容量を設定します。

アラームがトリガーされると、自動スケーリングプロセスは、取得済み容量およびスケーリングポリシーを使用して新しいターゲット容量を計算し、必要に応じてターゲット容量を更新します。例えば、ターゲット容量と取得済み容量がそれぞれ 10 で、スケーリングポリシーが 1 を加算するとします。アラームがトリガーされると、自動スケーリングプロセスは 10 に 1 を加えて 11 を得るため、スポットフリートは 1 インスタンスを起動します。

ターゲット容量が減ったためにスポットフリートがインスタンスを終了する場合、インスタンスはスポットインスタンスの中断通知を受け取ります。

制限

スポットフリートリクエストには、タイプが `maintain` のリクエストが必要です。自動スケーリングはタイプ `request` のリクエストまたはスポットブロックではサポートされていません。

前提条件

- アプリケーションにとってどの CloudWatch メトリクスが重要化を検討します。AWS または独自のカスタムメトリクスが提供するメトリクスに基づいて、CloudWatch アラームを作成できます。
- スケーリングポリシーで使用する AWS メトリクスについて、メトリクスを提供するサービスがデフォルトで有効にならない場合、CloudWatch メトリクスの収集を有効にします。

CloudWatch アラームを作成するには

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで、[Alarms] を選択します。
3. [アラームの作成] を選択します。
4. [Specify metric and conditions (メトリクスと条件を指定)] ページで、[メトリクスの選択] を選択します。
5. [EC2 スポット]、[フリートリクエストのメトリクス] の順に選択し、メトリクス (CPUUtilization など) を選択して [メトリクスの選択] を選択します。

   [Specify metric and conditions (メトリクスと条件の指定)] ページに、選択したメトリクスに関するグラフや他の情報が表示されます。
6. [期間] でアラームの評価期間 (1 分など) を選択します。アラームを評価する場合、各期間は 1 つのデータポイントに集約されます。

   > **Note**
   >
   > 期間が短いほど、作成されるアラームの感度が高くなります。
7. [条件] で、しきい値条件を定義してアラームを定義します。例えば、メトリクスの値が 80% 以上になるたびにアラームをトリガーするように、しきい値を定義できます。
8. [Additional configuration (追加設定)] の [Datapoints to alarm (アラームするデータポイント)] で、アラームをトリガーするために ALARM 状態になる必要があるデータポイント (評価期間) の数を指定します (3 個の評価期間のうち 1 個または 2 個の評価期間など)。これでアラームが作成されます。このアラームは、指定した数の期間で連続してしきい値を超過すると、ALARM 状態に移行します。詳細については、Amazon CloudWatch ユーザーガイド のアラームを評価するを参照してください。
9. [Missing data treatment (不足しているデータの扱い)] で、いずれかのオプションを選択します (または、デフォルトの [Treat missing data as missing (不足しているデータを不足として扱う)] のままにします)。詳細については、Amazon CloudWatch ユーザーガイドの「CloudWatch アラームが欠落データを処理する方法の設定」を参照してください。
10. [Next] を選択します。
11. (オプション) スケーリングイベントの通知を受け取る場合は、[通知] で、通知を受け取るために使用する Amazon SNS トピックを選択または作成できます。それ以外の場合は、通知を削除し、必要に応じて後で追加できます。

12. [Next] を選択します。
13. [Add a description (説明の追加)] にアラームの名前と説明を入力し、[次へ] を選択します。
14. [アラームの作成] を選択します。

スポットフリートのステップスケーリングポリシーを設定するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、[Auto Scaling] を選択します。
4. 自動スケーリングが設定されていない場合は、[Configure] を選択します。
5. スポットフリートの最小容量および最大容量を設定するには、[Scale capacity between] を使用します。スケーリングポリシーにより、最小容量未満に、または最大容量を超えてフリートがスケールされることはありません。
6. [スケーリングポリシー]、[ポリシータイプ] で [ステップスケーリングポリシー] を選択します。
7. 初期状態では、[スケーリングポリシー] には ScaleUp と ScaleDown という名前のポリシーが含まれています。これらのポリシーは、完了するか、[Remove policy] を選択して削除できます。[Add policy] を選択することもできます。
8. ポリシーを定義するには、以下の作業を行います。

    a. [Policy Name] にこのポリシーの名前を入力します。

    b. [ポリシートリガー] で、既存のアラームを選択するか、[アラームを作成] を選択して Amazon CloudWatch コンソールを開き、アラームを作成します。

    c. [容量の変更] では、スケーリングする量と、ステップ調整の下限と上限を指定します。特定の数のインスタンスまたは既存のグループサイズに対する割合を追加または削除したり、フリートを正確なサイズに設定したりできます。

    例えば、フリートのキャパシティを 30% 増やすステップスケーリングポリシーを作成するには、Add を選択し、次のフィールドに 30 を入力後 percent を選択します。デフォルトでは、[ポリシーを追加] の下限はアラームしきい値であり、上限は正 (+) の無限大です。デフォルトでは、[ポリシーを削除] の上限はアラームしきい値であり、下限は負 (-) の無限大です。

    d. (オプション) 別のステップを追加するには、[ステップを追加] を選択します。

    e. [クールダウン期間] には、新しい値 (秒単位) を指定するか、デフォルトのままにします。
9. [Save] を選択します。

AWS CLI を使用して、スポットフリートのステップスケーリングポリシーを設定するには

1. register-scalable-target コマンドを使用して、スケーラブルなターゲットとしてスポットフリートリクエストを登録します。
2. put-scaling-policy コマンドを使用して、スケーリングポリシーを作成します。
3. put-metric-alarm コマンドを使用してスケーリングポリシーをトリガーするアラームを作成します。https://docs.aws.amazon.com/cli/latest/reference/cloudwatch/put-metric-alarm.html

スケジュールに基づくスケーリングを使用して、スポットフリートをスケーリングします。

スケジュールに基づくスケーリングにより、予想可能な需要の変化に応じてアプリケーションを拡張することができます。スケジュールに基づくスケーリングを使用するには、スポットフリートに指定された時間にスケーリングアクティビティを行うよう伝える、スケジュールされたアクションを作成します。スケジュールされたアクションを作成するとき、既存のスポットフリート、スケーリングアクティビティが起こる時刻、最小容量、最大容量を指定します。スケジュールされたアクションは1回だけ、または反復して行われるように作成できます。

既に存在する スポットフリート 用のスケジュールされたアクションのみを作成できます。スケジュールされたアクションは、スポットフリートの作成と同時に作成することはできません。

制限

スポットフリートリクエストには、タイプが `maintain` のリクエストが必要です。自動スケーリングはタイプ `request` のリクエストまたはスポットブロックではサポートされていません。

1 回のアクションを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、画面の下部にある [スケジュールされたスケーリング] タブを選択します。
4. [予定アクションの作成] を選択します。
5. [名前] に、予定アクションの名前を指定します。
6. [最小容量]、[最大容量]、または両方の値を入力します。
7. [繰り返し] で、[1 回] を選択します。

8. (オプション) [開始時刻]、[終了時刻]、またはその両方の日付と時刻を選択します。
9. [Submit] を選択します。

定期的なスケジュールでスケールするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、画面の下部にある [スケジュールされたスケーリング] タブを選択します。
4. [繰り返し] で、事前定義済みのスケジュール (例えば、[毎日]) のいずれかを選択するか、[カスタム] を選択して cron 式を入力します。スケジュールに基づくスケーリングがサポートする cron 式の詳細については、Amazon CloudWatch Events ユーザーガイドの「cron 式」を参照してください。https://docs.aws.amazon.com/AmazonCloudWatch/latest/events/ScheduledEvents.html#CronExpressions
5. (オプション) [開始時刻]、[終了時刻]、またはその両方の日付と時刻を選択します。
6. [Submit] を選択します。

スケジュールされたアクションを編集するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、画面の下部にある [スケジュールされたスケーリング] タブを選択します。
4. スケジュールされたアクション を選択して、[Actions]、[Edit] の順に選択します。
5. 必要な変更を加えて、[Submit] を選択します。

スケジュールされたアクションを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Spot Requests] を選択します。
3. スポットフリートリクエストを選択し、画面の下部にある [スケジュールされたスケーリング] タブを選択します。
4. スケジュールされたアクションを選択して、[アクション]、[削除] の順に選択します。
5. 確認を求めるメッセージが表示されたら、[削除] を選択します。

AWS CLI を使用してスケジュールされたスケーリングを管理するには

次のコマンドを使用します。

- put-scheduled-actionhttps://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/put-scheduled-action.html
- describe-scheduled-actionshttps://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/describe-scheduled-actions.html
- delete-scheduled-actionhttps://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/delete-scheduled-action.html

Amazon EventBridge を使用したフリートイベントのモニタリング

EC2 フリートかスポットフリートの状態に変化があった場合、そのフリートから通知が発せられます。通知は、Amazon EventBridge (旧称 Amazon CloudWatch Events) に送信されるイベントとして利用できます。イベントは、ベストエフォートベースで出力されます。

Amazon EventBridge では、イベントに応答してプログラムによるアクションをトリガーするルールを作成できます。例えば、フリートの状態が変更されたときにトリガーされるルールと、フリートのインスタンスが終了したときにトリガーされるルールという 2 つの EventBridge ルールを作成できます。1 番目のルールでは、フリートの状態が変更された際に、そのルールが SNS トピックを呼び出して E メール通知を送信するように設定します。2 番目のルールでは、インスタンスが終了した場合に Lambda 関数を呼び出して、新しいインスタンスを起動するように設定します。

トピック

- EC2 フリート イベントタイプ
- スポットフリートイベントタイプ
- Amazon EventBridge ルールを作成する

EC2 フリート イベントタイプ

> **Note**
>
> タイプ `maintain` と `request` のフリートのみがイベントを発行します。タイプ `instant` のフリートは、同期された 1 回限りのリクエストを送信するため、イベントを発行しません。フリートの状態は応答ですぐに認識されます。

5 つの EC2 フリート イベントタイプがあります。イベントタイプごとに、いくつかのサブタイプがあります。

イベントは JSON 形式で EventBridge に送信されます。イベント内の次のフィールドは、ルールで定義され、アクションをトリガーするイベントパターンを形成します。

`"source": "aws.ec2fleet"`

イベントが EC2 フリート からのものであることを特定します。

`"detail-type": "EC2 Fleet State Change"`

イベントタイプを特定します。

`"detail": { "sub-type": "submitted" }`

イベントのサブタイプを特定します。

イベントタイプ

- EC2 フリートの状態の変更
- EC2 フリートのスポットインスタンスリクエストの変更
- EC2 フリートインスタンスの変更
- EC2 フリート情報
- EC2 フリートエラー

EC2 フリートの状態の変更

EC2 フリート は、EC2 フリート の状態が変更されたときに `EC2 Fleet State Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "715ed6b3-b8fc-27fe-fad6-528c7b8bf8a2",
    "detail-type": "EC2 Fleet State Change",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T09:00:20Z",
    "region": "us-east-1",
```

```
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-598fb973-87b7-422d-
be4d-6b0809bfff0a"
    ],
    "detail": {
        "sub-type": "active"
    }
}
```

sub-type に指定できる値は、次のとおりです。

active

EC2 フリート リクエストは検証済みです。Amazon EC2 は実行中のインスタンスをターゲット数分、確保しようとしています。

deleted

EC2 フリート リクエストが削除され、実行中のインスタンスがありません。EC2 フリート は、そのインスタンスが終了してから 2 日後に削除されます。

deleted_running

EC2 フリート リクエストが削除され、追加のインスタンスは起動されません。その既存のインスタンスは、中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。

deleted_terminating

EC2 フリートリクエストが削除され、そのインスタンスが終了します。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

expired

EC2 フリートリクエストが期限切れです。このリクエストが TerminateInstancesWithExpiration セットを使用して作成されている場合、後続の terminated イベントは、インスタンスが終了済みなことを示します。

modify_in_progress

EC2 フリート リクエストは変更中です。リクエストは、この変更が完全に処理されるまで、同じ状態を維持します。

modify_succeeded

EC2 フリートリクエストが変更されました。

`submitted`

EC2 フリート リクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。

`progress`

EC2 フリートリクエストは受理中です。

EC2 フリートのスポットインスタンスリクエストの変更

EC2 フリート は、フリート内のスポットインスタンスリクエストの状態が変更されたときに `EC2 Fleet Spot Instance Request Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "19331f74-bf4b-a3dd-0f1b-ddb1422032b9",
    "detail-type": "EC2 Fleet Spot Instance Request Change",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T09:00:05Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/
fleet-83fd4e48-552a-40ef-9532-82a3acca5f10"
    ],
    "detail": {
        "spot-instance-request-id": "sir-rmqske6h",
        "description": "SpotInstanceRequestId sir-rmqske6h, PreviousState:
 cancelled_running",
        "sub-type": "cancelled"
    }
}
```

`sub-type` に指定できる値は、次のとおりです。

`active`

スポットインスタンスリクエストは受理された状態であり、スポットインスタンスの関連付けが完了しています。

`cancelled`

スポットインスタンスリクエストがキャンセルされている、あるいは、そのリクエストの有効期限が切れています。

`disabled`

スポットインスタンスが停止されています。

`submitted`

スポットインスタンスリクエストは送信済みです。

EC2 フリートインスタンスの変更

EC2 フリート は、フリート内のインスタンスの状態が変更されたときに `EC2 Fleet Instance Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "542ce428-c8f1-0608-c015-e8ed6522c5bc",
    "detail-type": "EC2 Fleet Instance Change",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T09:00:23Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-598fb973-87b7-422d-
be4d-6b0809bfff0a"
    ],
    "detail": {
        "instance-id": "i-0c594155dd5ff1829",
        "description": "{\"instanceType\":\"c5.large\",\"image\":\"ami-6057e21a\",
\"productDescription\":\"Linux/UNIX\",\"availabilityZone\":\"us-east-1d\"}",
        "sub-type": "launched"
    }
}
```

`sub-type` に指定できる値は、次のとおりです。

`launched`

新しいインスタンスが起動されました。

`terminated`

このインスタンスは終了しています。

`termination_notified`

フリートのターゲット容量のスケールダウン中 (ターゲット容量 が 4 から 3 に変更される場合など) に、Amazon EC2 によってスポットインスタンスが終了されたので、インスタンス終了通知が送信されました。

EC2 フリート情報

EC2 フリート は、受理中にエラーが発生したときに `EC2 Fleet Information` イベントを Amazon EventBridge に送信します。情報イベントは、フリートがターゲット容量を満たすことをブロックしません。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "76529817-d605-4571-7224-d36cc1b2c0c4",
    "detail-type": "EC2 Fleet Information",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-11-09T08:17:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-8becf5fe-
bb9e-415d-8f54-3fa5a8628b91"
    ],
    "detail": {
        "description": "c4.xlarge, ami-0947d2ba12ee1ff75, Linux/UNIX, us-east-1a,
 Spot price in either SpotFleetRequestConfigData or SpotFleetLaunchSpecification or
 LaunchTemplate or LaunchTemplateOverrides is less than Spot market price $0.0619",
        "sub-type": "launchSpecUnusable"
    }
}
```

`sub-type` に指定できる値は、次のとおりです。

`fleetProgressHalted`

すべての起動仕様の料金は、スポット料金を下回っているため無効です (すべての起動仕様が `launchSpecUnusable` イベントを生成しました)。スポット料金が変更されると、起動仕様が有効になる場合があります。

`launchSpecTemporarilyBlacklisted`

設定が有効ではなく、インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

`launchSpecUnusable`

この起動仕様の料金は、スポット料金を下回っているため無効です。

`registerWithLoadBalancersFailed`

ロードバランサーにインスタンスを登録しようとして失敗しました。詳細については、イベントの説明をご覧ください。

EC2 フリートエラー

EC2 フリート は、受理中にエラーが発生したときに `EC2 Fleet Error` イベントを Amazon EventBridge に送信します。エラーイベントは、フリートがターゲット容量を満たすことをブロックします。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "69849a22-6d0f-d4ce-602b-b47c1c98240e",
    "detail-type": "EC2 Fleet Error",
    "source": "aws.ec2fleet",
    "account": "123456789012",
    "time": "2020-10-07T01:44:24Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:fleet/fleet-9bb19bc6-60d3-4fd2-ae47-
d33e68eafa08"
    ],
    "detail": {
        "description": "m3.large, ami-00068cd7555f543d5, Linux/UNIX: IPv6 is not
 supported for the instance type 'm3.large'. ",
```

```
        "sub-type": "spotFleetRequestConfigurationInvalid"
    }
}
```

sub-type に指定できる値は、次のとおりです。

iamFleetRoleInvalid

この EC2 フリートには、インスタンスの起動または終了に必要なアクセス許可がありません。

allLaunchSpecsTemporarilyBlacklisted

有効な設定はありません。インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

spotInstanceCountLimitExceeded

起動できるスポットインスタンスの数の上限に達しました。

spotFleetRequestConfigurationInvalid

設定が有効ではありません。詳細については、イベントの説明をご覧ください。

スポットフリートイベントタイプ

5 つのスポットフリートイベントタイプがあります。イベントタイプごとに、いくつかのサブタイプがあります。

イベントは JSON 形式で EventBridge に送信されます。イベント内の次のフィールドは、ルールで定義され、アクションをトリガーするイベントパターンを形成します。

"source": "aws.ec2spotfleet"

イベントがスポットフリートからのものであることを特定します

"detail-type": "*EC2 Spot Fleet State Change*"

イベントタイプを特定します。

"detail": { "sub-type": "*submitted*" }

イベントのサブタイプを特定します。

イベントタイプ

- EC2 スポットフリートの状態の変更
- EC2 スポットフリートのスポットインスタンスリクエストの変更
- EC2 スポットフリートインスタンスの変更
- EC2 スポットフリート情報
- EC2 スポットフリートのエラー

EC2 スポットフリートの状態の変更

スポットフリートは、スポットフリートの状態が変更されたときに EC2 Spot Fleet State Change イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "d1af1091-6cc3-2e24-203a-3b870e455d5b",
    "detail-type": "EC2 Spot Fleet State Change",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T08:57:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-4b6d274d-0cea-4b2c-
b3be-9dc627ad1f55"
    ],
    "detail": {
        "sub-type": "submitted"
    }
}
```

sub-type に指定できる値は、次のとおりです。

`active`

このスポットフリートリクエストは検証済みです。Amazon EC2 は実行中のインスタンスを目標数分、確保しようとしています。

`cancelled`

このスポットフリートリクエストはキャンセルされており、実行中のインスタンスはありません。スポットフリートは、そのインスタンスが終了されてから 2 日後に削除されます。

`cancelled_running`

このスポットフリートリクエストはキャンセルされており、追加のインスタンスは起動されません。その既存のインスタンスは、中断または終了されるまで実行され続けます。リクエストは、すべてのインスタンスが中断されるか終了されるまで、この状態のままになります。

`cancelled_terminating`

このスポットフリートリクエストはキャンセルされており、対象のインスタンスを終了中です。リクエストは、すべてのインスタンスが終了されるまで、この状態のままになります。

`expired`

スポットフリートリクエストの有効期限が切れました。このリクエストが `TerminateInstancesWithExpiration` セットを使用して作成されている場合、後続の `terminated` イベントは、インスタンスが終了済みなことを示します。

`modify_in_progress`

スポットフリートリクエストは変更中です。リクエストは、この変更が完全に処理されるまで、同じ状態を維持します。

`modify_succeeded`

スポットフリートリクエストが変更されました。

`submitted`

スポットフリートリクエストは評価中です。Amazon EC2 は目標数のインスタンスを起動する準備をしています。

`progress`

スポットフリートリクエストは受理中です。

EC2 スポットフリートのスポットインスタンスリクエストの変更

スポットフリートは、フリート内のスポットインスタンスリクエストの状態が変更されたときに `EC2 Spot Fleet Spot Instance Request Change` イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
```

```
    "id": "cd141ef0-14af-d670-a71d-fe46e9971bd2",
    "detail-type": "EC2 Spot Fleet Spot Instance Request Change",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T08:53:21Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-
a98d2133-941a-47dc-8b03-0f94c6852ad1"
    ],
    "detail": {
        "spot-instance-request-id": "sir-a2w9gc5h",
        "description": "SpotInstanceRequestId sir-a2w9gc5h, PreviousState:
 cancelled_running",
        "sub-type": "cancelled"
    }
}
```

sub-type に指定できる値は、次のとおりです。

active

スポットインスタンスリクエストは受理された状態であり、スポットインスタンスの関連付けが完了しています。

cancelled

スポットインスタンスリクエストがキャンセルされている、あるいは、そのリクエストの有効期限が切れています。

disabled

スポットインスタンスが停止されています。

submitted

スポットインスタンスリクエストは送信済みです。

EC2 スポットフリートインスタンスの変更

スポットフリートは、フリート内のインスタンスの状態が変更されたときに EC2 Spot Fleet Instance Change イベントを Amazon EventBridge に送信します。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "11591686-5bd7-bbaa-eb40-d46529c2710f",
    "detail-type": "EC2 Spot Fleet Instance Change",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T07:25:02Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-c8a764a4-bedc-4b62-
af9c-0095e6e3ba61"
    ],
    "detail": {
        "instance-id": "i-08b90df1e09c30c9b",
        "description": "{\"instanceType\":\"r4.2xlarge\",\"image\":
\"ami-032930428bf1abbff\",\"productDescription\":\"Linux/UNIX\",\"availabilityZone\":
\"us-east-1a\"}",
        "sub-type": "launched"
    }
}
```

sub-type に指定できる値は、次のとおりです。

`launched`

新しいインスタンスが起動されました。

`terminated`

このインスタンスは終了しています。

`termination_notified`

フリートのターゲット容量のスケールダウン中 (ターゲット容量 が 4 から 3 に変更される場合など) に、Amazon EC2 によってスポットインスタンスが終了されたので、インスタンス終了通知が送信されました。

EC2 スポットフリート情報

スポットフリートは、受理中にエラーが発生したときに EC2 Spot Fleet Information イベントを Amazon EventBridge に送信します。情報イベントは、フリートがターゲット容量を満たすことをブロックしません。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "73a60f70-3409-a66c-635c-7f66c5f5b669",
    "detail-type": "EC2 Spot Fleet Information",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-08T20:56:12Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/sfr-2531ea06-
af18-4647-8757-7d69c94971b1"
    ],
    "detail": {
        "description": "r3.8xlarge, ami-032930428bf1abbff, Linux/UNIX, us-east-1a, Spot
 bid price is less than Spot market price $0.5291",
        "sub-type": "launchSpecUnusable"
    }
}
```

sub-type に指定できる値は、次のとおりです。

fleetProgressHalted

すべての起動仕様の料金は、スポット料金を下回っているため無効です (すべての起動仕様が launchSpecUnusable イベントを生成しました)。スポット料金が変更されると、起動仕様が有効になる場合があります。

launchSpecTemporarilyBlacklisted

設定が有効ではなく、インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

launchSpecUnusable

この起動仕様の料金は、スポット料金を下回っているため無効です。

registerWithLoadBalancersFailed

ロードバランサーにインスタンスを登録しようとして失敗しました。詳細については、イベントの説明をご覧ください。

EC2 スポットフリートのエラー

スポットフリートは、受理中にエラーが発生したときに EC2 Spot Fleet Error イベントを Amazon EventBridge に送信します。エラーイベントは、フリートがターゲット容量を満たすことをブロックします。

以下はこのイベントのサンプルデータです。

```
{
    "version": "0",
    "id": "10adc4e7-675c-643e-125c-5bfa1b1ba5d2",
    "detail-type": "EC2 Spot Fleet Error",
    "source": "aws.ec2spotfleet",
    "account": "123456789012",
    "time": "2020-11-09T06:56:07Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ec2:us-east-1:123456789012:spot-fleet-request/
sfr-38725d30-25f1-4f30-83ce-2907c56dba17"
    ],
    "detail": {
        "description": "r4.2xlarge, ami-032930428bf1abbff, Linux/UNIX: The
 associatePublicIPAddress parameter can only be specified for the network interface
 with DeviceIndex 0. ",
        "sub-type": "spotFleetRequestConfigurationInvalid"
    }
}
```

sub-type に指定できる値は、次のとおりです。

iamFleetRoleInvalid

このスポットフリートには、インスタンスの起動または終了に必要なアクセス許可がありません。

allLaunchSpecsTemporarilyBlacklisted

有効な設定はありません。インスタンスを起動しようとして何回か失敗しました。詳細については、イベントの説明をご覧ください。

spotInstanceCountLimitExceeded

起動できるスポットインスタンスの数の上限に達しました。

`spotFleetRequestConfigurationInvalid`

設定が有効ではありません。詳細については、イベントの説明をご覧ください。

Amazon EventBridge ルールを作成する

EC2 フリートもしくはスポットフリート の状態変更に関する通知が送信されると、その通知のためのイベントが Amazon EventBridge に対し送信されます。EventBridge がルールで定義されているパターンに一致するイベントパターンを検出すると、EventBridge はルールで指定されているターゲットを呼び出します。

EventBridge ルールを作成し、イベントパターンがルールに一致したときに実行するアクションを自動化できます。

トピック

- EC2 フリート イベントをモニタリングする Amazon EventBridge を作成する
- スポットフリートイベントをモニタリングする Amazon EventBridge の作成

EC2 フリート イベントをモニタリングする Amazon EventBridge を作成する

EC2 フリートの状態変更に関する通知が送信されると、その通知のためのイベントが、JSON ファイルとして Amazon EventBridge に対し送信されます。EventBridge ルールを作成し、イベントパターンがルールに一致したときに実行するアクションを自動化できます。EventBridge がルールで定義されているパターンに一致するイベントパターンを検出すると、EventBridge はルールで指定されているターゲットを呼び出します。

次のフィールドは、ルールで定義されているイベントパターンになります。

`"source": "aws.ec2fleet"`

イベントが EC2 フリート からのものであることを特定します。

`"detail-type": "EC2 Fleet State Change"`

イベントタイプを特定します。

`"detail": { "sub-type": "submitted" }`

イベントのサブタイプを特定します。

EC2 フリートのイベントの一覧とイベントデータの例については、「」を参照してください。the section called “EC2 フリート イベントタイプ”

例

- 通知を送信する EventBridge ルールを作成する
- Lambda 関数をトリガーする EventBridge ルールの作成

通知を送信する EventBridge ルールを作成する

次の例では、Amazon EC2 が EC2 フリート状態変更通知を発するたびに、E メール、テキストメッセージ、またはモバイルプッシュ通知を送信する EventBridge ルールを作成します。この例のシグナルは `EC2 Fleet State Change` イベントとして送信され、ルールによって定義されたアクションがトリガーされます。

EventBridge ルールを作成する前に、E メール、テキストメッセージ、またはモバイルプッシュ通知用の Amazon SNS トピックを作成する必要があります。

EventBridge ルールを作成して EC2 フリート状態が変更されたときに通知を送信するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. [Create rule] を選択します。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択します。アカウント内の AWS のサービスで生成されたイベントは、常に、そのアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。
   d. [Next] を選択します。
4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。
   a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択します。
   b. この例では [Event pattern] (イベントパターン) で、`EC2 Fleet Instance Change` イベントと一致するように以下のイベントパターンを指定します。

```
{
 "source": ["aws.ec2fleet"],
 "detail-type": ["EC2 Fleet Instance Change"]
}
```

イベントパターンを追加するには、以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには、以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択します。

B. [イベントパターンフォーム] では、AWS[サービス] を選択します。

C. [AWS Service] (サービス) で、[EC2 Fleet] (EC2 フリート) を選択します。

D. [Event type] (イベントタイプ) で、[EC2 Fleet Instance Change] (EC2 フリートインスタンスの変更) を選択します。

E. テンプレートをカスタマイズするには、[Edit pattern] (パターンを編集) を選択した上で、この例のイベントパターンに合わせた変更を行います。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択します。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択します。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. ターゲットタイプ] では、AWSサービス] を選択します。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択します。

c. [Topic (トピック)] で、既存のトピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択します。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

7. [Review and create] (確認して作成) で、以下の操作を行いす。

a. ルールの詳細を確認し、必要な場合は変更を行います。

b. [ルールを作成] を選択します。

詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge ルール」と「Amazon EventBridge イベントパターン」を参照してください。

Lambda 関数をトリガーする EventBridge ルールの作成

次の例では、Amazon EC2 が EC2 フリートインスタンスが起動したときのインスタンス変更通知を発するたびに Lambda 関数をトリガーする EventBridge ルールを作成します。この例のシグナルは `EC2 Fleet Instance Change` イベント、サブタイプ `launched` として発され、ルールによって定義されたアクションがトリガーされます。

EventBridge ルールを作成する前に、Lambda 関数を作成する必要があります。

EventBridge ルールで使用する Lambda 関数を作成するには

1. AWS Lambda コンソール (https://console.aws.amazon.com/lambda/) を開きます。
2. [関数の作成] を選択します。
3. 関数の名前を入力し、コードを設定し、[Create function (関数の作成)] を選択します。

Lambda の使用の詳細については、AWS Lambda デベロッパーガイドの「コンソールで Lambda 関数を作成する」を参照してください。

EC2 フリート のインスタンスの状態が変更されたときに Lambda 関数をトリガーする EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. [Create rule] を選択します。

3. [Define rule detail] (詳細の定義) で、次の操作を行います。

   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。

   b. [イベントバス] として、[デフォルト] を選択します。アカウント内の AWS のサービスで生成されたイベントは、常に、そのアカウントのデフォルトのイベントバスに送られます。

   c. ルールタイプ では、イベントパターンを持つルール] を選択します。

   d. [Next] を選択します。

4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。

   a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択します。

   b. この例では [Event pattern] (イベントパターン) で `EC2 Fleet Instance Change` イベントと `launched` サブタイプに一致するよう、以下のイベントパターンを指定します。

```
{
 "source": ["aws.ec2fleet"],
 "detail-type": ["EC2 Fleet Instance Change"],
 "detail": {
   "sub-type": ["launched"]
}
```

      イベントパターンを追加するには、以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

      i. テンプレートを使用してイベントパターンを作成するには、以下の操作を行います。

         A. [Event pattern form] (イベントパターンフォーム) を選択します。

         B. [イベントパターンフォーム] では、AWS[サービス] を選択します。

         C. [AWS Service] (サービス) で、[EC2 Fleet] (EC2 フリート) を選択します。

         D. [Event type] (イベントタイプ) で、[EC2 Fleet Instance Change] (EC2 フリートインスタンスの変更) を選択します。

E. [Edit pattern] (パターンの編集) を選択し、サンプルのイベントパターンに一致するように `"detail": {"sub-type": ["launched"]` を追加します。適切な JSON 形式を得るには、前にある角括弧 (]) の後にコンマ (,) を入力します。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択します。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択します。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. ターゲットタイプ] では、AWSサービス] を選択します。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択します。

c. [Topic] (トピック) で [Lambda function] (Lambda 関数) を選択し、[Function] (関数) では、イベント発生時に応答するように作成した関数を選択します。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択します。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

7. [Review and create] (確認して作成) で、以下の操作を行いす。

a. ルールの詳細を確認し、必要な場合は変更を行います。

b. [ルールを作成] を選択します。

Lambda 関数を作成する方法のチュートリアルと Lambda 関数を実行する EventBridge ルールについては、AWS Lambda デベロッパーガイドの「チュートリアル: EventBridge を使用して Amazon EC2 インスタンスの状態をログに記録する」を参照してください。

スポットフリートイベントをモニタリングする Amazon EventBridge の作成

スポットフリートの状態変更に関する通知が送信されると、その通知に関するイベントが、JSON ファイルとして Amazon EventBridge に対し送信されます。EventBridge ルールを作成し、イベントパターンがルールに一致したときに実行するアクションを自動化できます。EventBridge がルールで定義されているパターンに一致するイベントパターンを検出すると、EventBridge はルールで指定されているターゲットを呼び出します。

次のフィールドは、ルールで定義されているイベントパターンになります。

`"source": "aws.ec2spotfleet"`

イベントがスポットフリートからのものであることを特定します

`"detail-type": "EC2 Spot Fleet State Change"`

イベントタイプを特定します。

`"detail": { "sub-type": "submitted" }`

イベントのサブタイプを特定します。

スポットフリートのイベントの一覧とイベントデータの例については、「」を参照してください。the section called “スポットフリートイベントタイプ”

例

- 通知を送信する EventBridge ルールを作成する
- Lambda 関数をトリガーする EventBridge ルールの作成

通知を送信する EventBridge ルールを作成する

次の例では、Amazon EC2 がスポットフリート状態変更通知を発するたびに、E メール、テキストメッセージ、またはモバイルプッシュ通知を送信する EventBridge ルールを作成します。この例のシグナルは `EC2 Spot Fleet State Change` イベントとして送信され、ルールによって定義されたアクションがトリガーされます。EventBridge ルールを作成する前に、E メール、テキストメッセージ、またはモバイルプッシュ通知用の Amazon SNS トピックを作成する必要があります。

スポットフリート状態が変更されたときに通知を送信する EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。

2. [Create rule] を選択します。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。

   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。

   b. [イベントバス] として、[デフォルト] を選択します。アカウント内の AWS のサービスで生成されたイベントは、常に、そのアカウントのデフォルトのイベントバスに送られます。

   c. ルールタイプ では、イベントパターンを持つルール] を選択します。

   d. [Next] を選択します。
4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。

   a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択します。

   b. この例では [Event pattern] (イベントパターン) で、`EC2 Spot Fleet Instance Change` イベントと一致するように以下のイベントパターンを指定します。

      ```
      {
       "source": ["aws.ec2spotfleet"],
       "detail-type": ["EC2 Spot Fleet Instance Change"]
      }
      ```

      イベントパターンを追加するには、以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

      i. テンプレートを使用してイベントパターンを作成するには、以下の操作を行います。

         A. [Event pattern form] (イベントパターンフォーム) を選択します。

         B. [イベントパターンフォーム] では、AWS[サービス] を選択します。

         C. [AWS Service] (サービス) で、[EC2 Spot Fleet] (EC2 スポットフリート) を選択します。

         D. [Event type] (イベントタイプ) で、[EC2 Spot Fleet Instance Change] (EC2 スポットフリートインスタンスの変更) を選択します。

         E. テンプレートをカスタマイズするには、[Edit pattern] (パターンを編集) を選択した上で、この例のイベントパターンに合わせた変更を行います。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択します。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択します。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. ターゲットタイプ] では、AWSサービス] を選択します。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択します。

c. [Topic (トピック)] で、既存のトピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択します。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

7. [Review and create] (確認して作成) で、以下の操作を行いす。

a. ルールの詳細を確認し、必要な場合は変更を行います。

b. [ルールを作成] を選択します。

詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge ルール」と「Amazon EventBridge イベントパターン」を参照してください。

Lambda 関数をトリガーする EventBridge ルールの作成

次の例では、Amazon EC2 がスポットフリートインスタンスが起動したときのインスタンス変更通知を発するたびに Lambda 関数をトリガーする EventBridge ルールを作成します。この例のシグナ

ルは EC2 Spot Fleet Instance Change イベント、サブタイプ launched として発され、ルールによって定義されたアクションがトリガーされます。

EventBridge ルールを作成する前に、Lambda 関数を作成する必要があります。

EventBridge ルールで使用する Lambda 関数を作成するには

1. AWS Lambda コンソール (https://console.aws.amazon.com/lambda/) を開きます。
2. [関数の作成] を選択します。
3. 関数の名前を入力し、コードを設定し、[Create function (関数の作成)] を選択します。

   Lambda の使用の詳細については、AWS Lambda デベロッパーガイドの「コンソールで Lambda 関数を作成する」を参照してください。

スポットフリートのインスタンスの状態が変更されたときに Lambda 関数をトリガーする EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. [Create rule] を選択します。
3. [Define rule detail] (詳細の定義) で、次の操作を行います。
   a. ルールの [Name (名前)] を入力し、必要に応じて説明を入力します。

      ルールには、同じリージョン内および同じイベントバス上の別のルールと同じ名前を付けることはできません。
   b. [イベントバス] として、[デフォルト] を選択します。アカウント内の AWS のサービスで生成されたイベントは、常に、そのアカウントのデフォルトのイベントバスに送られます。
   c. ルールタイプ では、イベントパターンを持つルール] を選択します。
   d. [Next] を選択します。
4. [Build event pattern] (イベントパターンの作成) で、次の操作を行います。
   a. [Event source] (イベントソース) で、[AWS events or EventBridge partner events] (イベントまたは EventBridge パートナーイベント) を選択します。
   b. この例では [Event pattern] (イベントパターン) で EC2 Spot Fleet Instance Change イベントと launched サブタイプに一致するよう、以下のイベントパターンを指定します。

```
{
 "source": ["aws.ec2spotfleet"],
 "detail-type": ["EC2 Spot Fleet Instance Change"],
 "detail": {
   "sub-type": ["launched"]
}
```

イベントパターンを追加するには、以下のように [Event pattern form] (イベントパターンフォーム) を選択してテンプレートを使用するか、[Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択して独自のパターンを指定します。

i. テンプレートを使用してイベントパターンを作成するには、以下の操作を行います。

A. [Event pattern form] (イベントパターンフォーム) を選択します。

B. [イベントパターンフォーム] では、AWS[サービス] を選択します。

C. [AWS Service] (サービス) で、[EC2 Spot Fleet] (EC2 スポットフリート) を選択します。

D. [Event type] (イベントタイプ) で、[EC2 Spot Fleet Instance Change] (EC2 スポットフリートインスタンスの変更) を選択します。

E. [Edit pattern] (パターンの編集) を選択し、イベントパターン例と一致するよう、`"detail": {"sub-type": ["launched"]` を追加します。適切な JSON 形式を得るには、前にある角括弧 (]) の後にコンマ (,) を入力します。

ii. (代替案) 以下の操作を行って、カスタムイベントパターンを指定します。

A. [Custom pattern (JSON editor)] (カスタムパターン (JSON エディター)) を選択します。

B. [Event pattern] (イベントパターン) ボックスに、この例のイベントパターンを追加します。

c. [Next] を選択します。

5. [Select target(s)] (ターゲットを選択) で、以下の操作を行います。

a. ターゲットタイプ] では、AWSサービス] を選択します。

b. イベントの発生時に E メール、テキストメッセージ、またはモバイルプッシュ通知を送信するために、[Select a target] (ターゲットを選択) で、[SNS topic] (SNS トピック) を選択します。

c. [Topic] (トピック) で [Lambda function] (Lambda 関数) を選択し、[Function] (関数) では、イベント発生時に応答するように作成した関数を選択します。

d. (オプション) [Additional settings] (追加設定) で、その他の設定を行うこともできます。詳細については、「Amazon EventBridge ユーザーガイド」の「イベントに反応する Amazon EventBridge ルールの作成」(ステップ 16) を参照してください。

e. [Next] を選択します。

6. (オプション) 必要な場合は、[Tags] (タグ) で 1 つ以上のタグを作成したルールに割り当て、[Next] (次へ) を選択します。

7. [Review and create] (確認して作成) で、以下の操作を行います。

a. ルールの詳細を確認し、必要な場合は変更を行います。

b. [ルールを作成] を選択します。

Lambda 関数を作成する方法のチュートリアルと Lambda 関数を実行する EventBridge ルールについては、AWS Lambda デベロッパーガイドの「チュートリアル: EventBridge を使用して Amazon EC2 インスタンスの状態を口グに記録する」を参照してください。

EC2 フリートとスポットフリートのチュートリアル

以下のチュートリアルでは、EC2 フリートとスポットフリートを作成するための一般的なプロセスを説明します。

チュートリアル

- チュートリアル: EC2 フリートを使ったインスタンスの分量指定
- チュートリアル: プライマリ容量としてオンデマンドの EC2 フリート を使用する
- チュートリアル:ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動する
- チュートリアル: キャパシティブロックでインスタンスを起動する
- チュートリアル: スポットフリートを使ったインスタンスの分量の指定

チュートリアル: EC2 フリートを使ったインスタンスの分量指定

このチュートリアルでは、サンプル株式会社という名前の架空の会社を使用して、インスタンスの分量指定を使った EC2 フリートリクエストのプロセスを説明します。

目的

製薬会社であるサンプル株式会社は、癌と闘うために使用できる可能性のある化合物をスクリーニングするために Amazon EC2 の計算処理能力を活用したいと考えています。

計画

サンプル株式会社はまず、「Spot Best Practices」を確認します。次に、サンプル株式会社は EC2 フリート に関する要件を確認します。

インスタンスタイプ

サンプル株式会社には、60 GB 以上のメモリと 8 つの仮想 CPU (vCPU) で最適に実行される、計算能力とメモリに負担がかかるアプリケーションがあります。同社は、できるだけ低価格でアプリケーション用のこれらのリソースを最大化したいと考えています。サンプル株式会社は、以下のいずれかの EC2 インスタンスタイプがそのニーズを満たすと判断します。

インスタンスタイプ	メモリ (GiB)	vCPU
r3.2xlarge	61	8
r3.4xlarge	122	16
r3.8xlarge	244	32

ユニット単位のターゲット容量

インスタンスの分量指定を使用すると、ターゲット容量はインスタンスの数 (デフォルト)、またはコア (vCPU)、メモリ (GiB) とストレージ (GB) との要素の組み合わせで表すことができます。アプリケーションのベース (60 GB の RAM と 8 個の vCPU) を 1 ユニットとして考えることで、サンプル株式会社はこの量の 20 倍で十分ニーズに合うと決定します。これにより、会社は EC2 フリート リクエストのターゲット容量を 20 に設定します。

インスタンスの分量

ターゲット容量の決定後、サンプル株式会社はインスタンスの分量を計算します。各インスタンスタイプのインスタンスの分量を計算することは、以下のように、ターゲット容量に達するために必要な各インスタンスタイプのユニットの数を決定することです。

- r3.2xlarge (61.0 GB、8 個の vCPU) = 1/20 ユニット
- r3.4xlarge (122.0 GB、16 個の vCPU) = 2/20 ユニット
- r3.8xlarge (244.0 GB、32 個の vCPU) = 4/20 ユニット

これよりサンプル株式会社は、1、2 と 4 のインスタンス分量を EC2 フリート リクエストのそれぞれの起動設定に割り当てます。

ユニット時間あたりの価格

サンプル株式会社は、料金の出発点としてインスタンス時間あたりの「オンデマンド料金」を使用します。最近のスポット料金または 2 つの組み合わせを使用することもできます。ユニット時間あたりの料金を計算するために、インスタンス時間あたりの出発点の料金を分量で割ります。次に例を示します。

インスタンスタイプ	オンデマンド価格	インスタンスの分量	ユニット時間あたりの価格
r3.2xLarge	$0.7	1	$0.7
r3.4xLarge	$1.4	2	$0.7
r3.8xLarge	$2.8	4	$0.7

サンプル株式会社は、ユニット時間あたりのグローバルな料金として 0.7 USD を使用し、3 つのインスタンスタイプすべてで競争力を高めることもできます。また、`r3.8xlarge` の起動条件のなかで、1 ユニット時間あたりの全体料金を 0.7 USD、そして 1 ユニット時間あたりの指定入力料金を 0.9 USD とすることもできます。

アクセス許可の確認

EC2 フリート を作成する前に、サンプル株式会社は必要なアクセス許可の IAM ロールがあることを確認します。詳細については、「EC2 フリートの前提条件」を参照してください。

起動テンプレートの作成

次に、Example Corp は起動テンプレートを作成します。起動テンプレート ID は、次のステップで使用されます。詳細については、「起動テンプレートの作成」を参照してください。

EC2 フリート の作成

サンプル株式会社は、その EC2 フリート用に次の設定を使用して config.json ファイルを作成します。次の例では、リソース識別子を独自のリソース識別子に置き換えます。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-07b3bc7625cdab851",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "r3.2xlarge",
                    "SubnetId": "subnet-482e4972",
                    "WeightedCapacity": 1
                },
                {
                    "InstanceType": "r3.4xlarge",
                    "SubnetId": "subnet-482e4972",
                    "WeightedCapacity": 2
                },
                {
                    "InstanceType": "r3.8xlarge",
                    "MaxPrice": "0.90",
                    "SubnetId": "subnet-482e4972",
                    "WeightedCapacity": 4
                }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "DefaultTargetCapacityType": "spot"
    }
}
```

サンプル株式会社は、次の create-fleet コマンドを使用して EC2 フリート を作成します。https://docs.aws.amazon.com/cli/latest/reference/ec2/create-fleet.html

```
aws ec2 create-fleet \
    --cli-input-json file://config.json
```

詳細については、「EC2 フリートの作成」を参照してください。

フルフィルメント

配分戦略では、スポットインスタンスの提供元となるスポットキャパシティプールが決定されます。

lowest-price 戦略 (デフォルトの戦略) では、受理時にユニットあたりの料金が最低値であるプールから スポットインスタンス が取得されます。20 ユニットの容量を提供するためには、20 個の r3.2xlarge インスタンス (20 ÷ 1)、10 個の r3.4xlarge インスタンス (20 ÷ 2)、あるいは 5 個の r3.8xlarge インスタンス (20 ÷ 4) が EC2 フリート から起動されることになります。

サンプル株式会社が diversified 戦略を採用する場合、スポットインスタンス は 3 つのすべてのプールから取得されます。EC2 フリート は、6 個の r3.2xlarge インスタンス (6 ユニットを提供)、3 個の r3.4xlarge インスタンス (6 ユニットを提供)、そして 2 個の r3.8xlarge インスタンス (8 ユニットを提供) の全部で 20 ユニットを起動します。

チュートリアル: プライマリ容量としてオンデマンドの EC2 フリート を使用する

このチュートリアルでは、ABC Online という架空の会社を使用して、プライマリ容量および使用可能な場合はスポット容量としてオンデマンドの EC2 フリートをリクエストするプロセスを説明します。

目的

レストラン向け配達会社である ABC Online は、EC2 インスタンスタイプおよび購入オプション間で Amazon EC2 容量をプロビジョンし、必要なスケール、パフォーマンス、コストを実現したいと思っています。

計画

ABC Online は、ピーク期間中の運用のために固定容量を要求していますが、低価格での容量増加という恩恵を得たいと考えています。ABC Online は、EC2 フリートについて以下の要件を設定しました。

- オンデマンドインスタンス容量 - ABC Online には、ピーク期間のトラフィックに対応できるように、15 個のオンデマンドインスタンスが必要です。
- スポットインスタンス容量 - ABC Online は、5 個のスポットインスタンスをプロビジョニングすることで、低価格でパフォーマンスを改善したいと考えています。

アクセス許可の確認

EC2 フリート を作成する前に、ABC Online は必要なアクセス許可の IAM ロールがあることを確認します。詳細については、「EC2 フリートの前提条件」を参照してください。

起動テンプレートの作成

次に、ABC Online によって起動テンプレートが作成されます。起動テンプレート ID は、次のステップで使用されます。詳細については、「起動テンプレートの作成」を参照してください。

EC2 フリート の作成

ABC Online は、その EC2 フリート用に次の設定を使用して config.json ファイルを作成します。次の例では、リソース識別子を独自のリソース識別子に置き換えます。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-07b3bc7625cdab851",
                "Version": "2"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 20,
        "OnDemandTargetCapacity":15,
        "DefaultTargetCapacityType": "spot"
    }
}
```

ABC Online は、次の create-fleet コマンドを使用して EC2 フリート を作成します。https://docs.aws.amazon.com/cli/latest/reference/ec2/create-fleet.html

```
aws ec2 create-fleet \
```

```
    --cli-input-json file://config.json
```

詳細については、「EC2 フリートの作成」を参照してください。

フルフィルメント

配分戦略により、オンデマンド容量が常に受理され、容量と可用性がある場合はターゲット容量がスポットとして受理されることが決定されます。

チュートリアル:ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動する

このチュートリアルでは実行するべきステップを段階的に解説しています。それにより、EC2 フリートがオンデマンドインスタンスをtargetedキャパシティー予約で起動できるようにします。

オンデマンドインスタンスの起動時に、最初にtargetedオンデマンドキャパシティー予約を使用するようにフリートを設定する方法を学習します。また、オンデマンドターゲット容量の合計が使用可能な未使用のキャパシティ予約数を超えた場合、フリートは指定された割り当て戦略を使用して、残りのターゲット容量を起動するインスタンスプールを選択するようにフリートを設定する方法についても学習します。

EC2 フリートの設定

このチュートリアルでは、フリートの設定は次のとおりです：

- ターゲット容量:10 オンデマンドインスタンス
- 未使用の合計targetedキャパシティー予約:6 (フリートの目標オンデマンド容量である 10 オンデマンドインスタンスを下回っています)
- キャパシティ予約のプールの数:2 (us-east-1aおよびus-east-1b)
- プールあたりのキャパシティ予約数:3
- オンデマンド割り当て戦略：lowest-price(未使用キャパシティーの予約の数が目標オンデマンド容量より少ない場合、フリートは、オンデマンド配分戦略に基づいて、残りのオンデマンド容量を起動するプールを決定します)。

  また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

targetedキャパシティ予約にオンデマンドインスタンスを起動するには、次のように、いくつかの手順を実行する必要があります：

ステップ 1: キャパシティー予約を作成する

キャパシティー予約の作成コマンドを使用してキャパシティ予約を作成します。3 つはus-east-1aの目的に、別の3つはus-east-1bの目的にします。アベイラビリティーゾーンを除き、キャパシティー予約の他の属性は同じです。

us-east-1aでの 3 つのキャパシティー予約

```
aws ec2 create-capacity-reservation \
    --availability-zone us-east-1a\
    --instance-type c5.xlarge\
    --instance-platform Linux/UNIX \
    --instance-count 3 \
    --instance-match-criteria targeted
```

キャパシティー予約 ID の結果の例

```
cr-1234567890abcdef1
```

us-east-1bでの 3 つのキャパシティー予約

```
aws ec2 create-capacity-reservation \
    --availability-zone us-east-1b\
    --instance-type c5.xlarge\
    --instance-platform Linux/UNIX \
    --instance-count 3 \
```

```
    --instance-match-criteria targeted
```

キャパシティー予約 ID の結果の例

```
cr-54321abcdef567890
```

ステップ 2: キャパシティー予約のリソースグループを作成する

resource-groupsサービスを使用する、およびグループを作成するコマンドを使用して、キャパシティー予約のリソースグループを作成します。この例では、プレイスメントグループ名はmy-cr-groupです。リソースグループを作成する必要がある理由の詳細については、オンデマンドインスタンスのためのキャパシティ予約の使用を参照してください。

```
aws resource-groups create-group \
    --name my-cr-group \
    --configuration '{"Type":"AWS::EC2::CapacityReservationPool"}'
 '{"Type":"AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-resource-
types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

ステップ 3: キャパシティ予約リソースグループにキャパシティ予約を追加する

resource-groupsサービス、およびグループリソースコマンドを使用して、手順 1 で作成したキャパシティ予約をキャパシティ予約リソースグループに追加します。オンデマンドキャパシティ予約は、ARN ごとに参照する必要があります。

```
aws resource-groups group-resources \
    --group my-cr-group \
    --resource-arns \
      arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1 \
      arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-54321abcdef567890
```

出力例

```
{
   "Failed": [],
   "Succeeded": [
   "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-1234567890abcdef1",
   "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/cr-54321abcdef567890"
   ]
```

```
}
```

(オプション) ステップ 4: リソースグループのキャパシティーの予約を表示する

resource-groupsサービス、およびグループリソースを表示するコマンドを使用して、キャパシティ予約を表示するリソースグループをオプションで記述します。

```
aws resource-groups list-group-resources --group my-cr-group
```

出力例

```
{
    "ResourceIdentifiers": [
        {
            "ResourceType": "AWS::EC2::CapacityReservation",
            "ResourceArn": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/
cr-1234567890abcdef1"
        },
        {
            "ResourceType": "AWS::EC2::CapacityReservation",
            "ResourceArn": "arn:aws:ec2:us-east-1:123456789012:capacity-reservation/
cr-54321abcdef567890"
        }
    ]
}
```

ステップ 5: キャパシティ予約が特定のリソースグループをターゲットに指定する起動テンプレートを作成する

起動テンプレートを作成するコマンドを使用して、使用するキャパシティー予約を指定する起動テンプレートを作成します。この例では、フリートはtargetedキャパシティー予約を使用して、ソースグループに追加されます。したがって、起動テンプレートデータでは、キャパシティ予約が特定のリソースグループをターゲットに指定します。次の例で、プレイスメントグループ名はmy-launch-templateです。

```
aws ec2 create-launch-template \
    --launch-template-name my-launch-template \
    --launch-template-data \
        '{"ImageId": "ami-0123456789example",
          "CapacityReservationSpecification":
            {"CapacityReservationTarget":
```

```
                { "CapacityReservationResourceGroupArn": "arn:aws:resource-groups:us-
east-1:123456789012:group/my-cr-group" }
            }
        }'
```

(オプション) ステップ 6: 起動テンプレートを説明する

起動テンプレートを説明するコマンドを使用して、オプションで、その設定を表示するための起動テンプレートを説明します。

```
aws ec2 describe-launch-template-versions --launch-template-name my-launch-template
```

出力例

```
{
    "LaunchTemplateVersions": [
        {
            "LaunchTemplateId": "lt-01234567890example",
            "LaunchTemplateName": "my-launch-template",
            "VersionNumber": 1,
            "CreateTime": "2021-01-19T20:50:19.000Z",
            "CreatedBy": "arn:aws:iam::123456789012:user/Admin",
            "DefaultVersion": true,
            "LaunchTemplateData": {
                "ImageId": "ami-0947d2ba12ee1ff75",
                "CapacityReservationSpecification": {
                    "CapacityReservationTarget": {
                        "CapacityReservationResourceGroupArn": "arn:aws:resource-
groups:us-east-1:123456789012:group/my-cr-group"
                    }
                }
            }
        }
    ]
}
```

ステップ 7: EC2 フリートを作成する

起動するインスタンスの設定情報を指定する EC2 フリートを作成します。以下の EC2 フリート設定は、この例に関連する設定のみを示しています。起動テンプレートmy-launch-templateは、ステップ 5 で作成した起動テンプレートです。2 つのインスタンスプールがあり、それぞれ同じイ

ンスタンスタイプ (c5.xlarge)ですが、異なるアベイラビリティーゾーン (us-east-1aおよびus-east-1b)にあります。料金は、アベイラビリティーゾーンごとではなく、リージョンに対して定義されるため、インスタンスプールの料金は同じです。合計ターゲット容量は 10 で、デフォルトのターゲット容量タイプはon-demandです。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

> **Note**
>
> フリートタイプはinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
            "Overrides": [
               {
                   "InstanceType": "c5.xlarge",
                   "AvailabilityZone": "us-east-1a"
               },
               {
                    "InstanceType": "c5.xlarge",
                    "AvailabilityZone": "us-east-1b"
               }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price",
        "CapacityReservationOptions": {
             "UsageStrategy": "use-capacity-reservations-first"
         }
    },
    "Type": "instant"
```

```
}
```

上記の設定を使用してinstantフリートを作成すると、目標容量を満たすために以下の 10 個のインスタンスが起動されます。

- 次のように、6 つのオンデマンドインスタンスを起動するために、キャパシティ予約が最初に使用されます。
  - 3 つのオンデマンドインスタンスが、us-east-1aでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
  - 3 つのオンデマンドインスタンスが、us-east-1bでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
- ターゲット容量を満たすために、4 つの追加のオンデマンドインスタンスは、オンデマンド配分戦略 (この例ではlowest-price) に従って通常のオンデマンド容量で起動します。ただし、プールの価格は同じであるため (価格はアベイラビリティーゾーンごとではなく、リージョンごとであるため)、フリートは残りの 4 つのオンデマンドインスタンスをいずれかのプールで起動します。

(オプション) ステップ 8: 未使用のキャパシティ予約の残りの数を表示する

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティー予約の残りの数を確認できます。この例では、以下のレスポンスが表示されます。これは、すべてのプール内のすべてのキャパシティ予約が使用されたことを示しています。

```
{ "CapacityReservationId":  "cr-111",
     "InstanceType":  "c5.xlarge",
     "AvailableInstanceCount":  0
}

 { "CapacityReservationId":  "cr-222",
     "InstanceType":  "c5.xlarge",
     "AvailableInstanceCount":  0
}
```

チュートリアル: キャパシティブロックでインスタンスを起動する

このチュートリアルでは実行すべきステップを段階的に解説しています。これらのステップを実行すると、EC2 フリートがキャパシティブロックでインスタンスを起動します。

EC2 フリートインスタントタイプを使用して、キャパシティブロックにインスタンスを起動できます。詳細については、「EC2フリート「インスタント」タイプの使用」を参照してください。

ほとんどの場合、EC2 フリートリクエストのターゲットキャパシティは、ターゲットとするキャパシティブロック予約の空き容量以下でなければなりません。キャパシティブロック予約の制限を超えるターゲットキャパシティリクエストは受理されません。ターゲットキャパシティリクエストがキャパシティブロック予約の制限を超えると、キャパシティブロック予約の制限を超える容量に対して、容量不足の例外が発生します。

> **Note**
>
> キャパシティブロックの場合、EC2 フリートは希望するターゲットキャパシティの残りをオンデマンドインスタンスの起動にフォールバックしません。

EC2 フリートが利用可能なキャパシティブロック予約で要求されたターゲットキャパシティを満たすことができない場合、EC2 フリートは可能な限り多くの容量を満たし、起動できたインスタンスを返します。すべてのインスタンスがプロビジョニングされるまで EC2 フリート の呼び出しを繰り返すことができます。

EC2 フリートリクエストを設定したら、キャパシティブロック予約の開始日まで待つ必要があります。まだ開始されていないキャパシティブロックで EC2 フリートに起動するようリクエストすると、容量不足エラーが表示されます。

キャパシティブロック予約が有効になったら、EC2 フリートの API コールを行い、選択したパラメータに基づいてキャパシティブロックにインスタンスをプロビジョニングできます。キャパシティブロックで実行されているインスタンスは、別の Amazon EC2 API コールで停止あるいは終了するまで、またはキャパシティブロック予約が完了して Amazon EC2 がインスタンスを終了するまで実行され続けます。

考慮事項

- 同じ `CreateFleet` リクエストでは、複数のキャパシティブロックはサポートされません。
- `OnDemandTargetCapacity` または `SpotTargetCapacity` を使用しながら `DefaultTargetCapacity` として `capacity-block` を設定することはサポートされていません。
- `DefaultTargetCapacityType` が `capacity-block` に設定されている場合、`OnDemandOptions::CapacityReservationOptions` は提供できません。例外が発生します。

起動テンプレートの作成

起動テンプレート ID は、次のステップで使用されます。詳細については、「起動テンプレートの作成」を参照してください。

起動テンプレートを設定するには、InstanceMarketOptionsRequest に対して MarketType を capacity-block に設定します。CapacityReservationID パラメータを設定して、対象のキャパシティブロック予約 ID を指定します。

EC2 フリート の作成

EC2 フリート 用に次の設定を使用して config.json ファイルを作成します。次の例では、リソース識別子を独自のリソース識別子に置き換えます。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "CBR-launch-template",
                "Version": "1"
            },
            "Overrides": [
                {
                    "InstanceType": "p5.48xlarge",
                    "AvailabilityZone": "us-east-1a"
                },
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "capacity-block"
    },
    "Type": "instant"
}
```

次の create-fleet コマンドを使用します。

```
aws ec2 create-fleet \
     --cli-input-json file://config.json
```

詳細については、「EC2 フリートの作成」を参照してください。

チュートリアル: スポットフリートを使ったインスタンスの分量の指定

このチュートリアルでは、サンプル株式会社という名前の架空の会社を使用して、インスタンス分量指定を使ったスポットフリートリクエストのプロセスを説明します。

目的

製薬会社であるサンプル株式会社は、癌と闘うために使用される可能性のある化合物を選別するために Amazon EC2 の計算処理能力を利用したいと考えています。

計画

サンプル株式会社はまず、「Spot Best Practices」を確認します。次に、サンプル株式会社はスポットフリートに関する以下の要件を確認します。

インスタンスタイプ

サンプル株式会社には、60 GB 以上のメモリと 8 つの仮想 CPU (vCPU) で最適に実行される、計算能力とメモリに負担がかかるアプリケーションがあります。同社は、できるだけ低価格でアプリケーション用のこれらのリソースを最大化したいと考えています。サンプル株式会社は、以下のいずれかの EC2 インスタンスタイプがそのニーズを満たすと判断します。

インスタンスタイプ	メモリ (GiB)	vCPU
r3.2xlarge	61	8
r3.4xlarge	122	16
r3.8xlarge	244	32

ユニット単位のターゲット容量

インスタンスの分量指定を使用すると、ターゲット容量はインスタンスの数 (デフォルト)、またはコア (vCPU)、メモリ (GiB) とストレージ (GB) との要素の組み合わせで表すことができます。アプリケーションのベース (60 GB の RAM と 8 個の vCPU) を 1 ユニットとして考えることで、サンプル株式会社はこの量の 20 倍で十分ニーズに合うと決定します。これにより、会社はスポットフリートリクエストのターゲット容量を 20 に設定します。

インスタンスの分量

ターゲット容量の決定後、サンプル株式会社はインスタンスの分量を計算します。各インスタンスタイプのインスタンスの分量を計算することは、以下のように、ターゲット容量に達するために必要な各インスタンスタイプのユニットの数を決定することです。

- r3.2xlarge (61.0 GB、8 個の vCPU) = 1/20 ユニット
- r3.4xlarge (122.0 GB、16 個の vCPU) = 2/20 ユニット
- r3.8xlarge (244.0 GB、32 個の vCPU) = 4/20 ユニット

その結果、サンプル株式会社は、1、2、4 のインスタンス分量をスポットフリートリクエストの各起動設定に割り当てます。

ユニット時間あたりの価格

サンプル株式会社は、料金の出発点としてインスタンス時間あたりの「オンデマンド料金」を使用します。最近のスポット料金または 2 つの組み合わせを使用することもできます。ユニット時間あたりの料金を計算するために、インスタンス時間あたりの出発点の料金を分量で割ります。次に例を示します。

インスタンスタイプ	オンデマンド価格	インスタンスの分量	ユニット時間あたりの価格
r3.2xLarge	$0.7	1	$0.7
r3.4xLarge	$1.4	2	$0.7
r3.8xLarge	$2.8	4	$0.7

サンプル株式会社は、ユニット時間あたりのグローバルな料金として 0.7 USD を使用し、3 つのインスタンスタイプすべてで競争力を高めることもできます。また、`r3.8xlarge` の起動条件のなかで、1 ユニット時間あたりの全体料金を 0.7 USD、そして 1 ユニット時間あたりの指定入力料金を 0.9 USD とすることもできます。

アクセス許可の確認

スポットフリートのリクエストを作成する前に、サンプル株式会社は必要アクセス許可の IAM ロールがあることを確認します。詳細については、「スポットフリートアクセス許可」を参照してください。

リクエストの作成

サンプル株式会社は、スポットフリートリクエスト用に次の設定を使用して config.json ファイルを作成します。

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
    {
      "ImageId": "ami-1a2b3c4d",
      "InstanceType": "r3.2xlarge",
      "SubnetId": "subnet-482e4972",
      "WeightedCapacity": 1
    },
    {
      "ImageId": "ami-1a2b3c4d",
      "InstanceType": "r3.4xlarge",
      "SubnetId": "subnet-482e4972",
      "WeightedCapacity": 2
    },
    {
      "ImageId": "ami-1a2b3c4d",
      "InstanceType": "r3.8xlarge",
      "SubnetId": "subnet-482e4972",
      "SpotPrice": "0.90",
      "WeightedCapacity": 4
    }
  ]
}
```

サンプル株式会社は、request-spot-fleet コマンドを使用してスポットフリートリクエストを作成します。https://docs.aws.amazon.com/cli/latest/reference/ec2/request-spot-fleet.html

```
aws ec2 request-spot-fleet --spot-fleet-request-config file://config.json
```

詳細については、「スポットフリートのリクエストタイプ」を参照してください。

フルフィルメント

配分戦略では、スポットインスタンスの提供元となるスポットキャパシティプールが決定されます。

lowestPrice 戦略 (デフォルトの戦略) では、受理時にユニットあたりの料金が最低値であるプールから スポットインスタンス が取得されます。20 ユニットの容量を提供するために、スポットフリートは、20 個の r3.2xlarge インスタンス (20 ÷ 1)、10 個の r3.4xlarge インスタンス (20 ÷ 2)、または 5 個の r3.8xlarge インスタンス (20 ÷ 4) を起動します。

サンプル株式会社が diversified 戦略を採用する場合、スポットインスタンス は 3 つのすべてのプールから取得されます。スポットフリートは、6 個の r3.2xlarge インスタンス (6 ユニットを提供)、3 個の r3.4xlarge インスタンス (6 ユニットを提供)、2 個の r3.8xlarge インスタンス (8 ユニットを提供)、合計 20 ユニットを起動します。

EC2 フリートとスポットフリートの設定例

以下の例では、EC2 フリートおよびスポットフリートの作成に使用できる起動設定を示します。

トピック

EC2 フリートの設定例

以下の例で示しているのは、EC2 フリートを作成するために create-fleet コマンドで使用できる起動設定です。パラメータの詳細については、「AWS CLI コマンドリファレンス」の「create-fleet」を参照してください。

例

- 例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する
- 例 11: price-capacity-optimized フリートでスポットインスタンスを起動する
- 例 12: 属性ベースのインスタンスタイプの選択を設定する

例 1: スポットインスタンスをデフォルト購入オプションとして起動する

次の例では、EC2 フリートで必要な最小限のパラメータ (起動テンプレート、ターゲットキャパシティ、デフォルト購入オプション) を指定します。起動テンプレートは、起動テンプレート ID とバージョン番号によって識別されます。フリートのターゲット容量は 2 インスタンスであり、デフォルト購入オプションは spot です。この結果、フリートは 2 スポットインスタンス を起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "DefaultTargetCapacityType": "spot"
    }
}
```

例 2: オンデマンドインスタンスをデフォルト購入オプションとして起動する

次の例では、EC2 フリート で必要な最小限のパラメータ (起動テンプレート、ターゲット容量、デフォルト購入オプション) を指定します。起動テンプレートは、起動テンプレート ID とバージョン番号によって識別されます。フリートのターゲット容量は 2 インスタンスであり、デフォルト購入オプションは on-demand です。この結果、フリートは 2 オンデマンドインスタンス を起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
```

```
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "DefaultTargetCapacityType": "on-demand"
    }
}
```

例 3: オンデマンドインスタンスをプライマリ容量として起動する

次の例では、フリートの合計ターゲット容量 2 インスタンス、ターゲット容量を 1 オンデマンドインスタンスとして指定します。デフォルト購入オプションは spot です。フリートは指定されたとおり 1 オンデマンドインスタンス を起動しますが、合計ターゲット容量を満たすために、さらに 1 つ以上のインスタンスを起動する必要があります。差額の購入オプションは、TotalTargetCapacity - OnDemandTargetCapacity = DefaultTargetCapacityType で計算されます。この結果、フリートはスポットインスタンスを起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "OnDemandTargetCapacity": 1,
        "DefaultTargetCapacityType": "spot"
    }
}
```

例 4: **lowest-price** 配分戦略を使用して スポットインスタンス を起動する

スポットインスタンスの配分戦略を指定しない場合、デフォルト配分戦略である lowest-price が使用されます。次の例では、lowest-price の配分戦略を使用します。起動テンプレートを

オーバーライドする 3 つの起動条件は、インスタンスタイプが異なりますが、加重容量とサブネットは同じです。合計ターゲット容量は 2 インスタンスで、デフォルト購入オプションは spot です。EC2 フリートは、最低価格の起動条件のインスタンスタイプを使用して 2 つのスポットインスタンスを起動します。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }
  "Overrides": [
        {
          "InstanceType": "c4.large",
          "WeightedCapacity": 1,
          "SubnetId": "subnet-a4f6c5d3"
        },
        {
          "InstanceType": "c3.large",
          "WeightedCapacity": 1,
          "SubnetId": "subnet-a4f6c5d3"
        },
        {
          "InstanceType": "c5.large",
          "WeightedCapacity": 1,
          "SubnetId": "subnet-a4f6c5d3"
        }
      ]

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 2,
        "DefaultTargetCapacityType": "spot"
    }
}
```

例 5: 複数のキャパシティー予約を使用して オンデマンドインスタンス を起動する

キャパシティ予約の使用戦略を use-capacity-reservations-first に設定することで、オンデマンドインスタンスの起動時に最初にオンデマンドキャパシティ予約を使用するようにフリートを

設定できます。この例では、目標容量を満たすために必要以上のキャパシティ予約がある場合に、フリートが使用するキャパシティ予約を選択する方法を示します。

この例では、フリート設定は次のようになります。

- ターゲット容量:12 オンデマンドインスタンス
- 未使用のキャパシティー予約の合計:15 (フリートの目標容量である 12 オンデマンドインスタンスを超えています)
- キャパシティ予約プールの数:3 (`m5.large`、`m4.xlarge`、および`m4.2xlarge`)
- プールあたりのキャパシティ予約数:5
- オンデマンド割り当て戦略：`lowest-price`(複数のインスタンスプールに未使用のキャパシティ予約が複数ある場合、フリートはオンデマンド割り当て戦略に基づいてオンデマンドインスタンスを起動するプールを決定します)。

  また、`lowest-price`割り当て戦略の代わりに`prioritized`割り当て戦略を使用することもできます。

キャパシティ予約

アカウントには、3 つの異なるプールに以下の 15 個の未使用のキャパシティ予約があります。各プールのキャパシティ予約の数は `AvailableInstanceCount` で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
```

```
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount":5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

フリート設定

以下のフリート設定は、この例に関連する設定のみを示しています。合計ターゲット容量は 12 で、デフォルトのターゲット容量タイプは on-demand です。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

この例では、オンデマンドインスタンスの料金は以下のようになります。

- m5.large – 1 時間あたり 0.096 USD
- m4.xlarge – 1 時間あたり 0.20 USD
- m4.2xlarge – 1 時間あたり 0.40 USD

> **Note**
>
> フリートタイプはタイプinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-abc1234567example",
                "Version": "1"
            }
            "Overrides": [
                {
                  "InstanceType": "m5.large",
```

```
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                },
                {
                  "InstanceType": "m4.xlarge",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                },
                {
                  "InstanceType": "m4.2xlarge",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                }
              ]

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 12,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price"
        "CapacityReservationOptions": {
            "UsageStrategy": "use-capacity-reservations-first"
        }
    },
    "Type": "instant",
}
```

上記の設定を使用して instant フリートを作成すると、目標容量を満たすために以下の 12 個のインスタンスが起動されます。

- us-east-1aの中のus-east-1a – m5.largeにある 5 つのm5.largeオンデマンドインスタンスが最低価格です。そしてそこに 5 つの利用可能な未使用のm5.largeキャパシティー予約があります。
- 5 つのm4.xlargeオンデマンドインスタンス (us-east-1a) – m4.xlarge (us-east-1a) は次に低い料金であり、利用可能な未使用m4.xlargeキャパシティーの予約が 5 つあります。
- 2 つのm4.2xlargeオンデマンドインスタンス (us-east-1a) – m4.2xlarge (us-east-1a) は 3 番目に低い料金であり、利用可能な未使用m4.2xlargeキャパシティーの予約は 5 つあります。そのうちの 2 つのみが目標容量を満たすために必要です。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、m5.largeおよびm4.xlargeのすべてのキャパシティーの予約が使用され、m4.2xlargeの 3 つのキャパシティーの予約が未使用のままであることを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "AvailableInstanceCount": 3
}
```

例 6: 合計ターゲット容量が未使用キャパシティーの予約の数を超えたときに、キャパシティーの予約を使用してオンデマンドインスタンスを起動する

キャパシティ予約の使用戦略を use-capacity-reservations-first に設定することで、オンデマンドインスタンスの起動時に最初にオンデマンドキャパシティ予約を使用するようにフリートを設定できます。この例では、総ターゲット容量が使用可能な未使用のキャパシティ予約数を超えた場合に、オンデマンドインスタンスを起動するインスタンスプールをフリートがどのように選択するかを示します。

この例では、フリート設定は次のようになります。

- ターゲット容量:16 オンデマンドインスタンス
- 未使用キャパシティー予約の合計:15 (フリートのターゲット容量である 16 オンデマンドインスタンスを下回っています)
- キャパシティ予約プールの数:3 (m5.large、m4.xlarge、およびm4.2xlarge)
- プールあたりのキャパシティ予約数:5

- オンデマンド割り当て戦略：lowest-price(未使用キャパシティーの予約の数が目標オンデマンド容量より少ない場合、フリートは、オンデマンド配分戦略に基づいて、残りのオンデマンド容量を起動するプールを決定します)。

  また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

キャパシティ予約

アカウントには、3 つの異なるプールに以下の 15 個の未使用のキャパシティ予約があります。各プールのキャパシティ予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "m5.large",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount":5,
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

フリート設定

以下のフリート設定は、この例に関連する設定のみを示しています。合計ターゲット容量は 16 で、デフォルトのターゲット容量タイプは on-demand です。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

この例では、オンデマンドインスタンスの料金は以下のようになります。

- m5.large – 0.096 USD/時間
- m4.xlarge – 0.20 USD/時間
- m4.2xlarge – 0.40 USD/時間

> **Note**
>
> フリートタイプはinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateId": "lt-0e8c754449b27161c",
                "Version": "1"
            }
            "Overrides": [
                {
                  "InstanceType": "m5.large",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                },
                {
                  "InstanceType": "m4.xlarge",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
                },
                {
                  "InstanceType": "m4.2xlarge",
                  "AvailabilityZone": "us-east-1a",
                  "WeightedCapacity": 1
```

```
                }
            ]

        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 16,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price"
        "CapacityReservationOptions": {
            "UsageStrategy": "use-capacity-reservations-first"
        }
    },
    "Type": "instant",
}
```

上記の設定を使用して instant フリートを作成すると、目標容量を満たすために以下の 16 個のインスタンスが起動されます。

- 6 つのm5.largeオンデマンドインスタンス (us-east-1aのus-east-1a – m5.large) が最低価格です。そして 5 つの利用可能な未使用m5.largeキャパシティーの予約があります。5 つのオンデマンドインスタンスを起動するために、キャパシティ予約が最初に使用されます。残りのm4.xlargeおよびm4.2xlargeキャパシティーの予約を使用してターゲット容量を満たすために、追加のオンデマンドインスタンスは、オンデマンド配分戦略 (この例ではlowest-price) に従って起動します。
- 5 つのm4.xlargeオンデマンドインスタンス (us-east-1a の us-east-1a – m4.xlarge) が次に低い料金であり、5 つの利用可能な未使用m4.xlargeキャパシティーの予約があります。
- 5 つのm4.2xlargeオンデマンドインスタンス (us-east-1a の us-east-1a – m4.2xlarge) が 3 番目に低い料金であり、5 つの利用可能な未使用m4.2xlargeキャパシティーの予約があります。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、すべてのプール内のすべてのキャパシティーの予約 が使用されたことを示しています。

```
{
    "CapacityReservationId": "cr-111",
```

```
    "InstanceType": "m5.large",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "m4.xlarge",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-333",
    "InstanceType": "m4.2xlarge",
    "AvailableInstanceCount": 0
}
```

例 7: ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動する

キャパシティーの予約の使用戦略をuse-capacity-reservations-firstに設定することで、オンデマンドインスタンスの起動時に最初にtargetedオンデマンドキャパシティー予約を使用するようにフリートを設定できます。この例では、オンデマンドインスタンスをtargetedキャパシティ予約で起動する方法を示します。キャパシティ予約の属性は、アベイラビリティーゾーン (us-east-1aおよびus-east-1b) を除いて同じになります。また、総ターゲット容量が使用可能な未使用のキャパシティ予約数を超えた場合に、オンデマンドインスタンスを起動するインスタンスプールをフリートがどのように選択するかについても説明します。

この例では、フリート設定は次のようになります。

- ターゲット容量:10 オンデマンドインスタンス
- 未使用の合計targetedキャパシティー予約:6 (フリートの目標オンデマンド容量である 10 オンデマンドインスタンスを下回っています)
- キャパシティ予約のプールの数:2 (us-east-1aおよびus-east-1b)
- プールあたりのキャパシティ予約数:3
- オンデマンド割り当て戦略：lowest-price(未使用キャパシティーの予約の数が目標オンデマンド容量より少ない場合、フリートは、オンデマンド配分戦略に基づいて、残りのオンデマンド容量を起動するプールを決定します)。

また、lowest-price割り当て戦略の代わりにprioritized割り当て戦略を使用することもできます。

この例を実行するために必要な手順のチュートリアルについては、チュートリアル:ターゲットのキャパシティー予約を使用してオンデマンドインスタンスを起動するを参照してください。

キャパシティ予約

アカウントには、2 つの異なるプールに以下の 6 個の未使用キャパシティーの予約があります。この例では、プールはアベイラビリティーゾーンによって異なります。各プールのキャパシティ予約の数は AvailableInstanceCount で示されます。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "c5.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1a",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.xlarge",
    "InstancePlatform": "Linux/UNIX",
    "AvailabilityZone": "us-east-1b",
    "AvailableInstanceCount": 3,
    "InstanceMatchCriteria": "open",
    "State": "active"
}
```

フリート設定

以下のフリート設定は、この例に関連する設定のみを示しています。合計ターゲット容量は 10 で、デフォルトのターゲット容量タイプはon-demandです。オンデマンド配分戦略はlowest-priceです。キャパシティ予約の使用戦略はuse-capacity-reservations-firstです。

この例では、us-east-1でのc5.xlargeのオンデマンドインスタンスの料金は時間当たり 0.17 USD になります。

> **Note**
>
> フリートタイプはinstantである必要があります。他のフリートタイプはuse-capacity-reservations-firstをサポートしていません。

```
{
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
            "Overrides": [
               {
                   "InstanceType": "c5.xlarge",
                   "AvailabilityZone": "us-east-1a"
               },
               {
                    "InstanceType": "c5.xlarge",
                    "AvailabilityZone": "us-east-1b"
               }
            ]
        }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 10,
        "DefaultTargetCapacityType": "on-demand"
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price",
        "CapacityReservationOptions": {
             "UsageStrategy": "use-capacity-reservations-first"
         }
    },
    "Type": "instant"
}
```

上記の設定を使用してinstantフリートを作成すると、目標容量を満たすために以下の 10 個のインスタンスが起動されます。

- 次のように、6 つのオンデマンドインスタンスを起動するために、キャパシティ予約が最初に使用されます。
  - 3 つのオンデマンドインスタンスが、us-east-1aでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
  - 3 つのオンデマンドインスタンスが、us-east-1bでの 3 つのc5.xlargetargetedキャパシティー予約で起動します。
- ターゲット容量を満たすために、4 つの追加のオンデマンドインスタンスは、オンデマンド配分戦略 (この例ではlowest-price) に従って通常のオンデマンド容量で起動します。ただし、プールの価格は同じであるため (価格はアベイラビリティーゾーンごとではなく、リージョンごとであるため)、フリートは残りの 4 つのオンデマンドインスタンスをいずれかのプールで起動します。

フリートの起動後、describe-capacity-reservations を実行して、未使用のキャパシティ予約の数を確認できます。この例では、以下のレスポンスが表示されます。これは、すべてのプール内のすべてのキャパシティーの予約 が使用されたことを示しています。

```
{
    "CapacityReservationId": "cr-111",
    "InstanceType": "c5.xlarge",
    "AvailableInstanceCount": 0
}

{
    "CapacityReservationId": "cr-222",
    "InstanceType": "c5.xlarge",
    "AvailableInstanceCount": 0
}
```

例 8: 容量の再調整を設定して代替スポットインスタンスを起動する

次の例では、Amazon EC2 がフリートのスポットインスタンスに再調整に関する推奨を送信したときに、代替スポットインスタンスを起動するように EC2 フリートを設定します。スポットインスタンスの自動代替を設定するには、ReplacementStrategy で、launch-before-terminate を指定します。置換用の新しいスポットインスタンスが起動してから、古いスポットインスタンスが自動的に削除されるまでの時間を設定するには、termination-delay に値を秒単位で指定します。詳細については、「設定オプション」を参照してください。

> **Note**
>
> launch-before-terminate を使用するのは、インスタンスのシャットダウン処理にかかる時間を予測できる場合に限り、これらの処理が完了した後に古いインスタンスが終了するようにすることをお勧めします。実行中は、すべてのインスタンスに対して課金されます。

容量の再調整戦略の有効性は、EC2 フリートリクエストで指定されたスポットキャパシティプールの数に左右されます。インスタンスタイプとアベイラビリティーゾーンの多様なセットを使ってフリートを設定し、AllocationStrategy では capacity-optimized を指定することをお勧めします。EC2 フリート の容量の再調整を行う際に考慮すべき事項の詳細については、「」を参照してください。容量の再調整

```
{
    "ExcessCapacityTerminationPolicy": "termination",
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "LaunchTemplate",
                "Version": "1"
            },
                 "Overrides": [
                       {
                           "InstanceType": "c3.large",
                           "WeightedCapacity": 1,
                            "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       },
                       {
                           "InstanceType": "c4.large",
                           "WeightedCapacity": 1,
                            "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
                       },
                       {
                           "InstanceType": "c5.large",
                           "WeightedCapacity": 1,
                            "Placement": {
                               "AvailabilityZone": "us-east-1a"
                           }
```

```
                        }
                ]
            }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 5,
        "DefaultTargetCapacityType": "spot"
    },
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        "MaintenanceStrategies": {
            "CapacityRebalance": {
                "ReplacementStrategy": "launch-before-terminate",
                "TerminationDelay": "720"
            }
        }
    }
}
```

例 9: 容量最適化フリートでスポットインスタンスを起動する

次の例は、容量を最適化するスポット配分戦略で、EC2 フリートを設定する方法を示しています。容量を最適化するには、AllocationStrategy を capacity-optimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量はスポットインスタンス 50 個です。EC2 フリートは、起動中のインスタンス数の最適な容量のスポットキャパシティープールに 50 個のスポットインスタンスを起動しようとします。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized",
        },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                       {
                           "InstanceType": "r4.2xlarge",
                           "Placement": {
```

```
                            "AvailabilityZone": "us-west-2a"
                        },
                    },
                     {
                        "InstanceType": "m4.2xlarge",
                        "Placement": {
                            "AvailabilityZone": "us-west-2b"
                        },
                     },
                     {
                        "InstanceType": "c5.2xlarge",
                        "Placement": {
                            "AvailabilityZone": "us-west-2b"
                        }
                     }
                ]
            }
    ],
    "TargetCapacitySpecification": {
            "TotalTargetCapacity": 50,
            "DefaultTargetCapacityType": "spot"

    }
}
```

例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する

次の例は、ベスト工フォートベースで優先順位を使用しながら、容量を最適化するスポット配分戦略を使用して、EC2 フリートを設定する方法を示しています。

capacity-optimized-prioritized 配分戦略を使用する場合は、Priority パラメータを使用して、スポットキャパシティプールの優先順位を指定します。数値が小さいほど優先順位が高くなります。また、優先度が同じならば、複数のスポットキャパシティープールに同じ優先順位を設定することもできます。プールに優先順位を設定しない場合、そのプールは優先順位が最も低いとみなされます。

スポットキャパシティプールに優先順位を付けるには、AllocationStrategy を capacity-optimized-prioritized に設定する必要があります。EC2 フリートは最初に容量を最適化しますが、インスタンスタイプの優先順位をベスト工フォートベースで決定します (例えば、優先順位を尊重しても、EC2 フリートの最適な容量をプロビジョニングする能力に大きな影響を与えない場合など) 。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティープールが指定されています。各プールには優先順位が設定されています。数値が小さいほど優先順位が高くなります。ターゲット容量は 50 個のスポットインスタンスです。EC2 フリートは、ベスト工フォートベースで優先順位が最も高いスポットキャパシティープールに 50 個のスポットインスタンスを起動しようとしますが、最初に容量を最適化します。

```
{
    "SpotOptions": {
        "AllocationStrategy": "capacity-optimized-prioritized"
        },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                        {
                            "InstanceType": "r4.2xlarge",
                            "Priority": 1,
                            "Placement": {
                                "AvailabilityZone": "us-west-2a"
                            },
                       },
                        {
                            "InstanceType": "m4.2xlarge",
                            "Priority": 2,
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            },
                        },
                        {
                            "InstanceType": "c5.2xlarge",
                            "Priority": 3,
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            }
                        }
                    ]
             }
    ],
    "TargetCapacitySpecification": {
            "TotalTargetCapacity": 50,
```

```
            "DefaultTargetCapacityType": "spot"
}
```

例 11: price-capacity-optimized フリートでスポットインスタンスを起動する

次の例は、容量と価格の両方を最適化するスポット配分戦略で、EC2 フリートを設定する方法を示しています。価格を考慮しながら容量を最適化するには、スポット `AllocationStrategy` を `price-capacity-optimized` に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量は 50 個のスポットインスタンスです。EC2 フリートは、起動するインスタンス数に最適な容量を持つスポットキャパシティプールに 50 個のスポットインスタンスを起動し、同時に価格が最も低いプールを選択することを試みます。

```
{
    "SpotOptions": {
        "AllocationStrategy": "price-capacity-optimized",
        "MinTargetCapacity": 2,
        "SingleInstanceType": true
    },
    "OnDemandOptions": {
        "AllocationStrategy": "lowest-price"
    },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                       {
                           "InstanceType": "r4.2xlarge",
                           "Placement": {
                               "AvailabilityZone": "us-west-2a"
                           },
                      },
                       {
                           "InstanceType": "m4.2xlarge",
                           "Placement": {
                               "AvailabilityZone": "us-west-2b"
                           },
                       },
```

```
                        {
                            "InstanceType": "c5.2xlarge",
                            "Placement": {
                                "AvailabilityZone": "us-west-2b"
                            }
                        }
                ]
            }
    ],
    "TargetCapacitySpecification": {
        "TotalTargetCapacity": 50,
        "OnDemandTargetCapacity":0,
        "SpotTargetCapacity":50,
        "DefaultTargetCapacityType": "spot"
    },
    "Type": "instant"
}
```

例 12: 属性ベースのインスタンスタイプの選択を設定する

次の例は、インスタンスタイプの識別に属性ベースのインスタンスタイプ選択を使用するように EC2 フリート を設定する方法を示しています。必要なインスタンス属性を指定するには、`InstanceRequirements` 構造に属性を指定します。

次の例では、2 つのインスタンス属性が指定されています。

- `VCpuCount` — 最低 2 つの vCPUs が指定されています。最大値は指定されていないため、上限はありません。
- `MemoryMiB` — 4 MiB 以上のメモリが指定されています。最大値は指定されていないため、上限はありません。

2 つ以上の vCPUs と 4 MiB 以上のメモリを持つすべてのインスタンスタイプが識別されます。ただし、EC2 フリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

指定できるすべての属性のリストと説明については、「Amazon EC2 API リファレンス」の「インスタンス要件」を参照してください。

```
{
 "SpotOptions": {
```

```
   "AllocationStrategy": "price-capacity-optimized"
  },
  "LaunchTemplateConfigs": [{
   "LaunchTemplateSpecification": {
    "LaunchTemplateName": "my-launch-template",
    "Version": "1"
   },
   "Overrides": [{
    "InstanceRequirements": {
     "VCpuCount": {
      "Min": 2
     },
     "MemoryMiB": {
      "Min": 4
     }
    }
   }]
  }],
  "TargetCapacitySpecification": {
   "TotalTargetCapacity": 20,
   "DefaultTargetCapacityType": "spot"
  },
  "Type": "instant"
}
```

スポットフリートの設定例

以下の例で示しているのは、スポットフリートリクエストを作成するための request-spot-fleet コマンドで使用できる起動設定です。https://docs.aws.amazon.com/cli/latest/reference/ec2/request-spot-fleet.html詳細については、「スポットフリートリクエストを作成します。」を参照してください。

> **Note**
>
> スポットフリートでは、ネットワークインターフェイス ID を起動テンプレートか起動仕様に指定できません。起動テンプレートまたは起動仕様から NetworkInterfaceID パラメータを必ず省略してください。

例

- 例 1: リージョンの最低価格のアベイラビリティーゾーンあるいはサブネットを使用してスポットインスタンスを起動する

例 1: リージョンの最低価格のアベイラビリティーゾーンあるいはサブネットを使用してスポットインスタンスを起動する

以下の例では、アベイラビリティーゾーンまたはサブネットを使用しない 1 つの起動仕様を指定しています。スポットフリートはデフォルトのサブネットを持つ最低価格のアベイラビリティーゾーンでインスタンスを起動します。お支払いいただく料金はオンデマンド価格を上回りません。

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
```

```
        }
    ]
}
```

例 2: 指定したリスト内で最低価格のアベイラビリティーゾーンまたはサブネットを使用してスポットインスタンスを起動する

以下の例では、アベイラビリティーゾーン/サブネットは異なるがインスタンスタイプおよび AMI が同じである、2 つの起動仕様を指定しています。

アベイラビリティーゾーン

スポットフリートは、指定した最低価格のアベイラビリティーゾーンのデフォルトサブネットでインスタンスを起動します。

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "Placement": {
              "AvailabilityZone": "us-west-2a, us-west-2b"
          },
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

Subnets

デフォルトのサブネットまたはデフォルト以外のサブネットを指定できますが、デフォルト以外のサブネットは、デフォルトの VPC またはデフォルト以外の VPC 内から選択できます。スポットサー

ビスは、最低価格のアベイラビリティーゾーンにあるいずれかのサブネットでインスタンスを起動します。

スポットフリートリクエストで、同じアベイラビリティーゾーンから異なるサブネットを指定することはできません。

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "m3.medium",
          "SubnetId": "subnet-a61dafcf, subnet-65ea5f08",
          "IamInstanceProfile": {
              "Arn": "arn:aws:iam::123456789012:instance-profile/my-iam-role"
          }
      }
  ]
}
```

インスタンスがデフォルトの VPC で起動される場合は、デフォルトでパブリック IPv4 アドレスが割り当てられます。インスタンスがデフォルト以外の VPC で起動される場合は、デフォルトでパブリック IPv4 アドレスは割り当てられません。起動仕様でネットワークインターフェイスを使用して、デフォルト以外の VPC で起動されるインスタンスにパブリック IPv4 アドレスを割り当てます。ネットワークインターフェイスの指定時、ネットワークインターフェイスを使用してサブネット ID とセキュリティグループ ID を含める必要があります。

```
  ...
      {
          "ImageId": "ami-1a2b3c4d",
          "KeyName": "my-key-pair",
          "InstanceType": "m3.medium",
          "NetworkInterfaces": [
              {
                  "DeviceIndex": 0,
```

```
                "SubnetId": "subnet-1a2b3c4d",
                "Groups": [ "sg-1a2b3c4d" ],
                "AssociatePublicIpAddress": true
            }
        ],
        "IamInstanceProfile": {
            "Arn": "arn:aws:iam::880185128111:instance-profile/my-iam-role"
        }
    }
...
```

例 3: 指定したリスト内で最低価格のインスタンスタイプを使用してスポットインスタンスを起動する

次の例では、同じ AMI と アベイラビリティーゾーンまたはサブネットで、複数の異なるインスタンスタイプを使用する 2 つの起動設定を指定します。スポットフリートは、指定された最低価格のインスタンスタイプを使用してインスタンスを起動します。

アベイラビリティーゾーン

```
{
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
          ],
          "InstanceType": "c5.4xlarge",
          "Placement": {
            "AvailabilityZone": "us-west-2b"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "SecurityGroups": [
              {
                  "GroupId": "sg-1a2b3c4d"
              }
```

```
            ],
            "InstanceType": "r3.8xlarge",
            "Placement": {
                "AvailabilityZone": "us-west-2b"
            }
        }
    ]
}
```

サブネット

```
{
    "TargetCapacity": 20,
    "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
    "LaunchSpecifications": [
        {
            "ImageId": "ami-1a2b3c4d",
            "SecurityGroups": [
                {
                    "GroupId": "sg-1a2b3c4d"
                }
            ],
            "InstanceType": "c5.4xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "SecurityGroups": [
                {
                    "GroupId": "sg-1a2b3c4d"
                }
            ],
            "InstanceType": "r3.8xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        }
    ]
}
```

例 4。リクエストの料金を上書きする

オンデマンド価格であるデフォルトの上限料金を使用することをお勧めします。必要に応じて、フリートリクエストの上限料金と個々の起動条件の上限料金を指定することができます。

以下の例は、フリートリクエストの上限料金と、3 つの起動条件のうちの 2 つの上限料金を指定しています。フリートリクエストの上限料金は、上限料金を指定しないすべての起動条件に適用されます。スポットフリートは、最低価格のインスタンスタイプを使用してインスタンスを起動します。

アベイラビリティーゾーン

```
{
  "SpotPrice": "1.00",
  "TargetCapacity": 30,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "SpotPrice": "0.10"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.4xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "SpotPrice": "0.20"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.8xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      }
    ]
}
```

サブネット

```
{
  "SpotPrice": "1.00",
  "TargetCapacity": 30,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
```

```
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.2xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "SpotPrice": "0.10"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.4xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "SpotPrice": "0.20"
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.8xlarge",
          "SubnetId": "subnet-1a2b3c4d"
      }
  ]
}
```

例 5: 分散配分戦略を使用して、スポットフリートを起動する

次の例では、diversified の配分戦略を使用します。これらの起動仕様では、インスタンスタイプは異なりますが、AMI およびアベイラビリティーゾーン/サブネットは同じです。スポットフリートは、各タイプのインスタンスが 10 個になるように、3 個の起動仕様全体に 30 個のインスタンスを分散します。詳細については、「スポットインスタンスの配分戦略」を参照してください。

アベイラビリティーゾーン

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c4.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      },
```

```
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "m3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      }
  ]
}
```

サブネット

```
{
    "SpotPrice": "0.70",
    "TargetCapacity": 30,
    "AllocationStrategy": "diversified",
    "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
    "LaunchSpecifications": [
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "c4.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "m3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        }
    ]
}
```

アベイラビリティーゾーンの 1 つで機能停止が発生した場合にスポットリクエストが EC2 のキャパシティーによって満たされる可能性を高めるためのベストプラクティスは、ゾーン間で多様化することです。このシナリオでは、使用可能な各アベイラビリティーゾーンを起動仕様に含めます。また、毎回同じサブネットを使用するのではなく、3 つの固有のサブネット (それぞれ異なるゾーンへのマッピング) を使用してください。

アベイラビリティーゾーン

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 30,
  "AllocationStrategy": "diversified",
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c4.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2a"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "m3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          }
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2c"
          }
      }
  ]
}
```

サブネット

```
{
    "SpotPrice": "0.70",
```

```
    "TargetCapacity": 30,
    "AllocationStrategy": "diversified",
    "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
    "LaunchSpecifications": [
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "c4.2xlarge",
            "SubnetId": "subnet-1a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "m3.2xlarge",
            "SubnetId": "subnet-2a2b3c4d"
        },
        {
            "ImageId": "ami-1a2b3c4d",
            "InstanceType": "r3.2xlarge",
            "SubnetId": "subnet-3a2b3c4d"
        }
    ]
}
```

例 6: インスタンスの分量指定を使用して、スポットフリートを起動する

次の例では、インスタンス分量指定を使っています。これは、料金が 1 インスタンス時間当たりではなく、1 ユニット時間当たりであることを意味します。それぞれの起動設定には、異なるインスタンスタイプおよび異なる分量がリストされます。スポットフリートはユニット時間の最低価格のインスタンスタイプを選択します。スポットフリートは、ターゲット容量をインスタンス分量で割ることで起動するスポットインスタンス数を計算します。結果が整数でない場合、スポットフリートはその数を次の整数に切り上げ、そのためフリートのサイズがターゲット容量以上になります。

`r3.2xlarge` のリクエストが成功すると、スポットはこれらのインスタンスのうち、4 つをプロビジョニングします。3.33 インスタンスまで 20 を 6 で割り、そして残りの 4 つのインスタンスを切り上げます。

`c3.xlarge` のリクエストが成功すると、スポットはこれらのインスタンスのうち、7 つをプロビジョニングします。6.66 インスタンスまで 20 を 3 で割り、そして残りの 7 つのインスタンスを切り上げます。

詳細については、「スポットフリートインスタンスの分量指定」を参照してください。

アベイラビリティーゾーン

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "WeightedCapacity": 6
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.xlarge",
          "Placement": {
              "AvailabilityZone": "us-west-2b"
          },
          "WeightedCapacity": 3
      }
    ]
}
```

サブネット

```
{
  "SpotPrice": "0.70",
  "TargetCapacity": 20,
  "IamFleetRole": "arn:aws:iam::123456789012:role/aws-ec2-spot-fleet-tagging-role",
  "LaunchSpecifications": [
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "r3.2xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "WeightedCapacity": 6
      },
      {
          "ImageId": "ami-1a2b3c4d",
          "InstanceType": "c3.xlarge",
          "SubnetId": "subnet-1a2b3c4d",
          "WeightedCapacity": 3
      }
```

```
  ]
}
```

例 7: オンデマンド容量でスポットフリートを起動する

インスタンス容量を常に確保するには、オンデマンド容量のリクエストをスポットフリートリクエストに含めることができます。オンデマンドリクエストは、容量がある限り、常に実行されます。ターゲット容量は、キャパシティーと可用性がある場合にスポットとして実行されます。

次の例では、希望するターゲット容量を 10 とし、そのうち 5 をオンデマンドキャパシティーとして指定する必要があります。スポットキャパシティーは指定しません。これは、ターゲット容量からオンデマンド容量を引いたバランスを意味します。Amazon EC2 は、利用可能な Amazon EC2 容量および可用性がある場合、オンデマンドとして 5 容量単位を、スポットとして 5 容量単位 (10-5=5) をスポットとして起動します。

詳細については、「スポットフリートでのオンデマンド」を参照してください。

```
{
  "IamFleetRole": "arn:aws:iam::781603563322:role/aws-ec2-spot-fleet-tagging-role",
  "AllocationStrategy": "lowestPrice",
  "TargetCapacity": 10,
  "SpotPrice": null,
  "ValidFrom": "2018-04-04T15:58:13Z",
  "ValidUntil": "2019-04-04T15:58:13Z",
  "TerminateInstancesWithExpiration": true,
  "LaunchSpecifications": [],
  "Type": "maintain",
  "OnDemandTargetCapacity": 5,
  "LaunchTemplateConfigs": [
    {
      "LaunchTemplateSpecification": {
        "LaunchTemplateId": "lt-0dbb04d4a6cca5ad1",
        "Version": "2"
      },
      "Overrides": [
        {
          "InstanceType": "t2.medium",
          "WeightedCapacity": 1,
          "SubnetId": "subnet-d0dc51fb"
        }
      ]
    }
```

```
  ]
}
```

例 8: 容量の再調整を設定して代替 スポットインスタンス を開始する

次の例では、Amazon EC2 がフリートのスポットインスタンスに再調整の推奨を発したときに、スポットフリートが代替スポットインスタンスを起動するように設定します。スポットインスタンスの自動代替を設定するには、ReplacementStrategy で、launch-before-terminate を指定します。新しい交換用スポットインスタンスが起動してから古いスポットインスタンスが自動削除されるまでの時間を設定するには、termination-delay に秒単位で値を指定します。詳細については、「設定オプション」を参照してください。

> **Note**
>
> launch-before-terminate は、インスタンスのシャットダウン手順が完了するまでの時間が予測できる場合にのみ使用することをお勧めします。これにより、古いインスタンスは、シャットダウン手順が完了した後にのみ終了されます。実行中は、すべてのインスタンスに対して課金されます。

容量の再調整戦略の有効性は、スポットフリートリクエストで指定されたスポットキャパシティプールの数に左右されます。インスタンスタイプとアベイラビリティーゾーンの多様なセットを使ってフリートを設定し、AllocationStrategy では capacityOptimized を指定することをお勧めします。スポットフリートの容量の再調整を行うときに考慮すべき事項の詳細については、「容量の再調整」を参照してください。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "capacityOptimized",
        "IamFleetRole": "arn:aws:iam::000000000000:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateName": "LaunchTemplate",
                    "Version": "1"
                },
                 "Overrides": [
                       {
                           "InstanceType": "c3.large",
```

```
                            "WeightedCapacity": 1,
                            "Placement": {
                                "AvailabilityZone": "us-east-1a"
                            }
                        },
                        {
                            "InstanceType": "c4.large",
                            "WeightedCapacity": 1,
                            "Placement": {
                                "AvailabilityZone": "us-east-1a"
                            }
                        },
                        {
                            "InstanceType": "c5.large",
                            "WeightedCapacity": 1,
                            "Placement": {
                                "AvailabilityZone": "us-east-1a"
                            }
                        }
                ]
          }
    ],
        "TargetCapacity": 5,
        "SpotMaintenanceStrategies": {
            "CapacityRebalance": {
                "ReplacementStrategy": "launch-before-terminate",
                "TerminationDelay": "720"
            }
        }
    }
}
```

例 9: 容量最適化フリートでスポットインスタンスを起動する

以下の例は、容量を最適化するスポット配分戦略で、スポットフリートを設定する方法を示しています。容量を最適化するには、AllocationStrategy を capacityOptimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量は 50 個のスポットインスタンスです。スポットインスタンスは、起動中のインスタンス数に最適な容量のスポットキャパシティプールに、50 個のスポットインスタンスを起動しようとします。

```
{
```

```
    "TargetCapacity": "50",
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "capacityOptimized",
    },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                       {
                           "InstanceType": "r4.2xlarge",
                           "AvailabilityZone": "us-west-2a"
                      },
                       {
                           "InstanceType": "m4.2xlarge",
                           "AvailabilityZone": "us-west-2b"
                       },
                       {
                           "InstanceType": "c5.2xlarge",
                           "AvailabilityZone": "us-west-2b"
                       }
                ]
           }
      ]
}
```

例 10: 優先順位のある容量最適化フリートでスポットインスタンスを起動する

次の例は、ベストエフォートベースで優先順位を使用しながら、容量を最適化するスポット配分戦略を使用して、スポットフリートを設定する方法を示しています。

capacityOptimizedPrioritized 配分戦略を使用する場合は、Priority パラメータを使用して、スポットキャパシティプールの優先順位を指定します。数値が小さいほど優先順位が高くなります。また、優先度が同じならば、複数のスポットキャパシティープールに同じ優先順位を設定することもできます。プールに優先順位を設定しない場合、そのプールは優先順位が最も低いとみなされます。

スポットキャパシティプールに優先順位を付けるには、AllocationStrategy を capacityOptimizedPrioritized に設定する必要があります。スポットフリートは最初に容量を最適化しますが、優先順位をベストエフォートベースで決定します (例えば、優先順位を尊重し

ても、スポットフリートの最適な容量をプロビジョニングする能力に大きな影響を与えない場合など)。これは、中断の可能性を最小限に抑える必要があり、特定のインスタンスタイプを優先することが重要なワークロードに適したオプションです。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティープールが指定されています。各プールには優先順位が設定されています。数値が小さいほど優先順位が高くなります。ターゲット容量は 50 個のスポットインスタンスです。スポットフリートは、ベストエフォートベースで優先順位が最も高いスポットキャパシティープールに 50 個のスポットインスタンスを起動しようとしますが、最初に容量を最適化します。

```
{
    "TargetCapacity": "50",
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "capacityOptimizedPrioritized"
    },
    "LaunchTemplateConfigs": [
        {
            "LaunchTemplateSpecification": {
                "LaunchTemplateName": "my-launch-template",
                "Version": "1"
            },
                 "Overrides": [
                        {
                            "InstanceType": "r4.2xlarge",
                            "Priority": 1,
                            "AvailabilityZone": "us-west-2a"
                        },
                        {
                            "InstanceType": "m4.2xlarge",
                            "Priority": 2,
                            "AvailabilityZone": "us-west-2b"
                        },
                        {
                            "InstanceType": "c5.2xlarge",
                            "Priority": 3,
                            "AvailabilityZone": "us-west-2b"
                        }
                    ]
             }
       ]
}
```

例 11: priceCapacityOptimized フリートでスポットインスタンスを起動する

次の例は、容量と最低価格の両方を最適化するスポット配分戦略を使用するスポットフリートを設定する方法を示しています。価格を考慮しながら容量を最適化するには、スポット AllocationStrategy を priceCapacityOptimized に設定する必要があります。

次の例では、3 つの起動仕様で 3 つのスポットキャパシティプールが指定されています。ターゲット容量は 50 個のスポットインスタンスです。スポットフリートは、起動するインスタンス数に最適な容量を持つスポットキャパシティプールに 50 個のスポットインスタンスを起動し、同時に価格が最も低いプールを選択することを試みます。

```
{
    "SpotFleetRequestConfig": {
        "AllocationStrategy": "priceCapacityOptimized",
        "OnDemandAllocationStrategy": "lowestPrice",
        "ExcessCapacityTerminationPolicy": "default",
        "IamFleetRole": "arn:aws:iam::111111111111:role/aws-ec2-spot-fleet-tagging-
role",
        "LaunchTemplateConfigs": [
            {
                "LaunchTemplateSpecification": {
                    "LaunchTemplateId": "lt-0123456789example",
                    "Version": "1"
                },
                "Overrides": [
                     {
                           "InstanceType": "r4.2xlarge",
                           "AvailabilityZone": "us-west-2a"
                     },
                     {
                           "InstanceType": "m4.2xlarge",
                           "AvailabilityZone": "us-west-2b"
                     },
                     {
                           "InstanceType": "c5.2xlarge",
                           "AvailabilityZone": "us-west-2b"
                     }
                ]
            }
        ],
        "TargetCapacity": 50,
        "Type": "request"
    }
```

```
}
```

例 12: 属性ベースのインスタンスタイプの選択を設定する

次の例は、インスタンスタイプの識別に属性ベースのインスタンスタイプ選択を使用するようにスポットフリートを設定する方法を示しています。必要なインスタンス属性を指定するには、InstanceRequirements 構造に属性を指定します。

次の例では、2 つのインスタンス属性が指定されています。

- VCpuCount — 最低 2 つの vCPUs が指定されています。最大値は指定されていないため、上限はありません。
- MemoryMiB — 4 MiB 以上のメモリが指定されています。最大値は指定されていないため、上限はありません。

2 つ以上の vCPUs と 4 MiB 以上のメモリを持つすべてのインスタンスタイプが識別されます。ただし、スポットフリートがフリートをプロビジョニングする場合、価格保護と配分戦略によって一部のインスタンスタイプが除外される場合があります。

指定できるすべての属性のリストと説明については、「Amazon EC2 API リファレンス」の「インスタンス要件」を参照してください。

```
{
 "AllocationStrategy": "priceCapacityOptimized",
 "TargetCapacity": 20,
 "Type": "request",
 "LaunchTemplateConfigs": [{
  "LaunchTemplateSpecification": {
   "LaunchTemplateName": "my-launch-template",
   "Version": "1"
  },
  "Overrides": [{
   "InstanceRequirements": {
    "VCpuCount": {
     "Min": 2
    },
    "MemoryMiB": {
     "Min": 4
    }
   }
  }]
```

```
 }]
}
```

フリートのクォータ

通常の Amazon EC2 のクォータは、EC2 フリートまたはスポットフリートが起動したインスタンスの、[Spot Instance limits] (スポットインスタンスの制限) や [volume limits] (ボリュームの制限) などに適用されます。

また、次のクォータも適用されます。

クォータの説明	クォータ
active、deleted_running 、およびcancelled_running 状態のリージョンあたりの EC2 フリートおよびスポットフリートの数	1,000 [1] [2] [3] [4]
スポットキャパシティープールの数 (インスタンスタイプとサブネットの一意の組み合わせ)	300 [1] [4]
起動仕様内のユーザーデータのサイズ	16 KB [2]
EC2 フリートまたはスポットフリートあたりのターゲットキャパシティ	10,000
リージョン内のすべての EC2 フリートおよびスポットフリートにおけるターゲットキャパシティ	100,000 [1]
EC2 フリートリクエストまたはスポットフリートリクエストは、リージョンにまたがることはできません。	
EC2 フリートリクエストまたはスポットフリートリクエストは、同じアベイラビリティーゾーンからの複数の異なるサブネットにまたがることはできません。	

[1] これらのクォータは、EC2 フリートとスポットフリートの両方に適用されます。

[2] これらはハードクォータです。これらのクォータの引き上げをリクエストできません。

[3] EC2 フリートを削除した後、またはスポットフリートリクエストをキャンセルした後、リクエストを削除またはキャンセルしたときにスポットインスタンスを終了すべきではないことを指定した場合、フリートリクエストは`deleted_running`(EC2 フリート) または`cancelled_running`(スポットフリート) 状態になり、インスタンスは中断または手動終了されるまで、引き続き実行されます。インスタンスを終了する場合、フリートリクエストは `deleted_terminating` (EC2 フリート) または `cancelled_terminating` (スポットフリート) 状態になるため、このクォータにはカウントされません。詳細については、EC2 フリート の削除およびスポットフリートリクエストをキャンセルします。を参照してください。

[4] このクォータはタイプ`request`または`maintain`のフリートにのみ適用されます。このクォータは`instant`フリートに適用されません。

ターゲットキャパシティのクォータ引き上げをリクエストします

ターゲット容量のデフォルトクォータを超える容量が必要な場合は、クォータの引き上げをリクエストできます。

ターゲットキャパシティのクォータ引き上げをリクエストするには

1. AWS Support中央の [Create case] (ケースの作成) フォームを開きます。
2. [Service Limit increase] (サービス制限の緩和) を選択します。
3. [Limit type] (制限タイプ) には、[EC2 Fleet] (EC2 フリート) を選択します。
4. [Region] (リージョン) には、クォータの増加をリクエストする AWS リージョンを選択します。
5. [Limit] (制限) には、どちらのクォータを増やしたいかに応じて、[Target Fleet Capacity per Fleet (in units)] (フリートごとのターゲットフリート容量 (ユニット))、または [Target Fleet Capacity per Region (in units)] (リージョンごとのターゲットフリート容量 (ユニット)) のいずれかを選択します。
6. [新しい制限値] (New limit value) の場合、任意の値を入力します。
7. 別のクォータの引き上げを要求するには、[Add another request] (別のリクエストを追加) を選択し、ステップ 4 ~ 6 を繰り返します。
8. [Use case description] (ユースケースの説明) には、クォータの引き上げをリクエストする理由を入力します。
9. [Contact options] (連絡先オプション)で、希望する連絡言語と連絡方法を指定します。

10. [送信] を選択します。

Amazon EC2 のモニタリング

モニタリングは、Amazon Elastic Compute Cloud (Amazon EC2) インスタンスおよび AWS ソリューションの信頼性、可用性、およびパフォーマンスを維持する上で重要な部分です。マルチポイント障害が発生した場合は、その障害をより簡単にデバッグできるように、AWS ソリューションのすべての部分からモニタリングデータを収集する必要があります。ただし、Amazon EC2 のモニタリングを開始する前に、次の内容を盛り込んだモニタリング計画を作成する必要があります。

- モニタリングの目的とは?
- モニタリングの対象となるリソースとは?
- どのくらいの頻度でこれらのリソースをモニタリングしますか?
- どのモニタリングツールを利用しますか?
- 誰がモニタリングタスクを実行しますか?
- 問題が発生したときに誰が通知を受け取りますか?

モニタリングの目的を定義し、モニタリングの計画を作成したら、次のステップとして、お客様の環境内で通常の Amazon EC2 パフォーマンスのベースラインを確立します。さまざまな時間帯に、さまざまな負荷条件で Amazon EC2 パフォーマンスを測定します。Amazon EC2 をモニタリングしながら、収集したモニタリングデータの履歴を保存します。現在の Amazon EC2 パフォーマンスをこの履歴データと比較して、通常のパフォーマンスパターンとパフォーマンス異常を識別することで、異常への対処方法を考案することが容易になります。例えば、EC2 インスタンスの CPU 使用率、ディスク I/O、およびネットワーク使用率をモニタリングすることができます。確立したベースラインからパフォーマンスが外れた場合は、インスタンスの再設定または最適化を行って CPU 使用率の抑制、ディスク I/O の改善、またはネットワークトラフィックの低減を行うことが必要な場合があります。

ベースラインを確立するには、少なくとも、次の項目をモニタリングする必要があります。

モニタリング対象の項目	Amazon EC2 のメトリクス	エージェント/CloudWatch Logs のモニタリング
CPU 使用率	CPUUtilization	
ネットワーク使用率	NetworkIn NetworkOut	

モニタリング対象の項目	Amazon EC2 のメトリクス	エージェント/CloudWatch Logs のモニタリング
ディスクパフォーマンス	DiskReadOps DiskWriteOps	
ディスクの読み書き	DiskReadBytes DiskWriteBytes	
メモリの使用率、ディスクスワップの使用率、ディスクスペースの使用状況、ページファイルの使用状況、ログ収集		[Linux および Windows Server インスタンス] CloudWatch エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスを収集する [Windows Server インスタンスでの以前の CloudWatch Logs エージェントからの移行] Windows Server インスタンスのログ収集を CloudWatch エージェントに移行する

自動モニタリングと手動モニタリング

AWS は、Amazon EC2 のモニタリングに使用できるさまざまなツールを提供します。これらのツールの中には、自動モニタリングを設定できるものもあれば、手操作を必要とするものもあります。

モニタリングツール

- 自動モニタリングツール
- 手動モニタリングツール

自動モニタリングツール

次に示す自動化されたモニタリングツールを使用すると、Amazon EC2 の監視が行われ、問題が検出されたときにレポートが返されます。

- システムステータスチェック - インスタンスを使用する際に必要な AWS システムをモニタリングして、正常に実行されていることを確認します。これらのチェックでは、修復には AWS の関与が必要なインスタンスの根本的な問題が検出されます。システムステータスチェックが失敗した場合、AWS によって問題が修正されるのを待つか、自分自身で (例えば、インスタンスを停止、再起動、終了、置換するなどによって) 問題を解決できます。システムステータスチェックの失敗の原因となる問題には、次のようなものがあります。
  - ネットワーク接続の喪失
  - システム電源の喪失
  - 物理ホストのソフトウェアの問題
  - ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

  詳細については、インスタンスのステータスチェックを参照してください。
- [インスタンスステータスのチェック] – 個々のインスタンスのソフトウェアとネットワークの設定をモニタリングします。これらのチェックでは、ユーザーが関与して修復する必要のある問題が検出されます。インスタンスステータスチェックが失敗した場合、通常はお客様ご自身で (インスタンスの再起動、オペレーティングシステムの修正など) 問題を修復する必要があります。インスタンスステータスチェックの失敗の原因となる問題には、次のようなものがあります。
  - 失敗したシステムステータスチェック
  - 誤って設定されたネットワークまたは起動設定
  - メモリの枯渇
  - 破損したファイルシステム
  - 互換性のないカーネル

  詳細については、インスタンスのステータスチェックを参照してください。
- [Amazon CloudWatch アラーム] – 指定された期間にわたって単一のメトリクスを監視し、複数の期間にわたり既定のしきい値に関連するメトリクス値に基づいて 1 つ以上のアクションを実行します。アクションは、Amazon Simple Notification Service (Amazon SNS) のトピックまたは Amazon EC2 Auto Scaling のポリシーに送信される通知です。アラームは、持続している状態変化に対してのみアクションを呼び出します。CloudWatch アラームは、メトリクスが特定の状態になっただけではアクションを呼び出しません。アクションを呼び出すには、状態が変化して、指定

した期間継続している必要があります。詳細については、CloudWatch を使用したインスタンスのモニタリングを参照してください。

- Amazon EventBridge - AWS サービスを自動化し、システムイベントに自動的に応答します。AWS サービスからのイベントはほぼリアルタイムに EventBridge に提供され、イベントが記述したルールと一致したときに実行する自動アクションを指定できます。詳細については、Amazon EventBridge とは を参照してください。
- Amazon CloudWatch Logs - Amazon EC2 インスタンス、AWS CloudTrail、またはその他のソースのログファイルの監視、保存、アクセスができます。詳細については、 Amazon CloudWatch Logs ユーザーガイドを参照してください。
- CloudWatch agent – EC2 インスタンスとオンプレミスサーバー上のホストとゲストの両方からログとシステムレベルのメトリクスを収集します。詳細については、Amazon CloudWatch ユーザーガイドのCloudWatch エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスとログを収集するを参照してください。

手動モニタリングツール

Amazon EC2 のモニタリングにおけるもう 1 つの重要な部分は、モニタリングスクリプト、ステータスチェック、および CloudWatch アラームで網羅されていない項目を手動でモニタリングすることです。Amazon EC2 および CloudWatch のコンソールダッシュボードには、Amazon EC2 環境の状態が一目でわかるビューが表示されます。

- Amazon EC2 ダッシュボードには次の内容が表示されます。
  - リージョンごとのサービス状態とスケジュールされたイベント
  - インスタンスの状態
  - ステータスチェック
  - アラームステータス
  - インスタンスメトリクスの詳細 (ナビゲーションペインで、[Instances] を選択し、インスタンスを選択して、[Monitoring] タブを選択します)
  - ボリュームメトリクスの詳細 (ナビゲーションペインの [Volumes] を選択し、ボリュームを選択して、[Monitoring] タブを選択します)
- Amazon CloudWatch ダッシュボードには、次の内容が表示されます。
  - 現在のアラームとステータス
  - アラームとリソースのグラフ
  - サービスのヘルスステータス

また、CloudWatch を使用して以下のことを行えます。

- Amazon EC2 モニタリングデータをグラフ化して、問題のトラブルシューティングを行い、傾向を確認する
- AWS リソースのすべてのメトリクスを検索して、参照する
- 問題があることを通知するアラームを作成/編集する
- アラームおよび AWS リソースが一目でわかる概要を表示する

モニタリングのベストプラクティス

次に示すモニタリングのベストプラクティスを使用すると、Amazon EC2 のモニタリングタスクが容易になります。

- モニタリングの優先順位を設定し、小さな問題が大きな問題に発展する前に阻止します。
- AWS ソリューションのすべての部分からモニタリングデータを収集するモニタリング計画を作成し、実施すると、マルチポイント障害が発生した場合に、その障害をより簡単にデバッグできます。モニタリング計画には、少なくとも、次の質問に対する回答を盛り込む必要があります。
  - モニタリングの目的とは?
  - モニタリングの対象となるリソースとは?
  - どのくらいの頻度でこれらのリソースをモニタリングしますか?
  - どのモニタリングツールを利用しますか?
  - 誰がモニタリングタスクを実行しますか?
  - 問題が発生したときに誰が通知を受け取りますか?
- モニタリングタスクは可能な限り自動化します。
- EC2 インスタンスでログファイルを確認します。

インスタンスのステータスのモニタリング

インスタンスのステータスをモニタリングして、インスタンスのステータスチェックや、インスタンスにスケジュールされたイベントを表示できます。

ステータスチェックでは、Amazon EC2 によって実行される自動化されたチェックからの情報が提供されます。これらの自動化されたチェックは、特定の問題がインスタンスに影響を与えているかど

うかを検出します。ステータスチェックの情報と、Amazon CloudWatch で提供されるデータによって、各インスタンスの詳細な動作状況を把握できます。

インスタンスに予定されている特定イベントのステータスも表示できます。イベントのステータスは、再起動やリタイアなど、インスタンスに対して予定されている今後のアクティビティに関する情報を提供します。また、各イベントの予定開始予定時刻および終了時刻も提供されています。

コンテンツ

インスタンスのステータスチェック

インスタンスのステータスのモニタリングでは、インスタンスによるアプリケーションの実行を妨げる可能性のある問題をAmazon EC2 が検出したかどうかをすばやく判断できます。Amazon EC2 は、稼働中のすべての EC2 インスタンスに対して自動チェックを実行して、ハードウェアおよびソフトウェアの問題を特定します。これらのステータスチェックの結果を表示して、具体的で検出可能な問題を識別できます。このイベントステータスデータは、各インスタンス (`pending`、`running`、`stopping`) の状態に関して Amazon EC2 が既に提供している情報と、Amazon CloudWatch が監視している使用状況メトリクス (CPU 使用率、ネットワークトラフィック、ディスクアクティビティ) を補足するものです。

ステータスチェックは 1 分ごとに実行され、それぞれ成功または失敗のステータスが返ります。すべてのチェックが成功すると、インスタンス全体のステータスが OK になります。1つ以上のチェックが失敗すると、全体のステータスが impaired になります。ステータスチェックは Amazon EC2 に組み込まれています。そのため、無効にしたり、削除したりすることはできません。

ステータスチェックに失敗すると、ステータスチェックの対応する CloudWatch メトリクスは増加します。詳細については、ステータスチェックメトリクスを参照してください。このようなメトリクスを使用して、ステータスチェックの結果に基づいてトリガーされる CloudWatch アラームを作成することができます。例えば、特定のインスタンスでステータスチェックが失敗したときに警告するアラームを作成できます。詳細については、ステータスチェックアラームの作成と編集を参照してください。

また、Amazon EC2 インスタンスをモニタリングし、基になる問題によりインスタンスが正常に機能しなくなった場合に、自動的にインスタンスを復旧する Amazon CloudWatch アラームを作成できます。詳細については、インスタンスの復旧を参照してください。

コンテンツ

ステータスチェックのタイプ

ステータスチェックには 3 種類あります。

- システムステータスのチェック
- インスタンスステータスのチェック
- アタッチ済みの EBS ステータスチェック

システムステータスのチェック

システムステータスチェックは、インスタンスが実行されている AWS システムをモニタリングします。これらのチェックでは、修復には AWS の関与が必要なインスタンスの基盤の問題が検出されます。システムステータスチェックが失敗した場合、AWS が問題を解決するのを待つか、自分で解決できるかを選択できます。Amazon EBS でバックアップされたインスタンスの場合は、インスタンスを自分で停止および起動することができます。通常、インスタンスは新しいホストに移行されます。Linux インスタンスストアによってサポートされているインスタンスの場合、インスタンスを終了して置き換えることができます。Windows インスタンスの場合、ルートボリュームは Amazon EBS ボリュームであることが必要です。インスタンスストアはルートボリュームではサポートされません。インスタンスストアボリュームは短期のものであり、インスタンスが停止するとすべてのデータが失われることに注意してください。

システムステータスチェックの失敗の原因となる問題の例を次に示します。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのソフトウェアの問題
- ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

システムステータスチェックが失敗した場合、StatusCheckFailed_System メトリクスをインクリメントします。

ベアメタルインスタンス

ベアメタルインスタンス上のオペレーティングシステムから再起動を実行すると、システムステータスチェックが一時的に失敗ステータスを返すことがあります。インスタンスが使用可能になると、システムステータスチェックからは成功ステータスが返されます。

インスタンスステータスのチェック

[インスタンスステータスのチェック] 個々のインスタンスのソフトウェアとネットワークの設定をモニタリングします。Amazon EC2 は、ネットワークインターフェイス (NIC) にアドレス解決プロトコル (ARP) リクエストを送信することでインスタンスのヘルスをチェックします。これらのチェックでは、ユーザーが関与して修復する必要のある問題が検出されます。インスタンスステータスチェックが失敗した場合は通常、自分自身で (例えば、インスタンスを再起動する、インスタンス設定を変更するなどによって) 問題に対処する必要があります。

インスタンスステータスチェックの失敗の原因となる問題の例を次に示します。

- 失敗したシステムステータスチェック
- 正しくないネットワークまたは起動設定
- メモリの枯渇
- 破損したファイルシステム
- 互換性のないカーネル

インスタンスのステータスチェックが失敗した場合、StatusCheckFailed_Instance メトリクスをインクリメントします。

ベアメタルインスタンス

ベアメタルインスタンス上のオペレーティングシステムから再起動を実行すると、インスタンスのステータスチェックが一時的に失敗ステータスを返すことがあります。インスタンスが使用可能になると、インスタンスステータスチェックからは成功ステータスが返されます。

アタッチ済みの EBS ステータスチェック

アタッチ済みの EBS ステータスチェックは、インスタンスにアタッチされている Amazon EBS ボリュームが到達可能かどうか、および I/O 操作を完了できるかどうかをモニタリングします。StatusCheckFailed_AttachedEBS メトリクスは、インスタンスにアタッチされている 1 つ以上の EBS ボリュームが I/O 操作を完了できない場合に障害が発生することを示すバイナリ値です。これらのステータスチェックは、コンピューティングまたは Amazon EBS インフラストラクチャの根本的な問題を検出します。アタッチ済みの EBS ステータスチェックメトリクスが失敗した

場合は、AWS を待って問題を解決するか、影響を受けたボリュームの置き換えやインスタンスの停止および再起動などのアクションを取ることができます。

アタッチ済みの EBS ステータスチェックが失敗する原因となる問題の例を次に示します。

- EBS ボリュームの基盤となるストレージサブシステムのハードウェアまたはソフトウェアの問題
- EBS ボリュームの到達可能性に影響する、物理ホスト上のハードウェアの問題
- インスタンスと EBS ボリューム間の接続に関する問題

`StatusCheckFailed_AttachedEBS` メトリクスを使うことで、ワークロードの耐障害性を向上できます。このメトリクスを使用して、ステータスチェックの結果に基づいてトリガーされる Amazon CloudWatch アラームを作成することができます。例えば、長期にわたる影響を検出した場合は、セカンダリインスタンスまたはアベイラビリティーゾーンにフェイルオーバーできます。または、EBS CloudWatch メトリクスを使用してアタッチされた各ボリュームの I/O パフォーマンスをモニタリングし、障害のあるボリュームを検出して置き換えることもできます。ワークロードがインスタンスにアタッチされたどの EBS ボリュームに対しても I/O を提供していない上に、アタッチ済みの EBS ステータスチェックで障害が判明した場合は、インスタンスを停止して起動することで、EBS ボリュームの到達可能性に影響を与えている物理ホストの問題に対処できます。詳細については、「Amazon EBS の Amazon CloudWatch メトリクス」を参照してください。

> **Note**
>
> - アタッチ済みの EBS ステータスチェックメトリクスは、Nitro インスタンスでのみ使用できます。
> - `StatusCheckFailed_AttachedEBS` メトリックスに基づいて CloudWatch アラームを作成することにより、アタッチ済みの EBS ステータスチェックメトリクスをモニタリングできます。describe-instance-status (AWS CLI) コマンドを使用しても、このステータスチェックは表示できません。

ステータスチェックの操作

ステータスチェックは、AWS CLI などのコンソールおよびコマンドラインツールを使用して実行できます。

トピック

- ステータスチェックの表示

- ステータスチェックアラームの作成と編集

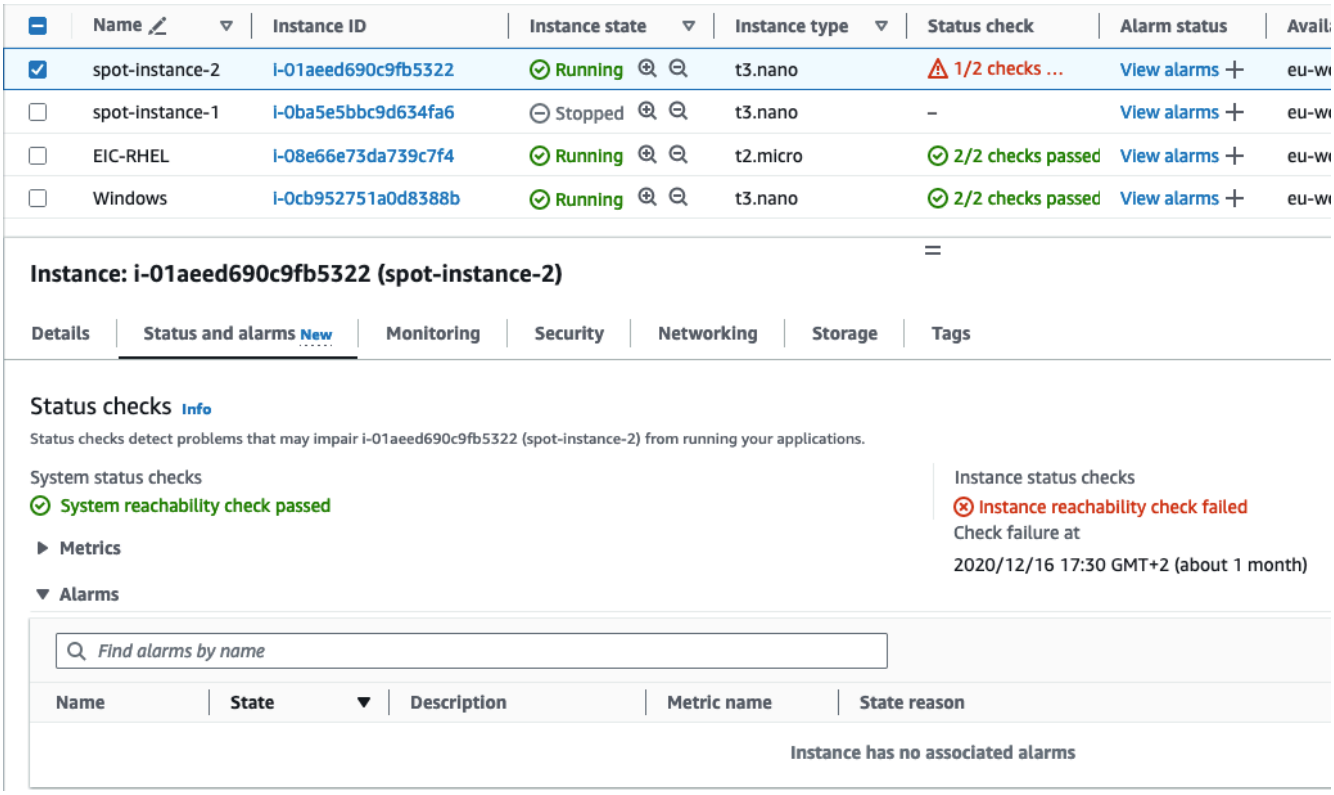
ステータスチェックの表示

ステータスチェックを表示するには、以下のいずれかの方法を使用します。

Console

ステータスチェックを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. [インスタンス] ページで、[Status check (ステータスチェック)] 列には、各インスタンスの動作状況が表示されます。
4. 特定のインスタンスのステータスを表示するには、インスタンスを選択して、[ステータスとアラーム] タブを選択します。



インスタンスに失敗したステータスチェックがある場合、通常は、自分自身で (例えば、インスタンスを再起動する、インスタンス設定を変更するなどによって) 問題に対処する必要があります。ご自分でシステムまたはインスタンスのステータスチェック失敗のトラブ

ルシューティングを行う場合は、ステータスチェックに失敗したインスタンスのトラブルシューティングを参照してください。

5. ステータスチェックで CloudWatch メトリクスを確認するには、[ステータスとアラーム] タブで [メトリクス] を展開し、以下のメトリクスのグラフを表示します。

   - [システムのステータスチェックの失敗]
   - [インスタンスのステータスチェックの失敗]

   詳細については、「the section called “ステータスチェックメトリクス”」を参照してください。

Command line

describe-instance-status (AWS CLI) コマンドを使用すると、実行中のインスタンスのステータスチェックを表示できます。

すべてのインスタンスのステータスを表示するには、次のコマンドを使用します。

```
aws ec2 describe-instance-status
```

インスタンスステータスが impaired であるすべてのインスタンスのステータスを取得するには、次のコマンドを使用します。

```
aws ec2 describe-instance-status \
    --filters Name=instance-status.status,Values=impaired
```

単一のインスタンスのステータスを取得するには、以下のコマンドを使用します。

```
aws ec2 describe-instance-status \
    --instance-ids i-1234567890abcdef0
```

または、以下の コマンドを使用します。

- Get-EC2InstanceStatus (AWS Tools for Windows PowerShell)
- DescribeInstanceStatus (Amazon EC2 クエリ API)

ステータスチェックが失敗したインスタンスがある場合は、ステータスチェックに失敗したインスタンスのトラブルシューティングを参照してください。

ステータスチェックアラームの作成と編集

ステータスチェックメトリクスを使用して、インスタンスのステータスチェックに失敗したときに通知されるように CloudWatch アラームを作成することができます。

ステータスチェックアラームを作成するには、以下のいずれかの方法を使用します。

Console

次の手順に従って、E メールで通知を送信するか、ステータスチェックに失敗したときにインスタンスを停止、終了、または回復するアラームを設定します。

ステータスチェックアラームを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択して、[Status Checks (ステータスチェック)] タブを選択し、[アクション]、[Create status check alarm (ステータスチェックアラームの作成)] を選択します。
4. [Manage CloudWatch alarms (CloudWatch アラームの管理)] ページの [Add or edit alarm (アラームの追加または編集)] で、[Create an alarm (新しいアラームの作成)] を選択します。
5. [Alarm notification (アラーム通知)] で、トグルをオンにして Amazon Simple Notification Service (Amazon SNS) 通知を設定します。既存の Amazon SNS トピックを選択するか、名前を入力して新しいトピックを作成します。

   受信者のリストに E メールアドレスを追加したか、トピックを新規作成した場合、Amazon SNS から追加した各 E メールアドレスにサブスクリプションの確認メールメッセージが送信されます。各受信者は、そのメッセージに記載されているリンクを選択してサブスクリプションを確認する必要があります。アラート通知は確認されたアドレスにのみ送信されます。
6. [Alarm action (アラームアクション)] で、トグルをオンにして、アラームがトリガーされたときに実行するアクションを指定します。アクションを選択します。
7. [Alarm thresholds (アラームのしきい値)] で、アラームのメトリクスと条件を指定します。

   [Group samples] (サンプルグループ化) ([Average] (平均)) と [Type of data to sample] (サンプリングするデータのタイプ) (ステータスチェックも失敗) をデフォルト設定のままにするか、または必要に応じて変更することもできます。

   [Consecutive period] (連続期間) の場合、評価する期間数を設定し、[Period] (期間) で、アラームをトリガーして E メールを送信するまでの評価の間隔を入力します。

8. (オプション) [Sample metric data] (サンプルメトリクスデータ) の場合、[Add to dashboard] (ダッシュボードに追加) を選択します。
9. [Create] (作成) を選択します。

インスタンスステータスのアラームを変更する必要がある場合は、そのアラームを編集できます。

ステータスチェックアラームを編集するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Actions (アクション)]、[Monitoring (モニタリング)]、[Manage CloudWatch alarms (CloudWatch アラームの管理)] の順に選択します。
4. [Manage CloudWatch alarms (CloudWatch アラームの管理)] ページの [Add or edit alarm (アラームの追加または編集)] で、[Edit an alarm (新しいアラームの編集)] を選択します。
5. [Search for alarm (アラームの検索)] で、アラームを選択します。
6. 変更が完了したら、[Update (更新)] を選択します。

Command line

次の例では、インスタンスが少なくとも 2 つの連続する期間内にインスタンスチェックまたはシステムステータスチェックに失敗した場合、アラームが SNS トピックに通知 `arn:aws:sns:us-west-2:111122223333:my-sns-topic` を発行します。使用する CloudWatch メトリクスは `StatusCheckFailed` です。

AWS CLI を使用してステータスチェックアラームを作成するには

1. 既存の SNS トピックを選択するか、新しいキーペアを作成することができます。詳細については、AWS Command Line Interface ユーザーガイドのAmazon SNS での AWS CLI の使用を参照してください。
2. Amazon EC2 の使用可能な Amazon CloudWatch メトリクスを表示するには、list-metrics コマンドを使用します。

```
aws cloudwatch list-metrics --namespace AWS/EC2
```

3. アラームを作成するには、次の put-metric-alarm コマンドを使用します。

```
aws cloudwatch put-metric-alarm \
    --alarm-name StatusCheckFailed-Alarm-for-i-1234567890abcdef0 \
    --metric-name StatusCheckFailed \
    --namespace AWS/EC2 \
    --statistic Maximum \
    --dimensions Name=InstanceId,Value=i-1234567890abcdef0 \
    --unit Count \
    --period 300 \
    --evaluation-periods 2 \
    --threshold 1 \
    --comparison-operator GreaterThanOrEqualToThreshold \
    --alarm-actions arn:aws:sns:us-west-2:111122223333:my-sns-topic
```

期間は Amazon CloudWatch メトリクスが収集される期間 (秒) です。この例では、60 秒に 5 分を乗算した 300 を使用します。評価期間は、メトリクスの値がしきい値と比較されなければならない連続した期間の数です。この例では 2 を使用します。アラームアクションは、このアラームがトリガーされたときに実行するアクションです。この例では、Amazon SNS を使用してメールを送信するようにアラームを設定します。

インスタンスの状態変更イベント

インスタンスの状態が変化すると、Amazon EC2 は Amazon EventBridge に `EC2 Instance State-change Notification` イベントを送信します。

以下はこのイベントのサンプルデータです。この例では、インスタンスは pending 状態に入りました。

```
{
   "id":"7bf73129-1428-4cd3-a780-95db273d1602",
   "detail-type":"EC2 Instance State-change Notification",
   "source":"aws.ec2",
   "account":"123456789012",
   "time":"2021-11-11T21:29:54Z",
   "region":"us-east-1",
   "resources":[
      "arn:aws:ec2:us-east-1:123456789012:instance/i-abcd1111"
   ],
   "detail":{
      "instance-id":"i-abcd1111",
      "state":"pending"
```

```
    }
}
```

state に指定できる値は、次のとおりです。

- pending
- running
- stopping
- stopped
- shutting-down
- terminated

インスタンスを起動または開始した場合、インスタンスは pending 状態に移行してから、running 状態になります。インスタンスを停止した場合、インスタンスは stopping 状態に移行してから、stopped 状態になります。インスタンスを終了した場合、インスタンスは shutting-down 状態に移行してから、terminated 状態になります。

インスタンスの状態が変化したらメール通知を受け取る

インスタンスの状態が変化したときに E メール通知を受け取るには、Amazon SNS トピックを作成してから、EC2 Instance State-change Notification イベントの EventBridge ルールを作成します。

SNS トピックを作成するには

1. Amazon SNS コンソール (https://console.aws.amazon.com/sns/v3/home) を開きます。
2. ナビゲーションペインで、[トピック] を選択します。
3. [Create topic] (トピックの作成) を選択します。
4. [Type (タイプ)] で、[Standard (標準)] を選択します。
5. [Name] (名前) で、トピックの名前を入力します。
6. [Create topic] (トピックの作成) を選択します。
7. [サブスクリプションを作成] を選択します。
8. [Protocol (プロトコル)] として [Email (E メール)] を選択します。
9. [Endpoint] (エンドポイント) で、通知を受信するメールアドレスを入力します。
10. [サブスクリプションを作成] を選択します。

11. 次の件名の E メールメッセージが届きます: AWS Notification - Subscription Confirmation。指示に沿って操作し、登録を確認します。

EventBridge ルールを作成するには

1. Amazon EventBridge コンソール (https://console.aws.amazon.com/events/) を開きます。
2. [Create rule] を選択します。
3. [Name] (名前) に、ルールの名前を入力します。
4. ルールタイプ では、イベントパターンを持つルール] を選択します。
5. [Next] を選択します。
6. [Event pattern] (イベントパターン) の場合は、次のいずれかを実行します。
   a. イベントソース で AWS のサービス を選択します。
   b. [AWS のサービス] で、[EC2] を選択します。
   c. [イベントタイプ] に、[EC2 インスタンスの状態変更通知] を選択します。
   d. デフォルトでは、すべてのインスタンスの状態変更に関する通知が送信されます。必要に応じて、特定の状態またはインスタンスを選択できます。
7. [Next] を選択します。
8. 次のようにターゲットを指定します。
   a. [Target types] (ターゲットタイプ) には、[AWS のサービス] を選択します。
   b. [Select a target] (ターゲットの選択) には、[SNS topic] (SNS トピック) を選択します。
   c. [Topic] (トピック) で、前の手順で作成した SNS トピックを選択します。
9. [Next] を選択します。
10. (オプション) ルールにタグを追加します。
11. [Next] を選択します。
12. [ルールを作成] を選択します。
13. ルールをテストするには、状態変更を開始します。例えば、停止されたインスタンスを開始したり、実行中のインスタンスを停止したり、インスタンスを起動したりします。次の件名の E メールメッセージが届きます: AWS Notification Message。E メールの本文には、イベントデータが含まれます。

インスタンスの予定されたイベント

AWS は、再起動、停止/開始、またはリタイアなど、インスタンスのイベントを予定できます。これらのイベントは頻繁には発生しません。インスタンスのいずれかが予定されたイベントの影響を受ける場合、予定されたイベントの前に AWS アカウントに関連付けられた E メールアドレスに E メールが AWS から送信されます。この E メールは、開始日と終了日などのイベントの詳細を提供します。イベントによっては、イベントのタイミングを管理するアクションを実行できる場合があります。AWS は、Amazon CloudWatch Events によるモニタリングと管理が可能な AWS Health イベントも送信します。CloudWatch による AWS Health イベントのモニタリングの詳細については、CloudWatch Events による AWS Health イベントのモニタリングを参照してください。

スケジュールされたイベントは AWS によって管理されます。インスタンスのイベントをスケジュールすることはできません。AWS によりスケジュールされたイベントを表示したり、スケジュールされたイベント通知をカスタマイズして、E メールの通知からタグを追加または削除できます。また、スケジュールされた時刻にインスタンスの再起動やリタイア、停止などのアクションを実行できます。

予定されたイベントに通知を受け取ることができるようにアカウントの連絡先情報を更新するには、アカウント設定ページを参照してください。

> **Note**
>
> インスタンスがスケジュールされたイベントの影響を受け、それが Auto Scaling グループの一部である場合、Amazon EC2 Auto Scaling はヘルスチェックの一部として最終的にそのインスタンスを置き換えるので、追加のアクションは必要ありません。Amazon EC2 Auto Scaling によって実行されるヘルスチェックの詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Auto Scaling インスタンスのヘルスチェック」を参照してください。

コンテンツ

- スケジュールされたイベントの再スケジュール
- スケジュールしたイベント用のイベントウィンドウの定義

予定されたイベントのタイプ

Amazon EC2 では、インスタンスに関連して以下のタイプのイベントがスケジュールされた時刻に発生するようにできます。

- インスタンスの停止: スケジュールされた時刻になると、インスタンスは停止します。再度起動すると、新しいホストに移行されます。Amazon EBS によってバックアップされるインスタンスにのみ適用されます。
- Instance retirement (インスタンスのリタイア): スケジュールされた時刻に、インスタンスは、Amazon EBS によってバックアップされると停止し、インスタンスストアによってバックアップされると削除されます。
- インスタンスの再起動: スケジュールされた時刻になると、インスタンスは再起動されます。
- システムの再起動: スケジュールされた時刻になると、インスタンスのホストは再起動されます。
- [System maintenance]: スケジュールされた時刻になると、インスタンスは、ネットワークメンテナンスまたは電源のメンテナンスの影響を一時的に受ける場合があります。

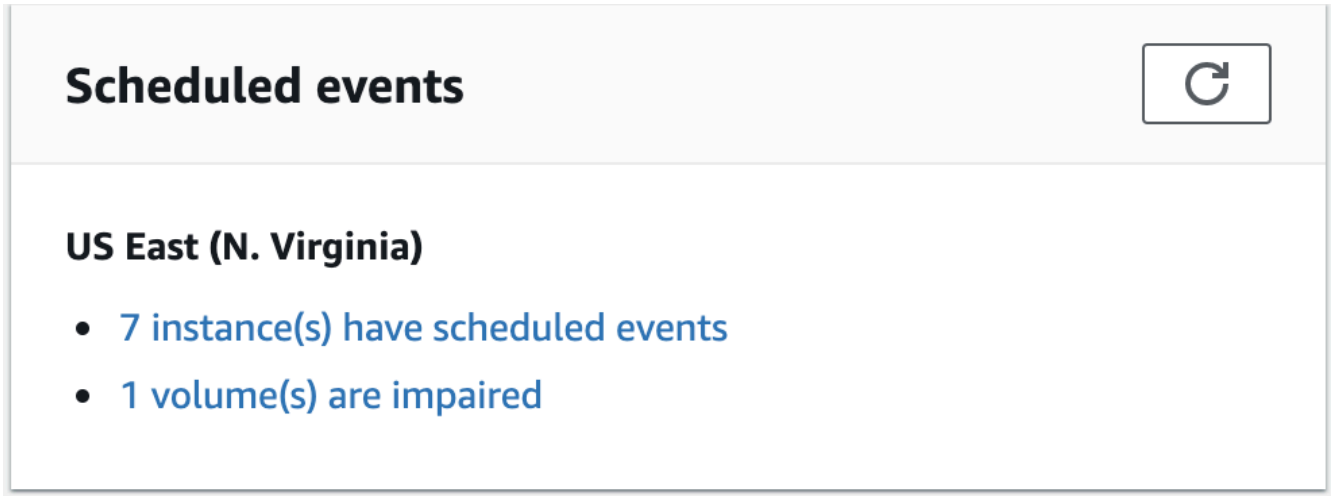
予定されたイベントの表示

予定されたイベントの通知を E メールで受信することに加え、以下のいずれかの方法を使用して予定されたイベントを確認できます。

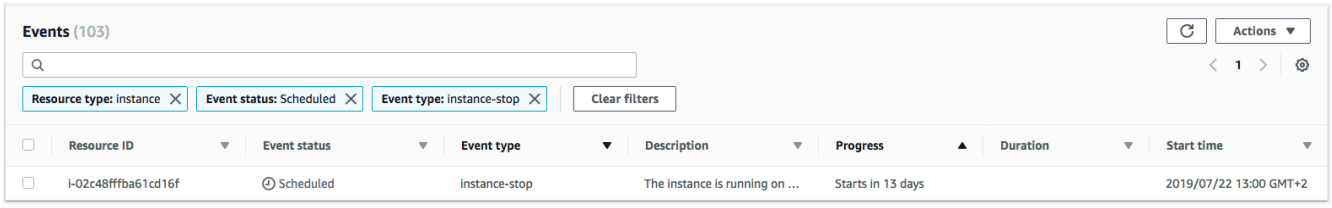
Console

インスタンスに予定されたイベントを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ダッシュボードには、[Scheduled events] の下に関連付けられているイベントを持つすべてのリソースが表示されます。



3. 詳細については、ナビゲーションペインで [イベント] を選択してください。イベントに関連付けられたリソースがすべて表示されます。イベントタイプ、リソースタイプ、アベイラビリティーゾーンなどの特性でフィルタリングできます。



AWS CLI

インスタンスに予定されたイベントを表示するには

describe-instance-status コマンドを使用します。

```
aws ec2 describe-instance-status \
    --instance-id i-1234567890abcdef0 \
    --query "InstanceStatuses[].Events"
```

以下の出力例は、再起動イベントを示しています。

```
[
    "Events": [
        {
            "InstanceEventId": "instance-event-0d59937288b749b32",
            "Code": "system-reboot",
            "Description": "The instance is scheduled for a reboot",
            "NotAfter": "2019-03-15T22:00:00.000Z",
```

```
            "NotBefore": "2019-03-14T20:00:00.000Z",
            "NotBeforeDeadline": "2019-04-05T11:00:00.000Z"
         }

    ]
]
```

インスタンスのリタイアイベントを示す出力例を次に示します。

```
[
    "Events": [
        {
            "InstanceEventId": "instance-event-0e439355b779n26",

            "Code": "instance-stop",
            "Description": "The instance is running on degraded hardware",
            "NotBefore": "2015-05-23T00:00:00.000Z"
        }
    ]
]
```

PowerShell

AWS Tools for Windows PowerShell を使用してインスタンスに予定されたイベントを表示するには

次の Get-EC2InstanceStatus コマンドを使用します。

```
PS C:\> (Get-EC2InstanceStatus -InstanceId i-1234567890abcdef0).Events
```

インスタンスのリタイアイベントを示す出力例を次に示します。

```
Code         : instance-stop
Description  : The instance is running on degraded hardware
NotBefore    : 5/23/2015 12:00:00 AM
```

Instance metadata

インスタンスメタデータを使用してインスタンスに予定されたイベントを表示するには

インスタンスのアクティブなメンテナンスイベントに関する情報は、インスタンスメタデータサービスバージョン 2 または インスタンスメタデータサービスバージョン 1 を使用してインスタンスメタデータから取得できます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/events/maintenance/scheduled
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/events/maintenance/scheduled
```

以下は、予定されたシステムの再起動イベントに関する情報を JSON 形式で出力した例です。

```
[
  {
    "NotBefore" : "21 Jan 2019 09:00:43 GMT",
    "Code" : "system-reboot",
    "Description" : "scheduled reboot",
    "EventId" : "instance-event-0d59937288b749b32",
    "NotAfter" : "21 Jan 2019 09:17:23 GMT",
    "State" : "active"
  }
]
```

インスタンスメタデータを使用して、インスタンスの完了またはキャンセルされたイベントのイベント履歴を表示するには

インスタンスの完了済みまたはキャンセル済みイベントに関する情報は、インスタンスメタデータサービスバージョン 2 または インスタンスメタデータサービスバージョン 1 を使用してインスタンスメタデータから取得できます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
```

```
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/events/maintenance/history
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/events/maintenance/history
```

以下は、取り消されたシステム再起動イベントおよび完了したシステム再起動イベントに関する情報を JSON 形式で出力した例です。

```
[
  {
    "NotBefore" : "21 Jan 2019 09:00:43 GMT",
    "Code" : "system-reboot",
    "Description" : "[Canceled] scheduled reboot",
    "EventId" : "instance-event-0d59937288b749b32",
    "NotAfter" : "21 Jan 2019 09:17:23 GMT",
    "State" : "canceled"
  },
  {
    "NotBefore" : "29 Jan 2019 09:00:43 GMT",
    "Code" : "system-reboot",
    "Description" : "[Completed] scheduled reboot",
    "EventId" : "instance-event-0d59937288b749b32",
    "NotAfter" : "29 Jan 2019 09:17:23 GMT",
    "State" : "completed"
  }
]
```

AWS Health

AWS Health Dashboard を使用して、インスタンスに影響を与える可能性があるイベントについて確認できます。AWS Health Dashboard では、未解決の問題、予定された変更、その他の通知という 3 つのグループに問題が分類されます。予定された変更には、進行中または予定されている変更が含まれます。

詳細については、AWS Health ユーザーガイドのAWS Health Dashboard の開始を参照してください。

スケジュールされたイベント通知のカスタマイズ

スケジュールされたイベント通知をカスタマイズして、メール通知にタグを含めることができます。これにより、影響を受けるリソース (インスタンスまたは Dedicated Hosts) を特定して、その後のイベントに対するアクションに優先順位を付けやすくなります。

タグを含めるようにイベント通知をカスタマイズする場合、次のいずれかを含めることができます。

- 影響を受けるリソースに関連付けられているすべてのタグ
- 影響を受けるリソースに関連付けられている特定のタグのみ

例えば、`application`、`costcenter`、`project`、`owner` タグをすべてのインスタンスに割り当てるとします。イベント通知には、これらのすべてのタグを含めることができます。また、イベント通知に `owner` タグと `project` タグのみを表示したい場合は、それらのタグのみを含めることもできます。

含めるタグを選択すると、イベント通知には、影響を受けるリソースに関連付けられているリソース ID (インスタンス ID または Dedicated Host ID) とタグのキーと値のペアが含まれます。

タスク

- イベント通知にタグを含める
- イベント通知からのタグの削除
- イベント通知に含めるタグの表示

イベント通知にタグを含める

含めるように選択したタグは、選択したリージョンのすべてのリソース (インスタンスと Dedicated Hosts) に適用されます。他のリージョンのイベント通知をカスタマイズするには、まず必要なリージョンを選択してから、次の手順を実行します。

イベント通知のタグは、次のいずれかの方法で含めることができます。

Console

イベント通知にタグを含めるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。

3. [アクション]、[Manage event notifications (イベント通知の管理)] の順に選択します。
4. [イベント通知にタグを含める]をオンにします。
5. イベント通知に含めるタグに応じて、次のいずれかの操作を行います。
   - 影響を受けるインスタンスまたは専有ホストに関連付けられている全タグを含めるには、[全タグを含める] を選択します。
   - 含めるタグを選択するには [含めるタグを選択] を選択し、タグキーを選択または入力します。
6. [Save] を選択します。

AWS CLI

イベント通知にすべてのタグを含めるには

AWS CLI コマンドの register-instance-event-notification-attributes を使用して、`IncludeAllTagsOfInstance` パラメータを `true` に設定します。

```
aws ec2 register-instance-event-notification-attributes \
    --instance-tag-attribute "IncludeAllTagsOfInstance=true"
```

イベント通知に特定のタグを含めるには

AWS CLI コマンドの register-instance-event-notification-attributes を使用して、`InstanceTagKeys` パラメータを使用して含めるタグを指定します。

```
aws ec2 register-instance-event-notification-attributes \
    --instance-tag-attribute 'InstanceTagKeys=["tag_key_1", "tag_key_2",
 "tag_key_3"]'
```

イベント通知からのタグの削除

イベント通知のタグは、次のいずれかの方法で削除することができます。

Console

イベント通知からタグを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインの [Events] を選択します。
3. [アクション]、[Manage event notifications (イベント通知の管理)] の順に選択します。
4. イベント通知からすべてのタグを削除するには、[イベント通知にタグを含める] をオフにします。
5. イベント通知から特定のタグを削除するには、対応するタグキーの [X] を選択します。
6. [Save] を選択します。

AWS CLI

イベント通知からすべてのタグを削除するには

AWS CLI コマンドの deregister-instance-event-notification-attributes を使用して、`IncludeAllTagsOfInstance` パラメータを `false` に設定します。

```
aws ec2 deregister-instance-event-notification-attributes \
    --instance-tag-attribute "IncludeAllTagsOfInstance=false"
```

イベント通知から特定のタグを削除するには

AWS CLI コマンドの deregister-instance-event-notification-attributes を使用して、`InstanceTagKeys` パラメータを使用して削除するタグを指定します。

```
aws ec2 deregister-instance-event-notification-attributes \
    --instance-tag-attribute 'InstanceTagKeys=["tag_key_1", "tag_key_2",
 "tag_key_3"]'
```

イベント通知に含めるタグの表示

イベント通知に含めるタグは、次のいずれかの方法で表示することができます。

Console

イベント通知に含めるタグを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. [アクション]、[Manage event notifications (イベント通知の管理)] の順に選択します。

AWS CLI

イベント通知に含めるタグを表示するには

AWS CLI コマンドの describe-instance-event-notification-attributes を使用します。

```
aws ec2 describe-instance-event-notification-attributes
```

停止またはリタイアが予定されているインスタンスの操作

AWS は、インスタンスの基盤となるホストの回復不能な障害を検出すると、インスタンスのルートデバイスのタイプに応じて、インスタンスの停止または削除を予定します。ルートデバイスが EBS ボリュームの場合、インスタンスが停止するように予定されます。ルートデバイスがインスタンスストアボリュームの場合、インスタンスは終了するように予定されます。詳細については、インスタンスのリタイアを参照してください。

> **Important**
>
> インスタンスストアボリュームに格納されているデータはいずれも、インスタンスが停止、休止、または終了されると失われます。これには、EBS ボリュームをルートデバイスとするインスタンスにアタッチされたインスタンスストアボリュームも含まれます。インスタンスが停止、休止、または終了される前に、後で必要となるインスタンスストアボリュームからデータを必ず保存しておきます。

Amazon EBS によりバックアップされたインスタンスのアクション

インスタンスが予定どおりに停止されるのを待機できます。または、インスタンスを自分で停止および起動して、新しいホストに移行することもできます。インスタンスが停止したときにインスタンス設定を変更する方法に加えて、インスタンスの停止についての詳細は、インスタンスの停止と起動を参照してください。

スケジュールされたインスタンスの停止イベントに対応した、即時の停止と開始を自動化することができます。詳細については、「AWS Health ユーザーガイド」の「Amazon EC2 インスタンスのアクションの自動化」を参照してください。

インスタンスストアによりバックアップされたインスタンスのアクション

最新の AMI から代替インスタンスを起動し、インスタンスの削除を予定する前に必要なすべてのデータを代替インスタンスに移行することをお勧めします。その後、元のインスタンスを終了するか、予定どおりに終了されるのを待機することができます。

再起動が予定されているインスタンスの操作

AWS は、更新のインストールや基盤となるホストのメンテナンスなどのタスクを実行する必要があるとき、インスタンスまたは基盤となるホストの再起動を予定できます。都合に合わせて指定する日付と時刻にインスタンスが再起動するように、ほとんどの再起動イベントを再スケジュールできます。

再起動イベントタイプの表示

次のいずれかの方法を使用して、再起動イベントがインスタンスの再起動またはシステムの再起動であるかを確認できます。

Console

予定された再起動イベントのタイプを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. フィルターリストから [リソースタイプ: インスタンス] を選択します。
4. インスタンスごとに、[イベントタイプ] 列の値を表示します。値は system-reboot または instance-reboot のいずれかです。

AWS CLI

予定された再起動イベントのタイプを表示するには

describe-instance-status コマンドを使用します。

```
aws ec2 describe-instance-status \
    --instance-id i-1234567890abcdef0
```

スケジュールされた再起動イベントでは、Code の値は `system-reboot` あるいは `instance-reboot` です。次の出力例は `system-reboot` イベントを示しています。

```
[
```

```
    "Events": [
                {
                    "InstanceEventId": "instance-event-0d59937288b749b32",
                    "Code": "system-reboot",
                    "Description": "The instance is scheduled for a reboot",
                    "NotAfter": "2019-03-14T22:00:00.000Z",
                    "NotBefore": "2019-03-14T20:00:00.000Z",
                    "NotBeforeDeadline": "2019-04-05T11:00:00.000Z"
                }
            ]
]
```

インスタンス再起動のアクション

予定されたメンテナンスウィンドウ内でのインスタンスの再起動まで待機することも、都合に合わせた日付と時刻にインスタンスの再起動を再スケジュールすることも、または都合のよい時間にインスタンスを手動で再起動することもできます。

インスタンスが再起動されると、予定されたイベントがクリアになり、このイベントの説明が更新されます。基になるホストに対する保留中のメンテナンスが完了し、インスタンスが完全に起動したら、インスタンスの使用を再開できます。

システム再起動のアクション

システムを自分で再起動することはできません。予定されたメンテナンスウィンドウ中におけるシステムの再起動まで待機することも、都合に合わせた日付と時刻でシステムの再起動を再スケジュールすることもできます。システムの再起動は通常数分で完了します。システムの再起動後、インスタンスの IP アドレスと DNS 名、およびローカルインスタンスストアボリュームのデータは保持されます。システムの再起動が完了すると、インスタンスに予定されているイベントはクリアされ、インスタンスのソフトウェアが正常に動作していることを確認できます。

または、インスタンスのメンテナンス時間を変更する必要があり、システムの再起動を再スケジュールできない場合は、Amazon EBS-backed インスタンスを停止して再起動すると、新しいホストに移行できます。ただし、ローカルインスタンスストアボリュームのデータは保持されません。また、スケジュールされたシステム再起動イベントに対応した、インスタンスの即時の停止と開始を自動化することができます。詳細については、AWS Health ユーザーガイドのEC2 インスタンスのアクションの自動化を参照してください。Instance Store-Backed インスタンスでシステムの再起動を再スケジュールできない場合、最新の AMI から代替インスタンスを起動し、予定されたメンテナンス期間より前に必要なデータをすべて代替インスタンスに移行した後、元のインスタンスを削除できます。

メンテナンスが予定されているインスタンスの操作

AWS は、インスタンスの基盤となるホストをメンテナンスする必要があるときに、インスタンスのメンテナンスを予定します。2 種類のメンテナンスイベントがあります。1 つはネットワークメンテナンスで、もう 1 つは電源のメンテナンスです。

ネットワークメンテナンス中は、短い期間、予定されたインスタンスのネットワーク接続が切断されます。メンテナンスが終了すると、インスタンスとの通常のネットワーク接続が回復します。

電源のメンテナンス中は、短い期間、予定されたインスタンスはオフラインになり、その後再起動されます。再起動されると、インスタンスの設定内容はすべて維持されます。

インスタンスが再起動したら (通常、数分かかります)、アプリケーションが正常に動作していることを確認します。この時点で、インスタンスにスケジュールされたイベントは残っていません。残っている場合は、スケジュールされたイベントの説明の先頭に [Completed] と表示されます。インスタンスのステータス説明が更新するのに、最大で 1 時間ほどかかる場合があります。完了したメンテナンスイベントは、最長で 1 週間、Amazon EC2 コンソールのダッシュボードに表示されます。

Amazon EBS によりバックアップされたインスタンスのアクション

メンテナンスが予定どおりに実行されるのを待機できます。または、インスタンスを停止および起動して、新しいホストに移行することもできます。インスタンスが停止したときにインスタンス設定を変更する方法に加えて、インスタンスの停止についての詳細は、インスタンスの停止と起動を参照してください。

スケジュールされたメンテナンスイベントに対応した、即時の停止と開始を自動化することができます。詳細については、AWS Health ユーザーガイドのEC2 インスタンスのアクションの自動化を参照してください。

インスタンスストアによりバックアップされたインスタンスのアクション

メンテナンスが予定どおりに実行されるのを待機できます。または、予定されたメンテナンス期間中に通常の運用を維持する場合、最新の AMI から代替インスタンスを起動し、予定されたメンテナンス期間より前に必要なデータをすべて代替インスタンスに移行した後、元のインスタンスを終了できます。

スケジュールされたイベントの再スケジュール

都合の良い日時にイベントが発生するように、予定を再スケジュールできます。期限が設定されているイベントのみを再スケジュールできます。イベントの再スケジュールに適用される制限は他にもあります。

イベントは、次のいずれかの方法で再スケジュールできます。

Console

イベントを再スケジュールするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. フィルターリストから [リソースタイプ: インスタンス] を選択します。
4. 1 つ以上のインスタンスを選択し、[アクション]、[Schedule event] の順に選択します。

   [期限] でイベント期限を設定したイベントのみを再スケジュールできます。選択したイベントのいずれかに期限がない場合、[アクション]、[Schedule event] は無効になります。
5. [New start time] に、イベントの新しい日時を入力します。新しい日時は、[Event deadline] より前に設定する必要があります。
6. [Save] を選択します。

   更新されたイベント開始時刻がコンソールに反映されるまで、1～2 分かかることがあります。

AWS CLI

イベントを再スケジュールするには

1. `NotBeforeDeadline` の値で示されるイベント期限があるイベントのみ、再スケジュールできます。describe-instance-status コマンドを使用して `NotBeforeDeadline` パラメータ値を表示します。

```
aws ec2 describe-instance-status \
    --instance-id i-1234567890abcdef0
```

   次の出力例は、`system-reboot` に値があるため再スケジュールできる `NotBeforeDeadline` イベントを示しています。

```
[
    "Events": [
                {
                    "InstanceEventId": "instance-event-0d59937288b749b32",
```

```
                    "Code": "system-reboot",
                    "Description": "The instance is scheduled for a reboot",
                    "NotAfter": "2019-03-14T22:00:00.000Z",
                    "NotBefore": "2019-03-14T20:00:00.000Z",
                    "NotBeforeDeadline": "2019-04-05T11:00:00.000Z"
                }
            ]
]
```

2. イベントを再スケジュールするには、modify-instance-event-start-time コマンドを使用します。not-before パラメータを使用して新しいイベント開始時刻を指定します。新しいイベント開始時刻は、NotBeforeDeadline より前にする必要があります。

```
aws ec2 modify-instance-event-start-time \
    --instance-id i-1234567890abcdef0 \
    --instance-event-id instance-event-0d59937288b749b32 \
    --not-before 2019-03-25T10:00:00.000
```

describe-instance-status コマンドが更新された not-before パラメータ値を返すまでに、1 ～2 分かかることがあります。

制限事項

- イベント期限があるイベントのみ再スケジュールできます。イベントは、イベント期限日まで再スケジュールできます。コンソールの [期限] 列と NotBeforeDeadline の AWS CLI フィールドは、イベントに期限が設定されていることを示します。
- まだ開始していないイベントのみ再スケジュールできます。コンソールの [開始時刻] 列と NotBefore の AWS CLI フィールドは、イベントの開始時刻を示します。あと 5 分で開始するようにスケジュールされているイベントは、再スケジュールできません。
- 新しいイベント開始時刻は、現在の時刻から少なくとも 60 分後にする必要があります。
- コンソールを使用して複数のイベントを再スケジュールすると、イベント期限は最も早い期限日のイベントによって決定されます。

スケジュールしたイベント用のイベントウィンドウの定義

スケジュールされたイベントに対して、週ごとに繰り返されるカスタムのイベントウィンドウを定義して、Amazon EC2 インスタンスを再起動、停止、終了させることができます。イベントウィンドウには、1 つ以上のインスタンスを関連付けることができます。これらのインスタンスにスケジュー

ルされたイベントが設定されている場合、AWS は、関連するイベントウィンドウ内でイベントをスケジュールします。

ワークロードのオフピーク期間にイベントウィンドウを指定することで、ワークロードの可用性を最大化できます。また、内部的な保守スケジュールにイベントウィンドウを合わせることもできます。

イベントウィンドウを定義するには、一連の時間範囲を指定します。最小期間は 2 時間です。全体を合計した時間範囲は、最小で 4 時間必要です。

インスタンス ID またはインスタンスタグを使用して、1 つ以上のインスタンスをイベントウィンドウに関連付けることができます。また、ホスト ID を使用して、Dedicated Hosts をイベントウィンドウに関連付けることもできます。

> **Warning**
>
> イベントウィンドウは、インスタンスを停止、再起動、または終了する、スケジュールされたイベントにのみ適用されます。
> イベントウィンドウは、以下には適用されません。
>
> - 繰り上げられた、スケジュールされたイベントとネットワーク保守イベント。
> - AutoRecovery や予期しない再起動などのスケジュール外の保守作業。

イベントウィンドウの使用

- 考慮事項
- イベントウィンドウの表示
- イベントウィンドウの作成
- イベントウィンドウの変更
- イベントウィンドウの削除
- イベントウィンドウのタグ付け

考慮事項

- イベントウィンドウの時刻はすべて UTC で表示されます。
- 週ごとのイベントウィンドウの最小期間は 4 時間です。
- 各イベント期間内の時間範囲は、少なくとも 2 時間に設定する必要があります。

- イベントウィンドウには、ターゲットタイプ (インスタンスID、Dedicated Host ID、またはインスタンスタグ) を 1 つだけ関連付けることができます。
- 1 つのターゲット (インスタンス ID、Dedicated Host ID、またはインスタンスタグ) は、1 つのイベントウィンドウにのみ関連付けることが可能です。
- 1 つのイベントウィンドウには、最大 100 個のインスタンス ID、または 50 個の Dedicated Host ID、または 50 個のインスタンスタグを関連付けることができます。インスタンスタグは、任意の数のインスタンスに関連付けることができます。
- 個々の AWS リージョンで、最大 200 個までのイベントウィンドウを作成できます。
- 複数のインスタンスがイベントウィンドウに関連付けられている場合、スケジュールされたイベントが同時に発生する可能性があります。
- 既に AWS によりスケジュールされたイベントが存在する場合、イベントウィンドウを変更しても、スケジュールされたイベントの時間は変更されません。イベントに締め切り日がある場合は、イベントの再スケジュールが行えます。
- インスタンスを新しいホストに移行するためのスケジュールされたイベントの前に、そのインスタンスを停止および開始することで、スケジュールされたイベントを発生しないようにできます。

イベントウィンドウの表示

次のいずれかの方法で、イベントウィンドウを表示できます。

Console

イベントウィンドウを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. [Actions] (アクション)、[Manage event windows] (イベントウィンドウの管理) を選択します。
4. イベントウィンドウを選択し詳細を表示します。

AWS CLI

すべてのイベントウィンドウを表示するには

describe-instance-event-windows コマンドを使用します。

```
aws ec2 describe-instance-event-windows \
```

```
    --region us-east-1
```

正常な出力

```
{
    "InstanceEventWindows": [
        {
            "InstanceEventWindowId": "iew-0abcdef1234567890",
            "Name": "myEventWindowName",
            "CronExpression": "* 21-23 * * 2,3",
            "AssociationTarget": {
                "InstanceIds": [
                    "i-1234567890abcdef0",
                    "i-0598c7d356eba48d7"
                ],
                "Tags": [],
                "DedicatedHostIds": []
            },
            "State": "active",
            "Tags": []
        }

        ...

    ],
    "NextToken": "9d624e0c-388b-4862-a31e-a85c64fc1d4a"
}
```

特定のイベントウィンドウを表示するには

describe-instance-event-windows コマンドで --instance-event-window-idパラメータを使用して、特定のイベントウィンドウの詳細を表示します。

```
aws ec2 describe-instance-event-windows \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890
```

1 つ以上のフィルターに一致するイベントウィンドウを表示するには

describe-instance-event-windows コマンドで --filters パラメータを使用します。以下の例では、指定されたインスタンスに関連付けられているすべてのイベントウィンドウを表示するために、instance-id フィルターを使用しています。

フィルタを使用すると、直接的な一致が評価されます。ただし、instance-id フィルターの場合は異なります。直接一致するインスタンス ID が見つからない場合は、インスタンスタグや Dedicated Host ID (インスタンスが Dedicated Host 上にある場合) など、イベントウィンドウとの間接的な関連付けまでが評価されます。

サポートされているフィルタの一覧については、AWS CLIリファレンスのdescribe-instance-event-windowsを参照してください。

```
aws ec2 describe-instance-event-windows \
    --region us-east-1 \
    --filters Name=instance-id,Values=i-1234567890abcdef0 \
    --max-results 100 \
    --next-token <next-token-value>
```

正常な出力

次の例では、インスタンスはイベントウィンドウに関連付けられた Dedicated Host 上に置かれています。

```
{
    "InstanceEventWindows": [
        {
            "InstanceEventWindowId": "iew-0dbc0adb66f235982",
            "TimeRanges": [
                {
                    "StartWeekDay": "sunday",
                    "StartHour": 2,
                    "EndWeekDay": "sunday",
                    "EndHour": 8
                }
            ],
            "Name": "myEventWindowName",
            "AssociationTarget": {
                "InstanceIds": [],
                "Tags": [],
                "DedicatedHostIds": [
                    "h-0140d9a7ecbd102dd"
                ]
            },
            "State": "active",
            "Tags": []
        }
```

```
    ]
}
```

イベントウィンドウの作成

イベントウィンドウは、複数作成できます。イベントウィンドウごとに、1 つ以上の時間ブロックを指定します。例えば、毎日の午前 4 時に発生し 2 時間継続する時間ブロックを持つイベントウィンドウを作成できます。あるいは、日曜日の午前 2 時から午前 4 時、および水曜日の午前 3 時から午前 5 時に発生する時間ブロックを持つイベントウィンドウを作成することもできます。

イベントウィンドウの制限については、このトピックの前半で 考慮事項 を参照してください。

イベントウィンドウは、削除されない限り毎週繰り返されます。

イベントウィンドウを作成するには、次のいずれかの方法を使用します。

Console

イベントウィンドウを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. [Actions] (アクション)、[Manage event windows] (イベントウィンドウの管理) を選択します。
4. [インスタンスのイベントウィンドウを作成] をクリックします。
5. [イベントウィンドウ名] に、イベントウィンドウのわかりやすい名前を入力します。
6. [イベントウィンドウのスケジュール] で、イベントウィンドウ内の時間ブロックを指定するために、Cron スケジュールビルダーを使用するか、あるいは時間範囲で指定するかを選択します。

   - [Cron スケジュールビルダー] を選択した場合は以下を指定します。
     1. [曜日 (UTC)] で、イベントウィンドウを発生させる曜日を指定します。
     2. [開始時刻 (UTC)] で、イベントウィンドウが開始する時刻を指定します。
     3. [期間] で、イベントウィンドウ内の時間ブロックの継続時間を指定します。各時間ブロックに設定できる最小期間は 2 時間です。イベントウィンドウの最小期間は、合計で 4 時間以上にする必要があります。すべての時刻は協定世界時 (UTC) です。

- [時間範囲] を選択した場合は、[新しい時間範囲の追加] をクリックし、開始する日時ならびに終了する日時を指定します。各時間範囲に対して、これを繰り返します。各時間範囲に設定できる最小期間は 2 時間です。時間範囲の最小期間は、全体を合計して 4 時間以上にする必要があります。

7. (オプション) [ターゲットの詳細] では、1 つ以上のインスタンスをイベントウィンドウに関連付けます。これにより、インスタンスでメンテナンスがスケジュールされている場合、関連付けられたイベントウィンドウ中に、スケジュールされたイベントが発生するように設定できます。インスタンス ID またはインスタンスタグを使用して、1 つ以上のインスタンスをイベントウィンドウに関連付けることができます。Dedicated Hosts をイベントウィンドウに関連付けるには、ホスト ID を使用します。

   イベントウィンドウの作成時、そのウィンドウとターゲットの関連付けは必須ではありません。作成後、ウィンドウを変更して、1 つ以上のターゲットを関連付けることができます。
8. (オプション) [イベントウィンドウのタグ] で、[タグを追加] をクリックし、タグのキーおよび値を入力します。各タグについて、これを繰り返します。
9. [イベントウィンドウの作成] をクリックします。

AWS CLI

AWS CLI を使用してイベントウィンドウを作成するには、まずイベントウィンドウを作成した後で、そのイベントウィンドウに 1 つ以上のターゲットを関連付けます。

イベントウィンドウを作成する

イベントウィンドウの作成時は、時間範囲のセットを指定するか、cron 式を使用するかのいずれかを定義できますが、両方を定義することはできません。

時間範囲を設定したイベントウィンドウを作成するには

`--time-range` パラメータを指定しながら create-instance-event-window コマンドを実行します。また、`--cron-expression` パラメータを指定することはできません。

```
aws ec2 create-instance-event-window \
    --region us-east-1 \
    --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8 \
    --tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
    --name myEventWindowName
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

cron 式を指定したイベントウィンドウを作成するには

`--cron-expression` パラメータを指定しながら create-instance-event-window コマンドを実行します。また、`--time-range` パラメータを指定することはできません。

```
aws ec2 create-instance-event-window \
    --region us-east-1 \
    --cron-expression "* 21-23 * * 2,3" \
    --tag-specifications "ResourceType=instance-event-
window,Tags=[{Key=K1,Value=V1}]" \
    --name myEventWindowName
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
```

```
        "CronExpression": "* 21-23 * * 2,3",
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

イベントウィンドウとターゲットの関連付け

イベントウィンドウには、1 つのタイプのターゲッ (インスタンス ID、Dedicated Host ID、またはインスタンスタグ) のみを関連付けることができます。

イベントウィンドウとインスタンスタグを関連付けるには

`instance-event-window-id` パラメータによりイベントウィンドウを指定しながら、associate-instance-event-window コマンドを実行します。インスタンスタグを関連付けるには、`--association-target` パラメータを使用し、その値に 1 つ以上のタグを指定します。

```
aws ec2 associate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [
                {
                    "Key": "k2",
                    "Value": "v2"
                },
                {
```

```
                    "Key": "k1",
                    "Value": "v1"
                }
            ],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

イベントウィンドウに 1 つ以上のインスタンスを関連付けるには

instance-event-window-id パラメータによりイベントウィンドウを指定しながら、associate-instance-event-window コマンドを実行します。インスタンスを関連付けるには --association-target パラメータを使用し、その値に 1 つ以上のインスタンス ID を指定します。

```
aws ec2 associate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [
                "i-1234567890abcdef0",
                "i-0598c7d356eba48d7"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

専有ホストとイベントウィンドウを関連付けるには

instance-event-window-id パラメータによりイベントウィンドウを指定しながら、associate-instance-event-window コマンドを実行します。専有ホストを関連付けるには、--association-targetパラメータを使用し、その値に 1 つ以上の Dedicated Host ID を指定します。

```
aws ec2 associate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "DedicatedHostIds=h-029fa35a02b99801d"
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": [
                "h-029fa35a02b99801d"
            ]
        },
        "State": "creating"
    }
}
```

イベントウィンドウの変更

イベントウィンドウに関しては、その ID 以外のすべてのフィールドを変更できます。例えば、夏時間の開始時に、イベントウィンドウのスケジュールを変更できます。既存のイベントウィンドウに対しては、ターゲットの追加または削除が必要になることもあります。

イベントウィンドウを変更するには、次のいずれかの方法を使用します。

Console

イベントウィンドウを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインの [Events] を選択します。
3. [Actions] (アクション)、[Manage event windows] (イベントウィンドウの管理) を選択します。
4. 変更するイベントウィンドウを選択し、そして[Actions] (アクション)、[Modify instance event window] (インスタンスイベントウィンドウの変更) を選択します。
5. イベントウィンドウの各フィールドを変更した後、[イベントウィンドウの変更] をクリックします。

AWS CLI

AWS CLI を使用してイベントウィンドウを変更する場合は、時間範囲または cron 式の変更や、1 つ以上のターゲットのイベントウィンドウへの関連付け、あるいはその関連付けの解除が可能です。

イベントウィンドウ時間の変更

イベントウィンドウでは、時間範囲または cron 式のいずれかの変更が可能ですが、両方を変更することはできません。

イベントウィンドウの時間範囲を変更するには

変更するイベントウィンドウを指定しながら、modify-instance-event-window コマンドを実行します。`--time-range` パラメータにより時間範囲を変更します。また、`--cron-expression` パラメータを指定することはできません。

```
aws ec2 modify-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --time-range StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
```

```
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

イベントウィンドウの時間範囲のセットを変更するには

変更するイベントウィンドウを指定しながら、modify-instance-event-window コマンドを実行します。--time-range パラメータにより時間範囲を変更します。また、--cron-expression パラメータを同じ呼び出しで指定することはできません。

```
aws ec2 modify-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --time-range '[{"StartWeekDay": "monday", "StartHour": 2, "EndWeekDay":
 "wednesday", "EndHour": 8},
   {"StartWeekDay": "thursday", "StartHour": 2, "EndWeekDay": "friday",
 "EndHour": 8}]'
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "TimeRanges": [
            {
```

```
                "StartWeekDay": "monday",
                "StartHour": 2,
                "EndWeekDay": "wednesday",
                "EndHour": 8
            },
            {
                "StartWeekDay": "thursday",
                "StartHour": 2,
                "EndWeekDay": "friday",
                "EndHour": 8
            }
        ],
        "Name": "myEventWindowName",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

イベントウィンドウの cron 式を変更するには

変更するイベントウィンドウを指定しながら、modify-instance-event-window コマンドを実行します。`--cron-expression` パラメータにより cron 式を変更します。また、`--time-range` パラメータを指定することはできません。

```
aws ec2 modify-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --cron-expression "* 21-23 * * 2,3"
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [
                "i-0abcdef1234567890",
                "i-0be35f9acb8ba01f0"
            ],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating",
        "Tags": [
            {
                "Key": "K1",
                "Value": "V1"
            }
        ]
    }
}
```

イベントウィンドウに関連付けられたターゲットの変更

イベントウィンドウには、追加のターゲットを関連付けることができます。また、イベントウィンドウで、ターゲットとの既存の関連付けを解除することもできます。ただし、イベントウィンドウには、1 つのタイプのターゲット (インスタンス ID、Dedicated Host ID、またはインスタンスタグ) のみを関連付けることができます。

イベントウィンドウに追加ターゲットを関連付けるには

ターゲットをイベントウィンドウに関連付ける手順については、Associate a target with an event windowを参照してください。

イベントウィンドウからインスタンスタグの関連付けを解除するには

`instance-event-window-id` パラメータを使用してイベントウィンドウを指定しながら、disassociate-instance-event-window コマンドを実行します。インスタンスタグの関連付けを解除するには、`--association-target` パラメータを使用し、その値に 1 つ以上のタグを指定します。

```
aws ec2 disassociate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceTags=[{Key=k2,Value=v2},{Key=k1,Value=v1}]"
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

イベントウィンドウから 1 つ以上のインスタンスの関連付けを解除するには

instance-event-window-id パラメータを使用してイベントウィンドウを指定しながら、disassociate-instance-event-window コマンドを実行します。インスタンスの関連付けを解除するには、--association-target パラメータを使用し、その値に 1 つ以上のインスタンス ID を指定します。

```
aws ec2 disassociate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target "InstanceIds=i-1234567890abcdef0,i-0598c7d356eba48d7"
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
```

```
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

専有ホストとイベントウィンドウとの関連付けを解除するには

`instance-event-window-id` パラメータを使用してイベントウィンドウを指定しながら、disassociate-instance-event-window コマンドを実行します。Dedicated Host の関連付けを解除するには、`--association-target` パラメータを使用し、その値に 1 つ以上の Dedicated Host ID を指定します。

```
aws ec2 disassociate-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --association-target DedicatedHostIds=h-029fa35a02b99801d
```

正常な出力

```
{
    "InstanceEventWindow": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "Name": "myEventWindowName",
        "CronExpression": "* 21-23 * * 2,3",
        "AssociationTarget": {
            "InstanceIds": [],
            "Tags": [],
            "DedicatedHostIds": []
        },
        "State": "creating"
    }
}
```

イベントウィンドウの削除

次のいずれかの方法を使用して、一度に 1 つのイベントウィンドウを削除できます。

Console

イベントウィンドウを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. [アクション]、[イベントウィンドウの管理] の順にクリックします。
4. 削除するイベントウィンドウを選択し、[アクション]、[インスタンスのイベントウィンドウの削除] の順にクリックします。
5. 確認を求めるメッセージが表示されたら、**delete**と入力し、[削除] を選択します。

AWS CLI

イベントウィンドウを削除するには

削除するイベントウィンドウを指定しながら、delete-instance-event-window コマンドを実行します。

```
aws ec2 delete-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890
```

イベントウィンドウを強制的に削除するには

現在、イベントウィンドウがターゲットに関連付けられている場合には、--force-delete パラメータを使用します。

```
aws ec2 delete-instance-event-window \
    --region us-east-1 \
    --instance-event-window-id iew-0abcdef1234567890 \
    --force-delete
```

正常な出力

```
{
    "InstanceEventWindowState": {
        "InstanceEventWindowId": "iew-0abcdef1234567890",
        "State": "deleting"
    }
```

```
}
```

イベントウィンドウのタグ付け

イベントウィンドウは、作成時またはその後にタグ付けすることができます。

作成時にイベントウィンドウのタグ付けを行うには、イベントウィンドウの作成を参照してください。

イベントウィンドウにタグ付けを行うには、次のいずれかの方法を使用します。

Console

既存のイベントウィンドウにタグ付けを行うには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Events] を選択します。
3. [アクション]、[イベントウィンドウの管理] の順にクリックします。
4. タグ付けするイベントウィンドウを選択し、[アクション]、[インスタンスのイベントウィンドウタグの管理] の順にクリックします。
5. [タグを追加] をクリックしタグを追加します。各タグについて、これを繰り返します。
6. [Save] を選択します。

AWS CLI

既存のイベントウィンドウにタグ付けを行うには

create-tags コマンドを使用して、既存のリソースにタグを付けます。以下の例では、既存のイベントウィンドウに、1 つのタグ (Key=purpose and Value=test) が付けられます。

```
aws ec2 create-tags \
    --resources iew-0abcdef1234567890 \
    --tags Key=purpose,Value=test
```

CloudWatch を使用したインスタンスのモニタリング

Amazon CloudWatch を使用してインスタンスをモニタリングすることで、Amazon EC2 から未加工データを収集し、リアルタイムに近い読み取り可能なメトリクスに加工することができます。これら

の統計は 15 か月間記録されるため、履歴情報にアクセスしてウェブアプリケーションやサービスの動作をより的確に把握できます。

デフォルトでは、Amazon EC2 は 5 分ごとにメトリクスデータを CloudWatch に送信します。1 分ごとにインスタンスのメトリクスデータを CloudWatch に送信するには、インスタンスで詳細モニタリングを有効にできます。詳細については、インスタンスの詳細モニタリングを有効または無効にするを参照してください。

Amazon EC2 コンソールには、Amazon CloudWatch の未加工データに基づいて一連のグラフが表示されます。必要に応じて、コンソールのグラフではなく Amazon CloudWatch からインスタンスのデータを取得することもできます。

Amazon CloudWatch の請求とコストに関する情報については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch の請求とコスト」を参照してください。

コンテンツ

Amazon EC2 インスタンスアラーム

インスタンスの Amazon CloudWatch アラームは、Amazon EC2 コンソールの [インスタンス] 画面で表示できます。

ListMetrics API コールのコスト

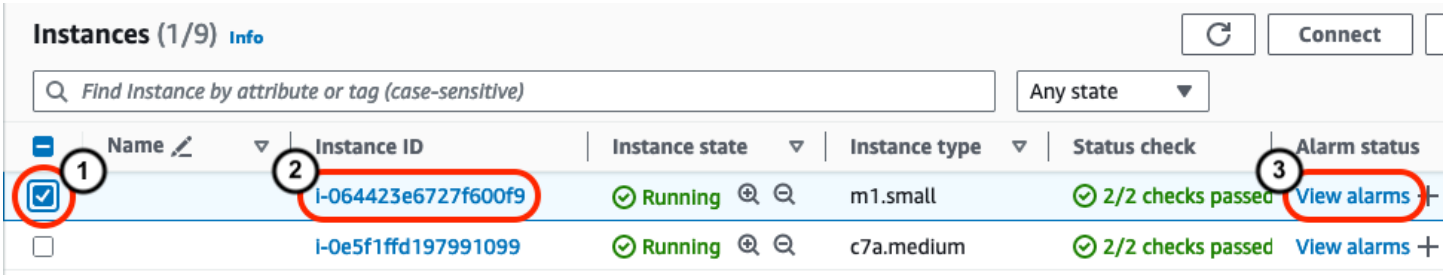
1,000 件の `ListMetrics` API リクエストごとに、AWS 無料利用枠 内にいるかどうかによって、0.01 USD のコストが発生する可能性があります。無料利用枠では、100 万件の無料の CloudWatch API リクエスト (常に課金される `GetMetricData`、`GetInsightRuleReport`、および `GetMetricWidgetImage` を除く) が得られます。詳細については、「Amazon CloudWatch 料金表」ページの「無料利用枠」を参照してください。

EC2 コンソールで次のアクションを実行すると、Amazon EC2 は CloudWatch `ListMetrics` API リクエストを行います。

- [インスタンス] テーブルのインスタンスのチェックボックスをオンにする (以下のスクリーンショットの 1 で示されています)。
- [インスタンス] テーブルで ID を選択してインスタンスを選択する (以下のスクリーンショットの 2 で示されています)。
- [インスタンス] テーブルで [アラームの表示] を選択して、[i-*1234567890example* のアラーム詳細] ウィンドウを開く (以下のスクリーンショットの 3 で示されています)。

> **Note**
>
> [アラームの表示] を選択すると、インスタンスのチェックボックス (以下のスクリーンショットの 1 で示されている) が自動的に選択され、別の `ListMetrics` API リクエストが生成されます。

次のスクリーンショットは、1、2、および 3 の番号が付いたコンソールコントロールが選択されると `ListMetrics` API を呼び出すことを示しています。



インスタンスの詳細モニタリングを有効または無効にする

デフォルトでは、インスタンスで基本モニタリングが有効になります。オプションで詳細モニタリングを有効にできます。

次のテーブルは、インスタンスの基本モニタリングと詳細モニタリングの違いを示しています。

モニタリングタイプ	説明	料金
基本モニターリング	ステータスチェックメトリクスに限り 1 分間隔で利用できます。	料金は発生しません。

モニタリングタイプ	説明	料金
	その他のメトリクスはすべて 5 分間隔で利用できます。	
詳細モニターリング	ステータスチェックメトリクスを含むすべてのメトリクスは、1 分間隔で利用できます。このレベルのデータを取得するには、インスタンスのデータ取得を明確に有効にする必要があります。詳細モニタリングを有効にしたインスタンスでは、同様のインスタンスグループの集約データを取得することもできます。	料金は、CloudWatch に送信されるメトリックスごとに発生します。データストレージに対しては料金が発生しません。詳細については、Amazon CloudWatch の料金ページの、有料利用枠および例 1 –EC2 の詳細モニタリングを参照してください。

トピック

- 必要な IAM アクセス許可
- 詳細モニタリングを有効にする
- 詳細モニタリングの無効化

必要な IAM アクセス許可

インスタンスの詳細モニタリングを有効にするには、ユーザーに MonitorInstances API アクションを使用するための許可が必要です。インスタンスの詳細モニタリングをオフにするには、ユーザーに UnmonitorInstances API アクションを使用するための許可が必要です。

詳細モニタリングを有効にする

インスタンスが実行または停止された後で、起動時にインスタンスの詳細モニタリングを有効にできます。インスタンスで詳細モニタリングを有効にしても、そのインスタンスに接続されている EBS ボリュームのモニタリングには影響しません。詳細については、「Amazon EBS の Amazon CloudWatch メトリクス」を参照してください。

Console

既存のインスタンスの詳細モニタリングを有効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Actions] (アクション)、[Monitor and troubleshoot] (モニタリングとトラブルシューティング)、[Manage detailed monitoring] (詳細モニタリングの管理) の順に選択します。
4. [Detailed monitoring (詳細モニタリング)] 詳細 ページの [Detailed monitoring (詳細モニタリング)] で、[Enable (有効)] チェックボックスをオンにします。
5. [Save] を選択します。

インスタンスの起動時に詳細モニタリングを有効にするには

Amazon EC2 コンソールを使用してインスタンスを起動する場合は、[高度な詳細] で、[CloudWatch モニタリングの詳細] チェックボックスを選択します。

AWS CLI

既存のインスタンスの詳細モニタリングを有効にするには

次の monitor-instances コマンドを使用して、指定したインスタンスの詳細モニタリングを有効にします。

```
aws ec2 monitor-instances --instance-ids i-1234567890abcdef0
```

インスタンスの起動時に詳細モニタリングを有効にするには

run-instances コマンドを `--monitoring` フラグとともに使用して詳細モニタリングを有効にします。

```
aws ec2 run-instances --image-id ami-09092360 --monitoring Enabled=true...
```

詳細モニタリングの無効化

インスタンスが実行または停止された後で、起動時にインスタンスの詳細モニタリングを無効にできます。

Console

詳細モニタリングを無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Actions] (アクション)、[Monitor and troubleshoot] (モニタリングとトラブルシューティング)、[Manage detailed monitoring] (詳細モニタリングの管理) の順に選択します。
4. [Detailed monitoring (詳細モニタリング)] 詳細 ページの [Detailed monitoring (詳細モニタリング)] で、[Enable (有効)] チェックボックスをオフにします。
5. [Save] を選択します。

AWS CLI

詳細モニタリングを無効にするには

次の unmonitor-instances コマンドを使用して、指定したインスタンスの詳細モニタリングを無効にします。

```
aws ec2 unmonitor-instances --instance-ids i-1234567890abcdef0
```

インスタンスの利用可能な CloudWatch メトリクスのリスト表示

Amazon EC2 はメトリクスを Amazon CloudWatch に送信します。AWS Management Console、AWS CLI、または API を使用して、Amazon EC2 が CloudWatch に送信するメトリクスを一覧表示できます。デフォルトで、各データポイントではインスタンスのアクティビティの開始後 5 分間が対象となります。詳細モニタリングを有効にした場合、各データポイントは開始後 1 分間のアクティビティを対象とします。注意事項[最小]、[最大]、[平均] の統計では、EC2 が提供するメトリックスの最小粒度は 1 分であることに注意します。

これらのメトリクスの統計の取得については、インスタンスのメトリクスの統計情報を取得するを参照してください。

コンテンツ

インスタンスメトリクス

AWS/EC2 名前空間には、次のインスタンスメトリクスが含まれます。

メトリクス	説明	単位	有意義な統計
CPUUtilization	Amazon EC2 が EC2 インスタンスを実行するために使用する物理 CPU 時間の割合。これには、ユーザーコードと Amazon EC2 コードの両方を実行するために費やされた時間が含まれます。 非常に高いレベルでは、CPUUtilization はゲスト CPUUtilization とハイパーバイザー CPUUtilization の合計です。 オペレーティングシステムのツールは CloudWatch と異なる割合を表示することがあります。これは、レガシーデバイスのシミュレーション、レガシーではないデバイスの設定、中断の多いワークロード、ライブ移行、ライブアップデートなどが原因です。	割合 (%)	• [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
DiskReadOps	指定された期間にインスタンスで利用できるすべてのインスタンスストアボリュームでの、完了した読み取り操作。 その期間の 1 秒あたりの I/O 操作回数 (IOPS) の平均を算出するには、その期間の操作回数の合計をその期間の秒数で割ります。 インスタンスストアボリュームがない場合は、値が 0 であるか、メトリクスがレポートされません。	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値
DiskWriteOps	指定された期間にインスタンスで利用できるすべてのインスタンスストアボリュームへの、完了した書き込み操作。 その期間の 1 秒あたりの I/O 操作回数 (IOPS) の平均を算出するには、その期間の操作回数の合計をその期間の秒数で割ります。 インスタンスストアボリュームがない場合は、値が 0 であるか、メトリクスがレポートされません。	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
DiskReadBytes	インスタンスで利用できるすべてのインスタンスストアボリュームから読み取られたバイト数。 このメトリクスを使用すると、このインスタンスのハードディスクからアプリケーションが読み取るデータの量がわかります。これを利用すると、アプリケーションの速度がわかります。 報告された数は、期間中に受信されたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算してバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で DiskReadBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。 インスタンスストアボリュームがない場合は、値が 0 であるか、メトリクスがレポートされません。	バイト	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
DiskWriteBytes	インスタンスで利用できるすべてのインスタンスストアボリュームに書き込まれたバイト数。 このメトリクスを使用すると、このインスタンスのハードディスクにアプリケーションが書き込むデータの量がわかります。これを利用すると、アプリケーションの速度がわかります。 報告された数は、期間中に受信されたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算してバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で DiskWriteBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。 インスタンスストアボリュームがない場合は、値が 0 であるか、メトリクスがレポートされません。	バイト	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
MetadataNoToken	トークンを使用しないメソッドを使用してインスタンスメタデータサービス (IMDS) に正常にアクセスした回数。 このメトリクスにより、トークンを使用しないインスタンスメタデータサービスバージョン 1 (IMDSv1) を使用してインスタンスメタデータにアクセスするプロセスがあるかどうかがわかります。すべてのリクエストがトークン支援のセッション (インスタンスメタデータサービスバージョン 2 (IMDSv2)) を使用している場合、値は 0 になります。詳細については、「インスタンスメタデータサービスバージョン 2 の使用への移行」を参照してください。	Count (カウント)	• 合計 • パーセンタイル
MetadataNoTokenRejected	IMDSv1 が無効になった後に IMDSv1 呼び出しが試行された回数。 このメトリクスが表示された場合は、IMDSv1 呼び出しが試行され、拒否されたことを示します。IMDSv1 を再度有効にするか、すべての呼び出しで IMDSv2 が使用されていることを確認します。詳細については、「インスタンスメタデータサービスバージョン 2 の使用への移行」を参照してください。	Count (カウント)	• 合計 • パーセンタイル

メトリクス	説明	単位	有意義な統計
NetworkIn	すべてのネットワークインターフェイスを通じ、このインスタンスによって受信されたバイトの数。このメトリクスは、1 つのインスタンスへの受信ネットワークトラフィックの量を表しています。 報告された数は、期間中に受信されたバイト数です。基本 (5 分) のモニタリングで統計情報に Sum 使用している場合であれば、この数を 300 で除算してバイト/秒の値を求めることができます。詳細 (1 分) のモニタリングで統計情報に Sum 使用している場合合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で NetworkIn のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	バイト	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
NetworkOut	すべてのネットワークインターフェイスを通じ、このインスタンスから送信されたバイトの数。このメトリクスは、1 つのインスタンスからの送信ネットワークトラフィックの量を表しています。 報告された数は、期間中に送信されたバイト数です。基本 (5 分) のモニタリングで統計情報に Sum 使用している場合であれば、この数を 300 で除算してバイト/秒の値を求めることができます。詳細 (1 分) のモニタリングで統計情報に Sum 使用している場合合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で NetworkOut のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	バイト	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
NetworkPacketsIn	すべてのネットワークインターフェイスを通じ、このインスタンスによって受信されたパケットの数。このメトリクスは、受信トラフィックのボリュームを単一インスタンスでのパケット数として識別します。 このメトリクスは、基本モニタリング (5分間) でのみ使用が可能です。5 分間にインスタンスが受信した 1 秒あたりのパケット数 (PPS) は、Sum統計値を 300 で割ることで算出されます。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのパケット数を求めることもできます。例えば、CloudWatch で NetworkPacketsIn のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをパケット/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
NetworkPackets Out	すべてのネットワークインターフェイスを通じ、このインスタンスから送信されたパケットの数。このメトリクスは、送信トラフィックのボリュームを単一インスタンスでのパケット数として識別します。 このメトリクスは、基本モニタリング (5分間) でのみ使用が可能です。5 分間にインスタンスが受信した 1 秒あたりのパケット数 (PPS) を計算するには、Sum 統計値を 300 で割ります。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのパケット数を求めることもできます。例えば、CloudWatch で NetworkPacketsOut のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをパケット/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値

CPU クレジットメトリクス

AWS/EC2 名前空間は、バーストパフォーマンスインスタンスの以下の CPU クレジットメトリクスを含みます。

メトリクス	説明	単位	有意義な統計
CPUCredit Usage	CPU 使用率に関してインスタンスで消費される CPU クレジットの数。1 つの CPU クレジットは、1 個の vCPU が 100% の使用率で 1 分間実行されること、または、vCPU、使用率、時間の同等の組み合わせ (例えば、1 個の	クレジット (vCPU 分)	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
	vCPU が 50% の使用率で 2 分間実行されるか、2 個の vCPU が 25% の使用率で 2 分間実行される) に相当します。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。5 分を超える期間を指定する場合は、Average 統計の代わりに Sum 統計を使用します。		

メトリクス	説明	単位	有意義な統計
CPUCredit Balance	インスタンスが起動または開始後に蓄積した獲得 CPU クレジットの数。T2 スタンダードの場合、CPUCreditBalance には蓄積された起動クレジットの数も含まれます。 クレジットは、獲得後にクレジット残高に蓄積され、消費されるとクレジット残高から削除されます。クレジット残高には、インスタンスサイズによって決まる上限があります。制限に到達すると、獲得された新しいクレジットはすべて破棄されます。T2 スタンダードの場合、起動クレジットは制限に対してカウントされません。 CPUCreditBalance のクレジットは、インスタンスがそのベースライン CPU 使用率を超えてバーストするために消費できます。 インスタンスが実行中の場合、CPUCredit Balance のクレジットは期限切れになりません。T3 または T3a インスタンスが停止すると、CPUCreditBalance 値は 7 日間保持されます。その後、蓄積されたすべてのクレジットが失われます。T2 インスタンスが停止すると、CPUCreditBalance 値は保持されず、蓄積されたすべてのクレジットが失われます。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。	クレジット (vCPU 分)	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
CPUSurplusCreditBalance	unlimited 値がゼロの場合に CPUCreditBalance インスタンスによって消費された余剰クレジットの数。 CPUSurplusCreditBalance 値は獲得した CPU クレジットによって支払われます。余剰クレジットの数が、24 時間にインスタンスが獲得できるクレジットの最大数を超えている場合、最大数を超えて消費された余剰クレジットに対しては料金が発生します。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。	クレジット (vCPU 分)	• 合計 • [Average] (平均) • 最小値 • 最大値
CPUSurplusCreditsCharged	獲得 CPU クレジットにより支払われないために追加料金が発生した、消費された余剰クレジットの数。 消費された余剰クレジットは、以下のいずれかの状況に当てはまると料金が発生します。 • 消費された余剰クレジットが、インスタンスが 24 時間に獲得できる最大クレジット数を超えている。最大数を越えて消費された余剰クレジットは、時間の最後に課金されます。 • インスタンスが停止または終了した。 • インスタンスは unlimited から standard に切り替わります。 CPU クレジットメトリクスは、5 分間隔でのみ利用可能です。	クレジット (vCPU 分)	• 合計 • [Average] (平均) • 最小値 • 最大値

Dedicated Hostsメトリクス

AWS/EC2 名前空間には、T3 Dedicated Hosts のための以下のメトリクスが含まれます。

メトリクス	説明	単位	有意義な統計
DedicatedHostCPUUtilization	Dedicated Host で実行されているインスタンスによって現在使用されている割り当て済みコンピューティング容量の割合。	割合 (%)	• 合計 • [Average] (平均) • 最小値 • 最大値

Nitro ベースのインスタンスの Amazon EBS メトリクス

AWS/EC2 名前空間には、ベアメタルインスタンスではない、Nitro ベースのインスタンスにアタッチされているボリュームに関する、追加の Amazon EBS メトリクスが含まれます。

メトリクス	説明	単位	有意義な統計
EBSReadOps	指定された期間にインスタンスに接続されたすべての Amazon EBS ボリュームからの、完了した読み込みオペレーション。 その期間の 1 秒あたりの読み込み I/O 操作回数 (読み込み IOPS) の平均を算出するには、その期間の操作回数の合計をその期間の秒数で割ります。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して読み込み IOPS を計算することができます。詳細 (1 分) モニタリングを使用している場合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのオペレーション数を求めることもできます。例えば、CloudWatch で EBSReadOps のグラフを m1 として作成した場合、メトリクスの計算関数 m1/(DIFF_TIME(m1)) はメトリクスをオペレーション/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
	ユーザーガイド」の「メトリクス数式の使用」を参照してください。		
EBSWriteOps	指定された期間にインスタンスに接続されたすべての EBS ボリュームからの、完了した書き込み操作。 その期間の 1 秒あたりの書き込み I/O 操作回数 (書き込み IOPS) の平均を算出するには、その期間の操作回数の合計をその期間の秒数で割ります。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して書き込み IOPS を計算することができます。詳細 (1 分) モニタリングを使用している場合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのオペレーション数を求めることもできます。例えば、CloudWatch で EBSWriteOps のグラフを m1 として作成した場合、メトリクスの計算関数 m1/(DIFF_TIME(m1)) はメトリクスをオペレーション/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
EBSReadBytes	指定した期間内にインスタンスに接続されたすべての EBS ボリュームから読み取られたバイト数。 報告された数は、期間中に読み取られたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して読み込みバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で EBSReadBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	バイト	• 合計 • [Average] (平均) • 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
EBSWriteBytes	指定した期間内にインスタンスに接続されたすべての EBS ボリュームに書き込まれたバイト数。 報告された数は、期間中に書き込まれたバイト数です。基本 (5 分) モニタリングを使用している場合、この数を 300 で除算して書き込みバイト/秒を求めることができます。詳細 (1 分) モニタリングを使用している場合は、この数を 60 で除算します。CloudWatch メトリクスの計算関数 DIFF_TIME を使用して、1 秒あたりのバイト数を求めることもできます。例えば、CloudWatch で EBSWriteBytes のグラフを m1 として作成した場合、メトリクスの数式 m1/(DIFF_TIME(m1)) はメトリクスをバイト/秒単位で返します。DIFF_TIME およびメトリクス計算関数の詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクス数式の使用」を参照してください。	バイト	• 合計 • [Average] (平均) • 最小値 • 最大値
EBSIOBalance%	バーストバケットの I/O 残りクレジットの割合に関する情報を提供します。このメトリクスは基本モニタリング専用です。 このメトリクスは、少なくとも 24 時間に 1 回、30 分間だけ最大パフォーマンスにバーストする一部の *.4xlarge インスタンスサイズ以下でのみ使用できます。 Sum 統計は、このメトリクスに該当しません。	割合 (%)	• 最小値 • 最大値

メトリクス	説明	単位	有意義な統計
EBSByteBalance%	バーストバケットのスループット残りクレジットの割合に関する情報を提供します。このメトリクスは基本モニタリング専用です。 このメトリクスは、少なくとも 24 時間に 1 回、30 分間だけ最大パフォーマンスにバーストする一部の *.4xlarge インスタンスサイズ以下でのみ使用できます。 Sum 統計は、このメトリクスに該当しません。	割合 (%)	• 最小値 • 最大値

EBS ボリューム用のメトリクスの詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS ボリュームのメトリクス」を参照してください。スポットフリートに提供されるメトリクスの詳細については、スポットフリートの CloudWatch メトリクスを参照してください。

ステータスチェックメトリクス

デフォルトでは、ステータスチェックメトリクスは無料で 1 分の頻度で利用できます。新しく起動したインスタンスの場合、ステータスチェックメトリクスデータは、インスタンスが初期化状態を完了するまで使用できません (インスタンスが running の状態になってから数分以内)。EC2 ステータスチェックの詳細については、インスタンスのステータスチェックを参照してください。

AWS/EC2 名前空間には、次のステータスチェックメトリクスが含まれます。

メトリクス	説明	単位	有意義な統計
StatusCheckFailed	インスタンスが過去 1 分間にインスタンスのステータスチェックとシステムステータスチェックの両方に合格したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトでは、このメトリクスは無料で 1 分の頻度で利用できます。	Count (カウント)	• 合計 • [Average] (平均)

メトリクス	説明	単位	有意義な統計
StatusCheckFailed_Instance	最近 1 分間にインスタンスがインスタンスステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトでは、このメトリクスは無料で 1 分の頻度で利用できます。	Count (カウント)	• 合計 • [Average] (平均)
StatusCheckFailed_System	最近 1 分間にインスタンスが システムステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトでは、このメトリクスは無料で 1 分の頻度で利用できます。	Count (カウント)	• 合計 • [Average] (平均)
StatusCheckFailed_AttachedEBS	直近 1 分間でインスタンスがアタッチ済みの EBS ステータスチェックに成功したかどうかを報告します。 このメトリクスは 0 (合格) または 1 (失敗) となります。 デフォルトでは、このメトリクスは無料で 1 分の頻度で利用できます。	Count (カウント)	• 合計 • [Average] (平均)

AWS/EBS 名前空間には、次のステータスチェックメトリクスが含まれます。

メトリクス	説明	単位	有意義な統計
VolumeStalledIOCheck	注: Nitro インスタンスのみが対象です。Amazon ECS と AWS Fargate タスクにアタッチされたボリュームについては公開されていません。 ボリュームが過去 1 分間のストールした IO のチェックに合格したか失敗したかを報告します。このメトリクスは 0 (合格) または 1 (失敗) となります。	Count (カウント)	• 合計 • [Average] (平均) • 最小値 • 最大値

トラフィックミラーリングのメトリクス

AWS/EC2 名前空間には、ミラートラフィックのメトリクスが含まれます。詳細については、「Amazon VPC トラフィックミラーリングガイド」の「Monitor mirrored traffic using Amazon CloudWatch」(Amazon CloudWatch によるミラーリングされたトラフィックのモニタリング) を参照してください。

Auto Scaling グループメトリクス

AWS/AutoScaling 名前空間には、Auto Scaling グループのメトリクスが含まれます。詳細については、「Amazon EC2 Auto Scaling ユーザーガイド」の「Monitor CloudWatch metrics for your Auto Scaling groups and instances (Auto Scaling グループとインスタンスの CloudWatch メトリクスのモニタリング)」ご参照してください。

Amazon EC2 メトリクスディメンション

以下のディメンションを使用して、前の表に示したメトリクスを絞り込むことができます。

ディメンション	説明
AutoScalingGroupName	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定したキャパシティーグループ内のインスタンスのものだけになります。Auto Scaling グループは、Auto Scaling を使用する場合に定義するインスタンスのコレクションです。このディメンションを Amazon EC2 のメトリクスに対して使用できるのは、インスタンスが Auto Scaling

ディメンション	説明
	グループ内にあるときに限られます。詳細モニタリングまたは基本モニタリングが有効になっているインスタンスに対して使用できます。
ImageId	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、この Amazon EC2 Amazon Machine Image (AMI) を実行しているインスタンスのものだけになります。詳細モニタリングが有効になっているインスタンスに対して使用できます。
InstanceId	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定のインスタンスのものだけになります。これを利用すると、どのインスタンスからのデータをモニタリングするかを指定できます。
InstanceType	このディメンションを指定すると、リクエストしたデータがフィルタリングされて、指定のインスタンスタイプで実行されているインスタンスのものだけになります。これを利用すると、実行されているインスタンスのタイプでデータを分類することができます。例えば、m1.small インスタンスと m1.large インスタンスのデータを比較して、アプリケーションに対するビジネス価値はどちらが上かを判断します。詳細モニタリングが有効になっているインスタンスに対して使用できます。

Amazon EC2 使用状況メトリクス

CloudWatch 使用状況メトリクスを使用して、アカウントのリソースの使用状況を把握できます。これらのメトリクスを使用して、CloudWatch グラフやダッシュボードで現在のサービスの使用状況を可視化できます。

Amazon EC2 使用状況メトリクスは、AWS のサービスクォータに対応しています。使用量がサービスクォータに近づいたときに警告するアラームを設定することもできます。CloudWatch とサービスクオータの統合については、「Amazon CloudWatch ユーザーガイド」の「AWS 使用状況メトリクス」を参照してください。

Amazon EC2 は、AWS/Usage 名前空間に以下のメトリクスを公開します。

メトリクス	説明
ResourceCount	アカウントで実行されている指定されたリソースの数。リソースは、メトリクスに関連付けられたディメンションによって定義されます。 このメトリクスで最も役に立つ統計は MAXIMUM です。これは、1 分間の期間中に使用されるリソースの最大数を表します。

次のディメンションは、Amazon EC2 によって発行される使用状況メトリクスを絞り込むために使用されます。

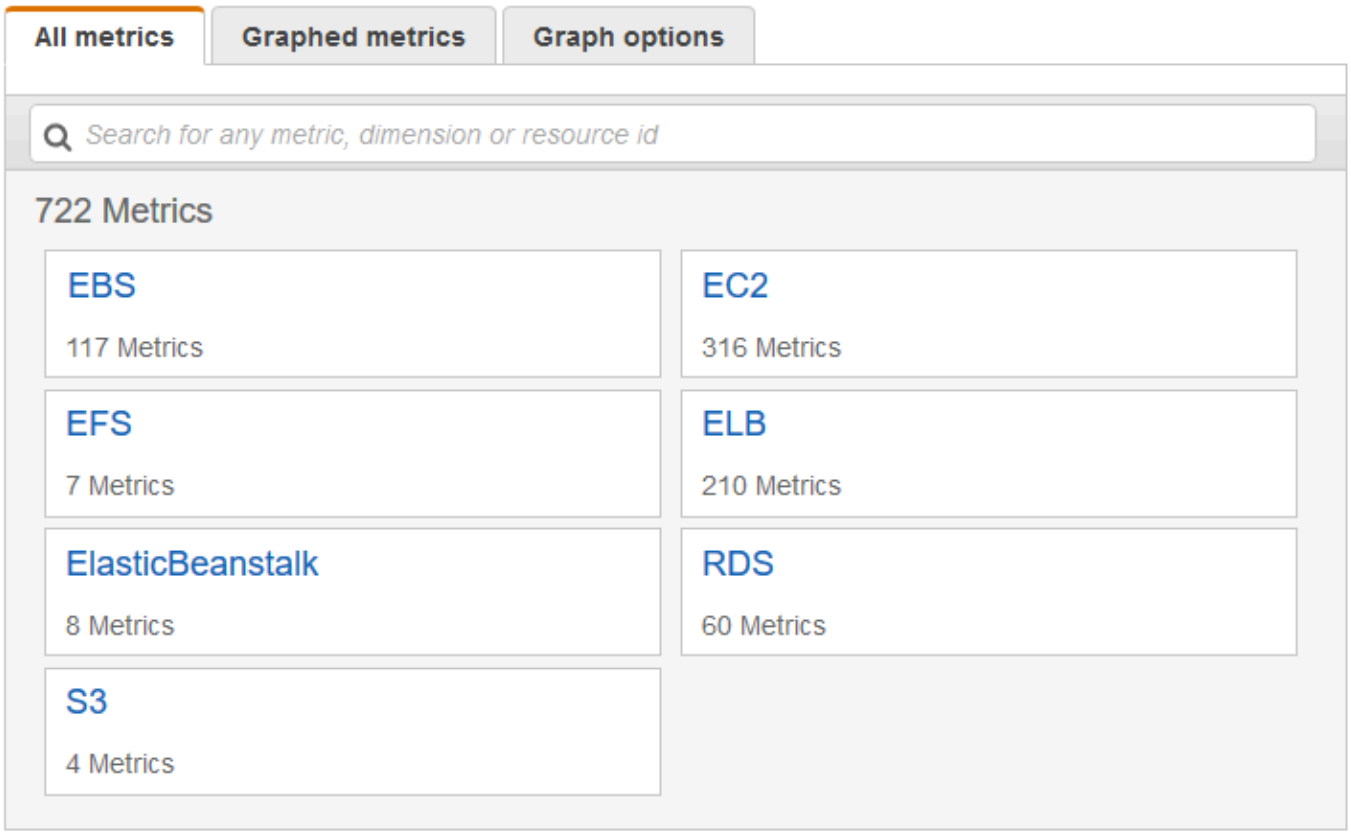
ディメンション	説明
Service	リソースを含む AWS のサービスの名前。Amazon EC2 使用状況メトリクスの場合、このディメンションの値は EC2 です。
Type	レポートされるエンティティのタイプ。現在、Amazon EC2 使用状況メトリクスの有効な値は Resource のみです。
Resource	実行中のリソースのタイプ。現在、Amazon EC2 使用状況メトリクスの有効な値は vCPU のみです。これは、実行中のインスタンスに関する情報を返します。
Class	追跡されるリソースのクラス。vCPU ディメンションの値として Resource を使用する Amazon EC2 使用状況メトリクスの場合、有効な値は、Standard/OnDemand 、F/OnDemand 、G/OnDemand 、Inf/OnDemand 、P/OnDemand 、および X/OnDemand です。 このディメンションの値は、メトリクスによって報告されるインスタンスタイプの最初の文字を定義します。例えば、Standard/OnDemand は、A、C、D、H、I、M、R、T、Z で始まるタイプのすべての実行中のインスタンスに関する情報を返し、G/OnDemand は G で始まるタイプのすべてのインスタンスに関する情報を返します。

コンソールを使用したメトリクスの一覧表示

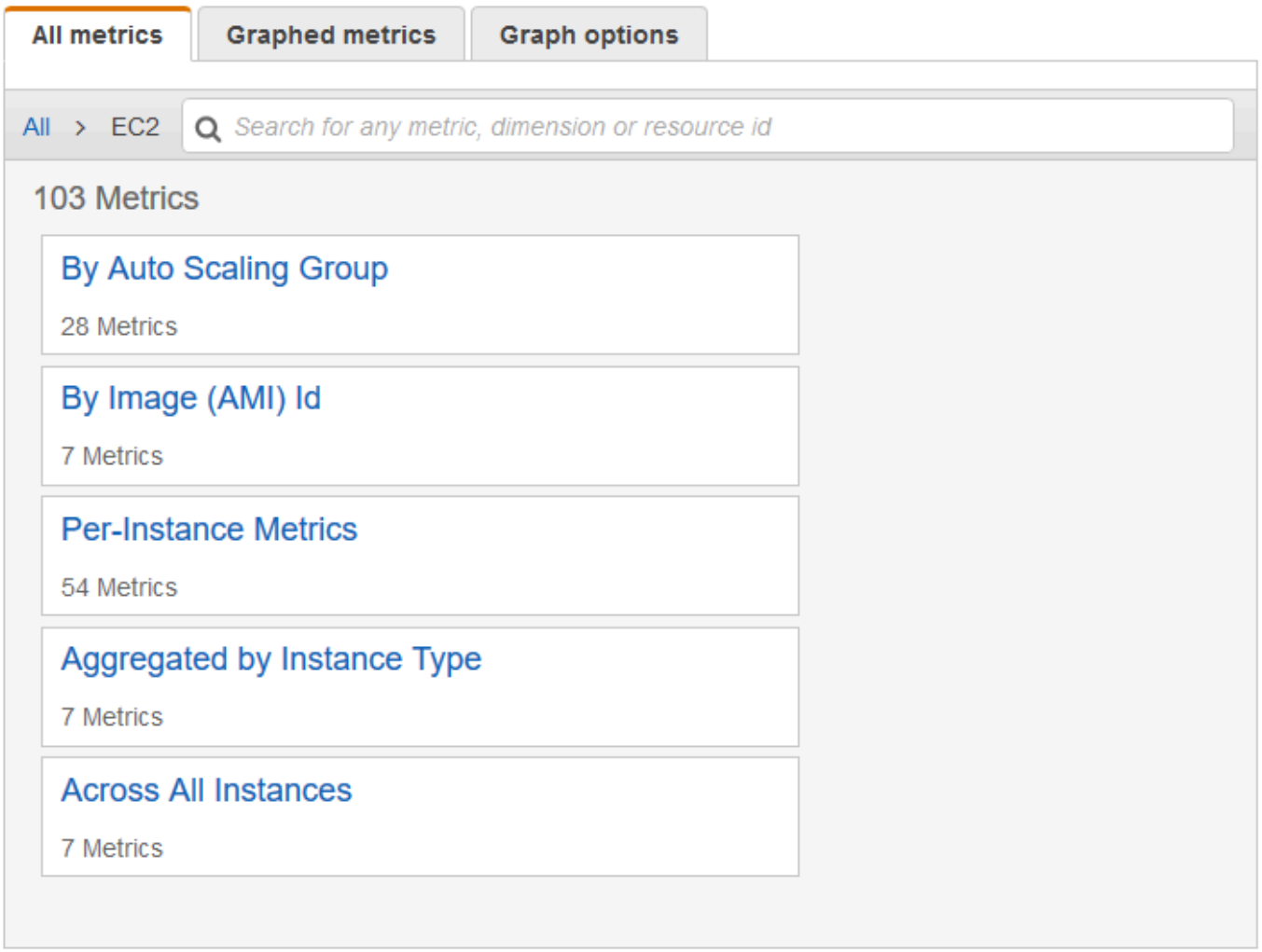
メトリクスはまず名前空間ごとにグループ化され、次に各名前空間内の種々のディメンションの組み合わせごとにグループ化されます。例えば、Amazon EC2 で提供されるすべてのメトリクスを表示させることもできれば、インスタンス ID、インスタンスタイプ、イメージ (AMI) ID、Auto Scaling グループでグループ化された EC2 メトリクスを表示することもできます。

利用可能なメトリクスをカテゴリー別に表示するには (コンソール)

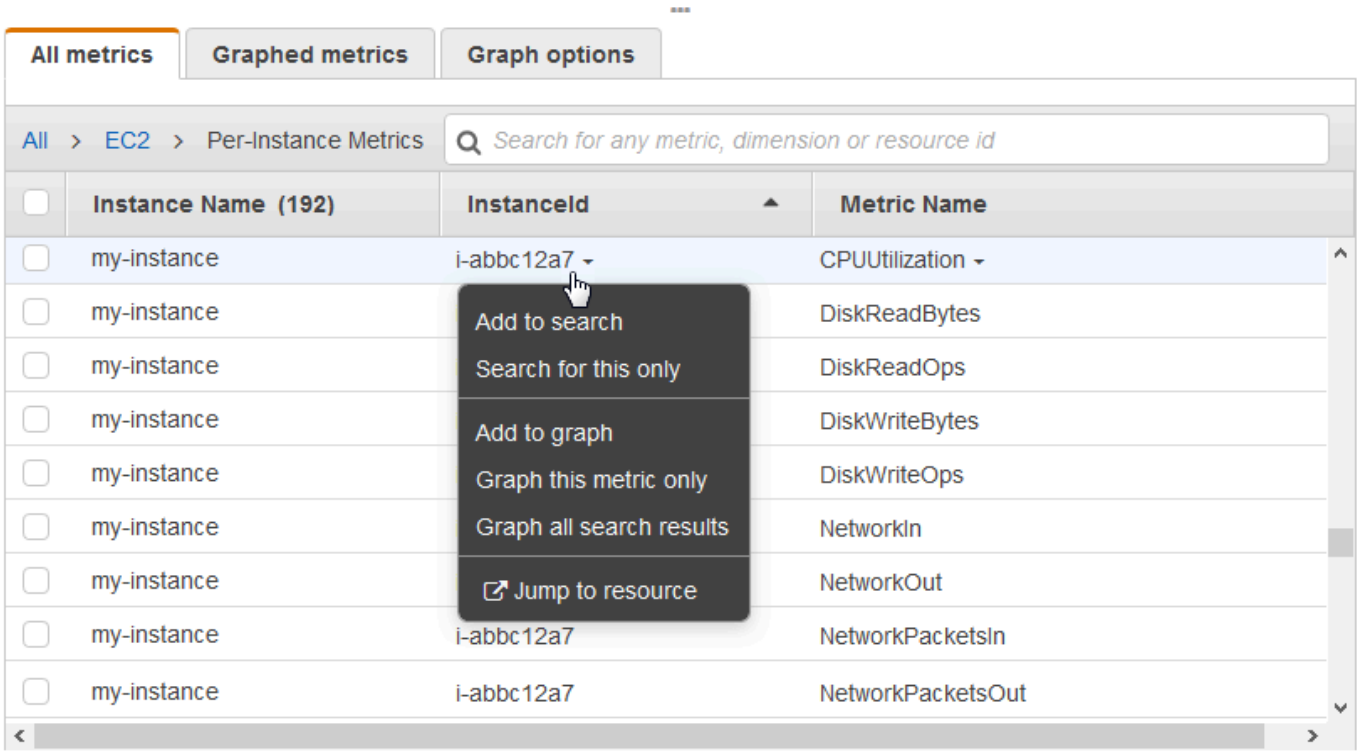
1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
3. EC2 のメトリクスの名前空間を選択します。



4. メトリクスのディメンション (例えば、インスタンス別メトリクス) を選択します。



5. メトリクスを並べ替えるには、列見出しを使用します。メトリクスをグラフ表示するには、メトリクスの横にあるチェックボックスを選択します。リソースでフィルタするには、リソース ID を選択し、[Add to search] を選択します。メトリクスでフィルタするには、メトリクスの名前を選択し、[Add to search] を選択します。



AWS CLI を使用したメトリクスの一覧表示

list-metrics コマンドを使用して、インスタンスの CloudWatch メトリクスをリスト表示します。

Amazon EC2 の利用可能なすべてのメトリクスを一覧表示するには (AWS CLI)

次の例では、Amazon EC2 のすべてのメトリクスを表示する目的で AWS/EC2 名前空間を指定します。

```
aws cloudwatch list-metrics --namespace AWS/EC2
```

出力例を次に示します。

```
{
  "Metrics": [
    {
        "Namespace": "AWS/EC2",
        "Dimensions": [
            {
                "Name": "InstanceId",
                "Value": "i-1234567890abcdef0"
```

```
            }
        ],
        "MetricName": "NetworkOut"
    },
    {
        "Namespace": "AWS/EC2",
        "Dimensions": [
            {
                "Name": "InstanceId",
                "Value": "i-1234567890abcdef0"
            }
        ],
        "MetricName": "CPUUtilization"
    },
    {
        "Namespace": "AWS/EC2",
        "Dimensions": [
            {
                "Name": "InstanceId",
                "Value": "i-1234567890abcdef0"
            }
        ],
        "MetricName": "NetworkIn"
    },
    ...
  ]
}
```

インスタンスで利用可能なすべてのメトリクスをリスト表示するには (AWS CLI)

次の例では、指定のインスタンスの結果だけを表示する目的で AWS/EC2 名前空間と InstanceId ディメンションを指定します。

```
aws cloudwatch list-metrics --namespace AWS/EC2 --dimensions
 Name=InstanceId,Value=i-1234567890abcdef0
```

すべてのインスタンス間でメトリクスをリスト表示するには (AWS CLI)

次の例では、指定のメトリクスの結果だけを表示する目的で AWS/EC2 名前空間とメトリクス名を指定します。

```
aws cloudwatch list-metrics --namespace AWS/EC2 --metric-name CPUUtilization
```

インスタンスのメトリクスの統計情報を取得する

インスタンスの CloudWatch メトリクスの統計情報を取得できます。

コンテンツ

統計の概要

統計とは、メトリクスデータを指定した期間で集計したものです。CloudWatch は、カスタムデータまたは AWS の他のサービスから CloudWatch に提供された、メトリクスデータポイントに基づく統計を提供します。集約は、指定した期間内に、名前空間、メトリクス名、ディメンション、データポイントの測定単位を用いて行われます。次の表は利用可能な統計を説明しています。

統計	説明
Minimum	指定された期間に認められた最小値です。この値を用いて、アプリケーションの低ボリュームのアクティビティを判断できます。
Maximum	指定された期間に認められた最大値です。この値を用いて、アプリケーションの高ボリュームのアクティビティを判断できます。
Sum	該当するメトリクスで加算されたすべての合計値です。この統計は、メトリクスの合計ボリュームを判断するのに役立ちます。
Average	指定した期間の Sum/SampleCount の値です。この統計を Minimum および Maximum と比較することで、メトリクスの全容、および平均使用量がどれくらい Minimum と Maximum に近いかを判断できます。この比較は、必要に応じていつリソースを増減させるべきかを知るのに役立ちます。
SampleCount	統計計算で使用するデータポイントのカウント (数) です。

統計	説明
pNN.NN	指定されたパーセンタイルの値。小数点以下最大 2 桁を使用して、任意のパーセンタイルを指定できます (p95.45 など)。

特定のインスタンスの統計を取得する

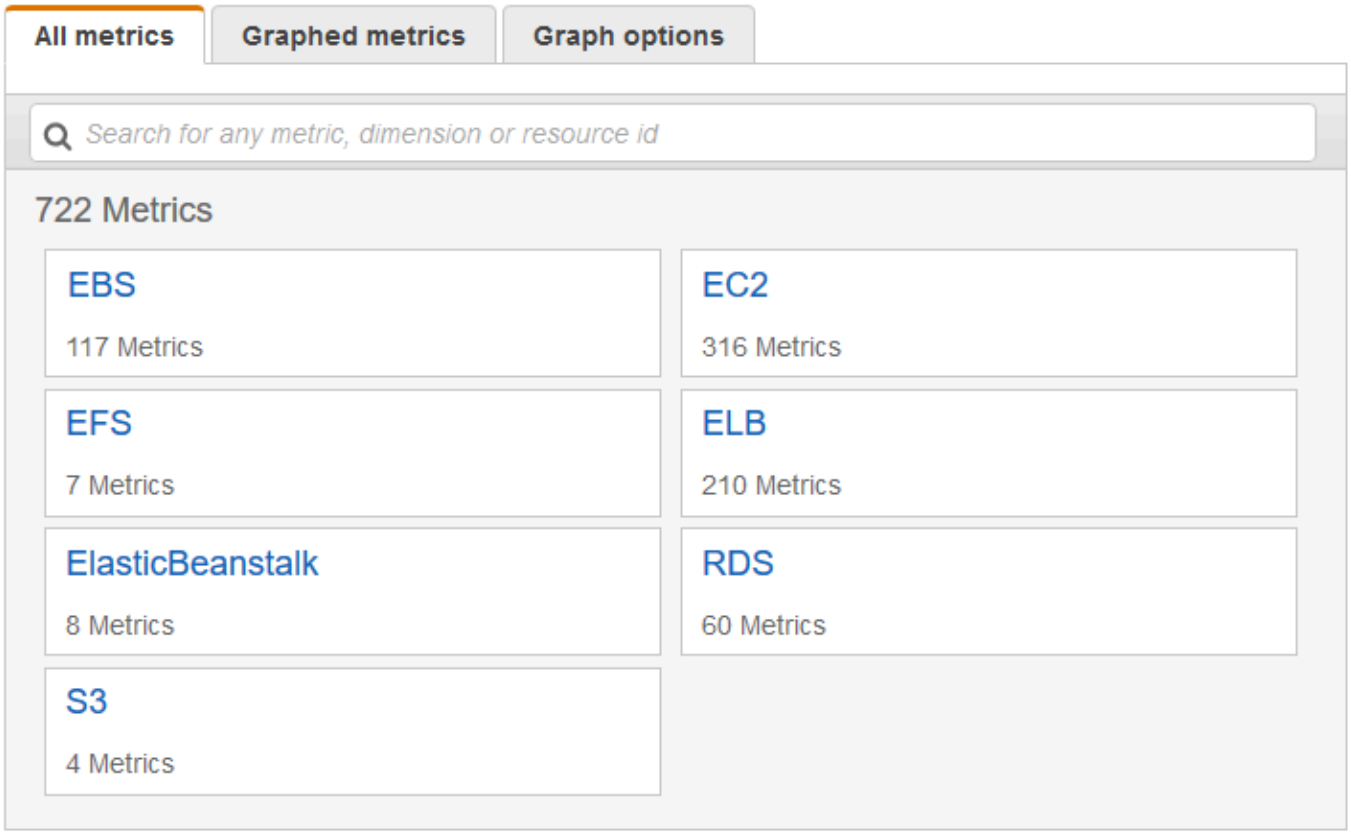
次の例では、AWS Management Console または AWS CLI を使用して、特定の EC2 インスタンスの最大 CPU 使用率を決定することができます。

要件

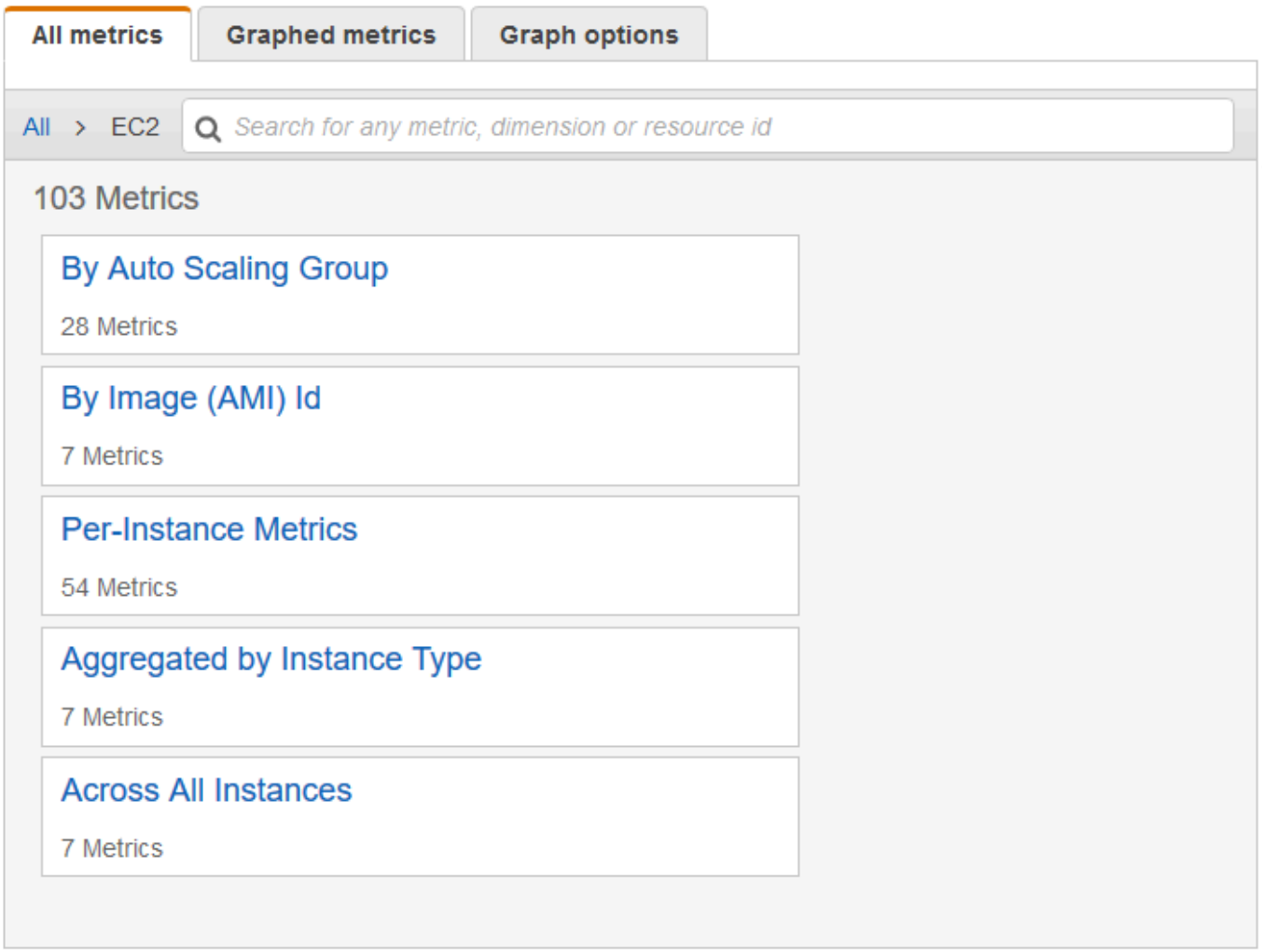
- インスタンスの ID が必要です。インスタンス ID は、AWS Management Console コンソールまたは describe-instances コマンドを使って取得します。
- デフォルトでは、基本モニタリングが有効化されていますが、詳細モニタリングを有効化することもできます。詳細については、インスタンスの詳細モニタリングを有効または無効にするを参照してください。

特定のインスタンスの CPU 使用率を表示するには (コンソール)

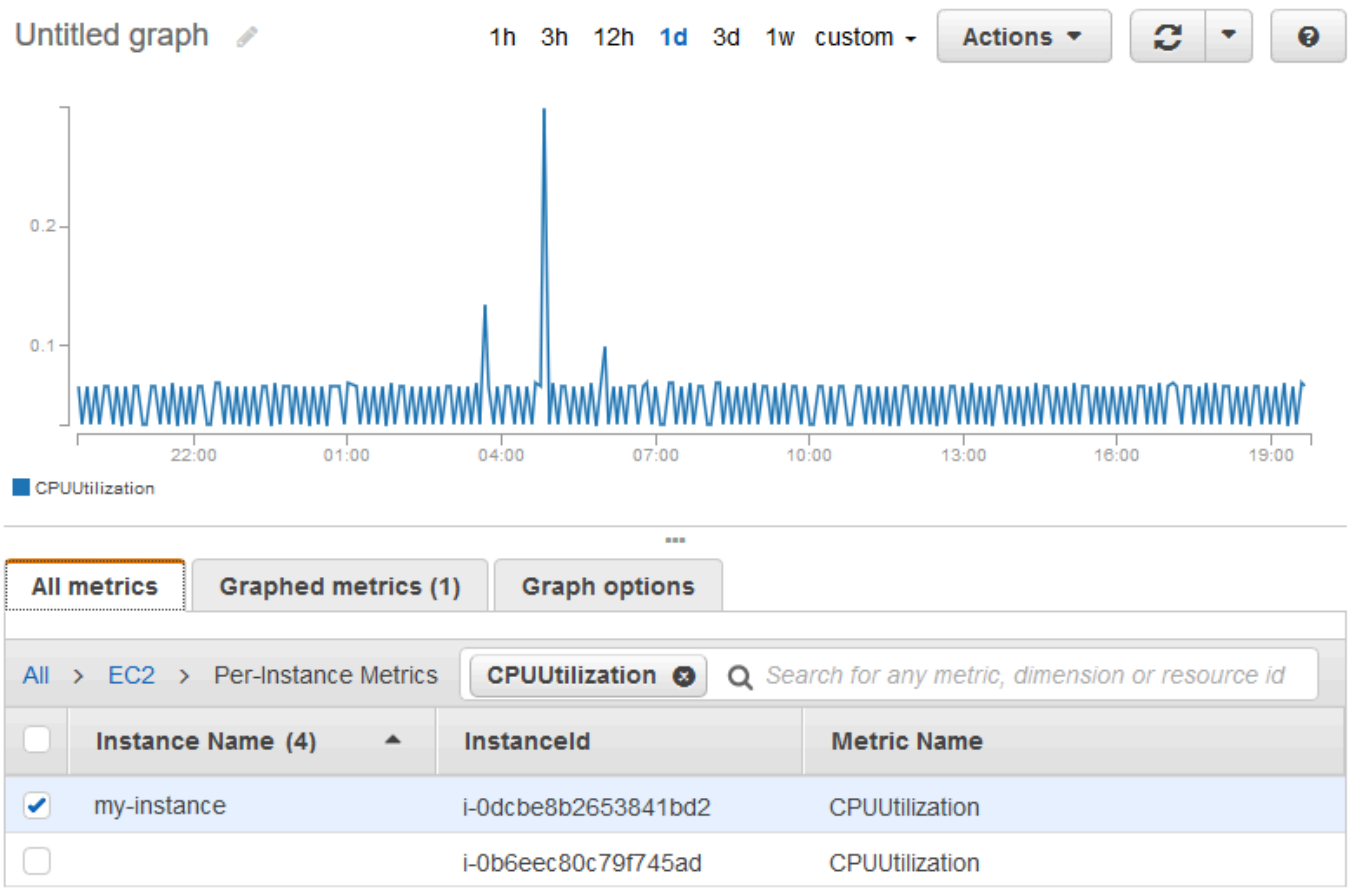
1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
3. EC2 のメトリクスの名前空間を選択します。



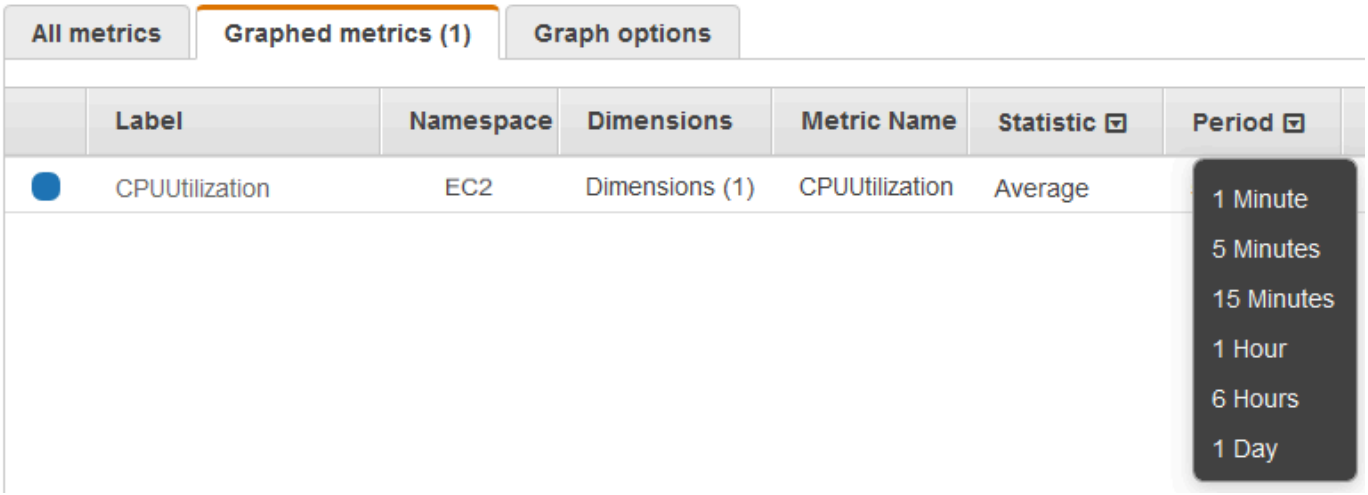
4. インスタンス別メトリクスのディメンションを選択します。



5. 検索フィールドに **`CPUUtilization`** と入力して Enter キーを押します。特定のインスタンスの行を選択します。すると、そのインスタンスの [CPUUtilization] メトリクスのグラフが表示されます。グラフに名前を付けるには、鉛筆アイコンを選択します。時間範囲を変更するには、事前定義済みの値を選択するか、[custom] を選択します。



6. メトリクスの統計または期間を変更するには、[Graphed metrics] タブを選択します。列見出しまたは個々の値を選択し、次に異なる値を選択します。



特定のインスタンスの CPU 使用率を取得するには (AWS CLI)

次の get-metric-statistics コマンドを使用すると、期間と時間間隔を指定して、特定のインスタンスの [CPUUtilization] メトリクスを取得できます。

```
aws cloudwatch get-metric-statistics --namespace AWS/EC2 --metric-name CPUUtilization
 --period 3600 \
--statistics Maximum --dimensions Name=InstanceId,Value=i-1234567890abcdef0 \
--start-time 2022-10-18T23:18:00 --end-time 2022-10-19T23:18:00
```

出力例を次に示します。それぞれの値は、単一の EC2 インスタンスの最大 CPU 使用率を表しています。

```
{
    "Datapoints": [
        {
            "Timestamp": "2022-10-19T00:18:00Z",
            "Maximum": 0.33000000000000002,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-19T03:18:00Z",
            "Maximum": 99.670000000000002,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-19T07:18:00Z",
            "Maximum": 0.34000000000000002,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-19T12:18:00Z",
            "Maximum": 0.34000000000000002,
            "Unit": "Percent"
        },
        ...
    ],
    "Label": "CPUUtilization"
}
```

インスタンス全体の統計の集約

集約された統計は、詳細モニタリングが有効になっているインスタンスで利用が可能です。基本モニタリングを使用するインスタンスは集約されません。インスタンス全体から集約された統計情報を取

得するには、詳細モニタリングを事前に有効化し、1 分間隔でデータが提供されるようにしておく必要があります (追加料金がかかります)。

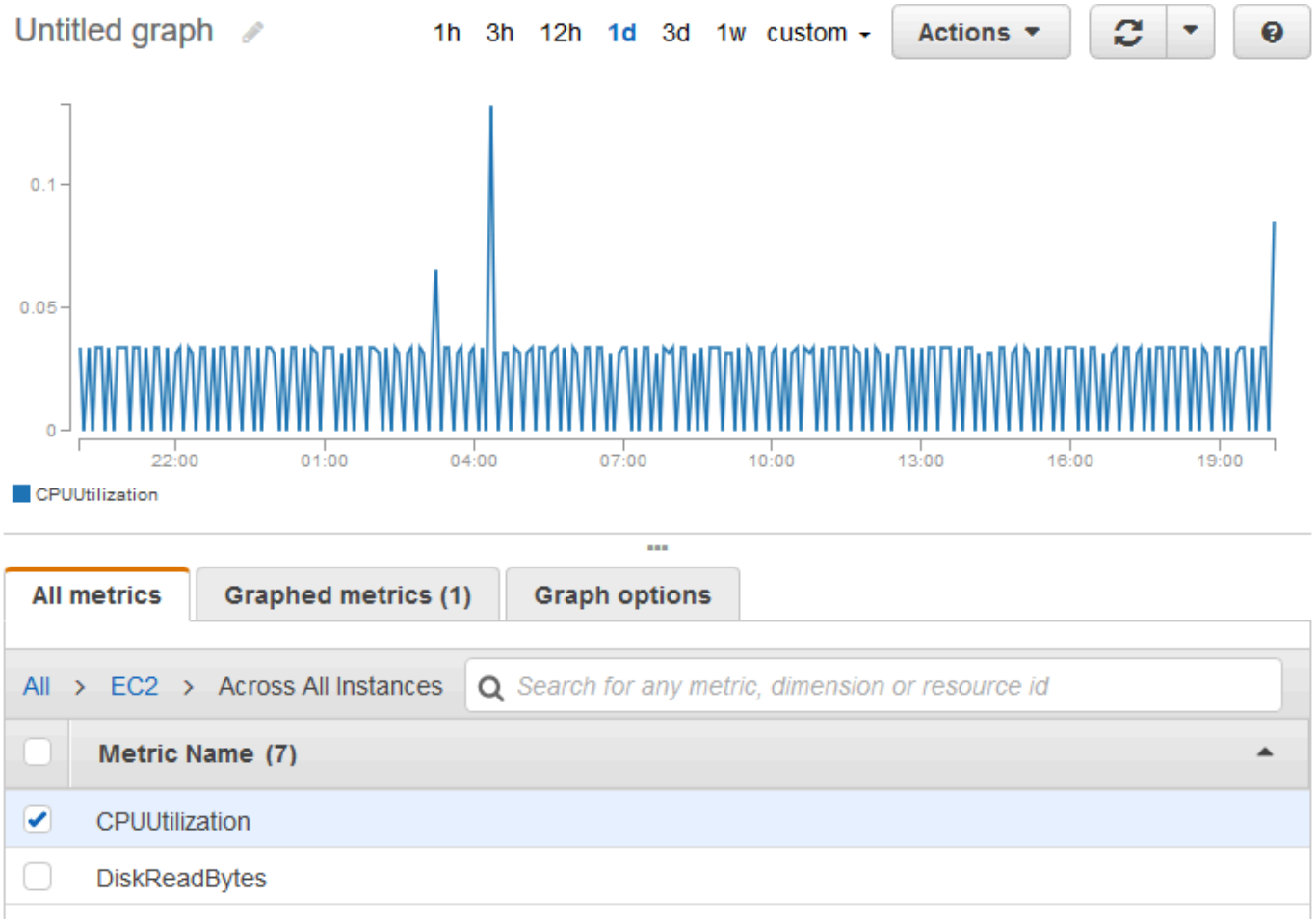
Amazon CloudWatch は、AWS リージョンをまたがってデータを集約することはできません。メトリクスは、リージョン間で完全に独立しています。

この例では、EC2 インスタンスの平均 CPU 使用率を取得するために詳細モニタリングを使用する方法について示します。ディメンションを指定していないため、CloudWatch は、AWS/EC2 名前空間にある全ディメンションの統計を返します。

> **Important**
>
> AWS 名前空間にあるすべてのディメンションを取得するこの手法は、Amazon CloudWatch に発行するカスタム名前空間では機能しません。カスタム名前空間の場合、データポイントを含む統計を取得するには、そのデータポイントに関連付けられたディメンション一式をすべて指定する必要があります。

インスタンスの平均 CPU 使用率を表示するには (コンソール)

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
3. [EC2] 名前空間を選択して、[Across All Instances] を選択します。
4. [CPUUtilization] を含む行を選択します。すべての EC2 インスタンスのメトリクスがグラフとして表示されます。グラフに名前を付けるには、鉛筆アイコンを選択します。時間範囲を変更するには、事前定義済みの値を選択するか、[custom] を選択します。



5. メトリクスの統計または期間を変更するには、[Graphed metrics] タブを選択します。列見出しまたは個々の値を選択し、次に異なる値を選択します。

複数のインスタンスの平均 CPU 使用率を取得するには (AWS CLI)

次のように get-metric-statistics コマンドを使用し、インスタンス全体の平均 [CPUUtilization] メトリクスを取得します。

```
aws cloudwatch get-metric-statistics \
    --namespace AWS/EC2 \
    --metric-name CPUUtilization \
    --period 3600  --statistics "Average" "SampleCount" \
    --start-time 2022-10-11T23:18:00 \
    --end-time 2022-10-12T23:18:00
```

出力例を次に示します。

```
{
    "Datapoints": [
        {
            "SampleCount": 238.0,
            "Timestamp": "2022-10-12T07:18:00Z",
            "Average": 0.038235294117647062,
            "Unit": "Percent"
        },
        {
            "SampleCount": 240.0,
            "Timestamp": "2022-10-12T09:18:00Z",
            "Average": 0.16670833333333332,
            "Unit": "Percent"
        },
        {
            "SampleCount": 238.0,
            "Timestamp": "2022-10-11T23:18:00Z",
            "Average": 0.041596638655462197,
            "Unit": "Percent"
        },
        ...
    ],
    "Label": "CPUUtilization"
}
```


Auto Scaling グループ別に統計を集計する

Auto Scaling グループ内で EC2 インスタンスの統計を集計することができます。Amazon CloudWatch は、AWS リージョンをまたがってデータを集約することはできません。メトリクスは、リージョン間で完全に独立しています。

この例では、1 つの Auto Scaling グループについて、ディスクに書き込まれた総バイト数を取得する方法を説明します。合計は、指定された Auto Scaling グループのすべての EC2 インスタンスで、24 時間おきに 1 分間に対して算出されます。

Auto Scaling グループ内のインスタンスの DiskWriteBytes を表示するには (コンソール)

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
3. [EC2] 名前空間を選択し、次に [By Auto Scaling Group] を選択します。

4. [DiskWriteBytes] メトリクスの行と特定の Auto Scaling グループを選択します。すると、その Auto Scaling グループ内にあるインスタンスのメトリクスがグラフとして表示されます。グラフに名前を付けるには、鉛筆アイコンを選択します。時間範囲を変更するには、事前定義済みの値を選択するか、[custom] を選択します。
5. メトリクスの統計または期間を変更するには、[Graphed metrics] タブを選択します。列見出しまたは個々の値を選択し、次に異なる値を選択します。

Auto Scaling グループのインスタンスの DiskWriteBytes を表示するには (AWS CLI)

以下のように get-metric-statistics コマンドを使用します。

```
aws cloudwatch get-metric-statistics --namespace AWS/EC2 --metric-name DiskWriteBytes
 --period 360 \
--statistics "Sum" "SampleCount" --dimensions Name=AutoScalingGroupName,Value=my-asg --
start-time 2022-10-16T23:18:00 --end-time 2022-10-18T23:18:00
```

出力例を次に示します。

```
{
    "Datapoints": [
        {
            "SampleCount": 18.0,
            "Timestamp": "2022-10-19T21:36:00Z",
            "Sum": 0.0,
            "Unit": "Bytes"
        },
        {
            "SampleCount": 5.0,
            "Timestamp": "2022-10-19T21:42:00Z",
            "Sum": 0.0,
            "Unit": "Bytes"
        }
    ],
    "Label": "DiskWriteBytes"
}
```

AMI 別に統計を集計する

統計の集計は、詳細モニタリングが有効化されているインスタンスに対して行うことができます。基本モニタリングを使用するインスタンスは集約されません。インスタンス全体から集約された統計情

報を取得するには、詳細モニタリングを事前に有効化し、1 分間隔でデータが提供されるようにしておく必要があります (追加料金がかかります)。

Amazon CloudWatch は、AWS リージョンをまたがってデータを集約することはできません。メトリクスは、リージョン間で完全に独立しています。

この例では、特定の Amazon Machine Image (AMI) を使用するすべてのインスタンスの平均 CPU 使用率を特定する方法を説明します。平均値は、1 日間、60 秒間隔の平均値です。

AMI 別の平均 CPU 使用率を表示するには (コンソール)

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Metrics (メトリクス)] を選択します。
3. [EC2] 名前空間を選択し、次に [By Image (AMI) Id] を選択します。
4. [CPUUtilization] メトリクスの行と特定の AMI を選択します。すると、その AMI のメトリクスがグラフとして表示されます。グラフに名前を付けるには、鉛筆アイコンを選択します。時間範囲を変更するには、事前定義済みの値を選択するか、[custom] を選択します。
5. メトリクスの統計または期間を変更するには、[Graphed metrics] タブを選択します。列見出しまたは個々の値を選択し、次に異なる値を選択します。

特定のイメージ ID の平均 CPU 使用率を取得するには (AWS CLI)

以下のように get-metric-statistics コマンドを使用します。

```
aws cloudwatch get-metric-statistics --namespace AWS/EC2 --metric-name CPUUtilization
 --period 3600 \
--statistics Average --dimensions Name=ImageId,Value=ami-3c47a355 --start-
time 2022-10-10T00:00:00 --end-time 2022-10-11T00:00:00
```

出力例を次に示します。それぞれの値は、指定した AMI を実行する EC2 インスタンスの平均 CPU 使用率 (%) を表します。

```
{
    "Datapoints": [
        {
            "Timestamp": "2022-10-10T07:00:00Z",
            "Average": 0.041000000000000009,
            "Unit": "Percent"
```

```
        },
        {
            "Timestamp": "2022-10-10T14:00:00Z",
            "Average": 0.079579831932773085,
            "Unit": "Percent"
        },
        {
            "Timestamp": "2022-10-10T06:00:00Z",
            "Average": 0.036000000000000011,
            "Unit": "Percent"
        },
        ...
    ],
    "Label": "CPUUtilization"
}
```

インスタンスのグラフメトリクス

インスタンスを起動した後は、Amazon EC2 コンソールの [モニタリング] タブを開いて、インスタンスのモニタリンググラフを表示できます。各グラフは、利用可能な Amazon EC2 メトリクスのいずれかに基づいています。

以下のグラフが利用可能です。

- 平均 CPU 使用率 (パーセント)
- 平均ディスク読み込み (バイト)
- 平均ディスク書き込み (バイト)
- 最大ネットワーク受信 (バイト)
- 最大ネットワーク送信 (Bytes)
- 要約ディスク読み取り操作 (カウント)
- 要約ディスク書き込み操作 (カウント)
- 要約ステータス (任意)
- 要約ステータスインスタンス (カウント)
- 要約ステータスシステム (カウント)

グラフに表示されるメトリクスおよびデータの詳細については、インスタンスの利用可能な CloudWatch メトリクスのリスト表示を参照してください。

CloudWatch コンソールを使用したメトリクスのグラフ化

CloudWatch コンソールを使用して、Amazon EC2 や他の AWS のサービスによって生成されたメトリクスデータをグラフ化できます。詳細については、「Amazon CloudWatch ユーザーガイド」の「メトリクスのグラフ化」を参照してください。

インスタンスの CloudWatch アラームを作成する

インスタンスの 1 つのCloudWatch メトリクスをモニタリングする、CloudWatch アラームを作成できます。CloudWatch は、メトリクスが指定したしきい値に到達すると、自動的に通知を送信します。CloudWatch アラームは、Amazon EC2 コンソールを使用するか、CloudWatch コンソールに用意されている高度なオプションを使用して作成できます。

CloudWatch コンソールを使用してアラームを作成するには

例については、Amazon CloudWatch ユーザーガイドのAmazon CloudWatch アラームの作成を参照してください。

Amazon EC2 コンソールを使用してアラームを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。
4. [Manage CloudWatch alarms (CloudWatch アラームの管理)] 詳細ページの [Add or edit alarm (アラームの追加または編集)] で、[Create an alarm (アラームの作成)] を選択します。
5. [アラーム通知] で、Amazon Simple Notification Service (Amazon SNS) 通知を設定するかどうかを選択します。既存の Amazon SNS トピックを入力するか、名前を入力して新しいトピックを作成します。
6. [アラームアクション] で、アラームがトリガーされた際に実行するアクションを指定するかどうかを選択します。リストからアクションを選択します。
7. [Alarm thresholds (アラームのしきい値)] で、アラームのメトリクスと条件を選択します。例えば、CPU 使用率が 80% に達した状態が 5 分間継続した場合にトリガーされるアラームを作成するには、次の操作を行います。

    a. [サンプルのグループ化基準] ([平均]) と [サンプリングするデータのタイプ] ([CPU 使用率]) の設定は、デフォルトのままにしてきます。

b. [アラームの条件] で [>=] を選択し、[割合] に「**0.80**」と入力します。

c. [連続した期間] に「**1**」と入力し、[期間] で [5 分間] を選択します。

8. (オプション) [Sample metric data] (サンプルメトリクスデータ) の場合、[Add to dashboard] (ダッシュボードに追加) を選択します。

9. [Create] (作成) を選択します。

CloudWatch アラーム設定は、Amazon EC2 コンソールまたは CloudWatch コンソールから編集できます。アラームを削除する場合は、CloudWatch コンソールから行えます。詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch アラームの編集または削除」を参照してください。

インスタンスを停止、終了、再起動、または復旧するアラームを作成する

Amazon CloudWatch アラームアクションを使用して、 インスタンスを自動的に停止、終了、再起動、または復旧するアラームを作成できます。停止または終了アクションを使用すると、今後インスタンスを実行する必要がなくなったときにコストを節約できます。再起動アクションを使用すると、これらのインスタンスを自動的に再起動でき、復旧アクションを使用すると、システムで障害が発生した場合に新しいハードウェアで復旧できます。

> **Note**
>
> Amazon CloudWatch アラームの請求および料金に関する情報については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch の請求とコスト」を参照してください。

サービスにリンクされたロール `AWSServiceRoleForCloudWatchEvents` を使用すると、AWS がお客様に代わってアラームアクションを実行できます。AWS Management Console、AWS CLI、または IAM API で初めてアラームを作成する場合、CloudWatch はサービスにリンクされたロールを作成します。

自動的にインスタンスを停止または終了するシナリオはいくつもあります。例えば、バッチ給与計算処理ジョブまたは科学計算タスクを専用に行うインスタンスを使用している場合が挙げられます。これらのインスタンスは一定期間動作して仕事を完了します。このようなインスタンスは、アイドル状態 (課金されている状態) にせずに、停止または終了するとコスト削減につながります。停止アラームアクションと終了アラームアクションの主な違いとして、停止したインスタンスは、後で再実行が必要な場合に起動しやすいことと、同じインスタンス ID およびルートボリュームを維持できることがあります。しかし、終了したインスタンスを起動することはできません。代わりに新しいインスタ

ンスを開始する必要があります。インスタンスストアボリュームのデータは、インスタンスの停止または終了に伴って失われます。

停止、終了、再起動、復旧の各アクションは、Amazon EC2 インスタンスメトリクスごとに設定されている任意のアラームに追加できます。これには、Amazon CloudWatch によって (AWS/EC2 名前空間で) 提供される基本モニタリングや詳細モニタリングのメトリクスが含まれます。また、InstanceId ディメンションを含む任意のカスタムメトリクスも (その値が実行中の有効な Amazon EC2 インスタンスを参照する場合に限り) 含まれます。

コンソールのサポート

Amazon EC2 コンソールまたは CloudWatch コンソールを使用してアラームを作成できます。このドキュメントの手順では、Amazon EC2 コンソールを使用します。CloudWatch コンソールを使用する手順については、「Amazon CloudWatch ユーザーガイド」の「インスタンスを停止、終了、再起動、または復旧するアラームを作成する」を参照してください。

アクセス許可

EC2 アラームアクションを実行するアラームを作成または変更するには、iam:CreateServiceLinkedRole が必要です。サービスロールとは、サービスがユーザーに代わってアクションを実行するために引き受ける IAM ロールです。IAM 管理者は、IAM 内からサービスロールを作成、変更、削除できます。詳細については、「IAM ユーザーガイド」の「AWS のサービス にアクセス許可を委任するロールの作成」を参照してください。

コンテンツ

Amazon CloudWatch アラームへの停止アクションの追加

一定のしきい値に達したときに Amazon EC2 インスタンスを停止するアラームを作成できます。例えば、開発またはテスト用のインスタンスを実行したまま、終了するのを忘れることがたまにあります。平均 CPU 利用率が 24 時間 10% 未満である場合に、インスタンスがアイドル状態で使用され

ていないという信号を発してトリガーするアラームを作成できます。しきい値、持続時間、期間をニーズに合わせて調整し、アラームがトリガーされたときにメールを受信するよう Amazon Simple Notification Service (Amazon SNS) 通知を追加できます。

Amazon EBS ボリュームをルートデバイスとして使用するインスタンスは停止または終了できますが、インスタンスストアをルートデバイスとして使用するインスタンスでは終了のみ行えます。インスタンスストアボリュームのデータは、インスタンスの終了または停止に伴って失われます。

アイドル状態のインスタンスを停止させるアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号
   (+)
   を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

   c. [アラームアクション] をオンにして、[停止] を選択します。

   d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [CPU 使用率] を選択します。

   e. [アラーム発生時] と [パーセント] で、メトリクスのしきい値を指定します。この例では、<= と 10 % を指定します。

   f. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、5 分間の 1 連続期間を指定します。

   g. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

> **Note**
>
> アラーム設定は、アラームを作成する前に実際の要件に基づいて調整することも、アラーム作成後に編集することもできます。これにはメトリクス、しきい値、持続時間、アクション、通知設定などがあります。ただし、アラームの作成後のアラーム名の編集はできません。

h. [Create] (作成) を選択します。

Amazon CloudWatch アラームへの終了アクションの追加

(インスタンスで終了保護が有効になっていない限り)、一定のしきい値に達したときに EC2 インスタンスを自動的に終了させるアラームを作成することができます。例えば、インスタンスが仕事を終え、再びそのインスタンスを使用する必要がない場合は、インスタンスを削除することをお勧めします。後でインスタンスを使用する可能性がある場合は、インスタンスを削除するのではなく、停止するほうが良いでしょう。インスタンスストアボリュームのデータは、インスタンスの終了に伴って失われます。インスタンスの削除保護の有効化と無効化については、「終了保護を有効化する」を参照してください。

アイドル状態のインスタンスを終了するアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号
   ()
   を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification

Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

c. [アラームアクション] をオンにして、[終了] を選択します。

d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [CPU 使用率] を選択します。

e. [アラーム発生時] と [パーセント] で、メトリクスのしきい値を指定します。この例では、>= と 10 % を指定します。

f. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、1 時間の 24 連続期間を指定します。

g. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

> **Note**
>
> アラーム設定は、アラームを作成する前に実際の要件に基づいて調整することも、アラーム作成後に編集することもできます。これにはメトリクス、しきい値、持続時間、アクション、通知設定などがあります。ただし、アラームの作成後のアラーム名の編集はできません。

h. [Create] (作成) を選択します。

Amazon CloudWatch アラームへの再起動アクションの追加

Amazon EC2 インスタンスをモニタリングし、自動的に再起動する Amazon CloudWatch アラームを作成できます。再起動アラームアクションは、インスタンスのヘルスチェックが失敗した場合に推奨されます (システムのヘルスチェックが失敗した場合には、復旧アラームアクションが推奨されます) 。インスタンスの再起動は、オペレーティングシステムの再起動と同等です。ほとんどの場合、インスタンスの再起動には数分しかかかりません。インスタンスを再起動すると、インスタンスは同じホスト上で保持されるため、インスタンスのパブリック DNS 名、プライベート IP アドレス、およびインスタンスストアボリューム上のすべてのデータは保持されます。

インスタンスの停止と再起動の場合とは違って、インスタンスを再起動しても、インスタンスの新しい (1 分間分の最低料金がある) 課金期間は開始されません。インスタンスストアボリュームのデータは、インスタンスの再起動しても保持されます。インスタンスストアボリュームは、再起動後に

ファイルシステムに再マウントする必要があります。詳細については、「インスタンスの再起動」を参照してください。

> **Important**
>
> 再起動と復旧アクション間で不具合が発生するのを回避するには、再起動アラームと復旧アラームを同じ評価期間に設定するのを避けます。再起動アラームを各 1 分間の 3 回の評価期間に設定することをお勧めします。詳細については、Amazon CloudWatch ユーザーガイドのアラームを評価するを参照してください。

インスタンスを再起動するアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号 (+) を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

   c. [アラームアクション] をオンにして、[再起動] を選択します。

   d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [ステータスチェックに失敗しました: インスタンス] を選択しています。

   e. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、5 分間の 3 連続期間と入力しています。

f. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

g. [Create] (作成) を選択します。

Amazon CloudWatch アラームへの復旧アクションの追加

Amazon EC2 インスタンスをモニタリングする Amazon CloudWatch アラームを作成できます。下層のハードウェア障害または修復に AWS を必要とする問題によりインスタンスが正常に機能しなくなった場合に、自動的にインスタンスを復旧できます。終了したインスタンスは復旧できません。復旧されたインスタンスは、インスタンス ID、プライベート IP アドレス、Elastic IP アドレス、すべてのインスタンスメタデータを含め、元のインスタンスと同じです。

CloudWatch では、復旧アクションをサポートしていないインスタンスにあるアラームに、復旧アクションを追加することはできません。

`StatusCheckFailed_System` アラームがトリガーされ、復旧アクションが開始されると、アラームを作成し、復旧アクションに関連付けたときに選択した Amazon SNS トピックによって通知されます。インスタンスを復旧する際、インスタンスを再起動するときにインスタンスは移行され、メモリ内にあるデータは失われます。プロセスが完了すると、情報はアラームに設定された SNS トピックに発行されます。この SNS トピックをサブスクライブしているすべてのユーザーは、復旧処理のステータスと、それ以降の手順を含むメールの通知を受け取ります。インスタンスが復旧した時点でインスタンスが再起動されたことがわかります。

> **Note**
>
> 復旧アクションは、`StatusCheckFailed_System` でのみ使用できます。`StatusCheckFailed_Instance` では使用できません。

以下の問題が発生すると、システムステータスのチェックに失敗する可能性があります。

- ネットワーク接続の喪失
- システム電源の喪失
- 物理ホストのソフトウェアの問題
- ネットワーク到達可能性に影響する、物理ホスト上のハードウェアの問題

復旧アクションは、特定の特性を持つインスタンスでのみサポートされます。詳細については、「インスタンスの復旧」を参照してください。

インスタンスにパブリック IP アドレスが割り当てられている場合、復旧後にパブリック IP アドレスが維持されます。

> **Important**
>
> 再起動と復旧アクション間で不具合が発生するのを回避するには、再起動アラームと復旧アラームを同じ評価期間に設定するのを避けます。復旧アラームを各 1 分間の 2 回の評価期間に設定することをお勧めします。詳細については、Amazon CloudWatch ユーザーガイドのアラームを評価するを参照してください。

インスタンスを復旧するアラームを作成するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[モニタリングとトラブルシューティング]、[CloudWatch アラームの管理] の順に選択します。

   または、[アラームステータス] 列でプラス記号 (+) を選択できます。
4. [CloudWatch アラームの管理] ページで、次の操作を行います。

   a. [アラームの作成] を選択します。

   b. アラームがトリガーされたときに E メールを受信するには、[アラーム通知] で既存の Amazon SNS トピックを選択します。先ず、Amazon SNS コンソールを使用して Amazon SNS トピックを作成する必要があります。詳細については、Amazon Simple Notification Service デベロッパーガイド のAmazon SNS を使用した Application-to-Person (A2P) メッセージングを参照してください。

   > **Note**
   >
   > 今後、アラームがトリガーされたときにメール通知を受信するためには、指定された SNS トピックをサブスクライブする必要があります。AWS アカウントのルートユーザー は、自動インスタンス復旧アクションが発生すると、SNS トピックが指

定されていない場合や、ルートユーザーが指定した SNS トピックにサブスクライブしていない場合でも、常に E メール通知を受信します。

c. [アラームアクション] をオンにして、[復元] を選択します。

d. [サンプルのグループ化基準] と [サンプリングするデータのタイプ] で、統計とメトリクスを選択します。この例では、[平均] と [ステータスチェックに失敗しました: システム] を選択しています。

e. [連続期間] と [期間] で、アラームの評価期間を指定します。この例では、5 分間の 2 連続期間と入力しています。

f. Amazon CloudWatch は、自動的にアラーム名を作成します。名前を変更するには、[アラーム名] に新しい名前を入力します。アラーム名には ASCII 文字のみを使用する必要があります。

g. [Create] (作成) を選択します。

Amazon CloudWatch コンソールを使用してアラームとアクションの履歴を確認する

Amazon CloudWatch コンソールで、アラームとアクションの履歴を表示できます。Amazon CloudWatch は、過去 2 週間分のアラームとアクションの履歴を保持します。

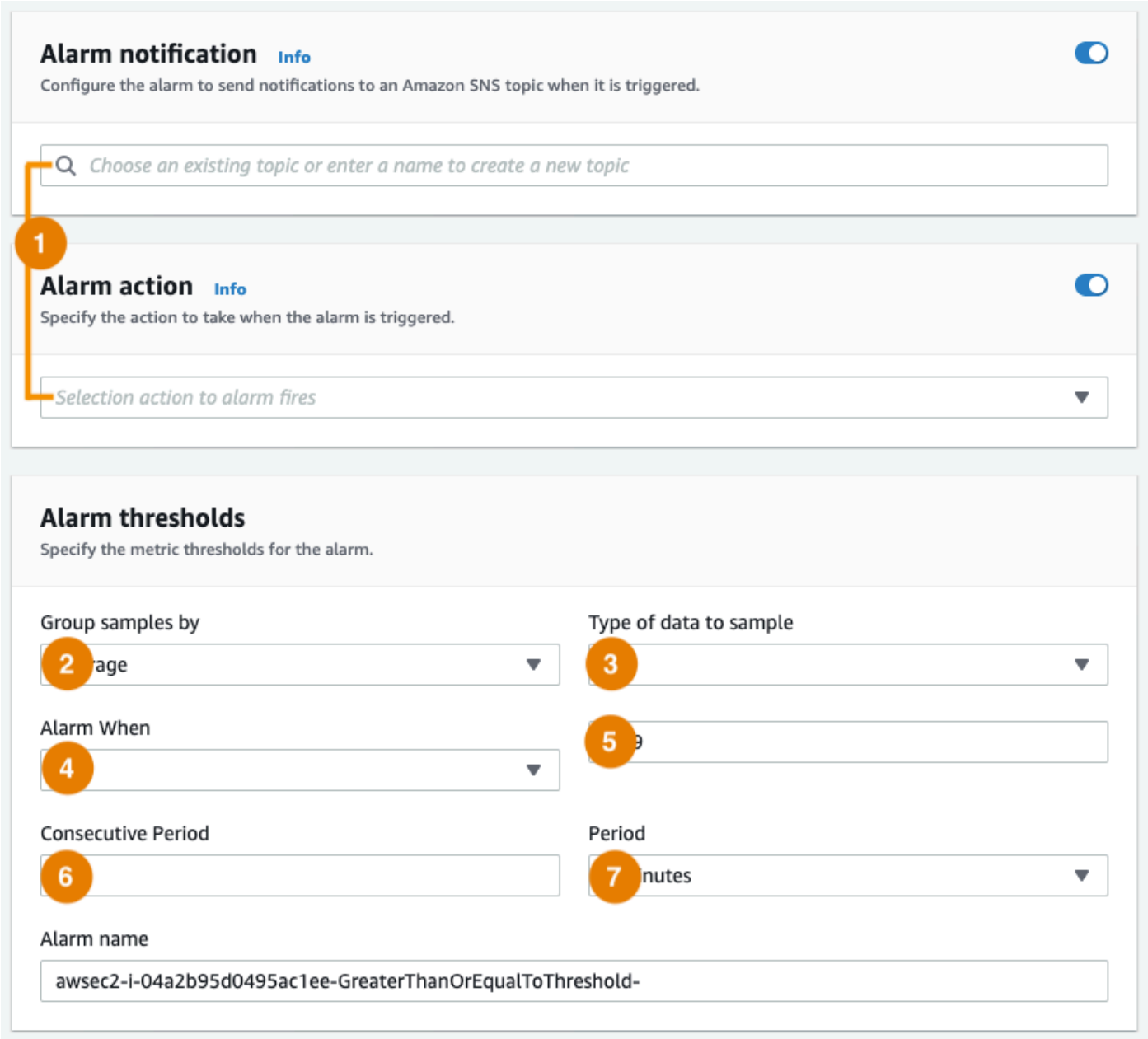
トリガーされたアラームとアクションを表示するには (CloudWatch コンソール)

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで、[Alarms] を選択します。
3. アラームを選択します。
4. [Details] タブには、直近の状態遷移、および時間とメトリクス値が表示されます。
5. 直近の履歴のエントリを表示するには、[History] タブを選択します。

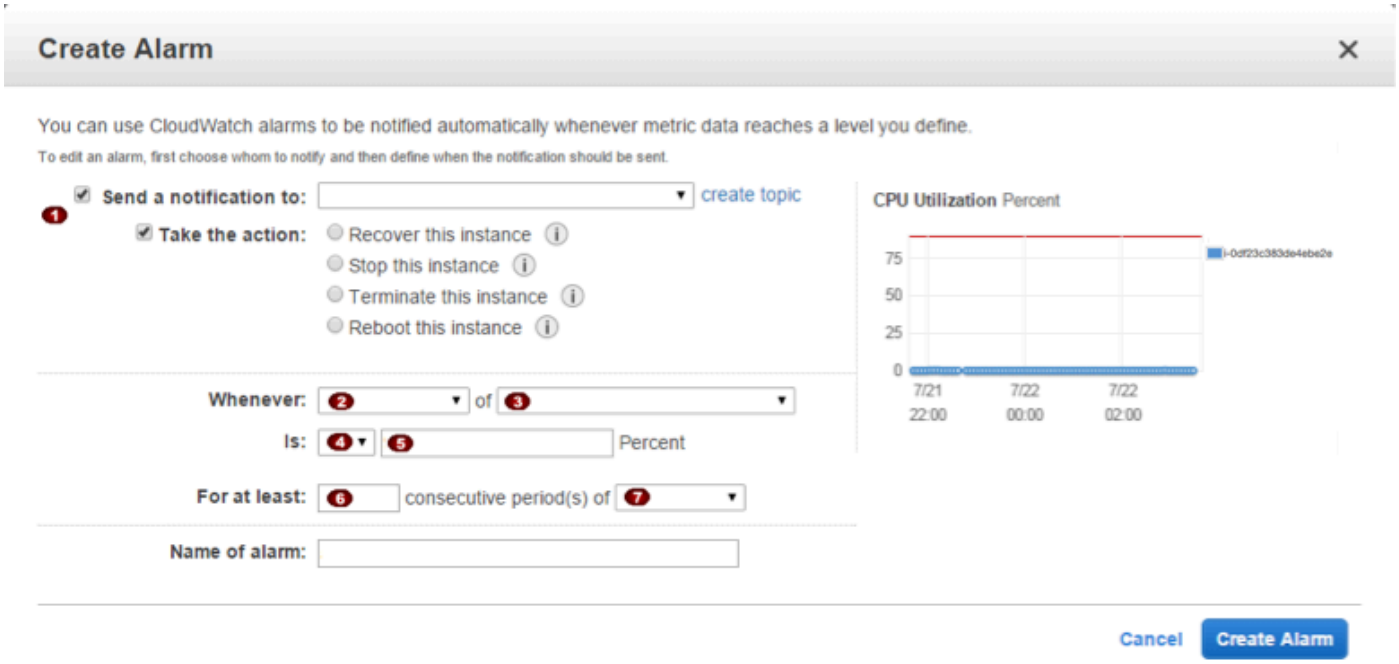
Amazon CloudWatch のアラームアクションのシナリオ

Amazon EC2 (Amazon EC2) コンソールを使用して、一定の条件が満たされたときにインスタンスを停止または終了させるアラームアクションを作成することができます。アラームアクションが設定する以下のコンソールページの画面キャプチャー内に、設定の順番を付けました。また、アクションを適切に作成できるよう、次のシナリオの設定にも順番を付けました。

New console


Alarm notification Info
Configure the alarm to send notifications to an Amazon SNS topic when it is triggered.
Choose an existing topic or enter a name to create a new topic
1
Alarm action Info
Specify the action to take when the alarm is triggered.
Selection action to alarm fires
Alarm thresholds
Specify the metric thresholds for the alarm.
Group samples by
2
Type of data to sample
3
Alarm When
4
5
Consecutive Period
6
Period
7
Alarm name
awsec2-i-04a2b95d0495ac1ee-GreaterThanOrEqualToThreshold-

Old console



シナリオ 1: アイドル状態の開発インスタンスおよびテストインスタンスを停止する

ソフトウェアの開発またはテストに使用するインスタンスが 1 時間以上アイドル状態である場合に停止するアラームを作成します。

設定	値
1	停止
2	最大
3	CPU 使用率 (%)
4	<=
5	10%
6	1
7	1 時間

シナリオ 2: アイドル状態のインスタンスを停止する

インスタンスが 24 時間アイドル状態である場合、インスタンスを停止し、メールを送信するアラームを作成します。

設定	値
1	停止および E メール
2	平均
3	CPU 使用率 (%)
4	<=
5	5%
6	24
7	1 時間

シナリオ 3: トラフィック量が異常に多いウェブサーバーについて E メールを送信する

インスタンスの 1 日当たりのアウトバウンドネットワークトラフィックが 10 GB を超える場合にメールを送信するアラームを作成します。

設定	値
1	メール
2	合計
3	ネットワーク出力
4	>
5	10 GB
6	24

設定	値
7	1 時間

シナリオ 4: トラフィック量が異常に多いウェブサーバーを停止する

アウトバウンドトラフィックが 1 時間当たり 1 GB を超えた場合にインスタンスを停止し、テキストメッセージ (SMS) を送信するアラームを作成します。

設定	値
1	Stop and send SMS
2	合計
3	ネットワーク出力
4	>
5	1 GB
6	1
7	1 時間

シナリオ 5: 障害のあるインスタンスを停止する

3 回連続で状態チェック (5 分間隔で実施) が不合格のインスタンスを停止するアラームを作成します。

設定	値
1	停止
2	平均
3	ステータスチェックに失敗: システム
4	-

設定	値
5	-
6	1
7	15 分

シナリオ 6: バッチ処理ジョブの完了時にインスタンスを削除する

バッチジョブを実行するインスタンスが結果データを送信しなくなったときに、そのインスタンスを削除するアラームを作成します。

設定	値
1	終了
2	最大
3	ネットワーク出力
4	<=
5	100,000 bytes
6	1
7	5 分

EventBridge を使用して Amazon EC2 を自動化する

Amazon EventBridge を使用すると、AWS のサービス を自動化し、アプリケーションの可用性の問題やリソースの変更などのシステムイベントに自動的に対応できます。AWS のサービスからのイベントは、ほぼリアルタイムに EventBridge に提供されます。ルールを作成して、注目しているイベントと、イベントがルールに一致した場合に実行するアクションを指定できます。自動的にトリガーできるオペレーションには、以下が含まれます。

- AWS Lambda 関数を呼び出す

- Amazon EC2 コマンドを実行 を呼び出す
- Amazon Kinesis Data Streams へのイベントを中継する
- AWS Step Functions ステートマシンをアクティブ化する
- Amazon SNS トピックを通知する
- Amazon SQS キューを通知する

Amazon EC2 での EventBridge の使用例を次に示します。

- インスタンスが実行状態になるたびに Lambda 関数をアクティブ化します。
- Amazon EBS ボリュームの作成時または変更時に Amazon SNS トピックを通知します。
- AWS の別のサービスで特定のイベントが発生するたびに、Amazon EC2 Run Command を使用して 1 つ以上の Amazon EC2 インスタンスにコマンドを送信します。

詳細については、「Amazon EventBridge ユーザーガイド」を参照してください。

Amazon EC2 イベントタイプ

Amazon EC2 は、次のイベントタイプをサポートします。

- EC2 AMI の状態変更
- EC2 フリートエラー
- EC2 フリート情報
- EC2 フリートインスタンスの変更
- EC2 フリートのスポットインスタンスリクエストの変更
- EC2 フリートの状態の変更
- EC2 インスタンスの再調整に関するレコメンデーション
- EC2 インスタンスの状態変更通知
- EC2 スポットフリートのエラー
- EC2 スポットフリート情報
- EC2 スポットフリートインスタンスの変更
- EC2 スポットフリートのスポットインスタンスリクエストの変更
- EC2 スポットフリートの状態の変更
- EC2 スポットインスタンスの中断の警告

- EC2 スポットインスタンスリクエストのフルフィルメント
- EC2 ODCR 低使用率通知

Amazon EBS でサポートされているイベントタイプについては、「EventBridge for Amazon EBS」を参照してください。

Amazon EC2 Linux インスタンスのメモリとディスクのメトリクスのモニタリング

Amazon CloudWatch を使用して、EC2 インスタンスのオペレーティングシステムからメトリクスおよびログを収集できます。

> **Important**
>
> CloudWatch モニタリングスクリプトは非推奨です。CloudWatch エージェントを使用してメトリクスおよびログを収集することを強くお勧めします。詳細については、Amazon CloudWatch ユーザーガイドのCloudWatch エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスを収集するを参照してください。
> 非推奨のモニタリングスクリプトからエージェントに移行し、モニタリングスクリプトに関する情報が必要な場合は、非推奨: CloudWatch モニタリングスクリプトを使用したメトリクスの収集を参照してください。

CloudWatch エージェントを使用したメトリクスの収集

CloudWatch エージェントを使用して、Amazon EC2 インスタンスとオンプレミスサーバーからシステムメトリクスとログファイルの両方を収集できます。エージェントでは Windows Server と Linux の両方がサポートされ、CPU あたりのコアのようなサブリソースメトリクスなど、収集するメトリクスを選択できます。メトリクスおよびログは、非推奨のモニタリングスクリプトを使用せずに、エージェントを使用して収集することをお勧めします。詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスとログを収集する」を参照してください。

非推奨: CloudWatch モニタリングスクリプトを使用したメトリクスの収集

> **Important**
>
> CloudWatch モニタリングスクリプトは非推奨です。使用できなくなったが、これらの監視スクリプトをまだ使用しているお客様向けに、これらの監視スクリプトに関する情報をこのページで提供します。
> CloudWatch エージェントを使用してメトリクスおよびログを収集することを強くお勧めします。詳細については、「Amazon CloudWatch ユーザーガイド」の「CloudWatch エージェントを使用して Amazon EC2 インスタンスとオンプレミスサーバーからメトリクスとログを収集する」を参照してください。

モニタリングスクリプトは、Amazon CloudWatch のカスタムメトリクスを作成して利用する方法を示しています。これらの Perl スクリプトのサンプルは、Linux インスタンスのメモリ、スワップ、およびディスクスペースの使用状況メトリクスをレポートする、完全に機能する例で構成されます。

Amazon CloudWatch 標準のカスタムメトリクスの利用料金が、これらのスクリプトの使用に適用されます。詳細については、Amazon CloudWatch料金表ページを参照してください。

コンテンツ

サポートされているシステム

モニタリングスクリプトは、以下のシステムを使用してインスタンスでテストされました。その他のオペレーティングシステムでのモニタリングスクリプトの使用はサポートされていません。

- Amazon Linux 2

- Amazon Linux AMI 2014.09.2 以降
- Red Hat Enterprise Linux 6.9 および 7.4
- SUSE Linux Enterprise Server 12
- Ubuntu Server 14.04 および 16.04

必要なアクセス許可

IAM ロールをインスタンスに関連付けて、次のアクションを呼び出すアクセス許可がスクリプトにあることを確認します。

- `cloudwatch:PutMetricData`
- `cloudwatch:GetMetricStatistics`
- `cloudwatch:ListMetrics`
- `ec2:DescribeTags`

詳細については、「IAM ロールの使用」を参照してください。

必要なパッケージをインストールする

Linux の一部のバージョンでは、モニタリングスクリプトを使用する前に、Perl モジュールをインストールする必要があります。

Amazon Linux 2 および Amazon Linux AMI に必要なパッケージをインストールするには

1. インスタンスにログオンします。詳細については、「Linux インスタンスへの接続」を参照してください。
2. コマンドプロンプトで以下のようにパッケージをインストールします。

```
sudo yum install -y perl-Switch perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-https perl-Digest-SHA.x86_64
```

Ubuntu に必要なパッケージをインストールするには

1. インスタンスにログオンします。詳細については、「Linux インスタンスへの接続」を参照してください。
2. コマンドプロンプトで以下のようにパッケージをインストールします。

```
sudo apt-get update
sudo apt-get install unzip
sudo apt-get install libwww-perl libdatetime-perl
```

Red Hat Enterprise Linux 7 に必要なパッケージをインストールするには

1. インスタンスにログオンします。詳細については、「Linux インスタンスへの接続」を参照してください。
2. コマンドプロンプトで以下のようにパッケージをインストールします。

```
sudo yum install perl-Switch perl-DateTime perl-Sys-Syslog perl-LWP-Protocol-https
 perl-Digest-SHA --enablerepo="rhui-REGION-rhel-server-optional" -y
sudo yum install zip unzip
```

Red Hat Enterprise Linux 6.9 に必要なパッケージをインストールするには

1. インスタンスにログオンします。詳細については、「Linux インスタンスへの接続」を参照してください。
2. コマンドプロンプトで以下のようにパッケージをインストールします。

```
sudo yum install perl-DateTime perl-CPAN perl-Net-SSLeay perl-IO-Socket-SSL perl-
Digest-SHA gcc -y
sudo yum install zip unzip
```

3. 昇格されたユーザーとして CPAN を実行します。

```
sudo cpan
```

次のプロンプトが表示されるまで、各プロンプトで Enter キーを押します。

```
cpan[1]>
```

4. CPAN プロンプトで、次の各コマンドを実行します。1 つのコマンドを実行してインストールを実行し、CPAN プロンプトに戻ったら次のコマンドを実行します。次の処理に進むことを求めるプロンプトが表示されたら、Enter キーを押します。

```
cpan[1]> install YAML
```

```
cpan[2]> install LWP::Protocol::https
cpan[3]> install Sys::Syslog
cpan[4]> install Switch
```

SUSE に必要なパッケージをインストールするには

1. インスタンスにログオンします。詳細については、「Linux インスタンスへの接続」を参照してください。
2. SUSE Linux Enterprise Server 12 が実行されているサーバーでは、perl-Switch パッケージのダウンロードが必要な場合があります。次のコマンドを使用して、このパッケージをダウンロードおよびインストールできます。

```
wget http://download.opensuse.org/repositories/devel:/languages:/perl/SLE_12_SP3/
noarch/perl-Switch-2.17-32.1.noarch.rpm
sudo rpm -i perl-Switch-2.17-32.1.noarch.rpm
```

3. 次のように、必要なパッケージをインストールします。

```
sudo zypper install perl-Switch perl-DateTime
sudo zypper install -y "perl(LWP::Protocol::https)"
```

モニタリングスクリプトをインストールする

以下の手順では、EC2 Linux インスタンスで CloudWatch モニタリングスクリプトのダウンロード、解凍、構成を行う方法について示します。

モニタリングスクリプトのダウンロード、インストール、設定を行うには

1. コマンドプロンプトで、モニタリングスクリプトを保存するフォルダに移動し、次のコマンドを実行してモニタリングスクリプトをダウンロードします。

```
curl https://aws-cloudwatch.s3.amazonaws.com/downloads/
CloudWatchMonitoringScripts-1.2.2.zip -O
```

2. ダウンロードしたモニタリングスクリプトをインストールするには、以下のコマンドを実行します。

```
unzip CloudWatchMonitoringScripts-1.2.2.zip && \
rm CloudWatchMonitoringScripts-1.2.2.zip && \
```

```
cd aws-scripts-mon
```

モニタリングスクリプトのパッケージに、以下のファイルが含まれます。

- CloudWatchClient.pm - 他のスクリプトから Amazon CloudWatch を簡単に呼び出せる共通 Perl モジュールです。
- mon-put-instance-data.pl - Amazon EC2 インスタンス (メモリ、スワップ、ディスクスペースの使用状況) のシステムメトリクスを収集し、Amazon CloudWatch に送信します。
- mon-get-instance-stats.pl - Amazon CloudWatch にクエリを実行して、このスクリプトが実行される EC2 インスタンスの最新の使用状況統計を表示します。
- awscreds.template - アクセスキー ID とシークレットアクセスキーを保存する AWS 認証情報のファイルテンプレートです。
- LICENSE.txt - Apache 2.0 のライセンスを含むテキストファイルです。
- NOTICE.txt 著作権情報です。

mon-put-instance-data.pl

このスクリプトは、現行システムにあるメモリ、スワップ、ディスクスペースの使用状況のデータを収集します。その後、Amazon CloudWatch へのリモート呼び出しを行って、収集したデータをカスタムメトリクスとしてレポートします。

オプション

名前	説明
`--mem-util`	MemoryUtilization メトリクスをパーセント (%) 単位で収集し、送信します。このメトリクスには、使用されているアプリケーションとオペレーティングシステムによって割り当てられたメモリがカウントされるほか、`--mem-used-incl-cache-buff` オプションを指定した場合は、使用されているキャッシュとバッファメモリもカウントされます。
`--mem-used`	メガバイト (MB) 単位でレポートされる MemoryUsed メトリクスを収集し、送信します。このメトリクスには、使用されているアプリケーションとオペレーティングシステムによって割り当てられたメモリがカウントされるほか、`--mem-used-`

名前	説明
	incl-cache-buff オプションを指定した場合は、使用されているキャッシュとバッファメモリもカウントされます。
--mem-used-incl-cache-buff	このオプションを含めると、キャッシュおよびバッファに現在使用されているメモリは、--mem-util 、--mem-used 、--mem-avail のメトリクスがレポートされるときに、「used」としてカウントされます。
--mem-avail	メガバイト (MB) 単位でレポートされる MemoryAvailable メトリクスを収集し、送信します。このメトリクスには、使用されているアプリケーションとオペレーティングシステムによって割り当てられたメモリがカウントされるほか、--mem-used-incl-cache-buff オプションを指定した場合は、使用されているキャッシュとバッファメモリもカウントされます。
--swap-util	パーセント (%) 単位でレポートされる SwapUtilization メトリクスを収集し、送信します。
--swap-used	メガバイト (MB) 単位でレポートされる SwapUsed メトリクスを収集し、送信します。
--disk-path=PATH	レポートするディスクを選択します。 PATH では、マウントポイント、またはレポートが必要なファイルシステムのマウントポイントにあるファイルを指定できます。複数のディスクを選択するには、それぞれに対して --disk-path=PATH を指定します。 / および /home にマウントされたファイルシステムのディスクを選択するには、パラメータを使用します。 --disk-path=/ --disk-path=/home

名前	説明
--disk-space-util	選択したディスクについて、DiskSpaceUtilization メトリクスを収集し送信します。メトリクスはパーセンテージでレポートされます。 このスクリプトによって計算されたディスクの使用状況メトリクスは、df -k -l コマンドによって計算された値とは異なることに注意してください。df -k -l の値のほうが有用であると判断した場合は、スクリプトのほうの計算値を変更できます。
--disk-space-used	選択したディスクについて、DiskSpaceUsed メトリクスを収集し送信します。メトリクスは、デフォルトにより、ギガバイトでレポートされます。 Linux オペレーティングシステムには予約ディスクスペースがあるため、使用済みディスクスペースと使用可能なディスクスペースを合計しても正確なディスクスペースの合計にならないことがあります。
--disk-space-avail	選択したディスクについて、DiskSpaceAvailable メトリクスを収集し送信します。メトリクスはギガバイトでレポートされます。 Linux オペレーティングシステムには予約ディスクスペースがあるため、使用済みディスクスペースと使用可能なディスクスペースを合計しても正確なディスクスペースの合計にならないことがあります。
--memory-units=UNITS	メモリ使用量をレポートする単位を指定します。指定がない場合、メモリはメガバイト (MB) でレポートされます。UNITS は、バイト (B)、キロバイト (KB)、メガバイト (MB)、ギガバイト (GB) のいずれかになります。
--disk-space-units =UNITS	ディスクスペース使用量をレポートする単位を指定します。指定がない場合、ディスクスペースはギガバイト (GB) でレポートされます。UNITS は、バイト (B)、キロバイト (KB)、メガバイト (MB)、ギガバイト (GB) のいずれかになります。

名前	説明
--aws-credential-file=PATH	AWS 認証情報を持っているファイルの場所を提供します。 このパラメータは、--aws-access-key-id および --aws-secret-key パラメータと一緒には使用できません。
--aws-access-key-id=VALUE	発信者を識別するために使用する AWS アクセスキー ID を指定します。--aws-secret-key オプションと一緒に使用する必要があります。このオプションを --aws-credential-file パラメータと一緒に使用しないでください。
--aws-secret-key=VALUE	CloudWatch へのリクエストの署名に使用する AWS シークレットアクセスキーを指定します。--aws-access-key-id オプションと一緒に使用する必要があります。このオプションを --aws-credential-file パラメータと一緒に使用しないでください。
--aws-iam-role=VALUE	AWS 認証情報を提供するために使用する IAM ロールを指定します。値 =VALUE が必要です。認証情報が指定されていない場合、EC2 インスタンスに関連付けられたデフォルトの IAM ロールが適用されます。使用できる IAM ロールは 1 つのみです。IAM ロールが検出されない場合、または 2 つ以上の IAM ロールが検出された場合、スクリプトはエラーを返します。 このオプションを --aws-credential-file 、--aws-access-key-id 、または --aws-secret-key パラメータと併せて使用しないでください。
--aggregated[=only]	インスタンスタイプ、AMI ID、リージョン全体の集約されたメトリクスを追加します。値 =only はオプションです。指定した場合、スクリプトは集約されたメトリクスのみをレポートします。

名前	説明
`--auto-scaling[=only]`	Auto Scaling グループの集約されたメトリクスを追加します。値 =only はオプションです。指定すると、スクリプトは Auto Scaling メトリクスのみをレポートします。スクリプトを使って IAM ユーザーまたはロールに関連付けられている IAM ポリシーには、EC2 アクション DescribeTags を呼び出す許可が必要になります。
`--verify`	メトリクスを収集するスクリプトのテストランを実行したり、完全な HTTP リクエストを用意したりしますが、実際に CloudWatch を呼び出してデータをレポートすることはありません。このオプションで、認証情報が提供されていることも確認できます。冗長モードで実行すると、このオプションは CloudWatch に送信するメトリクスを出力します。
`--from-cron`	cron からスクリプトを呼び出す際はこのオプションを使用します。このオプションを使用すると、すべての診断出力が抑えられますが、エラーメッセージがユーザーのローカルシステムログに送信されます。
`--verbose`	スクリプトの実行内容の詳細を表示します。
`--help`	使用状況の情報を表示します。
`--version`	スクリプトのバージョン番号を表示します。

例

次の例では、IAM ロールまたは `awscreds.conf` ファイルを指定していることを前提としています。それ以外の場合は、これらのコマンドで `--aws-access-key-id` および `--aws-secret-key` パラメータを使用して認証情報を指定する必要があります。

次の例では、CloudWatch にデータを送信せずに簡単なテストを実行します。

```
./mon-put-instance-data.pl --mem-util --verify --verbose
```

以下の例では、使用可能なメモリメトリクスをすべて収集し、CloudWatch に送信して、使用されているキャッシュとバッファメモリをカウントします。

```
./mon-put-instance-data.pl --mem-used-incl-cache-buff --mem-util --mem-used --mem-avail
```

以下の例では、Auto Scaling グループの集約メトリクスを収集し、個々のインスタンスメトリクスをレポートすることなく Amazon CloudWatch に送信します。

```
./mon-put-instance-data.pl --mem-util --mem-used --mem-avail --auto-scaling=only
```

以下の例では、インスタンスタイプ、AMI ID、リージョンの集約されたメトリクスを収集し、個々のインスタンスメトリクスをレポートすることなく Amazon CloudWatch に送信します。

```
./mon-put-instance-data.pl --mem-util --mem-used --mem-avail --aggregated=only
```

CloudWatch にレポートされたメトリクスの cron スケジュールを設定するには、crontab -e コマンドを使用して crontab の編集を開始します。5 分ごとにメモリとディスクスペースの使用状況を CloudWatch にレポートするには、以下のコマンドを追加します。

```
*/5 * * * * ~/aws-scripts-mon/mon-put-instance-data.pl --mem-used-incl-cache-buff --
mem-util --disk-space-util --disk-path=/ --from-cron
```

スクリプトにエラーが発生した場合、スクリプトのシステムログにエラーメッセージが書き込まれます。

mon-get-instance-stats.pl

このスクリプトは、直近の時間数を用いて、指定された時間間隔内で、メモリ、スワップ、ディスクスペースメトリクスの統計について CloudWatch に問い合わせます。このデータは、このスクリプトが実行される Amazon EC2 インスタンスに関するものです。

オプション

名前	説明
`--recent-hours=N`	レポートする直近の時間数を N で表記して指定します。ここで N は整数です。

名前	説明
--aws-credential-file=PATH	AWS 認証情報を持っているファイルの場所を提供します。
--aws-access-key-id=VALUE	発信者を識別するために使用する AWS アクセスキー ID を指定します。--aws-secret-key オプションと一緒に使用する必要があります。このオプションを --aws-credential-file オプションと併せて使用しないでください。
--aws-secret-key=VALUE	CloudWatch へのリクエストの署名に使用する AWS シークレットアクセスキーを指定します。--aws-access-key-id オプションと一緒に使用する必要があります。このオプションを --aws-credential-file オプションと併せて使用しないでください。
--aws-iam-role=VALUE	AWS 認証情報を提供するために使用する IAM ロールを指定します。値 =VALUE が必要です。認証情報が指定されていない場合、EC2 インスタンスに関連付けられたデフォルトの IAM ロールが適用されます。使用できる IAM ロールは 1 つのみです。IAM ロールが検出されない場合、または 2 つ以上の IAM ロールが検出された場合、スクリプトはエラーを返します。 このオプションを --aws-credential-file 、--aws-access-key-id 、または --aws-secret-key パラメータと併せて使用しないでください。
--verify	スクリプトのテストを実行します。このオプションで、認証情報が提供されていることも確認できます。
--verbose	スクリプトの実行内容の詳細を表示します。
--help	使用状況の情報を表示します。
--version	スクリプトのバージョン番号を表示します。

例

過去 12 時間の利用統計情報を取得するには、次のコマンドを実行します。

```
./mon-get-instance-stats.pl --recent-hours=12
```

以下に、応答の例を示します。

```
Instance metric statistics for the last 12 hours.

CPU Utilization
    Average: 1.06%, Minimum: 0.00%, Maximum: 15.22%

Memory Utilization
    Average: 6.84%, Minimum: 6.82%, Maximum: 6.89%

Swap Utilization
    Average: N/A, Minimum: N/A, Maximum: N/A

Disk Space Utilization on /dev/xvda1 mounted as /
    Average: 9.69%, Minimum: 9.69%, Maximum: 9.69%
```

コンソールでのカスタムメトリクスの表示

正常に `mon-put-instance-data.pl` スクリプトを実行すると、Amazon CloudWatch コンソールでカスタムメトリックスを確認できます。

カスタムメトリクスを表示するには

1. 前述のとおりに `mon-put-instance-data.pl` を実行します。
2. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
3. [View Metrics] を選択します。
4. [Viewing] (表示中) では、スクリプトによって投入されたカスタムメトリクスが `System/Linux` というプレフィックス付きで表示されます。

トラブルシューティング

CloudWatchClient.pm モジュールは、インスタンスのメタデータをローカルでキャッシュします。モニタリングスクリプトを実行しているインスタンスから AMI を作成すると、キャッシュ TTL (デフォルト: 6 時間、Auto Scaling グループでは 24 時間) 以内にこの AMI から起動したすべてのインスタンスは、元のインスタンスのインスタンス ID を使用してメトリクスを出力します。キャッシュ TTL 期間が経過した後は、スクリプトは新しいデータを取得し、モニタリングスクリプトは現在のインスタ

ンスのインスタンス ID を使用します。これをすぐに修正するには、次のコマンドを使用してキャッシュされたデータを削除します。

```
rm /var/tmp/aws-mon/instance-id
```

AWS CloudTrail による Amazon EC2 および Amazon EBS の API コールのログ記録

Amazon EC2 および Amazon EBS は、Amazon EC2 および Amazon EBS のユーザー、ロール、または AWS CloudTrail のサービスによって実行されたアクションの記録を提供するサービスである AWS と統合されます。CloudTrail は、コンソールからの呼び出し、および API へのコード呼び出しを含む、Amazon EC2 および Amazon EBS のすべての API 呼び出しをイベントとしてキャプチャします。証跡を作成する場合は、Amazon EC2 や Amazon EBS のイベントなど、Amazon S3 バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [Event history (イベント履歴)] で最新のイベントを表示できます。CloudTrail によって収集された情報を使用して、リクエストの作成元の IP アドレス、リクエストの実行者、リクエストの実行日時などの詳細を調べて、Amazon EC2 および Amazon EBS に対してどのようなリクエストが行われたかを判断できます。

CloudTrail の詳細については、AWS CloudTrail ユーザーガイドを参照してください。

CloudTrail での Amazon EC2 と Amazon EBS に関する情報

CloudTrail は、AWS アカウントを作成すると、その中で有効になります。Amazon EC2 および Amazon EBS でアクティビティが発生すると、そのアクティビティは CloudTrail イベントに、[イベント履歴] の他の AWS のサービス のイベントと共に記録されます。最近のイベントは、AWS アカウント で表示、検索、ダウンロードできます。詳細については、 CloudTrail イベント履歴でのイベントの表示を参照してください。

Amazon EC2 および Amazon EBS のイベントなど、AWS アカウント のイベントの継続的な記録用に追跡を作成します。証跡により、CloudTrail はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョン に適用されます。証跡は、AWS パーティションのすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、CloudTrail ログで収集したイベントデータをより詳細に分析し、それに基づく対応するためにその他の AWS のサービスを設定できます。詳細については、以下を参照してください。

- AWS アカウント の追跡の作成

- AWS のサービス と CloudTrail ログの統合
- CloudTrail の Amazon SNS 通知の設定
- CloudTrail ログファイルを複数のリージョンから受け取ると複数のアカウントから CloudTrail ログファイルを受け取る

すべての Amazon EC2 アクションと Amazon EBS 管理アクションは CloudTrail によってログが記録されます。これらは、Amazon EC2 API リファレンスにドキュメント化されています。例えば、RunInstances、DescribeInstances または CreateImage の各アクションを呼び出すと、CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、リクエストの生成者に関する情報が含まれます。アイデンティティ情報は、以下を判別するために役立ちます。

- リクエストが、ルートユーザーまたは IAM ユーザーのどちらの認証情報を使用して送信されたかどうか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが、別の AWS のサービス によって送信されたかどうか。

詳細については、CloudTrail userIdentity 要素を参照してください。

Amazon EC2 と Amazon EBS のログファイルエントリについて

証跡は、指定した Amazon S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail のログファイルは、単一か複数のログエントリを含みます。イベントはあらゆるソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストのパラメータなどの情報が含まれます。CloudTrail ログファイルは、パブリック API コールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次のログファイルレコードは、ユーザーがインスタンスを終了したことを示しています。

```
{
   "Records":[
      {
         "eventVersion":"1.03",
         "userIdentity":{
            "type":"Root",
            "principalId":"123456789012",
```

```
            "arn":"arn:aws:iam::123456789012:root",
            "accountId":"123456789012",
            "accessKeyId":"AKIAIOSFODNN7EXAMPLE",
            "userName":"user"
         },
         "eventTime":"2016-05-20T08:27:45Z",
         "eventSource":"ec2.amazonaws.com",
         "eventName":"TerminateInstances",
         "awsRegion":"us-west-2",
         "sourceIPAddress":"198.51.100.1",
         "userAgent":"aws-cli/1.10.10 Python/2.7.9 Windows/7botocore/1.4.1",
         "requestParameters":{
            "instancesSet":{
               "items":[{
                  "instanceId":"i-1a2b3c4d"
               }]
            }
         },
         "responseElements":{
            "instancesSet":{
               "items":[{
                  "instanceId":"i-1a2b3c4d",
                  "currentState":{
                     "code":32,
                     "name":"shutting-down"
                  },
                  "previousState":{
                     "code":16,
                     "name":"running"
                  }
               }]
            }
         },
         "requestID":"be112233-1ba5-4ae0-8e2b-1c302EXAMPLE",
         "eventID":"6e12345-2a4e-417c-aa78-7594fEXAMPLE",
         "eventType":"AwsApiCall",
         "recipientAccountId":"123456789012"
      }
   ]
}
```

AWS CloudTrail を使用して、EC2 Instance Connect 経由で接続するユーザーを監査する

AWS CloudTrail を使用して、EC2 Instance Connect 経由でインスタンスに接続するユーザーを監査します。

AWS CloudTrail コンソールを使用して、EC2 Instance Connect 経由で SSH アクティビティを監査するには

1. https://console.aws.amazon.com/cloudtrail/ で AWS CloudTrail コンソールを開きます。
2. 正しいリージョンを使用していることを確認します。
3. ナビゲーションペインで [Event history (イベント履歴)] を選択します。
4. [Filter (フィルター)] で、[Event source (イベントソース)]、[ec2-instance-connect.amazonaws.com] の順に選択します。
5. (オプション) [Time range (時間範囲)] で、時間範囲を選択します。
6. [Refresh events (イベントの更新)] アイコンを選択します。
7. SendSSHPublicKey API コールに対応するイベントがページに表示されます。矢印を使用してイベントを展開します。ユーザー名、SSH 接続を行うために使用した AWS アクセスキー、ソース IP アドレスなどの詳細が表示されます。
8. すべてのイベント情報を JSON 形式で表示するには、[View event (イベントの表示)] を選択します。[requestParameters] フィールドに、SSH 接続を行うために使用されたターゲットインスタンス ID、OS ユーザー名、およびパブリックキーが表示されます。

```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "ABCDEFGONGNOMOOCB6XYTQEXAMPLE",
        "arn": "arn:aws:iam::1234567890120:user/IAM-friendly-name",
        "accountId": "123456789012",
        "accessKeyId": "ABCDEFGUKZHNAW4OSN2AEXAMPLE",
        "userName": "IAM-friendly-name",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-09-21T21:37:58Z"}
        }
    },
```

```
    "eventTime": "2018-09-21T21:38:00Z",
    "eventSource": "ec2-instance-connect.amazonaws.com",
    "eventName": "SendSSHPublicKey ",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "123.456.789.012",
    "userAgent": "aws-cli/1.15.61 Python/2.7.10 Darwin/16.7.0 botocore/1.10.60",
    "requestParameters": {
        "instanceId": "i-0123456789EXAMPLE",
        "osUser": "ec2-user",
        "SSHKey": {
            "publicKey": "ssh-rsa ABCDEFGHIJKLMNO01234567890EXAMPLE"
        }
     },
    "responseElements": null,
    "requestID": "1a2s3d4f-bde6-11e8-a892-f7ec64543add",
    "eventID": "1a2w3d4r5-a88f-4e28-b3bf-30161f75be34",
    "eventType": "AwsApiCall",
    "recipientAccountId": "0987654321"
}
```

CloudTrail イベントを S3 バケットに収集するために AWS アカウントを設定している場合は、プログラムで情報をダウンロードして監査できます。詳細については、「AWS CloudTrail ユーザーガイド」の「CloudTrail ログファイルの取得と表示」を参照してください。

Amazon EC2におけるネットワーク

Amazon VPC を使用すると、Virtual Private Cloud (VPC) と呼ばれる AWS アカウント専用の仮想ネットワークに対して Amazon EC2 インスタンスなどの AWS リソースを起動できます。インスタンスを起動するときに、VPC からサブネットを選択できます。インスタンスには、論理的な仮想ネットワークカードであるプライマリネットワークインターフェイスが設定されています。インスタンスは、サブネットの IPv4 アドレスからプライマリプライベート IP アドレスを受け取ります。そのアドレスは、プライマリネットワークインターフェイスに割り当てられます。

インスタンスが Amazon のパブリック IP アドレスのプールからパブリック IP アドレスを受け取るかどうかをコントロールできます。インスタンスのパブリック IP アドレスは、インスタンスが停止または終了するまでに限り、インスタンスに関連付けられます。永続的なパブリック IP アドレスが必要な場合は、AWS アカウントに Elastic IP アドレスを割り当て、インスタンスまたはネットワークインターフェイスに関連付けることができます。Elastic IP アドレスは、ユーザーが AWS アカウントをリリースするまでアカウントに関連付けられたままであり、必要に応じてインスタンス間でそのアドレスを移動できます。独自の IP アドレスの範囲を AWS アカウントに持ち込み、アドレスプールとして表示して、そこから Elastic IP アドレスを割り当てることができます。

ネットワークのパフォーマンスを向上させ、レイテンシーを低減するために、プレイスメントグループ内でインスタンスを起動できます。拡張ネットワーキングを使用すると、1 秒あたりのパケット (PPS) のパフォーマンスが大幅に向上します。Elastic Fabric Adapter (EFA) を使用すると、ハイパフォーマンスコンピューティングおよび機械学習アプリケーションを高速化できます。EFA は、サポートされているインスタンスタイプにアタッチできるネットワークデバイスです。

機能

- リージョンとゾーン
- Amazon EC2 インスタンスの IP アドレス指定
- Amazon EC2 インスタンスのホスト名タイプ
- Amazon EC2 で自分の IP アドレスを使用する (BYOIP)
- Elastic IP アドレス
- Elastic Network Interface
- Amazon EC2 インスタンスのネットワーク帯域幅
- Linux での拡張ネットワーキング
- Elastic Fabric Adapter
- Amazon EC2 インスタンストポロジー

- プレイスメントグループ
- EC2 インスタンスのネットワークの最大送信単位 (MTU)
- 仮想プライベートクラウド

リージョンとゾーン

Amazon EC2 は、世界各地の場所でホスティングされています。これらの場所は、AWS リージョン、アベイラビリティーゾーン、Local Zones、AWS Outposts、および Wavelength Zones で構成されます。

- リージョンはそれぞれ、地理的に離れた領域です。
- アベイラビリティーゾーンは、各リージョン内の複数の独立した場所です。
- Local Zones を使用すると、コンピューティングやストレージなどのリソースをエンドユーザーに近い複数の場所に配置できます。
- AWS Outposts では、ネイティブの AWS のサービス、インフラストラクチャ、運用モデルをほぼすべてのデータセンター、コロケーションスペース、オンプレミスの施設で利用できます。
- Wavelength Zones を使用すると、デベロッパーは 5G デバイスやエンドユーザーに非常に低いレイテンシーを提供するアプリケーションを構築できます。Wavelength は、標準の AWS コンピューティングおよびストレージサービスを通信事業者の 5G ネットワークのエッジにデプロイします。

AWS は、最新の高可用性のデータセンターを運用しています。しかし、非常にまれですが、同じ場所にあるインスタンスすべての可用性に影響する障害が発生することもあります。すべてのインスタンスを 1 か所でホストしている場合、そのような障害が起きると、すべてのインスタンスが利用できなくなります。

最適なデプロイを確認するには、AWS Wavelength に関するよくある質問を参照してください。

コンテンツ

- リージョン
- アベイラビリティーゾーン
- Local Zones
- Wavelength Zone
- AWS Outposts

リージョン

各リージョンは、他のリージョンと完全に分離されるように設計されています。これにより、最大限の耐障害性と安定性が達成されます。

リソースを表示すると、指定したリージョンに結び付けられているリソースのみが表示されます。これは、リージョンが相互に分離されており、リージョン間ではリソースが自動的にレプリケートされないためです。

インスタンスを起動するときは、同じリージョン内にある AMI を選択する必要があります。AMI が別のリージョンにある場合は、使用しているリージョンに AMI をコピーできます。詳細については、AMI のコピーを参照してください。

リージョン間のデータ転送には料金がかかることに注意してください。詳細については、Amazon EC2 料金表 - データ転送を参照してください。

コンテンツ

利用できるリージョン

アカウントにより、利用できるリージョンが決まります。

- AWS アカウント は複数のリージョンを提供するため、要件に合った場所で Amazon EC2 インスタンスを起動できます。例えば、ヨーロッパの顧客に近づけるため、または法的要件を満たすために、ヨーロッパでインスタンスを起動することができます。
- AWS GovCloud (米国西部) アカウントでは、AWS GovCloud (米国西部) リージョンおよび AWS GovCloud (米国東部) リージョンにアクセスできます。詳細については、「AWS GovCloud (US)」を参照してください。
- Amazon AWS (中国) アカウントでは、北京および寧夏リージョンにのみアクセスできます。詳細については、「Amazon Web Services in China」(中国でのアマゾン ウェブ サービス) を参照してください。

次の表は、AWS アカウント で提供されるリージョンの一覧です。AWS GovCloud (US) Regions や中国のリージョンなど、追加のリージョンを AWS アカウント から表示またはアクセスすることはできません。2019 年 3 月 20 日より後に導入されたリージョンを使用するには、そのリージョンを有効にする必要があります。詳細については、「AWS Account Management リファレンスガイド」の「アカウントで使用できる AWS リージョンを指定する」を参照してください。

Code	名前	オプトインステータス
us-east-2	米国東部 (オハイオ)	不要
us-east-1	米国東部 (バージニア)	不要
us-west-1	米国西部 (北カリフォルニア)	不要
us-west-2	米国西部 (オレゴン)	不要
af-south-1	アフリカ (ケープタウン)	必須
ap-east-1	アジアパシフィック (香港)	必須
ap-south-2	アジアパシフィック (ハイデラバード)	必須
ap-southeast-3	アジアパシフィック (ジャカルタ)	必須
ap-southeast-4	アジアパシフィック (メルボルン)	必須
ap-south-1	アジアパシフィック (ムンバイ)	不要
ap-northeast-3	アジアパシフィック (大阪)	不要
ap-northeast-2	アジアパシフィック (ソウル)	不要
ap-southeast-1	アジアパシフィック (シンガポール)	不要
ap-southeast-2	アジアパシフィック (シドニー)	不要
ap-northeast-1	アジアパシフィック (東京)	不要
ca-central-1	カナダ (中部)	不要
ca-west-1	カナダ西部 (カルガリー)	必須

Code	名前	オプトインステータス
eu-central-1	欧州 (フランクフルト)	不要
eu-west-1	欧州 (アイルランド)	不要
eu-west-2	欧州 (ロンドン)	不要
eu-south-1	欧州 (ミラノ)	必須
eu-west-3	欧州 (パリ)	不要
eu-south-2	欧州 (スペイン)	必須
eu-north-1	欧州 (ストックホルム)	不要
eu-central-2	欧州 (チューリッヒ)	必須
il-central-1	イスラエル (テルアビブ)	必須
me-south-1	中東 (バーレーン)	必須
me-central-1	中東 (アラブ首長国連邦)	必須
sa-east-1	南米 (サンパウロ)	不要

詳細については、AWS グローバルインフラストラクチャを参照してください。

リージョンごとのアベイラビリティーゾーンの数とマッピングは、リージョンごとに AWS アカウント 間で異なる場合があります。アカウントで使用可能なアベイラビリティーゾーンのリストを取得するには、Amazon EC2 コンソールまたはコマンドラインインターフェイスを使用できます。詳細については、「リージョンの説明」を参照してください。

リージョンとエンドポイント

コマンドラインインターフェイスまたは API アクションを使用してインスタンスを操作するときは、そのリージョンエンドポイントを指定する必要があります。Amazon EC2 のリージョンとエンドポイントの詳細については、「Amazon Web Services 全般のリファレンス」の「Amazon EC2 エンドポイントとクォータ」を参照してください。

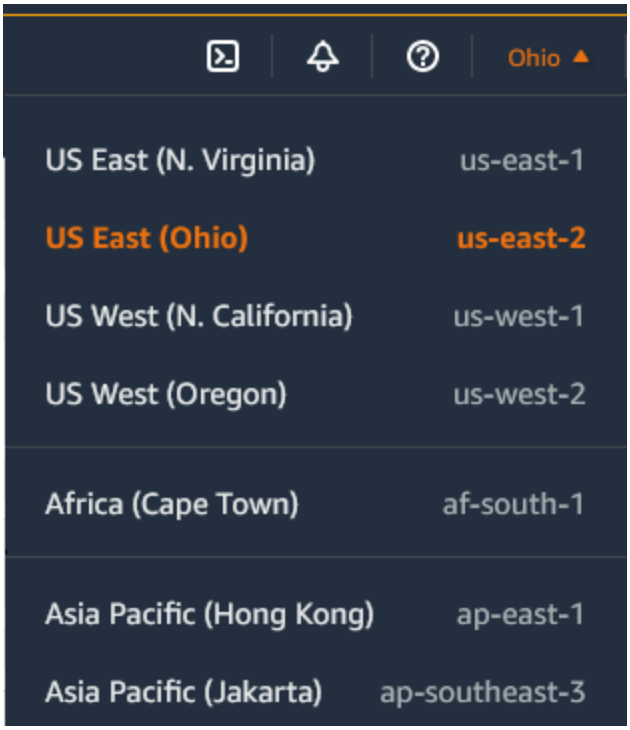
AWS GovCloud (米国西部) のエンドポイントとプロトコルの詳細については、「AWS GovCloud (US) ユーザーガイド」の「Service Endpoints」(サービスエンドポイント) を参照してください。

リージョンの説明

Amazon EC2 コンソールまたはコマンドラインインターフェイスを使用して、アカウントで使用できるリージョンを確認できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

コンソールを使用してリージョンを検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択します。



3. 選択したリージョンの EC2 リソースは、[リソース] セクションの [EC2 ダッシュボード] に表示されます。

AWS CLI を使用してリージョンを検索するには

次のように describe-regions コマンドを使用して、アカウントに対して有効になっているリージョンを記述します。

```
aws ec2 describe-regions
```

アカウントに対して無効になっているリージョンも含めてすべてのリージョンを記述するには、次のように --all-regions オプションを追加します。

```
aws ec2 describe-regions --all-regions
```

リージョンの表示名を取得

AWS Systems Manager パラメータストアを使用して、リージョンの表示名を確認できます。各リージョンには、以下のパスにパブリックパラメータがあります。

```
/aws/service/global-infrastructure/regions/region-code
```

リージョンのパブリックパラメータには以下が含まれます。

- /aws/service/global-infrastructure/regions/*region-code*/domain
- /aws/service/global-infrastructure/regions/*region-code*/geolocationCountry
- /aws/service/global-infrastructure/regions/*region-code*/geolocationRegion
- /aws/service/global-infrastructure/regions/*region-code*/longName
- /aws/service/global-infrastructure/regions/*region-code*/partition

longName パラメータにはリージョンの表示名が含まれます。以下の get-parameters-by-path コマンドは、af-south-1 リージョンの表示名を返します。--query オプションを使用して、出力の範囲をリージョンの名前に限定します。Linux ではクエリ文字列を一重引用符で囲む必要があります。Windows コマンドプロンプトを使用してこのコマンドを実行するには、一重引用符を省略するか、二重引用符に変更してください。

Linux

```
aws ssm get-parameters-by-path \
    --path /aws/service/global-infrastructure/regions/af-south-1 \
    --query 'Parameters[?Name.contains(@,`longName`)].Value' \
    --output text
```

Windows

```
aws ssm get-parameters-by-path ^
    --path /aws/service/global-infrastructure/regions/af-south-1 ^
    --query "Parameters[?Name.contains(@,`longName`)].Value" ^
```

```
    --output text
```

以下は出力例です。

```
Africa (Cape Town)
```

詳細については、「AWS Systems Manager ユーザーガイド」の「Working with public parameters」を参照してください。

リソースのリージョンの指定

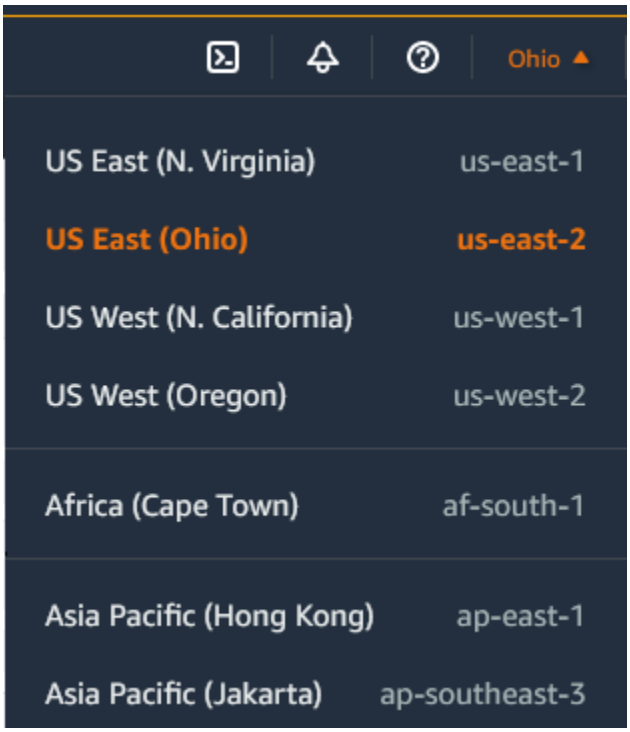
Amazon EC2 リソースを作成するたびに、リソースのリージョンを指定できます。リソースのリージョンは AWS Management Console またはコマンドラインを使用して指定できます。

考慮事項

一部の AWS リソースは、一部のリージョンで利用できない場合があります。インスタンスを起動する前に、該当するリージョンで必要なリソースを作成できることを確認してください。

コンソールを使用してリソースのリージョンを指定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、リージョンを選択します。



コマンドラインを使用してデフォルトのリージョンを指定するには

環境変数の値を、目的のリージョンエンドポイント (`https://ec2.us-east-2.amazonaws.com` など) に設定できます。

- `AWS_DEFAULT_REGION` (AWS CLI)
- `Set-AWSDefaultRegion` (AWS Tools for Windows PowerShell)

各コマンドで、`--region` (AWS CLI) または `-Region` (AWS Tools for Windows PowerShell) のコマンドラインオプションを使用することもできます。例えば、`--region us-east-2` と指定します。

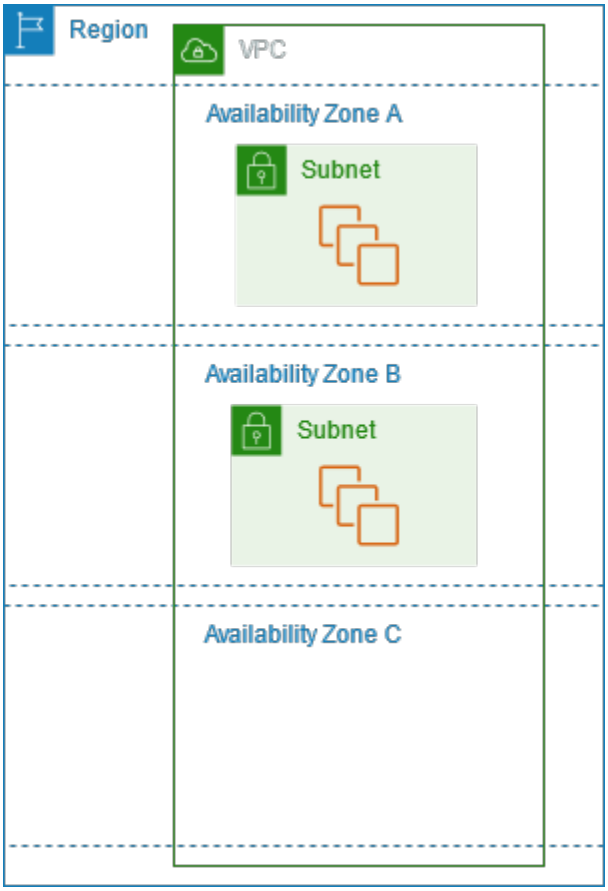
Amazon EC2 のエンドポイントの詳細については、「AWS 全般のリファレンス」の「Amazon EC2 エンドポイントとクォータ」を参照してください。

アベイラビリティーゾーン

リージョンごとにアベイラビリティーゾーンと呼ばれる複数の独立した場所があります。アベイラビリティーゾーンのコードは、リージョンコードとそれに続く文字識別子です。例えば、`us-east-1a` と指定します。

インスタンスを起動するときに、リージョンと仮想プライベートクラウド (VPC) を選択し、いずれかのアベイラビリティーゾーンからサブネットを自ら選択するか、またはサブネットが選択されることを許可します。インスタンスを複数のアベイラビリティーゾーンに配布する場合は、1 つのインスタンスで障害が発生したら別のアベイラビリティーゾーンのインスタンスが要求を処理するように、アプリケーションを設計できます。また、Elastic IP アドレスを使用すると、あるアベイラビリティーゾーンのインスタンスの障害を、別のアベイラビリティーゾーンのインスタンスにアドレスをすばやく再マッピングすることによってマスクできます。

次の図は、AWS リージョン内の複数のアベイラビリティーゾーンを示しています。アベイラビリティーゾーン A とアベイラビリティーゾーン B にはそれぞれ 1 つのサブネットがあり、各サブネットにはインスタンスがあります。アベイラビリティーゾーン C にはサブネットがないため、このアベイラビリティーゾーンにインスタンスを起動することはできません。



アベイラビリティーゾーンが拡大すると、アベイラビリティーゾーンを拡張しにくくなる場合があります。その場合、ユーザーがアベイラビリティーゾーンに既にインスタンスを持っているのでない場合は、制約のあるアベイラビリティーゾーンでのインスタンスの起動を制限する場合があります。最終的に、制約のあるアベイラビリティーゾーンを新しいアカウントに対するアベイラビリティーゾーンのリストから削除することもあります。したがって、アカウントによってリージョン内で使用できるアベイラビリティーゾーンの数が異なる場合があります。

コンテンツ

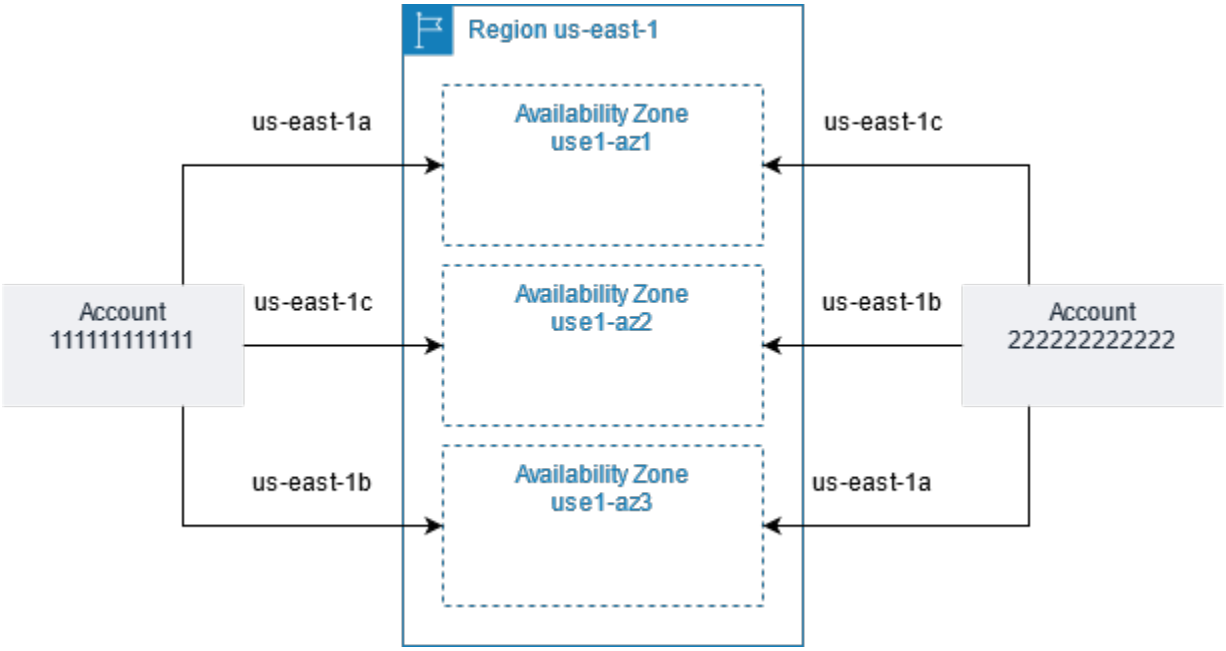
AZ ID

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 AWS アカウント のコードに個別にマッピングされます。例えば、AWS ア

カウント のアベイラビリティーゾーン us-east-1a の物理的な場所は、別の AWS アカウント の us-east-1a の場所と異なる可能性があります。

アカウント間でアベイラビリティーゾーンを調整するには、アベイラビリティーゾーンの一意で一貫性のある識別子である AZ ID を使用する必要があります。例えば、use1-az1 は、us-east-1 リージョンの AZ ID で、すべての AWS アカウント で同じ物理的な場所になります。アカウントの AZ ID を表示して、別のアカウントのリソースに対するリソースの物理的な場所を特定できます。例えば、AZ ID use1-az2 のアベイラビリティーゾーンにあるサブネットを別のアカウントと共有する場合、このサブネットは AZ ID が同じく use1-az2 であるアベイラビリティーゾーンのそのアカウントでも利用できます。

次の図は、アベイラビリティーゾーンのコードの AZ ID に対するマッピングが異なる 2 つのアカウントを示しています。



アベイラビリティーゾーンの説明

Amazon EC2 コンソールまたはコマンドラインインターフェイスを使用して、アカウントで使用できるアベイラビリティーゾーンを確認できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

Console を使用してアベイラビリティーゾーンを検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、リージョンを選択します。
3. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。

4. アベイラビリティーゾーンは、[サービスヘルス] ペインに一覧表示されます。

AWS CLI を使用してアベイラビリティーゾーンを検索するには

- 次のように describe-availability-zones コマンドを使用して、アカウントで有効な指定されたリージョン内のアベイラビリティーゾーンを記述します。

```
aws ec2 describe-availability-zones --region region-name
```

- オプトインのステータスに関係なしにアベイラビリティーゾーンを表示するには、次のように describe-availability-zones コマンドを使用します。

```
aws ec2 describe-availability-zones --all-availability-zones
```

アベイラビリティーゾーンでのインスタンスの起動

インスタンスを起動するときは、インスタンスと特定のお客様を近づけるリージョン、または法的要件や他の要件を満たすリージョンを選択します。個別のアベイラビリティーゾーンでインスタンスを起動することにより、1 つの場所で障害が発生しても、アプリケーションを保護することができます。

インスタンスを起動するときは、必要に応じて、使用するリージョン内のアベイラビリティーゾーンを指定できます。アベイラビリティーゾーンを指定しないと、アベイラビリティーゾーンが自動的に選択されます。初期インスタンスを起動する場合は、デフォルトのアベイラビリティーゾーンを受け入れることをお勧めします。これにより、システムの状態や利用可能なキャパシティーに基づいて、最適なアベイラビリティーゾーンを選択できます。追加のインスタンスを起動する場合、アベイラビリティーゾーンを指定するのは、新しいインスタンスを実行中のインスタンスと近づけるか、分離することが必要な場合に限ります。

別のアベイラビリティーゾーンへのインスタンスの移行

必要に応じて、アベイラビリティーゾーン間でインスタンスを移行できます。例えば、インスタンスのインスタンスタイプを変更しようとしたときに、現在のアベイラビリティーゾーンで新しいインスタンスタイプのインスタンスを起動できない場合は、新しいインスタンスタイプの容量を持つアベイラビリティーゾーンにインスタンスを移行できます。

移行プロセスは、次の作業を伴います。

- 元のインスタンスからの AMI の作成

- 新しいアベイラビリティーゾーンでのインスタンスの起動
- 新しいインスタンスの設定の更新 (次の手順で示します)

別のアベイラビリティーゾーンにインスタンスを移行するには

1. インスタンスから AMI を作成します。手順は、オペレーティングシステムとインスタンスのルートデバイスボリュームの種類によって異なります。詳細については、使用しているオペレーティングシステムとルートデバイスボリュームに対応するドキュメントを参照してください。
   - Amazon EBS-backed Linux AMI を作成する
   - instance store-backed Linux AMI を作成する
   - カスタム Windows AMI を作成する
2. インスタンスのプライベート IPv4 アドレスを維持する必要がある場合は、現在のアベイラビリティーゾーンのサブネットを削除してから、新しいアベイラビリティーゾーンに元のサブネットと同じ IPv4 アドレス範囲のサブネットを作成する必要があります。サブネットを削除する前に、その中のすべてのインスタンスを終了する必要があります。したがって、サブネットのすべてのインスタンスから AMI を作成し、現在のサブネットのすべてのインスタンスを新しいサブネットに移動できるようにする必要があります。
3. 新しいアベイラビリティーゾーンまたはサブネットを指定して、作成した AMI からインスタンスを起動します。インスタンスタイプは、元のインスタンスと同じにすることも、新しいインスタンスタイプを選択することもできます。詳細については、アベイラビリティーゾーンでのインスタンスの起動を参照してください。
4. 元のインスタンスに Elastic IP アドレスが関連付けられていた場合は、それを新しいインスタンスに関連付けます。詳細については、Elastic IP アドレスの関連付けを解除するを参照してください。
5. 元のインスタンスが リザーブドインスタンス の場合は、予約のアベイラビリティーゾーンを変更します。(また、インスタンスタイプも変更する場合は、予約のインスタンスタイプも変更できます)。詳細については、変更リクエストの送信を参照してください。
6. (オプション) 元のインスタンスを終了します。詳細については、インスタンスの終了を参照してください。

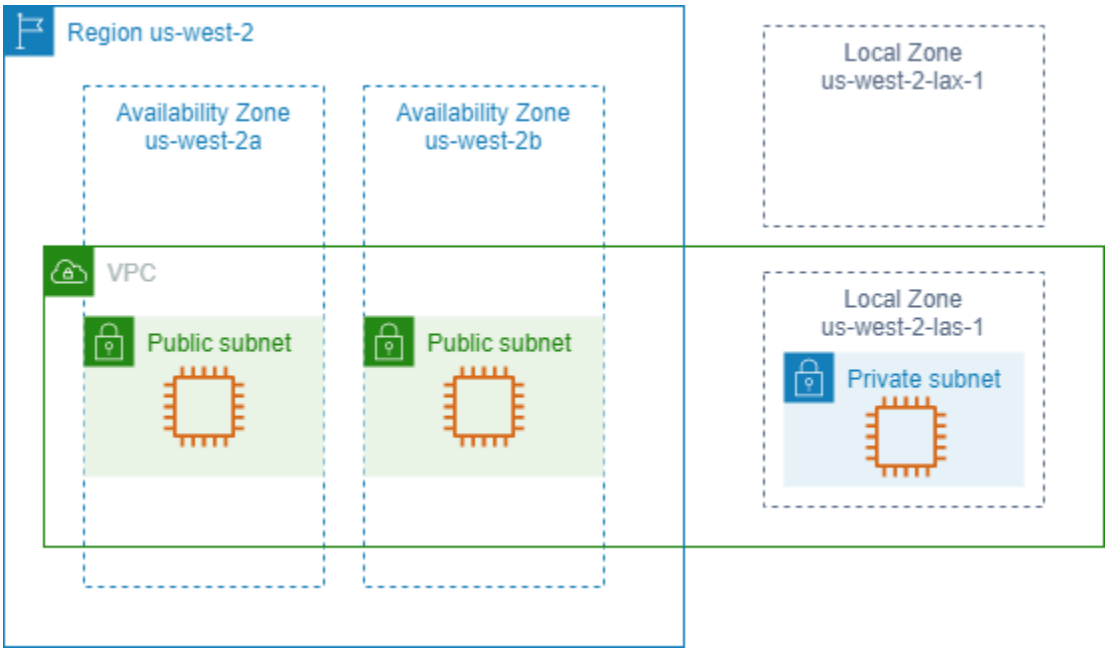
Local Zones

ローカルゾーンは、地理的にユーザーに近い場所に位置する AWS リージョンを拡張したものです。Local Zones はインターネットへの独自の接続を持ち、AWS Direct Connect をサポートしてい

るため、Local Zonesで作成されたリソースは、低レイテンシーの通信でローカルユーザーにサービスを提供できます。詳細については、AWS Local Zonesを参照してください。

ローカルゾーンのコードは、そのリージョンコードの後に、物理的な場所を示す識別子が続きます。例えば、ロサンゼルスの us-west-2-lax-1 です。

次の図は、AWS リージョン us-west-2、そのアベイラビリティーゾーンのうちの 2 つ、およびそのローカルゾーンのうちの 2 つを示しています。VPC は、アベイラビリティーゾーンといずれかのローカルゾーンにまたがっています。VPC 内の各ゾーンには 1 つのサブネットがあり、各サブネットにはインスタンスがあります。



ローカルゾーンを使用するには、最初にそれを有効にする必要があります。詳細については、the section called “Local Zones へのオプトイン”を参照してください。次に、ローカルゾーン内にサブネットを作成します。最後に、インスタンスなどのローカルゾーンサブネットでリソースを起動して、アプリケーションとユーザーを近づけます。

コンテンツ

利用可能な Local Zones

Amazon EC2 コンソールまたはコマンドラインインターフェイスを使用して、アカウントで利用できるローカルゾーンを確認できます。詳細な一覧については、「AWS Local Zones ロケーション」を参照してください。

コンソールを使用して Local Zones を検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、親リージョンを選択します。
3. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
4. ページの右上で、[アカウントの属性]、[ゾーン] の順に選択します。

AWS CLI を使用してローカルゾーンを検索するには

次のように describe-availability-zones コマンドを使用して、指定したリージョン内のすべてのローカルゾーンを、有効でないものも含めて、表示します。有効にしたローカルゾーンのみを表示するには、`--all-availability-zones` オプションを省略します。

```
aws ec2 describe-availability-zones --region region-name --filters Name=zone-
type,Values=local-zone --all-availability-zones
```

Local Zones へのオプトイン

リソースまたはサービスのLocal Zonesを指定する前に、Local Zonesにオプトインする必要があります。

考慮事項

一部の AWS リソースは、一部のリージョンで利用できない場合があります。特定のLocal Zonesでインスタンスを起動する前に、目的のリージョンまたはLocal Zonesで必要なリソースを作成できることを確認してください。各ローカルゾーンでサポートされているサービスのリストについては、「AWS Local Zones の機能」を参照してください。

コンソールを使用して Local Zones へオプトインするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ページの左上で、[新しい EC2 エクスペリエンス] を選択します。このタスクを実行するのに、コンソールの古いエクスペリエンスを使用することはできません。
3. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、親リージョンを選択します。
4. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
5. ページの右上で、[アカウントの属性]、[ゾーン] の順に選択します。
6. ローカルゾーンを有効にするには、[Manage] (管理) を選択します。
7. [ゾーングループ] で [有効] をクリックします。
8. [ゾーングループの更新] をクリックします。

AWS CLI を使用してローカルゾーンにオプトインするには

modify-availability-zone-group コマンドを使用します。

ローカルゾーンでのインスタンスの起動

インスタンスの起動時に、Local Zones 内のサブネットを指定します。また、ネットワークボーダーグループから次の IP アドレスも割り当てます。ネットワークボーダーグループは、AWS が IP アドレスをアドバタイズするアベイラビリティーゾーン、Local Zones、または Wavelength Zones の一意のセットです (例: `us-west-2-lax-1a`)。

ネットワークボーダーグループから次の IP アドレスを割り当てることができます。

- Amazon が提供する Elastic IPv4 アドレス
- Amazon が提供する IPv6 VPC アドレス (ロサンゼルスゾーンのみで利用可能)

Local Zones でインスタンスを起動する方法の詳細については、「AWS Local Zones ユーザーガイド」の「AWS Local Zones 入門」を参照してください。

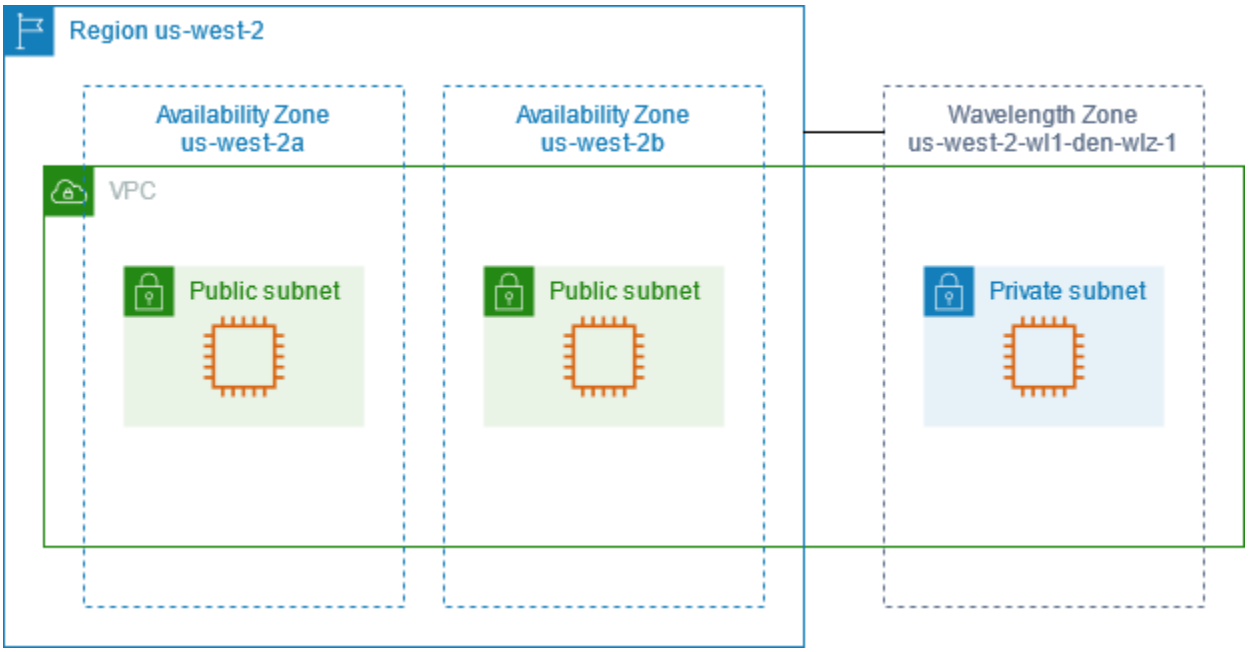
Wavelength Zone

AWS Wavelength を使用することで、デベロッパーは、モバイルデバイスおよびエンドユーザー向けに、非常にレイテンシーが低いアプリケーションを構築できます。Wavelength は、標準の AWS コンピューティングおよびストレージサービスを通信事業者の 5G ネットワークのエッジにデプロイします。デベロッパーは、Virtual Private Cloud (VPC) を 1 つ以上の Wavelength Zones に拡張し、Amazon EC2 インスタンスなどの AWS リソースを使用して、超低レイテンシーやリージョンの AWS サービスへの接続を必要とするアプリケーションを実行できます。

Wavelength Zone は、Wavelength インフラストラクチャをデプロイする先のキャリアロケーション内の独立したゾーンです。Wavelength Zone は、リージョンに関連付けられています。Wavelength Zone は、リージョンの論理的な拡張であり、リージョンの制御プレーンによって管理されます。

Wavelength Zone のコードは、そのリージョンコードの後に、物理的な場所を示す識別子が続きます。例えば、ボストンの `us-east-1-wl1-bos-wlz-1` です。

次の図は、AWS リージョン `us-west-2`、そのアベイラビリティーゾーンのうちの 2 つ、および Wavelength Zone を示しています。VPC はアベイラビリティーゾーンと Wavelength Zone にまたがっています。VPC 内の各ゾーンには 1 つのサブネットがあり、各サブネットにはインスタンスがあります。



Wavelength Zone を使用するには、まずゾーンにオプトインする必要があります。詳細については、the section called “Wavelength Zone の有効化”を参照してください。次に、Wavelength Zone にサブネットを作成します。最後に、Wavelength Zone のサブネットでリソースを起動し、アプリケーションとエンドユーザーを近づけます。

Wavelength Zone は、すべてのリージョンで利用できるわけではありません。Wavelength Zone をサポートするリージョンについては、AWS Wavelength デベロッパーガイドの利用可能な Wavelength Zoneを参照してください。

コンテンツ

- Wavelength Zone でのインスタンスの起動

Wavelength Zone の説明

Amazon EC2 コンソールまたはコマンドラインインターフェイスを使用して、アカウントで利用できる Wavelength Zone を確認できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

コンソールを使用して Wavelength Zone を検索するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、リージョンを選択します。
3. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
4. ページの右上で、[アカウントの属性]、[ゾーン] の順に選択します。

AWS CLI を使用して Wavelength Zone を検索するには

- 次のように describe-availability-zones コマンドを使用して、アカウントで有効な指定したリージョン内の Wavelength Zone を表示します。

```
aws ec2 describe-availability-zones --region region-name
```

- オプトインのステータスに関係なしに Wavelength Zone を表示するには、次のように describe-availability-zones コマンドを使用します。

```
aws ec2 describe-availability-zones --all-availability-zones
```

Wavelength Zone の有効化

リソースまたはサービスの Wavelength Zone を指定する前に、Wavelength Zone にオプトインする必要があります。

考慮事項

- 一部の AWS リソースは、リージョンによっては利用できません。特定の Wavelength Zone でインスタンスを起動する前に、目的のリージョンまたは Wavelength Zone で必要なリソースを作成できることを確認してください。

コンソールを使用して Wavelength Zone にオプトインするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ページの左上で、[新しい EC2 エクスペリエンス] を選択します。このタスクを実行するのに、コンソールの古いエクスペリエンスを使用することはできません。
3. ナビゲーションバーで、[Regions] (リージョン) セレクタを選択し、リージョンを選択します。
4. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
5. ページの右上で、[アカウントの属性]、[ゾーン] の順に選択します。
6. Wavelength Zone で、Wabelenth Zone の 管理 を選択します。
7. [Enable] (有効化) を選択します。
8. [ゾーングループの更新] をクリックします。

AWS CLI を使用して Wavelength Zone を有効にするには

modify-availability-zone-group コマンドを使用します。

Wavelength Zone でのインスタンスの起動

インスタンスの起動時に、Wavelength Zone にあるサブネットを指定できます。また、ネットワークボーダーグループからキャリア IP アドレスを割り当てます。これは、AWS が IP アドレスをアドバタイズするアベイラビリティーゾーン、Local Zones、または Wavelength Zones の一意のセットです (例: `us-east-1-wl1-bos-wlz-1` など)。

Wavelength Zone でインスタンスを起動する方法については、AWS Wavelength デベロッパーガイドのAWS Wavelength の開始方法を参照してください。

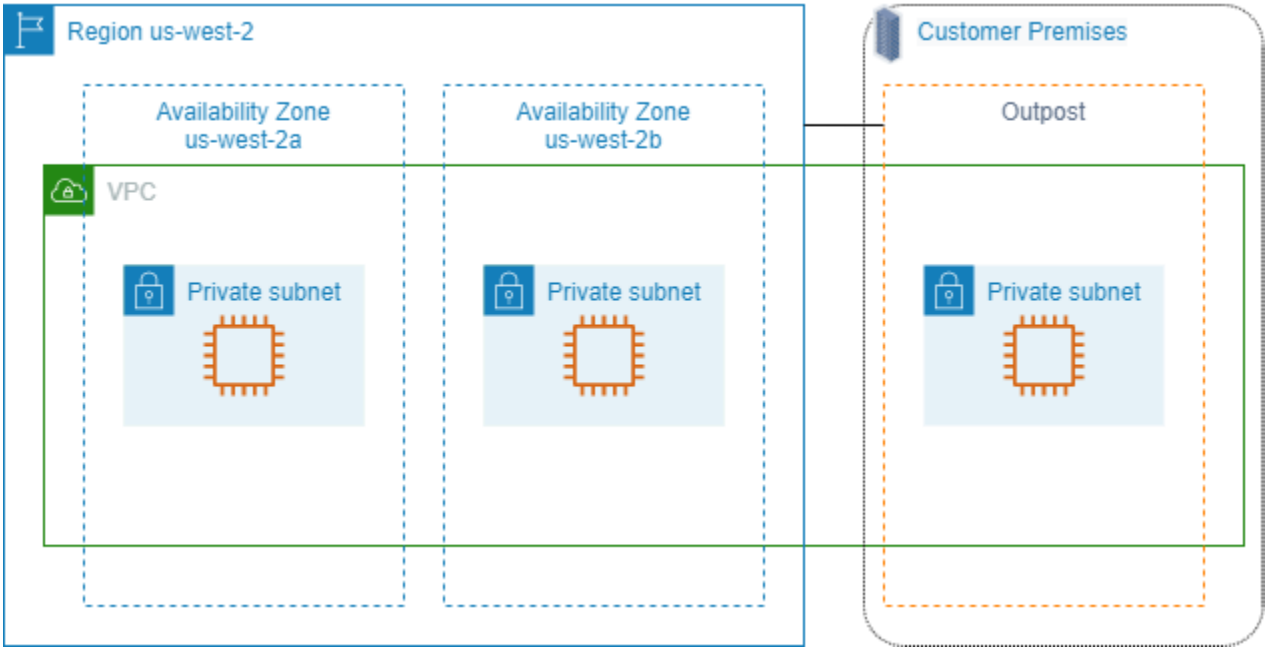
AWS Outposts

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをお客様のオンプレミスまで拡張するフルマネージドサービスです。AWS は、AWS Outposts マネージドインフラストラクチャへのローカルアクセスを提供することで、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築して実行できるようにします。同時に、コンピューティングとストレージのローカルリソースを使用して、レイテンシーを短縮し、ローカルのデータ処理ニーズに対応します。

Outpost とは、お客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS は、AWS リージョンの一部としてこのキャパシティーを運営、監

視、管理します。Outpost にサブネットを作成し、AWS リソースを作成したときにこれらのサブネットを指定します。Outpost サブネット内のインスタンスは、プライベート IP アドレスを使用して、AWS リージョン内の他のインスタンスと通信します。これらはすべて同じ VPC 内にあります。

次の図は、AWS リージョン us-west-2、そのアベイラビリティーゾーンのうちの 2 つ、および Outpost を示しています。VPC はアベイラビリティーゾーンと Outpost にまたがっています。Outpost は、オンプレミスの顧客データセンターにあります。VPC 内の各ゾーンには 1 つのサブネットがあり、各サブネットにはインスタンスがあります。



AWS Outposts の使用を開始するには、Outpost を作成し、Outpost 容量を注文する必要があります。Outpost の設定の詳細については、カタログを参照してください。Outpost 機器をインストールすると、Outpost で Amazon EC2 インスタンスを起動するときに、コンピューティング容量とストレージ容量を使用できます。

Outpost でのインスタンスの起動

作成した Outpost サブネットで EC2 インスタンスを起動できます。セキュリティグループは、アベイラビリティーゾーンサブネットのインスタンスと同様に、Outpost サブネットにある Elastic Network インスタンスのインバウンドトラフィックとアウトバウンドトラフィックを制御します。Outpost サブネットの EC2 インスタンスに接続するには、アベイラビリティーゾーンサブネットのインスタンスの場合と同様に、インスタンスの起動時にキーペアを指定できます。

Outpost ラック上のインスタンスのルートボリュームを 30 GiB 以下に制限することをお勧めします。AMI またはインスタンスのブロックデバイスマッピングでデータボリュームを指定し、追加の

ストレージを提供できます。ブートボリュームから未使用のブロックを削除するには、AWS パートナーネットワークブログのSparse EBS Volumeの構築方法を参照してください。

ルートボリュームの NVMe タイムアウトを増やすことをお勧めします。詳細については、「I/O オペレーションタイムアウト」を参照してください。

Outpost の作成方法の詳細については、「AWS Outposts ユーザーガイド」の「AWS Outposts の開始方法」を参照してください。

Outpost でのボリュームの作成

AWS Outposts は、ラックおよびサーバのフォームファクタを提供します。容量が Outpost ラックにある場合、作成した Outpost サブネットに EBS ボリュームを作成できます。ボリュームの作成時に、Outpost の Amazon リソースネーム (ARN) を指定します。

次の create-volume コマンドは、指定した Outpost に空の 50 GB ボリュームを作成します。

```
aws ec2 create-volume --availability-zone us-east-2a --outpost-arn arn:aws:outposts:us-east-2:123456789012:outpost/op-03e6fecad652a6138 --size 50
```

Amazon EBS gp2 ボリュームのサイズは、ボリュームをデタッチする必要なく動的に変更することができます。ボリュームをデタッチせずに変更する方法の詳細については、「EBS ボリュームへの変更のリクエスト」を参照してください。

Amazon EC2 インスタンスの IP アドレス指定

Amazon EC2 と Amazon VPC は、IPv4 と IPv6 の両方のアドレス設定プロトコルをサポートします。デフォルトでは、Amazon VPC は IPv4 アドレス設定プロトコルを使用します。この動作を無効にすることはできません。VPC の作成時には IPv4 CIDR ブロック (プライベート IPv4 アドレスの範囲) を指定する必要があります。必要に応じて、IPv6 CIDR ブロックを VPC に割り当て、そのブロックからサブネットのインスタンスに IPv6 アドレスを割り当てることができます。

コンテンツ

- インスタンスの IPv4 アドレスの操作
- インスタンスの IPv6 アドレスの操作
- 複数の IP アドレス
- EC2 インスタンスのホスト名
- リンクローカルアドレス

プライベート IPv4 アドレス

プライベート IPv4 アドレスは、インターネットから到達できない IP アドレスです。プライベート IPv4 アドレスは、同じ VPC 内のインスタンス間の通信に使用できます。プライベート IPv4 アドレスの標準および仕様については、RFC 1918 を参照してください。DHCP を使用してインスタンスにプライベート IPv4 アドレスが割り当てられます。

> **Note**
>
> RFC 1918 に指定されているプライベート IPv4 アドレスの範囲に含まれない、パブリックにルーティングできる CIDR ブロックを持つ VPC を作成できます。ただし、このドキュメントでプライベート IPv4 アドレス (またはプライベート IP アドレス) と言う場合は、VPC の IPv4 CIDR 範囲に含まれる IP アドレスを指します。

VPC サブネットは、次のいずれかのタイプです。

- IPv4 専用サブネット: IPv4 アドレスが割り当てられたこれらのサブネット内のリソースのみを作成できます。
- IPv6 専用サブネット: これらのサブネットには、IPv6 アドレスが割り当てられたリソースのみを作成できます。
- IPv4 および IPv6 サブネット: IPv4 または IPv6 アドレスのいずれかを割り当てて、これらのサブネットにリソースを作成できます。

EC2 インスタンスをIPv4 専用サブネットまたはデュアルスタック (IPv4 および IPv6) サブネットで起動すると、インスタンスはサブネットの IPv4 アドレス範囲からプライマリプライベート IP アドレスを受け取ります。詳細については、[Amazon VPC User Guide] (Amazon VPC ユーザーガイド) の[IP addressing] (IP アドレス指定) を参照してください。プライマリプライベート IP アドレスを指定しないでインスタンスを起動すると、サブネットの IPv4 範囲内で使用可能な IP アドレスが自動

的に選択されます。各インスタンスには、プライマリプライベート IPv4 アドレスが割り当てられたデフォルトのネットワークインターフェイス (eth0) があります。追加のプライベート IPv4 アドレス (セカンダリプライベート IPv4 アドレス) も指定できます。プライマリプライベート IP アドレスとは異なり、セカンダリプライベート IP アドレスは、別のインスタンスに割り当て直すことができます。詳細については、複数の IP アドレスを参照してください。

プライベート IPv4 アドレスは、プライマリアドレスまたはセカンダリアドレスを問わず、インスタンスが停止して起動、または休止して起動した際に、ネットワークインターフェイスに関連付けられたままになり、インスタンスを終了するとリリースされます。

パブリック IPv4 アドレス

パブリック IP アドレスは、インターネットから到達可能な IPv4 アドレスです。インスタンスとインターネット間で通信するには、パブリック アドレスを使用できます。

デフォルトの VPC でインスタンスを起動すると、デフォルトでパブリック IP アドレスが割り当てられます。デフォルト以外の VPC でインスタンスを起動するとき、サブネットには、そのサブネットで起動するインスタンスがパブリック IPv4 アドレスプールからパブリック IP アドレスを受け取るかどうかを決定する属性があります。デフォルトでは、デフォルト以外のサブネットで起動されたインスタンスにパブリック IP アドレスを割り当てません。

インスタンスがパブリック IP アドレスを割り当てられるかどうかを制御するには、以下の方法を使用します。

- サブネットのパブリック IP アドレス属性を変更する。詳細については、「Amazon VPC ユーザーガイド」の「サブネットのパブリック IPv4 アドレス指定属性の変更」を参照してください。
- 起動時にパブリック IP アドレス機能を有効または無効にする。これにより、サブネットのパブリック IP アドレス属性がオーバーライドされます。詳細については、インスタンス起動時のパブリック IPv4 アドレスの割り当てを参照してください。

パブリック IP アドレスは、Amazon のパブリック IPv4 アドレスプールからインスタンスに割り当てられ、お客様の AWS アカウントには関連付けられません。パブリック IP アドレスをインスタンスから割り当て解除すると、そのパブリック IPv4 アドレスはパブリック IP アドレスプールに戻され、再利用することはできません。

パブリック IP (IPv4) アドレスを、手動でインスタンスに関連付けたり、手動でインスタンスから割り当て解除することはできません。場合によって、パブリック IP アドレスはインスタンスからリリースされたり、新しいインスタンスに割り当てられたりします。

- インスタンスのパブリック IP アドレスが停止、休止または終了すると、インスタンスのパブリック IP アドレスがリリースされます。停止または休止状態のインスタンスは、起動時に、新しいパブリック IP アドレスを受け取ります。
- Elastic IP アドレスをこれに関連付けると、インスタンスのパブリック IP アドレスがリリースされます。Elastic IP アドレスをインスタンスから割り当て解除すると、そのインスタンスには新しいパブリック IP アドレスが送信されます。
- VPC 内のインスタンスのパブリック IP アドレスが既にリリースされている場合には、複数のネットワークインターフェイスがインスタンスにアタッチされていると、インスタンスに新しいパブリック IP アドレスは送信されません。
- インスタンスのパブリック IP アドレスがリリースされ、Elastic IP アドレスに関連付けられたセカンダリプライベート IP アドレスがある場合、インスタンスは新しいパブリック IP アドレスを受信しません。

必要に応じて、インスタンスに関連付けおよびインスタンスから関連付けできる永続的なパブリック IP アドレスが必要な場合は、Elastic IP アドレスを使用します。

動的 DNS を使用して既存の DNS 名を新しいインスタンスのパブリック IP アドレスにマッピングした場合、その IP アドレスがインターネット内に伝達されるまでに最大 24 時間かかることがあります。その結果、新しいインスタンスはトラフィックを受信せず、終了したインスタンスがリクエストの受信を継続することがあります。この問題を解決するには、Elastic IP アドレスを使用します。独自の Elastic IP アドレスを割り当てて、それをインスタンスに関連付けることができます。詳細については、「Elastic IP アドレス」を参照してください。

> **Note**
>
> - AWS では、実行中のインスタンスに関連付けられているパブリック IPv4 アドレスと Elastic IP アドレスを含む、すべてのパブリック IPv4 アドレスに対して料金が課されます。詳細については、「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。
> - インスタンスがパブリック NAT IP アドレスを使用して他のインスタンスにアクセスする場合、アクセス先のインスタンスが同じリージョンにあるかどうかによって、リージョンデータ転送またはインターネットデータ転送に対して課金されます。

Elastic IP アドレス (IPv4)

Elastic IP アドレスは、アカウントに割り当てることができるパブリック IPv4 アドレスです。このアドレスとインスタンスを関連付けたり、その関連付けを解除したりできます。アドレスはアカウントに割り当てられ、割り当てを解除するまでアカウントに残ります。Elastic IP アドレスとその使用方法の詳細については、Elastic IP アドレスを参照してください。

IPv6 に対する Elastic IP アドレスはサポートされていません。

IPv6 アドレス

必要に応じて、IPv6 CIDR ブロックを VPC と関連付けることができます。また、IPv6 CIDR ブロックをサブネットと関連付けることができます。VPC の IPv6 CIDR ブロックは、Amazon の IPv6 アドレスのプールから自動的に割り当てられます。独自にアドレス範囲を選択することはできません。詳細については、「Amazon VPC ユーザーガイド」の次のトピックを参照してください。

- VPC とサブネットの IP アドレス指定
- VPC に IPv6 CIDR ブロックを追加する
- サブネットに IPv6 CIDR ブロックを追加する

IPv6 アドレスはグローバルに一意であり、プライベートのまま、またはインターネット経由で到達可能になるように設定できます。IPv6 CIDR ブロックが VPC およびサブネットと関連付けられていて、以下のいずれかに該当する場合、インスタンスには IPv6 アドレスが割り当てられます。

- 起動時にサブネットからインスタンスに IPv6 アドレスが自動的に割り当てられるように設定されている。詳細については、サブネットの IPv6 アドレス指定属性の変更を参照してください。
- 起動時に IPv6 アドレスをインスタンスに割り当てる。
- 起動後に IPv6 アドレスをインスタンスのプライマリネットワークインターフェイスに割り当てる。
- 起動後に IPv6 アドレスを同じサブネットのネットワークインターフェイスに割り当て、そのネットワークインターフェイスをインスタンスにアタッチする。

起動時にインスタンスに IPv6 アドレスが割り当てられると、そのアドレスはインスタンスのプライマリネットワークインターフェイス (eth0) と関連付けられます。インスタンスのプライマリネットワークインターフェイス (eth0) の IPv6 アドレスは、次の方法で管理できます。

- ネットワークインターフェイスへ IPv6 アドレスを割り当て/割り当て解除します。ネットワークインターフェイスに割り当てることができる IPv6 アドレスの数と、インスタンスにアタッチできるネットワークインターフェイスの数は、インスタンスタイプごとに異なります。詳細については、「各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数」を参照してください。
- プライマリ IPv6 アドレスを有効にします。プライマリ IPv6 アドレスにより、インスタンスまたは ENI へのトラフィックの中断を回避できます。詳細については、「ネットワークインターフェイスの作成」または「IP アドレスの管理」を参照してください。

IPv6 アドレスは、インスタンスの停止して起動、または休止して起動する際には保持され、インスタンスの終了時にリリースされます。IPv6 アドレスは、別のネットワークインターフェイスに割り当てられている間は再割り当てできません。最初に割り当てを解除する必要があります。

インスタンスのサブネットのルーティングを制御するか、セキュリティグループとネットワーク ACL ルールを使用することで、IPv6 アドレスを介してインスタンスに接続できるかどうかを制御できます。詳細については、「Amazon VPC User Guide」のインターネットワークトラフィックのプライバシーを参照してください。

予約済み IPv6 アドレスの範囲については、「IANA IPv6 Special-Purpose Address Registry」と「RFC4291」を参照してください。

インスタンスの IPv4 アドレスの操作

インスタンスを起動するときに、パブリック IPv4 アドレスをインスタンスに割り当てることができます。インスタンスの IPv4 アドレスをコンソールに表示するには、Instances (インスタンス) ページまたは [Network Interfaces] (ネットワークインターフェイス) ページを使用します。

コンテンツ

IPv4 アドレスの表示

Amazon EC2 コンソールを使用して、インスタンスのプライベート IPv4 アドレスおよびパブリック IPv4 アドレスを表示できます。また、インスタンスメタデータを使用して、インスタンス内からインスタンスのパブリック IPv4 アドレスとプライベート IPv4 アドレスを確認することもできます。詳細については、インスタンスメタデータとユーザーデータを参照してください。

パブリック IPv4 アドレスは、コンソールのネットワークインターフェイスのプロパティとして表示されますが、NAT によってプライマリプライベート IPv4 アドレスにマッピングされます。したがって、インスタンスのネットワークインターフェイスのプロパティを、例えば `ifconfig` (Linux) または `ipconfig` (Windows) を通して調べてみると、パブリック IPv4 アドレスは表示されていません。インスタンスからインスタンスのパブリック IPv4 アドレスを判断するには、インスタンスメタデータを使用します。

コンソールを使用してインスタンスの IPv4 アドレスを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. [Networking (ネットワーク)] タブでは、次の情報が表示されます。
   - パブリック IPv4 アドレス — パブリック IPv4 アドレス。Elastic IP アドレスをインスタンスまたはプライマリネットワークインターフェイスに関連付けた場合、これが Elastic IP アドレスになります。
   - プライベート IPv4 アドレス — プライベート IPv4 アドレス。
   - セカンダリプライベート IPv4 アドレス — 任意のセカンダリプライベート IPv4 アドレス。
4. 詳細情報を表示するには、[ネットワーキング] タブで、プライマリネットワークインターフェイスの ID を選択して [ネットワークインターフェイス] ページを開き、ネットワークインターフェイスの ID を選択して詳細ページを開きます。

コマンドラインを使用してインスタンスの IPv4 アドレスを表示するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、「Amazon EC2 へのアクセス」を参照してください。

- describe-instances (AWS CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

インスタンスのメタデータを使用してインスタンスの IPv4 アドレスを確認するには

1. インスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。
2. プライベート IP アドレスにアクセスするには、次のコマンドを使用します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
 "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/local-ipv4
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/local-ipv4
```

3. パブリック IP アドレスにアクセスするには、次のコマンドを使用します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
 "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/public-ipv4
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-ipv4
```

インスタンスに Elastic IP アドレスが関連付けられている場合、返される値は Elastic IP アドレスの値です。

インスタンス起動時のパブリック IPv4 アドレスの割り当て

各サブネットに、そのサブネット内で起動されるインスタンスにパブリック IP アドレスが割り当てられるかどうかを決定する属性があります。デフォルトでは、デフォルト以外のサブネットではこの属性が false に設定されており、デフォルトのサブネットではこの属性が true に設定されています。インスタンスを起動する場合、パブリック IPv4 アドレス指定機能を使用してインスタンスにパブリック IPv4 アドレスを割り当てるかどうかを制御することもできます。サブネットの IP アドレス指定属性のデフォルトの動作をオーバーライドできます。パブリック IPv4 アドレスは、Amazon のパブリック IPv4 アドレスプールから割り当てられ、デバイスインデックス eth0 を持つネットワー

クインターフェイスに割り当てられます。この機能は、インスタンス起動時の特定の条件により異なります。

考慮事項

- 起動後に、インスタンスからパブリック IP アドレスの割り当てを手動で解除することはできません。ただし、特定の場合に、アドレスが自動的にリリースされ、その後再利用できなくなります。詳細については、パブリック IPv4 アドレスを参照してください。お客様の意志で関連付けしたり関連付けを解除したりできる永続的なパブリック IP アドレスを必要とする場合は、起動してからインスタンスに Elastic IP アドレスを割り当てます。詳細については、Elastic IP アドレスを参照してください。
- 複数のネットワークインターフェイスを指定した場合、パブリック IP アドレスを自動割り当てすることはできません。さらに、eth0 のように既存のネットワークインターフェイスを指定すると、パブリック IP の自動割り当て機能を使用してサブネット設定をオーバーライドすることはできません。
- パブリック IP アドレス機能は起動時にのみ使用できます。ただし、起動時にパブリック IP アドレスをインスタンスに割り当てるかどうかにかかわらず、起動後に Elastic IP アドレスをインスタンスに関連付けることができます。詳細については、Elastic IP アドレスを参照してください。サブネットのパブリック IPv4 アドレス指定動作を変更することもできます。詳細については、「サブネットの IPv4 アドレス指定属性の変更」を参照してください。

コンソールを使用してインスタンス起動時にパブリック IPv4 アドレスを割り当てるには

手順に従ってインスタンスを起動し、[Network Settings] (ネットワーク設定) を設定するときに、[Auto-assign Public IP] (パブリック IP を自動的に割り当てる) オプションを選択します。

コマンドラインを使用してパブリック IP アドレス指定機能を有効または無効にするには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- run-instances コマンド (AWS CLI) で `--associate-public-ip-address` または `--no-associate-public-ip-address` オプションを使用します。
- New-EC2Instance コマンド (AWS Tools for Windows PowerShell) で `-AssociatePublicIp` パラメータを使用します。

インスタンスの IPv6 アドレスの操作

インスタンスに割り当てられた IPv6 アドレスを表示したり、インスタンスにパブリック IPv6 アドレスを割り当てたり、インスタンスから IPv6 アドレスの割り当てを解除したりできます。これらのアドレスは、[Instances] (インスタンス) ページまたは [Network Interfaces] (ネットワークインターフェイス) ページを使用してコンソールに表示できます。

コンテンツ

IPv6 アドレスの表示

Amazon EC2 コンソール、AWS CLI、インスタンスメタデータを使用して、インスタンスの IPv6 アドレスを表示できます。

コンソールを使用してインスタンスの IPv6 アドレスを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択します。
4. [Networking] (ネットワーキング) タブで、[IPv6 addresses] (IPv6 アドレス) を見つけます。

コマンドラインを使用してインスタンスの IPv6 アドレスを表示するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、「Amazon EC2 へのアクセス」を参照してください。

- describe-instances (AWS CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

インスタンスメタデータを使用してインスタンスの IPv6 アドレスを表示するには

1. インスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。

2. 次のコマンドを使用して IPv6 アドレスを表示します (http://169.254.169.254/latest/meta-data/network/interfaces/macs/ から MAC アドレスを取得できます)。

   IMDSv2

   ```
   [ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
    "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
   meta-data/network/interfaces/macs/mac-address/ipv6s
   ```

   IMDSv1

   ```
   [ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/network/interfaces/
   macs/mac-address/ipv6s
   ```

インスタンスへの IPv6 アドレスの割り当て

VPC とサブネットに IPv6 CIDR ブロックが関連付けられている場合は、起動時または起動後に IPv6 アドレスをインスタンスに割り当てることができます。IPv6 アドレスは、サブネットの IPv6 アドレス範囲から割り当てられ、eth0 のデバイスインデックスを持つネットワークインターフェイスに割り当てられます。

インスタンス起動時に IPv6 アドレスを割り当てるには

手順に従ってインスタンスを起動し、[Network Settings] (ネットワーク設定) を設定するときに、[Auto-assign IPv6 IP] (IPv6 IP を自動的に割り当てる) オプションを選択します。

起動後に IPv6 アドレスをインスタンスに割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択後、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv6 addresses] (IPv6 アドレス) で、[Assign new IP address] (新しい IP アドレスの割り当て) を選択します。サブネットの範囲から IPv6 アドレスを入力します。また、フィールドを空白のままにすると Amazon によって IPv6 アドレスが自動的に選択されます。
5. [Save] を選択します。

コマンドラインを使用して IPv6 アドレスを割り当てるには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- run-instances コマンド (AWS CLI) で `--ipv6-addresses` オプションを使用する
- New-EC2Instance コマンド (AWS Tools for Windows PowerShell) で `-NetworkInterface` の `Ipv6Addresses` プロパティを使用する
- assign-ipv6-addresses (AWS CLI)
- Register-EC2Ipv6AddressList (AWS Tools for Windows PowerShell)

インスタンスからの IPv6 アドレスの割り当て解除

IPv6 アドレスは、インスタンスからいつでも割り当て解除できます。

コンソールを使用してインスタンスから IPv6 アドレスを割り当て解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択後、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv6 addresses] (IPv6 アドレス) で、IPv6 アドレスの横にある [Unassign] (割り当て解除) を選択します。
5. [Save] を選択します。

コマンドラインを使用してインスタンスから IPv6 アドレスを割り当て解除するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- unassign-ipv6-addresses (AWS CLI)
- Unregister-EC2Ipv6AddressList (AWS Tools for Windows PowerShell)

複数の IP アドレス

インスタンスに複数のプライベート IPv4 および IPv6 アドレスを指定できます。インスタンスに指定できるネットワークインターフェイスとプライベート IPv4 および IPv6 アドレスの数は、インス

タンスタイプによって異なります。詳細については、各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数を参照してください。

次のような場合、複数の IP アドレスを VPC 内のインスタンスに割り当てると便利です。

- 1 つのサーバーで複数の SSL 証明書を使用し、各インターフェイスに各 IP アドレスに割り当てることで、1 つのサーバーで複数のウェブサイトをホストする。
- 各ネットワークインターフェイス用に複数の IP アドレスを持つネットワークアプライアンス (ファイアウォールやロードバランサーなど) を運用する。
- インスタンスでエラーが発生した場合に、セカンダリ IP アドレスをスタンバイインスタンスに再割り当てすることによって、内部トラフィックをスタンバイインスタンスにリダイレクトする。

コンテンツ

複数の IP アドレスを使用する方法

次の一覧は、ネットワークインターフェイスで複数の IP アドレスを使用する方法の説明です。

- セカンダリプライベート IPv4 アドレスをネットワークインターフェイスに割り当てることができます。
- IPv6 CIDR ブロックが関連付けられているサブネット内のネットワークインターフェイスに複数の IPv6 アドレスを割り当てることができます。
- ネットワークインターフェイスのサブネットの IPv4 CIDR ブロック範囲からセカンダリ IPv4 アドレスを選択する必要があります。
- ネットワークインターフェイスのサブネットの IPv6 CIDR ブロック範囲から IPv6 アドレスを選択する必要があります。
- セキュリティグループを関連付けるのは、個々の IP アドレスではなく、ネットワークインターフェイスです。そのため、ネットワークインターフェイスで指定した各 IP アドレスは、そのネットワークインターフェイスのセキュリティグループの対象です。
- 複数の IP アドレスは、実行中または停止したインスタンスにアタッチされたネットワークインターフェイスに割り当てたり、割り当て解除したりできます。

- ネットワークインターフェイスに割り当てられているセカンダリプライベート IPv4 アドレスは、明示的に許可された場合、別のネットワークインターフェイスに割り当て直すことができます。
- IPv6 アドレスは、最初に既存のネットワークインターフェイスから割り当て解除しない限り、別のネットワークインターフェイスに再割り当てすることはできません。
- コマンドラインツールまたは API を使用して複数の IP アドレスをネットワークインターフェイスに割り当てるときに、いずれかの IP アドレスを割り当てることができない場合、オペレーション全体が失敗します。
- プライマリプライベート IPv4 アドレス、セカンダリプライベート IPv4 アドレス、Elastic IP アドレス、および IPv6 アドレスは、セカンダリネットワークインターフェイスをインスタンスからデタッチしたり、インスタンスにアタッチしても、セカンダリネットワークインターフェイスへの割り当ては維持します。
- プライマリネットワークインターフェイスをインスタンスからデタッチすることはできませんが、プライマリネットワークインターフェイスのセカンダリプライベート IPv4 アドレスを別のネットワークインターフェイスに再割り当てすることはできます。

次の一覧は、Elastic IP アドレスで複数の IP アドレスを使用する方法の説明です (IPv4 のみ)。

- 各プライベート IPv4 アドレスを関連付けることができる Elastic IP アドレスは 1 つであり、逆に各 Elastic IP アドレスを関連付けることができるプライベート IPv4 アドレスは 1 つです。
- セカンダリプライベート IPv4 アドレスを別のインターフェイスに再割り当てした場合、セカンダリプライベート IPv4 アドレスと Elastic IP アドレスの関連付けは維持されます。
- セカンダリプライベート IPv4 アドレスとインターフェイスの割り当てを解除すると、関連付けられた Elastic IP アドレスとセカンダリプライベート IPv4 アドレスとの関連付けは自動的に解除されます。

複数の IPv4 アドレスの使用

セカンダリプライベート IPv4 アドレスは、インスタンスに割り当てたり、Elastic IPv4 アドレスと関連付けたり、割り当て解除したりできます。

タスク

- セカンダリプライベート IPv4 アドレスの割り当て
- セカンダリプライベート IPv4 アドレスを認識するようにインスタンスのオペレーティングシステムを設定する
- セカンダリプライベート IPv4 アドレスへの Elastic IP アドレスの割り当て

- セカンダリプライベート IPv4 アドレスの表示
- セカンダリプライベート IPv4 アドレスの割り当て解除

セカンダリプライベート IPv4 アドレスの割り当て

セカンダリプライベート IPv4 アドレスは、インスタンスの起動時または起動後に、インスタンスのネットワークインターフェイスに割り当てることができます。このセクションには、以下の手順が含まれます。

- インスタンスの起動時にセカンダリプライベート IPv4 アドレスを割り当てるには
- コマンドラインを使用して起動時にセカンダリ IPv4 アドレスを割り当てるには
- セカンダリプライベート IPv4 アドレスをネットワークインターフェイスに割り当てるには
- コマンドラインを使用して既存のインスタンスにセカンダリプライベート IPv4 アドレスを割り当てるには

New console

インスタンスの起動時にセカンダリプライベート IPv4 アドレスを割り当てるには

1. インスタンスを起動するための手順に従います。[ネットワーク設定] で、[編集] を選択します。
2. VPC とサブネットを選択します。
3. 高度なネットワーク設定の拡張。
4. [セカンダリ IP] で、[自動で割り当て] を選択して IP アドレスの数を入力するか (Amazon が自動的にセカンダリ IPv4 アドレスを割り当てます)、[手動で割り当て] を選択して IPv4 アドレスを入力します。
5. インスタンスを起動する残りのステップを完了します。

Old console

インスタンスの起動時にセカンダリプライベート IPv4 アドレスを割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [Launch Instance] (インスタンスの起動) を選択します。
3. AMI を選択し、次にインスタンスタイプを選択して、[Next: Configure Instance Details] を選択します。

4. [Configure Instance Details] ページで、[Network] から VPC を選択し、[Subnet] からサブネットを選択します。
5. [Network Interfaces] セクションで、次の手順を実行し、[Next: Add Storage] を選択します。
   - 別のネットワークインターフェイスを追加するには、[Add Device] を選択します。コンソールでは、インスタンス起動時のネットワークインターフェイスを最大 2 つ指定できます。インスタンスを起動したら、ナビゲーションペインで [Network Interfaces] を選択し、ネットワークインターフェイスを追加します。アタッチできるネットワークインターフェイスの合計数はインスタンスタイプによって異なります。詳細については、各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数を参照してください。

     > **Important**
     >
     > 2 つ目のネットワークインターフェイスを追加すると、システムは、パブリック IPv4 アドレスを自動的に割り当てることができなくなります。プライマリネットワークインターフェイス (eth0) に Elastic IP アドレスを割り当てない限り、IPv4 経由でインスタンスに接続することはできません。Launch wizard 完了した後は、Elastic IP アドレスを割り当てることができます。詳細については、Elastic IP アドレスの操作を参照してください。

   - ネットワークインターフェイスごとに、[Secondary IP addresses] の下にある [Add IP] を選択し、サブネットの範囲に含まれるプライベート IP アドレスを入力するか、デフォルトの `Auto-assign` のままにして Amazon によってアドレスが自動的に選択されるようにします。
6. 次の [Add Storage] ページで、AMI によって指定されるボリューム (ルートデバイスボリュームなど) 以外にインスタンスにアタッチするボリュームを指定し、[Next: Add Tags] を選択します。
7. [Add Tags] ページで、ユーザーフレンドリーな名前などを使ってインスタンスのタグを指定し、[Next: Configure Security Group] を選択します。
8. [Configure Security Group] ページで、既存のセキュリティグループを選択するか、新しいグループを作成します。[Review and Launch] (確認と起動) を選択します。
9. [Review Instance Launch] ページで、設定内容を確認します。[Launch] を選択して、キーペアを選択し、インスタンスを起動します。Amazon EC2 を初めて使用する場合、これまでにキーペアを作成したことがなければ、ウィザードによってキーペアを作成するよう求めるメッセージが表示されます。

> **Important**
>
> セカンダリプライベート IP アドレスをネットワークインターフェイスに追加した後、インスタンスに接続して、インスタンス自体でセカンダリプライベート IP アドレスを設定する必要があります。詳細については、セカンダリプライベート IPv4 アドレスを認識するようにインスタンスのオペレーティングシステムを設定するを参照してください。

コマンドラインを使用して起動時にセカンダリ IPv4 アドレスを割り当てるには

- 次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。
  - run-instances コマンド (AWS CLI) の `--secondary-private-ip-addresses` オプション
  - `-NetworkInterface` を定義し、New-EC2Instance コマンド (AWS Tools for Windows PowerShell) に `PrivateIpAddresses` パラメータを指定します。

セカンダリプライベート IPv4 アドレスをネットワークインターフェイスに割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択し、インスタンスのネットワークインターフェイスを選択します。
3. [Actions]、[Manage IP Addresses] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv4 アドレス] で、[新しい IP アドレスの割り当て] を選択します。
5. インスタンスのサブネットの範囲に含まれる特定の IPv4 アドレスを入力するか、フィールドを空のままにして Amazon によって IPv4 アドレスが自動的に選択されるようにします。
6. (省略可能) セカンダリプライベート IP アドレスが既に別のネットワークインターフェイスに割り当てられている場合、[許可] を選択して、セカンダリプライベート IP アドレスを割り当て直すことができます。
7. [Save] を選択します。

または、インスタンスにセカンダリプライベート IPv4 アドレスを割り当てることができます。ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。次に、[アクション] を選択し、[ネットワーク]、[IP アドレスの管理] の順に選択します。上記のステップに従って、同じ情報

を設定できます。IP アドレスは、インスタンスのプライマリネットワークインターフェイス (eth0) に割り当てられます。

コマンドラインを使用して既存のインスタンスにセカンダリプライベート IPv4 アドレスを割り当てるには

- 次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。
  - assign-private-ip-addresses (AWS CLI)
  - Register-EC2PrivateIpAddress (AWS Tools for Windows PowerShell)

セカンダリプライベート IPv4 アドレスを認識するようにインスタンスのオペレーティングシステムを設定する

セカンダリプライベート IPv4 アドレスをインスタンスに割り当てたら、セカンダリプライベート IP アドレスを認識するようにインスタンスのオペレーティングシステムを設定する必要があります。

- Amazon Linux を使用している場合、ec2-net-utils パッケージがこの処理を自動実行します。このパッケージは、インスタンスの実行中にアタッチされる追加のネットワークインターフェイスを設定し、DHCP リースの更新中にセカンダリ IPv4 アドレスを更新して、関連するルーティングルールを更新します。コマンド `sudo service network restart` を使用して即座にインターフェースの一覧を更新し、`ip addr li` を使用することで最新の一覧を表示することができます。ネットワーク構成を手動で構成する必要がある場合、ec2-net-utils パッケージを削除できます。詳細については、Amazon Linux 2 向けに ec2-net-utils を使用してネットワークインターフェイスを設定するを参照してください。
- 別の Linux ディストリビューションを使用している場合、Linux ディストリビューションのドキュメントを参照してください。追加のネットワークインターフェイスとセカンダリ IPv4 アドレスの設定に関する情報が記載されています。同じサブネットのインスタンスに複数のインターフェイスがある場合、非対称のルーティングに対処する方法については、ルーティングルールの使用に関する情報を検索してください。

Windows インスタンスの設定については、Windows インスタンスの Amazon EC2 ユーザーガイドのVPC 内の Windows インスタンスのセカンダリプライベート IP アドレスの設定を参照してください。

セカンダリプライベート IPv4 アドレスへの Elastic IP アドレスの割り当て

Elastic IP アドレスをセカンダリプライベート IPv4 アドレスに関連付けるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. Elastic IP アドレスのチェックボックスをオンにします。
4. [アクション]、[Elastic IP アドレスの関連付け] の順に選択します。
5. [リソースタイプ] で [ネットワークインターフェイス] を選択します。ネットワークインターフェイスを選択し、[プライベート IP アドレス] リストからセカンダリ IP アドレスを選択します。
6. [ネットワークインターフェイス] でネットワークインターフェイスを選択し、[プライベート IP アドレス] リストからセカンダリ IP アドレスを選択します。
7. [プライベート IP アドレス] でセカンダリ IP アドレスを選択します。
8. [関連付ける] を選択します。

コマンドラインを使用して Elastic IP アドレスにセカンダリプライベート IPv4 アドレスを関連付けるには

- 次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。
  - associate-address (AWS CLI)
  - Register-EC2Address (AWS Tools for Windows PowerShell)

セカンダリプライベート IPv4 アドレスの表示

ネットワークインターフェイスに割り当てられたプライベート IPv4 アドレスを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [詳細] タブの [IP アドレス] で、[プライベート IPv4 アドレス] と [セカンダリプライベート IPv4 アドレス] を見つけます。

インスタンスに割り当てられたプライベート IPv4 アドレスを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスのチェックボックスをオンにします。
4. [ネットワーキング] タブの [ネットワーキングの詳細] で、[プライベート IPv4 アドレス] と [セカンダリプライベート IPv4 アドレス] を見つけます。

セカンダリプライベート IPv4 アドレスの割り当て解除

セカンダリプライベート IPv4 アドレスが不要になった場合、インスタンスやネットワークインターフェイスから割り当て解除できます。セカンダリプライベート IPv4 アドレスをネットワークインターフェイスから割り当て解除した場合、Elastic IP アドレス (存在する場合) の関連付けも解除されます。

インスタンスからセカンダリプライベート IPv4 アドレスを割り当て解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択し、[アクション]、[ネットワーク]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv4 アドレス] で、割り当て解除する IPv4 アドレスに対して [割り当て解除] を選択します。
5. [Save] を選択します。

ネットワークインターフェイスからセカンダリプライベート IPv4 アドレスを割り当て解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. ネットワークインターフェイスを選択し、[アクション]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv4 アドレス] で、割り当て解除する IPv4 アドレスに対して [割り当て解除] を選択します。
5. [Save] を選択します。

コマンドラインを使用してセカンダリプライベート IPv4 アドレスを割り当て解除するには

- 次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。
  - unassign-private-ip-addresses (AWS CLI)
  - Unregister-EC2PrivateIpAddress (AWS Tools for Windows PowerShell)

複数の IPv6 アドレスの使用

インスタンスに複数の IPv6 アドレスを割り当て、インスタンスに割り当てられている IPv6 アドレスを表示したり、インスタンスから IPv6 アドレスを割り当て解除したりできます。

コンテンツ

複数の IPv6 アドレスの割り当て

起動時または起動後のインスタンスに 1 つ以上の IPv6 アドレスを割り当てることができます。IPv6 アドレスをインスタンスに割り当てるには、インスタンスを起動した VPC およびサブネットに IPv6 CIDR ブロックが関連付けられている必要があります。

New console

起動時に複数の IPv6 アドレスを割り当てるには

1. インスタンスを起動するための手順に従います。[ネットワーク設定] で、[編集] を選択します。
2. VPC とサブネットを選択します。
3. 高度なネットワーク設定の拡張。
4. [IPv6 IP] で、[自動で割り当て] を選択して IP アドレスの数を入力するか (Amazon が自動的に IPv6 アドレスを割り当てます)、[手動で割り当て] を選択して IPv6 アドレスを入力します。
5. インスタンスを起動する残りのステップを完了します。

Old console

起動時に複数の IPv6 アドレスを割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ダッシュボードから、[Launch Instance] を選択します。
3. AMI を選択し、次にインスタンスタイプを選択して、[Next: Configure Instance Details] を選択します。IPv6 をサポートするインスタンスタイプを必ず選択します。詳細については、インスタンスタイプを参照してください。
4. [Configure Instance Details] ページで、[Network] リストから VPC を選択し、[Subnet] リストからサブネットを選択します。
5. [Network Interfaces] セクションで、次の手順を実行し、[Next: Add Storage] を選択します。
   - IPv6 アドレスをプライマリネットワークインターフェイス (eth0) に割り当てるには、[IPv6 IPs]、[Add IP] の順に選択します。セカンダリ IPv6 アドレスを追加するには、再度 [Add IP] 選択します。サブネットの範囲から IPv6 アドレスを入力するか、デフォルトの [Auto-assign] を使用すると Amazon がサブネットから自動的に IPv6 アドレスを選択します。
   - [Add Device] を選択して別のネットワークインターフェイスを追加し、上記のステップを繰り返してそのネットワークインターフェイスに 1 つ以上の IPv6 アドレスを追加します。コンソールでは、インスタンス起動時のネットワークインターフェイスを最大 2 つ指定できます。インスタンスを起動したら、ナビゲーションペインで [Network Interfaces] を選択し、ネットワークインターフェイスを追加します。アタッチできるネットワークインターフェイスの合計数はインスタンスタイプによって異なります。詳細については、各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数を参照してください。
6. ボリュームをアタッチしてインスタンスにタグを付けるには、ウィザードの以下のステップに従ってください。
7. [Configure Security Group] ページで、既存のセキュリティグループを選択するか、新しいグループを作成します。IPv6 経由でインスタンスに到達可能にする場合は、IPv6 アドレスからのアクセスを許可するルールがセキュリティグループにあることを確認します。詳細については、さまざまなユースケースのセキュリティグループのルールを参照してください。[Review and Launch] (確認と起動) を選択します。
8. [Review Instance Launch] ページで、設定内容を確認します。[Launch] を選択して、キーペアを選択し、インスタンスを起動します。Amazon EC2 を初めて使用する場合、これまで

にキーペアを作成したことがなければ、ウィザードによってキーペアを作成するよう求めるメッセージが表示されます。

Amazon EC2 コンソールの [インスタンス] 画面を使用して、既存のインスタンスに複数の IPv6 アドレスを割り当てることができます。IPv6 アドレスは、インスタンスのプライマリネットワークインターフェイス (eth0) に割り当てられます。IPv6 アドレスをインスタンスに割り当てるには、IPv6 アドレスが別のインスタンスやネットワークインターフェイスにまだ割り当てられていないことを確認します。

複数の IPv6 アドレスを既存のインスタンスに割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択し、[アクション]、[ネットワーキング]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv6 アドレス] で、追加する IPv6 アドレスごとに [新しい IP アドレスの割り当て] を選択します。サブネットの範囲から IPv6 アドレスを指定します。また、フィールドを空のままにすると Amazon によって IPv6 アドレスが自動的に選択されます。
5. [Save] を選択します。

また、既存のネットワークインターフェイスに複数の IPv6 アドレスを割り当てることができます。そのネットワークインターフェイスは、IPv6 CIDR ブロックが関連付けられているサブネットで作成されている必要があります。特定の IPv6 アドレスをネットワークインターフェイスに割り当てるには、その IPv6 アドレスが別のネットワークインターフェイスにまだ割り当てられていないことを確認します。

複数の IPv6 アドレスをネットワークインターフェイスに割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. ネットワークインターフェイスを選択し、[アクション]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv6 アドレス] で、追加する IPv6 アドレスごとに [新しい IP アドレスの割り当て] を選択します。サブネットの範囲から IPv6 アドレスを指定

します。また、フィールドを空のままにすると Amazon によって IPv6 アドレスが自動的に選択されます。

5. [Save] を選択します。

CLI の概要

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- 起動時に IPv6 アドレスを割り当てる:
  - run-instances コマンド (AWS CLI) で、`--ipv6-addresses` または `--ipv6-address-count` オプションを使用する
  - `-NetworkInterface`を定義し、New-EC2Instance コマンド (AWS Tools for Windows PowerShell) で、`Ipv6Addresses`パラメータまたは`Ipv6AddressCount`パラメータを指定します
- IPv6 アドレスをネットワークインターフェイスに割り当てる:
  - assign-ipv6-addresses (AWS CLI)
  - Register-EC2Ipv6AddressList (AWS Tools for Windows PowerShell)

IPv6 アドレスの表示

インスタンスまたはネットワークインターフェイスの IPv6 アドレスを確認できます。

インスタンスに割り当てられた IPv6 アドレスを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスのチェックボックスをオンにします。
4. [ネットワーキング] タブで、[IPv6 アドレス] フィールドを見つけます。

ネットワークインターフェイスに割り当てられた IPv6 アドレスを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。

4. [詳細] タブの [IP アドレス] で、[IPv6 アドレス] フィールドを見つけます。

CLI の概要

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- インスタンスの IPv6 アドレスを確認する場合
  - describe-instances (AWS CLI)
  - Get-EC2Instance (AWS Tools for Windows PowerShell)
- ネットワークインターフェイスの IPv6 アドレスを確認する場合
  - describe-network-interfaces (AWS CLI)
  - Get-EC2NetworkInterface (AWS Tools for Windows PowerShell)

IPv6 アドレスの割り当て解除

インスタンスのプライマリネットワークインターフェイスから IPv6 アドレスを割り当て解除できます。またはネットワークインターフェイスから IPv6 アドレスを割り当て解除できます。

インスタンスから IPv6 アドレスを割り当て解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスのチェックボックスを選択後、[アクション]、[ネットワーキング]、[IP アドレスの管理] の順に選択します。
4. ネットワークインターフェイスを展開します。[IPv6 addresses] (IPv6 アドレス) で、IPv6 アドレスの横にある [Unassign] (割り当て解除) を選択します。
5. [Save] を選択します。

ネットワークインターフェイスから IPv6 アドレスを割り当て解除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. ネットワークインターフェイスのチェックボックスを選択後、[アクション]、[IP アドレスの管理] の順に選択します。

4. ネットワークインターフェイスを展開します。[IPv6 addresses] (IPv6 アドレス) で、IPv6 アドレスの横にある [Unassign] (割り当て解除) を選択します。
5. [Save] を選択します。

CLI の概要

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- unassign-ipv6-addresses (AWS CLI)
- Unregister-EC2Ipv6AddressList (AWS Tools for Windows PowerShell)

EC2 インスタンスのホスト名

EC2 インスタンスを作成すると、AWS は、そのインスタンスのホスト名を作成します。ホスト名のタイプと、AWS によるプロビジョニング方法についての詳細Amazon EC2 インスタンスのホスト名タイプを参照してください。Amazon は、Amazon が提供するホスト名を IPv4 および IPv6 アドレスに解決する DNS サーバーを提供します。Amazon DNS サーバーは VPC ネットワークの範囲に 2 をプラスしたアドレスにあります。詳細については、Amazon VPC ユーザーガイドの「DNS attributes for your VPC」(VPC の DNS 属性) を参照してください。

リンクローカルアドレス

リンクローカルアドレスはよく知られた、ルーティング不可の IP アドレスです。Amazon EC2 は、リンクローカルアドレス空間のアドレスを使用して、EC2 インスタンスからのみアクセスできるサービスを提供します。これらのサービスはインスタンス上では実行されず、基盤となるホスト上で実行されます。これらのサービスのリンクローカルアドレスにアクセスすると、Xen ハイパーバイザーまたは Nitro コントローラーのどちらかと通信することになります。

リンクローカルアドレスの範囲

- IPv4 — 169.254.0.0/16 (169.254.0.0 ~ 169.254.255.255)
- IPv6 – fe80::/10

リンクローカルアドレスを使用してアクセスするサービス

- インスタンスメタデータサービス

- Amazon Route 53 Resolver (Amazon DNS サーバーとも呼ばれます)
- Amazon Time Sync Service

Amazon EC2 インスタンスのホスト名タイプ

このセクションでは、VPC サブネットでインスタンスを起動する際に使用できる Amazon EC2 インスタンスのゲスト OS ホスト名のタイプについて説明します。

ホスト名によって、ネットワーク上の EC2 インスタンスが区別されます。例えば、ネットワーク上の一部またはすべてのインスタンスと通信するスクリプトを実行する場合、インスタンスのホスト名を使用できます。

コンテンツ

EC2 ホスト名のタイプ

EC2 インスタンスが VPC で起動されるときのゲスト OS ホスト名には、次の 2 つのホスト名タイプがあります。

- [IP name] (IP 名) : 従来の命名スキームでは、インスタンスの起動時に、インスタンスのプライベート IPv4 アドレスがインスタンスのホスト名に含まれます。IP 名は EC2 インスタンスの存続中に存在します。プライベート DNS ホスト名として使用すると、プライベート IPv4 アドレス (A レコード) のみが返されます。
- [Resource name] (リソース名): インスタンスを起動すると、EC2 インスタンス ID がインスタンスのホスト名に含まれます。リソース名はEC2 インスタンスの存続中に存在します。プライベート DNS ホスト名として使用すると、プライベート IPv4 アドレス (A レコード) と IPv6 グローバルユニキャストアドレス (AAAA レコード) の両方を返すことができます。

EC2 インスタンスのゲスト OS ホスト名 のタイプはサブネット設定によって異なります。

- インスタンスが IPv4 専用サブネットで起動された場合、IP 名またはリソース名を選択できます。

- インスタンスがデュアルスタック (IPv4+IPv6) サブネットで起動されている場合、IP 名またはリソース名を選択できます。
- インスタンスが IPv6 専用サブネットで起動された場合、リソース名が自動的に使用されます。

コンテンツ

IP 名

[IP name] (IP 名) の [Hostname type] (ホスト名タイプ) を使用して EC2 インスタンスを起動すると、ゲスト OS ホスト名がプライベート IPv4 アドレスを使用するように設定されます。

- us-east-1 でのインスタンスのフォーマット: *private-ipv4-address*.ec2.internal
- 例 : *ip-10-24-34-0*.ec2.internal
- その他の AWS リージョンのインスタンスのフォーマット: *private-ipv4-address.region*.compute.internal
- 例 : *ip-10-24-34-0.us-west-2*.compute.internal

リソース名

EC2 インスタンスを IPv6 専用サブネットで起動すると、[Resource name] (リソース名) の[Hostname type] (ホスト名タイプ) がデフォルトで選択されます。IPv4 専用またはデュアルスタック (IPv4+IPv6) サブネットでインスタンスを起動すると、[Resource name] (リソース名) は選択できるオプションです。インスタンスを起動してから、ホスト名設定を管理できます。詳細については、ホスト名のタイプと DNS ホスト名の設定を変更しますを参照してください。

[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) を使用して EC2 インスタンスを起動すると、ゲスト OS ホスト名が EC2 インスタンス ID を使用するように設定されます。

- us-east-1 でのインスタンスのフォーマット: *ec2-instance-id*.ec2.internal
- 例 : *i-0123456789abcdef*.ec2.internal
- その他の AWS リージョンのインスタンスのフォーマット: *ec2-instance-id.region*.compute.internal

- 例：*i-0123456789abcdef.us-west-2*.compute.internal

IP 名とリソース名の違い

IP 名とリソース名の両方の DNS クエリが共存して、下位互換性を確保し、ホスト名に対する IP ベースの命名 からリソースベースの命名に移行できます。IP 名に基づくプライベート DNS ホスト名の場合、インスタンスの DNS A レコードクエリに応答するかどうかを設定することはできません。DNS A レコードクエリは、ゲスト OS ホスト名の設定に関係なく、常に応答されます。対照的に、リソース名に基づくプライベート DNS ホスト名の場合、インスタンスの DNS A または DNS AAAA クエリに応答するかどうかを設定できます。インスタンスの起動時またはサブネットの変更時に、レスポンスの動作を設定します。詳細については、ホスト名のタイプと DNS ホスト名の設定を変更しますを参照してください。

リソース名と IP 名が表示される場所

このセクションでは、EC2 コンソールでホスト名タイプのリソース名と IP 名が表示される場所について説明します。

コンテンツ

- EC2 インスタンスを作成する場合
- 既存の EC2 インスタンスの詳細を表示する場合

EC2 インスタンスを作成する場合

EC2 インスタンスを作成する際、選択したサブネットのタイプに応じて、[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) が使用可能になるか、または選択されて変更できない場合があります。このセクションでは、ホスト名タイプのリソース名と IP 名が表示されるシナリオについて説明します。

シナリオ 1

ウィザード (新しいインスタンス起動ウィザードを使用してインスタンスを起動するを参照) で EC2 インスタンスを作成し、詳細を設定するときに IPv6 専用に設定したサブネットを選択します。

この場合、[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) は自動的に選択され、変更できません。[Enable IP name IPv4 (A record) DNS requests] (IP 名 IPv4 (A レコード) DNS リクエストを有効にする) の [DNS Hostname] (DNS ホスト名) オプションと [Enable resource-based

IPv4 (A record) DNS requests] (リソースベースの IPv4 (A レコード) DNS リクエストを有効にする) は自動的に選択解除され、変更できません。[Enable resource-based IPv6 (AAAA record) DNS requests] (リソースベースの IPv6 (AAAA レコード) DNS リクエストを有効にする) がデフォルトで選択されていますが、変更可能です。選択した場合、リソース名への DNS リクエストはこの EC2 インスタンスの IPv6 アドレス (AAAA レコード) に解決されます。

シナリオ 2

ウィザード (新しいインスタンス起動ウィザードを使用してインスタンスを起動するを参照) で EC2 インスタンスを作成し、詳細を設定するときに、IPv4 CIDR ブロックまたは IPv4 と IPv6 の両方の CIDR ブロック (「デュアルスタック」) で構成されたサブネットを選択します。

この場合、[Enable IP name IPV4 (A record) DNS requests] (IP 名 IPV4 (A レコード) DNS リクエストを有効にする) は自動的に選択され、変更できません。つまり、IP 名へのリクエストは、この EC2 インスタンスの IPv4 アドレス (A レコード) に解決されます。

オプションはサブネットの設定にデフォルト設定されますが、サブネットの設定に応じてこのインスタンスのオプションを変更できます。

- [Hostname type] (ホスト名タイプ): EC2 インスタンスのゲスト OS ホスト名をリソース名 をリソース名または IP 名にするかを決定します。デフォルト値は [IP name] (IP 名) です。
- [Enable resource-based IPv4 (A record) DNS requests] (リソースベースの IPv4 (A レコード) DNS リクエストを有効にする): リソース名へのリクエストが、この EC2 インスタンスのプライベート IPv4 アドレス (A レコード) に解決されるかどうかを決定します。このオプションはデフォルトで選択されていません。
- [Enable resource-based IPv6 (AAAA record) DNS requests] (リソースベースの IPv6 (AAAA レコード) DNS リクエストを有効にする): リソース名へのリクエストが、この EC2 インスタンスの IPv6 GUA アドレス (AAAA レコード) に解決するかどうかを決定します。このオプションはデフォルトで選択されていません。

既存の EC2 インスタンスの詳細を表示する場合

既存の EC2 インスタンスのホスト名の値は、EC2 インスタンスの [Details] (詳細) タブで確認できます。

- [Hostname type] (ホスト名タイプ): IP 名またはリソース名形式のホスト名
- [Private IP DNS name (IPv4 only)] (プライベート IP DNS 名 (IPv4 専用)): インスタンスのプライベート IPv4 アドレスに常に解決される IP 名

- [Private resource DNS name] (プライベートリソース DNS 名): このインスタンス用に選択された DNS レコードに解決されるリソース名
- [Answer private resource DNS name] (プライベートリソース DNS 名に応答する): リソース名は IPv4 (A)、IPv6 (AAAA)、または IPv4 と IPv6 (A と AAAA) の DNS レコードに解決されます。

さらに、SSH 経由で直接 EC2 インスタンスに接続して、`hostname`コマンドを入力すると、ホスト名が IP 名またはリソース名形式で表示されます。

リソース名または IP 名のどちらを選択するかを決めるには

EC2 インスタンス (新しいインスタンス起動ウィザードを使用してインスタンスを起動するを参照) を起動する際に [Resource name] (リソース名) の [Hostname type] (ホスト名) を選んだ場合、EC2 インスタンスはリソース名形式のホスト名を使用して起動します。このような場合、この EC2 インスタンスの DNS レコードはリソース名を指すこともあります。これにより、そのホスト名はインスタンスの IPv4 アドレス、IPv6 アドレス、または IPv4 アドレスと IPv6 アドレスの両方に解決されるように選択する柔軟さを実現します。今後 IPv6 を使用する予定がある場合、あるいは現在デュアルスタックのサブネットを使用している場合、[Resource name] (リソース名) の [Hostname type] (ホスト名タイプ) の使用により、DNS レコード自体に変更を加えることなく、インスタンスのホスト名の DNS 解決を変更できます。リソース名を使用すると、EC2 インスタンスで IPv4 および IPv6 DNS リゾリューションを追加して削除できます。

代わりに [IP name] (IP 名) の [Hostname type] (ホスト名タイプ) を選択し、DNS ホスト名として使用する場合、インスタンスの IPv4 アドレスにのみ解決されます。インスタンスに IPv4 アドレスと IPv6 アドレスの両方が関連付けられている場合でも、インスタンスの IPv6 アドレスには解決されません。

ホスト名のタイプと DNS ホスト名の設定を変更します

このセクションの手順に従って、サブネットまたは EC2 インスタンスの起動後にホスト名タイプと DNS ホスト名設定を変更します。

コンテンツ

サブネット

VPC コンソールでサブネットを選択し、[Actions] (アクション)、[Edit subnet settings] (サブネット設定の編集) を選択して、サブネットの設定を変更します。

> **Note**
>
> サブネット設定を変更しても、サブネットですでに起動されている EC2 インスタンスの設定は変更されません。

- [Hostname type] (ホスト名タイプ): サブネットで起動される EC2 インスタンスのゲスト OS ホスト名のデフォルト設定をリソース名または IP 名にするかを決定します。
- [Enable DNS hostname IPv4 (A record) requests] (DNS ホスト名 IPv4 (A レコード) リクエストを有効にする): リソース名への DNS リクエスト/クエリがこの EC2 インスタンスのプライベート IPv4 アドレス (A レコード) に解決されるかどうかを決定します。
- [Enable DNS hostname IPv6 (AAAA record) requests] (DNS ホスト名 IPv6 (AAAA レコード) 要求を有効にする): リソース名への DNS リクエスト/クエリがこの EC2 インスタンスの IPv6 アドレス (AAAA レコード) に解決されるかどうかを決定します。

EC2インスタンス

このセクションのステップに従って、EC2 インスタンスのホスト名タイプと DNS ホスト名設定を変更します。

> **Important**
>
> - [Use resource based name as guest OS hostname] (リソースベース命名をゲスト OS ホスト名として使用) の設定を変更するには、まずインスタンスを停止する必要があります。[Answer DNS hostname IPv4 (A record) request] (DNS ホスト名 IPv4 (A レコード) 要求に応答する) または [Answer DNS hostname IPv6 (AAAA record) requests] (DNS ホスト名 IPv6 (AAAA レコード) 要求に応答する) の設定を変更するには、インスタンスを停止する必要はありません。
> - 非 EBS backed EC2 インスタンスタイプの 設定を変更するには、インスタンスを停止できません。インスタンスを終了し、目的のホスト名タイプと DNS ホスト名の設定で新しいインスタンスを起動する必要があります。

EC2 インスタンスのホスト名タイプと DNS ホスト名の設定を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [Use resource based naming as guest OS hostname] (リソースベース命名をゲスト OS ホスト名として使用) の設定を変更する場合、まず EC2 インスタンスを停止してください。それ以外の場合は、この手順をスキップしてください。

   インスタンスを停止し、インスタンスを選択後、[Instance state] (インスタンスの状態)、[Stop instance] (インスタンスの停止) の順にクリックします。
3. インスタンスを選択し、そして [Actions] (アクション)、[Instance settings] (インスタンス設定)、[Change resource based naming options] (リソースベースの命名オプションの変更) を選択します。

   - [Use resource based naming as guest OS hostname] (リソースベース命名をゲスト OS ホスト名として使用) : EC2 インスタンスのゲスト OS ホスト名をリソース名または IP 名にするかを決定します。
   - [Answer DNS hostname IPv4 (A record) requests] (DNS ホスト名 IPv4 (A レコード) リクエストに応答する): リソース名への DNS リクエスト/クエリがこの EC2 インスタンスのプライベート IPv4 アドレスに解決されるかどうかを決定します。
   - [Answer DNS hostname IPv6 (AAAA record) requests] (DNS ホスト名 IPv6 (AAAA レコード) 要求に応答する): リソース名への DNS リクエスト/クエリがこの EC2 インスタンスの IPv6 アドレス (AAAA レコード) に解決されるかどうかを決定します。
4. [Save] を選択します。
5. インスタンスを停止した後は、インスタンスを再起動します。

Amazon EC2 で自分の IP アドレスを使用する (BYOIP)

パブリックにルーティング可能な IPv4 または IPv6 アドレス範囲の一部または全部を、オンプレミスのネットワークから AWS アカウントに導入することができます。アドレス範囲は引き続き管理でき、AWS を通じてインターネット上でアドレス範囲をアドバタイズできます。アドレス範囲を AWS に設定すると、そのアドレス範囲はアドレスプールとして AWS アカウントに表示されます。

BYOIP を使用できるリージョンのリストについては、「リージョナルな可用性」を参照してください。

Windows インスタンスの BYOIP 情報を確認するには、Windows インスタンス用 Amazon EC2 ユーザーガイドのページ「Amazon EC2 で自分の IP アドレスを使用する (BYOIP)」に切り替えます。

> **Note**
>
> - このページのステップでは、独自の IP アドレス範囲を Amazon EC2 でのみ使用する方法について説明します。
> - AWS Global Acceleratorで使用するために独自の IP アドレス範囲を使用するには、「AWS Global Accelerator デベロッパーガイド」の「独自 IP アドレス (BYOIP) の使用」を参照してください。
> - Amazon VPC IP Address Manager で使用する独自の IP アドレス範囲を導入するには、「Amazon VPC IPAM ユーザーガイド」の「チュートリアル: BYOIP アドレス CIDR を IPAM へ」を参照してください。

コンテンツ

BYOIP の定義

- X.509 自己署名証明書 — ネットワーク内のデータを暗号化および認証するために最も一般的に使用される証明書標準。これは、RDAP レコードからの IP スペースの制御を検証するために AWS によって使用される証明書です。X.509 証明書の詳細については、「RFC 3280」を参照してください。
- AS 番号 (ASN) — 明確に定義された単一のルーティングポリシーを維持する 1 つ以上のネットワークオペレーターによって実行される IP プレフィックスのグループを定義するグローバル一意識別子。

- 地域インターネットレジストリ (RIR) — 世界のある地域内の IP アドレスと ASN の割り当てと登録を管理する組織。
- レジストリデータアクセスプロトコル (RDAP) — RIR 内の現在の登録データを照会する、読み取り専用プロトコルです。クエリされた RIR データベース内のエントリは「RDAP レコード」と呼ばれます。特定のレコードタイプは、RIR が提供するメカニズムを使用して顧客により更新される必要があります。これらのレコードは、RIR 内のアドレス空間の制御を確認するために AWS によりクエリされます。
- Route Origin Authorization (ROA) — お客様が特定の自律システムで IP アドバタイズメントを認証するために RIR によって作成されたオブジェクト。概要については、ARIN ウェブサイトの「Route Origin Authorizations (ROAs)」(ルートオリジン認証 (ROA)) を参照してください。
- ローカルインターネットレジストリ (LIR) — RIR からの IP アドレスのブロックをお客様に割り当てるインターネットサービスプロバイダーなどの組織。

要件とクォータ

- アドレス範囲は、地域インターネットレジストリ (RIR) に登録する必要があります。地理的リージョンに関するポリシーについては、RIR を参照してください。BYOIP は、現在、American Registry for Internet Numbers (ARIN)、Réseaux IP Européens Network Coordination Centre (RIPE) または Asia-Pacific Network Information Centre (APNIC) への登録をサポートしています。アドレス範囲は、事業体または機関エンティティについて登録を受ける必要があり、個人については登録を受けられない場合があります。
- 取得できる最も具体的な IPv4 アドレス範囲は /24 です。
- 提供できる最も具体的な IPv6 アドレス範囲は、パブリックにアドバタイズ可能な CIDR の場合は /48、パブリックにアドバタイズ可能でない CIDR の場合は /56 です。
- ROA はパブリックにアドバタイズ可能でない CIDR 範囲には必要ありませんが、RDAP レコードは更新する必要があります。
- 各アドレス範囲は、一度に 1 つの AWS リージョンで使用できます。
- AWS リージョンごとに合計 5 つの BYOIP IPv4 および IPv6 アドレス範囲を AWS アカウントに取り込むことができます。Service Quotas コンソールを使用して BYOIP CIDR のクォータを調整することはできませんが、「AWS 全般のリファレンス」の「AWS サービスクォータ」で説明されているように、AWS サポートセンターに連絡してクォータの引き上げをリクエストすることはできます。
- Amazon VPC IP Address Manager (IPAM) を使用し、IPAM と AWS RAM Organizationsを統合しない限り、AWS を使用する他のアカウントとIPアドレス範囲を共有することはできません。詳細

については、Amazon VPC IP アドレス管理ユーザーガイドのAWS Organizations と IPAM を統合するを参照してください。

- IP アドレス範囲内のアドレスには、消去履歴が含まれている必要があります。弊社は、IP アドレス範囲に評価が低いまたは悪意のある挙動に関連付けられている IP アドレスが含まれている場合、当該範囲の評価を調査したり、当該範囲を拒否する権利を留保したりすることがあります。
- レガシーアドレススペース、つまり Regional Internet Registry (RIR) システムの形成前に Internet Assigned Numbers Authority's (IANA) の中央レジストリによって配布された IPv4 アドレススペースには、引き続き対応する ROA オブジェクトが必要です。
- LIR では、手動プロセスを使用してレコードを更新するのが一般的です。LIR によっては、デプロイに数日かかることがあります。
- 大規模な CIDR ブロックには、単一の ROA オブジェクトと RDP レコードが必要です。単一のオブジェクトとレコードを使用して、その範囲から AWS まで (複数の AWS リージョンにまたがることもできます) 複数の小さな CIDR ブロックを使用できます。
- BYOIP は、Wavelength Zones または AWS Outposts ではサポートされていません。
- RADb やその他の IRR の BYOIP を手動で変更しないでください。BYOIP は RADb を自動的に更新します。BYOIP ASN を含む手動変更を行うと、BYOIP プロビジョニング操作が失敗します。

BYOIP アドレス範囲のオンボーディングの前提条件

BYOIP のオンボーディングプロセスには 2 つのフェーズがあり、そのためには 3 つのステップを実行する必要があります。これらの手順は、次の図に示す手順に対応しています。このドキュメントには手動のステップが含まれていますが、RIR はこれらのステップをサポートするマネージドサービスを提供している場合があります。

準備フェーズ

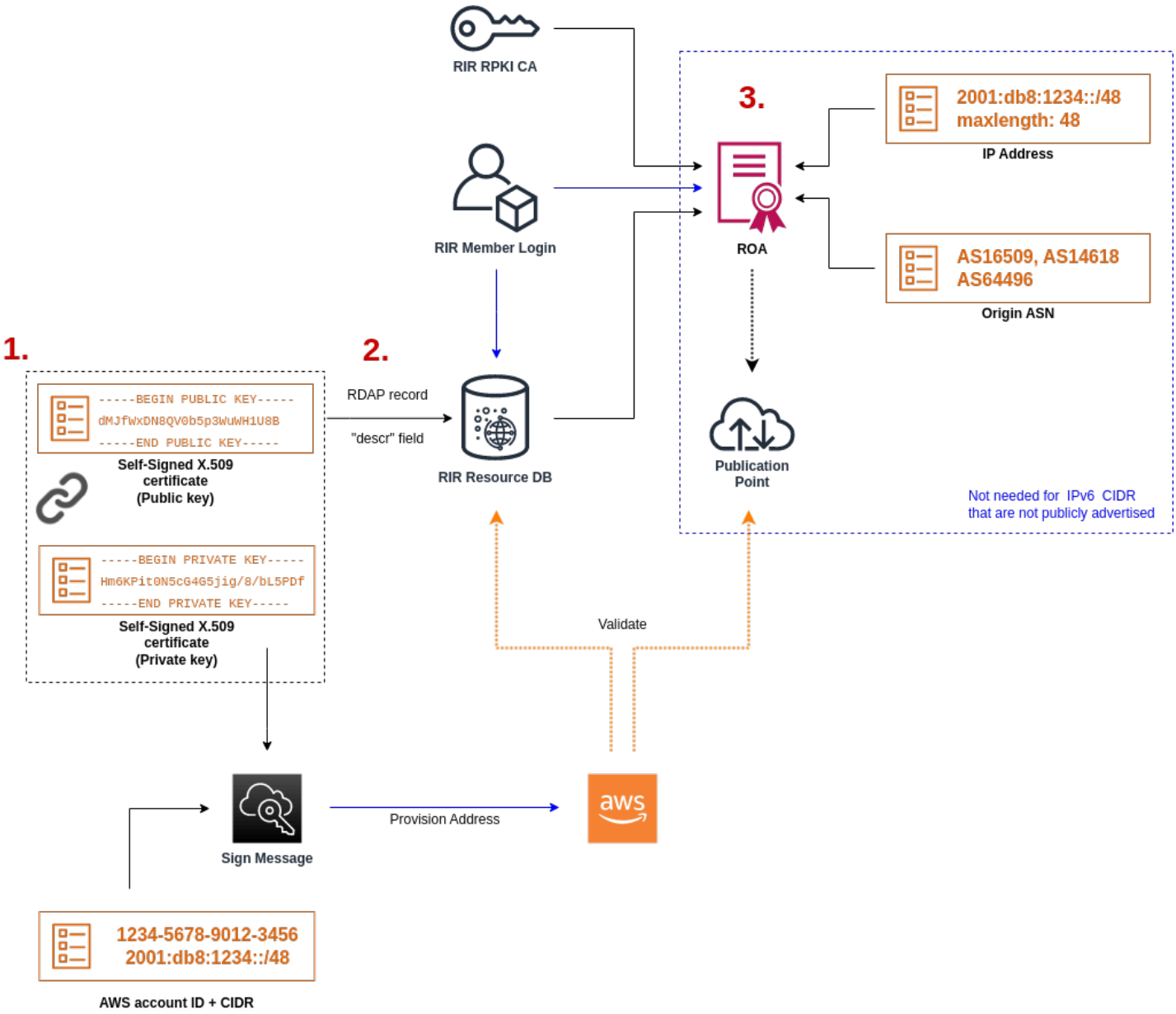
1. 認証のために、プライベートキーを作成し、それを使用して自己署名 X.509 証明書を生成します。この証明書は、プロビジョニング段階でのみ使用されます。

RIR 設定フェーズ

2. 自己署名証明書を RDAP レコードのコメントにアップロードします。

3. RIR に ROA オブジェクトを作成します。ROA は、目的のアドレス範囲、アドレス範囲のアドバタイズを許可する自律システム番号 (ASN)、および RIR の Resource Public Key Infrastructure (RPKI) に登録する有効期限を定義します。

> **Note**
>
> パブリックにアドバタイズ可能でない IPv6 アドレス空間には、ROA は必要ありません。



複数のアドレス範囲を使用する場合は、それぞれの不連続のアドレス範囲に対し、このプロセスを繰り返します。ただし、連続したブロックを複数の異なる AWS リージョンに分割する場合は、準備の手順と RIR 設定手順を繰り返す必要はありません。

新しくアドレス範囲を追加しても、以前に追加済みのアドレス範囲には影響を与えません。

> **Important**
>
> アドレス範囲をオンボーディングする前に、次の前提条件を満たしていることを確認してください。このセクションのタスクには Linux ターミナルが必要で、Linux、AWS CloudShell、または Windows Subsystem for Linux を使用して実行できます。

1. プライベートキーを作成し、X.509 証明書を生成します。

次の手順を使用して、自己署名 X509 証明書を作成し、RIR の RDAP レコードに追加します。RIR を使用してアドレス範囲を認証するには、このキーペアを使用します。openssl コマンドには、OpenSSL バージョン 1.0.2 以降が必要です。

次のコマンドをコピーし、プレースホルダー値 (色付きの斜体テキスト) のみを置換します。

この手順では、プライベート RSA キーを暗号化し、アクセスするためにパスフレーズを要求するベストプラクティスに従います。

1. 以下に示すように RSA 2048 ビットのプライベートキーを生成します。

```
$ openssl genpkey -aes256 -algorithm RSA -pkeyopt rsa_keygen_bits:2048 -out
 private-key.pem
```

-aes256 パラメータは、プライベートキーの暗号化に使用されるアルゴリズムを指定します。コマンドの出力は次の通りです。これには、パスフレーズを設定するためのプロンプトが含まれます。

```
......+++
.+++
Enter PEM pass phrase: xxxxxxx
Verifying - Enter PEM pass phrase: xxxxxxx
```

次のコマンドを使用して、キーを検査します。

```
$ openssl pkey -in private-key.pem -text
```

これは、次のようなパスフレーズプロンプトとキーの内容を返します。

```
Enter pass phrase for private-key.pem: xxxxxxx
```

```
-----BEGIN PRIVATE KEY-----
MIIEvgIBADANBgkqhkiG9w0BAQEFAASCBKgwggSkAgEAAoIBAQDFBXHRI4HVKAhh
3seiciooizCRTbJe1+YsxNTja4XyKypVGIFWDGhZs44FCHlPOOSVJ+NqP74w96oM
7DPS3xo9kaQyZBFn2YEp2EBq5vf307KHNRmZZUmkn0zHOSEpNmY2fMxISBxewlxR
FAniwmSd/8TDvHJMY9FvAIvWuTsv5l0tJKk+a91K4+tO3UdDR7Sno5WXExfsBrW3
g1ydo3TBsx8i5/YiVOcNApy7ge2/FiwY3aCXJB6r6nuF6H8mRgI4r4vkMRsOlAhJ
DnZPNeweboo+K3Q3lwbgbmOKD/z9svk8N/+hUTBtIX0fRtbG+PLIw3xWRHGrMSn2
BzsPVuDLAgMBAAECggEACiJUj2hfJkKv47Dc3es3Zex67A5uDVjXmxfox2Xhdupn
fAcNqAptV6fXt0SPUNbhUxbBKNbshoJGufFwXPli1SXnpzvkdU4Hyco4zgbhXFsE
RNYjYfOGzTPwdBLpNMB6k3Tp4RHse6dNrlH0jDhpioL8cQEBdBJyVF5X0wymEbmV
mC0jgH/MxsBAPWW6ZKicg9ULMlWiAZ3MRAZPjHHgpYkAAsUWKAbCBwVQcVjGO59W
jfZjzTX5pQtVVH68ruciH88DTZCwjCkjBhxg+OIkJBLE5wkh82jIHSivZ63flwLw
z+E0+HhELSZJrn2MY6Jxmik3qNNUOF/Z+3msdj2luQKBgQDjwlC/3jxp8zJy6P8o
JQKv7TdvMwUj4VSWOHZBHLv4evJaaia0uQjIo1UDa8AYitqhX1NmCCehGH8yuXj/
v6V3CzMKDkmRr1NrONnSz5QsndQ04Z6ihAQlPmJ96g4wKtgoC7AYpyP0g1a+4/sj
b1+o3YQI4pD/F71c+qaztH7PRwKBgQDdc23yNmT3+Jyptf0fKjEvONK+xwUKzi9c
L/OzBq5yOIC1Pz2T85gOe1i8kwZws+xlpG6uBT6lmIJELd0k59FyupNu4dPvX5SD
6GGqdx4jk9KvI74usGeOBohmF0phTHkrWKBxXiyT0oS8zjnJlEn8ysIpGgO28jjr
LpaHNZ/MXQKBgQDfLNcnS0LzpsS2aK0tzyZU8SMyqVHOGMxj7quhneBq2T6FbiLD
T9TVlYaGNZ0j71vQaLI19qOubWymbautH0Op5KV8owdf4+bf1/NJaPIOzhDUSIjD
Qo01WW31Z9XDSRhKFTnWzmCjBdeIcajyzf10YKsycaAW9lItu8aBrMndnQKBgQDb
nNp/JyRwqjOrNljk7DHEs+SD39kHQzzCfqd+dnTPv2sc06+cpym3yulQcbokULpy
fmRo3bin/pvJQ3aZX/Bdh9woTXqhXDdrrSwWInVYMQPyPk8f/D9mIOJp5FUWMwHD
U+whIZSxsEeE+jtixlWtheKRYkQmzQZXbWdIhYyI3QKBgD+F/6wcZ85QW8nAUykA
3WrSIx/3cwDGdm4NRGct8ZOZjTHjiy9ojMOD1L7iMhRQ/3k3hUsin5LDMp/ryWGG
x4uIaLat40kiC7T4I66DM7P59euqdz3w0PD+VU+h7GSivvsFDdySUt7bNK0AUVLh
dMJfWxDN8QV0b5p3WuWH1U8B
-----END PRIVATE KEY-----
Private-Key: (2048 bit)
modulus:
    00:c5:05:71:d1:23:81:d5:28:08:61:de:c7:a2:72:
    2a:28:8b:30:91:4d:b2:5e:d7:e6:2c:c4:d4:e3:6b:
    85:f2:2b:2a:55:18:81:56:0c:68:59:b3:8e:05:08:
    79:4f:38:e4:95:27:e3:6a:3f:be:30:f7:aa:0c:ec:
    33:d2:df:1a:3d:91:a4:32:64:11:67:d9:81:29:d8:
    40:6a:e6:f7:f7:d3:b2:87:35:19:99:65:49:a4:9f:
    4c:c7:39:21:29:36:66:36:7c:cc:48:48:1c:5e:c2:
    5c:51:14:09:e2:c2:64:9d:ff:c4:c3:bc:72:4c:63:
    d1:6f:00:8b:d6:b9:3b:2f:e6:5d:2d:24:a9:3e:6b:
    dd:4a:e3:eb:4e:dd:47:43:47:b4:a7:a3:95:97:13:
    17:ec:06:b5:b7:83:5c:9d:a3:74:c1:b3:1f:22:e7:
    f6:22:54:e7:0d:02:9c:bb:81:ed:bf:16:2c:18:dd:
    a0:97:24:1e:ab:ea:7b:85:e8:7f:26:46:02:38:af:
    8b:e4:31:1b:0e:94:08:49:0e:76:4f:35:ec:1e:6e:
```

```
    8a:3e:2b:74:37:97:06:e0:6e:63:8a:0f:fc:fd:b2:
    f9:3c:37:ff:a1:51:30:6d:21:7d:1f:46:d6:c6:f8:
    f2:c8:c3:7c:56:44:71:ab:31:29:f6:07:3b:0f:56:
    e0:cb
publicExponent: 65537 (0x10001)
privateExponent:
    0a:22:54:8f:68:5f:26:42:af:e3:b0:dc:dd:eb:37:
    65:ec:7a:ec:0e:6e:0d:58:d7:9b:17:e8:c7:65:e1:
    76:ea:67:7c:07:0d:a8:0a:6d:57:a7:d7:b7:44:8f:
    50:d6:e1:53:16:c1:28:d6:ec:86:82:46:b9:f1:70:
    5c:f9:62:d5:25:e7:a7:3b:e4:75:4e:07:c9:ca:38:
    ce:06:e1:5c:5b:04:44:d6:23:61:f3:86:cd:33:f0:
    74:12:e9:34:c0:7a:93:74:e9:e1:11:ec:7b:a7:4d:
    ae:51:f4:8c:38:69:8a:82:fc:71:01:01:74:12:72:
    54:5e:57:d3:0c:a6:11:b9:95:98:2d:23:80:7f:cc:
    c6:c0:40:3d:65:ba:64:a8:9c:83:d5:0b:32:55:a2:
    01:9d:cc:44:06:4f:8c:71:e0:a5:89:00:02:c5:16:
    28:06:c2:07:05:50:71:58:c6:3b:9f:56:8d:f6:63:
    cd:35:f9:a5:0b:55:54:7e:bc:ae:e7:22:1f:cf:03:
    4d:90:b0:8c:29:23:06:1c:60:f8:e2:24:24:12:c4:
    e7:09:21:f3:68:c8:1d:28:af:67:ad:df:97:02:f0:
    cf:e1:34:f8:78:44:2d:26:49:ae:7d:8c:63:a2:71:
    9a:29:37:a8:d3:54:38:5f:d9:fb:79:ac:76:3d:a5:
    b9
prime1:
    00:e3:c2:50:bf:de:3c:69:f3:32:72:e8:ff:28:25:
    02:af:ed:37:6f:33:05:23:e1:54:96:38:76:41:1c:
    bb:f8:7a:f2:5a:6a:26:b4:b9:08:c8:a3:55:03:6b:
    c0:18:8a:da:a1:5f:53:66:08:27:a1:18:7f:32:b9:
    78:ff:bf:a5:77:0b:33:0a:0e:49:91:af:53:6b:38:
    d9:d2:cf:94:2c:9d:d4:34:e1:9e:a2:84:04:25:3e:
    62:7d:ea:0e:30:2a:d8:28:0b:b0:18:a7:23:f4:83:
    56:be:e3:fb:23:6f:5f:a8:dd:84:08:e2:90:ff:17:
    bd:5c:fa:a6:b3:b4:7e:cf:47
prime2:
    00:dd:73:6d:f2:36:64:f7:f8:9c:a9:b5:fd:1f:2a:
    31:2f:38:d2:be:c7:05:0a:ce:2f:5c:2f:f3:b3:06:
    ae:72:38:80:b5:3f:3d:93:f3:98:0e:7b:58:bc:93:
    06:70:b3:ec:65:a4:6e:ae:05:3e:a5:98:82:44:2d:
    dd:24:e7:d1:72:ba:93:6e:e1:d3:ef:5f:94:83:e8:
    61:aa:77:1e:23:93:d2:af:23:be:2e:b0:67:8e:06:
    88:66:17:4a:61:4c:79:2b:58:a0:71:5e:2c:93:d2:
    84:bc:ce:39:c9:94:49:fc:ca:c2:29:1a:03:b6:f2:
    38:eb:2e:96:87:35:9f:cc:5d
```

```
exponent1:
    00:df:2c:d7:27:4b:42:f3:a6:c4:b6:68:ad:2d:cf:
    26:54:f1:23:32:a9:51:ce:18:cc:63:ee:ab:a1:9d:
    e0:6a:d9:3e:85:6e:22:c3:4f:d4:d5:95:86:86:35:
    9d:23:ef:5b:d0:68:b2:35:f6:a3:ae:6d:6c:a6:6d:
    ab:ad:1f:43:a9:e4:a5:7c:a3:07:5f:e3:e6:df:d7:
    f3:49:68:f2:0e:ce:10:d4:48:88:c3:42:8d:35:59:
    6d:f5:67:d5:c3:49:18:4a:15:39:d6:ce:60:a3:05:
    d7:88:71:a8:f2:cd:fd:74:60:ab:32:71:a0:16:f6:
    52:2d:bb:c6:81:ac:c9:dd:9d
exponent2:
    00:db:9c:da:7f:27:24:70:aa:33:ab:36:58:e4:ec:
    31:c4:b3:e4:83:df:d9:07:43:3c:c2:7e:a7:7e:76:
    74:cf:bf:6b:1c:d3:af:9c:a7:29:b7:ca:e9:50:71:
    ba:24:50:ba:72:7e:64:68:dd:b8:a7:fe:9b:c9:43:
    76:99:5f:f0:5d:87:dc:28:4d:7a:a1:5c:37:6b:ad:
    2c:16:22:75:58:31:03:f2:3e:4f:1f:fc:3f:66:20:
    e2:69:e4:55:16:33:01:c3:53:ec:21:21:94:b1:b0:
    47:84:fa:3b:62:c6:55:ad:85:e2:91:62:44:26:cd:
    06:57:6d:67:48:85:8c:88:dd
coefficient:
    3f:85:ff:ac:1c:67:ce:50:5b:c9:c0:53:29:00:dd:
    6a:d2:23:1f:f7:73:00:c6:76:6e:0d:44:67:2d:f1:
    93:99:8d:31:e3:8b:2f:68:8c:c3:83:d4:be:e2:32:
    14:50:ff:79:37:85:4b:22:9f:92:c3:32:9f:eb:c9:
    61:86:c7:8b:88:68:b6:ad:e3:49:22:0b:b4:f8:23:
    ae:83:33:b3:f9:f5:eb:aa:77:3d:f0:d0:f0:fe:55:
    4f:a1:ec:64:a2:be:fb:05:0d:dc:92:52:de:db:34:
    ad:00:51:52:e1:74:c2:5f:5b:10:cd:f1:05:74:6f:
    9a:77:5a:e5:87:d5:4f:01
```

プライベートキーは、使用しないときは安全な場所に保管してください。

2. 以前のステップで作成したプライベートキーを使用して、X.509 証明書を生成します。この例では、証明書は 365 日で期限切れになり、それ以降は信頼されません。有効期限は適切に設定してください。`tr -d "\n"` コマンドは、出力から改行文字 (改行) を削除します。プロンプトが表示されたら、共通名を指定する必要がありますが、その他のフィールドは空白のままにしておくことができます。

```
$ openssl req -new -x509 -key private-key.pem -days 365 | tr -d "\n" >
 certificate.pem
```

この結果、次のような出力が得られます。

```
Enter pass phrase for private-key.pem: xxxxxxx
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) []:
State or Province Name (full name) []:
Locality Name (eg, city) []:
Organization Name (eg, company) []:
Organizational Unit Name (eg, section) []:
Common Name (eg, fully qualified host name) []:example.com
Email Address []:
```

> **Note**
>
> AWS プロビジョニングには共通名は必要ありません。内部ドメイン名またはパブリックドメイン名は任意です。

次のコマンドを使用して、証明書を取得できます。

```
$ cat certificate.pem
```

出力は、改行のない長い PEM エンコード文字列で、先頭が `-----BEGIN CERTIFICATE-----` で、その後に `-----END CERTIFICATE-----` が続きます。

2. X.509 証明書を RIR の RDAP レコードにアップロードする

以前に作成した証明書を、RIR の RDAP レコードに追加します。エンコードされた部分の前後の `-----BEGIN CERTIFICATE-----` および `-----END CERTIFICATE-----` 文字列を、必ず含めます。このコンテンツはすべて、長い 1 行にする必要があります。RDAP を更新する手順は、ご使用の RIR によって異なります。

- ARIN の場合は、Account Manager ポータルを使用して、アドレス範囲を表す「ネットワーク情報」オブジェクトの「パブリックコメント」セクションに証明書を追加してください。組織の [comments] セクションには追加しないでください。
- RIPE の場合は、証明書を新しい「descr」フィールドとして、アドレス範囲を表す「inetnum」または「inet6num」オブジェクトに追加します。これらは通常、RIPE Database ポータルの「マイリソース」セクションにあります。組織の [コメント] セクションや上記オブジェクトの「備考」フィールドには追加しないでください。
- APNIC の場合は、証明書を電子メールで helpdesk@apnic.net に送信し、アドレス範囲の "remarks" フィールドに手動で追加します。APNIC の IP アドレスに関する正規連絡先に電子メールを送信します。

以下のプロビジョニング段階が完了したら、RIR の記録から証明書を削除できます。

3. RIR に ROA オブジェクトを作成する

アドレス範囲をアドバタイズするために Amazon ASN 16509 および 14618 を承認し、また、アドレス範囲をアドバタイズすることが現在許可されている ASN も承認するために、ROA オブジェクトを作成します。AWS GovCloud (US) Regions については、16509 および 14618 ではなく ASN 8987 を承認してください。持ち込む CIDR のサイズに最大長を設定する必要があります。持ち込める最も具体的な IPv4 プレフィクスは /24 です。提供できる最も具体的な IPv6 アドレス範囲は、パブリックにアドバタイズ可能な CIDR の場合は /48、パブリックにアドバタイズ可能でない CIDR の場合は /56 です。

> **Important**
>
> Amazon VPC IP Address Manager (IPAM) 用の ROA オブジェクトを作成する場合、ROA を作成するときには、IPv4 CIDR に対して、/24 のIP アドレスのプレフィックスの最大長を設定する必要があります。IPv6 CIDR については、アドバタイズ可能なプールに追加する場合、IP アドレスのプレフィックスの最大長は /48 である必要があります。これにより、パブリック IP アドレスを AWS リージョンごとに分割して利用する柔軟性がもたらされます。IPAM では、設定した最大長が適用されます。IPAM への BYOIP アドレスの詳細については、Amazon VPC IPAM ユーザーガイド の「チュートリアル: IPAM への BYOIP アドレス CIDR」を参照してください。

ROA が Amazon で使用できるようになるまで最大 24 時間かかる場合があります。詳細については、RIRを参照してください。

- ARIN — ROA のリクエスト数
- RIPE — ROA の管理
- APNIC— 経路管理

アドバタイズメントをオンプレミスのワークロードから AWS に移行する場合は、Amazon の ASN の ROA を作成する前に、既存の ASN 向けの ROA を作成する必要があります。これを行わないと、既存のルーティングとアドバタイズメントに影響を与える可能性があります。

> **Important**
>
> Amazon がお客様の IP アドレス範囲をアドバタイズし、引き続きアドバタイズするには、Amazon ASN の ROA が上記のガイドラインに準拠している必要があります。お客様の ROA が基準を満たしていない場合、Amazon はお客様の IP アドレス範囲のアドバタイズを停止する権利を留保します。

> **Note**
>
> このステップは、パブリックにアドバタイズ可能でない IPv6 アドレス空間には必要ありません。

BYOIP をオンボーディングする

BYOIP のオンボーディングプロセスには、ニーズに応じて次のタスクがあります。

トピック

- AWS でパブリックにアドバタイズ可能なアドレス範囲をプロビジョニングする
- パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングする
- AWS を通じてアドレス範囲をアドバタイズする
- アドレス範囲のプロビジョニング解除

AWS でパブリックにアドバタイズ可能なアドレス範囲をプロビジョニングする

AWS で使用するアドレス範囲をプロビジョニングする場合は、当該範囲の管理者であることを証明し、Amazon による当該範囲のアドバタイズを承認します。また、署名済みの認可メッセージを使

用して、アドレス範囲を管理していることを確認します。このメッセージは、X.509 証明書で RDAP レコードを更新するときに使用した自己署名 X.509 キーペアで署名されます。AWS には、RIR に提示する暗号署名付き認可メッセージが必要です。RIR は、RDAP に追加した証明書に対して署名を認証し、ROA に対して認証の詳細をチェックします。

アドレス範囲をプロビジョニングするには

1. メッセージを構成する

   プレーンテキスト認可メッセージを作成します。メッセージの形式は以下のとおりです。日付はメッセージの有効期限日になります。

   ```
   1|aws|account|cidr|YYYYMMDD|SHA256|RSAPSS
   ```

   アカウント番号、アドレス範囲、および有効期限日を独自の値に置き換え、次のようなメッセージを作成します。

   ```
   text_message="1|aws|0123456789AB|198.51.100.0/24|20211231|SHA256|RSAPSS"
   ```

   これは ROA メッセージと外観が似ているので、混同しないでください。

2. メッセージに署名する

   以前に作成したプライベートキーを使用して、プレーンテキストメッセージに署名します。このコマンドが返す署名は長い文字列となります。また、次の手順で使用する必要があります。

   > **Important**
   >
   > このコマンドをコピーして貼り付けることをお勧めします。メッセージの内容を除いて、いずれの値も変更または置換しないでください。

   ```
   signed_message=$( echo -n $text_message | openssl dgst -sha256 -sigopt
    rsa_padding_mode:pss -sigopt rsa_pss_saltlen:-1 -sign private-key.pem -keyform PEM
    | openssl base64 | tr -- '+=/' '-_~' | tr -d "\n")
   ```

3. アドレスのプロビジョニング

   AWS CLIprovision-byoip-cidr コマンドを使用して、アドレス範囲をプロビジョニングします。--cidr-authorization-context オプションは、以前に作成したメッセージと署名の文字列を使用します。

   > **Important**
   >
   > AWS CLI 設定 Default region name と異なる場合に BYOIP 範囲をプロビジョニングする AWS リージョンを指定する必要があります。

   ```
   aws ec2 provision-byoip-cidr --cidr address-range --cidr-authorization-context
    Message="$text_message",Signature="$signed_message" --region us-east-1
   ```

   アドレス範囲のプロビジョニングは非同期オペレーションであるため、呼び出しはすぐに戻りますが、アドレスの範囲は、そのステータスが pending-provision から provisioned に変わるまで使用できません。

4. 進行状況をモニタリングする

   ほとんどのプロビジョニングは 2 時間以内に完了しますが、パブリックにアドバタイズ可能な範囲のプロビジョニングプロセスが完了するまでに最大 1 週間かかる場合があります。この例のように、describe-byoip-cidrs コマンドを使用して進行状況をモニタリングします。

   ```
   aws ec2 describe-byoip-cidrs --max-results 5 --region us-east-1
   ```

   プロビジョニング中に問題が発生してステータスが failed-provision になった場合は、問題の解決後に provision-byoip-cidr コマンドを再度実行する必要があります。

パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングする

デフォルトでは、アドレス範囲はインターネットにパブリックにアドバタイズ可能になるようにプロビジョニングされます。パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングできます。パブリックにアドバタイズできないルートの場合、プロビジョニングプロセスは通常、数分以内に完了します。非パブリックアドレス範囲の IPv6 CIDR ブロックを VPC に関連付ける場合、IPv6 CIDR には、AWS Direct Connect、AWS Site-to-Site VPN、Amazon VPC トランジット

ゲートウェイなどの IPv6 をサポートするハイブリッド接続オプションを介してのみアクセスできます。

非パブリックアドレス範囲をプロビジョニングする場合、ROA は必要ありません。

> **Important**
>
> - ユーザーは、プロビジョニング中にアドレス範囲がパブリックにアドバタイズ可能かどうかのみ指定できます。アドバタイズ可能なステータスは、後で変更できません。
> - Amazon VPC は、ユニークローカルアドレス (ULA) CIDR をサポートしていません。すべての VPC には一意の IPv6 CIDR が必要です。2 つの VPC が同じ IPv6 CIDR 範囲を持つことはできません。

パブリックにアドバタイズ可能でない IPv6 アドレス範囲をプロビジョニングするには、次の provision-byoip-cidr コマンドを使用します。

```
aws ec2 provision-byoip-cidr --cidr address-range --cidr-authorization-context
 Message="$text_message",Signature="$signed_message" --no-publicly-advertisable --
region us-east-1
```

AWS を通じてアドレス範囲をアドバタイズする

アドレス範囲をプロビジョニングすると、公開することができるようになります。プロビジョンした正確なアドレス範囲をアドバタイズする必要があります。プロビジョンしたアドレス範囲の一部のみアドバタイズすることはできません。

パブリックにアドバタイズされない IPv6 アドレス範囲をプロビジョニングした場合、このステップを実行する必要はありません。

アドレス範囲は、AWS からアドバタイズする前に、他の場所からのアドバタイズを停止することをお勧めします。他の場所から IP アドレス範囲を公開し続ける場合、当社では、その IP アドレス範囲を信頼してサポートしたり、問題をトラブルシューティングすることができなくなります。具体的には、そのアドレス範囲へのトラフィックが当社のネットワークに入るのを保証できません。

ダウンタイムを最小限に抑えるには、アドレス範囲がアドバタイズされる前にご使用のアドレスプールからアドレスを使用するように AWS リソースを設定してから、同時に現在の場所からのアドバタイズを停止して、AWS からのアドバタイズを開始します。アドレスプールからの Elastic IP アドレスの割り当ての詳細については、Elastic IP アドレスを割り当てるを参照してください。

制限事項

- アドレス範囲が毎回異なる場合でも、advertise-byoip-cidr コマンドは 10 秒ごとに最大 1 回しか実行できません。
- アドレス範囲が毎回異なる場合でも、withdraw-byoip-cidr コマンドは 10 秒ごとに最大 1 回しか実行できません。

アドレス範囲を公開するには、以下のadvertise-byoip-cidrコマンドを使用します。

```
aws ec2 advertise-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲の公開を停止するには、以下のwithdraw-byoip-cidrコマンドを使用します。

```
aws ec2 withdraw-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲のプロビジョニング解除

AWS でアドレス範囲の使用を停止するには、まず Elastic IP アドレスをリリースし、アドレスプールからまだ割り当てられている IPv6 CIDR ブロックの関連付けを解除します。次に、アドレス範囲のアドバタイズを停止し、最後にアドレス範囲のプロビジョニングを解除します。

アドレス範囲のプロビジョニングを部分的に解除することはできません。AWS でより具体的なアドレス範囲を使用する場合は、アドレス範囲全体のプロビジョニングを解除し、より具体的なアドレス範囲をプロビジョニングします。

(IPv4) 各 Elastic IP アドレスをリリースするには、以下の release-address コマンドを使用します。

```
aws ec2 release-address --allocation-id eipalloc-12345678abcabcabc --region us-east-1
```

(IPv6) IPv6 CIDR ブロックの関連付けを解除するには、次の disassociate-vpc-cidr-block コマンドを使用します。

```
aws ec2 disassociate-vpc-cidr-block --association-id vpc-cidr-assoc-12345abcd1234abc1
 --region us-east-1
```

アドレス範囲の公開を停止するには、以下のwithdraw-byoip-cidrコマンドを使用します。

```
aws ec2 withdraw-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲のプロビジョニングを解除するには、以下のdeprovision-byoip-cidrコマンドを使用します。

```
aws ec2 deprovision-byoip-cidr --cidr address-range --region us-east-1
```

アドレス範囲のプロビジョニングを解除するには、最長 1 日かかります。

アドレス範囲を操作する

ユーザーは、アカウントでプロビジョニングした IPv4 と IPv6 のアドレス範囲の表示と使用が可能です。

IPv4 アドレス範囲

IPv4 アドレスプールから Elastic IP アドレスを作成し、EC2 インスタンス、NAT ゲートウェイ、Network Load Balancer などの AWS リソースで使用できます。

アカウントでプロビジョニングした IPv4 アドレスプールに関する情報を表示するには、次の describe-public-ipv4-pools コマンドを使用します。

```
aws ec2 describe-public-ipv4-pools --region us-east-1
```

IPv4 アドレスプールから Elastic IP アドレスを作成するには、allocate-address コマンドを使用します。--public-ipv4-poolオプションを使用して、describe-byoip-cidrsが返すアドレスプールの ID を指定したり、--address オプションを使用して、プロビジョニングしたアドレス範囲からのアドレスを指定したりすることができます。

IPv6 アドレス範囲

アカウントでプロビジョニングした IPv6 アドレスプールに関する情報を表示するには、次の describe-ipv6-pools コマンドを使用します。

```
aws ec2 describe-ipv6-pools --region us-east-1
```

VPC を作成し、IPv6 アドレスプールから IPv6 CIDR を指定するには、次の create-vpc コマンドを使用します。Amazon が IPv6 アドレスプールから IPv6 CIDR を選択できるようにするには、[--ipv6-cidr-block] オプションを省略します。

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16 --ipv6-cidr-block ipv6-cidr --ipv6-
pool pool-id --region us-east-1
```

IPv6 アドレスプールからの IPv6 CIDR ブロックを VPC に関連付けるには、次の associate-vpc-cidr-block コマンドを使用します。Amazon が IPv6 アドレスプールから IPv6 CIDR を選択できるようにするには、[--ipv6-cidr-block] オプションを省略します。

```
aws ec2 associate-vpc-cidr-block --vpc-id vpc-123456789abc123ab --ipv6-cidr-block ipv6-cidr --ipv6-pool pool-id --region us-east-1
```

VPC および関連する IPv6 アドレスプール情報を表示するには、describe-vpcs コマンドを使用します。特定の IPv6 アドレスプールから関連付けられた IPv6 CIDR ブロックに関する情報を表示するには、次の get-associated-ipv6-pool-cidrs コマンドを使用します。

```
aws ec2 get-associated-ipv6-pool-cidrs --pool-id pool-id --region us-east-1
```

VPC から IPv6 CIDR ブロックの関連付けを解除すると、IPv6 アドレスプールに戻されます。

BYOIP を検証する

1. 自己署名 x.509 キーペアを検証する

   whois コマンドで、証明書がアップロードされており、かつ、有効であることを確認します。

   ARIN の場合、whois -h whois.arin.net r + *2001:0DB8:6172::/48* を使用してアドレス範囲の RDAP レコードを検索します。コマンド出力の NetRange (ネットワーク範囲) については、「Public Comments」セクションを確認してください。証明書は、アドレス範囲の「Public Comments」セクションに追加する必要があります。

   次のコマンドを使用して、証明書を含む Public Comments を検査できます。

   ```
   whois -h whois.arin.net r + 2001:0DB8:6172::/48 | grep Comments | grep BEGIN
   ```

   これにより、キーの内容を含む出力が返されます。これは次のようになります。

   ```
   Public Comments:
   -----BEGIN CERTIFICATE-----
   MIID1zCCAr+gAwIBAgIUBkRPNSLrPqbRAFP8RDAHSP+I1TowDQYJKoZIhvcNAQE
   LBQAwezELMAkGA1UEBhMCTloxETAPBgNVBAgMCEF1Y2tsYW5kMREwDwYDVQQHDA
   hBdWNrbGFuZDEcMBoGA1UECgwTQW1hem9uIFdlYiBTZXJ2aWNlczETMBEGA1UEC
   wwKQllPSVAgRGVtbzETMBEGA1UEAwwKQllPSVAgRGVtbzAeFw0yMTEyMDcyMDI0
   NTRaFw0yMjEyMDcyMDI0NTRaMHsxCzAJBgNVBAYTAk5aMREwDwYDVQQIDAhBdWN
   rbGFuZDERMA8GA1UEBwwIQXVja2xhbmQxHDAaBgNVBAoME0FtYXpvbiBXZWIgU2
   ```

```
VydmljZXMxEzARBgNVBAsMCkJZT0lQIERlbW8xEzARBgNVBAMMCkJZT0lQIERlb
W8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCfmacvDp0wZ0ceiXXc
R/q27mHI/U5HKt7SST4X2eAqufR9wXkfNanAEskgAseyFypwEEQr4CJijI/5hp9
prh+jsWHWwkFRoBRR9FBtwcU/45XDXLga7D3stsI5QesHVRwOaXUdprAnndaTug
mDPkD0vrl475JWDSIm+PUxGWLy+60aBqiaZq35wU/x+wXlAqBXg4MZK2KoUu27k
Yt2zhmy0S7Ky+oRfRJ9QbAiSu/RwhQbh5Mkp1ZnVIc7NqnhdeIW48QaYjhMlUEf
xdaqYUinzz8KpjfADZ4Hvqj9jWZ/eXo/9b2rGlHWkJsbhr0VEUyAGu1bwkgcdww
3A7NjOxQbAgMBAAGjUzBRMB0GA1UdDgQWBBStFyujN6SYBr2glHpGt0XGF7GbGT
AfBgNVHSMEGDAWgBStFyujN6SYBr2glHpGt0XGF7GbGTAPBgNVHRMBAf8EBTADA
QH/MA0GCSqGSIb3DQEBCwUAA4IBAQBX6nn6YLhz521lfyVfxY0t6o3410bQAeAF
08ud+ICtmQ4IO4A4B7zV3zIVYr0clrOOaFyLxngwMYN0XY5tVhDQqk4/gmDNEKS
Zy2QkX4Eg0YUWVzOyt6fPzjOvJLcsqc1hcF9wySL507XQz76Uk5cFypBOzbnk35
UkWrzA9KK97cXckfIESgK/k1N4ecwxwG6VQ8mBGqVpPpey+dXpzzzv1iBKN/VY4
ydjgH/LBfdTsVarmmy2vtWBxwrqkFvpdhSGCvRDl/qdO/GIDJi77dmZWkh/ic90
MNk1f38gs1jrCj8lThoar17Uo9y/Q5qJIsoNPyQrJRzqFU9F3FBjiPJF
-----END CERTIFICATE-----
```

RIPE の場合、whois -r -h whois.ripe.net *2001:0DB8:7269::/48* を使用してアドレス範囲の RDAP レコードを検索します。コマンド出力の inetnum オブジェクト (ネットワーク範囲) については、「descr」セクションを確認してください。証明書は、アドレス範囲の新しい descr フィールドとして追加する必要があります。

次のコマンドを使用して、証明書を含む descr を検査できます。

```
whois -r -h whois.ripe.net 2001:0DB8:7269::/48 | grep descr | grep BEGIN
```

これにより、キーの内容を含む出力が返されます。これは次のようになります。

```
descr:
-----BEGIN CERTIFICATE-----MIID1zCCAr+gAwIBAgIUBkRPNSLrPqbRAFP8
RDAHSP+I1TowDQYJKoZIhvcNAQELBQAwezELMAkGA1UEBhMCTloxETAPBgNVBAg
MCEF1Y2tsYW5kMREwDwYDVQQHDAhBdWNrbGFuZDEcMBoGA1UECgwTQW1hem9uIF
dlYiBTZXJ2aWNlczETMBEGA1UECwwKQllPSVAgRGVtbzETMBEGA1UEAwwKQllPS
VAgRGVtbzAeFw0yMTEyMDcyMDI0NTRaFw0yMjEyMDcyMDI0NTRaMHsxCzAJBgNV
BAYTAk5aMREwDwYDVQQIDAhBdWNrbGFuZDERMA8GA1UEBwwIQXVja2xhbmQxHDA
aBgNVBAoME0FtYXpvbiBXZWIgU2VydmljZXMxEzARBgNVBAsMCkJZT0lQIERlbW
8xEzARBgNVBAMMCkJZT0lQIERlbW8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwg
gEKAoIBAQCfmacvDp0wZ0ceiXXcR/q27mHI/U5HKt7SST4X2eAqufR9wXkfNanA
EskgAseyFypwEEQr4CJijI/5hp9prh+jsWHWwkFRoBRR9FBtwcU/45XDXLga7D3
stsI5QesHVRwOaXUdprAnndaTugmDPkD0vrl475JWDSIm+PUxGWLy+60aBqiaZq
35wU/x+wXlAqBXg4MZK2KoUu27kYt2zhmy0S7Ky+oRfRJ9QbAiSu/RwhQbh5Mkp
1ZnVIc7NqnhdeIW48QaYjhMlUEfxdaqYUinzz8KpjfADZ4Hvqj9jWZ/eXo/9b2r
```

```
GlHWkJsbhr0VEUyAGu1bwkgcdww3A7NjOxQbAgMBAAGjUzBRMB0GA1UdDgQWBBS
tFyujN6SYBr2glHpGt0XGF7GbGTAfBgNVHSMEGDAWgBStFyujN6SYBr2glHpGt0
XGF7GbGTAPBgNVHRMBAf8EBTADAQH/MA0GCSqGSIb3DQEBCwUAA4IBAQBX6nn6Y
Lhz521lfyVfxY0t6o3410bQAeAF08ud+ICtmQ4IO4A4B7zV3zIVYr0clrOOaFyL
xngwMYN0XY5tVhDQqk4/gmDNEKSZy2QkX4Eg0YUWVzOyt6fPzjOvJLcsqc1hcF9
wySL507XQz76Uk5cFypBOzbnk35UkWrzA9KK97cXckfIESgK/k1N4ecwxwG6VQ8
mBGqVpPpey+dXpzzzv1iBKN/VY4ydjgH/LBfdTsVarmmy2vtWBxwrqkFvpdhSGC
vRDl/qdO/GIDJi77dmZWkh/ic90MNk1f38gs1jrCj8lThoar17Uo9y/Q5qJIsoN
PyQrJRzqFU9F3FBjiPJF
-----END CERTIFICATE-----
```

APNIC の場合、`whois -h whois.apnic.net` *`2001:0DB8:6170::/48`* を使用して BYOIP アドレス範囲の RDAP レコードを検索します。コマンド出力の `inetnum` オブジェクト (ネットワーク範囲) については、「`remarks`」セクションを確認してください。証明書は、アドレス範囲の新しい `remarks` フィールドとして追加する必要があります。

次のコマンドを使用して、証明書を含む `remarks` を検査できます。

```
whois -h whois.apnic.net 2001:0DB8:6170::/48 | grep remarks | grep BEGIN
```

これにより、キーの内容を含む出力が返されます。これは次のようになります。

```
remarks:
-----BEGIN CERTIFICATE-----
MIID1zCCAr+gAwIBAgIUBkRPNSLrPqbRAFP8RDAHSP+I1TowDQYJKoZIhvcNAQE
LBQAwezELMAkGA1UEBhMCTloxETAPBgNVBAgMCEF1Y2tsYW5kMREwDwYDVQQHDA
hBdWNrbGFuZDEcMBoGA1UECgwTQW1hem9uIFdlYiBTZXJ2aWNlczETMBEGA1UEC
wwKQllPSVAgRGVtbzETMBEGA1UEAwwKQllPSVAgRGVtbzAeFw0yMTEyMDcyMDI0
NTRaFw0yMjEyMDcyMDI0NTRaMHsxCzAJBgNVBAYTAk5aMREwDwYDVQQIDAhBdWN
rbGFuZDERMA8GA1UEBwwIQXVja2xhbmQxHDAaBgNVBAoME0FtYXpvbiBXZWIgU2
VydmljZXMxEzARBgNVBAsMCkJZT0lQIERlbW8xEzARBgNVBAMMCkJZT0lQIERlb
W8wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCfmacvDp0wZ0ceiXXc
R/q27mHI/U5HKt7SST4X2eAqufR9wXkfNanAEskgAseyFypwEEQr4CJijI/5hp9
prh+jsWHWwkFRoBRR9FBtwcU/45XDXLga7D3stsI5QesHVRwOaXUdprAnndaTug
mDPkD0vrl475JWDSIm+PUxGWLy+60aBqiaZq35wU/x+wXlAqBXg4MZK2KoUu27k
Yt2zhmy0S7Ky+oRfRJ9QbAiSu/RwhQbh5Mkp1ZnVIc7NqnhdeIW48QaYjhMlUEf
xdaqYUinzz8KpjfADZ4Hvqj9jWZ/eXo/9b2rGlHWkJsbhr0VEUyAGu1bwkgcdww
3A7NjOxQbAgMBAAGjUzBRMB0GA1UdDgQWBBStFyujN6SYBr2glHpGt0XGF7GbGT
AfBgNVHSMEGDAWgBStFyujN6SYBr2glHpGt0XGF7GbGTAPBgNVHRMBAf8EBTADA
QH/MA0GCSqGSIb3DQEBCwUAA4IBAQBX6nn6YLhz521lfyVfxY0t6o3410bQAeAF
08ud+ICtmQ4IO4A4B7zV3zIVYr0clrOOaFyLxngwMYN0XY5tVhDQqk4/gmDNEKS
Zy2QkX4Eg0YUWVzOyt6fPzjOvJLcsqc1hcF9wySL507XQz76Uk5cFypBOzbnk35
```

```
UkWrzA9KK97cXckfIESgK/k1N4ecwxwG6VQ8mBGqVpPpey+dXpzzzv1iBKN/VY4
ydjgH/LBfdTsVarmmy2vtWBxwrqkFvpdhSGCvRDl/qdO/GIDJi77dmZWkh/ic90
MNk1f38gs1jrCj8lThoar17Uo9y/Q5qJIsoNPyQrJRzqFU9F3FBjiPJF
-----END CERTIFICATE-----
```

2. ROA オブジェクトの作成を検証する

   RIPEstat データ API コマンドを使用して、ROA オブジェクトが正常に作成されたことを検証します。Amazon ASN 16509 および 14618、ならびにそのアドレス範囲をアドバタイズすることが現在承認されている ASN に対して、アドレス範囲をテストしてください。

   次のコマンドを使用して、アドレス範囲で異なる Amazon ASN の ROA オブジェクトを検査できます。

   ```
   curl --location --request GET "https://stat.ripe.net/data/rpki-validation/data.json?resource=ASN&prefix=CIDR
   ```

   この出力例では、レスポンスには、Amazon ASN 16509 について "status": "valid" の結果があります。これは、アドレス範囲についての ROA オブジェクトが正常に作成されたことを示します。

   ```
   {
       "messages": [],
       "see_also": [],
       "version": "0.3",
       "data_call_name": "rpki-validation",
       "data_call_status": "supported",
       "cached": false,
       "data": {
           "validating_roas": [
               {
                   "origin": "16509",
                   "prefix": "2001:0DB8::/32",
                   "max_length": 48,
                   "validity": "valid"
               },
               {
                   "origin": "14618",
                   "prefix": "2001:0DB8::/32",
                   "max_length": 48,
                   "validity": "invalid_asn"
               },
   ```

```
            {
                "origin": "64496",
                "prefix": "2001:0DB8::/32",
                "max_length": 48,
                "validity": "invalid_asn"
            }
        ],
        "status": "valid",
        "validator": "routinator",
        "resource": "16509",
        "prefix": "2001:0DB8::/32"
    },
    "query_id": "20230224152430-81e6384e-21ba-4a86-852a-31850787105f",
    "process_time": 58,
    "server_id": "app116",
    "build_version": "live.2023.2.1.142",
    "status": "ok",
    "status_code": 200,
    "time": "2023-02-24T15:24:30.773654"
}
```


“unknown” の状態は、アドレス範囲についての ROA オブジェクトが作成されていないことを示します。“invalid_asn” の状態は、アドレス範囲についての ROA オブジェクトが正常に作成されなかったことを示します。

リージョナルな可用性

BYOIP 機能は現在、AWS中国リージョンを除くすべての商用リージョンで利用できます。

Local Zone の可用性

Local Zone は、地理的にユーザーに近い場所に位置する AWS リージョンを拡張したものです。Local Zones は「ネットワークボーダーグループ」にグループ化されます。AWS で、ネットワークボーダーグループは、AWS がパブリック IP アドレスをアドバタイズするアベイラビリティーゾーン (AZ)、Local Zones、Wavelength Zones のコレクションです。Local Zones は、AWS ネットワークとこれらのゾーンのリソースにアクセスするお客様との間のレイテンシーや物理的距離を最小限に抑えるために、AWS リージョン内の AZ とは異なるネットワークボーダーグループを持つ場合があります。

`--network-border-group` オプションを使用すると、以下の Local Zone ネットワークボーダーグループに BYOIPv4 アドレス範囲をプロビジョニングしてアドバタイズできます。

- us-east-1-dfw-2
- us-west-2-lax-1
- us-west-2-phx-2

Local Zones を有効にしている場合 (「Enable a Local Zone」を参照)、BYOIPv4 CIDR のプロビジョニングとアドバタイズをするときにネットワークボーダーグループを選択できます。EIP とそれが関連付けられている AWS リソースは同じネットワークボーダーグループに属している必要があるため、ネットワークボーダーグループは慎重に選択してください。

> **Note**
>
> 現時点では、Local Zones で BYOIPv6 アドレス範囲をプロビジョニングまたはアドバタイズすることはできません。

詳細

詳細については、AWS オンラインテックトークの「自分の IP アドレス使用の詳細」を参照してください。

Elastic IP アドレス

Elastic IP アドレスは、動的なクラウドコンピューティングのために設計された静的 IPv4 アドレスです。Elastic IP アドレスはユーザーの AWS アカウントに割り当てられ、リリースするまでユーザーのアドレスになります。Elastic IP アドレスを使用すると、アドレスをアカウント内の別のインスタンスに迅速に再マッピングすることで、インスタンスやソフトウェアの障害をマスクできます。または、ドメインがインスタンスを参照するように、ドメインの DNS レコードに Elastic IP アドレスを指定することもできます。詳細については、ドメインレジストラのドキュメント、または Amazon Linux インスタンスでの動的な DNS のセットアップ を参照してください。

Elastic IP アドレスは、インターネットからアクセス可能なパブリック IPv4 アドレスです。インスタンスにパブリック IPv4 アドレスがない場合は、Elastic IP アドレスをインスタンスに関連付けて、インターネットとの通信を有効にできます。例えば、これにより、ローカルコンピュータからインスタンスに接続できます。

コンテンツ

Elastic IP アドレスの料金

AWS では、実行中のインスタンスに関連付けられているパブリック IPv4 アドレスと Elastic IP アドレスを含む、すべてのパブリック IPv4 アドレスに対して料金が課されます。詳細については、「Amazon VPC の料金」ページの「パブリック IPv4 アドレス」タブを参照してください。

Elastic IP アドレスの基本

Elastic IP アドレスの基本的な特徴を次に示します。

- Elastic IP アドレスは静的であり、時間の経過とともに変わることはありません。
- Elastic IP アドレスは特定のリージョン専用であり、別のリージョンに移動することはできません。
- Elastic IP アドレスは、Amazon が持っている IPv4 アドレスのプールまたはお客様が AWS アカウントに持ち込んだカスタム IPv4 アドレスのプールから割り当てることができます。
- Elastic IP アドレスを使用するには、まずアカウントに 1 つ割り当ててから、それをインスタンスまたはネットワークインターフェイスに関連付けます。
- Elastic IP アドレスをインスタンスと関連付けると、インスタンスのプライマリネットワークインターフェイスとも関連付けられます。Elastic IP アドレスをインスタンスにアタッチされたネットワークインターフェイスと関連付けると、インスタンスとも関連付けられます。
- Elastic IP アドレスをインスタンスまたはそのプライマリネットワークインターフェイスに関連付けると、インスタンスのパブリック IPv4 アドレス (既に割り当てられていた場合) が Amazon のパブリック IPv4 アドレスのプールに戻されます。パブリック IPv4 アドレスを再利用することはできず、パブリック IPv4 アドレスを Elastic IP アドレスに変換することはできません。詳細については、パブリック IPv4 アドレスを参照してください。
- リソースから Elastic IP アドレスの関連付けを解除し、別のリソースと関連付けることができます。予期しない動作を避けるため、変更を行う前に、既存の関連付けで指定されたリソースへのアクティブな接続をすべて閉じていることを確認してください。Elastic IP アドレスを別のリソースに関連付けた後、新しく関連付けられたリソースへの接続を再度開くことができます。

- 関連付けが解除された Elastic IP アドレスは、明示的にリリースするまでアカウントに割り当てられたままです。実行中のインスタンスに関連付けられていない Elastic IP アドレスには、時間単位で少額の料金が課されます。
- パブリック IPv4 アドレスが前回割り当てられたインスタンスに Elastic IP アドレスを関連付けると、インスタンスのパブリック DNS ホスト名は、Elastic IP アドレスに一致するように変更されます。
- パブリック DNS ホスト名を解決すると、インスタンスのパブリック IPv4 アドレスまたは Elastic IP アドレス (インスタンスのネットワークの外部の場合)、およびインスタンスのプライベート IPv4 アドレス (インスタンスのネットワーク内からの場合) となります。
- AWS アカウントに持ち込んだ IP アドレスプールから Elastic IP アドレスを割り当てた場合、Elastic IP アドレス制限にカウントされません。詳細については、Elastic IP アドレスのクォータを参照してください。
- Elastic IP アドレスを割り当てると、Elastic IP アドレスをネットワークボーダーグループに関連付けることができます。これは、CIDR ブロックをアドバタイズする場所です。ネットワークボーダーグループを設定すると、CIDR ブロックがこのグループに制限されます。ネットワークボーダーグループを指定しない場合は、リージョン (`us-west-2` など) のすべてのアベイラビリティーゾーンを含むボーダーグループが自動的に設定されます。
- Elastic IP アドレスは、ネットワークボーダーグループ別に専用になっています。

Elastic IP アドレスの操作

以下のセクションでは、Elastic IP アドレスの使用方法について説明します。

タスク

- Elastic IP アドレスを割り当てる
- Elastic IP アドレスの説明
- Elastic IP アドレスにタグを適用する
- Elastic IP アドレスをインスタンスまたはネットワークインターフェイスに関連付ける
- Elastic IP アドレスの関連付けを解除する
- Elastic IP アドレスを移管する
- Elastic IP アドレスをリリース
- Elastic IP アドレスの復元
- E メールアプリケーション用の逆引き DNS の使用

Elastic IP アドレスを割り当てる

Elastic IP アドレスは、Amazon のパブリック IPv4 アドレスのプールまたは AWS アカウントに持ち込んだカスタム IP アドレスプールから割り当てることができます。AWS アカウントへの独自の IP アドレス範囲の持ち込みの詳細については、Amazon EC2 で自分の IP アドレスを使用する (BYOIP) をご参照ください。

以下のいずれかの方法を使用して、Elastic IP アドレスを割り当てることができます。

Console

Elastic IP アドレスを割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network & Security]、[Elastic IPs] の順に選択します。
3. [Allocate Elastic IP address] を選択します。
4. (オプション) Elastic IP アドレス (EIP) を割り当てるときは、EIP を割り当てるネットワークボーダーグループを選択します。ネットワークボーダーグループは、AWS がパブリック IP アドレスをアドバタイズするアベイラビリティーゾーン (AZ)、Local Zones、または Wavelength Zones のコレクションです。Local Zones と Wavelength Zones は、AWS ネットワークとこれらのゾーンのリソースにアクセスする顧客との間のレイテンシーや物理的距離を最小限に抑えるために、リージョン内の AZ とは異なるネットワークボーダーグループを持つ場合があります。

   > **Important**
   >
   > EIP に関連付ける AWS リソースと同じネットワークボーダーグループに EIP を割り当てる必要があります。あるネットワークボーダーグループ内の EIP は、そのネットワークボーダーグループ内のゾーンでのみアドバタイズでき、他のネットワークボーダーグループで表される他のゾーンではアドバタイズできません。

   Local Zones または Wavelength Zones を有効にしている場合 (詳細については、「Local Zone を有効にする」または「Wavelength Zones を有効にする」を参照)、AZ、Local Zones、または Wavelength Zones のネットワークボーダーグループを選択できます。EIP とそれが関連付けられている AWS リソースは同じネットワークボーダーグループに属している必要があるため、ネットワークボーダーグループは慎重に選択してください。EC2 コンソールを使用して、アベイラビリティーゾーン、Local Zones、または Wavelength Zones が

属するネットワークボーダーグループを表示できます (「Local Zones」を参照)。通常、リージョン内のすべてのアベイラビリティーゾーンは同じネットワークボーダーグループに属しますが、Local Zones や Wavelength Zones はそれぞれ別のネットワークボーダーグループに属します。

Local Zones または Wavelength Zones が有効になっていない場合、EIP を割り当てると、リージョン (`us-west-2` など) のすべての AZ を表すネットワークボーダーグループが定義済みになり、変更することはできません。つまり、このネットワークボーダーグループに割り当てた EIP は、現在のリージョンのすべての AZ でアドバタイズされます。

5. [Public IPv4 address pool (パブリック IPv4 アドレスのプール)] で、以下のいずれかを選択します。

   - [Amazon's pool of IPv4 addresses (Amazon の IP アドレスのプール)] — Amazon の IPv4 アドレスのプールから IPv4 アドレスを割り当てる場合。
   - AWS アカウントに持ち込むパブリック IPv4 アドレス - AWS アカウントに持ち込んだ IP アドレスプールから IPv4 アドレスを割り当てる場合。IP アドレスプールがない場合、このオプションは無効になります。
   - ユーザー所有の IPv4 アドレスのプール - AWS Outpost で使用するために、オンプレミスネットワークから作成したプールから IPv4 アドレスを割り当てる場合。AWS Outpost がない場合、このオプションは無効になります。

6. (オプション) タグを追加または削除します。

   [タグの追加] [新しいタグの追加] を選択して、以下を実行します。

   - [キー] にはキー名を入力します。
   - [値] にキー値を入力します。

   [タグを削除] タグのキーと値の右側にある [削除] を選択します。

7. [Allocate] を選択します。

AWS CLI

Elastic IP アドレスを割り当てるには

allocate-address AWS CLI コマンドを使用します。

PowerShell

Elastic IP アドレスを割り当てるには

New-EC2Address AWS Tools for Windows PowerShell コマンドを使用します。

Elastic IP アドレスの説明

以下のいずれかの方法を使用して、Elastic IP アドレスの情報を取得できます。

Console

Elastic IP アドレスの情報を取得するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. 表示する Elastic IP アドレスを選択してから、[Actions (アクション)]、[View details (詳細の表示)] の順に選択します。

AWS CLI

Elastic IP アドレスの情報を取得するには

describe-addresses AWS CLI コマンドを使用します。

PowerShell

Elastic IP アドレスの情報を取得するには

Get-EC2Address AWS Tools for Windows PowerShell コマンドを使用します。

Elastic IP アドレスにタグを適用する

Elastic IP アドレスにカスタムタグを割り当てて、目的、所有者、環境など、さまざまな方法で分類できます。これにより、割り当てたカスタムタグに基づいて特定の Elastic IP アドレスをすばやく見つけることができるようになります。

Elastic IP アドレスタグを使用したコスト配分の追跡はサポートされていません。

以下のいずれかの方法を使用して、Elastic IP アドレスにタグ付けできます。

Console

Elastic IP アドレスにタグを適用するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. タグ付けする Elastic IP アドレスを選択してから、[Actions (アクション)]、[View details (詳細の表示)] の順に選択します。
4. [Tags (タグ)] タブで、[Manage tags (タグの管理)] を選択します。
5. タグのキーと値のペアを指定します。
6. (オプション) [タグの追加] を選択して、タグを追加します。
7. [Save] を選択します。

AWS CLI

Elastic IP アドレスにタグを適用するには

create-tags AWS CLI コマンドを使用します。

```
aws ec2 create-tags --resources eipalloc-12345678 --tags Key=Owner,Value=TeamA
```

PowerShell

Elastic IP アドレスにタグを適用するには

New-EC2Tag AWS Tools for Windows PowerShell コマンドを使用します。

New-EC2Tag コマンドには、Elastic IP アドレスのタグに使用するキーと値のペアを指定する Tag パラメータが必要です。以下のコマンドでは、Tag パラメータを作成します。

```
PS C:\> $tag = New-Object Amazon.EC2.Model.Tag
 PS C:\> $tag.Key = "Owner"
 PS C:\> $tag.Value = "TeamA"
```

```
PS C:\> New-EC2Tag -Resource eipalloc-12345678 -Tag $tag
```

Elastic IP アドレスをインスタンスまたはネットワークインターフェイスに関連付ける

Elastic IP アドレスをインスタンスに関連付けてインターネットとの通信を有効にする場合、インスタンスがパブリックサブネットに属していることも確認する必要があります。詳細については、Amazon VPC ユーザーガイドの「インターネットゲートウェイ」を参照してください。

以下のいずれかの方法を使用して、Elastic IP アドレスをインスタンスまたはネットワークインターフェイスに関連付けることができます。

Console

Elastic IP アドレスをインスタンスに関連付けるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. 関連付ける Elastic IP アドレスを選択してから、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択します。
4. [リソースタイプ] で、[Instance (インスタンス)] を選択します。
5. 例えば、Elastic IP アドレスを関連付けるインスタンスを選択します。テキストを入力して特定のインスタンスを検索することもできます。
6. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。
7. [Associate] を選択します。

Elastic IP アドレスとネットワークインターフェイスを関連付けるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. 関連付ける Elastic IP アドレスを選択してから、[Actions (アクション)]、[Associate Elastic IP address (Elastic IP アドレスの関連付け)] の順に選択します。
4. [リソースタイプ] で、[ネットワークインターフェイス] を選択します。
5. [ネットワークインターフェイス] で、Elastic IP アドレスを関連付けるネットワークインターフェイスを選択します。テキストを入力して、特定のネットワークインターフェイスを検索することもできます。
6. (オプション) [プライベート IP アドレス] で、Elastic IP アドレスを関連付けるプライベート IP アドレスを指定します。

7. [Associate] を選択します。

AWS CLI

Elastic IP アドレスを関連付けるには

associate-address AWS CLI コマンドを使用します。

PowerShell

Elastic IP アドレスを関連付けるには

Register-EC2Address AWS Tools for Windows PowerShell コマンドを使用します。

Elastic IP アドレスの関連付けを解除する

インスタンスまたはネットワークインターフェイスから Elastic IP アドレスの関連付けをいつでも解除できます。Elastic IP アドレスの関連付けを解除した後、そのアドレスを別のリソースに再度関連付けることができます。

以下のいずれかの方法を使用して、Elastic IP アドレスの関連付けを解除できます。

Console

Elastic IP アドレスの関連付けを解除して再度関連付けするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. 関連付けを解除する Elastic IP アドレスを選択してから、[Actions (アクション)]、[Elastic IP アドレスの関連付けの解除] の順に選択します。
4. [関連付け解除] を選択します。

AWS CLI

Elastic IP アドレスの関連付けを解除するには

disassociate-address AWS CLI コマンドを使用します。

PowerShell

Elastic IP アドレスの関連付けを解除するには

Unregister-EC2Address AWS Tools for Windows PowerShell コマンドを使用します。

Elastic IP アドレスを移管する

このセクションでは、Elastic IP アドレスを ある AWS アカウント から別のアカウントに転送する方法について説明します。Elastic IP アドレスの移管は、次のような状況で役に立ちます。

- 組織の再構築 - Elastic IP アドレス転送を使用すると、ある AWS アカウント から別のアカウントにワークロードをすばやく移動できます。新しい Elastic IP アドレスがセキュリティグループと NACL の許可リストに追加されるのを待つ必要がありません。
- 一元的なセキュリティ管理 - 一元化された AWS セキュリティアカウントを使用して、セキュリティコンプライアンスのために精査された Elastic IP アドレスを追跡および移管できます。
- ディザスタリカバリ – 緊急時には、Elastic IP アドレス移管を使用することで、一般向けインターネットワークロードの IP アドレスをすばやく再マッピングできます。

Elastic IP アドレスの移管には料金はかかりません。

タスク

- Elastic IP アドレスの移管を有効にする
- Elastic IP アドレスの移管を無効にする
- 移管された Elastic IP アドレスを承諾する

Elastic IP アドレスの移管を有効にする

このセクションでは、移管された Elastic IP アドレスを承諾する方法について説明します。Elastic IP アドレスの移管を有効にする際には、以下の制限に注意してください。

- 任意の AWS アカウント (ソースアカウント) から同じ AWS リージョン内の他の AWS アカウント (転送先アカウント) に Elastic IP アドレスを転送できます。
- Elastic IP アドレスを転送する場合、AWS アカウント の間で 2 段階のハンドシェイクが行われます。ソースアカウントが移管を開始してから 7 日間は、転送先アカウントが Elastic IP アドレス移管を受け入れることができます。この 7 日間、ソースアカウントは保留中の移管を (AWS コンソールや AWS CLI コマンドの describe-address-transfers などを使用して) 確認できます。7 日後、移管の有効期限が切れ、Elastic IP アドレスの所有権がソースアカウントに戻ります。
- 移管が受け入れられてから 3 日間、ソースアカウントは受け入れられた移管を (AWS コンソールや AWS CLI コマンドの describe-address-transfers などを使用して) 表示できます。

- AWS は、保留中の Elastic IP アドレス転送リクエストについて、転送先アカウントに通知しません。ソースアカウントの所有者は、承諾する必要がある Elastic IP アドレス転送リクエストがあることを転送先アカウントの所有者に通知する必要があります。
- 転送中の Elastic IP アドレスに関連付けられているタグは、転送が完了するとリセットされます。
- AWS アカウント に持ち込んだパブリック IPv4 アドレスプール (一般的に Bring-Your-Own-IP (BYOIP) アドレスプールと呼ばれる) から割り当てられた Elastic IP アドレスは転送できません。
- リバース DNS レコードが関連付けられている Elastic IP アドレスを移管しようとする場合、移管プロセスを開始することはできますが、関連付けられている DNS レコードが削除されるまで、転送先アカウントは移管を受け入れることができません。
- AWS Outposts を有効にして設定している場合は、カスタマー所有の IP アドレスプール (CoIP) から Elastic IP アドレスを割り当てている可能性があります。CoIP から割り当てられた Elastic IP アドレスを転送することはできません。ただし、AWS RAM を使用して CoIP を別のアカウントと共有することはできます。CoIP の詳細については、AWS Outposts ユーザーガイドの「カスタマー所有 IP アドレス」を参照してください 。
- Amazon VPC IPAM を使用して、AWS Organizations から組織内のアカウントへの Elastic IP アドレスの転送を追跡することができます。詳細については、「IP アドレスの履歴の表示」を参照してください。Elastic IP アドレスが組織外の AWS アカウント に転送されると、その Elastic IP アドレスの IPAM 監査履歴は失われます。

これらのステップは、ソースアカウントで実行する必要があります。

Console

Elastic IP アドレスの移管を有効にするには

1. 転送元となる AWS アカウントを使用していることを確認してください。
2. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
3. ナビゲーションペインで [Elastic IP] を選択します。
4. 移管を有効にする Elastic IP アドレスを 1 つ以上選択し、[Action] (アクション)、[Enable transfer] (移管を有効にする) を選択します。
5. 複数の Elastic IP アドレスを移管する場合は、[Transfer type] (転送タイプ) オプションが表示されます。以下のオプションのいずれかを選択します。
   - Elastic IP アドレスを単一の AWS アカウントに移管する場合は、[Single account] (単一アカウント) を選択します。

- Elastic IP アドレスを複数の AWS アカウントに移管する場合は、[Multiple accounts] (複数アカウント) を選択します。

6. [Transfer account ID] (アカウント ID の移管) に、Elastic IP アドレスの転送先の AWS アカウント ID を入力します。
7. テキストボックスに「**enable**」と入力して移管を確定します。
8. 送信 を選択します。
9. 移管を承諾するには、「移管された Elastic IP アドレスを承諾する」を参照してください。移管を無効にするには、「Elastic IP アドレスの移管を無効にする」を参照してください。

AWS CLI

Elastic IP アドレスの移管を有効にするには

enable-address-transfer コマンドを使用します。

PowerShell

Elastic IP アドレスの移管を有効にするには

Enable-EC2AddressTransfer コマンドを使用します。

Elastic IP アドレスの移管を無効にする

このセクションでは、Elastic IP 移管を有効にした後に Elastic IP 転送を無効にする方法について説明します。

これらのステップは、移管を有効にしたソースアカウントが実行する必要があります。

Console

Elastic IP アドレス移管を無効にするには

1. 転送元となる AWS アカウントを使用していることを確認してください。
2. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
3. ナビゲーションペインで [Elastic IP] を選択します。
4. Elastic IP のリソースリストで、[Transfer status] (移管ステータス) 列を表示するプロパティが有効になっていることを確認します。

5. [Transfer status] (移管ステータス) が [Pending] (保留中) の Elastic IP アドレスを 1 つ以上選択し、[Action] (アクション)、[Disable transfer] (移管を無効にする) を選択します。
6. テキストボックスに「**disable**」と入力して確認します。
7. 送信 を選択します。

AWS CLI

Elastic IP アドレスの移管を無効にするには

disable-address-transfer コマンドを使用します。

PowerShell

Elastic IP アドレスの移管を無効にするには

Disable-EC2AddressTransfer コマンドを使用します。

移管された Elastic IP アドレスを承諾する

このセクションでは、移管された Elastic IP アドレスを承諾する方法について説明します。

Elastic IP アドレスを転送する場合、AWS アカウント の間で 2 段階のハンドシェイクが行われます。ソースアカウントが移管を開始してから 7 日間は、転送先アカウントが Elastic IP アドレス移管を受け入れることができます。この 7 日間、ソースアカウントは保留中の移管を (AWS コンソールや AWS CLI コマンドの describe-address-transfers などを使用して) 確認できます。7 日後、移管の有効期限が切れ、Elastic IP アドレスの所有権がソースアカウントに戻ります。

転送を承諾する際に発生する可能性のある例外と、解決する方法は次のとおりです。

- AddressLimitExceeded: 転送先アカウントが Elastic IP アドレスのクォータを超えている場合、ソースアカウントは Elastic IP アドレス移管を有効にできますが、この例外は転送先アカウントが移管を承諾しようとした場合に発生します。デフォルトでは、すべての AWS アカウントはリージョンあたり 5 つの Elastic IP アドレスに制限されています。制限を増やす方法については、「Elastic IP アドレスのクォータ」を参照してください。
- InvalidTransfer.addressCustomPtrSet: お客様または組織内の誰かが、移管しようとしている Elastic IP アドレスをリバース DNS ルックアップを使用するように設定している場合、ソースアカウントは Elastic IP アドレスの移管を有効にできますが、転送元アカウントが転送を受け入れようとするとこの例外が発生します。この問題を解決するには、転送元アカウントで Elastic IP アド

レスの DNS レコードを削除する必要があります。詳細については、「E メールアプリケーション用の逆引き DNS の使用」を参照してください。

- InvalidTransfer.AddressAssociated: Elastic IP アドレスが ENI や EC2 インスタンスと関連付けられている場合、転送元アカウントはその Elastic IP アドレスに対して移管を有効にできますが、転送元アカウントが移管を受け入れようとするとこの例外が発生します。この問題を解決するには、ソースアカウントが Elastic IP アドレスの関連付けを解除する必要があります。詳細については、「Elastic IP アドレスの関連付けを解除する」を参照してください。

その他の例外については、AWS Support にお問い合わせください。

これらのステップは、転送先アカウントで実行する必要があります。

Console

Elastic IP アドレスの移管を承諾するには

1. 転送先アカウントを使用していることを確認してください。
2. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
3. ナビゲーションペインで [Elastic IP] を選択します。
4. [Action] (アクション)、[Accept transfer] (移管を許可する) を選択します。
5. 転送を受け入れると、移管される Elastic IP アドレスに関連付けられたタグは転送されません。承諾する Elastic IP アドレスの [Name] (名前) タグを定義する場合は、[Create a tag with a key of 'Name' and a value that you specify] ('Name'のキーと指定した値を使用してタグを作成) を選択します。
6. 移管する Elastic IP アドレスを入力します。
7. 複数の移管された Elastic IP アドレスを受け入れる場合は、[Add address] (アドレスを追加) を選択して追加の Elastic IP アドレスを入力します。
8. 送信 を選択します。

AWS CLI

Elastic IP アドレスの移管を承諾するには

accept-address-transfer コマンドを使用します。

PowerShell

Elastic IP アドレスの移管を承諾するには

Approve-EC2AddressTransfer コマンドを使用します。

Elastic IP アドレスをリリース

Elastic IP アドレスが不要になった場合は、以下のいずれかの方法を使用してリリースすることをお勧めします。リリースするアドレスは、EC2 インスタンス、NAT ゲートウェイ、Network Load Balancer などの AWS リソースに現在関連付けられていないものに限ります。

> **Note**
>
> AWS サポートに問い合わせて Elastic IP (EIP) アドレスの逆引き DNS を設定する場合、逆引き DNS を削除することはできますが、Elastic IP アドレスは AWS サポートによってロックされているためリリースできません。Elastic IP アドレスのロックを解除するには、AWS Support にお問い合わせください。Elastic IP アドレスのロックが解除されたら、Elastic IP アドレス (EIP) をリリースできます。

Console

Elastic IP アドレスをリリースするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. リリースする Elastic IP アドレスを選択してから、[アクション]、 [Elastic IP アドレスのリリース] の順に選択します。
4. [リリース] を選択します。

AWS CLI

Elastic IP アドレスをリリースするには

release-address AWS CLI コマンドを使用します。

PowerShell

Elastic IP アドレスをリリースするには

Remove-EC2Address AWS Tools for Windows PowerShell コマンドを使用します。

Elastic IP アドレスの復元

Elastic IP アドレスをリリースした場合でも、復元できる可能性があります。以下のルールが適用されます。

- Elastic IP アドレスが別の AWS アカウントに割り当てられている場合や Elastic IP アドレスの制限を超過する場合は、Elastic IP アドレスを復元できません。
- Elastic IP アドレスに関連付けられたタグを復旧することはできません。
- Elastic IP アドレスは、Amazon EC2 API コンソールまたはコマンドラインツールでのみ復元できます。

AWS CLI

Elastic IP アドレスを復元するには

以下のように、AWS CLI パラメータを指定した allocate-address `--address` コマンドを使用して、IP アドレスを指定します。

```
aws ec2 allocate-address --domain vpc --address 203.0.113.3
```

PowerShell

Elastic IP アドレスを復元するには

以下のように、AWS Tools for Windows PowerShell パラメータを指定した New-EC2Address `-Address` コマンドを使用して、IP アドレスを指定します。

```
PS C:\> New-EC2Address -Address 203.0.113.3 -Domain vpc -Region us-east-1
```

E メールアプリケーション用の逆引き DNS の使用

インスタンスから第三者に E メールを送信する場合は、1 つ以上の Elastic IP アドレスをプロビジョニングし、E メールの送信に使用する Elastic IP アドレスに静的な逆引き DNS レコードを割り当てることをお勧めします。これにより、電子メールが一部のスパム対策組織によりスパムとしてフラグ付けされるのを防ぐことができます。AWSは、ISP およびインターネットアンチスパム組織と協力して、これらのアドレスから送信された E メールにスパムのフラグが付く可能性を減らしています。

考慮事項

- 逆引き DNS レコードを作成する前に、Elastic IP アドレスを参照する、対応するフォワード DNS レコード (レコードタイプ A) を設定する必要があります。
- 逆引き DNS レコードが Elastic IP アドレスに関連付けられている場合、その Elastic IP アドレスはアカウントにロックされ、レコードが削除されるまでアカウントからリリースすることはできません。
- AWS GovCloud (US) Region

  コンソールまたは AWS CLI を使用して逆引き DNS レコードを作成することはできません。AWS から静的な逆引き DNS レコードが割り当てられる必要があります。リバースDNSとEメール送信制限を削除し、Elastic IP アドレスや逆引きDNSレコードを提供するリクエストを開きます。

逆引き DNS レコードを作成する

逆引き DNS レコードを作成するには、使用する方法に一致するタブを選択します。

Console

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. Elastic IP アドレスを選択し、[Actions]、[Update reverse DNS] の順に選択します。
4. [Reverse DNS domain name] (リバース DNS ドメイン名) の場合、ドメイン名を入力します。
5. **update** を入力して確定します。
6. [更新] を選択します。

AWS CLI

次の例に示されているように、AWS CLI で modify-address-attribute コマンドを使用します。

```
aws ec2 modify-address-attribute --allocation-id eipalloc-abcdef01234567890  --
domain-name example.com
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
```

```
            "PtrRecord": "example.net."
            "PtrRecordUpdate": {
                "Value": "example.com.",
                "Status": "PENDING"
        }
    ]
}
```

逆引き DNS レコードを削除する

逆引き DNS レコードを削除するには、使用する方法に一致するタブを選択します。

Console

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Elastic IP] を選択します。
3. Elastic IP アドレスを選択し、[Actions]、[Update reverse DNS] の順に選択します。
4. [Reverse DNS domain name] (リバース DNS ドメイン名) の場合、ドメイン名をクリアします。
5. **update** を入力して確定します。
6. [更新] を選択します。

AWS CLI

次の例に示されているように、AWS CLI で reset-address-attribute コマンドを使用します。

```
aws ec2 reset-address-attribute --allocation-id eipalloc-abcdef01234567890 --
attribute domain-name
{
    "Addresses": [
        {
            "PublicIp": "192.0.2.0",
            "AllocationId": "eipalloc-abcdef01234567890",
            "PtrRecord": "example.com."
            "PtrRecordUpdate": {
                "Value": "example.net.",
                "Status": "PENDING"
        }
    ]
```

```
}
```

> **Note**
>
> コマンドの実行時に次のエラーが表示された場合は、E メール送信制限の解除リクエストを AWS Supportに送信してサポートを受けてください。
> 割り当て ID が含まれるアドレスは、アカウントにロックされているためリリースすることはできません。

Elastic IP アドレスのクォータ

デフォルトでは、すべての AWS アカウントでリージョンあたり 5 つの Elastic IP アドレスが割り当てられています。これは、パブリック (IPv4) インターネットアドレスが数に限りのあるパブリックリソースであるためです。インスタンスに障害が発生した場合にアドレスを他のインスタンスに再マップする機能のために主に Elastic IP アドレスを使用し、他のすべてのノード間通信には DNS ホスト名を使用することを強くお勧めします。

使用中の Elastic IP アドレスの数を確認するには

https://console.aws.amazon.com/ec2/ でAmazon EC2 コンソールを開き、ナビゲーションペインで [Elastic IP] を選択します。

現在のアカウントに割り当てられている Elastic IP アドレスを確認するには

1. Service Quotas のコンソールを開きます。https://console.aws.amazon.com/servicequotas/
2. 画面上部のナビゲーションバーで、リージョンを選択します。
3. ダッシュボードで [Amazon Elastic Compute Cloud (Amazon EC2)] を選択します。

   ダッシュボードに Amazon Elastic Compute Cloud (Amazon EC2) が一覧表示されていない場合は、[AWS サービス] を選択し、検索フィールドに**EC2**と入力して、[Amazon Elastic Compute Cloud (Amazon EC2)] を選択します。
4. Amazon EC2 のサービスクォータのページで、検索フィールドに**IP**と入力します。制限は [EC2-VPC Elastic IPs (EC2-VPC Elastic IP の数)] です。詳細については、制限のリンクを選択してください。

お客様のアーキテクチャで追加の Elastic IP アドレスが必要な場合、クォータの引き上げを Service Quotas コンソールから直接リクエストできます。クォータの引き上げをリクエストするには、[ア

カウントレベルでの引き上げをリクエスト] を選択します。詳細については、「Amazon EC2 の Service Quotas」を参照してください。

Elastic Network Interface

Elastic Network Interface は、仮想ネットワークカードを表す VPC 内の論理ネットワーキングコンポーネントです。次の属性を含めることができます。

- VPC の IPv4 アドレス範囲からのプライマリプライベート IPv4 アドレス
- VPC の IPv6 アドレス範囲からのプライマリ IPv6 アドレス
- VPC の IPv4 アドレス範囲からの 1 つ以上のセカンダリプライベート IPv4 アドレス
- プライベート IPv4 アドレスごとに 1 つの Elastic IP アドレス (IPv4)
- 1 つのパブリック IPv4 アドレス
- 1 つ以上の IPv6 アドレス
- 1 つ以上のセキュリティグループ
- MAC アドレス
- 送信元/送信先チェックフラグ
- 説明

ネットワークインターフェイスを作成および設定して、同じアベイラビリティーゾーンのインスタンスにアタッチできます。アカウントでは、AWS のサービスで作成および管理されるリクエスタマネージド型のネットワークインターフェイスも使用できます。これらを通じて他のリソースやサービスを利用できます。これらは、ユーザーが直接管理できないネットワークインターフェイスです。詳細については、リクエスタマネージド型のネットワークインターフェイスを参照してください。

この AWS リソースは、AWS Management Console および Amazon EC2 API ではネットワークインターフェイスと呼ばれます。したがって、このドキュメントでは Elastic Network Interface ではなく「ネットワークインターフェイス」を使用します。このドキュメントの「ネットワークインターフェイス」という用語は、常に Elastic Network Interface を意味します。

コンテンツ

- ネットワークインターフェイスの設定に関するベストプラクティス
- ネットワークインターフェイスのシナリオ
- リクエスタマネージド型のネットワークインターフェイス
- Amazon EC2 ネットワークインターフェイスへのプレフィクスの割り当て

ネットワークインターフェイスの基本

ネットワークインターフェイスを作成したり、インスタンスにアタッチしたり、インスタンスからデタッチしたり、別のインスタンスにアタッチしたりできます。ネットワークインターフェイスをインスタンスにアタッチしたり、インスタンスからデタッチして別のインスタンスに再アタッチしたりするときには、ネットワークインターフェイスの属性が保持されます。インスタンス間でネットワークインターフェイスを移動すると、ネットワークトラフィックは新しいインスタンスにリダイレクトされます。

プライマリネットワークインターフェイス

各インスタンスには、プライマリネットワークインターフェイスと呼ばれるデフォルトのネットワークインターフェイスがあります。プライマリネットワークインターフェイスをインスタンスからデタッチすることはできません。追加のネットワークインターフェイスを作成し、アタッチできます。使用できるネットワークインターフェイスの最大数はインスタンスタイプによって異なります。詳細については、各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数を参照してください。

ネットワークインターフェイスのパブリック IPv4 アドレス

VPC では、すべてのサブネットに、そのサブネットで作成されるネットワークインターフェイス (結果的にそのサブネットで起動されるインスタンス) にパブリック IPv4 アドレスを割り当てるかどうかを決定する、変更可能な属性があります。詳細については、「Amazon VPC ユーザーガイド」の「サブネット設定」を参照してください。パブリック IPv4 アドレスは Amazon のパブリック IPv4 アドレスのプールから割り当てられます。インスタンスを起動すると、作成されたプライマリネットワークインターフェイスに IP アドレスが割り当てられます。

ネットワークインターフェイスを作成すると、サブネットからパブリック IPv4 アドレス指定属性を継承します。後でサブネットのパブリック IPv4 アドレス指定属性を変更しても、ネットワークインターフェイスでは作成時に有効だった設定が保持されます。インスタンスを起動し、既存のネットワークインターフェイスをプライマリネットワークインターフェイスとして指定する場合、パブリック IPv4 アドレス属性はこのネットワークインターフェイスによって決定されます。

詳細については、パブリック IPv4 アドレスを参照してください。

ネットワークインターフェイスの Elastic IP アドレス

Elastic IP アドレスが与えられている場合、ネットワークインターフェイスのプライベート IPv4 アドレスの 1 つをそれと関連付けることができます。1 つの Elastic IP アドレスと各プライベート IPv4 アドレスを関連付けることができます。

ネットワークインターフェイスから Elastic IP アドレスの関連付けを解除すると、そのアドレスをアドレスプールに戻すことができます。ネットワークインターフェイスはサブネットに固有であるため、これが Elastic IP アドレスを別のサブネットまたは VPC のインスタンスに関連付ける唯一の方法です。

ネットワークインターフェイスの IPv6 アドレス

IPv6 CIDR ブロックを VPC とサブネットに関連付けると、サブネットの範囲から 1 つ以上の IPv6 アドレスをネットワークインターフェイスに割り当てることができます。各 IPv6 アドレスは、1 つのネットワークインターフェイスに割り当てることができます。

すべてのサブネットには、そのサブネットで作成されるネットワークインターフェイス (結果的にそのサブネットで起動されるインスタンス) にサブネットの範囲から IPv6 アドレスを自動的に割り当てるかどうかを決定する、変更可能な属性があります。詳細については、「Amazon VPC ユーザーガイド」の「サブネット設定」を参照してください。インスタンスを起動すると、作成されたプライマリネットワークインターフェイスに IPv6 アドレスが割り当てられます。

詳細については、IPv6 アドレスを参照してください。

プレフィクスの委任

プレフィクス委任プレフィクスとは、予約済みのプライベート IPv4 または IPv6 CIDR 範囲のことで、インスタンスに関連付けられたネットワークインターフェイスへ自動または手動で割り当てるためのものです。委任プレフィクスを使用すると、IP アドレスの範囲を単一のプレフィクスとして割り当てることで、サービスを迅速に起動できます。

終了動作

インスタンスにアタッチされているネットワークインターフェイスの終了動作を設定できます。アタッチしたインスタンスの終了時に、ネットワークインターフェイスを自動的に削除するかどうかを指定できます。

送信元/送信先チェック

送信元/送信先チェックを有効または無効にできます。これにより、受信するすべてのトラフィックに関して、そのインスタンスが送信元なのか、あるいは送信先であるのかを確認できます。送信元/送信先チェックはデフォルトで有効化されています。ネットワークアドレス変換、ルーティング、ファイアウォールなどのサービスを実行するインスタンスでは、送信元/送信先チェックを無効にする必要があります。

IP トラフィックのモニタリング

ネットワークインターフェイスで VPC フローログを有効にして、ネットワークインターフェイスとの間で行き来する IP トラフィックに関する情報をキャプチャできます。フローログを作成すると、そのデータを Amazon CloudWatch Logs で表示し、取得できます。詳細については、Amazon VPC ユーザーガイド のVPC フローログを参照してください。

ネットワークカード

複数のネットワークカードを持つインスタンスは、100 Gbps を超える帯域幅機能やパケットレートのパフォーマンスの向上など、より高いネットワークパフォーマンスを提供します。各ネットワークインターフェイスは、ネットワークカードにアタッチされています。プライマリネットワークインターフェイスは、ネットワークカードインデックス 0 に割り当てる必要があります。

複数のネットワークカードをサポートするインスタンスを起動するときに Elastic Fabric Adapter (EFA) を有効にした場合、すべてのネットワークカードが使用可能になります。ネットワークカードごとに EFA を 1 つまで割り当てることができます。EFA はネットワークインターフェイスとしてカウントされます。

次のインスタンスは、複数のネットワークカードをサポートします。他のすべてのインスタンスタイプは、1 つのネットワークカードをサポートします。

インスタンスタイプ	ネットワークカードの数
c6in.32xlarge	2
c6in.metal	2
dl1.24xlarge	4
hpc6id.32xlarge	2
hpc7a.12xlarge	2

インスタンスタイプ	ネットワークカードの数
hpc7a.24xlarge	2
hpc7a.48xlarge	2
hpc7a.96xlarge	2
m6idn.32xlarge	2
m6idn.metal	2
m6in.32xlarge	2
m6in.metal	2
p4d.24xlarge	4
p4de.24xlarge	4
p5.48xlarge	32
r6idn.32xlarge	2
r6idn.metal	2
r6in.32xlarge	2
r6in.metal	2
trn1.32xlarge	8
trn1n.32xlarge	16

各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数

以下の表に示しているのは、各インスタンスタイプのネットワークインターフェイスの最大数と、ネットワークインターフェイスあたりのプライベート IPv4 アドレスと IPv6 アドレスの最大数です。ネットワークインターフェイスあたりの IPv6 アドレスとプライベート IPv4 アドレスの制限は

異なります。すべてのインスタンスタイプで IPv6 アドレス指定がサポートされているわけではありません。

コンテンツ

汎用

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
a1.medium	2	4	4
a1.large	3	10	10
a1.xlarge	4	15	15
a1.2xlarge	4	15	15
a1.4xlarge	8	30	30
a1.metal	8	30	30
m1.small	2	4	IPv6 はサポートされていません
m1.medium	2	6	IPv6 はサポートされていません
m1.large	3	10	IPv6 はサポートされていません

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m1.xlarge	4	15	IPv6 はサポートされていません
m2.xlarge	4	15	IPv6 はサポートされていません
m2.2xlarge	4	30	IPv6 はサポートされていません
m2.4xlarge	8	30	IPv6 はサポートされていません
m3.medium	2	6	IPv6 はサポートされていません
m3.large	3	10	IPv6 はサポートされていません
m3.xlarge	4	15	IPv6 はサポートされていません
m3.2xlarge	4	30	IPv6 はサポートされていません
m4.large	2	10	10
m4.xlarge	4	15	15
m4.2xlarge	4	15	15
m4.4xlarge	8	30	30
m4.10xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m4.16xlarge	8	30	30
m5.large	3	10	10
m5.xlarge	4	15	15
m5.2xlarge	4	15	15
m5.4xlarge	8	30	30
m5.8xlarge	8	30	30
m5.12xlarge	8	30	30
m5.16xlarge	15	50	50
m5.24xlarge	15	50	50
m5.metal	15	50	50
m5a.large	3	10	10
m5a.xlarge	4	15	15
m5a.2xlarge	4	15	15
m5a.4xlarge	8	30	30
m5a.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m5a.12xlarge	8	30	30
m5a.16xlarge	15	50	50
m5a.24xlarge	15	50	50
m5ad.large	3	10	10
m5ad.xlarge	4	15	15
m5ad.2xlarge	4	15	15
m5ad.4xlarge	8	30	30
m5ad.8xlarge	8	30	30
m5ad.12xlarge	8	30	30
m5ad.16xlarge	15	50	50
m5ad.24xlarge	15	50	50
m5d.large	3	10	10
m5d.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m5d.2xlarge	4	15	15
m5d.4xlarge	8	30	30
m5d.8xlarge	8	30	30
m5d.12xlarge	8	30	30
m5d.16xlarge	15	50	50
m5d.24xlarge	15	50	50
m5d.metal	15	50	50
m5dn.large	3	10	10
m5dn.xlarge	4	15	15
m5dn.2xlarge	4	15	15
m5dn.4xlarge	8	30	30
m5dn.8xlarge	8	30	30
m5dn.12xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m5dn.16xlarge	15	50	50
m5dn.24xlarge	15	50	50
m5dn.metal	15	50	50
m5n.large	3	10	10
m5n.xlarge	4	15	15
m5n.2xlarge	4	15	15
m5n.4xlarge	8	30	30
m5n.8xlarge	8	30	30
m5n.12xlarge	8	30	30
m5n.16xlarge	15	50	50
m5n.24xlarge	15	50	50
m5n.metal	15	50	50
m5zn.large	3	10	10
m5zn.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m5zn.2xlarge	4	15	15
m5zn.3xlarge	8	30	30
m5zn.6xlarge	8	30	30
m5zn.12xlarge	15	50	50
m5zn.metal	15	50	50
m6a.large	3	10	10
m6a.xlarge	4	15	15
m6a.2xlarge	4	15	15
m6a.4xlarge	8	30	30
m6a.8xlarge	8	30	30
m6a.12xlarge	8	30	30
m6a.16xlarge	15	50	50
m6a.24xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m6a.32xlarge	15	50	50
m6a.48xlarge	15	50	50
m6a.metal	15	50	50
m6g.medium	2	4	4
m6g.large	3	10	10
m6g.xlarge	4	15	15
m6g.2xlarge	4	15	15
m6g.4xlarge	8	30	30
m6g.8xlarge	8	30	30
m6g.12xlarge	8	30	30
m6g.16xlarge	15	50	50
m6g.metal	15	50	50
m6gd.medium	2	4	4
m6gd.large	3	10	10

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m6gd.xlarge	4	15	15
m6gd.2xlarge	4	15	15
m6gd.4xlarge	8	30	30
m6gd.8xlarge	8	30	30
m6gd.12xlarge	8	30	30
m6gd.16xlarge	15	50	50
m6gd.metal	15	50	50
m6i.large	3	10	10
m6i.xlarge	4	15	15
m6i.2xlarge	4	15	15
m6i.4xlarge	8	30	30
m6i.8xlarge	8	30	30
m6i.12xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m6i.16xlarge	15	50	50
m6i.24xlarge	15	50	50
m6i.32xlarge	15	50	50
m6i.metal	15	50	50
m6id.large	3	10	10
m6id.xlarge	4	15	15
m6id.2xlarge	4	15	15
m6id.4xlarge	8	30	30
m6id.8xlarge	8	30	30
m6id.12xlarge	8	30	30
m6id.16xlarge	15	50	50
m6id.24xlarge	15	50	50
m6id.32xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m6id.metal	15	50	50
m6idn.large	3	10	10
m6idn.xlarge	4	15	15
m6idn.2xlarge	4	15	15
m6idn.4xlarge	8	30	30
m6idn.8xlarge	8	30	30
m6idn.12xlarge	8	30	30
m6idn.16xlarge	15	50	50
m6idn.24xlarge	15	50	50
m6idn.32xlarge	14	50	50
m6idn.metal	14	50	50
m6in.large	3	10	10
m6in.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m6in.2xlarge	4	15	15
m6in.4xlarge	8	30	30
m6in.8xlarge	8	30	30
m6in.12xlarge	8	30	30
m6in.16xlarge	15	50	50
m6in.24xlarge	15	50	50
m6in.32xlarge	14	50	50
m6in.metal	14	50	50
m7a.medium	2	4	4
m7a.large	3	10	10
m7a.xlarge	4	15	15
m7a.2xlarge	4	15	15
m7a.4xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m7a.8xlarge	8	30	30
m7a.12xlarge	8	30	30
m7a.16xlarge	15	50	50
m7a.24xlarge	15	50	50
m7a.32xlarge	15	50	50
m7a.48xlarge	15	50	50
m7a.metal-48xl	15	50	50
m7g.medium	2	4	4
m7g.large	3	10	10
m7g.xlarge	4	15	15
m7g.2xlarge	4	15	15
m7g.4xlarge	8	30	30
m7g.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m7g.12xlarge	8	30	30
m7g.16xlarge	15	50	50
m7g.metal	15	50	50
m7gd.medium	2	4	4
m7gd.large	3	10	10
m7gd.xlarge	4	15	15
m7gd.2xlarge	4	15	15
m7gd.4xlarge	8	30	30
m7gd.8xlarge	8	30	30
m7gd.12xlarge	8	30	30
m7gd.16xlarge	15	50	50
m7gd.metal	15	50	50
m7i.large	3	10	10
m7i.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m7i.2xlarge	4	15	15
m7i.4xlarge	8	30	30
m7i.8xlarge	8	30	30
m7i.12xlarge	8	30	30
m7i.16xlarge	15	50	50
m7i.24xlarge	15	50	50
m7i.48xlarge	15	50	50
m7i.metal-24xl	15	50	50
m7i.metal-48xl	15	50	50
m7i-flex.large	3	10	10
m7i-flex.xlarge	4	15	15
m7i-flex.2xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
m7i-flex.4xlarge	8	30	30
m7i-flex.8xlarge	8	30	30
mac1.metal	8	30	30
mac2.metal	8	30	30
mac2-m2.metal	8	30	30
mac2-m2pro.metal	8	30	30
t1.micro	2	2	IPv6 はサポートされていません
t2.nano	2	2	2
t2.micro	2	2	2
t2.small	3	4	4
t2.medium	3	6	6
t2.large	3	12	12
t2.xlarge	3	15	15
t2.2xlarge	3	15	15
t3.nano	2	2	2
t3.micro	2	2	2

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
t3.small	3	4	4
t3.medium	3	6	6
t3.large	3	12	12
t3.xlarge	4	15	15
t3.2xlarge	4	15	15
t3a.nano	2	2	2
t3a.micro	2	2	2
t3a.small	2	4	4
t3a.medium	3	6	6
t3a.large	3	12	12
t3a.xlarge	4	15	15
t3a.2xlarge	4	15	15
t4g.nano	2	2	2
t4g.micro	2	2	2
t4g.small	3	4	4
t4g.medium	3	6	6
t4g.large	3	12	12
t4g.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
t4g.2xlarge	4	15	15

コンピューティングの最適化

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c1.medium	2	6	IPv6 はサポートされていません
c1.xlarge	4	15	IPv6 はサポートされていません
c3.large	3	10	10
c3.xlarge	4	15	15
c3.2xlarge	4	15	15
c3.4xlarge	8	30	30
c3.8xlarge	8	30	30
c4.large	3	10	10
c4.xlarge	4	15	15
c4.2xlarge	4	15	15
c4.4xlarge	8	30	30
c4.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c5.large	3	10	10
c5.xlarge	4	15	15
c5.2xlarge	4	15	15
c5.4xlarge	8	30	30
c5.9xlarge	8	30	30
c5.12xlarge	8	30	30
c5.18xlarge	15	50	50
c5.24xlarge	15	50	50
c5.metal	15	50	50
c5a.large	3	10	10
c5a.xlarge	4	15	15
c5a.2xlarge	4	15	15
c5a.4xlarge	8	30	30
c5a.8xlarge	8	30	30
c5a.12xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c5a.16xlarge	15	50	50
c5a.24xlarge	15	50	50
c5ad.large	3	10	10
c5ad.xlarge	4	15	15
c5ad.2xlarge	4	15	15
c5ad.4xlarge	8	30	30
c5ad.8xlarge	8	30	30
c5ad.12xlarge	8	30	30
c5ad.16xlarge	15	50	50
c5ad.24xlarge	15	50	50
c5d.large	3	10	10
c5d.xlarge	4	15	15
c5d.2xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c5d.4xlarge	8	30	30
c5d.9xlarge	8	30	30
c5d.12xlarge	8	30	30
c5d.18xlarge	15	50	50
c5d.24xlarge	15	50	50
c5d.metal	15	50	50
c5n.large	3	10	10
c5n.xlarge	4	15	15
c5n.2xlarge	4	15	15
c5n.4xlarge	8	30	30
c5n.9xlarge	8	30	30
c5n.18xlarge	15	50	50
c5n.metal	15	50	50
c6a.large	3	10	10

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c6a.xlarge	4	15	15
c6a.2xlarge	4	15	15
c6a.4xlarge	8	30	30
c6a.8xlarge	8	30	30
c6a.12xlarge	8	30	30
c6a.16xlarge	15	50	50
c6a.24xlarge	15	50	50
c6a.32xlarge	15	50	50
c6a.48xlarge	15	50	50
c6a.metal	15	50	50
c6g.medium	2	4	4
c6g.large	3	10	10
c6g.xlarge	4	15	15
c6g.2xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c6g.4xlarge	8	30	30
c6g.8xlarge	8	30	30
c6g.12xlarge	8	30	30
c6g.16xlarge	15	50	50
c6g.metal	15	50	50
c6gd.medium	2	4	4
c6gd.large	3	10	10
c6gd.xlarge	4	15	15
c6gd.2xlarge	4	15	15
c6gd.4xlarge	8	30	30
c6gd.8xlarge	8	30	30
c6gd.12xlarge	8	30	30
c6gd.16xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c6gd.metal	15	50	50
c6gn.medium	2	4	4
c6gn.large	3	10	10
c6gn.xlarge	4	15	15
c6gn.2xlarge	4	15	15
c6gn.4xlarge	8	30	30
c6gn.8xlarge	8	30	30
c6gn.12xlarge	8	30	30
c6gn.16xlarge	15	50	50
c6i.large	3	10	10
c6i.xlarge	4	15	15
c6i.2xlarge	4	15	15
c6i.4xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c6i.8xlarge	8	30	30
c6i.12xlarge	8	30	30
c6i.16xlarge	15	50	50
c6i.24xlarge	15	50	50
c6i.32xlarge	15	50	50
c6i.metal	15	50	50
c6id.large	3	10	10
c6id.xlarge	4	15	15
c6id.2xlarge	4	15	15
c6id.4xlarge	8	30	30
c6id.8xlarge	8	30	30
c6id.12xlarge	8	30	30
c6id.16xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c6id.24xlarge	15	50	50
c6id.32xlarge	15	50	50
c6id.metal	15	50	50
c6in.large	3	10	10
c6in.xlarge	4	15	15
c6in.2xlarge	4	15	15
c6in.4xlarge	8	30	30
c6in.8xlarge	8	30	30
c6in.12xlarge	8	30	30
c6in.16xlarge	15	50	50
c6in.24xlarge	15	50	50
c6in.32xlarge	14	50	50
c6in.metal	14	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c7a.medium	2	4	4
c7a.large	3	10	10
c7a.xlarge	4	15	15
c7a.2xlarge	4	15	15
c7a.4xlarge	8	30	30
c7a.8xlarge	8	30	30
c7a.12xlarge	8	30	30
c7a.16xlarge	15	50	50
c7a.24xlarge	15	50	50
c7a.32xlarge	15	50	50
c7a.48xlarge	15	50	50
c7a.metal-48xl	15	50	50
c7g.medium	2	4	4
c7g.large	3	10	10

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c7g.xlarge	4	15	15
c7g.2xlarge	4	15	15
c7g.4xlarge	8	30	30
c7g.8xlarge	8	30	30
c7g.12xlarge	8	30	30
c7g.16xlarge	15	50	50
c7g.metal	15	50	50
c7gd.medium	2	4	4
c7gd.large	3	10	10
c7gd.xlarge	4	15	15
c7gd.2xlarge	4	15	15
c7gd.4xlarge	8	30	30
c7gd.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c7gd.12xlarge	8	30	30
c7gd.16xlarge	15	50	50
c7gd.metal	15	50	50
c7gn.medium	2	4	4
c7gn.large	3	10	10
c7gn.xlarge	4	15	15
c7gn.2xlarge	4	15	15
c7gn.4xlarge	8	30	30
c7gn.8xlarge	8	30	30
c7gn.12xlarge	8	30	30
c7gn.16xlarge	15	50	50
c7i.large	3	10	10
c7i.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
c7i.2xlarge	4	15	15
c7i.4xlarge	8	30	30
c7i.8xlarge	8	30	30
c7i.12xlarge	8	30	30
c7i.16xlarge	15	50	50
c7i.24xlarge	15	50	50
c7i.48xlarge	15	50	50
c7i.metal-24xl	15	50	50
c7i.metal-48xl	15	50	50

メモリ最適化

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r3.large	3	10	10

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r3.xlarge	4	15	15
r3.2xlarge	4	15	15
r3.4xlarge	8	30	30
r3.8xlarge	8	30	30
r4.large	3	10	10
r4.xlarge	4	15	15
r4.2xlarge	4	15	15
r4.4xlarge	8	30	30
r4.8xlarge	8	30	30
r4.16xlarge	15	50	50
r5.large	3	10	10
r5.xlarge	4	15	15
r5.2xlarge	4	15	15
r5.4xlarge	8	30	30
r5.8xlarge	8	30	30
r5.12xlarge	8	30	30
r5.16xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r5.24xlarge	15	50	50
r5.metal	15	50	50
r5a.large	3	10	10
r5a.xlarge	4	15	15
r5a.2xlarge	4	15	15
r5a.4xlarge	8	30	30
r5a.8xlarge	8	30	30
r5a.12xlarge	8	30	30
r5a.16xlarge	15	50	50
r5a.24xlarge	15	50	50
r5ad.large	3	10	10
r5ad.xlarge	4	15	15
r5ad.2xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r5ad.4xlarge	8	30	30
r5ad.8xlarge	8	30	30
r5ad.12xlarge	8	30	30
r5ad.16xlarge	15	50	50
r5ad.24xlarge	15	50	50
r5b.large	3	10	10
r5b.xlarge	4	15	15
r5b.2xlarge	4	15	15
r5b.4xlarge	8	30	30
r5b.8xlarge	8	30	30
r5b.12xlarge	8	30	30
r5b.16xlarge	15	50	50
r5b.24xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r5b.metal	15	50	50
r5d.large	3	10	10
r5d.xlarge	4	15	15
r5d.2xlarge	4	15	15
r5d.4xlarge	8	30	30
r5d.8xlarge	8	30	30
r5d.12xlarge	8	30	30
r5d.16xlarge	15	50	50
r5d.24xlarge	15	50	50
r5d.metal	15	50	50
r5dn.large	3	10	10
r5dn.xlarge	4	15	15
r5dn.2xlarge	4	15	15
r5dn.4xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r5dn.8xlarge	8	30	30
r5dn.12xlarge	8	30	30
r5dn.16xlarge	15	50	50
r5dn.24xlarge	15	50	50
r5dn.metal	15	50	50
r5n.large	3	10	10
r5n.xlarge	4	15	15
r5n.2xlarge	4	15	15
r5n.4xlarge	8	30	30
r5n.8xlarge	8	30	30
r5n.12xlarge	8	30	30
r5n.16xlarge	15	50	50
r5n.24xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r5n.metal	15	50	50
r6a.large	3	10	10
r6a.xlarge	4	15	15
r6a.2xlarge	4	15	15
r6a.4xlarge	8	30	30
r6a.8xlarge	8	30	30
r6a.12xlarge	8	30	30
r6a.16xlarge	15	50	50
r6a.24xlarge	15	50	50
r6a.32xlarge	15	50	50
r6a.48xlarge	15	50	50
r6a.metal	15	50	50
r6g.medium	2	4	4
r6g.large	3	10	10

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r6g.xlarge	4	15	15
r6g.2xlarge	4	15	15
r6g.4xlarge	8	30	30
r6g.8xlarge	8	30	30
r6g.12xlarge	8	30	30
r6g.16xlarge	15	50	50
r6g.metal	15	50	50
r6gd.medium	2	4	4
r6gd.large	3	10	10
r6gd.xlarge	4	15	15
r6gd.2xlarge	4	15	15
r6gd.4xlarge	8	30	30
r6gd.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r6gd.12xlarge	8	30	30
r6gd.16xlarge	15	50	50
r6gd.metal	15	50	50
r6i.large	3	10	10
r6i.xlarge	4	15	15
r6i.2xlarge	4	15	15
r6i.4xlarge	8	30	30
r6i.8xlarge	8	30	30
r6i.12xlarge	8	30	30
r6i.16xlarge	15	50	50
r6i.24xlarge	15	50	50
r6i.32xlarge	15	50	50
r6i.metal	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r6idn.large	3	10	10
r6idn.xlarge	4	15	15
r6idn.2xlarge	4	15	15
r6idn.4xlarge	8	30	30
r6idn.8xlarge	8	30	30
r6idn.12xlarge	8	30	30
r6idn.16xlarge	15	50	50
r6idn.24xlarge	15	50	50
r6idn.32xlarge	14	50	50
r6idn.metal	14	50	50
r6in.large	3	10	10
r6in.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r6in.2xlarge	4	15	15
r6in.4xlarge	8	30	30
r6in.8xlarge	8	30	30
r6in.12xlarge	8	30	30
r6in.16xlarge	15	50	50
r6in.24xlarge	15	50	50
r6in.32xlarge	14	50	50
r6in.metal	14	50	50
r6id.large	3	10	10
r6id.xlarge	4	15	15
r6id.2xlarge	4	15	15
r6id.4xlarge	8	30	30
r6id.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r6id.12xlarge	8	30	30
r6id.16xlarge	15	50	50
r6id.24xlarge	15	50	50
r6id.32xlarge	15	50	50
r6id.metal	15	50	50
r7a.medium	2	4	4
r7a.large	3	10	10
r7a.xlarge	4	15	15
r7a.2xlarge	4	15	15
r7a.4xlarge	8	30	30
r7a.8xlarge	8	30	30
r7a.12xlarge	8	30	30
r7a.16xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r7a.24xlarge	15	50	50
r7a.32xlarge	15	50	50
r7a.48xlarge	15	50	50
r7a.metal-48xl	15	50	50
r7g.medium	2	4	4
r7g.large	3	10	10
r7g.xlarge	4	15	15
r7g.2xlarge	4	15	15
r7g.4xlarge	8	30	30
r7g.8xlarge	8	30	30
r7g.12xlarge	8	30	30
r7g.16xlarge	15	50	50
r7g.metal	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r7gd.medium	2	4	4
r7gd.large	3	10	10
r7gd.xlarge	4	15	15
r7gd.2xlarge	4	15	15
r7gd.4xlarge	8	30	30
r7gd.8xlarge	8	30	30
r7gd.12xlarge	8	30	30
r7gd.16xlarge	15	50	50
r7gd.metal	15	50	50
r7i.large	3	10	10
r7i.xlarge	4	15	15
r7i.2xlarge	4	15	15
r7i.4xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r7i.8xlarge	8	30	30
r7i.12xlarge	8	30	30
r7i.16xlarge	15	50	50
r7i.24xlarge	15	50	50
r7i.48xlarge	15	50	50
r7i.metal-24xl	15	50	50
r7i.metal-48xl	15	50	50
r7iz.large	3	10	10
r7iz.xlarge	4	15	15
r7iz.2xlarge	4	15	15
r7iz.4xlarge	8	30	30
r7iz.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
r7iz.12xlarge	8	30	30
r7iz.16xlarge	15	50	50
r7iz.32xlarge	15	50	50
r7iz.metal-16xl	15	50	50
r7iz.metal-32xl	15	50	50
u-3tb1.56xlarge	8	30	30
u-6tb1.56xlarge	15	50	50
u-6tb1.112xlarge	15	50	50
u-6tb1.metal	5	30	30
u-9tb1.112xlarge	15	50	50
u-9tb1.metal	5	30	30
u-12tb1.112xlarge	15	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
u-12tb1.metal	5	30	30
u-18tb1.112xlarge	15	50	50
u-18tb1.metal	15	50	50
u-24tb1.112xlarge	15	50	50
u-24tb1.metal	15	50	50
x1.16xlarge	8	30	30
x1.32xlarge	8	30	30
x2gd.medium	2	4	4
x2gd.large	3	10	10
x2gd.xlarge	4	15	15
x2gd.2xlarge	4	15	15
x2gd.4xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
x2gd.8xlarge	8	30	30
x2gd.12xlarge	8	30	30
x2gd.16xlarge	15	50	50
x2gd.metal	15	50	50
x2idn.16xlarge	15	50	50
x2idn.24xlarge	15	50	50
x2idn.32xlarge	15	50	50
x2idn.metal	15	50	50
x2iedn.xlarge	4	15	15
x2iedn.2xlarge	4	15	15
x2iedn.4xlarge	8	30	30
x2iedn.8xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
x2iedn.16xlarge	15	50	50
x2iedn.24xlarge	15	50	50
x2iedn.32xlarge	15	50	50
x2iedn.metal	15	50	50
x2iezn.2xlarge	4	15	15
x2iezn.4xlarge	8	30	30
x2iezn.6xlarge	8	30	30
x2iezn.8xlarge	8	30	30
x2iezn.12xlarge	15	50	50
x2iezn.metal	15	50	50
x1e.xlarge	3	10	10
x1e.2xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
x1e.4xlarge	4	15	15
x1e.8xlarge	4	15	15
x1e.16xlarge	8	30	30
x1e.32xlarge	8	30	30
z1d.large	3	10	10
z1d.xlarge	4	15	15
z1d.2xlarge	4	15	15
z1d.3xlarge	8	30	30
z1d.6xlarge	8	30	30
z1d.12xlarge	15	50	50
z1d.metal	15	50	50

ストレージの最適化

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
d2.xlarge	4	15	15
d2.2xlarge	4	15	15
d2.4xlarge	8	30	30
d2.8xlarge	8	30	30
d3.xlarge	4	3	3
d3.2xlarge	4	5	5
d3.4xlarge	4	10	10
d3.8xlarge	3	20	20
d3en.xlarge	4	3	3
d3en.2xlarge	4	5	5
d3en.4xlarge	4	10	10
d3en.6xlarge	4	15	15
d3en.8xlarge	4	20	20
d3en.12xlarge	3	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
h1.2xlarge	4	15	15
h1.4xlarge	8	30	30
h1.8xlarge	8	30	30
h1.16xlarge	8	50	50
i2.xlarge	4	15	15
i2.2xlarge	4	15	15
i2.4xlarge	8	30	30
i2.8xlarge	8	30	30
i3.large	3	10	10
i3.xlarge	4	15	15
i3.2xlarge	4	15	15
i3.4xlarge	8	30	30
i3.8xlarge	8	30	30
i3.16xlarge	15	50	50
i3.metal	15	50	50
i3en.large	3	10	10
i3en.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
i3en.2xlarge	4	15	15
i3en.3xlarge	4	15	15
i3en.6xlarge	8	30	30
i3en.12xlarge	8	30	30
i3en.24xlarge	15	50	50
i3en.metal	15	50	50
i4g.large	3	10	10
i4g.xlarge	4	15	15
i4g.2xlarge	4	15	15
i4g.4xlarge	8	30	30
i4g.8xlarge	8	30	30
i4g.16xlarge	15	50	50
i4i.large	3	10	10
i4i.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
i4i.2xlarge	4	15	15
i4i.4xlarge	8	30	30
i4i.8xlarge	8	30	30
i4i.12xlarge	8	30	30
i4i.16xlarge	15	50	50
i4i.24xlarge	15	30	30
i4i.32xlarge	15	50	50
i4i.metal	15	50	50
im4gn.large	3	10	10
im4gn.xlarge	4	15	15
im4gn.2xlarge	4	15	15
im4gn.4xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
im4gn.8xlarge	8	30	30
im4gn.16xlarge	15	50	50
is4gen.medium	2	4	4
is4gen.large	3	10	10
is4gen.xlarge	4	15	15
is4gen.2xlarge	4	15	15
is4gen.4xlarge	8	30	30
is4gen.8xlarge	8	30	30

高速コンピューティング

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
dl1.24xlarge	60	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
dl2q.24xlarge	15	50	50
f1.2xlarge	4	15	15
f1.4xlarge	8	30	30
f1.16xlarge	8	50	50
g2.2xlarge	4	15	IPv6 はサポートされていません
g2.8xlarge	8	30	IPv6 はサポートされていません
g3.4xlarge	8	30	30
g3.8xlarge	8	30	30
g3.16xlarge	15	50	50
g4ad.xlarge	2	4	4
g4ad.2xlarge	2	4	4
g4ad.4xlarge	3	10	10
g4ad.8xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
g4ad.16xlarge	8	30	30
g4dn.xlarge	3	10	10
g4dn.2xlarge	3	10	10
g4dn.4xlarge	3	10	10
g4dn.8xlarge	4	15	15
g4dn.12xlarge	8	30	30
g4dn.16xlarge	4	15	15
g4dn.metal	15	50	50
g5.xlarge	4	15	15
g5.2xlarge	4	15	15
g5.4xlarge	8	30	30
g5.8xlarge	8	30	30
g5.12xlarge	15	50	50
g5.16xlarge	8	30	30

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
g5.24xlarge	15	50	50
g5.48xlarge	7	50	50
g5g.xlarge	4	15	15
g5g.2xlarge	4	15	15
g5g.4xlarge	8	30	30
g5g.8xlarge	8	30	30
g5g.16xlarge	15	50	50
g5g.metal	15	50	50
inf1.xlarge	4	10	10
inf1.2xlarge	4	10	10
inf1.6xlarge	8	30	30
inf1.24xlarge	11	30	30
inf2.xlarge	4	15	15

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
inf2.8xlarge	8	30	30
inf2.24xlarge	15	50	50
inf2.48xlarge	15	50	50
p2.xlarge	4	15	15
p2.8xlarge	8	30	30
p2.16xlarge	8	30	30
p3.2xlarge	4	15	15
p3.8xlarge	8	30	30
p3.16xlarge	8	30	30
p3dn.24xlarge	15	50	50
p4d.24xlarge	60	50	50
p4de.24xlarge	60	50	50
p5.48xlarge	64	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
trn1.2xlarge	4	15	15
trn1.32xlarge	40	50	50
trn1n.32xlarge	80	50	50
vt1.3xlarge	4	15	15
vt1.6xlarge	8	30	30
vt1.24xlarge	15	50	50

高性能コンピューティング

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
hpc6a.48xlarge	2	50	50
hpc6id.32xlarge	2	50	50
hpc7a.12xlarge	4	50	50

インスタンスタイプ	ネットワークインターフェイスの最大数	インターフェイスあたりのプライベート IPv4 アドレス数	インターフェイスあたりの IPv6 アドレス数
hpc7a.24xlarge	4	50	50
hpc7a.48xlarge	4	50	50
hpc7a.96xlarge	4	50	50
hpc7g.4xlarge	4	50	50
hpc7g.8xlarge	4	50	50
hpc7g.16xlarge	4	50	50

describe-instance-types AWS CLI コマンドを使用して、サポートされるネットワークインターフェイスやインターフェイスごとの IP アドレスなど、インスタンスタイプに関する情報を表示できます。次の例では、すべての C5 インスタンスでこれらの情報を表示します。

```
aws ec2 describe-instance-types --filters "Name=instance-type,Values=c5.*" --query
 "InstanceTypes[].{Type: InstanceType, MaxENI: NetworkInfo.MaximumNetworkInterfaces,
 IPv4addr: NetworkInfo.Ipv4AddressesPerInterface}" --output table
-------------------------------------
|        DescribeInstanceTypes        |
+----------+----------+---------------+
| IPv4addr | MaxENI   |     Type      |
+----------+----------+---------------+
|  30      |  8       |  c5.4xlarge   |
|  50      |  15      |  c5.24xlarge  |
|  15      |  4       |  c5.xlarge    |
|  30      |  8       |  c5.12xlarge  |
|  10      |  3       |  c5.large     |
|  15      |  4       |  c5.2xlarge   |
```

```
|  50      |  15      |  c5.metal     |
|  30      |  8       |  c5.9xlarge   |
|  50      |  15      |  c5.18xlarge  |
+----------+----------+---------------+
```

ネットワークインターフェイスの操作

ネットワークインターフェイスは、Amazon EC2 コンソールまたはコマンドラインを使用して操作できます。

コンテンツ

ネットワークインターフェイスの作成

ネットワークインターフェイスはサブネットで作成できます。作成後のネットワークインターフェイスは、別のサブネットに移動できません。ネットワークインターフェイスは、同じアベイラビリティーゾーンのインスタンスにアタッチする必要があります。

コンソールを使用してネットワークインターフェイスを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. [ネットワークインターフェイスの作成] をクリックします。
4. (オプション) [説明] に分かりやすい名前を入力します。
5. [Subnet (サブネット)] で、サブネットを選択します。以降のステップで使用できるオプションは、選択したサブネットのタイプ (IPv4 専用、IPv6 専用、またはデュアルスタック (IPv4 および IPv6)) によって異なります。
6. [プライベート IPv4 アドレス] で、次のいずれかの操作を実行します。

- サブネットからの IPv4 アドレスの選択を Amazon EC2 に許可するには、[自動割り当て] を選択します。
- [カスタム] を選択した場合は、続いてサブネットから選択した IPv4 アドレスを入力します。

7. (IPv6 アドレスを持つサブネットのみ) [IPv6 アドレス] で、次のいずれかの操作を行います。
   - ネットワークインターフェイスに IPv6 アドレスを割り当てたくない場合は、[なし] を選択します。
   - サブネットからの IPv6 アドレスの選択を Amazon EC2 に許可するには、[自動割り当て] を選択します。
   - [カスタム] を選択した場合は、続いてサブネットから選択した IPv6 アドレスを入力します。
8. (オプション) デュアルスタックまたは IPv6 専用サブネットにネットワーク インターフェイスを作成している場合は、プライマリ IPv6 IP の割り当てるオプションがあります。これにより、プライマリ IPv6 グローバルユニキャストアドレス (GUA) がネットワークインターフェイスに割り当てられます。プライマリ IPv6 アドレスを割り当てると、インスタンスまたは ENI へのトラフィックの中断を回避できます。この ENI が接続されるインスタンスが IPv6 アドレスが変更されないことに依存する場合は、[有効化] を選択します。 AWS は、インスタンスにアタッチされている ENI に関連付けられた IPv6 アドレスをプライマリ IPv6 アドレスとして自動的に割り当てます。IPv6 GUA アドレスをプライマリ IPv6 として有効にすると、無効にすることはできません。IPv6 GUA アドレスをプライマリ IPv6 にすることを有効にすると、インスタンスが終了するか、ネットワークインターフェースがデタッチされるまで、最初の IPv6 GUA がプライマリ IPv6 アドレスになります。インスタンスに複数の IPv6 アドレスがアタッチされていて、プライマリ IPv6 アドレスを有効にすると、ENI に関連付けられた最初の IPv6 GUA アドレスがプライマリ IPv6 アドレスになります。
9. (オプション) Elastic Fabric Adapter を作成するには、Elastic Fabric Adapter、[有効化] の順にクリックします。
10. (オプション) [詳細設定] の [アイドル接続追跡タイムアウト] で、デフォルトのアイドル接続のタイムアウトを変更します。これらのパラメータの詳細については、「アイドル接続追跡タイムアウト」を参照してください。
    - TCP 確立タイムアウト: 確立された状態のアイドル TCP 接続のタイムアウト (秒単位)。最小: 60 秒。最大: 432000 秒 (5 日間)。デフォルト: 432000 秒。推奨: 432000 秒未満。
    - UDP タイムアウト: 単一方向、または 1 つのリクエスト-レスポンストランザクションのみのトラフィックが発生した、アイドル状態にある UDP フローのタイムアウト (秒単位)。最小: 30 秒。最大: 60 秒。デフォルト: 30 秒。

- UDP ストリームタイムアウト: 複数のリクエスト-レスポンストランザクションが発生したストリームとして分類される、アイドル状態にある UDP フローのタイムアウト (秒単位)。最小: 60 秒。最大: 180 秒 (3 分)。デフォルト: 180 秒。

11. [Security groups] で、1 つまたは複数のセキュリティグループを選択します。
12. (オプション) タグごとに [新しいタグを追加] をクリックし、タグキーとオプションのタグ値を入力します。
13. [ネットワークインターフェイスの作成] をクリックします。

コマンドラインを使用してネットワークインターフェイスを作成するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- create-network-interface (AWS CLI)
- New-EC2NetworkInterface (AWS Tools for Windows PowerShell)

ネットワークインターフェイスに関する詳細の表示

アカウントのすべてのネットワークインターフェイスを表示できます。

コンソールを使用してネットワークインターフェイスを記述するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスの詳細ページを表示するには、そのネットワークインターフェイスの ID を選択します。または、ネットワークインタフェース ページを離れずに情報を表示するには、ネットワークインターフェイスのチェックボックスをオンにします。

コマンドラインを使用してネットワークインターフェイスを記述するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- describe-network-interfaces (AWS CLI)
- Get-EC2NetworkInterface (AWS Tools for Windows PowerShell)

コマンドラインを使用してネットワークインターフェイス属性を記述するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- describe-network-interface-attribute (AWS CLI)
- Get-EC2NetworkInterfaceAttribute (AWS Tools for Windows PowerShell)

インスタンスへのネットワークインターフェイスのアタッチ

同じアベイラビリティーゾーンにあるインスタンスであれば、ネットワークインターフェイスをアタッチできます。アタッチするには、Amazon EC2 コンソールの [インスタンス] または [ネットワークインターフェイス] ページを開きます。または、インスタンスを起動する際に、既存のネットワークインターフェイスを指定することもできます。

> **Important**
>
> IPv6 専用サブネット内の EC2 インスタンスの場合、セカンダリネットワークインターフェイスをインスタンスにアタッチすると、2 番目のネットワークインターフェイスのプライベート DNS ホスト名は、インスタンスの最初のネットワークインターフェイスの最初の IPv6 アドレスに解決されます。EC2 インスタンスのプライベート DNS ホスト名の詳細については、Amazon EC2 インスタンスのホスト名タイプを参照してください。

インスタンスのパブリック IPv4 アドレスがリリースされる場合、複数のネットワークインターフェイスがそのインスタンスにアタッチされていると、インスタンスに新しいパブリック IP アドレスは送信されません。パブリック IPv4 アドレスの動作の詳細については、パブリック IPv4 アドレスを参照してください。

Instances page

インスタンスページを使用してネットワークインターフェイスをインスタンスにアタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスのチェックボックスをオンにします。

4. [Actions (アクション)]、[Networking (ネットワーク)]、[Attach Network Interface (ネットワークインターフェイスのアタッチ)] の順に選択します。
5. VPC を選択します。セカンダリネットワークインターフェイスをインスタンスにアタッチする場合、ネットワークインターフェイスはインスタンスと同じ VPC にあっても、または所有している別の VPC にあってもかまいません (ネットワークインターフェイスがインスタンスと同じアベイラビリティーゾーンにあるサブネットにある場合)。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。
6. ネットワークインターフェイスを選択します。インスタンスが複数のネットワークカードをサポートしている場合は、ネットワークカードを選択できます。
7. [アタッチ] を選択します。

Network Interfaces page

ネットワークインターフェイスページを使用してネットワークインターフェイスをインスタンスにアタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[アタッチ] の順にクリックします。
5. インスタンスを選択します。インスタンスが複数のネットワークカードをサポートしている場合は、ネットワークカードを選択できます。
6. [アタッチ] を選択します。

コマンドラインを使用してインスタンスにネットワークインターフェイスをアタッチするには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

> **Note**
>
> attach-network-interface AWS CLI コマンドを使用して、別の VPC (ただし同じアベイラビリティーゾーンにある) にあるネットワークインターフェースをインスタンスにアタッチできます。AWS Management Console を使用してこれを行うことはできません。

- attach-network-interface (AWS CLI)
- Add-EC2NetworkInterface (AWS Tools for Windows PowerShell)

インスタンスからのネットワークインターフェイスのデタッチ

ある時点で EC2 インスタンスにアタッチされているセカンダリネットワークインターフェイスは、Amazon EC2 コンソールの [インスタンス] ページまたは [ネットワークインターフェイス] ページを使用して、いつでもデタッチできます。

Elastic Load Balancing ロードバランサー、Lambda 関数、WorkSpace、NAT ゲートウェイなど、別のサービスからリソースにアタッチされているネットワークインターフェイスをデタッチしようとすると、そのリソースに対するアクセス許可がないことを知らせるエラーが表示されます。ネットワークインターフェイスにアタッチされているリソースを作成したサービスを特定するには、そのネットワークインターフェイスの説明を確認します。リソースを削除すると、そのネットワークインターフェイスも削除されます。

Instances page

インスタンスページを使用してネットワークインターフェイスをインスタンスからデタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスのチェックボックスをオンにします。[ネットワーキング] タブにある [ネットワークインタフェース] セクションを選択して、そのネットワークインターフェイスが、セカンダリネットワークインターフェイスとしてインスタンスにアタッチされていることを確認します。
4. [Actions (アクション)]、[Networking (ネットワーク)]、[Detach Network Interface (ネットワークインターフェイスのデタッチ)] の順に選択します。
5. ネットワークインターフェイスを選択し、[デタッチ] を選択します。

Network Interfaces page

ネットワークインターフェイスページを使用してネットワークインターフェイスをインスタンスからデタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。[詳細] タブの [インスタンスの詳細] セクションを選択し、そのネットワークインターフェイスが、セカンダリネットワークインターフェイスとしてインスタンスにアタッチされていることを確認します。
4. [アクション]、[デタッチ] の順にクリックします。
5. 確認を求められたら、[デタッチ] を選択します。
6. ネットワークインターフェイスをインスタンスからデタッチできなかった場合は、[強制デタッチ]、[有効化] の順にクリックし再試行します。強制デタッチは、最終的手段としてのみご使用になることをお勧めします。デタッチを強制すると、インスタンスを再起動するまで、同じインデックスに別のネットワークインターフェイスをアタッチできなくなります。また、インスタンスを再起動するまで、ネットワークインターフェイスがデタッチされたことをインスタンスメタデータに反映しないようにできます。

コマンドラインを使用してネットワークインターフェイスをデタッチするには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- detach-network-interface (AWS CLI)
- Dismount-EC2NetworkInterface (AWS Tools for Windows PowerShell)

IP アドレスの管理

ネットワークインターフェースの次の IP アドレスを管理できます。

- Elastic IP アドレス (プライベート IPv4 アドレスごとに 1 つ)
- IPv4 アドレス
- IPv6 アドレス
- プライマリ IPv6 アドレス

コンソールを使用してネットワークインターフェイスの Elastic IP アドレスを管理するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。

4. Elastic IP アドレスを関連付けるには、次の操作を行います。

   a. [アクション]、[アドレスの関連付け] の順にクリックします。

   b. [Elastic IP アドレス] で、Elastic IP アドレスを選択します。

   c. [プライベート IPv4 アドレス] で、Elastic IP アドレスに関連付けるプライベート IPv4 アドレスを選択します。

   d. (オプション) 現在、ネットワークインターフェイスが別のインスタンスまたはネットワークインターフェイスに関連付けられている場合は、[Elastic IP アドレスの再関連付けを許可する] をクリックします。

   e. [Associate] を選択します。

5. Elastic IP アドレスの関連付けを解除するには、次の手順を実行します。

   a. [Actions]、[Disassociate address] の順に選択します。

   b. [パブリック IP アドレス] で、Elastic IP アドレスを選択します。

   c. [関連付け解除] を選択します。

コンソールを使用してネットワークインターフェイスの IPv4 アドレスと IPv6 アドレスを管理するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスを選択します。
4. [アクション]、[IP アドレスの管理] の順にクリックします。
5. ネットワークインターフェイスを展開します。
6. [IPv4 アドレス] で、必要に応じて IP アドレスを変更します。IPv4 アドレスを割り当てるには、[新しい IP アドレスの割り当て] をクリックし、サブネット範囲にある IPv4 アドレスを指定します。あるいは、AWS により自動的に選択させます。IPv4 アドレスの割り当てを解除するには、アドレスの横にある [Unassign (割り当て解除)] を選択します。
7. [IPv6 アドレス] で、必要に応じて IP アドレスを変更します。IPv6 アドレスを割り当てるには、[新しい IP アドレスの割り当て] を選択し、サブネット範囲にある IPv6 アドレスを指定します。あるいは、AWS により自動的に選択させます。IPv6 アドレスの割り当てを解除するには、アドレスの横にある [Unassign (割り当て解除)] を選択します。
8. (オプション) デュアルスタックまたは IPv6 専用サブネット内のネットワークインターフェイスを変更する場合は、プライマリ IPv6 IP の割り当てるオプションがあります。プライマリ

IPv6 アドレスを割り当てると、インスタンスまたは ENI へのトラフィックの中断を回避できます。この ENI が接続されるインスタンスが IPv6 アドレスが変更されないことに依存する場合は、[有効化] を選択します。 AWS は、インスタンスにアタッチされている ENI に関連付けられた IPv6 アドレスをプライマリ IPv6 アドレスとして自動的に割り当てます。IPv6 GUA アドレスをプライマリ IPv6 として有効にすると、無効にすることはできません。IPv6 GUA アドレスをプライマリ IPv6 にすることを有効にすると、インスタンスが終了するか、ネットワークインターフェースがデタッチされるまで、最初の IPv6 GUA がプライマリ IPv6 アドレスになります。インスタンスに複数の IPv6 アドレスがアタッチされていて、プライマリ IPv6 アドレスを有効にすると、ENI に関連付けられた最初の IPv6 GUA アドレスがプライマリ IPv6 アドレスになります。

9. [Save] を選択します。

AWS CLI を使用してネットワークインターフェースの IP アドレスを管理するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- assign-ipv6-addresses
- associate-address
- disassociate-address
- unassign-ipv6-addresses

Tools for Windows PowerShell を使用してネットワークインターフェースの IP アドレスを管理するには

次のいずれかのコマンドを使用できます。

- Register-EC2Address
- Register-EC2Ipv6AddressList
- Unregister-EC2Address
- Unregister-EC2Ipv6AddressList

ネットワークインターフェイス属性の変更

次のネットワークインターフェース属性を変更できます。

- 説明
- セキュリティグループ
- 終了時に削除
- 送信元/送信先チェック

コンソールを使用してネットワークインターフェイスの説明を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[説明の変更] の順にクリックします。
5. [説明] に、ネットワークインターフェイスの説明を入力します。
6. [Save] を選択します。

コンソールを使用してネットワークインターフェイスのセキュリティグループを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[セキュリティグループの変更] の順にクリックします。
5. [関連付けられたセキュリティグループ] で、使用するセキュリティグループを選択し、[保存] をクリックします。

   セキュリティグループとネットワークインターフェイスは、同じ VPC に対して作成する必要があります。Elastic Load Balancing などの他のサービスが所有するインターフェイスのセキュリティグループを変更するには、そのサービスを通じて変更します。

コンソールを使用してネットワークインターフェイスの終了時の動作を変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。

3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[終了時の動作を変更] の順にクリックします。
5. 必要に応じて、[終了時に削除] を選択またはクリアし [有効化] をクリックした上で、[保存] をクリックします。

コンソールを使用してネットワークインターフェイスの送信元/送信先チェックを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[送信元/送信先チェックの変更] の順にクリックします。
5. 必要に応じて、[送信元/送信先チェック] を選択またはクリアし [有効化] をクリックした上で、[保存]をクリックします。

アイドル接続追跡タイムアウトを変更するには:

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [アクション]、[接続タイムアウトの変更] を選択します。
5. アイドル接続追跡タイムアウトを変更します。これらのパラメータの詳細については、「アイドル接続追跡タイムアウト」を参照してください。

   - TCP 確立タイムアウト: 確立された状態のアイドル TCP 接続のタイムアウト (秒単位)。最小: 60 秒。最大: 432000 秒 (5 日間)。デフォルト: 432000 秒。推奨: 432000 秒未満。
   - UDP タイムアウト: 単一方向、または 1 つのリクエスト-レスポンストランザクションのみのトラフィックが発生した、アイドル状態にある UDP フローのタイムアウト (秒単位)。最小: 30 秒。最大: 60 秒。デフォルト: 30 秒。
   - UDP ストリームタイムアウト: 複数のリクエスト-レスポンストランザクションが発生したストリームとして分類される、アイドル状態にある UDP フローのタイムアウト (秒単位)。最小: 60 秒。最大: 180 秒 (3 分)。デフォルト: 180 秒。

6. [Save] を選択します。

コマンドラインを使用してネットワークインターフェース属性を変更するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- modify-network-interface-attribute (AWS CLI)
- Edit-EC2NetworkInterfaceAttribute (AWS Tools for Windows PowerShell)

タグの追加または編集

タグとは、ネットワークインターフェイスに追加できるメタデータです。タグはプライベートとして扱われ、アカウントでのみ表示できます。各タグはキーとオプションの値で構成されます。タグの詳細については、Amazon EC2 リソースのタグ付けを参照してください。

コンソールを使用してネットワークインターフェイスのタグを追加または編集するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにします。
4. [タグ] タブで、[タグを管理] をクリックします。
5. 作成するタグごとに、[新しいタグを追加] をクリックし、キーとオプションの値を入力します。完了したら、[Save] を選択します。

コマンドラインを使用してネットワークインターフェイスのタグを追加または編集するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- create-tags (AWS CLI)
- New-EC2Tag (AWS Tools for Windows PowerShell)

ネットワークインターフェイスの削除

ネットワークインターフェイスを削除すると、そのインターフェイスに関連付けられているすべての属性がリリースされ、別のインスタンスで使用できるように、プライベート IP アドレスまたは Elastic IP アドレスがリリースされます。

使用中のネットワークインターフェイスは削除できません。先に、ネットワークインターフェイスをデタッチする必要があります。

コンソールを使用してネットワークインターフェイスを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. ネットワークインターフェイスのチェックボックスをオンにし、[アクション]、[削除] の順にクリックします。
4. 確認を求めるメッセージが表示されたら、[削除] を選択します。

コマンドラインを使用してネットワークインターフェイスを削除するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- delete-network-interface (AWS CLI)
- Remove-EC2NetworkInterface (AWS Tools for Windows PowerShell)

ネットワークインターフェイスの設定に関するベストプラクティス

- ネットワークインターフェイスは、インスタンスの実行中、インスタンスの停止中、インスタンスの起動中にインスタンスにアタッチできます (それぞれ、ホットアタッチ、ウォームアタッチ、コールドアタッチと呼ばれています)。
- セカンダリネットワークインターフェイスは、インスタンスの実行中または停止中にデタッチできます。ただし、プライマリネットワークインターフェイスをデタッチすることはできません。
- インスタンスが同じアベイラビリティーゾーンと VPC にあるが異なるサブネットにある場合、セカンダリネットワークインターフェイスを 1 つのインスタンスから別のインスタンスに移動できます。
- CLI、API、または SDK を使用してインスタンスを起動する場合、プライマリネットワークインターフェイスおよび追加のネットワークインターフェイスを指定できます。
- 複数のネットワークインターフェイスを使用して Amazon Linux または Windows Server インスタンスを起動すると、インスタンスのオペレーティングシステム上でインターフェイス、プライベート IPv4 アドレス、ルートテーブルが自動的に設定されます。

- 追加ネットワークインターフェイスをウォームアタッチまたはホットアタッチする際、場合によっては、手動で 2 つ目のインターフェイスを起動し、プライベート IPv4 アドレスを設定し、ルートテーブルを適宜変更する必要が生じます。Amazon Linux または Windows Server を実行するインスタンスは、ウォームアタッチまたはホットアタッチを自動的に認識し、それらのインスタンス自体を設定します。
- 別のネットワークインターフェイスをインスタンスにアタッチすること (NIC チーミング設定など) で、デュアルホーム接続インスタンスに対するネットワーク帯域幅を、増加または倍増させることはできません。
- 同じサブネットから複数のネットワークインターフェイスをインスタンスにアタッチすると、非対称ルーティングなどのネットワーク問題が発生する場合があります。可能であれば、代わりにプライマリネットワークインターフェイス上でセカンダリプライベート IPv4 アドレスを使用します。

Amazon Linux 2 向けに ec2-net-utils を使用してネットワークインターフェイスを設定する

> **Note**
>
> AL2023 の場合、`amazon-ec2-net-utils` パッケージはインターフェイス固有の設定を `/run/systemd/network` ディレクトリに生成します。詳細については、「 Amazon Linux 2023 ユーザーガイド」の「ネットワーキングサービス」を参照してください。

Amazon Linux 2 AMI には、ec2-net-utils という、AWS がインストールした追加のスクリプトが含まれることがあります。これらのスクリプトはオプションで、ネットワークインターフェイスの設定を自動化します。これらのスクリプトは Amazon Linux 2 でのみ使用できます。

パッケージをまだインストールしていない場合は、以下のコマンドを使用して Amazon Linux 2 にインストールします。インストール済みの場合、追加の更新があれば、更新します。

```
$ yum install ec2-net-utils
```

ec2-net-utils には、以下のコンポーネントが含まれます。

udev ルール (`/etc/udev/rules.d`)

実行中のインスタンスにネットワークインターフェイスがアタッチ、デタッチ、または再アタッチされたときに、そのネットワークインターフェイスを特定し、ホットプラグスクリプト

が実行されることを確認します (53-ec2-network-interfaces.rules)。MAC アドレスをデバイス名にマッピングします (75-persistent-net-generator.rules を生成する 70-persistent-net.rules)。

ホットプラグスクリプト

DHCP での使用に適したインターフェイス設定ファイルを生成します (/etc/sysconfig/network-scripts/ifcfg-ethN)。また、ルート設定ファイルも生成します (/etc/sysconfig/network-scripts/route-ethN)。

DHCP スクリプト

ネットワークインターフェイスが新しい DHCP リースを受け取るたびに、このスクリプトがインスタンスメタデータに対し、Elastic IP アドレスを求めるクエリを実行します。これにより、各 Elastic IP アドレスごとに、そのアドレスからのアウトバンドトラフィックが正しいネットワークインターフェイスを使用するよう、ルーティングポリシーデータベースにルールが追加されます。また、各プライベート IP アドレスを、セカンダリアドレスとしてネットワークインターフェイスに追加します。

ec2ifup ethN (/usr/sbin/)

標準の ifup の機能を拡張します。このスクリプトが設定ファイル ifcfg-ethN および route-ethN を書き換えた後、ifup を実行します。

ec2ifdown ethN (/usr/sbin/)

標準の ifdown の機能を拡張します。このスクリプトがルーティングポリシーデータベースからネットワークインターフェイスのルールをすべて削除した後、ifdown を実行します。

ec2ifscan (/usr/sbin/)

まだ設定されていないネットワークインターフェイスを探して、それらを設定します。

このスクリプトは、ec2-net-utils の初期リリースでは提供されていません。

ec2-net-utils によって生成された設定ファイルをリストするには、以下のコマンドを使用します。

```
$ ls -l /etc/sysconfig/network-scripts/*-eth?
```

オートメーションを無効にするには、対応する ifcfg-ethN ファイルに EC2SYNC=no を追加します。例えば、eth1 インターフェイスの自動化を無効にするには、以下のコマンドを使用します。

```
$ sed -i -e 's/^EC2SYNC=yes/EC2SYNC=no/' /etc/sysconfig/network-scripts/ifcfg-eth1
```

オートメーションを完全に無効にするには、次のコマンドを使用してパッケージを削除できます。

```
$ yum remove ec2-net-utils
```

ネットワークインターフェイスのシナリオ

次の作業を行う場合、複数のネットワークインターフェイスをインスタンスにアタッチすると便利です。

- 管理用ネットワークを作成する。
- 仮想プライベートクラウド (VPC) 内でネットワークアプライアンスやセキュリティアプライアンスを使用します。
- 別個のサブネット上のワークロード/ロールを使用するデュアルホーム接続インスタンスを作成する。
- 低予算で可用性の高いソリューションを作成する。

管理用ネットワークの作成

このシナリオでは、次の基準と設定に基づいて、ネットワークインターフェイスを備えた管理ネットワークを作成する方法について説明します (次の図を参照してください)。

条件

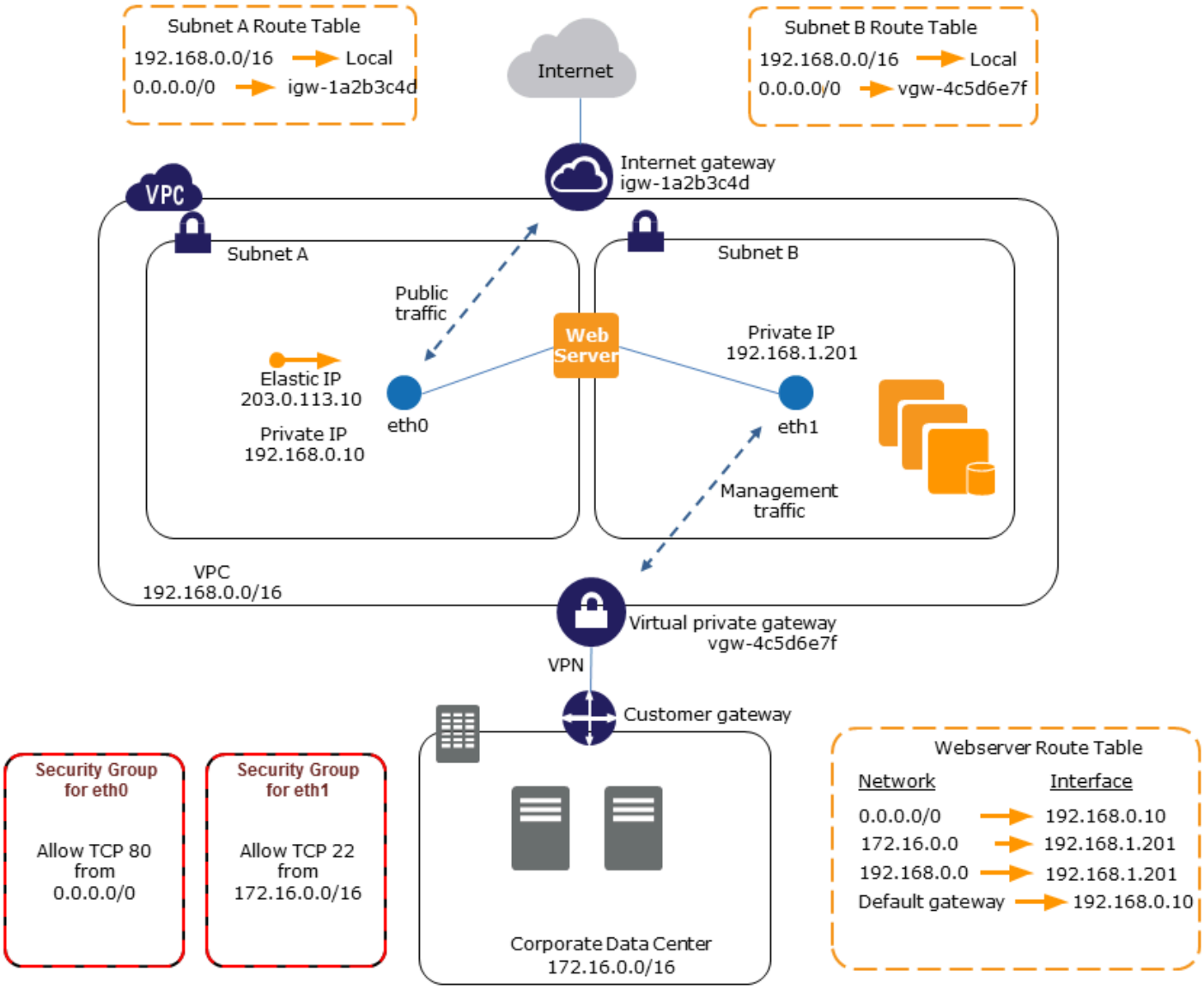
- インスタンスのプライマリネットワークインターフェイス (eth0) が、パブリックトラフィックを処理します。
- インスタンスのセカンダリネットワークインターフェイス (eth1) が、バックエンド管理トラフィックを処理します。より制限の厳しいアクセス制御を使用した個別のサブネットに接続されており、プライマリネットワークインターフェイスと同じアベイラビリティーゾーン (AZ) 内にあります。

設定

- プライマリネットワークインターフェイスは、ロードバランサーの背後にある場合とそうでない場合があり、インターネットからのサーバーへのアクセスを許可するセキュリティグループが関連付けられています。例えば、`0.0.0.0/0` またはロードバランサーからの TCP ポート 80 と 443 を許可します。

- セカンダリネットワークインターフェイスには、次のいずれかの場所から開始された SSH アクセスのみを許可するセキュリティグループが関連付けられています。
  - VPC 内またはインターネット上の IP アドレスの許容範囲。
  - プライマリネットワークインターフェースと同じ AZ 内のプライベートサブネット。
  - 仮想プライベートゲートウェイ。

> **Note**
>
> フェイルオーバー機能が確実に動作するように、ネットワークインターフェイスの受信トラフィックに対してセカンダリプライベート IPv4 を使用することをお勧めします。インスタンスに障害が発生した場合は、インターフェイスまたはセカンダリプライベート IPv4 アドレスあるいはその両方をスタンバイ用のインスタンスに移行できます。



VPC 内でネットワークアプライアンスとセキュリティアプライアンスを使用する

ロードバランサー、ネットワークアドレス変換 (NAT) サーバー、プロキシサーバーなど、ネットワークアプライアンスやセキュリティアプライアンスの中には、複数のネットワークインターフェイスを使用した構成が優先されるものがあります。セカンダリネットワークインターフェイスを作成して、これらのタイプのアプリケーションを実行するインスタンスにアタッチし、専用のパブリック IP アドレスとプライベート IP アドレス、セキュリティグループ、およびソース/デスティネーションチェックを使用するインターフェイスを、追加で構成することができます。

別個のサブネット上のワークロード/ロールを使用するデュアルホーム接続インスタンスを作成する

アプリケーションサーバーが存在するミッドティアネットワークに接続する Web サーバーのそれぞれにネットワークインターフェイスを置くことができます。このアプリケーションサーバーは、データベースサーバーが存在するバックエンドネットワーク (サブネット) にデュアルホーム接続することもできます。デュアルホーム接続されたインスタンスを介してネットワークパケットをルーティングする代わりに、デュアルホーム接続された各インスタンスは、フロントエンドでリクエストを受信して処理し、バックエンドとの接続を開始して、バックエンドネットワーク上のサーバーにリクエストを送信します。

同一アカウント内の個別の VPC にあるワークロード/ロールを使用するデュアルホームインスタンスを作成する

ある VPC で EC2 インスタンスを起動し、別の VPC (ただし同じアベイラビリティーゾーンにある) のセカンダリ ENI をインスタンスにアタッチできます。これにより、ネットワークとセキュリティの設定が異なる VPC にまたがるマルチホームインスタンスを作成できます。異なる AWS アカウントの VPC にまたがってマルチホームインスタンスを作成することはできません。

VPC にまたがるデュアルホームインスタンスは、次のユースケースで使用できます。

- 相互にピアリングできない 2 つの VPC 間における CIDR の重複を克服: VPC のセカンダリ CIDR を活用して、重複しない 2 つの IP 範囲でインスタンスが通信できるようにします。
- 単一アカウント内で複数の VPC を接続: 通常は VPC の境界で区切られている個々のリソース間における通信を可能にします。

低予算で可用性の高いソリューションを構築する

特定の機能にサービスを提供しているインスタンスのいずれかが機能しなくなった場合は、そのネットワークインターフェイスを同じ役割で構成された交換用またはホットスタンバイ用のインスタンスにアタッチすることで、サービスを迅速に回復できます。例えば、データベースインスタンスや NAT インスタンスなどの重要なサービスに対するプライマリまたはセカンダリのネットワークインターフェイスとしてネットワークインターフェイスを使用することができます。そのインスタンスが機能しなくなった場合、お客様 (通常はお客様に代わって実行されるコード) がネットワークインターフェイスをホットスタンバイ用のインスタンスにアタッチすることができます。インターフェイスでは、プライベート IP アドレス、Elastic IP アドレス、および MAC アドレスがそのまま維持されるため、交換用のインスタンスにネットワークインターフェイスを接続するとすぐに、ネットワーク

トラフィックはスタンバイ用のインスタンスに流れ始めます。インスタンスに障害が発生してから、ネットワークインターフェイスがスタンバイ用のインスタンスにアタッチされるまで一時的な接続断が発生しますが、ルートテーブルや DNS サーバーに変更を加える必要はありません。

リクエスタマネージド型のネットワークインターフェイス

リクエスタマネージド型ネットワークインターフェイスは、AWS のサービスがユーザーに代わって VPC 内に作成するネットワークインターフェイスです。ネットワークインターフェイスは、Amazon RDS の DB インスタンス、NAT ゲートウェイ、または AWS PrivateLink のインターフェイス VPC エンドポイントなど、別のサービスのリソースに関連付けられています。

考慮事項

- アカウントにあるリクエスタマネージド型ネットワークインターフェイスを確認できます。タグを追加または削除することはできますが、リクエスタマネージド型ネットワークインターフェイスの他のプロパティは変更できません。
- リクエスタマネジード型ネットワークインターフェイスをデタッチすることはできません。
- リクエスタマネジード型ネットワークインターフェイスに関連付けられているリソースを削除すると、AWS のサービスはネットワークインターフェイスをデタッチして削除します。サービスがネットワークインターフェースをデタッチしたが、削除しなかった場合は、デタッチされたネットワークインターフェイスを削除できます。

コンソールを使用してリクエスタマネージド型のネットワークインターフェイスを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network & Security] (ネットワークとセキュリティ)、[Network Interfaces] (ネットワークインターフェイス) を選択します。
3. ネットワークインターフェイスの ID を選択して、その詳細ページを開きます。
4. ネットワークインターフェイスの目的を特定するために使用できる主要なフィールドを次に示します。

    - [Description] (説明): インターフェイスを作成した AWS のサービスによって提供される説明。例えば、「VPC Endpoint Interface vpce 089f2123488812123」です。
    - Requester-managed：ネットワークインターフェイスが AWS によって管理されているかどうかを示します。
    - [Requester ID] (リクエスタ ID)：ネットワークインターフェイスを作成したプリンシパルまたはサービスのエイリアスまたはAWSアカウント ID ネットワークインターフェイスを作成した

場合、これはユーザのAWS アカウントID です。それ以外の場合は、別のプリンシパルまたはサービスによって作成されています。

AWS CLI を使用してリクエスタマネージド型のネットワークインターフェイスを表示するには

次のように、describe-network-interfaces コマンドを使用します。

```
aws ec2 describe-network-interfaces --filters Name=requester-managed,Values=true
```

ネットワークインターフェイス Description および InterfaceType の目的を決定するために使用できるキーフィールドを示す出力例を次に示します。

```
{
     ...
    "Description": "VPC Endpoint Interface vpce-089f2123488812123",
    ...
    "InterfaceType": "vpc_endpoint",
    ...
    "NetworkInterfaceId": "eni-0d11e3ccd2c0e6c57",
    ...
    "RequesterId": "727180483921",
    "RequesterManaged": true,
    ...
}
```

Tools for Windows PowerShell を使用してリクエスタマネージド型のネットワークインターフェイスを表示するには

次のように、Get-EC2NetworkInterface cmdlet を使用します。

```
Get-EC2NetworkInterface -Filter @{ Name="requester-managed"; Values="true" }
```

ネットワークインターフェイス Description および InterfaceType の目的を決定するために使用できるキーフィールドを示す出力例を次に示します。

```
Description        : VPC Endpoint Interface vpce-089f2123488812123
...
InterfaceType      : vpc_endpoint
...
```

```
NetworkInterfaceId : eni-0d11e3ccd2c0e6c57
...
RequesterId        : 727180483921
RequesterManaged   : True
...
```

Amazon EC2 ネットワークインターフェイスへのプレフィクスの割り当て

プライベート IPv4 または IPv6 CIDR 範囲は、自動または手動で、ネットワークインターフェイスに割り当てることができます。プレフィクスを割り当てることにより、インスタンス上で複数の IP アドレスを必要とするコンテナおよびネットワークアプリケーションなど、アプリケーションの管理を拡張および簡素化します。IPv4 アドレスと IPv6 アドレスの詳細については、「Amazon EC2 インスタンスの IP アドレス指定」を参照してください。

以下の割り当てオプションが利用できます。

- 自動割り当て — AWS が、VPC サブネットの IPv4 または IPv6 CIDR ブロックからプレフィクスを選択し、ネットワークインターフェイスに割り当てます。
- 手動割り当て — ユーザーが VPC サブネットの IPv4 または IPv6 CIDR ブロックからプレフィクスを指定し、AWS はそのプレフィクスが他のリソースに割り当てられていないことを確認してから、そのプレフィクスをネットワークインターフェイスに割り当てます。

プレフィクスの割り当てには次の利点があります。

- ネットワークインターフェイスの IP アドレスの増加 — プレフィクスを使用する場合は、個々の IP アドレスではなく IP アドレスのブロックを割り当てます。これにより、ネットワークインターフェイスの IP アドレスの数が増加します。
- コンテナの VPC 管理の簡素化 — コンテナアプリケーションでは、各コンテナに一意の IP アドレスが必要です。インスタンスにプレフィクスを割り当てることで、個々の IP 割り当てに対して Amazon EC2 API を呼び出すことなくコンテナを起動および終了できるため、VPC の管理が簡素化されます。

コンテンツ

プレフィクス割り当ての基本

- 新しいネットワークインターフェイスまたは既存のネットワークインターフェイスにプレフィクスを割り当てることができます。
- プレフィクスを使用するには、ネットワークインターフェイスにプレフィクスを割り当て、次にそのネットワークインターフェイスをインスタンスにアタッチしてから、オペレーティングシステムを設定します。
- プレフィクスを指定するオプションを選択する場合、プレフィクスは次の要件を満たしている必要があります。
  - 指定できる IPv4 プレフィクスは /28。
  - 指定できる IPv6 プレフィクスは /80。
  - プレフィクスがネットワークインターフェイスのサブネット CIDR にあり、サブネット内の既存のリソースに割り当てられた他のプレフィクスまたは IP アドレスと重複していないこと。
- プレフィクスは、プライマリまたはセカンダリのネットワークインターフェイスに割り当てることができます。
- プレフィクスが割り当てられているネットワークインターフェイスに Elastic IP アドレスを割り当てることができます。
- 割り当てられたプレフィクスの IP アドレス部分に Elastic IP アドレスを割り当てることもできます。
- インスタンスのプライベート DNS ホスト名をプライマリのプライベート IPv4 アドレスに解決します。
- プレフィクスからのものを含め、ネットワークインターフェイスの各プライベート IPv4 アドレスは、次の形式を使用して割り当てます。
  - us-east-1 リージョン

    ```
    ip-private-ipv4-address.ec2.internal
    ```

  - その他のすべてのリージョン

    ```
    ip-private-ipv4-address.region.compute.internal
    ```

プレフィクスの考慮事項と制限

プレフィクスを使用する場合は、次の点を考慮してください:

- プレフィクス付きのネットワークインターフェイスは、Nitro システムに内蔵されたインスタンスでサポートされます。
- ネットワークインターフェイスのプレフィクスは、IPv6 アドレスとプライベート IPv4 アドレスに制限されます。
- ネットワークインターフェイスに割り当てることができる IP アドレスの数は、インスタンスタイプによって異なります。ネットワークインターフェイスに割り当てる各プレフィクスが、1 つの IP アドレスとしてカウントされます。例えば、`c5.large` インスタンスで、ネットワークインターフェイスあたりの IPv4 アドレス数が `10` に制限されているとします。このインスタンスの各ネットワークインターフェイスに、プライマリ IPv4 アドレスが存在します。ネットワークインターフェイスにセカンダリ IPv4 アドレスがない場合は、ネットワークインターフェイスに最大 9 つのプレフィクスを割り当てることができます。ネットワークインターフェイスに割り当てる IPv4 アドレスを追加する度に、ネットワークインターフェイスに割り当てるプレフィクスの数が 1 つ少なくなります。詳細については、各インスタンスタイプのネットワークインターフェイスあたりの IP アドレス数 を参照してください。
- プレフィクスは、送信元/送信先チェックに含まれます。

プレフィクスの使用

ネットワークインターフェースでは、次のようにプレフィックスを使用できます。

タスク

- ネットワークインターフェイスの作成時にプレフィクスを割り当てる
- 既存のネットワークインターフェイスにプレフィクスを割り当てる
- プレフィクス付きのネットワークインターフェイス用にオペレーティングシステムを設定する
- ネットワークインターフェイスに割り当てられたプレフィクスの表示
- ネットワークインターフェイスからプレフィクスを削除する

ネットワークインターフェイスの作成時にプレフィクスを割り当てる

自動割り当てオプションを使用する場合は、サブネット内の IP アドレスのブロックを予約できます。AWS は、このブロックからプレフィクスを選択します。詳細については、Amazon VPC ユーザーガイドのサブネット CIDR の予約を参照してください。

ネットワークインターフェイスを作成したら、attach-network-interfaceAWS CLIコマンドを使用して、ネットワークインターフェイスをインスタンスにアタッチします。プレフィクス付きのネット

ワークインターフェイス用にオペレーティングシステムを設定する必要があります。詳細については、「プレフィクス付きのネットワークインターフェイス用にオペレーティングシステムを設定する」を参照してください。

タスク

- ネットワークインターフェイスの作成時に自動的にプレフィクスを割り当てる
- ネットワークインターフェイスの作成時に特定のプレフィクスを割り当てる

ネットワークインターフェイスの作成時に自動的にプレフィクスを割り当てる

以下のいずれかの方法を使用して、ネットワークインターフェイスの作成中に自動プレフィクスを割り当てることができます。

Console

ネットワークインターフェース作成時に自動的にプレフィクスを割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [ネットワークインターフェース] 選択し、それから[ネットワークインターフェイス]を選択します。
3. ネットワークインターフェイスの説明を指定し、ネットワークインターフェイスを作成するサブネットを選択し、プライベート IPv4 アドレスと IPv6 アドレスを設定します。
4. [詳細設定]を展開し、以下の操作を行います。

   a. IPv4 プレフィクスを自動的に割り当てるには、[IPv4 プレフィクスの委任] で、[自動割り当て]を選択します。その後、IPv4 プレフィクスの数で、割り当てるプレフィクスの数を指定します。

   b. IPv6 プレフィクスの自動割り当てるには、IPv6 プレフィクスの委任] で、[自動割り当て]を選択します。その後、IPv6 プレフィクスの数で、割り当てるプレフィクスの数を指定します。

   > **Note**
   >
   > [IPv6 プレフィクスの委任]は、選択したサブネットがIPv6 に対して有効になっている場合にのみ表示されます。

5. ネットワークインターフェイスに関連付けるセキュリティグループを選択し、必要に応じてリソースタグを割り当てます。

6. [ネットワークインターフェイスの作成] をクリックします。

AWS CLI

ネットワークインターフェース作成時に自動的に IPv4 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、AWS が割り当てるプレフィクスの数を --ipv4-prefix-count に設定します。次の例では、AWS が 1 プレフィクスを割り当てています。

```
$ aws ec2 create-network-interface \
--subnet-id subnet-047cfed18eEXAMPLE \
--description "IPv4 automatic example" \
--ipv4-prefix-count 1
```

出力例

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "IPv4 automatic example",
        "Groups": [
            {
                "GroupName": "default",
                "GroupId": "sg-044c2de2c4EXAMPLE"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "02:98:65:dd:18:47",
        "NetworkInterfaceId": "eni-02b80b4668EXAMPLE",
        "OwnerId": "123456789012",
        "PrivateIpAddress": "10.0.0.62",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateIpAddress": "10.0.0.62"
            }
        ],
        "Ipv4Prefixes": [
            {
                "Ipv4Prefix": "10.0.0.208/28"
            }
```

```
        ],
        "RequesterId": "AIDAIV5AJI5LXF5XXDPCO",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-047cfed18eEXAMPLE",
        "TagSet": [],
        "VpcId": "vpc-0e12f52b21EXAMPLE"
    }
}
```

ネットワークインターフェース作成時に自動的に IPv6 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、AWS が割り当てるプレフィクスの数を --ipv6-prefix-count に設定します。次の例では、AWS が 1 プレフィクスを割り当てています。

```
$ aws ec2 create-network-interface \
--subnet-id subnet-047cfed18eEXAMPLE \
--description "IPv6 automatic example" \
--ipv6-prefix-count 1
```

出力例

```
{
    "NetworkInterface": {
        "AvailabilityZone": "us-west-2a",
        "Description": "IPv6 automatic example",
        "Groups": [
            {
                "GroupName": "default",
                "GroupId": "sg-044c2de2c4EXAMPLE"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "02:bb:e4:31:fe:09",
        "NetworkInterfaceId": "eni-006edbcfa4EXAMPLE",
        "OwnerId": "123456789012",
        "PrivateIpAddress": "10.0.0.73",
        "PrivateIpAddresses": [
            {
                "Primary": true,
```

```
                "PrivateIpAddress": "10.0.0.73"
            }
        ],
        "Ipv6Prefixes": [
            {
                "Ipv6Prefix": "2600:1f13:fc2:a700:1768::/80"
            }
        ],
        "RequesterId": "AIDAIV5AJI5LXF5XXDPCO",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "pending",
        "SubnetId": "subnet-047cfed18eEXAMPLE",
        "TagSet": [],
        "VpcId": "vpc-0e12f52b21EXAMPLE"
    }
}
```

ネットワークインターフェイスの作成時に特定のプレフィクスを割り当てる

以下のいずれかの方法を使用して、ネットワークインターフェイスの作成中に特定のプレフィクスを割り当てることができます。

Console

ネットワークインターフェース作成時に特定のプレフィクスを割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [ネットワークインターフェース] 選択し、それから[ネットワークインターフェイス]を選択します。
3. ネットワークインターフェイスの説明を指定し、ネットワークインターフェイスを作成するサブネットを選択し、プライベート IPv4 アドレスと IPv6 アドレスを設定します。
4. [アドバンスト設定] で、以下の操作を行います。

    a. 特定の IPv4 プレフィクスを割り当てるには、[IPv4 プレフィクスの委任] で、[カスタム] を選択します。次に[新しいプレフィクスの追加] を選択し、使用するプレフィクスの入力を行います。

    b. 特定の IPv6 プレフィクスを割り当てるには、[IPv6 プレフィクスの委任] で、[カスタム] を選択します。次に [新しいプレフィクスの追加] を選択し、使用するプレフィクスの入力を行います。

> **Note**
>
> [IPv6 プレフィクスの委任]は、IPv6 に対して選択したサブネットが有効になっている場合にのみ表示されます。

5. ネットワークインターフェイスに関連付けるセキュリティグループを選択し、必要に応じてリソースタグを割り当てます。
6. [ネットワークインターフェイスの作成] をクリックします。

AWS CLI

ネットワークインターフェース作成時に特定の IPv4 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、`--ipv4-prefixes` にプレフィクスを設定します。AWS はこの範囲から IP アドレスを選択します。次の例では、プレフィクス CIDR は `10.0.0.208/28` です。

```
$ aws ec2 create-network-interface \
    --subnet-id subnet-047cfed18eEXAMPLE \
    --description "IPv4 manual example" \
    --ipv4-prefixes Ipv4Prefix=10.0.0.208/28
```

出力例

```
{
        "NetworkInterface": {
            "AvailabilityZone": "us-west-2a",
            "Description": "IPv4 manual example",
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-044c2de2c4EXAMPLE"
                }
            ],
            "InterfaceType": "interface",
            "Ipv6Addresses": [],
            "MacAddress": "02:98:65:dd:18:47",
            "NetworkInterfaceId": "eni-02b80b4668EXAMPLE",
            "OwnerId": "123456789012",
            "PrivateIpAddress": "10.0.0.62",
```

```
            "PrivateIpAddresses": [
                {
                    "Primary": true,
                    "PrivateIpAddress": "10.0.0.62"
                }
            ],
            "Ipv4Prefixes": [
                {
                    "Ipv4Prefix": "10.0.0.208/28"
                }
            ],
            "RequesterId": "AIDAIV5AJI5LXF5XXDPCO",
            "RequesterManaged": false,
            "SourceDestCheck": true,
            "Status": "pending",
            "SubnetId": "subnet-047cfed18eEXAMPLE",
            "TagSet": [],
            "VpcId": "vpc-0e12f52b21EXAMPLE"
        }
    }
```

ネットワークインターフェース作成時に特定の IPv6 プレフィクスを割り当てるには

create-network-interface コマンドを使用し、--ipv6-prefixes にプレフィクスを設定します。AWS はこの範囲から IP アドレスを選択します。次の例では、プレフィクス CIDR は 2600:1f13:fc2:a700:1768::/80 です。

```
$ aws ec2 create-network-interface \
    --subnet-id subnet-047cfed18eEXAMPLE \
    --description "IPv6 manual example" \
    --ipv6-prefixes Ipv6Prefix=2600:1f13:fc2:a700:1768::/80
```

出力例

```
{
        "NetworkInterface": {
            "AvailabilityZone": "us-west-2a",
            "Description": "IPv6 automatic example",
            "Groups": [
                {
                    "GroupName": "default",
```

```
                    "GroupId": "sg-044c2de2c4EXAMPLE"
                }
            ],
            "InterfaceType": "interface",
            "Ipv6Addresses": [],
            "MacAddress": "02:bb:e4:31:fe:09",
            "NetworkInterfaceId": "eni-006edbcfa4EXAMPLE",
            "OwnerId": "123456789012",
            "PrivateIpAddress": "10.0.0.73",
            "PrivateIpAddresses": [
                {
                    "Primary": true,
                    "PrivateIpAddress": "10.0.0.73"
                }
            ],
            "Ipv6Prefixes": [
                {
                    "Ipv6Prefix": "2600:1f13:fc2:a700:1768::/80"
                }
            ],
            "RequesterId": "AIDAIV5AJI5LXF5XXDPCO",
            "RequesterManaged": false,
            "SourceDestCheck": true,
            "Status": "pending",
            "SubnetId": "subnet-047cfed18eEXAMPLE",
            "TagSet": [],
            "VpcId": "vpc-0e12f52b21EXAMPLE"
        }
    }
```


既存のネットワークインターフェイスにプレフィクスを割り当てる

プレフィクスを割り当てた後、(attach-network-interface)AWS CLIコマンドを使用して、ネットワークインターフェイスをインスタンスにアタッチします。プレフィクス付きのネットワークインターフェイス用にオペレーティングシステムを設定する必要があります。詳細については、「プレフィクス付きのネットワークインターフェイス用にオペレーティングシステムを設定する」を参照してください。

タスク

- 既存のネットワークインターフェイスに自動的にプレフィクスを割り当てる
- 既存のネットワークインターフェイスに特定のプレフィクスを割り当てる

既存のネットワークインターフェイスに自動的にプレフィクスを割り当てる

以下のいずれかの方法を使用して、自動プレフィクスを既存のネットワークインターフェイスに割り当てることができます。

Console

既存のネットワークインターフェイスに自動的にプレフィクスを割り当てるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. プレフィクスを割り当てるネットワークインターフェイスを選択し、[アクション],[プレフィクスの管理]を選択します。
4. IPv4 プレフィクスを自動的に割り当てるには、IPv4 プレフィクスの委任] で、[自動割り当て]を選択します。その後、IPv4 プレフィクスの数で、割り当てるプレフィクスの数を指定します。
5. IPv6 プレフィクスの自動割り当てるには、IPv6 プレフィクスの委任] で、[自動割り当て]を選択します。その後、IPv6 プレフィクスの数で、割り当てるプレフィクスの数を指定します。

   > **Note**
   >
   > [IPv6 プレフィクスの委任]は、IPv6 に対して選択したサブネットが有効になっている場合にのみ表示されます。

6. [Save] を選択します。

AWS CLI

assign-ipv6-addresses コマンドで IPv6 プレフィクスを割り当て、assign-private-ip-addresses コマンドで既存のネットワークインターフェイスに IPv4 プレフィクスを割り当てることができます。

既存のネットワークインターフェイスに自動的に IPv4 プレフィクスを割り当てるには

assign-private-ip-addresses コマンドを使用し、AWS が割り当てるプレフィクスの数を `--ipv4-prefix-count` に設定します。次の例では、AWS が 1 IPv4 プレフィクスを割り当てています。

```
$ aws ec2 assign-private-ip-addresses \
--network-interface-id eni-081fbb4095EXAMPLE \
--ipv4-prefix-count 1
```

出力例

```
{
    "NetworkInterfaceId": "eni-081fbb4095EXAMPLE",
    "AssignedIpv4Prefixes": [
        {
            "Ipv4Prefix": "10.0.0.176/28"
        }
    ]
}
```

既存のネットワークインターフェイスに自動的に IPv6 プレフィクスを割り当てるには

assign-ipv6-addresses コマンドを使用し、AWS が割り当てるプレフィクスの数を --ipv6-prefix-count に設定します。次の例では、AWS が 1 IPv6 プレフィクスを割り当てています。

```
$ aws ec2 assign-ipv6-addresses \
--network-interface-id eni-00d577338cEXAMPLE \
--ipv6-prefix-count 1
```

出力例

```
{
    "AssignedIpv6Prefixes": [
        "2600:1f13:fc2:a700:18bb::/80"
    ],
    "NetworkInterfaceId": "eni-00d577338cEXAMPLE"
}
```

既存のネットワークインターフェイスに特定のプレフィクスを割り当てる

次のいずれかの方法を使用して、既存のネットワークインターフェイスに特定のプレフィクスの割り当てを行うことができます。

Console

既存のネットワークインターフェイスに特定のプレフィクスを割り当てるには、

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. プレフィクスを割り当てるネットワークインターフェイスを選択し、[アクション],[プレフィクスの管理]を選択します。
4. 特定の IPv4 プレフィクスを割り当てるには、[IPv4 プレフィクスの委任] で、[カスタム]をを選択します。次に[新しいプレフィクスの追加] を選択し、使用するプレフィクスの入力を行います。
5. 特定の IPv6 プレフィクスを割り当てるには、[IPv6 プレフィクスの委任] で、[カスタム]を選択します。次に [新しいプレフィクスの追加] を選択し、使用するプレフィクスの入力を行います。

> **Note**
>
> [IPv6 プレフィクスの委任]は、IPv6 に対して選択したサブネットが有効になっている場合にのみ表示されます。

6. [Save] を選択します。

AWS CLI

既存のネットワークインターフェイスに特定の IPv4 プレフィクスを割り当てる

assign-private-ip-addresses コマンドを使用し、`--ipv4-prefixes` にプレフィクスを設定します。AWS はこの範囲から IPv4 アドレスを選択します。次の例では、プレフィクス CIDR は `10.0.0.208/28` です。

```
$ aws ec2 assign-private-ip-addresses \
--network-interface-id eni-081fbb4095EXAMPLE \
--ipv4-prefixes 10.0.0.208/28
```

出力例

```
{
    "NetworkInterfaceId": "eni-081fbb4095EXAMPLE",
```

```
    "AssignedIpv4Prefixes": [
        {
            "Ipv4Prefix": "10.0.0.208/28"
        }
    ]
}
```

既存のネットワークインターフェイスに特定の IPv6 プレフィクスを割り当てる

assign-ipv6-addresses コマンドを使用し、--ipv6-prefixes にプレフィクスを設定します。AWS はこの範囲から IPv6 アドレスを選択します。次の例では、プレフィクス CIDR は 2600:1f13:fc2:a700:18bb::/80 です。

```
$ aws ec2 assign-ipv6-addresses \
--network-interface-id eni-00d577338cEXAMPLE \
--ipv6-prefixes 2600:1f13:fc2:a700:18bb::/80
```

出力例

```
{
    "NetworkInterfaceId": "eni-00d577338cEXAMPLE",
    "AssignedIpv6Prefixes": [
        {
            "Ipv6Prefix": "2600:1f13:fc2:a700:18bb::/80"
        }
    ]
}
```

プレフィクス付きのネットワークインターフェイス用にオペレーティングシステムを設定する

Amazon Linux AMI には、AWS がインストールした ec2-net-utils という追加のスクリプトが含まれることがあります。これらのスクリプトはオプションで、ネットワークインターフェイスの設定を自動化します。これらは Amazon Linux でのみ使用できます。

Amazon Linux を使用していない場合は、Kubernetes 用 Container Network Interface (CNI) プラグイン、Docker を使用してコンテナを管理している場合は dockerd を使用することができます。

ネットワークインターフェイスに割り当てられたプレフィクスの表示

ネットワークインターフェイスに割り当てられたプレフィクスの表示は、次のいずれかの方法で行うことができます。

Console

既存のネットワークインターフェイスに割り当てられた自動プレフィクスを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. プレフィクスの表示するネットワークインターフェイスを選択し、[詳細]タブを選択します。
4. [IPv4 プレフィクスの委任]フィールドには割り当てられた IPv4 プレフィクスが一覧表示され、IPv6 プレフィクスの委任フィールドには、割り当てられた IPv6 プレフィクスが一覧表示されます。

AWS CLI

describe-network-interfaces AWS CLI コマンドを使用して、ネットワークインターフェイスに割り当てられたプレフィクスを表示することができます。

```
$ aws ec2 describe-network-interfaces
```

出力例

```
{
    "NetworkInterfaces": [
        {
            "AvailabilityZone": "us-west-2a",
            "Description": "IPv4 automatic example",
            "Groups": [
                {
                    "GroupName": "default",
                    "GroupId": "sg-044c2de2c4EXAMPLE"
                }
            ],
            "InterfaceType": "interface",
            "Ipv6Addresses": [],
            "MacAddress": "02:98:65:dd:18:47",
            "NetworkInterfaceId": "eni-02b80b4668EXAMPLE",
            "OwnerId": "123456789012",
            "PrivateIpAddress": "10.0.0.62",
            "PrivateIpAddresses": [
                {
```

```
                "Primary": true,
                "PrivateIpAddress": "10.0.0.62"
            }
        ],
        "Ipv4Prefixes": [
            {
                "Ipv4Prefix": "10.0.0.208/28"
            }
        ],
        "Ipv6Prefixes": [],
        "RequesterId": "AIDAIV5AJI5LXF5XXDPCO",
        "RequesterManaged": false,
        "SourceDestCheck": true,
        "Status": "available",
        "SubnetId": "subnet-05eef9fb78EXAMPLE",
        "TagSet": [],
        "VpcId": "vpc-0e12f52b2146bf252"
    },
    {
        "AvailabilityZone": "us-west-2a",
        "Description": "IPv6 automatic example",
        "Groups": [
            {
                "GroupName": "default",
                "GroupId": "sg-044c2de2c411c91b5"
            }
        ],
        "InterfaceType": "interface",
        "Ipv6Addresses": [],
        "MacAddress": "02:bb:e4:31:fe:09",
        "NetworkInterfaceId": "eni-006edbcfa4EXAMPLE",
        "OwnerId": "123456789012",
        "PrivateIpAddress": "10.0.0.73",
        "PrivateIpAddresses": [
            {
                "Primary": true,
                "PrivateIpAddress": "10.0.0.73"
            }
        ],
        "Ipv4Prefixes": [],
        "Ipv6Prefixes": [
            {
                "Ipv6Prefix": "2600:1f13:fc2:a700:1768::/80"
            }
```

```
            ],
            "RequesterId": "AIDAIV5AJI5LXF5XXDPCO",
            "RequesterManaged": false,
            "SourceDestCheck": true,
            "Status": "available",
            "SubnetId": "subnet-05eef9fb78EXAMPLE",
            "TagSet": [],
            "VpcId": "vpc-0e12f52b21EXAMPLE"
        }
    ]
}
```

ネットワークインターフェイスからプレフィクスを削除する

ネットワークインターフェイスからプレフィクスの削除は、次のいずれかの方法で行うことができます。

Console

ネットワークインターフェイスからプレフィクスを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークインターフェイス] を選択します。
3. プレフィクスの削除元となるネットワークインターフェイスを選択し、[アクション],[プレフィクスの管理]を選択します。
4. 次のいずれかを行います。

   - 割り当てられたプレフィクスをすべて削除するには、[IPv4 プレフィクスの委任]および [IPv6 プレフィクスの委任] で、[割り当てない]を選択します。
   - 割り当てられた特定のプレフィクスを削除するには、[IPv4 prefix delegation] (IPv4 プレフィックスの委任) または [IPv6 prefix delegation] (IPv6 プレフィックスの委任) で、[Custome] (カスタム) を選択し、削除するプレフィックスの横にある [Unassign] (割り当て解除) を選択します。

> **Note**
>
> [IPv6 プレフィクスの委任]は、IPv6 に対して選択したサブネットが有効になっている場合にのみ表示されます。

5. [Save] を選択します。

AWS CLI

unassign-ipv6-addresses コマンドで IPv6 プレフィクスを削除し、unassign-private-ip-addresses コマンドで既存のネットワークインターフェイスから IPv4 プレフィクスを削除することができます。

ネットワークインターフェイスから IPv4 プレフィクスを削除するには

unassign-private-ip-addresses コマンドを使用し、`--ipv4-prefix` に削除したいアドレスを設定します。

```
$ aws ec2 unassign-private-ip-addresses \
--network-interface-id eni-081fbb4095EXAMPLE \
--ipv4-prefixes 10.0.0.176/28
```

ネットワークインターフェイスから IPv6 プレフィクスを削除するには

unassign-ipv6-addresses コマンドを使用し、`--ipv6-prefix` に削除したいアドレスを設定します。

```
$ aws ec2 unassign-ipv6-addresses \
--network-interface-id eni-00d577338cEXAMPLE \
--ipv6-prefix 2600:1f13:fc2:a700:18bb::/80
```

Amazon EC2 インスタンスのネットワーク帯域幅

インスタンスの帯域幅の仕様は、インスタンスのインバウンドトラフィックとアウトバウンドトラフィックの両方に適用されます。例えば、インスタンスが最大 10 Gbps の帯域幅を指定した場合、インバウンドトラフィックには最大 10 Gbps、アウトバウンドトラフィックには最大 10 Gbps の帯域幅になります。EC2 インスタンスで使用できるネットワーク帯域幅は、次のようにいくつかの要因によって異なります。

マルチフロートラフィック

インスタンスで使用できる集約マルチフロートラフィックの帯域幅は、トラフィックの宛先によって異なります。

- リージョン内 – トラフィックは、インスタンスで使用可能な全ネットワーク帯域幅を利用することができます。
- 他のリージョンへの通信、インターネットゲートウェイ、Direct Connect、ローカルゲートウェイ (LGW) – トラフィックは、最低 32 個の vCPUs を搭載した現行世代のインスタンスで利用可能なネットワーク帯域幅の最大 50% を利用できます。32vCPUs未満の現世代のインスタンスの帯域幅は 5 Gbps に制限されています。

シングルフロートラフィック

インスタンスが同じクラスタープレイスメントグループにない場合、シングルフロートラフィックのベースライン帯域幅は 5 Gbps に制限されます。レイテンシーを減らし、シングルフロー帯域幅を増やすには、以下のいずれかをお試しください。

- クラスタープレイスメントグループを使用すると、同じプレイスメントグループ内のインスタンスで最大 10 Gbps の帯域幅を実現できます。
- または、任意の 2 つのエンドポイント間に複数のパスを設定することで、Multipath TCP (MPTCP) を使用して高帯域幅を実現できます。
- 同じサブネット内の対象インスタンスに ENA Express を設定して、それらのインスタンス間で最大 25 Gbps を実現します。

使用可能なインスタンスの帯域幅

使用可能なインスタンスのネットワーク帯域幅は、その vCPU の数によって異なります。例えば、`m5.8xlarge` インスタンスには 32 個の vCPU と 10 Gbps のネットワーク帯域幅があり、`m5.16xlarge` インスタンスには 64 個の vCPU と 20 Gbps のネットワーク帯域幅があります。ただし、インスタンスがこの帯域幅を達成できない場合があります。例えば、インスタンスレベルでネットワーク許容量 (1 秒あたりのパケット数や追跡される接続数など) を超えた場合などです。トラフィックが使用できる帯域幅の量は、vCPUsの数と宛先によって異なります。例えば、`m5.16xlarge` インスタンスは 64 vCPUs のため、リージョン内の別のインスタンスへのトラフィックは、使用可能な全帯域幅 (20 Gbps) を利用できます。ただし、異なるリージョンの別のインスタンスへのトラフィックは、使用可能な帯域幅の 50% (10 Gbps) しか利用できません。

通常、vCPU が 16 個以下のインスタンス (サイズ `4xlarge` 以下) の場合、指定の帯域幅に「最大」と文書化されています。例えば、最大 10 Gbpsなどです。これらのインスタンスには、ベースライン帯域幅があります。追加需要を満たすために、ネットワーク I/O クレジットメカニズムを使用して、ベースライン帯域幅を超えてバーストすることができます。インスタンスは、インスタンスのサイズに応じて、バースト帯域幅を限られた期間 (通常は 5～60 分) 使用できます。

インスタンスは、起動時にネットワーク I/O クレジットの最大数を受け取ります。インスタンスがネットワーク I/O クレジットを使い果たすと、ベースライン帯域幅に戻ります。実行中のインスタンスは、ベースライン帯域幅よりも少ないネットワーク帯域幅を使用するたびに、ネットワーク I/O クレジットを取得します。停止したインスタンスは、ネットワーク I/O クレジットを取得しません。バースト帯域幅は共有リソースであるため、インスタンスにクレジットが使用可能な場合でも、インスタンスのバーストはベストエフォートベースで行われます。

インバウンドトラフィックとアウトバウンドトラフィックには個別のネットワーク I/O クレジットバケットがあります。

ベースおよびバーストネットワークパフォーマンス

以下のドキュメントでは、すべてのインスタンスのネットワークパフォーマンスと、バースト帯域幅を使用できるインスタンスで利用可能なベースラインネットワーク帯域幅について説明します。

- 汎用インスタンス
- コンピュート最適化インスタンス
- メモリ最適化インスタンス
- ストレージ最適化インスタンス
- 高速コンピューティングインスタンス

AWS CLI を使用してネットワークパフォーマンスを表示するには

describe-instance-types AWS CLI コマンドを使用して、インスタンスタイプに関する情報を表示できます。次の例では、すべての C5 インスタンスのネットワークパフォーマンス情報を表示します。

```
aws ec2 describe-instance-types --filters "Name=instance-type,Values=c5.*"
 --query "InstanceTypes[].[InstanceType, NetworkInfo.NetworkPerformance,
 NetworkInfo.NetworkCards[0].BaselineBandwidthInGbps]" --output table
-----------------------------------------------
|           DescribeInstanceTypes             |
+--------------+--------------------+---------+
|  c5.4xlarge  |  Up to 10 Gigabit  |  5.0    |
|  c5.xlarge   |  Up to 10 Gigabit  |  1.25   |
|  c5.12xlarge |  12 Gigabit        |  12.0   |
|  c5.24xlarge |  25 Gigabit        |  25.0   |
|  c5.metal    |  25 Gigabit        |  25.0   |
|  c5.9xlarge  |  12 Gigabit        |  12.0   |
|  c5.2xlarge  |  Up to 10 Gigabit  |  2.5    |
```

```
|  c5.large    |  Up to 10 Gigabit  |  0.75 |
|  c5.18xlarge |  25 Gigabit        |  25.0 |
+--------------+--------------------+-------+
```

インスタンスの帯域幅をモニタリングします。

CloudWatch メトリクスを使用して、インスタンスのネットワーク帯域幅と送受信されたパケットをモニタリングできます。Elastic Network Adapter (ENA) ドライバーが提供するネットワークパフォーマンスメトリクスを使用して、トラフィックが Amazon EC2 がインスタンスレベルで定義するネットワーク許容量を超えるかモニタリングできます。

Amazon EC2 がインスタンスのメトリクスデータを CloudWatch に送信するときに、1 分単位か 5 分単位かを設定できます。CloudWatch インスタンスメトリクスでは、許容量を超え、パケットがドロップされたことがネットワークパフォーマンスメトリクスに表示されることがあります。これは、インスタンスのネットワークリソースに対する需要が短時間で急増し (マイクロバーストと呼ばれる)、CloudWatch メトリクスがこのようなマイクロ秒単位の急増を反映するのに十分な粒度を持っていない場合に発生します。

詳細はこちら

- インスタンスメトリクス
- ネットワークパフォーマンスメトリクス

Linux での拡張ネットワーキング

拡張ネットワーキングでは、シングルルート I/O 仮想化 (SR-IOV) を使用して、サポートされるインスタンスタイプにおける高性能ネットワーキング機能が提供されます。SR-IOV は、従来の仮想化ネットワークインターフェイスと比較し、I/O パフォーマンスが高く、CPU 利用率が低いデバイス仮想化の手法です。拡張ネットワーキングは、高い帯域幅、1 秒あたりのパケット (PPS) の高いパフォーマンス、常に低いインスタンス間レイテンシーを実現します。拡張ネットワーキングは追加料金なしで使用できます。

各インスタンスタイプでサポートされているネットワーク速度については、Amazon EC2インスタンスタイプを参照してください。

コンテンツ

- 拡張ネットワークのサポート
- インスタンスでの拡張ネットワーキングの有効化

- Linux インスタンスにおける Elastic Network Adapter (ENA) を使用した拡張ネットワーキングの有効化
- Linux インスタンスで ENA Express を使用してネットワークパフォーマンスを向上させる
- Linux インスタンスで Intel 82599 VF インターフェイスを使用した拡張ネットワーキングの有効化
- オペレーティングシステムの最適化
- EC2 インスタンスのネットワークパフォーマンスをモニタリングします。
- Elastic Network Adapter (ENA) のトラブルシューティング
- Linux ベースの Amazon EC2 インスタンスのネットワークレイテンシーを改善する

拡張ネットワークのサポート

現行世代のすべてのインスタンスタイプ (T2 インスタンスを除く) は、拡張ネットワークをサポートしています。

次のいずれかのメカニズムを使用して、拡張ネットワークを有効にすることができます。

Elastic Network Adapter (ENA)

Elastic Network Adapter (ENA) は、サポート対象のインスタンスタイプに対して最大 100 Gbps のネットワーク速度をサポートします。

すべての Nitro System ベースのインスタンスは ENA を使用してネットワークを強化しています。さらに、H1、G3、`m4.16xlarge`、P2、P3、P3dn、R4 の Xen インスタンスタイプは ENA をサポートしています。

Intel 82599 Virtual Function (VF) インターフェイス

Intel 82599 Virtual Function インターフェイスでは、サポートされているインスタンスタイプについて最大 10 Gbps のネットワーク速度がサポートされています。

インスタンスタイプ C3、C4、D2、I2、M4 (m4.16xlarge を除く)、R3 では、拡張ネットワーキングに Intel 82599 VF インターフェイスが使用されます。

インスタンスタイプ別の拡張ネットワークメカニズムの概要については、ネットワーキング機能とストレージ機能の概要を参照してください。

インスタンスでの拡張ネットワーキングの有効化

ご使用のインスタンスタイプで拡張ネットワーキングに Elastic Network Adapter がサポートされている場合、Linux インスタンスにおける Elastic Network Adapter (ENA) を使用した拡張ネットワーキングの有効化の手順に従います。

ご使用のインスタンスタイプで拡張ネットワーキングに Intel 82599 VF インターフェイスがサポートされている場合、Linux インスタンスで Intel 82599 VF インターフェイスを使用した拡張ネットワーキングの有効化の手順に従います。

Linux インスタンスにおける Elastic Network Adapter (ENA) を使用した拡張ネットワーキングの有効化

Amazon EC2 は、Elastic Network Adapter (ENA) を介してネットワーキング機能を提供します。拡張ネットワーキングを使用するには、必要な ENA モジュールをインストールし、ENA のサポートを有効にする必要があります。

コンテンツ

要件

ENA を使用した拡張ネットワーキングを準備するには、次のようにインスタンスをセットアップします。

- Nitro System ベースのインスタンスを起動します。
- サポートされているバージョンの Linux カーネルとサポートされているディストリビューションを使用してインスタンスを起動します。インスタンスに対して ENA 拡張ネットワーキングが自動的

に有効化されます。詳細については、ENA Linux Kernel Driver リリースノートを参照してください。

- インスタンスがインターネットに接続されていることを確認します。
- 選択した任意のコンピュータ、できればローカルのデスクトップまたはノートパソコンで、AWS Management Console から AWS CloudShell を使用するか、AWS CLI もしくは AWS Tools for Windows PowerShell をインストールし設定します。詳細については、Amazon EC2 へのアクセスもしくは AWS CloudShell ユーザーガイドを参照してください。拡張ネットワーキングは、Amazon EC2 コンソールから管理することはできません。
- 保持する必要がある重要なデータがインスタンスにある場合、インスタンスから AMI を作成してそのデータをバックアップする必要があります。`enaSupport` 属性を有効にするとともに、カーネルおよびカーネルモジュールを更新すると、互換性のないインスタンスがレンダリングされたり、オペレーティングシステムに接続できなくなったりする可能性があります。最近のバックアップがある場合は、これが発生してもデータは保持されます。

拡張ネットワーキングのパフォーマンス

以下のドキュメントには、ENA 拡張ネットワーキングをサポートするインスタンスタイプのネットワークパフォーマンスの概要が記載されています。

- 高速コンピューティングインスタンスのネットワークパフォーマンス
- コンピューティング最適化インスタンスのネットワークパフォーマンス
- 汎用インスタンスのネットワークパフォーマンス
- メモリ最適化インスタンスのネットワークパフォーマンス
- ストレージ最適化インスタンスのネットワークパフォーマンス

拡張ネットワーキングが有効化されているかどうかのテスト

次の AMI には必要な ENA モジュールが含まれており、ENA のサポートが有効になっています。

- AL2023
- Amazon Linux 2
- Amazon Linux AMI 2018.03 以降
- `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

> **Note**
>
> AWS Graviton ベースのインスタンスタイプには、linux-aws カーネル搭載の Ubuntu 18.04 以降が必要です

- Red Hat Enterprise Linux 7.4 以降
- SUSE Linux Enterprise Server 12 SP2 以降
- CentOS 7.4.1708 以降
- FreeBSD 11.1 以降
- Debian GNU/Linux 9 以降

拡張ネットワーキングが既に有効になっているかどうかをテストするには、ena モジュール がインスタンスにインストールされていることと、enaSupport 属性が設定されていることを確認します。インスタンスがこれら 2 つの条件を満たしている場合は、ethtool -i eth*n* コマンドによって、ネットワークインターフェイスで使用されているモジュールが表示されます。

カーネルモジュール (ena)

ena モジュールがインストールされたことを確認するには、以下の例に示されるように modinfo コマンドを使用します。

```
[ec2-user ~]$ modinfo ena
filename:    /lib/modules/4.14.33-59.37.amzn2.x86_64/kernel/drivers/amazon/net/ena/
ena.ko
version:  1.5.0g
license:  GPL
description: Elastic Network Adapter (ENA)
author:   Amazon.com, Inc. or its affiliates
srcversion:  692C7C68B8A9001CB3F31D0
alias:    pci:v00001D0Fd0000EC21sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd0000EC20sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd00001EC2sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd00000EC2sv*sd*bc*sc*i*
depends:
retpoline:   Y
intree:   Y
name:     ena
...
```

上の Amazon Linux のケースでは、ena モジュールはインストールされています。

```
ubuntu:~$ modinfo ena
ERROR: modinfo: could not find module ena
```

上の Ubuntu インスタンスでは、モジュールはインストールされていないため、まずインストールする必要があります。詳細については、Ubuntu での拡張ネットワーキングの有効化を参照してください。

インスタンス属性 (enaSupport)

インスタンスに拡張ネットワーキングの enaSupport 属性が設定されているかどうかを確認するには、次のいずれかのコマンドを使用します。属性が設定されている場合、レスポンスは true です。

- describe-instances (AWS CLI/AWS CloudShell)

```
aws ec2 describe-instances --instance-ids instance_id --query
 "Reservations[].Instances[].EnaSupport"
```

- Get-EC2Instance (Windows PowerShell 用のツール)

```
(Get-EC2Instance -InstanceId instance-id).Instances.EnaSupport
```

イメージ属性 (enaSupport)

AMI に拡張ネットワーキングの enaSupport 属性が設定されているかどうかを確認するには、次のいずれかのコマンドを使用します。属性が設定されている場合、レスポンスは true です。

- describe-images (AWS CLI/AWS CloudShell)

```
aws ec2 describe-images --image-id ami_id --query "Images[].EnaSupport"
```

- Get-EC2Image (Windows PowerShell 用のツール)

```
(Get-EC2Image -ImageId ami_id).EnaSupport
```

ネットワークインターフェイスドライバー

次のコマンドを使用して、ena モジュールが特定のインターフェイスで使用されていることを確認し、確認するインターフェイス名に置き換えます。単一のインターフェイス (デフォルト) を使用している場合は、eth0 です。オペレーティングシステムで予測可能なネットワーク名がサポートされている場合は、ens5 のような名前にすることができます。

次の例で、リストされているドライバーは vif であるため、ena モジュールはロードされていません。

```
[ec2-user ~]$ ethtool -i eth0
driver: vif
version:
firmware-version:
bus-info: vif-0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

この例では、ena モジュールがロードされており、最小推奨バージョンです。このインスタンスでは、拡張ネットワーキングが適切に設定されています。

```
[ec2-user ~]$ ethtool -i eth0
driver: ena
version: 1.5.0g
firmware-version:
expansion-rom-version:
bus-info: 0000:00:05.0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

Amazon Linux AMI での拡張ネットワーキングの有効化

Amazon Linux 2 および Amazon Linux AMI の最新バージョンには、ENA がインストールされた拡張ネットワーキングに必要なモジュールが含まれており、ENA のサポートが有効になっています。したがって、サポートされるインスタンスタイプで HVM バージョンの Amazon Linux を使用してインスタンスを起動した場合、拡張ネットワーキングは既にインスタンスで有効になっています。詳細については、拡張ネットワーキングが有効化されているかどうかのテストを参照してください。

以前の Amazon Linux AMI を使用してインスタンスを起動し、まだ拡張ネットワーキングが有効になっていない場合、拡張ネットワーキングを有効にするには次の手順を実行します。

Amazon Linux AMI で拡張ネットワーキングを有効化するには

1. インスタンスに接続します。
2. インスタンスから、次のコマンドを実行して、ena を含む最新のカーネルとカーネルモジュールでインスタンスを更新します。

   ```
   [ec2-user ~]$ sudo yum update
   ```

3. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを再起動します。reboot-instances (AWS CLI)、Restart-EC2Instance (AWS Tools for Windows PowerShell)。
4. インスタンスに再接続し、enaの modinfo ena コマンドを使用して、拡張ネットワーキングが有効化されているかどうかのテスト モジュールがインストールされ、最小推奨バージョンであることを確認します。
5. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: Amazon Linux AMI で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。
6. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

   - modify-instance-attribute (AWS CLI)

     ```
     aws ec2 modify-instance-attribute --instance-id instance_id --ena-support
     ```

   - Edit-EC2InstanceAttribute (Windows PowerShell 用のツール)

     ```
     Edit-EC2InstanceAttribute -InstanceId instance-id -EnaSupport $true
     ```

7. (オプション) Amazon EBS-backed Linux AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング `enaSupport` 属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。
8. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。
9. インスタンスに接続し、拡張ネットワーキングが有効化されているかどうかのテスト の ethtool -i eth*n* コマンドを使用して、ena モジュールがインストールされ、ネットワークインターフェイスにロードされていることを確認します。

   拡張ネットワーキングを有効にした後にインスタンスに接続できない場合、Elastic Network Adapter (ENA) のトラブルシューティングを参照してください。

Amazon Linux AMI で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。instance store-backed Linux AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

- register-image (AWS CLI)

```
aws ec2 register-image --ena-support ...
```

- Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -EnaSupport $true ...
```

Ubuntu での拡張ネットワーキングの有効化

最新の Ubuntu HVM AMI には、ENA がインストールされた拡張ネットワーキングに必要なモジュールが含まれており、ENA のサポートが有効になっています。したがって、サポートされるインスタンスタイプで最新の Ubuntu HVM AMI を使用してインスタンスを起動した場合、拡張ネットワーキングは既にインスタンスで有効になっています。詳細については、拡張ネットワーキングが有効化されているかどうかのテストを参照してください。

以前の AMI を使用してインスタンスを起動した場合、まだ拡張ネットワーキングが有効になっていなければ、linux-aws カーネルパッケージをインストールして最新の拡張ネットワーキングドライバーを取得して、必要な属性を更新できます。

linux-aws カーネルパッケージをインストールするには (Ubuntu 16.04 以降)

Ubuntu 16.04 および 18.04 には、Ubuntu カスタムカーネル (linux-aws カーネルパッケージ) が付属しています。別のカーネルを使用するには、AWS Support にお問い合わせください。

linux-aws カーネルパッケージをインストールするには (Ubuntu Trusty 14.04)

1. インスタンスに接続します。
2. パッケージキャッシュおよびパッケージを更新します。

```
ubuntu:~$ sudo apt-get update && sudo apt-get upgrade -y linux-aws
```

> **Important**
>
> 更新プロセス中に grub をインストールするよう求められた場合は、/dev/xvda のインストール先として grub を使用し、現在のバージョンの /boot/grub/menu.lst を保持することを選択します。

3. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: Ubuntu で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。
4. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

   - modify-instance-attribute (AWS CLI)

     ```
     aws ec2 modify-instance-attribute --instance-id instance_id --ena-support
     ```

   - Edit-EC2InstanceAttribute (Windows PowerShell 用のツール)

```
Edit-EC2InstanceAttribute -InstanceId instance-id -EnaSupport $true
```

5. (オプション) Amazon EBS-backed Linux AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング enaSupport 属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。
6. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

Ubuntu で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。instance store-backed Linux AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

- register-image (AWS CLI)

```
aws ec2 register-image --ena-support ...
```

- Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -EnaSupport $true ...
```

Linux での拡張ネットワーキングの有効化

Red Hat Enterprise Linux、SUSE Linux Enterprise Server、および CentOS 用の最新の AMI には、ENA を使用した拡張ネットワーキングに必要なモジュールが含まれており、ENA のサポートが有効になっています。したがって、サポートされるインスタンスタイプで最新の AMI を使用してインスタンスを起動した場合、拡張ネットワーキングは既にインスタンスで有効になっています。詳細については、拡張ネットワーキングが有効化されているかどうかのテストを参照してください。

次の手順では、Amazon Linux AMI または Ubuntu 以外の Linux ディストリビューションで拡張ネットワーキングを有効にするための一般的なステップを説明します。コマンドの詳細な構文、ファイ

ルの場所、パッケージやツールのサポートなどの詳細については、使用する Linux ディストリビューションのドキュメントを参照してください。

Linux で拡張ネットワーキングを有効化するには

1. インスタンスに接続します。
2. https://github.com/amzn/amzn-drivers の GitHub からインスタンスで ena モジュールのソースコードのクローンを作成します。(SUSE Linux Enterprise Server 12 SP2 以降には、デフォルトで ENA 2.02 が含まれているため、ENA ドライバーをダウンロードしてコンパイルする必要はありません。SUSE Linux Enterprise Server 12 SP2 以降では、必要なドライバーバージョンを標準カーネルに追加するためリクエストを申請する必要があります。)

   ```
   git clone https://github.com/amzn/amzn-drivers
   ```

3. インスタンスで ena モジュールをコンパイルし、インストールします。これらの手順は Linux ディストリビューションによって異なります。Red Hat Enterprise Linux でのモジュールのコンパイルの詳細については、AWS ナレッジセンターの記事を参照してください。
4. sudo depmod コマンドを実行して、モジュールの依存関係を更新します。
5. 起動時に新しいモジュールがロードされるように、インスタンスの `initramfs` を更新します。例えば、ディストリビューションで dracut がサポートされる場合、次のコマンドを使用できます。

   ```
   dracut -f -v
   ```

6. システムがデフォルトで予測可能なネットワークインターフェイス名を使用するかどうかを確認します。systemd または udev のバージョン 197 以上を使用するシステムの場合、イーサネットデバイスの名前を変更でき、単一ネットワークインターフェイスの名前が `eth0` になることは保証されません。この動作は、インスタンスに接続する際に問題の原因となる可能性があります。詳細と他の設定オプションについては、freedesktop.org ウェブサイトでPredictable Network Interface Names/を参照してください。

   a. 次のコマンドを使用して、RPM ベースのシステムで systemd または udev のバージョンを確認できます。

      ```
      rpm -qa | grep -e '^systemd-[0-9]\+\|^udev-[0-9]\+'
      systemd-208-11.el7_0.2.x86_64
      ```

上記の Red Hat Enterprise Linux 7 の例では、systemd のバージョンは 208 であるため、予測可能なネットワークインターフェイス名は無効になっている必要があります。

b. `net.ifnames=0` オプションを `GRUB_CMDLINE_LINUX` の `/etc/default/grub` 行に追加することによって、予測可能なネットワークインターフェイス名を無効にします。

```
sudo sed -i '/^GRUB\_CMDLINE\_LINUX/s/\"$/\ net\.ifnames\=0\"/' /etc/default/grub
```

c. grub の設定ファイルを再ビルドします。

```
sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

7. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: Linux で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。

8. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの `enaSupport` 属性を有効化します。

   - modify-instance-attribute (AWS CLI)

   ```
   aws ec2 modify-instance-attribute --instance-id instance_id --ena-support
   ```

   - Edit-EC2InstanceAttribute (Windows PowerShell 用のツール)

   ```
   Edit-EC2InstanceAttribute -InstanceId instance-id -EnaSupport $true
   ```

9. (オプション) Amazon EBS-backed Linux AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング `enaSupport` 属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。

> **Important**
>
> インスタンスオペレーティングシステムに `/etc/udev/rules.d/70-persistent-net.rules` が含まれている場合には、AMI を作成する前にそれを削除する必要があります。このファイルには、元のインスタンスのイーサネットアダプターの MAC アドレスが保存されています。別のインスタンスがこのファイルを使用して起動した場合、オペレーティングシステムがそのデバイスを検出できなくなり、`eth0` が失敗して、起動に関する問題が発生することがあります。このファイルは次の起動サイクルで再び生成され、AMI から起動されるインスタンスごとに独自のバージョンが作成されます。

10. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。
11. (オプション) インスタンスに接続し、モジュールがインストールされていることを確認します。

    拡張ネットワーキングを有効にした後にインスタンスに接続できない場合、Elastic Network Adapter (ENA) のトラブルシューティングを参照してください。

Linux で拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。instance store-backed Linux AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

- register-image (AWS CLI)

```
aws ec2 register-image --ena-support ...
```

- Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -EnaSupport ...
```

DKMS を使用した Ubuntu での拡張ネットワーキングの有効化

この方法は、テストおよびフィードバックのみを目的としています。本番稼働用デプロイによる使用を目的としていません。本番稼働デプロイについては、Ubuntu での拡張ネットワーキングの有効化を参照してください。

> **Important**
>
> DKMS を使用すると、サブスクリプションのサポート契約が無効になります。本稼働環境では使用しないでください。

Ubuntu で ENA を使用した拡張ネットワーキングを有効にするには (EBS-backed インスタンス)

1. Ubuntu での拡張ネットワーキングの有効化 のステップ 1 および 2 を行います。
2. build-essential パッケージをインストールしてカーネルモジュールと dkms パッケージをコンパイルし、カーネルが更新されるたびに ena モジュールが再構築されるようにします。

   ```
   ubuntu:~$ sudo apt-get install -y build-essential dkms
   ```

3. の GitHub からインスタンスで ena モジュールのソースのクローンを作成します。。https://github.com/amzn/amzn-drivers

   ```
   ubuntu:~$ git clone https://github.com/amzn/amzn-drivers
   ```

4. amzn-drivers パッケージを /usr/src/ ディレクトリに移動して、カーネルの更新のたびに DKMS がこのパッケージを見つけて構築できるようにします。ソースコードのバージョン番号 (現在のバージョン番号はリリースノートにあります) をディレクトリ名に付加します。例えば、バージョン 1.0.0 は以下のようになります。

   ```
   ubuntu:~$ sudo mv amzn-drivers /usr/src/amzn-drivers-1.0.0
   ```

5. 以下の値を使用して DKMS 設定ファイルを作成し、ena のバージョンに置き換えます。

   ファイルを作成します。

   ```
   ubuntu:~$ sudo touch /usr/src/amzn-drivers-1.0.0/dkms.conf
   ```

   ファイルを編集し、次の値を追加します。

```
ubuntu:~$ sudo vim /usr/src/amzn-drivers-1.0.0/dkms.conf
PACKAGE_NAME="ena"
PACKAGE_VERSION="1.0.0"
CLEAN="make -C kernel/linux/ena clean"
MAKE="make -C kernel/linux/ena/ BUILD_KERNEL=${kernelver}"
BUILT_MODULE_NAME[0]="ena"
BUILT_MODULE_LOCATION="kernel/linux/ena"
DEST_MODULE_LOCATION[0]="/updates"
DEST_MODULE_NAME[0]="ena"
AUTOINSTALL="yes"
```

6. DKMS を使用して、インスタンスで ena モジュールを追加、構築、インストールします。

   DKMS にモジュールを追加します。

   ```
   ubuntu:~$ sudo dkms add -m amzn-drivers -v 1.0.0
   ```

   dkms コマンドを使用してモジュールを構築します。

   ```
   ubuntu:~$ sudo dkms build -m amzn-drivers -v 1.0.0
   ```

   dkms を使用してモジュールをインストールします。

   ```
   ubuntu:~$ sudo dkms install -m amzn-drivers -v 1.0.0
   ```

7. 起動時に正しいモジュールがロードされるように、initramfs を再ビルドします。

   ```
   ubuntu:~$ sudo update-initramfs -u -k all
   ```

8. 拡張ネットワーキングが有効化されているかどうかのテスト から modinfo ena コマンドを使用して、ena モジュールがインストールされていることを確認します。

   ```
   ubuntu:~$ modinfo ena
   filename:    /lib/modules/3.13.0-74-generic/updates/dkms/ena.ko
   version:  1.0.0
   license:  GPL
   description: Elastic Network Adapter (ENA)
   author:   Amazon.com, Inc. or its affiliates
   srcversion:  9693C876C54CA64AE48F0CA
   alias:    pci:v00001D0Fd0000EC21sv*sd*bc*sc*i*
   ```

```
alias:    pci:v00001D0Fd0000EC20sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd00001EC2sv*sd*bc*sc*i*
alias:    pci:v00001D0Fd00000EC2sv*sd*bc*sc*i*
depends:
vermagic:    3.13.0-74-generic SMP mod_unload modversions
parm:     debug:Debug level (0=none,...,16=all) (int)
parm:     push_mode:Descriptor / header push mode (0=automatic,1=disable,3=enable)
     0 - Automatically choose according to device capability (default)
     1 - Don't push anything to device memory
     3 - Push descriptors and header buffer to device memory (int)
parm:     enable_wd:Enable keepalive watchdog (0=disable,1=enable,default=1) (int)
parm:     enable_missing_tx_detection:Enable missing Tx completions. (default=1)
 (int)
parm:     numa_node_override_array:Numa node override map
 (array of int)
parm:     numa_node_override:Enable/Disable numa node override (0=disable)
 (int)
```

9. Ubuntu での拡張ネットワーキングの有効化 のステップ 3 に進みます。

ドライバーのリリースノート

Linux ENA ドライバーのバージョンについては、ENA Linux カーネルドライバーのリリースノートを参照してください 。

トラブルシューティング

トラブルシューティング情報については、Elastic Network Adapter (ENA) のトラブルシューティングを参照してください。

Linux インスタンスで ENA Express を使用してネットワークパフォーマンスを向上させる

ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD は、動的ルーティングを使用してスループットを向上させ、テールレイテンシーを最小限に抑える高性能なネットワークトランスポートプロトコルです。ENA Express を使用すると、同じサブネット内の 2 つの EC2 インスタンス間で通信できます。

ENA Express の利点

- サブネット内で 1 つのフローで使用できる集約インスタンスの制限までの最大帯域幅を、5 Gbps から 25 Gbps に拡大します。
- 特にネットワーク負荷が高い期間に、EC2 インスタンス間のネットワークトラフィックのテールレイテンシーを短縮します。
- 混雑したネットワークパスを検出して回避します。
- 受信側でのパケットの並べ替えや、必要とされるほとんどの再送信など、一部のタスクをネットワーク層で直接処理します。これにより、アプリケーション層が解放され、他の作業に充てることができるようになります。

> **Note**
>
> アプリケーションが 1 秒間に大量のパケットを送受信し、ほとんどの場合、特にネットワークに輻輳がない時間帯にレイテンシーを最適化する必要がある場合は、拡張ネットワーキング の方がネットワークに適している場合があります。

ネットワークトラフィックが少ない時間帯に、パケットが ENA Express を使用すると、パケットのレイテンシーがわずかに増加することがあります (数十マイクロ秒)。このような場合、特定のネットワークパフォーマンス特性を優先するアプリケーションには、次のような ENA Express の利点があります。

- プロセスは、集約インスタンスの制限までの同じサブネット内におけるシングルフローの最大帯域幅を 5 Gbps から 25 Gbps に拡大するという利点を得られます。たとえば、特定のインスタンスタイプが最大 12.5 Gbps までサポートする場合、シングルフローの帯域幅も 12.5 Gbps までに制限されます。
- 実行時間が長いプロセスでは、ネットワークが混雑している間のテールレイテンシーが減少するはずです。
- プロセスには、ネットワークの応答時間をよりスムーズに、より標準的にディストリビューションできるという利点があります。

前提条件

ENA Express が効果的に動作するように、インスタンスの次の設定を更新してください。

- インスタンスでジャンボフレームを使用している場合は、次のコマンドを実行して最大送信単位 (MTU) を 8900 に設定します。

```
[ec2-user ~]$ sudo ip link set dev eth0 mtu 8900
```

- 受信側 (Rx) のリングサイズを次のように大きくします。

```
[ec2-user ~]$ ethtool -G device rx 8192
```

- ENA Expressの帯域幅を最大化するには、TCP キュー制限を次のように設定します。

  1. TCP の小規模なキューの制限を 1 MB 以上に設定します。これにより、ソケット上で送信キューのデータが増えます。

```
sudo sh -c 'echo 1048576 > /proc/sys/net/ipv4/tcp_limit_output_bytes'
```

  2. お使いの Linux ディストリビューションで eth デバイスのバイトキュー制限が有効になっている場合は、それを無効にしてください。これにより、デバイスキューの送信待ちのデータが増加します。

```
sudo sh -c 'for txq in /sys/class/net/eth0/queues/tx-*; do echo max > ${txq}/
byte_queue_limits/limit_min; done'
```

> **Note**
>
> Amazon Linux ディストリビューションの ENA ドライバーは、デフォルトでバイトキュー制限を無効にします。

ENA Express の仕組み

ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。各ネットワークフローのパケットをさまざまな AWS ネットワークパスに分散し、輻輳の兆候を検出すると配信を動的に調整します。また、受信側でのパケットの並べ替えも管理します。

ENA Express がネットワークトラフィックを意図したとおりに管理できるようにするには、送受信インスタンスと受信側インスタンス間の通信が次の要件をすべて満たしている必要があります。

- 送信側と受信側の両方のインスタンスタイプがサポートされています。詳細については「ENA Express でサポートされるインスタンスタイプ」の表を参照してください。

- 送信側と受信側の両方のインスタンスに ENA Express が設定されている必要があります。設定に違いがあると、トラフィックがデフォルトで標準の ENA 送信になる状況が発生する可能性があります。発生し得る状況を次のシナリオで説明します。

  シナリオ: 設定の違い

インスタンス	ENA Express が有効になっている	UDP は ENA Express を使用する
インスタンス 1	はい	はい
インスタンス 2	はい	いいえ

  この場合、2 つのインスタンスが ENA Express を有効にするので、両方のインスタンス間の TCP トラフィックで、ENA Express を使用できます。ただし、一方のインスタンスは UDP トラフィックに ENA Express を使用しないため、これら 2 つのインスタンス間の UDP 経由の通信には標準の ENA 送信が使用されます。
- 送信側と受信側のインスタンスは同じサブネットで実行する必要があります。
- インスタンス間のネットワークパスには、ミドルウェアボックスを含めないようにしてください。ENA Express は現在、ミドルウェアボックスをサポートしていません。
- 帯域幅を最大限に活用するには、ドライバーバージョン 2.2.9 以降を使用してください。
- メトリクスを生成するには、ドライバーバージョン 2.8 以降を使用してください。

いずれかの要件が満たされていない場合、インスタンスは標準の TCP/UDP プロトコルを使用して通信しますが、SRD は使用しません。

インスタンスのネットワークドライバーが最適なパフォーマンスを発揮できるように構成するには、ENA ドライバーの推奨ベストプラクティスを確認してください。これらのベストプラクティスは ENA Express にも当てはまります。詳細については、GitHub ウェブサイトの「ENA Linux Driver Best Practices and Performance Optimization Guide」(ENA Linux ドライバーのベストプラクティスとパフォーマンス最適化ガイド) を参照してください。

> **Note**
>
> Amazon EC2 では、インスタンスとそれにアタッチされたネットワークインターフェイスとの関係をアタッチメントと呼びます。ENA Express の設定がアタッチメントに適用されま

す。ネットワークインターフェースがインスタンスからデタッチされると、アタッチメントは存在しなくなり、そのアタッチメントに適用されていた ENA Express 設定は無効になります。ネットワークインターフェースが残っていても、インスタンスが終了した場合、同様になります。

ENA Express でサポートされるインスタンスタイプ

次の表には、ENA Express をサポートするインスタンスタイプが含まれています。

インスタンスタイプ	アーキテクチャ
汎用	
m6a.48xlarge	x86_64
m6a.metal	x86_64
m6i.8xlarge	x86_64
m6i.12xlarge	x86_64
m6i.16xlarge	x86_64
m6i.24xlarge	x86_64
m6i.32xlarge	x86_64
m6i.metal	x86_64
m6id.8xlarge	x86_64
m6id.12xlarge	x86_64
m6id.16xlarge	x86_64
m6id.24xlarge	x86_64
m6id.32xlarge	x86_64
m6id.metal	x86_64

インスタンスタイプ	アーキテクチャ
m7g.12xlarge	arm64
m7g.16xlarge	arm64
m7g.metal	arm64
m7gd.12xlarge	arm64
m7gd.16xlarge	arm64
m7gd.metal	arm64
m7i.48xlarge	x86_64
m7i.metal-48xl	x86_64
コンピューティングの最適化	
c6a.48xlarge	x86_64
c6a.metal	x86_64
c6gn.16xlarge	arm64
c6i.8xlarge	x86_64
c6i.12xlarge	x86_64
c6i.16xlarge	x86_64
c6i.24xlarge	x86_64
c6i.32xlarge	x86_64
c6i.metal	x86_64
c6id.8xlarge	x86_64
c6id.12xlarge	x86_64

インスタンスタイプ	アーキテクチャ
c6id.16xlarge	x86_64
c6id.24xlarge	x86_64
c6id.32xlarge	x86_64
c6id.metal	x86_64
c7g.12xlarge	arm64
c7g.16xlarge	arm64
c7g.metal	arm64
c7gd.12xlarge	arm64
c7gd.16xlarge	arm64
c7gd.metal	arm64
c7i.48xlarge	x86_64
c7i.metal-48xl	x86_64
メモリ最適化	
r6a.48xlarge	x86_64
r6a.metal	x86_64
r6i.8xlarge	x86_64
r6i.12xlarge	x86_64
r6i.16xlarge	x86_64
r6i.24xlarge	x86_64
r6i.32xlarge	x86_64

インスタンスタイプ	アーキテクチャ
r6i.metal	x86_64
r6id.8xlarge	x86_64
r6id.12xlarge	x86_64
r6id.16xlarge	x86_64
r6id.24xlarge	x86_64
r6id.32xlarge	x86_64
r6id.metal	x86_64
r7g.12xlarge	arm64
r7g.16xlarge	arm64
r7g.metal	arm64
r7gd.12xlarge	arm64
r7gd.16xlarge	arm64
r7gd.metal	arm64
r7i.48xlarge	x86_64
r7i.metal-48xl	x86_64
x2idn.16xlarge	x86_64
x2idn.24xlarge	x86_64
x2idn.32xlarge	x86_64
x2idn.metal	x86_64
x2iedn.8xlarge	x86_64

インスタンスタイプ	アーキテクチャ
x2iedn.16xlarge	x86_64
x2iedn.24xlarge	x86_64
x2iedn.32xlarge	x86_64
x2iedn.metal	x86_64
ストレージの最適化	
i4g.4xlarge	arm64
i4g.8xlarge	arm64
i4g.16xlarge	arm64
i4i.8xlarge	x86_64
i4i.12xlarge	x86_64
i4i.16xlarge	x86_64
i4i.24xlarge	x86_64
i4i.32xlarge	x86_64
i4i.metal	x86_64
im4gn.4xlarge	arm64
im4gn.8xlarge	arm64
im4gn.16xlarge	arm64

ENA Express の設定を一覧化して表示する

このセクションでは、AWS Management Console または AWS CLI から ENA Express 情報を一覧化して表示する方法について説明します。詳細については、使用する方法に一致するタブを選択してください。

Console

このタブでは、現在の ENA Express 設定に関する情報を確認する方法と、AWS Management Console でインスタンスタイプのサポートを確認する方法について説明します。

インスタンスタイプのサポートを表示する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで、[Instance types] (インスタンスタイプ) を選択します。
3. インスタンスタイプを選択すると、そのインスタンスの詳細が表示されます。[Instance type] (インスタンスタイプ) のリンクを選択して詳細ページを開くか、リストの左側のチェックボックスを選択してページ下部の詳細ペインで詳細を表示できます。
4. [Networking] タブまたは詳細ページのそのセクションの [ENA Express Support] (ENA Express サポート) に、インスタンスタイプがこの機能をサポートしているかどうかを示す値が true または false と表示されます。

ネットワークインターフェースリストから設定を表示する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側のナビゲーションペインで、[Network Interfaces] (ネットワークインターフェース) を選択します。
3. ネットワークインターフェイスを選択すると、そのインスタンスの詳細が表示されます。[Network interface ID] (ネットワークインターフェース ID) リンクを選択して詳細ページを開くか、リストの左側のチェックボックスを選択してページ下部の詳細ペインで詳細を表示できます。
4. [Details] (詳細) タブまたは詳細ページの [Network interface attachment] (ネットワークインターフェース接続) セクションで、[ENA Express] と [ENA Express UDP] の設定を確認します。

インスタンスから設定を表示する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックします。
3. インスタンスを選択すると、そのインスタンスの詳細が表示されます。[Instance ID] (インスタンス ID) リンクを選択して詳細ページを開くことも、リストの左側のチェックボックスを選択してページ下部の詳細ペインに詳細を表示することもできます。

4. [Networking] (ネットワーキング) タブの [Network interfaces] (ネットワークインターフェース) セクションを右にスクロールして [ENA Express] と [ENA Express UDP] の設定を確認します。

AWS CLI

このタブでは、現在の ENA Express 設定に関する情報を確認する方法と、AWS CLI でインスタンスタイプのサポートを確認する方法について説明します。

インスタンスタイプを記述する

特定のインスタンスタイプのインスタンスタイプ設定については、AWS CLI で describe-instance-types コマンドを実行し、インスタンスタイプを次のように置き換えてください。

```
[ec2-user ~]$ aws ec2 describe-instance-types --instance-types m6i.metal
{
"InstanceTypes": [
{
 "InstanceType": "m6i.metal",
 "CurrentGeneration": true,
 ...
 },
 "NetworkInfo": {
  ...
  "EnaSrdSupported": true
 },
 ...
}
]
}
```

ネットワークインターフェース記述する

インスタンスタイプの設定については、次のように AWS CLI で describe-network-interfaces コマンドを実行してください。

```
[ec2-user ~]$ aws ec2 describe-network-interfaces
{
"NetworkInterfaces": [
{
 "Association": {
```

```
  ....IPs, DNS...
 },
 "Attachment": {
  "AttachTime": "2022-11-17T09:04:28+00:00",
  "AttachmentId": "eni-attach-0ab1c23456d78e9f0",
  "DeleteOnTermination": true,
  "DeviceIndex": 0,
  "NetworkCardIndex": 0,
  "InstanceId": "i-0abcd123e456fabcd",
  "InstanceOwnerId": "111122223333",
  "Status": "attached",
  "EnaSrdSpecification": {
   "EnaSrdEnabled": true,
   "EnaSrdUdpSpecification": {
    "EnaSrdUdpEnabled": true
   }
  }
 },
 ...
 "NetworkInterfaceId": "eni-0d1234e5f6a78901b",
 "OwnerId": "111122223333",
 ...
}
]
}
```

PowerShell

このタブでは、現在の ENA Express 設定に関する情報を確認する方法と、PowerShell を使用してインスタンスタイプのサポートを確認する方法について説明します。

インスタンスタイプを記述する

特定のインスタンスタイプのインスタンスタイプ設定については、Tools for PowerShell で Get-EC2InstanceType Cmdlet コマンドを実行し、インスタンスタイプを次のように置き換えてください。

```
PS C:\> Get-EC2InstanceType -InstanceType m6i.metal | `
Select-Object `
    InstanceType,
    CurrentGeneration,
    @{Name = 'EnaSrdSupported'; Expression = { $_.NetworkInfo.EnaSrdSupported } } |
 `
```

```
Format-List

InstanceType      : m6i.metal
CurrentGeneration : True
EnaSrdSupported   : True
```

ENA Express が有効になっている場合、値 `True` が返されます。

ネットワークインターフェースを記述する

ネットワークインターフェイスの ENA Express 設定の詳細については、Tools for PowerShell で Get-EC2NetworkInterface Cmdlet を次のように実行してください。

```
PS C:\> Get-EC2NetworkInterface -NetworkInterfaceId eni-0d1234e5f6a78901b | `
Select-Object `
    Association,
    NetworkInterfaceId,
    OwnerId,
    @{Name = 'AttachTime'; Expression = { $_.Attachment.AttachTime } },
    @{Name = 'AttachmentId'; Expression = { $_.Attachment.AttachmentId } },
    @{Name = 'DeleteOnTermination'; Expression =
 { $_.Attachment.DeleteOnTermination } },
    @{Name = 'NetworkCardIndex'; Expression = { $_.Attachment.NetworkCardIndex } },
    @{Name = 'InstanceId'; Expression = { $_.Attachment.InstanceId } },
    @{Name = 'InstanceOwnerId'; Expression = { $_.Attachment.InstanceOwnerId } },
    @{Name = 'Status'; Expression = { $_.Attachment.Status } },
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled } },
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled } }

Association         :
NetworkInterfaceId  : eni-0d1234e5f6a78901b
OwnerId             : 111122223333
AttachTime          : 6/11/2022 1:13:11 AM
AttachmentId        : eni-attach-0d1234e5f6a78901b
DeleteOnTermination : True
NetworkCardIndex    : 0
InstanceId          : i-0d1234e5f6a78901b
InstanceOwnerId     : 111122223333
Status              : attached
EnaSrdEnabled       : True
EnaSrdUdpEnabled    : False
```

ENA Express の設定を行う

ENA Express は、対応する EC2 インスタンスタイプに対して、追加のソフトウェアをインストールすることなく設定できます。このセクションでは、AWS Management Console または AWS CLI から ENA Express を設定する方法について説明します。詳細については、使用する方法に一致するタブを選択してください。

Console

このタブでは、インスタンスにアタッチされているネットワークインターフェースの ENA Express 設定を管理する方法について説明します。

ネットワークインターフェースリストから ENA Express を管理する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側のナビゲーションペインで、[Network Interfaces] (ネットワークインターフェース) を選択します。
3. インスタンスにアタッチされるネットワークインターフェースを指定します。[Network interface ID] (ネットワークインターフェース ID) リンクを選択して詳細ページを開くことも、リストの左側にあるチェックボックスを選択することもできます。
4. ページ右上の [Action] (アクション) メニューから [Manage ENA Express] (ENA Express の管理) を選択します。これにより、選択したネットワークインターフェイス ID と現在の設定が表示された [Manage ENA Express] (ENA Express の管理) ダイアログが開きます。

   > **Note**
   >
   > 選択したネットワークインターフェースがインスタンスに接続されていない場合、このアクションはメニューに表示されません。

5. [ENA Express] を使用するには、[Enable] (有効にする) チェックボックスを選択します。
6. ENA Express が有効になっている場合、UDP 設定を構成できます。[ENA Express UDP] を使用するには、[Enable] (有効にする) チェックボックスを選択します。
7. 設定を保存するには [Save] (保存) を選択します。

インスタンスリストから ENA Express を管理する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックします。
3. 管理するインスタンスを選択します。[Instance ID] (インスタンス ID) を選択して詳細ページを開くか、リストの左側にあるチェックボックスを選択します。
4. インスタンスに設定する[Network interface] (ネットワークインターフェイス) を選択します。
5. ページ右上の [Action] (アクション) メニューから [Manage ENA Express] (ENA Express の管理) を選択します。
6. インスタンスにアタッチされているネットワークインターフェイスに ENA Express を設定するには、[Network interface] (ネットワークインターフェイス) リストから選択します。
7. 選択したネットワークインターフェースアタッチメントに [ENA Express] を使用するには、[Enable] (有効にする) チェックボックスを選択します。
8. ENA Express が有効になっている場合、UDP 設定を構成できます。[ENA Express UDP] を使用するには、[Enable] (有効にする) チェックボックスを選択します。
9. 設定を保存するには [Save] (保存) を選択します。

ネットワークインターフェースを EC2 インスタンスにアタッチする際に ENA Express を設定する

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側のナビゲーションペインで、[Network Interfaces] (ネットワークインターフェース) を選択します。
3. インスタンスにアタッチされていないネットワークインターフェイスを選択します ([Status] (ステータス) が [Available] (利用可) のもの)。[Network interface ID] (ネットワークインターフェース ID) リンクを選択して詳細ページを開くことも、リストの左側にあるチェックボックスを選択することもできます。
4. アタッチする [Instance] (インスタンス) を選択します。
5. ネットワークインターフェースをインスタンスに接続した後に [ENA Express] を使用するには、[Enable] (有効にする) チェックボックスを選択します。
6. ENA Express が有効になっている場合、UDP 設定を構成できます。[ENA Express UDP] を使用するには、[Enable] (有効にする) チェックボックスを選択します。
7. ネットワークインターフェイスをインスタンスにアタッチし、ENA Express 設定を保存するには、[Attach] (アタッチ) を選択します。

AWS CLI

このタブでは、AWS CLI で ENA Express の設定を行う方法について説明します。

ネットワークインターフェースをアタッチする際の ENA Express の設定

ネットワークインターフェイスをインスタンスに接続するときに ENA Express を設定するには、次の例に示すように AWS CLI で、attach-network-interface コマンドを実行します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では、EnaSrdEnabled を true に設定し、EnaSrdUdpEnabled をデフォルトで false になるように設定します。

```
[ec2-user ~]$ aws ec2 attach-network-interface --network-interface-
id eni-0123f4567890a1b23 --instance-id i-0f1a234b5cd67e890 --device-index 1 --ena-
srd-specification 'EnaSrdEnabled=true'
{
"AttachmentId": "eni-attach-012c3d45e678f9012"
}
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
[ec2-user ~]$ aws ec2 attach-network-interface --network-interface-
id eni-0123f4567890a1b23 --instance-id i-0f1a234b5cd67e890 --device-index 1 --ena-
srd-specification
 'EnaSrdEnabled=true,EnaSrdUdpSpecification={EnaSrdUdpEnabled=true}'
{
"AttachmentId": "eni-attach-012c3d45e678f9012"
}
```

ネットワークインターフェースアタッチメントの ENA Express 設定を更新する

インスタンスにアタッチされているネットワークインターフェイスの ENA Express 設定を更新するには、次の例に示すように、AWS CLI で modify-network-interface-attribute コマンドを実行します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では EnaSrdEnabled を true に設定し、以前に設定したことがない場合は EnaSrdUdpEnabled をデフォルトで false になるよう設定します。

```
[ec2-user ~]$ aws ec2 modify-network-interface-attribute --network-interface-
id eni-0123f4567890a1b23 --ena-srd-specification 'EnaSrdEnabled=true'
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
[ec2-user ~]$ aws ec2 modify-network-interface-attribute --
network-interface-id eni-0123f4567890a1b23 --ena-srd-specification
 'EnaSrdEnabled=true,EnaSrdUdpSpecification={EnaSrdUdpEnabled=true}'
```

例 3: UDP トラフィックでの ENA Express の使用を停止する

この例では、EnaSrdUdpEnabled を false に設定しています。

```
[ec2-user ~]$ aws ec2 modify-network-interface-attribute --
network-interface-id eni-0123f4567890a1b23 --ena-srd-specification
 'EnaSrdUdpSpecification={EnaSrdUdpEnabled=false}'
```

PowerShell

このタブでは、PowerShell を使用して ENA Express の設定を行う方法について説明します。

ネットワークインターフェースをアタッチする際の ENA Express の設定

ネットワークインターフェイスの ENA Express 設定を構成するには、次の例に示すように、Tools for PowerShell で Add-EC2NetworkInterface Cmdlet を実行します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では、EnaSrdEnabled を true に設定し、EnaSrdUdpEnabled をデフォルトで false になるように設定します。

```
PS C:\> Add-EC2NetworkInterface `
-NetworkInterfaceId eni-0123f4567890a1b23 `
-InstanceId i-0f1a234b5cd67e890 `
-DeviceIndex 1 `
```

```
-EnaSrdSpecification_EnaSrdEnabled $true

eni-attach-012c3d45e678f9012
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
PS C:\> Add-EC2NetworkInterface `
-NetworkInterfaceId eni-0123f4567890a1b23 `
-InstanceId i-0f1a234b5cd67e890 `
-DeviceIndex 1 `
-EnaSrdSpecification_EnaSrdEnabled $true `
-EnaSrdUdpSpecification_EnaSrdUdpEnabled $true

eni-attach-012c3d45e678f9012
```

ネットワークインターフェースアタッチメントの ENA Express 設定を更新する

インスタンスにアタッチされているネットワークインターフェイスの ENA Express 設定を更新するには、次の例に示すように、Tools for PowerShell で Add-EC2NetworkInterface Cmdlet コマンドを実行します。

例 1: TCP トラフィックには ENA Express を使用するが、UDP トラフィックには使用しない

この例では EnaSrdEnabled を true に設定し、以前に設定したことがない場合は EnaSrdUdpEnabled をデフォルトで false になるよう設定します。

```
PS C:\> Edit-EC2NetworkInterfaceAttribute `
-NetworkInterfaceId eni-0123f4567890a1b23 `
-EnaSrdSpecification_EnaSrdEnabled $true ;
Get-EC2NetworkInterface -NetworkInterfaceId eni-0123f4567890a1b23 | `
Select-Object `
    NetworkInterfaceId,
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled }},
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled }} | `
Format-List

NetworkInterfaceId : eni-0123f4567890a1b23
```

```
EnaSrdEnabled      : True
EnaSrdUdpEnabled   : False
```

例 2: TCP トラフィックと UDP トラフィックの両方に ENA Express を使用する

この例では、EnaSrdEnabled と EnaSrdUdpEnabled の両方を true に設定します。

```
PS C:\> Edit-EC2NetworkInterfaceAttribute `
-NetworkInterfaceId eni-0123f4567890a1b23 `
-EnaSrdSpecification_EnaSrdEnabled $true `
-EnaSrdSpecification_EnaSrdUdpSpecification_EnaSrdUdpEnabled $true ;
Get-EC2NetworkInterface -NetworkInterfaceId eni-0123f4567890a1b23 | `
Select-Object `
    NetworkInterfaceId,
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled }},
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled }} | `
Format-List

NetworkInterfaceId : eni-0123f4567890a1b23
EnaSrdEnabled      : True
EnaSrdUdpEnabled   : True
```

例 3: UDP トラフィックでの ENA Express の使用を停止する

この例では、EnaSrdUdpEnabled を false に設定しています。

```
PS C:\> Edit-EC2NetworkInterfaceAttribute `
-NetworkInterfaceId eni-0123f4567890a1b23 `
-EnaSrdSpecification_EnaSrdUdpSpecification_EnaSrdUdpEnabled $false ;
Get-EC2NetworkInterface -NetworkInterfaceId eni-0123f4567890a1b23 | `
Select-Object `
    NetworkInterfaceId,
    @{Name = 'EnaSrdEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdEnabled }},
    @{Name = 'EnaSrdUdpEnabled'; Expression =
 { $_.Attachment.EnaSrdSpecification.EnaSrdUdpSpecification.EnaSrdUdpEnabled }} | `
Format-List

NetworkInterfaceId : eni-0123f4567890a1b23
EnaSrdEnabled      : True
```

```
EnaSrdUdpEnabled    : False
```

EC2 インスタンスの起動時に ENA Express を設定する

AWS Management Console からインスタンスを起動する際、次の方法のいずれかを使用して AMI に ENA Express を設定できます。

- インスタンス起動ウィザードでのインスタンスの起動時に、AMI に ENA Express を設定できます。設定の詳細については、インスタンス起動ウィザードの ネットワーク設定、「Advanced network configuration」を参照してください。
- 起動テンプレートの使用時に、AMI に ENA Express を設定できます。起動テンプレートの設定についての詳細は、起動テンプレートの ネットワーク設定、「Advanced network configuration」を参照してください。

ENA Express のパフォーマンスをモニタリングする

送信側インスタンスと受信側インスタンスの両方で、ネットワークインターフェースアタッチメントの ENA Express を有効にしたら、ENA Express メトリックを使用して、SRD テクノロジーによるパフォーマンスの向上をインスタンスが最大限に活用できるようにします。

ENA Express 用にフィルタリングされたメトリクスのリストを表示するには、ネットワークインターフェースで以下の ethtool コマンドを実行します (ここでは `eth0` として表示されています)。

```
[ec2-user ~]$ ethtool -S eth0 | grep ena_srd
NIC statistics:
 ena_srd_mode: 0
 ena_srd_tx_pkts: 0
 ena_srd_eligible_tx_pkts: 0
 ena_srd_rx_pkts: 0
 ena_srd_resource_utilization: 0
```

インスタンスの ENA Express 設定を確認する

インスタンスのネットワークインターフェースアタッチメントの現在の ENA Express 設定を確認するには、`ethtool` コマンドを実行して ENA Express メトリクスを一覧表示し、`ena_srd_mode` メトリクスの値を書き留めます。値は次のとおりです。

- `0` = ENA Express がオフ、UDP がオフ

- 1 = ENA Express がオン、UDP がオフ
- 2 = ENA Express がオフ、UDP がオン

> **Note**
>
> これは、ENA Express が最初に有効になっていて、UDP がそれを使用するように設定されている場合にのみ発生します。UDP トラフィックの以前の値は保持されます。

- 3 = ENA Express がオン、UDP がオン

インスタンスのネットワークインターフェイスアタッチメントで ENA Express を有効にした後、送信側インスタンスは受信側インスタンスとの通信を開始し、SRD は ENA Express が送信側インスタンスと受信側インスタンスの両方で動作しているかどうかを検出します。ENA Express が動作している場合、通信に SRD 送信を使用できます。ENA Express が動作していない場合、通信は標準の ENA 送信にフォールバックします。パケット送信に SRD が使用されているかどうかを確認するには、対象パケットの数 (`ena_srd_eligible_tx_pkts` メトリクス) と特定の期間に送信された SRD パケットの数 (`ena_srd_tx_pkts` メトリクス) を比較します。

`ena_srd_resource_utilization` メトリクスを使用して SRD リソースの使用率をモニタリングできます。インスタンスで SRD リソースが使い果たされそうになっていたら、インスタンスをスケールアウトする時期であると判断できます。

ENA Express のメトリクスに関する詳細は、「ENA Express のメトリクス」を参照してください。

Linux インスタンスで Intel 82599 VF インターフェイスを使用した拡張ネットワーキングの有効化

Amazon EC2 は Intel 82599 VF インターフェイスを通じて拡張ネットワーキング機能を提供しますが、この機能では Intel `ixgbevf` ドライバーを使用します。

コンテンツ

要件

Intel 82599 VF インターフェイスを使用した拡張ネットワーキングを準備するには、次のようにインスタンスをセットアップします。

- サポートされているインスタンスタイプ C3、C4、D2、I2、M4 (`m4.16xlarge` を除く)、R3 から選択します。
- Linux カーネルバージョン 2.6.32 以降を使用して、HVM AMI からインスタンスを起動します。最新の Amazon Linux HVM AMI では、拡張ネットワーキングに必要なモジュールがインストールされており、必要な属性も設定されています。したがって、最新の Amazon Linux HVM AMI を使用して、拡張ネットワーキングがサポートされている Amazon EBS–backed インスタンスを起動した場合は、インスタンスで拡張ネットワーキングが既に有効化されています。

  > **Warning**
  >
  > 拡張ネットワーキングは、HVM インスタンスでのみサポートされています。PV インスタンスで拡張ネットワーキングを有効にすると、このインスタンスに到達できなくなります。また、適切なモジュールまたはモジュールバージョンを使用せずにこの属性を設定すると、インスタンスにアクセスできなくなる場合があります。

- インスタンスがインターネットに接続されていることを確認します。
- 選択した任意のコンピュータ、できればローカルのデスクトップまたはノートパソコンで、AWS Management Console から AWS CloudShell を使用するか、AWS CLI もしくは AWS Tools for Windows PowerShell をインストールし設定します。詳細については、Amazon EC2 へのアクセスもしくは AWS CloudShell ユーザーガイドを参照してください。拡張ネットワーキングは、Amazon EC2 コンソールから管理することはできません。
- 保持する必要がある重要なデータがインスタンスにある場合、インスタンスから AMI を作成してそのデータをバックアップする必要があります。`sriovNetSupport` 属性を有効にするとともに、カーネルおよびカーネルモジュールを更新すると、互換性のないインスタンスがレンダリングされたり、オペレーティングシステムに接続できなくなったりする可能性があります。最近のバックアップがある場合は、これが発生してもデータは保持されます。

拡張ネットワーキングが有効化されているかどうかのテスト

`ixgbevf` モジュールがインスタンスにインストールされており、`sriovNetSupport` 属性が設定されている場合は、Intel 82599 VF インターフェイスを使用した拡張ネットワーキングが既に有効になっています。

インスタンス属性 (sriovNetSupport)

インスタンスに拡張ネットワーキングの sriovNetSupport 属性が設定されているかどうかを確認するには、次のいずれかのコマンドを使用します。

AWS CLI

describe-instance-attribute (AWS CLI/AWS CloudShell)

```
aws ec2 describe-instance-attribute --instance-id instance_id --attribute
 sriovNetSupport
```

PowerShell

Get-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

```
Get-EC2InstanceAttribute -InstanceId instance-id -Attribute sriovNetSupport
```

属性が設定されていない場合、SriovNetSupport は空です。属性が設定されている場合、以下の出力例に示すように、値は simple です。

```
"SriovNetSupport": {
 "Value": "simple"
},
```

イメージ属性 (sriovNetSupport)

AMI に拡張ネットワーキングの sriovNetSupport 属性がすでに設定されているかどうかを確認するには、次のいずれかのコマンドを使用します。

AWS CLI

describe-images (AWS CLI/AWS CloudShell)

```
aws ec2 describe-images --image-id ami_id --query "Images[].SriovNetSupport"
```

PowerShell

Get-EC2Image (AWS Tools for Windows PowerShell)

```
(Get-EC2Image -ImageId ami-id).SriovNetSupport
```

属性が設定されていない場合、SriovNetSupport は空です。属性が設定されている場合、値は simple です。

ネットワークインターフェイスドライバー

次のコマンドを使用して、モジュールが特定のインターフェイスで使用されていることを確認し、確認するインターフェイス名に置き換えます。単一のインターフェイス (デフォルト) を使用している場合は、eth0 です。オペレーティングシステムで予測可能なネットワーク名がサポートされている場合は、ens5 のような名前にすることができます。

次の例で、リストされているドライバーは vif であるため、ixgbevf モジュールはロードされていません。

```
[ec2-user ~]$ ethtool -i eth0
driver: vif
version:
firmware-version:
bus-info: vif-0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

この例では、ixgbevf モジュールがロードされます。このインスタンスでは、拡張ネットワーキングが適切に設定されています。

```
[ec2-user ~]$ ethtool -i eth0
driver: ixgbevf
version: 4.0.3
firmware-version: N/A
bus-info: 0000:00:03.0
supports-statistics: yes
supports-test: yes
supports-eeprom-access: no
supports-register-dump: yes
supports-priv-flags: no
```

Amazon Linux での拡張ネットワーキングの有効化

最新の Amazon Linux HVM AMI では、拡張ネットワーキングに必要な ixgbevf モジュールがインストールされており、必要な sriovNetSupport 属性も設定されています。したがって、最新の

Amazon Linux HVM AMI を使用してインスタンスタイプを起動した場合は、拡張ネットワーキングが既にインスタンスに対して有効になっています。詳細については、拡張ネットワーキングが有効化されているかどうかのテストを参照してください。

以前の Amazon Linux AMI を使用してインスタンスを起動し、まだ拡張ネットワーキングが有効になっていない場合、拡張ネットワーキングを有効にするには次の手順を実行します。

> ⚠ Warning
>
> 拡張ネットワーキング属性は、いったん有効にすると無効にする方法はありません。

拡張ネットワーキングを有効にするには

1. インスタンスに接続します。
2. インスタンスから、次のコマンドを実行して、ixgbevf を含む最新のカーネルとカーネルモジュールでインスタンスを更新します。

   ```
   [ec2-user ~]$ sudo yum update
   ```

3. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを再起動します。reboot-instances (AWS CLI)、Restart-EC2Instance (AWS Tools for Windows PowerShell)。
4. インスタンスに再接続し、ixgbevfの modinfo ixgbevf コマンドを使用して、拡張ネットワーキングが有効化されているかどうかのテスト モジュールがインストールされ、最小推奨バージョンであることを確認します。
5. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: 拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。
6. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

AWS CLI

modify-instance-attribute (AWS CLI/AWS CloudShell)

```
aws ec2 modify-instance-attribute --instance-id instance_id --sriov-net-support
 simple
```

PowerShell

Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

```
Edit-EC2InstanceAttribute -InstanceId instance_id -SriovNetSupport "simple"
```

7. (オプション) Amazon EBS-backed Linux AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング属性を継承します。このため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。
8. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。
9. インスタンスに接続し、拡張ネットワーキングが有効化されているかどうかのテスト の ethtool -i eth*n* コマンドを使用して、`ixgbevf` モジュールがインストールされ、ネットワークインターフェイスにロードされていることを確認します。

拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。instance store-backed Linux AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

AWS CLI

register-image (AWS CLI/AWS CloudShell)

```
aws ec2 register-image --sriov-net-support simple ...
```

PowerShell

Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -SriovNetSupport "simple" ...
```

Ubuntu での拡張ネットワーキングの有効化

開始する前に、インスタンスで拡張ネットワーキングがすでに有効になっているかどうかを確認します。

クイックスタート Ubuntu HVM AMI には、拡張ネットワーキングに必要なドライバーが搭載されています。ixgbevf 2.16.4 より前のバージョンを使用している場合は、linux-aws カーネルパッケージをインストールして最新の拡張ネットワーキングドライバーを取得できます。

以下の手順は、Ubuntu インスタンスで ixgbevf モジュールをコンパイルするための一般的なステップを示しています。

linux-aws カーネルパッケージをインストールするには

1. インスタンスに接続します。
2. パッケージキャッシュおよびパッケージを更新します。

```
ubuntu:~$ sudo apt-get update && sudo apt-get upgrade -y linux-aws
```

> **Important**
>
> 更新プロセス中に grub をインストールするよう求められた場合は、/dev/xvda のインストール先として grub を使用し、現在のバージョンの /boot/grub/menu.lst を保持することを選択します。

他の Linux ディストリビューションでの拡張ネットワーキングの有効化

開始する前に、インスタンスで拡張ネットワーキングがすでに有効になっているかどうかを確認します。最新のクイックスタート HVM AMI には、拡張ネットワーキングに必要なドライバーが含まれているため、追加ステップを実行する必要はありません。

次の手順では、Amazon Linux または Ubuntu 以外の Linux ディストリビューションで Intel 82599 VF インターフェイスを使用した拡張ネットワーキングを有効にする必要がある場合の一般的なステップを説明します。コマンドの詳細な構文、ファイルの場所、パッケージやツールのサポートなどの詳細については、使用する Linux ディストリビューションのドキュメントを参照してください。

Linux で拡張ネットワーキングを有効化するには

1. インスタンスに接続します。
2. Sourceforge (https://sourceforge.net/projects/e1000/files/ixgbevf%20stable/) からインスタンスに `ixgbevf` モジュールのソースをダウンロードします。

   `ixgbevf` の 2.16.4 より前のバージョン (バージョン 2.14.2 を含む) は、一部の Linux ディストリビューション (特定のバージョンの Ubuntu など) では適切にビルドされません。
3. インスタンスで `ixgbevf` モジュールをコンパイルし、インストールします。

   > ⚠ Warning
   >
   > 現在のカーネルに `ixgbevf` モジュールをコンパイルし、新しいカーネルをドライバを再構築しないで更新すると、システムは次回の再起動の際にディストリビューション固有の `ixgbevf` モジュールに戻る場合があります。これにより、ディストリビューション固有のバージョンが拡張ネットワーキングと互換性がない場合に、システムに接続できなくなります。
4. sudo depmod コマンドを実行して、モジュールの依存関係を更新します。
5. 起動時に新しいモジュールがロードされるように、インスタンスの `initramfs` を更新します。
6. システムがデフォルトで予測可能なネットワークインターフェイス名を使用するかどうかを確認します。systemd または udev のバージョン 197 以上を使用するシステムの場合、イーサネットデバイスの名前を変更でき、単一ネットワークインターフェイスの名前が `eth0` になることは保証されません。この動作は、インスタンスに接続する際に問題の原因となる可能性があります。詳細と他の設定オプションについては、freedesktop.org ウェブサイトでPredictable Network Interface Names/を参照してください。

   a. 次のコマンドを使用して、RPM ベースのシステムで systemd または udev のバージョンを確認できます。

   ```
   [ec2-user ~]$ rpm -qa | grep -e '^systemd-[0-9]\+\|^udev-[0-9]\+'
   systemd-208-11.el7_0.2.x86_64
   ```

上記の Red Hat Enterprise Linux 7 の例では、systemd のバージョンは 208 であるため、予測可能なネットワークインターフェイス名は無効になっている必要があります。

b. `net.ifnames=0` オプションを `GRUB_CMDLINE_LINUX` の `/etc/default/grub` 行に追加することによって、予測可能なネットワークインターフェイス名を無効にします。

```
[ec2-user ~]$ sudo sed -i '/^GRUB\_CMDLINE\_LINUX/s/\"$/\ net\.ifnames\=0\"/' /etc/default/grub
```

c. grub の設定ファイルを再ビルドします。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

7. [EBS-backed インスタンス] ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用してインスタンスを停止します。stop-instances (AWS CLI/AWS CloudShell)、Stop-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

   [Instance store-backed インスタンス] インスタンスを停止して属性を変更することはできません。代わりに、この手順に進んでください: 拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)。

8. ローカルコンピュータから、次のいずれかのコマンドを使用して拡張ネットワーキングの属性を有効化します。

   AWS CLI

   modify-instance-attribute (AWS CLI/AWS CloudShell)

   ```
   aws ec2 modify-instance-attribute --instance-id instance_id --sriov-net-support simple
   ```

   PowerShell

   Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

   ```
   Edit-EC2InstanceAttribute -InstanceId instance_id -SriovNetSupport "simple"
   ```

9. (オプション) Amazon EBS-backed Linux AMI を作成するの説明に従って、インスタンスから AMI を作成します。AMI は、インスタンスから拡張ネットワーキング属性を継承します。この

ため、この AMI を使用することで、拡張ネットワーキングがデフォルトで有効になっている別のインスタンスを起動できます。

> **Important**
>
> インスタンスオペレーティングシステムに `/etc/udev/rules.d/70-persistent-net.rules` が含まれている場合には、AMI を作成する前にそれを削除する必要があります。このファイルには、元のインスタンスのイーサネットアダプターの MAC アドレスが保存されています。別のインスタンスがこのファイルを使用して起動した場合、オペレーティングシステムがそのデバイスを検出できなくなり、`eth0` が失敗して、起動に関する問題が発生することがあります。このファイルは次の起動サイクルで再び生成され、AMI から起動されるインスタンスごとに独自のバージョンが作成されます。

10. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを開始します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell)。インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。
11. (オプション) インスタンスに接続し、モジュールがインストールされていることを確認します。

拡張ネットワーキングを有効にするには (Instance store-backed インスタンス)

インスタンスを停止するステップまで、前の手順に従います。instance store-backed Linux AMI を作成するに記述されているように、新しい AMI を作成します。AMI を登録するときに拡張ネットワーキング属性を有効にしてください。

AWS CLI

register-image (AWS CLI/AWS CloudShell)

```
aws ec2 register-image --sriov-net-support simple ...
```

PowerShell

Register-EC2Image (AWS Tools for Windows PowerShell)

```
Register-EC2Image -SriovNetSupport "simple" ...
```

接続に関する問題のトラブルシューティング

拡張ネットワーキングを有効化しているときに接続が失われると、ixgbevf モジュールとカーネルの互換性が保たれない可能性があります。この場合、インスタンスの Linux ディストリビューションに含まれる ixgbevf モジュールのバージョンをインストールしてみます。

PV インスタンスまたは AMI で拡張ネットワーキングを有効にすると、お使いのインスタンスに到達できなくなります。

詳細については、EC2 で拡張ネットワーキングを有効化および設定する方法を参照してください。

オペレーティングシステムの最適化

ネットワーキングが拡張されたインスタンスで最大のネットワークパフォーマンスを実現するには、デフォルトのオペレーティングシステムの設定を変更する必要がある場合があります。詳細については、GitHub の「ENA Linux ドライバーのベストプラクティスとパフォーマンス最適化ガイド」を参照してください。

EC2 インスタンスのネットワークパフォーマンスをモニタリングします。

Elastic Network Adapter (ENA) ドライバーは、有効になっているインスタンスからネットワークパフォーマンスメトリクスを公開します。このようなメトリクスを使用して、インスタンスのパフォーマンスの問題のトラブルシューティング、ワークロードに適したインスタンスサイズの選択、スケーリングアクティビティの事前計画、およびアプリケーションのベンチマークにより、メトリクスがインスタンスで利用できるパフォーマンスを最大化するかどうかを判断できます。

Amazon EC2 は、インスタンスレベルでネットワーク最大値を定義し、インスタンスサイズ全体で一貫したネットワークパフォーマンスを含め、質の高いネットワークエクスペリエンスを実現します。AWS は、各インスタンスに次の最大値を提供します。

- 帯域幅機能 - 各 EC2 インスタンスには、インスタンスタイプとサイズに基づいて、集計したインバウンドトラフィックとアウトバウンドトラフィックの最大帯域幅があります。インスタンスの一部は、ネットワーク I/O クレジットメカニズムを使用して、平均帯域幅使用率に基づいて、ネットワーク帯域幅を割り当てます。また、Amazon EC2 には、AWS Direct Connect およびインターネットへのトラフィックに最大帯域幅があります。詳細については、「Amazon EC2 インスタンスのネットワーク帯域幅」を参照してください。
- Packet-per-second (PPS) パフォーマンス - 各 EC2 インスタンスには、インスタンスタイプとサイズに基づいて、最大 PPS パフォーマンスがあります。

- 追跡された接続 - セキュリティグループは、確立された各接続を追跡し、リターンパケットが期待どおりに配信されることを確認します。インスタンスごとに追跡できる接続の最大数があります。詳細については、「セキュリティグループの接続の追跡」を参照してください。
- リンクローカルサービスアクセス - Amazon EC2 は、DNS サービス、インスタンスメタデータサービス、Amazon Time Sync Service などのサービスへのトラフィックに対して、ネットワークインターフェイスごとに最大 PPS を提供します。

インスタンスのネットワークトラフィックが最大値を超えると、AWS はネットワークパケットをキューイングしてから破棄することによって、最大値を超えるトラフィックを調整します。ネットワークパフォーマンスメトリクスを使用して、トラフィックが最大値を超えるタイミングをモニタリングできます。これらのメトリクスは、ネットワークトラフィックへの影響、およびネットワークパフォーマンスの問題の可能性をリアルタイムで通知します。

コンテンツ

要件

Linux インスタンスには、次の要件が適用されます。

- ENA ドライババージョン 2.2.10 以降をインストールします。インストールしたバージョンを検証するには、ethtool コマンドを使用します。次の例では、バージョンは最小要件を満たしています。

```
[ec2-user ~]$ ethtool -i eth0 | grep version
version: 2.2.10
```

  ENA ドライバーをアップグレードするには、拡張ネットワーキングを参照してください。

- これらのメトリクスを Amazon CloudWatch にインポートするには、CloudWatch エージェントをインストールします。詳細については、「Amazon CloudWatch ユーザーガイド」の「ネットワークパフォーマンスメトリクスの収集」を参照してください。

- conntrack_allowance_available メトリクスをサポートするには、ENA ドライバーのバージョン 2.8.1 をインストールします。

ENA ドライバーのメトリクス

ENA ドライバーは、次のメトリクスをリアルタイムでインスタンスに配信します。前回のドライバーのリセット以降に、各ネットワークインターフェイスでキューまたはドロップされたパケットの累積数を示します。

Linux インスタンス、FreeBSD インスタンス、および DPDK 環境では、次のメトリクスを使用できます。

メトリクス	説明	以下でサポートされます
bw_in_allowance_exceeded	インバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。	すべてのインスタンスタイプ
bw_out_allowance_exceeded	アウトバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。	すべてのインスタンスタイプ
conntrack_allowance_exceeded	接続トラッキングがインスタンスの最大数を超え、新しい接続を確立できなかったためにドロップされたパケットの数。これにより、インスタンスとの間で送受信されるトラフィックのパケット損失が発生する可能性があります。	すべてのインスタンスタイプ
conntrack_allowance_available	そのインスタンスタイプの Connections Tracked 許容量に達す	Nitro ベースのインスタンスタイプのみ。

メトリクス	説明	以下でサポートされます
	る前にインスタンスが確立できる接続トラッキング数。	FreeBSD インスタンスまたは DPDK 環境ではサポートされません。
linklocal_allowance_exceeded	ローカルプロキシサービスへのトラフィックの PPS がネットワークインターフェイスの最大値を超えたためにドロップされたパケットの数。これは、DNS サービス、インスタンスメタデータサービス、および Amazon Time Sync Service へのトラフィックに影響します。	すべてのインスタンスタイプ
pps_allowance_exceeded	双方向 PPS がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。	すべてのインスタンスタイプ

Linux インスタンスのネットワークパフォーマンスメトリクスを表示します。

メトリクスをお気に入りのツールに公開して、メトリクスデータを視覚化できます。例えば、CloudWatch エージェントを使用してメトリックスを Amazon CloudWatch に公開できます。エージェントにより、個々のメトリクスを選択し、公開を制御できます。

ethtool を使用して、次のように eth0 などの各ネットワークインターフェイスのメトリクスを取得することもできます。

```
[ec2-user ~]$ ethtool -S eth0
     bw_in_allowance_exceeded: 0
     bw_out_allowance_exceeded: 0
     pps_allowance_exceeded: 0
     conntrack_allowance_exceeded: 0
     linklocal_allowance_exceeded: 0
     conntrack_allowance_available: 136812
```

ENA Express のメトリクス

ENA Express は、AWS Scalable Reliable Datagram (SRD) テクノロジーを搭載しています。SRD は、動的ルーティングを使用してスループットを向上させ、テールレイテンシーを最小限に抑える高性能なネットワークトランスポートプロトコルです。ENA Express メトリクスを使用すると、SRD テクノロジーがもたらすパフォーマンスの向上をインスタンスが最大限に活用できるようになります。次に例を示します。

- より多くの SRD 接続を確立するのに十分な容量があることを確認するために、リソースを評価します。
- 対象となる送信パケットで SRD を使用できない原因となる潜在的な問題がある箇所を特定します。
- インスタンスに SRD を使用する送信トラフィックの割合を計算します。
- インスタンスに SRD を使用する受信トラフィックの割合を計算します。

> **Note**
>
> メトリクスを生成するには、ドライバーバージョン 2.8 以降を使用してください。

Linux ベースのインスタンスでは、ethtool コマンドを使用することで以下の ENA Express メトリクスを利用できます。

- `ena_srd_mode` — ENA Express のどの機能が有効になっているかを説明します。値は次のとおりです。
  - `0` = ENA Express がオフ、UDP がオフ
  - `1` = ENA Express がオン、UDP がオフ
  - `2` = ENA Express がオフ、UDP がオン

    > **Note**
    >
    > これは、ENA Express が最初に有効になっていて、UDP がそれを使用するように設定されている場合にのみ発生します。UDP トラフィックの以前の値は保持されます。

  - `3` = ENA Express がオン、UDP がオン

- ena_srd_eligible_tx_pkts — 一定期間内に送信された SRD の資格要件を満たすネットワークパケットの数。次のようになります。
  - 送信側と受信側の両方のインスタンスタイプがサポートされています。詳細については「ENA Express でサポートされるインスタンスタイプ」の表を参照してください。
  - 送信側と受信側の両方のインスタンスに ENA Express が設定されている必要があります。
  - 送信側と受信側のインスタンスは同じサブネットで実行する必要があります。
  - インスタンス間のネットワークパスには、ミドルウェアボックスを含めないようにしてください。ENA Express は現在、ミドルウェアボックスをサポートしていません。

> **Note**
>
> ENA Express の適格性メトリクスには、ソースと送信先の要件、および 2 つのエンドポイント間のネットワークが含まれます。対象となるパケットは、既にカウントされた後でも失格となる可能性があります。例えば、対象となるパケットが最大送信単位 (MTU) の制限を超えている場合、そのパケットはカウンターに適格として反映されますが、標準の ENA 送信にフォールバックします。

- ena_srd_tx_pkts — 一定期間内に送信した SRD パケット数。
- ena_srd_rx_pkts — 一定期間内に受信した SRD パケット数。
- ena_srd_resource_utilization — インスタンスが消費した同時 SRD 接続の最大許容メモリ使用量の割合。

ENA Express 用にフィルタリングされたメトリクスのリストを表示するには、ネットワークインターフェースで以下の ethtool コマンドを実行します (ここでは eth0 として表示されています)。

```
[ec2-user ~]$ ethtool -S eth0 | grep ena_srd
NIC statistics:
 ena_srd_mode: 0
 ena_srd_tx_pkts: 0
 ena_srd_eligible_tx_pkts: 0
 ena_srd_rx_pkts: 0
 ena_srd_resource_utilization: 0
```

エグレストラフィック (送信パケット)

エグレストラフィックが想定どおりに SRD を使用するようにするには、SRD 送信に適格なパケットの数 (ena_srd_eligible_tx_pkts) と、特定の期間に送信された SRD パケットの数 (ena_srd_tx_pkts) を比較します。

使用可能なパケット数と送信された SRD パケット数の差が大きい場合、リソース使用率の問題が原因である可能性が高いです。インスタンスにアタッチされたネットワークカードが最大リソースを使い果たしている場合、またはパケットが MTU 制限を超えている場合、適格なパケットでも SRD 経由で送信できないため、標準の ENA 送信にフォールバックする必要があります。ライブ移行中やライブサーバー更新中にもパケットがこのギャップに陥る可能性があります。根本原因を特定するには、追加のトラブルシューティングが必要です。

> **Note**
>
> 適格なパケット数と SRD パケット数のわずかな違いは無視してかまいません。これは、例えば、インスタンスが SRD トラフィック用に別のインスタンスへの接続を確立した場合に発生する可能性があります。

特定の期間における総エグレストラフィックの何パーセントで SRD が使用されているかを調べるには、送信された SRD パケット数 (ena_srd_tx_pkts) と、その期間にインスタンスに送信されたパケットの総数 (NetworkPacketOut) を比較します。

イングレストラフィック (受信パケット)

総イングレストラフィックの何パーセントで SRD が使用されているかを調べるには、特定の期間に受信した SRD パケット数 (ena_srd_rx_pkts) と、その期間にインスタンスで受信されたパケットの総数 (NetworkPacketIn) を比較します。

リソース使用率

リソース使用率は、1 つのインスタンスが一定時間に保持できる SRD の同時接続数に基づいています。リソース使用率メトリクス (ena_srd_resource_utilization) は、インスタンスの現在の使用率を追跡します。使用率が 100% に近づくと、パフォーマンスの問題が発生することが予想されます。ENA Express は SRD から標準の ENA 送信にフォールバックし、パケットドロップの可能性が高まります。リソース使用率が高い場合は、ネットワークパフォーマンスを向上させるためにインスタンスをスケールアウトする時期が来たと判断できます。

> **Note**
>
> インスタンスのネットワークトラフィックが最大値を超えると、AWS はネットワークパケットをキューイングしてから破棄することによって、最大値を超えるトラフィックを調整します。

永続的

エグレスメトリクスとイングレスメトリクスは、インスタンスで ENA Express が有効になっている間に発生します。ENA Express が非アクティブ化されるとメトリクスの発生しなくなりますが、インスタンスがまだ実行されている限り持続します。インスタンスが再起動または終了した場合、またはネットワークインターフェイスがインスタンスから切り離されると、メトリクスはリセットされます。

ENA 用の DPDK ドライバーを備えたネットワークパフォーマンスメトリクス

ENA ドライババージョン 2.2.0 以降では、ネットワークメトリクスのレポートがサポートされています。DPDK 20.11 には ENA ドライバー 2.2.0 が含まれており、この機能をサポートする最初の DPDK バージョンです。

サンプルアプリケーションを使用して、DPDK 統計を表示できます。サンプルアプリケーションの対話型バージョンを開始するには、次のコマンドを実行します。

```
./app/dpdk-testpmd -- -i
```

この対話型セッションでは、コマンドを入力してポートの拡張統計情報を取得できます。次のコマンド例では、ポート 0 の統計情報を取得します。

```
show port xstats 0
```

次に、DPDK サンプルアプリケーションとの対話型セッションの例を示します。

```
[root@ip-192.0.2.0 build]# ./app/dpdk-testpmd -- -i
        EAL: Detected 4 lcore(s)
        EAL: Detected 1 NUMA nodes
        EAL: Multi-process socket /var/run/dpdk/rte/mp_socket
        EAL: Selected IOVA mode 'PA'
        EAL: Probing VFIO support...
        EAL:   Invalid NUMA socket, default to 0
```

```
        EAL:   Invalid NUMA socket, default to 0
        EAL: Probe PCI driver: net_ena (1d0f:ec20) device: 0000:00:06.0
(socket 0)
        EAL: No legacy callbacks, legacy socket not created
        Interactive-mode selected

        Port 0: link state change event
        testpmd: create a new mbuf pool <mb_pool_0>: n=171456,
size=2176, socket=0
        testpmd: preferred mempool ops selected: ring_mp_mc

        Warning! port-topology=paired and odd forward ports number, the
last port will pair with itself.

        Configuring Port 0 (socket 0)
        Port 0: 02:C7:17:A2:60:B1
        Checking link statuses...
        Done
        Error during enabling promiscuous mode for port 0: Operation
not supported - ignore
        testpmd> show port xstats 0
        ###### NIC extended statistics for port 0
        rx_good_packets: 0
        tx_good_packets: 0
        rx_good_bytes: 0
        tx_good_bytes: 0
        rx_missed_errors: 0
        rx_errors: 0
        tx_errors: 0
        rx_mbuf_allocation_errors: 0
        rx_q0_packets: 0
        rx_q0_bytes: 0
        rx_q0_errors: 0
        tx_q0_packets: 0
        tx_q0_bytes: 0
        wd_expired: 0
        dev_start: 1
        dev_stop: 0
        tx_drops: 0
        bw_in_allowance_exceeded: 0
        bw_out_allowance_exceeded: 0
        pps_allowance_exceeded: 0
        conntrack_allowance_exceeded: 0
        linklocal_allowance_exceeded: 0
```

```
        rx_q0_cnt: 0
        rx_q0_bytes: 0
        rx_q0_refill_partial: 0
        rx_q0_bad_csum: 0
        rx_q0_mbuf_alloc_fail: 0
        rx_q0_bad_desc_num: 0
        rx_q0_bad_req_id: 0
        tx_q0_cnt: 0
        tx_q0_bytes: 0
        tx_q0_prepare_ctx_err: 0
        tx_q0_linearize: 0
        tx_q0_linearize_failed: 0
        tx_q0_tx_poll: 0
        tx_q0_doorbells: 0
        tx_q0_bad_req_id: 0
        tx_q0_available_desc: 1023
        testpmd>
```

サンプルアプリケーションとその使用による拡張統計情報の取得の詳細については、DPDK ドキュメントのTestpmd アプリケーションユーザーガイドを参照してください。

FreeBSD を実行しているインスタンスのメトリクス

ENA FreeBSD ドライバーのバージョン 2.3.0 以降では、FreeBSD を実行しているインスタンスでのネットワークパフォーマンスメトリクスの収集をサポートしています。FreeBSD メトリクスの収集を有効にするには、次のコマンドを入力し、[*##*]を 1 ~ 3600 の値に設定します。FreeBSD メトリクスを収集する頻度を秒単位で特定します。

```
sysctl dev.ena.network_interface.eni_metrics.sample_interval=interval
```

例えば、次のコマンドは、ネットワークインターフェイス 1 の FreeBSD メトリクスを 10 秒ごとに収集するようにドライバーを設定します。

```
sysctl dev.ena.1.eni_metrics.sample_interval=10
```

FreeBSD メトリクスの収集をオフにするには、上記のコマンドを実行し、*##*を `0` に指定します。

FreeBSD メトリクスを収集したら、次のコマンドを実行して、収集されたメトリクスの最新セットを取得できます。

```
sysctl dev.ena.network_interface.eni_metrics
```

Elastic Network Adapter (ENA) のトラブルシューティング

Elastic Network Adapter (ENA) は、オペレーティング システムのヘルスを向上し、予期しないハードウェア動作や障害による長期的な停止の可能性を減らすように設計されています。ENA アーキテクチャでは、デバイスやドライバーの障害がシステムに対して可能な限り透過的に保持されます。このトピックでは、ENA のトラブルシューティングについて説明します。

インスタンスに接続できない場合は、まず接続に関する問題のトラブルシューティングセクションを参照してください。

第 6 世代のインスタンスタイプに移行した後にパフォーマンスの低下が発生した場合は、AWS ナレッジセンターの記事「最大のネットワークパフォーマンスを得られるようにするには、EC2 インスタンスを第 6 世代インスタンスに移行する前に何をしておく必要がありますか?」を参照してください。

インスタンスに接続できる場合、このトピックの以降のセクションに記載されている障害検出/復旧メカニズムを使用して診断情報を収集することができます。

コンテンツ

接続に関する問題のトラブルシューティング

拡張ネットワーキングを有効化しているときに接続が失われると、ena モジュールとインスタンスの現在実行中のカーネルの互換性が保たれない可能性があります。これは、特定のカーネルバージョンのモジュールをインストール (dkms を使用しないか、不適切な設定の dkms.conf ファイルを使用) したため、インスタンスカーネルが更新された場合に発生します。起動時にロードされるインスタンスカーネルにより、ena モジュールが正しくインストールされない場合、インスタンスがネットワークアダプタを認識せず、インスタンスが到達不可能になります。

PV インスタンスまたは AMI で拡張ネットワーキングを有効にすると、お使いのインスタンスにも到達できなくなります。

ENA を使用して拡張ネットワーキングを有効した後インスタンスが到達不可能になった場合、インスタンスの enaSupport 属性を無効にすると、ストックネットワークアダプタにフォールバックできます。

ENA を使用して拡張ネットワーキングを無効にするには (EBS-backed インスタンス)

1. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを停止します。stop-instances (AWS CLI)、Stop-EC2Instance (AWS Tools for Windows PowerShell) インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。

   > **Important**
   >
   > instance store-backed インスタンスを使用している場合、インスタンスを停止することはできません。代わりに、ENA を使用して拡張ネットワーキングを無効にするには (Instance store-backed インスタンス)に進みます。

2. ローカルコンピュータから、次のコマンドを使用して拡張ネットワーキングの属性を無効化します。

   - modify-instance-attribute (AWS CLI)

   ```
   $ aws ec2 modify-instance-attribute --instance-id instance_id --no-ena-support
   ```

3. ローカルコンピュータから、Amazon EC2 コンソールまたは次のいずれかのコマンドを使用して、インスタンスを起動します。start-instances (AWS CLI)、Start-EC2Instance (AWS Tools for Windows PowerShell) インスタンスを AWS OpsWorks で管理する場合、インスタンスの状態が同期し続けるために、AWS OpsWorks コンソールでインスタンスを停止する必要があります。
4. (オプション) インスタンスに接続し、enaのステップに従って、現在のカーネルバージョンを使用して Linux インスタンスにおける Elastic Network Adapter (ENA) を使用した拡張ネットワーキングの有効化 モジュールの再インストールを試みます。

ENA を使用して拡張ネットワーキングを無効にするには (Instance store-backed インスタンス)

インスタンスが instance store-backed インスタンスの場合、instance store-backed Linux AMI を作成するの説明に従って新しい AMI を作成します。AMI を登録するときに、必ず拡張ネットワーキング enaSupport 属性を無効化してください。

- register-image (AWS CLI)

```
$ aws ec2 register-image --no-ena-support ...
```

- Register-EC2Image (AWS Tools for Windows PowerShell)

```
C:\> Register-EC2Image -EnaSupport $false ...
```

キープアライブメカニズム

ENA デバイスは、キープアライブイベントを一定の速度 (通常は 1 秒に 1 回) で送信します。ENA ドライバーは、これらのキープアライブメッセージの存在を確認するウォッチドッグメカニズムを実装します。メッセージが存在する場合、ウォッチドッグが再実装されます。存在しない場合、ドライバーはデバイスで障害が発生したと判断し、次の処理を行います。

- 現在の統計を syslog にダンプする
- ENA デバイスをリセットする
- ENA のドライバー状態をリセットする

上記のリセット手順を実行すると、トラフィックが短時間失われる可能性がありますが (TCP 接続は回復可能です)、ユーザーに影響は及びません。

ENA デバイスは、キープアライブ通知を送信しないことによりデバイスリセット手順を間接的にリクエストすることがあります。例えば、ENA デバイスが回復不可能な設定をロードした後に不明な状態になった場合などです。

リセット手順の例を以下に示します。

```
[18509.800135] ena 0000:00:07.0 eth1: Keep alive watchdog timeout. // The watchdog
 process initiates a reset
[18509.815244] ena 0000:00:07.0 eth1: Trigger reset is on
[18509.825589] ena 0000:00:07.0 eth1: tx_timeout: 0 // The driver logs the current
 statistics
[18509.834253] ena 0000:00:07.0 eth1: io_suspend: 0
[18509.842674] ena 0000:00:07.0 eth1: io_resume: 0
[18509.850275] ena 0000:00:07.0 eth1: wd_expired: 1
[18509.857855] ena 0000:00:07.0 eth1: interface_up: 1
[18509.865415] ena 0000:00:07.0 eth1: interface_down: 0
[18509.873468] ena 0000:00:07.0 eth1: admin_q_pause: 0
[18509.881075] ena 0000:00:07.0 eth1: queue_0_tx_cnt: 0
```

```
[18509.888629] ena 0000:00:07.0 eth1: queue_0_tx_bytes: 0
[18509.895286] ena 0000:00:07.0 eth1: queue_0_tx_queue_stop: 0
.......
........
[18511.280972] ena 0000:00:07.0 eth1: free uncompleted tx skb qid 3 idx 0x7 // At the
 end of the down process, the driver discards incomplete packets.
[18511.420112] [ENA_COM: ena_com_validate_version] ena device version: 0.10 //The
 driver begins its up process
[18511.420119] [ENA_COM: ena_com_validate_version] ena controller version: 0.0.1
 implementation version 1
[18511.420127] [ENA_COM: ena_com_admin_init] ena_defs : Version:[b9692e8] Build date
 [Wed Apr  6 09:54:21 IDT 2016]
[18512.252108] ena 0000:00:07.0: Device watchdog is Enabled
[18512.674877] ena 0000:00:07.0: irq 46 for MSI/MSI-X
[18512.674933] ena 0000:00:07.0: irq 47 for MSI/MSI-X
[18512.674990] ena 0000:00:07.0: irq 48 for MSI/MSI-X
[18512.675037] ena 0000:00:07.0: irq 49 for MSI/MSI-X
[18512.675085] ena 0000:00:07.0: irq 50 for MSI/MSI-X
[18512.675141] ena 0000:00:07.0: irq 51 for MSI/MSI-X
[18512.675188] ena 0000:00:07.0: irq 52 for MSI/MSI-X
[18512.675233] ena 0000:00:07.0: irq 53 for MSI/MSI-X
[18512.675279] ena 0000:00:07.0: irq 54 for MSI/MSI-X
[18512.772641] [ENA_COM: ena_com_set_hash_function] Feature 10 isn't supported
[18512.772647] [ENA_COM: ena_com_set_hash_ctrl] Feature 18 isn't supported
[18512.775945] ena 0000:00:07.0: Device reset completed successfully // The reset
 process is complete
```

読み取りタイムアウトの登録

ENA アーキテクチャでは、Memory Mapped I/O (MMIO) の読み取りオペレーションの限定的に使用することが推奨されます。MMIO レジスタには、初期化手順中のみ ENA デバイスドライバーがアクセスします。

ドライバーログ (dmesg 出力にあります) が読み取りオペレーションの失敗を示している場合、互換性のないドライバーまたは適切にコンパイルされていないドライバー、ビジー状態のハードウェアドライバー、ハードウェア障害が原因の可能性があります。

読み取りオペレーションの失敗を示すログエントリが断続的に発生する場合は、問題とみなさないでください。この場合はドライバーによって再試行されます。ただし、読み取りの失敗を含むログエントリが連続して発生する場合は、ドライバーまたはハードウェアの問題を示しています。

タイムアウトによる読み取りオペレーション失敗を示すドライバーログエントリの例を以下に示します。

```
[ 47.113698] [ENA_COM: ena_com_reg_bar_read32] reading reg failed for timeout.
 expected: req id[1] offset[88] actual: req id[57006] offset[0]
[ 47.333715] [ENA_COM: ena_com_reg_bar_read32] reading reg failed for timeout.
 expected: req id[2] offset[8] actual: req id[57007] offset[0]
[ 47.346221] [ENA_COM: ena_com_dev_reset] Reg read32 timeout occurred
```

統計

ネットワークパフォーマンスが不十分な場合やレイテンシーの問題がある場合、デバイス統計情報を取得して調査する必要があります。これらの統計は、以下に示すように ethtool を使用して取得できます。

```
[ec2-user ~]$ ethtool -S ethN
NIC statistics:
    tx_timeout: 0
    suspend: 0
    resume: 0
    wd_expired: 0
    interface_up: 1
    interface_down: 0
    admin_q_pause: 0
    bw_in_allowance_exceeded: 0
    bw_out_allowance_exceeded: 0
    pps_allowance_exceeded: 0
    conntrack_allowance_available: 450878
    conntrack_allowance_exceeded: 0
    linklocal_allowance_exceeded: 0
    queue_0_tx_cnt: 4329
    queue_0_tx_bytes: 1075749
    queue_0_tx_queue_stop: 0
    ...
```

次のコマンド出力パラメータの説明を以下に示します。

tx_timeout: *N*

Netdev ウォッチドッグがアクティブになった回数。

suspend: *N*

ドライバーが停止操作を実行した回数。

`resume`: *N*

ドライバーが再開操作を実行した回数。

`wd_expired`: *N*

ドライバーが直近 3 秒以内にキープアライブイベントを受け取らなかった回数。

`interface_up`: *N*

ENA インターフェイスが起動された回数。

`interface_down`: *N*

ENA インターフェイスが停止された回数。

`admin_q_pause`: *N*

管理キューが実行状態で見つからなかった回数。

`bw_in_allowance_exceeded`: *N*

インバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。

`bw_out_allowance_exceeded`: *N*

アウトバウンド集計帯域幅がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。

`pps_allowance_exceeded`: *N*

双方向 PPS がインスタンスの最大値を超えたためにキューまたはドロップされたパケットの数。

`conntrack_allowance_available`: *N*

そのインスタンスタイプの Connections Tracked 許容量に達する前にインスタンスが確立できる接続トラッキング数。Nitro ベースのインスタンスでのみ使用できます。FreeBSD インスタンスまたは DPDK 環境ではサポートされません。

`conntrack_allowance_exceeded`: *N*

接続トラッキングがインスタンスの最大数を超え、新しい接続を確立できなかったためにドロップされたパケットの数。これにより、インスタンスとの間で送受信されるトラフィックのパケット損失が発生する可能性があります。

`linklocal_allowance_exceeded: N`

ローカルプロキシサービスへのトラフィックの PPS がネットワークインターフェイスの最大値を超えたためにドロップされたパケットの数。これは、DNS サービス、インスタンスメタデータサービス、および Amazon Time Sync Service へのトラフィックに影響します。

`queue_N_tx_cnt: N`

このキューの送信パケット数。

`queue_N_tx_bytes: N`

このキューの送信バイト数。

`queue_N_tx_queue_stop: N`

キュー *N* がいっぱいになって停止された回数。

`queue_N_tx_queue_wakeup: N`

停止後にキュー *N* が再開された回数。

`queue_N_tx_dma_mapping_err: N`

直接メモリアクセスエラーの数。この値が 0 の場合は、システムリソースが低いことを示しています。

`queue_N_tx_linearize: N`

このキューに SKB 線形化が試行された回数。

`queue_N_tx_linearize_failed: N`

このキューで SKB 線形化が失敗した回数。

`queue_N_tx_napi_comp: N`

`napi` ハンドラーがこのキューの `napi_complete` を呼び出した回数。

`queue_N_tx_tx_poll: N`

`napi` ハンドラーがこのキューにスケジュールされた回数。

`queue_N_tx_doorbells: N`

このキューの送信ドアベルの数。

`queue_N_tx_prepare_ctx_err: N`

このキューで `ena_com_prepare_tx` が失敗した回数。

queue_*N*_tx_bad_req_id: *N*

このキューの req_id が無効です。有効な req_id は 0、マイナス queue_size、マイナス 1 です。

queue_*N*_tx_llq_buffer_copy: *N*

このキューの llq エントリよりもヘッダーサイズが大きいパケットの数。

queue_*N*_tx_missed_tx: *N*

このキューの未処理のパケット数。

queue_*N*_tx_unmask_interrupt: *N*

このキューで tx 割り込みがマスク解除された回数。

queue_*N*_rx_cnt: *N*

このキューで受信したパケット数。

queue_*N*_rx_bytes: *N*

このキューの受信バイト数。

queue_*N*_rx_rx_copybreak_pkt: *N*

rx キューが、このキューの rx_copybreak パケットサイズより小さいパケットを受信した回数。

queue_*N*_rx_csum_good: *N*

rx キューが、チェックサムをチェックし、このキューに対して正しいパケットを受信した回数。

queue_*N*_rx_refil_partial: *N*

ドライバーが rx キューの空いている部分にこのキューのバッファーを補充できなかった回数。この値が 0 でない場合、メモリリソースが低いことを示しています。

queue_*N*_rx_bad_csum: *N*

rx キューに、このキューの不良なチェックサムがあった回数 (rx チェックサムオフロードがサポートされている場合のみ)。

queue_*N*_rx_page_alloc_fail: *N*

このキューのページ割り当てに失敗した回数。この値が 0 でない場合、メモリリソースが低いことを示しています。

queue_*N*_rx_skb_alloc_fail: *N*

このキューの SKB 割り当てに失敗した回数。この値が 0 でない場合、システムリソースが低いことを示しています。

queue_*N*_rx_dma_mapping_err: *N*

直接メモリアクセスエラーの数。この値が 0 の場合は、システムリソースが低いことを示しています。

queue_*N*_rx_bad_desc_num: *N*

パケットあたりのバッファーが多すぎます。この値が 0 でない場合、バッファーの使用量が非常に少ないことを示しています。

queue_*N*_rx_bad_req_id: *N*

このキューの req_id は無効です。有効な req_id は [0, queue_size - 1] です。

queue_*N*_rx_empty_rx_ring: *N*

このキューの rx キューが空だった回数。

queue_*N*_rx_csum_unchecked: *N*

rx キューが、このキューに対してチェックサムがチェックされなかったパケットを受信した回数。

queue_*N*_rx_xdp_aborted: *N*

XDP パケットが XDP_ABORT として分類された回数。

queue_*N*_rx_xdp_drop: *N*

XDP パケットが XDP_DROP として分類された回数。

queue_*N*_rx_xdp_pass: *N*

XDP パケットが XDP_PASS として分類された回数。

queue_*N*_rx_xdp_tx: *N*

XDP パケットが XDP_TX として分類された回数。

queue_*N*_rx_xdp_invalid: *N*

パケットの XDP リターンコードが無効な回数。

queue_*N*_rx_xdp_redirect: *N*

XDP パケットが XDP_REDIRECT として分類された回数。

queue_*N*_xdp_tx_cnt: *N*

このキューの送信パケット数。

queue_*N*_xdp_tx_bytes: *N*

このキューの送信バイト数。

queue_*N*_xdp_tx_queue_stop: *N*

このキューがいっぱいになって停止した回数。

queue_*N*_xdp_tx_queue_wakeup: *N*

停止後にこのキューが再開された回数。

queue_*N*_xdp_tx_dma_mapping_err: *N*

直接メモリアクセスエラーの数。この値が 0 の場合は、システムリソースが低いことを示しています。

queue_*N*_xdp_tx_linearize: *N*

このキューに XDP バッファ線形化が試行された回数。

queue_*N*_xdp_tx_linearize_failed: *N*

このキューで XDP バッファ線形化が失敗した回数。

queue_*N*_xdp_tx_napi_comp: *N*

このキューで napi ハンドラーが napi_complete を呼び出した回数。

queue_*N*_xdp_tx_tx_poll: *N*

このキューで napi ハンドラーがスケジュールされた回数。

queue_*N*_xdp_tx_doorbells: *N*

このキューの送信ドアベルの数。

queue_*N*_xdp_tx_prepare_ctx_err: *N*

このキューで ena_com_prepare_tx が失敗した回数。この値は、常に 0 になる必要があります。そうでない場合はドライバーログを参照してください。

queue_*N*_xdp_tx_bad_req_id: *N*

このキューの req_id は無効です。有効な req_id は [0, queue_size - 1] です。

queue_*N*_xdp_tx_llq_buffer_copy: *N*

このキューの llq バッファコピーを使用してヘッダーをコピーしたパケット数。

queue_*N*_xdp_tx_missed_tx: *N*

tx キューエントリがこのキューの完了タイムアウトを逃した回数。

queue_*N*_xdp_tx_unmask_interrupt: *N*

このキューで tx 割り込みがマスク解除された回数。

ena_admin_q_aborted_cmd: *N*

中断された管理コマンドの数。これは、通常自動リカバリ手順中に発生します。

ena_admin_q_submitted_cmd: *N*

管理者キューのドアベルの数。

ena_admin_q_completed_cmd: *N*

管理者キューの完了数。

ena_admin_q_out_of_space: *N*

ドライバーが新しい管理コマンドの送信を試みたが、キューがいっぱいであった回数。

ena_admin_q_no_completion: *N*

ドライバーが管理コマンドの完了を取得しなかった回数。

syslog のドライバーエラーログ

ENA ドライバーは、システム起動時にログメッセージを syslog に書き込みます。問題が発生した場合、これらのログを調べてエラーを探すことができます。システム起動時に ENA ドライバーにより syslog に記録される情報の例と、特定のメッセージの注釈の一部を以下に示します。

```
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  478.416939] [ENA_COM:
 ena_com_validate_version] ena device version: 0.10
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  478.420915] [ENA_COM:
 ena_com_validate_version] ena controller version: 0.0.1 implementation version 1
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.256831] ena 0000:00:03.0: Device
 watchdog is Enabled
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.672947] ena 0000:00:03.0: creating 8 io
 queues. queue size: 1024
```

```
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.680885] [ENA_COM:
 ena_com_init_interrupt_moderation] Feature 20 isn't supported  // Interrupt moderation
 is not supported by the device
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.691609] [ENA_COM:
 ena_com_get_feature_ex] Feature 10 isn't supported // RSS HASH function configuration
 is not supported by the device
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.694583] [ENA_COM:
 ena_com_get_feature_ex] Feature 18 isn't supported //RSS HASH input source
 configuration is not supported by the device
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.697433] [ENA_COM:
 ena_com_set_host_attributes] Set host attribute isn't supported
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.701064] ena 0000:00:03.0 (unnamed
 net_device) (uninitialized): Cannot set host attributes
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  479.704917] ena 0000:00:03.0: Elastic
 Network Adapter (ENA) found at mem f3000000, mac addr 02:8a:3c:1e:13:b5 Queues 8
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  480.805037] EXT4-fs (xvda1): re-mounted.
 Opts: (null)
Jun  3 22:37:46 ip-172-31-2-186 kernel: [  481.025842] NET: Registered protocol family
 10
```

無視可能なエラー

システムのエラーログに記録される可能性がある以下のエラーは、Elastic Network Adapter では無視できます。

ホスト属性の設定がサポートされない

ホスト属性は、このデバイスではサポートされていません。

rx キューのバッファの割り当てに失敗した

これは復元可能なエラーであり、エラーがスローされたときにメモリプレッシャーの問題が発生した可能性があることを示します。

機能 *X* はサポートされていない

言及されている機能は、Elastic Network Adapter ではサポートされていません。*X* に指定できる値は、以下のとおりです。

- **10**: RSS ハッシュ関数設定は、このデバイスではサポートされていません。
- **12**: RSS 間接テーブル設定は、このデバイスではサポートされていません。
- **18**: RSS ハッシュ入力設定は、このデバイスではサポートされていません。
- **20**: 割り込みモデレーションは、このデバイスではサポートされていません。

- **27**: Elastic Network Adapter ドライバーは、snmpd からのイーサネット機能のポーリングをサポートしていません。

AENQ の設定に失敗した

Elastic Network Adapter では、AENQ 設定がサポートされていません。

サポートされていない AENQ のイベントを設定しようとしている

このエラーは、Elastic Network Adapter によりサポートされていない AENQ イベントグループを設定しようとしたことを示しています。

最適とは言えない構成に関する通知

ENA デバイスは、変更可能なドライバー内の最適ではない構成設定を検出します。デバイスは ENA ドライバーに通知し、コンソールに警告を記録します。次の例は、警告メッセージの形式を示しています。

```
Sub-optimal configuration notification code: 1. Refer to AWS ENA documentation for
 additional details and mitigation options.
```

次のリストは、通知コードの詳細と、最適ではない構成が検出された場合の推奨アクションを示しています。

- コード **1**: ワイド LLQ 構成の ENA Express は推奨されません。

  ENA Express ENI はワイド LLQ で設定されています。この構成は最適とは言えず、ENA Express のパフォーマンスに影響を与える可能性があります。ENA Express ENI を使用するときは、次のようにワイド LLQ 設定を無効にすることをお勧めします。

  ```
  sudo rmmod ena && sudo modprobe ena force_large_llq_header=0
  ```

  ENA Express の最適な構成の詳細については、「Linux インスタンスで ENA Express を使用してネットワークパフォーマンスを向上させる」を参照してください。
- コード **2**: Tx キューの深さが最適ではない ENA Express ENI は推奨されません

  ENA Express ENI が最適ではない Tx キューの深さで設定されています。この設定は、ENA Express のパフォーマンスに影響を与える可能性があります。ENA Express ENI を使用する際は、次のようにすべての Tx キューをネットワークインターフェイスの最大値に拡大することをお勧めします。

Tx キューの深さを最大化するには:

```
ethtool -g interface
```

Tx キューを最大の深さまで拡大するには :

```
ethtool -G interface tx depth
```

ENA Express の最適な構成の詳細については、「Linux インスタンスで ENA Express を使用してネットワークパフォーマンスを向上させる」を参照してください。

Linux ベースの Amazon EC2 インスタンスのネットワークレイテンシーを改善する

ネットワークレイテンシーとは、データのパケットが送信元から送信先に移動するまでにかかる時間です。ネットワークを介してデータを送信するアプリケーションは、快適なユーザーエクスペリエンスを提供するためにタイムリーな応答が不可欠です。ネットワークレイテンシーが長くなると、次のようなさまざまな問題が発生する可能性があります。

- ウェブページのロードが遅い
- ビデオストリーミングのタイムラグ
- オンラインリソースへのアクセスが難しい

このセクションでは、Linux で実行される Amazon EC2 インスタンスのネットワークレイテンシーを改善するために実行できる手順の概要を説明します。最適なレイテンシーを実現するには、以下の手順に従ってインスタンス、カーネル、および ENA ドライバーの設定を行います。その他の設定ガイダンスについては、GitHub の「ENA Linux ドライバーのベストプラクティスとパフォーマンス最適化ガイド」を参照してください。

> **Note**
>
> 手順と設定は、特定のネットワークハードウェア、インスタンスを起動した AMI、およびアプリケーションのユースケースによって若干異なる場合があります。変更を加える前に、ネットワークパフォーマンスを徹底的にテストおよびモニタリングして、望ましい結果が得られることを確認してください。

ネットワークホップを削減する

データパケットがルーター間を移動する (ホップする) たびにネットワークレイテンシーが増加します。通常、トラフィックが送信先に到達するには、複数のホップを経由する必要があります。Amazon EC2 インスタンスのネットワークホップを減らすには、次の 2 つの方法があります。

- クラスタープレイスメントグループ – クラスタープレイスメントグループを指定すると、Amazon EC2 は、同じアベイラビリティーゾーン (AZ) 内の物理的に互いに近いインスタンスをよりタイトなパッキングで起動します。グループ内のインスタンスが物理的に近接していることにより、高速接続を利用できるため、レイテンシーは低く、単一フローのスループットは高くなります。
- 専有ホスト – 専有ホストはお客様専用の物理サーバーです。専有ホストを使用すると、インスタンスを起動して同じ物理サーバー上で実行できます。同じ専有ホストで実行されるインスタンス間では、余分なネットワークホップなしで通信できます。

Linux カーネル設定

Linux カーネル設定では、ネットワークレイテンシーを増減できます。レイテンシー最適化の目標を達成するには、ワークロードの特定の要件に応じて Linux カーネル設定を微調整することが重要です。

Linux カーネルには、ネットワークレイテンシーを減らすのに役立つ設定オプションが多数あります。最も影響の大きいオプションは次のとおりです。

- ビジーポーリングモードを有効にする – ビジーポーリングモードを使用すると、ネットワーク受信パスのレーテンシーが減少します。ビジーポーリングモードを有効にすると、ソケットレイヤーコードはネットワークデバイスの受信キューを直接ポーリングできます。ビジーポーリングの欠点は、タイトなループで新しいデータをポーリングすることにより、ホストの CPU 使用率が高くなることです。すべてのインターフェイスでパケットを待機するマイクロ秒数を制御するグローバル設定は 2 つあります。

  `busy_read`

  ソケット読み込み時の低レイテンシーのビジーポーリングタイムアウト。これにより、ソケット層がデバイスキューのパケットを読み取るまでのマイクロ秒数を制御します。sysctl コマンドを使用して機能をグローバルに有効にするには、Linux カーネル組織は 50 マイクロ秒の値を推奨しています。詳細については、「Linux カーネルユーザーおよび管理者ガイド」の「busy_read」を参照してください。

```
$ sudo sysctl -w net.core.busy_read=50
```

busy_poll

ポーリングとセレクティングの低レイテンシーのビジーポーリングタイムアウト。これにより、イベント待ち時間のマイクロ秒数を制御します。推奨値は 50 ~ 100 マイクロ秒で、ポーリングするソケットの数によって異なります。追加するソケットが多いほど、数値は大きい必要があります。

```
$ sudo sysctl -w net.core.busy_poll=50
```

- CPU 電源状態の設定 (C ステート) – C ステートは非アクティブ時のコアのスリープレベルを制御します。C ステートを制御して、システムのレイテンシーとパフォーマンスを調整することができます。より深い C ステートでは、CPU は基本的に「スリープ」状態になり、起動してアクティブ状態に戻るまでリクエストに応答できません。コアをスリープ状態にするには時間がかかります。また、スリープ状態のコアによって、別のコアが高い周波数で動作するための余裕が生まれますが、そのスリープ状態にあるコアが再び稼働し処理を実行するのにも時間がかかります。

  例えば、ネットワークパケットの中断を処理するように割り当てられたコアがスリープ状態である場合、その中断の処理に遅延が生じる可能性があります。より深い C ステートを使用しないようにシステムを設定できます。ただし、この設定では、プロセッサの反応レイテンシーは短縮されますが、他のコアの Turbo Boost 用のヘッドルームも減少します。

  プロセッサーの反応レイテンシーを短縮するために、C ステートが深くなるのを制限できます。詳細については、「深い C ステートの制限による高パフォーマンスと低レイテンシー」を参照してください。

ENA ドライバー設定

ENA ネットワークドライバーを使用すると、インスタンスとネットワーク間の通信が可能になります。ドライバーはネットワークパケットを処理し、ネットワークスタックまたは Nitro Card に渡します。ネットワークパケットが受信されると、Nitro Card は割り込みを生成して CPU がソフトウェアにイベントを通知します。

割り込み

割り込みは、デバイスまたはアプリケーションがプロセッサに送信する信号です。割り込みは、イベントが発生したこと、または即時の注意を要する条件が満たされたことをプロセッサに通知

します。割り込みにより、ネットワークインターフェイスからのデータ受信、ハードウェアイベントの処理、他のデバイスからのリクエストの処理など、時間的制約のあるタスクを処理できます。

割り込みモデレーション

割り込みモデレーションは、割り込みを集約または遅延させることにより、デバイスが生成する割り込みの数を減らす手法です。割り込みモデレーションの目的は、多数の割り込みの処理に伴うオーバーヘッドを削減することによってシステムパフォーマンスを向上させることです。割り込みが多すぎると CPU 使用率が高くなり、スループットに悪影響を及ぼします。一方、割り込みが少なすぎると、レイテンシーが長くなります。

動的割り込みモデレーション

動的割り込みモデレーションは、現在のシステム負荷とトラフィックパターンに基づいて割り込みレートを動的に調整する、割り込みモデレーションの拡張形です。割り込みオーバーヘッドの削減と、1 秒あたりのパケット数 (つまり帯域幅) とのバランスをとることを目的としています。

> **Note**
>
> 動的割り込みモデレーションは、一部の AMI ではデフォルトで有効になっています (ただし、すべての AMI で有効または無効にできます)。

ネットワークレイテンシーを最小限に抑えるために、割り込みモデレーションを無効にする必要があることがあります。ただし、これによって割り込み処理のオーバーヘッドが増加する可能性もあります。レイテンシーの低減とオーバーヘッドの最小化とのバランス点を見つけることが重要です。ethtool コマンドは割り込みモデレーションの設定に役立ちます。デフォルトでは、rx-usecs は 20 に設定され、tx-usecs は 64 に設定されています。

現在の割り込み変更設定を取得するには、次のコマンドを使用します。

```
$ ethtool -c interface | egrep "rx-usecs:|tx-usecs:|Adaptive RX"
Adaptive RX: on  TX: off
rx-usecs: 20
tx-usecs: 64
```

割り込み変更および動的割り込みモデレーションを無効にするには、次のコマンドを使用します。

```
$ sudo ethtool -C interface adaptive-rx off rx-usecs 0 tx-usecs 0
```

Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) は、High Performance Computing (HPC) と機械学習アプリケーションを高速化するために Amazon EC2 インスタンスにアタッチできるネットワークデバイスです。EFA では、AWS クラウドが提供するスケーラビリティ、柔軟性、伸縮性により、オンプレミス HPC クラスターのアプリケーションパフォーマンスを実現できます。

EFA では、クラウドベースの HPC システムで従来使用されていた TCP トランスポートよりも低く、一貫性の高いレイテンシーを提供し、高いスループットが得られます。HPC と機械学習アプリケーションのスケーリングに不可欠なインスタンス間通信のパフォーマンスが向上します。既存の AWS ネットワークインフラストラクチャで動作するように最適化されており、アプリケーション要件に応じてスケーリングすることができます。

EFA は、Libfabric 1.7.0 と統合されており、HPC アプリケーション向けに Open MPI 5 以降とインテル MPI 2019 Update 5 以降をサポートし、さらに機械学習アプリケーション向けに NVIDIA Collective Communications Library (NCCL) をサポートしています。

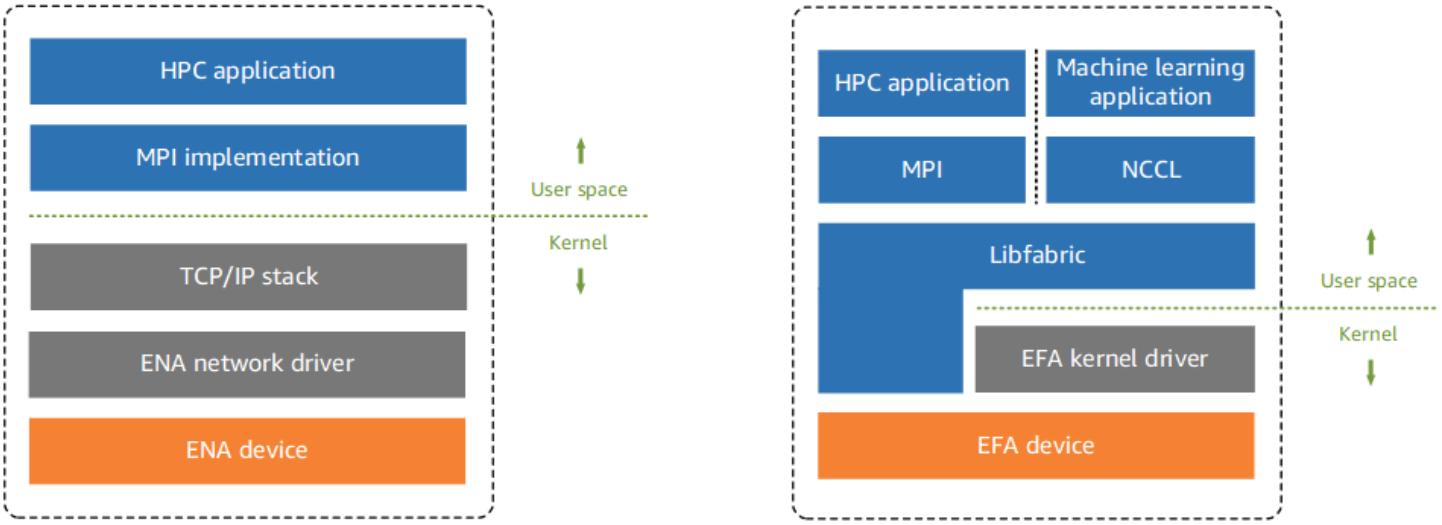
> **Note**
>
> EFAs の OS バイパス機能は、Windows インスタンスではサポートされていません。EFA を Windows インスタンスにアタッチした場合、インスタンスは、Elastic Network Adapter として動作し、EFA 機能は追加されません。

コンテンツ

- チェックサムを使用した EFA インストーラの検証

EFA の基本

EFA は、機能が追加された Elastic Network Adapter (ENA) です。ENA のすべての機能に OS バイパス機能が追加されています。OS バイパスは、HPC と機械学習アプリケーションがネットワークインターフェイスハードウェアと直接通信して、レイテンシーが低く、信頼性の高い転送機能を実現できるようにするアクセスモデルです。



Traditional HPC software stack in EC2

HPC software stack in EC2 with EFA

従来、HPC アプリケーションは、Message Passing Interface (MPI) を使用してシステムのネットワーク転送と通信していました。AWS クラウドでは、アプリケーションが MPI と通信することを意味します。MPI はオペレーティングシステムの TCP/IP スタックと ENA デバイスドライバーを使用して、インスタンス間のネットワーク通信を行います。

EFA の場合、HPC アプリケーションは MPI または NCCL を使用して Libfabric API と連携します。Libfabric API はオペレーティングシステムのカーネルをバイパスし、EFA デバイスと直接通信してパケットをネットワークに送ります。これにより、オーバーヘッドが削減され、HPC アプリケーションを効率的に実行できるようになります。

> **Note**
>
> Libfabric は、OpenFabrics Interface (OFI) フレームワークのコアコンポーネントで、OFI のユーザースペース API を定義およびエクスポートします。詳細については、Libfabric OpenFabricsウェブサイトを参照してください。

EFAs と ENA の違い

Elastic Network Adapters (ENA) は、VPC ネットワーキングをサポートするために必要な従来の IP ネットワーキング機能を提供します。EFAは、ENA と同じ従来の IP ネットワーキング機能すべてに加えて、OS バイパス機能をサポートしています。OS バイパスにより、HPC と機械学習アプリケーションはオペレーティングシステムのカーネルをバイパスして EFA デバイスと直接通信できます。

サポートされているインターフェイスとライブラリ

EFA は、以下のインターフェイスとライブラリをサポートしています。

- Open MPI 5 以降
- Graviton には、Open MPI 4.0 以降が推奨されます
- Intel MPI 2019 Update 5 以降
- NVIDIA Collective Communications Library (NCCL) 2.4.2 以降

サポートされるインスタンスタイプ

EFAs をサポートしているインスタンスタイプ:

- 凡用: m5dn.24xlarge | m5dn.metal | m5n.24xlarge | m5n.metal | m5zn.12xlarge | m5zn.metal | m6a.48xlarge | m6a.metal | m6i.32xlarge | m6i.metal | m6id.32xlarge | m6id.metal | m6idn.32xlarge | m6idn.metal | m6in.32xlarge | m6in.metal | m7a.48xlarge | m7a.metal-48xl | m7g.16xlarge | m7g.metal | m7gd.16xlarge | m7gd.metal | m7i.48xlarge | m7i.metal-48xl
- コンピューティング最適化: c5n.9xlarge | c5n.18xlarge | c5n.metal | c6a.48xlarge | c6a.metal | c6gn.16xlarge | c6i.32xlarge | c6i.metal | c6id.32xlarge | c6id.metal | c6in.32xlarge | c6in.metal | c7a.48xlarge | c7a.metal-48xl | c7g.16xlarge | c7g.metal | c7gd.16xlarge | c7gd.metal | c7gn.16xlarge | c7i.48xlarge | c7i.metal-48xl
- メモリ最適化: r5dn.24xlarge | r5dn.metal | r5n.24xlarge | r5n.metal | r6a.48xlarge | r6a.metal | r6i.32xlarge | r6i.metal | r6idn.32xlarge | r6idn.metal | r6in.32xlarge | r6in.metal | r6id.32xlarge | r6id.metal | r7a.48xlarge | r7a.metal-48xl | r7g.16xlarge | r7g.metal | r7gd.16xlarge | r7gd.metal | r7i.48xlarge | r7i.metal-48xl | r7iz.32xlarge | r7iz.metal-32xl | x2idn.32xlarge | x2idn.metal | x2iedn.32xlarge | x2iedn.metal | x2iezn.12xlarge | x2iezn.metal

- ストレージ最適化: i3en.12xlarge | i3en.24xlarge | i3en.metal | i4g.16xlarge | i4i.32xlarge | i4i.metal | im4gn.16xlarge
- 高速コンピューティング: dl1.24xlarge | dl2q.24xlarge | g4dn.8xlarge | g4dn.12xlarge | g4dn.16xlarge | g4dn.metal | g5.8xlarge | g5.12xlarge | g5.16xlarge | g5.24xlarge | g5.48xlarge | inf1.24xlarge | p3dn.24xlarge | p4d.24xlarge | p4de.24xlarge | p5.48xlarge | trn1.32xlarge | trn1n.32xlarge | vt1.24xlarge
- 高性能コンピューティング: hpc6a.48xlarge | hpc6id.32xlarge | hpc7a.12xlarge | hpc7a.24xlarge | hpc7a.48xlarge | hpc7a.96xlarge | hpc7g.4xlarge | hpc7g.8xlarge | hpc7g.16xlarge

特定のリージョンで EFA をサポートする利用可能なインスタンスタイプを確認するには

利用可能なインスタンスタイプは、リージョンごとに異なります。リージョンで EFA をサポートする使用可能なインスタンスタイプを確認するには、--region パラメーターを指定して describe-instance-types コマンドを使用します。結果を EFA をサポートするインスタンスタイプにスコープする --filters パラメーターと、出力を InstanceType の値にスコープする --query パラメーターを含めます。

```
aws ec2 describe-instance-types  --region us-east-1  --filters Name=network-info.efa-supported,Values=true  --query "InstanceTypes[*].[InstanceType]"  --output text | sort
```

サポートされるオペレーティングシステム

次のオペレーティングシステムは、Intel/AMD x86 ベースのインスタンスタイプを持つ EFA をサポートしています。

- Amazon Linux 2023
- Amazon Linux 2
- CentOS 7
- RHEL 7、8、および 9
- [Debian 10 と 11]
- Rocky Linux 8 および 9
- Ubuntu 20.04 および 22.04
- SUSE Linux Enterprise 15 SP2 以降
- OpenSUSE Leap 15.4 以降

> **Note**
>
> Ubuntu 20.04 では、dl1.24xlarge インスタンスと併用した場合、ピアダイレクトサポートがサポートされます。

次のオペレーティングシステムは、ARM ベース (Graviton) インスタンスタイプを持つ EFA をサポートしています。

- Amazon Linux 2023
- Amazon Linux 2
- RHEL 8/9 と Rocky Linux 8/9
- [Debian 10 と 11]
- Ubuntu 20.04 および 22.04
- SUSE Linux Enterprise 15 SP2 以降

EFA の制限事項

EFA には次の制限があります。

- すべての P4d および P5 インスタンスのタイプは、NVIDIA GPUDirect Remote Direct Memory Access (RDMA) をサポートします。
- P4d/P4de/DL1 インスタンスと他のインスタンスタイプ間の EFA トラフィックは、現在サポートされていません。
- 複数のネットワークカードをサポートするインスタンスタイプは、ネットワークカードごとに 1 つの EFA で設定できます。その他のサポートされているインスタンスタイプはすべて、インスタンスごとに 1 つの EFA のみをサポートしています。
- EFA がアタッチされている場合、c7g.16xlarge、m7g.16xlarge、r7g.16xlarge Dedicated Instances および Dedicated Hosts はサポートされません。
- EFA OS バイパストラフィックは、1 つのサブネットに制限されています。つまり、EFA トラフィックをサブネット間で送信することはできません。EFA の通常の IP トラフィックは、サブネット間で送信することができます。
- EFA OS バイパストラフィックは、ルーティングできません。EFA の通常の IP トラフィックは、引き続きルーティングできます。

- EFA は、セキュリティグループ自体との間のインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループのメンバーである必要があります。
- EFA は、AWS Outposts ではサポートされていません。

EFA 価格設定

EFA はオプションの Amazon EC2 ネットワーキング機能として利用でき、サポートされているどのインスタンスでも追加費用なしで有効にできます。

P5 インスタンスと EFA の使用を開始する

P5 インスタンスは、複数の EFA インターフェイスを使用して 3200 Gbps のネットワーク帯域幅を提供します。P5 インスタンスは 32 枚のネットワークカードをサポートします。P5 インスタンスの開始方法の詳細については、「P5 インスタンスの使用を開始する」を参照してください。

ネットワークカードごとに 1 つの EFA ネットワークインターフェイスを定義することをお勧めします。起動時にこれらのインターフェイスを設定するには、以下の設定をお勧めします。

- ネットワークインターフェイス `0` の場合、デバイスインデックス `0` を指定する
- 31 を介したネットワークインターフェイス 1 の場合、デバイスインデックス 1 を指定する

Amazon EC2 コンソールを使用している場合は、インスタンス起動ウィザードの [ネットワーク設定] セクションで [編集] を選択します。[高度なネットワーク設定] を展開し、[ネットワークインターフェイスを追加] を選択して、必要な数のネットワークインターフェイスを追加します。各ネットワークインターフェイスの [EFA] で、[有効化] を選択します。プライマリネットワークインターフェイスを除くすべてのネットワークインターフェイスの [デバイスインデックス] に、1 を指定します。必要に応じて、残りの設定を設定します。

AWS CLI を使用している場合、`--network-interfaces` に run-instances コマンドを使用し、必要な数のネットワークインターフェイスを指定します。各ネットワークインターフェイスの `InterfaceType` に `efa` を指定します。プライマリネットワークインターフェイスの `NetworkCardIndex` と `DeviceIndex` に `0` を指定します。残りのネットワークインターフェイスの `NetworkCardIndex` に、1 から 31 までの一意の値、`DeviceIndex` に 1 を指定します。

以下のコマンドスニペットの例は、32 の EFA ネットワークインターフェイスによるリクエストを示しています。

```
$ aws --region $REGION ec2 run-instances \
```

```
--instance-type p5.48xlarge \
--count 1 \
--key-name key_pair_name \
--image-id ami_id \
--network-interfaces
"NetworkCardIndex=0,DeviceIndex=0,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=1,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=2,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=3,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=4,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=5,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=6,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=7,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=8,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=9,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=ef
\

"NetworkCardIndex=10,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=11,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=12,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\
```

```
"NetworkCardIndex=13,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=14,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=15,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=16,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=17,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=18,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=19,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=20,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=21,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=22,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=23,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=24,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=25,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\

"NetworkCardIndex=26,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
\
```

```
 "NetworkCardIndex=27,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
 \
 "NetworkCardIndex=28,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
 \
 "NetworkCardIndex=29,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
 \
 "NetworkCardIndex=30,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
 \
 "NetworkCardIndex=31,DeviceIndex=1,Groups=security_group_id,SubnetId=subnet_id,InterfaceType=e
...
```

起動テンプレートを使用している場合は、起動テンプレートに必要な数のネットワークインターフェイスを指定します。各ネットワークインターフェイスの InterfaceType に efa を指定します。プライマリネットワークインターフェイスの NetworkCardIndex と DeviceIndex に 0 を指定します。残りのネットワークインターフェイスの NetworkCardIndex に、1 から 31 までの一意の値、DeviceIndex に 1 を指定します。次のスニペットは、設定可能な 32 のネットワークインターフェイスのうち 3 つのネットワークインターフェイスを使用した例を示しています。

```
"NetworkInterfaces":[
{
  "NetworkCardIndex":0,
  "DeviceIndex":0,
  "InterfaceType": "efa",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
},
{
  "NetworkCardIndex": 1,
  "DeviceIndex": 1,
  "InterfaceType": "efa",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
```

```
  "DeleteOnTermination":true
},
{
  "NetworkCardIndex": 2,
  "DeviceIndex": 1,
  "InterfaceType": "efa",
  "AssociatePublicIpAddress":false,
  "Groups":[
    "security_group_id"
  ],
  "DeleteOnTermination":true
}
...
```

複数のネットワークインターフェイスを持つ P5 インスタンスを起動する場合、パブリック IP アドレスを自動割り当てすることはできません。ただし、インターネット接続の起動後に、Elastic IP アドレスをプライマリネットワークインターフェイス (NetworkCardIndex=0、DeviceIndex=0) にアタッチすることはできます。Ubuntu 20.04 以降と Amazon Linux 2 以降はどちらも、上記に推奨されているようにインスタンスの起動時にインターネットトラフィックにプライマリネットワークインターフェイスを使用するように設定されています。

EFAと MPI の開始方法

本チュートリアルは、EFA と HPC ワークロードの MPI 対応インスタンスクラスターの起動に役立ちます。本チュートリアルでは、次の手順を実行します。

コンテンツ

ステップ 1: EFA 対応のセキュリティグループを準備する

EFA には、セキュリティグループ自体とのインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループが必要です。以下の手順では、セキュリティグループを作成します。このセキュリティグループでは、セキュリティグループ自体とのすべてのインバウンドおよびアウトバウンドのトラフィックと、SSH 接続用の任意の IPv4 アドレスからのインバウンド SSH トラフィックを許可します。

> **Important**
>
> このセキュリティグループは、テストのみを目的としています。本番環境では、コンピュータの IP アドレスやローカルネットワークの IP アドレスの範囲など、接続元の IP アドレスからのトラフィックのみを許可するインバウンド SSH ルールを作成することをお勧めします。

その他のシナリオについては、さまざまなユースケースのセキュリティグループのルールを参照してください。

EFA 対応のセキュリティグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create security group] (セキュリティグループの作成) を選択します。
3. [Create security group] (セキュリティグループの作成) ウィンドウで、以下を行います。
   a. [セキュリティグループ名] に、`EFA-enabled security group` のような、分かりやすいセキュリティグループ名を入力します。
   b. (オプション) [説明] に、セキュリティグループの簡単な説明を入力します。
   c. [VPC] で、EFA 対応のインスタンスを起動する VPC を選択します。
   d. [Create Security Group] を選択します。
4. 作成したセキュリティグループを選択し、[Details] (詳細) タブで [Security group ID] (セキュリティグループ ID) をコピーします。

5. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit inbound rules] (インバウンドルールの編集) の順に選択し、次の手順を実行します。

   a. [Add rule] を選択します。

   b. [Type] で、[All traffic] を選択します。

   c. [Source type] (送信元タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。

   d. [ルールを追加] を選択します。

   e. タイプ] で SSH] を選択します。

   f. [Source type] (ソースタイプ) で、[Anywhere-IPv4] を選択します。

   g. [Save Rules] (ルールの保存) を選択します。

6. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールの編集) の順に選択し、次の手順を実行します。

   a. [Add rule] を選択します。

   b. [Type] で、[All traffic] を選択します。

   c. [Destination type] (送信先タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。

   d. [Save Rules] (ルールの保存) を選択します。

ステップ 2: 一時インスタンスを作成する

EFA ソフトウェアコンポーネントのインストールおよび設定に使用する一時インスタンスを起動します。このインスタンスを使用して、EFA 対応のインスタンスを起動する EFA 対応の AMI を作成します。

一時インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前は、リソースタグとしてインスタンスに割り当てられます (Name=*EFA-instance*)。

4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、サポートされるオペレーティングシステムを選択します。
5. [Instance type] (インスタンスタイプ) セクションで、サポートされているインスタンスタイプを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択します。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択します。

   c. [Advanced network configuration] (高度なネットワーク設定) セクションを展開し、[Elastic Fabric Adapter] の [Enable] (有効) を選択します。
8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. 右側の [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

ステップ 3: EFA ソフトウェアをインストールする

一時インスタンスで EFA をサポートするために必要な EFA 対応のカーネル、EFA ドライバー、Libfabric、および Open MPI スタックをインストールします。

このステップは、EFA で Open MPI、Intel MPI、または Open MPI と Intel MPI のどれを使用するかによって異なります。

EFA ソフトウェアをインストールするには

1. 起動したインスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。
2. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。このプロセスには数分かかることがあります。

   - Amazon Linux 2023、Amazon Linux 2、RHEL 7/8/9、CentOS 7、Rocky Linux 8/9

```
$ sudo yum update -y
```

- Ubuntu 20.04/22.04 および Debian 10/11

```
$ sudo apt-get update && sudo apt-get upgrade -y
```

- SUSE Linux Enterprise

```
$ sudo zypper update -y
```

3. インスタンスを再起動して、そのインスタンスに再接続します。
4. EFA ソフトウェアのインストールファイルをダウンロードします。ソフトウェアのインストールファイルは、圧縮された tar (`.tar.gz`) ファイルにパッケージ化されています。次のコマンドを使用して、安定している最新バージョンをダウンロードします。

```
$ curl -O https://efa-installer.amazonaws.com/aws-efa-installer-1.31.0.tar.gz
```

前述のコマンドのバージョン番号を `latest` に置き換えることで最新バージョンを取得することもできます。

5. (オプション) EFA tarball (`.tar.gz`) ファイルの認証と完全性を検証します。

ソフトウェア発行元の ID を検証し、発行後にファイルの改変や破損がないことを確認するために、これを行うことをお勧めします。tar ファイルを検証しない場合は、この手順をスキップします。

> **Note**
>
> 代わりに、MD5 または SHA256 チェックサムを使用して tar ファイルを検証する場合は、チェックサムを使用した EFA インストーラの検証を参照してください。

a. パブリック GPG キーをダウンロードして、キーリングにインポートします。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer.key && gpg --
import aws-efa-installer.key
```

コマンドはキーの値を返します。次の手順で必要になるため、キーの値を書きとめておきます。

b. GPG キーのフィンガープリントを検証します。次のコマンドを実行し、前のステップで作成したキーの値を指定します。

```
$ gpg --fingerprint key_value
```

コマンドは、4E90 91BC BB97 A96B 26B1 5E59 A054 80B1 DD2D 3CCC と同じフィンガープリントを返します。フィンガープリントが一致しない場合は、EFA インストールスクリプトを実行せず、AWS Support にお問い合わせください。

c. 署名ファイルをダウンロードし、EFA tar ファイルの署名を検証します。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer-1.31.0.tar.gz.sig
 && gpg --verify ./aws-efa-installer-1.31.0.tar.gz.sig
```

出力例を次に示します。

```
gpg: Signature made Wed 29 Jul 2020 12:50:13 AM UTC using RSA key ID DD2D3CCC
gpg: Good signature from "Amazon EC2 EFA <ec2-efa-maintainers@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 4E90 91BC BB97 A96B 26B1  5E59 A054 80B1 DD2D 3CCC
```

結果に Good signature が含まれ、フィンガープリントが前のステップで返されたフィンガープリントと一致する場合は、次のステップに進みます。そうでない場合は、EFA インストールスクリプトを実行せず、AWS Support にお問い合わせください。

6. 圧縮された .tar.gz ファイルからファイルを展開し、展開されたディレクトリに移動します。

```
$ tar -xf aws-efa-installer-1.31.0.tar.gz && cd aws-efa-installer
```

7. EFA ソフトウェアをインストールします。使用するユースケースに応じて、以下のいずれかを実行します。

> **Note**
>
> EFA は SUSE Linux での NVIDIA GPUDirect をサポートしていません。SUSE Linux を使用している場合は、さらに kmod のインストールを防止する --skip-kmod オプショ

ンを指定する必要があります。デフォルトでは、SUSE Linux はツリー外のカーネルモジュールを許可しません。

Open MPI and Intel MPI

EFA で Open MPI と Intel MPI を使用する場合は、EFA ソフトウェアと共に Libfabric と Open MPI をインストールする必要があります。また、「ステップ 5: (オプション) インテル MPI をインストールする」を完了する必要があります。

Libfabric および Open MPI と共に EFA ソフトウェアをインストールするには、次のコマンドを実行します。

> **Note**
>
> EFA 1.30.0 からは、オープン MPI 4 と Open MPI 5 の両方がデフォルトでインストールされます。任意で、インストールする Open MPI のバージョンを指定できます。Open MPI 4 のみをインストールするには、`--mpi=openmpi4` を含めてください。Open MPI 5 のみをインストールするには、`--mpi=openmpi5` を含めてください。両方をインストールする場合は、`--mpi` オプションを省略してください。

```
$ sudo ./efa_installer.sh -y
```

Libfabric は `/opt/amazon/efa` にインストールされます。Open MPI 4 は `/opt/amazon/openmpi` にインストールされます。Open MPI 5 は `/opt/amazon/openmpi5` にインストールされます。

Open MPI only

EFA で Open MPI のみを使用する場合は、EFA ソフトウェアと共に Libfabric と Open MPI をインストールする必要があります。また、「ステップ 5: (オプション) インテル MPI をインストールする」はスキップできます。Libfabric および Open MPI と共に EFA ソフトウェアをインストールするには、次のコマンドを実行します。

> **Note**
>
> EFA 1.30.0 からは、オープン MPI 4 と Open MPI 5 の両方がデフォルトでインストールされます。任意で、インストールする Open MPI のバージョンを指定できます。Open MPI 4 のみをインストールするには、`--mpi=openmpi4` を含めてください。Open MPI 5 のみをインストールするには、`--mpi=openmpi5` を含めてください。両方をインストールする場合は、`--mpi` オプションを省略してください。

```
$ sudo ./efa_installer.sh -y
```

Libfabric は `/opt/amazon/efa` にインストールされます。Open MPI 4 は `/opt/amazon/openmpi` にインストールされます。Open MPI 5 は `/opt/amazon/openmpi5` にインストールされます。

Intel MPI only

EFA で Intel MPI のみを使用する場合は、Libfabric および Open MPI を使用せずに EFA ソフトウェアをインストールできます。この場合、Intel MPI は埋め込まれている Libfabric を使用します。これを選択した場合は、「ステップ 5: (オプション) インテル MPI をインストールする」を完了する必要があります。

Libfabric および Open MPI を使用せずに EFA ソフトウェアをインストールするには、次のコマンドを実行します。

```
$ sudo ./efa_installer.sh -y --minimal
```

8. EFA インストーラーでインスタンスの再起動を求めるメッセージが表示された場合は、再起動してからインスタンスに再接続します。それ以外の場合は、インスタンスからログアウトし、再度ログインしてインストールを完了します。

ステップ 4: (オプション) Open MPI 5 を有効にする

> **Note**
>
> このステップは、Open MPI 5 を使用する場合にのみ実行します。

EFA 1.30.0 からは、オープン MPI 4 と Open MPI 5 の両方がデフォルトでインストールされます。あるいは、Open MPI 4 または Open MPI 5 のみをインストールするように選択することもできます。

「ステップ 3: EFA ソフトウェアをインストールする」で Open MPI 5 のインストールを選択し、これを使用する場合は、次の手順を実行して有効にする必要があります。

Open MPI 5 を有効にするには

1. Open MPI 5 を PATH 環境変数に追加します。

   ```
   $ module load openmpi5
   ```

2. Open MPI 5 の使用が有効になっていることを確認します。

   ```
   $ which mpicc
   ```

   このコマンドは Open MPI 5 のインストールディレクトリ - `/opt/amazon/openmpi5` を返すはずです。

3. (オプション) インスタンスが起動するたびに Open MPI 5 が PATH 環境変数に追加されるようにするには、次の操作を行います。

   bash shell

   `module load openmpi5` を `/home/username/.bashrc` と `/home/username/.bash_profile` に追加します。

   csh and tcsh shells

   `module load openmpi5` を `/home/username/.cshrc` に追加します。

Open MPI 5 を PATH 環境変数から削除する必要がある場合は、次のコマンドを実行して、シェルスタートアップスクリプトからそのコマンドを削除します。

```
$ module unload openmpi5
```

ステップ 5: (オプション) インテル MPI をインストールする

> **Important**
>
> このステップは、Intel MPI を使用する場合にのみ実行します。Open MPI のみを使用する場合は、このステップをスキップしてください。

Intel MPI を使用するには、追加のインストールと環境変数設定が必要です。

前提条件

以下のステップは、sudo アクセス許可を持つユーザーが実行してください。

Intel MPI をインストールするには

1. インテル MPI のインストールスクリプトをダウンロードするには、次の手順を実行します。
   a. インテルのウェブサイトにアクセスします。
   b. ウェブページの「Intel MPI Library」(インテル MPI ライブラリ) セクションで、Intel MPI Library for Linux Offline インストーラのリンクを選択します。
2. 前のステップでダウンロードしたインストールスクリプトを実行します。

```
$ sudo bash installation_script_name.sh
```

3. インストーラで、[Accept & install] (承諾してインストール) を選択します。
4. インテル Improvement Program を読み、適切なオプションを選択してから、[Begin Installation] (インストールを開始) を選択します。
5. インストールが完了したら、[Close] を選択します。
6. デフォルトでは、インテル MPI は埋め込まれている (内部) Libfabric を使用します。代わりに、EFA インストーラに同梱されている Libfabric を使用するようにインテル MPI を設定できます。通常、EFA インストーラには、インテル MPI よりも新しいバージョンの Libfabric が同梱されています。場合によっては、EFA インストーラに同梱されている Libfabric は、インテル MPI よりもパフォーマンスが高い場合があります。EFA インストーラに同梱されている Libfabric を使用するようにインテル MPI を設定するには、シェルに応じて次のいずれかを実行します。

bash shells

次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export I_MPI_OFI_LIBRARY_INTERNAL=0
```

csh and tcsh shells

次のステートメントを /home/*username*/.cshrc に追加します。

```
setenv I_MPI_OFI_LIBRARY_INTERNAL 0
```

7. 次のソースコマンドをシェルスクリプトに追加して、インストールディレクトリから vars.sh スクリプトを読み込み、インスタンスが開始するたびにコンパイラ環境をセットアップします。使用するシェルに応じて、以下のいずれかを実行します。

   bash shells

   次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   source /opt/intel/oneapi/mpi/latest/env/vars.sh
   ```

   csh and tcsh shells

   次のステートメントを /home/*username*/.cshrc に追加します。

   ```
   source /opt/intel/oneapi/mpi/latest/env/vars.csh
   ```

8. デフォルトでは、設定ミスにより EFA が使用できない場合、インテル MPI はデフォルトで TCP/IP ネットワークスタックを使用するため、アプリケーションのパフォーマンスが低下する可能性があります。I_MPI_OFI_PROVIDER を efa に設定することでこれを防ぐことができます。これにより、EFA が利用できない場合、インテル MPI は次のエラーで失敗します。

   ```
   Abort (XXXXXX) on node 0 (rank 0 in comm 0): Fatal error in PMPI_Init: OtherMPI
    error,
   MPIR_Init_thread (XXX).....:
   MPID_Init (XXXX)..............:
   ```

```
MPIDI_OFI_mpi_init_hook (XXXX):
open_fabric (XXXX).............:
find_provider (XXXX)..........:
OFI fi_getinfo() failed (ofi_init.c:2684:find_provider:
```

使用するシェルに応じて、以下のいずれかを実行します。

bash shells

次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export I_MPI_OFI_PROVIDER=efa
```

csh and tcsh shells

次のステートメントを /home/*username*/.cshrc に追加します。

```
setenv I_MPI_OFI_PROVIDER efa
```

9. デフォルトでは、インテル MPI はデバッグ情報を出力しません。さまざまな詳細レベルを指定して、デバッグ情報を制御できます。可能な値は次のとおりです (提供される詳細の量の順): `0` (デフォルト)、1、2、3、4、5。レベル 1 以上は `libfabric version` と `libfabric provider` を出力します。インテル MPI が内部 Libfabric を使用しているか、または EFA インストーラに同梱されている Libfabric を使用しているかを確認するために `libfabric version` を使用します。内部 Libfabric を使用している場合、バージョンのサフィックスは `impi` です。インテル MPI が EFA または TCP/IP ネットワークを使用しているかどうかを確認するために `libfabric provider` を使用します。EFA を使用している場合、値は `efa` です。TCP/IP を使用している場合、値は `tcp;ofi_rxm` です。

   デバッグ情報を有効にするには、使用するシェルに応じて、次のいずれかを実行します。

   bash shells

   次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   export I_MPI_DEBUG=value
   ```

csh and tcsh shells

次のステートメントを /home/*username*/.cshrc に追加します。

```
setenv I_MPI_DEBUG value
```

10. デフォルトでは、インテル MPI はノード内通信にオペレーティングシステムの共有メモリ (shm) を使用し、ノード間通信にのみ Libfabric (ofi) を使用します。通常、この設定は、最高のパフォーマンスを提供します。ただし、場合によっては、インテル MPI shm ファブリックによって特定のアプリケーションが無期限にハングすることがあります。

    この問題を解決するために、インテル MPI がノード内通信とノード間通信の両方に Libfabric を使用するように強制できます。これを実行するには、使用するシェルに応じて、次のいずれかを実行します。

    bash shells

    次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

    ```
    export I_MPI_FABRICS=ofi
    ```

    csh and tcsh shells

    次のステートメントを /home/*username*/.cshrc に追加します。

    ```
    setenv I_MPI_FABRICS ofi
    ```

    > **Note**
    >
    > EFA Libfabric プロバイダーは、オペレーティングシステムの共有メモリをノード内通信に使用します。これは、I_MPI_FABRICS を ofi に設定すると、デフォルトの shm:ofi 設定と同様のパフォーマンスが得られることを意味します。

11. インスタンスからログアウトしてからログインし直します。

Intel MPI が不要になった場合は、シェル起動スクリプトから環境変数を削除してください。

ステップ 6: ptrace 保護を無効にする

HPC アプリケーションのパフォーマンスを向上させるために、Libfabric は、プロセスが同じインスタンスで実行されている場合、プロセス間通信にインスタンスのローカルメモリを使用します。

共有メモリ機能では、ptrace 保護ではサポートされない Cross-Memory Attach (CMA) が使用されます。Ubuntu など、ptrace 保護がデフォルトで有効になっている Linux ディストリビューションを使用している場合は、無効にする必要があります。Linux ディストリビューションで ptrace 保護がデフォルトで有効になっていない場合は、このステップをスキップします。

ptrace 保護を無効にするには

次のいずれかを行ってください。

- テストのために ptrace 保護を一時的に無効にするには、次のコマンドを実行します。

```
$ sudo sysctl -w kernel.yama.ptrace_scope=0
```

- ptrace 保護を完全に無効にするには、`kernel.yama.ptrace_scope = 0` を `/etc/sysctl.d/10-ptrace.conf` に追加してインスタンスを再起動します。

ステップ 7。インストールを確認する

インストールが正常に完了したことを確認するには

1. MPI が正常にインストールされていることを確認するには、次のコマンドを実行します。

```
$ which mpicc
```

   - Open MPI の場合、返されるパスには `/opt/amazon/` が含まれている必要があります。
   - Intel MPI の場合、返されるパスには `/opt/intel/` が含まれている必要があります。期待どおりの出力が得られない場合は、Intel MPI `vars.sh` スクリプトをソースとしていることを確認してください。

2. EFA ソフトウェアコンポーネントと Libfabric が正常にインストールされたことを確認するには、以下のコマンドを実行します。

```
$ fi_info -p efa -t FI_EP_RDM
```

コマンドによって、Libfabric の EFA インターフェイスに関する情報が返ります。以下の例は、コマンド出力を示しています。

```
provider: efa
    fabric: EFA-fe80::94:3dff:fe89:1b70
    domain: efa_0-rdm
    version: 2.0
    type: FI_EP_RDM
    protocol: FI_PROTO_EFA
```

ステップ 8: HPC アプリケーションをインストールする

HPC アプリケーションを一時インスタンスにインストールします。インストール手順は、特定の HPC アプリケーションによって異なります。詳細については、「Amazon Linux インスタンスでのソフトウェアの管理」を参照してください。

> **Note**
>
> インストール手順については、HPC アプリケーションのドキュメントを参照してください。

ステップ 9: EFA 対応の AMI を作成する

必要なソフトウェアコンポーネントのインストール後、EFA 対応のインスタンスの起動に再利用できる AMI を作成します。

一時インスタンスから AMI を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 作成した一時インスタンスを選択し、[アクション]、[イメージ]、[イメージの作成] の順に選択します。
4. [イメージの作成] で、次を行います。
   a. [イメージ名] に、の分かりやすい AMI 名を入力します。
   b. (オプション) [イメージの説明] に、AMI の簡単な説明を入力します。
   c. [イメージを作成] を選択します。

5. ナビゲーションペインで [AMIs] を選択します。
6. リストで作成した AMI を探します。ステータスが `pending` から `available` に変わるまで待ってから、次のステップに進みます。

ステップ 10: クラスタープレイスメントグループで EFA 対応のインスタンスを作成する

ステップ 7 で作成した EFA 対応の AMI と、ステップ 1 で作成した EFA 対応のセキュリティグループを使用して、EFA 対応のインスタンスをクラスタープレイスメントグループ内で起動します。

> **Note**
>
> - EFA 対応のインスタンスをクラスターのプレイスメントグループに起動することは絶対的な要件ではありません。ただし、EFA 対応のインスタンスは、1 つのアベイラビリティーゾーン内の低レイテンシーグループに起動されるため、クラスタープレイスメントグループで実行することをお勧めします。
> - クラスターのインスタンスをスケールするときにキャパシティを使用できるようにするには、クラスタープレイスメントグループのキャパシティ予約を作成します。詳細については、クラスタープレイスメントグループでのキャパシティ予約 を参照してください。

一時インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前は、リソースタグとしてインスタンスに割り当てられます (Name=*`EFA-instance`*)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、[My AMIs] (マイ AMI) をクリックし、前のステップで作成した AMI を選択します。
5. [Instance type] (インスタンスタイプ) セクションで、サポートされているインスタンスタイプを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。

7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択します。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択します。

   c. [Advanced network configuration] (高度なネットワーク設定) セクションを展開し、[Elastic Fabric Adapter] の [Enable] (有効) を選択します。

8. (オプション) [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。

9. [Advanced details] (高度な詳細) セクションの [Placement group name] (プレイスメントグループ名) で、インスタンスを起動するクラスタープレイスメントグループを選択します。新しいクラスタープレイスメントグループを作成する必要がある場合は、[Create new placement group] (新しいプレイスメントグループの作成) を選択します。

10. 右側の [Summary] (サマリー) パネルで、[Number of instances] (インスタンス数) に、起動する EFA 対応のインスタンスの数を入力し、[Launch Instance] (インスタンスの起動) を選択します。

ステップ 11: 一時インスタンスを終了する

この時点で、起動した一時インスタンスは不要になります。インスタンスを終了して、料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 作成した一時インスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了] の順に選択します。
4. 確認を求めるメッセージが表示されたら、[終了] を選択します。

ステップ 12: パスワードレス SSH を有効にする

クラスター内のすべてのインスタンスでアプリケーションを実行できるようにするには、リーダーノードからメンバーノードへのパスワードなしの SSH アクセスを有効にする必要があります。リー

ダーノードは、アプリケーションを実行するインスタンスです。クラスター内の残りのインスタンスはメンバーノードです。

クラスター内のインスタンス間でパスワードなしの SSH を有効にするには

1. クラスター内の 1 つのインスタンスをリーダーノードとして選択し、それに接続します。
2. リーダーノード上で strictHostKeyChecking を無効にし ForwardAgent を有効にします。任意のテキストエディタを使用して ~/.ssh/config ファイルを開き、以下を追加します。

   ```
   Host *
       ForwardAgent yes
   Host *
       StrictHostKeyChecking no
   ```

3. RSA キーペアを生成します。

   ```
   $ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
   ```

   キーペアは、$HOME/.ssh/ ディレクトリで作成されます。
4. リーダーノードのプライベートキーの許可を変更します。

   ```
   $ chmod 600 ~/.ssh/id_rsa
   chmod 600 ~/.ssh/config
   ```

5. 任意のテキストエディタで ~/.ssh/id_rsa.pub を開き、キーをコピーします。
6. クラスター内の各メンバーノードについて、次の操作を行います。

   a. インスタンスに接続します。

   b. 任意のテキストエディタで ~/.ssh/authorized_keys を開き、前にコピーしたパブリックキーを追加します。
7. パスワードレス SSH が正常に機能していることをテストするには、リーダーノードに接続して、次のコマンドを実行します。

   ```
   $ ssh member_node_private_ip
   ```

   キーまたはパスワードの入力を求められずに、メンバーノードに接続できるはずです。

EFAとNCCL の開始方法

NVIDIA Collective Communications Library (NCCL) は、単一のノードまたは複数のノードの複数の GPU のための集合的な標準コミュニケーションルーチンのライブラリです。NCCL は、各種の機械学習のワークロードをサポートするために、EFA、Libfabric、MPI と共に使用できます。詳細については、NCCL のウェブサイトを参照してください。

> **Note**
>
> - EFA を持つ NCCL は、`p3dn.24xlarge`、`p4d.24xlarge`、`p5.48xlarge` でのみサポートされています。
> - NCCL EFA 以降のみが 2.4.2 でサポートされています。

以下のチュートリアルは、機械学習のワークロードの EFA と NCCL 対応のインスタンスクラスターの起動に役立ちます。

- ベース AMI の使用
- AWS 深層学習 AMI の使用

ベース AMI の使用

次の手順で、いずれかのサポートされているベースオペレーティングシステムの AMI を使用して、Elastic Fabric Adapter を開始できます。

> **Note**
>
> - `p3dn.24xlarge`、`p4d.24xlarge` および `p5.48xlarge` インスタンスタイプのみがサポートされています。
> - Amazon Linux 2、RHEL 7/8/9、CentOS 7、Rocky Linux 8/9、Ubuntu 20.04/22.04 ベース AMI のみがサポートされています。

コンテンツ

ステップ 1: EFA 対応のセキュリティグループを準備する

EFA には、セキュリティグループ自体とのインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループが必要です。以下の手順では、セキュリティグループを作成します。このセキュリティグループでは、セキュリティグループ自体とのすべてのインバウンドおよびアウトバウンドのトラフィックと、SSH 接続用の任意の IPv4 アドレスからのインバウンド SSH トラフィックを許可します。

> **Important**
>
> このセキュリティグループは、テストのみを目的としています。本番環境では、コンピュータの IP アドレスやローカルネットワークの IP アドレスの範囲など、接続元の IP アドレスからのトラフィックのみを許可するインバウンド SSH ルールを作成することをお勧めします。

その他のシナリオについては、さまざまなユースケースのセキュリティグループのルールを参照してください。

EFA 対応のセキュリティグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create security group] (セキュリティグループの作成) を選択します。
3. [Create security group] (セキュリティグループの作成) ウィンドウで、以下を行います。
   a. [セキュリティグループ名] に、`EFA-enabled security group` のような、分かりやすいセキュリティグループ名を入力します。
   b. (オプション) [説明] に、セキュリティグループの簡単な説明を入力します。
   c. [VPC] で、EFA 対応のインスタンスを起動する VPC を選択します。
   d. [Create Security Group] を選択します。
4. 作成したセキュリティグループを選択し、[Details] (詳細) タブで [Security group ID] (セキュリティグループ ID) をコピーします。
5. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit inbound rules] (インバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択します。
   b. [Type] で、[All traffic] を選択します。
   c. [Source type] (送信元タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
   d. [ルールを追加] を選択します。
   e. タイプ] で SSH] を選択します。
   f. [Source type] (ソースタイプ) で、[Anywhere-IPv4] を選択します。
   g. [Save Rules] (ルールの保存) を選択します。
6. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択します。
   b. [Type] で、[All traffic] を選択します。
   c. [Destination type] (送信先タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
   d. [Save Rules] (ルールの保存) を選択します。

ステップ 2: 一時インスタンスを作成する

EFA ソフトウェアコンポーネントのインストールおよび設定に使用する一時インスタンスを起動します。このインスタンスを使用して、EFA 対応のインスタンスを起動する EFA 対応の AMI を作成します。

一時インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前は、リソースタグとしてインスタンスに割り当てられます (Name=*EFA-instance*)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、サポートされるオペレーティングシステムを選択します。
5. [インスタンスタイプ] セクションで、`p3dn.24xlarge`、`p4d.24xlarge` または `p5.48xlarge` のいずれかを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。
   a. [サブネット] で、インスタンスを起動するサブネットを選択します。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。
   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択します。
   c. [Advanced network configuration] (高度なネットワーク設定) セクションを展開し、[Elastic Fabric Adapter] の [Enable] (有効) を選択します。
8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。

> **Note**
>
> Nvidia CUDA ツールキットには、追加の 10 ~ 20 GiB のストレージをプロビジョニングする必要があります。十分な量のストレージをプロビジョニングしないと、Nvidia ド

ライバーと CUDA ツールキットをインストールしようとしたときに、`insufficient disk space` エラーが発生します。

9. 右側の [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

ステップ 3: Nvidia GPU ドライバー、Nvidia CUDA ツールキットおよび cuDNN をインストールする

Amazon Linux 2

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

```
$ sudo yum upgrade -y && sudo reboot
```

インスタンスの再起動後に、再接続します。

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

```
$ sudo yum groupinstall 'Development Tools' -y
```

3. nouveau オープンソースドライバーを無効にします。

   a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo yum install -y wget kernel-devel-$(uname -r) kernel-headers-$(uname -r)
```

   b. nouveau 拒否リストファイルに `/etc/modprobe.d/blacklist.conf` を追加します。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
```

```
blacklist rivatv
EOF
```

c. grub ファイルに GRUB_CMDLINE_LINUX="rdblacklist=nouveau" を追加し、Grub 設定を再構成します。

```
$ echo 'GRUB_CMDLINE_LINUX="rdblacklist=nouveau"' | sudo tee -a /etc/
default/grub \
&& sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

4. インスタンスを再起動して、そのインスタンスに再接続します。
5. 必要なリポジトリを準備する

a. DKMS 用 EPEL リポジトリをインストールし、Linux ディストリビューションのオプションリポジトリを有効にします。

```
$ sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-7.noarch.rpm
```

b. CUDA リポジトリのパブリック GPG キーをインストールします。

```
$ distribution='rhel7'
```

c. CUDA ネットワークリポジトリを設定し、リポジトリキャッシュを更新します。

```
$ ARCH=$( /bin/arch ) \
&& sudo yum-config-manager --add-repo http://developer.download.nvidia.com/
compute/cuda/repos/$distribution/${ARCH}/cuda-$distribution.repo \
&& sudo yum clean expire-cache
```

d. (カーネルバージョン 5.10 のみ) 以下の手順は、Amazon Linux 2 をカーネルバージョン 5.10 で使用している場合にのみ実行します。Amazon Linux 2 をカーネルバージョン 4.12 で使用している場合は、以下の手順をスキップします。カーネルバージョンを確認するには、uname -r を実行します。

i. /etc/dkms/nvidia.conf という名前で Nvidia ドライバ設定ファイルを作成します。

```
$ sudo mkdir -p /etc/dkms \
```

```
&& echo "MAKE[0]=\"'make' -j2 module SYSSRC=\${kernel_source_dir}
 IGNORE_XEN_PRESENCE=1 IGNORE_PREEMPT_RT_PRESENCE=1 IGNORE_CC_MISMATCH=1
 CC=/usr/bin/gcc10-gcc\"" | sudo tee /etc/dkms/nvidia.conf
```

ii. (p4d.24xlarge と p5.48xlarge のみ) NVIDIA ドライバー設定ファイルをコピーします。

```
$ sudo cp /etc/dkms/nvidia.conf /etc/dkms/nvidia-open.conf
```

6. NVIDIA GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

   - p3dn.24xlarge

   ```
   $ sudo yum clean all \
   && sudo yum -y install kmod-nvidia-latest-dkms nvidia-driver-latest-dkms \
   && sudo yum -y install cuda-drivers-fabricmanager cuda libcudnn8-devel
   ```

   - p4d.24xlarge および p5.48xlarge

   ```
   $ sudo yum clean all \
   && sudo yum -y install kmod-nvidia-open-dkms nvidia-driver-latest-dkms \
   && sudo yum -y install cuda-drivers-fabricmanager cuda libcudnn8-devel
   ```

7. インスタンスを再起動して、そのインスタンスに再接続します。

8. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager は、NV Switch Management に必要です。

   ```
   $ sudo systemctl enable nvidia-fabricmanager && sudo systemctl start nvidia-
   fabricmanager
   ```

9. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。

   - bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   export PATH=/usr/local/cuda/bin:$PATH
   export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
   lib64:$LD_LIBRARY_PATH
   ```

- tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

```
setenv PATH=/usr/local/cuda/bin:$PATH
setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

10. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

```
$ nvidia-smi -q | head
```

このコマンドは、Nvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

CentOS 7

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

```
$ sudo yum upgrade -y && sudo reboot
```

インスタンスの再起動後に、再接続します。

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

```
$ sudo yum groupinstall 'Development Tools' -y \
&& sudo yum install -y tar bzip2 make automake pciutils elfutils-libelf-devel
 libglvnd-devel iptables firewalld vim bind-utils
```

3. Nvidia GPU ドライバを使用するには、まず、nouveau オープンソースドライバを無効にする必要があります。

a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo yum install -y wget kernel-devel-$(uname -r) kernel-headers-$(uname -
r)
```

b. nouveau 拒否リストファイルに /etc/modprobe.d/blacklist.conf を追加します。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

c. 任意のテキストエディタを使用して /etc/default/grub ファイルを開き、以下を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

d. Grub 設定を再構築します。

```
$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

4. インスタンスを再起動して、そのインスタンスに再接続します。
5. NVIDIA GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

a. DKMS 用 EPEL リポジトリをインストールし、Linux ディストリビューションのオプションリポジトリを有効にします。

```
$ sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-7.noarch.rpm
```

b. CUDA リポジトリのパブリック GPG キーをインストールします。

```
$ distribution='rhel7'
```

c. CUDA ネットワークリポジトリを設定し、リポジトリキャッシュを更新します。

```
$ ARCH=$( /bin/arch ) \
&& sudo yum-config-manager --add-repo http://developer.download.nvidia.com/
compute/cuda/repos/$distribution/${ARCH}/cuda-$distribution.repo \
&& sudo yum clean expire-cache
```

d. NVIDIA、CUDA ドライバー、および cuDNN をインストールします。

```
$ sudo yum clean all \
&& sudo yum -y install cuda-drivers-fabricmanager cuda libcudnn8-devel
```

6. インスタンスを再起動して、そのインスタンスに再接続します。
7. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager は、NV Switch Management に必要です。

```
$ sudo systemctl start nvidia-fabricmanager \
&& sudo systemctl enable nvidia-fabricmanager
```

8. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。

   - bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

   ```
   export PATH=/usr/local/cuda/bin:$PATH
   export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
   lib64:$LD_LIBRARY_PATH
   ```

   - tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

   ```
   setenv PATH=/usr/local/cuda/bin:$PATH
   setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
   lib64:$LD_LIBRARY_PATH
   ```

9. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

```
$ nvidia-smi -q | head
```

このコマンドは、Nvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

RHEL 7/8/9 and Rocky Linux 8/9

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

```
$ sudo yum upgrade -y && sudo reboot
```

   インスタンスの再起動後に、再接続します。

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

```
$ sudo yum groupinstall 'Development Tools' -y
```

3. Nvidia GPU ドライバを使用するには、まず、nouveau オープンソースドライバを無効にする必要があります。

   a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo yum install -y wget kernel-devel-$(uname -r) kernel-headers-$(uname -r)
```

   b. nouveau 拒否リストファイルに /etc/modprobe.d/blacklist.conf を追加します。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   c. 任意のテキストエディタを使用して /etc/default/grub ファイルを開き、以下を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

d. Grub 設定を再構築します。

```
$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

4. インスタンスを再起動して、そのインスタンスに再接続します。
5. NVIDIA GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

a. DKMS用 EPELリポジトリをインストールし、Linux ディストリビューションのオプションリポジトリを有効にします。

- RHEL 7

```
$ sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-7.noarch.rpm
```

- RHEL 8 と Rocky Linux 8/9

```
$ sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-8.noarch.rpm
```

- RHEL 9

```
$ sudo yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-
latest-9.noarch.rpm
```

b. CUDA リポジトリのパブリック GPG キーをインストールします。

```
$ distribution=$(. /etc/os-release;echo $ID`rpm -E "%{?rhel}%{?fedora}"`)
```

c. CUDA ネットワークリポジトリを設定し、リポジトリキャッシュを更新します。

```
$ ARCH=$( /bin/arch ) \
&& sudo yum-config-manager --add-repo http://developer.download.nvidia.com/
compute/cuda/repos/$distribution/${ARCH}/cuda-$distribution.repo \
&& sudo yum clean expire-cache
```

d. NVIDIA、CUDA ドライバー、および cuDNN をインストールします。

```
$ sudo yum clean all \
&& sudo yum -y install cuda-drivers-fabricmanager cuda libcudnn8-devel
```

6. インスタンスを再起動して、そのインスタンスに再接続します。
7. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager は、NV Switch Management に必要です。

```
$ sudo systemctl start nvidia-fabricmanager \
&& sudo systemctl enable nvidia-fabricmanager
```

8. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。
   - bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export PATH=/usr/local/cuda/bin:$PATH
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

   - tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

```
setenv PATH=/usr/local/cuda/bin:$PATH
setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

9. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

```
$ nvidia-smi -q | head
```

このコマンドは、Nvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

Ubuntu 20.04/22.04

NVIDIA GPU ドライバー、NVIDIA CUDA ツールキットおよび cuDNN をインストールするには

1. すべてのソフトウェアパッケージが最新の状態であることを確認するため、インスタンスでソフトウェアの更新を実行します。

```
$ sudo apt-get update && sudo apt-get upgrade -y
```

2. Nvidia GPU ドライバと Nvidia CUDA ツールキットをインストールするために必要なユーティリティをインストールします。

```
$ sudo apt-get update && sudo apt-get install build-essential -y
```

3. Nvidia GPU ドライバを使用するには、まず、nouveau オープンソースドライバを無効にする必要があります。

   a. 必要なユーティリティ、および現在実行しているカーネルのバージョン用のカーネルヘッダーパッケージをインストールします。

```
$ sudo apt-get install -y gcc make linux-headers-$(uname -r)
```

   b. nouveau 拒否リストファイルに /etc/modprobe.d/blacklist.conf を追加します。

```
$ cat << EOF | sudo tee --append /etc/modprobe.d/blacklist.conf
blacklist vga16fb
blacklist nouveau
blacklist rivafb
blacklist nvidiafb
blacklist rivatv
EOF
```

   c. 任意のテキストエディタを使用して /etc/default/grub ファイルを開き、以下を追加します。

```
GRUB_CMDLINE_LINUX="rdblacklist=nouveau"
```

   d. Grub 設定を再構築します。

```
$ sudo update-grub
```

4. インスタンスを再起動して、そのインスタンスに再接続します。
5. CUDA リポジトリを追加し、Nvidia GPU ドライバー、NVIDIA CUDA ツールキット、および cuDNN をインストールします。

   - p3dn.24xlarge

```
$ sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/7fa2af80.pub \
```

```
&& wget -O /tmp/deeplearning.deb http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/nvidia-machine-learning-repo-
ubuntu2004_1.0.0-1_amd64.deb \
&& sudo dpkg -i /tmp/deeplearning.deb \
&& wget -O /tmp/cuda.pin https://developer.download.nvidia.com/compute/cuda/
repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin \
&& sudo mv /tmp/cuda.pin /etc/apt/preferences.d/cuda-repository-pin-600 \
&& sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/
compute/cuda/repos/ubuntu2004/x86_64/3bf863cc.pub \
&& sudo add-apt-repository 'deb http://developer.download.nvidia.com/compute/
cuda/repos/ubuntu2004/x86_64/ /' \
&& sudo apt update \
&& sudo apt install nvidia-dkms-535 \
&& sudo apt install -o Dpkg::Options::='--force-overwrite' cuda-drivers-535
 cuda-toolkit-12-3 libcudnn8 libcudnn8-dev -y
```

- p4d.24xlarge および p5.48xlarge

```
$ sudo apt-key adv --fetch-keys http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/7fa2af80.pub \
&& wget -O /tmp/deeplearning.deb http://developer.download.nvidia.com/compute/
machine-learning/repos/ubuntu2004/x86_64/nvidia-machine-learning-repo-
ubuntu2004_1.0.0-1_amd64.deb \
&& sudo dpkg -i /tmp/deeplearning.deb \
&& wget -O /tmp/cuda.pin https://developer.download.nvidia.com/compute/cuda/
repos/ubuntu2004/x86_64/cuda-ubuntu2004.pin \
&& sudo mv /tmp/cuda.pin /etc/apt/preferences.d/cuda-repository-pin-600 \
&& sudo apt-key adv --fetch-keys https://developer.download.nvidia.com/
compute/cuda/repos/ubuntu2004/x86_64/3bf863cc.pub \
&& sudo add-apt-repository 'deb http://developer.download.nvidia.com/compute/
cuda/repos/ubuntu2004/x86_64/ /' \
&& sudo apt update \
&& sudo apt install nvidia-kernel-open-535 \
&& sudo apt install -o Dpkg::Options::='--force-overwrite' cuda-drivers-535
 cuda-toolkit-12-3 libcudnn8 libcudnn8-dev -y
```

6. インスタンスを再起動して、そのインスタンスに再接続します。
7. (p4d.24xlarge および p5.48xlarge のみ) NVIDIA Fabric Manager をインストールします。

   a. 前の手順でインストールした Nvidia カーネルモジュールのバージョンと一致する Nvidia Fabric Manager のバージョンをインストールする必要があります。

Nvidia カーネルモジュールのバージョンを確認するには、次のコマンドを実行します。

```
$ cat /proc/driver/nvidia/version | grep "Kernel Module"
```

以下は出力例です。

```
NVRM version: NVIDIA UNIX x86_64 Kernel Module  450.42.01  Tue Jun 15
 21:26:37 UTC 2021
```

上記の例では、メジャーバージョン 450 のカーネルモジュールがインストールされました。これは、Nvidia Fabric Manager のバージョン 450 をインストールする必要があることを意味します。

b. Nvidia Fabric Manager をインストールする 次のコマンドを、前の手順で識別されたメジャーバージョンを指定して実行します。

```
$ sudo apt install -o Dpkg::Options::='--force-overwrite' nvidia-
fabricmanager-major_version_number
```

例えば、メジャーバージョン 450 のカーネルモジュールがインストールされた場合、以下のコマンドを使用して、一致するバージョンの Nvidia Fabric Manager をインストールします。

```
$ sudo apt install -o Dpkg::Options::='--force-overwrite' nvidia-
fabricmanager-450
```

c. サービスを開始し、インスタンスの起動時に自動的に起動することを確認します。NVIDIA Fabric Manager は、NV Switch Management に必要です。

```
$ sudo systemctl start nvidia-fabricmanager && sudo systemctl enable nvidia-
fabricmanager
```

8. インスタンスが起動するたびに CUDA パスが設定されていることを確認します。

- bash シェルの場合、次のステートメントを /home/*username*/.bashrc と /home/*username*/.bash_profile に追加します。

```
export PATH=/usr/local/cuda/bin:$PATH
```

```
export LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

- tcsh シェルの場合、次の文を /home/*username*/.cshrc に追加します。

```
setenv PATH=/usr/local/cuda/bin:$PATH
setenv LD_LIBRARY_PATH=/usr/local/cuda/lib64:/usr/local/cuda/extras/CUPTI/
lib64:$LD_LIBRARY_PATH
```

9. 以下のコマンドを実行して、Nvidia GPU ドライバが機能することを確認します。

```
$ nvidia-smi -q | head
```

このコマンドは、Nvidia GPU、Nvidia GPU ドライバ、Nvidia CUDA ツールキットの情報を返します。

ステップ 4: GDRCopy をインストールする

GDRCopy をインストールして Libfabric のパフォーマンスを向上させます。GDRCopy の詳細については、「GDRCopy レポジトリ」を参照してください。

Amazon Linux 2, CentOS 7, RHEL 7/8/9, and Rocky Linux 8/9

GDRCopy をインストールするには

1. 必要な依存ファイルをインストールします。

```
$ sudo yum -y install dkms rpm-build make check check-devel subunit subunit-
devel
```

2. GDRCopy パッケージをダウンロードして解凍します。

```
$ wget https://github.com/NVIDIA/gdrcopy/archive/refs/tags/v2.4.tar.gz \
&& tar xf v2.4.tar.gz ; cd gdrcopy-2.4/packages
```

3. GDRCopy RPM パッケージをビルドします。

```
$ CUDA=/usr/local/cuda ./build-rpm-packages.sh
```

4. GDRCopy RPM パッケージをインストールします。

```
$ sudo rpm -Uvh gdrcopy-kmod-2.4-1dkms.noarch*.rpm \
&& sudo rpm -Uvh gdrcopy-2.4-1.x86_64*.rpm \
&& sudo rpm -Uvh gdrcopy-devel-2.4-1.noarch*.rpm
```

Ubuntu 20.04/22.04

GDRCopy をインストールするには

1. 必要な依存ファイルをインストールします。

```
$ sudo apt -y install build-essential devscripts debhelper check libsubunit-dev
 fakeroot pkg-config dkms
```

2. GDRCopy パッケージをダウンロードして解凍します。

```
$ wget https://github.com/NVIDIA/gdrcopy/archive/refs/tags/v2.4.tar.gz \
&& tar xf v2.4.tar.gz \
&& cd gdrcopy-2.4/packages
```

3. GDRCopy RPM パッケージをビルドします。

```
$ CUDA=/usr/local/cuda ./build-deb-packages.sh
```

4. GDRCopy RPM パッケージをインストールします。

```
$ sudo dpkg -i gdrdrv-dkms_2.4-1_amd64.*.deb \
&& sudo dpkg -i libgdrapi_2.4-1_amd64.*.deb \
&& sudo dpkg -i gdrcopy-tests_2.4-1_amd64.*.deb \
&& sudo dpkg -i gdrcopy_2.4-1_amd64.*.deb
```

ステップ 5: EFA ソフトウェアをインストールする

一時インスタンスで EFA をサポートするために必要な EFA 対応のカーネル、EFA ドライバー、Libfabric、および Open MPI スタックをインストールします。

EFA ソフトウェアをインストールするには

1. 起動したインスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。

2. EFA ソフトウェアのインストールファイルをダウンロードします。ソフトウェアのインストールファイルは、圧縮された tar (`.tar.gz`) ファイルにパッケージ化されています。次のコマンドを使用して、安定している最新バージョンをダウンロードします。

```
$ curl -O https://efa-installer.amazonaws.com/aws-efa-installer-1.31.0.tar.gz
```

前述のコマンドのバージョン番号を `latest` に置き換えることで最新バージョンを取得することもできます。

3. (オプション) EFA tarball (`.tar.gz`) ファイルの認証と完全性を検証します。

ソフトウェア発行元の ID を検証し、発行後にファイルの改変や破損がないことを確認するために、これを行うことをお勧めします。tar ファイルを検証しない場合は、この手順をスキップします。

> **Note**
>
> 代わりに、MD5 または SHA256 チェックサムを使用して tar ファイルを検証する場合は、チェックサムを使用した EFA インストーラの検証を参照してください。

a. パブリック GPG キーをダウンロードして、キーリングにインポートします。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer.key && gpg --
import aws-efa-installer.key
```

コマンドはキーの値を返します。次の手順で必要になるため、キーの値を書きとめておきます。

b. GPG キーのフィンガープリントを検証します。次のコマンドを実行し、前のステップで作成したキーの値を指定します。

```
$ gpg --fingerprint key_value
```

コマンドは、`4E90 91BC BB97 A96B 26B1 5E59 A054 80B1 DD2D 3CCC` と同じフィンガープリントを返します。フィンガープリントが一致しない場合は、EFA インストールスクリプトを実行せず、AWS Support にお問い合わせください。

c. 署名ファイルをダウンロードし、EFA tar ファイルの署名を検証します。

```
$ wget https://efa-installer.amazonaws.com/aws-efa-installer-1.31.0.tar.gz.sig
 && gpg --verify ./aws-efa-installer-1.31.0.tar.gz.sig
```

出力例を次に示します。

```
gpg: Signature made Wed 29 Jul 2020 12:50:13 AM UTC using RSA key ID DD2D3CCC
gpg: Good signature from "Amazon EC2 EFA <ec2-efa-maintainers@amazon.com>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 4E90 91BC BB97 A96B 26B1  5E59 A054 80B1 DD2D 3CCC
```

結果に Good signature が含まれ、フィンガープリントが前のステップで返されたフィンガープリントと一致する場合は、次のステップに進みます。そうでない場合は、EFA インストールスクリプトを実行せず、AWS Support にお問い合わせください。

4. 圧縮された .tar.gz ファイルからファイルを展開し、展開されたディレクトリに移動します。

```
$ tar -xf aws-efa-installer-1.31.0.tar.gz && cd aws-efa-installer
```

5. EFA ソフトウェアのインストールスクリプトを実行します。

> **Note**
>
> EFA 1.30.0 からは、オープン MPI 4 と Open MPI 5 の両方がデフォルトでインストールされます。Open MPI 5 が必要でない限り、Open MPI 4 のみをインストールすることをお勧めします。以下のコマンドは Open MPI 4 のみをインストールします。Open MPI 4 と Open MPI 5 をインストールする場合は、--mpi=openmpi4 を削除してください。

```
$ sudo ./efa_installer.sh -y --mpi=openmpi4
```

Libfabric は、/opt/amazon/efa ディレクトリにインストールされているのに対し、Open MPI は /opt/amazon/openmpi ディレクトリにインストールされています。

6. EFA インストーラーでインスタンスの再起動を求めるメッセージが表示された場合は、再起動してからインスタンスに再接続します。それ以外の場合は、インスタンスからログアウトし、再度ログインしてインストールを完了します。

7. EFA ソフトウェアコンポーネントが正常にインストールされたことを確認します。

```
$ fi_info -p efa -t FI_EP_RDM
```

コマンドによって、Libfabric の EFA インターフェイスに関する情報が返ります。以下の例は、コマンド出力を示しています。

- 単一のネットワークインターフェイスを持つ p3dn.24xlarge

```
provider: efa
fabric: EFA-fe80::94:3dff:fe89:1b70
domain: efa_0-rdm
version: 2.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
```

- 複数のネットワークインターフェイスを持つ p4d.24xlarge および p5.48xlarge

```
provider: efa
fabric: EFA-fe80::c6e:8fff:fef6:e7ff
domain: efa_0-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::c34:3eff:feb2:3c35
domain: efa_1-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::c0f:7bff:fe68:a775
domain: efa_2-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
provider: efa
fabric: EFA-fe80::ca7:b0ff:fea6:5e99
domain: efa_3-rdm
version: 111.0
type: FI_EP_RDM
protocol: FI_PROTO_EFA
```

ステップ 6: NCCL をインストールする

NCCL をインストールします。NCCL に関する詳細については、NCCL repositoryを参照してください。

NCCL をインストールするには

1. /opt ディレクトリに移動します。

```
$ cd /opt
```

2. 公式の NCCL リポジトリをインスタンスにクローンし、ローカルのクローンされたリポジトリに移動します。

```
$ sudo git clone https://github.com/NVIDIA/nccl.git && cd nccl
```

3. NCCL を構築およびインストールし、CUDA インストールディレクトリを指定します。

```
$ sudo make -j src.build CUDA_HOME=/usr/local/cuda
```

ステップ 7: aws-ofi-nccl プラグインをインストールする

aws-ofi-nccl プラグインは、NCCL の接続目的のトランスポート API を、Llibfabric の接続がなく信頼性の高いインターフェイスにマップします。これにより、NCCL ベースのアプリケーションの実行中に、Libfabric をネットワークプロバイダーとして使用できます。aws-ofi-nccl プラグインに関する詳細については、aws-ofi-nccl リポジトリを参照してください。

aws-ofi-nccl プラグインをインストールするには

1. ホームディレクトリに移動します。

```
$ cd $HOME
```

2. (Amazon Linux 2 と Ubuntu のみ) 必要なユーティリティをインストールします。

   - Amazon Linux 2

```
$ sudo yum install hwloc-devel
```

   - Ubuntu 20.04

```
$ sudo apt-get install libhwloc-dev
```

3. aws-ofi-nccl プラグインファイルをダウンロードします。ファイルは、圧縮された tar (.tar.gz) にパッケージ化されています。

```
$ wget https://github.com/aws/aws-ofi-nccl/releases/download/v1.8.1-aws/aws-ofi-nccl-1.8.1-aws.tar.gz
```

4. 圧縮された .tar.gz ファイルからファイルを展開し、展開されたディレクトリに移動します。

```
$ tar -xf aws-ofi-nccl-1.8.1-aws.tar.gz && cd aws-ofi-nccl-1.8.1-aws
```

5. make ファイルを生成するには、configure スクリプトを実行し、MPI、Libfabric、NCCL、CUDA インストールディレクトリを指定します。

```
$ ./configure --prefix=/opt/aws-ofi-nccl --with-mpi=/opt/amazon/openmpi \
--with-libfabric=/opt/amazon/efa \
--with-cuda=/usr/local/cuda \
--enable-platform-aws
```

6. Open MPI ディレクトリを PATH 変数に追加します。

```
$ export PATH=/opt/amazon/openmpi/bin/:$PATH
```

7. aws-ofi-nccl プラグインをインストールします。

```
$ make && sudo make install
```

ステップ 8: NCCL テストをインストールする

NCCL テストをインストールします。NCCL テストでは、NCCL が適切にインストールされていることを確認し、想定どおりに機能していることを確認できます。NCCL テストに関する詳細については、nccl-tests リポジトリを参照してください。

NCCL テストをインストールするには

1. ホームディレクトリに移動します。

```
$ cd $HOME
```

2. 公式の nccl-tests リポジトリをインスタンスにクローンし、ローカルのクローンされたリポジトリに移動します。

```
$ git clone https://github.com/NVIDIA/nccl-tests.git && cd nccl-tests
```

3. Libfabric ディレクトリを LD_LIBRARY_PATH 変数に追加します。

   - Amazon Linux、Amazon Linux 2、RHEL、Rocky Linux 8/9、CentOS

```
$ export LD_LIBRARY_PATH=/opt/amazon/efa/lib64:$LD_LIBRARY_PATH
```

   - Ubuntu

```
$ export LD_LIBRARY_PATH=/opt/amazon/efa/lib:$LD_LIBRARY_PATH
```

4. NCCL テストをインストールし、MPI、NCCL、CUDA インストールディレクトリを指定します。

```
$ make MPI=1 MPI_HOME=/opt/amazon/openmpi NCCL_HOME=/opt/nccl/build CUDA_HOME=/usr/
local/cuda
```

ステップ 9: EFA と NCCL の設定をテストする

テストを実行し、EFA と NCCL に一時インスタンスが適切に設定されていることを確認します。

EFA と NCCL 設定をテストするには

1. テストを実行するホストを指定するホストファイルを作成します。以下のコマンドは、インスタンス自体へのリファレンスを含む my-hosts と呼ばれるホストファイルを作成します。

   IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
 "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/
meta-data/local-ipv4 >> my-hosts
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/local-ipv4 >> my-hosts
```

2. テストを実行し、ホストファイル (--hostfile) と使用する GPU の数 (-n) を指定します。以下のコマンドは、インスタンス自体の 8 つの GPU で all_reduce_perf テストを実行し、以下の環境変数を指定します。

   - FI_EFA_USE_DEVICE_RDMA=1 — (p4d.24xlarge のみ) 片側転送および両側転送にデバイスの RDMA 機能を使用します。
   - NCCL_DEBUG=INFO – 詳細なデバッグ出力を有効にします。また、テストの開始時に NCCL バージョンのみをプリントするために VERSION を指定したり、エラーメッセージのみを受信するために WARN を指定したりすることもできます。

   NCCL テスト引数に関する詳細は、公式の nccl-tests リポジトリのNCCL Tests READMEを参照してください。

   - p3dn.24xlarge

   ```
   $ /opt/amazon/openmpi/bin/mpirun \
       -x LD_LIBRARY_PATH=/opt/nccl/build/lib:/usr/local/cuda/lib64:/opt/amazon/efa/lib:/opt/amazon/openmpi/lib:/opt/aws-ofi-nccl/lib:$LD_LIBRARY_PATH \
       -x NCCL_DEBUG=INFO \
       --hostfile my-hosts -n 8 -N 8 \
       --mca pml ^cm --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to none \
       $HOME/nccl-tests/build/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n 100
   ```

   - p4d.24xlarge および p5.48xlarge

   ```
   $ /opt/amazon/openmpi/bin/mpirun \
       -x FI_EFA_USE_DEVICE_RDMA=1 \
       -x LD_LIBRARY_PATH=/opt/nccl/build/lib:/usr/local/cuda/lib64:/opt/amazon/efa/lib:/opt/amazon/openmpi/lib:/opt/aws-ofi-nccl/lib:$LD_LIBRARY_PATH \
       -x NCCL_DEBUG=INFO \
       --hostfile my-hosts -n 8 -N 8 \
       --mca pml ^cm --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to none \
   ```

```
$HOME/nccl-tests/build/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n 100
```

3. NCCL_DEBUG ログが出力されるときに、EFA が NCCL の基盤となるプロバイダーとしてアクティブであることを確認できます。

```
ip-192-168-2-54:14:14 [0] NCCL INFO NET/OFI Selected Provider is efa*
```

p4d.24xlarge インスタンスの使用時に、次の追加情報が表示されます。

```
ip-192-168-2-54:14:14 [0] NCCL INFO NET/OFI Running on P4d platform, Setting
 NCCL_TOPO_FILE environment variable to /home/ec2-user/install/plugin/share/aws-
ofi-nccl/xml/p4d-24xl-topo.xml
```

ステップ 10: 機械学習アプリケーションをインストールする

機械学習アプリケーションを一時インスタンスにインストールします。インストール手順は、それぞれの機械学習アプリケーションによって異なります。Linux インスタンスへのソフトウェアのインストールの詳細については、Linux インスタンスでのソフトウェアの管理を参照してください。

> **Note**
>
> インストール手順については、機械学習アプリケーションのドキュメントを参照してください。

ステップ 11: EFA および NCCL 対応 AMI を作成する

必要なソフトウェアコンポーネントのインストール後、EFA 対応のインスタンスの起動に再利用できる AMI を作成します。

一時インスタンスから AMI を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 作成した一時インスタンスを選択し、[アクション]、[イメージ]、[イメージの作成] の順に選択します。
4. [イメージの作成] で、次を行います。

   a. [イメージ名] に、の分かりやすい AMI 名を入力します。

b. (オプション) [イメージの説明] に、AMI の簡単な説明を入力します。

c. [イメージを作成] を選択します。

5. ナビゲーションペインで [AMIs] を選択します。

6. リストで作成した AMI を探します。ステータスが `pending` から `available` に変わるまで待ってから、次のステップに進みます。

ステップ 12: 一時インスタンスを終了する

この時点で、起動した一時インスタンスは不要になります。インスタンスを終了して、料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[インスタンス] を選択します。

3. 作成した一時インスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了] の順に選択します。

4. 確認を求めるメッセージが表示されたら、[終了] を選択します。

ステップ 13: クラスタープレイスメントグループに EFA および NCCL 対応インスタンスを起動する

前に作成した EFA 対応の AMI と EFA 対応のセキュリティグループを使用して、EFA および NCCL 対応のインスタンスをクラスタープレイスメントグループ内で起動します。

> **Note**
>
> - EFA 対応のインスタンスをクラスターのプレイスメントグループに起動することは絶対的な要件ではありません。ただし、EFA 対応のインスタンスは、1 つのアベイラビリティーゾーン内の低レイテンシーグループに起動されるため、クラスタープレイスメントグループで実行することをお勧めします。
> - クラスターのインスタンスをスケールするときにキャパシティを使用できるようにするには、クラスタープレイスメントグループのキャパシティ予約を作成します。詳細については、クラスタープレイスメントグループでのキャパシティ予約 を参照してください。

New console

一時インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前は、リソースタグとしてインスタンスに割り当てられます (`Name=EFA-instance`)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、[My AMIs] (マイ AMI) をクリックし、前のステップで作成した AMI を選択します。
5. [Instance type] (インスタンスタイプ) セクションで、`p3dn.24xlarge` または `p4d.24xlarge` のいずれかを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。
   a. [サブネット] で、インスタンスを起動するサブネットを選択します。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。
   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択します。
   c. [Advanced network configuration] (高度なネットワーク設定) セクションを展開し、[Elastic Fabric Adapter] の [Enable] (有効) を選択します。
8. (オプション) [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. [Advanced details] (高度な詳細) セクションの [Placement group name] (プレイスメントグループ名) で、インスタンスを起動するクラスタープレイスメントグループを選択します。新しいクラスタープレイスメントグループを作成する必要がある場合は、[Create new placement group] (新しいプレイスメントグループの作成) を選択します。
10. 右側の [Summary] (サマリー) パネルで、[Number of instances] (インスタンス数) に、起動する EFA 対応のインスタンスの数を入力し、[Launch Instance] (インスタンスの起動) を選択します。

Old console

EFA および NCCL 対応のインスタンスをクラスタープレイスメントグループに起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [インスタンスの作成] を選択します。
3. [AMI の選択] ページで、[マイ AMI] を選択し、前に作成した AMI を見つけて、[選択] をクリックします。
4. [インスタンスタイプの選択] ページで [p3dn.24xlarge] を選択し、[次へ: インスタンスの詳細の設定] を選択します。
5. [インスタンスの詳細設定] ページで、以下を行います。
   a. [インスタンス数] に、起動する EFA および NCCL 対応のインスタンスの数を入力します。
   b. [ネットワーク] および [サブネット] で、インスタンスを起動する VPC およびサブネットを選択します。
   c. [プレイスメントグループ] で、[インスタンスをプレイスメントグループに追加します] チェックボックスをオンにします。
   d. [プレイスメントグループ名] で、[新しいプレイスメントグループに追加する] チェックボックスをオンにし、分かりやすいプレイスメントグループ名を入力します。次に、[プレイスメントグループ戦略] で [クラスター] を選択します。
   e. [EFA] で、[有効化] を選択します。
   f. [ネットワークインターフェイス] セクションの [eth0] で、[新しいネットワークインターフェイス] を選択します。必要に応じて、プライマリ IPv4 アドレスと 1 つ以上のセカンダリ IPv4 アドレスを指定できます。関連付けられている IPv6 CIDR ブロックを持つサブネットにインスタンスを起動する場合は、必要に応じて、プライマリ IPv6 アドレスと 1 つ以上のセカンダリ IPv6 アドレスを指定できます。
   g. [次の手順: ストレージの追加] を選択します。
6. [ストレージの追加] ページで、AMI が指定するボリューム (ルートデバイスボリュームなど) に加えて、インスタンスにアタッチするボリュームを指定します。次に、[次の手順: タグの追加] を選択します。
7. [Add Tags] ページで、ユーザーフレンドリーな名前などを使ってインスタンスのタグを指定し、[Next: Configure Security Group] を選択します。
8. [セキュリティグループの設定] ページの [セキュリティグループの割り当て] で、[既存のセキュリティグループの選択] を選択し、前に作成したセキュリティグループを選択します。

9. [Review and Launch] を選択します。
10. [インスタンス作成の確認] ページで設定を確認し、[起動] を選択してキーペアを選択し、インスタンスを起動します。

ステップ 14: パスワードレス SSH を有効にする

クラスター内のすべてのインスタンスでアプリケーションを実行できるようにするには、リーダーノードからメンバーノードへのパスワードなしの SSH アクセスを有効にする必要があります。リーダーノードは、アプリケーションを実行するインスタンスです。クラスター内の残りのインスタンスはメンバーノードです。

クラスター内のインスタンス間でパスワードなしの SSH を有効にするには

1. クラスター内の 1 つのインスタンスをリーダーノードとして選択し、それに接続します。
2. リーダーノード上で `strictHostKeyChecking` を無効にし `ForwardAgent` を有効にします。任意のテキストエディタを使用して `~/.ssh/config` ファイルを開き、以下を追加します。

```
Host *
    ForwardAgent yes
Host *
    StrictHostKeyChecking no
```

3. RSA キーペアを生成します。

```
$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
```

キーペアは、`$HOME/.ssh/` ディレクトリで作成されます。

4. リーダーノードのプライベートキーの許可を変更します。

```
$ chmod 600 ~/.ssh/id_rsa
chmod 600 ~/.ssh/config
```

5. 任意のテキストエディタで `~/.ssh/id_rsa.pub` を開き、キーをコピーします。
6. クラスター内の各メンバーノードについて、次の操作を行います。
   a. インスタンスに接続します。
   b. 任意のテキストエディタで `~/.ssh/authorized_keys` を開き、前にコピーしたパブリックキーを追加します。

7. パスワードレス SSH が正常に機能していることをテストするには、リーダーノードに接続して、次のコマンドを実行します。

```
$ ssh member_node_private_ip
```

キーまたはパスワードの入力を求められずに、メンバーノードに接続できるはずです。

AWS 深層学習 AMI の使用

以下の手順は、以下の AWS Deep Learning AMI のいずれかを開始するのに役立ちます。

- Deep Learning AMI (Amazon Linux 2)
- Deep Learning AMI (Ubuntu 20.04)

詳細については、AWS Deep Learning AMI ユーザーガイドを参照してください。

> **Note**
>
> `p3dn.24xlarge` および `p4d.24xlarge` インスタンスタイプのみがサポートされています。

コンテンツ

ステップ 1: EFA 対応のセキュリティグループを準備する

EFA には、セキュリティグループ自体とのインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループが必要です。以下の手順では、セキュリティグループを作成します。このセキュリティグループでは、セキュリティグループ自体とのすべてのインバウンドおよびアウトバウンドのトラフィックと、SSH 接続用の任意の IPv4 アドレスからのインバウンド SSH トラフィックを許可します。

> **Important**
>
> このセキュリティグループは、テストのみを目的としています。本番環境では、コンピュータの IP アドレスやローカルネットワークの IP アドレスの範囲など、接続元の IP アドレスからのトラフィックのみを許可するインバウンド SSH ルールを作成することをお勧めします。

その他のシナリオについては、さまざまなユースケースのセキュリティグループのルールを参照してください。

EFA 対応のセキュリティグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Security Groups] (セキュリティグループ) を選択して、[Create security group] (セキュリティグループの作成) を選択します。
3. [Create security group] (セキュリティグループの作成) ウィンドウで、以下を行います。
   a. [セキュリティグループ名] に、`EFA-enabled security group` のような、分かりやすいセキュリティグループ名を入力します。
   b. (オプション) [説明] に、セキュリティグループの簡単な説明を入力します。
   c. [VPC] で、EFA 対応のインスタンスを起動する VPC を選択します。
   d. [Create Security Group] を選択します。
4. 作成したセキュリティグループを選択し、[Details] (詳細) タブで [Security group ID] (セキュリティグループ ID) をコピーします。
5. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit inbound rules] (インバウンドルールの編集) の順に選択し、次の手順を実行します。
   a. [Add rule] を選択します。

b. [Type] で、[All traffic] を選択します。
c. [Source type] (送信元タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
d. [ルールを追加] を選択します。
e. タイプ] で SSH] を選択します。
f. [Source type] (ソースタイプ) で、[Anywhere-IPv4] を選択します。
g. [Save Rules] (ルールの保存) を選択します。

6. セキュリティグループが選択された状態で、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールの編集) の順に選択し、次の手順を実行します。

a. [Add rule] を選択します。
b. [Type] で、[All traffic] を選択します。
c. [Destination type] (送信先タイプ) で、[Custom] (カスタム) を選択し、コピーしたセキュリティグループ ID をフィールドに貼り付けます。
d. [Save Rules] (ルールの保存) を選択します。

ステップ 2: 一時インスタンスを作成する

EFA ソフトウェアコンポーネントのインストールおよび設定に使用する一時インスタンスを起動します。このインスタンスを使用して、EFA 対応のインスタンスを起動する EFA 対応の AMI を作成します。

一時インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前は、リソースタグとしてインスタンスに割り当てられます (Name=*`EFA-instance`*)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、サポートされている AWS Deep Learning AMI バージョン 25.0 以降を選択します。
5. [Instance type] (インスタンスタイプ) セクションで、`p3dn.24xlarge` または `p4d.24xlarge` のいずれかを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。

7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。

   a. [サブネット] で、インスタンスを起動するサブネットを選択します。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。

   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択します。

   c. [Advanced network configuration] (高度なネットワーク設定) セクションを展開し、[Elastic Fabric Adapter] の [Enable] (有効) を選択します。

8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。

   > **Note**
   >
   > Nvidia CUDA ツールキットには、追加の 10 ~ 20 GiB のストレージをプロビジョニングする必要があります。十分な量のストレージをプロビジョニングしないと、Nvidia ドライバーと CUDA ツールキットをインストールしようとしたときに、`insufficient disk space` エラーが発生します。

9. 右側の [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

ステップ 3: EFA と NCCL の設定をテストする

テストを実行し、EFA と NCCL に一時インスタンスが適切に設定されていることを確認します。

EFA と NCCL 設定をテストするには

1. テストを実行するホストを指定するホストファイルを作成します。以下のコマンドは、インスタンス自体へのリファレンスを含む `my-hosts` と呼ばれるホストファイルを作成します。

   IMDSv2

   ```
   [ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
    "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
   && curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/latest/
   meta-data/local-ipv4 >> my-hosts
   ```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/local-ipv4 >> my-hosts
```

2. テストを実行し、ホストファイル (--hostfile) と使用する GPU の数 (-n) を指定します。以下のコマンドは、インスタンス自体の 8 つの GPU で all_reduce_perf テストを実行し、以下の環境変数を指定します。

   - FI_EFA_USE_DEVICE_RDMA=1 — (p4d.24xlarge のみ) 片側転送および両側転送にデバイスの RDMA 機能を使用します。
   - NCCL_DEBUG=INFO – 詳細なデバッグ出力を有効にします。また、テストの開始時に NCCL バージョンのみをプリントするために VERSION を指定したり、エラーメッセージのみを受信するために WARN を指定したりすることもできます。

   NCCL テスト引数に関する詳細は、公式の nccl-tests リポジトリのNCCL Tests READMEを参照してください。

   - p3dn.24xlarge

   ```
   $ /opt/amazon/openmpi/bin/mpirun \
       -x LD_LIBRARY_PATH=/opt/nccl/build/lib:/usr/local/cuda/lib64:/opt/amazon/efa/lib:/opt/amazon/openmpi/lib:/opt/aws-ofi-nccl/lib:$LD_LIBRARY_PATH \
       -x NCCL_DEBUG=INFO \
       --hostfile my-hosts -n 8 -N 8 \
       --mca pml ^cm --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to none \
       $HOME/nccl-tests/build/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n 100
   ```

   - p4d.24xlarge

   ```
   $ /opt/amazon/openmpi/bin/mpirun \
       -x FI_EFA_USE_DEVICE_RDMA=1 \
       -x LD_LIBRARY_PATH=/opt/nccl/build/lib:/usr/local/cuda/lib64:/opt/amazon/efa/lib:/opt/amazon/openmpi/lib:/opt/aws-ofi-nccl/lib:$LD_LIBRARY_PATH \
       -x NCCL_DEBUG=INFO \
       --hostfile my-hosts -n 8 -N 8 \
       --mca pml ^cm --mca btl tcp,self --mca btl_tcp_if_exclude lo,docker0 --bind-to none \
   ```

```
$HOME/nccl-tests/build/all_reduce_perf -b 8 -e 1G -f 2 -g 1 -c 1 -n 100
```

3. NCCL_DEBUG ログが出力されるときに、EFA が NCCL の基盤となるプロバイダーとしてアクティブであることを確認できます。

```
ip-192-168-2-54:14:14 [0] NCCL INFO NET/OFI Selected Provider is efa*
```

p4d.24xlarge インスタンスの使用時に、次の追加情報が表示されます。

```
ip-192-168-2-54:14:14 [0] NCCL INFO NET/OFI Running on P4d platform, Setting
 NCCL_TOPO_FILE environment variable to /home/ec2-user/install/plugin/share/aws-
ofi-nccl/xml/p4d-24xl-topo.xml
```

ステップ 4: 機械学習アプリケーションをインストールする

機械学習アプリケーションを一時インスタンスにインストールします。インストール手順は、それぞれの機械学習アプリケーションによって異なります。Linux インスタンスへのソフトウェアのインストールの詳細については、Linux インスタンスでのソフトウェアの管理を参照してください。

> **Note**
>
> インストール手順については、機械学習アプリケーションのドキュメントを参照してください。

ステップ 5: EFA および NCCL 対応 AMI を作成する

必要なソフトウェアコンポーネントのインストール後、EFA 対応のインスタンスの起動に再利用できる AMI を作成します。

一時インスタンスから AMI を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 作成した一時インスタンスを選択し、[アクション]、[イメージ]、[イメージの作成] の順に選択します。
4. [イメージの作成] で、次を行います。

   a. [イメージ名] に、の分かりやすい AMI 名を入力します。

b. (オプション) [イメージの説明] に、AMI の簡単な説明を入力します。

c. [イメージを作成] を選択します。

5. ナビゲーションペインで [AMIs] を選択します。

6. リストで作成した AMI を探します。ステータスが `pending` から `available` に変わるまで待ってから、次のステップに進みます。

ステップ 6: 一時インスタンスを終了する

この時点で、起動した一時インスタンスは不要になります。インスタンスを終了して、料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[インスタンス] を選択します。

3. 作成した一時インスタンスを選択し、[アクション]、[インスタンスの状態]、[インスタンスの終了] の順に選択します。

4. 確認を求めるメッセージが表示されたら、[終了] を選択します。

ステップ 7: クラスタープレイスメントグループで EFA および NCCL 対応のインスタンスを作成する

前に作成した EFA 対応の AMI と EFA 対応のセキュリティグループを使用して、EFA および NCCL 対応のインスタンスをクラスタープレイスメントグループ内で起動します。

> **Note**
>
> - EFA 対応のインスタンスをクラスターのプレイスメントグループに起動することは絶対的な要件ではありません。ただし、EFA 対応のインスタンスは、1 つのアベイラビリティーゾーン内の低レイテンシーグループに起動されるため、クラスタープレイスメントグループで実行することをお勧めします。
> - クラスターのインスタンスをスケールするときにキャパシティを使用できるようにするには、クラスタープレイスメントグループのキャパシティ予約を作成します。詳細については、クラスタープレイスメントグループでのキャパシティ予約 を参照してください。

New console

一時インスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) [Name and tags] (名前とタグ) セクションで、`EFA-instance` などのインスタンス名を指定します。指定した名前は、リソースタグとしてインスタンスに割り当てられます (`Name=EFA-instance`)。
4. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、[My AMIs] (マイ AMI) をクリックし、前のステップで作成した AMI を選択します。
5. [Instance type] (インスタンスタイプ) セクションで、`p3dn.24xlarge` または `p4d.24xlarge` のいずれかを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。
7. [Network settings] (ネットワーク設定) セクションで、[Edit] (編集) を選択し、次の操作を行います。
   a. [サブネット] で、インスタンスを起動するサブネットを選択します。サブネットを選択しない場合、EFA のインスタンスを有効にすることはできません。
   b. [Firewall (security groups)] (ファイアウォール (セキュリティグループ)) の場合、[Select existing security group] (既存のセキュリティグループの選択) を選択し、前のステップで作成したセキュリティグループを選択します。
   c. [Advanced network configuration] (高度なネットワーク設定) セクションを展開し、[Elastic Fabric Adapter] の [Enable] (有効) を選択します。
8. (オプション) [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。
9. [Advanced details] (高度な詳細) セクションの [Placement group name] (プレイスメントグループ名) で、インスタンスを起動するクラスタープレイスメントグループを選択します。新しいクラスタープレイスメントグループを作成する必要がある場合は、[Create new placement group] (新しいプレイスメントグループの作成) を選択します。
10. 右側の [Summary] (サマリー) パネルで、[Number of instances] (インスタンス数) に、起動する EFA 対応のインスタンスの数を入力し、[Launch Instance] (インスタンスの起動) を選択します。

Old console

EFA および NCCL 対応のインスタンスをクラスタープレイスメントグループに起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [インスタンスの作成] を選択します。
3. [AMI の選択] ページで、[マイ AMI] を選択し、前に作成した AMI を見つけて、[選択] をクリックします。
4. [インスタンスタイプの選択] ページで [p3dn.24xlarge] を選択し、[次へ: インスタンスの詳細の設定] を選択します。
5. [インスタンスの詳細設定] ページで、以下を行います。
   a. [インスタンス数] に、起動する EFA および NCCL 対応のインスタンスの数を入力します。
   b. [ネットワーク] および [サブネット] で、インスタンスを起動する VPC およびサブネットを選択します。
   c. [プレイスメントグループ] で、[インスタンスをプレイスメントグループに追加します] チェックボックスをオンにします。
   d. [プレイスメントグループ名] で、[新しいプレイスメントグループに追加する] チェックボックスをオンにし、分かりやすいプレイスメントグループ名を入力します。次に、[プレイスメントグループ戦略] で [クラスター] を選択します。
   e. [EFA] で、[有効化] を選択します。
   f. [ネットワークインターフェイス] セクションの [eth0] で、[新しいネットワークインターフェイス] を選択します。必要に応じて、プライマリ IPv4 アドレスと 1 つ以上のセカンダリ IPv4 アドレスを指定できます。関連付けられている IPv6 CIDR ブロックを持つサブネットにインスタンスを起動する場合は、必要に応じて、プライマリ IPv6 アドレスと 1 つ以上のセカンダリ IPv6 アドレスを指定できます。
   g. [次の手順: ストレージの追加] を選択します。
6. [ストレージの追加] ページで、AMI が指定するボリューム (ルートデバイスボリュームなど) に加えて、インスタンスにアタッチするボリュームを指定します。次に、[次の手順: タグの追加] を選択します。
7. [Add Tags] ページで、ユーザーフレンドリーな名前などを使ってインスタンスのタグを指定し、[Next: Configure Security Group] を選択します。
8. [セキュリティグループの設定] ページの [セキュリティグループの割り当て] で、[既存のセキュリティグループの選択] を選択し、前に作成したセキュリティグループを選択します。

9. [Review and Launch] を選択します。
10. [インスタンス作成の確認] ページで設定を確認し、[起動] を選択してキーペアを選択し、インスタンスを起動します。

ステップ 8: パスワードレス SSH を有効にする

クラスター内のすべてのインスタンスでアプリケーションを実行できるようにするには、リーダーノードからメンバーノードへのパスワードなしの SSH アクセスを有効にする必要があります。リーダーノードは、アプリケーションを実行するインスタンスです。クラスター内の残りのインスタンスはメンバーノードです。

クラスター内のインスタンス間でパスワードなしの SSH を有効にするには

1. クラスター内の 1 つのインスタンスをリーダーノードとして選択し、それに接続します。
2. リーダーノード上で strictHostKeyChecking を無効にし ForwardAgent を有効にします。任意のテキストエディタを使用して ~/.ssh/config ファイルを開き、以下を追加します。

```
Host *
    ForwardAgent yes
Host *
    StrictHostKeyChecking no
```

3. RSA キーペアを生成します。

```
$ ssh-keygen -t rsa -N "" -f ~/.ssh/id_rsa
```

キーペアは、$HOME/.ssh/ ディレクトリで作成されます。

4. リーダーノードのプライベートキーの許可を変更します。

```
$ chmod 600 ~/.ssh/id_rsa
chmod 600 ~/.ssh/config
```

5. 任意のテキストエディタで ~/.ssh/id_rsa.pub を開き、キーをコピーします。
6. クラスター内の各メンバーノードについて、次の操作を行います。
   a. インスタンスに接続します。
   b. 任意のテキストエディタで ~/.ssh/authorized_keys を開き、前にコピーしたパブリックキーを追加します。

7. パスワードレス SSH が正常に機能していることをテストするには、リーダーノードに接続して、次のコマンドを実行します。

```
$ ssh member_node_private_ip
```

キーまたはパスワードの入力を求められずに、メンバーノードに接続できるはずです。

EFA の操作

EFA は、Amazon EC2 の他の Elastic Network Interface と同じように作成、使用、管理することができます。ただし、Elastic Network Interface とは異なり、EFAs は、実行中状態のインスタンスにアタッチしたり、実行中状態のインスタンスからデタッチしたりすることはできません。

EFA の要件

EFA を使用するには、以下の操作を行う必要があります。

- サポートされているいずれかのインスタンスタイプを選択します。
- いずれかのサポートされているオペレーティングシステムの AMI を使用してください。
- EFA ソフトウェアコンポーネントをインストールします。詳細については、ステップ 3: EFA ソフトウェアをインストールするおよびステップ 5: (オプション) インテル MPI をインストールするを参照してください。
- セキュリティグループ自体との間のインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループを使用します。詳細については、ステップ 1: EFA 対応のセキュリティグループを準備するを参照してください。

コンテンツ

- EFAs の表示
- EFA の削除

EFA の作成

EFA は、VPC のサブネットに作成することができます。作成後に EFA を別のサブネットに移動することはできません。また、アタッチできるのは、同じアベイラビリティーゾーンの停止したインスタンスに限ります。

コンソールを使用して新しい EFA を作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Network Interfaces] を選択します。
3. [Create Network Interface] を選択します。
4. [説明] に、EFA の分かりやすい名前を入力します。
5. [サブネット] で、EFA を作成するサブネットを選択します。
6. [プライベート IP] に、プライマリのプライベート IPv4 アドレスを入力します。IPv4 アドレスを指定しない場合、選択されているサブネット内で使用可能なプライベート IPv4 アドレスが選択されます。
7. (IPv6 のみ) IPv6 CIDR ブロックが関連付けられているサブネットを選択した場合は、オプションで [IPv6 IP] フィールドに IPv6 アドレスを指定できます。
8. [Security groups] で、1 つまたは複数のセキュリティグループを選択します。
9. [EFA] で、[有効化] を選択します。
10. [Yes, Create] を選択します。

AWS CLI を使用して新しい EFA を作成するには

次の例に示されているように、create-network-interface コマンドを使用し、`interface-type` で `efa` を指定します。

```
aws ec2 create-network-interface --subnet-id subnet-01234567890 --
description example_efa --interface-type efa
```

停止したインスタンスへの EFA のアタッチ

EFA は、サポート対象の stopped 状態のインスタンスにアタッチすることができます。running 状態のインスタンスに EFA をアタッチすることはできません。サポートされるインスタンスタイプの詳細については、サポートされるインスタンスタイプを参照してください。

ネットワークインターフェイスをインスタンスにアタッチするのと同じ方法で、EFA をインスタンスにアタッチできます。詳細については、インスタンスへのネットワークインターフェイスのアタッチを参照してください。

インスタンス起動時の EFA のアタッチ

インスタンス起動時に既存の EFA をアタッチするには (AWS CLI)

次の例に示されているように、run-instances コマンドを使用し、NetworkInterfaceId で EFA の ID を指定します。

```
aws ec2 run-instances --image-id ami_id --count 1 --instance-
type c5n.18xlarge --key-name my_key_pair --network-interfaces
 DeviceIndex=0,NetworkInterfaceId=efa_id,Groups=sg_id,SubnetId=subnet_id
```

インスタンス起動時に新しい EFA をアタッチするには (AWS CLI)

次の例に示されているように、run-instances コマンドを使用し、InterfaceType で efa を指定します。

```
aws ec2 run-instances --image-id ami_id --count 1 --instance-
type c5n.18xlarge --key-name my_key_pair --network-interfaces
 DeviceIndex=0,InterfaceType=efa,Groups=sg_id,SubnetId=subnet_id
```

起動テンプレートへの EFA 追加

EFA 対応のインスタンスの起動に必要な設定情報を含む起動テンプレートを作成できます。EFA 対応の起動テンプレートを作成するには、新しい起動テンプレートを作成し、サポート対象のインスタンスタイプ、EFA 対応の AMI、および EFA 対応のセキュリティグループを指定します。詳細については、EFAと MPI の開始方法 を参照してください。

起動テンプレートを利用すると、AWS や AWS Batch など他の AWS ParallelCluster サービスで EFA 対応のインスタンスを起動できます。

起動テンプレートの作成の詳細については、起動テンプレートの作成を参照してください。

EFA の IP アドレスの管理

EFA に関連付けられた IP アドレスは変更できます。Elastic IP アドレスをお持ちの場合は、EFA に関連付けることができます。IPv6 CIDR ブロックに関連付けられているサブネットで EFA をプロビジョンしている場合は、1 つ以上の IPv6 アドレスを EFA に割り当てることができます。

IP アドレスを Elastic Network Interface に割り当てるのと同じ方法で、Elastic IP (IPv4) および IPv6 アドレスを EFA に割り当てます。詳細については、IP アドレスの管理を参照してください。

EFA のセキュリティグループの変更

EFA に関連付けられているセキュリティグループは変更することができます。OS バイパス機能を有効にするには、EFA が、セキュリティグループ自体との間のインバウンドおよびアウトバウンドのトラフィックをすべて許可するセキュリティグループのメンバーである必要があります。

Elastic Network Interface に関連付けられているセキュリティグループを変更するのと同じ方法で、EFA に関連付けられているセキュリティグループを変更します。詳細については、セキュリティグループの変更を参照してください。

EFA のデタッチ

EFA をインスタンスからデタッチするには、まずインスタンスを停止する必要があります。実行中状態のインスタンスから EFA をデタッチすることはできません。

インスタンスから Elastic Network Interface をデタッチするのと同じ方法で、EFA をインスタンスからデタッチします。詳細については、インスタンスからのネットワークインターフェイスのデタッチを参照してください。

EFAs の表示

アカウントのすべての EFAs を表示できます。

Elastic Network Interface を表示するのと同じ方法で EFAs を表示します。詳細については、ネットワークインターフェイスに関する詳細の表示を参照してください。

EFA の削除

EFA を削除するには、まずインスタンスから削除する必要があります。インスタンスにアタッチされている場合は、EFA を削除する必要があります。

Elastic Network Interface を削除するのと同じ方法で EFAs を削除します。詳細については、ネットワークインターフェイスの削除を参照してください。

EFA のモニタリング

Elastic Fabric Adapter のパフォーマンスをモニタリングするには、次の機能を使用できます。

Amazon VPC フローログ

Amazon VPC フローログを作成することで、EFA との間で送受信されるトラフィックに関する情報を取得できます。フローログデータは Amazon CloudWatch Logs と Amazon S3 に発行できます。フローログを作成したら、選択した送信先でそのデータを取得して表示できます。詳細については、Amazon VPC ユーザーガイドのVPC フローログを参照してください。

EFA のフローログを作成する方法は、Elastic Network Interface のフローログを作成する場合と同じです。詳細については、「Amazon VPC ユーザーガイド」の「フローログの作成」を参照してください。

フローログエントリで、EFA エントリは、`srcAddress` および `destAddress` で識別されます。次の例に示されているように、これらはいずれも MAC アドレス形式になります。

```
version accountId  eniId        srcAddress        destAddress       sourcePort destPort
 protocol packets bytes start      end        action log-status
2       3794735123 eni-10000001 01:23:45:67:89:ab 05:23:45:67:89:ab -          -
 -        9       5689  1521232534 1524512343 ACCEPT OK
```

Amazon CloudWatch

Amazon CloudWatch には、リアルタイムで EFAs をモニタリングできるメトリクスが用意されています。メトリクスの収集と追跡、カスタマイズしたダッシュボードの作成、および指定したメトリクスが指定したしきい値に達したときに通知またはアクションを実行するアラームの設定を行うことができます。詳細については、CloudWatch を使用したインスタンスのモニタリングを参照してください。

チェックサムを使用した EFA インストーラの検証

オプションで、MD5 または SHA256 チェックサムを使用して EFA tarball (.tar.gz ファイル) を検証できます。ソフトウェア発行元の ID を確認し、発行後にアプリケーションの変更または破損がないことを確認するために、この操作を行うことをお勧めします。

tarball を検証するには

MD5 チェックサムには md5sum ユーティリティを使用します。SHA256 チェックサムには sha256sum ユーティリティを使用し、tarball のファイル名を指定します。このコマンドは、tarball ファイルを保存したディレクトリから実行する必要があります。

- MD5

```
$  md5sum tarball_filename.tar.gz
```

- SHA256

```
$  sha256sum tarball_filename.tar.gz
```

このコマンドは、次の形式でチェックサム値を返します。

```
checksum_value tarball_filename.tar.gz
```

コマンドによって返されるチェックサム値を、次の表に示すチェックサム値と比較します。チェックサムが一致する場合は、インストールスクリプトを実行しても安全です。チェックサムが一致しない場合は、インストールスクリプトを実行せず、AWS Support にお問い合わせください。

例えば、次のコマンドは SHA256 チェックサムを使用して EFA 1.9.4 tarball を検証します。

```
$  sha256sum aws-efa-installer-1.9.4.tar.gz
```

```
1009b5182693490d908ef0ed2c1dd4f813cc310a5d2062ce9619c4c12b5a7f14 aws-efa-
installer-1.9.4.tar.gz
```

次の表に、最新バージョンの EFA のチェックサムを示します。

バージョン	URL のダウンロード	チェックサム
EFA 1.31.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.31.0.tar.gz	MD5: 856352f12bef2ccbadcd75e35aa52aaf SHA256: 943325bd37902a4300ac9e5715163537d56ecb4e7b87b37827c3e547aa1897bf

バージョン	URL のダウンロード	チェックサム
EFA 1.30.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.30.0.tar.gz	MD5: 31f48e1a47fe93ede8ebd273fb747358 SHA256: 876ab9403e07a0c3c91a1a34685a52eced890ae052df94857f6081c5f6c78a0a
EFA 1.29.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.29.1.tar.gz	MD5: e1872ca815d752c1d7c2b5c175e52a16 SHA256: 178b263b8c25845b63dc93b25bcdff5870df5204ec509af26f43e8d283488744
EFA 1.29.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.29.0.tar.gz	MD5: 39d06a002154d94cd982ed348133f385 SHA256: 836655f87015547e733e7d9f7c760e4e24697f8bbc261bb5f3560abd4206bc36
EFA 1.28.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.28.0.tar.gz	MD5: 9dc13b7446665822605e66febe074035 SHA256: 2e625d2d6d3e073b5178e8e861891273d896b66d03cb1a32244fd56789f1c435

バージョン	URL のダウンロード	チェックサム
EFA 1.27.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.27.0.tar.gz	MD5: 98bfb515ea3e8d93f554020f3837fa15 SHA256: 1d49a97b0bf8d964d91652a79ac851f2550e33a5bf9d0cf86ec9357ff6579aa3
EFA 1.26.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.26.1.tar.gz	MD5: 884e74671fdef4725501f7cd2d451d0c SHA256: c616994c924f54ebfabfab32b7fe8ac56947fae00a0ff453d975e298d174fc96
EFA 1.26.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.26.0.tar.gz	MD5: f8839f12ff2e3b9ba09ae8a82b30e663 SHA256: bc1abc1f76e97d204d3755d2a9ca307fc423e51c63141f798c2f15be3715aa11
EFA 1.25.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.25.1.tar.gz	MD5: 6d876b894547847a45bb8854d4431f18 SHA256: d2abc553d22b89a4ce92882052c1fa6de450d3a801fe005da718b7d4b9602b06

バージョン	URL のダウンロード	チェックサム
EFA 1.25.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.25.0.tar.gz	MD5: 1993836ca749596051da04694ea0d00c SHA256: 98b7b26ce031a2d6a93de2297cc71b03af647194866369ca53b60d82d45ad342
EFA 1.24.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.24.1.tar.gz	MD5: 211b249f39d53086f3cb0c07665f4e6f SHA256: 120cfeec233af0955623ac7133b674143329f9561a9a8193e473060f596aec62
EFA 1.24.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.24.0.tar.gz	MD5: 7afe0187951e2dd2c9cc4b572e62f924 SHA256: 878623f819a0d9099d76ecd41cf4f569d4c3aac0c9bb7ba9536347c50b6bf88e
EFA 1.23.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.23.1.tar.gz	MD5: 22491e114b6ee7160a8290145dca0c28 SHA256: 5ca848d8e0ff4d1571cd443c36f8d27c8cdf2a0c97e9068ebf000c303fc40797

バージョン	URL のダウンロード	チェックサム
EFA 1.23.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.23.0.tar.gz	MD5: 38a6d7c1861f5038dba4e441ca7683ca SHA256: 555d497a60f22e3857fdeb3dfc53aa86d05926023c68c916d15d2dc3df6525bd
EFA 1.22.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.22.1.tar.gz	MD5: 600c0ad7cdbc06e8e846cb763f92901b SHA256: f90f3d5f59c031b9a964466b5401e86fd0429272408f6c207c3f9048254e9665
EFA 1.22.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.22.0.tar.gz	MD5: 8f100c93dc8ab519c2aeb5dab89e98f8 SHA256: f329e7d54a86a03ea51da6ea9a5b68fb354fbae4a57a02f9592e21fce431dc3a
EFA 1.21.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.21.0.tar.gz	MD5: 959ccc3a4347461909ec02ed3ba7c372 SHA256: c64e6ca34ccfc3ebe8e82d08899ae8442b3ef552541cf5429c43d11a04333050

バージョン	URL のダウンロード	チェックサム
EFA 1.20.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.20.0.tar.gz	MD5: 7ebfbb8e85f1b94709df4ab3db47913b SHA256: aeefd2681ffd5c4c631d1502867db5b831621d6eb85b61fe3ec80df983d1dcf0
EFA 1.19.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.19.0.tar.gz	MD5: 2fd45324953347ec5518da7e3fefa0ec SHA256: 99b77821b9e72c8dea015cc92c96193e8db307deee05b91a58094cc331f16709
EFA 1.18.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.18.0.tar.gz	MD5: fc2571a72f5d3c7b7b576ce2de38d91e SHA256: acb18a0808aedb9a5e485f1469225b9ac97f21db9af78e4cd6939700debe1cb6
EFA 1.17.3	https://efa-installer.amazonaws.com/aws-efa-installer-1.17.3.tar.gz	MD5: 0517df4a190356ab559235147174cafd SHA256: 5130998b0d2883bbae189b21ab215ecbc1b01ae0231659a9b4a17b0a33ebc6ca

バージョン	URL のダウンロード	チェックサム
EFA 1.17.2	https://efa-installer.amazonaws.com/aws-efa-installer-1.17.2.tar.gz	MD5: a329dedab53c4832df218a24449f4c9a SHA256: bca1fdde8b32b00346e175e597ffab32a09a08ee9ab136875fb38283cc4cd099
EFA 1.17.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.17.1.tar.gz	MD5: 733ae2cfc9d14b52017eaf0a2ab6b0ff SHA256: f29322640a88ae9279805993cb836276ea240623820848463ca686c8ce02136f
EFA 1.17.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.17.0.tar.gz	MD5: d430fc841563c11c3805c5f82a4746b1 SHA256: 75ab0cee4fb6bd38889dce313183f5d3a83bd233e0a6ef6205d8352821ea901d
EFA 1.16.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.16.0.tar.gz	MD5: 399548d3b0d2e812d74dd67937b696b4 SHA256: cecec36495a1bc6fdc82f97761a541e4fb6c9a3cbf3cfcb145acf25ea5dbd45b

バージョン	URL のダウンロード	チェックサム
EFA 1.15.2	https://efa-installer.amazonaws.com/aws-efa-installer-1.15.2.tar.gz	MD5: 955fea580d5170b05823d51acde7ca21 SHA256: 84df4fbc1b3741b6c073176287789a601a589313accc8e6653434e8d4c20bd49
EFA 1.15.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.15.1.tar.gz	MD5: c4610267039f72bbe4e35d7bf53519bc SHA256: be871781a1b9a15fca342a9d169219260069942a8bda7a8ad06d4baeb5e2efd7
EFA 1.15.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.15.0.tar.gz	MD5: 9861694e1cc00d884fadac07d22898be SHA256: b329862dd5729d2d098d0507fb486bf859d7c70ce18b61c302982234a3a5c88f
EFA 1.14.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.14.1.tar.gz	MD5: 50ba56397d359e57872fde1f74d4168a SHA256: c7b1b48e86fe4b3eaa4299d3600930919c4fe6d88cc6e2c7e4a408a3f16452c7

バージョン	URL のダウンロード	チェックサム
EFA 1.14.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.14.0.tar.gz	MD5: 40805e7fd842c36ececb9fd7f921b1ae SHA256: 662d62c12de85116df33780d40e0533ef7dad92709f4f613907475a7a1b60a97
EFA 1.13.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.13.0.tar.gz	MD5: c91d16556f4fd53becadbb345828221e SHA256: ad6705eb23a3fce44af3afc0f7643091595653a723ad0374084f4f2b715192e1
EFA 1.12.3	https://efa-installer.amazonaws.com/aws-efa-installer-1.12.3.tar.gz	MD5: 818aee81f097918cfaebd724eddea678 SHA256: 2c225321824788b8ca3fbc118207b944cdb096b847e1e0d1d853ef2f0d727172
EFA 1.12.2	https://efa-installer.amazonaws.com/aws-efa-installer-1.12.2.tar.gz	MD5: 956bb1fc5ae0d6f0f87d2e481d49fccf SHA256: 083a868a2c212a5a4fcf3e4d732b685ce39cceb3ca7e5d50d0b74e7788d06259

バージョン	URL のダウンロード	チェックサム
EFA 1.12.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.12.1.tar.gz	MD5: f5bfe52779df435188b0a2874d0633ea SHA256: 5665795c2b4f09d5f3f767506d4d4c429695b36d4a17e5758b27f033aee58900
EFA 1.12.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.12.0.tar.gz	MD5: d6c6b49fafb39b770297e1cc44fe68a6 SHA256: 28256c57e9ecc0b0778b41c1f777a9982b4e8eae782343dfe1246079933dca59
EFA 1.11.2	https://efa-installer.amazonaws.com/aws-efa-installer-1.11.2.tar.gz	MD5: 2376cf18d1353a4551e35c33d269c404 SHA256: a25786f98a3628f7f54f7f74ee2b39bc6734ea9374720507d37d3e8bf8ee1371
EFA 1.11.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.11.1.tar.gz	MD5: 026b0d9a0a48780cc7406bd51997b1c0 SHA256: 6cb04baf5ffc58ddf319e956b5461289199c8dd805fe216f8f9ab8d102f6d02a

バージョン	URL のダウンロード	チェックサム
EFA 1.11.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.11.0.tar.gz	MD5: 7d9058e010ad65bf2e14259214a36949 SHA256: 7891f6d45ae33e822189511c4ea1d14c9d54d000f6696f97be54e915ce2c9dfa
EFA 1.10.1	https://efa-installer.amazonaws.com/aws-efa-installer-1.10.1.tar.gz	MD5: 78521d3d668be22976f46c6fecc7b730 SHA256: 61564582de7320b21de319f532c3a677d26cc46785378eb3b95c636506b9bcb4
EFA 1.10.0	https://efa-installer.amazonaws.com/aws-efa-installer-1.10.0.tar.gz	MD5: 46f73f5a7afe41b4bb918c81888fefa9 SHA256: 136612f96f2a085a7d98296da0afb6fa807b38142e2fc0c548fa986c41186282
EFA 1.9.5	https://efa-installer.amazonaws.com/aws-efa-installer-1.9.5.tar.gz	MD5: 95edb8a209c18ba8d250409846eb6ef4 SHA256: a4343308d7ea4dc943ccc21bcebed913e8868e59bfb2ac93599c61a7c87d7d25

バージョン	URL のダウンロード	チェックサム
EFA 1.9.4	https://efa-installer.amazonaws.com/aws-efa-installer-1.9.4.tar.gz	MD5: f26dd5c350422c1a985e35947fa5aa28 SHA256: 1009b5182693490d908ef0ed2c1dd4f813cc310a5d2062ce9619c4c12b5a7f14
EFA 1.9.3	https://efa-installer.amazonaws.com/aws-efa-installer-1.9.3.tar.gz	MD5: 95755765a097802d3e6d5018d1a5d3d6 SHA256: 46ce732d6f3fcc9edf6a6e9f9df0ad136054328e24675567f7029edab90c68f1
EFA 1.8.4	https://efa-installer.amazonaws.com/aws-efa-installer-1.8.4.tar.gz	MD5: 85d594c41e831afc6c9305263140457e SHA256: 0d974655a09b213d7859e658965e56dc4f23a0eee2dc44bb41b6d039cc5bab45

Amazon EC2 インスタンストポロジー

インスタンストポロジーを表示すると、インスタンス間の相対的な近接性を階層的に確認できます。この情報を使用して、ハイパフォーマンスコンピューティング (HPC) と機械学習 (ML) のコンピューティングインフラストラクチャを大規模に管理しながら、ジョブプレイスメントを最適化できます。HPC と ML のジョブはレイテンシーとスループットの影響を受けます。インスタンストポロジーを使用してインスタンスの場所を検出できます。また、HPC ジョブと ML ジョブを互いに物理的に近いインスタンスで実行することで、これらの情報をジョブの最適化のために使用できます。

インスタンストポロジーを使用して既存のインスタンスの場所を検出することはできますが、これを使用して既存のインスタンスに物理的に近い場所に新しいインスタンスを起動することはできませ

ん。インスタンスの配置に影響を与える場合は、クラスタープレイスメントグループでのキャパシティ予約 を使用できます。

料金

インスタンストポロジーを表示するための追加コストはかかりません。

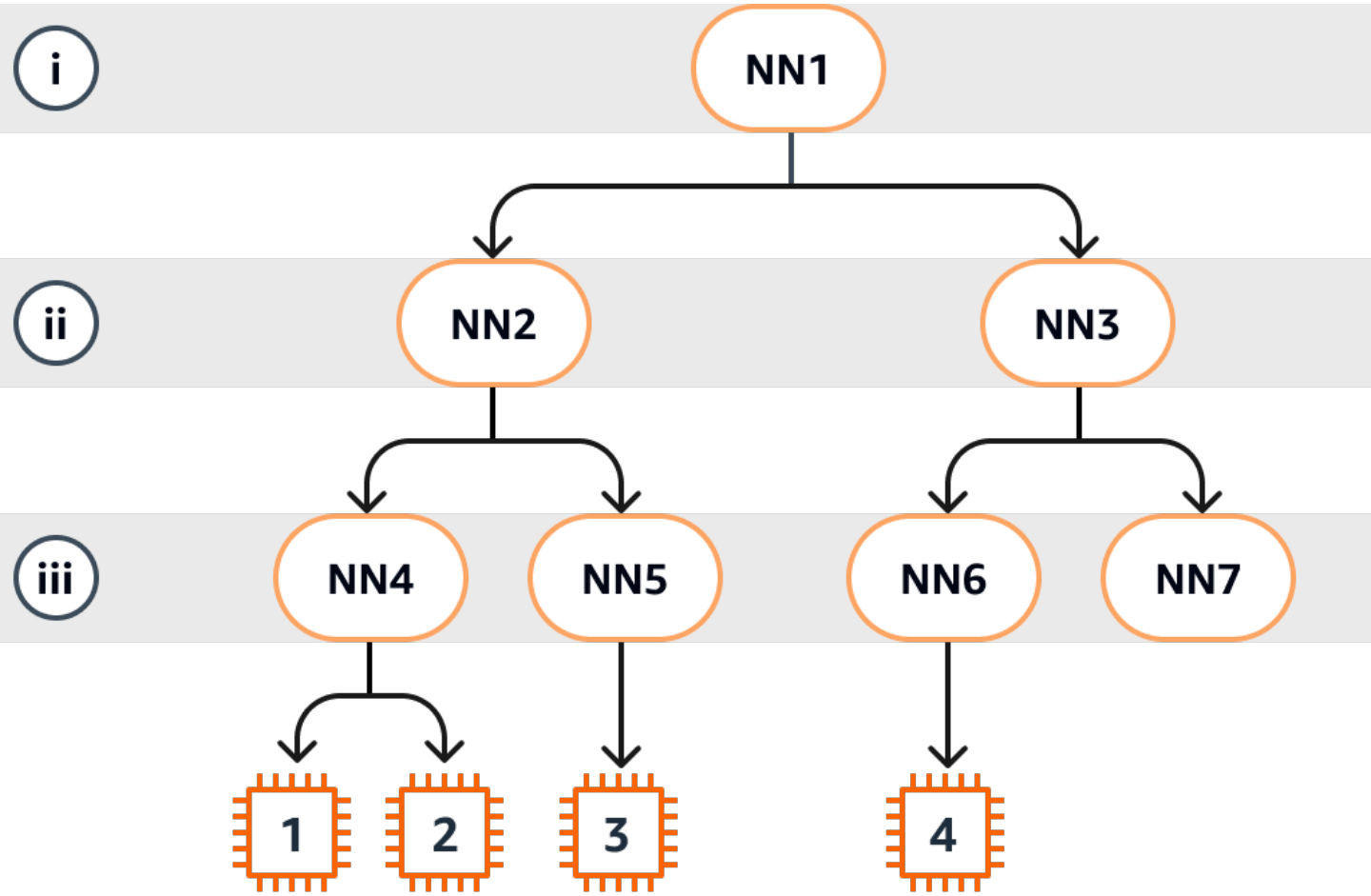
コンテンツ

インスタンストポロジーの仕組み

すべての EC2 インスタンスは 1 つのノードセットに接続します。ノードセットは 3 つのネットワークノードで構成され、各ノードは AWS ネットワーク内の異なるレイヤーを表します。ネットワークレイヤーは 3 つ以上のレイヤーの階層に配置されています。ノードセットでは、この階層のトップダウンビューが示され、最下層のレイヤーがインスタンスに最も近く接続されています。

ノードセットに関する情報はインスタンストポロジーと呼ばれます。

次の図は、インスタンストポロジーを視覚的に表しており、インスタンストポロジーの理解に役立ちます。ネットワークノードは NN1 ~ NN7 として示されます。数字 i、ii、iii は、ネットワークレイヤーを示しています。数字 1、2、3、4 は、EC2 インスタンスを示しています。インスタンスは、iii と示されている最下層のレイヤーのノードに接続しています。複数のインスタンスが同じノードに接続できます。



この例では、以下のようになっています。

- インスタンス 1 はレイヤー iii のネットワークノード 4 (NN4) に接続しています。NN4 はレイヤー ii のネットワークノード 2 (NN2) に接続しています。また NN2 は、この例でネットワーク階層の一番上にあるレイヤー i のネットワークノード 1 (NN1) に接続しています。ネットワークノードセットは NN1、NN2、NN4 で構成され、上位層から最下層まで階層的に示されています。
- インスタンス 2 はネットワークノード 4 (NN4) にも接続しています。インスタンス 1 とインスタンス 2 は同じネットワークノードセット (NN1、NN2、NN4) を共有しています。
- インスタンス 3 はネットワークノード 5 (NN5) に接続しています。NN5 は NN2 に接続しており、NN2 は NN1 に接続しています。インスタンス 3 に設定されているネットワークノードは、NN1、NN2、NN5 です。
- インスタンス 4 はネットワークノード 6 (NN6) に接続しています。このネットワークノードセットは NN1、NN3、NN6 です。

インスタンス 1、2、3 の近接性を考えると、インスタンス 1 と 2 は同じネットワークノード (NN4) に接続しているため互いに近く、インスタンス 3 は別のネットワークノード (NN5) に接続しているため遠くなります。

この図のすべてのインスタンスの近接性を考えると、インスタンス 1、2、3 はネットワークノードセットで NN2 を共有しているため、インスタンス 4 よりも互いに近くなります。

原則として、いずれかの 2 つのインスタンスに接続されているネットワークノードが同じ場合、インスタンス 1 と 2 の場合と同様に、これらのインスタンスは互いに物理的に近くなります。さらに、ネットワークノード間のホップ数が少ないほど、インスタンスは互いに近くなります。例えば、インスタンス 1 と 3 では、インスタンス 4 との共通のネットワークノード (NN1) へよりも共通のネットワークノード (NN2) への方がホップ数が少ないため、これらはインスタンス 4 よりも互いに近くなります。

この例では、ネットワークノード 7 (NN7) ではインスタンスが実行されていないため、API 出力に NN7 は含まれません。

出力の解釈方法

インスタンストポロジー情報は、DescribeInstanceTopology API を使用して取得します。出力では、インスタンスの基盤となるネットワークトポロジーが階層的に示されます。

次の出力例は、上の図にある 4 つのインスタンスにおけるネットワークトポロジーの情報に対応します。この例のため、出力例にはコメントが含まれています。

出力に含まれる次の情報に注意してください。

- `NetworkNodes` では、インスタンスのネットワークノードセットについて記述されます。
- 各ネットワークノードセットでは、ネットワークノードは上から下に階層的に一覧表示されます。
- インスタンスに接続されているネットワークノードは、一覧にある最後のネットワークノード (最下層) です。
- 互いに近いインスタンスを調べるにはまず、最下層にある共通のネットワークノードを見つけます。最下層に共通のネットワークノードがない場合、上位層で共通のネットワークノードを探します。

次の出力例で `i-1111111111example` と `i-2222222222example` は、最下層に共通のネットワークノード `nn-4444444444example` があるため、この例における他のインスタンスと比較して互いに最も近い位置にあります。

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",  //Corresponds to instance 1
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",           //Corresponds to NN1 in layer i
                "nn-2222222222example",           //Corresponds to NN2 in layer ii
                "nn-4444444444example"            //Corresponds to NN4 in layer iii -
 bottom layer, connected to the instance
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",  //Corresponds to instance 2
            "InstanceType": "p4d.24xlarge",
            "NetworkNodes": [
                "nn-1111111111example",           //Corresponds to NN1 - layer i
                "nn-2222222222example",           //Corresponds to NN2 - layer ii
                "nn-4444444444example"            //Corresponds to NN4 - layer iii -
 connected to instance
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-3333333333example",  //Corresponds to instance 3
            "InstanceType": "trn1.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",           //Corresponds to NN1 - layer i
                "nn-2222222222example",           //Corresponds to NN2 - layer ii
                "nn-5555555555example"            //Corresponds to NN5 - layer iii -
 connected to instance
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-444444444example",  //Corresponds to instance 4
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
```

```
                "nn-1111111111example",          //Corresponds to NN1 - layer i
                "nn-3333333333example",          //Corresponds to NN3 - layer ii
                "nn-6666666666example"           //Corresponds to NN6 - layer iii -
 connected to instance
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

制限事項

以下の制限が適用されます。

- インスタンスは running の状態である必要があります。
- 各インスタンストポロジーのビューはアカウントごとに異なります。
- AWS Management Consoleでは、インスタンストポロジーの表示はサポートされていません。

インスタンストポロジーの前提条件

インスタンスのインスタンストポロジーを表示する前に、インスタンスが次の要件を満たしていることを確認します。

インスタンスのトポロジーを表示するための要件

- AWS リージョン
- インスタンスのタイプ
- インスタンスの状態

AWS リージョン

サポートされている AWS リージョン:

- 米国東部 (バージニア北部)、米国東部 (オハイオ)、米国西部 (北カリフォルニア)、米国西部 (オレゴン)
- アジアパシフィック (ソウル)、アジアパシフィック (東京)

- カナダ (中部)
- 欧州 (フランクフルト)、欧州 (アイルランド)、欧州 (ストックホルム)

インスタンスのタイプ

サポートされるインスタンスタイプ:

- hpc6a.48xlarge | hpc6id.32xlarge | hpc7a.12xlarge | hpc7a.24xlarge | hpc7a.48xlarge | hpc7a.96xlarge | hpc7g.4xlarge | hpc7g.8xlarge | hpc7g.16xlarge
- p3dn.24xlarge | p4d.24xlarge | p4de.24xlarge | p5.48xlarge
- trn1.2xlarge | trn1.32xlarge | trn1n.32xlarge

特定のリージョンで利用可能なインスタンスタイプを確認するには

利用可能なインスタンスタイプは、リージョンごとに異なります。あるリージョンでインスタンスタイプが利用可能かどうかを確認するには、--region パラメータとともに describe-instance-types-offerings コマンドを使用します。結果を興味のあるインスタンスタイプまたはインスタンスファミリーにスコープする --filters パラメータと、出力を InstanceType の値にスコープする --query パラメータを含めます。

```
aws ec2 describe-instance-type-offerings \
    --region us-east-2 \
    --filters 'Name=instance-type, Values=trn1*' \
    --query 'InstanceTypeOfferings[].InstanceType'
```

正常な出力

```
[
    "trn1.2xlarge",
    "trn1.32xlarge",
    "trn1n.32xlarge"
]
```

インスタンスの状態

インスタンスは running の状態である必要があります。別の状態にあるインスタンスのインスタンストポロジーの情報は取得できません。

Amazon EC2 インスタンストポロジーの例

describe-instance-topology CLI コマンドを使用して、EC2 インスタンスのインスタンストポロジーを表示できます。

パラメータやフィルターなしで `describe-instance-topology` コマンドを使用すると、指定したリージョン内のこのコマンドで利用可能なインスタンスタイプに一致する、すべてのインスタンスが応答に含まれます。リージョンを設定するには、`--region` パラメータを含めるかデフォルトのリージョンを設定できます。デフォルトのリージョンの設定についての詳細は、「リソースのリージョンの指定」を参照してください。

指定したインスタンス ID またはプレイスメントグループ名と一致するインスタンスを返すパラメータを含めることができます。また、指定したインスタンスタイプやインスタンスファミリーに一致するインスタンス、または指定したアベイラビリティーゾーンやローカルゾーン内のインスタンスを返すフィルターを含めることもできます。1 つのパラメータまたはフィルター、もしくはパラメータとフィルターの組み合わせを含めることができます。

出力はページ分割されます。デフォルトでは、1 ページあたり最大 20 インスタンスです。`--max-results` パラメータを使用すると、1 ページあたり最大 100 インスタンスまで指定できます。

詳細については、AWS CLI コマンドリファレンスの describe-instance-topology を参照してください。

必要なアクセス許可

インスタンストポロジーを表示するには、次のアクセス許可が必要です。

- `ec2:DescribeInstanceTopology`

例

- 例 1 - パラメータもフィルターもない
- 例 2 — instance-type フィルター
  - 例 2a — 指定したインスタンスタイプの完全一致フィルター
  - 例 2b — インスタンスファミリーのワイルドカードフィルター
  - 例 2c — インスタンスファミリーと完全一致フィルターの組み合わせ
- 例 3 — zone-id フィルター
  - 例 3a — アベイラビリティーゾーンフィルター

- 例 3b — ローカルゾーンフィルター
- 例 3c — アベイラビリティーゾーンフィルターとローカルゾーンフィルターの組み合わせ
- 例 4 — instance-type フィルターと zone-id フィルターの組み合わせ
- 例 5 — プレイスメントグループ名パラメーター
- 例6 — インスタンス ID

例 1 - パラメータもフィルターもない

すべてのインスタンスのインスタントポロジーを記述するには

パラメータやフィルターを指定せずに、describe-instance-topology CLI コマンドを使用します。

```
aws ec2 describe-instance-topology --region us-west-2
```

レスポンスは、この API でサポートされているインスタンスタイプと一致するインスタンスのみを返します。インスタンスは、異なるアベイラビリティーゾーン、ローカルゾーン (ZoneId)、およびプレイスメントグループ (GroupName) に配置できます。インスタンスがプレイスメントグループ内にない場合、GroupName フィールドは出力に表示されません。この出力例では、プレイスメントグループ内には 1 つのインスタンスのみが存在します。

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "my-ml-cpg",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "p4d.24xlarge",
            "NetworkNodes": [
```

```
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-3333333333example",
            "InstanceType": "trn1.32xlarge",
            "NetworkNodes": [
                "nn-1212121212example",
                "nn-1211122211example",
                "nn-1311133311example"
            ],
            "ZoneId": "usw2-az4",
            "AvailabilityZone": "us-west-2d"
        },
        {
            "InstanceId": "i-444444444example",
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-5434334334example",
                "nn-1235301234example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 2 — instance-type フィルター

指定したインスタンスタイプ (完全一致)、またはインスタンスファミリーでフィルタリング (ワイルドカードを使用) できます。指定したインスタンスタイプフィルターとインスタンスファミリーのフィルターを組み合わせることもできます。

例 2a — 指定したインスタンスタイプの完全一致フィルター

指定したインスタンスタイプに一致するすべてのインスタンスのインスタントポロジーを記述するには

instance-type フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力は trn1n.32xlarge インスタンスに対してフィルタリングされます。レスポンスは、指定したインスタンスタイプと一致するインスタンスのみを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters Name=instance-type,Values=trn1n.32xlarge
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 2b — インスタンスファミリーのワイルドカードフィルター

インスタンスファミリーに一致するすべてのインスタンスのインスタントポロジーを記述するには

instance-type フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力は trn1* インスタンスに対してフィルタリングされます。レスポンスは、指定したインスタンスファミリーに一致するインスタンスのみを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters Name=instance-type,Values="trn1*"
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-3333333333example",
            "InstanceType": "trn1.32xlarge",
            "NetworkNodes": [
                "nn-1212121212example",
                "nn-1211122211example",
                "nn-1311133311example"
            ],
            "ZoneId": "usw2-az4",
            "AvailabilityZone": "us-west-2d"
        },
        {
            "InstanceId": "i-444444444example",
            "InstanceType": "trn1.2xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-5434334334example",
                "nn-1235301234example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 2c — インスタンスファミリーと完全一致フィルターの組み合わせ

インスタンスファミリーまたは指定したインスタンスタイプに一致するすべてのインスタンスのインスタンストポロジーを記述するには

instance-type フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力は pd4d* または trn1n.32xlarge インスタンスに対してフィルタリングされます。レスポンスは、指定したフィルターのいずれかに一致するインスタンスを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters "Name=instance-type,Values=p4d*,trn1n.32xlarge"
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-4343434343example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 3 — zone-id フィルター

zone-id フィルターを使用して、アベイラビリティーゾーンまたはローカルゾーンでフィルタリングできます。アベイラビリティーゾーンフィルターとローカルゾーンフィルターを組み合わせることもできます。

例 3a — アベイラビリティーゾーンフィルター

指定したアベイラビリティーゾーンに一致するすべてのインスタンスのインスタントポロジーを記述するには

zone-id フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力はアベイラビリティーゾーン us-west-2a に対してフィルタリングされます。レスポンスは、指定したアベイラビリティーゾーンに一致するインスタンスのみを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters Name=zone-id,Values="us-west-2a"
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 3b — ローカルゾーンフィルター

指定したローカルゾーンに一致するすべてのインスタンスのインスタントポロジーを記述するには

zone-id フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力はローカルゾーン usw2-az2 に対してフィルタリングされます。レスポンスは、指定したローカルゾーンに一致するインスタンスのみを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters Name=zone-id,Values=usw2-az2
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 3c — アベイラビリティーゾーンフィルターとローカルゾーンフィルターの組み合わせ

指定したアベイラビリティーゾーンまたはローカルゾーンに一致するすべてのインスタンスのインスタンストポロジーを記述するには

zone-id フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力はアベイラビリティーゾーン us-west-2a およびローカルゾーン usw2-az2 に対してフィルタリングされます。レスポンスは、指定したフィルターのいずれかに一致するインスタンスを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters "Name=zone-id,Values=us-west-2a,usw2-az2"
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 4 — instance-type フィルターと zone-id フィルターの組み合わせ

1 つのコマンドですべてのフィルターを組み合わせることができます。

指定したインスタンスタイプ、インスタンスファミリー、アベイラビリティーゾーンまたはローカルゾーンに一致するすべてのインスタンスのインスタントポロジーを記述するには

`instance-type` および `zone-id` フィルターとともに describe-instance-topology CLI コマンドを使用します。この例では、出力は `p4d*` インスタンスファミリー、`trn1n.32xlarge` インスタンスタイプ、アベイラビリティーゾーン `us-west-2a` およびローカルゾーン `usw2-az2` に対して

フィルタリングされます。レスポンスは、us-west-2a または usw2-az2 ゾーン内の p4d* または trn1n.32xlarge インスタンスに一致するインスタンスを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --filters "Name=instance-type,Values=p4d*,trn1n.32xlarge" "Name=zone-id,Values=us-west-2a,usw2-az2"
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
    "NextToken": "SomeEncryptedToken"
}
```

例 5 — プレイスメントグループ名パラメーター

指定したプレイスメントグループ内のすべてのインスタンスのインスタンストポロジーを記述するには

group-names パラメータとともに describe-instance-topology CLI コマンドを使用します。次の例では、インスタンスは ML-group または HPC-group プレイスメントグループに属すことができます。レスポンスは、いずれかのプレイスメントグループに属するインスタンスを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --group-names ML-group HPC-group
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "GroupName": "HPC-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
    ],
```

```
    "NextToken": "SomeEncryptedToken"
}
```

例6 — インスタンス ID

指定したインスタンスのインスタントポロジーを記述するには

--instance-ids パラメータとともに describe-instance-topology CLI コマンドを使用します。レスポンスは、指定したインスタンス ID と一致するインスタンスを返します。

```
aws ec2 describe-instance-topology \
    --region us-west-2 \
    --instance-ids i-1111111111example i-2222222222example
```

出力例

```
{
    "Instances": [
        {
            "InstanceId": "i-1111111111example",
            "InstanceType": "p4d.24xlarge",
            "GroupName": "ML-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3333333333example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        },
        {
            "InstanceId": "i-2222222222example",
            "InstanceType": "trn1n.32xlarge",
            "GroupName": "HPC-group",
            "NetworkNodes": [
                "nn-1111111111example",
                "nn-2222222222example",
                "nn-3214313214example"
            ],
            "ZoneId": "usw2-az2",
            "AvailabilityZone": "us-west-2a"
        }
```

```
    ],
    "NextToken": "SomeEncryptedToken"
}
```

プレイスメントグループ

ワークロードのニーズを対応するために、相互に依存する EC2 インスタンスのグループをプレイスメントグループ内に作成して、そのプレイスメントに影響を与えることができます。

ワークロードのタイプに応じて、以下のいずれかのプレイスメント戦略によりプレイスメントグループを作成できます。

- [クラスター] – アベイラビリティーゾーン内でインスタンスをまとめます。この戦略により、ワークロードは、ハイパフォーマンスコンピューティング (HPC) アプリケーションで典型的な緊密に組み合わされたノード間通信に必要な低レイテンシーネットワークパフォーマンスを実現できます。
- パーティション – インスタンスを複数の論理パーティションに分散させ、1 つのパーティション内のインスタンスのグループが基盤となるハードウェアを別のパーティション内のインスタンスのグループと共有しないようにします。この戦略は、Hadoop、Cassandra、Kafka などの大規模な分散および複製ワークロードで一般的に使用されます。
- スプレッド 相関性のエラーを減らすために、少数のインスタンスを基盤となるハードウェア全体に厳密に配置します。

プレイスメントグループは任意で選択します。インスタンスをリプレイスメントグループに作成しない場合、EC2 は、関連する障害を最小限に抑えるために、すべてのインスタンスが基盤となるハードウェア全体に分散されるような方法でインスタンスを配置しようとします。

プレイスメントグループを作成するための料金は発生しません。

プレイスメント戦略

プレイスメントグループは、次のいずれかのプレイスメント戦略を使用して作成できます。

プレイスメント戦略:

- クラスタープレイスメントグループ
- パーティションプレイスメントグループ

- スプレッドプレイスメントグループ

クラスタープレイスメントグループ

クラスタープレイスメントグループは、単一のアベイラビリティーゾーン内のインスタンスを論理的にグループ化したものです。クラスタープレイスメントグループは、同じリージョン内の複数のピア接続 VPC にまたがることができます。同じクラスタープレイスメントグループ内のインスタンスは、TCP/IP トラフィックのフローあたりのスループット上限が高くなり、ネットワークの二分帯域幅の広い同じセグメントに配置されます。

次の図は、クラスタープレイスメントグループに配置されたインスタンスを示しています。

低いネットワークレイテンシー、高いネットワークスループット、またはその両方からメリットを受けるアプリケーションの場合は、クラスタープレイスメントグループの使用をお勧めします。また、ネットワークトラフィックの大部分がグループ内のインスタンス間で発生している場合にもお勧めします。プレイスメントグループで、最も低いレイテンシーと最も高いネットワークパフォーマンス (1 秒あたりパケット数) を実現するためには、拡張ネットワーキングをサポートするインスタンスタイプを選択します。詳細については、拡張ネットワーキングを参照してください。

インスタンスは、次の方法で起動することをお勧めします。

- プレイスメントグループ内で必要な数のインスタンスを起動するには、1 つの起動リクエストを使用します。
- プレイスメントグループ内のすべてのインスタンスに同じインスタンスタイプを使用します。

後でプレイスメントグループにさらにインスタンスを追加しようとした場合、またはプレイスメントグループ内で複数のインスタンスタイプを起動しようとした場合、容量不足エラーが発生する可能性が高くなります。

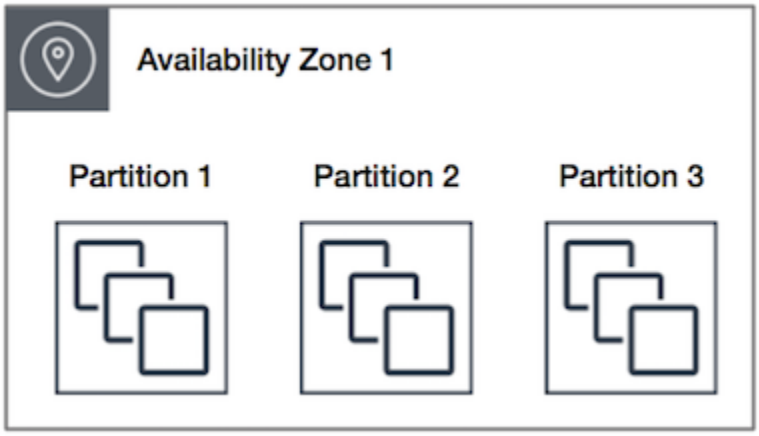
プレイスメントグループ内のインスタンスを停止して再起動しても、そのインスタンスは同じプレイスメントグループ内で実行されます。ただし、インスタンスに対して十分な容量がない場合、起動は失敗します。

既にインスタンスを実行中のプレイスメントグループ内のインスタンスを起動するときに容量エラーを受け取った場合は、プレイスメントグループ内のすべてのインスタンスを停止して開始し、もう一度起動を試みてください。インスタンスを起動すると、すべてのリクエストしたインスタンスに応じた容量があるハードウェアにインスタンスが移行される場合があります。

パーティションプレイスメントグループ

パーティションプレイスメントグループは、アプリケーションに関連するハードウェア障害の頻度を軽減するために役立ちます。パーティションプレイスメントグループを使用する場合、Amazon EC2 は各グループをパーティションと呼ばれる論理的なセグメントに分割します。Amazon EC2 では、プレイスメントグループ内の各パーティションにそれぞれ一連のラックがあります。各ラックには独自のネットワークおよび電源があります。プレイスメントグループ内のパーティションどうしが同じラックを共有することはありません。これにより、アプリケーション内でのハードウェア障害による影響を隔離できます。

次のイメージは、単一のアベイラビリティーゾーン内のパーティションプレイスメントグループのシンプルな描写を示しています。ここでは、3 つのパーティション (パーティション 1、パーティション 2、パーティション 3) があるパーティションプレイスメントグループに配置されたインスタンスを示しています。各パーティションは複数のインスタンスで構成されています。各パーティション内のインスタンスは、他のパーティション内のラックを共有しないため、単一のハードウェア障害の影響は関連付けられたパーティションのみに留まります。



パーティションプレイスメントグループは、HDFS、HBase、Cassandra などの大規模な分散および複製ワークロードを異なるラック間でデプロイするために使用できます。インスタンスをパーティ

ションプレイスメントグループに起動すると、Amazon EC2 は、指定したパーティション数全体にインスタンスを均等に分散しようとします。インスタンスを特定のパーティションに起動して、インスタンスの配置場所をより細かく制御することもできます。

パーティションプレイスメントグループは、同じリージョン内の複数のアベイラビリティーゾーンにパーティションを持つことができます。パーティションプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つのパーティションを持つことができます。パーティションプレイスメントグループで起動できるインスタンス数の制限は、アカウントの制限のみです。

また、パーティションプレイスメントグループでは各パーティションが可視化されるため、どのインスタンスがどのパーティションにあるかを確認できます。この情報は、HDFS、HBase、Cassandra などトポロジー対応アプリケーションと共有できます。これらのアプリケーションはこの情報を利用してインテリジェントなデータレプリケーションの決定を行い、データの可用性と耐久性を向上します。

パーティションプレイスメントグループでインスタンスを開始または起動し、リクエストを実行するための固有のハードウェアが不足している場合、そのリクエストは失敗します。Amazon EC2 では、時間の経過とともに、より明確なハードウェアを利用できるようになるため、後でリクエストを再試行できます。

スプレッドプレイスメントグループ

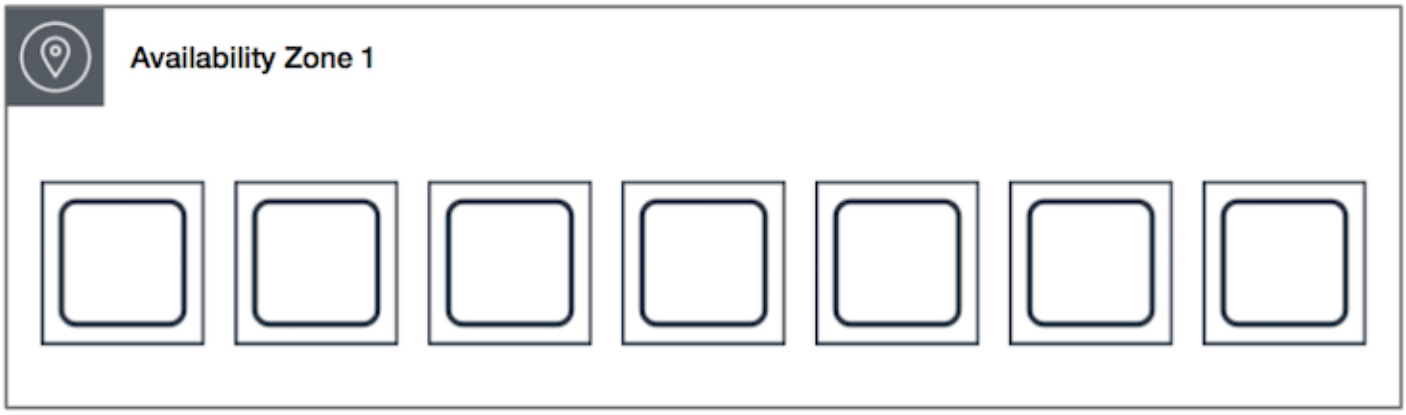
スプレッドプレイスメントグループは、それぞれ異なるハードウェアに配置されるインスタンスのグループです。

スプレッドプレイスメントグループは、少数の重要なインスタンスが互いに分離して保持される必要があるアプリケーションに推奨されます。スプレッドレベルのプレイスメントグループでインスタンスを起動すると、インスタンスが同じ機器を共有するときに発生し得る同時障害のリスクが軽減されます。スプレッドレベルのプレイスメントグループは、異なるハードウェアへのアクセスを提供するため、長時間のインスタンスタイプの混合やインスタンスの起動に適しています。

スプレッドプレイスメントグループでインスタンスを開始または起動し、リクエストを実行するための固有のハードウェアが不足している場合、そのリクエストは失敗します。Amazon EC2 では、時間の経過とともに、より明確なハードウェアを利用できるようになるため、後でリクエストを再試行できます。プレイスメントグループは、ラックまたはホスト全体でインスタンスを分散できます。ラックレベルのスプレッドプレイスメントグループは、AWS リージョンおよび AWS Outposts で使用できます。ホストレベルのスプレッドプレイスメントグループは、AWS Outposts を使用する場合にのみ使用できます。

ラックレベルのスプレッドプレイスメントグループ

次の図は、1 つのアベイラビリティーゾーン内の、スプレッドプレイスメントグループに配置された 7 つのインスタンスを示しています。7 つのインスタンスは、7 つの異なるラックに配置され、各ラックは独自のネットワークおよび電源を備えています。



ラックレベルのスプレッドプレイスメントグループは、同じリージョン内の複数のアベイラビリティーゾーンに分散できます。リージョンでは、ラックレベルのスプレッドプレイスメントグループについては、グループごとのアベイラビリティーゾーンごとに、最大 7 つの実行中のインスタンスを持つことができます。Outposts では、ラックレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のラックと同じ数のインスタンスを保持できます。

ホストレベルのスプレッドプレイスメントグループ

ホストレベルのスプレッドプレイスメントグループは、AWS Outposts を使用する場合にのみ使用できます。ホストスプレッドレベル配置グループは、Outpost デプロイメント内のホストと同じ数のインスタンスを保持できます。詳細については、「the section called “AWS Outposts のプレイスメントグループ”」を参照してください。

プレイスメントグループのルールと制限

トピック

- 一般的なルールと制限
- クラスタープレイスメントグループのルールと制限
- パーティションプレイスメントグループのルールと制限
- スプレッドプレイスメントグループのルールと制限

一般的なルールと制限

プレイスメントグループを使用する前に、次のルールに注意してください。

- リージョンごとにアカウントあたり、最大 500 個のプレイスメントグループを作成できます。
- プレイスメントグループには、リージョンの AWS アカウント内で固有の名前を付ける必要があります。
- プレイスメントグループをマージすることはできません。
- インスタンスは、1 つのプレイスメントグループ内で一度に起動できます。複数のプレイスメントグループにまたがることはできません。
- オンデマンドキャパシティ予約およびゾーンリザーブドインスタンスを使用すると、アベイラビリティーゾーンの EC2 インスタンスに対してキャパシティを予約できます。インスタンスを起動するときに、インスタンス属性がオンデマンドキャパシティ予約またはゾーンリザーブドインスタンスで指定された属性と一致する場合、リザーブドキャパシティはインスタンスによって自動的に使用されます。これは、プレイスメントグループにインスタンスを起動する場合にも当てはまります。

  クラスタープレイスメントグループにインスタンスを起動する場合は、クラスタープレイスメントグループでキャパシティを明示的に予約することをお勧めします。これを行うには、指定したクラスタープレイスメントグループにオンデマンドキャパシティ予約を作成します。この方法ではオンデマンドキャパシティ予約を使用してキャパシティを予約できますが、プレイスメントグループでキャパシティを明示的に予約できないため、ゾーンリザーブドインスタンスでは同じ操作を行うことはできません。
- Dedicated Hosts をプレイスメントグループで起動することはできません。
- プレイスメントグループの中断時に停止または休止するように設定されたスポットインスタンスは起動できません。

クラスタープレイスメントグループのルールと制限

クラスタープレイスメントグループには、以下のルールが適用されます。

- 以下のインスタンスタイプがサポートされています。
  - バーストパフォーマンスインスタンス (T2 など)、Mac1 インスタンス、および M7i-flex インスタンスを除く、現世代のインスタンス。
  - 以下は旧世代のインスタンスです: A1、C3、C4、G2、I2、M4、R3、および R4。

- クラスタープレイスメントグループを、複数のアベイラビリティーゾーンで設定することはできません。
- クラスタープレイスメントグループの 2 つのインスタンス間のトラフィックの最大ネットワークスループット速度は、2 つのインスタンスのうち遅い方に制限されます。高スループットの要件があるアプリケーションの場合、要件に適合するネットワーク接続を備えたインスタンスタイプを選択します。
- 拡張ネットワーキングに対して有効になっているインスタンスには、以下のルールが適用されます。
  - クラスタープレイスメントグループ内のインスタンス間では、シングルフロートラフィックに最大 10 Gbps を使用できます。クラスタープレイスメントグループ内にないインスタンスは、シングルフロートラフィックに最大 5 Gbps を使用できます。
  - 同じリージョン内でのインスタンスと Amazon S3 バケットとの間では、パブリック IP アドレス空間または VPC エンドポイントを介したトラフィックに、使用可能なすべてのインスタンスの集計帯域幅を使用できます。
- 複数のインスタンスタイプをクラスタープレイスメントグループに起動できます。ただし、これにより起動に成功するために必要な容量が使用可能になる可能性が低くなります。クラスタープレイスメントグループ内ですべてのインスタンスで同じインスタンスタイプを使用することをお勧めします。
- インターネットへのネットワークトラフィックとオンプレミスリソースへの AWS Direct Connect 接続は、クラスタープレイスメントグループに対して 5 Gbps に制限されます。

パーティションプレイスメントグループのルールと制限

パーティションプレイスメントグループには、以下のルールが適用されます。

- パーティションプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つのパーティションをサポートします。パーティションプレイスメントグループで起動できるインスタンス数の制限は、アカウントの制限のみです。
- インスタンスをパーティションプレイスメントグループに起動すると、Amazon EC2 は、すべてのパーティションにインスタンスを均等に分散しようとします。Amazon EC2 では、すべてのパーティションにインスタンスが均等に分散されるとは限りません。
- ハードウェア専有インスタンス を持つパーティションプレイスメントグループは、最大 2 つのパーティションを持つことができます。
- [Capacity Reservations] (キャパシティー予約) を使用して、パーティションプレイスメントグループでキャパシティを予約することはできません。

スプレッドプレイスメントグループのルールと制限

スプレッドプレイスメントグループには、以下のルールが適用されます。

- ラックスプレッドプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つの実行インスタンスをサポートします。例えば、3 つのアベイラビリティーゾーンがあるリージョンでは、グループ内で合計 21 個のインスタンスを実行でき、各アベイラビリティーゾーンに 7 個のインスタンスがあります。同じアベイラビリティーゾーンと同じスプレッドプレイスメントグループで 8 番目のインスタンスを開始しようとすると、インスタンスは起動しません。アベイラビリティーゾーンに 7 個を超えるインスタンスが必要な場合は、複数のスプレッドプレイスメントグループを使用することをお勧めします。複数のプレイスメントグループに分散しても、グループ間でインスタンスが分散されるとは限りませんが、グループごとの分散が確実になされるようにできるため、特定の障害クラスからの影響は制限されます。
- ハードウェア専有インスタンス では、スプレッドプレイスメントグループはサポートされていません。
- ホストレベルのスプレッドプレイスメントグループは、AWS Outposts のプレイスメントグループでのみサポートされます。ホストレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のホストと同じ数のインスタンスを保持できます。
- リージョンでは、ラックレベルのスプレッドプレイスメントグループについては、グループごとのアベイラビリティーゾーンごとに、最大 7 つの実行中のインスタンスを持つことができます。AWS Outposts では、ラックレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のラックと同じ数のインスタンスを保持できます。
- [Capacity Reservations] (キャパシティー予約) を使用して、スプレッドプレイスメントグループでキャパシティを予約することはできません。

プレイスメントグループの操作

コンテンツ

- プレイスメントグループの削除

プレイスメントグループの作成

プレイスメントグループは、次のいずれかの方法で作成できます。

Console

コンソールを使用してプレイスメントグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Placement Groups] を選択します。
3. [プレイスメントグループを作成] を選択します。
4. グループの名前を指定します。
5. グループのプレイスメント方法を選択します。
   - [Spread] (スプレッド) を選択する場合は、スプレッドレベルを選択します。
     - [ラック] - 制限なし
     - [ホスト] - Outposts のみ
   - [パーティション] を選択した場合は、グループ内のパーティション数を指定します。
6. プレースメントグループにタグを付けるには、タグの追加 を選択してから、キーと値を入力します。追加するタグごとに [Add tag] (タグを追加) を選択します。
7. [グループを作成] を選択します。

AWS CLI

AWS CLI を使用してプレイスメントグループを作成するには

create-placement-group コマンドを使用します。次の例では、`cluster` プレイスメント戦略を使用する、`my-cluster` という名前のプレイスメントグループを作成し、キー `purpose` と値 `production` を持つタグを適用します。

```
aws ec2 create-placement-group \
    --group-name my-cluster \
    --strategy cluster \
    --tag-specifications 'ResourceType=placement-
group,Tags={Key=purpose,Value=production}'
```

AWS CLI を使用してパーティションプレイスメントグループを作成するには

create-placement-group コマンドを使用します。--strategy パラメータに値として partition を指定し、--partition-count パラメータに必要なパーティション数を指定します。この例では、パーティションプレイスメントグループは HDFS-Group-A という名で、パーティションは 5 つ作成されています。

```
aws ec2 create-placement-group \
    --group-name HDFS-Group-A \
    --strategy partition \
    --partition-count 5
```

PowerShell

AWS Tools for Windows PowerShell を使用してプレイスメントグループを作成するには

New-EC2PlacementGroup コマンドを使用します。

プレイスメントグループ情報を表示する

すべてのプレイスメントグループおよびそれらに関する情報は、次のいずれかの方法で表示できます。

Console

1 つまたは複数のプレイスメントグループに関する情報を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[ネットワークとセキュリティ] の下にある [プレイスメントグループ] を選択します。
3. [プレイスメントグループ] テーブルでは、プレイスメントグループごとに次の情報を表示できます。

   - [グループ名] – プレイスメントグループに付けた名前。
   - [グループ ID] - プレイスメントグループの ID。
   - [戦略] – プレイスメントグループのプレイスメント戦略。
   - [状態] — プレイスメントグループの状態。
   - [パーティション] – パーティションの数。戦略がパーティションの場合にのみ有効です。
   - [グループ ARN] – プレイスメントグループの Amazon リソースネーム (ARN)。

AWS CLI

すべてのプレイスメントグループの説明を表示するには

describe-placement-groups AWS CLI コマンドを使用します。

```
aws ec2 describe-placement-groups
```

レスポンスの例

```
{
    "PlacementGroups": [
        {
            "GroupName": "my-cluster-pg",
            "State": "available",
            "Strategy": "cluster",
            "GroupId": "pg-0123456789example",
            "GroupArn": "arn:aws:ec2:eu-west-1:111111111111:placement-group/my-
cluster-pg"
        },
            ...
     ]
}
```

特定のプレイスメントグループの説明を表示するには

describe-placement-groups AWS CLI コマンドを使用します。--group-id または --group-name パラメータを指定できます。

プレイスメントグループ ID を指定します。

```
aws ec2 describe-placement-groups --group-id pg-0123456789example
```

プレイスメントグループ名を指定します。

```
aws ec2 describe-placement-groups --group-name my-cluster-pg
```

レスポンスの例

```
{
```

```
    "PlacementGroups": [
        {
            "GroupName": "my-cluster-pg",
            "State": "available",
            "Strategy": "cluster",
            "GroupId": "pg-0123456789example",
            "GroupArn": "arn:aws:ec2:eu-west-1:111111111111:placement-group/my-
cluster-pg"
        }
    ]
}
```

プレイスメントグループのタグ付け

既存のプレイスメントグループを分類および管理しやすくするために、カスタムメタデータでタグ付けできます。タグの仕組みの詳細については、Amazon EC2 リソースのタグ付けを参照してください。

プレイスメントグループにタグを付けると、プレイスメントグループに起動されたインスタンスは自動的にタグ付けされなくなります。プレイスメントグループに起動されるインスタンスには、明示的にタグを付ける必要があります。詳細については、インスタンスを起動するときのタグの追加を参照してください。

タグの表示、追加、および削除は、以下のいずれかの方法で行います。

Console

既存のプレイスメントグループのタグを表示、追加、または削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Placement Groups] を選択します。
3. プレイスメントグループを選択し、[アクション]、[タグの管理] の順に選択します。
4. [タグを管理] 画面には、プレイスメントグループに割り当てられているタグが表示されます。

   - タグを追加するには、[Add tag] を選択し、タグのキーと値を入力します。プレイスメントグループごとに最大 50 個のタグを追加できます。詳細については、タグの制限を参照してください。
   - タグを削除するには、削除するタグの横にある [Remove] を選択します。

5. [Save] を選択します。

AWS CLI

プレイスメントグループタグを表示するには

describe-tags コマンドを使用して、指定したリソースのタグを表示します。次の例では、すべてのプレイスメントグループのタグの説明を表示します。

```
aws ec2 describe-tags \
    --filters Name=resource-type,Values=placement-group
```

```
{
    "Tags": [
        {
            "Key": "Environment",
            "ResourceId": "pg-0123456789EXAMPLE",
            "ResourceType": "placement-group",
            "Value": "Production"
        },
        {
            "Key": "Environment",
            "ResourceId": "pg-9876543210EXAMPLE",
            "ResourceType": "placement-group",
            "Value": "Production"
        }
    ]
}
```

describe-tags コマンドを使用し、ID を指定してプレイスメントグループのタグを表示することもできます。次の例では、pg-0123456789EXAMPLE のタグの説明を表示します。

```
aws ec2 describe-tags \

    --filters Name=resource-id,Values=pg-0123456789EXAMPLE
```

```
{
    "Tags": [
        {
            "Key": "Environment",
```

```
            "ResourceId": "pg-0123456789EXAMPLE",
            "ResourceType": "placement-group",
            "Value": "Production"
        }
    ]
}
```

プレイスメントグループの説明を表示して、プレイスメントグループのタグを表示することもできます。

describe-placement-groups コマンドを使用して、指定したプレイスメントグループの設定を表示します。この設定には、プレイスメントグループに指定されたタグがすべて含まれます。

```
aws ec2 describe-placement-groups \
    --group-name my-cluster
```

```
{
    "PlacementGroups": [
        {
            "GroupName": "my-cluster",
            "State": "available",
            "Strategy": "cluster",
            "GroupId": "pg-0123456789EXAMPLE",
            "Tags": [
                {
                    "Key": "Environment",
                    "Value": "Production"
                }
            ]
        }
    ]
}
```

AWS CLI を使用して既存のプレイスメントグループにタグを付けるには

create-tags コマンドを使用して、既存のリソースにタグ付けできます。次の例では、既存のプレイスメントグループに Key=Cost-Center と Value=CC-123 のタグが付けられています。

```
aws ec2 create-tags \
    --resources pg-0123456789EXAMPLE \
    --tags Key=Cost-Center,Value=CC-123
```

AWS CLI を使用してタグをプレイスメントグループから削除するには

delete-tags コマンドを使用して、既存のリソースからタグを削除できます。例については、AWS CLI コマンドリファレンスの例を参照してください。

PowerShell

プレイスメントグループタグを表示するには

Get-EC2Tag コマンドを使用します。

特定のプレイスメントグループのタグの説明を表示するには

Get-EC2PlacementGroup コマンドを使用します。

既存のプレイスメントグループ名にタグを付けるには

New-EC2Tag コマンドを使用します。

プレイスメントグループからタグを削除するには

Remove-EC2Tag コマンドを使用します。

プレイスメントグループ内でインスタンスを起動する方法

プレイスメントグループのルールと制限が満たされている場合、次のいずれかの方法を使用してプレイスメントグループ内でインスタンスを起動できます。

Console

プレイスメントグループ内でインスタンスを起動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. EC2 コンソールダッシュボードの [インスタンスの起動] ボックスで、[インスタンスの起動] を選択します。指示どおりにフォームを完了し、次の操作を行うように注意します。

   - [Instance type] (インスタンスタイプ) で、プレイスメントグループに起動できるインスタンスタイプを選択します。
   - [Summary] (概要) ボックスの [Number of instances] (インスタンスの数) で、このプレイスメントグループで必要なインスタンスの総数を入力します。これは、後でプレイスメントグループにインスタンスを追加できない場合があるためです。

- [Advanced details] (高度な詳細) の [Placement group name] (プレイスメントグループ名) で、インスタンスを新規または既存のプレイスメントグループに追加することを選択できます。パーティション戦略のあるプレイスメントグループを選択する場合は、[Target partition] (ターゲットパーティション) で、インスタンスを起動するパーティションを選択します。

AWS CLI

プレイスメントグループ内でインスタンスを起動するには

run-instances コマンドを使用し、`--placement "GroupName = my-cluster"` パラメータを使用してプレイスメントグループ名を指定します。次の例で、プレイスメントグループ名は `my-cluster` です。

```
aws ec2 run-instances --placement "GroupName = my-cluster"
```

AWS CLI を使用してパーティションプレイスメントグループの特定のパーティション内でインスタンスを起動するには

run-instances コマンドを使用して、`--placement "GroupName = HDFS-Group-A, PartitionNumber = 3"` パラメータを使用するグループプレイスメントグループ名とパーティションを指定します。この例では、パーティションプレイスメントグループは `HDFS-Group-A` という名で、パーティション数は `3` です。

```
aws ec2 run-instances --placement "GroupName = HDFS-Group-A, PartitionNumber = 3"
```

PowerShell

AWS Tools for Windows PowerShell を使用してプレイスメントグループ内でインスタンスを起動するには

New-EC2Instance コマンドを使用し、`-Placement_GroupName` パラメータを使用してプレイスメントグループ名を指定します。

プレイスメントグループのインスタンスの説明

次のいずれかの方法を使用して、インスタンスのプレイスメント情報を表示できます。AWS CLI を使用して、パーティション番号でパーティションプレイスメントグループをフィルターすることもできます。

Console

インスタンスのプレイスメントグループとパーティション番号を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択します。
4. [Details] (詳細) タブの [Host and placement group] (ホストとプレイスメントグループ) で、[Placement group] (プレイスメントグループ) を見つけます。プレイスメントグループにインスタンスがない場合、フィールドは空になります。それ以外の場合は、プレイスメントグループの名前が含まれます。プレイスメントグループがパーティションプレイスメントグループの場合、[Partition number (パーティション番号)] にはインスタンスのパーティション番号が含まれます。

AWS CLI

パーティションプレイスメントグループのインスタンスのパーティション番号を表示するには

describe-instancesコマンドを使用して `--instance-id` パラメータを指定します。

```
aws ec2 describe-instances --instance-id i-0123a456700123456
```

レスポンスにはプレイスメント情報が含まれています。この情報にはインスタンスのプレイスメントグループ名とパーティション番号が含まれます。

```
"Placement": {
     "AvailabilityZone": "us-east-1c",
     "GroupName": "HDFS-Group-A",
     "PartitionNumber": 3,
     "Tenancy": "default"
}
```

特定のパーティションプレイスメントグループとパーティション番号のインスタンスにフィルターを適用するには

describe-instances コマンドを使用して、`--filters` および `placement-group-name` フィルターを持つ `placement-partition-number` パラメータを指定します。この例では、パーティションプレイスメントグループは `HDFS-Group-A` という名で、パーティション数は `7` です。

```
aws ec2 describe-instances --filters "Name = placement-group-name, Values = HDFS-Group-A" "Name = placement-partition-number, Values = 7"
```

レスポンスは、指定されたプレイスメントグループ内の指定されたパーティション内にあるすべてのインスタンスをリストします。次の出力例は、返されたインスタンスのインスタンス ID、インスタンスタイプ、および配置情報のみを示しています。

```
"Instances": [
                {
                    "InstanceId": "i-0a1bc23d4567e8f90",
                    "InstanceType": "r4.large",
                    },

                    "Placement": {
                        "AvailabilityZone": "us-east-1c",
                        "GroupName": "HDFS-Group-A",
                        "PartitionNumber": 7,
                        "Tenancy": "default"
                    }

                {
                    "InstanceId": "i-0a9b876cd5d4ef321",
                    "InstanceType": "r4.large",
                    },

                    "Placement": {
                        "AvailabilityZone": "us-east-1c",
                        "GroupName": "HDFS-Group-A",
                        "PartitionNumber": 7,
                        "Tenancy": "default"
                    }
            ],
```

インスタンスのプレイスメントグループの変更

インスタンスのプレイスメントグループは、次の方法で変更できます。

- 既存のインスタンスをプレイスメントグループに移動する
- プレイスメントグループ間でインスタンスを移動する

インスタンスを移動できるようになる前に、インスタンスを stopped 状態にする必要があります。

Console

プレイスメントグループにインスタンスを移動するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[インスタンスの状態]、[インスタンスを停止] を選択します。
4. 選択したインスタンスについて、[アクション]、[インスタンス設定]、[インスタンスの配置の変更] を選択します。
5. [配置グループ] について、インスタンスの移動先のプレイスメントグループを選択します。
6. [保存] を選択します。

AWS CLI

プレイスメントグループにインスタンスを移動するには

1. stop-instances コマンドを使用して、インスタンスを停止します。
2. modify-instance-placement コマンドを使用し、インスタンスの移動先プレイスメントグループの名前を指定します。

```
aws ec2 modify-instance-placement \
    --instance-id i-0123a456700123456 \
    --group-name MySpreadGroup
```

3. start-instances コマンドを使用してインスタンスを起動します。

PowerShell

AWS Tools for Windows PowerShell を使用してプレイスメントグループにインスタンスを移動するには

1. Stop-EC2Instance コマンドを使用してインスタンスを停止します。
2. Edit-EC2InstancePlacement コマンドを使用し、インスタンスの移動先のプレイスメントグループの名前を指定します。
3. Start-EC2Instance コマンドを使用してインスタンスを起動します。

プレイスメントグループからインスタンスを削除する

プレイスメントグループから、次のいずれかの方法でインスタンスを削除できます。

インスタンスを移動または削除するには、まずインスタンスが stopped 状態になっている必要があります。

Console

プレイスメントグループからインスタンスを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[インスタンスの状態]、[インスタンスを停止] を選択します。
4. 選択したインスタンスについて、[アクション]、[インスタンス設定]、[インスタンスの配置の変更] を選択します。
5. [配置グループ] には [なし] を選択します。
6. [Save] を選択します。

AWS CLI

プレイスメントグループからインスタンスを削除するには

1. stop-instances コマンドを使用して、インスタンスを停止します。
2. modify-instance-placement コマンドを使用し、プレイスメントグループ名に空の文字列を指定します。

```
aws ec2 modify-instance-placement \
    --instance-id i-0123a456700123456 \
    --group-name ""
```

3. start-instances コマンドを使用してインスタンスを起動します。

PowerShell

AWS Tools for Windows PowerShell を使用してプレイスメントグループからインスタンスを削除するには

1. Stop-EC2Instance コマンドを使用してインスタンスを停止します。

2. Edit-EC2InstancePlacement コマンドを使用し、プレイスメントグループ名に空の文字列を指定します。
3. Start-EC2Instance コマンドを使用してインスタンスを起動します。

プレイスメントグループの削除

プレイスメントグループを交換する必要がある場合、または不要になった場合は、そのプレイスメントグループを削除できます。プレイスメントグループを削除するには、次のいずれかの方法を使用できます。

前提条件

削除するプレイスメントグループにはインスタンスが含まれていないことが必要です。プレイスメントグループ内で起動したすべてのインスタンスを終了し、インスタンスを別のプレイスメントグループに移動するか、プレイスメントグループから削除することができます。

Console

プレイスメントグループを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Placement Groups] を選択します。
3. プレイスメントグループを選択し、[Actions (アクション)]、[Delete (削除)] の順に選択します。
4. 確認を求められたら、**Delete**と入力し、[削除] を選択します。

AWS CLI

プレイスメントグループを削除するには

delete-placement-group コマンドを使用し、削除するプレイスメントグループの名前を指定します。次の例で、プレイスメントグループ名は `my-cluster` です。

```
aws ec2 delete-placement-group --group-name my-cluster
```

PowerShell

AWS Tools for Windows PowerShell を使用してプレイスメントグループを削除するには

Remove-EC2PlacementGroup コマンドを使用してプレイスメントグループを削除します。

プレイスメントグループの共有

プレイスメントグループを共有すると、別々の AWS アカウントが所有する相互に依存するインスタンスの配置を変更できます。プレイスメントグループは複数の AWS アカウントや自分の組織で共有できます。共有されたプレイスメントグループ内でインスタンスを起動することができます。

プレイスメントグループの所有者は、プレイスメントグループを次の人と共有できます。

- 組織内または組織外の特定の AWS アカウント
- 組織内の組織単位
- 組織全体

> **Note**
>
> プレイスメントグループを共有する AWS アカウントには、IAM ポリシーで次の権限が必要です。
>
> - `ec2:PutResourcePolicy`
> - `ec2:DeleteResourcePolicy`

トピック

ルールと制限

プレイスメントグループを共有する場合、またはプレイスメントグループが自分と共有される場合は、次のルールと制限が適用されます。

- プレイメントグループを共有するには、AWS アカウント内で所有している必要があります。自分に共有されているプレイスメントグループは共有できません。
- パーティションまたはスプレッドプレイスメントグループを共有しても、プレイスメントグループの制限は変わりません。共有パーティションプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つのパーティションをサポートし、共有スプレッドプレイスメントグループは、アベイラビリティーゾーンごとに最大 7 つの実行インスタンスをサポートします。
- ユーザーの組織や組織内の組織単位とプレイスメントグループを共有するには、AWS Organizations との共有を有効にする必要があります。詳細については、「AWS リソースの共有」を参照してください。
- 共有プレイスメントグループで所有するインスタンスを管理する責任はお客様にあります。
- 共有プレイスメントグループに関連付けられているが、自分が所有していないインスタンスやキャパシティ予約を表示または変更することはできません。

アベイラビリティーゾーン間での共有

リソースがリージョンの複数のアベイラビリティーゾーンに分散されるようにするために、アベイラビリティーゾーンは各 アカウントの名前に個別にマッピングされます。このため、アカウントが異なると、アベイラビリティーゾーンの命名方法が異なる場合があります。例えば、us-east-1a アカウントのアベイラビリティーゾーン AWS の場所は、別の us-east-1a アカウントのアベイラビリティーゾーン AWS の場所と異なる可能性があります。

自己のアカウントを基準にして Dedicated Hosts の場所を特定するには、アベイラビリティーゾーン ID (AZ ID) を使用する必要があります。アベイラビリティーゾーン ID は、すべての AWS アカウントにわたって各アベイラビリティーゾーンを一意に識別する ID です。例えば、use1-az1 は us-east-1 リージョンのアベイラビリティーゾーン ID であり、すべての AWS アカウントで同じ場所を示します。

アカウントのアベイラビリティーゾーンのアベイラビリティーゾーン ID を表示するには

1. AWS RAM コンソール (https://console.aws.amazon.com/ram) を開きます。
2. 現在のリージョンのアベイラビリティーゾーン ID は、画面の右側のパネルにある [お客様の AZ ID] に表示されます。

プレイスメントグループの共有

プレイスメントグループを共有するには、リソース共有に追加する必要があります。リソース共有とは、AWS RAM アカウント間で自身のリソースを共有するための AWS リソースです。リソース共有では、共有対象のリソースと、共有先のコンシューマーを指定します。

AWS Organizations の組織に属しており、組織内での共有が有効化されている場合、組織内のコンシューマーに対し、共有プレイスメントグループへのアクセス権が付与されます。

プレイスメントグループが、組織外の AWS アカウントと共有されている場合、AWS アカウント所有者はリソース共有に参加するための招待状を受け取ります。招待を承諾すると、共有プレイスメントグループにアクセスできます。

https://console.aws.amazon.com/ram または AWS CLI を使用して、AWS アカウント間でプレイスメントグループを共有できます。

AWS RAM console

> https://console.aws.amazon.com/ram を使用して所有している [share a placement group] (プレイスメントグループを共有する) には、「リソース共有の作成」を参照してください。

AWS CLI

> 所有しているプレイスメントグループを共有するには、create-resource-share コマンドを使用します。

共有プレイスメントグループを特定する

プレイスメントグループの Amazon リソースネーム (ARN) には、プレイスメントグループを所有しているアカウントの、12 桁のアカウント ID が含まれています。このアカウント ID を使用することで、自分に共有されたプレイスメントグループの所有者を特定することができます。

プレイスメントグループの ARN は、次のいずれかの方法で特定できます。詳細については、「プレイスメントグループ情報を表示する」を参照してください。

Amazon EC2 console

> 共有のプレイスメントグループを特定するには
>
> 1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[ネットワークとセキュリティ] の下にある [プレイスメントグループ] を選択します。
3. [プレイスメントグループ] の表には、自分が所有しているプレイスメントグループと、自分に共有されたプレイスメントグループが一覧表示されています。[グループ ARN] 列には、プレイスメントグループ ARN が表示されています。

   [グループ ARN] 列が表示されない場合は、右上の設定 (
   
) をクリックし、[グループ ARN] をオンにして [確認] をクリックします。

AWS CLI

共有のプレイスメントグループを特定するには

自分が所有するプレイスメントグループと自分に共有されたプレイスメントグループを一覧表示するときは、`describe-placement-groups` コマンドを使用します。レスポンスでは、`GroupId` パラメータにプレイスメントグループの ARN が表示されます。

共有プレイスメントグループ内でインスタンスを起動する

> **Important**
>
> AWS CLI を使用して、共有されたプレイスメントグループ内のインスタンスを起動するときは、`GroupId` パラメータを使用してプレイスメントグループ ID を指定する必要があります。

プレイスメントグループ名は、ユーザーが、共有されているプレイスメントグループの所有者である場合にのみ使用できます。AWS アカウント間でプレイスメントグループ名が重複する可能性を避けるため、プレイスメントグループ ID を使用することが推奨されます。

プレイスメントグループの ID は、[プレイスメントグループ] 画面の Amazon EC2 コンソールから、または `describe-placement-groups` AWS CLI コマンドを使用して確認できます。詳細については、「プレイスメントグループ情報を表示する」を参照してください。

Console

共有されたプレイスメントグループでインスタンスを起動するには

1. 手順に従ってインスタンスを起動します。ただし、次のステップを完了してプレイスメントグループの設定を指定するまでインスタンスを起動しないでください。
2. [Instance type] (インスタンスタイプ) で、サポートされているインスタンスタイプを選択します。詳細については、「プレイスメントグループのルールと制限」を参照してください。
3. [高度な詳細] を展開し、プレイスメントグループ設定を以下のように行います。
   a. [プレイスメントグループ] で、自分に共有されたプレイスメントグループを選択します。

      > **Note**
      > 同じ名前を持つプレイスメントグループがある場合は、プレイスメントグループ ID をチェックし、正しいプレイスメントグループを選択していることを確認します。

   b. パーティション戦略を持つプレイスメントグループを選択する場合は、[ターゲットパーティション] で、インスタンスを起動するパーティションを選択します。
4. [概要] パネルで以下を実行します。
   a. [インスタンス数] で、このプレイスメントグループ内で必要なインスタンスの総数を入力します。これは、後でプレイスメントグループにインスタンスを追加できない場合があるためです。
   b. インスタンスの設定を確認し、[インスタンスを起動] を選択します。

詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

AWS CLI

[To launch instances in a shared placement group] (プレイスメントグループ内でインスタンスを起動する)

`run-instances` コマンドを使用して、共有されたプレイスメントグループの、プレイスメントグループ ID を指定します。

```
aws ec2 run-instances –-placement "GroupId = pg-0123456789example"
```

[To launch instances into a specific partition of a shared partition placement group] (共有パーティションプレイスメントグループの特定のパーティションでインスタンスを起動するには)

run-instances コマンドを使用して、共有されたプレイスメントグループの、プレイスメントグループ ID とパーティション番号を指定します。

```
aws ec2 run-instances –-placement "GroupId = pg-0123456789example, PartitionNumber
 = 3"
```

> **Tip**
>
> VPC ピアリングを使用して別の AWS アカウントが所有するインスタンスを接続することで、共有クラスタープレイスメントグループが提供するレイテンシーの利点を最大限に活用できます。詳細については、「VPC ピア機能とは」を参照してください。

共有プレイスメントグループの共有解除

プレイスメントグループの所有者は、共有プレイスメントグループをいつでも共有解除することができます。

共有プレイスメントグループの共有を解除すると、次の変更が有効になります。

- プレイスメントグループを共有していた AWS アカウントでは、インスタンスを起動したり、容量を予約したりできなくなります。
- インスタンスを共有プレイスメントグループで実行していた場合、そのインスタンスはプレイスメントグループとの関連付けが解除されますが、AWS アカウントでは引き続き正常に実行されます。
- 共有プレイスメントグループでキャパシティを予約していた場合、そのキャパシティはプレイスメントグループとの関連付けが解除されますが、AWS アカウントでは引き続きアクセスできます。

共有プレイスメントグループは、次のいずれかの方法で共有解除できます。

AWS RAM console

https://console.aws.amazon.com/ram を使用して共有プレイスメントグループの共有を解除するには、「リソース共有の削除」を参照してください。

AWS CLI

AWS Command Line Interface を使用して共有プレイスメントグループの共有を解除するには、`disassociate-resource-share` コマンドを使用します。

AWS Outposts のプレイスメントグループ

AWS Outposts は、AWS のインフラストラクチャ、サービス、API、ツールをお客様のオンプレミスまで拡張するフルマネージドサービスです。AWS は、AWS Outposts マネージドインフラストラクチャへのローカルアクセスを提供することで、AWS リージョンと同じプログラミングインターフェイスを使用してオンプレミスでアプリケーションを構築して実行できるようにします。同時に、コンピューティングとストレージのローカルリソースを使用して、レイテンシーを短縮し、ローカルのデータ処理ニーズに対応します。

Outpost とは、お客様のサイトにデプロイされる AWS のコンピューティングおよびストレージキャパシティーのプールです。AWS は、AWS リージョンの一部としてこのキャパシティーを運営、監視、管理します。

ユーザーは、自分のアカウントで作成した Outposts にプレイスメントグループを作成できます。これにより、自分のサイトにある Outpost において、基盤となるハードウェア全体でインスタンスを分散できるようになります。通常のアベイラビリティーゾーンでプレイスメントグループを作成して使用するのと同じ方法で、Outposts でプレイスメントグループを作成して使用します。Outpost で分散戦略を使用してプレイスメントグループを作成する場合、プレイスメントグループがホストまたはラック全体でインスタンスを分散するように選択できます。ホスト全体でインスタンスを分散すると、単一ラックの Outpost で分散戦略を使用できます。

考慮事項

- ラックレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のラックと同じ数のインスタンスを保持できます。
- ホストレベルのスプレッドプレイスメントグループは、Outpost デプロイメント内のホストと同じ数のインスタンスを保持できます。

前提条件

Outpost は、自分のサイトにインストールする必要があります。詳細については、AWS Outposts ユーザーガイドの「Outpost を作成し、Outpost 容量を注文する」を参照してください。

Outpost でプレイスメントグループを使用するには

1. Outpost にサブネットを作成します。詳細については、AWS Outposts ユーザーガイドの「サブネットの作成」を参照してください。
2. Outpost の関連付けられたリージョンでプレイスメントグループを作成します。スプレッド戦略を使用してプレイスメントグループを作成する場合は、ホストまたはラックレベルのスプレッドを選択して、Outpost の基盤となるハードウェア全体にグループがインスタンスを分散する方法を決定できます。詳細については、「the section called “プレイスメントグループの作成”」を参照してください。
3. プレイスメントグループにインスタンスを起動します。[Subnet] (サブネット) には、ステップ 1 で作成したサブネットを選択し、[Placement group name] (プレイスメントグループ名) には、ステップ 2 で作成したプレイスメントグループを選択します。詳細については、AWS Outposts ユーザーガイドの、「Outposts でインスタンスを起動する」を参照してください。

EC2 インスタンスのネットワークの最大送信単位 (MTU)

ネットワーク接続の最大送信単位 (MTU) とは、接続を介して渡すことができる最大許容パケットサイズ (バイト単位) です。接続の MTU が大きいほど、より多くのデータを単一のパケットで渡すことができます。イーサネットフレームは、パケット (送信している実際のデータ) とそれを囲むネットワークオーバーヘッド情報で構成されています。

イーサネットフレームの形式はさまざまで、最も一般的な形式は、標準イーサネット v2 フレーム形式です。これはインターネットのほとんどでサポートされている最大のイーサネットパケットサイズである 1500 MTU をサポートします。インスタンスでサポートされている最大 MTU は、インスタンスタイプによって異なります。

Wavelength Zone にあるインスタンスには、次のルールが適用されます。

- 同じ Wavelength Zone 内の VPC で、あるインスタンスから別のインスタンスへ送られるトラフィックの MTU は 1300 です。
- Wavelength Zone 内のキャリア IP を使用し、あるインスタンスから別のインスタンスへ送られるトラフィックの MTU は 1500 です。
- Wavelength Zone とパブリック IP アドレスを使用するリージョン間で、あるインスタンスから別のインスタンスへ送られるトラフィックの MTU は 1500 です。

- Wavelength Zone とプライベート IP アドレスを使用するリージョン間で、あるインスタンスから別のインスタンスへ送られるトラフィックの MTU は 1300 です。

Outposts にあるインスタンスには、次のルールが適用されます。

- Outposts のインスタンスからリージョンのインスタンスへ送られるトラフィックの MTU は 1300 です。

Windows インスタンスのネットワーク MTU 情報を表示するには、Windows インスタンスの Amazon EC2 ユーザーガイド ガイド:EC2 インスタンスのネットワークの最大送信単位 (MTU)でこのページに切り替えます。

コンテンツ

ジャンボフレーム (9001 MTU)

ジャンボフレームでは、パケットあたりのペイロードサイズを拡張し、パケットオーバーヘッド以外のパケットの割合を高めることによって、1500 バイトを超えるデータを送信できます。同じ量の使用可能なデータを少ないパケットで送信することができます。ただし次の場合には、トラフィックの MTU は最大 1500 に制限されます。

- インターネットゲートウェイ経由のトラフィック
- リージョン間 VPC ピアリング接続経由のトラフィック
- VPN 接続経由のトラフィック
- EC2-Classic 用の特定の AWSリージョン外部にあるトラフィック

パケットが 1500 バイト以上ある場合は、フラグメント化されます。または、`Don't Fragment` フラグが IP ヘッダーに設定されている場合は削除されます。

ジャンボフレームを、インターネットバウンドトラフィックや VPC を出るトラフィックに使用する場合には慎重に行ってください。パケットは中間システムによってフラグメント化されるため、このトラフィックの速度が低下します。VPC 外に向かうトラフィックの速度を低下させずに VPC 内のジャンボ フレームを使用するには、ルートごとに MTU サイズを設定するか、または MTU サイズやルートの異なる複数の Elastic ネットワークインターフェイスを使用します。

クラスタープレイスメントグループ内にコロケーションされたインスタンスでは、考えられる最大のネットワークスループットの実現するうえでジャンボフレームが役立ちます。この場合は、ジャンボフレームを使用することが推奨されています。詳細については、プレイスメントグループを参照してください。

AWS Direct Connect を経由した VPC とオンプレミスのネットワーク間のトラフィックにはジャンボフレームを使用できます。詳細や、Jumbo Frame 機能を確認する方法については、AWS Direct Connect ユーザーガイドのネットワーク MTU 設定を参照してください。

すべての Amazon EC2 インスタンスタイプは 1500 MTU をサポートし、すべての現行世代のインスタンスタイプはジャンボフレームをサポートします。次の前世代のインスタンスタイプは、A1、C3、G2、I2、M3、および R3 のジャンボフレームをサポートしています。

サポート対象の MTU サイズの詳細については、次を参照してください。

- NAT ゲートウェイについては、「Amazon VPC ユーザーガイド」の「NAT ゲートウェイの基本」を参照してください。
- Transit Gateway の詳細については、「Amazon VPC Transit Gateway ユーザーガイド」の「MTU」を参照してください。
- ローカルゾーンについては、「AWS ローカルゾーンユーザーガイド」の「考慮事項」を参照してください。

パス MTU 検出

2 つのデバイス間のパス MTU を判断するために、パス MTU 検出 (PMTUD) が使用されます。パス MTU は、送信側ホストと受信側ホスト間のパスでサポートされている最大のパケットサイズです。2 つのホスト間のネットワークで MTU サイズに違いがある場合、PMTUD は、受信側ホストが ICMP メッセージで送信側ホストに応答するのを可能にします。この ICMP メッセージは、送信側ホストがネットワークパスに沿って最低の MTU サイズを使用し、リクエストを再送信するように指示します。このネゴシエーションがないと、リクエストが大きすぎて受信側ホストが受け取れないため、パケットドロップが発生する可能性があります。

IPv4 の場合、ホストがパスに沿って送信するパケットが、受信側ホストの MTU、あるいはデバイスの MTU よりも大きな場合、受信側ホストまたはデバイスはそのパケットをドロップし、次のような ICMP メッセージ `Destination Unreachable: Fragmentation Needed and Don't Fragment was Set` (タイプ 3、コード 4) を返します。このメッセージは送信側ホストに対し、ペイロードを複数の小さなパケットに分割し再送信することを指示します。

IPv6 プロトコルは、ネットワークのフラグメンテーションをサポートしていません。ホストがパスに沿って送信するパケットが、受信側ホストの MTU、あるいはデバイスの MTU よりも大きな場合、受信側ホストまたはデバイスはそのパケットをドロップし、次のような ICMP メッセージ `ICMPv6 Packet Too Big (PTB)` (タイプ 2) を返します。このメッセージは送信側ホストに対し、ペイロードを複数の小さなパケットに分割し再送信することを指示します。

NAT ゲートウェイやロードバランサーなどの一部のコンポーネントを介して行われる接続は、自動追跡されます。つまり、セキュリティグループの追跡は、アウトバウンド接続を試みると自動的に有効になります。接続が自動追跡されるか、セキュリティグループのルールでインバウンド ICMP トラフィックが許可されている場合は、PMTUD 応答を受信できます。

サブネットへの ICMP トラフィックを拒否するネットワークアクセスコントロールリストのエントリがある場合など、セキュリティグループレベルでトラフィックが許可されている場合でも、ICMP トラフィックはブロックされる可能性があることに注意してください。

> **Important**
>
> パス MTU 検出は、ジャンボフレームが一部のルーターによって破棄されないことを保証するものではありません。VPC のインターネットゲートウェイでは、最大 1500 バイトのパケットだけが転送されます。インターネットトラフィックでは、MTU が 1500 のパケットが推奨されています。

2 つのホスト間のパス MTU の確認

tracepath コマンドを使用して 2 つのホスト間のパス MTU を確認できます。このコマンドは、Amazon Linux を含む、多くの Linux ディストリビューションでデフォルトで提供されている `iputils` パッケージの一部です。

tracepath を使用してパス MTU を確認するには

次のコマンドを使用して、EC2 インスタンスと別のホスト間のパス MTU を確認します。宛先として DNS 名または IP アドレスを使用できます。宛先が別の EC2 インスタンスの場合、セキュリ

ティグループによりインバウンド UDP トラフィックが許可されていることを確認します。次の例では、EC2 インスタンスと amazon.com の間のパス MTU を確認します。

```
[ec2-user ~]$ tracepath amazon.com
 1?: [LOCALHOST]      pmtu 9001
 1:  ip-172-31-16-1.us-west-1.compute.internal (172.31.16.1)   0.187ms pmtu 1500
 1:  no reply
 2:  no reply
 3:  no reply
 4:  100.64.16.241 (100.64.16.241)                              0.574ms
 5:  72.21.222.221 (72.21.222.221)                             84.447ms asymm 21
 6:  205.251.229.97 (205.251.229.97)                           79.970ms asymm 19
 7:  72.21.222.194 (72.21.222.194)                             96.546ms asymm 16
 8:  72.21.222.239 (72.21.222.239)                             79.244ms asymm 15
 9:  205.251.225.73 (205.251.225.73)                           91.867ms asymm 16
...
31:  no reply
     Too many hops: pmtu 1500
     Resume: pmtu 1500
```

この例では、パス MTU は 1500 です。

Linux インスタンスの MTU の確認および設定

一部のインスタンスでは、ジャンボフレームを使用し、それ以外のドライバには標準フレームサイズを使用するように設定されています。VPC 内のネットワークトラフィックにジャンボフレームを使用したり、インターネットトラフィックに標準フレームを使用したりしたい場合があります。いずれにしても、予想したとおりにインスタンスが動作することを確認することをお勧めします。このセクションの手順に従って、ネットワークインターフェイスの MTU 設定を確認し、必要に応じてそれらを変更することができます。

Linux インスタンス上の MTU 設定を確認するには

以下の ip コマンドを使用して、現在の MTU 値を確認できます。出力例では、*mtu 9001* が、このインスタンスにジャンボフレームが使用されていると示していることに注意してください。

```
[ec2-user ~]$ ip link show eth0
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9001 qdisc pfifo_fast state UP mode
 DEFAULT group default qlen 1000
    link/ether 02:90:c0:b7:9e:d1 brd ff:ff:ff:ff:ff:ff
```

Linux インスタンス上の MTU 値を設定するには

1. MTU 値は、ip コマンドを使用して設定できます。次のコマンドで、目的の MTU 値を 1500 に設定できますが、代わりに 9001 を使用します。

```
[ec2-user ~]$ sudo ip link set dev eth0 mtu 1500
```

2. (オプション) 再起動後もネットワーク MTU 設定を維持するには、オペレーティングシステムのタイプに基づいて、次の設定ファイルを変更します。

   - Amazon Linux 2 の場合、次の行を /etc/sysconfig/network-scripts/ifcfg-eth0 ファイルに追加します。

     ```
     MTU=1500
     ```

     次の行を /etc/dhcp/dhclient.conf ファイルに追加します。

     ```
     request subnet-mask, broadcast-address, time-offset, routers, domain-name,
      domain-search, domain-name-servers, host-name, nis-domain, nis-servers, ntp-
     servers;
     ```

   - Amazon Linux の場合は、以下の行を /etc/dhcp/dhclient-eth0.conf ファイルに追加します。

     ```
     interface "eth0" {
     supersede interface-mtu 1500;
     }
     ```

   - その他の Linux ディストリビューションの場合は、特定のドキュメントを参照してください。

3. (オプション) インスタンスを再起動し、MTU 設定が正しいことを確認します。

トラブルシューティング

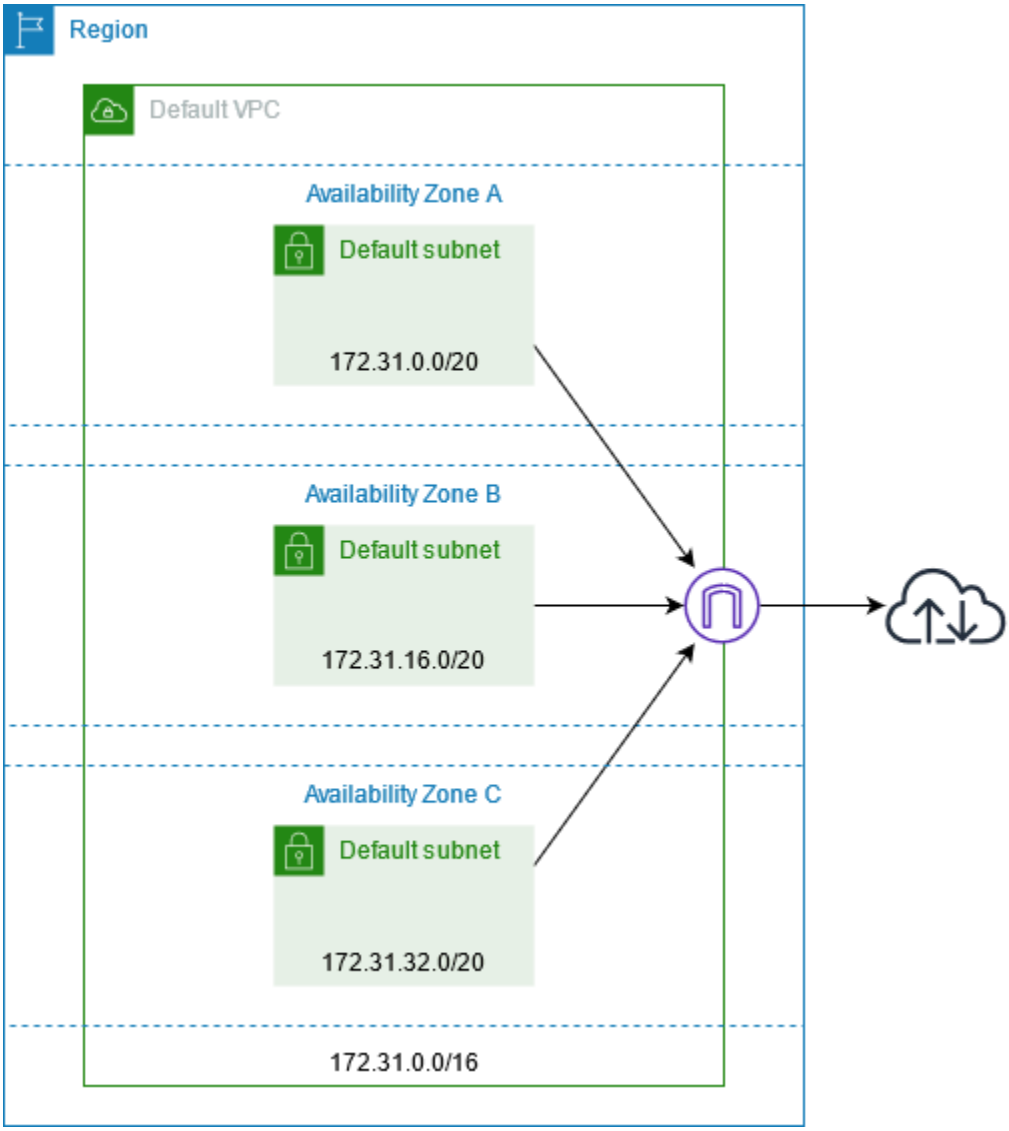
ジャンボフレームを使用したときに EC2 インスタンスと Amazon Redshift クラスター間の接続に問題が発生する場合は、「Amazon Redshift 管理ガイド」の「クエリがハングしたようになる」を参照してください。

仮想プライベートクラウド

Amazon Virtual Private Cloud (Amazon VPC) を使用すると、AWS クラウドで論理的に分離された独自の領域内に、仮想プライベートクラウド (VPC) と呼ばれる仮想ネットワークを定義できます。AWS のリソース (Amazon EC2 インスタンスなど) を VPC のサブネット内に作成できます。VPC は、お客様自身のデータセンターで運用されている従来のネットワークによく似ていますが、AWS からスケーラブルなインフラストラクチャを使用できるというメリットがあります。お客様の VPC はお客様が設定できます。IP アドレスレンジの選択、サブネットの作成、ルートテーブル、ネットワークゲートウェイ、セキュリティの設定ができます。VPC のインスタンスをインターネットまたは独自のデータセンターに接続できます。

デフォルトの VPC

AWS アカウントを作成すると、各リージョンにデフォルト VPC が作成されます。デフォルトの VPC は、設定済みですぐに使用できる VPC です。例えば、それぞれのデフォルトの VPC では、各アベイラビリティーゾーンがデフォルトのサブネットを持ちます。この VPC には、インターネットゲートウェイがアタッチされ、メインルートテーブルでは、すべて (0.0.0.0/0) のトラフィックをインターネットゲートウェイに送信するルートが定義されています。または、必要に応じた独自の VPC を作成および設定をすることができます。



追加の VCP を作成する

以下の手順で、必要なサブネット、ゲートウェイ、ルーティング構成を持つ VPC を作成します。

VPC を作成するには

1. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
2. [Create VPC (VPC の作成)] を選択します。
3. Resources to create (作成するリソース) で、VPC only (VPC など) を選択します。
4. [名前タグの自動生成] に、VPC の名前を入力します。
5. [IPv4 CIDR block] (IPv4 CIDR ブロック) の場合は、デフォルトの候補のままにするか、アプリケーションまたはネットワークが必要とする CIDR ブロックを入力します。

6. [アベイラビリティーゾーンの数] で [2] を選択すると、複数のアベイラビリティーゾーンでインスタンスを起動して高可用性を確保できます。
7. インスタンスをインターネットからアクセスできるようにするには、次のいずれかを実行します。
   - インスタンスをパブリックサブネットに配置できる場合は、[Number of public subnets] (パブリックサブネットの数) に 0 以外の値を選択します。[DNS options] (DNS オプション) で両方のオプションを選択したままにします。今すぐまたは後で、オプションでプライベートサブネットを追加することができます。
   - インスタンスがプライベートサブネット内にある必要がある場合は、[Number of public subnets] (パブリックサブネットの数) で [0] を選択します。[プライベートサブネットの数] には、必要に応じて数を選択します (使用可能な値としては、アベイラビリティーゾーンごとに 1 つまたは 2 つのプライベートサブネットに対応しています)。[NAT ゲートウェイ] の場合、両方のアベイラビリティーゾーンのインスタンスがアベイラビリティーゾーン間で大量のトラフィックを送受信する場合は、[アベイラビリティーゾーンごとに 1 つ] を選択します。それ以外の場合は、[1 つのアベイラビリティーゾーンで] を選択し、NAT ゲートウェイと同じアベイラビリティーゾーンでクロスゾーントラフィックを送受信するインスタンスを起動します。
8. [Customize subnet CIDR blocks] (サブネット CIDR ブロックのカスタマイズ) を展開します。デフォルトの候補をそのまま使用するか、各サブネットの CIDR ブロックを入力します。詳細については、「Amazon VPC ユーザーガイド」の「サブネット CIDR ブロック」を参照してください。
9. 選択した内容に基づいて作成される VPC リソースが表示される [Preview] (プレビュー) ペインを確認してください。
10. [Create VPC (VPC の作成)] を選択します。

インスタンスからインターネットにアクセスする

VPC はパブリック IP アドレスと DNS ホスト名を割り当てるように設定され、メインルートテーブルには VPC にアタッチされたインターネットゲートウェイへのルートが設定されているため、デフォルトサブネットで起動されたインスタンスは、インターネットにアクセスすることが可能です。

VPC で作成したサブネットについて以下のいずれかを実行して、これらのサブネットで起動したインスタンスから、インターネットにアクセスできることを確認します。

- インターネットゲートウェイを設定します。詳細については、「Amazon VPC ユーザーガイド」の「インターネットゲートウェイを使用してインターネットに接続する」 を参照してください。

- パブリックな NAT ゲートウェイを設定します。詳細については、Amazon VPC ユーザーガイドのプライベートサブネットからインターネットにアクセスするを参照してください。

インスタンスへの SSH アクセス

インスタンスに接続するには、ネットワークからインスタンスへの SSH トラフィックを承認する必要があります。インスタンスを起動するときはキーペアを指定し、インスタンスに接続するときは .pem ファイルを指定する必要があります。詳細については、「プライベートキーを見つけ、許可を設定する」を参照してください。

共有サブネット

EC2 インスタンスを共有 VPC サブネットで起動するときは、次の点に注意してください:

- 参加者は、共有サブネット ID を渡すことで、共有 VPC サブネットでインスタンスを実行できます。参加者がインスタンスを実行するときにセキュリティグループ ID またはネットワークインターフェイス ID を渡す場合、参加者はセキュリティグループまたはネットワークインターフェイスを所有している必要があります。
- 参加者は、共有 VPC サブネットで作成したインスタンスを起動、停止、終了、説明できます。参加者は、共有 VPC サブネットで作成したインスタンスを起動、停止、終了、説明できません。
- VPCオーナーは、共有 VPC サブネットで作成したインスタンスを起動、停止、終了、説明できません。

詳細については、「Amazon VPC ユーザーガイド」の「他のアカウントとVPCを共有する」を参照してください。

IPv6 専用サブネット

IPv6 のみのサブネットで起動される EC2 インスタンスは、IPv6 アドレスを受信しますが、IPv4 アドレスは受信しません。IPv6 のみのサブネットで起動するインスタンスは、Nitro システム上に構築されたインスタンスである必要があります。

セキュリティとコンプライアンスの目標を満たすようにAmazon EC2 を設定し、Amazon EC2 リソースの保護に役立つ他の サービスの使用方法を学びます。

AWS では、クラウドのセキュリティが最優先事項です。AWS のお客様は、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャから利点を得られます。

セキュリティは、AWS とお客様の間の共有責任です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ - AWS は、AWS クラウドで AWS サービスを実行するインフラストラクチャを保護する責任を負います。また AWS は、安全に使用できるサービスを提供します。AWS コンプライアンスプログラムの一環として、サードパーティーの監査が定期的にセキュリティの有効性をテストおよび検証しています。Amazon EC2 に適用するコンプライアンスプログラムの詳細については、AWSコンプライアンスプログラムによる対象範囲の
- クラウド内のセキュリティ - お客様は以下の事項について責任を負います。
  - VPC とセキュリティグループの設定など、インスタンスへのネットワークアクセスの制御。詳細については、ネットワークトラフィックの制御を参照してください。
  - インスタンスへの接続に使用する認証情報の管理。
  - ゲスト OS と、ゲスト OS にデプロイされたソフトウェア (更新およびセキュリティパッチを含む) の管理。詳細については、Amazon EC2 での更新管理を参照してください。
  - インスタンスにアタッチされた IAM ロールと、それらのロールに関連付けられたアクセス許可の設定。詳細については、Amazon EC2 の IAM ロールを参照してください。

このドキュメントは、Amazon EC2使用時における責任共有モデルの適用法を理解するのに役立ちます。ここでは、セキュリティやコンプライアンスに関する目標を達成できるようにAmazon EC2を設定する方法について説明します。Amazon EC2リソースのモニタリングやセキュリティ確保に役立つ他の AWS サービスの使用方法についても説明します。

コンテンツ

- Amazon EC2 の Identity and Access Management
- Amazon EC2 のキーペアと Amazon EC2 インスタンス
- Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ
- インターフェイス VPC エンドポイントを使用して Amazon EC2 にアクセスします。
- Amazon EC2 での更新管理
- Amazon EC2 のコンプライアンス検証
- NitroTPM

Amazon EC2 でのインフラストラクチャセキュリティ

マネージドサービスとして、Amazon Elastic Compute Cloud は AWS グローバルネットワークセキュリティによって保護されています。AWS セキュリティサービスと AWS がインフラストラクチャを保護する方法については、「AWS クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「セキュリティの柱 - AWS Well-Architected フレームワーク」の「インフラストラクチャ保護」を参照してください。

AWS が公開している API コールを使用し、ネットワーク経由で Amazon EC2にアクセスします。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS)。TLS 1.2、できれば TLS 1.3 が必要です。
- DHE (Ephemeral Diffie-Hellman) や ECDHE (Elliptic Curve Ephemeral Diffie-Hellman) などの Perfect Forward Secrecy (PFS) を使用した暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

また、リクエストには、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

詳細については、セキュリティの柱 - AWS Well-Architected フレームワークの「Infrastructure Protection」(インフラストラクチャの保護) を参照してください。

ネットワークの隔離

仮想プライベートクラウド (VPC) は、AWS クラウド内の論理的に隔離された領域にある仮想ネットワークです。ワークロードまたは組織エンティティ単位でインフラストラクチャを隔離するには、個別の VPC を使用します。

サブネットは、ある範囲の IP アドレスが示す VPC 内の領域です。インスタンスを起動する場合には、VPC 内のあるサブネットにおいて起動することになります。サブネットを使用すると、単一の VPC 内で多階層ウェブアプリケーションの各階層 (ウェブサーバー、アプリケーションサーバーおよびデータベースサーバーなど) を隔離できます。インターネットからの直接アクセスを認めるべきでないインスタンスには、プライベートサブネットを使用します。

プライベート IP アドレスを使用して VPC から Amazon EC2 API を呼び出すには、AWS PrivateLink を使用します。詳細については、「インターフェイス VPC エンドポイントを使用して Amazon EC2 にアクセスします。」を参照してください。

物理ホストでの分離

同じ物理ホストで実行される異なる EC2 インスタンスは、個別の物理ホストで実行されるかのように隔離されます。ハイパーバイザーが CPU およびメモリを隔離し、各インスタンスには、生ディスクデバイスへのアクセスに代わる仮想ディスクへのアクセスが提供されます。

インスタンスを停止または終了すると、そのインスタンスに割り当てられていたメモリをハイパーバイザーがスクラブ (ゼロに設定) し、そのメモリが新たなインスタンスに割り当てられ、すべてのストレージブロックがリセットされます。これは、お客様のデータが誤って他のインスタンスに引き渡されないようにするための処理です。

ネットワーク MAC アドレスは、AWS ネットワークインフラストラクチャが各インスタンスに対し動的に割り当てます。IP アドレスは、AWS ネットワークインフラストラクチャが各インスタンスに対し動的に割り当てるか、要認証 API リクエストを介して EC2 管理者が割り当てます。AWS ネットワークは、インスタンスは割り当てられた MAC および IP アドレスからのみトラフィックを送信できます。それ以外のトラフィックは除外されます。

デフォルトでは、インスタンスは、そのインスタンス宛ではないトラフィックを受信することはできません。インスタンスにおいて、ネットワークアドレス変換 (NAT、network address translation)、ルーティングまたはファイアウォールといったサービスの実行が必要な場合には、ネットワークインターフェースの送信元/送信先チェックを無効化できます。

ネットワークトラフィックの制御

EC2 インスタンスへのネットワークトラフックを制御するには、以下のオプションを検討します。

- セキュリティグループを使用してインスタンスへのアクセスを制限する。この方法を使うと、例えば、社内ネットワークのアドレス範囲に属するアドレスからのトラフィックのみ認めるといったことができます。
- インターネットからの直接アクセスを認めるべきでないインスタンスには、プライベートサブネットを使用します。プライベートサブネット内にあるインスタンスからのインターネットアクセスに、要塞ホストまたは NAT ゲートウェイを使用する。
- AWS Virtual Private Network または AWS Direct Connectを使用して、リモートネットワークから VPC へのプライベート接続を確立する。詳細については、ネットワークから Amazon VPC への接続オプションを参照してください。
- VPC フローログを使用して、インスタンスに到達するトラフィックを監視します。
- AWS Security Hubを使用して、インスタンスからの意図しないネットワークアクセスを確認する。
- EC2 Instance Connectを使用して、SSH キーの共有および管理が不要な Secure Shell (SSH) を使いインスタンスに接続する。
- インバウンドSSH ポートを開き、SSH キーを管理する代わりに、AWS Systems Managerセッションマネージャーを使用してインスタンスにリモートアクセスする。
- インバウンドSSH ポートを開き、SSH キーを管理する代わりに、AWS Systems Manager 実行コマンドを使用して、共通の管理タスクを自動化する。

各 Amazon EC2 インスタンスへのネットワークアクセスの制限に加えて、Amazon VPC は、インラインゲートウェイ、プロキシサーバー、さまざまなネットワークモニタリングオプションなど、追加のネットワークセキュリティ管理の実装をサポートしています。

Amazon EC2の耐障害性

AWS のグローバルインフラストラクチャは AWS リージョンとアベイラビリティーゾーンを中心として構築されます。リージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている複数の物理的に独立および隔離されたアベイラビリティーゾーンがあります。アベイラビリティーゾーンでは、ゾーン間で中断することなく自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用することができます。アベイラビリティーゾーン

は、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、耐障害性、および拡張性が優れています。

データまたはアプリケーションをより広範な地理的距離にわたってレプリケートする必要がある場合は、AWS Local Zonesを使用します。AWS ローカルゾーンは AWS リージョンの拡張であり、ユーザーに近い場所に配置されます。Local Zones は、インターネットへの独自の接続を持ち、AWS Direct Connect をサポートします。すべての AWS リージョンと同じように、AWS Local Zonesは他の AWS リージョンから完全に分離されています。

AWS ローカルゾーン内でデータまたはアプリケーションをレプリケートする必要がある場合、次のいずれかのゾーンをフェイルオーバーゾーンとして使用することを AWS はお勧めします。

- 別のローカルゾーン
- 親ゾーンではないリージョン内のアベイラビリティーゾーン。親ゾーンを確認するには、describe-availability-zones コマンドを使用できます。

AWS リージョンとアベイラビリティーゾーンの詳細については、AWS グローバルインフラストラクチャを参照してください。

Amazon EC2 は、AWS グローバルインフラストラクチャに加え、データ耐障害性をサポートする以下の機能を提供します。

- リージョンで AMI をコピーする機能
- リージョン間の EBS スナップショットをコピーする機能
- Amazon Data Lifecycle Manager を使用した EBS-backed AMI の自動化
- Amazon Data Lifecycle Managerを使用して EBS スナップショットを自動化する機能
- Amazon EC2 Auto Scalingを使用してフリートの健全性や可用性を維持する機能
- Elastic Load Balancing を使用して、単一のまたは複数のアベイラビリティーゾーンにある複数のインスタンスの間で受信トラフィックを分散する機能

Amazon EC2 でのデータ保護

AWS 責任共有モデルは、Amazon Elastic Compute Cloud のデータ保護に適用されます。このモデルで説明されているように、AWS は、AWS クラウド のすべてを実行するグローバルインフラストラクチャを保護する責任を担います。このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任はユーザーにあります。また、使用する AWS のサービスのセキュリティ設

定と管理タスクもユーザーの責任となります。データプライバシーの詳細については、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、「AWS セキュリティブログ」に投稿された「AWS 責任共有モデルおよび GDPR」のブログ記事を参照してください。

データを保護するため、AWS アカウント の認証情報を保護し、AWS IAM Identity Center または AWS Identity and Access Management (IAM) を使用して個々のユーザーをセットアップすることをお勧めします。こうすると、それぞれのジョブを遂行するために必要なアクセス許可のみを各ユーザーに付与できます。また、以下の方法でデータを保護することをお勧めします。

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 が必須です。TLS 1.3 が推奨されます。
- AWS CloudTrail で API とユーザーアクティビティロギングをセットアップします。
- AWS のサービス 内でデフォルトである、すべてのセキュリティ制御に加え、AWS の暗号化ソリューションを使用します。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密データの検出と保護を支援します。
- コマンドラインインターフェイスまたは API により AWS にアクセスするときに FIPS 140-2 検証済み暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。利用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

お客様の E メールアドレスなどの機密情報やセンシティブ情報は、タグや名前フィールドなどの自由形式のフィールドに配置しないことを強くお勧めします。これには、コンソール、API、AWS CLI、または AWS SDK を使用して Amazon EC2 またはその他の AWS のサービス で作業する場合が含まれます。名前に使用する自由記述のテキストフィールドやタグに入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

トピック

- Amazon EBS のデータセキュリティ
- 保管中の暗号化
- 転送中の暗号化

Amazon EBS のデータセキュリティ

Amazon EBS ボリュームは、初期化されていない raw ブロックデバイスとして表示されます。これらのデバイスは、EBS インフラストラクチャ上に作成される論理デバイスであり、Amazon EBS サービスは、お客様による利用または再利用の前に、デバイスが論理的に空になっている (つまり、raw ブロックがゼロになっている、または暗号で擬似ランダムデータが含まれている) ようにします。

DoD 5220.22-M (National Industrial Security Program Operating Manual) や NIST 800-88 (Guidelines for Media Sanitization) に詳述されているような、使用後もしくは使用前 (またはその両方) に特定の方法を使用してすべてのデータを消去する必要がある手順がある場合、Amazon EBS でこれを行うことができます。ブロックレベルのアクティビティは、Amazon EBS サービス内の基盤となるストレージメディアに反映されます。

保管中の暗号化

EBS ボリューム

Amazon EBS暗号化は、EBS ボリュームおよびスナップショット向けの暗号化ソリューションです。それは AWS KMS keys を使用します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS 暗号化」を参照してください。

インスタンスストアボリューム

NVMe インスタンスストアボリューム内のデータは、インスタンスのハードウェアモジュールに実装されている XTS-AES-256 暗号を使用して暗号化されます。ローカルに接続された NVMe ストレージデバイスに書き込まれるデータの暗号化に使用されるキーは、お客様ごと、ボリュームごとに異なります。キーはハードウェアモジュールによって生成され、ハードウェアモジュールの内部にのみ存在します。AWS ユーザーはハードウェアモジュールにはアクセスできません。暗号化キーは、インスタンスが停止または終了して復元できないときに破棄されます。この暗号化を無効にしたり、独自の暗号キーを指定したりすることはできません。

H1、D3、D3en インスタンス上にある HDD インスタンスストアボリュームのデータは、XTS-AES-256 とワンタイムキーを使用して暗号化されます。

インスタンスを、停止、休止、または終了するとき、インスタンスストアボリュームのストレージの各ブロックはリセットされます。そのため、別のインスタンスのインスタンスストアを通じてデータにアクセスすることはできません。

「メモリ」

メモリの暗号化は、次のインスタンスで有効になります。

- AWS Graviton プロセッサを搭載したインスタンス。 AWSGraviton2、AWS Graviton3、AWS Graviton3E は常時オンのメモリ暗号化をサポートしています。暗号化キーは、ホストシステム内で安全に生成され、ホストシステムから離れることはなく、ホストの再起動または電源切断時に破棄されます。詳細については、「AWS Graviton プロセッサ」を参照してください。
- M6i インスタンスなどの第 3 世代 Intel Xeon スケーラブルプロセッサ (Ice Lake) と M7i インスタンスなどの第 4 世代 Intel Xeon スケーラブルプロセッサ (Sapphire Rapids) を搭載したインスタンス。これらのプロセッサーは、インテル· トータル· メモリー暗号化 (TME) を使用した常時オンのメモリー暗号化をサポートします。
- M6a インスタンスなどの第 3 世代 AMD EPYC プロセッサ (Milan) と M7a インスタンスなどの第 4 世代 AMD EPYC プロセッサ (Genoa) を搭載したインスタンス。これらのプロセッサーは、AMD Secure Memory Encryption (SME) を使用した常時オンのメモリー暗号化をサポートします。第 3 世代 AMD EPYC プロセッサ (Milan) を搭載したインスタンスは、AMD Secure Encrypted Virtualization-Secure Nested Paging (SEV-SNP) もサポートしています。

転送中の暗号化

物理レイヤーでの暗号化

AWS グローバルネットワーク上の AWS リージョンを流れるすべてのデータは、AWS の安全な施設を離れる前に、物理層で自動的に暗号化されます。AZ 間のトラフィックはすべて暗号化されます。追加的な暗号化レイヤーでは、このセクションに記載されているもの以外にも、保護が提供されている場合があります。

Amazon VPC および Transit Gateway のクロスリージョンピアリング接続によって提供される暗号化

Amazon VPC およびTransit Gateway のピアリング接続を使用する、すべてのクロスリージョントラフィックは、リージョンからの送信時に自動的に一括で暗号化されます。このセクションで前述したように、すべてのクロスリージョントラフィックの物理レイヤーには、追加の暗号化レイヤーが自動的に提供されます。

インスタンス間での暗号化

AWS では、すべてのタイプの EC2 インスタンス間において安全でプライベートな接続を提供しています。さらに、一部のインスタンスタイプでは、基盤となる Nitro System ハードウェアのオフ

ロード機能を使用して、インスタンス間の転送中のトラフィックを自動的に暗号化します。この暗号化では、256 ビットの暗号化による関連データによる認証暗号化 (AEAD) アルゴリズムを使用します。ネットワークのパフォーマンスには影響しません。インスタンス間でこの追加の転送中トラフィック暗号化をサポートするには、次の要件を満たす必要があります。

- インスタンスは、次のインスタンスタイプを使用します。
  - 汎用: M5dn、M5n、M5zn、M6a、M6i、M6id、M6idn、M6in、M7a、M7g、M7gd、M7i、M7i-flex
  - コンピューティング最適化:
    C5a、C5ad、C5n、C6a、C6gn、C6i、C6id、C6in、C7a、C7g、C7gd、C7gn、C7i
  - メモリ最適化:
    R5dn、R5n、R6a、R6i、R6idn、R6in、R6id、R7a、R7g、R7gd、R7i、R7iz、U-3tb1、U-6tb1、U-9tb
  - ストレージ最適化: D3、D3en、I3en、I4g、I4i、Im4gn、Is4gen
  - 高速コンピューティング:
    DL1、DL2q、G4ad、G4dn、G5、Inf1、Inf2、P3dn、P4d、P4de、P5、Trn1、Trn1n、VT1
  - ハイパフォーマンスコンピューティング: Hpc6a、Hpc6id、Hpc7a、Hpc7g
- 各インスタンスは同じリージョンにあるものとします。
- 各インスタンスは同じ VPC 内、あるいはピア接続された VPC 内にあり、トラフィックは仮想ネットワークのデバイスもしくはサービス (ロードバランサーや Transit Gateway など) を通過しないものとします。

このセクションで先に述べたように、すべてのトラフィックにおける物理レイヤーには、そのトラフィックが AWS の保護された設備を離れる前に、追加の暗号化レイヤーが自動的に提供されています。

AWS CLIを使用してインスタンス間のトランジットトラフィックを暗号化するインスタンスタイプを表示するには、以下のようにします。

次の describe-instances コマンドを使用します。

```
aws ec2 describe-instance-types \
    --filters Name=network-info.encryption-in-transit-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

AWS Outposts との間の暗号化

Outpost は、AWS ホームリージョンとの間にサービスリンクと呼ばれる特別なネットワーク接続を作成し、オプションとして指定した VPC サブネットとのプライベート接続も可能です。これらの接続上のすべてのトラフィックは完全に暗号化されます。詳細については、AWS Outposts ユーザーガイドのサービスリンクによる接続および転送中の暗号化を参照してください。

リモートアクセスの暗号化

SSH は、 Linux や インスタンスへのリモートアクセスのために、直接的もしくは EC2 Instance Connect 経由での安全な通信チャネルを提供します。AWS Systems Manager Session Manager および Run Command を使用したインスタンスへのリモートアクセスは、TLS 1.2 を使用して暗号化されます。また、接続を確立するリクエストは SigV4 を使用して署名され、AWS Identity and Access Management により認証ならびに許可されます。

クライアントと Amazon EC2 インスタンスの間で送受信される機密データを、Transport Layer Security (TLS) などの暗号化プロトコルを使用して暗号化することは、お客様の責任範囲です。

Amazon EC2 の Identity and Access Management

セキュリティ認証情報により、AWS サービスでユーザーの身分が証明され、Amazon EC2 リソースなどの AWS リソースを無制限に使用できる許可が付与されます。Amazon EC2 および AWS Identity and Access Management (IAM) の機能を使用して、他のユーザー、サービス、およびアプリケーションがユーザーの Amazon EC2 リソースを使用できるようにします。その際、ユーザーのセキュリティ認証情報は共有されません。他のユーザーが AWS アカウントのリソースを使用する方法を制御するには IAM を、Amazon EC2 インスタンスへのアクセスを制御するにはセキュリティグループを使用できます。Amazon EC2 のリソースの完全使用または制限付き使用のどちらを許可するか選択できます。

IAM を使用して AWS リソースを保護するためのベストプラクティスについては、「IAM でのセキュリティのベストプラクティス」を参照してください。

コンテンツ

- Amazon EC2 の IAM ロール
- Linux インスタンス用のインバウンドトラフィックの承認

インスタンスへのネットワークアクセス

セキュリティグループは、1 つ以上のインスタンスに到達できるトラフィックを制御するファイアウォールとして機能します。インスタンスを起動するときに、そのインスタンスに 1 つまたは複数のセキュリティグループを割り当てることができます。セキュリティグループのそれぞれに、そのインスタンスへのトラフィックを制御するルールを追加できます。セキュリティグループルールはいつでも変更できます。新しいルールは、そのセキュリティグループが割り当てられているインスタンスすべてに自動的に適用されます。

詳細については、Linux インスタンス用のインバウンドトラフィックの承認を参照してください。

Amazon EC2 のアクセス許可属性

ユーザーの組織には、複数のAWS アカウントがある場合があります。Amazon EC2 では、Amazon Machine Image (AMI) および Amazon EBS スナップショットを使用できる追加の AWS アカウントを指定できます。このアクセス権限は AWS アカウントレベルでのみ有効です。特定の AWS アカウント内の特定ユーザーのアクセス権限を制限することはできません。指定した AWS アカウントのすべてのユーザーが、AMI またはスナップショットを使用できます。

AMI ごとに `LaunchPermission` 属性があり、AMI にアクセスできる AWS アカウントを制御します。詳細については、を参照してください。AMI の公開

Amazon EBS スナップショットごとに `createVolumePermission` 属性があり、スナップショットを使用できる AWS アカウントを制御します。詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS スナップショットの共有」を参照してください。

IAM および Amazon EC2

IAM を使って以下を行えます。

- AWS アカウント にユーザーとグループを作成する
- お客様の AWS アカウント でユーザーごとに固有のセキュリティ認証情報を割り当てる
- AWS のリソースを使用してタスクを実行するために各ユーザーのアクセス権限を制御する
- 別の AWS アカウント のユーザーがお客様の AWS のリソースを共有できるようにする

- AWS アカウント にロールを作成し、それを行えるユーザーまたはサービスを定義する
- お客様の企業用の既存のアイデンティティを使用し、AWS のリソースを使用してタスクを実行するようにアクセス権限を与える

Amazon EC2 と組み合わせて IAM を使用すると、組織のユーザーが特定の Amazon EC2 API アクションを使用してタスクを実行できるか、そして、特定の AWS リソースを使用できるかを制御できます。

このトピックには、以下の質問に対する回答があります。

- IAM でグループとユーザーを作成するには、どうすればよいですか?
- ポリシーを作成するには、どうすればよいですか?
- Amazon EC2 でタスクを実行するには、どのような IAM ポリシーが必要ですか?
- Amazon EC2 でアクションを実行するための許可を与えるには、どうすればよいですか?
- Amazon EC2 の特定のリソースでアクションを実行するための許可を与えるには、どうすればよいですか?

ユーザー、グループ、ロールを作成する

AWS アカウント 用のユーザーとグループを作成して、必要な許可を割り当てることができます。ベストプラクティスとして、ユーザーは IAM ロールを引き受けて許可を取得する必要があります。AWS アカウント のユーザーとグループを設定する方法の詳細については、「Amazon EC2 を使用するように設定する」を参照してください。

IAM ロールは、特定の許可があり、アカウントで作成できるもう 1 つの IAM アイデンティティです。IAM ロールは、ID が AWS で実行できることとできないことを決定する許可ポリシーを持つ AWS ID であるという点で IAM ユーザーと似ています。ただし、ユーザーは 1 人の特定の人に一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。また、ロールには標準の長期認証情報 (パスワードやアクセスキーなど) も関連付けられません。代わりに、ロールを引き受けると、ロールセッション用の一時的なセキュリティ認証情報が提供されます。IAM ロールを作成して許可を付与する方法の詳細については、「Amazon EC2 の IAM ロール」を参照してください。

関連トピック

IAM の詳細については、以下を参照してください。

- Amazon EC2 の IAM ポリシー
- Amazon EC2 の IAM ロール
- AWS Identity and Access Management (IAM)
- IAM ユーザーガイド

Amazon EC2 の IAM ポリシー

デフォルトでは、ユーザーには Amazon EC2 リソースを作成または変更、または Amazon EC2 API、Amazon EC2 コンソールあるいは CLI を使用するタスクを実行する許可がありません。ユーザーがリソースを作成または変更、およびタスクを実行できるようにするには、IAM ポリシーを作成する必要があります。これによって、必要な特定のリソースおよび API アクションを使用するための許可をユーザーに付与し、その後、ポリシーをその許可が必要なユーザー、グループまたは IAM ロールにアタッチします。

ポリシーをユーザー、ユーザーのグループ、またはロールにアタッチする場合、ポリシーによって特定リソースの特定タスクを実行するユーザーの許可が許可または拒否されます。IAM ポリシーの一般的な情報については、「IAM ユーザーガイド」の「IAM の許可とポリシー」を参照してください。カスタム IAM ポリシーの管理と作成の詳細については、「IAM ポリシーの管理」を参照してください。

ご利用開始にあたって

IAM ポリシーは、1 つ以上の Amazon EC2 アクションを使用するアクセス許可を付与または拒否する必要があります。さらに、このアクションで使用できるリソース (すべてのリソースか、場合によっては特定のリソース) も指定する必要があります。このポリシーには、リソースに適用する条件も含めることができます。

Amazon EC2 では、リソースレベルのアクセス許可が部分的にサポートされます。これは、一部の EC2 API アクションでは、ユーザーがそのアクションに使用できるリソースを指定できないことを意味します。代わりに、ユーザーがそのアクションにすべてのリソースを使用することを許可する必要があります。

タスク	トピック
ポリシーの基本構造について	ポリシー構文
ポリシーでのアクションの定義	Amazon EC2 のアクション

タスク	トピック
ポリシーでの特定のリソースの定義	Amazon EC2 用の Amazon リソースネーム (ARN)
リソースの使用への条件の適用	Amazon EC2 の条件キー
Amazon EC2 での使用可能なリソースレベルのアクセス許可の使用	Amazon EC2 のアクション、リソース、条件キー
ポリシーのテスト	ユーザーが必要なアクセス許可を持っているかどうかの確認
IAM ポリシーを生成する	アクセスアクティビティに基づいてポリシーを生成する
CLI または SDK のサンプルポリシー	AWS CLI または AWS SDK で使用するサンプルポリシー
Amazon EC2 コンソールのサンプルポリシー	Amazon EC2 コンソールで機能するサンプルポリシー

ユーザー、グループ、およびロールに許可を付与する

以下は、ニーズを満たす場合に利用できる AWS 管理ポリシーの例です。

- `PowerUserAccess`
- `ReadOnlyAccess`
- `AmazonEC2FullAccess`
- `AmazonEC2ReadOnlyAccess`

Amazon EC2 で使用できる AWS 管理ポリシーの詳細については、「Amazon Elastic Compute Cloud の AWS 管理ポリシー」を参照してください。

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「アクセス許可一式を作成」の手順を実行します。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。
- IAM ユーザー:
  - ユーザーが継承できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

ポリシーの構造

次のトピックでは、IAM ポリシーの簡単な構造について説明します。

コンテンツ

- ポリシー構文
- Amazon EC2 のアクション
- Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可
- Amazon EC2 用の Amazon リソースネーム (ARN)
- Amazon EC2 の条件キー
- ユーザーが必要なアクセス許可を持っているかどうかの確認

ポリシー構文

IAM ポリシーは 1 つ以上のステートメントで構成される JSON ドキュメントです。各ステートメントは次のように構成されます。

```
{
  "Statement":[{
    "Effect":"effect",
    "Action":"action",
    "Resource":"arn",
    "Condition":{
```

```
      "condition":{
        "key":"value"
        }
      }
    }
  ]
}
```

ステートメントはさまざまなエレメントで構成されます。

- Effect: effect は、Allow または Deny にすることができます。デフォルトでは、 ユーザーはリソースおよび API アクションを使用するアクセス許可がないため、リクエストはすべて拒否されます。明示的な許可はデフォルトに上書きされます。明示的な拒否はすべての許可に優先します。
- [Action]: action は、アクセス許可を付与または拒否する対象とする、特定の API アクションです。action の指定については、Amazon EC2 のアクションを参照してください。
- [Resource]: アクションによって影響を及ぼされるリソースです。Amazon EC2 API アクションの中には、アクションによって作成/変更できるリソースをポリシー内で特定できるものもあります。Amazon リソースネーム (ARN) を使用して、またはステートメントがすべてのリソースに適用されることを示すワイルドカード (*) を使用して、リソースを指定します。詳細については、Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可を参照してください。
- [Condition]: condition はオプションです。ポリシーの発効条件を指定するために使用します。Amazon EC2 の条件を指定する方法については、Amazon EC2 の条件キーを参照してください。

ポリシー要件の詳細については、「IAM ユーザーガイド」の「IAM JSON ポリシーリファレンス」を参照してください。Amazon EC2 の IAM ポリシーステートメントの例については、「AWS CLI または AWS SDK で使用するサンプルポリシー」を参照してください。

Amazon EC2 のアクション

IAM ポリシーステートメントで、IAM をサポートするすべてのサービスから任意の API アクションを指定できます。Amazon EC2 の場合、API アクション ec2: の名前に次のプレフィクスを使用します。例: ec2:RunInstances および ec2:CreateImage。

単一のステートメントに複数のアクションを指定するには、次のようにコンマで区切ります。

```
"Action": ["ec2:action1", "ec2:action2"]
```

ワイルドカードを使用して複数のアクションを指定することもできます。例えば、以下のようにDescribeという単語で始まる名前のすべてのアクションを指定できます。

```
"Action": "ec2:Describe*"
```

> **Note**
>
> 現在、すべての Amazon EC2 Describe* API アクションがリソースレベルのアクセス許可をサポートしているわけではありません。Amazon EC2のリソースレベルの許可の詳細については、Amazon EC2 の IAM ポリシーを参照してください。

Amazon EC2 API アクションをすべて指定するには、* ワイルドカードを以下のように使用します。

```
"Action": "ec2:*"
```

Amazon EC2 アクションのリストを確認するには、サービス認証リファレンスのAmazon EC2 で定義されるアクションを参照してください。

Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可

リソースレベルのアクセス許可とは、ユーザーがアクションを実行できるリソースを指定できる機能を意味します。Amazon EC2 は、リソースレベルのアクセス許可を部分的にサポートします。これは、特定の Amazon EC2 アクションでは、満たす必要がある条件、またはユーザーが使用できる特定のリソースに基づいて、ユーザーがそれらのアクションをいつ使用できるかを制御できることを意味します。例えば、特定の AMI のみを使用して、特定のタイプのインスタンスだけを起動するアクセス許可をユーザーに付与できます。

IAM ポリシーステートメントでリソースを指定するには、Amazon リソースネーム (ARN) を使用します。ARN 値の指定については、Amazon EC2 用の Amazon リソースネーム (ARN)を参照してください。API アクションが個々の ARN をサポートしていない場合は、ワイルドカード (*) を使用して、アクションによってすべてのリソースが影響を受ける可能性があることを指定する必要があります。

リソースレベルのアクセス許可をサポートする Amazon EC2 API アクション、およびポリシーで使用できる ARN と条件キーがわかる表を見るには、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

Amazon EC2 API アクションに対して使用する IAM ポリシーで、タグベースのリソースレベルアクセス許可を適用できます。これにより、ユーザーがどのリソースを作成、変更、または使用できるかを制御しやすくなります。詳細については、リソース作成時にタグ付けするアクセス許可の付与を参照してください。

Amazon EC2 用の Amazon リソースネーム (ARN)

各 IAM ポリシーステートメントは、ARN を使用して指定したリソースに適用されます。

ARN には以下の一般的な構文があります。

```
arn:aws:[service]:[region]:[account-id]:resourceType/resourcePath
```

service

サービス (例: ec2)。

region

リソースのリージョン (例: us-east-1)。

account-id

ハイフンなしの AWS アカウント ID (例: 123456789012)。

resourceType

リソースの種類 (例: instance)。

resourcePath

リソースを識別するパス。パスにワイルドカードの * が使用できます。

例えば、以下のように ARN を使用して、ステートメント内で特定のインスタンス (i-1234567890abcdef0) を指定することができます。

```
"Resource": "arn:aws:ec2:us-east-1:123456789012:instance/i-1234567890abcdef0"
```

以下のように * ワイルドカードを使用して、特定のアカウントに属するすべてのインスタンスを指定できます。

```
"Resource": "arn:aws:ec2:us-east-1:123456789012:instance/*"
```

また、以下のように * ワイルドカードを使用して、特定のアカウントに属するすべての Amazon EC2 リソースを指定することもできます。

```
"Resource": "arn:aws:ec2:us-east-1:123456789012:*"
```

すべてのリソースを指定する場合、または特定の API アクションが ARN をサポートしていない場合は、以下のように、Resource エレメント内で * ワイルドカードを使用します。

```
"Resource": "*"
```

Amazon EC2 API アクションの多くが複数のリソースと関連します。例えば、AttachVolume では Amazon EBS ボリュームをインスタンスにアタッチするため、ユーザーはボリュームおよびインスタンスを使用する許可が必要です。1 つのステートメントで複数のリソースを指定するには、次のように ARN をカンマで区切ります。

```
"Resource": ["arn1", "arn2"]
```

Amazon EC2 リソースの ARN のリストについては、Amazon EC2 で定義されるリソースタイプを参照してください。

Amazon EC2 の条件キー

ポリシーステートメントでは、オプションで有効になるタイミングを制御する条件を指定できます。各条件には 1 つ以上のキーと値のペアが含まれます。条件キーは大文字小文字を区別しません。AWS グローバル条件キーに加え、追加のサービス固有の条件キーも定義されています。

Amazon EC2 のサービス固有の条件キーのリストについては、Amazon EC2 の条件キーを参照してください。Amazon EC2 では、AWS グローバル条件キーも実装されています。詳細については、IAM ユーザーガイドのすべてのリクエストで利用可能な情報を参照してください。

IAM ポリシーで条件キーを使用するには、Condition ステートメントを使用します。例えば、次のポリシーは、セキュリティグループのインバウンドルールとアウトバウンドルールを追加および削除するアクセス許可をユーザーに付与します。ec2:Vpc 条件キーを使用して、これらのアクションを実行できる対象は、特定の VPC 内のセキュリティグループに限ることを指定します。

```
{
"Version": "2012-10-17",
  "Statement":[{
    "Effect":"Allow",
    "Action": [
```

```
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress"],
      "Resource": "arn:aws:ec2:region:account:security-group/*",
       "Condition": {
         "StringEquals": {
           "ec2:Vpc": "arn:aws:ec2:region:account:vpc/vpc-11223344556677889"
         }
       }
     }
  ]
}
```

複数の条件、または単一の条件に複数のキーを指定する場合、論理 AND 演算を使用してそれらを評価します。1 つのキーに複数の値を使用して単一の条件を指定する場合、論理 OR 演算を使用して条件を評価します。アクセス許可が付与されるには、すべての条件を満たしている必要があります。

条件を指定する際にプレースホルダーも使用できます。詳細については、IAM ユーザーガイドのIAM ポリシーエレメント: 変数およびタグを参照してください。

> **Important**
>
> 多くの条件キーはリソースに固有のものであり、一部の API アクションでは複数のリソースを使用します。条件キーを使用してポリシーを作成する場合は、ポリシーステートメントの `Resource` 要素で、条件キーが適用されるリソースを指定します。指定しない場合、そのポリシーはユーザーに対してすべてのアクションの実行を禁止します。これは、条件キーが適用されないリソースに対して条件チェックが失敗するためです。リソースを指定しない場合や、ポリシーの `Action` 要素に複数の API アクションを含めている場合は、`...IfExists` 条件タイプを使用して、条件キーが適用されないリソースに対して無視されるようにする必要があります。詳細については、IAM ユーザーガイドの..IfExists 条件を参照してください。

すべての Amazon EC2 アクションは、`aws:RequestedRegion` および `ec2:Region` 条件キーをサポートします。詳細については、「例: 特定のリージョンへのアクセスの制限」を参照してください。

`ec2:SourceInstanceARN` 条件キー

`ec2:SourceInstanceARN` 条件キーは、リクエストの生成元インスタンスの ARN を指定する条件に使用できます。これは、AWS グローバル条件キーであり、サービス固有ではありません。ポリ

シーの例については、EC2 インスタンス: Amazon EC2 インスタンスへのボリューのアタッチまたはデタッチと例: 特定のインスタンスが他の AWS サービスでリソースを表示できるようにするを参照してください。ec2:SourceInstanceARN キーは、ステートメントの Resource 要素に ARN を入力する変数として使用することはできません。

Amazon EC2 のポリシーステートメントの例については、AWS CLI または AWS SDK で使用するサンプルポリシーを参照してください。

ec2:Attribute 条件キー

ec2:Attribute 条件キーは、リソースの属性によってアクセスをフィルタリングする条件に使用できます。条件キーは、プリミティブデータ型のプロパティ (文字列や整数など)、または [値] のプロパティのみを持つ複雑な AttributeValue オブジェクト (ModifyImageAttribute API アクションの Description または ImdsSupport オブジェクトなど) に使用できます。

> **Important**
>
> 条件キーは、ModifyImageAttribute API アクションの LaunchPermission オブジェクトなど、複数のプロパティを持つ複雑なオブジェクトには使用できません。

例えば、次のポリシーでは、ModifyImageAttributeAPI アクションの複雑な Description オブジェクトによるアクセスをフィルタリングするために ec2:Attribute/Description 条件キーを使用します。条件キーは、イメージの説明を Production または Development のいずれかに変更するリクエストのみを許可します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:ModifyImageAttribute",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-*",
      "Condition": {
        "StringEquals": {
          "ec2:Attribute/Description": [
            "Production",
            "Development"
          ]
        }
      }
```

```
    }
  ]
}
```

次のポリシー例では、ModifyImageAttribute API アクションのプリミティブな Attribute プロパティによるアクセスをフィルタリングするために ec2:Attribute 条件キーを使用します。この条件キーは、イメージの説明を変更しようとするすべてのリクエストを拒否します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Deny",
      "Action": "ec2:ModifyImageAttribute",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-*",
      "Condition": {
        "StringEquals": {
          "ec2:Attribute": "Description"
        }
      }
    }
  ]
}
```

ec2:ResourceID 条件キー

指定された API アクションで次の ec2:*Resource*ID 条件キーを使用する場合、条件キーの値は、API アクションによって作成される結果のリソースを指定するために使用されます。ec2:*Resource*ID 条件キーを使用して、API リクエストで指定されたソース リソースを指定することはできません。指定された API で次の ec2:*Resource*ID 条件キーのいずれかを使用する場合は、常にワイルドカード (*) を指定する必要があります。別の値を指定した場合、条件はランタイム中に常に * に解決されます。例えば、CopyImage API で ec2:ImageId 条件キーを使用するには、次のように条件キーを指定する必要があります。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CopyImage",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-*",
```

```
      "Condition": {
        "StringEquals": {
          ec2:ImageID": "*"
        }
      }
    }
  ]
}
```

条件キー	API アクション			
ec2:DhcpOptionsID	• CreateDhcpOptions			
ec2:ImageID	• CopyImage • CreateImage • ImportImage • RegisterImage			
ec2:InstanceID	• RunInstances • ImportInstance			
ec2:InternetGatewayID	• CreateInternetGateway			
ec2:NetworkAclID	•			

条件キー	API アクション			
	CreateNetworkAcl			
ec2:NetworkInterfaceID	• CreateNetworkInterface			
ec2:PlacementGroupName	• CreatePlacementGroup			
ec2:RouteTableID	• CreateRouteTable			
ec2:SecurityGroupID	• CreateSecurityGroup			
ec2:SnapshotID	• CopySnapshot • CreateSnapshot • CreateSnapshots • ImportSnapshots			
ec2:SubnetID	• CreateSubnet			

条件キー	API アクション			
ec2:VolumeID	• CreateVolume • ImportVolume			
ec2:VpcID	• CreateVpc			
ec2:VpcPeeringConnectionID	• CreateVpcPeeringConnection			

これらの API アクションでは ec2:*Resource*ID 条件キーを使用しないことをお勧めします。代わりに、特定のリソース ID に基づいてアクセスをフィルタリングする必要がある場合は、次のように Resource ポリシー要素を使用してフィルタリングすることをお勧めします。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:CopyImage",
      "Resource": "arn:aws:ec2:us-east-1::image/ami-01234567890abcdef"
    }
  ]
}
```

ユーザーが必要なアクセス許可を持っているかどうかの確認

IAM ポリシーを作成したら、ポリシーを本稼働環境に置く前に、そのポリシーがユーザーに特定の API アクションおよび必要なリソースを使用するアクセス許可を付与しているかどうかを確認することをお勧めします。

まずテスト目的のユーザーを作成し、作成した IAM ポリシーをテストユーザーにアタッチします。次に、テストユーザーとしてリクエストを作成します。

テスト中の Amazon EC2 アクションがリソースを作成または変更する場合、DryRun パラメータ を使用してリクエストを作成する (または AWS CLI オプションで --dry-run コマンドを実行する) 必要があります。この場合、発信者は認証チェックを行いますが、操作は完了しません。例えば、実際に終了させることなく、ユーザーが特定のインスタンスを終了できるかどうかを確認できます。テストユーザーに必要なアクセス許可がある場合、リクエストで DryRunOperation が返されます。必要なアクセス許可がない場合は UnauthorizedOperation が返されます。

ポリシーが想定したアクセス許可をユーザーに付与していない場合、または過度に許可されている場合、必要に応じてポリシーを調整し、必要な結果を得るまで再テストできます。

> **Important**
>
> ポリシーの変更が反映され、有効になるには数分間かかります。したがって、ポリシーの更新をテストするには 5 分かかると見ておいてください。

認証チェックが失敗した場合、リクエストでは診断情報でエンコードされたメッセージが返されます。DecodeAuthorizationMessage アクションを使用してメッセージをデコードできます。詳細については、AWS Security Token Service API リファレンス のDecodeAuthorizationMessage、およびAWS CLI コマンドリファレンスのdecode-authorization-messageを参照してください。

リソース作成時にタグ付けするアクセス許可の付与

一部のリソース作成 Amazon EC2 API アクションでは、リソースの作成時にタグを指定できます。リソースタグを使用して、属性ベースの制御 (ABAC) を実装できます。詳細については、リソースのタグ付けおよびリソースタグを使用した EC2 リソースへのアクセスの制御を参照してください。

ユーザーがリソースの作成時にタグを付けるには、リソースを作成するアクション (ec2:RunInstances や ec2:CreateVolume など) を使用するためのアクセス許可が必要です。タグがリソース作成アクションで指定されている場合、Amazon は ec2:CreateTags アクションで追加の認可を実行してユーザーがタグを作成するアクセス許可を持っているかどうかを確認します。そのため、ユーザーには、ec2:CreateTags アクションを使用する明示的なアクセス権限が必要です。

ec2:CreateTags アクションの IAM ポリシー定義で、Condition 要素と ec2:CreateAction 条件キーを使用して、リソースを作成するアクションにタグ付けのアクセス許可を付与します。

次のポリシー例では、インスタンスを起動し、起動時にインスタンスとボリュームにタグを適用することをユーザーに許可します。ユーザーには、既存のリソースへのタグ付けが許可されません (ec2:CreateTags アクションを直接呼び出すことはできません)。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:region:account:*/*",
      "Condition": {
         "StringEquals": {
             "ec2:CreateAction" : "RunInstances"
          }
       }
    }
  ]
}
```

同様に、次のポリシーでは、ユーザーがボリュームを作成し、ボリューム作成時にボリュームにタグを適用することができます。ユーザーには、既存のリソースへのタグ付けが許可されません (ec2:CreateTags アクションを直接呼び出すことはできません)。

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateVolume"
      ],
      "Resource": "*"
    },
    {
```

```
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:region:account:*/*",
      "Condition": {
         "StringEquals": {
             "ec2:CreateAction" : "CreateVolume"
          }
       }
    }
  ]
}
```

ec2:CreateTags アクションは、タグがリソース作成アクション時に適用された場合のみ評価されます。したがって、リクエストでタグが指定されていない場合、リソースを作成するアクセス許可を持っているユーザー (タグ付け条件がないと仮定) には、ec2:CreateTags アクションを実行するアクセス許可は必要ありません。ただし、ユーザーがタグを使用してリソースを作成しようとした場合、ユーザーが ec2:CreateTags アクションを使用するアクセス権限を持っていない場合はリクエストに失敗します。

ec2:CreateTags アクションは、タグが起動テンプレートに指定されている場合にも評価されます。ポリシーの例については、起動テンプレートのタグを参照してください。

特定のタグへのアクセスの制御

IAM ポリシーの Condition 要素で追加の条件を使用して、リソースに適用できるタグキーとタグ値を制御できます。

次の条件キーは、前のセクションの例で使用できます。

- aws:RequestTag: 特定のタグキーまたはタグキーと値がリクエストに存在している必要があることを指定する場合に使用します。リクエストでは他のタグも指定できます。
  - StringEquals 条件演算子とともに使用して、特定のタグキーと値の組み合わせを適用します。例えば、タグ cost-center=cc123: を適用します。

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - StringLike 条件演算子とともに使用して、リクエストで特定のタグキーを適用します。例えば、タグキー purpose を適用します。

```
"StringLike": { "aws:RequestTag/purpose": "*" }
```

- aws:TagKeys: リクエストで使用されるタグキーを適用する場合に使用します。
  - リクエストにタグが指定されている場合は、ForAllValues 修飾子を使用して特定のタグキーのみを適用します (リクエストにタグが指定されている場合、特定のタグキーのみが許可されます。他のタグは許可されません)。例えば、タグキー environment または cost-center が適用されます:

    ```
    "ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
    ```

  - ForAnyValue 修飾子とともに使用して、指定されたタグキーの少なくとも 1 つがリクエストに存在することを要求します。例えば、タグキー environment または webserver のうち少なくとも 1 つがリクエストに存在している必要があります。

    ```
    "ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
    ```

これらの条件キータグ付けをサポートするリソース作成アクションと、ec2:CreateTags および ec2:DeleteTags アクションに適用できます。Amazon EC2 API アクションがタグ付けをサポートしているかどうかについては、Amazon EC2 のアクション、リソース、および条件キーを参照してください

リソースの作成時にタグを指定するようにユーザーに強制するには、リソース作成アクションで aws:RequestTag 修飾子とともに aws:TagKeys 条件キーまたは ForAnyValue 条件キーを使用する必要があります。ユーザーがリソース作成アクションのタグを指定しない場合、ec2:CreateTags アクションは評価されません。

条件においては、条件キーでは大文字と小文字が区別されず、条件値では大文字と小文字が区別されます。したがって、タグキーの大文字と小文字を区別するには、条件の値としてタグキーが指定される aws:TagKeys 条件キーを使用します。

IAM ポリシーの例は、AWS CLI または AWS SDK で使用するサンプルポリシーを参照してください。複数値条件の詳細については、IAM ユーザーガイドの複数のキーバリューをテストする条件を作成するを参照してください。

リソースタグを使用した EC2 リソースへのアクセスの制御

EC2 リソースを使用するための許可をユーザーに付与する IAM ポリシーを作成する場合、ポリシーの Condition 要素にタグ情報を含めることで、タグに基づいてアクセスをコントロールできます。

これは、属性ベースのアクセス制御 (ABAC) と呼ばれます。ABAC を使用すると、ユーザーが変更、使用、または削除できるリソースをより適切に制御できます。詳細については、AWS の ABAC とは を参照してください。

例えば、インスタンスを終了することをユーザーに許可するが、インスタンスに environment=production タグが付いている場合はアクションを拒否するポリシーを作成できます。これを行うには、aws:ResourceTag 条件キーを使用し、リソースにアタッチされているタグに基づいてリソースへのアクセスを許可または拒否します。

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Amazon EC2 API アクションが aws:ResourceTag 条件キーを使用したアクセスの制御をサポートしているかどうかについては、Amazon EC2 のアクション、リソース、および条件キーを参照してください。Describe アクションはリソースレベルのアクセス権限をサポートしないため、条件のない別のステートメントでそれらのアクセス権限を指定する必要があることに注意してください。

IAM ポリシーの例は、AWS CLI または AWS SDK で使用するサンプルポリシーを参照してください。

タグに基づいてリソースへのユーザーのアクセスを許可または拒否する場合は、ユーザーが同じリソースに対してそれらのタグを追加または削除することを明示的に拒否することを検討する必要があります。そうしないと、ユーザーはそのリソースのタグを変更することで、制限を回避してリソースにアクセスできてしまいます。

AWS CLI または AWS SDK で使用するサンプルポリシー

IAM ポリシーを使用して Amazon EC2 に必要な許可をユーザーに付与する必要があります。以下の例では、Amazon EC2 に対してユーザーが所有する許可をコントロールするために使用できるポリシーステートメントを示しています。これらのポリシーは、AWS CLI または AWS SDK で行われたリクエスト向けに設計されています。詳細については、「IAM ユーザーガイド」の「IAM ポリシーの作成」を参照してください。Amazon EC2 コンソールで機能するポリシーの例については、Amazon EC2 コンソールで機能するサンプル ポリシーを参照してください。Amazon VPC に固有の IAM ポリシーの例については、Amazon VPC の Identity and Access Management を参照してください。

次の例では、###############をユーザー自身の情報で置き換えます。

例

- 例: 読み取り専用アクセス

- 例: 特定のリージョンへのアクセスの制限
- インスタンスの使用
- インスタンスの起動 (RunInstances)
- スポットインスタンス の操作
- 例: リザーブドインスタンス の操作
- 例: リソースのタグ付け
- 例: IAM ロールの使用
- 例: ルートテーブルの使用
- 例: 特定のインスタンスが他の AWS サービスでリソースを表示できるようにする
- 例: 起動テンプレートの使用
- インスタンスメタデータの使用
- Amazon EBS ボリュームとスナップショットの使用

例: 読み取り専用アクセス

次のポリシーでは、名前が Describe で始まるすべての Amazon EC2 API アクションを使用できるアクセス許可をユーザーに与えます。Resource エレメントにワイルドカードを使用します。これは、ユーザーが API アクションですべてのリソースを指定できることを示します。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

デフォルトで API アクションを使用するアクセス許可が拒否されているため、ユーザーには (別のステートメントでアクセス許可が与えられない限り) そのリソースに対してアクションを実行するアクセス許可がありません。

```
{
   "Version": "2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:Describe*",
      "Resource": "*"
    }
   ]
```

```
}
```

例: 特定のリージョンへのアクセスの制限

次のポリシーでは、リージョンが 欧州 (フランクフルト) でない限り、すべての Amazon EC2 API アクションを使用するアクセス許可をユーザーに拒否します。これにはグローバル条件キー aws:RequestedRegion が使用され、このキーはすべての Amazon EC2 API アクションでサポートされています。

```
{
  "Version":"2012-10-17",
  "Statement": [
       {
      "Effect": "Deny",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "aws:RequestedRegion": "eu-central-1"
        }
      }
    }
  ]
}
```

または、条件キー ec2:Region を使用することもできます。これは、Amazon EC2 に固有のもので、すべての Amazon EC2 API アクションでサポートされています。

```
{
  "Version":"2012-10-17",
  "Statement": [
       {
      "Effect": "Deny",
      "Action": "ec2:*",
      "Resource": "*",
      "Condition": {
        "StringNotEquals": {
          "ec2:Region": "eu-central-1"
        }
      }
    }
  ]
```

```
}
```

インスタンスの使用

例

- 例: すべてのインスタンスを記述、起動、停止、開始、および終了する
- 例: すべてのインスタンスを記述し、特定のインスタンスのみを停止、開始、および終了する

例: すべてのインスタンスを記述、起動、停止、開始、および終了する

次のポリシーでは、Action エレメントで指定された API アクションを使用するアクセス許可をユーザーに与えます。Resource エレメントでは * ワイルドカードを使用して、ユーザーが API アクションですべてのリソースを指定できることを示します。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

ユーザーはデフォルトで API アクションを使用するアクセス許可を拒否されているため、ユーザーには (別のステートメントでユーザーにそのアクセス許可を与えない限り) その他の API アクションを使用するアクセス許可がありません。

```
{
   "Version": "2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeInstances",
        "ec2:DescribeImages",
        "ec2:DescribeKeyPairs",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeAvailabilityZones",
        "ec2:RunInstances",
        "ec2:TerminateInstances",
        "ec2:StopInstances",
        "ec2:StartInstances"
      ],
      "Resource": "*"
    }
   ]
```

```
}
```

例: すべてのインスタンスを記述し、特定のインスタンスのみを停止、開始、および終了する

次のポリシーでは、すべてのインスタンスを表示し、i-1234567890abcdef0 と i-0598c7d356eba48d7 インスタンスのみを開始および停止し、米国東部 (バージニア北部) リージョン (us-east-1) 内でリソースタグ 「purpose=test」 の付いたインスタンスのみを終了する許可をユーザーに与えます。

最初のステートメントでは、Resource エレメントに * ワイルドカードを使用して、ユーザーがそのアクションにすべてのリソースを指定できることを示しています。この場合、すべてのインスタンスをリストできます。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です (この場合は、ec2:DescribeInstances)。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

2 番目のステートメントでは、StopInstances および StartInstances アクションに対してリソースレベルのアクセス許可を使用しています。Resource エレメント内で、ARN によって特定のインスタンスが指定されています。

3 番目のステートメントでは、ユーザーは指定された us-east-1 アカウントに属する 米国東部 (バージニア北部) リージョン (AWS) 内のすべてのインスタンスを終了できますが、インスタンスにタグ "purpose=test" が付けられている場合に限ります。Condition エレメントは、ポリシーステートメントの発効条件を指定します。

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
   "Effect": "Allow",
      "Action": "ec2:DescribeInstances",
      "Resource": "*"
   },
   {
      "Effect": "Allow",
      "Action": [
        "ec2:StopInstances",
        "ec2:StartInstances"
      ],
      "Resource": [
        "arn:aws:ec2:us-east-1:account-id:instance/i-1234567890abcdef0",
```

```
        "arn:aws:ec2:us-east-1:account-id:instance/i-0598c7d356eba48d7"
      ]
    },
    {
      "Effect": "Allow",
      "Action": "ec2:TerminateInstances",
      "Resource": "arn:aws:ec2:us-east-1:account-id:instance/*",
      "Condition": {
         "StringEquals": {
            "aws:ResourceTag/purpose": "test"
         }
      }
   }

   ]
}
```

インスタンスの起動 (RunInstances)

RunInstances API アクションは、1 つ以上の オンデマンドインスタンス または 1 つ以上の スポットインスタンス を起動します。RunInstances は AMI を必要とし、インスタンスを作成します。ユーザーは、リクエストでキーペアとセキュリティグループを指定できます。VPC 内に起動するにはサブネットが必要であり、起動されるとネットワークインターフェイスが作成されます。Amazon EBS-backed AMI から起動すると、ボリュームが作成されます。そのため、ユーザーにはこれらの Amazon EC2 リソースを使用するアクセス許可が必要です。ユーザーが RunInstances に対してオプションのパラメータを指定する必要がある、またはユーザーからパラメータの特定の値を制限するポリシーステートメントを作成できます。

インスタンスの起動に必要なリソースレベルのアクセス許可の詳細については、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

デフォルトでは、作成したインスタンスを記述、開始、停止、または終了するアクセス許可はユーザーに付与されていません。作成したインスタンスを管理するアクセス許可をユーザーに付与する 1 つの方法としては、インスタンスごとに特定のタグを作成し、そのタグでインスタンスを管理できるようにステートメントを作成します。詳細については、インスタンスの使用を参照してください。

リソース

- AMI
- インスタンスタイプ
- サブネット

- EBS ボリューム
- タグ
- 起動テンプレートのタグ
- Elastic GPU
- 起動テンプレート

AMI

次のポリシーでは、指定された AMI (ami-9e1670f7 および ami-45cf5c3c) のみを使用してインスタンスを起動できます。(別のステートメントでユーザーに起動するアクセス許可が付与されない限り) ユーザーはその他の AMI を使用してインスタンスを起動することはできません。

```
{
   "Version": "2012-10-17",
   "Statement": [
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:region::image/ami-9e1670f7",
        "arn:aws:ec2:region::image/ami-45cf5c3c",
        "arn:aws:ec2:region:account-id:instance/*",
        "arn:aws:ec2:region:account-id:volume/*",
        "arn:aws:ec2:region:account-id:key-pair/*",
        "arn:aws:ec2:region:account-id:security-group/*",
        "arn:aws:ec2:region:account-id:subnet/*",
        "arn:aws:ec2:region:account-id:network-interface/*"
      ]
    }
   ]
}
```

または、次のポリシーを使用すると、ユーザーは Amazon、または特定の信頼できる検証済みのパートナーが所有するすべての AMI からインスタンスを起動できます。最初のステートメントの Condition エレメントは、ec2:Owner が amazon であるかどうかをテストします。(別のステートメントでユーザーに起動するアクセス許可が付与されない限り) ユーザーはその他の AMI を使用してインスタンスを起動することはできません。

```
{
```

```
   "Version": "2012-10-17",
   "Statement": [
         {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:region::image/ami-*"
      ],
      "Condition": {
         "StringEquals": {
            "ec2:Owner": "amazon"
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:region:account-id:instance/*",
         "arn:aws:ec2:region:account-id:subnet/*",
         "arn:aws:ec2:region:account-id:volume/*",
         "arn:aws:ec2:region:account-id:network-interface/*",
         "arn:aws:ec2:region:account-id:key-pair/*",
         "arn:aws:ec2:region:account-id:security-group/*"
         ]
      }
   ]
}
```

インスタンスタイプ

次のポリシーにより、ユーザーは t2.micro または t2.small インスタンスタイプのみを使用してインスタンスを起動できます。これにより、コストを管理することができます。最初のステートメントの Condition エレメントは ec2:InstanceType が t2.micro または t2.small のどちらであるかをテストするため、ユーザーは大きなインスタンスを起動することはできません。

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
```

```
            "arn:aws:ec2:region:account-id:instance/*"
         ],
         "Condition": {
            "StringEquals": {
               "ec2:InstanceType": ["t2.micro", "t2.small"]
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": "ec2:RunInstances",
         "Resource": [
            "arn:aws:ec2:region::image/ami-*",
            "arn:aws:ec2:region:account-id:subnet/*",
            "arn:aws:ec2:region:account-id:network-interface/*",
            "arn:aws:ec2:region:account-id:volume/*",
            "arn:aws:ec2:region:account-id:key-pair/*",
            "arn:aws:ec2:region:account-id:security-group/*"
            ]
         }
      ]
}
```

また、ユーザーが t2.micro と t2.small のインスタンスタイプ以外のすべてのインスタンス起動へのアクセスを拒否するポリシーを作成することもできます。

```
{
   "Version": "2012-10-17",
   "Statement": [
        {
         "Effect": "Deny",
         "Action": "ec2:RunInstances",
         "Resource": [
            "arn:aws:ec2:region:account-id:instance/*"
         ],
         "Condition": {
            "StringNotEquals": {
               "ec2:InstanceType": ["t2.micro", "t2.small"]
            }
         }
      },
      {
         "Effect": "Allow",
```

```
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:region::image/ami-*",
         "arn:aws:ec2:region:account-id:network-interface/*",
         "arn:aws:ec2:region:account-id:instance/*",
         "arn:aws:ec2:region:account-id:subnet/*",
         "arn:aws:ec2:region:account-id:volume/*",
         "arn:aws:ec2:region:account-id:key-pair/*",
         "arn:aws:ec2:region:account-id:security-group/*"
         ]
      }
   ]
}
```

サブネット

次のポリシーにより、ユーザーは指定したサブネット subnet-*12345678* のみを使用してインスタンスを起動できます。グループは、インスタンスを他のサブネットに起動することはできません (他のステートメントがそのような許可をユーザーに与えている場合はその限りではありません)。

```
{
   "Version": "2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:region:account-id:subnet/subnet-12345678",
        "arn:aws:ec2:region:account-id:network-interface/*",
        "arn:aws:ec2:region:account-id:instance/*",
        "arn:aws:ec2:region:account-id:volume/*",
        "arn:aws:ec2:region::image/ami-*",
        "arn:aws:ec2:region:account-id:key-pair/*",
        "arn:aws:ec2:region:account-id:security-group/*"
      ]
    }
   ]
}
```

また、ユーザーがその他のサブネットにインスタンスを起動するアクセス許可を拒否するポリシーを作成することもできます。ステートメントでは、サブネット subnet-*12345678* が指定されてい

る場合以外は、ネットワークインターフェイスの作成を拒否することでこれを実行します。この拒否は、他のサブネットへのインスタンスの起動を許可する他のすべてのポリシーよりも優先されます。

```
{
   "Version": "2012-10-17",
   "Statement": [
         {
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:region:account-id:network-interface/*"
      ],
      "Condition": {
         "ArnNotEquals": {
            "ec2:Subnet": "arn:aws:ec2:region:account-id:subnet/subnet-12345678"
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:region::image/ami-*",
         "arn:aws:ec2:region:account-id:network-interface/*",
         "arn:aws:ec2:region:account-id:instance/*",
         "arn:aws:ec2:region:account-id:subnet/*",
         "arn:aws:ec2:region:account-id:volume/*",
         "arn:aws:ec2:region:account-id:key-pair/*",
         "arn:aws:ec2:region:account-id:security-group/*"
         ]
      }
   ]
}
```

EBS ボリューム

次のポリシーでは、インスタンスの EBS ボリュームが暗号化されている場合のみユーザーがインスタンスを起動できます。ユーザーは、ルートボリュームが暗号化されるように、暗号化されたスナップショットを使用して作成された AMI からインスタンスを起動する必要があります。ユーザーが起動時にインスタンスにアタッチする追加ボリュームも暗号化されている必要があります。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
                {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*:*:volume/*"
            ],
            "Condition": {
                "Bool": {
                    "ec2:Encrypted": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:instance/*",
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        }
    ]
}
```

タグ

インスタンスの作成時にタグを付ける

次のポリシーでは、ユーザーがインスタンスを起動し、作成時にインスタンスにタグ付けすることができます。タグを適用するリソース作成アクションには、ユーザーが CreateTags アクションを使用するアクセス権限を持っていることが必要です。2 番目のステートメントは、ec2:CreateAction 条件キーを使用し、ユーザーが RunInstances のコンテキストでのみ、インスタンスに対してのみタグを作成できるようにします。ユーザーは、既存のリソースにタグ付けすることができず、RunInstances リクエストを使用してボリュームにタグ付けすることもできません。

詳細については、リソース作成時にタグ付けするアクセス許可の付与を参照してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:us-east-1:account-id:instance/*",
      "Condition": {
         "StringEquals": {
             "ec2:CreateAction" : "RunInstances"
          }
       }
    }
  ]
}
```

インスタンスやボリュームの作成時に特定のタグを付ける

次のポリシーには、aws:RequestTag および RunInstances タグを使用して environment=production により作成されたすべてのインスタンスおよびボリュームへのタグ付けをユーザーに求める purpose=webserver 条件キーが含まれています。ユーザーがこれらのタグを渡さないか、タグをまったく指定しない場合、リクエストは失敗します。

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": [
         "arn:aws:ec2:region::image/*",
```

```
            "arn:aws:ec2:region:account-id:subnet/*",
            "arn:aws:ec2:region:account-id:network-interface/*",
            "arn:aws:ec2:region:account-id:security-group/*",
            "arn:aws:ec2:region:account-id:key-pair/*"
        ]
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:RunInstances"
        ],
        "Resource": [
            "arn:aws:ec2:region:account-id:volume/*",
            "arn:aws:ec2:region:account-id:instance/*"
        ],
        "Condition": {
            "StringEquals": {
                "aws:RequestTag/environment": "production" ,
                "aws:RequestTag/purpose": "webserver"
            }
        }
    },
    {
        "Effect": "Allow",
        "Action": [
            "ec2:CreateTags"
        ],
        "Resource": "arn:aws:ec2:region:account-id:*/*",
        "Condition": {
            "StringEquals": {
                "ec2:CreateAction" : "RunInstances"
            }
        }
    }
  ]
}
```

インスタンスやボリュームの作成時に特定のタグを少なくとも 1 つ付ける

次のポリシーは、ForAnyValue 条件で aws:TagKeys 修飾子を使用して、リクエストで少なくとも 1 つのタグが指定されている必要があり、キー environment または webserver が含まれている必要があることを示します。タグは、インスタンスとボリュームの両方に適用される必要があります。リクエストでは、任意のタグ値を指定できます。

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": [
         "arn:aws:ec2:region::image/*",
         "arn:aws:ec2:region:account-id:subnet/*",
         "arn:aws:ec2:region:account-id:network-interface/*",
         "arn:aws:ec2:region:account-id:security-group/*",
         "arn:aws:ec2:region:account-id:key-pair/*"
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
          "ec2:RunInstances"
      ],
      "Resource": [
          "arn:aws:ec2:region:account-id:volume/*",
          "arn:aws:ec2:region:account-id:instance/*"
      ],
      "Condition": {
          "ForAnyValue:StringEquals": {
              "aws:TagKeys": ["environment","webserver"]
          }
       }
    },
    {
      "Effect": "Allow",
      "Action": [
          "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:region:account-id:*/*",
      "Condition": {
          "StringEquals": {
              "ec2:CreateAction" : "RunInstances"
          }
       }
    }
```

```
  ]
}
```

インスタンスの作成時にタグを付ける場合は、特定のタグを使用してタグ付けする必要がある

次のポリシーでは、ユーザーはリクエストでタグを指定する必要はありませんが、指定する場合は purpose=test タグを指定する必要があります。他のタグは許可されません。ユーザーは、RunInstances リクエストでタグ付け可能なリソースにタグを適用できます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:CreateTags"
      ],
      "Resource": "arn:aws:ec2:region:account-id:*/*",
      "Condition": {
         "StringEquals": {
             "aws:RequestTag/purpose": "test",
             "ec2:CreateAction" : "RunInstances"
          },
          "ForAllValues:StringEquals": {
              "aws:TagKeys": "purpose"
          }
       }
    }
  ]
}
```

RunInstances の呼び出しで作成時のタグ付けをユーザーを禁止するには

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "VisualEditor0",
            "Effect": "Deny",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

spot-instances-request の特定のタグのみを許可します。ここで矛盾数 2 が意外な効果を発揮します。通常の状況では、タグを指定しないと非認証になります。spot-instances-request の場合、spot-instances-request タグがないと、このポリシーは評価されないため、タグなしの Spot on Run リクエストが成功します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
```

```
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
            ]
        },
        {
            "Sid": "VisualEditor0",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        }
    ]
}
```

起動テンプレートのタグ

次の例で、ユーザーはインスタンスを起動できますが、特定の起動テンプレートを使用する場合に限ります (lt-09477bcd97b0d310e)。ec2:IsLaunchTemplateResource 条件キーは、ユーザーが起動テンプレートで指定されたリソースを上書きしないようにします。ステートメントの 2 番目の部分では、ユーザーは作成時にインスタンスにタグ付けできます — ステートメントのこの部分は、起動テンプレートでタグがインスタンスに対して指定されている場合に必要になります。

```
{
  "Version": "2012-10-17",
  "Statement": [
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:region:account-id:launch-template/
lt-09477bcd97b0d310e"
          },
```

```
            "Bool": {
               "ec2:IsLaunchTemplateResource": "true"
            }
         }
      },
      {
        "Effect": "Allow",
        "Action": [
           "ec2:CreateTags"
        ],
        "Resource": "arn:aws:ec2:region:account-id:instance/*",
        "Condition": {
           "StringEquals": {
               "ec2:CreateAction" : "RunInstances"
            }
         }
      }
   ]
}
```

Elastic GPU

次のポリシーでは、ユーザーはインスタンスを起動させ、インスタンスにアタッチする Elastic GPU を指定できます。ユーザーは任意のリージョンでインスタンスを起動できますが、Elastic GPU をアタッチできるのはその us-east-2 リージョンでの起動中に限られます。

ec2:ElasticGpuType条件キーは、eg1.mediumeg1.largeインスタンスがまたはエラスティック GPU タイプのいずれかを使用することを保証します。

```
{
    "Version": "2012-10-17",
    "Statement": [
             {
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:*:account-id:elastic-gpu/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:Region": "us-east-2",
```

```
                    "ec2:ElasticGpuType": [
                        "eg1.medium",
                        "eg1.large"
                    ]
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:*::image/ami-*",
                "arn:aws:ec2:*:account-id:network-interface/*",
                "arn:aws:ec2:*:account-id:instance/*",
                "arn:aws:ec2:*:account-id:subnet/*",
                "arn:aws:ec2:*:account-id:volume/*",
                "arn:aws:ec2:*:account-id:key-pair/*",
                "arn:aws:ec2:*:account-id:security-group/*"
            ]
        }
    ]
}
```

起動テンプレート

次の例で、ユーザーはインスタンスを起動できますが、特定の起動テンプレートを使用する場合に限ります (lt-09477bcd97b0d310e)。ユーザーは、RunInstances アクションでパラメータを指定することで、起動テンプレートのパラメータを上書きできます。

```
{
  "Version": "2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:region:account-id:launch-template/
lt-09477bcd97b0d310e"
          }
       }
    }
```

```
  ]
}
```

この例で、ユーザーは、起動テンプレートを使用する場合に限りインスタンスを起動できます。ポリシーでは ec2:IsLaunchTemplateResource 条件キーを使用して、ユーザーが起動テンプレート内の既存の ARN を上書きできないようにします。

```
{
  "Version": "2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:region:account-id:launch-template/*"
          },
          "Bool": {
             "ec2:IsLaunchTemplateResource": "true"
          }
       }
    }
  ]
}
```

次のサンプルポリシーによりユーザーはインスタンスを起動できますが、起動テンプレートを使用する場合に限ります。ユーザーは、リクエストでサブネットおよびネットワークインターフェイスのパラメータを上書きすることはできません。これらのパラメータは、起動テンプレートでのみ指定できます。ステートメントの最初の部分は、NotResource 要素を使用して、サブネットやネットワークインターフェイスを除くその他のすべてのリソースを許可します。ステートメントの 2 番目の部分は、サブネットおよびネットワークインターフェイスのリソースを許可しますが、これは起動テンプレートから取得された場合に限ります。

```
{
  "Version": "2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "NotResource": ["arn:aws:ec2:region:account-id:subnet/*",
```

```
                    "arn:aws:ec2:region:account-id:network-interface/*" ],
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:region:account-id:launch-template/*"
          }
       }
    },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": ["arn:aws:ec2:region:account-id:subnet/*",
                  "arn:aws:ec2:region:account-id:network-interface/*" ],
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:region:account-id:launch-template/*"
          },
          "Bool": {
             "ec2:IsLaunchTemplateResource": "true"
          }
       }
    }
  ]
}
```

次の例では、起動テンプレートを使用していて、また起動テンプレートにタグがある場合に限り、ユーザーはインスタンスを起動できるようになります Purpose=Webservers。ユーザーは、RunInstances アクションで起動テンプレートパラメータを上書きすることはできません。

```
{
  "Version": "2012-10-17",
  "Statement": [
        {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "NotResource": "arn:aws:ec2:region:account-id:launch-template/*",
      "Condition": {
         "ArnLike": {
             "ec2:LaunchTemplate": "arn:aws:ec2:region:account-id:launch-template/*"
          },
         "Bool": {
             "ec2:IsLaunchTemplateResource": "true"
          }
       }
```

```
    },
    {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": "arn:aws:ec2:region:account-id:launch-template/*",
      "Condition": {
       "StringEquals": {
           "aws:ResourceTag/Purpose": "Webservers"
        }
       }
     }
  ]
}
```

スポットインスタンス の操作

RunInstances アクションを使用してスポットインスタンスリクエストを作成し、作成時にスポットインスタンスリクエストにタグ付けできます。RunInstances に指定するリソースは `spot-instances-request` です。

`spot-instances-request` リソースは、IAM ポリシーで次のように評価されます。

- スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 は RunInstances ステートメント内の `spot-instances-request` リソースを評価しません。
- スポットインスタンスリクエストの作成時にタグを付けると、 RunInstances ステートメント内の `spot-instances-request` リソースが、Amazon EC2 により評価されます。

したがって、`spot-instances-request` リソースの場合、次のルールが IAM ポリシーに適用されます。

- RunInstances を使用してスポットインスタンスリクエストを作成し、その際リクエストにタグを付けない場合は、`spot-instances-request` リソースを明示的に許可しなくても、その呼び出しは成功します。
- RunInstances を使用してスポットインスタンスリクエストを作成する際に、そのリクエストにタグを付ける場合には、RunInstances の許可ステートメントに `spot-instances-request` リソースを含める必要があります。これがない場合は呼び出しが失敗します。
- RunInstances を使用してスポットインスタンスリクエストを作成し、作成時にタグを付ける場合は、CreateTags 許可ステートメントに `spot-instances-request` リソースまたは * ワイルドカードを指定する必要があります。指定しないと、呼び出しは失敗します。

スポットインスタンス は、RunInstances または RequestSpotInstances を使用してリクエストできます。次の例の IAM ポリシーは、RunInstances を使用して スポットインスタンス をリクエストする場合にのみ適用されます。

例: RunInstances を使用して スポットインスタンス をリクエストする

次のポリシーでは、RunInstances アクションを使用して スポットインスタンス をリクエストすることをユーザーに許可します。spot-instances-request リソースは、RunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。

> **Note**
>
> RunInstances を使用してスポットインスタンスリクエストを作成し、作成時にタグを付けない場合は、spot-instances-request リストから Resource を省略できます。これは、スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 は RunInstances ステートメント内の spot-instances-request リソースを評価しないためです。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        }
    ]
}
```

Warning

サポート対象外 – 例: RunInstances を使用して スポットインスタンス をリクエストするためのアクセス許可をユーザーに拒否する
次のポリシーは、spot-instances-requestリソースではサポートされません。
次のポリシーでは、ユーザーに オンデマンドインスタンス を起動するためのアクセス許可を付与しますが、スポットインスタンス をリクエストするためのアクセス許可を拒否します。spot-instances-request リソースは、RunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。2 番目のステートメントでは、spot-instances-request リソースに対する RunInstances アクションを拒否します。ただし、スポットインスタンスリクエストの作成時にタグを付けない場合、Amazon EC2 が RunInstances ステートメントの spot-instances-request リソースを評価しないため、この条件はサポートされません。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*"
            ]
        },
        {
            "Sid": "DenySpotInstancesRequests - NOT SUPPORTED - DO NOT USE!",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
        }
    ]
```

```
}
```

例: スポットインスタンスリクエストの作成時にタグを付ける

次のポリシーでは、インスタンスの起動時に作成されるすべてのリソースにタグを付けることをユーザーに許可します。最初のステートメントでは、一覧表示されたリソースの作成を RunInstances に許可します。spot-instances-request リソースは、RunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。2 番目のステートメントでは、* ワイルドカードを指定し、インスタンスの起動時に作成されるすべてのリソースのタグ付けを許可します。

> **Note**
>
> スポットインスタンスリクエストの作成時にタグを付けると、 RunInstances ステートメント内の spot-instances-request リソースが、Amazon EC2 により評価されます。したがって、RunInstances アクションで spot-instances-request リソースを明示的に許可する必要があります。許可しないと、呼び出しは失敗します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
```

```
            "Sid": "TagResources",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

例: スポットインスタンスリクエストの作成時にタグ付けを拒否する

次のポリシーでは、インスタンスの起動時に作成されるリソースにタグを付けるためのアクセス許可をユーザーに拒否します。

最初のステートメントでは、一覧表示されたリソースの作成を RunInstances に許可します。spot-instances-request リソースは、RunInstances によって作成されます。このリソースは スポットインスタンス をリクエストします。2 番目のステートメントでは、* ワイルドカードを指定し、インスタンスの起動時に作成されるすべてのリソースのタグ付けを拒否します。spot-instances-request リソースまたは他のリソースの作成時にタグを付けた場合、RunInstances の呼び出しは失敗します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "DenyTagResources",
```

```
            "Effect": "Deny",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

Warning

サポート対象外 - 例: スポットインスタンスリクエストに特定のタグを割り当てる場合にのみ、リクエストの作成を許可する
次のポリシーは、spot-instances-request リソースではサポートされません。
次のポリシーは、スポットインスタンスリクエストに特定のタグを付ける場合にのみ、リクエストを作成するためのアクセス許可を RunInstances に付与することを想定しています。最初のステートメントでは、一覧表示されたリソースの作成を RunInstances に許可します。
2 番目のステートメントでは、スポットインスタンスリクエストにタグ environment=production が付いている場合にのみ、リクエストを作成するためのアクセス許可をユーザーに付与することを想定しています。この条件を RunInstances によって作成された他のリソースに適用する場合、タグを指定しないと、Unauthenticated エラーが発生します。ただし、スポットインスタンスリクエストにタグを指定しない場合、Amazon EC2 は RunInstances ステートメントの spot-instances-request リソースを評価しないため、RunInstances がタグなしのスポットインスタンスリクエストを作成します。environment=production 以外の別のタグを指定すると、Unauthenticated エラーが発生することに注意してください。これは、ユーザーがスポットインスタンスリクエストにタグを付けると、Amazon EC2 が RunInstances ステートメントの spot-instances-request リソースを評価するためです。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
```

```
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*"
            ]
        },
        {
            "Sid": "RequestSpotInstancesOnlyIfTagIs_environment=production - NOT
 SUPPORTED - DO NOT USE!",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        },
        {
            "Sid": "TagResources",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }

    ]
}
```

例: スポットインスタンスリクエストに特定のタグが割り当てられている場合、このリクエストの作成を拒否する

次のポリシーは、スポットインスタンスリクエストにタグ environment=production が付いている場合、このリクエストを作成するためのアクセス許可を RunInstances に拒否します。

最初のステートメントでは、一覧表示されたリソースの作成を RunInstances に許可します。

2 番目のステートメントでは、スポットインスタンスリクエストにタグ environment=production が付いている場合、このリクエストを作成するためのアクセス許可をユーザーに拒否します。environment=production をタグとして指定する

と、Unauthenticated エラーが発生します。他のタグを指定するか、タグを指定しないと、スポットインスタンスリクエストが作成されます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowRun",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances"
            ],
            "Resource": [
                "arn:aws:ec2:us-east-1::image/*",
                "arn:aws:ec2:us-east-1:*:subnet/*",
                "arn:aws:ec2:us-east-1:*:network-interface/*",
                "arn:aws:ec2:us-east-1:*:security-group/*",
                "arn:aws:ec2:us-east-1:*:key-pair/*",
                "arn:aws:ec2:us-east-1:*:volume/*",
                "arn:aws:ec2:us-east-1:*:instance/*",
                "arn:aws:ec2:us-east-1:*:spot-instances-request/*"
            ]
        },
        {
            "Sid": "DenySpotInstancesRequests",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:us-east-1:*:spot-instances-request/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        },
        {
            "Sid": "TagResources",
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

例: リザーブドインスタンス の操作

次のポリシーでは、アカウントで リザーブドインスタンス を表示、変更、購入するアクセス許可をユーザーに与えます。

個別の リザーブドインスタンス にリソースレベルのアクセス許可を設定することはできません。このポリシーは、ユーザーがアカウントのすべての リザーブドインスタンス にアクセスできることを意味します。

Resource 要素は * ワイルドカードを使用して、ユーザーがそのアクションにすべてのリソースを指定できることを示しています。この場合、アカウントのすべての リザーブドインスタンス をリストして変更できます。ユーザーは、アカウント認証情報を使用して リザーブドインスタンス を購入することもできます。また、API アクションがリソースレベルのアクセス許可をサポートしていない場合も、* ワイルドカードが必要です。

```
{
   "Version": "2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeReservedInstances",
        "ec2:ModifyReservedInstances",
        "ec2:PurchaseReservedInstancesOffering",
        "ec2:DescribeAvailabilityZones",
        "ec2:DescribeReservedInstancesOfferings"
      ],
      "Resource": "*"
    }
   ]
}
```

次のコードでは、アカウント内の リザーブドインスタンス を表示および変更できるようにユーザーに許可しています。新しい リザーブドインスタンス の購入は、許可していません。

```
{
  "Version": "2012-10-17",
   "Statement": [
    {
      "Effect": "Allow",
      "Action": [
```

```
        "ec2:DescribeReservedInstances",
        "ec2:ModifyReservedInstances",
        "ec2:DescribeAvailabilityZones"
      ],
      "Resource": "*"
    }
  ]
}
```

例: リソースのタグ付け

次のポリシーでは、タグにキー CreateTags および値 environment が含まれている場合のみ、ユーザーが production アクションを使用してインスタンスにタグを適用できます。他のタグは許可されず、ユーザーは他のリソースタイプをタグ付けすることはできません。

```
{
    "Version": "2012-10-17",
    "Statement": [
              {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/environment": "production"
                }
            }
        }
    ]
}
```

次のポリシーでは、ユーザーは owner のキーとユーザー名の値を使用したタグが既に適用されているタグ付け可能なリソースにタグ付けできます。加えて、ユーザーはリクエストで anycompany:environment-type のキーと値 test または prod を持つタグを指定する必要があります。ユーザーは、リクエストで追加のタグを指定できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
```

```
            {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:region:account-id:*/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/anycompany:environment-type": ["test","prod"],
                    "aws:ResourceTag/owner": "${aws:username}"
                }
            }
        }
    ]
}
```

ユーザーがリソースの特定のタグを指定できるようにする IAM ポリシーを作成できます。例えば、次のポリシーでは、リクエストで指定されたタグキーが environment または cost-center の場合、ユーザーがボリュームのタグを削除できます。タグにはどの値でも指定できますが、指定されたキーのいずれかにタグキーが一致する必要があります。

> **Note**
>
> リソースを削除すると、リソースに関連付けられているすべてのタグも削除されます。タグ付きのリソースを削除する場合、ユーザーは ec2:DeleteTags アクションを使用するためのアクセス許可は必要ありません。削除アクションを実行するためのアクセス許可のみが必要です。

```
{
  "Version": "2012-10-17",
  "Statement": [
       {
      "Effect": "Allow",
      "Action": "ec2:DeleteTags",
      "Resource": "arn:aws:ec2:us-east-1:account-id:volume/*",
      "Condition": {
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["environment","cost-center"]
        }
      }
```

```
    }
  ]
}
```

このポリシーでは、リソースが owner のキーとユーザー名の値で既にタグ付けされている場合のみ、ユーザーが任意のリソースで environment=prod タグのみ削除できます。ユーザーは、リソースの他のタグを削除することはできません。

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteTags"
      ],
      "Resource": "arn:aws:ec2:region:account-id:*/*",
      "Condition": {
        "StringEquals": {
          "aws:RequestTag/environment": "prod",
          "aws:ResourceTag/owner": "${aws:username}"
        },
        "ForAllValues:StringEquals": {
          "aws:TagKeys": ["environment"]
        }
      }
    }
  ]
}
```

例: IAM ロールの使用

次のポリシーでは、department=test タグを持つインスタンス対して IAM ロールのアタッチ、置換、デタッチを行うことをユーザーに許可します。IAM ロールの置換またはデタッチには関連 ID が必要であるため、ポリシーでは ec2:DescribeIamInstanceProfileAssociations アクションを使用するアクセス許可もユーザーに付与します。

ユーザーは、ロールをインスタンスに渡すために iam:PassRole アクションを使用するための許可が必要です。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ec2:AssociateIamInstanceProfile",
        "ec2:ReplaceIamInstanceProfileAssociation",
        "ec2:DisassociateIamInstanceProfile"
      ],
      "Resource": "arn:aws:ec2:us-east-1:account-id:instance/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/department":"test"
        }
      }
    },
    {
      "Effect": "Allow",
      "Action": "ec2:DescribeIamInstanceProfileAssociations",
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::account-id:role/DevTeam*"
    }
  ]
}
```

次のポリシーでは、どのインスタンスに対しても IAM ロールのアタッチまたは置換を行うことをユーザーに許可します。ユーザーは、TestRole- で始まる名前の IAM ロールのみアタッチまたは置換できます。IAM アクションでは、インスタンスプロファイルではなく iam:PassRole ロールの名前を指定します (両方の名前が異なる場合)。詳細については、インスタンスプロファイルを参照してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AssociateIamInstanceProfile",
                "ec2:ReplaceIamInstanceProfileAssociation"
```

```
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DescribeIamInstanceProfileAssociations",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::account-id:role/TestRole-*"
        }
    ]
}
```

例: ルートテーブルの使用

次のポリシーでは、VPC (vpc-ec43eb89) のみに関連付けられているルートテーブルのルートの追加、削除、置換を行うことができます。ec2:Vpc 条件キーの VPC を指定するには、VPC の完全な ARN を指定する必要があります。

```
{
    "Version": "2012-10-17",
    "Statement": [
              {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteRoute",
                "ec2:CreateRoute",
                "ec2:ReplaceRoute"
            ],
            "Resource": [
                "arn:aws:ec2:region:account-id:route-table/*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:Vpc": "arn:aws:ec2:region:account-id:vpc/vpc-ec43eb89"
                }
            }
        }
    ]
}
```

例: 特定のインスタンスが他の AWS サービスでリソースを表示できるようにする

次に示すのは、IAM ロールにアタッチできるポリシーの例です。ポリシーにより、インスタンスは AWS サービスのさまざまなリソースを表示できるようになります。ec2:SourceInstanceARN 条件キーを使用して、リクエストの実行元インスタンスが i-093452212644b0dd6 インスタンスになるように指定します。同じ IAM ロールが別のインスタンスと関連付けられている場合、他のインスタンスはこれらのどのアクションも実行できません。

ec2:SourceInstanceARN キーは AWS グローバル条件キーであるため、Amazon EC2 だけではなく他のサービスアクションにも使用できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
              {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeVolumes",
                "s3:ListAllMyBuckets",
                "dynamodb:ListTables",
                "rds:DescribeDBInstances"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "ArnEquals": {
                    "ec2:SourceInstanceARN": "arn:aws:ec2:region:account-id:instance/
i-093452212644b0dd6"
                }
            }
        }
    ]
}
```

例: 起動テンプレートの使用

次のポリシーでは、ユーザーは起動テンプレートのバージョンを作成して起動テンプレートを変更することができます。ただし、特定の起動テンプレートに限られます (lt-*09477bcd97b0d3abc*)。ユーザーは、他の起動テンプレートを使用することはできません。

```
{
```

```
  "Version": "2012-10-17",
  "Statement": [
   {
      "Action": [
        "ec2:CreateLaunchTemplateVersion",
        "ec2:ModifyLaunchTemplate"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ec2:region:account-id:launch-template/lt-09477bcd97b0d3abc"
    }
  ]
}
```

次のポリシーでは、ユーザーは任意の起動テンプレートと起動テンプレートのバージョンを削除できます。ただし、起動テンプレートに Purpose=Testing のタグがある場合に限ります。

```
{
  "Version": "2012-10-17",
  "Statement": [
       {
      "Action": [
        "ec2:DeleteLaunchTemplate",
        "ec2:DeleteLaunchTemplateVersions"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:ec2:region:account-id:launch-template/*",
      "Condition": {
        "StringEquals": {
          "aws:ResourceTag/Purpose": "Testing"
        }
      }
    }
  ]
}
```

インスタンスメタデータの使用

以下のポリシーでは、インスタンスメタデータサービスバージョン 2 (IMDSv2) を使用して、ユーザーがインスタンスメタデータのみを取得できるようにします。以下の 4 つのポリシーは、4 つのステートメントを使用する 1 つのポリシーに結合できます。1 つのポリシーとして結合すると、このポリシーをサービスコントロールポリシー (SCP) として使用できます。これは、既存の IAM ポリシーに適用する拒否ポリシーとして (既存のアクセス許可を削除して制限するために) 使用したり、アカ

ウント、部門単位 (OU)、組織全体にグローバルに適用する SCP として使用したりすることもできます。

> **Note**
>
> 以下の RunInstances メタデータオプションポリシーは、RunInstances を使用してインスタンスを起動するアクセス許可をプリンシパルに付与するポリシーと組み合わせて使用する必要があります。プリンシパルに RunInstances アクセス許可もない場合、インスタンスを起動することはできません。詳細については、インスタンスの使用とインスタンスの起動 (RunInstances)のポリシーを参照してください。

> **Important**
>
> Auto Scaling グループを使用し、すべての新しいインスタンスで IMDSv2 の使用を要求する必要がある場合は、Auto Scaling グループで 起動テンプレートを使用する必要があります。Auto Scaling グループが起動テンプレートを使用する場合、新しい Auto Scaling グループが作成されるときに IAM プリンシパルの `ec2:RunInstances` アクセス許可がチェックされます。また、既存の Auto Scaling グループが更新され、新しい起動テンプレートまたは新しいバージョンの起動テンプレートが使用される場合にもチェックされます。`RunInstances` の IAM プリンシパルでの IMDSv1 の使用に関する制限は、起動テンプレートを使用している Auto Scaling グループが作成または更新された場合にのみチェックされます。`Latest` または `Default` 起動テンプレートを使用するように設定された Auto Scaling グループでは、起動テンプレートの新しいバージョンが作成されたときにアクセス許可はチェックされません。アクセス許可をチェックするには、特定のバージョンの起動テンプレートを使用するように Auto Scaling グループを設定する必要があります。
>
> Auto Scaling グループによって起動されるインスタンスで IMDSv2 の使用を強制するには、以下の追加ステップが必要です。
>
> 1. 作成された新しいプリンシパルのサービスコントロールポリシー (SCP) または IAM アクセス許可の境界を使用して、組織内のすべてのアカウントの起動設定の使用を無効にします。Auto Scaling グループアクセス許可を持つ既存の IAM プリンシパルの場合、関連するポリシーをこの条件キーで更新します。起動設定の使用を無効にするには、値が `"autoscaling:LaunchConfigurationName"` として指定された `null` 条件キーを使用して、関連する SCP、アクセス許可の境界、または IAM ポリシーを作成または変更します。

2. 新しい起動テンプレートの場合は、起動テンプレートでインスタンスメタデータオプションを設定します。既存の起動テンプレートの場合は、新しいバージョンの起動テンプレートを作成し、新しいバージョンでインスタンスメタデータオプションを設定します。
3. 起動テンプレートを使用するアクセス許可を任意のプリンシパルに付与するポリシーで、`$latest` を指定して `$default` と `"autoscaling:LaunchTemplateVersionSpecified": "true"` の関連付けを制限します。使用を特定のバージョンの起動テンプレートに制限することで、インスタンスメタデータオプションが設定されているバージョンを使用して新しいインスタンスを確実に起動できます。詳細については、Amazon EC2 Auto Scaling API リファレンス (具体的には `Version` パラメータ) のLaunchTemplateSpecificationを参照してください。
4. 起動設定を使用する Auto Scaling グループの場合、起動設定を起動テンプレートに置き換えます。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの起動設定を起動テンプレートと置き換えるを参照してください。
5. 起動テンプレートを使用する Auto Scaling グループの場合、インスタンスメタデータオプションが設定された新しい起動テンプレートを使用するか、インスタンスメタデータオプションが設定された現在の起動テンプレートの新しいバージョンを使用します。詳細については、AWS CLI コマンドリファレンスのupdate-auto-scaling-groupを参照してください。

例

- IMDSv2 の使用を要求する
- IMDSv2 のオプトアウトを拒否する
- ホップ制限の最大値の指定
- インスタンスメタデータオプションを変更できるユーザーの制限
- IMDSv2 からロール認証情報を取得することを要求する

IMDSv2 の使用を要求する

次のポリシーでは、インスタンスが IMDSv2 の使用を要求するようにオプトインされていない限り (`"ec2:MetadataHttpTokens": "required"` で指定)、RunInstances API を呼び出せないように指定します。インスタンスが IMDSv2 を要求するように指定しないと、RunInstances API を呼び出したときに `UnauthorizedOperation` エラーが発生します。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "RequireImdsV2",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "StringNotEquals": {
                    "ec2:MetadataHttpTokens": "required"
                }
            }
        }
    ]
}
```

IMDSv2 のオプトアウトを拒否する

次のポリシーでは、ModifyInstanceMetadataOptions API を呼び出さないように指定し、IMDSv1 または IMDSv2 のオプションを許可します。ModifyInstanceMetadataOptions API を呼び出す場合は、HttpTokens 属性を required に設定する必要があります。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Sid": "DenyIMDSv1HttpTokensModification",
        "Effect": "Deny",
        "Action": "ec2:ModifyInstanceMetadataOptions",
        "Resource": "arn:aws:ec2:*:*:instance/*",
        "Condition": {
            "StringNotEquals": {
                "ec2:Attribute/HttpTokens": "required"
            },
            "Null": {
                "ec2:Attribute/HttpTokens": false
            }
        }
    }]
}
```

ホップ制限の最大値の指定

次のポリシーでは、ホップ制限を指定しない限り、RunInstances API を呼び出せないように指定します。また、ホップ制限を 3 以下にするように指定します。これを指定しないと、RunInstances API を呼び出したときに UnauthorizedOperation エラーが発生します。

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "MaxImdsHopLimit",
            "Effect": "Deny",
            "Action": "ec2:RunInstances",
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "NumericGreaterThan": {
                    "ec2:MetadataHttpPutResponseHopLimit": "3"
                }
            }
        }
    ]
}
```

インスタンスメタデータオプションを変更できるユーザーの制限

次のポリシーでは、一般の管理者がインスタンスメタデータオプションを変更するロール ec2-imds-admins を持つユーザーのみに変更を行うことを許可します。ec2-imds-admins ロール以外のプリンシパルが ModifyInstanceMetadataOptions API を呼び出そうとすると、UnauthorizedOperation エラーが発生します。このステートメントは、ModifyInstanceMetadataOptions API の使用を制御するために使用できます。現在、ModifyInstanceMetadataOptions API 用の詳細なアクセスコントロール (条件) はありません。

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "AllowOnlyImdsAdminsToModifySettings",
            "Effect": "Deny",
            "Action": "ec2:ModifyInstanceMetadataOptions",
            "Resource": "*",
            "Condition": {
                "StringNotLike": {
```

```
                    "aws:PrincipalARN": "arn:aws:iam::*:role/ec2-imds-admins"
                }
            }
        }
    ]
}
```

IMDSv2 からロール認証情報を取得することを要求する

次のポリシーでは、このポリシーを適用したロールを EC2 サービスが引き受けて、結果の認証情報をリクエストの署名に使用する場合は、IMDSv2 から取得した EC2 ロールの認証情報を使用してリクエストに署名する必要があることを指定します。それ以外の場合は、すべての API コールで `UnauthorizedOperation` エラーが発生します。このステートメント/ポリシーは、リクエストが EC2 ロールの認証情報によって署名されていない場合は効果がないため、一般的に適用できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
                {
            "Sid": "RequireAllEc2RolesToUseV2",
            "Effect": "Deny",
            "Action": "*",
            "Resource": "*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:RoleDelivery": "2.0"
                }
            }
        }
    ]
}
```

Amazon EBS ボリュームとスナップショットの使用

Amazon EBS ボリュームとスナップショットを使用するポリシーの例については、「Amazon EBS のアイデンティティベースのポリシー例」を参照してください。

Amazon EC2 コンソールで機能するサンプル ポリシー

IAM ポリシーを使用して Amazon EC2 に必要な許可をユーザーに付与する必要があります。IAM ポリシーを使用して、Amazon EC2 コンソールで特定のリソースを表示、および操作するアクセス許

可をユーザーに付与することができます。上記のセクションのサンプルポリシーを使用することはできますが、これらは AWS CLI または AWS SDK で作成されたリクエスト向けに設計されています。詳細については、「IAM ユーザーガイド」の「AWS CLI または AWS SDK で使用するサンプルポリシー」および「IAM ポリシーの作成」を参照してください。

コンソールではこの機能を実行するために追加の API アクションを使用するので、これらのポリシーは正常に動作しない可能性があります。例えば、`DescribeVolumes` API アクションのみを使用するアクセス許可を持つユーザーがコンソールでボリュームを表示しようとすると、エラーが発生します。このセクションでは、コンソールの特定の部分をユーザーが操作できるようになるポリシーを説明します。Amazon EC2 向けのポリシー作成の詳細については、以下の AWS セキュリティブログの投稿Granting Users Permission to Work in the Amazon EC2 Consoleを参照してください。

> **Tip**
>
> コンソールでタスクを実行するために必要な API アクションを探すには、AWS CloudTrail などのサービスを使用できます。詳細については、AWS CloudTrail ユーザーガイドを参照してください。ポリシーにより特定のリソースを作成または変更するアクセス許可が付与されない場合、コンソールではエンコードされた診断情報のメッセージが表示されます。の DecodeAuthorizationMessage API アクションAWS STS、または AWS CLI の decode-authorization-message コマンドを使用してメッセージをデコードできます。

例

- 例: 読み取り専用アクセス
- 例: EC2 起動インスタンスウィザードの使用
- 例: セキュリティグループの操作
- 例: Elastic IP アドレスの操作
- 例: リザーブドインスタンス の操作

例: 読み取り専用アクセス

ユーザーが Amazon EC2 コンソールですべてのリソースを表示できるようにするには、次の例と同じポリシーを使用します: 例: 読み取り専用アクセス。別のステートメントによりユーザーにアクセス許可が与えられない限り、ユーザーはリソースのアクションを実行したり新しいリソースを作成することができません。

インスタンス、AMI、スナップショットを表示する

代わりに、リソースのサブセットへの読み取り専用アクセスを提供できます。これを行うには、ec2:Describe API アクションの * (ワイルドカード) を各リソースの固有の ec2:Describe アクションに置き換えます。次のポリシーによりユーザーは Amazon EC2 コンソールですべてのインスタンス、AMI、およびスナップショットを表示できます。ec2:DescribeTags アクションにより、ユーザーはパブリック AMI を表示できます。コンソールでタグ付け情報にパブリック AMI を表示させる必要がありますが、ユーザーがプライベート AMI だけを表示できるようにするには、このアクションを削除できます。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeInstances",
         "ec2:DescribeImages",
         "ec2:DescribeTags",
         "ec2:DescribeSnapshots"
      ],
      "Resource": "*"
   }
   ]
}
```

> **Note**
>
> Amazon EC2 ec2:Describe* API アクションは、リソースレベルのアクセス許可をサポートしていません。そのため、ユーザーがコンソールで表示できる個人のリソースを制御できません。したがって、上記のステートメントの Resource エレメントには、* (ワイルドカード) が必要です。どの Amazon EC2 API アクションでどの ARN を使用できるかの詳細については、Amazon EC2 のアクション、リソース、および条件キーを参照してください。

インスタンスと CloudWatch メトリクスを表示する

以下のポリシーは、ユーザーに対して Amazon EC2 コンソールでのインスタンスの表示、[Instances] ページの [Monitoring] タブでの CloudWatch アラームおよびメトリクスの表示を許可します。Amazon EC2 コンソールでは、アラームとメトリクスの表示に CloudWatch API を使用するため、ユーザーに対して

cloudwatch:DescribeAlarms、cloudwatch:DescribeAlarmsForMetric、cloudwatch:ListMet
および cloudwatch:GetMetricData のアクションを使用する許可を付与する必要があります。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeInstances",
         "ec2:DescribeInstanceTypes",
         "cloudwatch:DescribeAlarms",
         "cloudwatch:DescribeAlarmsForMetric",
         "cloudwatch:ListMetrics",
         "cloudwatch:GetMetricStatistics",
         "cloudwatch:GetMetricData"
      ],
      "Resource": "*"
   }
   ]
}
```

例: EC2 起動インスタンスウィザードの使用

Amazon EC2 起動インスタンスウィザードは、インスタンスを設定し、起動するためのオプションを提供する画面です。ユーザーがウィザードのオプションを操作できるように、API アクションを使用するアクセス許可をポリシーに含める必要があります。ポリシーにそれらのアクションを使用するアクセス許可が含まれない場合、ウィザードの一部の項目は適切にロードされず、ユーザーは起動を完了できません。

基本のインスタンス起動ウィザードのアクセス

起動を正常に完了させるには、ユーザーに ec2:RunInstances API アクションを使用するアクセス許可を付与し、少なくとも以下の API アクションを使用できるようにする必要があります。

- ec2:DescribeImages: AMI を表示して選択します。
- ec2:DescribeInstanceTypes: インスタンスタイプを表示および選択します。
- ec2:DescribeVpcs: 使用できるネットワークオプションを表示します。
- ec2:DescribeSubnets: 選択した VPC のすべての使用可能なサブネットを表示します。
- ec2:DescribeSecurityGroups または ec2:CreateSecurityGroup: 既存のセキュリティグループを表示および選択する、または新しいセキュリティグループを作成します。

- ec2:DescribeKeyPairs または ec2:CreateKeyPair: 既存のキーペアを選択する、または新しいキーペアを作成します。
- ec2:AuthorizeSecurityGroupIngress: インバウンドルールを追加します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:CreateSecurityGroup",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateKeyPair"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": "*"
        }
    ]
}
```

ポリシーに次のような API アクションを追加して、ユーザーに追加のオプションを提供できます。

- ec2:DescribeAvailabilityZones: 特定のアベイラビリティーゾーンを選択します。
- ec2:DescribeNetworkInterfaces: 選択したサブネットの既存のネットワークインターフェイスを表示および選択します。
- VPC セキュリティグループにアウトバウンドルールを追加するには、ユーザーに ec2:AuthorizeSecurityGroupEgress API アクションを使用するアクセス許可を付与する必要があります。既存のルールを変更または削除するには、ユーザーに関連する

ec2:RevokeSecurityGroup* API アクションを使用するアクセス許可を付与する必要があります。

- ec2:CreateTags: により作成されたリソースにタグ付けする場合に使用します。RunInstances詳細については、「リソース作成時にタグ付けするアクセス許可の付与」を参照してください。ユーザーにこのアクションを使用する許可がなく、起動インスタンスウィザードのタグ付けページでてタグを適用しようとした場合、起動に失敗します。

> **Important**
>
> インスタンスの起動中に [Name] (名前) を指定すると、タグが作成され、ec2:CreateTags アクションが必要になります。ユーザーに ec2:CreateTags アクションを使用するアクセス許可を付与すると、aws:ResourceTag 条件キーを使用してユーザーによる他のリソースの使用を制限する能力が制限されるため、注意が必要です。ユーザーに ec2:CreateTags アクションを使用するアクセス許可を付与すると、ユーザーがそれらの制限を回避するためにリソースのタグを変更できます。詳細については、「リソースタグを使用した EC2 リソースへのアクセスの制御」を参照してください。

- AMI を選択するときに Systems Manager パラメータを使用するには、ポリシーに ssm:DescribeParameters と ssm:GetParameters を追加する必要があります。ssm:DescribeParameters は、ユーザーに Systems Manager パラメータを表示および選択する許可を付与します。ssm:GetParameters は、ユーザーに Systems Manager パラメータの値を取得する許可を付与します。また、特定の Systems Manager パラメータへのアクセスを制限することもできます。詳細については、このセクションの後半の特定の Systems Manager パラメータへのアクセスの制限を参照してください。

現在、Amazon EC2 Describe* API アクションは、リソースレベルの許可をサポートしていません。そのため、ユーザーが起動インスタンスウィザードで表示できる個人のリソースを制限することはできません。ただし、ec2:RunInstances API アクションにリソースレベルのアクセス許可を適用して、ユーザーがインスタンスの起動に使用できるリソースを制限できます。ユーザーが使用する権限がないオプションを選択すると、起動は失敗します。

特定のインスタンスタイプ、サブネット、リージョンへのアクセスの制限

次のポリシーにより、ユーザーは Amazon が所有する AMI を使用して t2.micro インスタンスを特定のサブネット (subnet-1a2b3c4d) でのみ起動することができます。ユーザーは sa-east-1 リージョンでのみ起動できます。ユーザーが異なるリージョンを選択するか、起動インスタンスウィザードで異なるインスタンスタイプ、AMI、サブネットを選択すると、起動は失敗します。

最初のステートメントでは、上記の例で説明したように、起動インスタンスウィザードでオプションを表示する許可または新しいオプションを作成する許可がユーザーに付与されます。2 番目のステートメントでは、ec2:RunInstances アクションでネットワークインターフェイス、ボリューム、キーペア、セキュリティグループ、サブネットリソースを使用するアクセス許可が付与されます。これは、ユーザーが VPC でインスタンスを起動するために必要です。ec2:RunInstances アクションの使用方法の詳細については、インスタンスの起動 (RunInstances)を参照してください。3 番目と 4 番目のステートメントは、それぞれインスタンスと AMI リソースを使用するための許可をユーザーに付与しますが、これは、インスタンスが t2.micro インスタンスであり、AMI が Amazon または特定の信頼できる検証済みのパートナーによって所有されている場合に限られます。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeInstances",
         "ec2:DescribeImages",
         "ec2:DescribeInstanceTypes",
         "ec2:DescribeKeyPairs",
         "ec2:CreateKeyPair",
         "ec2:DescribeVpcs",
         "ec2:DescribeSubnets",
         "ec2:DescribeSecurityGroups",
         "ec2:CreateSecurityGroup",
         "ec2:AuthorizeSecurityGroupIngress"
   ],
   "Resource": "*"
   },
   {
      "Effect": "Allow",
      "Action":"ec2:RunInstances",
      "Resource": [
         "arn:aws:ec2:sa-east-1:111122223333:network-interface/*",
         "arn:aws:ec2:sa-east-1:111122223333:volume/*",
         "arn:aws:ec2:sa-east-1:111122223333:key-pair/*",
         "arn:aws:ec2:sa-east-1:111122223333:security-group/*",
         "arn:aws:ec2:sa-east-1:111122223333:subnet/subnet-1a2b3c4d"
      ]
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
```

```
      "Resource": [
         "arn:aws:ec2:sa-east-1:111122223333:instance/*"
      ],
      "Condition": {
         "StringEquals": {
            "ec2:InstanceType": "t2.micro"
         }
      }
   },
   {
      "Effect": "Allow",
      "Action": "ec2:RunInstances",
      "Resource": [
            "arn:aws:ec2:sa-east-1::image/ami-*"
      ],
      "Condition": {
         "StringEquals": {
            "ec2:Owner": "amazon"
         }
      }
   }
   ]
}
```

特定の Systems Manager パラメータへのアクセスの制限

次のポリシーは、特定の名前の Systems Manager パラメータを使用するアクセスを許可します。

1 つ目のステートメントは、起動インスタンスウィザードで AMI を選択するときに Systems Manager パラメータを表示する許可をユーザーに付与します。2 つ目のステートメントは、`prod-*` という名前のパラメータのみを使用するアクセス許可をユーザーに付与します。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ssm:DescribeParameters"
      ],
      "Resource": "*"
   },
   {
      "Effect": "Allow",
```

```
      "Action": [
         "ssm:GetParameters"
      ],
     "Resource": "arn:aws:ssm:us-east-2:123456123:parameter/prod-*"
   }
   ]
}
```

例: セキュリティグループの操作

セキュリティグループを表示し、ルールを追加/削除する

次のポリシーは、Amazon EC2 コンソールでセキュリティグループを表示し、インバウンドおよびアウトバウンドのルールを追加および削除し、タグ Department=Test を含む既存のセキュリティグループのルール説明を変更するアクセス許可をユーザーに付与します。

最初のステートメントの ec2:DescribeTags アクションにより、ユーザーはコンソールでタグを表示できます。これにより、ユーザーは変更できるセキュリティグループをより簡単に識別できます。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeSecurityGroups",
         "ec2:DescribeSecurityGroupRules",
         "ec2:DescribeTags"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ec2:AuthorizeSecurityGroupIngress",
         "ec2:RevokeSecurityGroupIngress",
         "ec2:AuthorizeSecurityGroupEgress",
         "ec2:RevokeSecurityGroupEgress",
         "ec2:ModifySecurityGroupRules",
         "ec2:UpdateSecurityGroupRuleDescriptionsIngress",
         "ec2:UpdateSecurityGroupRuleDescriptionsEgress"
      ],
      "Resource": [
```

```
            "arn:aws:ec2:region:111122223333:security-group/*"
         ],
         "Condition": {
            "StringEquals": {
               "aws:ResourceTag/Department": "Test"
            }
         }
      },
      {
         "Effect": "Allow",
         "Action": [
            "ec2:ModifySecurityGroupRules"
         ],
         "Resource": [
            "arn:aws:ec2:region:111122223333:security-group-rule/*"
         ]
      }
]}
```

[Create Security Group] ダイアログボックスの使用

ユーザーが Amazon EC2 コンソールの [Create Security Group] ダイアログボックスを使用して作業できるようにするポリシーを作成できます。このダイアログボックスを使用するには、ユーザーに少なくとも以下の API アクションを使用するアクセス許可を付与する必要があります。

- `ec2:CreateSecurityGroup`: 新しいセキュリティグループを作成するには
- `ec2:DescribeVpcs`: [VPC] リストに既存の VPC のリストを表示します。

これらのアクセス許可で、ユーザーは新しいセキュリティグループを正常に作成できますが、ルールを追加することはできません。[Create Security Group] ダイアログボックスでルールを操作するには、ポリシーに次の API アクションを追加します。

- `ec2:AuthorizeSecurityGroupIngress`: インバウンドルールを追加します。
- `ec2:AuthorizeSecurityGroupEgress`: VPC セキュリティグループにアウトバウンドルールを追加します。
- `ec2:RevokeSecurityGroupIngress`: 既存のインバウンドルールを変更または削除します。これは、ユーザーがコンソールで [Copy to new] 機能を使用できるようにするために役に立ちます。この機能により、[Create Security Group] ダイアログボックスが開き、選択したセキュリティグループと同じルールが追加されます。

- ec2:RevokeSecurityGroupEgress: VPC セキュリティグループのアウトバウンドルールを変更または削除します。これは、すべてのアウトバウンドトラフィックを許可するデフォルトのアウトバウンドルールを変更または削除する場合に役に立ちます。
- ec2:DeleteSecurityGroup: 無効なルールを保存できないときに対応します。コンソールでは、最初にセキュリティグループを作成し、次に指定されたルールを追加します。ルールが無効である場合、アクションは失敗し、コンソールによってセキュリティグループの削除が試行されます。引き続き、[Create Security Group] ダイアログボックスが利用できるため、ユーザーは無効なルールを修正してセキュリティグループを再作成できます。この API アクションは必須ではありませんが、ユーザーにこのアクションを使用するアクセス許可が付与されておらず、無効なルールを持つセキュリティグループを作成しようとすると、ルールのないセキュリティグループが作成され、後でルールを追加することが必要になります。
- ec2:UpdateSecurityGroupRuleDescriptionsIngress: 入力 (受信) セキュリティグループルールの説明を追加または更新するには
- ec2:UpdateSecurityGroupRuleDescriptionsEgress: 出力 (送信) セキュリティグループルールの説明を追加または更新するには
- ec2:ModifySecurityGroupRules: セキュリティグループのルールを変更します。
- ec2:DescribeSecurityGroupRules: セキュリティグループのルールを一覧表示します。

次のポリシーは、[Create Security Group] ダイアログボックスを使用し、特定の VPC (vpc-1a2b3c4d) に関連付けられたセキュリティグループに対してインバウンドおよびアウトバウンドのルールを作成するアクセス許可をユーザーに付与します。ユーザーは VPC のセキュリティグループを作成できますが、ルールを追加することはできません。同様に、ユーザーは VPC vpc-1a2b3c4d に関連付けられていないの既存のセキュリティグループにルールを追加することもできません。ユーザーには、コンソールですべてのセキュリティグループを表示するアクセス許可も付与されます。これにより、ユーザーはインバウンドルールを追加するセキュリティグループをより簡単に識別できるようになります。このポリシーは、ユーザーに VPC vpc-1a2b3c4d に関連付けられたセキュリティグループを削除するアクセス許可も付与します。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
        "ec2:DescribeSecurityGroups",
        "ec2:CreateSecurityGroup",
        "ec2:DescribeVpcs"
      ],
```

```
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteSecurityGroup",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:AuthorizeSecurityGroupEgress"
      ],
      "Resource": "arn:aws:ec2:region:111122223333:security-group/*",
      "Condition":{
         "ArnEquals": {
            "ec2:Vpc": "arn:aws:ec2:region:111122223333:vpc/vpc-1a2b3c4d"
         }
      }
    }
   ]
}
```

例: Elastic IP アドレスの操作

Amazon EC2 コンソールで Elastic IP アドレスを確認することをユーザーに許可するには、`ec2:DescribeAddresses` アクションを使用するためのアクセス許可をユーザーに付与します。

Elastic IP アドレスの使用をユーザーに許可する場合は、ポリシーに次のアクションを追加できます。

- `ec2:AllocateAddress`: Elastic IP アドレスを割り当てます。
- `ec2:ReleaseAddress`: Elastic IP アドレスをリリースするには。
- `ec2:AssociateAddress`: Elastic IP アドレスをインスタンスまたはネットワークインターフェイスに関連付けます。
- `ec2:DescribeNetworkInterfaces` と `ec2:DescribeInstances`: [Associate address] で使用します。この画面には、Elastic IP アドレスを関連付けることができるインスタンスまたはネットワークインターフェイスが表示されます。
- `ec2:DisassociateAddress`: Elastic IP アドレスとインスタンスまたはネットワークインターフェイスの関連付けを解除します。

次のポリシーでは、Elastic IP アドレスの表示、割り当て、インスタンスとの関連付けを行うことができます。ユーザーは Elastic IP アドレスとネットワークインターフェイスの関連付け、Elastic IP アドレスの関連付けの解除、または Elastic IP アドレスのリリースを行うことはできません。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAddresses",
                "ec2:AllocateAddress",
                "ec2:DescribeInstances",
                "ec2:AssociateAddress"
            ],
            "Resource": "*"
        }
    ]
}
```

例: リザーブドインスタンス の操作

次のポリシーにより、アカウントのリザーブドインスタンスの表示と変更、および AWS Management Console での新しいリザーブドインスタンスの購入をすることができます。

このポリシーにより、ユーザーがアカウント内のすべての リザーブドインスタンス と オンデマンドインスタンス を表示できるようになります。個別の リザーブドインスタンス にリソースレベルのアクセス許可を設定することはできません。

```
{
   "Version": "2012-10-17",
   "Statement": [{
      "Effect": "Allow",
      "Action": [
         "ec2:DescribeReservedInstances",
         "ec2:ModifyReservedInstances",
         "ec2:PurchaseReservedInstancesOffering",
         "ec2:DescribeInstances",
         "ec2:DescribeInstanceTypes",
         "ec2:DescribeAvailabilityZones",
         "ec2:DescribeReservedInstancesOfferings"
      ],
```

```
      "Resource": "*"
   }
   ]
}
```

ec2:DescribeAvailabilityZones アクションは、リザーブドインスタンス を購入できるアベイラビリティーゾーンに関する情報を Amazon EC2 コンソールで表示できるようにするために必要です。ec2:DescribeInstances アクションは必須ではありませんが、このアクションにより、ユーザーがアカウントのインスタンスを表示し、正しい仕様に合わせて予約を購入できるようになります。

ec2:DescribeInstances を削除するなど、API アクションを調整してユーザーアクセスを制限できます。ec2:DescribeAvailabilityZones はユーザーが読み取り専用アクセスを持っていることを意味します。

Amazon Elastic Compute Cloud での AWS の管理ポリシー

ユーザー、グループ、ロールにアクセス許可を追加するには、自分でポリシーを作成するよりも、AWS 管理ポリシーを使用する方が簡単です。チームに必要な権限のみを提供する IAM カスタマーマネージドポリシーを作成するには、時間と専門知識が必要です。すぐに使用を開始するために、AWS マネージドポリシーを使用できます。これらのポリシーは、一般的なユースケースを対象範囲に含めており、AWS アカウントで利用できます。AWS マネージドポリシーの詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

AWS のサービスは、AWS マネージドポリシーを維持および更新します。AWS マネージドポリシーの許可を変更することはできません。サービスでは、新しい機能を利用できるようにするために、AWS マネージドポリシーに権限が追加されることがあります。この種類の更新は、ポリシーがアタッチされている、すべてのアイデンティティ (ユーザー、グループおよびロール) に影響を与えます。新しい機能が立ち上げられた場合や、新しいオペレーションが使用可能になった場合に、各サービスが AWS マネージドポリシーを更新する可能性が最も高くなります。サービスは、AWS マネージドポリシーから権限を削除しないため、ポリシーの更新によって既存の権限が破棄されることはありません。

さらに、AWS は、複数のサービスにまたがるジョブ機能の特徴に対するマネージドポリシーもサポートしています。例えば、ReadOnlyAccess AWS マネージドポリシーでは、すべての AWS のサービスおよびリソースへの読み取り専用アクセスを許可します。サービスが新しい機能を起動する場合、AWS は、新たなオペレーションとリソース用に、読み取り専用の許可を追加します。ジョブ機能ポリシーのリストと説明については、IAM ユーザーガイドのジョブ機能の AWS 管理ポリシーを参照してください。

AWS 管理ポリシー: AmazonEC2FullAccess

AmazonEC2FullAccess ポリシーは IAM アイデンティティにアタッチできます。このポリシーでは、Amazon EC2 への完全なアクセスを可能にする許可を付与します。

このポリシーの許可を確認するには、「AWS マネージドポリシーリファレンス」の「AmazonEC2FullAccess」を参照してください。

AWS 管理ポリシー: AmazonEC2ReadOnlyAccess

AmazonEC2ReadOnlyAccess ポリシーは IAM アイデンティティにアタッチできます。このポリシーでは、Amazon EC2 に対する読み取り専用アクセスを可能にする許可を付与します。

このポリシーの許可を確認するには、「AWS マネージドポリシーリファレンス」の「AmazonEC2ReadOnlyAccess」を参照してください。

AWS マネージドポリシー: AWSEC2CapacityReservationFleetRolePolicy

このポリシーは、AWSServiceRoleForEC2CapacityReservationFleet という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにキャパシティ予約を作成、変更、およびキャンセルすることを、キャパシティ予約に許可します。詳細については、「キャパシティ予約フリートのサービスにリンクされたロール」を参照してください。

AWS マネージドポリシー: AWSEC2FleetServiceRolePolicy

このポリシーは、AWSServiceRoleForEC2Fleet という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにインスタンスのリクエスト、起動、終了、タグ付けを行うことを、EC2 フリートに許可します。詳細については、「EC2 フリート用のサービスにリンクされたロール」を参照してください。

AWS マネージドポリシー: AWSEC2SpotFleetServiceRolePolicy

このポリシーは、AWSServiceRoleForEC2SpotFleet という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにインスタンスの起動および管理を行うことをスポットフリートに許可します。詳細については、「スポットフリート用のサービスにリンクされたロール」を参照してください。

AWS マネージドポリシー: AWSEC2SpotServiceRolePolicy

このポリシーは、AWSServiceRoleForEC2Spot という名前のサービスにリンクされたロールにアタッチされ、ユーザーの代わりにスポットインスタンス の起動および管理を行うことを、Amazon

EC2 に許可します。詳細については、「スポットインスタンスリクエスト向けのサービスにリンクされたロール」を参照してください。

Amazon EC2 での AWS 管理ポリシーに関する更新

Amazon EC2 の AWS 管理ポリシーに対する更新の詳細について、このサービスがこれらの変更の追跡を開始した以降のものを示します。

変更	説明	日付
Amazon EC2 が変更の追跡を開始しました。	Amazon EC2 が AWS 管理ポリシーの変更の追跡を開始しました	2021 年 3 月 1 日

Amazon EC2 の IAM ロール

アプリケーションは AWS 認証情報を使用して API リクエストに署名する必要があります。したがって、アプリケーションデベロッパーである場合、EC2 インスタンスで実行するアプリケーションの認証情報を管理する戦略が必要です。例えば、AWS 認証情報をインスタンスに安全に配布することで、他のユーザーから認証情報を保護しながら、それらのインスタンスのアプリケーションで認証情報を使用してリクエストに署名できます。ただし、各インスタンスに認証情報を安全に配布することは難しく、特に AWS がユーザーの代わりに作成するスポットインスタンスや Auto Scaling グループのインスタンスなどではそれが顕著です。また、AWS 認証情報を循環させる場合、各インスタンスの認証情報を更新できる必要もあります。

> **Note**
>
> Amazon EC2 ワークロードでは、次に説明する方法を使用してセッション認証情報を取得することをお勧めします。これらの認証情報により、インスタンスに既に関連付けられている同じロールを引き受けるために `sts:AssumeRole` を使用する必要なしに、ワークロードが AWS API リクエストを実行できるようにすることができます。属性ベースのアクセスコントロール (ABAC) のためにセッションタグを渡す必要がある場合や、ロールの許可をさらに制限するためにセッションポリシーを渡す必要がある場合でない限り、このようなロール引き受け呼び出しは不要です。これは、同じ一時的なロールセッション認証情報の新しいセットを作成するためです。
> ワークロードがロールを使用してそれ自体を引き受ける場合は、その旨を明示的に許可する信頼ポリシーを作成する必要があります。信頼ポリシーを作成しない場合、`AccessDenied`

エラーが発生します。詳細については、「IAM ユーザーガイド」の「Modifying a role trust policy」(ロールの信頼ポリシーの変更) を参照してください。

アプリケーションが使用するセキュリティ認証情報をお客様が管理する必要なく、アプリケーションがインスタンスから API リクエストを安全に作成できるように、IAM ロールをデザインしました。AWS 認証情報を作成および配布する代わりに、以下の方法で、IAM ロールを使用して API リクエストを作成するアクセス許可を委任できます。

1. IAM ロールを作成します。
2. ロールを行うアカウントまたは AWS のサービスを定義する
3. ロールを引き受けた後で、アプリケーションで使用できる API アクションおよびリソースを定義します。
4. インスタンスの起動時にロールを指定するか、既存のインスタンスにロールをアタッチします。
5. アプリケーションで一時的な認証情報のセットを取得して使用します。

例えば、IAM ロールを使用し、Amazon S3 のバケットを使用する必要のあるインスタンスで実行中のアプリケーションに、アクセス許可を与えることができます。JSON 形式のポリシーを作成することにより、IAM ロールのアクセス許可を指定できます。これらのポリシーは、 ユーザー用に作成するポリシーに類似しています。ロールを変更すると、その変更はすべてのインスタンスに反映されます。

> **Note**
>
> Amazon EC2 IAM ロールの認証情報は、ロールで設定された最大セッション期間の対象にはなりません。詳細については、「IAM ユーザーガイド」の「IAM ロールの使用」を参照してください。

IAM ロールを作成するとき、アプリケーションが必要とする特定の API コールへのアクセスを制限する最小権限の IAM ポリシーを関連付けます。

インスタンスにアタッチできる IAM ロールは 1 つだけですが、同じロールを複数のインスタンスにアタッチできます。IAM ロールの作成と使用の詳細については、IAM ユーザーガイドのロールを参照してください。

リソースレベルのアクセス許可を IAM ポリシーに適用して、インスタンスの IAM ロールのアタッチ、置換、またはデタッチをユーザーに許可するかどうを制御できます。詳細については、Amazon EC2 API アクションでサポートされるリソースレベルのアクセス許可と、例: IAM ロールの使用の例を参照してください。

コンテンツ

インスタンスプロファイル

Amazon EC2 は、IAM ロールのコンテナとしてインスタンスプロファイルを使用します。IAM コンソールを使用して IAM ロールを作成すると、コンソールによりインスタンスプロファイルが自動的に作成され、対応するロールと同じ名前が付けられます。Amazon EC2 コンソールを使用して IAM ロールを持つインスタンスを起動する場合、またはインスタンスに IAM ロールをアタッチする場合は、インスタンスプロファイル名のリストに基づいてロールを選択します。

AWS CLI、API、または AWS SDK を使用してロールを作成する場合、ロールとインスタンスプロファイルを別個のアクションとして作成します。名前は異なる可能性があります。次に AWS CLI、API、または AWS SDK を使用して IAM ロールを持つインスタンスを起動する場合、またはインスタンスに IAM ロールをアタッチする場合は、インスタンスプロファイル名を指定します。

インスタンスプロファイルに含めることができる IAM ロールの数は 1 つのみです。この制限を増やすことはできません。

詳細については、IAM ユーザーガイドのインスタンスプロファイルを参照してください。

インスタンスメタデータからのセキュリティ認証情報の取得

インスタンスのアプリケーションは、インスタンスメタデータアイテム `iam/security-credentials/role-name` のロールから提供されたセキュリティ認証情報を取得します。アプリケーションには、ロールに関連付けられたセキュリティ認証情報によって、ロールに対して定義したアクションおよびリソースのアクセス許可が付与されます。これらのセキュリティ認証情報は一時的なものであり、私たちが自動的に循環させます。新しい認証情報は、古い認証情報が失効する少なくとも 5 分前から有効になるようにします。

> **Warning**
>
> IAM ロールでインスタンスメタデータを使用するサービスを使用する場合は、サービスで HTTP 呼び出しが行われるときに認証情報を公開しないように注意する必要があります。認証情報を公開できるサービスの種類には、HTTP プロキシ、HTML/CSS 検証サービス、および XML インクルードをサポートする XML プロセッサーが含まれます。

以下のコマンドでは、s3access という名前の IAM ロールのセキュリティ認証情報を取得します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/iam/security-credentials/s3access
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/iam/security-credentials/s3access
```

出力例を次に示します。

```
{
  "Code" : "Success",
  "LastUpdated" : "2012-04-26T16:39:16Z",
  "Type" : "AWS-HMAC",
  "AccessKeyId" : "ASIAIOSFODNN7EXAMPLE",
  "SecretAccessKey" : "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
  "Token" : "token",
  "Expiration" : "2017-05-17T15:09:54Z"
}
```

インスタンスで実行されるアプリケーション、AWS CLI、および Tools for Windows PowerShell コマンドのために、一時的なセキュリティ認証情報を明示的に取得する必要はありません。AWS SDK、AWS CLI、および Tools for Windows PowerShell は、EC2 インスタンスメタデータサービスから自動的に認証情報を取得し、使用します。一時的なセキュリティ認証情報を使用してインスタンスの外部で呼び出しを行う (IAM ポリシーをテストするなど) には、アクセスキー、秘密キー、およ

びセッショントークンを提供する必要があります。詳細については、IAM ユーザーガイドのAWS リソースへのアクセスを要求するための一時的なセキュリティ認証情報の使用を参照してください。

インスタンスのメタデータの詳細については、インスタンスメタデータとユーザーデータを参照してください。インスタンスメタデータの IP アドレスについては、インスタンスメタデータの取得を参照してください。

IAM ロールをインスタンスに渡すための許可をユーザーに付与する

ユーザーに対して、IAM ロールを持つインスタンスの起動、および既存インスタンスへの IAM ロールのアタッチと置き換えを許可するには、以下の API アクションを実行するための許可を付与します。

- `iam:PassRole`
- `ec2:AssociateIamInstanceProfile`
- `ec2:ReplaceIamInstanceProfileAssociation`

例えば、次の IAM ポリシーでは、IAM ロールを持つインスタンスの起動や、既存のインスタンスへの IAM ロールのアタッチならびに置換を行うためのアクセス許可を、AWS CLI を使用しながらユーザーに付与しています。

> **Note**
>
> ユーザーにすべてのロールへのアクセス許可を付与するポリシーが必要な場合は、そのポリシーの中でリソースを * として指定します。ただし、最小権限の原則によるベストプラクティスを考慮してください。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ec2:RunInstances",
         "ec2:AssociateIamInstanceProfile",
         "ec2:ReplaceIamInstanceProfileAssociation"
      ],
      "Resource": "*"
```

```
    },
    {
      "Effect": "Allow",
      "Action": "iam:PassRole",
      "Resource": "arn:aws:iam::123456789012:role/DevTeam*"
    }
  ]
}
```

Amazon EC2 コンソールを使用して、IAM ロールを持つインスタンスを起動したり、既存のインスタンスで IAM ロールをアタッチおよび置換したりするための、アクセス許可をユーザーに付与するには、必要であれば他の権限を追加しながら `iam:ListInstanceProfiles`、`iam:PassRole`、`ec2:AssociateIamInstanceProfile`、および `ec2:ReplaceIamInstanceProfileAssociation` を使用します。エンドポイントポリシーの例については、Amazon EC2 コンソールで機能するサンプル ポリシーを参照してください。

IAM ロールの使用

IAM ロールは、インスタンスの起動時または起動後に作成してインスタンスにアタッチできます。インスタンスの IAM ロールは、置換またはデタッチすることもできます。

コンテンツ

IAM ロールを作成する

IAM ロールを持つインスタンスを起動したり、インスタンスにアタッチしたりするには、そのロールを事前に作成する必要があります。

Console

IAM コンソールを使用して IAM ロールを作成するには

1. IAM コンソール (https://console.aws.amazon.com/iam/) を開きます。

2. ナビゲーションペインで、[ロール]、[ロールの作成] の順に選択します。
3. [信頼できるエンティティの選択] ページで [AWS のサービス] を選択し、[EC2] ユースケースを選択します。[Next] を選択します。
4. [許可を追加する] ページで、必要なリソースに対するアクセス許可をインスタンスに付与するポリシーを選択します。[Next] を選択します。
5. [名前、確認、および作成] ページでロールの名前と説明を入力します。必要に応じて、ロールにタグを追加します。[ロールの作成] を選択します。

Command line

次の例では、IAM ロールを作成し、このロールに Amazon S3 バケットの使用を許可するポリシーを割り当てます。

IAM ロールおよびインスタンスプロファイルを作成するには (AWS CLI)

1. 以下の信頼ポリシーを作成し、`ec2-role-trust-policy.json` という名前のテキストファイルに保存します。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "Service": "ec2.amazonaws.com"},
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. `s3access` ロールを作成し、create-role コマンドを使用して作成した信頼ポリシーを指定します。

```
aws iam create-role \
    --role-name s3access \
    --assume-role-policy-document file://ec2-role-trust-policy.json
```

レスポンスの例

```
{
```

```
    "Role": {
        "AssumeRolePolicyDocument": {
            "Version": "2012-10-17",
            "Statement": [
                {
                    "Action": "sts:AssumeRole",
                    "Effect": "Allow",
                    "Principal": {
                        "Service": "ec2.amazonaws.com"
                    }
                }
            ]
        },
        "RoleId": "AROAIIZKPBKS2LEXAMPLE",
        "CreateDate": "2013-12-12T23:46:37.247Z",
        "RoleName": "s3access",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:role/s3access"
    }
}
```

3. アクセスポリシーを作成し、ec2-role-access-policy.json という名前のテキストファイルに保存します。例えば、このポリシーは、インスタンスで実行しているアプリケーションに対し、Amazon S3 の管理権限を与えます。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": ["s3:*"],
      "Resource": ["*"]
    }
  ]
}
```

4. put-role-policy コマンドを使用して、アクセスポリシーをロールにアタッチします。

```
aws iam put-role-policy \
    --role-name s3access \
    --policy-name S3-Permissions \
    --policy-document file://ec2-role-access-policy.json
```

5. create-instance-profile コマンドを使用して、s3access-profile という名前のインスタンスプロファイルを作成します。

```
aws iam create-instance-profile --instance-profile-name s3access-profile
```

レスポンスの例

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJTLBPJLEGREXAMPLE",
        "Roles": [],
        "CreateDate": "2013-12-12T23:53:34.093Z",
        "InstanceProfileName": "s3access-profile",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/s3access-profile"
    }
}
```

6. s3access インスタンスプロファイルに s3access-profile ロールを追加します。

```
aws iam add-role-to-instance-profile \
    --instance-profile-name s3access-profile \
    --role-name s3access
```

または、以下の AWS Tools for Windows PowerShell コマンドを使用することもできます。

- New-IAMRole
- Register-IAMRolePolicy
- New-IAMInstanceProfile

IAM ロールを使用したインスタンスの起動

IAM ロールを作成した後、インスタンスを起動して、起動中にそのロールをインスタンスに関連付けることができます。

> **Important**
>
> IAM ロールを作成した後、適切なアクセス許可が反映されるまで数秒ほどかかります。ロールを使用した最初のインスタンスの起動が失敗した場合は、数秒待ってからもう一度試して

ください。詳細については、「IAM ユーザーガイド」の「IAM ロールのトラブルシューティング」を参照してください。

New console

IAM ロールを使用してインスタンスを起動するには (コンソール)

1. インスタンスを起動するための手順に従います。
2. [Advanced details] (高度な詳細) を展開し、[IAM instance profile] (IAM インスタンスプロファイル) で、作成した IAM ロールを選択します。

> **Note**
>
> [IAM instance profile] (IAM インスタンスプロファイル) リストには、IAM ロールの作成時に作成したインスタンスプロファイルの名前が表示されます。コンソールを使用して IAM ロールを作成した場合、インスタンスプロファイルが自動的に作成され、ロールと同じ名前が付けられます。AWS CLI、API、または AWS SDK を使用して IAM ロールを作成した場合、インスタンスプロファイルに異なる名前を付けた可能性があります。

3. インスタンスに必要なその他の詳細を設定するか、デフォルトを受け入れて、キーペアを選択します。インスタンス起動ウィザードのフィールドについては、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
4. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。
5. アプリケーションで Amazon EC2 API アクションを使用している場合、インスタンスで有効にされている AWS セキュリティ認証情報を取得し、それを使用しリクエストに署名します。これは、AWS SDK によって行われます。

   IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
 "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/iam/security-credentials/role_name
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/iam/security-credentials/role_name
```

Old console

IAM ロールを使用してインスタンスを起動するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ダッシュボードで、[Launch Instance (インスタンスの起動)] を選択します。
3. AMI およびインスタンスタイプを選択し、[Next: Configure Instance Details] を選択します。
4. [Configure Instance Details] ページの [IAM role] で、作成した IAM ロールを選択します。

> **Note**
>
> [IAM role] リストには、IAM ロールの作成時に作成したインスタンスプロファイルの名前が表示されます。コンソールを使用して IAM ロールを作成した場合、インスタンスプロファイルが自動的に作成され、ロールと同じ名前が付けられます。AWS CLI、API、または AWS SDK を使用して IAM ロールを作成した場合、インスタンスプロファイルに異なる名前を付けた可能性があります。

5. その他の詳細を設定し、ウィザードの残りの部分の指示に従うか、[Review and Launch] を選択してデフォルト設定を受け入れ、直接 [Review Instance Launch] ページに移動します。
6. 設定を確認して [Launch] を選択し、キーペアを選択してインスタンスを起動します。
7. アプリケーションで Amazon EC2 API アクションを使用している場合、インスタンスで有効にされている AWS セキュリティ認証情報を取得し、それを使用しリクエストに署名します。これは、AWS SDK によって行われます。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H
 "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/
meta-data/iam/security-credentials/role_name
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/iam/security-credentials/role_name
```

Command line

AWS CLI を使用して起動時にロールをインスタンスに関連付けることもできます。コマンド内でインスタンスプロファイルを指定する必要があります。

IAM ロールを使用してインスタンスを起動するには (AWS CLI)

1. run-instances コマンドでインスタンスプロファイルを使用してインスタンスを起動します。以下の例は、インスタンスプロファイルを使用してインスタンスを起動する方法を示しています。

   ```
   aws ec2 run-instances \
       --image-id ami-11aa22bb \
       --iam-instance-profile Name="s3access-profile" \
       --key-name my-key-pair \
       --security-groups my-security-group \
       --subnet-id subnet-1a2b3c4d
   ```

   または、New-EC2Instance Tools for Windows PowerShell コマンドを使用することもできます。

2. アプリケーションで Amazon EC2 API アクションを使用している場合、インスタンスで有効にされている AWS セキュリティ認証情報を取得し、それを使用しリクエストに署名します。これは、AWS SDK によって行われます。

   ```
   curl http://169.254.169.254/latest/meta-data/iam/security-credentials/role_name
   ```

インスタンスへの IAM ロールのアタッチ

ロールを持たないインスタンスに IAM ロールをアタッチするには、そのインスタンスを `stopped` または `running` の状態にします。

Console

IAM ロールをインスタンスにアタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択し、[アクション]、[セキュリティ]、[IAM ロールの変更] の順に選択します。
4. インスタンスにアタッチする IAM ロールを選択して、[保存] を選択します。

Command line

IAM ロールをインスタンスにアタッチするには (AWS CLI)

1. 必要に応じて、インスタンスを記述して、ロールをアタッチするインスタンスの ID を取得します。

```
aws ec2 describe-instances
```

2. associate-iam-instance-profile コマンドでインスタンスプロファイルを指定して、IAM ロールをインスタンスにアタッチします。インスタンスプロファイルの Amazon リソースネーム (ARN) またはプロファイル名を使用できます。

```
aws ec2 associate-iam-instance-profile \
    --instance-id i-1234567890abcdef0 \
    --iam-instance-profile Name="TestRole-1"
```

レスポンスの例

```
{
    "IamInstanceProfileAssociation": {
        "InstanceId": "i-1234567890abcdef0",
        "State": "associating",
        "AssociationId": "iip-assoc-0dbd8529a48294120",
        "IamInstanceProfile": {
            "Id": "AIPAJLNLDX3AMYZNWYYAY",
            "Arn": "arn:aws:iam::123456789012:instance-profile/TestRole-1"
        }
    }
```

```
}
```

または、以下の Tools for Windows PowerShell コマンドを使用します。

- Get-EC2Instance
- Register-EC2IamInstanceProfile

IAM ロールの置換

既に IAM ロールが割り当てられているインスタンスで IAM ロールを置き換えるには、インスタンスは `running` 状態になっている必要があります。既存のロールをデタッチしないでインスタンスの IAM ロールを変更する場合に、これを行うことができます。例えば、インスタンスで実行しているアプリケーションが実行する API アクションが中断されないようにするために、これを行うことができます。

Console

インスタンスの IAM ロールを置き換えるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択し、[アクション]、[セキュリティ]、[IAM ロールの変更] の順に選択します。
4. インスタンスにアタッチする IAM ロールを選択して、[保存] を選択します。

Command line

インスタンスの IAM ロールを置き換えるには (AWS CLI)

1. 必要に応じて、IAM インスタンスプロファイルの関連付けを記述し、置き換える IAM インスタンスプロファイルの関連 ID を取得します。

```
aws ec2 describe-iam-instance-profile-associations
```

2. replace-iam-instance-profile-association コマンドで置換元のインスタンスプロファイルの関連 ID と置換先のインスタンスプロファイルの ARN 名またはプロファイル名を指定して、IAM インスタンスプロファイルを置き換えます。

```
aws ec2 replace-iam-instance-profile-association \
    --association-id iip-assoc-0044d817db6c0a4ba \
    --iam-instance-profile Name="TestRole-2"
```

レスポンスの例

```
{
    "IamInstanceProfileAssociation": {
        "InstanceId": "i-087711ddaf98f9489",
        "State": "associating",
        "AssociationId": "iip-assoc-09654be48e33b91e0",
        "IamInstanceProfile": {
            "Id": "AIPAJCJEDKX7QYHWYK7GS",
            "Arn": "arn:aws:iam::123456789012:instance-profile/TestRole-2"
        }
    }
}
```

または、以下の Tools for Windows PowerShell コマンドを使用します。

- Get-EC2IamInstanceProfileAssociation
- Set-EC2IamInstanceProfileAssociation

IAM ロールのデタッチ

実行中または停止中のインスタンスから IAM ロールをデタッチできます。

Console

インスタンスから IAM ロールをデタッチするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択します。
3. インスタンスを選択し、[アクション]、[セキュリティ]、[IAM ロールの変更] の順に選択します。
4. [AM ロール] で、[IAM ロールがありません] を選択します。[Save] を選択します。
5. 確認ダイアログボックスで、Detachと入力し、[デタッチ] を選択します。

Command line

インスタンスから IAM ロールをデタッチするには (AWS CLI)

1. 必要に応じて、describe-iam-instance-profile-associations で IAM インスタンスプロファイルの関連付けを記述し、デタッチする IAM インスタンスプロファイルの関連 ID を取得します。

```
aws ec2 describe-iam-instance-profile-associations
```

レスポンスの例

```
{
    "IamInstanceProfileAssociations": [
        {
            "InstanceId": "i-088ce778fbfeb4361",
            "State": "associated",
            "AssociationId": "iip-assoc-0044d817db6c0a4ba",
            "IamInstanceProfile": {
                "Id": "AIPAJEDNCAA64SSD265D6",
                "Arn": "arn:aws:iam::123456789012:instance-profile/TestRole-2"
            }
        }
    ]
}
```

2. disassociate-iam-instance-profile コマンドで関連 ID を使用して IAM インスタンスプロファイルをデタッチします。

```
aws ec2 disassociate-iam-instance-profile --association-id iip-
assoc-0044d817db6c0a4ba
```

レスポンスの例

```
{
    "IamInstanceProfileAssociation": {
        "InstanceId": "i-087711ddaf98f9489",
        "State": "disassociating",
        "AssociationId": "iip-assoc-0044d817db6c0a4ba",
        "IamInstanceProfile": {
            "Id": "AIPAJEDNCAA64SSD265D6",
```

```
            "Arn": "arn:aws:iam::123456789012:instance-profile/TestRole-2"
        }
    }
}
```

または、以下の Tools for Windows PowerShell コマンドを使用します。

- Get-EC2IamInstanceProfileAssociation
- Unregister-EC2IamInstanceProfile

アクセスアクティビティに基づいて IAM ロールのポリシーを生成する

アプリケーションの IAM ロールを最初に作成するときに、必要な範囲を超えたアクセス権限を付与することがあります。本番環境でアプリケーションを起動する前に、IAM ロールのアクセスアクティビティに基づく IAM ポリシーを生成できます。IAM Access Analyzer は AWS CloudTrail ログを確認し、指定した日付範囲内のロールが使用したアクセス許可を含むポリシーテンプレートを生成します。テンプレートを使用して、きめ細かなアクセス権限で管理ポリシーを作成し、それを IAM ロールにアタッチできます。これにより、特定のユースケースでロールが AWS リソースとインタラクションするために必要なアクセス権限のみを付与します。これは、最小アクセス権限の付与のベストプラクティスに準拠するのに役立ちます。詳細については、「IAM ユーザーガイド」の「アクセスアクティビティに基づいてポリシーを生成する」を参照してください。

Linux インスタンス用のインバウンドトラフィックの承認

セキュリティグループを使用すると、どのトラフィックがインスタンスに到達できるかなど、インスタンスへのトラフィックを制御できます。例えば、ホームネットワークからのコンピュータのみが、SSH/ を使用してインスタンスにアクセスするように許可できます。インスタンスがウェブサーバーの場合、すべての IP アドレスが HTTP または HTTPS を使用してインスタンスにアクセスできるようにすることで、外部ユーザーはウェブサーバーのコンテンツを閲覧できるようなります。

デフォルトのセキュリティグループと新しく作成されたセキュリティグループには、インターネットからインスタンスにアクセスできないデフォルトのルールが含まれます。詳細については、デフォルトのセキュリティグループおよびCustom security groupsを参照してください。インスタンスへのネットワークアクセスを有効にするには、インスタンスへのインバウンドトラフィックを許可する必要があります。受信トラフィック用のポートを開くには、起動時にインスタンスに関連付けたセキュリティグループにルールを追加します。

インスタンスに接続するには、コンピュータのパブリック IPv4 アドレスからの SSH/ トラフィックを承認するルールをセットアップする必要があります。追加の IP アドレス範囲からの SSH/ トラフィックを許可するには、承認する必要がある各範囲に別のルールを追加します。

IPv6 の VPC を有効にして IPv6 アドレスを使用してインスタンスを起動している場合は、パブリック IPv4 アドレスではなくインスタンスの IPv6 アドレスを使用してインスタンスに接続できます。ローカルコンピュータに IPv6 アドレスがあり、IPv6 を使用するように設定されている必要があります。

Windows インスタンスへのネットワークアクセスを有効にする必要がある場合は、Windows インスタンス用 Amazon EC2 ユーザーガイドのWindows インスタンス用の受信トラフィックの承認を参照してください。

開始する前に

インスタンスへのアクセスの要求元 (例: ローカルコンピュータのパブリック IPv4 アドレスなど、信頼する単一のホストや特定のネットワーク) を判断します。Amazon EC2 コンソールのセキュリティグループエディタは、ローカルコンピュータのパブリック IPv4 アドレスを自動的に検出できます。別の方法として、インターネットブラウザで検索文字列として私の「 IP アドレスは何ですか?」を使用するか、次のサービス: Check IP を使用することもできます。ISP 経由で、またはファイアウォールの内側から静的な IP アドレスなしで接続している場合は、クライアントコンピュータで使用されている IP アドレスの範囲を見つける必要があります。

> **Warning**
>
> `0.0.0.0/0` を使用すると、すべての IPv4 アドレスが SSH/ を使用して、インスタンスにアクセスすることを許可されます。`::/0` を使用すると、すべての IPv6 アドレスからインスタンスにアクセスできるようになります。特定の IP アドレスまたは特定のアドレス範囲にのみ、インスタンスへのアクセスを限定してください。

EC2 Instance Connect を使用してインスタンスへの SSH アクセスをサポートするかどうかを決定します。EC2 Instance Connect を使用しない場合は、アンインストールするか、IAM ポリシーで次のアクションを拒否することを検討してください。`ec2-instance-connect:SendSSHPublicKey`。詳細については、EC2 Instance Connect のアンインストールおよびEC2 Instance Connect の IAM アクセス権限を設定するを参照してください。

Linux インスタンスに対するインバウンド SSH トラフィックのルールを追加する

セキュリティグループは、関連付けられたインスタンスのファイアウォールとして動作し、インバウンドトラフィックとアウトバウンドトラフィックの両方をインスタンスレベルでコントロールします。SSH/ を使用して IP アドレスから Linux/ インスタンスへの接続を有効にするためのルールをセキュリティグループに追加します。

IPv4 でインバウンド SSH トラフィック用のルールをセキュリティグループに追加するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 上部のナビゲーションバーで、セキュリティグループのリージョンを選択します。セキュリティグループはリージョンに固有であるため、インスタンスを作成したリージョンと同じリージョンを選択する必要があります。
3. ナビゲーションペインで、[インスタンス] を選択します。
4. インスタンスを選択し、画面の下半分で [セキュリティ] タブを選択します。[セキュリティグループ] には、インスタンスに関連付けられているセキュリティグループが一覧表示されます。[インバウンドルール] には、インスタンスに有効なインバウンドルールのリストが表示されます。
5. 新しいルールを追加するセキュリティグループについて、セキュリティグループ ID リンクを選択してセキュリティグループを開きます。
6. [インバウンドルール] タブで、[インバウンドルールの編集] を選択します。
7. [インバウンドルールの編集] ページで、次の操作を行います。

   a. [Add rule] を選択します。

   b. [Type] で [SSH] を選択します。

   c. [ソース] で [マイ IP] を選択すると、ローカルコンピュータのパブリック IPv4 アドレスが自動的にフィールドに入力されます。

      別の方法として、[ソース] で [カスタム] を選択してコンピュータまたはネットワークのパブリック IPv4 アドレスを CIDR 表記で入力することもできます。例えば、IPv4 アドレスが `203.0.113.25` である場合、この単一の IPv4 アドレスを CIDR 表記で示すには `203.0.113.25/32` と入力します。会社が特定の範囲からアドレスを割り当てている場合、範囲全体 (`203.0.113.0/24` など) を入力します。

      IP アドレスを見つける方法については、開始する前にを参照してください。

   d. [Save Rules (ルールの保存)] を選択します。

IPv6 アドレスを持つインスタンスを起動して、その IPv6 アドレスを使用してインスタンスに接続する場合は、SSH/ でインバウンド IPv6 トラフィックを許可するルールを追加する必要があります。

IPv6 でインバウンド SSH トラフィック用のルールをセキュリティグループに追加するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 上部のナビゲーションバーで、セキュリティグループのリージョンを選択します。セキュリティグループはリージョンに固有であるため、インスタンスを作成したリージョンと同じリージョンを選択する必要があります。
3. ナビゲーションペインで、[インスタンス] を選択します。
4. インスタンスを選択し、画面の下半分で [セキュリティ] タブを選択します。[セキュリティグループ] には、インスタンスに関連付けられているセキュリティグループが一覧表示されます。[インバウンドルール] には、インスタンスに有効なインバウンドルールのリストが表示されます。
5. 新しいルールを追加するセキュリティグループについて、セキュリティグループ ID リンクを選択してセキュリティグループを開きます。
6. [インバウンドルール] タブで、[インバウンドルールの編集] を選択します。
7. [インバウンドルールの編集] ページで、次の操作を行います。

    a. [Add rule] を選択します。

    b. [Type] で [SSH] を選択します。

    c. [ソース] で [カスタム] を選択し、コンピュータの IPv6 アドレスを CIDR 表記で入力します。例えば、IPv6 アドレスが `2001:db8:1234:1a00:9691:9503:25ad:1761` である場合、この単一の IP アドレスを CIDR 表記で示すには `2001:db8:1234:1a00:9691:9503:25ad:1761/128` と入力します。会社が特定の範囲からアドレスを割り当てている場合、範囲全体 (`2001:db8:1234:1a00::/64` など) を入力します。

    d. [Save Rules (ルールの保存)] を選択します。

> **Note**
>
> 次のコマンドが、インスタンスではなく、ローカルシステムで実行されていることを確認してください。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

コマンドラインを使用してセキュリティグループにルールを追加するには

1. 以下のいずれかのコマンドを使用してインスタンスに関連付けられるセキュリティグループを見つける:

   - describe-instance-attribute (AWS CLI)

     ```
     aws ec2 describe-instance-attribute --region region --instance-id instance_id --
     attribute groupSet
     ```

   - Get-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

     ```
     PS C:\> (Get-EC2InstanceAttribute -Region region -InstanceId instance_id -
     Attribute groupSet).Groups
     ```

   どちらのコマンドも、次のステップで使用できるセキュリティ グループ ID を返します。

2. 以下のいずれかのコマンドを使用してセキュリティグループにルールを追加します。

   - authorize-security-group-ingress (AWS CLI)

     ```
     aws ec2 authorize-security-group-ingress --region region --group-
     id security_group_id --protocol tcp --port 22 --cidr cidr_ip_range
     ```

   - Grant-EC2SecurityGroupIngress (AWS Tools for Windows PowerShell)

     `Grant-EC2SecurityGroupIngress` コマンドには、`IpPermission` パラメーターが必要です。このパラメーターは、セキュリティグループのルールに使用するプロトコル、ポート範囲、IP アドレス範囲を定義します。次のコマンドでは、`IpPermission` パラメーターが作成されます。

     ```
     PS C:\> $ip1 = @{ IpProtocol="tcp"; FromPort="22"; ToPort="22";
      IpRanges="cidr_ip_range" }
     ```

     ```
     PS C:\> Grant-EC2SecurityGroupIngress -Region region -GroupId security_group_id -
     IpPermission @($ip1)
     ```

インスタンスへのセキュリティグループの割り当て

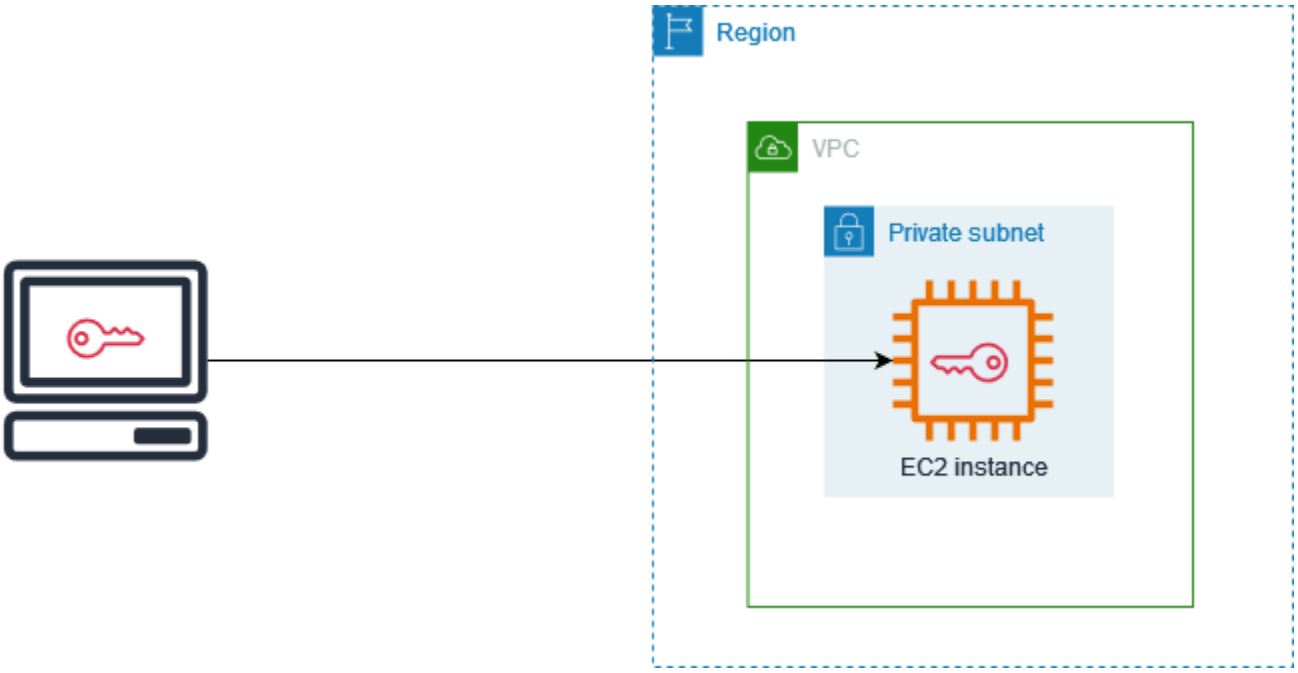
インスタンスを起動する際に、インスタンスにセキュリティグループを割り当てることができます。ルールを追加または削除すると、それらの変更は、そのセキュリティグループを割り当てたすべてのインスタンスに自動的に適用されます。

インスタンスを起動した後、そのセキュリティグループを変更することができます。詳細については、「the section called “インスタンスのセキュリティグループの変更”」を参照してください。

Amazon EC2 のキーペアと Amazon EC2 インスタンス

キーペアには、プライベートキーと公開キーを含んでおり、 Amazon EC2 インスタンスへの接続時の身分証明に使用する、セキュリティ認証情報のセットを構成しています。Linux インスタンスの場合、プライベートキーを使用すると、インスタンスに安全に SSH 接続できます。Windows インスタンスの場合、管理者パスワードを復号化するにはプライベートキーが必要です。これを使用してインスタンスに接続します。

次の図に示すように、パブリックキーは、Amazon EC2 によりお客様のインスタンス内に保管されます。またプライベートキーは、お客様自身が保管します。プライベートキーを所有するすべてのユーザーは、キーペアを使用するインスタンスに接続できるため、プライベートキーを安全な場所に保存することが重要です。



インスタンスを起動するとき、キーペアを指定できます。SSH/ を使用してインスタンスに接続する予定の場合は、キーペアを指定する必要があります。セキュリティの管理方法に応じて、すべての

インスタンスに同じ key pair を指定することも、異なるキーペアを指定することもできます。インスタンスを初めて起動する際に、お客様が初期に指定したパブリックキーが Linux インスタンスの `~/.ssh/authorized_keys` 内のエントリに配置されます。SSH を使用して Linux インスタンスに接続する際のログインには、パブリックキーに対応するプライベートキーを指定する必要があります。インスタンスへの接続方法の詳細については、「Linux インスタンスへの接続」を参照してください。

> **Important**
>
> Amazon EC2 ではプライベートキーのコピーが保持されないため、プライベートキーを失った場合、復元することはできません。ただし、プライベートキーを失くしたインスタンスに接続する方法はまだあります。詳細については、「プライベートキーを紛失しました。Linux インスタンスに接続するにはどうすればよいですか?」を参照してください。

キーペアの代わりに、インタラクティブなワンクリックのブラウザベースのシェルまたは AWS Command Line Interface (AWS CLI) を使用してインスタンスに接続するために、AWS Systems Manager Session Manager を使用できます。

コンテンツ

Amazon EC2 インスタンスのキーペアを作成する

Amazon EC2 を使用してキーペアを作成できます。または、サードパーティー製のツールを使用して、キーペアを作成してから、Amazon EC2 にインポートすることもできます。

Amazon EC2 は、Linux インスタンス向けの、ED25519 および 2048-bit SSH-2 RSA キーをサポートしています。Amazon EC2 は、Windows インスタンス向けの、2048-bit SSH-2 RSA キーをサポートしています。Windows インスタンスでは、ED25519 キーはサポートされていません。

キーペアを作成した後に SSH を使用して Linux インスタンスに接続する手順については、「Linux インスタンスへの接続」を参照してください。

コンテンツ

Amazon EC2 を使用してキーペアを作成する

Amazon EC2 を使用してキーペアを作成すると、パブリックキーは Amazon EC2 内に保存されます。プライベートキーは、自分で保存します。

リージョンごとに最大 5,000 のキーペアを作成できます。増加をリクエストするには、サポートケースを作成します。詳細については、「AWS Support ユーザーガイド」の「サポートケースの作成」を参照してください。

Console

Amazon EC2 を使用してキーペアを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Network & Security] で、[Key Pairs] を選択します。
3. [キーペアの作成] を選択します。
4. [Name (名前)] に、キーペアのわかりやすい名前を入力します。Amazon EC2 は、キー名として指定した名前にパブリックキーを関連付けます。キー名には、最大 255 文字の ASCII 文字を含めることができます。先頭または末尾にスペースを含めることはできません。
5. [キーペアタイプ]を使用する場合には、[RSA]または[25519]を選択します。
6. [Private key ファイル形式] に、プライベートキーを保存する形式を選択します。OpenSSH で使用できる形式でプライベートキーを保存するには、[pem] を選択します。プライベートキーを PuTTY で使用できる形式で保存するには、[ppk] を選択します。
7. タグを追加するには、[タグの追加] ページで [タグの追加] をクリックし、タグのキーと値を入力します。各タグについて、これを繰り返します。
8. [キーペアの作成] を選択します。

9. ブラウザによって秘密キーファイルが自動的にダウンロードされます。ベースファイル名は、キーペアの名前として指定した名前で、ファイル名拡張子は選択したファイル形式によって決まります。ダウンロードしたプライベートキーのファイルを安全な場所に保存します。

   > **Important**
   >
   > プライベートキーのファイルを保存できるのは、このタイミングだけです。

10. macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は、自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

    ```
    chmod 400 key-pair-name.pem
    ```

    これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については、「エラー: Unprotected Private Key File (保護されていないプライベートキーファイル)」を参照してください。

AWS CLI

Amazon EC2 を使用してキーペアを作成するには

1. create-key-pair コマンドを使用し、次のようにキーペアを生成してプライベートキーを .pem ファイルに保存します。

   --key-nameを使用する場合、公開キーの名前を指定します。名前には、最大 255 文字の ASCII 文字を含めることができます。

   --key-typeを使用する場合で、rsaと ed25519のいずれかを指定します。--key-typeパラメータを含まない場合は、デフォルトで rsa キーが作成されます。Windows インスタンスでは、ED25519 キーはサポートされていません。

   --key-formatを使用する場合で、pemと ppkのいずれかを指定します。--key-format パラメータを含まない場合は、デフォルトで pem キーが作成されます。

   --query "KeyMaterial"はプライベートキーのマテリアルを出力します。

`--output text > my-key-pair.pem` は、プライベートキーのマテリアルを拡張機能とともにファイルに保存します。拡張子は、.pem または .ppk のいずれかです。プライベートキーには、公開キーの名前とは異なる名前を指定できますが、使いやすくするために、同じ名前を使用してください。

```
aws ec2 create-key-pair \
    --key-name my-key-pair \
    --key-type rsa \
    --key-format pem \
    --query "KeyMaterial" \
    --output text > my-key-pair.pem
```

2. macOS または Linux コンピュータの SSH クライアントを使用して Linux インスタンスに接続する予定がある場合は、自分以外のユーザーが読み込むことができないように、次のコマンドを使用してプライベートキーファイルの許可を設定します。

```
chmod 400 key-pair-name.pem
```

これらのアクセス権限を設定しないと、このキーペアを使用してインスタンスに接続できません。詳細については、「エラー: Unprotected Private Key File (保護されていないプライベートキーファイル)」を参照してください。

PowerShell

Amazon EC2 を使用してキーペアを作成するには

New-EC2KeyPair AWS Tools for Windows PowerShell コマンドを使用し、次のようにキーを生成して .pem または .ppk ファイルに保存します。

-KeyNameを使用する場合、公開キーの名前を指定します。名前には、最大 255 文字の ASCII 文字を含めることができます。

-KeyTypeを使用する場合で、rsaと ed25519のいずれかを指定します。-KeyTypeパラメータを含まない場合は、デフォルトで rsa キーが作成されます。Windows インスタンスでは、ED25519 キーはサポートされていません。

-KeyFormatを使用する場合で、pemと ppkのいずれかを指定します。-KeyFormat パラメータを含まない場合は、デフォルトで pem キーが作成されます。

KeyMaterialはプライベートキーのマテリアルを出力します。

`Out-File -Encoding ascii -FilePath C:\path\my-key-pair.pem` は、プライベートキーのマテリアルを拡張機能とともにファイルに保存します。拡張子は、`.pem` または `.ppk` にすることができます。プライベートキーには、公開キーの名前とは異なる名前を指定できますが、使いやすくするために、同じ名前を使用してください。

```
PS C:\> (New-EC2KeyPair -KeyName "my-key-pair" -KeyType "rsa" -KeyFormat
 "pem").KeyMaterial | Out-File -Encoding ascii -FilePath C:\path\my-key-pair.pem
```

AWS CloudFormation を使用してキーペアを作成する

AWS CloudFormation を使用して新しいキーペアを作成すると、プライベートキーは AWS Systems Manager パラメータストアに保存されます。パラメータ名の形式は次のとおりです。

```
/ec2/keypair/key_pair_id
```

詳細については、「AWS Systems Manager ユーザーガイド」の「AWS Systems Manager パラメータストア」を参照してください。

AWS CloudFormation を使用してキーペアを作成するには

1. テンプレートに AWS::EC2::KeyPair リソースを指定します。

   ```
   Resources:
     NewKeyPair:
       Type: 'AWS::EC2::KeyPair'
       Properties:
         KeyName: new-key-pair
   ```

2. キーペアの ID を取得するには、次のように describe-key-pairs コマンドを使用します。

   ```
   aws ec2 describe-key-pairs --filters Name=key-name,Values=new-key-pair --query
    KeyPairs[*].KeyPairId --output text
   ```

   以下は出力例です。

   ```
   key-05abb699beEXAMPLE
   ```

3. キーのパラメータを取得するために、次のように get-parameter コマンドを使用して、キーマテリアルを `.pem` ファイルに保存します。

```
aws ssm get-parameter --name /ec2/keypair/key-05abb699beEXAMPLE --with-decryption
 --query Parameter.Value --output text > new-key-pair.pem
```

必要な IAM 許可

AWS CloudFormation がユーザーに代わって Parameter Store パラメータを管理できるようにするには、AWS CloudFormation またはユーザーにより引き受けられた IAM ロールは、次の許可を持っている必要があります。

- `ssm:PutParameter` – プライベートキーマテリアル用パラメーターの削除を許可します。
- `ssm:DeleteParameter` - プライベートキーマテリアルを保存したパラメータの削除する許可を付与します。この権限は、キーペアが AWS CloudFormation によってインポートまたは作成されたかに関係なく必要です。

スタックによって作成またはインポートされたキーペアを AWS CloudFormation が削除する場合、AWS CloudFormation がキーペアをインポートする際ではなく、キーペアを作成する際にのみパラメーターを作成する場合でも権限チェックが実行され、パラメータを削除する権限があるかどうかが判断されます。AWS CloudFormation はアカウント内のどのパラメータとも一致しない偽造パラメータ名を使用して必要なアクセス許可をテストします。そのため、`AccessDeniedException` エラーメッセージに偽造されたパラメータ名が表示されることがあります。

サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする

Amazon EC2 を使用してキーペアを作成する代わりに、サードパーティー製のツールで RSA または ED25519 のキーペアを作成してから、パブリックキーを Amazon EC2 にインポートすることもできます。

キーペアの要件

- サポートされているタイプ:RSA および ED25519。Amazon EC2 は DSA キーを受け付けません。

> **Note**
>
> Windows インスタンスでは、ED25519 キーはサポートされていません。

- サポートされる形式:

  - OpenSSH パブリックキー形式 (~/.ssh/authorized_keys 形式) EC2 Instance Connect API の使用中に SSH を使用して接続する場合は、SSH2 形式もサポートされます。
  - SSH プライベートキーファイル形式は PEM または PPK である必要があります
  - (RSA のみ)Base64 でエンコードされた DER 形式
  - SSH パブリックキーファイル形式 [RFC4716] で指定
- サポートされているキーの長さ 1024、2048、および 4096。EC2 Instance Connect API の使用中に SSH を使用して接続する場合は、長さ 2048 および 4096 がサポートされます。

サードパーティーツールを使用してキーペアを作成するには

1. 選択したサードパーティ製のツールでキーペアを生成します。例えば、ssh-keygen (標準 OpenSSH インストールで提供されるツール) を使用しできます。また、Java、Ruby、Python などのさまざまなプログラミング言語では、RSA または ED25519 のキーペアを作成するために使用できる標準ライブラリが提供されています。

   > **Important**
   >
   > プライベートキーは、PEM または PPK 形式である必要があります。例えば、`ssh-keygen -m PEM` を使用して OpenSSH キーを PEM 形式で生成します。

2. ローカルファイルにパブリックキーを保存します。例えば、`~/.ssh/my-key-pair.pub` と指定します。このファイル名の拡張子は重要ではありません。
3. `.pem` または `.ppk` 拡張子を持つローカルファイルにプライベートキーを保存します。例えば、`~/.ssh/my-key-pair.pem`、または `~/.ssh/my-key-pair.ppk`.

   > **Important**
   >
   > プライベートキーファイルを安全な場所に保存します。インスタンスと対応するプライベートキーの起動時には、毎回インスタンスに接続するたびに、パブリックキーの名前を入力する必要があります。

キーペアを作成したら、次のいずれかの方法を使用してパブリックキーを Amazon EC2 にインポートします。

Console

パブリックキーを Amazon EC2 にインポートするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キーペア] を選択します。
3. [Import Key Pair (キーペアのインポート)] を選択します。
4. [Name (名前)] に、パブリックキーのわかりやすい名前を入力します。名前には、最大 255 文字の ASCII 文字を含めることができます。先頭または末尾にスペースを含めることはできません。

   > **Note**
   >
   > EC2 コンソールからインスタンスに接続すると、コンソールはプライベートキーファイルの名前としてこの名前が提示します。

5. [Browse (参照)] を選択してパブリックキーに移動して選択するか、パブリックキーのコンテンツを [Public key contents (パブリックキーのコンテンツ)] フィールドに貼り付けます。
6. [Import Key Pair (キーペアのインポート)] を選択します。
7. インポートしたパブリックキーがキーペアのリストに表示されていることを確認します。

AWS CLI

パブリックキーを Amazon EC2 にインポートするには

import-key-pair AWS CLI コマンドを実行します。

キーペアが正常にインポートされたことを確認するには

describe-key-pairs AWS CLI コマンドを実行します。

PowerShell

パブリックキーを Amazon EC2 にインポートするには

Import-EC2KeyPair AWS Tools for Windows PowerShell コマンドを実行します。

キーペアが正常にインポートされたことを確認するには

Get-EC2KeyPair AWS Tools for Windows PowerShell コマンドを実行します。

キーペアのタグ付け

Amazon EC2 を使用して作成した、または Amazon EC2 にインポートしたキーペアの分類と管理には、カスタムメタデータによるタグ付けが役立ちます。タグの仕組みの詳細については、Amazon EC2 リソースのタグ付けを参照してください。

Console

キーペアのタグを表示、追加、または削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キーペア] を選択します。
3. パブリックキーを選択した上で、[アクション]、[タグを管理] の順にクリックします。
4. [タグの管理] ページには、対象のパブリックキーに割り当てられているすべてのタグが表示されます。
   - タグを追加するには、[Add tag] を選択し、タグのキーと値を入力します。キーごとに最大 50 個のタグを追加できます。詳細については、を参照してくださいタグの制限
   - タグを削除するには、削除するタグの横にある [Remove (削除)] を選択します。
5. [Save] を選択します。

AWS CLI

キーペアのタグを表示するには

describe-tags AWS CLI コマンドを実行します。次の例では、すべてのパブリックキーのタグを詳細表示します。

```
$ aws ec2 describe-tags --filters "Name=resource-type,Values=key-pair"
```

```
{
    "Tags": [
    {
        "Key": "Environment",
        "ResourceId": "key-0123456789EXAMPLE",
        "ResourceType": "key-pair",
        "Value": "Production"
    },
```

```
    {
        "Key": "Environment",
        "ResourceId": "key-9876543210EXAMPLE",
        "ResourceType": "key-pair",
        "Value": "Production"
    }]
}
```

キーペアのタグの説明を表示するには

describe-key-pairs AWS CLI コマンドを実行します。

```
$ aws ec2 describe-key-pairs --key-pair-ids key-0123456789EXAMPLE
```

```
{
    "KeyPairs": [
    {
        "KeyName": "MyKeyPair",
        "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
        "KeyPairId": "key-0123456789EXAMPLE",
        "Tags": [
        {
            "Key": "Environment",
            "Value": "Production"
        }]
    }]
}
```

キーペアをタグ付けするには

create-tags AWS CLI コマンドを実行します。次の例では、パブリックキーに Key=Cost-Center と Value=CC-123 のタグが付けられています。

```
$ aws ec2 create-tags --resources key-0123456789EXAMPLE --tags Key=Cost-
Center,Value=CC-123
```

キーペアからタグを削除するには

delete-tags AWS CLI コマンドを実行します。例については、AWS CLI コマンドリファレンスの例を参照してください。

PowerShell

キーペアのタグを表示するには

Get-EC2Tagコマンドを実行します。

キーペアのタグの説明を表示するには

Get-EC2KeyPairコマンドを実行します。

キーペアをタグ付けするには

New-EC2Tagコマンドを実行します。

キーペアからタグを削除するには

Remove-EC2Tagコマンドを実行します。

キーペアの詳細表示

Amazon EC2 に保存されているキーペアの詳細情報を表示できます。また、パブリックキーマテリアルを取得し、起動時に指定されたパブリックキーを特定することもできます。

トピック

キーペアの詳細表示

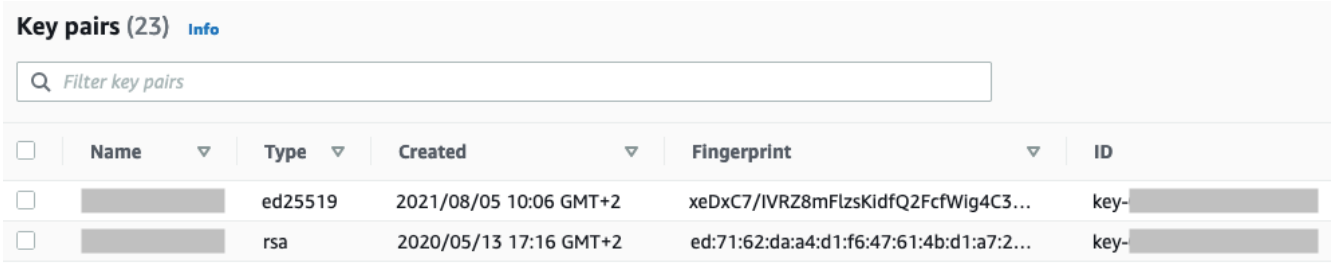
Amazon EC2 に保存されているパブリックキーに関する次の情報を表示できます。パブリックキーの名前、ID、キーの種類、フィンガープリント、パブリックキーのマテリアル、Amazon EC2 によるキーの作成日時 (UTC タイムゾーン) (キーがサードパーティのツールで作成された場合は、そのキーが Amazon EC2 にインポートされた日時)、およびパブリックキーに関連付けられているすべてのタグ。

Amazon EC2 コンソールまたは AWS CLI を使用して、パブリックキーに関する情報を表示できます。

Console

パブリックキーに関する情報を表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側のナビゲータで、[Key Pairs] (キーペア) を選択します。
3. [Key pairs] (キーペア) テーブルで各パブリックキーに関する情報を確認できます。



4. パブリックキーのタグを表示するには、キーの横にあるチェックボックスをオンにし、[Actions] (アクション)、[Manage tags] (タグの管理) の順に選択します。

AWS CLI

パブリックキーの説明を記述するには

describe-key-pairs コマンドを使用して `--key-names` パラメータを指定します。

```
aws ec2 describe-key-pairs --key-names key-pair-name
```

出力例

```
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0123456789example",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
            "KeyName": "key-pair-name",
            "KeyType": "rsa",
            "Tags": [],
            "CreateTime": "2022-04-28T11:37:26.000Z"
        }
    ]
}
```

あるいは、--key-names の代わりに --key-pair-ids パラメーターを指定してパブリックキーを識別することもできます。

```
aws ec2 describe-key-pairs --key-pair-ids key-0123456789example
```

パブリックキーのマテリアルを出力に表示するには、--include-public-key パラメータを指定する必要があります。

```
aws ec2 describe-key-pairs --key-names key-pair-name --include-public-key
```

出力例 — 出力では、PublicKey フィールドにパブリックキーマテリアルが含まれています。

```
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0123456789example",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
            "KeyName": "key-pair-name",
            "KeyType": "rsa",
            "Tags": [],
            "PublicKey": "ssh-ed25519
 AAAAC3NzaC1lZDI1NTE5AAAAIIj7azlDjVHAsSxgcpCRZ3oWnTm0nAFM64y9jd22ioI/ my-key-pair",
            "CreateTime": "2022-04-28T11:37:26.000Z"
        }
    ]
}
```

パブリックキーマテリアルを取得する

さまざまな方法を使用して、パブリックキーマテリアルにアクセスできます。パブリックキーマテリアルは、ローカルコンピュータ上の一致するプライベートキー、またはパブリックキーで起動したインスタンスのインスタンスメタデータまたは authorized_keys ファイルから取得するか、describe-key-pairs AWS CLI コマンドを使用することで取得できます。

次のいずれかの方法を使用して、パブリックキーマテリアルを取得します。

From the private key

プライベートキーからパブリックキーマテリアルを取得するには

ローカルの Linux または macOS コンピュータで、ssh-keygen コマンドを使用して、キーペアのパブリックキーを取得します。プライベートキーをダウンロードしたパスを指定します (.pem ファイル)。

```
ssh-keygen -y -f /path_to_key_pair/my-key-pair.pem
```

コマンドは、次の例に示すように、パブリックキーを返します。

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQClKsfkNkuSevGj3eYhCe53pcjqP3maAhDFcvBS7O6V
hz2ItxCih+PnDSUaw+WNQn/mZphTk/a/gU8jEzoOWbkM4yxyb/wB96xbiFveSFJuOp/d6RJhJOI0iBXr
lsLnBItntckiJ7FbtxJMXLvvwJryDUilBMTjYtwB+QhYXUMOzce5Pjz5/i8SeJtjnV3iAoG/cQk+0FzZ
qaeJAAHco+CY/5WrUBkrHmFJr6HcXkvJdWPkYQS3xqC0+FmUZofz221CBt5IMucxXPkX4rWi+z7wB3Rb
BQoQzd8v7yeb7OzlPnWOyN0qFU0XA246RA8QFYiCNYwI3f05p6KLxEXAMPLE
```

このコマンドが失敗した場合は、次のコマンドを実行して、自分だけがプライベートキーペアファイルを表示できるように、このファイルに対するアクセス許可が変更されていることを確認してください。

```
chmod 400 key-pair-name.pem
```

From the instance metadata

インスタンスメタデータサービスバージョン 2 またはインスタンスメタデータサービスバージョン 1 を使用して、インスタンスメタデータからパブリックキーを取得できます。

> **Note**
>
> インスタンスへの接続に使用するキーペアを変更しても、Amazon EC2 は新しいパブリックキーを表示するようにインスタンスメタデータを更新しません。インスタンスのメタデータには、インスタンスの起動時に指定したキーペアのパブリックキーが引き続き表示されます。

インスタンスメタデータからパブリックキーマテリアルを取得するには

インスタンスから次のいずれかのコマンドを使用します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/public-keys/0/openssh-key
```

出力例

```
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQClKsfkNkuSevGj3eYhCe53pcjqP3maAhDFcvBS7O6V
hz2ItxCih+PnDSUaw+WNQn/mZphTk/a/gU8jEzoOWbkM4yxyb/wB96xbiFveSFJuOp/d6RJhJOI0iBXr
lsLnBItntckiJ7FbtxJMXLvvwJryDUilBMTjYtwB+QhYXUMOzce5Pjz5/i8SeJtjnV3iAoG/cQk+0FzZ
qaeJAAHco+CY/5WrUBkrHmFJr6HcXkvJdWPkYQS3xqC0+FmUZofz221CBt5IMucxXPkX4rWi+z7wB3Rb
BQoQzd8v7yeb7OzlPnWOyN0qFU0XA246RA8QFYiCNYwI3f05p6KLxEXAMPLE key-pair-name
```

インスタンスのメタデータの詳細については、インスタンスメタデータの取得を参照してください。

From the instance

Linux インスタンスを起動するときにキーペアを指定した場合、そのインスタンスの最初の起動時に、パブリックキーの内容がインスタンスの ~/.ssh/authorized_keys 内にエントリとして配置されます。

インスタンスからパブリックキーマテリアルを取得するには

1. インスタンスに接続します。
2. ターミナルウィンドウで、任意のテキストエディタ (authorized_keys や vim など) を使用して nano ファイルを開きます。

   ```
   [ec2-user ~]$ nano ~/.ssh/authorized_keys
   ```

   authorized_keys ファイルが開き、パブリックキーと、その後にキーペアの名前が表示されます。以下に、*key-pair-name* という名前のキーペアのエントリの例を示します。

   ```
   ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQClKsfkNkuSevGj3eYhCe53pcjqP3maAhDFcvBS7O6V
   hz2ItxCih+PnDSUaw+WNQn/mZphTk/a/gU8jEzoOWbkM4yxyb/wB96xbiFveSFJuOp/d6RJhJOI0iBXr
   lsLnBItntckiJ7FbtxJMXLvvwJryDUilBMTjYtwB+QhYXUMOzce5Pjz5/i8SeJtjnV3iAoG/cQk+0FzZ
   ```

```
qaeJAAHco+CY/5WrUBkrHmFJr6HcXkvJdWPkYQS3xqC0+FmUZofz221CBt5IMucxXPkX4rWi+z7wB3Rb
BQoQzd8v7yeb7OzlPnWOyN0qFU0XA246RA8QFYiCNYwI3f05p6KLxEXAMPLE key-pair-name
```

From describe-key-pairs

describe-key-pairsAWS CLI コマンドからパブリックキーマテリアルを取得するには

describe-key-pairs コマンドを使用し、--key-names パラメータを指定してパブリックキーを識別します。パブリックキーのマテリアルを出力に含めるには、--include-public-key パラメータを指定する必要があります。

```
aws ec2 describe-key-pairs --key-names key-pair-name --include-public-key
```

出力例 — 出力では、PublicKey フィールドにパブリックキーマテリアルが含まれています。

```
{
    "KeyPairs": [
        {
            "KeyPairId": "key-0123456789example",
            "KeyFingerprint":
 "1f:51:ae:28:bf:89:e9:d8:1f:25:5d:37:2d:7d:b8:ca:9f:f5:f1:6f",
            "KeyName": "key-pair-name",
            "KeyType": "rsa",
            "Tags": [],
            "PublicKey": "ssh-ed25519
 AAAAC3NzaC1lZDI1NTE5AAAAIIj7azlDjVHAsSxgcpCRZ3oWnTm0nAFM64y9jd22ioI/ my-key-pair",
            "CreateTime": "2022-04-28T11:37:26.000Z"
        }
    ]
}
```

あるいは、--key-names の代わりに --key-pair-ids パラメーターを指定してパブリックキーを識別することもできます。

```
aws ec2 describe-key-pairs --key-pair-ids key-0123456789example --include-public-key
```

起動時に指定されたパブリックキーを特定する

インスタンスを起動する際にパブリックキーを指定した場合、インスタンスによりパブリックキー名が記録されます。

起動時に指定されたパブリックキーを特定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. [詳細] タブの [インスタンスの詳細] にある [起動時に割り当てられたキーペア] フィールドに、インスタンスの起動時に指定したパブリックキーの名前が表示されます。

> **Note**
>
> インスタンスのパブリックキーを変更したり、パブリックキーを追加したりしても、[起動時に割り当てられたキーペア] フィールドの値は変更されません。

キーペアの削除

キーペアは削除できます。これにより、Amazon EC2 に保存されているパブリックキーは削除されます。キーペアを削除しても、対応するプライベートキーは削除されません。

以下の方法でパブリックキーを削除すると、キーペアの作成時またはインポート時に Amazon EC2 に保存されたパブリックキーのみが削除されます。パブリックキーをインスタンスに追加していた場合、パブリックキーを削除しても、インスタンスの起動時またはそれ以降に、インスタンスからパブリックキーが削除されることはありません。ローカルコンピュータのプライベートキーも削除されません。Amazon EC2 から削除したパブリックキーを使用してインスタンスを起動していた場合、プライベートキー (`.pem`) ファイルを保持している限り、引き続きインスタンスに接続できます。

> **Important**
>
> Auto Scaling グループ (Elastic Beanstalk 環境など) を使用している場合、関連する起動テンプレートまたは起動設定に削除するパブリックキーが指定されていないことを確認します。Amazon EC2 Auto Scaling が異常なインスタンスを検出した場合、代替インスタンスを起動します。ただし、パブリックキーが見つからない場合、インスタンスの起動は失敗します。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの起動テンプレートを参照してください。

Console

Amazon EC2 上のパブリックキーを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[キーペア] を選択します。
3. 削除するキーペアを選択し、[Actions] (アクション)、[Delete] (削除) の順に選択します。
4. 確認フィールドで、`Delete` を入力し、[Delete (削除)] を選択します。

AWS CLI

Amazon EC2 上のパブリックキーを削除するには

delete-key-pair AWS CLI コマンドを実行します。

PowerShell

Amazon EC2 上のパブリックキーを削除するには

Remove-EC2KeyPair AWS Tools for Windows PowerShell コマンドを実行します。

インスタンスでパブリックキーを追加または削除する

プライベートキーを紛失した場合、キーペアを使用するインスタンスへのアクセスができなくなります。起動時に指定したキーペアとは異なるキーペアを使用してインスタンスに接続する方法の詳細については、「プライベートキーを紛失しました」を参照してください。

インスタンスを起動するとき、キーペアを指定できます。起動時にキーペアを指定した場合、インスタンスを初めて起動すると、パブリックキーマテリアルが Linux インスタンスの `~/.ssh/authorized_keys` 内のエントリに配置されます。

インスタンスのデフォルトシステムアカウントへのアクセスに使用されるキーペアを変更するには、インスタンスに新しいパブリックキーを追加するか、インスタンスでパブリックキー (既存のパブリックキーを削除して新しいパブリックキーを追加します) を置き換えます。インスタンスからすべてのパブリックキーを削除することもできます。キーペアを追加または置換するには、インスタンスに接続できる必要があります。

次の場合に、キーペアを追加するか、交換できます。

- 例えば、組織のユーザーが、別のキーペアを使用してシステムユーザーにアクセスする必要がある場合は、パブリックキーをインスタンスに追加できます。
- プライベートキー (.pem ファイル) のコピーを持つユーザーがインスタンスに接続するのを防ぐには (例えば、組織を去った場合)、インスタンスのパブリックキーを削除し、新しいものに交換することができます。
- インスタンスから Linux AMI を作成すると、パブリックキーマテリアルがインスタンスから AMI にコピーされます。AMI からインスタンスを起動すると、新しいインスタンスは元のインスタンスからのパブリックキーを含みます。プライベートキーを持つユーザーが新しいインスタンスに接続できないようにするには、AMI を作成する前に、元のインスタンスからパブリックキーを削除します。

次の手順を使用して、デフォルトのユーザー (例: ec2-user) のキーペアを変更します。インスタンスにユーザーを追加する方法については、インスタンスのオペレーティングシステムのドキュメントを参照してください。

キーペアの追加または交換

1. Amazon EC2 コンソールまたはサードパーティー製のツールで、新しいキーペアを作成します。
2. 新しいキーペアからパブリックキーを取得します。詳細については、パブリックキーマテリアルを取得するを参照してください。
3. 既存のプライベートキーを使用して、インスタンスに接続します。
4. 任意のテキストエディタを使用して、インスタンス上にある .ssh/authorized_keys ファイルを開きます。既存のパブリックキー情報の下の新しいキーペアからパブリックキーを貼り付けます。ファイルを保存します。
5. インスタンスから切断し、新しいプライベートキーファイルを使用してインスタンスに接続できることを確認します。
6. (オプション) 既存のキーペアを交換している場合は、インスタンスに接続し、.ssh/authorized_keys ファイルからオリジナルのキーペアのパブリックキー情報を削除します。

> **Important**
>
> Auto Scaling グループを使用している場合、交換するキーペアが起動テンプレートまたは起動設定で指定されていないことを確認します。Amazon EC2 Auto Scaling が異常なインスタンスを検出した場合、代替インスタンスを起動します。ただし、キーペアが見つからない

場合、インスタンスの起動は失敗します。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの起動テンプレートを参照してください。

インスタンスからパブリックキーを削除するには

1. インスタンスに接続します。
2. 任意のテキストエディタを使用して、インスタンス上にある .ssh/authorized_keys ファイルを開きます。パブリックキーの情報を削除し、ファイルを保存します。

> **Warning**
>
> インスタンスからすべてのパブリックキーを削除してインスタンスから切断すると、AMI から別のログイン方法が提供されない限り、インスタンスに再接続できません。

キーペアのフィンガープリントを確認する

キーペアのフィンガープリントを確認するには、Amazon EC2 コンソールの [キーペア] ページに表示される、または describe-key-pair コマンドによって返されるフィンガープリントを、ローカルコンピュータのプライベートキーを使用して生成したフィンガープリントと比較します。これらのフィンガープリントは一致するはずです。

Amazon EC2 がフィンガープリントを計算するとき、Amazon EC2 はフィンガープリントに = 文字でパディングを追加することがあります。ssh-keygen などのその他のツールでは、このパディングを省略することがあります。

キーペアのフィンガープリントではなく EC2 インスタンスのフィンガープリントを検証しようとしている場合は、「インスタンスフィンガープリントの取得」を参照してください。

フィンガープリントの計算方法

Amazon EC2 は、RSA および ED25519 キーペアのフィンガープリントを計算するために、さまざまなハッシュ関数を使用します。さらに、RSA キーペアの場合、Amazon EC2 は、キーペアが Amazon EC2 によって作成されたか、Amazon EC2 にインポートされたかに応じて、異なるハッシュ関数を使用して異なる方法でフィンガープリントを計算します。

次の表は、Amazon EC2 で作成され、Amazon EC2 にインポートされた RSA および ED25519 キーペアの、フィンガープリントの計算に使用されるハッシュ関数の一覧です。

フィンガープリントの計算に使用されるハッシュ関数

キーペアのソース	RSA キーペア	ED25519 キーペア
Amazon EC2 によって作成された	SHA-1	SHA-256
Amazon EC2 にインポートされた	MD5[1]	SHA-256

[1] パブリック RSA キーを Amazon EC2 にインポートした場合、フィンガープリントは、MD5 ハッシュ関数を使用して計算されます。これは、サードパーティーのツールを使用する、Amazon EC2 を使用して作成した既存のプライベートキーから新しいパブリックキーを生成するなど、キーペアの作成方法に関係なく当てはまります。

異なるリージョンで同じキーペアを使用する場合

同じキーペアを使用して、異なる AWS リージョン にあるインスタンスに接続する場合は、使用するすべてのリージョンにパブリックキーをインポートする必要があります。Amazon EC2 を使用してキーペアを作成する場合、パブリックキーを他のリージョンにインポートできるよう、パブリックキーマテリアルを取得する することができます。

> **Note**
>
> Amazon EC2 を使用して RSA キーペアを作成した場合、Amazon EC2 プライベートキーからパブリックキーを生成すると、インポートされたパブリックキーのフィンガープリントは、元のパブリックキーとは異なるものになります。これは、Amazon EC2 を使用して作成された元の RSA キーのフィンガープリントは SHA-1 ハッシュ関数を使用して計算されるのに対し、インポートされた RSA キーのフィンガープリントは MD5 ハッシュ関数を使用して計算されるためです。

ED25519 キーペアでは、同じ SHA-256 ハッシュ関数を使用してフィンガープリントが計算されるため、Amazon EC2 で作成されたか、Amazon EC2 にインポートされたかにかかわらず、フィンガープリントは同一になります。

プライベートキーからフィンガープリントを生成する

ローカルマシンのプライベートキーからフィンガープリントを生成するには、次のいずれかのコマンドを使用します。

Windows ローカルマシンを使用している場合、Windows Subsystem for Linux (WSL) を使用して、次のコマンドを実行できます。Windows 10 インストールガイドの手順を使用して、WSL と Linux ディストリビューションをインストールします。手順の例では、Linux の Ubuntu ディストリビューションをインストールしますが、任意のディストリビューションをインストールできます。コンピュータを再起動して変更を有効にすることが求められます。

- Amazon EC2 を使用してキーペアを作成した場合

  次の例のように、OpenSSL ツールを使用してフィンガープリントを生成します。

  RSA キーペアの場合:

  ```
  openssl pkcs8 -in path_to_private_key -inform PEM -outform DER -topk8 -nocrypt |
   openssl sha1 -c
  ```

  ED25519 キーペアの場合 :

  ```
  ssh-keygen -l -f path_to_private_key
  ```

- (RSA キーペアのみ) パブリックキーを Amazon EC2 にインポートした場合

  キーペアの作成方法 (サードパーティーのツールを使用する、Amazon EC2 を使用して作成した既存のプライベートキーから新しいパブリックキーを生成するなど) に関係なく、この手順に従うことができます。

  次の例のように、OpenSSL ツールを使用してフィンガープリントを生成します。

  ```
  openssl rsa -in path_to_private_key -pubout -outform DER | openssl md5 -c
  ```

- OpenSSH 7.8 以降を使用して OpenSSH キーペアを作成し、パブリックキーを Amazon EC2 にインポートした場合

  次の例のように、ssh-keygen を使用してフィンガープリントを生成します。

  RSA キーペアの場合:

```
ssh-keygen -ef path_to_private_key -m PEM | openssl rsa -RSAPublicKey_in -outform DER
 | openssl md5 -c
```

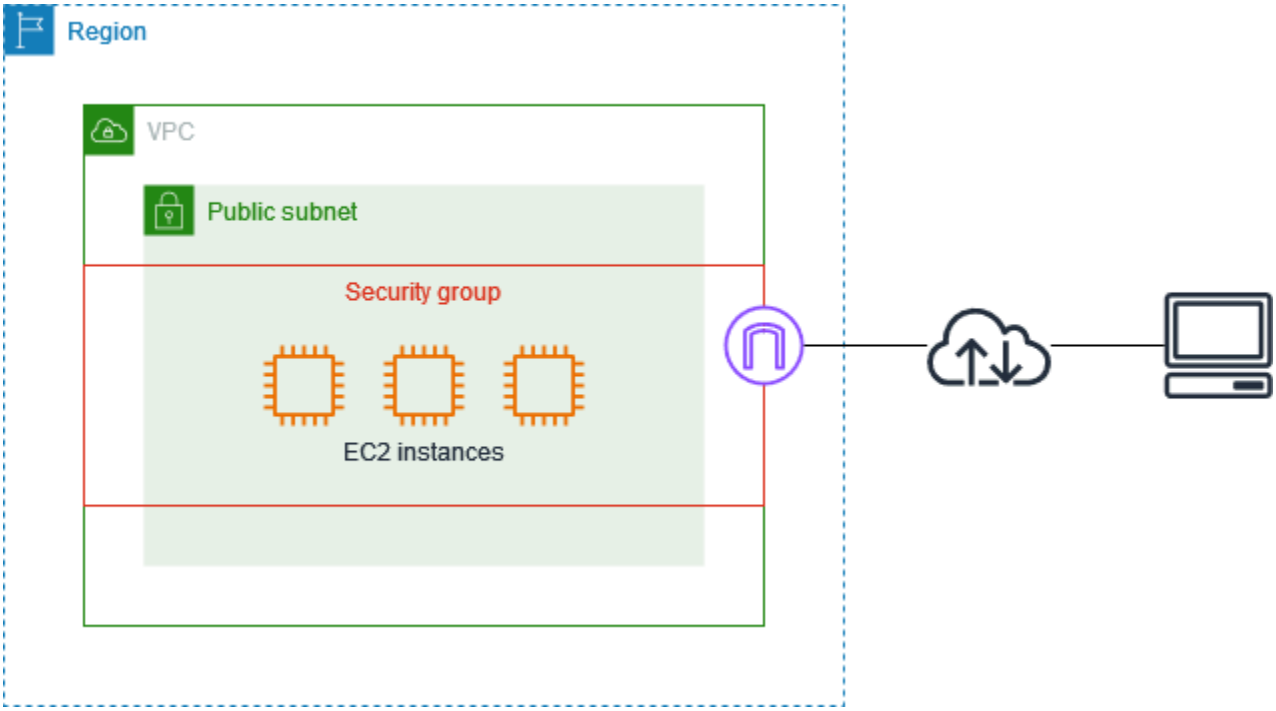
ED25519 キーペアの場合：

```
ssh-keygen -l -f path_to_private_key
```

Linux インスタンス用の Amazon EC2 Amazon セキュリティグループ

セキュリティグループは、EC2 インスタンスの仮想ファイアウォールとして機能し、受信トラフィックと送信トラフィックを制御します。インバウンドルールはインスタンスへの受信トラフィックを制御し、アウトバウンドルールはインスタンスからの送信トラフィックをコントロールします。インスタンスの起動時に 1 つ以上のセキュリティグループを指定できます。セキュリティグループを指定しない場合、Amazon EC2 はデフォルトの VPC のセキュリティグループを使用します。各セキュリティグループに対してルールを追加し、関連付けられたインスタンスに対するトラフィックを許可できます。セキュリティグループのルールはいつでも変更することができます。新規または変更したルールは、セキュリティグループに関連付けられたすべてのインスタンスに自動的に適用されます。トラフィックがインスタンスに到達することを許可するかどうかを Amazon EC2 が判断するとき、インスタンスに関連付けられているすべてのセキュリティグループのすべてのルールを評価します。

次の図は、サブネット、インターネットゲートウェイ、セキュリティグループを備えた VPC を示しています。サブネットには EC2 インスタンスが含まれています。セキュリティグループは、インスタンスに割り当てられます。インスタンスに到達するトラフィックは、セキュリティグループのルールで許可されているトラフィックだけです。例えば、ネットワークからの SSH トラフィックを許可するルールがセキュリティグループに含まれている場合は、お使いのコンピュータからインスタンスに SSH を使用して接続できます。割り当てられたリソースからのすべてのトラフィックを許可するルールがセキュリティグループに含まれている場合、各インスタンスは他のインスタンスから送信されたすべてのトラフィックを受信できます。



インスタンスを起動した後、そのセキュリティグループを変更することができます。セキュリティグループはネットワークインターフェイスに関連付けられます。インスタンスのセキュリティグループの変更は、プライマリネットワークインターフェイス (eth0) に関連付けられるセキュリティグループを変更することになります。詳細については、「インスタンスのセキュリティグループの変更」を参照してください。あらゆるネットワークインターフェイスに関連付けられているセキュリティグループも変更できます。詳細については、ネットワークインターフェイス属性の変更を参照してください。

セキュリティは、AWS とお客様の間の共有責任です。詳細については、セキュリティとコンプライアンスの目標を満たすように Amazon EC2 を設定し、Amazon EC2 リソースの保護に役立つ他の サービスの使用方法を学びます。を参照してください。AWS は、インスタンスをセキュリティで保護するためのツールの 1 つとしてセキュリティグループを提供しています。このセキュリティグループをセキュリティニーズに合わせて設定する必要があります。セキュリティグループでは十分に満たせない要件がある場合は、セキュリティグループの使用に加えて、どのインスタンスでも独自のファイアウォールを使用できます。

Windows インスタンスへのトラフィックを許可するには、Windows インスタンス用 Amazon EC2 セキュリティグループのWindows インスタンス用 Amazon EC2 ユーザーガイドを参照してください。

セキュリティグループは追加料金なしで使用できます。

コンテンツ

セキュリティグループのルール

セキュリティグループルールは、セキュリティグループに関連付けられたインスタンスに到達することを許可するインバウンドトラフィックを制御します。また、このルールによって、インスタンスから送信されるアウトバウンドトラフィックも制御されます。

セキュリティグループのルールの特徴を次に示します。

- デフォルトでは、セキュリティグループには、すべてのアウトバウンドトラフィックを許可するアウトバウンドルールが含まれています。これらのルールは削除できます。デフォルトで、Amazon EC2 はポート 25 のトラフィックをブロックすることに注意してください。詳細については、ポート 25 を使用した E メール送信の制限を参照してください。
- セキュリティグループのルールは常にパーミッシブです。アクセスを拒否するルールを作成することはできません。
- セキュリティグループルールを使用すると、プロトコルとポート番号に基づいてトラフィックをフィルタリングできます。
- セキュリティグループはステートフルです。インスタンスからリクエストを送信すると、そのリクエストに対するレスポンストラフィックは、セキュリティグループのインバウンドルールにかかわらず、流入できます。つまり、VPC セキュリティグループの場合、アウトバウンドルールにかかわらず、許可されたインバウンドトラフィックは流れることができます。詳細については、セキュリティグループの接続の追跡を参照してください。
- ルールの追加と削除は随時行うことができます。変更は、セキュリティグループに関連付けられたインスタンスに自動的に適用されます。

  一部のルール変更の影響は、トラフィックの追跡方法によって異なる場合があります。詳細については、セキュリティグループの接続の追跡を参照してください。
- 複数のセキュリティグループをインスタンスに関連付けると、各セキュリティグループのルールが効率的に集約され、1 つのルールセットが作成されます。Amazon EC2 はこのルールセットを使用して、アクセスを許可するかを判断します。

セキュリティグループは、1 つのインスタンスに複数割り当てることができます。そのため、1 つのインスタンスに数百のルールが適用される場合があります。結果として、インスタンスにアクセスするときに問題が発生する可能性があります。そのため、ルールは可能な限り要約することをお勧めします。

> **Note**
>
> セキュリティグループは、「VPC +2 IP アドレス」(「Amazon Route 53 デベロッパーガイド」の「Amazon Route 53 Resolver とは」を参照)、または「AmazonProvidedDNS」(「Amazon Virtual Private Cloud ユーザーガイド」の「DHCP オプションセットの使用」を参照) と呼ばれることもある Route 53 Resolver から送受信される DNS リクエストをブロックできません。Route 53 Resolver で DNS リクエストをフィルタリングしたい場合は、Route 53 Resolver DNS ファイアウォールを有効にできます (「Amazon Route 53 デベロッパーガイド」の「Route 53 Resolver DNS ファイアウォール」を参照)。

ルールごとに、以下の点について指定します。

- 名前: セキュリティグループの名前 (「my-security-group」など)。

  名前の最大長は 255 文字です。使用できる文字は、a ~ z、A ~ Z、0 ~ 9、スペース、._-:/()#,@[]+=;{}!$* です。名前の末尾にスペースが含まれている場合は、名前を保存するときにスペースが切り捨てられます。例えば、名前に「セキュリティグループのテスト」と入力すると、「セキュリティグループのテスト」として保存されます。
- プロトコル: 許可するプロトコル。最も一般的なプロトコルは、6 (TCP)、17 (UDP)、1 (ICMP) です。
- ポートの範囲: TCP、UDP、カスタムプロトコルの場合、許可するポートの範囲。1 つのポート番号 (22 など)、または一定範囲のポート番号 (`7000-8000` など) を指定できます。
- ICMP タイプおよびコード: ICMP の場合、ICMP タイプおよびコードです。例えば、ICMP エコー要求にはタイプ 8、ICMPv6 エコー要求にはタイプ 128 を使用します。
- Source or destination (送信元または送信先): 許可するトラフィックの送信元 (インバウンドルール) または送信先 (アウトバウンドルール)。次のいずれかを指定します。
  - 単一の IPv4 アドレス。`/32` プレフィクス長を使用する必要があります。例えば、`203.0.113.1/32` と指定します。

- 単一の IPv6 アドレス。/128 プレフィクス長を使用する必要があります。例えば、2001:db8:1234:1a00::123/128 と指定します。
- CIDR ブロック表記の IPv4 アドレスの範囲。例えば、203.0.113.0/24 と指定します。
- CIDR ブロック表記の IPv6 アドレスの範囲。例えば、2001:db8:1234:1a00::/64 と指定します。
- プレフィクスリストの ID。例えば、pl-1234abc1234abc123 と指定します。詳細については、「Amazon VPC ユーザーガイド」の「プレフィクスリスト」を参照してください。
- セキュリティグループの ID (ここでは、指定されたセキュリティグループと呼ばれます)。例えば、現在のセキュリティグループ、同じ VPC のセキュリティグループ、またはピア接続された VPC のセキュリティグループなどがあります。これにより、指定されたセキュリティグループに関連付けられたリソースのプライベート IP アドレスに基づくトラフィックが許可されます。その際、指定されたセキュリティグループから現在のセキュリティグループにルールが追加されることはありません。

- (オプション) 説明: 後で分かりやすいように、このルールの説明を追加できます。説明の長さは最大 255 文字とすることができます。使用できる文字は、a ~ z、A ~ Z、0 ~ 9、スペース、._-:/()#,@[]+=;{}!$* です。

セキュリティグループルールを作成する際、AWS により、一意の ID がそのルールに割り当てられます。このルールの ID は、API または CLI を使用してルールを変更または削除する際に使用します。

ルールに送信元または送信先としてセキュリティグループを指定する場合、ルールはセキュリティグループに関連付けられているすべてのインスタンスに影響します。着信トラフィックは、ソースセキュリティグループに関連付けられたインスタンスのプライベート IP アドレスに基づいて許可されます (パブリック IP アドレスまたは Elastic IP アドレスは考慮されません)。IP アドレスについては、Amazon EC2 インスタンスの IP アドレス指定を参照してください。セキュリティグループルールが、同じ VPC またはピア VPC 内の削除されたセキュリティグループを参照している場合、または VPC ピアリング接続が削除されたピア VPC のセキュリティグループを参照している場合は、古いルールとしてマークされます。詳細については、「Amazon VPC Peering ガイド」の「古いセキュリティグループルールの操作」を参照してください。

特定のポートに複数のルールがある場合、Amazon EC2 が最も許容度の大きいルールを適用します。例えば、IP アドレス 203.0.113.1 からの TCP ポート 22 (SSH) に対するアクセスを許可するルールがあり、全員からの TCP ポート 22 に対するアクセスを許可する別のルールがある場合、全員が TCP ポート 22 にアクセスできます。

ルールを追加、更新、または削除すると、セキュリティグループに関連付けられたすべてのインスタンスにこの変更が自動的に適用されます。

セキュリティグループの接続の追跡

セキュリティグループは、接続追跡を使用してインスタンスを出入りするトラフィックに関する情報を追跡します。ルールはトラフィックの接続の状態に基づいて適用され、トラフィックを許可するか拒否するかが判断されます。このアプローチでは、セキュリティグループはステートフルです。これは、セキュリティグループのアウトバウンドルールにかかわらず、インバウンドトラフィックに対するレスポンスがインスタンスから送信されることを許可することを意味します。逆も同じです。

例えば、自宅のコンピュータからインスタンスに対し netcat や同様の ICMP コマンドを開始する場合を考えます。この時、インバウンドセキュリティグループは、ICMP トラフィックを許可しているとします。接続に関する情報 (ポート情報を含む) が追跡されます。コマンドに対するインスタンスからのレスポンストラフィックは、新しいリクエストではなく確立済みの接続として追跡されます。また、セキュリティグループのアウトバウンドルールが、アウトバウンドの ICMP トラフィックを制限している場合でも、このトラフィックはインスタンスから外部に出力されることが許されます。

TCP、UDP、または ICMP 以外のプロトコルの場合は、IP アドレスとプロトコル番号のみが追跡されます。インスタンスが別のホストにトラフィックを送信し、そのホストが 600 秒以内に同じタイプのトラフィックをインスタンスに送信した場合、インスタンスのセキュリティグループはインバウンドセキュリティグループルールに関係なく、そのトラフィックを受け入れます。そのトラフィックが元のトラフィックのレスポンストラフィックとみなされるからです。

セキュリティグループルールを変更しても、そのルールで追跡された接続がすぐに中断されることはありません。セキュリティグループは、既存の接続がタイムアウトするまで引き続きパケットを許可します。トラフィックをすぐに中断するか、追跡状態に関係なくすべてのトラフィックをファイアウォールルールの対象にするには、サブネットにネットワーク ACL を使用します。ネットワーク ACL はステートレスであるため、レスポンスのトラフィックを自動的には許可しません。いずれかの方向のトラフィックをブロックするネットワーク ACL を追加すると、既存の接続が切断されます。詳細については、Amazon VPC ユーザーガイドのネットワーク ACLを参照してください。

> **Note**
>
> セキュリティグループは、「VPC +2 IP アドレス」(「Amazon Route 53 デベロッパーガイド」の「Amazon Route 53 Resolver とは」を参照)、または「AmazonProvidedDNS」(「Amazon Virtual Private Cloud ユーザーガイド」の「DHCP オプションセットの使用」を参照) と呼ばれることもある Route 53 Resolver から送受信される DNS トラフィックに影響を及ぼすことはありません。Route 53 Resolver で DNS リクエストをフィルタリングしたい

場合は、Route 53 Resolver DNS ファイアウォールを有効にできます (「Amazon Route 53 デベロッパーガイド」の「Route 53 Resolver DNS ファイアウォール」を参照)。

追跡されていない接続

すべてのトラフィックフローが追跡されるわけではありません。セキュリティグループのルールがすべてのトラフィック (0.0.0.0/0 または ::/0) の TCP または UDP フローを許可していて、片方の方向には任意のポート (0 ~ 65535) のすべての応答トラフィック (0.0.0.0/0 または ::/0) を許可するルールがある場合、そのトラフィックフローは自動的に追跡される接続の一部でない限り追跡されません。追跡されていないフローのレスポンストラフィックは、追跡情報ではなく、レスポンストラフィックを許可するインバウンドルールまたはアウトバウンドルールに基づいて許可されます。

追跡されていないトラフィックフローは、そのフローを有効にするルールが削除または変更されるとすぐに中断されます。例えば、オープン (0.0.0.0/0) のアウトバウンドルールがあり、インスタンスへのすべて(0.0.0.0/0) のインバウンドの SSH (TCP ポート 22) トラフィックを許可するルールを削除した場合 (または接続を許可しないように変更した場合)、インスタンスへの既存の SSH 接続はすぐに中断されます。接続はそれまで追跡されていないため、この変更によって接続が切断されます。一方、最初に細かく SSH 接続を許可する (つまり、接続を追跡する) インバウンドルールがある場合、現在の SSH クライアントのアドレスからの新しい接続を許可しないようにルールを変更しても、既存の SSH 接続は追跡対象であるため中断されません。

自動的に追跡される接続

本来ならセキュリティグループの設定で追跡を指定する必要がある場合でも、以下の手段で確立された接続は自動的に追跡されます。有効なリプライパスが複数存在する可能性があるため、対称ルーティングを確保するには、こうした接続を追跡する必要があります。

- Egress-Only インターネットゲートウェイ
- Gateway Load Balancer
- Global Accelerator アクセラレーター
- NAT ゲートウェイ
- Network Firewall ファイアウォールのエンドポイント
- Network Load Balancers
- AWS PrivateLink (インターフェイス VPC エンドポイント)
- Transit Gateway アタッチメント
- AWS Lambda (Hyperplane Elastic Network Interface)

スロットリング

Amazon EC2 では、インスタンスごとに追跡できる接続の最大数が定義されています。追跡が最大数に達すると、新しい接続が確立されることはないため、送受信されるパケットはすべてドロップされます。この場合、パケットを送受信するアプリケーションは正しく通信できません。`conntrack_allowance_available` ネットワークパフォーマンスメトリクスを使用して、そのインスタンスタイプでまだ利用可能な接続トラッキングの数を判断します。

インスタンスのネットワークトラフィックが追跡可能な接続の最大数を超えたために、パケットがドロップされたかどうかを判断するには、ネットワークパフォーマンスメトリクス `conntrack_allowance_exceeded` を参照します。詳細については、「EC2 インスタンスのネットワークパフォーマンスをモニタリングします。」を参照してください。

Elastic Load Balancing を実行している際にインスタンスごとに追跡できる接続の最大数を超える場合は、ロードバランサーに登録されているインスタンスの数、あるいは登録されているインスタンスのサイズのいずれかをスケールすることをお勧めします。

アイドル接続追跡タイムアウト

セキュリティグループは、確立された各接続を追跡し、リターンパケットが期待どおりに配信されることを保証します。インスタンスごとに追跡できる接続の最大数があります。接続がアイドル状態のままになると、接続追跡が使い果たされることで接続が追跡されなくなり、また、パケットがドロップされる原因となります。Elastic Network Interface では、アイドル接続の追跡にタイムアウトを設定できます。

> **Note**
>
> この機能は、Nitro ベースの EC2 インスタンスでのみ利用が可能です。

設定可能なタイムアウトは 3 つ用意されています。

- TCP 確立タイムアウト: 確立された状態のアイドル TCP 接続のタイムアウト (秒単位)。最小: 60 秒。最大: 432000 秒 (5 日間)。デフォルト: 432000 秒。推奨: 432000 秒未満。
- UDP タイムアウト: 単一方向、または 1 つのリクエスト-レスポンストランザクションのみのトラフィックが発生した、アイドル状態にある UDP フローのタイムアウト (秒単位)。最小: 30 秒。最大: 60 秒。デフォルト: 30 秒。

- UDP ストリームタイムアウト: 複数のリクエスト-レスポンストランザクションが発生したストリームとして分類される、アイドル状態にある UDP フローのタイムアウト (秒単位)。最小: 60 秒。最大: 180 秒 (3 分)。デフォルト: 180 秒。

以下のいずれかに当てはまる場合は、デフォルトのタイムアウトの変更が必要になる場合があります。

- Amazon EC2 のネットワークパフォーマンスメトリクスを使用して追跡接続をモニタリングしている場合は、conntrack_allowance_exceeded および conntrack_allowance_available メトリクスにより、ドロップされたパケットと追跡された接続使用率をモニタリングできるようになります。これにより、スケールアップまたはスケールアウトアクションにより EC2 インスタンスの容量を事前に管理し、パケットのドロップが発生する前にネットワーク接続の需要を満たすことができます。EC2 インスタンスで conntrack_allowance_exceeded のドロップが観測された場合は、不適切なクライアントやネットワークのミドルボックスが原因で TCP/UDP セッションの使用期間が長くなりすぎることを考慮して、TCP 確立のタイムアウトを低く設定すると、メリットが得られる場合があります。
- 通常、ロードバランサーまたはファイアウォールの TCP Established アイドルタイムアウトは、60 分 ~ 90 分の範囲に設定されています。ネットワークファイアウォールなどのアプライアンスからの、非常に多くの (10万件を超える) 接続を処理することが予想されるワークロードを実行している場合は、EC2 ネットワークインターフェイスでも同様のタイムアウトを設定することをお勧めします。
- 主に UDP を使用してリクエストを処理するサービス (例えば DNS、SIP、SNMP、Syslog、Radius) など、接続数が多いワークロードを実行している場合、「UDP ストリーム」のタイムアウトを 60 秒に設定すると、既存の容量のスケール対パフォーマンス比が向上し、グレーな障害を防ぐことができます。
- Network Load Balancer (NLB) とエラスティックロードバランサー (ELB) を介する TCP/UDP 接続では、すべての接続が追跡されます。TCP フローのアイドルタイムアウト値は 350 秒、UDP フローのアイドルタイムアウト値は 120 秒で、インターフェイスレベルのタイムアウト値により変化します。ELB/NLB のデフォルトよりも柔軟にタイムアウトを使用するために、ネットワークインターフェイスレベルでタイムアウトを設定することも考えられます。

以下の操作を行う際には、接続追跡のタイムアウト設定のオプションが用意されています。

- ネットワークインターフェイスの作成
- ネットワークインターフェイス属性の変更

- EC2 インスタンスの起動
- EC2 インスタンスの起動テンプレートの作成

例

次の例では、セキュリティグループに TCP および ICMP トラフィックを許可するインバウンドルールと、すべてのアウトバウンドトラフィックを許可するアウントバウンドルールがあります。

インバウンド

プロトコルのタイプ	ポート番号	ソース
TCP	22 (SSH)	203.0.113.1/32
TCP	80 (HTTP)	0.0.0.0/0
TCP	80 (HTTP)	::/0
ICMP	すべて	0.0.0.0/0

アウトバウンド

プロトコルのタイプ	ポート番号	デスティネーション
すべて	すべて	0.0.0.0/0
すべて	すべて	::/0

インスタンスまたはネットワークインターフェイスに対して直接ネットワーク接続を確立した場合、追跡動作は次のようになります。

- インバウンドルールでは 203.0.113.1/32 からのトラフィックのみ許可されるため、ポート 22 のインバウンドおよびアウトバウンド TCP トラフィック (SSH) は追跡されますが、必ずしもすべての IP アドレス (0.0.0.0/0) が追跡されるとは限りません。
- インバウンドルールとアウトバウンドルールですべての IP アドレスからのトラフィックが許可されるため、ポート 80 (HTTP) のインバウンドおよびアウトバウンド TCP トラフィックは追跡されません。
- ICMP トラフィックは常に追跡されます。

IPv4 トラフィックのアウトバウンドルールを削除すると、ポート 80 (HTTP) のトラフィックを含めすべてのインバウンドおよびアウトバウンド IPv4 トラフィックが追跡されます。IPv6 トラフィックのアウトバウンドルールを削除すると、IPv6 トラフィックでも同じことが起きます。

デフォルトセキュリティグループとカスタムセキュリティグループ

AWS アカウントには、各リージョンのデフォルト VPC のデフォルトセキュリティグループが自動的に設定されます。インスタンスを起動するときにセキュリティグループを指定しないと、そのインスタンスは VPC のデフォルトのセキュリティグループに自動的に関連付けられます。インスタンスでデフォルトのセキュリティグループを使用することを望まない場合、独自のカスタムセキュリティグループを作成して、インスタンスの起動時にそれらを指定することができます。

コンテンツ

デフォルトのセキュリティグループ

各 VPC には、デフォルトのセキュリティグループが付属しています。デフォルトのセキュリティグループを使用する代わりに、特定のインスタンスまたはインスタンスグループ用セキュリティグループを作成することをお勧めします。ただし、インスタンス起動時にセキュリティグループを指定しない場合、インスタンスにデフォルトの VPC 用セキュリティグループが関連付けられます。

デフォルトのセキュリティグループ名は「default」です。デフォルトのセキュリティグループごとのデフォルトルールを次に示します。

インバウンド

送信元	プロトコル	ポート範囲	説明
sg-1234567890abcdef0	すべて	すべて	このセキュリティグループに割り当てられたすべてのリソースからのインバウンドトラフィックを許可します。ソースは、このセキュリティグループの ID です。

アウトバウンド

送信先	プロトコル	ポート範囲	説明
0.0.0.0/0	すべて	すべて	すべてのアウトバウンド IPv4 トラフィックを許可します。
::/0	すべて	すべて	すべてのアウトバウンド IPv6 トラフィックを許可します。このルールは、VPC に IPv6 CIDR ブロックが関連付けられている場合にのみ追加されます。

デフォルトセキュリティグループの基本

- デフォルトのセキュリティグループのルールは変更できます。
- デフォルトのセキュリティグループを削除することはできません。デフォルトセキュリティグループを削除しようとした場合、次のエラーが発生します: `Client.CannotDelete`

Custom security groups

複数のセキュリティグループを作成して、インスタンスが果たすさまざまな役割 (例えば、Web サーバーまたはデータベースサーバー) を反映させることができます。

セキュリティグループを作成する場合、名前と説明を指定する必要があります。セキュリティグループには、255 文字以下の名前と説明を指定できます。また、次の特徴の制限があります。

a-z、A-Z、0-9、スペース、および._-:/()#,@[]+=&;{}!$*

セキュリティグループ名は、以下では開始できません: sg- セキュリティグループ名は VPC で一意である必要があります。

作成するセキュリティグループのデフォルトルールを次に示します。

- インバウンドトラフィックを許可しません
- すべてのアウトバウンドトラフィックを許可します

セキュリティグループを作成したら、関連するインスタンスに到達できる着信トラフィックのタイプを反映するように着信ルールを変更できます。アウトバウンドルールも変更できます。

セキュリティグループに追加できるルールのタイプの詳細については、さまざまなユースケースのセキュリティグループのルールを参照してください。

セキュリティグループの操作

インスタンスを起動する際に、インスタンスにセキュリティグループを割り当てることができます。ルールを追加または削除すると、それらの変更は、そのセキュリティグループを割り当てたすべてのインスタンスに自動的に適用されます。詳細については、インスタンスへのセキュリティグループの割り当てを参照してください。

インスタンスを起動した後、そのセキュリティグループを変更することができます。詳細については、インスタンスのセキュリティグループの変更を参照してください。

Amazon EC2 コンソールおよびコマンドラインツールを使用して、セキュリティグループとセキュリティグループルールを作成、表示、更新、削除できます。

タスク

- セキュリティグループの作成
- セキュリティグループのコピー
- セキュリティグループの表示
- セキュリティグループへのルールの追加
- セキュリティグループルールの更新
- セキュリティグループからのルールの削除
- セキュリティグループを削除する
- インスタンスへのセキュリティグループの割り当て
- インスタンスのセキュリティグループの変更

セキュリティグループの作成

インスタンスのデフォルトのセキュリティグループを使用できますが、独自のグループを作成し、システムにおけるインスタンスの様々な役割を反映させたい場合があります。

デフォルトでは、新しいセキュリティグループには、すべてのトラフィックがインスタンスを出ることを許可するアウトバウンドルールのみが設定されています。任意のインバウンドトラフィックを許可するには、またはアウトバウンドトラフィックを制限するには、ルールを追加する必要があります。

セキュリティグループは、それが対象としている VPC 内でのみ使用が可能です。

Console

セキュリティグループを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. [セキュリティグループの作成] を選択します。
4. [Basic details (基本情報)] セクションで、次の操作を行います。
   a. セキュリティグループの分かりやすい名前と簡単な説明を入力します。セキュリティグループの作成後は編集できません。名前と説明は最大 255 文字とすることができます。使用できる文字は、a-z、A-Z、0-9、スペース、および ._-:/()#,@[]+=&;{}!$* です。
   b. [VPC] で、VPC を選択します。
5. セキュリティグループルールはここで追加することも、後で追加することもできます。詳細については、を参照してくださいセキュリティグループへのルールの追加
6. タグはここで追加することも、後で追加することもできます。タグを追加するには、新しいタグを追加 をクリックし、タグのキーと値を入力します。
7. [セキュリティグループの作成] を選択します。

Command line

セキュリティグループを作成するには

以下のいずれかのコマンドを使用します。

- create-security-group (AWS CLI)
- New-EC2SecurityGroup (AWS Tools for Windows PowerShell)

セキュリティグループのコピー

既存のセキュリティグループのコピーを作成して、新しいセキュリティグループを作成することができます。セキュリティグループをコピーすると、元のセキュリティグループと同じインバウンドルールとアウトバウンドルールでコピーが作成されます。元のセキュリティグループが VPC にある場合、別の VPC を指定しない限り、コピーは同じ VPC に作成されます。

コピーには新しい一意のセキュリティグループ ID が割り当てられ、名前を指定する必要があります。また、説明を追加することもできます。

セキュリティグループは、あるリージョンから別のリージョンにコピーできません。

Amazon EC2 コンソールを使いセキュリティグループのコピーを作成することができます。

セキュリティグループをコピーするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. コピーするセキュリティグループを選択し、[アクション]、[Copy to new security group (新しいセキュリティグループにコピー)] の順に選択します。
4. 名前と任意で説明を指定し、必要に応じて VPC とセキュリティグループルールを変更します。
5. [Create] (作成) を選択します。

セキュリティグループの表示

セキュリティグループに関する情報は、次のいずれかの方法で表示できます。

Console

セキュリティグループを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループが一覧表示されます。インバウンドルールやアウトバウンドルールなど、特定のセキュリティグループの詳細を表示するには、[セキュリティグループ ID] 列でその ID を選択します。

Command line

セキュリティグループを表示するには

以下のいずれかのコマンドを使用します。

- describe-security-groups (AWS CLI)
- describe-security-group-rules (AWS CLI)

- Get-EC2SecurityGroup (AWS Tools for Windows PowerShell)

Amazon EC2 Global View

Amazon EC2 グローバルビューを使用して、AWSアカウントが有効になっているすべてのリージョンにわたってセキュリティグループを表示できます。詳細については、Amazon EC2 Global Viewを参照してください。

セキュリティグループへのルールの追加

ルールをセキュリティグループに追加すると、セキュリティグループに関連付けられているすべてのインスタンスに新しいルールが自動的に適用されます。ルールの適用には、少し時間がかかる場合があります。詳細については、さまざまなユースケースのセキュリティグループのルールおよびセキュリティグループのルールを参照してください。

Console

セキュリティグループにインバウンドルールを追加するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループを選択し、[アクション]、[インバウンドのルールの編集] の順にクリックします。
4. 各ルールについて [ルールを追加] を選択し、次の操作を行います。
   a. [タイプ] で、許可するプロトコルのタイプを選択します。
      - [カスタム TCP] または [カスタム UDP] の場合は、許可するポート範囲を入力する必要があります。例えば、0-99 と指定します。
      - [カスタム ICMP] の場合は、[プロトコル] から ICMP タイプを選択する必要があります。ポート範囲は自動的に設定されます。例えば、pingコマンドを許可する場合は [プロトコル] から、[Echoリクエスト] を選択します。
      - その他のタイプについては、プロトコルとポート範囲は自動的に設定されます。
   b. [送信元] で、次のいずれかの操作を行いトラフィックを許可します。
      - [カスタム] を選択し、IP アドレス (CIDR 表記)、CIDR ブロック、別のセキュリティグループ、あるいはプレフィクスリストを入力します。

- 指定したプロトコルのすべてのトラフィックがインスタンスに到達することを許可するには、[任意の場所] を選択します。このオプションでは、送信元として IPv4 の CIDR ブロック 0.0.0.0/0 が自動的に追加されます。セキュリティグループが、IPv6 が有効な VPC 内にある場合、このオプションでは ::/0 IPv6 CIDR ブロックのためのルールが自動的に追加されます。

> **Warning**
>
> [Anywhere] (どこでも) を選択した場合は、すべての IPv4 および IPv6 アドレスが、指定されたプロトコルでインスタンスにアクセスできるようにします。ポート 22 (SSH) または 3389 (RDP) のルールを追加する場合は、特定の IP アドレスまたはアドレスの範囲のみにインスタンスへのアクセスを許可する必要があります。

- ローカルコンピュータのパブリック IPv4 アドレスからのインバウンドトラフィックのみを許可するには、[My IP] (マイ IP) を選択します。

c. [説明] では、任意でルールの簡単な説明を指定できます。

5. [変更のプレビュー]、[ルールの保存] を選択します。

セキュリティグループにアウトバウンドルールを追加するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループを選択し、[アクション]、[アウトバウンドルールを編集] の順にクリックします。
4. 各ルールについて [ルールを追加] を選択し、次の操作を行います。

a. [タイプ] で、許可するプロトコルのタイプを選択します。

- [カスタム TCP] または [カスタム UDP] の場合は、許可するポート範囲を入力する必要があります。例えば、0-99 と指定します。
- [カスタム ICMP] の場合は、[プロトコル] から ICMP タイプを選択する必要があります。ポート範囲は自動的に設定されます。
- その他のタイプについては、プロトコルとポート範囲は自動的に設定されます。

b. [送信先] で、次のいずれかの操作を行います。

- [カスタム] をクリックし、アウトバウンドトラフィックを許可する IP アドレス (CIDR 表記)、CIDR ブロック、別のセキュリティグループ、プレフィクスリストのいずれかを入力します。
- すべての IP アドレスへのアウトバウンドトラフィックを許可するには、[任意の場所] を選択します。このオプションでは、送信先として、0.0.0.0/0 IPv4 CIDR ブロックが自動的に追加されます。

  セキュリティグループが、IPv6 が有効な VPC 内にある場合、このオプションでは ::/0 IPv6 CIDR ブロックのためのルールが自動的に追加されます。
- ローカルコンピュータのパブリック IPv4 アドレスへのアウトバウンドトラフィックのみを許可するには、[My IP] (マイ IP) を選択します。

c. (オプション) [説明] に、ルールの簡単な説明を入力します。

5. [変更のプレビュー]、[確認] の順に選択します。

Command line

セキュリティグループにルールを追加するには

以下のいずれかのコマンドを使用します。

- authorize-security-group-ingress (AWS CLI)
- Grant-EC2SecurityGroupIngress (AWS Tools for Windows PowerShell)

セキュリティグループに 1 つ以上の Egress ルールを追加するには

以下のいずれかのコマンドを使用します。

- authorize-security-group-egress (AWS CLI)
- Grant-EC2SecurityGroupEgress (AWS Tools for Windows PowerShell)

セキュリティグループルールの更新

セキュリティグループルールの更新は、次のいずれかの方法で行うことができます。更新されたルールは、セキュリティグループに関連付けられているすべてのインスタンスに自動的に適用されます。

Console

コンソールを使用して既存のセキュリティグループルールのプロトコル、ポート範囲、または送信元または送信先を変更すると、コンソールは既存のルールを削除し、新しいルールを追加します。

セキュリティグループルールを更新するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループを選択します。
4. [Actions] (アクション)、[Edit inbound rules] (インバウンドルールを編集) の順にクリックして、インバウンドトラフィックのルールを更新するか、[Actions] (アクション)、[Edit outbound rules] (アウトバウンドルールを編集) の順にクリックして、アウトバウンドトラフィックのルールを更新します。
5. 必要に応じてルールを更新します。
6. [変更のプレビュー]、[確認] の順に選択します。

セキュリティグループルールにタグ付けするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループを選択します。
4. [インバウンドルール] または [アウトバウンドルール] タブで、対象となるルールのチェックボックスを選択してから、[タグを管理] をクリックします。
5. [タグの管理] ページには、ルールに割り当てられているすべてのタグが表示されます。タグを追加するには、[タグの追加] を選択し、タグのキーと値を入力します。タグを削除するには、削除するタグの横にある [Remove] を選択します。
6. [Save changes] を選択します。

Command line

Amazon EC2 API またはコマンドラインツールを使用して、既存のルールのプロトコル、ポート範囲、送信元または送信先を変更することはできません。代わりに、既存のルールを削除して新しいルールを追加する必要があります。ただし、既存のルールの説明を更新することはできます。

ルールを更新するには

以下のいずれかのコマンドを使用します。

- modify-security-group-rules (AWS CLI)

既存のインバウンドルールの説明を更新するには

以下のいずれかのコマンドを使用します。

- update-security-group-rule-descriptions-ingress (AWS CLI)
- Update-EC2SecurityGroupRuleIngressDescription (AWS Tools for Windows PowerShell)

既存のアウトバウンドルールの説明を更新するには

以下のいずれかのコマンドを使用します。

- update-security-group-rule-descriptions-egress (AWS CLI)
- Update-EC2SecurityGroupRuleEgressDescription (AWS Tools for Windows PowerShell)

セキュリティグループルールにタグ付けするには

以下のいずれかのコマンドを使用します。

- create-tags (AWS CLI)
- New-EC2Tag (AWS Tools for Windows PowerShell)

セキュリティグループからのルールの削除

セキュリティグループからルールを削除すると、その変更内容が自動的にセキュリティグループに関連付けられているインスタンスに適用されます。

セキュリティグループからのルールの削除は、次のいずれかの方法で行うことができます。

Console

セキュリティグループルールを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Security Groups] を選択します。
3. 更新するセキュリティグループを選択し、[アクション] を選択してから、[インバウンドのルールの編集] を選択してインバウンドルールを削除するか、[アウトバウンドのルールの編集] を選択してアウトバウンドルールを削除します。
4. 削除するルールの右にある [削除] ボタンを選択します。
5. [Save Rules] (ルールの保存) を選択します。または、[変更をプレビュー] を選択し、変更を確認して [確認] を選択します。

Command line

セキュリティグループから 1 つ以上の Ingress ルールを削除するには

以下のいずれかのコマンドを使用します。

- revoke-security-group-ingress (AWS CLI)
- Revoke-EC2SecurityGroupIngress (AWS Tools for Windows PowerShell)

セキュリティグループから 1 つ以上の Egress ルールを削除するには

以下のいずれかのコマンドを使用します。

- revoke-security-group-egress (AWS CLI)
- Revoke-EC2SecurityGroupEgress (AWS Tools for Windows PowerShell)

セキュリティグループを削除する

インスタンスに関連付けられているセキュリティグループを削除することはできません。デフォルトセキュリティグループを削除することはできません。同じ VPC の他のセキュリティグループのルールによって参照されているセキュリティグループは削除できません。セキュリティグループが独自のいずれかのルールで参照されている場合は、セキュリティグループを削除する前に、まずルールを削除する必要があります。

Console

セキュリティグループを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。

2. ナビゲーションペインで、[Security Groups] を選択します。
3. セキュリティグループを選択して、[アクション]、[セキュリティグループを削除] を選択します。
4. 確認を求めるメッセージが表示されたら、[削除] を選択します。

Command line

セキュリティグループを削除するには

以下のいずれかのコマンドを使用します。

- delete-security-group (AWS CLI)
- Remove-EC2SecurityGroup (AWS Tools for Windows PowerShell)

インスタンスへのセキュリティグループの割り当て

インスタンスを起動する際に、インスタンスに 1 つ以上のセキュリティグループを割り当てることができます。起動テンプレートでは、1 つ以上のセキュリティグループを指定することもできます。セキュリティグループは、起動テンプレートを使用して起動されるすべてのインスタンスに割り当てられます。

- インスタンスを起動する際に、インスタンスにセキュリティグループを割り当てるには、「定義済みのパラメータを使用したインスタンスの起動」(新しいコンソール) または「ステップ 6: セキュリティグループを設定する」(古いコンソール) の「ネットワーク設定」を参照してください。
- 起動テンプレートでセキュリティグループを指定するには、「定義したパラメータを使用した新しい起動テンプレートの作成」の「ネットワーク設定」を参照してください。

インスタンスのセキュリティグループの変更

インスタンスを起動した後、セキュリティグループを追加または削除することで、セキュリティグループを変更できます。

要件

- インスタンスは、`running` または `stopped` 状態である必要があります。
- セキュリティグループは VPC に固有です。セキュリティグループを作成した VPC 内起動されている 1 つ以上のインスタンスにセキュリティグループを割り当てることができます。

Console

インスタンスのセキュリティグループを変更するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[セキュリティ]、[セキュリティグループの変更] の順に選択します。
4. [関連付けられたセキュリティグループ] で、リストからセキュリティグループを選択し、[セキュリティグループを追加] を選択します。

   すでに関連付けられているセキュリティグループを削除するには、そのセキュリティグループで [削除] を選択します。
5. [Save] を選択します。

Command line

インスタンスのセキュリティグループを変更するには

以下のいずれかのコマンドを使用します。

- modify-instance-attribute (AWS CLI)
- Edit-EC2InstanceAttribute (AWS Tools for Windows PowerShell)

さまざまなユースケースのセキュリティグループのルール

セキュリティグループを作成し、そのセキュリティグループに関連付けられたインスタンスのロールを反映したルールを追加できます。例えば、ウェブサーバーとして構成されているインスタンスには、インバウンドの HTTP および HTTPS アクセスを許可するセキュリティグループルールが必要です。同様に、データベースのインスタンスには、データベースのタイプに対するアクセス (MySQL のポート 3306 でのアクセスなど) を許可するルールが必要です。

以下は、特定の種類のアクセスのセキュリティグループに追加できるルールの種類の例です。

例

- ウェブサーバールール
- データベースサーバールール

- コンピュータからのインスタンスへの接続ルール
- 同じセキュリティグループを持つインスタンスからインスタンスに接続するためのルール
- Ping/ICMP のルール
- DNS サーバールール
- Amazon EFS ルール
- Elastic Load Balancing ルール
- VPC ピア接続ルール

ウェブサーバールール

次のインバウンドルールでは、任意の IP アドレスからの HTTP および HTTPS アクセスを許可します。VPC が IPv6 に対して有効になっている場合、IPv6 アドレスからインバウンド HTTP および HTTPS トラフィックを制御するルールを追加できます。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	80 (HTTP)	0.0.0.0/0	任意の IPv4 アドレスからのインバウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	0.0.0.0/0	任意の IPv4 アドレスからのインバウンド HTTPS アクセスを許可します
TCP	6	80 (HTTP)	::/0	任意の IPv6 アドレスからのインバウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	::/0	任意の IPv6 アドレスからのインバウンド HTTPS アクセスを許可します

データベースサーバールール

次のインバウンドルールは、インスタンスで実行中のデータベースのタイプに応じて、データベースアクセス用に追加するルールの例です。Amazon RDS インスタンスの詳細については、Amazon RDS ユーザーガイドを参照してください。

ソース IP には、次のいずれかを指定します。

- ローカルネットワークの特定の IP アドレスまたは IP アドレス範囲 (CIDR ブロック表記)
- データベースにアクセスするインスタンスのグループのセキュリティグループ ID

プロトコルのタイプ	プロトコル番号	ポート	コメント
TCP	6	1433 (MS SQL)	Amazon RDS インスタンス上など、Microsoft SQL Server データベースにアクセスするデフォルトのポート
TCP	6	3306 (MYSQL/Aurora)	Amazon RDS インスタンス上など、MySQL または Aurora データベースにアクセスするデフォルトのポート
TCP	6	5439 (Redshift)	Amazon Redshift クラスターデータベースにアクセスするデフォルトのポート。
TCP	6	5432 (PostgreSQL)	Amazon RDS インスタンス上など、PostgreSQL データベースにアクセスするデフォルトのポート
TCP	6	1521 (Oracle)	Amazon RDS インスタンス上など、Oracle データベースにアクセスするデフォルトのポート

必要に応じて、データベースサーバーからのアウトバウンドトラフィックを制限できます。例えば、ソフトウェアの更新ではインターネットへのアクセスを許可し、その他のトラフィックはすべて制限することができます。最初に、すべてのアウトバンドトラフィックを許可するデフォルトのアウトバウンドルールを削除する必要があります。

プロトコルのタイプ	プロトコル番号	ポート	送信先 IP	コメント
TCP	6	80 (HTTP)	0.0.0.0/0	任意の IPv4 アドレスへのアウトバウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	0.0.0.0/0	任意の IPv4 アドレスへのアウトバウンド HTTPS アクセスを許可します
TCP	6	80 (HTTP)	::/0	(IPv6 が有効な VPC のみ) 任意の IPv6 アドレスへのアウトバウンド HTTP アクセスを許可します
TCP	6	443 (HTTPS)	::/0	(IPv6 が有効な VPC のみ)、任意の IPv6 アドレスへのアウトバウンド HTTPS アクセスを許可します

コンピュータからのインスタンスへの接続ルール

インスタンスに接続するには、セキュリティグループに SSH アクセス (Linux インスタンスの場合) または RDP アクセス (Windows インスタンスの場合) を許可するインバウンドルールが必要です。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP
TCP	6	22 (SSH)	ローカルコンピュータのパブリック IPv4 アドレス、またはローカルネットワークの IP アドレスの範囲。V

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP
			PC が IPv6 に対して有効で、インスタンスに IPv6 アドレスがある場合、IPv6 アドレスまたは範囲を入力できます。
TCP	6	3389 (RDP)	ローカルコンピュータのパブリック IPv4 アドレス、またはローカルネットワークの IP アドレスの範囲。VPC が IPv6 に対して有効で、インスタンスに IPv6 アドレスがある場合、IPv6 アドレスまたは範囲を入力できます。

同じセキュリティグループを持つインスタンスからインスタンスに接続するためのルール

同じセキュリティグループに関連付けられたインスタンスが相互に通信できるようにするには、そのためのルールを明示的に追加する必要があります。

> **Note**
>
> ミドルボックスアプライアンスを介して異なるサブネット内の 2 つのインスタンス間のトラフィックを転送するようにルートを設定するには、両方のインスタンスのセキュリティグループでインスタンス間のトラフィックがフローできるようにする必要があります。各インスタンスのセキュリティグループは、他のインスタンスのプライベート IP アドレス、または他のインスタンスを含むサブネットの CIDR 範囲を送信元として参照する必要があります。他のインスタンスのセキュリティグループを送信元として参照する場合、インスタンス間のトラフィックは許可されません。

次の表は、関連付けられたインスタンスの相互通信を可能にするセキュリティグループのインバウンドルールを示します。このルールでは、すべてのタイプのトラフィックが許可されます。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP
-1 (すべて)	-1 (すべて)	-1 (すべて)	セキュリティグループの ID、または他のインスタンスを含むサブネットの CIDR 範囲 (注を参照)。

Ping/ICMP のルール

ping コマンドは、ICMP トラフィックの一種です。インスタンスで Ping を実行するには、次のインバウンド ICMP ルールのうち 1 つを追加する必要があります。

タイプ	プロトコル	ソース		
カスタム ICMP - IPv4	エコーリクエスト	お使いのコンピュータのパブリック IPv4 アドレス、特定の IPv4 アドレス、または任意の場所の IPv4 あるいは IPv6 アドレス。		
すべての ICMP - IPv4	IPv4 ICMP (1)	お使いのコンピュータのパブリック IPv4 アドレス、特定の IPv4 アドレス、または任意の場所の IPv4 あるいは IPv6 アドレス。		

ping6 コマンドを使用してインスタンスの IPv6 アドレスに ping を実行するには、次のインバウンド ICMPv6 ルールを追加する必要があります。

タイプ	プロトコル	ソース		
すべての ICMP - IPv6	IPv6 ICMP (58)	お使いのコンピュータの IPv6 アドレス、特定の IPv4 アドレス、または任意の場所の IPv4 あるいは IPv6 アドレス。		

DNS サーバールール

DNS サーバーとして EC2 インスタンスをセットアップした場合、TCP および UDP のトラフィックがポート 53 経由で DNS サーバーに到達できるようにする必要があります。

ソース IP には、次のいずれかを指定します。

- ネットワークの IP アドレスまたは IP アドレス範囲 (CIDR ブロック表記)
- ネットワークで、DNS サーバーにアクセスする必要がある一連のインスタンスのセキュリティグループの ID

プロトコルのタイプ	プロトコル番号	ポート
TCP	6	53
UDP	17	53

Amazon EFS ルール

Amazon EC2 インスタンスで Amazon EFS ファイルシステムを使用している場合、Amazon EFS マウントターゲットに関連付けるセキュリティグループは、NFS プロトコル経由のトラフィックを許可する必要があります。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	2049 (NFS)	セキュリティグループの ID	このセキュリティグループに関連付けられたリソース (マウントターゲットを含む) からのインバウンド NFS アクセスを許可します。

Amazon EC2 インスタンスに Amazon EFS ファイルシステムをマウントするには、インスタンスに接続する必要があります。したがって、インスタンスに関連付けられているセキュリティグループには、ローカルコンピュータまたはローカルネットワークからのインバウンド SSH を許可するルールが必要です。

プロトコルのタイプ	プロトコル番号	ポート	送信元 IP	コメント
TCP	6	22 (SSH)	ローカルコンピュータの IP アドレス範囲、またはネットワークの IP アドレス範囲 (CIDR ブロック表記)。	ローカルコンピュータからのインバウンド SSH アクセスを許可します。

Elastic Load Balancing ルール

ロードバランサーを使用している場合、ロードバランサーに関連付けられたセキュリティグループには、インスタンスやターゲットとの通信を許可するルールが必要です。詳細については、Classic Load Balancer のユーザーガイドのClassic Load Balancer のセキュリティグループの設定、およびApplication Load Balancer のユーザーガイドのApplication Load Balancer のセキュリティグループを参照してください。

VPC ピア接続ルール

VPC セキュリティグループのインバウンドルールまたはアウトバウンドルールを更新して、ピアリング接続 VPC のセキュリティグループを参照できます。これにより、トラフィックはピア VPC の

参照されるセキュリティグループに関連付けられたインスタンスに出入りできます。VPC ピア接続のセキュリティグループを設定する方法の詳細については、セキュリティグループの更新によるピア VPC グループの参照を参照してください。

インターフェイス VPC エンドポイントを使用して Amazon EC2 にアクセスします。

VPC と Amazon EC2 の間にプライベート接続を作成することで、VPC のセキュリティ体制を向上させることができます。インターネットゲートウェイ、NAT デバイス、VPN 接続、または AWS Direct Connect 接続を使用せずに、VPC 内にあるかのように Amazon EC2 にアクセスできます。VPC のインスタンスは、パブリック IP アドレスがなくても Amazon EC2 にアクセスできます。

詳細については、「AWS PrivateLink Guide (AWS PrivateLink ガイド)」の「Access an AWS のサービス using an interface VPC endpoint (インターフェイス VPC エンドポイントを使用して にアクセスする)」を参照してください。

コンテンツ

インターフェイス VPC エンドポイントを作成する

以下のサービス名を使用して、Amazon EC2 のインターフェイス エンドポイントを作成します。

- com.amazonaws.***region***.ec2 — Amazon EC2 API アクションのエンドポイントを作成します。

詳細については、[AWS PrivateLink Guide] (ガイド) の[Access an AWS のサービス using an interface VPC endpoint] (インターフェイス VPC エンドポイントを使用した へのアクセス) を参照してください。

エンドポイントポリシーを作成する

エンドポイントポリシーは、インターフェイスエンドポイントにアタッチできる IAM リソースです。デフォルトのエンドポイントポリシーでは、インターフェイスエンドポイント経由での Amazon EC2 API へのフルアクセスが許可されています。VPC から Amazon EC2 API へのアクセス許可をコ

ントロールするには、カスタムエンドポイントポリシーをインターフェイスエンドポイントにアタッチします。

エンドポイントポリシーは、以下の情報を指定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- このアクションを実行できるリソース。

> **Important**
>
> デフォルト以外のポリシーが Amazon EC2 のインターフェイス VPC エンドポイントに適用されると、`RequestLimitExceeded` からの失敗したリクエストなど、特定の失敗した API リクエストが AWS CloudTrail または Amazon CloudWatch にログ記録されない場合があります。

詳細については、[AWS PrivateLink Guide] (ガイド) の[Control access to services using endpoint policies] (エンドポイントポリシーを使用してサービスへのアクセスをコントロール) を参照してください。

次の例は、暗号化されていないボリュームを作成したり、暗号化されていないボリュームでインスタンスを起動したりするアクセス許可を拒否する VPC エンドポイントポリシーを示しています。このポリシー例では、他のすべての Amazon EC2 のアクションを実行するアクセス許可も付与しています。

```
{
    "Version": "2012-10-17",
    "Statement": [
    {
        "Action": "ec2:*",
        "Effect": "Allow",
        "Resource": "*",
        "Principal": "*"
    },
    {
        "Action": [
            "ec2:CreateVolume"
        ],
```

```
        "Effect": "Deny",
        "Resource": "*",
        "Principal": "*",
        "Condition": {
            "Bool": {
                "ec2:Encrypted": "false"
            }
        }
    },
    {
        "Action": [
            "ec2:RunInstances"
        ],
        "Effect": "Deny",
        "Resource": "*",
        "Principal": "*",
        "Condition": {
            "Bool": {
                "ec2:Encrypted": "false"
            }
        }
    }]
}
```

Amazon EC2 での更新管理

弊社は、EC2 インスタンスのオペレーティングシステムやアプリケーションに対するパッチ適用やそれらの更新およびセキュリティ確保を定期的に行うよう推奨しています。AWS Systems Manager パッチマネージャーを使うと、オペレーティングシステムとアプリケーションの双方に関するセキュリティ関連更新のインストールプロセスを自動化できます。

Auto Scaling グループの EC2 インスタンスの場合、AWS-PatchAsgInstance ランブックを使用すると、パッチ適用中のインスタンスが置き換えられるのを防ぐことができます。代わりに、アプリケーションベンダーが提供している、自動更新サービスまたは推奨更新インストールプロセスを使用することもできます。

Windows Server を実行している最新の Amazon EC2 AMI のリストについては、AWS Windows AMI バージョンの詳細を参照してください。

Amazon EC2 のコンプライアンス検証

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、「コンプライアンスプログラム別の範囲」の「AWS のサービス」と「」の「AWS のサービス」を参照し、関心のあるコンプライアンスプログラムを選択してください。一般的な情報については、「AWS コンプライアンスプログラム」を参照してください。

AWS Artifact を使用して、サードパーティーの監査レポートをダウンロードできます。詳細については、「AWS Artifact でレポートをダウンロードする」を参照してください。

AWS のサービス を使用する際のユーザーのコンプライアンス責任は、ユーザーのデータの機密性や貴社のコンプライアンス目的、適用される法律および規制によって決まります。AWS では、コンプライアンスに役立つ以下のリソースを提供しています。

- セキュリティとコンプライアンスのクイックスタートガイド — これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、ベースライン環境をセキュリティとコンプライアンスに重点を置いた AWS にデプロイするための手順を示します。
- Amazon Web Services での HIPAA のセキュリティとコンプライアンスのためのアーキテクチャ – このホワイトペーパーは、企業が AWS を使用して HIPAA 対応アプリケーションを作成する方法を説明しています。

  > **Note**
  >
  > すべての AWS のサービスが HIPAA 適格であるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。

- AWS コンプライアンスのリソース– このワークブックおよびガイドのコレクションは、お客様の業界と拠点に適用される場合があります。
- AWS Customer Compliance Guide — コンプライアンスの観点から見た責任共有モデルを理解できます。このガイドは、AWS のサービスを保護するためのベストプラクティスを要約したものであり、複数のフレームワーク (米国標準技術研究所 (NIST)、ペイメントカード業界セキュリティ標準評議会 (PCI)、国際標準化機構 (ISO) など) にわたるセキュリティ統制へのガイダンスがまとめられています。
- 「AWS Config デベロッパーガイド」の「ルールでのリソースの評価」 - AWS Config サービスは、自社のプラクティス、業界ガイドライン、および規制に対するリソースの設定の準拠状態を評価します。

- AWS Security Hub – この AWS のサービス は、AWS 内のセキュリティ状態の包括的なビューを提供します。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、「Security Hub のコントロールリファレンス」を参照してください。
- AWS Audit Manager - この AWS のサービスは、AWS の使用状況を継続的に監査して、リスクの管理方法や、規制および業界標準へのコンプライアンスの管理方法を簡素化するために役立ちます。

NitroTPM

Nitro Trusted Platform Module (NitroTPM) は、AWS Nitro System によって提供され TPM 2.0 仕様に準拠した仮想デバイスです。インスタンスの認証に使用されるアーティファクト (パスワード、証明書、暗号化キーなど) を安全に保存します。NitroTPM は、キーを生成し、暗号化機能 (ハッシング、署名、暗号化、復号化など) に使用できます。

NitroTPM は、測定されたブートを提供します。これは、ブートローダーとオペレーティングシステムがすべてのブートバイナリの暗号化ハッシュを作成し、それらを NitroTPM 内部プラットフォーム構成レジスタ (PCR) の以前の値と組み合わせるプロセスです。測定されたブートを使用することで、NitroTPM から署名された PCR 値を取得し、それらを使用してインスタンスのブートソフトウェアの整合性をリモートエンティティに証明することができます。これは、リモート認証と呼ばれます。

NitroTPM を使用することで、キーおよびシークレットに特定の PCR 値をタグ付けできるため、PCR の値、つまりインスタンスの整合性が変更された場合にそれらにアクセスすることはできません。この特別な形式の条件付きアクセスは、封印および開封と呼ばれます。BitLocker などのオペレーティングシステムのテクノロジーは、NitroTPM を使用してドライブの復号化キーを封印し、オペレーティングシステムが正しく起動し、正常な状態である場合にのみドライブを復号化できます。

NitroTPM を使用するには、NitroTPM サポート用に設定された Amazon マシンイメージ (AMI) を選択してから、AMI を使用して Nitro ベースのインスタンスを起動する必要があります。Amazon のビルド済みの AMI を選択するか、自分で作成することができます。

コスト

NitroTPM を使用しても追加コストはかかりません。お客様は、使用した基本リソースに対してのみ、料金を支払います。

トピック

考慮事項

以下の考慮事項は、NitroTPM を使用する場合に適用されます。

- NitrotPM ベースキーで暗号化された BitLocker ボリュームは、オリジナルインスタンスでのみ使用できます。
- NitroTPM 状態は Amazon EBS スナップショットには含まれていません。
- NitroTPM 状態は VM Import/Export イメージには含まれていません。
- NitroTPM サポートは、AMI の作成時に `tpm-support` パラメーターに `v2.0` 値を指定することで有効になります。AMI を使用してインスタンスを起動すると、インスタンスの属性を変更することはできません。NitroTPM を使用したインスタンスでは、ModifyInstanceAttribute API をサポートしていません。
- NitroTPM を設定した AMI の作成は、AWS CLI で RegisterImage API を使用した場合のみ可能で、Amazon EC2 コンソールではできません。
- NitroTPM は、Outposts ではサポートされていません。
- NitroTPMは、ローカルゾーン、または Wavelength Zone ではサポートされていません。

Linux インスタンスを起動する前提条件

NitroTPM を有効にした Linux インスタンスを起動するには、以下の前提条件を設定する必要があります。Linuxインスタンスを起動するための前提条件については、「Linux インスタンス用 Amazon EC2 ユーザーガイド」「」の「前提条件」「」を参照してください。

AMI

NitroTPM が有効な AMI が必要です。

現在、NitroTPM が有効な Amazon Linux AMI はありません。サポートされている AMI を使用するには、独自の Linux AMI でいくつかの設定手順を実行する必要があります。詳細については、「NitroTPM サポート用の Linux AMI を作成する」を参照してください。

オペレーティングシステム

AMI には、TPM 2.0 Command Response Buffer (CRB) ドライバーを搭載したオペレーティングシステムが含まれている必要があります。Amazon Linux 2 など、現在のほとんどのオペレーティングシステムには、TPM 2.0 CRB ドライバーが含まれています。

インスタンスタイプ

サポートされている仮想化インスタンスタイプ:

- 汎用: M5, M5a, M5ad, M5d, M5dn, M5n, M5zn, M6a, M6i, M6id, M6idn, M6in, M7a, M7i, M7i-flex, T3, T3a
- コンピューティング最適化: C5, C5a, C5ad, C5d, C5n, C6a, C6i, C6id, C6in, C7a, C7i
- メモリ最適化: R5, R5a, R5ad, R5b, R5d, R5dn, R5n, R6a, R6i, R6idn, R6in, R6id, R7a, R7i, R7iz, z1d
- ストレージ最適化:: D3、D3en、I3en、I4i
- 高速コンピューティング: G4dn、G5、Inf1、Inf2
- ハイパフォーマンスコンピューティング: Hpc6a、Hpc6id

サポート対象外:Graviton ベースのインスタンス、Xen インスタンス、Mac インスタンス、ベアメタルインスタンス

UEFI ブートモード

NitroTPM では、インスタンスが UEFI ブートモードで実行されている必要があります。そのためには、AMI が UEFI ブートモード用に設定されている必要があります。詳細については、「UEFI セキュアブート」を参照してください。

NitroTPM サポート用の Linux AMI を作成する

Linux AMI を登録するときに、NitroTPM サポート用の AMI を設定します。NitroTPM サポートを後で設定することはできません。

NitroTPM サポート用に事前設定された Windows AMI のリストについては、「Windows インスタンス用 Amazon EC2 ユーザーガイド」の「Windows インスタンスを起動する前提条件」を参照してください。

NitroTPM サポート用の Linux AMI を登録するには

1. 必要な Linux AMI を使用して一時インスタンスを起動します。
2. インスタンスが running 状態になったら、インスタンスのルートボリュームのスナップショットを作成します。
3. 新しい AMI を登録します。register-image コマンドを使用します。--tpm-support で、v2.0 を指定します。--boot-mode で、uefi を指定します。また、前のステップで作成したスナップショットを使用して、ルートボリュームのブロックデバイスマッピングを指定します。

```
aws ec2 register-image \
    --name my-image \
    --boot-mode uefi \
    --architecture x86_64 \
    --root-device-name /dev/xvda \
    --block-device-mappings DeviceName=/dev/xvda,Ebs={SnapshotId=snapshot_id} \
    --tpm-support v2.0
```

正常な出力

```
{
"ImageId": "ami-0123456789example"
}
```

4. ステップ 1 で起動した一時インスタンスは、不要になったら終了します。

AMI が NitroTPM に対して有効になっているかどうかを確認する

describe-images または describe-image-attributes を使用して、AMI が NitroTPM に対して有効になっているかどうかを検証します。

describe-images を使用して AMI が NitroTPM に対して有効になっているかどうかを確認する

describe-images コマンドを使用して、AMI の ID を指定します。

```
aws ec2 describe-images --image-ids ami-0123456789example
```

NitroTPM が AMI に対して有効になっている場合、"TpmSupport": "v2.0" が出力に表示されます。

```
{
    "Images": [
        {
        ...
        "BootMode": "uefi",
        ...
        "TpmSupport": "v2.0"
        }
    ]
}
```

describe-image-attribute を使用して AMI が NitroTPM に対して有効になっているかどうかを確認する

describe-image-attribute コマンドを使用して、attribute パラメータを tpmSupport 値で指定します。

> **Note**
>
> describe-image-attribute を呼び出すには、AMI の所有者である必要があります。

```
aws ec2 describe-image-attribute \
    --region us-east-1 \
    --image-id ami-0123456789example \
    --attribute tpmSupport
```

AMI に対して NitroTPM が有効になっている場合、TpmSupport 値は "v2.0" です。describe-image-attribute は、リクエストで指定された属性のみを返すことに注意してください。

```
{
    "ImageId": "ami-0123456789example",
    "TpmSupport": {
        "Value": "v2.0"
    }
}
```

インスタンスでの NitroTPM の使用を有効または停止する

NitroTPM サポートが有効になっている AMI からインスタンスを起動した場合、インスタンスは NitroTPM を有効にして起動します。NitroTPM の使用を停止するようにインスタンスを設定できます。インスタンスが NitroTPM に対して有効であるかどうかを確認できます。

トピック

NitroTPM を有効にしてインスタンスを起動する

前提条件を使用してインスタンスを起動すると、インスタンスで NitroTPM が自動的に有効になります。NitroTPM は、起動時にインスタンスでのみ有効にできます。インスタンスの起動方法についての詳細は、インスタンスの起動 を参照してください。

インスタンスでの NitroTPM の使用を停止する

NitroTPM を有効にしてインスタンスを起動した後、インスタンスの NitroTPM を無効にすることはできません。ただし、次のツールを使用して、インスタンスで TPM 2.0 デバイスドライバーを無効にすることで、NitroTPM の使用をオペレーティングシステムで停止するよう設定できます。

- Linux の場合は tpm-tools を使用してください。

デバイスドライバーを無効化する方法の詳細については、オペレーティングシステムのドキュメントを参照してください。

NitroTPM がインスタンス内でアクセス可能かどうかを検証する

AWS CLI を使用してインスタンスが NitroTPM サポートに対して有効になっているかどうかを検証する

describe-instances AWS CLI コマンドを使用して、インスタンス ID を指定します。現時点では、Amazon EC2 コンソールには `TpmSupport` フィールドは表示されません。

```
aws ec2 describe-instances --instance-ids i-0123456789example
```

NitroTPM サポートがインスタンスに対して有効になっている場合、"TpmSupport": "v2.0" が出力に表示されます。

```
"Instances": {
             "InstanceId":"0123456789example",
             "InstanceType":"c5.large",
             ...
             "BootMode": "uefi",
             "TpmSupport": "v2.0"
             ...
             }
```

Amazon EC2 インスタンスのストレージオプション

Amazon EC2 にはインスタンスを格納するための、柔軟で使いやすく、コスト効率の良いデータストレージオプションが用意されています。各オプションは独自のパフォーマンスと耐久性を備えています。これらのストレージオプションは、要件に応じて個別に使用することも、組み合わせて使用することもできます。

Amazon EBS

Amazon EBS は、インスタンスにアタッチまたはデタッチできる、耐久性の高いブロックレベルのストレージボリュームを提供します。複数の EBS ボリュームを 1 つのインスタンスにアタッチできます。EBS ボリュームは、関連するインスタンスの有効期間とは無関係に存続します。EBS ボリュームは暗号化できます。データのバックアップコピーを保持するには、EBS ボリュームからスナップショットを作成します。スナップショットは Amazon S3 に保存されます。スナップショットから EBS ボリュームを作成できます。

インスタンスストア

インスタンスストアは、インスタンス用のブロックレベルの一時ストレージを提供します。インスタンスストアボリュームの数、サイズ、タイプは、インスタンスタイプとインスタンスサイズによって決まります。インスタンスストアボリュームのデータは、関連するインスタンスの存続中にのみ保持されます。インスタンスを停止、休止、または終了すると、インスタンスストアボリュームのすべてのデータが失われます。

Amazon EFS

Amazon EFS は、Amazon EC2 と併用できるスケーラブルなファイルストレージを提供します。EFS ファイルシステムを作成し、ファイルシステムをマウントするためにインスタンスを設定できます。複数のインスタンスで実行している作業負荷やアプリケーションの一般的なデータソースとしてEFSファイルシステムを使用できます。

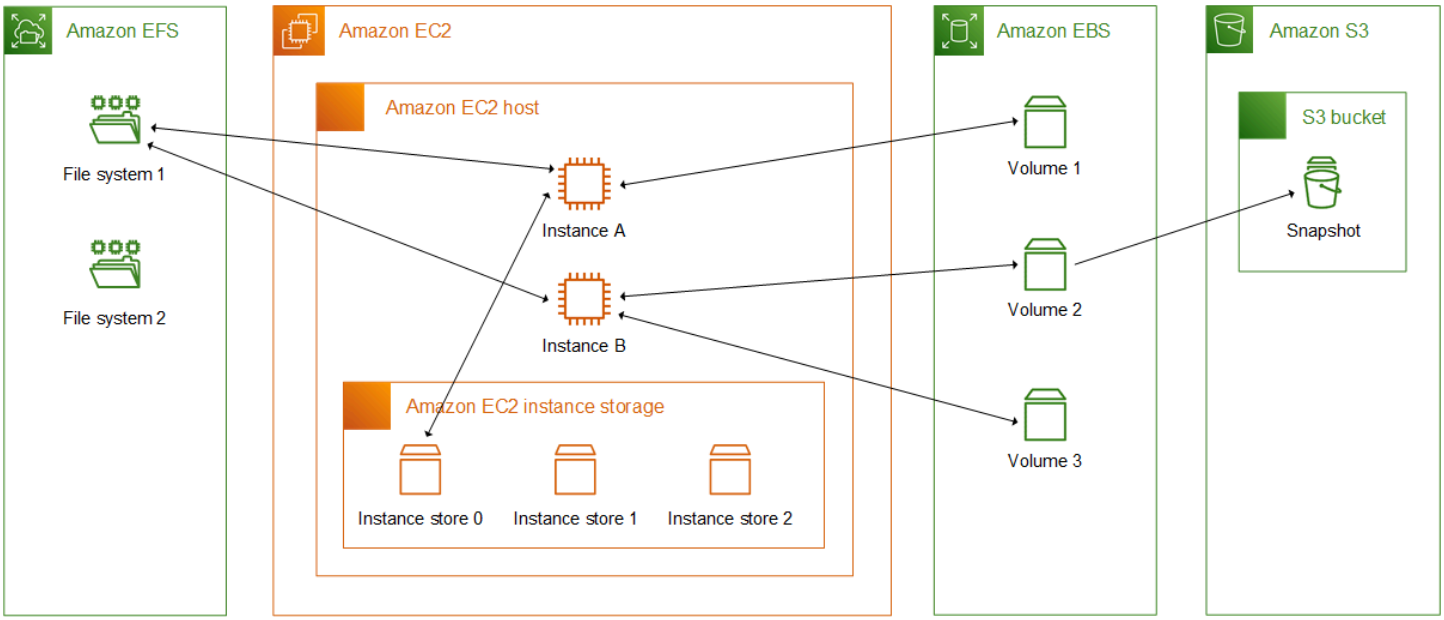
Amazon S3

Amazon S3 により、低コストで信頼性に優れたデータストレージインフラストラクチャが実現します。ウェブスケールのコンピューティングをさらに簡単に行えるように設計されており、Amazon EC2 内から、またはウェブ上のどこからでも、いつでも必要な量だけデータを格納および取得できます。例えば、Amazon S3 を使用して、データとアプリケーションのバックアップコピーを保存することができます。Amazon EC2 は、Amazon S3 を使用して EBS スナップショットと Instance Store-Backed AMI を保存します。

Amazon FSx

Amazon FSx を使用すると、多機能で高性能なファイルシステムをクラウドで起動、実行、およびスケールできます。Amazon FSx は、広範なワークロードをサポートするフルマネージド型サービスです。Lustre、NetApp ONTAP、OpenZFS、Windows File Server など、広く使用されているファイルシステムから選択できます。

各ストレージオプションとインスタンスの関係を下の図に示します。



ストレージ料金表

[AWS の料金] を開き、[AWS 製品の料金] までスクロールし、[ストレージ] を選択します。ストレージ製品を選択して、料金ページを開きます。

Amazon EC2 での Amazon EBS の使用

Amazon Elastic Block Store (Amazon EBS) は、Amazon Elastic Compute Cloud (Amazon EC2) インスタンスで使用できるスケーラブルな高性能ブロックストレージリソースを提供します。Amazon EBS では、次のブロックストレージリソースを作成および管理できます。

- Amazon EBS ボリューム - Amazon EC2 インスタンスにアタッチするストレージボリュームです。ボリュームをインスタンスにアタッチした後は、ブロックストレージを使用するのと同じ方法で使用できます。インスタンスはローカルドライブと同じようにボリュームとやり取りできます。
- Amazon EBS スナップショット - ボリューム自体とは独立して保持される Amazon EBS ボリュームのポイントインタイムバックアップです。スナップショットを作成して、Amazon EBS ボ

リュームのデータをバックアップできます。その後、それらのスナップショットからいつでも新しいボリュームを復元できます。

起動時に Amazon EBS ボリュームを作成してインスタンスにアタッチでき、起動後もいつでも EBS ボリュームを作成してインスタンスにアタッチできます。また、スナップショットはボリュームの作成後いつでも作成できます。

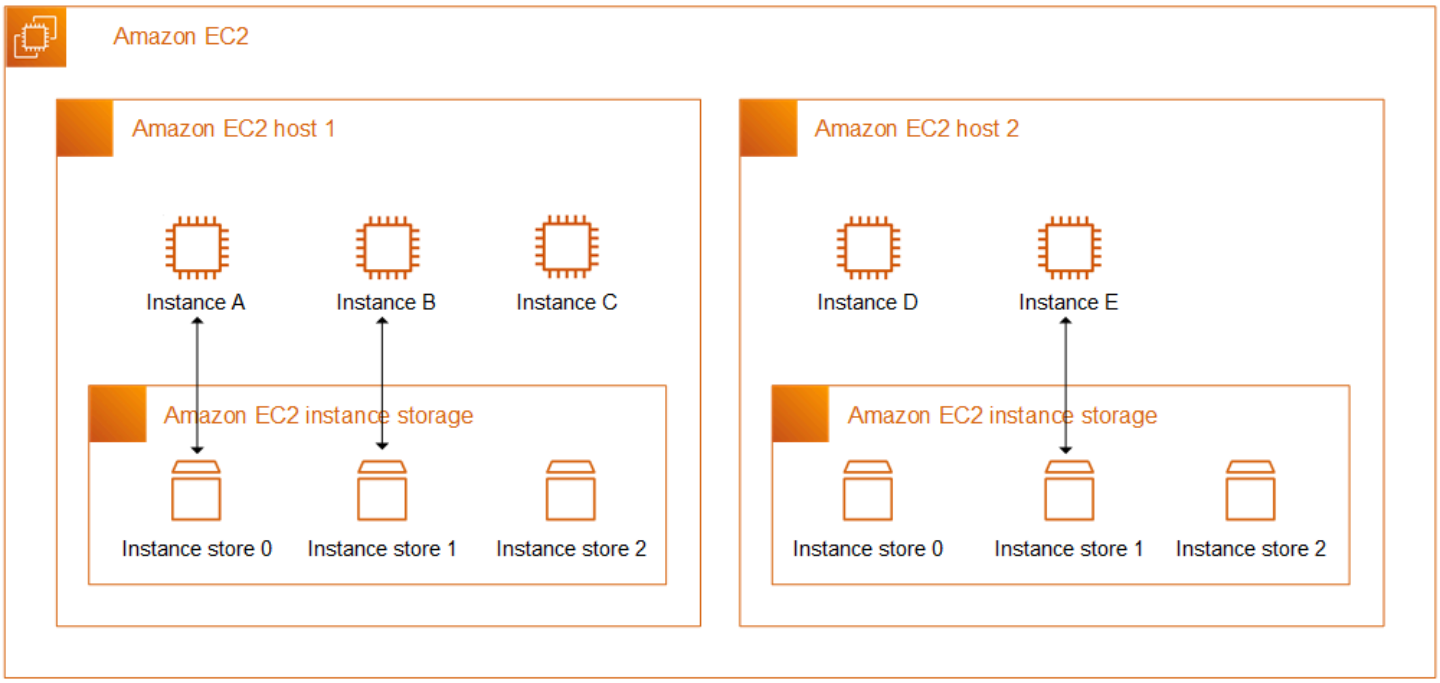
ボリュームとスナップショットの使用方法の詳細については、「Amazon EBS ユーザーガイド」を参照してください。

Amazon EC2 インスタンスストア

インスタンスストアは、インスタンス用のブロックレベルの一時ストレージを提供します。このストレージは、ホストコンピュータに物理的にアタッチされたディスク上にあります。インスタンスストアは、バッファ、キャッシュ、スクラッチデータ、その他の一時的データのように頻繁に変化する情報の一時的なストレージに最適です。また、負荷分散されたウェブサーバーのプールなど、インスタンスのフリート全体で複製する一時データを保存するためにも使用できます。

インスタンスストアは、ブロックデバイスとして表示される 1 つ以上のインスタンスストアボリュームで構成されます。インスタンスストアのサイズと、利用可能なデバイスの数は、インスタンスタイプおよびインスタンスサイズによって異なります。詳細については、「インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームの仮想デバイスは `ephemeral[0-23]` です。1 つのインスタンスストアボリュームをサポートするインスタンスタイプには、`ephemeral0` があります。2 つ以上のインスタンスストアボリュームをサポートするインスタンスタイプは、`ephemeral0`、`ephemeral1` などを持ちます。



インスタンスストアの価格

インスタンスストアボリュームは、インスタンスの使用料に含まれます。

コンテンツ

インスタンスストアボリュームとデータライフタイム

インスタンスストアボリュームの数、サイズ、タイプは、インスタンスタイプとインスタンスサイズによって決まります。詳細については、「インスタンスストアボリューム」を参照してください。

インスタンスストアボリュームは、インスタンスの起動時にのみアタッチされます。起動後にインスタンスストアボリュームをアタッチすることはできません。1 つのインスタンスからインスタンスストアをデタッチして別のインスタンスにアタッチすることはできません。

インスタンスストアボリュームは、アタッチされているインスタンスのライフタイム中にのみ存在します。インスタンスストアボリュームが、関連付けられたインスタンスのライフタイムを超えて維持されるように設定することはできません。

インスタンスストアボリューム上のデータは、インスタンスが再起動しても保持されます。ただし、インスタンスが停止、休止、終了するとデータは消滅します。インスタンスが停止、休止、終了した場合、インスタンスストアボリュームのすべてのブロックが暗号で消去されます。

このため、長期的に使用する重要なデータがある場合は、インスタンスストアに頼りすぎないようにしてください。インスタンスストアボリュームに保存されているデータを、インスタンスのライフタイムを超えて保持する必要がある場合は、そのデータを Amazon EBS ボリューム、Amazon S3 バケット、Amazon EFS ファイルシステムなどのより永続的なストレージに手動でコピーする必要があります。

イベントによっては、インスタンスのライフタイムを通じてデータが保持されなくなる場合があります。次の表は、仮想インスタンスとベアメタルインスタンスの両方について、特定のイベント中にインスタンスストアボリュームのデータが保持されるかどうかを示しています。

イベント	データはどうなりますか？
ユーザー主導のインスタンスライフサイクルイベント	
インスタンスが再起動されます。	The data persists
インスタンスが停止しました。	The data does not persist
インスタンスが休止しました。	The data does not persist
インスタンスが終了しました。	The data does not persist
インスタンスタイプが変更されます。	The data does not persist *
Windows AMI はインスタンスから作成されます。	The data does not persist in the created AMI **
EBS-backed AMI はインスタンスから作成されます。	The data does not persist in the created AMI **
Instance Store-Backed AMI はインスタンスから作成されます。	The data persists in the AMI bundle uploaded to Amazon S3 ***

イベント	データはどうなりますか？
ユーザー主導の OS イベント	
A shutdown is initiated	The data does not persist †
A restart is initiated	The data persists
AWS で予定されているイベント	
インスタンスの停止	The data does not persist
インスタンスの再起動	The data persists
システムの再起動	The data persists
インスタンスのリタイア	The data does not persist
想定外のイベント	
簡易自動復旧	The data does not persist
CloudWatch アクションに基づく復旧	The data does not persist
The underlying disk fails	The data on the failed disk does not persist
Power failure	The data persists upon reboot

* 新しいインスタンスタイプがインスタンスストアをサポートしている場合、インスタンスは新しいインスタンスタイプがサポートしているインスタンスストアボリュームの数を取得しますが、データは新しいインスタンスに転送されません。新しいインスタンスタイプがインスタンスストアをサポートしていない場合、インスタンスは、インスタンスストアボリュームを取得しません。

** データは EBS-backed AMI には含まれず、その AMI から起動されたインスタンスにアタッチされたインスタンスストアボリュームにも含まれません。

*** データは、Amazon S3 にアップロードされる AMI バンドルに含まれます。その AMI からインスタンスを起動すると、インスタンスは、AMI の作成時に含まれていたデータとともに AMI にバンドルされたインスタンスストアボリュームを取得します。

† 終了保護と停止保護は、インスタンスのオペレーティングシステムを通じて開始したシャットダウンの結果、インスタンスが停止または終了することに対してインスタンスを保護しません。インスタンスストアボリュームに保存されたデータは、インスタンスの停止イベントと終了イベントの両方で保持されません。

インスタンスストアボリューム

インスタンスストアボリュームの数、サイズ、タイプは、インスタンスタイプとインスタンスサイズによって決まります。M6、C6、R6 などの一部のインスタンスタイプはインスタンスストアボリュームをサポートしていませんが、M5d、C6gd、R6gd などのその他のインスタンスタイプはインスタンスストアボリュームをサポートしています。1 つのインスタンスに、そのインスタンスタイプでサポートされる量を超えるインスタンスストアボリュームをアタッチすることはできません。インスタンスストアボリュームをサポートするインスタンスタイプの場合、インスタンスストアボリュームの数とサイズは、インスタンスサイズによって異なります。例えば、`m5d.large` は 1 x 75 GB のインスタンスストアボリュームをサポートし、`m5d.24xlarge` は 4 x 900 GB のインスタンスストアボリュームをサポートします。

NVMe インスタンスストアボリュームを使用するインスタンスタイプでは、サポートされているすべてのインスタンスストアボリュームが、起動時に自動的にインスタンスにアタッチされます。C1、C3、M1、M2、M3、R3、D2、H1、I2、G2、X1、X1e など、NVMe 以外のインスタンスストアボリュームのインスタンスタイプでは、起動時にアタッチするインスタンスストアボリュームのブロックデバイスマッピングを手動で指定する必要があります。次に、インスタンスが起動したら、アタッチされたインスタンスストアボリュームを使用する前に、フォーマットしてマウントする必要があります。インスタンスの起動後にインスタンスストアボリュームをアタッチすることはできません。

インスタンスタイプには、NVMe または SATA ベースのソリッドステートドライブ (SSD) を使用するものと、SATA ベースのハードディスクドライブ (HDD) を使用するものがあります。SSD は、極めて低いレイテンシーで高いランダム I/O パフォーマンスを提供しますが、インスタンスの終了時にデータを保持する必要はなく、フォールトトレラントアーキテクチャを活用できます。詳細については、「SSD インスタンスストアボリューム」を参照してください。

NVMe インスタンスストアボリューム、および一部の HDD インスタンスストアボリュームにあるデータは、その保存時に暗号化されます。詳細については、「Amazon EC2 でのデータ保護」を参照してください。

使用可能なインスタンスストアボリューム

次の表は、サポートされている各インスタンスタイプで使用できるインスタンスストアボリュームの数量、サイズ、タイプ、パフォーマンス最適化を示しています。

トピック

汎用

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
m1.small	1 x 160 GB	HDD	✓	
m1.medium	1 x 410 GB	HDD	✓	
m1.large	2 x 420 GB	HDD	✓	
m1.xlarge	4 x 420 GB	HDD	✓	
m2.xlarge	1 x 420 GB	HDD	✓	
m2.2xlarge	1 x 850 GB	HDD	✓	
m2.4xlarge	2 x 840 GB	HDD	✓	
m3.medium	1 x 4 GB	SSD	✓	
m3.large	1 x 32 GB	[SSD]	✓	
m3.xlarge	2 x 40 GB	[SSD]	✓	
m3.2xlarge	2 x 80 GB	[SSD]	✓	

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
m5ad.large	1 x 75 GB	NVMe SSD		✓
m5ad.xlarge	1 x 150 GB	NVMe SSD		✓
m5ad.2xlarge	1 x 300 GB	NVMe SSD		✓
m5ad.4xlarge	2 x 300 GB	NVMe SSD		✓
m5ad.8xlarge	2 x 600 GB	NVMe SSD		✓
m5ad.12xlarge	2 x 900 GB	NVMe SSD		✓
m5ad.16xlarge	4 x 600 GB	NVMe SSD		✓
m5ad.24xlarge	4 x 900 GB	NVMe SSD		✓
m5d.large	1 x 75 GB	NVMe SSD		✓
m5d.xlarge	1 x 150 GB	NVMe SSD		✓
m5d.2xlarge	1 x 300 GB	NVMe SSD		✓
m5d.4xlarge	2 x 300 GB	NVMe SSD		✓
m5d.8xlarge	2 x 600 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
m5d.12xlarge	2 x 900 GB	NVMe SSD		✓
m5d.16xlarge	4 x 600 GB	NVMe SSD		✓
m5d.24xlarge	4 x 900 GB	NVMe SSD		✓
m5d.metal	4 x 900 GB	NVMe SSD		✓
m5dn.large	1 x 75 GB	NVMe SSD		✓
m5dn.xlarge	1 x 150 GB	NVMe SSD		✓
m5dn.2xlarge	1 x 300 GB	NVMe SSD		✓
m5dn.4xlarge	2 x 300 GB	NVMe SSD		✓
m5dn.8xlarge	2 x 600 GB	NVMe SSD		✓
m5dn.12xlarge	2 x 900 GB	NVMe SSD		✓
m5dn.16xlarge	4 x 600 GB	NVMe SSD		✓
m5dn.24xlarge	4 x 900 GB	NVMe SSD		✓
m5dn.metal	4 x 900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
m6gd.medium	1 x 59 GB	NVMe SSD		✓
m6gd.large	1 x 118 GB	NVMe SSD		✓
m6gd.xlarge	1 x 237 GB	NVMe SSD		✓
m6gd.2xlarge	1 x 474 GB	NVMe SSD		✓
m6gd.4xlarge	1 x 950 GB	NVMe SSD		✓
m6gd.8xlarge	1 x 1900 GB	NVMe SSD		✓
m6gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
m6gd.16xlarge	2 x 1900 GB	NVMe SSD		✓
m6gd.metal	2 x 1900 GB	NVMe SSD		✓
m6id.large	1 x 118 GB	NVMe SSD		✓
m6id.xlarge	1 x 237 GB	NVMe SSD		✓
m6id.2xlarge	1 x 474 GB	NVMe SSD		✓
m6id.4xlarge	1 x 950 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
m6id.8xlarge	1 x 1900 GB	NVMe SSD		✓
m6id.12xlarge	2 x 1425 GB	NVMe SSD		✓
m6id.16xlarge	2 x 1900 GB	NVMe SSD		✓
m6id.24xlarge	4 x 1425 GB	NVMe SSD		✓
m6id.32xlarge	4 x 1900 GB	NVMe SSD		✓
m6id.metal	4 x 1900 GB	NVMe SSD		✓
m6idn.large	1 x 118 GB	NVMe SSD		✓
m6idn.xlarge	1 x 237 GB	NVMe SSD		✓
m6idn.2xlarge	1 x 474 GB	NVMe SSD		✓
m6idn.4xlarge	1 x 950 GB	NVMe SSD		✓
m6idn.8xlarge	1 x 1900 GB	NVMe SSD		✓
m6idn.12xlarge	2 x 1425 GB	NVMe SSD		✓
m6idn.16xlarge	2 x 1900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
m6idn.24xlarge	4 x 1425 GB	NVMe SSD		✓
m6idn.32xlarge	4 x 1900 GB	NVMe SSD		✓
m6idn.metal	4 x 1900 GB	NVMe SSD		✓
m7gd.medium	1 x 59 GB	NVMe SSD		✓
m7gd.large	1 x 118 GB	NVMe SSD		✓
m7gd.xlarge	1 x 237 GB	NVMe SSD		✓
m7gd.2xlarge	1 x 474 GB	NVMe SSD		✓
m7gd.4xlarge	1 x 950 GB	NVMe SSD		✓
m7gd.8xlarge	1 x 1900 GB	NVMe SSD		✓
m7gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
m7gd.16xlarge	2 x 1900 GB	NVMe SSD		✓
m7gd.metal	2 x 1900 GB	NVMe SSD		✓

コンピューティングの最適化

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
c1.medium	1 x 350 GB	HDD	✓	
c1.xlarge	4 x 420 GB	HDD	✓	
c3.large	2 x 16 GB	[SSD]	✓	
c3.xlarge	2 x 40 GB	[SSD]	✓	
c3.2xlarge	2 x 80 GB	[SSD]	✓	
c3.4xlarge	2 x 160 GB	[SSD]	✓	
c3.8xlarge	2 x 320 GB	[SSD]	✓	
c5ad.large	1 x 75 GB	NVMe SSD		✓
c5ad.xlarge	1 x 150 GB	NVMe SSD		✓
c5ad.2xlarge	1 x 300 GB	NVMe SSD		✓
c5ad.4xlarge	2 x 300 GB	NVMe SSD		✓
c5ad.8xlarge	2 x 600 GB	NVMe SSD		✓
c5ad.12xlarge	2 x 900 GB	NVMe SSD		✓
c5ad.16xlarge	2 x 1200 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
c5ad.24xlarge	2 x 1900 GB	NVMe SSD		✓
c5d.large	1 x 50 GB	NVMe SSD		✓
c5d.xlarge	1 x 100 GB	NVMe SSD		✓
c5d.2xlarge	1 x 200 GB	NVMe SSD		✓
c5d.4xlarge	1 x 400 GB	NVMe SSD		✓
c5d.9xlarge	1 x 900 GB	NVMe SSD		✓
c5d.12xlarge	2 x 900 GB	NVMe SSD		✓
c5d.18xlarge	2 x 900 GB	NVMe SSD		✓
c5d.24xlarge	4 x 900 GB	NVMe SSD		✓
c5d.metal	4 x 900 GB	NVMe SSD		✓
c6gd.medium	1 x 59 GB	NVMe SSD		✓
c6gd.large	1 x 118 GB	NVMe SSD		✓
c6gd.xlarge	1 x 237 GB	NVMe SSD		✓
c6gd.2xlarge	1 x 474 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
c6gd.4xlarge	1 x 950 GB	NVMe SSD		✓
c6gd.8xlarge	1 x 1900 GB	NVMe SSD		✓
c6gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
c6gd.16xlarge	2 x 1900 GB	NVMe SSD		✓
c6gd.metal	2 x 1900 GB	NVMe SSD		✓
c6id.large	1 x 118 GB	NVMe SSD		✓
c6id.xlarge	1 x 237 GB	NVMe SSD		✓
c6id.2xlarge	1 x 474 GB	NVMe SSD		✓
c6id.4xlarge	1 x 950 GB	NVMe SSD		✓
c6id.8xlarge	1 x 1900 GB	NVMe SSD		✓
c6id.12xlarge	2 x 1425 GB	NVMe SSD		✓
c6id.16xlarge	2 x 1900 GB	NVMe SSD		✓
c6id.24xlarge	4 x 1425 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
c6id.32xlarge	4 x 1900 GB	NVMe SSD		✓
c6id.metal	4 x 1900 GB	NVMe SSD		✓
c7gd.medium	1 x 59 GB	NVMe SSD		✓
c7gd.large	1 x 118 GB	NVMe SSD		✓
c7gd.xlarge	1 x 237 GB	NVMe SSD		✓
c7gd.2xlarge	1 x 474 GB	NVMe SSD		✓
c7gd.4xlarge	1 x 950 GB	NVMe SSD		✓
c7gd.8xlarge	1 x 1900 GB	NVMe SSD		✓
c7gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
c7gd.16xlarge	2 x 1900 GB	NVMe SSD		✓
c7gd.metal	2 x 1900 GB	NVMe SSD		✓

メモリ最適化

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
r3.large	1 x 32 GB	SSD	✓	
r3.xlarge	1 x 80 GB	SSD	✓	
r3.2xlarge	1 x 160 GB	SSD	✓	
r3.4xlarge	1 x 320 GB	[SSD]	✓	
r3.8xlarge	2 x 320 GB	[SSD]	✓	
r5ad.large	1 x 75 GB	NVMe SSD		✓
r5ad.xlarge	1 x 150 GB	NVMe SSD		✓
r5ad.2xlarge	1 x 300 GB	NVMe SSD		✓
r5ad.4xlarge	2 x 300 GB	NVMe SSD		✓
r5ad.8xlarge	2 x 600 GB	NVMe SSD		✓
r5ad.12xlarge	2 x 900 GB	NVMe SSD		✓
r5ad.16xlarge	4 x 600 GB	NVMe SSD		✓
r5ad.24xlarge	4 x 900 GB	NVMe SSD		✓
r5d.large	1 x 75 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
r5d.xlarge	1 x 150 GB	NVMe SSD		✓
r5d.2xlarge	1 x 300 GB	NVMe SSD		✓
r5d.4xlarge	2 x 300 GB	NVMe SSD		✓
r5d.8xlarge	2 x 600 GB	NVMe SSD		✓
r5d.12xlarge	2 x 900 GB	NVMe SSD		✓
r5d.16xlarge	4 x 600 GB	NVMe SSD		✓
r5d.24xlarge	4 x 900 GB	NVMe SSD		✓
r5d.metal	4 x 900 GB	NVMe SSD		✓
r5dn.large	1 x 75 GB	NVMe SSD		✓
r5dn.xlarge	1 x 150 GB	NVMe SSD		✓
r5dn.2xlarge	1 x 300 GB	NVMe SSD		✓
r5dn.4xlarge	2 x 300 GB	NVMe SSD		✓
r5dn.8xlarge	2 x 600 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
r5dn.12xlarge	2 x 900 GB	NVMe SSD		✓
r5dn.16xlarge	4 x 600 GB	NVMe SSD		✓
r5dn.24xlarge	4 x 900 GB	NVMe SSD		✓
r5dn.metal	4 x 900 GB	NVMe SSD		✓
r6gd.medium	1 x 59 GB	NVMe SSD		✓
r6gd.large	1 x 118 GB	NVMe SSD		✓
r6gd.xlarge	1 x 237 GB	NVMe SSD		✓
r6gd.2xlarge	1 x 474 GB	NVMe SSD		✓
r6gd.4xlarge	1 x 950 GB	NVMe SSD		✓
r6gd.8xlarge	1 x 1900 GB	NVMe SSD		✓
r6gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
r6gd.16xlarge	2 x 1900 GB	NVMe SSD		✓
r6gd.metal	2 x 1900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
r6idn.large	1 x 118 GB	NVMe SSD		✓
r6idn.xlarge	1 x 237 GB	NVMe SSD		✓
r6idn.2xlarge	1 x 474 GB	NVMe SSD		✓
r6idn.4xlarge	1 x 950 GB	NVMe SSD		✓
r6idn.8xlarge	1 x 1900 GB	NVMe SSD		✓
r6idn.12xlarge	2 x 1425 GB	NVMe SSD		✓
r6idn.16xlarge	2 x 1900 GB	NVMe SSD		✓
r6idn.24xlarge	4 x 1425 GB	NVMe SSD		✓
r6idn.32xlarge	4 x 1900 GB	NVMe SSD		✓
r6idn.metal	4 x 1900 GB	NVMe SSD		✓
r6id.large	1 x 118 GB	NVMe SSD		✓
r6id.xlarge	1 x 237 GB	NVMe SSD		✓
r6id.2xlarge	1 x 474 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
r6id.4xlarge	1 x 950 GB	NVMe SSD		✓
r6id.8xlarge	1 x 1900 GB	NVMe SSD		✓
r6id.12xlarge	2 x 1425 GB	NVMe SSD		✓
r6id.16xlarge	2 x 1900 GB	NVMe SSD		✓
r6id.24xlarge	4 x 1425 GB	NVMe SSD		✓
r6id.32xlarge	4 x 1900 GB	NVMe SSD		✓
r6id.metal	4 x 1900 GB	NVMe SSD		✓
r7gd.medium	1 x 59 GB	NVMe SSD		✓
r7gd.large	1 x 118 GB	NVMe SSD		✓
r7gd.xlarge	1 x 237 GB	NVMe SSD		✓
r7gd.2xlarge	1 x 474 GB	NVMe SSD		✓
r7gd.4xlarge	1 x 950 GB	NVMe SSD		✓
r7gd.8xlarge	1 x 1900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
r7gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
r7gd.16xlarge	2 x 1900 GB	NVMe SSD		✓
r7gd.metal	2 x 1900 GB	NVMe SSD		✓
x1.16xlarge	1 x 1920 GB	[SSD]	✓	
x1.32xlarge	2 x 1920 GB	[SSD]	✓	
x2gd.medium	1 x 59 GB	NVMe SSD		✓
x2gd.large	1 x 118 GB	NVMe SSD		✓
x2gd.xlarge	1 x 237 GB	NVMe SSD		✓
x2gd.2xlarge	1 x 475 GB	NVMe SSD		✓
x2gd.4xlarge	1 x 950 GB	NVMe SSD		✓
x2gd.8xlarge	1 x 1900 GB	NVMe SSD		✓
x2gd.12xlarge	2 x 1425 GB	NVMe SSD		✓
x2gd.16xlarge	2 x 1900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
x2gd.metal	2 x 1900 GB	NVMe SSD		✓
x2idn.16xlarge	1 x 1900 GB	NVMe SSD		✓
x2idn.24xlarge	2 x 1425 GB	NVMe SSD		✓
x2idn.32xlarge	2 x 1900 GB	NVMe SSD		✓
x2idn.metal	2 x 1900 GB	NVMe SSD		✓
x2iedn.xlarge	1 x 118 GB	NVMe SSD		✓
x2iedn.2xlarge	1 x 237 GB	NVMe SSD		✓
x2iedn.4xlarge	1 x 475 GB	NVMe SSD		✓
x2iedn.8xlarge	1 x 950 GB	NVMe SSD		✓
x2iedn.16xlarge	1 x 1900 GB	NVMe SSD		✓
x2iedn.24xlarge	2 x 1425 GB	NVMe SSD		✓
x2iedn.32xlarge	2 x 1900 GB	NVMe SSD		✓
x2iedn.metal	2 x 1900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
x1e.xlarge	1 x 120 GB	SSD	✓	
x1e.2xlarge	1 x 240 GB	SSD	✓	
x1e.4xlarge	1 x 480 GB	SSD	✓	
x1e.8xlarge	1 x 960 GB	[SSD]	✓	
x1e.16xlarge	1 x 1920 GB	[SSD]	✓	
x1e.32xlarge	2 x 1920 GB	[SSD]	✓	
z1d.large	1 x 75 GB	NVMe SSD		✓
z1d.xlarge	1 x 150 GB	NVMe SSD		✓
z1d.2xlarge	1 x 300 GB	NVMe SSD		✓
z1d.3xlarge	1 x 450 GB	NVMe SSD		✓
z1d.6xlarge	1 x 900 GB	NVMe SSD		✓
z1d.12xlarge	2 x 900 GB	NVMe SSD		✓
z1d.metal	2 x 900 GB	NVMe SSD		✓

ストレージの最適化

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
d2.xlarge	3 x 2048 GB	HDD	✓	
d2.2xlarge	6 x 2048 GB	HDD	✓	
d2.4xlarge	12 x 2048 GB	HDD	✓	
d2.8xlarge	24 x 2048 GB	HDD	✓	
d3.xlarge	3 x 1980 GB	NVMe HDD		✓
d3.2xlarge	6 x 1980 GB	NVMe HDD		✓
d3.4xlarge	12 x 1980 GB	NVMe HDD		✓
d3.8xlarge	24 x 1980 GB	NVMe HDD		✓
d3en.xlarge	2 x 13980 GB	NVMe HDD		✓
d3en.2xlarge	4 x 13980 GB	NVMe HDD		✓
d3en.4xlarge	8 x 13980 GB	NVMe HDD		✓
d3en.6xlarge	12 x 13980 GB	NVMe HDD		✓
d3en.8xlarge	16 x 13980 GB	NVMe HDD		✓
d3en.12xlarge	24 x 13980 GB	NVMe HDD		✓
h1.2xlarge	1 x 2000 GB	HDD	✓	

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
h1.4xlarge	2 x 2000 GB	HDD	✓	
h1.8xlarge	4 x 2000 GB	HDD	✓	
h1.16xlarge	8 x 2000 GB	HDD	✓	
i2.xlarge	1 x 800 GB	[SSD]	✓	
i2.2xlarge	2 x 800 GB	[SSD]	✓	
i2.4xlarge	4 x 800 GB	[SSD]	✓	
i2.8xlarge	8 x 800 GB	[SSD]	✓	
i3.large	1 x 475 GB	NVMe SSD		✓
i3.xlarge	1 x 950 GB	NVMe SSD		✓
i3.2xlarge	1 x 1900 GB	NVMe SSD		✓
i3.4xlarge	2 x 1900 GB	NVMe SSD		✓
i3.8xlarge	4 x 1900 GB	NVMe SSD		✓
i3.16xlarge	8 x 1900 GB	NVMe SSD		✓
i3.metal	8 x 1900 GB	NVMe SSD		✓
i3en.large	1 x 1250 GB	NVMe SSD		✓
i3en.xlarge	1 x 2500 GB	NVMe SSD		✓
i3en.2xlarge	2 x 2500 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
i3en.3xlarge	1 x 7500 GB	NVMe SSD		✓
i3en.6xlarge	2 x 7500 GB	NVMe SSD		✓
i3en.12xlarge	4 x 7500 GB	NVMe SSD		✓
i3en.24xlarge	8 x 7500 GB	NVMe SSD		✓
i3en.metal	8 x 7500 GB	NVMe SSD		✓
i4g.large	1 × 468 GB	NVMe SSD		✓
i4g.xlarge	1 × 937 GB	NVMe SSD		✓
i4g.2xlarge	1 x 1875 GB	NVMe SSD		✓
i4g.4xlarge	1 x 3750 GB	NVMe SSD		✓
i4g.8xlarge	2 x 3750 GB	NVMe SSD		✓
i4g.16xlarge	4 x 3750 GB	NVMe SSD		✓
i4i.large	1 × 468 GB	NVMe SSD		✓
i4i.xlarge	1 × 937 GB	NVMe SSD		✓
i4i.2xlarge	1 x 1875 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
i4i.4xlarge	1 x 3750 GB	NVMe SSD		✓
i4i.8xlarge	2 x 3750 GB	NVMe SSD		✓
i4i.12xlarge	3 x 3750 GB	NVMe SSD		✓
i4i.16xlarge	4 x 3750 GB	NVMe SSD		✓
i4i.24xlarge	6 x 3750 GB	NVMe SSD		✓
i4i.32xlarge	8 x 3750 GB	NVMe SSD		✓
i4i.metal	8 x 3750 GB	NVMe SSD		✓
im4gn.large	1 × 937 GB	NVMe SSD		✓
im4gn.xlarge	1 x 1875 GB	NVMe SSD		✓
im4gn.2xlarge	1 x 3750 GB	NVMe SSD		✓
im4gn.4xlarge	1 x 7500 GB	NVMe SSD		✓
im4gn.8xlarge	2 x 7500 GB	NVMe SSD		✓
im4gn.16xlarge	4 x 7500 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
is4gen.medium	1 × 937 GB	NVMe SSD		✓
is4gen.large	1 x 1875 GB	NVMe SSD		✓
is4gen.xlarge	1 x 3750 GB	NVMe SSD		✓
is4gen.2xlarge	1 x 7500 GB	NVMe SSD		✓
is4gen.4xlarge	2 x 7500 GB	NVMe SSD		✓
is4gen.8xlarge	4 x 7500 GB	NVMe SSD		✓

高速コンピューティング

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
dl1.24xlarge	4 x 1000 GB	NVMe SSD		✓
f1.2xlarge	1 x 470 GB	NVMe SSD		✓
f1.4xlarge	1 x 940 GB	NVMe SSD		✓
f1.16xlarge	4 x 940 GB	NVMe SSD		✓
g2.2xlarge	1 x 60 GB	[SSD]	✓	

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
g2.8xlarge	2 x 120 GB	[SSD]	✓	
g4ad.xlarge	1 x 150 GB	NVMe SSD		✓
g4ad.2xlarge	1 x 300 GB	NVMe SSD		✓
g4ad.4xlarge	1 x 600 GB	NVMe SSD		✓
g4ad.8xlarge	1 x 1200 GB	NVMe SSD		✓
g4ad.16xlarge	2 x 1200 GB	NVMe SSD		✓
g4dn.xlarge	1 x 125 GB	NVMe SSD		✓
g4dn.2xlarge	1 x 225 GB	NVMe SSD		✓
g4dn.4xlarge	1 x 225 GB	NVMe SSD		✓
g4dn.8xlarge	1 x 900 GB	NVMe SSD		✓
g4dn.12xlarge	1 x 900 GB	NVMe SSD		✓
g4dn.16xlarge	1 x 900 GB	NVMe SSD		✓
g4dn.metal	2 x 900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
g5.xlarge	1 x 250 GB	NVMe SSD		✓
g5.2xlarge	1 x 450 GB	NVMe SSD		✓
g5.4xlarge	1 x 600 GB	NVMe SSD		✓
g5.8xlarge	1 x 900 GB	NVMe SSD		✓
g5.12xlarge	1 x 3800 GB	NVMe SSD		✓
g5.16xlarge	1 x 1900 GB	NVMe SSD		✓
g5.24xlarge	1 x 3800 GB	NVMe SSD		✓
g5.48xlarge	2 x 3800 GB	NVMe SSD		✓
p3dn.24xlarge	2 x 900 GB	NVMe SSD		✓
p4d.24xlarge	8 x 1000 GB	NVMe SSD		✓
p4de.24xlarge	8 x 1000 GB	NVMe SSD		✓
p5.48xlarge	8 x 3800 GB	NVMe SSD		✓
trn1.2xlarge	1 x 474 GB	NVMe SSD		✓
trn1.32xlarge	4 x 1900 GB	NVMe SSD		✓

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
trn1n.32xlarge	4 x 1900 GB	NVMe SSD		✓

高性能コンピューティング

インスタンスタイプ	インスタンスストアボリューム	タイプ	初期化が必要*	TRIM サポート**
hpc6id.32xlarge	4 x 3800 GB	NVMe SSD		✓

* 特定のインスタンスにアタッチされたボリュームは、初期化されないと初回書き込み時のパフォーマンスが低下します。詳細については、インスタンスストアボリュームのディスクパフォーマンスの最適化 をご参照ください。

** 詳細については、インスタンスストアボリュームの TRIM のサポートを参照してください。

† c1.medium および m1.small インスタンスタイプには、900 MB のインスタンスストアスワップボリュームも含まれます。これは起動時に自動的に有効にならない場合があります。詳細については、「インスタンスストアスワップボリューム」を参照してください。

インスタンスストアボリュームのパフォーマンス

次のドキュメントは、インスタンスストアボリュームの I/O パフォーマンスについて説明します。

- 汎用インスタンス
- コンピュート最適化インスタンス
- メモリ最適化インスタンス
- ストレージ最適化インスタンス
- 高速コンピューティングインスタンス

AWS CLI を使用してインスタンスストアボリューム情報をクエリするには

describe-instance-types AWS CLI コマンドを使用して、インスタンスストアボリュームなど、インスタンスタイプに関する情報を表示できます。次の例では、インスタンスストアボリュームを持つすべての R5 インスタンスのインスタンスストレージの合計サイズを表示します。

```
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=r5*" "Name=instance-storage-
supported,Values=true" \
    --query "InstanceTypes[].[InstanceType, InstanceStorageInfo.TotalSizeInGB]" \
    --output table
```

出力例

```
--------------------------
|  DescribeInstanceTypes  |
+----------------+--------+
|  r5ad.24xlarge |  3600  |
|  r5ad.12xlarge |  1800  |
|  r5dn.8xlarge  |  1200  |
|  r5ad.8xlarge  |  1200  |
|  r5ad.large    |  75    |
|  r5d.4xlarge   |  600   |
   . . .
|  r5dn.2xlarge  |  300   |
|  r5d.12xlarge  |  1800  |
+----------------+--------+
```

次の例では、指定されたインスタンスタイプの完全なインスタンスストレージの詳細を表示します。

```
aws ec2 describe-instance-types \
    --filters "Name=instance-type,Values=r5d.4xlarge" \
    --query "InstanceTypes[].InstanceStorageInfo"
```

出力例は、このインスタンスタイプに 300 GB の NVMe SSD ボリュームが 2 つあり、合計 600 GB のインスタンスストレージがあることを示しています。

```
[
    {
        "TotalSizeInGB": 600,
        "Disks": [
            {
                "SizeInGB": 300,
```

```
                "Count": 2,
                "Type": "ssd"
            }
        ],
        "NvmeSupport": "required"
    }
]
```

EC2 インスタンスにインスタンスストアボリュームを追加する

NVMe インスタンスストアボリュームを使用するインスタンスタイプでは、サポートされているすべてのインスタンスストアボリュームが、起動時に自動的にインスタンスにアタッチされます。NVMe インスタンスストアボリュームは、インスタンスの起動時に自動的に列挙され、デバイス名が割り当てられます。

C1、C3、M1、M2、M3、R3、D2、H1、I2、G2、X1、X1e など、NVMe 以外のインスタンスストアボリュームのインスタンスタイプでは、起動時にアタッチするインスタンスストアボリュームのブロックデバイスマッピングを手動で指定する必要があります。ブロックデバイスマッピングは、インスタンス起動リクエストで、またはインスタンスの起動に使用される AMI で指定できます。ブロックデバイスマッピングには、デバイス名とそれがマッピングされたボリュームが含まれます。詳細については、「ブロックデバイスマッピング」を参照してください。

> **Important**
>
> インスタンスを起動する場合にのみ、インスタンスストアボリュームをインスタンスにアタッチできます。また、起動後のインスタンスにインスタンスストアボリュームをアタッチすることはできません。

インスタンスを起動したら、使用する前に、インスタンスのインスタンスストアボリュームがフォーマットされ、マウントされていることを確認する必要があります。instance store-backed インスタンスのルートボリュームは自動的にマウントされます。

ルートボリュームに関する考慮事項

ブロックデバイスマッピングでは、常にインスタンスのルートボリュームを指定します。ルートボリュームは、Amazon EBS ボリュームまたはインスタンスストアボリュームのいずれかです。ルートボリュームは自動的にマウントされます。ルートボリュームのインスタンスストアボリュームを持つインスタンスの場合、このボリュームのサイズは AMI によって異なりますが、最大サイズは 10 GB です。詳細については、「ルートデバイスのストレージ」を参照してください。

コンテンツ

AMI へのインスタンスストアボリュームの追加

インスタンスストアボリュームが含まれる、ブロックデバイスマッピングを持つ AMI を作成できます。

インスタンスストアボリュームブロックデバイスマッピングを指定する AMI を使用して、非 NVMe のインスタンスストアボリュームをサポートするインスタンスを起動すると、インスタンスにインスタンスストアボリュームが含まれます。AMI のインスタンスストアボリュームブロックデバイスマッピングの数がインスタンスに利用できるインスタンスストアボリュームの数を超えた場合、追加のインスタンスストアボリュームブロックデバイスマッピングは無視されます。

インスタンスストアボリュームブロックデバイスマッピングを指定する AMI を使用して、NVMe インスタンスストアボリュームをサポートするインスタンスを起動した場合、インスタンスストアボリュームブロックデバイスマッピングは無視されます。NVMe インスタンスストアボリュームをサポートするインスタンスは、インスタンス起動リクエストと AMI で指定されたブロックデバイスマッピングに関らず、サポートされているすべてのインスタンスストアボリュームを取得します。

考慮事項

- M3 インスタンスの場合は、AMI ではなく、インスタンスのブロックデバイスマッピングにあるインスタンスストアボリュームを指定します。Amazon EC2 は、AMI のインスタンスストアボリュームブロックデバイスマッピングを無視することがあります。
- インスタンスを起動する際に、AMI ブロックデバイスマッピングで指定された 非 NVMe インスタンスストアボリュームを省略したり、インスタンスストアボリュームを追加したりできます。

New console

コンソールを使用して Amazon EBS-backed AMI にインスタンスストアボリュームを追加するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。

3. [アクション]、[Image and templates (イメージとテンプレート)]、[イメージの作成] の順に選択します。
4. [イメージの作成] ページで、イメージの意味のある名前と説明を追加します。
5. 追加する各インスタンスストアボリュームについて、[ボリュームの追加] を選択し、[ボリュームタイプ] からインスタンスストアボリュームを選択して、[デバイス] からデバイス名を選択します。(詳細については、Linux インスタンスでのデバイス名 を参照してください)。使用できるインスタンスストアボリュームの数は、インスタンスタイプによって異なります。NVMe インスタンスストアボリュームを持つインスタンスの場合、これらのボリュームのデバイスマッピングは、オペレーティングシステムがこれらのボリュームを列挙する順序によって決まります。
6. [イメージを作成] を選択します。

AWS CLI

コマンドラインを使用して AMI にインスタンスストアボリュームを追加するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- create-image または register-image (AWS CLI)
- New-EC2Image および Register-EC2Image (AWS Tools for Windows PowerShell)

インスタンスに非 NVMe インスタンスストアボリュームを追加する

非 NVMe インスタンスストアボリュームをサポートするインスタンスを起動するときは、アタッチするインスタンスストアボリュームのブロックデバイスマッピングを指定する必要があります。ブロックデバイスマッピングは、インスタンス起動リクエストで、またはインスタンスの起動に使用される AMI で指定する必要があります。

AMI にインスタンスストアボリュームのブロックデバイスマッピングが含まれている場合、AMI に含まれるよりも多くのインスタンスストアボリュームが必要でない限り、インスタンス起動リクエストでブロックデバイスマッピングを指定する必要はありません。

AMI にインスタンスストアボリュームのブロックデバイスマッピングが含まれていない場合は、インスタンス起動リクエストでブロックデバイスマッピングを指定する必要があります。

考慮事項

- M3 インスタンスの場合は、インスタンスのブロックデバイスマッピングで指定しなくても、インスタンスストアボリュームを受け取る可能性があります。

インスタンス起動リクエストでブロックデバイスマッピングを指定するには、次のいずれかの方法を使用します。

Amazon EC2 console

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ダッシュボードから、[インスタンスの作成] を選択します。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、使用する AMI を選択します。
4. [ストレージの設定] セクションの [インスタンスストアボリューム] セクションには、インスタンスにアタッチできるインスタンスストアボリュームが一覧表示されます。使用できるインスタンスストアボリュームの数は、インスタンスタイプによって異なります。
5. アタッチする各インスタンスストアボリュームの [デバイス名] で、使用するデバイス名を選択します。
6. 必要に応じて残りのインスタンスの設定を設定し、[インスタンスの起動] を選択します。

Command line

次のいずれかのオプションコマンドを、対応するオプションで使用できます。

- `--block-device-mappings` と run-instances (AWS CLI)
- `-BlockDeviceMapping` と New-EC2Instance (AWS Tools for Windows PowerShell)

インスタンスでインスタンスストアボリュームを使用できるようにする

インスタンスストアボリュームが接続しているインスタンスを起動したら、アクセスする前にボリュームをマウントする必要があります。

Linux インスタンスでは、多くのインスタンスストアボリュームは ext3 ファイルシステムで事前にフォーマットされています。TRIM コマンドをサポートする SSD ベースのインスタンスストアボリュームは、ファイルシステムを使用して事前にフォーマットされていません。ただし、インスタンスを起動してから、選択したファイルシステムでボリュームをフォーマットすることもで

きます。詳細については、「インスタンスストアボリュームの TRIM のサポート」を参照してください。Windows インスタンスでは、NTFS ファイルシステムでインスタンスストアボリュームをフォーマットします。

インスタンスストアデバイスが使用できるかどうかは、インスタンスメタデータを使用してインスタンスの内部から確認できます。詳細については、インスタンスストアボリュームのインスタンスブロックデバイスマッピングの表示を参照してください。

Linux インスタンスの場合、次の手順で説明されているように、インスタンスストアボリュームを表示してマウントできます。

Linux でインスタンスストアボリュームを使用できるようにするには

1. SSH クライアントを使用してインスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。
2. df -h コマンドを使用して、フォーマットおよびマウントされたボリュームを表示します。

```
$ df -h
Filesystem      Size  Used Avail Use% Mounted on
devtmpfs        3.8G   72K  3.8G   1% /dev
tmpfs           3.8G     0  3.8G   0% /dev/shm
/dev/nvme0n1p1  7.9G  1.2G  6.6G  15% /
```

3. lsblk を使用して、起動時にマッピングされたが、フォーマットおよびマウントされていないボリュームを表示します。

```
$ lsblk
NAME          MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
nvme0n1       259:1    0    8G  0 disk
##nvme0n1p1   259:2    0    8G  0 part /
##nvme0n1p128 259:3    0    1M  0 part
nvme1n1       259:0    0 69.9G  0 disk
```

4. マッピングされたのみのインスタンスストアボリュームをフォーマットしてマウントするには、以下の作業を行います。

   a. mkfs コマンドを使用してデバイスでファイルシステムを作成します。

   ```
   $ sudo mkfs -t xfs /dev/nvme1n1
   ```

   b. mkdir コマンドを使用してデバイスをマウントするディレクトリを作成します。

```
$ sudo mkdir /data
```

c. mount コマンドを使用して、新しく作成されたディレクトリにデバイスをマウントします。

```
$ sudo mount /dev/nvme1n1 /data
```

再起動後にアタッチされたボリュームを自動的にマウントする方法については、「Amazon EBS ユーザーガイド」の「再起動後に接続ボリュームを自動的にマウントする」を参照してください。

SSD インスタンスストアボリューム

Linux で SSD インスタンスストアボリュームから最高の IOPS 性能を得るには、最新バージョンの Amazon Linux、またはカーネルバージョン 3.8 以降の別の Linux AMI を使用することをお勧めします。カーネルバージョン 3.8 以降の Linux AMI を使用しないと、これらのインスタンスタイプで最大可能な IOPS パフォーマンスはインスタンスで実現されません。

他のインスタンスストアボリュームと同様に、インスタンスの SSD インスタンスストアボリュームを起動するときにマップする必要があります。SSD インスタンスボリューム上のデータは、関連するインスタンスの存続期間中のみ維持されます。詳細については、EC2 インスタンスにインスタンスストアボリュームを追加するを参照してください。

NVMe SSD ボリューム

インスタンスによっては、Non-Volatile Memory Express (NVMe) ソリッドステートドライブ (SSD) インスタンスストアボリュームを提供するものもあります。各インスタンスタイプによりサポートされるインスタンスストアボリュームのタイプの詳細については、インスタンスストアボリュームを参照してください。

NVMe ボリュームにアクセスするには、NVMe ドライバーをインストールする必要があります。以下の AMI はこの要件を満たしています。

- AL2023
- Amazon Linux 2
- Amazon Linux AMI 2018.03 以降
- `linux-aws` カーネルを搭載した Ubuntu 14.04 以降

> **Note**
>
> AWS Graviton ベースのインスタンスタイプには、linux-aws カーネル搭載の Ubuntu 18.04 以降が必要です

- Red Hat Enterprise Linux 7.4 以降
- SUSE Linux Enterprise Server 12 SP2 以降
- CentOS 7.4.1708 以降
- FreeBSD 11.1 以降
- Debian GNU/Linux 9 以降

- Bottlerocket

インスタンスに接続したら、lspci コマンドを使用して NVMe デバイスをリストできます。次に示すのは、4 つの NVMe デバイスをサポートする i3.8xlarge インスタンスの出力例です。

```
[ec2-user ~]$ lspci
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 01)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:03.0 Ethernet controller: Device 1d0f:ec20
00:17.0 Non-Volatile memory controller: Device 1d0f:cd01
00:18.0 Non-Volatile memory controller: Device 1d0f:cd01
00:19.0 Non-Volatile memory controller: Device 1d0f:cd01
00:1a.0 Non-Volatile memory controller: Device 1d0f:cd01
00:1f.0 Unassigned class [ff80]: XenSource, Inc. Xen Platform Device (rev 01)
```

サポートされているオペレーティングシステムを使用しているが、NVMe デバイスが表示されない場合は、次のコマンドを使用して NVMe モジュールが読み込まれていることを確認します。

- Amazon Linux、Amazon Linux 2、Ubuntu 14/16、Red Hat Enterprise Linux、SUSE Linux Enterprise Server、CentOS 7

```
$ lsmod | grep nvme
nvme           48813  0
```

- Ubuntu 18

```
$ cat /lib/modules/$(uname -r)/modules.builtin | grep nvme
s/nvme/host/nvme-core.ko
kernel/drivers/nvme/host/nvme.ko
kernel/drivers/nvmem/nvmem_core.ko
```

NVMe ボリュームは NVMe 1.0e 仕様に準拠しています。NVMe コマンドは NVMe ボリュームで使用できます。Amazon Linux では、nvme-cli コマンドを使用して repo から yum install パッケージをインストールできます。サポートされているバージョンの Linux では、イメージで利用可能でない場合は nvme-cli パッケージをダウンロードできます。

NVMe インスタンスストレージのデータは、インスタンスのハードウェアモジュールに実装されている XTS-AES-256 ブロック暗号を使用して暗号化されます。暗号化キーは、ハードウェアモジュールで作成され、NVMe インスタンスストレージデバイスごとに固有です。すべての暗号化キーは、インスタンスが停止または終了して復元できないときに破棄されます。この暗号化を無効にしたり、独自の暗号キーを指定したりすることはできません。

非 NVMe SSD ボリューム

C3、G2、I2、M3、R3、および X1 の各インスタンスは、非 NVMe SSD を使用するインスタンスストアボリュームをサポートすることで、高いランダム I/O パフォーマンスを実現しています。各インスタンスタイプによりサポートされるインスタンスストアボリュームの詳細については、インスタンスストアボリュームを参照してください。

インスタンスストアボリュームの TRIM のサポート

一部のインスタンスタイプでは、TRIM を持つ SSD ボリュームがサポートされます。詳細については、インスタンスストアボリュームを参照してください。

TRIM をサポートしているインスタンスストアボリュームは、インスタンスに割り当てられる前に完全に TRIM が実行されます。これらのボリュームは、インスタンスの起動時にファイルシステムを使用してフォーマットされないため、マウントして使用する前にボリュームをフォーマットする必要があります。これらのボリュームを迅速に使用できるようにするには、ボリュームをフォーマットするときに、TRIM 操作をスキップします。

TRIM をサポートするインスタンスストアボリュームでは、TRIM コマンドを使用して、書き込んだデータが不要になったときに SSD コントローラーに通知することができます。これにより、より多

くの空き領域がコントローラーに与えられ、その結果書き込み増幅が減り、パフォーマンスが向上します。Linux では、`fstrim` コマンドを使用して、定期的な TRIM を有効にします。

インスタンスストアスワップボリューム

Linux のスワップ空間は、システムで物理的に割り当てられたよりも多くのメモリを必要とする場合に使用できます。スワップ空間を有効にすると、Linux システムは頻繁に使用されないメモリページを物理メモリからスワップ空間 (既存のファイルシステムの専用パーティションまたはスワップファイル) にスワップし、高速なアクセスを必要とするメモリページのためにその空間を解放します。

> **Note**
>
> スワップ空間をメモリページングに使用しても、RAM 使用時ほど高速でも効率的でもありません。スワップ空間に定期的にメモリをページングするワークロードの場合は、RAM が多くサイズの大きいインスタンスタイプに移行することを検討してください。詳細については、インスタンスタイプを変更するを参照してください。

`c1.medium` および `m1.small` インスタンスタイプの物理メモリ容量は限られており、起動時には Linux AMIs の仮想メモリとして機能する 900 MiB スワップボリュームが与えられます。Linux カーネルはこのスワップ領域をルートデバイス上のパーティションとして認識しますが、ルートデバイスのタイプに関係なく、実際には別のインスタンスストアボリュームです。

Amazon Linux は自動的にこのスワップ空間を有効にして使用しますが、AMI では、このスワップ空間を認識して使用するために、追加のステップが必要になる場合があります。インスタンスがスワップ空間を使用しているかどうか確認するには、swapon -s コマンドを使用できます。

```
[ec2-user ~]$ swapon -s
Filename                                Type            Size    Used    Priority
/dev/xvda3                              partition       917500  0       -1
```

上記のインスタンスには、900 MiB のスワップボリュームがアタッチされ、有効になっています。このコマンドでスワップボリュームが表示されない場合は、そのデバイスに対してスワップ空間を有効しなければならない可能性があります。利用可能なディスクは、lsblk コマンドを使用して確認します。

```
[ec2-user ~]$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda1 202:1    0    8G  0 disk /
```

```
xvda3 202:3    0  896M  0 disk
```

ここで、スワップボリューム xvda3 はインスタンスで利用できますが、有効になっていません (MOUNTPOINT フィールドが空です)。スワップボリュームは swapon コマンドを使って有効にできます。

> **Note**
>
> /dev/ でリストされるデバイス名の先頭に lsblk を付加する必要があります。デバイスは、sda3、sde3、xvde3 など、異なる名前になる場合があります。システムのデバイス名は、次のコマンドで使用します。

```
[ec2-user ~]$ sudo swapon /dev/xvda3
```

これで、スワップ空間が lsblk および swapon -s 出力に表示されます。

```
[ec2-user ~]$ lsblk
NAME  MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda1 202:1    0    8G  0 disk /
xvda3 202:3    0  896M  0 disk [SWAP]
[ec2-user ~]$ swapon -s
Filename                                Type            Size    Used    Priority
/dev/xvda3                              partition       917500  0       -1
```

また、システムを起動する度にこのスワップ空間が自動的に有効になるように、/etc/fstab ファイルを編集する必要もあります。

```
[ec2-user ~]$ sudo vim /etc/fstab
```

(システムの swap デバイス名を使用して) 次の行を /etc/fstab ファイルに追加します。

```
/dev/xvda3       none    swap    sw  0       0
```

インスタンスストアボリュームをスワップ空間として使用するには

どのインスタンスストアボリュームもスワップ空間として使用できます。例えば、m3.medium インスタンスタイプは、スワップ空間に適した 4 GB の SSD インスタンスストアボリュームを含み

ます。インスタンスストアボリュームがはるかに大きい場合 (例えば、350 GB)、ボリュームに 4~8 GB の小さいスワップパーティションを作成し、残りをデータボリュームにすることもできます。

> **Note**
>
> この手順は、インスタンスストレージをサポートするインスタンスタイプのみに適用されます。サポートされているインスタンスタイプについては、インスタンスストアボリュームを参照してください。

1. インスタンスにアタッチされたブロックデバイスの一覧を表示して、インスタンスストアボリュームのデバイス名を取得します。

```
[ec2-user ~]$ lsblk -p
NAME       MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
/dev/xvdb  202:16   0   4G  0 disk /media/ephemeral0
/dev/xvda1 202:1    0   8G  0 disk /
```

   この例では、インスタンスストアボリュームは /dev/xvdb です。これは Amazon Linux インスタンスであるため、インスタンスストアボリュームはフォーマットされ、/media/ephemeral0 にマウントされます。すべての Linux オペレーティングシステムでこれが自動的に実行されるわけではありません。

2. (省略可能) インスタンスストアボリュームがマウントされている場合 (lsblk コマンドの出力に MOUNTPOINT が表示されます)、次のコマンドを使ってアンマウントする必要があります。

```
[ec2-user ~]$ sudo umount /dev/xvdb
```

3. mkswap コマンドを使って、デバイスに Linux スワップ領域をセットアップします。

```
[ec2-user ~]$ sudo mkswap /dev/xvdb
mkswap: /dev/xvdb: warning: wiping old ext3 signature.
Setting up swapspace version 1, size = 4188668 KiB
no label, UUID=b4f63d28-67ed-46f0-b5e5-6928319e620b
```

4. 新しいスワップ空間を有効にします。

```
[ec2-user ~]$ sudo swapon /dev/xvdb
```

5. 新しいスワップ空間が使用されていることを確認します。

```
[ec2-user ~]$ swapon -s
Filename    Type  Size Used Priority
/dev/xvdb                               partition 4188668 0 -1
```

6. システムを起動する度にこのスワップ空間が自動的に有効になるように、/etc/fstab ファイルを編集します。

```
[ec2-user ~]$ sudo vim /etc/fstab
```

/etc/fstab ファイルに /dev/xvdb (または /dev/sdb) 用の項目がある場合は、それを以下の行に合わせて変更します。このデバイス用の項目がない場合は、/etc/fstab ファイルに以下の行を追加します (システムのスワップデバイス名を使用します)。

```
/dev/xvdb       none    swap    sw  0       0
```

> **Important**
>
> インスタンスが停止または休止すると、インスタンスストアボリュームデータが失われます。これには、Step 3で作成したインスタンスストアスワップ領域のフォーマットが含まれます。インスタンスストアのスワップ領域を使用するように設定されたインスタンスを停止および再起動した場合、新しいインスタンスストアボリュームでStep 1からStep 5を繰り返す必要があります。

インスタンスストアボリュームのディスクパフォーマンスの最適化

Amazon EC2 でのディスクの仮想化方法が原因となり、一部のインスタンスストアボリュームに対する最初の書き込みは、書き込みの場所にかかわらず、それ以降の書き込みより速度が遅くなります。ほとんどのアプリケーションでは、インスタンスの存続期間全体でこのコストを負担することは、許容範囲内です。ただし、高いディスクパフォーマンスを必要とする場合は、本稼働環境での使用の前に、ドライブのすべての場所に一度書き込みを行うことで初期化することをお勧めします。

> **Note**
>
> インスタンスタイプの中には、初期化を行わずに、起動時に最大限のパフォーマンスを発揮する直接アタッチされた Solid State Drive (SSD) および TRIM サポートを使用するものがあ

ります。各インスタンスタイプのインスタンスストアについては、インスタンスストアボリュームを参照してください。

レイテンシーやスループットに関してさらに柔軟性が必要な場合は、Amazon EBS を使用することをお勧めします。

インスタンスストアボリュームを初期化するには、初期化するストア (例: dd または /dev/sdb) に応じて、次の /dev/nvme1n1 コマンドを使用します。

> **Note**
>
> 必ずドライブをアンマウントしてから、このコマンドを実行してください。
> 初期化には長い時間がかかる場合があります (エクストララージのインスタンスで約 8 時間)。

インスタンスストアボリュームを初期化するには、m1.large、m1.xlarge、c1.xlarge、m2.xlarge、m2.2xlarge、m2.4xlarge インスタンスタイプで次のコマンドを使用します。

```
dd if=/dev/zero of=/dev/sdb bs=1M
dd if=/dev/zero of=/dev/sdc bs=1M
dd if=/dev/zero of=/dev/sdd bs=1M
dd if=/dev/zero of=/dev/sde bs=1M
```

すべてのインスタンスストアボリュームに対して同時に初期化を実行するには、次のコマンドを使用します。

```
dd if=/dev/zero bs=1M|tee /dev/sdb|tee /dev/sdc|tee /dev/sde > /dev/sdd
```

ドライブを RAID 用に構成すると、ドライブのすべての場所に書き込みを行うことで、ドライブが初期化されます。ソフトウェアベースの RAID を構成するときは、再構築の最低速度を必ず変更してください。

```
echo $((30*1024)) > /proc/sys/dev/raid/speed_limit_min
```

ファイルストレージ

クラウドファイルストレージは、クラウド上にデータを保存する方法で、サーバーとアプリケーションは共有ファイルシステムを通してデータにアクセスできます。クラウドファイルストレージが持つこの互換性は共有ファイルシステムに依存するワークロードに最適で、コードを変更することなく統合を容易に行えます。

スケーラビリティを持たない、またはデータを保護するための冗長性がほとんどないブロックストレージを基盤として使用する、コンピューティングインスタンス上の単一ノードのファイルサーバーから、独自のクラスター化されたソリューション、完全マネージド型のソリューションまで、さまざまなファイルストレージソリューションがあります。次のコンテンツでは、Linux で使用するために AWS から提供されているストレージサービスの一部を紹介しています。

コンテンツ

Amazon EC2 での Amazon S3 の使用

Amazon Simple Storage Service (Amazon S3) は、業界をリードするスケーラビリティ、データ可用性、セキュリティ、およびパフォーマンスを提供するオブジェクトストレージサービスです。Amazon S3 を使用して、データレイク、ウェブサイト、バックアップ、ビッグデータ分析など、さまざまなユースケースの任意の量のデータを Amazon EC2 インスタンスから、またはインターネット経由でどこからでも保存および取得できます。詳細については、「Amazon S3 とは」を参照してください。

オブジェクトとは、Amazon S3 に格納される基本エンティティです。Amazon S3 に格納されるすべてのオブジェクトは、バケットに保管されます。バケットは Amazon S3 名前空間の最上位レベルを構成し、個々のストレージを所有するアカウントを識別します。Amazon S3 のバケットはインターネットのドメイン名に似ています。バケットに格納されたオブジェクトは一意のキー値を持ち、URL を使用して取得されます。例えば、キー値 (`/photos/mygarden.jpg`) を持つオブジェクトが *`DOC-EXAMPLE-BUCKET1`* バケットに格納されている場合、このオブジェクトは URL (`https://`*`DOC-EXAMPLE-BUCKET1`*`.s3.amazonaws.com/photos/mygarden.jpg`) を使用してアドレス解決できます。詳細については、「Amazon S3 の仕組み」を参照してください。

使用例

Amazon S3 にはストレージとしての利点があるため、場合によっては、このサービスを使用して、EC2 インスタンス用にファイルとデータセットを保存してもかまいません。Amazon S3 とインスタンスとの間でデータを移動するには、いくつかの方法があります。以下に説明する例以外にも、コンピュータやインスタンスから Amazon S3 のデータにアクセスできるさまざまなツールが、他のユーザーによって作成されています。一般的な一部のツールについては、AWS フォーラムで取り上げられています。

アクセス許可がある場合は、以下の方法を使用して、Amazon S3 とインスタンスとの間でファイルをコピーできます。

GET または wget

> **Note**
>
> この手法は、パブリックなオブジェクトに対してのみ有効です。オブジェクトがパブリックでない場合は、`ERROR 403: Forbidden` メッセージが出力されます。このエラーを受け取った場合は、Amazon S3 コンソール、AWS CLI、AWS API、AWS SDK、または AWS Tools for Windows PowerShell を使用する必要があります。この際は、適切なアクセス許可が必要です。詳細については、Amazon S3 ユーザーガイドのAmazon S3 での Identity and Access Managementおよびオブジェクトのダウンロードを参照してください。

wget ユーティリティは、Amazon S3 からパブリックオブジェクトをダウンロードできる HTTP および FTP のクライアントです。これは、Amazon Linux やその他のほとんどのディストリビューションにデフォルトでインストールされ、Windows ではダウンロード可能です。Amazon S3 オブジェクトをダウンロードするには、次のコマンドを入力し、ダウンロードするオブジェクトの URL に置き換えます。

```
[ec2-user ~]$ wget https://my_bucket.s3.amazonaws.com/path-to-file
```

AWS Command Line Interface

AWS Command Line Interface (AWS CLI) は、AWS サービスを管理するための統合ツールです。AWS CLI を使用すると、ユーザーは自分自身を認証し、限定された項目を Simple Storage Service (Amazon S3) からダウンロードしたり、項目をアップロードしたりできます。ツールのインストールおよび設定方法などの詳細については、AWS Command Line Interface の詳細ページを参照してください。

aws s3 cp コマンドは、Unix cp コマンドと似ています。ファイルを Amazon S3 からインスタンスにコピーしたり、ファイルをインスタンスから Amazon S3 にコピーしたりできるほか、ファイルを Amazon S3 の 1 つの場所から別の場所にコピーすることもできます。

オブジェクトを Amazon S3 からインスタンスにコピーするには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 cp s3://my_bucket/my_folder/my_file.ext my_copied_file.ext
```

オブジェクトをインスタンスから Amazon S3 にコピーして戻すには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 cp my_copied_file.ext s3://my_bucket/my_folder/my_file.ext
```

aws s3 sync コマンドは、Amazon S3 バケット全体をローカルディレクトリの場所に同期できます。この機能は、データセットをダウンロードし、リモートセットでローカルコピーを最新の状態に保つ際に役立ちます。Amazon S3 バケットに対して適切なアクセス許可がある場合は、コマンドで送信元と送信先の場所を入れ替えることで、終了時にローカルディレクトリバックアップをクラウドにプッシュできます。

Amazon S3 バケット全体をインスタンスのローカルディレクトリにダウンロードするには、次のコマンドを使用します。

```
[ec2-user ~]$ aws s3 sync s3://remote_S3_bucket local_directory
```

Amazon S3 API

デベロッパーは API を使用して、Amazon S3 のデータにアクセスできます。詳細については、Amazon Simple Storage Serviceユーザーガイドを参照してください。この API およびその例を使用すると、アプリケーションを開発し、boto Python インターフェイスなどのその他の API および SDK と統合するのに役立ちます。

Amazon EC2 での Amazon EFS の使用

> **Note**
>
> Amazon EFS は Windows インスタンスではサポートされていません。

Amazon EFS は、Amazon EC2 と併用できるスケーラブルなファイルストレージを提供します。複数のインスタンスで実行している作業負荷やアプリケーションの一般的なデータソースとしてEFS

ファイルシステムを使用できます。詳細については、Amazon Elastic File System 製品ページを参照してください。

このチュートリアルでは、インスタンスの起動時に、Amazon EFS クイック作成ウィザードを使用して Amazon EFS ファイルシステムを作成し、アタッチする方法について解説します。Amazon EFS コンソールを使用してファイルシステムを作成する方法に関するチュートリアルについては、「Amazon Elastic File System User Guide」(Amazon Elastic File System ユーザーガイド) の「Amazon Elastic File System の使用開始」を参照してください。

> **Note**
>
> EFS クイック作成を使用して EFS ファイルシステムを作成すると、次のサービス推奨設定でファイルシステムが作成されます。
>
> - 自動バックアップを有効化しました。
> - 選択した VPC の各デフォルトサブネットにターゲットをマウントします。
> - 汎用パフォーマンスモード。
> - バーストスループットモード。
> - Amazon EFS (`aws/elasticfilesystem`) のデフォルトのキーを使用して、保管中のデータの暗号化を有効にしました。
> - 30日間のポリシーで Amazon EFS ライフサイクル管理を有効にしました。

タスク

- Amazon EFS クイック作成を使用した EFS ファイルシステムの作成
- EFS ファイルシステムをテストする
- EFS ファイルシステムを削除する

Amazon EFS クイック作成を使用した EFS ファイルシステムの作成

Amazon EC2 インスタンス起動ウィザードの Amazon EFS クイック作成機能を使用してインスタンスを起動するときに、EFS ファイルシステムを作成してインスタンスにマウントできます。

Amazon EFS クイック作成を使用して EFS ファイルシステムを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. [インスタンスを起動] を選択します。

3. (オプション) [Names and tags] (名前とタグ) における [Name] (名前) では、インスタンスを識別するための名前を入力します。
4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で Linux オペレーティングシステムを選択し、[Amazon Machine Image (AMI)] (Amazon マシンイメージ (AMI)) で Linux AMI を選択します。
5. [Instance type] (インスタンスタイプ) の [Instance type](インスタンスタイプ) で、インスタンスタイプを選択するか、デフォルトのままにします。
6. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。
7. [Network settings] (ネットワーク設定) で [Edit] (編集) (右側) を選択し、[Subnet] (サブネット) でサブネットを選択します。

   > **Note**
   >
   > EFS ファイルシステムを追加する前に、サブネットを選択する必要があります。

8. [Configure storage] (ストレージを設定) で、[Edit](編集) (右下) を選択し、次の操作を実行します。

   a. [ファイルシステム] で EFS が選択されていることを確認し、[新しい共有ファイルシステムを作成] を選択します。

   b. [ファイルシステム名] に Amazon EFS ファイルシステムの名前を入力し、[ファイルシステムの作成] を選択します。

   c. [マウントポイント] で、カスタムのマウントポイントを指定するか、デフォルトのままにします。

   d. ファイルシステムへのアクセスを有効にするには、[Automatically create and attach security groups] (セキュリティグループを自動的に作成してアタッチ) を選択します。このチェックボックスをオンにすると、次のセキュリティグループが自動的に作成され、ファイルシステムのインスタンスとマウントターゲットにアタッチされます。

      - インスタンスセキュリティグループ – NFS 2049 ポート経由のトラフィックを許可するアウトバウンドルールは含まれますが、インバウンドルールは含まれません。
      - ファイルシステムマウントターゲットセキュリティグループ — インスタンスセキュリティグループ (上述) からの NFS 2049 ポート経由のトラフィックを許可するインバウンドルールと、NFS 2049 ポート経由のトラフィックを許可するアウトバウンドルールが含まれます。

> **Note**
>
> 代わりに、セキュリティグループを手動で作成してアタッチすることができます。セキュリティグループを手動で作成してアタッチする場合は、[Automatically create and attach the required security groups] (必要なセキュリティグループを自動的に作成してアタッチ) をオフにします。

e. インスタンスの起動時に共有ファイルシステムを自動的にマウントするには、[Automatically mount shared file system by attaching required user data script] (必要なユーザーデータスクリプトをアタッチして共有ファイルシステムを自動的にマウント) を選択します。自動的に生成されたユーザーデータを表示するには、[Advanced details] (高度な詳細) を展開し、[User data] (ユーザー データ) まで下方向にスクロールします。

> **Note**
>
> このチェックボックスをオンにする前にユーザーデータを追加すると、元のユーザーデータは、自動的に生成されたユーザーデータによって上書きされます。

9. 必要に応じて、その他のインスタンスの設定を行います。
10. [Summary] (概要) パネルでインスタンスの設定を確認し、[Launch instance] (インスタンスを起動) を選択します。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。

EFS ファイルシステムをテストする

インスタンスに接続し、ファイルシステムが指定したディレクトリにマウントされていることを確認します (例えば、/mnt/efs)。

ファイルシステムがマウントされていることを確認するには

1. インスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。
2. インスタンスのターミナルウィンドウから、df -T コマンドを実行して、EFS ファイルシステムがマウントされていることを確認します。

```
$ df -T
```

```
Filesystem      Type            1K-blocks    Used          Available Use% Mounted
 on
/dev/xvda1      ext4              8123812 1949800            6073764  25% /
devtmpfs        devtmpfs          4078468      56            4078412   1% /dev
tmpfs           tmpfs             4089312       0            4089312   0% /dev/shm
efs-dns         nfs4       9007199254740992       0   9007199254740992   0% /mnt/efs
```

なお、ファイルシステムの名前 (サンプル出力では *efs-dns* として表示) は次の形式になります。

```
file-system-id.efs.aws-region.amazonaws.com:/
```

3. (オプション) インスタンスからファイルシステムでファイルを作成し、別のインスタンスからファイルを表示できることを確認します。

   a. インスタンスから、次のコマンドを実行してファイルを作成します。

   ```
   $ sudo touch /mnt/efs/test-file.txt
   ```

   b. 他のインスタンスから、次のコマンドを実行してファイルを表示します。

   ```
   $ ls /mnt/efs
   test-file.txt
   ```

EFS ファイルシステムを削除する

ファイルシステムが不要になった場合には、それを削除することができます。

ファイルシステムを削除するには

1. Amazon Elastic File System コンソール (https://console.aws.amazon.com/efs/) を開きます。
2. 削除するファイル システムを選択します。
3. [Actions]、[Delete file system] の順に選択します。
4. 確認を求められたら、ファイルシステム ID を入力し、[Delete file system (ファイルシステムの削除)] を選択します。

Amazon EC2 での Amazon FSx の使用

Amazon FSx ファミリーのサービスにより、人気のある市販のファイルシステムやオープンソースのファイルシステムで動作する共有ストレージを簡単に起動、実行、スケーリングできます。新しいインスタンス起動ウィザードを使用すると、起動時に Amazon EC2 インスタンスに次のタイプの Amazon FSx ファイルシステムを自動的にアタッチできます。

- Amazon FSx for NetApp ONTAP により、AWS クラウドのフルマネージド共有ストレージで NetApp ONTAP の人気のあるデータアクセス機能と管理機能を利用できます。
- Amazon FSx for OpenZFS により、人気のある OpenZFS ファイルシステムでフルマネージドの費用対効果の高い共有ストレージを利用できます。

> **Note**
>
> - この機能は、新しいインスタンス起動ウィザードでのみ使用できます。詳細については、「新しいインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください
> - Amazon FSx for Windows File Server と Amazon FSx for Lustre ファイルシステムを起動時にマウントすることはできません。これらのファイルシステムは、起動後に手動でマウントする必要があります。

以前に作成した既存のファイルシステムをマウントすることも、起動時にインスタンスにマウントする新しいファイルシステムを作成することもできます。

トピック

セキュリティグループとユーザーデータスクリプト

インスタンス起動ウィザードを使用して Amazon FSx ファイルシステムをインスタンスにマウントする場合、ファイルシステムへのアクセスを有効にするために必要なセキュリティグループを自動的に作成してアタッチするかどうかを選択できます。また、ファイルシステムをマウントして使用可能にするために必要なユーザーデータスクリプトを自動的に含めるかどうかを選択できます。

トピック

セキュリティグループ

ファイルシステムへのアクセスを有効にするために必要なセキュリティグループを自動的に作成する場合は、インスタンス起動ウィザードが 2 つのセキュリティグループを作成してアタッチします。1 つはインスタンスにアタッチされ、もう 1 つはファイルシステムにアタッチされます。セキュリティグループの要件の詳細については、「FSx for ONTAP file system access control with Amazon VPC」(Amazon VPC での FSx for ONTAP ファイルシステムアクセスコントロール) と「FSx for OpenZFS file system access control with Amazon VPC」(Amazon VPC での FSx for OpenZFS ファイルシステムアクセスコントロール) を参照してください。

作成されてインスタンスにアタッチされたセキュリティグループにタグ `Name=instance-sg-1` を追加します。インスタンス起動ウィザードが Amazon FSx ファイルシステム用に新しいセキュリティグループを作成するたびに、 タグの値が自動的に増分されます。

セキュリティグループには次の出力ルールが含まれていますが、インバウンドルールは含まれていません。

アウトバウンドルール

プロトコルのタイプ	ポート番号	デスティネーション
UDP	111	##################
UDP	20001 ~ 20003	##################
UDP	4049	##################
UDP	2049	##################
UDP	635	##################
UDP	4045 ~ 4046	##################
TCP	4049	##################
TCP	635	##################

プロトコルのタイプ	ポート番号	デスティネーション
TCP	2049	##################
TCP	111	##################
TCP	4045 ~ 4046	##################
TCP	20001 ~ 20003	##################
すべて	すべて	##################

作成されてファイルシステムにアタッチされたセキュリティグループには、Name=fsx-sg-*1* というタグが付けられます。インスタンス起動ウィザードが Amazon FSx ファイルシステム用に新しいセキュリティグループを作成するたびに、 タグの値が自動的に増分されます。

セキュリティグループには次のルールが含まれます。

インバウンドルール

プロトコルのタイプ	ポート番号	ソース
UDP	2049	################
UDP	20001 ~ 20003	################
UDP	4049	################
UDP	111	################
UDP	635	################
UDP	4045 ~ 4046	################
TCP	4045 ~ 4046	################
TCP	635	################
TCP	2049	################
TCP	4049	################

プロトコルのタイプ	ポート番号	ソース
TCP	20001 ~ 20003	################
TCP	111	################

アウトバウンドルール

プロトコルのタイプ	ポート番号	デスティネーション
すべて	すべて	0.0.0.0/0

ユーザーデータスクリプト

ユーザーデータスクリプトを自動的にアタッチする場合は、インスタンス起動ウィザードが次のユーザーデータをインスタンスに追加します。このスクリプトは、必要なパッケージをインストールし、ファイルシステムをマウントします。また、インスタンスが再起動されるたびにファイルシステムが自動的に再マウントされるようにインスタンスの設定を更新します。

```
#cloud-config
package_update: true
package_upgrade: true
runcmd:
- yum install -y nfs-utils
- apt-get -y install nfs-common
- svm_id_1=svm_id
- file_system_id_1=file_system_id
- vol_path_1=/vol1
- fsx_mount_point_1=/mnt/fsx/fs1
- mkdir -p "${fsx_mount_point_1}"
- if [ -z "$svm_id_1" ]; then printf "\n${file_system_id_1}.fsx.eu-
north-1.amazonaws.com:/${vol_path_1} ${fsx_mount_point_1} nfs4
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev
 0 0\n" >> /etc/fstab; else printf "\n${svm_id_1}.${file_system_id_1}.fsx.eu-
north-1.amazonaws.com:/${vol_path_1} ${fsx_mount_point_1} nfs4
 nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2,noresvport,_netdev 0
 0\n" >> /etc/fstab; fi
- retryCnt=15; waitTime=30; while true; do mount -a -t nfs4 defaults; if [ $? = 0 ] ||
 [ $retryCnt -lt 1 ]; then echo File system mounted successfully; break; fi; echo File
 system not available, retrying to mount.; ((retryCnt--)); sleep $waitTime; done;
```

起動時に Amazon FSx ファイルシステムをマウントする

起動時に新規または既存の Amazon FSx ファイルシステムをマウントするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch instance] (インスタンスの起動) を選択して、インスタンス起動ウィザードを開きます。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、使用する AMI を選択します。
4. [Instance type] (インスタンスタイプ) セクションで、インスタンスタイプを選択します。
5. [Key pair] (キーペア) セクションで、既存のキーペアを選択するか、新規にキーペアを作成します。
6. [Network settings] (ネットワーク設定) セクションで、以下の操作を行います。

   a. [編集] を選択します。

   b. 既存のファイルシステムをマウントする場合は、[Subnet] (サブネット) でファイルシステムの優先サブネットを選択します。パフォーマンスを最適化するために、ファイルシステムの優先サブネットと同じアベイラビリティーゾーンでインスタンスを起動することをお勧めします。

      インスタンスにマウントするファイルシステムを新規に作成する場合は、[Subnet] (サブネット) で、インスタンスを起動するサブネットを選択します。

      > **Important**
      >
      > 新しいインスタンス起動ウィザードで Amazon FSx の機能を有効にするサブネットを選択する必要があります。サブネットを選択しないと、既存のファイルシステムをマウントすることも、新規にファイルシステムを作成することもできなくなります。

7. [Storage] (ストレージ) セクションで、以下の操作を行います。

   a. 必要に応じてボリュームを設定します。

   b. [File systems] (ファイルシステム) セクションを展開し、[FSx] を選択します。

   c. [Add shared file system] (共有ファイルシステムの追加) を選択します。

   d. [File system] (ファイルシステム) で、マウントするファイルシステムを選択します。

> **Note**
>
> 選択したリージョンのアカウントにあるすべての Amazon FSx for NetApp ONTAP ファイルシステムと Amazon FSx for OpenZFS ファイルシステムがリストに表示されます。

e. ファイルシステムへのアクセスを有効にするために必要なセキュリティグループを自動的に作成してアタッチするには、[Automatically create and attach security groups] (セキュリティグループを自動的に作成してアタッチする) を選択します。セキュリティグループを手動で作成する場合は、このチェックボックスをオフにします。詳細については、「セキュリティグループ」を参照してください。

f. ファイルシステムをマウントするために必要なユーザーデータスクリプトを自動的にアタッチするには、[Automatically mount shared file system by attaching required user data script] (必要なユーザーデータスクリプトをアタッチして、共有ファイルシステムを自動的にマウントする) を選択します。ユーザーデータスクリプトを手動で指定する場合は、このチェックボックスをオフにします。詳細については、「ユーザーデータスクリプト」を参照してください。

8. [Advanced] (詳細) セクションで、必要に応じて追加でインスタンスの設定を行います。
9. [Launch] (起動する) を選択します。

Amazon EC2 での Amazon File Cache の使用

Amazon File Cache は、データの保存場所にかかわらず、ファイルデータの処理に使用される AWS のフルマネージド型の高速キャッシュです。Amazon File Cache は、オンプレミスのファイルシステム、AWS ファイルシステム、Amazon Simple Storage Service (Amazon S3) バケットに保存されているデータの一時的な高性能ストレージの場所として機能します。この機能を使用すると、分散したデータセットを AWS 上のファイルベースのアプリケーションで統合ビューを使用して高速 (ミリ秒未満のレイテンシーと高いスループット) で利用できるようになります。詳細については、「What is Amazon File Cache?」を参照してください。

オープンソースの Lustre クライアントを使用して、Amazon EC2 インスタンスからキャッシュにアクセスできます。Amazon EC2 インスタンスは、同じ Amazon Virtual Private Cloud (Amazon VPC) 内の他のアベイラビリティーゾーンからキャッシュにアクセスできます。ただし、ネットワークが VPC 内のサブネットを越えてアクセスできる場合に限ります。キャッシュがマウントされたら、ローカルファイルシステムを使用する場合と同じように、ファイルやディレクトリを操作できるようになります。

開始するには、「Getting started with Amazon File Cache」を参照してください。

インスタンスボリューム数の制限

インスタンスにアタッチできる Amazon EBS ボリュームの最大数は、インスタンスのタイプとサイズによって異なります。インスタンスに追加するボリューム数を検討する際は、I/O 帯域幅とストレージ容量のどちらを増加するのかを、考慮する必要があります。

帯域幅と容量

整合性がとれており予測可能な帯域幅のユースケースでは、Amazon EBS 最適化インスタンスと、汎用 SSD ボリュームまたは Provisioned IOPS SSD ボリュームを使用します。パフォーマンスを最大化するためには、ボリュームに対してプロビジョニングした IOPS を、使用しているインスタンスで利用可能な帯域幅と一致させます。

RAID 構成では、I/O オーバーヘッドの増加が原因となり、8 つのボリュームよりも大きなアレイがパフォーマンスを低下させることがあります。個々のアプリケーションのパフォーマンスをテストし、必要に応じて調整してください。

トピック

- Nitro システム上に構築されたインスタンスにおけるボリューム制限
- Xen ベースのインスタンスのボリューム制限

Nitro システム上に構築されたインスタンスにおけるボリューム制限

専用 Amazon EBS ボリュームの制限

以下の Nitro インスタンスタイプには、インスタンスサイズに応じて異なる専用の Amazon EBS ボリューム制限があります。この制限は他のデバイスアタッチメントとは共有されません。つまり、NVMe インスタンスストアボリュームやネットワークインターフェイスなど接続されているデバイスの数に関係なく、ボリュームのアタッチ上限まで任意の数の Amazon EBS ボリュームをアタッチできます。

- 汎用: M7a, M7i, M7i-flex
- コンピューティングの最適化: C7a、C7i
- メモリの最適化: R7a、R7i、R7iz

ボリュームの制限は、インスタンスのサイズによって異なります。次の表に、インスタンスの各サイズ別の上限値を示します。

インスタンスサイズ	ボリュームの制限
medium | large | xlarge | 2xlarge | 4xlarge | 8xlarge | 12xlarge	32
16xlarge	48
24xlarge	64
32xlarge	88
48xlarge	128
metal-16xl | metal-24xl	39
metal-32xl | metal-48xl	79

Amazon EBS 共有ボリュームでの制限

その他のすべての Nitro インスタンスタイプには、Amazon EBS ボリューム、ネットワークインターフェイス、および NVMe インスタンスストアボリューム間で共有されるボリュームのアタッチ数に制限があります。その制限数から、アタッチ済みのネットワークインターフェイスと NVMe インスタンスストアボリュームの数を差し引いた数を上限として、Amazon EBS ボリュームをいくつでもアタッチできます。すべてのインスタンスには少なくとも 1 つのネットワークインターフェースが必要であり、NVMe インスタンスストアボリュームは起動時に自動的にアタッチされることに注意してください。

これらのインスタンスのほとんどは、最大 28 のアタッチメントをサポートします。例えば、m5.xlarge インスタンスに追加のネットワークインターフェイスのアタッチがない場合は、最大 27 個 (28 のボリューム上限 – 1 つのネットワークインターフェイス) の EBS ボリュームをアタッチできます。m5.xlargeインスタンスに 2 つのネットワークインターフェースが追加されている場合は、最大 25 個 (28 のボリューム上限 – 3つのネットワークインターフェース) の EBS ボリュームをアタッチできます。同様に、1 つの NVMe インスタンスストアボリュームを持つ m5d.xlarge イ

ンスタンスに、ネットワークインターフェースが 2 つ追加されている場合は、最大 24 個 (28 のボリューム上限 – 3 つのネットワークインターフェース – 1 つのNVMe インスタンスストアボリューム) の EBS ボリュームをアタッチできます。

以下の例外が適用されます。

- DL2q インスタンスは、最大 19 個の EBS ボリュームをサポートします。
- ほとんどのベアメタルインスタンスは、最大 31 の EBS ボリュームをサポートします。
- ハイメモリ仮想化インスタンスは、最大 27 の EBS ボリュームをサポートします。
- ハイメモリベアメタルインスタンスは、最大 19 の EBS ボリュームをサポートします。
- `inf1.xlarge` および `inf1.2xlarge` インスタンスは、最大 26 の EBS ボリュームをサポートします。
- `inf1.6xlarge` インスタンスは、最大 23 の EBS ボリュームをサポートします。
- `mac1.metal` インスタンスは、最大 16 の EBS ボリュームをサポートします。
- `mac2.metal`、`mac2-m2.metal`、および `mac2-m2pro.metal` インスタンスは、最大 10 の EBS ボリュームをサポートしています。
- `inf1.24xlarge` インスタンスは、最大 11 の EBS ボリュームをサポートします。
- `g5.48xlarge` インスタンスは、最大 9 つの EBS ボリュームをサポートします。
- `d3.8xlarge` および `d3en.12xlarge` インスタンスは、最大 3 の EBS ボリュームをサポートしています。
- 高速コンピューティングインスタンスの場合、アタッチされたアクセラレータ数が、共有ボリューム制限にカウントされます。例えば、共有ボリュームを上限の 28 個、GPU を 8 個、NVMe インスタンスストアボリュームを 8 個持つ `p4d.24xlarge` インスタンスでは、最大 11 個 (28 個のボリューム上限 – 1 個のネットワークインターフェース – 8 個の GPU – 8 個の NVMe インスタンスストアボリューム) の Amazon EBS ボリュームをアタッチできます。

Xen ベースのインスタンスのボリューム制限

Xen ベースの Linux インスタンスに 40 個を超えるボリュームをアタッチすると、ブートに失敗する可能性があります。この数には、ルートボリュームに加え、アタッチされたインスタンスストアボリュームと Amazon EBS ボリュームも含まれます。

多数のボリュームがアタッチされているインスタンスで起動の問題が発生した場合は、インスタンスを停止し、起動プロセスに不可欠ではないボリュームをデタッチした上で再起動し、インスタンスの実行後にそれらのボリュームを再度アタッチします。

> **Important**
>
> Xen ベースの Linux インスタンスに 40 より多くのボリュームをアタッチすることは、ベストエフォートベースでのみサポートされており、動作は保証されていません。

Amazon EC2 インスタンスのルートボリューム

インスタンスを起動すると、インスタンスのルートボリュームが作成されます。ルートボリュームには、インスタンスの起動に使用されるイメージが含まれています。各インスタンスには 1 つのルートボリュームがあります。ストレージボリュームは、インスタンスの起動時または実行後に追加できます。

特定のデバイス名がルートボリューム用に予約されています。詳細については、「Linux インスタンスでのデバイス名」を参照してください。

コンテンツ

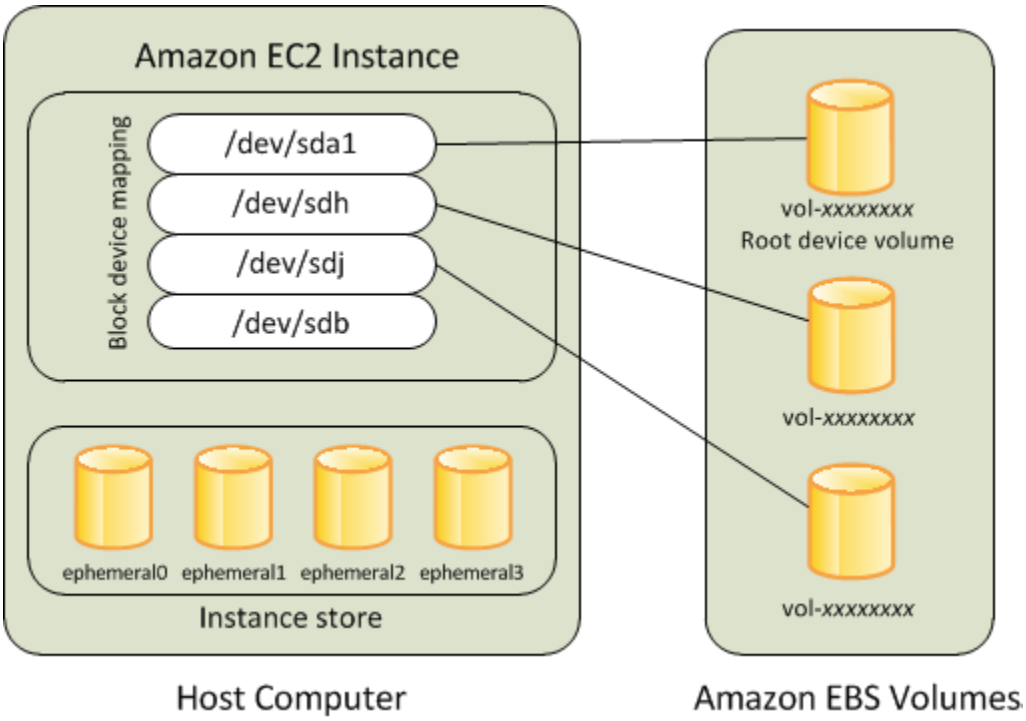
ルートボリュームタイプ

instance store-backed AMI または Amazon EBS-backed AMI のどちらからでもインスタンスを起動できます。AMI の説明にはそのタイプが含まれており、場所によってルートデバイスが `ebs` (Amazon EBS-Backed の場合) または `instance store` (Instance store-Backed の場合) と表示されます。各タイプの AMI を使用して実行できることには大きな違いがあるため、タイプを区別できることは重要です。違いについての詳細は ルートデバイスのストレージ を参照してください。Amazon EBS-backed AMI を使用することをお勧めします。これらのインスタンスは起動速度が速く、永続的ストレージを使用しているからです。

Amazon EBS-backed インスタンス

Amazon EBS をルートボリュームに使用するインスタンスには、Amazon EBS ボリュームが自動的にアタッチされます。Amazon EBS Backed インスタンスを起動するときに、AMI で参照されている Amazon EBS スナップショットごとに 1 つの Amazon EBS ボリュームが作成されます。インスタンスタイプによっては、Amazon EBS ボリュームまたはインスタンスストアボリュームをオプションで使用できます。

Amazon EBS-backed インスタンスは、停止後に再起動できます。アタッチされているボリュームに格納されているデータに影響を及ぼすこともありません。Amazon EBS-backed インスタンスが停止状態にあるときは、インスタンスおよびボリューム関連の様々なタスクを実行できます。例えば、インスタンスのプロパティの変更、そのサイズの変更、あるいは使用しているカーネルを更新できます。また、デバッグなどの目的で別の実行中インスタンスにルートボリュームをアタッチすることもできます。詳細については、「Amazon EBS ボリューム」を参照してください。



制限

st1 または sc1 EBS ボリュームをルートボリュームとして使用することはできません。

インスタンスの障害

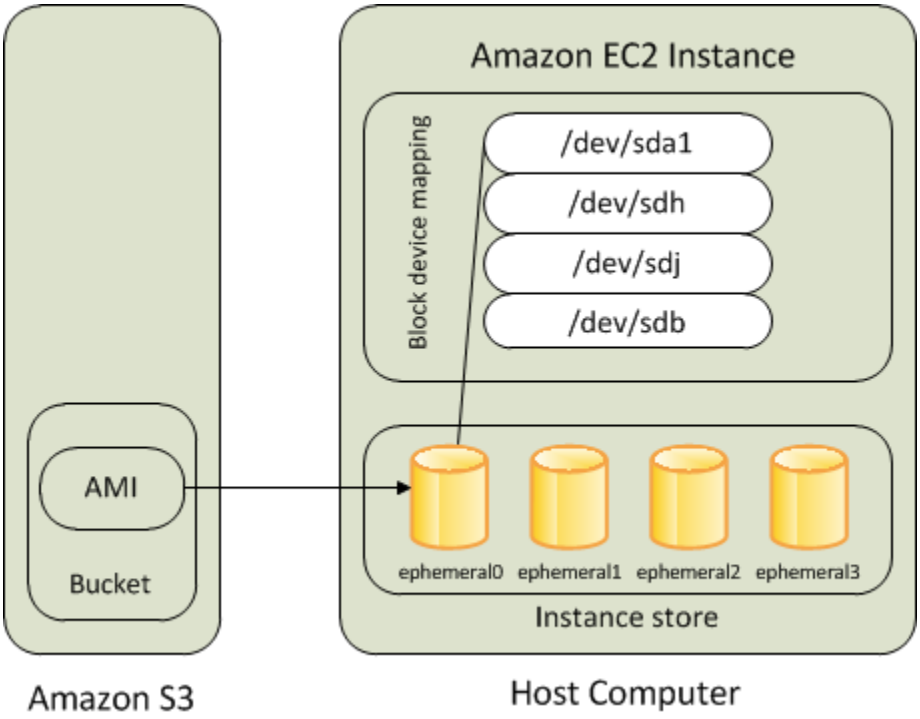
Amazon EBS-backed インスタンスに障害が発生した場合は、以下のいずれかの方法によってセッションを復元できます。

- 停止して再起動します (最初にこの方法を試してください)。
- 関連するすべてのボリュームのスナップショットを自動的に作成し、新しい AMI を作成します。詳細については、Amazon EBS-backed Linux AMI を作成するを参照してください。
- 以下の手順に従って、ボリュームを新しいインスタンスにアタッチします。
  1. ルートボリュームのスナップショットを作成します。
  2. 作成したスナップショットを使用して新しい AMI を登録します。
  3. 新しい AMI から新しいインスタンスを起動します。
  4. 残りの Amazon EBS ボリュームを古いインスタンスからデタッチします。
  5. Amazon EBS ボリュームを新しいインスタンスに再アタッチします。

Instance store-Backed インスタンス

インスタンスストアをルートデバイスに使用するインスタンスでは、自動的に 1 つまたは複数のインスタンスストアボリュームを利用できるようになり、そのボリュームの 1 つがルートボリュームとして機能します。インスタンスを起動すると、インスタンスのブートに使用されるイメージがルートボリュームにコピーされます。インスタンスタイプによっては、オプションで追加のインスタンスストアボリュームを使用できることに注意してください。

インスタンスストアボリュームのデータはインスタンスが実行している間は維持されますが、インスタンスが終了すると (Instance store-Backed インスタンスは [Stop] アクションをサポートしていません)、またはインスタンスが失敗すると (基盤となるドライブに問題がある場合など)、削除されます。詳細については、「Amazon EC2 インスタンスストア」を参照してください。



要件

インスタンスストアボリュームをルートボリュームとしてサポートするインスタンスタイプは C3、D2、G2、I2、M3、および R3 のみです。

インスタンスの障害

障害が発生したり終了されたりした instance store-backed インスタンスは復元できません。Amazon EC2 instance store-backed インスタンスの使用を予定している場合は、インスタンスストアのデータを複数のアベイラビリティーゾーンにまたがって分散させることを強くお勧めします。また、インスタンスストアボリュームからの重要データは永続的ストレージに定期的にバックアップする必要があります。

ルートボリュームタイプによる AMI の選択

インスタンスの起動時に指定する AMI によって、インスタンスのルートデバイスボリュームのタイプが決まります。次のいずれかの方法で、AMI をルートデバイスタイプ別に表示できます。

Console

コンソールを使用して Amazon EBS-backed AMI を選択するには

1. Amazon EC2 コンソールを開きます。

2. ナビゲーションペインで [AMIs] を選択します。
3. フィルタの一覧から、イメージタイプ ([Public images] など) を選択します。検索バーで、[プラットフォーム] を選択してオペレーティングシステム ([Amazon Linux] など) を選択し、[ルートデバイスタイプ] を選択してルートボリュームタイプ ([ebs]) を選択します。
4. (オプション) 選択の参考になる追加情報を表示するには、[設定] アイコンを選択し、表示する列をトグルして、[確認] を選択します。
5. AMI を選択し、その AMI ID を記録します。

コンソールを使用して instance store-backed AMI を選択するには

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインで [AMIs] を選択します。
3. フィルタの一覧から、イメージタイプ ([Public images] など) を選択します。検索バーで、[プラットフォーム] を選択してオペレーティングシステム ([Amazon Linux] など) を選択し、[ルートデバイスタイプ] を選択してルートボリュームタイプ ([instance-store]) を選択します。
4. (オプション) 選択の参考になる追加情報を表示するには、[設定] アイコンを選択し、表示する列をトグルして、[確認] を選択します。
5. AMI を選択し、その AMI ID を記録します。

AWS CLI

コマンドラインを使用して AMI のルートデバイスボリュームの種類を確認するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- describe-images (AWS CLI)
- Get-EC2Image (AWS Tools for Windows PowerShell)

インスタンスのルートデバイスタイプの判別

Console

コンソールを使用してインスタンスのルートデバイスタイプを判別するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. [Storage (ストレージ)] タブの [Root device details (ルートデバイスの詳細)] で、[Root device type (ルートデバイスタイプ)] の値を次のように確認します。
   - 値が `EBS` の場合は Amazon EBS-Backed インスタンスです。
   - 値が `INSTANCE-STORE` の場合、これは Instance store-Backed インスタンスです。

AWS CLI

コマンドラインを使用してインスタンスのルートデバイスタイプを判別するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、「Amazon EC2 へのアクセス」を参照してください。

- describe-instances (AWS CLI)
- Get-EC2Instance (AWS Tools for Windows PowerShell)

永続的ルートボリュームへの変更

デフォルトでは、Amazon EBS-backed AMI のルートボリュームは、インスタンスを終了すると削除されます。インスタンスの終了後もボリュームが永続化するように、デフォルトの動作を変更できます。デフォルトの動作を変更するには、ブロックデバイスマッピングを使用して、`DeleteOnTermination` 属性を `false` に設定します。

タスク

- インスタンスの起動時に永続化するためのルートボリュームの設定
- 既存のインスタンスで永続化するためのルートボリュームの設定
- ルートボリュームが永続化するように設定されていることの確認

インスタンスの起動時に永続化するためのルートボリュームの設定

Amazon EC2 コンソールまたはコマンドラインツールを使用して、インスタンスの起動時に永続化するようにルートボリュームを設定できます。

Console

コンソールを使用してインスタンスの起動時に永続化するようにルートボリュームを設定するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス]、[インスタンスの作成] の順に選択します。
3. Amazon マシンイメージ (AMI) を選択して、インスタンスタイプ、キーペアの順に選択し、ネットワークを設定します。
4. [ストレージを設定] には [アドバンスト] を選択します。
5. ルートボリュームを拡張します。
6. [終了時に削除] には、[いいえ] を選択します。
7. インスタンスの設定が終了したら、[インスタンスを起動] をクリックします。

AWS CLI

AWS CLI を使用してインスタンスの起動時に永続化するようにルートボリュームを設定するには

run-instances コマンドを使用して、DeleteOnTermination 属性を false に設定するブロックデバイスマッピングを含めます。

```
$ aws ec2 run-instances --block-device-mappings file://mapping.json ...other
 parameters...
```

mapping.json で、以下を指定します。

```
[
    {
        "DeviceName": "/dev/sda1",
        "Ebs": {
            "DeleteOnTermination": false
        }
    }
```

```
]
```

Tools for Windows PowerShell

Tools for Windows PowerShell を使用してインスタンスの起動時に永続化するようにルートボリュームを設定するには

New-EC2Instance コマンドを使用して、DeleteOnTermination 属性を false に設定するブロックデバイスマッピングを含めます。

```
C:\> $ebs = New-Object Amazon.EC2.Model.EbsBlockDevice
C:\> $ebs.DeleteOnTermination = $false
C:\> $bdm = New-Object Amazon.EC2.Model.BlockDeviceMapping
C:\> $bdm.DeviceName = "dev/xvda"
C:\> $bdm.Ebs = $ebs
C:\> New-EC2Instance -ImageId ami-0abcdef1234567890 -BlockDeviceMapping
 $bdm ...other parameters...
```

既存のインスタンスで永続化するためのルートボリュームの設定

コマンドラインツールのみを使用して、実行中のインスタンスで永続化するようにルートボリュームを設定できます。

AWS CLI

AWS CLI を使用して、既存のインスタンスで永続化するようにルートボリュームを設定するには

DeleteOnTermination 属性を false に設定するブロックデバイスマッピングを指定して modify-instance-attribute コマンドを使用します。

```
aws ec2 modify-instance-attribute --instance-id i-1234567890abcdef0 --block-device-
mappings file://mapping.json
```

mapping.json で、以下を指定します。

```
[
    {
        "DeviceName": "/dev/xvda",
        "Ebs": {
```

```
            "DeleteOnTermination": false
        }
    }
]
```

Tools for Windows PowerShell

AWS Tools for Windows PowerShell を使用して、既存のインスタンスで永続化するようにルートボリュームを設定するには

`DeleteOnTermination` 属性を `false` に設定するブロックデバイスマッピングを指定して Edit-EC2InstanceAttribute コマンドを使用します。

```
C:\> $ebs = New-Object Amazon.EC2.Model.EbsInstanceBlockDeviceSpecification
C:\> $ebs.DeleteOnTermination = $false
C:\> $bdm = New-Object Amazon.EC2.Model.InstanceBlockDeviceMappingSpecification
C:\> $bdm.DeviceName = "/dev/xvda"
C:\> $bdm.Ebs = $ebs
C:\> Edit-EC2InstanceAttribute -InstanceId i-1234567890abcdef0 -BlockDeviceMapping
 $bdm
```

ルートボリュームが永続化するように設定されていることの確認

Amazon EC2 コンソールまたはコマンドラインツールを使用して、ルートボリュームが永続化するように設定されていることを確認できます。

Console

Amazon EC2 コンソールを使用してルートボリュームが永続化するように設定されていることを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択してから、インスタンスを選択します。
3. [ストレージ] タブの [ブロックデバイス] で、ルートボリュームのエントリを見つけます。[合わせて削除] が No の場合、ボリュームは永続化するように設定されます。

AWS CLI

AWS CLI を使用してルートボリュームが永続化するように設定されていることを確認するには

describe-instances コマンドを使用して、DeleteOnTermination レスポンス要素の BlockDeviceMappings 属性が false に設定されていることを確認します。

```
$ aws ec2 describe-instances --instance-id i-1234567890abcdef0
```

```
...
    "BlockDeviceMappings": [
    {
        "DeviceName": "/dev/sda1",
        "Ebs": {
            "Status": "attached",
            "DeleteOnTermination": false,
            "VolumeId": "vol-1234567890abcdef0",
            "AttachTime": "2013-07-19T02:42:39.000Z"
        }
    }
...
```

Tools for Windows PowerShell

AWS Tools for Windows PowerShell を使用してルートボリュームが永続化するように設定されていることを確認するには

Get-EC2Instance コマンドを使用して、DeleteOnTermination レスポンス要素の BlockDeviceMappings 属性が false に設定されていることを確認します。

```
C:\> (Get-EC2Instance -InstanceId i-
i-1234567890abcdef0).Instances.BlockDeviceMappings.Ebs
```

ルートボリュームの初期サイズの変更

デフォルトでは、ルートボリュームのサイズはスナップショットのサイズによって決まります。次のようにインスタンスのブロックデバイスマッピングを使用して、ルートボリュームの初期サイズを増やすことができます。

1. AMI ブロックデバイスマッピングの EBS ボリュームの表示 の説明に従って、AMI で指定されているルートボリュームのデバイス名を決定します。
2. AMI ブロックデバイスマッピングで指定されたスナップショットのサイズを確認します。

3. インスタンス起動時のブロックデバイスマッピングの更新 の説明に従って、インスタンスブロックデバイスマッピングを使用してルートボリュームのサイズを上書きし、スナップショットサイズよりも大きいボリュームサイズを指定します。

例えば、インスタンスブロックデバイスマッピングの次のエントリは、ルートボリュームのサイズ /dev/xvda を 100 GiB に増やします。スナップショット ID は AMI ブロックデバイスマッピングで既に指定されているため、インスタンスブロックデバイスマッピングでスナップショット ID を省略できます。

```
{
    "DeviceName": "/dev/xvda",
    "Ebs": {
      "VolumeSize": 100
    }
}
```

詳細については、「ブロックデバイスマッピング」を参照してください。

EC2 インスタンスのルートボリュームを置き換える

Amazon EC2 では、実行中のインスタンスのルート EBS ボリュームを置き換えることができます。

- インスタンスストアボリュームで復元されたデータ – インスタンスストアボリュームは、ルートボリュームが復元された後も、インスタンスにアタッチされたままになります。
- データ (非ルート) Amazon EBS ボリュームに保存されたデータ - 非ルート Amazon EBS ボリュームは、ルートボリュームが復元された後もインスタンスに接続されたままです。
- ネットワーク設定 — すべてのネットワークインターフェイスはインスタンスにアタッチされたままとなり、IP アドレス、識別子、およびアタッチメント ID を保持します。インスタンスが利用可能になると、保留中のネットワークトラフィックがすべてフラッシュされます。さらに、インスタンスは同じ物理ホスト上に残るため、パブリック IP アドレス、プライベート IP アドレス、および DNS 名が保持されます。
- IAM ポリシー — インスタンスに関連付けられた IAM プロファイルとポリシー (タグベースのポリシーなど) は保持され、適用されます。

トピック

- 動作の仕組み
- ルートボリュームを置き換える

- ルートボリューム置換タスクを表示する

動作の仕組み

インスタンスのルートボリュームを置き換えると、新しい (置換) ルートボリュームは、次のいずれかの方法で復元されます。

- 初期起動状態へ — ボリュームはインスタンス起動時の初期状態に復元されます。詳細については、「ルートボリュームを開始状態に復元する」を参照してください。
- 現在のルートボリュームと同じ系統のスナップショットから — これにより、ルートボリュームの破損やゲスト OS ネットワーク設定エラーなどの問題を修正できます。詳細については、「スナップショットを使用したボリュームの置き換え」を参照してください。
- インスタンスと同じキー属性を持つ AMI から — これにより、オペレーティングシステムとアプリケーションのパッチ適用やアップグレードを実行できます。詳細については、「AMI を使用したボリュームの置き換え」を参照してください。

元のルートボリュームがインスタンスからデタッチされ、その代わりに新しいルートボリュームがインスタンスにアタッチされます。インスタンスのブロックデバイスマッピングは、置換用ルートボリュームの ID を反映するように更新されます。ルートボリュームの置換プロセスが完了した後も、元のルートボリュームを維持するかどうかを選択できます。置換プロセスの完了後に元のルートボリュームを削除すると、元のルートボリュームは自動的に削除され、回復できなくなります。処理が完了した後も元のルートボリュームを維持することを選択した場合、ボリュームはアカウントにプロビジョニングされたままになるため、不要になったら手動で削除する必要があります。

ルートボリュームの置き換えタスクが失敗した場合、インスタンスは再起動され、元のルートボリュームはインスタンスにアタッチされたままになります。

ルートボリュームの置き換えに関する考慮事項

- インスタンスは `running` の状態である必要があります。
- インスタンスは、プロセス中に自動的に再起動されます。メモリ (RAM) の内容は、再起動中に消去されます。手動で再起動する必要はありません。
- インスタンスストアボリュームの場合、ルートボリュームを置き換えることはできません。Amazon EBS ルートボリュームを持つインスタンスのみに対応しています。
- ルートボリュームを置き換えることができるのは、すべての仮想化インスタンスタイプと EC2 Mac ベアメタルインスタンスのみです。その他のベアメタルインスタンスタイプには対応していません。

- インスタンスの以前のルートボリュームのいずれかと同じ系統に属する任意のスナップショットを使用できます。
- 現在のリージョンで、アカウントのAmazon EBS の暗号化がデフォルトで有効になっていると、指定したスナップショットや指定した AMI のルートボリュームにおける暗号化ステータスに関係なく、ルートボリューム置換タスクによって作成された置換用ルートボリュームは常に暗号化されます。
- 次の表は、考えられる暗号化の結果をまとめたものです。

	元のルートボリューム	指定したスナップショットまたは AMI	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
置換用ルートボリュームを起動状態に復元する	暗号化された	該当しない	考慮しない	暗号化された	元のルートボリュームと同じ KMS キー
	暗号化されていない	該当しない	無効	暗号化されていない	該当しない
	暗号化されていない	該当しない	有効	暗号化された	アカウントの Amazon EBS 暗号化用のデフォルト KMS キー
スナップショットまたは AMI から置換用ルートボリュームを復元する	暗号化された	暗号化されていない	考慮しない	暗号化された	元のルートボリュームと同じ KMS キー
	暗号化された	暗号化された	考慮しない	暗号化された	元のルートボリュームと同じ KMS キー
	暗号化されていない	暗号化されていない	無効	暗号化されていない	該当しない

	元のルートボリューム	指定したスナップショットまたはAMI	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
	暗号化されていない	暗号化されていない	有効	暗号化された	アカウントのAmazon EBS暗号化用のデフォルトKMS キー

	元のルートボリューム	指定したスナップショットまたはAMI	デフォルトでの暗号化	置換用ルートボリューム	置換用ルートボリュームに使用される暗号化キー
	暗号化されていない	暗号化された	考慮しない	暗号化された	AMI またはスナップショットがアカウントで所有されている場合、置換用ボリュームは、その AMI またはスナップショットの KMS キーで暗号化されます。アカウントが AMI またはスナップショットを共有している場合、置換用ボリュームは、そのアカウントの デフォルトの Amazon EBS 暗号化の KMS キーで暗号化されます。

トピック

- ルートボリュームを開始状態に復元する

- スナップショットを使用したボリュームの置き換え
- AMI を使用したボリュームの置き換え

ルートボリュームを開始状態に復元する

インスタンスのルートボリュームを、元のルートボリュームの起動状態に復元された置換用ルートボリュームに置き換えるルートボリュームの置換を実行できます。置換用ボリュームは、インスタンスの起動中に元のボリュームの作成に使用されたスナップショットから自動的に復元されます。

置換用ルートボリュームには、元のルートボリュームと同じタイプ、サイズ、終了時の削除属性が割り当てられます。

スナップショットを使用したボリュームの置き換え

インスタンスのルートボリュームを、特定のスナップショットに復元された置換用ボリュームで置き換えるルートボリューム置換を実行することができます。これにより、インスタンスのルートボリュームを、そのルートボリュームから以前に作成した特定のスナップショットに復元できます。

置換用ルートボリュームには、元のルートボリュームと同じタイプ、サイズ、終了時の削除属性が割り当てられます。

スナップショット使用時の考慮事項

- インスタンスの現在のルートボリュームと同じ系統に属するスナップショットのみを使用できます。
- ルートボリュームから作成したスナップショットにより作成されたスナップショットのコピーは使用できません。
- ルートボリュームの置き換えに成功した後も、元のルートボリュームから取得したスナップショットを使用して、新しい (置換) ルートボリュームに置き換えることができます。

AMI を使用したボリュームの置き換え

ルートボリュームの置き換えは、所有する AMI、共有されている AMI を使用して実行できます。AMI には、インスタンスと同じ製品コード、請求情報、アーキテクチャタイプ、仮想化タイプが必要です。

インスタンスで NitroTPM、ENA、または sriov-net が有効になっている場合は、これらの機能をサポートする AMI を使用する必要があります。インスタンスで NitroTPM、ENA、または sriov-net が

有効になっていない場合は、それらの機能をサポートしない AMI を選択するか、それらをサポートする AMI を選択できます。その場合は、インスタンスにサポートが追加されます。

インスタンスが AMI のブートモードをサポートしている場合は、インスタンスとは異なるブートモードの AMI を選択できます。インスタンスがブートモードをサポートしていない場合、リクエストは失敗します。インスタンスがブートモードをサポートしている場合、新しいブートモードがインスタンスに伝達され、それに応じてその UEFI データが更新されます。ブート順序を手動で変更したり、プライベートカーネルモジュールをロードするためにプライベート UEFI セキュアブートキーを追加したりした場合、ルートボリュームの置換時に変更内容が失われます。

置換用ルートボリュームには、元のルートボリュームと同じボリュームタイプ、削除時の削除属性が割り当てられ、AMI ルートボリュームのブロックデバイスマッピングのサイズを取得します。

> **Note**
>
> AMI ルートボリュームのブロックデバイスマッピングのサイズは、元のルートボリュームのサイズ以上である必要があります。AMI ルートボリュームのブロックデバイスマッピングのサイズが元のルートボリュームのサイズよりも小さい場合、リクエストは失敗します。

ルートボリュームの置換タスクの完了後、コンソール、AWS CLI または AWS SDK を使用してインスタンスを記述すると、次の新しい情報および更新された情報が反映されます。

- 新しい AMI ID
- ルートボリュームの新しいボリューム ID
- 更新されたブートモード設定 (AMI によって変更された場合)
- 更新された NitroTPM 設定 (AMI によって有効になっている場合)
- 更新された ENA 設定 (AMI によって有効になっている場合)
- 更新された sriov-net 設定 (AMI によって有効になっている場合)

新しい AMI ID はインスタンスメタデータにも反映されます。

AMI を使用ためのする考慮事項

- ブロックデバイスマッピングが複数ある AMI を使用する場合、AMI のルートボリュームのみが使用されます。他の (非ルート) ボリュームは無視されます。

- この機能を使用できるのは、AMI とそれに関連するルートボリュームスナップショットに対するアクセス許可を持っている場合のみです。この機能は AWS Marketplace AMI では使用できません。
- インスタンスに製品コードがない場合にのみ、製品コードなしの AMI を使用できます。
- AMI ルートボリュームのブロックデバイスマッピングのサイズは、元のルートボリュームのサイズ以上である必要があります。AMI ルートボリュームのブロックデバイスマッピングのサイズが元のルートボリュームのサイズよりも小さい場合、リクエストは失敗します。
- インスタンスのインスタンス ID ドキュメントは自動的に更新されます。
- インスタンスが NitroTPM に対応している場合、インスタンスの NitroTPM データがリセットされ、新しいキーが生成されます。

ルートボリュームを置き換える

インスタンスのルートボリュームを復元すると、ルートボリュームの置換タスクが作成されます。ルートボリュームの置換タスクを使用して、置換プロセスの進行状況と結果をモニタリングできます。詳細については、「ルートボリューム置換タスクを表示する」を参照してください。

次のいずれかの方法を使用して、インスタンスのルートボリュームを置き換えることができます。

> **Note**
>
> Amazon EC2 コンソールを使用する場合、この機能は新しいコンソールでのみ使用できます。

New console

ルートボリュームを置き換えるには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. ルートボリュームを置き換えるインスタンスを選択し、[Actions] (アクション)、[Monitor and troubleshoot] (監視とトラブルシューティング)、[Replace root volume] (ルートボリュームを置換) の順に選択します。

> **Note**
>
> 選択したインスタンスが `running` 状態でない場合、[ルートボリュームを置き換える] アクションは無効です。

4. [Replace root volume] (ルートボリュームを置換) の画面で、次のいずれかの操作を行います。

   - 置換用のルートボリュームを初期起動状態に復元するには、スナップショットを選択せずに [Create replacement task] (置換タスクを作成) を選択します。
   - 置換用のルートボリュームを特定のスナップショットに復元するには、[Snapshot] (スナップショット) で、使用するスナップショットを選択し、[Create replacement task] (置換タスクを作成) を選択します。
   - AMI を使用して置換用ルートボリュームを復元するには、[AMI] で使用する AMI を選択し、[Create replacement task] (置換タスクを作成) を選択します。

5. 置換タスクの完了後に元のルートボリュームを削除するには、[Delete replaced root volume] (置き換えられたルートボリュームを削除) を選択します。

AWS CLI

置換用ルートボリュームを起動状態に復元するには

create-replace-root-volume-task コマンドを使用します。`--instance-id` には、ルートボリュームを置き換えるインスタンスの ID を指定します。`--snapshot-id` および `--image-id` のパラメータは省略します。置換後に元のルートボリュームを削除するには、`--delete-replaced-root-volume` を含めて、`true` を指定してください。

```
$ aws ec2 create-replace-root-volume-task \
--instance-id i-1234567890abcdef0 \
--delete-replaced-root-volume true
```

置換用ルートボリュームを特定のスナップショットに復元するには

create-replace-root-volume-task コマンドを使用します。`--instance-id` には、ルートボリュームを置き換えるインスタンスの ID を指定します。`--snapshot-id` には、使用するスナップショットの ID を指定します。置換後に元のルートボリュームを削除するには、`--delete-replaced-root-volume` を含めて、`true` を指定してください。

```
$ aws ec2 create-replace-root-volume-task \
--instance-id i-1234567890abcdef0 \
--snapshot-id snap-9876543210abcdef0 \
--delete-replaced-root-volume true
```

AMI を使用して置換用ルートボリュームを復元するには

create-replace-root-volume-task コマンドを使用します。--instance-id には、ルートボリュームを置き換えるインスタンスの ID を指定します。--image-id に使用する AMI の ID を指定します。置換後に元のルートボリュームを削除するには、--delete-replaced-root-volume を含めて、true を指定してください。

```
$ aws ec2 create-replace-root-volume-task \
--instance-id i-01234567890abcdef \
--image-id ami-09876543210abcdef \
--delete-replaced-root-volume true
```

Tools for Windows PowerShell

置換用ルートボリュームを起動状態に復元するには

New-EC2ReplaceRootVolumeTask コマンドを使用します。-InstanceId には、ルートボリュームを置き換えるインスタンスの ID を指定します。-SnapshotId および -ImageId のパラメータは省略します。置換後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めて、$true を指定してください。

```
PS C:\> New-EC2ReplaceRootVolumeTask -InstanceId i-1234567890abcdef0 -
DeleteReplacedRootVolume $true
```

置換用ルートボリュームを特定のスナップショットに復元するには

New-EC2ReplaceRootVolumeTask コマンドを使用します。--InstanceId には、ルートボリュームを置き換えるインスタンスの ID を指定します。-SnapshotId には、使用するスナップショットの ID を指定します。置換後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めて、$true を指定してください。

```
PS C:\> New-EC2ReplaceRootVolumeTask -InstanceId i-1234567890abcdef0 -
SnapshotId snap-9876543210abcdef0 -DeleteReplacedRootVolume $true
```

AMI を使用して置換用ルートボリュームを復元するには

New-EC2ReplaceRootVolumeTask コマンドを使用します。-InstanceId には、ルートボリュームを置き換えるインスタンスの ID を指定します。-ImageId に使用する AMI の ID を指定します。置換後に元のルートボリュームを削除するには、-DeleteReplacedRootVolume を含めて、$true を指定してください。

```
PS C:\> New-EC2ReplaceRootVolumeTask -InstanceId i-1234567890abcdef0 -
ImageId ami-09876543210abcdef -DeleteReplacedRootVolume $true
```

ルートボリューム置換タスクを表示する

インスタンスのルートボリュームを復元すると、ルートボリュームの置換タスクが作成されます。ルートボリュームの置き換えタスクは、プロセス中に次の状態に移行します。

- pending — 置換ボリュームが作成されています。
- in-progress — 元のボリュームがデタッチされ、置換ボリュームがアタッチされています。
- succeeded — 置換ボリュームはインスタンスに正常にアタッチされ、インスタンスは利用可能です。
- failing — 置換タスクが失敗を処理しています。
- failed — 置換タスクは失敗しましたが、元のルートボリュームはまだアタッチされています。
- failing-detached — 置換タスクが正常に処理されていません。インスタンスにルートボリュームがアタッチされていない可能性があります。
- failed-detached — 置換タスクが失敗し、インスタンスにルートボリュームがアタッチされていません。

次のいずれかの方法を使用して、インスタンスのルートボリューム置換タスクを表示できます。

> **Note**
>
> Amazon EC2 コンソールを使用する場合、この機能は新しいコンソールでのみ使用できます。

Console

ルートボリュームの置換タスクを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. ルートボリューム置換タスクを表示するインスタンスを選択し、[Storage] (ストレージ) タブを選択します。
4. [Storage] (ストレージ) タブで、[Recent root volume replacement tasks] (最近のルートボリューム置換タスク) を展開します。

AWS CLI

ルートボリューム置換タスクのステータスを表示するには

describe-replace-root-volume-task コマンドを使用して、表示するルートボリューム置換タスクの ID を指定します。

```
$ aws ec2 describe-replace-root-volume-tasks \
--replace-root-volume-task-ids replacevol-1234567890abcdef0
```

```
{
  "ReplaceRootVolumeTasks": [
  {
    "ReplaceRootVolumeTaskId": "replacevol-1234567890abcdef0",
    "InstanceId": "i-1234567890abcdef0",
    "TaskState": "succeeded",
    "StartTime": "2020-11-06 13:09:54.0",
    "CompleteTime": "2020-11-06 13:10:14.0",
    "SnapshotId": "snap-01234567890abcdef",
    "DeleteReplacedRootVolume": "True"
  }]
}
```

または、instance-id フィルターを指定して、結果をインスタンス別にフィルタリングします。

```
$ aws ec2 describe-replace-root-volume-tasks \
--filters Name=instance-id,Values=i-1234567890abcdef0
```

Tools for Windows PowerShell

ルートボリューム置換タスクのステータスを表示するには

Get-EC2ReplaceRootVolumeTask コマンドを使用して、表示するルートボリューム置換タスクの ID を指定します。

```
PS C:\> Get-EC2ReplaceRootVolumeTask -
ReplaceRootVolumeTaskIds replacevol-1234567890abcdef0
```

または、instance-id フィルターを指定して、結果をインスタンス別にフィルタリングします。

```
PS C:\> Get-EC2ReplaceRootVolumeTask -Filters @{Name = 'instance-id'; Values =
 'i-1234567890abcdef0'} | Format-Table
```

Linux インスタンスでのデバイス名

インスタンスにボリュームをアタッチする場合は、ボリュームのデバイス名を含めます。このデバイス名は Amazon EC2 によって使用されます。インスタンスのブロックデバイスドライバーは、ボリュームのマウント時に実際のボリューム名を割り当てますが、この割り当てられた名前は、Amazon EC2 が使用する名前とは異なる可能性があります。

インスタンスがサポートできるボリュームの数は、オペレーティングシステムによって決まります。詳細については、インスタンスボリューム数の制限を参照してください。

コンテンツ

Windows インスタンスのデバイス名の詳細については、Windows インスタンスの Amazon EC2 ユーザーガイド のWindows インスタンスでのデバイスの名前付けを参照してください。

使用できるデバイス名

Linux インスタンスでは、準仮想化 (PV) とハードウェア仮想マシン (HVM) の 2 種類の仮想化を使用できます。インスタンスの仮想化タイプは、インスタンスの起動に使用される AMI によって決まります。すべてのインスタンスタイプが HVM AMI をサポートしています。一部の前世代のインスタン

スタイプは PV AMI をサポートしています。使用できる推奨のデバイス名はインスタンスの仮想化タイプによって異なるため、必ず AMI の仮想化タイプを確認してください。詳細については、Linux AMI 仮想化タイプを参照してください。

次の表に、ブロックデバイスマッピングまたは EBS ボリュームの接続時に指定できる使用可能なデバイス名を示します。

仮想化タイプ	利用可能	ルートボリューム用に予約済み	EBS ボリュームとして推奨	インスタンスストアボリューム
準仮想化	/dev/sd[a-z] /dev/sd[a-z][1-15] /dev/hd[a-z] /dev/hd[a-z][1-15]	/dev/sda1	/dev/sd[f-p] /dev/sd[f-p][1-6]	/dev/sd[b-e]
HVM	/dev/sd[a-z] /dev/xvd[a-d][a-z] /dev/xvd[e-z]	AMI による違い /dev/sda1 or /dev/xvda	/dev/sd[f-p] *	/dev/sd[b-e] /dev/sd[b-h] (h1.16xlarge) /dev/sd[b-y] (d2.8xlarge) /dev/sd[b-i] (i2.8xlarge) **

* ブロックデバイスマッピングの NVMe EBS ボリュームで指定したデバイス名は、NVMe デバイス名 (`/dev/nvme[0-26]n1`) を使用して名称変更されます。ブロックデバイスドライバーは、ブロックデバイスマッピングのボリュームに指定した順序とは異なる順序で NVMe デバイス名を割り当てることができます。

** NVMe インスタンスストアボリュームは自動的に列挙され、NVMe デバイス名が割り当てられます。

インスタンスストアボリュームの詳細については、Amazon EC2 インスタンスストア を参照してください。EBS デバイスの識別方法を含む、NVMe EBS ボリューム (Nitro ベースのインスタンス) の詳細については、「Amazon EBS ユーザーガイド」の「Amazon EBS および NVMe」を参照してください。

デバイス名に関する考慮事項

デバイス名を選択するときは、以下の点を常に考慮する必要があります。

- インスタンスストアボリュームをアタッチするために使用されたデバイス名を使用して EBS ボリュームをアタッチすることはできますが、動作を予測できない場合があるため、この方法は使用しないことを強くお勧めします。
- インスタンスの NVMe インスタンスストアボリュームの数は、インスタンスのサイズによって異なります。NVMe インスタンスストアボリュームは自動的に列挙され、NVMe デバイス名 (/dev/nvme[0-26]n1)
- カーネルのブロックデバイスドライバによっては、指定した名前とは異なる名前でデバイスがアタッチされる可能性があります。例えば、デバイス名 (/dev/sdh) を指定した場合、デバイスの名前が /dev/xvdh や /dev/hdh に変更される場合があります。ほとんどの場合、末尾の文字は変更されません。Red Hat Enterprise Linux (および CentOS などのバリアント) の一部バージョンでは、末尾の文字が変更されることがあります (/dev/sda が /dev/xvde になることがあります)。このような場合、各デバイス名の末尾の文字は同じ規則で変更されます。例えば、/dev/sdb の名前が /dev/xvdf に変更されると、/dev/sdc の名前は /dev/xvdg に変更されます。Amazon Linux は、名前が変更されたデバイスに対して指定した名前のシンボリックデバイスを作成します。他のオペレーティングシステムの動作は異なる場合があります。
- HVM AMI では、ルートデバイス用に予約されているデバイス名 (/dev/sda1) や /dev/sda2 を除き、デバイス名の末尾に数字を使用することをサポートしていません。/dev/sda2 は使用できますが、HVM インスタンスでこのデバイスマッピングを使用することは推奨していません。
- PV AMI を使用する場合は、末尾の番号の有無にかかわらず、同じデバイス文字を共有するボリュームをアタッチすることはできません。例えば、あるボリュームを /dev/sdc としてアタッチし、別のボリュームを /dev/sdc1 としてアタッチした場合、インスタンスは /dev/sdc のみを認識します。デバイス名の末尾に数字を使用するには、同じベース文字を共有するすべてのデバイス名の末尾に数字を使用する必要があります (例:/dev/sdc1 、/dev/sdc2、/dev/sdc3)。
- 一部のカスタムカーネルでは、使用できるデバイス名が /dev/sd[f-p] や /dev/sd[f-p][1-6] に制限されている場合があります。/dev/sd[q-z] または /dev/sd[q-z][1-6] の使用に関して問題がある場合は、/dev/sd[f-p] または /dev/sd[f-p][1-6] に切り替えてみてください。

ブロックデバイスマッピング

起動する各インスタンスには、Amazon EBS ボリューム、またはインスタンスストアボリュームのどちらかのルートデバイスボリュームが関連付けられています。ブロックデバイスマッピングを使用すると、インスタンスの起動時にそのインスタンスにアタッチする追加の EBS ボリュームまたはインスタンスストアボリュームを指定できます。追加する EBS ボリュームは、実行中のインスタンスにアタッチすることもできます。ただし、インスタンスストアボリュームについては、ブロックデバイスマッピングを使用して、インスタンスの起動時にアタッチする以外方法はありません。

ルートデバイスボリュームの詳細については、永続的ルートボリュームへの変更を参照してください。

コンテンツ

ブロックデバイスマッピングの概念

ブロックデバイスは、一連のバイトまたはビット (ブロック) でデータを移動するストレージデバイスです。これらのデバイスはランダムアクセスをサポートし、バッファ付き I/O を使用します。例にはハードディスク、CD-ROM ドライブ、フラッシュドライブが含まれます。ブロックデバイスは物理的にコンピュータにアタッチできます。また、コンピュータに物理的にアタッチされているかのように、リモートでアクセスすることもできます。

Amazon EC2 は、2 種類のブロックデバイスをサポートしています。

- インスタンスストアボリューム (基盤となるハードウェアがインスタンスのホストコンピュータに物理的にアタッチされている仮想デバイス)
- EBS ボリューム (リモートストレージデバイス)

ブロックデバイスマッピングでは、インスタンスにアタッチするブロックデバイス (インスタンスストアボリュームと EBS ボリューム) を定義します。ブロックデバイスマッピングは、AMI 作成プロセスの一環として、AMI から起動されるすべてのインスタンスによって使用されるように指定できます。また、インスタンスの起動時にブロックデバイスマッピングを指定することもできます。起動したインスタンスの AMI ですでに指定されているマッピングは、このマッピングによって上書きさ

れます。インスタンスタイプによってサポートされるすべての NVMe インスタンスストアボリュームが自動的に列挙され、インスタンスの起動時にデバイス名が割り当てられることに注意してください。それらをブロックデバイスマッピングに含めます。含めないとインスタンスは効果がありません。

コンテンツ

ブロックデバイスマッピングのエントリ

ブロックデバイスマッピングを作成するとき、インスタンスにアタッチする必要があるブロックデバイスごとに以下の情報を指定します。

- Amazon EC2 内で使用されるデバイス名。インスタンスのブロックデバイスドライバーは、ボリュームをマウントするときに実際のボリューム名を割り当てます。この割り当てられた名前は、Amazon EC2 が推奨する名前とは異なる可能性があります。詳細については、Linux インスタンスでのデバイス名を参照してください。

インスタンスストアボリュームの場合は、次の情報も指定します。

- 仮想デバイスの名前: `ephemeral[0-23]`。インスタンスで使用できるインスタンスストアボリュームの数とサイズは、インスタンスタイプによって異なります。

NVMe インスタンスストアボリュームの場合は、次の情報も適用されます。

- これらのボリュームが自動的に列挙され、デバイス名が割り当てられます。それらをブロックデバイスマッピングに含めます。含めないとインスタンスは効果がありません。

EBS ボリュームの場合は、次の情報も指定します。

- ブロックデバイスを作成するときに使用するスナップショットの ID (snap-xxxxxxxx)。ボリュームサイズを指定する場合、この値はオプションです。アーカイブされたスナップショットの ID は指定できません。

- ボリュームのサイズ (GiB 単位) 。指定されたサイズは、指定されたスナップショットのサイズ以上である必要があります。
- インスタンス終了時にボリュームを削除するかどうか (`true` または `false`) デフォルト値は、ルートデバイスボリュームでは `true`、アタッチされたボリュームでは `false` です。AMI を作成するときは、そのブロックデバイスマッピングがインスタンスからこの設定を継承します。インスタンスを起動するときに、AMI からこの設定を継承します。
- ボリュームタイプ。汎用 SSD の場合は `gp2` および `gp3`、プロビジョンド IOPS SSD の場合は `io1` および `io2`、スループット最適化 HDD の場合は `st1`、Cold HDD の場合は `sc1`、磁気の場合は `standard` です。
- ボリュームがサポートする 1 秒あたりの入力/出力オペレーションの数 (IOPS) (`io1` および `io2` ボリュームでのみ使用)

ブロックデバイスマッピングのインスタンスストアの注意事項

ブロックデバイスマッピングでインスタンスストアボリュームがある場合は、インスタンスを AMIs から起動すると、いくつかの警告が表示されます。

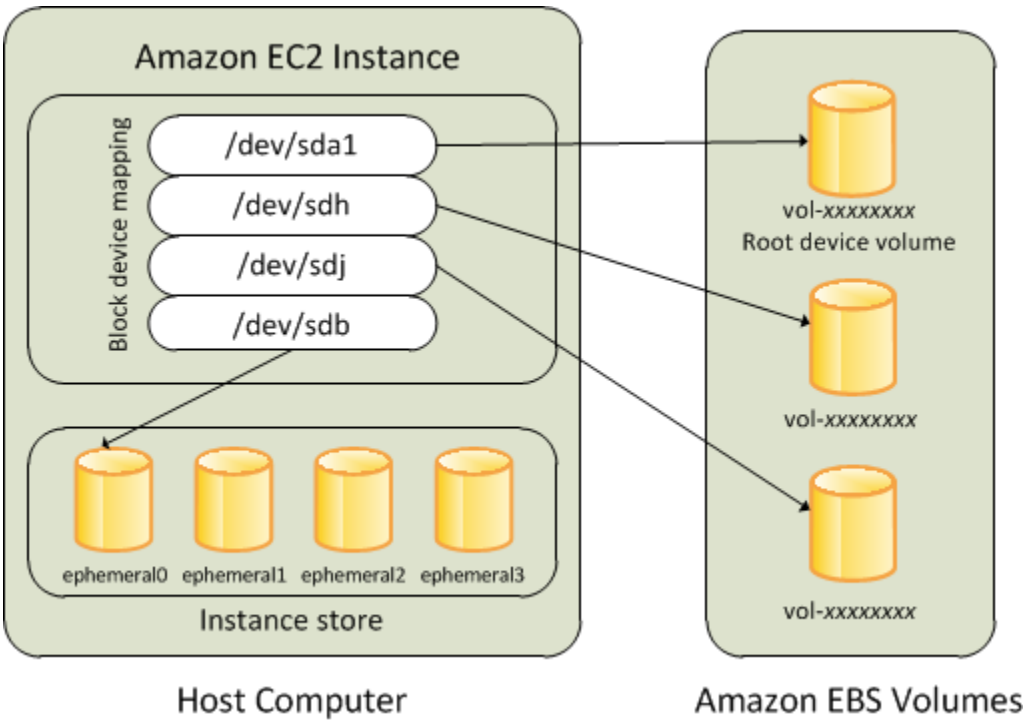
- インスタンスタイプによって中に含まれるインスタンスストアボリューム数が異なり、インスタンスストアボリュームがまったく含まれないインスタンスタイプもあります。単一インスタンスストアボリュームのみをサポートするインスタンスタイプで、AMI が 2 つのインスタンスストアボリュームにマッピングされている場合、インスタンスは単一のインスタンスストアボリュームのみで起動します。
- インスタンスストアボリュームをマッピングできるのは、起動時のみに限られます。インスタンスストアボリュームのないインスタンスを停止することはできません (`t2.micro` など)。インスタンスストアボリュームをサポートするインスタンスに変更し、インスタンスストアボリュームを含めて再起動します。ただし、AMI をインスタンスから作成し、インスタンスストアボリュームをサポートするインスタンスタイプで起動して、インスタンスストアボリュームをインスタンスにマッピングすることは可能です。
- インスタンスストアボリュームをマッピングしたインスタンスを起動し、インスタンスを停止して、インスタンスストアボリュームの少ないインスタンスタイプに変更して再開すれば、最初の起動からマッピングしたインスタンスストアボリュームもインスタンスのメタデータに表示されます。ただし、インスタンスに使用できるのは、そのインスタンスタイプでサポートされているインスタンスストアボリュームの最大数までです。

> **Note**
>
> インスタンスが停止されると、インスタンスストアボリュームのデータはすべて失われます。

- 起動時のインスタンスストア容量によっては、M3 インスタンスが AMI インスタンスストアブロックデバイスのマッピングを (起動時に指定されていない限り) 無視します。インスタンスの起動時にインスタンスストアボリュームを使用するには、起動する AMI ボリュームに AMI でインスタンスストアボリュームがマッピングされていたとしても、起動時にインスタンスストアブロックデバイスのマッピングを指定する必要があります。

ブロックデバイスマッピングの例

この図は、EBS-backed インスタンスのブロックデバイスマッピングの例を示しています。この例では、/dev/sdb を ephemeral0 にマッピングし、2 つの EBS ボリュームを 1 つは /dev/sdh に、もう 1 つは /dev/sdj にマッピングします。また、ルートデバイスボリュームである EBS ボリューム、/dev/sda1 も示しています。



このブロックデバイスマッピングの例は、このトピックのコマンドおよび API の例で使用されています。ブロックデバイスマッピングを作成するコマンドおよび API の例については、AMI 用のブ

ロックデバイスマッピングの指定およびインスタンス起動時のブロックデバイスマッピングの更新を参照してください。

オペレーティングシステムでデバイスを使用できるようにする方法

Amazon EC2 では、ブロックデバイスの記述に、/dev/sdh や xvdh などのデバイス名が使われます。また、Amazon EC2 では、EC2 インスタンスにアタッチするブロックデバイスを、ブロックデバイスマッピングで指定します。ストレージデバイスにアクセスするには、インスタンスにアタッチしたブロックデバイスが、オペレーティングシステムによって事前にマウントされていなければなりません。ブロックデバイスがインスタンスからデタッチされると、そのデバイスはオペレーティングシステムによってアンマウントされ、ストレージデバイスにアクセスできなくなります。

Linux インスタンスの場合、ブロックデバイスマッピングで指定されたデバイス名は、インスタンスの初回起動時に対応するブロックデバイスにマッピングされます。デフォルトでフォーマットおよびマウントされるインスタンスストアボリュームは、インスタンスタイプによって決まります。インスタンスタイプで使用できるインスタンスストアボリューム数を超えていない場合は、起動時に追加のインスタンスストアボリュームをマウントできます。詳細については、Amazon EC2 インスタンスストアを参照してください。ボリュームがフォーマットおよびマウントされるときに使用されるデバイスは、インスタンスのブロックデバイスドライバーによって決まります。

AMI ブロックデバイスマッピング

各 AMI にブロックデバイスマッピングがあります。このブロックデバイスマッピングは、AMI からのインスタンスの起動時にそのインスタンスにアタッチするブロックデバイスを指定します。追加のブロックデバイスを AMI に追加するには、独自の AMI を作成する必要があります。

コンテンツ

- AMI 用のブロックデバイスマッピングの指定
- AMI ブロックデバイスマッピングの EBS ボリュームの表示

AMI 用のブロックデバイスマッピングの指定

AMI を作成する場合に、ルートボリュームに加えて、ボリュームを指定するには、2 つの方法があります。インスタンスから AMI を作成する前に、実行中のインスタンスにすでにボリュームをアタッチしている場合、AMI のブロックデバイスマッピングにそれらの同じボリュームが含まれます。EBS ボリュームの場合、既存のデータが新しいスナップショットに保存され、それがブロックデバイスマッピングで指定される新しいスナップショットになります。インスタンスストアボリュームの場合、データは維持されません。

EBS-backed AMI の場合、ブロックデバイスマッピングを使用して、EBS ボリュームとインスタンスストアボリュームを追加できます。instance store-backed AMI の場合、イメージの登録時にイメージマニフェストファイルでブロックデバイスマッピングエントリを変更して、インスタンスストアボリュームのみを追加できます。

> **Note**
>
> M3 インスタンスの場合、インスタンスのブロックデバイスマッピングで起動時にインスタンスストアボリュームを指定する必要があります。AMI のブロックデバイスマッピングで指定したインスタンスストアボリュームは、インスタンスブロックデバイスマッピングの一部として指定されていない場合、M3 インスタンスを起動した際に無視される可能性があります。

コンソールを使用してボリュームを AMI に追加するには

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[アクション]、[イメージとテンプレート]、[イメージの作成] の順に選択します。
4. イメージの名前と説明を入力します。
5. インスタンスボリュームが [インスタンスボリューム] の下に表示されます。別のボリュームを追加するには、[ボリュームの追加] を選択します。
6. [Volume type] で、ボリュームタイプを選択します。[Device] で、デバイス名を選択します。EBS ボリュームでは、スナップショット、ボリュームサイズ、ボリュームタイプ、IOPS、暗号化状態などの追加の詳細を指定できます。
7. [イメージを作成] を選択します。

コマンドラインを使用して AMI にボリュームを追加するには

EBS-Backed AMI のブロックデバイスマッピングを指定するには、create-image AWS CLI コマンドを使用します。Instance Store-Backed AMI のブロックデバイスマッピングを指定するには、register-image AWS CLI コマンドを使用します。

`--block-device-mappings` パラメータを使用してブロックデバイスマッピングを指定します。JSON でエンコードされた引数は、コマンドラインで直接指定することも、ファイルを参照して指定することもできます。

```
--block-device-mappings [mapping, ...]
--block-device-mappings [file://mapping.json]
```

インスタンスストアボリュームを追加するには、次のマッピングを使用します。

```
{
    "DeviceName": "/dev/sdf",
    "VirtualName": "ephemeral0"
}
```

空の 100 GiB gp2 ボリュームを追加するには、次のマッピングを使用します。

```
{
    "DeviceName": "/dev/sdg",
    "Ebs": {
      "VolumeSize": 100
    }
}
```

スナップショットに基づいた EBS ボリュームを追加するには、次のマッピングを使用します。

```
{
    "DeviceName": "/dev/sdh",
    "Ebs": {
      "SnapshotId": "snap-xxxxxxxx"
    }
}
```

デバイスのマッピングを省略するには、次のマッピングを使用します。

```
{
    "DeviceName": "/dev/sdj",
    "NoDevice": ""
}
```

または、次のコマンド (AWS Tools for Windows PowerShell) で `-BlockDeviceMapping` パラメータを使用することもできます。

- New-EC2Image

- Register-EC2Image

AMI ブロックデバイスマッピングの EBS ボリュームの表示

AMI のブロックデバイスマッピングの EBS ボリュームを簡単に列挙できます。

コンソールを使用して AMI の EBS ボリュームを表示するには

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインで [AMIs] を選択します。
3. [Filter] リストから [EBS images] を選択して、EBS-Backed AMI のリストを取得します。
4. ご希望の AMI を選択し、[Details] タブを確認します。少なくとも、ルートデバイスでは次の情報を使用できます。

   - ルートデバイスタイプ (ebs)
   - [ルートデバイス名] (例: /dev/sda1)
   - [Block Devices] (例: /dev/sda1=snap-1234567890abcdef0:8:true)

   AMI がブロックデバイスマッピングを使用して追加の EBS ボリュームで作成された場合、[Block Devices] フィールドには、その追加の EBS ボリュームのマッピングも表示されます。(この画面には、インスタンスストアボリュームは表示されません。)

コマンドラインを使用して AMI の EBS ボリュームを表示するには

describe-images (AWS CLI) コマンドまたは Get-EC2Image (AWS Tools for Windows PowerShell) コマンドを使用して、AMI のブロックデバイスマッピング内の EBS ボリュームを列挙します。

インスタンスブロックデバイスマッピング

デフォルトでは、起動するインスタンスには、そのインスタンスを起動した AMI のブロックデバイスマッピングで指定されたストレージデバイスが含まれます。インスタンスを起動するときに、インスタンスのブロックデバイスマッピングへの変更を指定できます。この変更は AMI のブロックデバイスマッピングを上書きするか、このブロックデバイスマッピングに統合されます。

制限事項

- ルートボリュームの場合、変更できるのはボリュームサイズ、ボリュームタイプ、および [合わせて削除] フラグのみです。

- EBS ボリュームを変更する場合、そのサイズを小さくすることはできません。そのため、指定するスナップショットのサイズは、AMI のブロックデバイスマッピングで指定されたスナップショットのサイズ以上であることが必要です。

コンテンツ

インスタンス起動時のブロックデバイスマッピングの更新

インスタンスの起動時に、EBS ボリュームとインスタンスストアボリュームをインスタンスに追加できます。インスタンスのブロックデバイスマッピングを更新しても、そのインスタンスが起動された AMI のブロックデバイスマッピングは完全には変更されないことに注意してください。

コンソールを使用してボリュームをインスタンスに追加するには

1. Amazon EC2 コンソールを開きます。
2. ダッシュボードから、[Launch Instance] を選択します。
3. [Choose an Amazon Machine Image (AMI)] ページで、使用する AMI を選択し、[Select] を選択します。
4. ウィザードにしたがって [Choose an Instance Type] ページと [Configure Instance Details] ページを設定します。
5. [Add Storage] ページで、以下のようにルートボリューム、EBS ボリューム、およびインスタンスストアボリュームを変更できます。

   - ルートボリュームのサイズを変更するには、[Type] 列で [Root] ボリュームを見つけて、[Size] フィールドを変更します。
   - インスタンスの起動に使用された AMI のブロックデバイスマッピングで指定された EBS ボリュームを削除するには、ボリュームを見つけて、[Delete] アイコンをクリックします。
   - EBS ボリュームを追加するには、[Add New Volume] (新しいボリュームの追加]) を選択し、[Type] (タイプ) リストから [EBS] を選択して、各フィールド ([Device] (デバイス)、[Snapshot] (スナップショット) など) に入力します。

- インスタンスの起動に使用された AMI のブロックデバイスマッピングで指定されたインスタンスストアボリュームを削除するには、ボリュームを見つけて、[Delete] アイコンを選択します。
- インスタンスストアボリュームを追加するには、[新しいボリュームの追加] を選択し、[Type] リストから [インスタンスストア] を選択して、[Device] からデバイス名を選択します。

6. ウィザードの残りのページを完了した後、[起動] を選択します。

AWS CLI を使用してボリュームをインスタンスに追加するには

起動時に run-instances AWS CLI コマンドを `--block-device-mappings` オプションと共に使用し、インスタンスのブロックデバイスマッピングを指定します。

例えば、EBS-backed AMI が、次のブロックデバイスマッピングを指定するとします。

- /dev/sdb=ephemeral0
- /dev/sdh=snap-1234567890abcdef0
- /dev/sdj=:100

この AMI から起動したインスタンスに `/dev/sdj` がアタッチされないようにするには、次のマッピングを使用します。

```
{
    "DeviceName": "/dev/sdj",
    "NoDevice": ""
}
```

`/dev/sdh` のサイズを 300 GiB に増やすには、次のマッピングを指定します。デバイス名を指定することでボリュームを特定できるため、`/dev/sdh` のスナップショット ID を指定する必要はありません。

```
{
    "DeviceName": "/dev/sdh",
    "Ebs": {
      "VolumeSize": 300
    }
}
```

インスタンスの起動時にルートボリュームのサイズを増やすには、最初に AMI の ID を指定して describe-images を呼び出し、ルートボリュームのデバイス名を確認します。たとえば、"RootDeviceName": "/dev/xvda" と指定します。ルートボリュームのサイズを上書きするには、AMI が使用しているルートデバイスのデバイス名と、新しいボリュームサイズを指定します。

```
{
    "DeviceName": "/dev/xvda",
    "Ebs": {
      "VolumeSize": 100
    }
}
```

追加インスタンスストアボリューム /dev/sdc をアタッチするには、次のマッピングを指定します。インスタンスタイプが複数のインスタンスストアボリュームをサポートしていない場合、このマッピングは効果がありません。インスタンスが NVMe インスタンスストアボリュームをサポートしている場合、これらのボリュームは自動的に列挙され、NVMe デバイス名が割り当てられます。

```
{
    "DeviceName": "/dev/sdc",
    "VirtualName": "ephemeral1"
}
```

AWS Tools for Windows PowerShell を使用してボリュームをインスタンスに追加するには

New-EC2Instance コマンド (-BlockDeviceMapping) で AWS Tools for Windows PowerShell パラメータを使用します。

実行中のインスタンスのブロックデバイスマッピングの更新

modify-instance-attribute AWS CLI コマンドを使用して、実行中のインスタンスのブロックデバイスマッピングを更新できます。この属性を変更する前に、インスタンスを停止する必要はありません。

```
aws ec2 modify-instance-attribute --instance-id i-1a2b3c4d --block-device-mappings
 file://mapping.json
```

例えば、インスタンスの終了時にルートボリュームを保持するには、mapping.json で以下を指定します。

```
[
```

```
  {
    "DeviceName": "/dev/sda1",
    "Ebs": {
      "DeleteOnTermination": false
    }
  }
]
```

または、Edit-EC2InstanceAttribute コマンド (`-BlockDeviceMapping`) で AWS Tools for Windows PowerShell パラメータを使用することもできます。

インスタンスブロックデバイスマッピングの EBS ボリュームの表示

インスタンスにマッピングされた EBS ボリュームを簡単に列挙できます。

> **Note**
>
> 2009 年 10 月 31 日 API のリリースよりも前に起動されたインスタンスについては、AWS では、ブロックデバイスマッピングを表示できません。AWS がブロックデバイスマッピングを表示できるようにするには、ボリュームをデタッチしてから再アタッチする必要があります。

コンソールを使用してインスタンスの EBS ボリュームを表示するには

1. Amazon EC2 コンソールを開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 検索ボックスにルートデバイスタイプと入力し、[EBS] を選択します。これにより、EBS-backed インスタンスのリストが表示されます。
4. 目的のインスタンスを選択し、[ストレージ] タブに表示された詳細を確認します。少なくとも、ルートデバイスでは次の情報を使用できます。
   - ルートデバイスタイプ (例: EBS)
   - [ルートデバイス名] (例: `/dev/xvda`)
   - [ブロックデバイス] (例: `/dev/xvda`、`/dev/sdf`、`/dev/sdj`)

インスタンスがブロックデバイスマッピングを使用して追加の EBS ボリュームで起動した場合は、[ブロックデバイス] の下に表示されます。このタブには、インスタンスストアボリュームは表示されません。

5. EBS ボリュームに関する追加情報を表示するには、そのボリューム ID を選択して [ボリューム] ページに移動します。

コマンドラインを使用してインスタンスの EBS ボリュームを表示するには

describe-instances (AWS CLI) コマンドまたは Get-EC2Instance (AWS Tools for Windows PowerShell) コマンドを使用して、インスタンスのブロックデバイスマッピングで EBS ボリュームを列挙します。

インスタンスストアボリュームのインスタンスブロックデバイスマッピングの表示

インスタンスのブロックデバイスマッピングを表示した場合、EBS ボリュームのみが表示され、インスタンスストアボリュームは表示されません。インスタンスのインスタンスストアボリュームを表示する方法は、ボリュームタイプによって異なります。

NVMe インスタンスストアボリューム

ブロックデバイスマッピング内の NVMe インスタンスストアボリュームをクエリするには、NVMe コマンドラインパッケージ (nvme-cli) を使用します。パッケージをダウンロードし、インスタンスにインストールした上で、次のコマンドを実行します。

```
[ec2-user ~]$ sudo nvme list
```

インスタンスに関する出力例を次に示します。Model 列のテキストは、このボリュームが EBS ボリュームであるか、インスタンスストアボリュームであるかを示します。この例では、/dev/nvme1n1 および /dev/nvme2n1 がインスタンスストアボリュームです。

```
Node             SN                   Model
 Namespace
---------------- -------------------- ----------------------------------------
 ---------
/dev/nvme0n1     vol06afc3f8715b7a597 Amazon Elastic Block Store               1

/dev/nvme1n1     AWS2C1436F5159EB6614 Amazon EC2 NVMe Instance Storage         1
```

```
/dev/nvme2n1     AWSB1F4FF0C0A6C281EA Amazon EC2 NVMe Instance Storage         1
  ...
```

HDD もしくは SSD のインスタンスストアボリューム

ブロックデバイスマッピングで HDD もしくは SSD のインスタンスストアボリュームをクエリするには、インスタンスメタデータを使用します。NVMe インスタンスストアボリュームは含まれていません。

インスタンスメタデータのすべてのリクエストの基本 URI は `http://169.254.169.254/latest/` です。詳細については、インスタンスメタデータとユーザーデータを参照してください。

まず、実行中にインスタンスに接続します。インスタンスからこのクエリを使用して、そのブロックデバイスマッピングを取得します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/block-device-mapping/
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/block-device-mapping/
```

レスポンスには、インスタンスのブロックデバイスの名前が含まれます。例えば、instance store Backed `m1.small` インスタンスの出力は次のようになります。

```
ami
ephemeral0
root
swap
```

`ami` デバイスは、インスタンスによって判断されるルートデバイスです。インスタンスストアボリュームの名前は `ephemeral[0-23]` です。`swap` デバイスはページファイル用です。EBS ボリュームもマップした場合、そのボリュームは、`ebs1`、`ebs2` のように表示されます。

ブロックデバイスマッピングの個別のブロックデバイスの詳細を確認するには、ここで示すように、前のクエリにブロックデバイスの名前を追加します。

IMDSv2

```
[ec2-user ~]$ TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-metadata-token-ttl-seconds: 21600"` \
&& curl -H "X-aws-ec2-metadata-token: $TOKEN" http://169.254.169.254/latest/meta-data/block-device-mapping/ephemeral0
```

IMDSv1

```
[ec2-user ~]$ curl http://169.254.169.254/latest/meta-data/block-device-mapping/ephemeral0
```

インスタンスタイプは、インスタンスに利用できるインスタンスストアボリュームの数を決定します。ブロックデバイスマッピングのインスタンスストアボリュームの数が、インスタンスに利用できるインスタンスストアボリュームの数を超える場合は、追加のボリュームは無視されます。インスタンスにインスタンスストアボリュームを表示するには、lsblk コマンド を実行します。各インスタンスタイプがサポートするインスタンスストアボリュームの数は、インスタンスストアボリュームを参照してください。

Torn Write Prevention

Torn Write Prevention は、データの回復力に悪影響を及ぼすことなく、I/O 集約型リレーショナルデータベースのワークロードのパフォーマンスを向上させ、待ち時間を短縮するために AWS によって設計されたブロックストレージ機能です。MySQL や MariaDB など、データベースエンジンとして InnoDB や XtraDB を使用するリレーショナルデータベースは、Torn Write Prevention が有効です。

通常、ストレージデバイスの電源障害発生量を超えるページを使用するリレーショナルデータベースでは、データロギングメカニズムを使用して書き込み操作が中断されるのを防ぎます。MariaDB と MySQL は、データテーブルに書き込む前に二重書き込みバッファファイルを使用してデータを口グに記録します。書き込み処理中にオペレーティングシステムがクラッシュしたり、停電したりして、書き込みが不完全になる、または中断された場合、データベースは二重書き込みバッファからデータを回復できます。二重書き込みバッファへの書き込みに伴い追加で発生する I/O オーバーヘッドは、データベースのパフォーマンスとアプリケーションのレイテンシーに影響を与え、1 秒あたりに処理できるトランザクション数が減少します。二重書き込みバッファの詳細については、「MariaDB」と「MySQL」のドキュメントを参照してください。

Torn Write Prevention 機能では、データがオールオアナッシングの書き込みトランザクションでストレージに書き込まれるため、二重書き込みバッファを使用する必要がなくなります。これにより、オペレーティングシステムがクラッシュしたり、書き込みトランザクション中に停電したりした場合に、データの一部または破損したデータがストレージに書き込まれるのを防ぎます。ワークロードの回復力を損なうことなく、1 秒あたりに処理されるトランザクション数を最大 30% 増やし、書き込みレイテンシーを最大 50% 削減できます。

料金

Torn Write Prevention 機能の使用に、追加料金はかかりません。

サポートされているブロックサイズとブロック境界の配置

Torn Write Prevention は、4 KiB、8 KiB、16 KiB のデータブロックの書き込み操作をサポートします。データブロック開始論理ブロックアドレス (LBA) は、4 KiB、8 KiB、または 16 KiB のそれぞれのブロック境界サイズに合わせる必要があります。例えば、16 KiB の書き込み操作では、データブロック開始 LBA を 16 KiB のブロック境界サイズに合わせる必要があります。

次の表は、ストレージとインスタンスタイプによるサポートを示しています。

	4 KiB ブロック	8 KiB ブロック	16 KiB ブロック
インスタンスストアボリューム	現行世代の I-family インスタンスにアタッチされたすべての NVMe インスタンスストアボリューム。	I4i、Im4gn、および Is4gen インスタンスは AWS Nitro SSD でサポートされています。	
Amazon EBS ボリューム	Nitro ベースのインスタンスにアタッチされているすべての Amazon EBS ボリューム。		

インスタンスとボリュームが Torn Write Prevention をサポートしているかどうかを確認するには、クエリを実行して、インスタンスが Torn Write Prevention をサポートしているかどうか、およびサポートされているブロックサイズや境界サイズなどのその他の詳細を確認します。詳細については、「Torn Write Prevention のサポートと設定を確認する」を参照してください。

要件

Torn Write Prevention が正しく機能するには、NTWPU、NTWGU、NTWBU フィールドで指定されているサイズ、配置、および境界の要件を I/O 操作が満たしている必要があります。デバイスに送信する前に、特定のストレージサブシステム (ファイルシステム、LVM、RAID など) がストレージスタックの I/O プロパティ (ブロックマージ、分割、ブロックアドレスの再配置など) を変更しないように、オペレーティングシステムを設定する必要があります。

Torn Write Prevention は、次の設定でテスト済みです。

- 必要なブロックサイズをサポートするインスタンスタイプとストレージタイプ。
- カーネルバージョン 5.10 以降の Amazon Linux 2。
- `bigalloc` が有効になっていて、クラスターサイズが 16 KiB で、最新の ext4 ユーティリティ (e2fsprogs 1.46.5 以降) を使用している ext4。
- Linux カーネルバッファキャッシュをバイパスする `O_DIRECT` ファイルアクセスモード。

> **Note**
>
> MySQL と MariaDB のワークロードの I/O マージを無効にする必要はありません。

Torn Write Prevention のサポートと設定を確認する

インスタンスとボリュームが Torn Write Prevention をサポートしているかどうかを確認し、Torn Write Prevention に関する情報を含む NVMe 名前空間のベンダー固有のデータを表示するには、次のコマンドを使用します。

```
$ sudo nvme id-ns -v device_name
```

> **Note**
>
> このコマンドは、ベンダー固有の情報を ASCII 解釈の 16 進数で返します。出力の読み取りと解析ができる `ebsnvme-id` と似たツールをアプリケーションに組み込む必要がある場合があります。

例えば、次のコマンドは、/dev/nvme1n1 の Torn Write Prevention に関する情報を含む NVMe 名前空間のベンダー固有のデータを返します。

```
$ sudo nvme id-ns -v /dev/nvme1n1
```

インスタンスとボリュームが Torn Write Prevention をサポートしている場合、NVMe 名前空間のベンダー固有データに次の AWS Torn Write Prevention 情報が返されます。

> **Note**
>
> 次の表のバイトは、NVMe 名前空間のベンダー固有データの先頭からのオフセットをバイト単位で表しています。

バイト	説明
0:31	例えば /dev/xvda 、デバイス接続マウントポイントの名前。ボリュームアタッチメントリクエスト時にこれを指定すると、Amazon EC2 インスタンスが NVMe ブロックデバイス (nvmeXn1) へのシンボリックリンクを作成するために使用できます。
32:63	ボリューム ID。例えば、vol01234567890abcdef と指定します。このフィールドを使用して、NVMe デバイスを接続されたボリュームにマッピングできます。
64:255	将来の利用のために予約されています。
256:257	Torn Write Prevention 名前空間ユニットサイズ (NTWPU)。このフィールドは、停電またはエラー状態時に NVM にアトミックに書き込まれることが保証されている書き込み操作の名前空間固有のサイズを示します。このフィールドは、0 ベース値で表される論理ブロックで指定されます。
258:259	Torn Write Prevention 名前空間粒度サイズ (NTWPG)。このフィールドは、停電またはエラー状態時に NVM にアトミックに書き込まれることが保証されている書き込み操作の NTWPU 以下の名前空間固有のサイズ増分を示します。つまり、サイズは NTWPG * n <= NTWPU で、n は正の整数でなければなりません。書き込み操作の LBA オフセットもこのフ

バイト	説明
	ィールドに合わせる必要があります。このフィールドは、0 ベース値で表される論理ブロックで指定されます。
260:263	Torn Write Prevention 名前空間境界サイズ (NTWPB)。このフィールドは、この名前空間の NTWPU 値のアトミック境界サイズを示します。この名前空間への書き込みがアトミック境界を越える場合、停電やエラー時に NVM への書き込みがアトミックに行われることは保証されません。0h の値が、停電やエラー時にアトミックな境界がないことを示します。他のすべての値は、NTWPU フィールドと同じエンコーディングを使用して論理ブロック単位でサイズを指定します。

Torn Write Prevention 用のソフトウェアスタックを設定する

Torn Write Prevention は、サポートされているボリュームを持つサポートされているインスタンスタイプでは、デフォルトで有効になっています。ボリュームやインスタンスで Torn Write Prevention を有効にするために、追加の設定を有効にする必要はありません。

> **Note**
>
> Torn Write Prevention をサポートしていないワークロードでは、パフォーマンスに影響はありません。これらのワークロードでは何も変更する必要はありません。
> Torn Write Prevention はサポートしているが、使用するように設定されていないワークロードは、引き続き二重書き込みバッファを使用するため、パフォーマンス上のメリットはありません。

MySQL または MariaDB ソフトウェアスタックを設定して二重書き込みバッファを無効にし、Torn Write Prevention を使用するには、次の手順を実行します。

1. BigAlloc オプションで ext4 ファイルシステムを使用するようにボリュームを設定し、クラスターサイズを 4 KiB、8 KiB、または 16 KiB に設定します。クラスターサイズが 4 KiB、8 KiB、または 16 KiB の BigAlloc を使用すると、ファイルシステムがそれぞれの境界に合わせてファイルを割り当てることができます。

   ```
   $  mkfs.ext4 -O bigalloc -C 4096|8192|16384 device_name
   ```

> **Note**
>
> MySQL と MariaDB の場合は、データベースのページサイズに合わせて `-C 16384` を使用する必要があります。割り当ての粒度をページサイズの倍数以外の値に設定すると、ストレージデバイスの Torn Write Prevention の境界と割り当てが一致しなくなる可能性があります。

例:

```
$  mkfs.ext4 -O bigalloc -C 16384 /dev/nvme1n1
```

2. `O_DIRECT` フラッシュ方式を使用するように InnoDB を設定し、InnoDB の二重書き込みをオフにします。任意のテキストエディタを使用して `/etc/my.cnf` を開き、以下のように `innodb_flush_method` および `innodb_doublewrite` パラメータを更新します。

```
innodb_flush_method=O_DIRECT
innodb_doublewrite=0
```

> **Important**
>
> Logical Volume Manager (LVM) またはその他のストレージ仮想化レイヤーを使用している場合は、ボリュームの開始オフセットが 16 KiB の倍数になるように調整してください。これは、ストレージ仮想化レイヤーで使用されるメタデータヘッダーとスーパーブロックを考慮して、基盤となる NVMe ストレージを基準としています。LVM 物理ボリュームにオフセットを追加すると、ファイルシステムの割り当てと NVMe デバイスのオフセットがずれて、Torn Write Prevention が無効になる可能性があります。詳細については、「Linux 手動ページ」の「`--dataalignmentoffset`」を参照してください。

リソースとタグ

Amazon EC2 にはさまざまなリソースが用意されており、それらを作成して利用することができます。これらのリソースには、イメージ、インスタンス、ボリューム、スナップショットなどがあります。リソースを作成すると、リソースに一意のリソース ID が割り当てられます。

一部のリソースには、それらの整理と識別に役立つように、ユーザーが定義できる値で値にタグを付けることができます。

以下のトピックでは、リソースとタグ、およびそれらの使用方法について説明します。

コンテンツ

ごみ箱

ごみ箱は、誤って削除された Amazon EBS スナップショットと EBS-backed AMI を復元することを可能にするデータ復旧機能です。ごみ箱を使用する場合、リソースが削除されると、リソースは、完全に削除されるまでの時間として指定した期間、ごみ箱に保持されます。

リソースは、保持期間が終了する前であればいつでもごみ箱から復元できます。ごみ箱からリソースを復元すると、そのリソースはごみ箱から削除され、アカウント内の他のそのタイプのリソースと同じ方法で使用できます。保持期間が終了し、リソースが復元されない場合、リソースはごみ箱から完全に削除され、復旧できなくなります。

ごみ箱は、ビジネスクリティカルなデータを誤って削除しないように保護することで、ビジネス継続性を確保するのに役立ちます。

トピック

仕組み

ごみ箱を有効にして使用するには、リソースを保護する AWS リージョンに保持ルールを作成する必要があります。保持ルールでは、以下を指定します。

- 保護するリソースタイプ。
- 削除時にごみ箱に保持するリソース。
- リソースが完全に削除される前に、リソースをごみ箱に保持する保持期間。

ごみ箱では、2 種類の保持ルールを作成できます。

- タグレベルの保持ルール — タグレベルの保持ルールは、リソースタグを使用して、ごみ箱に保持されるリソースを識別します。保持ルールごとに、1 つ以上のタグのキーと値のペアを指定します。保持ルールで指定されたタグのキーと値のペアの少なくとも 1 つでタグ付けされた指定タイプのリソースは、削除時に自動的にごみ箱に保持されます。タグに基づいてアカウント内の特定のリソースを保護する場合は、このタイプの保持ルールを使用します。
- リージョンレベルの保持ルール — リージョンレベルの保持ルールでは、リソースタグは指定されません。リソースにタグが付いていなくても、ルールが作成されるリージョンにある指定タイプのすべてのリソースに適用されます。特定のリージョン内のすべての指定タイプのリソースを保護する場合は、このタイプの保持ルールを使用します。

リソースがごみ箱に入っている間は、いつでもそのリソースを復元して使用できます。

リソースは、次のいずれかの結果になるまで、ごみ箱に残ります。

- 使用するために手動で復元した場合。ごみ箱からリソースを復元すると、そのリソースはごみ箱から削除され、直ちに使用できるようになります。復旧されたリソースは、アカウント内のそのタイプの他のリソースと同じ方法で使用できます。
- 保持期間が終了した場合。保存期間が終了し、リソースがごみ箱から復元されていない場合、リソースはごみ箱から完全に削除され、表示や復元はできなくなります。

サポート リソース

ごみ箱は、次のリソースタイプをサポートしています。

- Amazon EBS スナップショット

> **Important**
>
> ごみ箱の保存ルールは、アーカイブストレージ階層のアーカイブされたスナップショットにも適用されます。ごみ箱の保持ルールに一致するアーカイブスナップショットを削除すると、アーカイブされたスナップショットは、保持ルールで定義されている保持期間中、ごみ箱に保持されます。アーカイブされたスナップショットは、ごみ箱に入っている間、アーカイブされたスナップショットの料金で請求されます。

- Amazon EBS-backed Amazon マシンイメージ (AMI)

> **Note**
>
> 保持ルールは無効になっている AMI にも適用されます。

考慮事項

ごみ箱および保持ルールを使用する場合は、次の考慮事項が適用されます。

一般的な考慮事項

- > **Important**
  >
  > 最初の保持ルールを作成すると、ルールがアクティブになり、リソースの保持が開始されるまでに 30 分ほどかかる場合があります。最初の保持ルールを作成すると、後続の保持ルールがアクティブになり、リソースの保持がほぼ即時に開始されます。

- 削除時にリソースが複数の保持ルールと一致する場合は、保持期間が最も長い保持ルールが優先されます。
- リソースをごみ箱から手動で削除することはできません。リソースは、保持期間が終了すると自動的に削除されます。
- リソースがごみ箱に入っている間は、そのリソースを表示したり、復元したり、タグを変更することしかできません。リソースを使用するには、まずリソースを復元する必要があります。
- AWS のサービス Backup や Amazon Data Lifecycle Manager などの、任意の AWS が保持ルールと一致するリソースを削除した場合、そのリソースは自動的にごみ箱に保持されます。
- リソースをごみ箱に送信すると、次のシステム生成タグがリソースに割り当てられます。
  - タグキー — `aws:recycle-bin:resource-in-bin`
  - タグ値 — `true`

  このタグを手動で編集または削除することはできません。リソースをごみ箱から復元すると、タグは自動的に削除されます。

スナップショットに関する考慮事項

- > **Important**
  >
  > AMI および関連するスナップショットの保持ルールがある場合は、スナップショットの保持期間を AMI の保持期間と同等以上にしてください。これにより、AMI 自体を削除する前に AMI に関連付けられたスナップショットがごみ箱で削除されないようになります。これは、このようなスナップショットを削除すると、AMI が復旧不能になるためです。

- スナップショットが削除されたときに高速スナップショット復元が有効になっている場合、スナップショットがごみ箱に送信された直後に、高速スナップショット復元は自動的に無効になります。
  - スナップショットの高速スナップショット復元が無効になる前にスナップショットを復元しても、そのスナップショットは有効なままになります。

  - スナップショットを復元すると、高速スナップショット復元が無効になった後も、無効のままになります。必要に応じて、高速スナップショット復元を手動で再度有効にする必要があります。
- 削除時に共有されていたスナップショットは、ごみ箱に移動されると自動的に共有が解除されます。スナップショットを復元すると、以前の共有権限がすべて自動的に復元されます。
- AWS Backupなど、別のAWSのサービスによって作成されたスナップショットがごみ箱に移動され、後でそのスナップショットをごみ箱から復旧する場合、それを作成したAWSサービスによって管理されなくなります。スナップショットが不要になった場合は、手動で削除する必要があります。

AMI に関する考慮事項

- Amazon EBS-backed AMI のみがサポートされます。
-
> **Important**
>
> AMI および関連するスナップショットの保持ルールがある場合は、スナップショットの保持期間を AMI の保持期間と同等以上にしてください。これにより、AMI 自体を削除する前に AMI に関連付けられたスナップショットがごみ箱で削除されないようになります。これは、このようなスナップショットを削除すると、AMI が復旧不能になるためです。

- 削除時に共有されていた AMI は、ごみ箱に移動されると自動的に共有が解除されます。AMI を復元すると、以前の共有許可がすべて自動的に復元されます。
- ごみ箱から AMI を復元する前に、まずごみ箱から関連するすべてのスナップショットを復旧し、それらが `available` 状態になっているようにする必要があります。
- AMI に関連付けられているスナップショットをごみ箱から削除すると、AMI は復旧できなくなります。AMI は、保持期間が経過すると削除されます。
- AWS Backup などの別の AWS のサービスによって作成された AMI がごみ箱に送信され、後でその AMI をごみ箱から復旧すると、それを作成した AWS のサービスによって管理されなくなります。AMI が不要になった場合は、手動で削除する必要があります。

Amazon Data Lifecycle Manager スナップショットポリシーに関する考慮事項

- Amazon Data Lifecycle Manager が保持ルールと一致するスナップショットを削除すると、そのスナップショットは自動的にごみ箱に保持されます。
- Amazon Data Lifecycle Manager がポリシーの保持しきい値に達したときにスナップショットを削除してごみ箱に移動し、そのスナップショットをごみ箱から手動で復元した場合は、スナップ

ショットが不要になったら手動で削除する必要があります。Amazon Data Lifecycle Manager は、スナップショットを管理しなくなります。

- ポリシーによって作成されたスナップショットを手動で削除し、ポリシーの保持しきい値に達したときにそのスナップショットがごみ箱にある場合、Amazon Data Lifecycle Manager はスナップショットを削除しません。Amazon Data Lifecycle Manager は、スナップショットがごみ箱に保存されている間は、スナップショットを管理しません。

  ポリシーの保持しきい値に達する前にスナップショットがごみ箱から復元された場合、Amazon Data Lifecycle Manager は、ポリシーの保持しきい値に達したときにスナップショットを削除します。

  ポリシーの保持しきい値に達した後にスナップショットがごみ箱から復元された場合、Amazon Data Lifecycle Manager はそのスナップショットを削除しません。スナップショットが不要になった場合は、手動で削除する必要があります。

AWS Backup の考慮事項

- AWS Backup が保持ルールと一致するスナップショットを削除すると、そのスナップショットは自動的にごみ箱に保持されます。

アーカイブされたスナップショットに関する考慮事項

- ごみ箱の保存ルールは、アーカイブストレージ階層のアーカイブされたスナップショットにも適用されます。ごみ箱の保持ルールに一致するアーカイブスナップショットを削除すると、アーカイブされたスナップショットは、保持ルールで定義されている保持期間中、ごみ箱に保持されます。

  アーカイブされたスナップショットは、ごみ箱に入っている間、アーカイブされたスナップショットの料金で請求されます。

  保持ルールによってアーカイブされたスナップショットがごみ箱から最低期間である 90 日前に削除された場合、残りの日分の料金が請求されます。詳細については、「Amazon EBS ユーザーガイド」の「Archived snapshot pricing and billing」を参照してください。

  ごみ箱にアーカイブされたスナップショットを使用するには、まずそのスナップショットをごみ箱から復元し、次にアーカイブ階層から標準階層に復元する必要があります。

クォータ

ごみ箱には、以下のクォータが適用されます。

クォータ	デフォルトのクォータ			
リージョンあたりの保持ルール数	250			
保持ルールごとにキーと値のペアにタグ付けする	50			

関連サービス

ごみ箱は以下のサービスと連携します。

- [AWS CloudTrail] — ごみ箱で発生したイベントを記録できます。詳細については、AWS CloudTrail を使用してごみ箱をモニタリングするを参照してください。

料金

ごみ箱内のリソースは、標準料金で請求されます。ごみ箱および保持ルールの使用には、追加料金はかかりません。詳細については、Amazon EBS の料金表を参照してください。

> **Note**
>
> 一部のリソースは、保持期間が終了して完全に削除された後も、ごみ箱コンソールや AWS CLI および API 出力に短期間表示される場合があります。これらのリソースの料金は請求されません。請求は、保持期間が終了するとすぐに停止します。

以下の AWS が生成コスト配分タグは、AWS Billing and Cost Management を使用する際のコストの追跡と配分の目的で使用できます。

- キー: `aws:recycle-bin:resource-in-bin`

- 値: `true`

詳細については、「AWS Billing and Cost Management ユーザーガイド」の「AWS 生成コスト配分タグ」を参照してください。

必要な IAM 許可

デフォルトでは、ユーザーには、ごみ箱、保持ルール、またはごみ箱にあるリソースを操作する許可はありません。ユーザーがこれらのリソースを利用するには、特定のリソースと API アクションを使用する許可を付与する IAM ポリシーを作成する必要があります。ポリシーを作成したら、ユーザー、グループ、ロールに許可を追加します。

トピック

- ごみ箱および保持ルールを操作するための許可
- ごみ箱内のリソースを操作するための許可
- [Condition keys for Recycle Bin] (ごみ箱の条件キー)

ごみ箱および保持ルールを操作するための許可

ごみ箱と保持ルールを使用するには、次の許可をユーザーに付与する必要があります。

- `rbin:CreateRule`
- `rbin:UpdateRule`
- `rbin:GetRule`
- `rbin:ListRules`
- `rbin:DeleteRule`
- `rbin:TagResource`
- `rbin:UntagResource`
- `rbin:ListTagsForResource`
- `rbin:LockRule`
- `rbin:UnlockRule`

ごみ箱コンソールを使用するには、ユーザーに `tag:GetResources` 許可が必要です。

以下は、コンソールユーザーの tag:GetResources 許可を含む IAM ポリシーの例です。一部の許可が不要な場合は、ポリシーから削除できます。

```
{
    "Version": "2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "rbin:CreateRule",
            "rbin:UpdateRule",
            "rbin:GetRule",
            "rbin:ListRules",
            "rbin:DeleteRule",
            "rbin:TagResource",
            "rbin:UntagResource",
            "rbin:ListTagsForResource",
            "rbin:LockRule",
            "rbin:UnlockRule",
            "tag:GetResources"
        ],
        "Resource": "*"
    }]
}
```

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「アクセス許可一式を作成」の手順を実行します。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

- IAM ユーザー:
  - ユーザーが継承できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

ごみ箱内のリソースを操作するための許可

ごみ箱内のリソースを操作するために必要な IAM 許可の詳細については、次を参照してください。

- ごみ箱のスナップショットを操作するための権限
- ごみ箱内の AMI を操作するための許可

[Condition keys for Recycle Bin] (ごみ箱の条件キー)

ごみ箱は、IAM ポリシーのCondition要素に使用できる次の条件キーを定義し、ポリシーステートメントが適用される条件を制御します。詳細については、[IAM User Guide] (IAM ユーザーガイド) の [IAM JSON policy elements: Condition] (IAM JSON ポリシー要素：条件) を参照してください。

トピック

- rbin:Request/ResourceType 条件キー
- rbin:Attribute/ResourceType 条件キー

rbin:Request/ResourceType 条件キー

このrbin:Request/ResourceType条件キーを使用して、ResourceTypeリクエストパラメータで指定された値に基づいて[CreateRule]と[ListRules]リクエストのアクセスをフィルタリングするために使用することができます。

例 1 - CreateRule

次のサンプルの IAM ポリシーは、ResourceTypeリクエストパラメーターに指定された値がEBS_SNAPSHOTまたはEC2_IMAGEである場合のみ IAM プリンシパルに [CreateRule] リクエストを行うことを許可します。これにより、プリンシパルはスナップショットと AMI に対してのみ新しい保存ルールを作成できます。

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:CreateRule"
            ],
            "Resource" : "*",
```

```
            "Condition" : {
                "StringEquals" : {
                    "rbin:Request/ResourceType" : ["EBS_SNAPSHOT", "EC2_IMAGE"]
                }
            }
        }
    ]
}
```

例 2 - ListRules

次のサンプル IAM ポリシーは、ResourceTypeリクエストパラメーターに指定した値がEBS_SNAPSHOTの場合にのみ、IAM プリンシパルが ListRules に要求を行うことを許可します。これにより、プリンシパルはスナップショットの保存ルールのみを一覧表示でき、他のリソースタイプの保存ルールを一覧表示できなくなります。

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:ListRules"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Request/ResourceType" : "EBS_SNAPSHOT"
                }
            }
        }
    ]
}
```

rbin:Attribute/ResourceType 条件キー

rbin:Attribute/ResourceType 条件キーを使用し、保存ルールの ResourceType 属性の値に基づいた DeleteRule、GetRule、UpdateRule、LockRule、UnlockRule、TagResource、UntagResource、ListTagsForResource リクエストへのアクセスをフィルタリングできます。

例 1 - UpdateRule

次のサンプル IAM ポリシーは、ResourceType要求されたリテンションルールの属性がEBS_SNAPSHOTまたはEC2_IMAGEの場合にのみ、IAM プリンシパルが UpdateRule に要求を行うことを許可します。これにより、プリンシパルはスナップショットと AMI の保持ルールのみを更新できます。

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:UpdateRule"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Attribute/ResourceType" : ["EBS_SNAPSHOT", "EC2_IMAGE"]
                }
            }
        }
    ]
}
```

例 2 - DeleteRule

次のサンプル IAM ポリシーは、ResourceType要求されたリテンションルールの属性がEBS_SNAPSHOTの場合にのみ、IAM プリンシパルが DeleteRule に要求を行うことを許可します。これにより、プリンシパルはスナップショットの保存ルールのみを削除できます。

```
{
    "Version" : "2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:DeleteRule"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Attribute/ResourceType" : "EBS_SNAPSHOT"
                }
```

```
            }
        }
    ]
}
```

保持ルールの操作

ごみ箱を有効にして使用するには、リソースを保護する AWS リージョンに保持ルールを作成する必要があります。保持ルールでは、以下を指定します。

- 保護するリソースタイプ。
- 削除時にごみ箱に保持するリソース。
- リソースが完全に削除される前に、リソースをごみ箱に保持する保持期間。

ごみ箱では、2 種類の保持ルールを作成できます。

- タグレベルの保持ルール — タグレベルの保持ルールは、リソースタグを使用して、ごみ箱に保持されるリソースを識別します。保持ルールごとに、1 つ以上のタグのキーと値のペアを指定します。保持ルールで指定されたタグのキーと値のペアの少なくとも 1 つでタグ付けされた指定タイプのリソースは、削除時に自動的にごみ箱に保持されます。タグに基づいてアカウント内の特定のリソースを保護する場合は、このタイプの保持ルールを使用します。
- リージョンレベルの保持ルール — リージョンレベルの保持ルールでは、リソースタグは指定されません。リソースにタグが付いていなくても、ルールが作成されるリージョンにある指定タイプのすべてのリソースに適用されます。特定のリージョン内のすべての指定タイプのリソースを保護する場合は、このタイプの保持ルールを使用します。

保持ルールを作成すると、条件に合致したリソースが削除された後に、指定された保持期間自動的にごみ箱に保持されます。

トピック

- 保持ルールを作成する
- ごみ箱の保持ルールの表示
- 保持ルールの更新
- 保持ルールのロック
- 保持ルールのロック解除

- タグ保持ルール
- 保持ルールのタグを表示する
- 保持ルールからタグを削除する
- ごみ箱の保持ルールの削除

保持ルールを作成する

保持ルールを作成するときは、次の必須パラメータを指定する必要があります。

- 保持ルールで保護されるリソースタイプ。
- 保持ルールで保護されるリソースタイプ。保持ルールは、タグレベルとリージョンレベルで作成できます。
  - タグレベルの保持ルールを作成するには、保護するリソースを特定するリソースタグを指定します。ルールごとに最大 50 つのタグを指定でき、同じタグのキーと値のペアは最大 5 つの保持ルールに追加できます。
  - リージョンレベルの保持ルールを作成するには、タグキーと値のペアを指定しないでください。この場合、指定されたタイプのすべてのリソースが保護されます。
- リソースが削除後にごみ箱に保持される期間。期間は、最大 1 年 (365 日) です。

オプションで次のパラメータを指定することもできます。

- 保持ルールの名前 (オプション)。名前の長さは最大 255 文字です。
- 保持ルールのオプション説明 説明は最大 255 文字とすることができます。

> **Note**
>
> 保持ルールの説明には、個人を特定する情報、機密情報、または機密情報を含めないことをお勧めします。

- 保持ルールの識別と整理に役立つ保持ルールタグ (オプション)。ルールごとに最大 50 個のタグを割り当てることができます。

オプションで、作成時に保持ルールをロックすることもできます。作成時に保持ルールをロックする場合は、ロック解除の遅延期間 (7 ~ 30 日) も指定する必要があります。保持ルールは、意図的にロックしない限り、デフォルトでロック解除されたままになります。

保持ルールは、作成されたリージョンでのみ機能します。他のリージョンでごみ箱を使用する場合は、そのリージョンに追加の保持ルールを作成する必要があります。

ごみ箱保持ルールは、次のいずれかの方法で作成できます。

Recycle Bin console

保持ルールを作成するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール)、[Create retention rule] (保持ルールの作成) の順に選択します。
3. [Rule details] (ルール詳細) セクションで、次の操作を行います。
   a. (オプション) [Retention rule name] (保持ルール名) に、保持ルールのわかりやすい名前を入力します。
   b. (オプション) [Retention rule descripion] (保持ルールの説明) に、保持ルールの簡単な説明を入力します。
4. [Rule settings] (ルールの設定) セクションで、以下の操作を行います。
   a. [Resource type] (リソースの種類) で、保護する保持ルールのリソースの種類を選択します。保持ルールは、このタイプのリソースのみをごみ箱に保持します。
   b. 次のいずれかを行います。
      - リージョン内の削除されたすべての指定タイプのリソースに対応するリージョンレベルの保持ルールを作成するには、[Apply to all resources] (すべてのリソースに適用する) を選択します。この保持ルールは、リソースにタグがない場合でも、削除時に指定リソースをすべてごみ箱に保持します。
      - タグレベルの保持ルールを作成するには、[Resource tags to match] (照合するリソースタグ) に、ごみ箱に保持される指定タイプのリソースの識別に使用するタグのキーと値のペアを入力します。保持ルールでは、指定されたタグのキーと値のペアが少なくとも 1 つ含まれている指定タイプのリソースのみが保持されます。
   c. [Retention period] (保持期間) に、保持ルールによってリソースをごみ箱に保持する日数を入力します。
5. (オプション) 保持ルールをロックするには、[Rule lock settings] (ルールのロックの設定) で [Lock] (ロック) を選択し、[Unlock delay period] (ロック解除の遅延期間) でロック解除の遅延期間を日単位で指定します。保持ルールを変更または削除することはできません。ルールを

変更または削除するには、まずルールをロック解除してから、ロック解除の遅延期間が終了するまで待つ必要があります。詳細については、「保持ルールのロック」を参照してください。

保持ルールをロック解除したままにするには、[Rule lock settings] (ルールのロックの設定) で [Unlock] (ロック解除) を選択したままにします。ロック解除された保持ルールは、いつでも変更または削除できます。詳細については、「保持ルールのロック解除」を参照してください。

6. (オプション) [Tags] (タグ) セクションで、以下の操作を行います。
   - ルールにカスタムタグをタグ付けするには、[Add tag] (タグの追加) を選択し、タグキーと値のペアを入力します。
7. [Create retention rule] (保持ルールの作成) を選択します。

AWS CLI

保持ルールを作成するには

create-rule AWS CLI コマンドを使用します。[--retention-period] に、ごみ箱に削除されたスナップショットを保持する日数を指定します。[--resource-type] で、スナップショットに [EBS_SNAPSHOT]、または AMI に [EC2_IMAGE] を指定します。タグレベルの保持ルールを作成するには、[--resource-tags] で、保持するスナップショットの識別に使用するタグを指定します。リージョンレベルの保持ルールを作成するには、--resource-tags を省略します。保持ルールをロックするには、--lock-configuration を含めて、ロック解除の遅延期間を日単位指定します。

```
$ aws rbin create-rule \
--retention-period RetentionPeriodValue=number_of_days,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT|EC2_IMAGE \
--description "rule_description" \
--lock-configuration
 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=unlock_delay_in_days}' \
--resource-tags ResourceTagKey=tag_key,ResourceTagValue=tag_value
```

例 1

次のコマンド例では、すべてのスナップショットを 7 日間保持するリージョンレベルのロック削除された保持ルールを作成します。

```
$ aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Match all snapshots"
```

例 2

次のコマンド例では、purpose=production でタグ付けされた削除済みのスナップショットを 7 日間保持するタグレベルのルールを作成します。

```
$ aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Match snapshots with a specific tag" \
--resource-tags ResourceTagKey=purpose,ResourceTagValue=production
```

例 3

次のコマンド例では、すべてのスナップショットを 7 日間保持するリージョンレベルのロックされた保持ルールを作成します。保持ルールは 7 日間のロック解除の遅延期間でロックされます。

```
$ aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Match all snapshots" \
--lock-configuration 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=7}'
```

ごみ箱の保持ルールの表示

ごみ箱の保持ルールは、次のいずれかの方法で表示できます。

Recycle Bin console

保持ルールを表示するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール) を選択します。
3. グリッドには、選択したリージョンのすべての保持ルールがリストされます。特定の保持ルールに関する詳細情報を表示するには、グリッドでその保持ルールを選択します。

AWS CLI

すべての保持ルールを表示するには

list-rules AWS CLI コマンドを使用し、--resource-type で、スナップショットには EBS_SNAPSHOT、または AMI には EC2_IMAGE を指定します。

```
$ aws rbin list-rules --resource-type EBS_SNAPSHOT|EC2_IMAGE
```

例

次のサンプルコマンドは、スナップショットを保持するすべての保持ルールを一覧表示します。

```
$ aws rbin list-rules --resource-type EBS_SNAPSHOT
```

特定の保持ルールの情報を表示するには

get-rule AWS CLI コマンドを使用します。

```
$ aws rbin get-rule --identifier rule_ID
```

例

次のコマンド例は、保持ルール pwxIkFcvge4 に関する情報を表示します。

```
$ aws rbin get-rule --identifier pwxIkFcvge4
```

保持ルールの更新

ロック解除された保持ルールの説明、リソースタグ、保持期間は、作成後にいつでも更新できます。保持ルールのリソースタイプやロック解除の遅延期間を、保持ルールがロック解除されていても、更新することはできません。

ロックされた保持ルールは、どのような方法でも更新できません。ロックされた保持ルールを変更する必要がある場合は、まずロックを解除し、ロック解除の遅延期間が終了するまで待つ必要があります。

ロックされた保持ルールのロック解除の遅延期間を変更する必要がある場合は、保持ルールをロック解除し、現在のロック解除の遅延期間が終了するまで待つ必要があります。ロック解除の遅延期間が終了したら、保持ルールを再ロックし、新しいロック解除の遅延期間を指定する必要があります。

> **Note**
>
> 保持ルールの説明には、個人を特定する情報、機密情報、または機密情報を含めないことをお勧めします。

保持ルールを更新すると、その変更は保持される新しいリソースにのみ適用されます。この変更は、ごみ箱に移動済みのリソースには影響しません。例えば、保持ルールの保持期間を更新すると、更新後に削除されたスナップショットのみが新しい保持期間、保持されます。更新前にごみ箱に送られたスナップショットは、以前の (古い) 保持期間にわたって保持されます。

保持ルールの更新は、次のいずれかの方法で行うことができます。

Recycle Bin console

保持ルールを更新するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール) を選択します。
3. グリッドで、更新する保持ルールを選択し、[Actions] (アクション)、[Edit retention rule] (保持ルールの編集) の順にクリックします。
4. [Rule details] (ルールの詳細) セクションで、[Retention rule name] (保持ルール名) そして [Retention rule description] (保持ルールの説明) を必要に応じて更新します。
5. [Rule settings] (ルール設定) セクションで、[Resource type] (リソースタイプ)、[Resource tags to match] (照合するリソースタグ)、[Retention period] (保持期間) を必要に応じて更新します。
6. [Tags] (タグ) セクションで、必要に応じて保持ルールタグを追加または削除します。
7. [Save retention rule] (保持ルールの保存) を選択します。

AWS CLI

保持ルールを更新するには

update-rule AWS CLI コマンドを使用します。[`--identifier`] で、更新する保持ルールの ID を指定します。[`--resource-types`] で、スナップショットに [`EBS_SNAPSHOT`]、または AMI に [`EC2_IMAGE`] を指定します。

```
$ aws rbin update-rule \
```

```
--identifier rule_ID \
--retention-period RetentionPeriodValue=number_of_days,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT|EC2_IMAGE \
--description "rule_description"
```

例

次の例では、保持ルール 6lsJ2Fa9nh9 を更新して、すべてのスナップショットを 7 日間保持するようにし、その説明を更新しています。

```
$ aws rbin update-rule \
--identifier 6lsJ2Fa9nh9 \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Retain for three weeks"
```

保持ルールのロック

ごみ箱を使用すると、リージョンレベルの保持ルールをいつでもロックできます。

> **Note**
>
> タグレベルの保持ルールはロックできません。

ロックされた保持ルールは、必要な IAM 許可を持つユーザーであっても変更または削除できません。保持ルールをロックすることで、偶発的な、または悪意のある変更や削除から保護できます。

保持ルールをロックするには、ロック解除の遅延期間を指定する必要があります。これは、保持ルールをロック解除してから変更または削除できるようになるまで待つ必要がある期間です。ロック解除の遅延期間中は、保持ルールを変更または削除することはできません。保持ルールの変更または削除は、ロック解除の遅延期間の終了後にのみ行えます。

保持ルールのロック後は、ロック解除の遅延期間を変更できません。アカウントの権限が侵害された場合、ロック解除の遅延期間を設けることで、セキュリティ上の脅威を検出して対応するための追加の時間を確保できます。この期間は、セキュリティ違反を特定して対応するのにかかる時間よりも長くする必要があります。過去のセキュリティインシデントと、アカウント侵害の特定と是正に必要な時間を確認することで、適切な期間を設定することができます。

保持ルールのロック状態が変更された場合に通知されるように、Amazon EventBridge ルールを使用することをお勧めします。詳細については、「Amazon EventBridge を使用してごみ箱をモニタリングする」を参照してください。

考慮事項

- ロックできるのはリージョンレベルの保持ルールだけです。
- 保持ルールのロックはいつでも解除できます。
- ロック解除の遅延期間は 7〜30 日でなければなりません。
- 保持ルールはロック解除の遅延期間中に再ロックできます。保持ルールを再ロックすると、ロック解除の遅延期間がリセットされます。

リージョンレベルの保持ルールは、次のいずれかの方法でロックできます。

Recycle Bin console

保持ルールをロックするには

1. ごみ箱コンソールを https://console.aws.amazon.com/rbin/home/ で開きます。
2. ナビゲーションパネルで、[Retention rules] (保持ルール) を選択します。
3. グリッドでロックする保持ルールを選択し、[Actions] (アクション)、[Edit retention rule lock] (保持ルールロックの編集) の順に選択します。
4. [Edit retention rule lock] (保持ルールロックの編集) 画面で [Lock] (ロック) を選択し、[Unlock delay period] (ロック解除の遅延期間) でロック解除の遅延期間を日単位で指定します。
5. [I acknowledge that locking the retention rule will prevent it from being modified or deleted] (保持ルールをロックすると変更や削除ができなくなることを確認) チェックボックスをオンにし、[Save] (保存) を選択します。

AWS CLI

ロック解除された保持ルールをロックするには

ロックルール AWS CLI コマンドを使用します。`--identifier` については、ロックする保持ルールの ID を指定します。`--lock-configuration` については、ロック解除の遅延期間を日単位で指定します。

```
$ aws rbin lock-rule \
```

```
--identifier rule_ID \
--lock-configuration
 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=number_of_days}'
```

例

次のコマンド例では、6lsJ2Fa9nh9 保持ルールをロックし、ロック解除の遅延期間を 15 日間に設定します。

```
$ aws rbin lock-rule \
--identifier 6lsJ2Fa9nh9 \
--lock-configuration 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=15}'
```

保持ルールのロック解除

ロックされた保持ルールの変更または削除はできません。ロックされた保持ルールを変更する必要がある場合は、まずロックを解除する必要があります。保持ルールをロック解除したら、ロック解除の遅延期間が終了するのを待ってから、変更または削除する必要があります。ロック解除の遅延期間中は、保持ルールを変更または削除することはできません。

ロック解除された保持ルールは、必要な IAM 許可を持つユーザーがいつでも変更および削除できます。保持ルールをロック解除したままにすると、偶発的または悪意のある変更や削除にさらされる可能性があります。

考慮事項

- 保持ルールはロック解除の遅延期間中に再ロックできます。
- ロック解除の遅延期間が過ぎた後で保持ルールを再ロックできます。
- ロック解除の遅延期間をバイパスすることはできません。
- 最初のロック後に、ロック解除の遅延時間を変更することはできません。

保持ルールのロック状態が変更された場合に通知されるように、Amazon EventBridge ルールを使用することをお勧めします。詳細については、「Amazon EventBridge を使用してごみ箱をモニタリングする」を参照してください。

リージョンレベルのロックされた保持ルールは、次のいずれかの方法でロック解除できます。

Recycle Bin console

保持ルールをロック解除するには

1. ごみ箱コンソールを https://console.aws.amazon.com/rbin/home/ で開きます。
2. ナビゲーションパネルで、[Retention rules] (保持ルール) を選択します。
3. グリッドでロック解除する保持ルールを選択し、[Actions] (アクション)、[Edit retention rule lock] (保持ルールロックの編集) の順に選択します。
4. [Edit retention rule lock] (保持ルールロックの編集) 画面で、[Unlock] (ロック解除) を選択し、[Save] (保存) を選択します。

AWS CLI

ロックされた保持ルールをロック解除するには

unlock-rule AWS CLI コマンドを使用します。`--identifier` で、ロック解除する保持ルールの ID を指定します。

```
$ aws rbin unlock-rule \
--identifier rule_ID
```

例

次のコマンド例では、保持ルール `6lsJ2Fa9nh9` をロック解除します。

```
$ aws rbin unlock-rule \
--identifier 6lsJ2Fa9nh9
```

タグ保持ルール

保持ルールにカスタムタグを割り当てて、目的、所有者、環境など、さまざまな方法で分類できます。これにより、割り当てたカスタムタグに基づいて特定の保持ルールを効率的に見つけることができます。

保持ルールにタグを割り当てるには、次のいずれかの方法を使用します。

Recycle Bin console

保持ルールにタグ付けするには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール) を選択します。
3. タグ付けする保持ルールを選択し、[Tags] (タグ) タブで、[Manage tags] (タグの管理) を選択します。
4. タグを追加 を選択します。[Key] (キー) に、タグキーを入力します。[Value] (値) に、タグの値を入力します。
5. [Save] (保存) を選択します。

AWS CLI

保持ルールにタグ付けするには

tag-resource AWS CLI コマンドを使用します。--resource-arn で、タグ付けする保持ルールの Amazon リソースネーム (ARN) を指定し、--tags で、タグのキーと値のペアを指定します。

```
$ aws rbin tag-resource \
--resource-arn retention_rule_arn \
--tags key=tag_key,value=tag_value
```

例

次のコマンド例では、保持ルール arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3 に purpose=production をタグ付けします。

```
$ aws rbin tag-resource \
--resource-arn arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3 \
--tags key=purpose,value=production
```

保持ルールのタグを表示する

保持ルールに割り当てられたタグは、次のいずれかの方法で表示できます。

Recycle Bin console

保持ルールのタグを表示するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール) を選択します。
3. タグを表示する保持ルールを選択し、[Tags] (タグ) タブを選択します。

AWS CLI

保持ルールに割り当てられたタグを表示するには

list-tags-for-resource AWS CLI コマンドを使用します。`--resource-arn` で、保持ルールの ARN を指定します。

```
$ aws rbin list-tags-for-resource \
--resource-arn retention_rule_arn
```

例

次のコマンド例では、保持ルール `arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3` のタグを一覧表示します。

```
$ aws rbin list-tags-for-resource \
--resource-arn arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3
```

保持ルールからタグを削除する

イベント通知のタグは、次のいずれかの方法で削除することができます。

Recycle Bin console

保持ルールからタグを削除するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール) を選択します。
3. タグを削除する保持ルールを選択し、[Tags] (タグ) タブで、[Manage tags] (タグの管理) を選択します。
4. 削除するタグの横にある [Remove] (削除) を選択します。

5. [Save] (保存) を選択します。

AWS CLI

保持ルールからタグを削除するには

untag-resource AWS CLI コマンドを使用します。--resource-arn で、保持ルールの ARN を指定します。--tagkeys で、削除するタグのタグキーを指定します。

```
$ aws rbin untag-resource \
--resource-arn retention_rule_arn \
--tagkeys tag_key
```

例

次のコマンド例では、キーが purpose のタグを保持ルール arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3 から削除します。

```
$ aws rbin untag-resource \
--resource-arn arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3 \
--tagkeys purpose
```

ごみ箱の保持ルールの削除

保持ルールはいつでも削除できます。保持ルールを削除すると、削除後にゴミ箱で新しいリソースが保持されなくなります。保持ルールが削除される前にごみ箱に移動されたリソースは、保持ルールで定義されている保持期間に従って、引き続きごみ箱に保持されます。期間が終了すると、リソースはごみ箱から完全に削除されます。

次のいずれかの方法を使用して、保持ルールを削除できます。

Recycle Bin console

保持ルールを削除するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Retention rules] (保持ルール) を選択します。
3. グリッドで削除する保持ルールを選択し、[Actions] (アクション)、[Delete retention rule] (保持ルールの削除) の順に選択します。

4. プロンプトが表示されたら、確認メッセージを入力し、[Delete retention rule] (保持ルールの削除) を選択します。

AWS CLI

保持ルールを削除するには

delete-rule AWS CLI コマンドを使用します。`--identifier` で、削除するリテンションルールの ID を指定します。

```
$ aws rbin delete-rule --identifier rule_ID
```

例

次のコマンド例では、保持ルール `6lsJ2Fa9nh9` を削除します。

```
$ aws rbin delete-rule --identifier 6lsJ2Fa9nh9
```

ごみ箱内のリソースを使用する

ごみ箱は、次のリソースタイプをサポートしています。

- Amazon EBS スナップショット
- Amazon EBS-backed Amazon マシンイメージ (AMI)

タスク

- ごみ箱からスナップショットを復元する
- ごみ箱から AMI を復旧する

ごみ箱からスナップショットを復元する

ごみ箱は、誤って削除された Amazon EBS スナップショットと EBS-backed AMI を復元することを可能にするデータ復旧機能です。ごみ箱を使用する場合、リソースが削除されると、リソースは、完全に削除されるまでの時間として指定した期間、ごみ箱に保持されます。

リソースは、保持期間が終了する前であればいつでもごみ箱から復元できます。ごみ箱からリソースを復元すると、そのリソースはごみ箱から削除され、アカウント内の他のそのタイプのリソースと同

じ方法で使用できます。保持期間が終了し、リソースが復元されない場合、リソースはごみ箱から完全に削除され、復旧できなくなります。

ごみ箱内のスナップショットは、アカウント内の通常のスナップショットと同じ料金で請求されます。ごみ箱および保持ルールの使用には、追加料金はかかりません。詳細については、Amazon EBS の料金表を参照してください。

詳細については、「ごみ箱」を参照してください。

トピック

- ごみ箱のスナップショットを操作するための権限
- ごみ箱のスナップショットを表示する
- ごみ箱からスナップショットを復元する

ごみ箱のスナップショットを操作するための権限

デフォルトでは、ユーザーには、ごみ箱にあるスナップショットを操作する許可はありません。ユーザーがこれらのリソースを利用するには、特定のリソースと API アクションを使用する許可を付与する IAM ポリシーを作成する必要があります。ポリシーを作成したら、ユーザー、グループ、ロールに許可を追加します。

ごみ箱にあるスナップショットを表示および復旧するには、ユーザーに次の許可が必要です。

- `ec2:ListSnapshotsInRecycleBin`
- `ec2:RestoreSnapshotFromRecycleBin`

ごみ箱内のスナップショットのタグを管理するには、次の追加の許可をユーザーに付与する必要があります。

- `ec2:CreateTags`
- `ec2:DeleteTags`

ごみ箱コンソールを使用するには、ユーザーに `ec2:DescribeTags` 許可が必要です。

IAM ポリシーの例を次に示します。これには、コンソールユーザーの `ec2:DescribeTags` 許可と、タグを管理するための `ec2:CreateTags` および `ec2:DeleteTags` の許可が含まれます。許可が不要な場合は、ポリシーから削除できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
            "ec2:ListSnapshotsInRecycleBin",
            "ec2:RestoreSnapshotFromRecycleBin"
        ],
        "Resource": "*"
      },
      {
        "Effect": "Allow",
        "Action": [
            "ec2:CreateTags",
            "ec2:DeleteTags",
            "ec2:DescribeTags"
        ],
        "Resource": "arn:aws:ec2:Region:account-id:snapshot/*"
      },

    ]
}
```

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「アクセス許可一式を作成」の手順を実行します。

- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。

- IAM ユーザー:
  - ユーザーが継承できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

ごみ箱を使用するために必要な許可の詳細については、「ごみ箱および保持ルールを操作するための許可」を参照してください。

ごみ箱のスナップショットを表示する

スナップショットがごみ箱に入っている間は、次のような限定された情報を表示できます。

- スナップショットの ID。
- スナップショットの説明。
- スナップショットを作成したボリュームの ID。
- スナップショットが削除され、ごみ箱に入った日時。
- 保持期間の有効期限が切れる日時。この時点で、スナップショットはごみ箱から完全に削除されます。

ごみ箱のスナップショットは、次のいずれかの方法を使用して表示できます。

Recycle Bin console

コンソールを使用して、ごみ箱にあるスナップショットを表示するには

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Recycle Bin] (ごみ箱) を選択します。
3. グリッドには、現在ごみ箱にあるすべてのスナップショットがリストされます。特定のスナップショットの詳細を表示するには、グリッドで選択し、[Actions] (アクション)、[View details] (詳細を表示) の順にクリックします。

AWS CLI

AWS CLI を使用してごみ箱のスナップショットを表示するには

list-snapshots-in-recycle-bin AWS CLI コマンドを使用します。`--snapshot-id` オプションを使用して、特定のスナップショットを表示します。または、`--snapshot-id` オプションを省略して、ごみ箱内のすべてのスナップショットを表示します。

```
$ aws ec2 list-snapshots-in-recycle-bin --snapshot-id snapshot_id
```

たとえば、次のコマンドは、ごみ箱にあるスナップショット `snap-01234567890abcdef` に関する情報を提供します。

```
$ aws ec2 list-snapshots-in-recycle-bin --snapshot-id snap-01234567890abcdef
```

出力例:

```
{
    "SnapshotRecycleBinInfo": [
        {
            "Description": "Monthly data backup snapshot",
            "RecycleBinEnterTime": "2021-12-01T13:00:00.000Z",
            "RecycleBinExitTime": "2021-12-15T13:00:00.000Z",
            "VolumeId": "vol-abcdef09876543210",
            "SnapshotId": "snap-01234567890abcdef"
        }
    ]
}
```

ごみ箱からスナップショットを復元する

スナップショットがごみ箱に入っている間は、いかなる方法でも使用することはできません。スナップショットを使用するには、まずスナップショットを復元する必要があります。ごみ箱からスナップショットを復元すると、そのスナップショットはすぐに使用でき、ごみ箱から削除されます。復元されたスナップショットは、アカウント内の他のスナップショットと同じ方法で使用できます。

次のいずれかの方法を使用して、ごみ箱からスナップショットを復元できます。

Recycle Bin console

コンソールを使用してごみ箱からスナップショットから復元する

1. ごみ箱コンソール (https://console.aws.amazon.com/rbin/home/) を開きます。
2. ナビゲーションペインで、[Recycle Bin] (ごみ箱) を選択します。
3. グリッドには、現在ごみ箱にあるすべてのスナップショットがリストされます。復元するスナップショットを選択し、[Recover] (復元) を選択します。
4. プロンプトが表示されたら、[Recover] (復元) を選択します。

AWS CLI

AWS CLI を使用して、削除したスナップショットをごみ箱から復元するには

restore-snapshot-from-recycle-bin AWS CLI コマンドを使用します。--snapshot-id に、復元するスナップショットの ID を指定します。

```
$ aws ec2 restore-snapshot-from-recycle-bin --snapshot-id snapshot_id
```

例えば次のコマンドでは、スナップショット を snap-01234567890abcdef をごみ箱から復元します。

```
$ aws ec2 restore-snapshot-from-recycle-bin --snapshot-id snap-01234567890abcdef
```

出力例:

```
{
    "SnapshotId": "snap-01234567890abcdef",
    "Description": "Monthly data backup snapshot",
    "Encrypted": false,
    "OwnerId": "111122223333",
    "Progress": "100%",
    "StartTime": "2021-12-01T13:00:00.000000+00:00",
    "State": "recovering",
    "VolumeId": "vol-ffffffff",
    "VolumeSize": 30
}
```

ごみ箱から AMI を復旧する

ごみ箱は、誤って削除された Amazon EBS スナップショットと EBS-backed AMI を復元することを可能にするデータ復旧機能です。ごみ箱を使用する場合、リソースが削除されると、リソースは、完全に削除されるまでの時間として指定した期間、ごみ箱に保持されます。

リソースは、保持期間が終了する前であればいつでもごみ箱から復元できます。ごみ箱からリソースを復元すると、そのリソースはごみ箱から削除され、アカウント内の他のそのタイプのリソースと同じ方法で使用できます。保持期間が終了し、リソースが復元されない場合、リソースはごみ箱から完全に削除され、復旧できなくなります。

ごみ箱内の AMI には追加料金は発生しません。

詳細については、ごみ箱 を参照してください。

トピック

ごみ箱内の AMI を操作するための許可

デフォルトでは、ユーザーには、ごみ箱にある AMI を操作する許可はありません。ユーザーがこれらのリソースを利用するには、特定のリソースと API アクションを使用する許可を付与する IAM ポリシーを作成する必要があります。ポリシーを作成したら、ユーザー、グループ、ロールに許可を追加します。

ごみ箱にある AMI を表示および復旧するには、ユーザーに次の許可が必要です。

- `ec2:ListImagesInRecycleBin`
- `ec2:RestoreImageFromRecycleBin`

ごみ箱内の AMI のタグを管理するには、次の追加の許可をユーザーに付与する必要があります。

- `ec2:CreateTags`
- `ec2:DeleteTags`

ごみ箱コンソールを使用するには、ユーザーに `ec2:DescribeTags` 許可が必要です。

IAM ポリシーの例を次に示します。これには、コンソールユーザーの `ec2:DescribeTags` 許可と、タグを管理するための `ec2:CreateTags` および `ec2:DeleteTags` の許可が含まれます。許可が不要な場合は、ポリシーから削除できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
      {
        "Effect": "Allow",
        "Action": [
            "ec2:ListImagesInRecycleBin",
            "ec2:RestoreImageFromRecycleBin"
        ],
        "Resource": "*"
      },
```

```
      {
        "Effect": "Allow",
        "Action": [
            "ec2:CreateTags",
            "ec2:DeleteTags",
            "ec2:DescribeTags"
        ],
        "Resource": "arn:aws:ec2:Region::image/*"
      }
    ]
}
```

アクセス権限を付与するには、ユーザー、グループ、またはロールにアクセス許可を追加します。

- AWS IAM Identity Center のユーザーとグループ:

  アクセス許可セットを作成します。「AWS IAM Identity Center ユーザーガイド」の「アクセス許可一式を作成」の手順を実行します。
- IAM 内で、ID プロバイダーによって管理されているユーザー:

  ID フェデレーションのロールを作成します。詳細については、「IAM ユーザーガイド」の「サードパーティー ID プロバイダー (フェデレーション) 用のロールの作成」を参照してください。
- IAM ユーザー:
  - ユーザーが継承できるロールを作成します。手順については、「IAM ユーザーガイド」の「IAM ユーザー用ロールの作成」を参照してください。
  - (お奨めできない方法) ポリシーをユーザーに直接アタッチするか、ユーザーをユーザーグループに追加する。詳細については、「IAM ユーザーガイド」の「ユーザー (コンソール) へのアクセス権限の追加」を参照してください。

ごみ箱を使用するために必要な許可の詳細については、「ごみ箱および保持ルールを操作するための許可」を参照してください。

ごみ箱内の AMI を表示する

AMI がごみ箱に入っている間は、次のような限定された情報を表示できます。

- AMI の名前、説明、および一意の ID。
- AMI が削除され、ごみ箱に入った日時。
- 保持期間の有効期限が切れる日時。この日時に AMI は完全に削除されます。

ごみ箱内の AMI は、次のいずれかの方法を使用して表示できます。

Recycle Bin console

コンソールを使用して、ごみ箱にある委任された AMI を表示するには

1. console.aws.amazon.com/rbin/home/ でごみ箱コンソールを開きます。
2. ナビゲーションペインで、[Recycle Bin] (ごみ箱) を選択します。
3. グリッドには、現在ごみ箱にあるすべてのリソースが一覧表示されます。特定の AMI の詳細を表示するには、グリッドで選択し、[Actions] (アクション)、[View details] (詳細を表示) の順に選択します。

AWS CLI

AWS CLI を使用して、ごみ箱にある委任された AMI を表示するには

list-images-in-recycle-bin AWS CLI コマンドを使用します。特定の AMI を表示するには、`--image-id` オプションを含めて、表示する AMI の ID を指定します。1 つのリクエストで最大 20 個の ID を指定できます。

ごみ箱内のすべての AMI を表示するには、`--image-id` オプションを省略します。`--max-items` の値を指定しない場合、コマンドはデフォルトで 1 ページあたり 1,000 個のアイテムを返します。詳細については、「Amazon EC2 API リファレンス」の「Pagination」(ページネーション) を参照してください。

```
$ aws ec2 list-images-in-recycle-bin --image-id ami_id
```

例えば、次のコマンドは、ごみ箱にある AMI `ami-01234567890abcdef` に関する情報を表示します。

```
$ aws ec2 list-images-in-recycle-bin --image-id ami-01234567890abcdef
```

出力例:

```
{
    "Images": [
        {
            "ImageId": "ami-0f740206c743d75df",
```

```
            "Name": "My AL2 AMI",
            "Description": "My Amazon Linux 2 AMI",
            "RecycleBinEnterTime": "2021-11-26T21:04:50+00:00",
            "RecycleBinExitTime": "2022-03-06T21:04:50+00:00"
        }
    ]
}
```

> **Important**
>
> 以下のエラーが発生した場合、AWS CLI でバージョンの更新が必要な場合があります。詳細については、「コマンドが見つからないエラー」を参照してください。
>
> ```
> aws.exe: error: argument operation: Invalid choice, valid choices are: ...
> ```

ごみ箱から AMI を復元する

AMI がごみ箱に入っている間は、いかなる方法でも使用できません。AMI を使用するには、まずスナップショットを復元する必要があります。ごみ箱から AMI を復元すると、その AMI はすぐに使用でき、ごみ箱からは削除されます。復元された AMI は、アカウント内の他の AMI と同じ方法で使用できます。

次のいずれかの方法を使用して、ごみ箱から AMI を復元できます。

Recycle Bin console

コンソールを使用してごみ箱から AMI を復元するには

1. console.aws.amazon.com/rbin/home/ でごみ箱コンソールを開きます。
2. ナビゲーションペインで、[Recycle Bin] (ごみ箱) を選択します。
3. グリッドには、現在ごみ箱にあるすべてのリソースが一覧表示されます。復元する AMI を選択し、[Recover] (復旧) を選択します。
4. プロンプトが表示されたら、[Recover] (復元) を選択します。

AWS CLI

AWS CLI を使用して、削除した AMI をごみ箱から復元するには

restore-image-from-recycle-bin AWS CLI コマンドを使用します。--image-id に復元する AMI の ID を指定します。

```
$ aws ec2 restore-image-from-recycle-bin --image-id ami_id
```

例えば、次のコマンドでは AMI ami-01234567890abcdef をごみ箱から復元します。

```
$ aws ec2 restore-image-from-recycle-bin --image-id ami-01234567890abcdef
```

コマンドが正常に完了した場合、出力を返しません。

> **Important**
>
> 以下のエラーが発生した場合、AWS CLI でバージョンの更新が必要な場合があります。詳細については、「コマンドが見つからないエラー」を参照してください。
>
> ```
> aws.exe: error: argument operation: Invalid choice, valid choices are: ...
> ```

ごみ箱をモニタリングする

ごみ箱をモニタリングするには、次の機能を使用することができます。

トピック

- Amazon EventBridge を使用してごみ箱をモニタリングする
- AWS CloudTrail を使用してごみ箱をモニタリングする

Amazon EventBridge を使用してごみ箱をモニタリングする

ごみ箱は、保持ルールに基づいて実行されるアクションのイベントを Amazon EventBridge に送信します。EventBridge を使用することで、これらのイベントへの対応でプログラム的なアクションや通知を呼び出すルールを設定できます。例えば、保持ルールがロック解除され、ロック解除の遅延期間に入ったときにメールに通知を送信する EventBridge ルールを作成できます。詳細については、「イベントに反応する Amazon EventBridge ルールの作成」を参照してください。

EventBridge でのイベントは、JSON オブジェクトとして表されます。イベント固有のフィールドは、JSON オブジェクトの detail セクションに表示されます。event フィールドにはイベント名が入ります。result フィールドには、イベントを開始したアクションの完了時のステータスが入り

ます。詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge のイベントパターン」を参照してください。

Amazon EventBridge の詳細については、「Amazon EventBridge ユーザーガイド」の「Amazon EventBridge とは」を参照してください。

イベント

- RuleLocked
- RuleChangeAttempted
- RuleUnlockScheduled
- RuleUnlockingNotice
- RuleUnlocked

RuleLocked

以下は、保持ルールが正常にロックされた場合にごみ箱が生成するイベントの例です。このイベントは、CreateRule リクエストと LockRule リクエストによって生成できます。イベントを生成した API が `api-name` フィールドに表示されます。

```
{
  "version": "0",
  "id": "exampleb-b491-4cf7-a9f1-bf370example",
  "detail-type": "Recycle Bin Rule Locked",
  "source": "aws.rbin",
  "account": "123456789012",
  "time": "2022-08-10T16:37:50Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
  ],
  "detail":
  {
    "detail-version": " 1.0.0",
    "rule-id": "a12345abcde",
    "rule-description": "locked account level rule",
    "unlock-delay-period": "30 days",
    "api-name": "CreateRule"
  }
}
```

RuleChangeAttempted

以下は、ロックされたルールを変更または削除しようとして失敗した場合にごみ箱が生成するイベントの例です。このイベントは、DeleteRule リクエストと UpdateRule リクエストによって生成できます。イベントを生成した API が `api-name` フィールドに表示されます。

```
{
  "version": "0",
  "id": "exampleb-b491-4cf7-a9f1-bf370example",
  "detail-type": "Recycle Bin Rule Change Attempted",
  "source": "aws.rbin",
  "account": "123456789012",
  "time": "2022-08-10T16:37:50Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
  ],
  "detail":
  {
    "detail-version": " 1.0.0",
    "rule-id": "a12345abcde",
    "rule-description": "locked account level rule",
    "unlock-delay-period": "30 days",
    "api-name": "DeleteRule"
  }
}
```

RuleUnlockScheduled

以下は、保持ルールがロックされロック解除の遅延期間が開始された場合にごみ箱が生成するイベントの例です。

```
{
  "version": "0",
  "id": "exampleb-b491-4cf7-a9f1-bf370example",
  "detail-type": "Recycle Bin Rule Unlock Scheduled",
  "source": "aws.rbin",
  "account": "123456789012",
  "time": "2022-08-10T16:37:50Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
  ],
```

```
  "detail":
  {
    "detail-version": " 1.0.0",
    "rule-id": "a12345abcde",
    "rule-description": "locked account level rule",
    "unlock-delay-period": "30 days",
    "scheduled-unlock-time": "2022-09-10T16:37:50Z",
  }
}
```

RuleUnlockingNotice

以下は、保持ルールがロック解除の遅延期間中に、ロック解除の遅延期間が終了する前日までごみ箱が毎日生成するイベントの例です。

```
{
  "version": "0",
  "id": "exampleb-b491-4cf7-a9f1-bf370example",
  "detail-type": "Recycle Bin Rule Unlocking Notice",
  "source": "aws.rbin",
  "account": "123456789012",
  "time": "2022-08-10T16:37:50Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
  ],
  "detail":
  {
    "detail-version": " 1.0.0",
    "rule-id": "a12345abcde",
    "rule-description": "locked account level rule",
    "unlock-delay-period": "30 days",
    "scheduled-unlock-time": "2022-09-10T16:37:50Z"
  }
}
```

RuleUnlocked

以下は、保存ルールのロック解除の遅延期間が終了し、保持ルールを変更または削除できるようになったときにごみ箱が生成するイベントの例です。

```
{
  "version": "0",
```

```
  "id": "exampleb-b491-4cf7-a9f1-bf370example",
  "detail-type": "Recycle Bin Rule Unlocked",
  "source": "aws.rbin",
  "account": "123456789012",
  "time": "2022-08-10T16:37:50Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
  ],
  "detail":
  {
    "detail-version": " 1.0.0",
    "rule-id": "a12345abcde",
    "rule-description": "locked account level rule",
    "unlock-delay-period": "30 days",
    "scheduled-unlock-time": "2022-09-10T16:37:50Z"
  }
}
```


AWS CloudTrail を使用してごみ箱をモニタリングする

ごみ箱サービスは AWS CloudTrail と統合しています。CloudTrail は、ユーザー、ロール、または AWS のサービスによって実行されたアクションを記録するサービスです。CloudTrail は、ごみ箱で実行されるすべての API コールをイベントとしてキャプチャします。証跡を作成する場合は、Amazon Simple Storage Service (Amazon S3) バケットへの CloudTrail イベントの継続的な配信を有効にすることができます。証跡を設定しない場合でも、CloudTrail コンソールの [Event history] (イベント履歴) で最新の管理イベントを表示できます。CloudTrail で収集された情報を使用して、ごみ箱に対するリクエスト、リクエスト元の IP アドレス、リクエストの実行者、リクエスト日時などの詳細を把握できます。

CloudTrail の詳細については、「AWS CloudTrail ユーザーガイド」を参照してください。

CloudTrail でのごみ箱情報

AWS アカウントを作成すると、そのアカウントに対して CloudTrail が有効になります。サポートされているイベントアクティビティがごみ箱で発生すると、そのアクティビティは [Event history] (イベント履歴) の他の AWS サービスのイベントとともに、CloudTrail イベントにレコードされます。最近のイベントは、AWSアカウントで表示、検索、ダウンロードできます。詳細については、「CloudTrailイベント履歴でのイベントの表示」を参照してください。

ごみ箱のイベントなど、AWS アカウントのイベントの継続的な記録については、証跡を作成します。証跡より、CloudTrail はログファイルを S3 バケットに配信できます。デフォルトでは、

コンソールで証跡を作成するときに、証跡がすべての AWS リージョンに適用されます。証跡では、AWS パーティションのすべてのリージョンからのイベントがログに記録され、指定した S3 バケットにログファイルが配信されます。さらに、CloudTrail ログで収集したイベントデータをより詳細に分析し、それに基づく対応するためにその他の AWS のサービスを設定できます。詳細については、AWS CloudTrail ユーザーガイドの証跡作成の概要を参照してください。

サポートされている API アクション

ごみ箱の場合、CloudTrail を使用して次の API アクションを管理イベントとしてログできます。

- CreateRule
- UpdateRule
- GetRules
- ListRule
- DeleteRule
- TagResource
- UntagResource
- ListTagsForResource
- LockRule
- UnlockRule

管理イベントの記録については、CloudTrail ユーザーガイドの証跡での管理イベントの記録を参照してください。

アイデンティティ情報

各イベントまたはログエントリには、リクエストの生成者に関する情報が含まれます。アイデンティティ情報は、以下を判別するために役立ちます。

- ルートユーザーまたはユーザー認証情報のどちらを使用してリクエストが送信されたか
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが、別の AWS サービスによって送信されたかどうか。

詳細については、 CloudTrail userIdentityElementを参照してください。

ごみ箱ログファイルエントリについて

証跡は、指定した S3 バケットにイベントをログファイルとして配信するように設定できます。CloudTrail ログファイルには、1 つ以上のログエントリがあります。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストのパラメータなどの情報が含まれます。CloudTrail ログファイルは、パブリック API 呼び出しの順序付けられたスタックトレースではないため、特定の順序では表示されません。

以下に CloudTrail ログエントリの例を示します。

CreateRule


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-08-02T21:43:38Z"
      }
    }
  },
  "eventTime": "2021-08-02T21:45:22Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "CreateRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
  "requestParameters": {
```

```
    "retentionPeriod": {
      "retentionPeriodValue": 7,
      "retentionPeriodUnit": "DAYS"
    },
    "description": "Match all snapshots",
    "resourceType": "EBS_SNAPSHOT"
  },
  "responseElements": {
    "identifier": "jkrnexample"
  },
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

GetRule

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
```

```
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-08-02T21:43:38Z"
      }
    }
  },
  "eventTime": "2021-08-02T21:45:33Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "GetRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
  "requestParameters": {
    "identifier": "jkrnexample"
  },
  "responseElements": null,
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

ListRules

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
```

```
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-08-02T21:43:38Z"
      }
    }
  },
  "eventTime": "2021-08-02T21:44:37Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "ListRules",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
  "requestParameters": {
    "resourceTags": [
      {
        "resourceTagKey": "test",
        "resourceTagValue": "test"
      }
    ]
  },
  "responseElements": null,
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

UpdateRule


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-08-02T21:43:38Z"
      }
    }
  },
  "eventTime": "2021-08-02T21:46:03Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "UpdateRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
  "requestParameters": {
    "identifier": "jkrnexample",
    "retentionPeriod": {
      "retentionPeriodValue": 365,
      "retentionPeriodUnit": "DAYS"
    },
    "description": "Match all snapshots",
    "resourceType": "EBS_SNAPSHOT"
  },
  "responseElements": null,
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
```

```
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

DeleteRule

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-08-02T21:43:38Z"
      }
    }
  },
  "eventTime": "2021-08-02T21:46:25Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "DeleteRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
```

```
  "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
  "requestParameters": {
    "identifier": "jkrnexample"
  },
  "responseElements": null,
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

TagResource

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-10-22T21:38:34Z"
      }
```

```
    }
  },
  "eventTime": "2021-10-22T21:43:15Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "TagResource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.26 Python/3.6.14
 Linux/4.9.273-0.1.ac.226.84.332.metal1.x86_64 botocore/1.21.26",
  "requestParameters": {
    "resourceArn": "arn:aws:rbin:us-west-2:123456789012:rule/ABCDEF01234",
    "tags": [
      {
        "key": "purpose",
        "value": "production"
      }
    ]
  },
  "responseElements": null,
  "requestID": "examplee-7962-49ec-8633-795efexample",
  "eventID": "example4-6826-4c0a-bdec-0bab1example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

UntagResource

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-10-22T21:38:34Z"
      }
    }
  },
  "eventTime": "2021-10-22T21:44:16Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "UntagResource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.26 Python/3.6.14
 Linux/4.9.273-0.1.ac.226.84.332.metal1.x86_64 botocore/1.21.26",
  "requestParameters": {
    "resourceArn": "arn:aws:rbin:us-west-2:123456789012:rule/ABCDEF01234",
    "tagKeys": [
      "purpose"
    ]
  },
  "responseElements": null,
  "requestID": "example7-6c1e-4f09-9e46-bb957example",
  "eventID": "example6-75ff-4c94-a1cd-4d5f5example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

ListTagsForResource


```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "mfaAuthenticated": "false",
        "creationDate": "2021-10-22T21:38:34Z"
      }
    }
  },
  "eventTime": "2021-10-22T21:42:31Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "ListTagsForResource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.26 Python/3.6.14
 Linux/4.9.273-0.1.ac.226.84.332.metal1.x86_64 botocore/1.21.26",
  "requestParameters": {
    "resourceArn": "arn:aws:rbin:us-west-2:123456789012:rule/ABCDEF01234"
  },
  "responseElements": null,
  "requestID": "example8-10c7-43d4-b147-3d9d9example",
  "eventID": "example2-24fc-4da7-a479-c9748example",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
```

```
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

LockRule

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2022-10-25T00:45:11Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2022-10-25T00:45:19Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "LockRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "python-requests/2.25.1",
  "requestParameters": {
    "identifier": "jkrnexample",
    "lockConfiguration": {
      "unlockDelay": {
        "unlockDelayValue": 7,
        "unlockDelayUnit": "DAYS"
```

```
      }
    }
  },
  "responseElements": {
    "identifier": "jkrnexample",
    "description": "",
    "resourceType": "EBS_SNAPSHOT",
    "retentionPeriod": {
      "retentionPeriodValue": 7,
      "retentionPeriodUnit": "DAYS"
    },
    "resourceTags": [],
    "status": "available",
    "lockConfiguration": {
      "unlockDelay": {
        "unlockDelayValue": 7,
        "unlockDelayUnit": "DAYS"
      }
    },
    "lockState": "locked"
  },
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

UnlockRule

```
{
  "eventVersion": "1.08",
  "userIdentity": {
    "type": "AssumedRole",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:root",
```

```
    "accountId": "123456789012",
    "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
    "sessionContext": {
      "sessionIssuer": {
        "type": "Role",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:role/Admin",
        "accountId": "123456789012",
        "userName": "Admin"
      },
      "webIdFederationData": {},
      "attributes": {
        "creationDate": "2022-10-25T00:45:11Z",
        "mfaAuthenticated": "false"
      }
    }
  },
  "eventTime": "2022-10-25T00:46:17Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "UnlockRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "python-requests/2.25.1",
  "requestParameters": {
    "identifier": "jkrnexample"
  },
  "responseElements": {
    "identifier": "jkrnexample",
    "description": "",
    "resourceType": "EC2_IMAGE",
    "retentionPeriod": {
      "retentionPeriodValue": 7,
      "retentionPeriodUnit": "DAYS"
    },
    "resourceTags": [],
    "status": "available",
    "lockConfiguration": {
      "unlockDelay": {
        "unlockDelayValue": 7,
        "unlockDelayUnit": "DAYS"
      }
    },
    "lockState": "pending_unlock",
    "lockEndTime": "Nov 1, 2022, 12:46:17 AM"
```

```
  },
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management",
  "tlsDetails": {
    "tlsVersion": "TLSv1.2",
    "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
    "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
}
```

リソースの場所

Amazon EC2 リソースはAWSそのリソースが存在するリージョンまたはアベイラビリティーゾーンに固有のものです。

リソース	タイプ	説明
Amazon EC2 リソース識別子	リージョン別	各リソース識別子 (AMI ID、インスタンス ID、EBS ボリューム ID、EBS スナップショット ID など) はリージョンに固定され、そのリソースを作成したリージョンでのみ使用できます。
ユーザーが指定したリソース名	リージョン別	各リソース名 (セキュリティグループ名前、キーペア名など) はリージョンに固定され、そのリソースを作成したリージョンでのみ使用できます。複数のリージョンで同じ名前のリソースを作成することはできますが、相互に関連付けられてはいません。
AMI	リージョン別	AMI は、Amazon S3 内でそのファイルが置かれているリージョンに固定されます。AMI は、別のリージョンにコピーできます。詳細については、AMI のコピーを参照してください。

リソース	タイプ	説明
EBS スナップショット	リージョン別	EBS スナップショットはリージョンに固定されており、同じリージョン内のボリュームの作成にのみ使用できます。スナップショットは、別のリージョンにコピーできます。
EBS ボリューム	アベイラビリティーゾーン	Amazon EBS ボリュームはアベイラビリティーゾーンに固定されており、同じアベイラビリティーゾーンのインスタンスにのみアタッチできます。
Elastic IP アドレス	リージョン別	Elastic IP アドレスはリージョンに固定されており、同じリージョンのインスタンスにのみ関連付けることができます。
インスタンス	アベイラビリティーゾーン	インスタンスは、そのインスタンスを起動したアベイラビリティーゾーンに固定されています。ただし、インスタンス ID はリージョンに固定されています。
キーペア	グローバルまたはリージョン別	Amazon EC2 を使用して作成したキーペアは、そのペアを作成したリージョンに関連付けられます。独自の RSA キーペアを作成し、それを使用するリージョンにアップロードできます。したがって、各リージョンにアップロードすることで、キーペアをグローバルに利用可能にすることができます。 詳細については、Amazon EC2 のキーペアと Amazon EC2 インスタンスを参照してください。
セキュリティグループ	リージョン別	セキュリティグループはリージョンに固定されており、同じリージョンのインスタンスにのみ割り当てることができます。インスタンスが、セキュリティグループルールを使用するリージョン以外のインスタンスと通信できるようにすることはできません。別のリージョン内のインスタンスからのトラフィックは、WAN 帯域幅とみなされます。

リソース ID

リソースが作成されると、各リソースに一意のリソース ID が割り当てられます。リソース ID は、リソース ID (スナップショットの snap など) にハイフンと英数字の一意の組み合わせが続く形式です。

各リソース識別子 (AMI ID、インスタンス ID、EBS ボリューム ID、EBS スナップショット ID など) はリージョンに固定され、そのリソースを作成したリージョンでのみ使用できます。

リソース ID を使用して、Amazon EC2 コンソールでリソースを見つけることができます。コマンドラインツールまたは Amazon EC2 API を使用して Amazon EC2 を操作している場合、特定のコマンドにはリソース ID が必要になります。例えば、インスタンスを停止するために stop-instances AWS CLI コマンドを使用している場合、コマンドでインスタンス ID を指定する必要があります。

リソース ID の長さ

2016 年 1 月以前に新規作成された特定のリソースタイプのリソースに割り当てられた ID には、ハイフンの後に 8 文字が使用されていました (例: i-1a2b3c4d)。2016 年 1 月から 2018 年 6 月にかけて、これらのリソースタイプの ID は、ハイフンの後に 17 文字を使用するように変更されました (例: `i-1234567890abcdef0`)。アカウントが作成された時期によっては、短い ID を持つ既存のリソースがいくつかある場合がありますが、すべての新しいリソースは長い ID を受け取ります。

リソースの一覧表示およびフィルタリング

Amazon EC2 コンソールを使用して、一部のタイプのリソースのリストを取得できます。対応するコマンドまたは API アクションを使用して、各タイプのリソースのリストを取得できます。リソースが多い場合は、所定の条件に一致するリソースのみを表示、または非表示にするように結果をフィルタリングすることができます。

コンテンツ

コンソールを使用したリソースの一覧表示およびフィルタリング

目次

コンソールを使用したリソースの一覧表示

コンソールを使用して、最も一般的に使用される Amazon EC2 リソースタイプを表示できます。その他のリソースを表示するには、コマンドラインインターフェイスまたは API アクションを使用します。

コンソールを使用して EC2 リソースをリスト表示するには

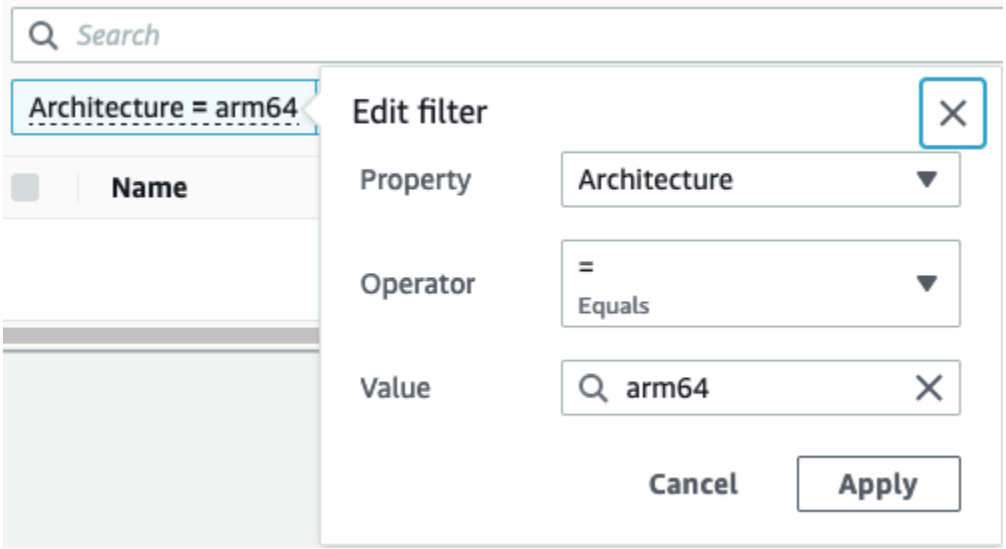
1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、リソースタイプに対応するオプションを選択します。例えば、インスタンスを一覧表示するには、[インスタンス] を選択します。

   このページには、選択したリソースタイプのすべてのリソースが表示されます。

コンソールを使用したリソースのフィルタリング

リソースリストをフィルタリングするには

1. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
2. 検索フィールドを選択します。
3. リストからフィルターを選択します。
4. = (等しい))、などの演算子を選択します。一部の属性では、他の演算子を選択することもできます。すべての画面で演算子が選択できるわけではないことに注意してください。
5. フィルター値を選択します。
6. 選択したフィルタを編集するには、フィルタートークン (青いボックス) を選択し、必要な編集を行った上で [Apply] (適用) をクリックします。すべての画面で、選択したフィルターの編集がサポートされているわけではないことに注意してください。



7. 完了したら、フィルターを削除します。

サポートされているフィルタ

Amazon EC2 コンソールでは、2 種類のフィルタリングがサポートされています。

- API フィルタリングはサーバー側で行われます。このフィルタリングは API コールに適用され、サーバーから返されるリソースの数が減少します。これにより、大量のリソースにわたる迅速なフィルタリングが可能になり、サーバーとブラウザ間のデータ転送時間とコストを削減できます。API に関するフィルタリングでは、[=] (等しい) および [:] (含む) の演算子が使用できます。また、常に大文字と小文字が区別されます。
- クライアントのフィルタリング は、クライアント側で行われます。これにより、ブラウザで既に使用可能なデータ (つまり、APIによって既に返されたデータ) をフィルタリングできます。クライアントフィルタリングは、ブラウザ内の小さなデータセットまでフィルタリングするために、API フィルタと併用してうまく機能します。[=] (等しい) および [:] (含む) の演算子に加え、クライアントフィルタリングでは、[>=] (以上) のような範囲演算子や、[! =] (等しくない) などの否定 (反転) 演算子も使用することができます。

Amazon EC2 コンソールでは、次のタイプの検索がサポートされます。

キーワードによる検索

キーワードによる検索は、検索する属性やタグキーを指定せずに、リソースの属性とタグの全体から値を検索できる、フリーテキスト検索です。

> **Note**
>
> すべてのキーワード検索では、クライアントフィルタリングが使用されます。

キーワードで検索するには、検索フィールドに検索したいキーワードを入力するか貼り付けて、Enter を選択します。例えば 123 を検索すると、IP アドレス、インスタンス ID、VPC ID、AMI ID などの属性の中、あるいは Name などのタグの中に 123 が含まれる、すべてのインスタンスが一致します。フリーテキスト検索で予期しない一致が返された場合は、追加のフィルタを適用します。

属性による検索

属性による検索では、すべてのリソースで特定の属性を検索できます。

> **Note**
>
> 属性検索では、選択した属性に応じて、API フィルタリングまたはクライアントフィルタリングが使用されます。属性検索を実行すると、属性はそれに応じてグループ化されます。

例えば、すべてのインスタンスの [インスタンスの状態] 属性を検索して、`stopped` 状態にあるインスタンスのみを取得することができます。目的:

1. インスタンス画面の検索フィールドで、「`Instance state`」の入力を開始します。文字を入力すると、[Instance state (インスタンスの状態)] には、[API filters (API フィルター)] と [Client filters (クライアントフィルター)] の 2 種類のフィルターが表示されます 。
2. サーバー側で検索するには、[API filters (API フィルター)] で [Instance state (インスタンスの状態)] を選択します。クライアント側で検索するには、[Client filters (クライアントフィルター)] で [Instance state (client) (インスタンスの状態 (クライアント))] を選択します。

   選択した属性に使用可能な演算子のリストが表示されます。
3. [=] (等しい) 演算子をクリックします。

   選択された属性と演算子に適合する可能性のある、値のリストが表示されます。
4. リストから [停止] を選択します。

タグによる検索

タグによる検索では、現在表示されているテーブル内のリソースを、タグキーまたはタグ値でフィルタリングできます。

タグ検索では、[Preferences] (設定) ウィンドウの設定に応じて、API フィルタリングまたはクライアントフィルタリングのどちらかが使用されます。

タグに対し API フィルタリングが使用されるようにするには

1. [Preferences] (設定) ウインドウを開きます。
2. [Use regular expression matching] (正規表現で検索する) チェックボックスをオフにします。このチェックボックスがオンの場合、クライアントのフィルタリングが実行されます。
3. [Use case sensitive matching] (大文字と小文字を区別する) チェックボックスをオンにします。このチェックボックスがオフの場合、クライアントのフィルタリングが実行されます。
4. [確認] を選択します。

タグに関する検索では以下の値を使用できます。

- [(empty)] ((空)) – 指定したタグキーを持ち、かつタグ値を持たないすべてのリソースを検索します。
- [All values] (すべての値) – 指定したタグキーと任意のタグ値を持つすべてのリソースを検索します。
- [Not tagged] (タグ付けなし) – 指定したタグキーを持たないすべてのリソースを検索します。
- [The displayed value] (表示された値) – 指定したタグキーと指定したタグ値を持つすべてのリソースを検索します。

次のテクニックを使用して、検索の精度を高めたり、絞り込んだりできます。

逆順検索

逆検索では、指定した値に一致しないリソースを検索できます。[Instances] (インスタンス) 画面および [AMIs] 画面で逆検索を実行するには、[!=] (等しくない) または [!:] (含まない) 演算子を選択した上で、値を選択します。他の場面では、検索キーワードのプレフィックスに感嘆符 (!) を付けることによって逆検索が実行されます。

> **Note**
>
> 逆検索は、クライアントフィルタのキーワード検索および属性検索でのみサポートされます。API フィルタの属性検索ではサポートされていません。

例えば、すべてのインスタンスの [インスタンスの状態] 属性を検索して、`terminated` 状態にあるインスタンスをすべて除外することができます。目的:

1. インスタンス画面の検索フィールドで、「`Instance state`」の入力を開始します。文字を入力すると、[Instance state (インスタンスの状態)] には、[API filters (API フィルター)] と [Client filters (クライアントフィルター)] の 2 種類のフィルターが表示されます 。
2. [Client filters] (クライアントフィルター) で、[Instance state (client)] (インスタンスの状態 (クライアント)) を選択します。逆検索は、クライアントフィルタでのみサポートされます。

   選択した属性に使用可能な演算子のリストが表示されます。
3. [! =] (等しくない) を選択した上で、[terminated] (終了) をクリックします。

インスタンス状態の属性に基づいてインスタンスをフィルタリングするには、[Instance state (インスタンスの状態)] 列の検索アイコン () を使用することもできます。プラス記号 (+) が付いた検索アイコンは、その属性に一致するすべてのインスタンスを表示します。マイナス記号 (-) が付いた検索アイコンは、その属性に一致するすべてのインスタンスを除外します。

もう 1 つ逆検索の例を挙げます。`launch-wizard-1` という名前のセキュリティグループが割り当てられていないすべてのインスタンスを一覧表示するには、[Client filters] (クライアントフィルター) で、[!=] を選択した上で検索バーに `launch-wizard-1` と入力し、[Security group name] (セキュリティグループ名) 属性による検索を行います。

部分検索

部分検索では、部分文字列値を検索できます。部分検索を実行するには、検索するキーワードの一部だけを入力します。[Instances] (インスタンス) 画面、および [AMIs] 画面では、[:] (含む) 演算子を使用する場合のみ部分検索ができます。他の画面では、クライアントフィルター属性を選択して、キーワードの一部だけを直接入力して検索できます。例えば、[Instance type] (インスタンスタイプ) 画面で `t2.micro`、`t2.small`、`t2.medium` のすべてのインスタンスを検索するには、キーワードに `t2` を入力し、[Instance Type] (インスタンスタイプ) 属性で検索します。

正規表現検索

正規表現検索を使用するには、[Preferences] (設定) ウィンドウで、 [Use regular expression matching] (正規表現で検索する) をオンにする必要があります。

フィールド内の値を特定のパターンと一致させる場合、正規表現が役立ちます。例えば、s で始まる値を検索するには、^s を検索します。xyz で終わる値を検索するには、xyz$ を検索します。または、1 つ以上の文字が続く数字で始まる値を検索するには、[0-9]+.* を検索します。

> **Note**
>
> 正規表現検索は、クライアントフィルタでのキーワード検索および属性検索でのみサポートされます。API フィルタの属性検索ではサポートされていません。

大文字と小文字を区別する検索

大文字と小文字を区別する検索を使用するには、[Preferences] (設定) ウィンドウで、[Use case sensitive matching] (大文字と小文字を区別する) のチェックボックスをオンにする必要があります。大文字と小文字を区別する設定は、クライアントフィルタとタグフィルタにのみ適用されます。

> **Note**
>
> API フィルターでは、常に大文字と小文字が区別されます。

ワイルドカード検索

0 文字以上の文字に一致させるには、* ワイルドカードを使用します。0 文字または 1 文字に一致させるには、? ワイルドカードを使用します。例えば、prod、prods、および production の値を含むデータセットの場合、prod* での検索はすべての値と一致しますが、prod? での検索は prod と prods にのみ一致します。リテラル値を使用するには、バックスラッシュ (\) でエスケープします。例えば、「prod*」 は「prod*」と一致します。

> **Note**
>
> ワイルドカード検索は、API フィルタの属性およびタグ検索でのみサポートされます。これは、キーワード検索、ならびに、クライアントフィルタでの属性とタグによる検索では使用できません。

検索を組み合わせる

一般に、同じ属性を持つ複数のフィルタは、OR で自動的に結合されます。例えば、`Instance State : Running` および `Instance State : Stopped` を検索すると、実行中または停止中のすべてのインスタンスが返されます。AND で検索を結合するには、さまざまな属性を検索します。例えば、`Instance State : Running` および `Instance Type : c4.large` を検索すると、タイプが `c4.large` で、かつ実行状態のインスタンスだけが返されます。

CLI と API を使用した一覧表示およびフィルタリング

リソースタイプごとに、そのタイプのリソースの一覧表示に使用する CLI コマンドまたは API アクションが用意されています。結果として得られるリソースのリストは長くなる場合があるため、特定の条件に一致するリソースのみを含めるように結果をフィルタリングする方がより速く、より便利です。

フィルタリングの考慮事項

- 1 つのリクエストで複数のフィルターと複数のフィルタの値を指定できます。
- フィルタの値には、ワイルドカードを使用することもできます。アスタリスク (*) は 0 個以上の文字、クエスチョンマーク (?) は 0 文字または 1 文字にマッチングします。
- フィルタの値は大文字と小文字が区別されます。
- 検索には、ワイルドカード文字のリテラル値を含めることができます。ただ、文字の前にバックスラッシュを使用してエスケープする必要があります。例えば、`\*amazon\?\\` という値では、リテラル文字列 `*amazon?\` が検索されます。

サポートされているフィルタ

各 Amazon EC2 リソースでサポートされているフィルタを確認するには、次のドキュメントを参照してください。

- AWS CLI: AWS CLI Command Reference-Amazon EC2 の `describe` コマンド。
- Tools for Windows PowerShell: AWS Tools for PowerShell Cmdlet Reference-Amazon EC2 の `Get` コマンド。
- Query API: Amazon EC2 API Reference の `Describe` API アクション。

Example 例: 単一のフィルタを指定する

describe-instances を使用して Amazon EC2 インスタンスを一覧表示できます。フィルタを使用しないと、レスポンスには、すべてのリソースに関する情報が含まれます。次のコマンドを使用して、実行中のインスタンスのみを出力に含めることができます。

```
aws ec2 describe-instances --filters Name=instance-state-name,Values=running
```

実行中のインスタンスのインスタンス ID のみを一覧表示するには、次のように `--query` パラメータを追加します。

```
aws ec2 describe-instances --filters Name=instance-state-name,Values=running --query
 "Reservations[*].Instances[*].InstanceId" --output text
```

出力例を次に示します。

```
i-0ef1f57f78d4775a4
i-0626d4edd54f1286d
i-04a636d18e83cfacb
```

Example 例: 複数のフィルタまたはフィルタ値の指定

複数のフィルタまたは複数のフィルタ値を指定する場合、リソースはすべてのフィルタに一致して結果に含める必要があります。

次のコマンドを使用すると、タイプが `m5.large` または `m5d.large` のいずれかであるすべてのインスタンスを一覧表示できます。

```
aws ec2 describe-instances --filters Name=instance-type,Values=m5.large,m5d.large
```

次のコマンドを使用して、タイプが `t2.micro` であるすべての停止したインスタンスを一覧表示できます。

```
aws ec2 describe-instances --filters Name=instance-state-name,Values=stopped
 Name=instance-type,Values=t2.micro
```

Example 例: フィルタ値でのワイルドカードの使用

describe-snapshots を使用して EBS スナップショットを記述するときに、database フィルタのフィルタ値として description を指定した場合、コマンドは記述が「database」であるスナップショットのみを返します。

```
aws ec2 describe-snapshots --filters Name=description,Values=database
```

ワイルドカード * は、ゼロ文字以上と一致します。フィルタ値として *database* を指定すると、このコマンドは記述に database という単語を含むスナップショットのみを返します。

```
aws ec2 describe-snapshots --filters Name=description,Values=*database*
```

ワイルドカード ? は厳密に 1 文字に一致します。database? をフィルタ値に指定した場合、コマンドは、記述が「database」または「database」の後に 1 文字続くスナップショットのみを返します。

```
aws ec2 describe-snapshots --filters Name=description,Values=database?
```

database???? を指定すると、コマンドは、記述が「database」の後に最大 4 文字続くスナップショットだけを返します。「database」の後に 5 文字以上続く記述は除外されます。

```
aws ec2 describe-snapshots --filters Name=description,Values=database????
```

Example 例: 日付に基づくフィルタリング

AWS CLI では、JMesPath を使用して、式を使用した結果をフィルタリングできます。例えば、次の describe-snapshots コマンドは、指定された日付 (*2020-03-31* で表記) より前に AWS アカウント によって作成されたすべてのスナップショット (*123456789012* で表記) の ID を表示します。所有者を指定しない場合、結果にはすべてのパブリックスナップショットが含まれます。

```
aws ec2 describe-snapshots --filters Name=owner-id,Values=123456789012 --query
 "Snapshots[?(StartTime<='2020-03-31')].[SnapshotId]" --output text
```

次のコマンドは、指定した日付範囲で作成されたすべてのスナップショットの ID を表示します。

```
aws ec2 describe-snapshots --filters Name=owner-id,Values=123456789012 --query
 "Snapshots[?(StartTime>='2019-01-01') && (StartTime<='2019-12-31')].[SnapshotId]" --
output text
```

タグに基づくフィルタリング

タグに従ってリソースのリストをフィルタリングする方法の例については、「コマンドラインによるタグの使用」を参照してください。

Amazon EC2 Global View を使用して、リージョン間のリソースを表示する

Amazon EC2 Global View では、単一の AWS リージョン、または単一のコンソールで同時に複数のリージョンの Amazon EC2 リソースおよび Amazon VPC リソースを表示および検索できます。詳細については、「Amazon EC2 Global View」を参照してください。

Amazon EC2 Global View

Amazon EC2 Global View では、Amazon EC2 および Amazon VPC リソースの一部を、単一のAWS リージョン、または単一のコンソールで複数のリージョンにまたがって表示できます。またAmazon EC2 Global View では、グローバル検索機能を使用して、特定のリソースまたは特定のリソースタイプを複数のリージョンにまたがって同時に検索できます。

Amazon EC2 Global View グローバルビューでは、いかなる方法でもリソースを変更することはできません。

サポート リソース

Amazon EC2 グローバルビューを使用すると、AWS アカウント が有効になっているすべてのリージョンの以下のリソースのグローバルサマリーを表示できます。

- 「Auto Scaling グループ」
- DHCP オプションセット
- Egress-Only インターネットゲートウェイ
- Elastic IP
- エンドポイントサービス
- インスタンス
- インターネットゲートウェイ

- マネージドプレフィックスリスト
- NAT ゲートウェイ
- ネットワーク ACL
- ネットワークインターフェイス
- ルートテーブル
- セキュリティグループ
- サブネット
- ボリューム
- VPC
- VPC エンドポイント
- VPC ピアリング接続

必要なアクセス許可

ユーザーが Amazon EC2 Global View を使用するには、次の許可が必要です。

```
{
  "Version": "2012-10-17",
  "Statement": [
  {
    "Effect": "Allow",
    "Action": [
    "autoscaling:DescribeAutoScalingGroups",
    "ec2:DescribeRegions",
    "ec2:DescribeDhcpOptions",
    "ec2:DescribeEgressOnlyInternetGateways",
    "ec2:DescribeAddresses",
    "ec2:DescribeVpcEndpointServices",
    "ec2:DescribeInstances",
    "ec2:DescribeInternetGateways",
    "ec2:DescribePrefixLists",
    "ec2:DescribeNatGateways",
    "ec2:DescribeNetworkAcls",
    "ec2:DescribeNetworkInterfaces",
    "ec2:DescribeRouteTables",
    "ec2:DescribeSecurityGroups",
    "ec2:DescribeSubnets",
    "ec2:DescribeVolumes",
```

```
    "ec2:DescribeVpcs",
    "ec2:DescribeVpcEndpoints",
    "ec2:DescribeVpcPeeringConnections"
  ],
  "Resource": "*"
  }]
}
```

Amazon EC2 Global Viewを使用するには

Amazon EC2 Global Viewコンソール https://console.aws.amazon.com/ec2globalview/homeを開きます。

> **Important**
>
> Firefox のプライベートウィンドウを使用して Amazon EC2 Global View にアクセスすることはできません。

このコンソールは以下のもので構成されています。

- リージョンエクスプローラー—このタブには、次のセクションがあります。
  - 概要 — すべてのリージョンにわたって、リソースの大まかな概要を提供します。

    有効なリージョンは AWS アカウント が有効になっているリージョンの数を示します。残りのフィールドは、それらのリージョンに現在あるリソースの数を示します。いずれかのリンクを選択すると、すべてのリージョンでそのタイプのリソースを表示します。たとえば、インスタンスラベルの下のリンクが10リージョンで29の場合は、現在リージョン29間のインスタンス10が存在しているということを示しています。リンクを選択して、29すべてのインスタンスのリストを表示します。
  - リージョンごとのリソース数 — すべての AWS リージョン (アカウントが有効になっていないリージョンを含む) を一覧に表示し、各リージョンのリソースタイプの合計を表示します。

    リージョン名を選択すると、特定のリージョンのすべてのタイプのリソースを表示します。例えば、アフリカ (ケープタウン) af-south-1 を選択すると、そのリージョン内の VPC、サブネット、インスタンス、セキュリティグループ、ボリューム、および Auto Scaling グループをすべて表示します。または、リージョンを選択し、選択したリージョンのリソースの表示を選びます。

特定のリージョン内の特定のリソースタイプの値を選択して、そのリージョン内のタイプのリソースのみを表示します。たとえば、アフリカ (ケープタウン) af-south-1のInstancesの値を選択して、そのリージョンのインスタンスのみを表示します。

- グローバル検索-このタブで、1 つのリージョンまたは複数のリージョンにわたって、特定のリソースまたは特定のリソースタイプを検索できます。また、特定のリソースの詳細を表示できます。

  リソースを検索するには、グリッドの前のフィールドに検索条件を入力します。リージョン、リソースタイプ、およびリソースに割り当てられたタグにより検索できます。

  特定のリソースの詳細を表示するには、グリッドでそのリソースを選択します。またリソースのリソース ID を選択して、それぞれのコンソールで開くこともできます。たとえば、インスタンス ID を選択して Amazon EC2 コンソールでインスタンスを開く、またはサブネット ID を選択して Amazon VPC コンソールでサブネットを開きます。

> **Tip**
>
> 特定のリージョンまたはリソースタイプのみを使用する場合は、Amazon EC2 グローバルビューをカスタマイズして、それらのリージョンとリソースタイプのみを表示できます。表示されるリージョンとリソースタイプをカスタマイズするには、ナビゲーションパネルで [設定] を選択し、[リソース] タブと [リージョン] タブで、Amazon EC2 グローバルビューに表示したくないリージョンとリソースタイプを選択します。

Amazon EC2 リソースのタグ付け

インスタンスやイメージなどの Amazon EC2 リソースを管理しやすくするために、独自のメタデータをタグとして各リソースに割り当てることができます。タグを使用すると、AWS リソースを用途、所有者、環境などのさまざまな方法で分類できます。これは、同じタイプのリソースが多数ある場合に役立ちます。割り当てたタグに基づいて、特定のリソースをすばやく識別できます。ここでは、タグとその作成方法について説明します。

> **Warning**
>
> タグのキーと値は、多くの異なる API コールから返されます。`DescribeTags` へのアクセスを拒否しても、他の API から返されるタグへのアクセスは自動的に拒否されません。ベストプラクティスとして、機密データをタグに含めないようお勧めします。

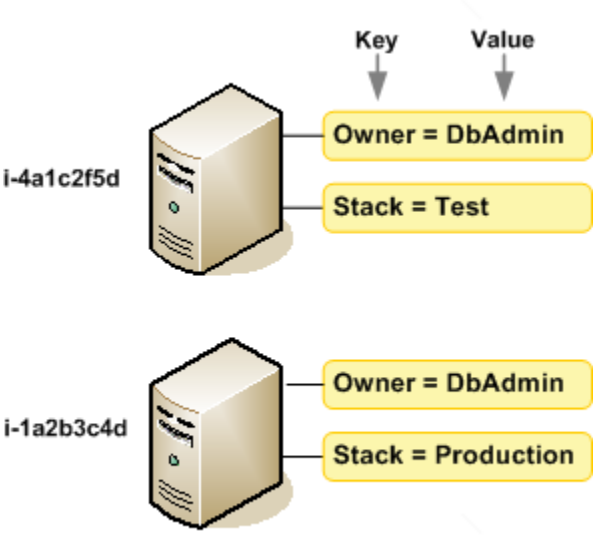
コンテンツ

タグの基本

タグとは、AWS リソースに割り当てるラベルです。タグはそれぞれ、1 つのキーとオプションの 1 つの値で構成されており、どちらもお客様側が定義します。

タグを使用すると、AWS リソースを用途、所有者、環境などのさまざまな方法で分類できます。例えば、アカウントの各インスタンスの所有者とスタックレベルを追跡しやすくするため、Amazon EC2 インスタンスに対して一連のタグを定義できます。

次の図は、タグの機能を示しています。図の中では、インスタンスのそれぞれに 2 つのタグを割り当てています。1 つは `Owner` のキーを使用、もう 1 つは `Stack` キーを使用します。各タグには値も関連付けられています。



ニーズを満たす一連のタグキーをリソースタイプごとに考案されることをお勧めします。一貫性のある一連のタグキーを使用することで、リソースの管理が容易になります。追加したタグに基づいてリソースを検索およびフィルタリングできます。効果的なリソースのタグ付け戦略を実装する方法の詳細については、AWS ホワイトペーパーの「タグ付けのベストプラクティス」を参照してください。

タグには、Amazon EC2 に関連する意味はなく、完全に文字列として解釈されます。また、タグは自動的にリソースに割り当てられます。タグのキーと値は編集でき、タグはリソースからいつでも削除できます。タグの値を空の文字列に設定することはできますが、タグの値を null に設定することはできません。特定のリソースについて既存のタグと同じキーを持つタグを追加した場合、以前の値は新しい値によって上書きされます。リソースを削除すると、リソースのタグも削除されます。

> **Note**
>
> リソースを削除した後も、そのタグがコンソール、API、および CLI の出力にしばらく表示されたままになる場合があります。これらのタグは徐々にリソースから切り離され、完全に削除されます。

リソースのタグ付け

アカウントにすでに存在するほとんどの Amazon EC2 リソースにタグ付けできます。以下の表に、タグ付けをサポートするリソースを示します。

Amazon EC2 コンソールを使用している場合は、関連するリソース画面で [タグ] タブを使用してリソースにタグを適用するか、または AWS Resource Groups コンソールで [タグエディタ] を使用

できます。一部のリソースの画面では、リソースの作成時にリソースのタグを指定できます。例えば、Name のキーと指定した値をタグ付けします。ほとんどの場合、リソースの作成後すぐに (リソースの作成時ではなく) コンソールによりタグが適用されます。コンソールは Name タグに従ってリソースを整理する場合がありますが、このタグには Amazon EC2 サービスに対する意味論的意味はありません。

Amazon EC2 API、AWS CLI、または AWS SDK を使用している場合、CreateTags EC2 API アクションを使用してタグを既存のリソースに適用できます。さらに、リソース作成アクションによっては、リソースの作成時にリソースのタグを指定できます。リソースの作成時にタグを適用できない場合は、リソース作成プロセスがロールバックされます。これにより、リソースがタグ付きで作成されるか、まったく作成されないようになるため、タグ付けされていないリソースが存在することがなくなります。作成時にリソースにタグ付けすることで、リソース作成後にカスタムタグ付けスクリプティングを実行する必要がなくなります。作成時にユーザーがリソースにタグ付けできるようにする方法については、「リソース作成時にタグ付けするアクセス許可の付与」を参照してください。

次の表では、タグ付け可能な Amazon EC2 リソースと、Amazon EC2 API、AWS CLI、または AWS SDK を使用した作成時にタグ付け可能なリソースについて説明します。

Amazon EC2 リソースのタグ付けのサポート

リソース	タグをサポート	作成時のタグ付けをサポート
AFI	はい	はい
AMI	はい	はい
バンドルタスク	いいえ	いいえ
Capacity Reservation	はい	はい
キャリアゲートウェイ	はい	はい
クライアント VPN エンドポイント	はい	はい
クライアント VPN ルート	いいえ	いいえ
カスタマーゲートウェイ	はい	はい
Dedicated Host	はい	はい

リソース	タグをサポート	作成時のタグ付けをサポート
Dedicated Host 予約	はい	はい
DHCP オプション	はい	はい
EBS スナップショット	はい	はい
EBS ボリューム	はい	はい
EC2 Fleet	はい	はい
Egress-only インターネットゲートウェイ	はい	はい
Elastic IP アドレス	はい	はい
Elastic Graphics アクセラレーター	はい	いいえ
インスタンス	はい	はい
インスタンスイベントウィンドウ	はい	はい
インスタンスストアボリューム	該当なし	該当なし
インターネットゲートウェイ	はい	はい
IP アドレスプール (BYOIP)	はい	はい
キーペア	はい	はい
起動テンプレート	はい	はい
起動テンプレートのバージョン	いいえ	いいえ
ローカルゲートウェイ	はい	いいえ

リソース	タグをサポート	作成時のタグ付けをサポート
ローカルゲートウェイルートテーブル	はい	いいえ
ローカルゲートウェイ仮想インターフェイス	はい	いいえ
ローカルゲートウェイ仮想インターフェイスグループ	はい	いいえ
ローカルゲートウェイルートテーブル VPC の関連付け	はい	はい
ローカルゲートウェイルートテーブル仮想インターフェイスグループの関連付け	はい	いいえ
NAT ゲートウェイ	はい	はい
ネットワーク ACL	はい	はい
ネットワークインターフェイス	はい	はい
配置グループ	はい	はい
プレフィックスリスト	はい	はい
Reserved Instance	はい	いいえ
リザーブドインスタンス出品	いいえ	いいえ
ルートテーブル	はい	はい
スポットフリートのリクエスト	はい	はい
スポットインスタンスリクエスト	はい	はい

リソース	タグをサポート	作成時のタグ付けをサポート
セキュリティグループ	はい	はい
セキュリティグループルール	はい	いいえ
サブネット	はい	はい
Traffic Mirror フィルタ	はい	はい
Traffic Mirror セッション	はい	はい
Traffic Mirror ターゲット	はい	はい
トランジットゲートウェイ	はい	はい
Transit Gateway のマルチキャストドメイン	はい	はい
トランジットゲートウェイルートテーブル	はい	はい
トランジットゲートウェイ VPC アタッチメント	はい	はい
仮想プライベートゲートウェイ	はい	はい
VPC	はい	はい
VPC エンドポイント	はい	はい
VPC エンドポイントサービス	はい	はい
VPC エンドポイントサービス設定	はい	はい
VPC フローログ	はい	はい
VPC ピア接続	はい	はい

リソース	タグをサポート	作成時のタグ付けをサポート
VPN 接続	はい	はい

作成時に、Amazon EC2 コンソールの Amazon EC2 インスタンス起動ウィザードを使用して、インスタンス、ボリューム、Elastic Graphics、ネットワークインターフェイス、スポットインスタンスリクエストにタグを付けることができます。[ボリューム] 画面を使用して作成時に EBS ボリュームにタグを付けたり、[スナップショット] 画面を使用して EBS スナップショットにタグを付けたりすることができます。または、リソースを作成するときに、リソース作成 Amazon EC2 API (RunInstances など) を使用してタグを適用することもできます。

IAM ポリシーでタグベースのリソースレベルアクセス許可を、作成時のタグ付けをサポートする Amazon EC2 API アクションに適用し、作成時にリソースにタグ付けできるユーザーとグループを細かく制御できます。リソースは、作成時から適切に保護されます。タグはリソースに即座に適用されるため、リソースの使用を制御するタグベースのリソースレベルアクセス権限がただちに有効になります。リソースは、より正確に追跡および報告されます。新しいリソースにタグ付けの使用を適用し、リソースで設定されるタグキーと値を制御できます。

さらに、リソースレベルのアクセス許可を IAM ポリシーの `CreateTags` および `DeleteTags` Amazon EC2 API アクションに適用し、既存のリソースで設定されるタグキーと値を制御することもできます。詳細については、「例: リソースのタグ付け」を参照してください。

請求用リソースへのタグ付けの詳細については、AWS Billing ユーザーガイドの「コスト配分タグの使用」を参照してください。

タグの制限

タグには以下のような基本制限があります。

- リソースあたりのタグの最大数 - 50 件
- タグキーは、リソースごとにそれぞれ一意である必要があります。また、各タグキーに設定できる値は 1 つのみです。
- キーの最大長 - UTF-8 の 128 Unicode 文字
- 値の最大長 - UTF-8 の 256 Unicode 文字
- 使用できる文字

  - EC2 ではタグ内に任意の文字を使用できますが、他の AWS のサービスでは制限があります。すべての AWS のサービスで使用できる文字は、UTF-8 で表現できる英字 (a-z、A-Z)、数字 (0-9)、スペース、および + - = . _ : / @ です。
  - インスタンスメタデータでインスタンスタグを有効にすると、インスタンスタグキーは文字 (a-z、A-Z)、数字 (0-9)、および次の文字のみを使用できます: + - = . , _ : @。インスタンスタグキーは、スペースまたは / を含めることはできず、. (1 つのピリオド)、.. (2 つのピリオド)、または _index のみを含めることはできません。詳細については、インスタンスメタデータ内のインスタンスタグの使用を参照してください。
- タグのキーと値は大文字と小文字が区別されます。
- aws: プレフィックスは AWS 用に限定されています。タグにこのプレフィックスが付いたタグキーがある場合、タグのキーまたは値を編集、削除することはできません。aws: プレフィックスを持つタグは、リソースあたりのタグ数の制限時には計算されません。

タグのみに基づいてリソースを終了、停止、終了することはできません。リソース識別子を指定する必要があります。例えば、DeleteMe というタグキーを使用してタグ付けしたスナップショットを削除するには、DeleteSnapshots のようなスナップショットのリソース識別子を指定して snap-1234567890abcdef0 アクションを使用する必要があります。

パブリックリソースまたは共有リソースにタグを付けると、割り当てたタグは、タグ付けを行った AWS アカウントだけが使用できます。他の AWS アカウントはそれらのタグにアクセスできません。共有リソースへのタグベースのアクセス制御では、リソースへのアクセスを制御するために、各 AWS アカウントに独自のタグセットを割り当てる必要があります。

すべてのリソースにタグ付けすることはできません。詳細については、「Amazon EC2 リソースのタグ付けのサポート」を参照してください。

タグとアクセス管理

AWS Identity and Access Management (IAM) を使用している場合は、タグを作成、編集、削除する許可を持つ AWS アカウントのユーザーを制御できます。詳細については、「リソース作成時にタグ付けするアクセス許可の付与」を参照してください。

リソースタグを使用して、属性ベースの制御 (ABAC) を実装することもできます。リソースのタグに基づいてオペレーションを許可する IAM ポリシーを作成できます。詳細については、「リソースタグを使用した EC2 リソースへのアクセスの制御」を参照してください。

請求用のリソースのタグ付け

タグを使用して AWS 請求書を整理し、自分のコスト構造を反映できます。そのためには、サインアップして、タグキー値が含まれた AWS アカウントの請求書を取得する必要があります。タグによるコスト配分レポートの設定の詳細については、『AWS Billing ユーザーガイド』の「コスト配分月次レポート」を参照してください。リソースを組み合わせたコストを確認するには、同じタグキー値を持つリソースに基づいて、請求情報を整理します。例えば、複数のリソースに特定のアプリケーション名のタグを付け、請求情報を整理することで、複数のサービスを利用しているアプリケーションの合計コストを確認することができます。詳細については、AWS Billing ユーザーガイド の「コスト配分タグの使用」を参照してください。

> **Note**
>
> レポートを有効にすると、約 24 時間後に、今月のデータを表示できるようになります。

コスト割り当てタグは、どのリソースがコストに貢献しているかを示すことができますが、リソースを削除または非アクティブ化にしてもコストは必ずしも削減されるわけではありません。例えば、元のデータを含むスナップショットが削除された場合でも、別のスナップショットによって参照されるスナップショットデータは維持されます。詳細については、AWS Billing ユーザーガイドの「Amazon Elastic Block Store のボリュームとスナップショット」を参照してください。

> **Note**
>
> タグされている Elastic IP アドレスは、コスト配分レポートには表示されません。

コンソールでのタグの使用

Amazon EC2 コンソールを使用して、個々のリソースのタグを表示し、一度に 1 つのリソースに対してタグを適用または削除できます。

AWS Resource Groups コンソールでタグエディタを使用すると、すべてのリージョンにおけるすべての Amazon EC2 リソースのタグを表示できます。リソース別およびリソースタイプ別にタグを表示でき、指定したタグにどのリソースタイプが関連付けられているかを確認できます。複数のリソースおよび複数のリソースタイプに対して一度にタグを適用または削除できます。タグエディタでは、統一された方法で一元的にタグを作成および管理できます。詳細については、「AWS リソースのタグ付けのユーザーガイド」を参照してください。

タスク

- タグの表示
- 個々のリソースのタグの追加および削除
- 複数のリソースのタグを追加および削除する
- インスタンスを起動するときのタグの追加
- タグによるリソースリストのフィルタリング

タグの表示

Amazon EC2 コンソールで個々のリソースのタグを表示できます。すべてのリソースのタグを表示するには、AWS Resource Groups コンソールのタグエディタを使用します。

個々のリソースのタグを表示する

Amazon EC2 コンソールでリソース固有のページを選択すると、リソースリストが表示されます。例えば、ナビゲーションペインの [Instances (インスタンス)] を選択すると、コンソールに Amazon EC2 インスタンスが表示されます。このようなリスト (インスタンスなど) からリソースを選択し、リソースがタグをサポートしている場合、タグを表示し、管理することができます。ほとんどのリソースページでは、[Tags] (タグ) タブを選択してタグを表示できます。

リソースリストに列を追加して、同じキーを持つタグのすべての値を表示できます。この列を使用して、タグによるリソースリストの並べ替えやフィルタリングを行うことができます。

New console

リソースリストに列を追加してタグを表示するには

1. EC2 コンソールで、画面右上にある歯車の形をした [詳細設定] アイコンを選択します。
2. [詳細設定] ダイアログボックスの [タグ] 列 (左下) で、1 つ以上のタグキーを選択し、[確認] を選択します。

Old console

新しい列をリソースリストに追加してタグを表示するには、2 つの方法があります。

- [Tags] タブで、[Show Column] を選択します。新しい列がコンソールに追加されます。
- [Show/Hide Columns] (歯車型のアイコン) を選択し、[Show/Hide Columns] ダイアログボックスの [Your Tag Keys] のタグキーを選択します。

複数のリソースのタグを表示する

AWS Resource Groups コンソールでタグエディタを使用すると、複数のリソースのタグを表示できます。詳細については、「AWS リソースのタグ付けのユーザーガイド」を参照してください。

個々のリソースのタグの追加および削除

リソースのページから、個々のリソースのタグを直接管理できます。

個々のリソースにタグを追加するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、タグ付けするリソースがあるリージョンを選択します。詳細については、「リソースの場所」を参照してください。
3. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
4. リソースリストからリソースを選択し、[Tags (タグ)] タブを選択します。
5. [タグを管理] を選択し、[新しいタグを追加] を選択します。タグのキーと値を入力します。追加するタグごとに [新しいタグを追加] を選択します。タグの追加を完了したら、[Save (保存)] を選択します。

個々のリソースからタグを削除するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションバーから、タグ付けを解除するリソースがあるリージョンを選択します。詳細については、「リソースの場所」を参照してください。
3. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
4. リソースリストからリソースを選択し、[Tags (タグ)] タブを選択します。
5. [Manage tags (タグの管理)] を選択します。削除する各タグについて、[削除] を選択します。タグの削除を完了したら、[Save (保存)] を選択します。

複数のリソースのタグを追加および削除する

複数のリソースにタグを追加するには

1. AWS リソースグループコンソール (https://console.aws.amazon.com/resource-groups/tag-editor) でタグエディタを開きます。

2. [リージョン] で、タグ付けするリソースが存在する 1 つ以上のリージョンを選択します。
3. [リソースタイプ] で、タグ付けするリソースのタイプを選択します (AWS::EC2::Instance など)。
4. [リソースを検索] を選択します。
5. [リソースの検索結果] で、タグ付けする各リソースの横にあるチェックボックスをオンにします。
6. [選択したリソースのタグの管理] を選択します。
7. [選択したすべてのリソースのタグを編集] で、[タグを追加] を選択し、新しいタグのキーと値を入力します。追加するタグごとに [Add tag] (タグを追加) を選択します。

> **Note**
>
> 既存のタグとタグキーが同じ新しいタグを追加すると、既存のタグは新しいタグで上書きされます。

8. [タグの変更を確認して適用] を選択します。
9. [Apply changes to all selected (選択したすべてに変更を適用)] を選択します。

複数のリソースからタグを削除するには

1. AWS リソースグループコンソール (https://console.aws.amazon.com/resource-groups/tag-editor) でタグエディタを開きます。
2. [リージョン] で、タグを解除するリソースが存在するリージョンを選択します。
3. [リソースタイプ] で、タグを解除するリソースのタイプを選択します (AWS::EC2::Instance など)。
4. [リソースを検索] を選択します。
5. [リソースの検索結果] で、タグを解除する各リソースの横にあるチェックボックスをオンにします。
6. [選択したリソースのタグの管理] を選択します。
7. [選択したすべてのリソースのタグを編集] で、削除するタグの横にある [タグを削除] を選択します。
8. [タグの変更を確認して適用] を選択します。
9. [Apply changes to all selected (選択したすべてに変更を適用)] を選択します。

インスタンスを起動するときのタグの追加

New console

インスタンスの起動ウィザードを使用してタグを追加するには

1. ナビゲーションバーで、インスタンスを起動するリージョンを選択します。一部の Amazon EC2 リソースはリージョン間で共有できるため、この選択は重要です。ニーズに合ったリージョンを選択します。詳細については、「リソースの場所」を参照してください。
2. [インスタンスの起動] を選択します。
3. [Names and tags] (名前とタグ) で、インスタンス用にわかりやすい名前を入力し、タグを指定します。

   インスタンス名はタグで、キーは [Name] (名前)、値は指定した名前です。インスタンス、ボリューム、Elastic Graphics、ネットワークインターフェイスにタグ付けできます。スポットインスタンスの場合、スポットインスタンスリクエストにのみタグを付けることができます。

   インスタンス名と追加のタグを指定することはオプションです。

   - [Name] (名前) に、インスタンスのわかりやすい名前を入力します。名前を指定しない場合は、インスタンスをその ID で識別できます。ID は、インスタンスの起動時に自動的に生成されます。
   - タグを追加するには、[Add additional tag] (追加のタグを追加) を選択します。[Add tag] (タグを追加) を選択し、キーと値を入力し、タグ付けするリソースタイプを選択します。追加するタグごとに [Add tag] (タグを追加) を選択します。
4. [Application and OS Images (Amazon Machine Image)] (アプリケーションおよび OS イメージ (Amazon マシンイメージ)) で、インスタンスと AMI 用のオペレーティングシステム (OS) を選択します。詳細については、「アプリケーションと OS イメージ (Amazon マシンイメージ)」を参照してください。
5. [Key pair (login)] (キーペア (ログイン)) の [Key pair name] (キーペア名) で、既存のキーペアを選択するか、新しいキーペアを作成します。
6. その他のフィールドはすべてデフォルト値のままにするか、希望するインスタンス設定に合わせて特定の値を選択します。各フィールドの詳細については、「定義済みのパラメータを使用したインスタンスの起動」を参照してください。
7. [Summary] (概要) パネルで設定を確認し、[Launch instance] (インスタンスを起動) を選択します。

Old console

インスタンスの起動ウィザードを使用してタグを追加するには

1. ナビゲーションバーで、インスタンスを起動するリージョンを選択します。一部の Amazon EC2 リソースはリージョン間で共有できるため、この選択は重要です。ニーズに合ったリージョンを選択します。詳細については、「リソースの場所」を参照してください。
2. [インスタンスの作成] を選択します。
3. [Choose an Amazon Machine Image (AMI)] ページには、Amazon マシンイメージ (AMI) と呼ばれる基本設定リストが表示されます。使用する AMI を選択し、[Select] を選択します。詳細については、「Linux AMI の検索」を参照してください。
4. [Configure Instance Details] ページで、必要に応じてインスタンスの設定を行い、[Next: Add Storage] を選択します。
5. [Add Storage] ページで、インスタンスに追加のストレージボリュームを指定できます。完了したら、[Next: Add Tags] を選択します。
6. [Add Tags] ページで、インスタンス、ボリューム、またはその両方のタグを指定します。インスタンスに複数のタグを追加するには、[Add another tag] を選択します。完了したら、[次の手順: セキュリティグループの設定] を選択します。
7. [Configure Security Group] ページで、所有する既存のセキュリティグループから選択するか、ウィザードで新しいセキュリティグループを作成します。完了したら、[Review and Launch] を選択します。
8. 設定を確認します。選択した内容でよければ、[Launch] を選択します。既存のキーペアを選択するか、新しいキーペアを作成し、確認のチェックボックスを選択して、[Launch Instances] を選択します。

タグによるリソースリストのフィルタリング

1 つまたは複数のタグキーとタグ値に基づいて、リソースリストをフィルタリングできます。

Amazon EC2 コンソールでリソースのリストをタグでフィルタリングするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、リソースタイプを選択します ([Instances] など)。
3. 検索フィールドを選択します。
4. リストの [タグ] で、タグキーを選択します。
5. リストから対応するタグ値を選択します。

6. 完了したら、フィルターを削除します。

Amazon EC2 コンソールでのフィルターの使用の詳細については、「リソースの一覧表示およびフィルタリング」を参照してください。

タグエディタを使用して、複数のリージョンにわたって複数のリソースをタグでフィルタリングするには

AWS リソースグループコンソールのタグエディタを使用して、複のリージョンにわたって複数のリソースをタグでフィルタリングできます。詳細については、「AWS リソースのタグ付けユーザーガイド」の「タグ付けするリソースの検索」を参照してください。

コマンドラインによるタグの使用

作成時に、create コマンドのタグ仕様パラメータを使用して、多くの EC2 リソースにタグを追加できます。リソースの describe コマンドを使用して、リソースのタグを表示できます。次のコマンドを使用して、既存のリソースのタグを追加、更新、または削除することもできます。

タスク	AWS CLI	AWS Tools for Windows PowerShell
1 つ以上のタグを追加、または上書きします	create-tags	New-EC2Tag
1 つ以上のタグを削除します	delete-tags	Remove-EC2Tag
1 つ以上のタグを記述します	describe-tags	Get-EC2Tag

タスク

- リソース作成時のタグの追加
- 既存のリソースへのタグの追加
- タグ付きリソースの説明

リソース作成時のタグの追加

次の例は、リソースの作成時にタグを適用する方法を示しています。

> **Note**
>
> コマンドラインで JSON 形式のパラメータを入力する方法はオペレーティングシステムによって異なります。
>
> - Linux、macOS、または Unix と Windows PowerShell - 一重引用符 (') を使用して JSON データ構造を囲みます。
> - Windows - Windows コマンドラインでコマンドを使用するときは一重引用符を省略します。
>
> 詳細については、「AWS CLI のパラメータ値の指定」を参照してください。

Example 例: インスタンスを起動し、インスタンスおよびボリュームにタグを適用する

次の run-instances コマンドは、インスタンスを起動し、キー **webserver** と値 **production** を含むタグをインスタンスに適用します。さらに、**cost-center** キーと **cc123** の値を持つタグを、作成された EBS ボリューム (この場合はルートボリューム) に適用します。

```
aws ec2 run-instances \
    --image-id ami-abc12345 \
    --count 1 \
    --instance-type t2.micro \
    --key-name MyKeyPair \
    --subnet-id subnet-6e7f829e \
    --tag-specifications
 'ResourceType=instance,Tags=[{Key=webserver,Value=production}]'
 'ResourceType=volume,Tags=[{Key=cost-center,Value=cc123}]'
```

起動時にインスタンスとボリュームの両方に同じタグキーと値を適用できます。次のコマンドは、インスタンスを起動し、**cost-center** のキーと **cc123** の値を持つタグを、作成されたインスタンスとすべての EBS ボリュームに適用します。

```
aws ec2 run-instances \
    --image-id ami-abc12345 \
    --count 1 \
    --instance-type t2.micro \
    --key-name MyKeyPair \
    --subnet-id subnet-6e7f829e \
```

```
    --tag-specifications 'ResourceType=instance,Tags=[{Key=cost-center,Value=cc123}]'
 'ResourceType=volume,Tags=[{Key=cost-center,Value=cc123}]'
```

Example 例: ボリュームを作成してタグを適用する

次の create-volume コマンドは、ボリュームを作成し、2 つのタグ **purpose=production** および **cost-center=cc123** を適用します。

```
aws ec2 create-volume \
    --availability-zone us-east-1a \
    --volume-type gp2 \
    --size 80 \
    --tag-specifications 'ResourceType=volume,Tags=[{Key=purpose,Value=production},
{Key=cost-center,Value=cc123}]'
```

既存のリソースへのタグの追加

次の例は、create-tags コマンドを使用して既存のリソースにタグを追加する方法を示しています。

Example 例: リソースにタグを追加する

次のコマンドでは、タグ (**Stack=production**) を指定されたイメージに追加するか、タグキーが **Stack** の AMI 用に既存のタグを上書きします。コマンドが成功した場合、出力は返りません。

```
aws ec2 create-tags \
    --resources ami-78a54011 \
    --tags Key=Stack,Value=production
```

Example 例: タグを複数のリソースに追加する

この例では、2 つのタグを AMI とインスタンス用に追加 (または上書き) します。一方のタグにはキー (**webserver**) のみ含まれており、値は設定されていません (値を空の文字列に設定)。もう 1 つのタグはキー (**stack**) と値 (**Production**) で構成されます。コマンドが成功した場合、出力は返りません。

```
aws ec2 create-tags \
    --resources ami-1a2b3c4d i-1234567890abcdef0 \
    --tags Key=webserver,Value=  Key=stack,Value=Production
```

Example 例: 特殊文字のタグを追加する

この例では、タグ (**[Group]=test**) をインスタンスに追加します。角括弧 (**[** および **]**) は特殊文字であり、エスケープする必要があります。

Linux または OS X を使用している場合、特殊文字をエスケープするには、特殊文字を含む要素を二重引用符 (**"**) で囲んでから、キーと値の構造全体を一重引用符 (**'**) で囲みます。

```
aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags 'Key="[Group]",Value=test'
```

Windows を使用している場合、特殊文字をエスケープするには、特殊文字を含む要素を二重引用符 (") で囲み、各二重引用符の前にバックスラッシュ (****) を付けます。

```
aws ec2 create-tags ^
    --resources i-1234567890abcdef0 ^
    --tags Key=\"[Group]\",Value=test
```

Windows PowerShell を使用している場合、特殊文字をエスケープするには、次のように特殊文字を含む値を二重引用符 (**"**) で囲み、各二重引用符の前にバックスラッシュ (****) を付けてから、キーと値の構造全体を一重引用符 (**'**) で囲みます。

```
aws ec2 create-tags `
    --resources i-1234567890abcdef0 `
    --tags 'Key=\"[Group]\",Value=test'
```

タグ付きリソースの説明

次の例は、describe-instances でフィルターを使用して、特定のタグを持つインスタンスを表示する方法を示しています。すべての EC2 describe コマンドは、この構文を使用して、1 つのリソースタイプ全体でタグに基づいてフィルタリングします。または、describe-tags コマンドを使用して、EC2 リソースタイプ間でタグに基づいてフィルタリングすることもできます。

Example 例: 特定のタグキーでインスタンスの詳細を示します。

次のコマンドは、タグの値にかかわらず **Stack** タグでインスタンスの詳細を示します。

```
aws ec2 describe-instances \
    --filters Name=tag-key,Values=Stack
```

Example 例: 特定のタグでインスタンスの詳細を示します。

次のコマンドは、**Stack=production** タグでインスタンスの詳細を示します。

```
aws ec2 describe-instances \
    --filters Name=tag:Stack,Values=production
```

Example 例: 特定のタグの値でインスタンスの詳細を示します。

次のコマンドは、タグキーにかかわらず値 **production** を持つタグでインスタンスの詳細を示します。

```
aws ec2 describe-instances \
    --filters Name=tag-value,Values=production
```

Example 例: 指定したタグを持つすべての EC2 リソースの詳細を示す

次のコマンドは、タグ **Stack=Test** を持つすべての EC2 リソースの詳細を示します。

```
aws ec2 describe-tags \
    --filters Name=key,Values=Stack Name=value,Values=Test
```

インスタンスメタデータ内のインスタンスタグの使用

インスタンスのメタデータからインスタンスのタグにアクセスできます。インスタンスメタデータからタグにアクセスすると、DescribeInstances または DescribeTags API コールを使用してタグ情報を取得する必要がなくなります。これにより、1 秒あたりの API トランザクションが削減され、制御するインスタンスの数に応じてタグ取得をスケーリングできるようになります。さらに、インスタンスで実行されているローカルプロセスは、インスタンスのメタデータからインスタンスのタグ情報を直接表示できます。

デフォルトでは、インスタンスメタデータからタグは使用できません。アクセスを明示的に許可する必要があります。インスタンスの起動時、または実行中または停止したインスタンスでの起動後にアクセスを許可できます。起動テンプレートでこれを指定することで、タグへのアクセスを許可することもできます。テンプレートを使用してインスタンスが起動されると、インスタンスメタデータ内のタグへのアクセスが許可されます。

インスタンスタグを追加または削除すると、インスタンスの実行中にインスタンスのメタデータが更新されます。インスタンスを停止して起動したりする必要はありません。

トピック

- インスタンスメタデータのタグへのアクセスを許可する
- インスタンスメタデータのタグへのアクセスを無効にする
- インスタンスメタデータでのタグへのアクセスが許可されるか確認する
- インスタンスメタデータからタグを取得する

インスタンスメタデータのタグへのアクセスを許可する

デフォルトでは、インスタンスメタデータ内のインスタンスタグへのアクセス権はありません。インスタンスごとに、次のいずれかの方法を使用してアクセスを明示的に許可する必要があります。

コンソールを使用してインスタンスメタデータのタグへのアクセスを許可するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Actions] (アクション)、[Instance settings] (インスタンス設定)、[Allow tags in instance metadata] (インスタンスメタデータ内のタグを許可する) の順に選択します。
4. インスタンスメタデータ内のタグへのアクセスを許可するには、[Allow] (許可) チェックボックスをオンにします。
5. [Save] を選択します。

AWS CLI を使用して起動時にインスタンスメタデータのタグへのアクセスを許可するには

run-instances コマンドを使用して、`InstanceMetadataTags` を `enabled` に設定します。

```
aws ec2 run-instances \
    --image-id ami-0abcdef1234567890 \
    --instance-type c3.large \
    ...
    --metadata-options "InstanceMetadataTags=enabled"
```

AWS CLI を使用して実行中または停止中のインスタンス上のインスタンスメタデータ内のタグへのアクセスを許可するには

modify-instance-metadata-options コマンドを使用して、`--instance-metadata-tags` を `enabled` に設定します。

```
aws ec2 modify-instance-metadata-options \
```

```
    --instance-id i-123456789example \
    --instance-metadata-tags enabled
```

インスタンスメタデータのタグへのアクセスを無効にする

インスタンスメタデータ内のインスタンスタグへのアクセスを無効にするには、次のいずれかの方法を使用します。インスタンスメタデータのインスタンスタグへのアクセスは、デフォルトでオフになっているため、起動時にインスタンスタグへのアクセスをオフにする必要はありません。

コンソールを使用してインスタンスメタデータのタグへのアクセスを無効にするには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Actions] (アクション)、[Instance settings] (インスタンス設定)、[Allow tags in instance metadata] (インスタンスメタデータ内のタグを許可する) の順に選択します。
4. インスタンスメタデータ内のタグへのアクセスを無効にするには、[Allow] (許可) チェックボックスをオフにします。
5. [Save] を選択します。

AWS CLI を使用してインスタンスメタデータのタグへのアクセスを無効にするには

modify-instance-metadata-options コマンドを使用して、`--instance-metadata-tags` を `disabled` に設定します。

```
aws ec2 modify-instance-metadata-options \
    --instance-id i-123456789example \
    --instance-metadata-tags disabled
```

インスタンスメタデータでのタグへのアクセスが許可されるか確認する

インスタンスごとに、Amazon EC2 コンソールまたは AWS CLI を使用し、インスタンスメタデータからのインスタンスタグへのアクセスが許可されているかどうかを確認できます。

コンソールを使用してインスタンスメタデータのタグへのアクセスが許可されているかを確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択してから、インスタンスを選択します。

3. [Detail] (詳細) タブで、[Allow tags in instance metadata] (インスタンスメタデータのタグを許可) フィールドをチェックします。値が [Enabled] (有効) の場合、インスタンスメタデータのタグを使用できます。値が [Disabled] (無効) の場合、インスタンスメタデータのタグを使用できません。

AWS CLI を使用してインスタンスメタデータのタグへのアクセスが許可されるか確認するには

describe-instances コマンドを使用して、インスタンス ID を指定します。

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0
```

次の例では、出力はスペースの都合上、表示されていません。"InstanceMetadataTags" パラメータは、インスタンスメタデータのタグを許可するかどうかを示します。値が enabled の場合、インスタンスメタデータのタグを使用できます。値が disabled の場合、インスタンスメタデータのタグを使用できません。

```
{
    "Reservations": [
        {
            "Groups": [],
            "Instances": [
                {
                    "AmiLaunchIndex": 0,
                    "ImageId": "ami-0abcdef1234567890",
                    "InstanceId": "i-1234567890abcdef0",
   ...

   "MetadataOptions": {
   "State": "applied",
   "HttpTokens": "optional",
   "HttpPutResponseHopLimit": 1,
   "HttpEndpoint": "enabled",
   "HttpProtocolIpv6": "disabled",
   "InstanceMetadataTags": "enabled"
   },
   ...
```

インスタンスメタデータからタグを取得する

インスタンスメタデータでインスタンスタグが許可されている場合、tags/instance カテゴリはインスタンスのメタデータからアクセスできます。インスタンスメタデータからタグを取得する方法の例については、インスタンスのインスタンスタグを取得するを参照してください。

CloudFormation を使用したリソースへのタグの追加

Amazon EC2 リソースタイプでは、Tags または TagSpecifications プロパティを使用してタグを指定します。

次の例では、**Stack=Production** プロパティを使用して AWS::EC2::Instance にタグ Tags を追加します。

Example 例: YAML のタグ

```
Tags:
  - Key: "Stack"
    Value: "Production"
```

Example 例: JSON のタグ

```
"Tags": [
    {
        "Key": "Stack",
        "Value": "Production"
    }
]
```

次の例では、**Stack=Production** プロパティを使用して AWS::EC2::LaunchTemplate LaunchTemplateData にタグ TagSpecifications を追加します。

Example 例: YAML のタグ仕様

```
TagSpecifications:
  - ResourceType: "instance"
    Tags:
    - Key: "Stack"
      Value: "Production"
```

Example 例: JSON のタグ仕様

```
"TagSpecifications": [
    {
        "ResourceType": "instance",
        "Tags": [
            {
                "Key": "Stack",
                "Value": "Production"
            }
        ]
    }
]
```

Amazon EC2 の Service Quotas

Amazon EC2 にはさまざまなリソースが用意されており、それらを利用することができます。リソースにはイメージ、インスタンス、ボリューム、スナップショットなどがあります。AWS アカウント を作成するときに、リージョンごとにこれらのリソースに対してデフォルトのクォータ (制限とも呼ばれます) が設定されます。例えば、リージョンで起動できるインスタンスの最大数があります。したがって、米国西部 (オレゴン) リージョンでインスタンスを起動するときなどは、リクエストによってインスタンスの使用数がそのリージョンでのインスタンスの最大数を超えないようにしてください。

Service Quotas コンソールは、AWS サービスのクォータを表示および管理したり、使用する多くのリソースのクォータの引き上げをリクエストしたりできる一元的な場所です。提供されるクォータとに関する情報を利用して、AWS インフラストラクチャを管理します。クォータの引き上げに対するリクエストは、クォータの引き上げが必要となる前に計画してください。

詳細については、「Amazon Web Services 全般のリファレンス」の「Amazon EC2 エンドポイントとクォータ」と「Amazon EBS エンドポイントとクォータ」を参照してください。

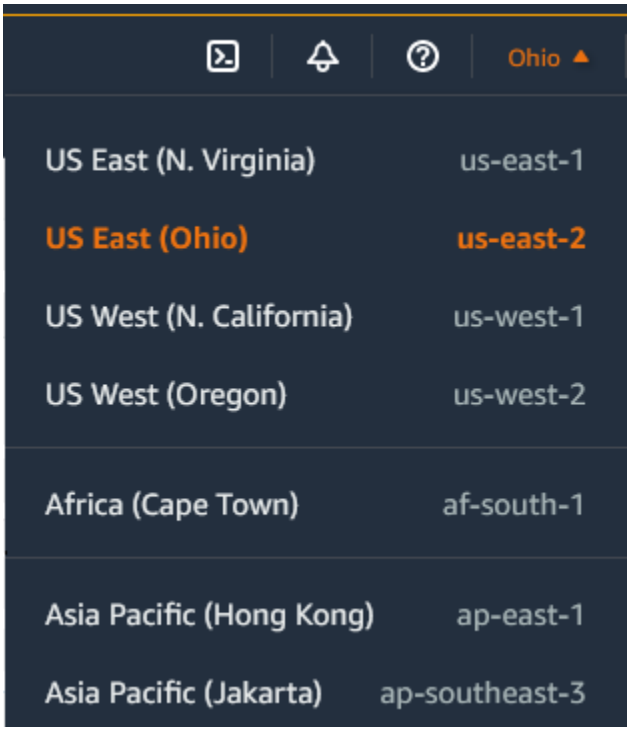
現在の制限を表示するには

の Service Quotas コンソールを使用して、各リージョンのクォータを表示できます。

Service Quotas コンソールを使用して現在のクォータを表示するには

1. Service Quotas コンソール (https://console.aws.amazon.com/servicequotas/home/services/ec2/quotas/) を開きます。

2. 画面上部のナビゲーションバーで、リージョンを選択します。



3. リソースネームごとにリストを絞り込むには、フィルターフィールドを使用します。例えば、**On-Demand** と入力してオンデマンドインスタンスのクォータを確認してください。
4. 詳細情報を表示するには、クォータ名を選択してクォータの詳細ページを開きます。

引き上げのリクエスト

各リージョンに対して、クォータの増加をリクエストすることができます。

増加をリクエストするには、Service Quotas コンソールを使用してください。

1. Service Quotas コンソール (https://console.aws.amazon.com/servicequotas/home/services/ec2/quotas/) を開きます。
2. 画面上部のナビゲーションバーで、リージョンを選択します。
3. リソースネームごとにリストを絞り込むには、フィルターフィールドを使用します。例えば、**On-Demand** と入力してオンデマンドインスタンスのクォータを確認してください。
4. クォータが調整可能な場合は、クォータを選択し、[クォータの引き上げをリクエスト] を選択します。
5. [クォータ値を変更] に、新しいクォータ値を入力します。
6. [リクエスト] を選択します。

7. コンソールで保留中または最近解決された要求を表示するには、ナビゲーションペインから [ダッシュボード] を選択します。保留中のリクエストの場合は、リクエストのステータスを選択してリクエストの受信をオープンします。リクエストの初期ステータスは [Pending] (保留中) です。ステータスが [要求されたクォータ] に変わると、AWS Support とケース番号が表示されます。リクエストのチケットを開くには、ケース番号を選択します。

AWS CLI や SDK を使用してクォータの増加をリクエストする方法などの詳細については、「Service Quotas ユーザーガイド」の「クォータ増加のリクエスト」を参照してください。

ポート 25 を使用した E メール送信の制限

すべてのインスタンスで、Amazon EC2 はデフォルトでポート 25 を介したパブリック IP アドレスへのアウトバウンドトラフィックを制限します。この制限の解除をリクエストできます。詳細については、AWS re:Post の「Amazon EC2 インスタンスまたは AWS Lambda 関数からポート 25 の制限を解除する方法」を参照してください。

> **Note**
>
> この制限は、ポート 25 経由で次の宛先に送信されるアウトバウンドトラフィックには適用されません。
>
> - 発信元のネットワークインターフェイスが存在する VPC のプライマリ CIDR ブロック内の IP アドレス。
> - RFC 1918、RFC 6598、および RFC 4193 で定義されている CIDR 内の IP アドレス。

Amazon EC2 使用状況レポート

AWS は、EC2 インスタンスのコストと使用状況、およびリザーブドインスタンスの使用状況を分析できる AWS Cost Explorer という無料レポートツールを提供します。過去 12 か月までのデータを表示でき、また次の 3 か月間にどのくらい使用する可能性があるかを予測します。時間の経過に伴う AWS リソースの使用量パターンを確認して、さらに照会が必要な分野を識別し、コストの把握に役立つ傾向を確認するには、Cost Explorer を使用します。データの時間範囲を指定したり、時間データを日または月ごとに表示することもできます。

以下に、Cost Explorer を使用すると回答を得られるいくつかの質問の例を示します。

- 各インスタンスタイプのインスタンスには、どのくらいのコストがかかりますか?

- 特定の部門でどのくらいのインスタンス時間が使用されていますか?
- 自分のインスタンスの使用は、複数のアベイラビリティーゾーンにわたってどのように分散されていますか?
- 自分のインスタンスの使用は、AWS アカウント にわたってどのように分散されていますか?
- 自分は リザーブドインスタンス をどのくらい適切に使用しているのでしょうか?
- リザーブドインスタンス によってコストを削減できているのでしょうか?

レポートの保存を含め、Cost Explorer でのレポートの使用の詳細については、「AWS Cost Management ユーザーガイド」の「AWS Cost Explorer によるコストの分析」を参照してください。

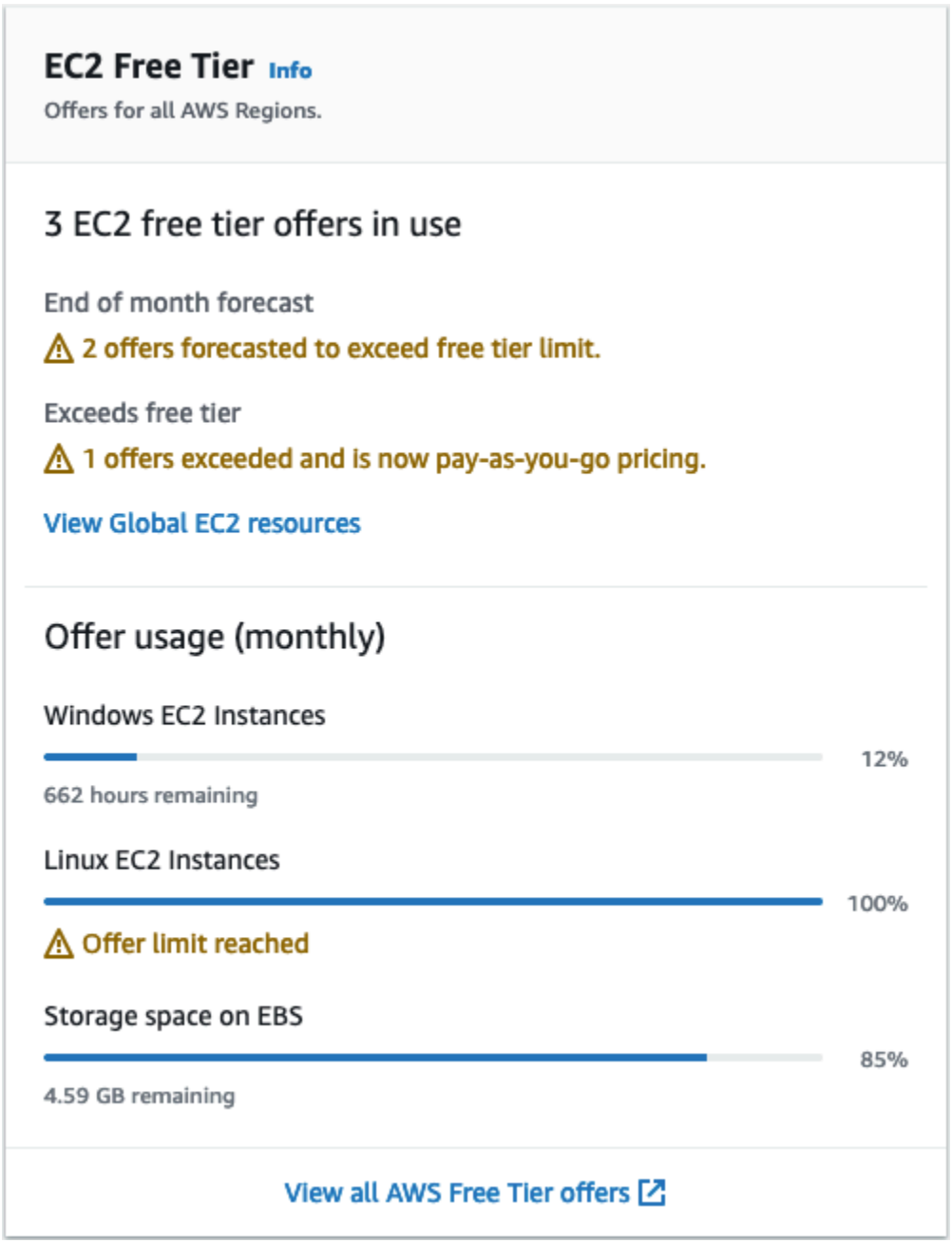
無料利用枠の使用状況の追跡

AWS のお客様になってから 12 か月未満で、AWS 無料利用枠 使用制限の範囲内であれば、Amazon EC2 は無料で使用できます。予期せぬ請求を避けるには、無料利用枠の使用状況を追跡することが重要です。無料利用枠の制限を超える場合、標準の従量課金制料金が発生します。

> **Note**
>
> 12 か月以上ご利用いただいている AWS カスタマーの場合は、無料利用枠の利用資格がなくなり、以下の手順で説明する [EC2 無料利用枠] ボックスも表示されなくなります。

無料利用枠の使用状況を追跡するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[EC2 ダッシュボード] を選択します。
3. [EC2 無料利用枠] ボックス (右上) を参照してください。



4. [EC2 無料利用枠] ボックスで、次のように無料利用枠の使用状況を確認します。

   - [使用中の EC2 無料利用枠のオファー] の警告に注意してください。
     - [月末の予測] — 現在の使用パターンを継続する場合、今月は料金が発生することを警告しています。
     - [無料利用枠超過] — 無料利用枠の制限を超え、すでに料金が発生していることを警告しています。

- [オファーの使用量 (月額)] で、Linux インスタンス、Windows インスタンス、EBS ストレージの使用状況を書き留めてください。このパーセンテージは、今月使用した無料利用枠の制限を示します。100% の場合、それ以降の使用には料金が発生します。

> **Note**
>
> この情報は、インスタンスを作成した後にのみ表示されます。ただし、使用状況情報はリアルタイムでは更新されず、1 日 3 回更新されます。

5. さらなる料金の発生を避けるには、現在料金が発生しているリソース、または無料利用枠の使用制限を超えた場合に料金が発生するリソースをすべて削除してください。

   - インスタンスを削除する手順については、このチュートリアルの次のステップに記載しています。
   - 料金が発生する可能性のあるリソースが他のリージョンにあるかどうかを確認するには、[EC2 無料利用枠] ボックスで [グローバル EC2 リソースを表示] を選択して [EC2 グローバルビュー] を開きます。詳細については、「Amazon EC2 Global View」を参照してください。

6. AWS 無料利用枠 にあるすべての AWS のサービス のリソース使用量を表示するには、[EC2 無料利用枠] ボックスの下部にある [すべての AWS 無料利用枠 オファーを表示] を選択します。詳細は、「AWS 料金ユーザーガイド」の「AWS 無料利用枠 を使用する」を参照してください。

EC2 インスタンスのトラブルシューティング

次のドキュメントは、インスタンスに関する問題を解決するために役立ちます。

コンテンツ

Windows インスタンスの詳細なヘルプについては、Windows インスタンスの Amazon EC2 ユーザーガイド の「Windows インスタンスのトラブルシューティング」を参照してください。

インスタンスの起動に関する問題のトラブルシューティング

以下の問題が発生すると、インスタンスを起動できなくなります。

起動に関する問題

無効なデバイス名

説明

新しいインスタンスを起動しようとすると、`Invalid device name` *`device_name`* エラーが発生します。

原因

インスタンスを起動しようとしたときにこのエラーが発生した場合、リクエストの 1 つ以上のボリュームのために指定されたデバイス名に無効なデバイス名が含まれています。エラーの原因として以下が考えられます。

- デバイス名は、選択した AMI によって使用されている可能性があります。
- デバイス名は、ルートボリューム用に予約されている可能性があります。
- デバイス名は、リクエスト内の別のボリュームのために使用される可能性があります。
- デバイス名は、オペレーティングシステム向けに有効でない可能性があります。

解決策

問題を解決するには:

- デバイス名が、選択した AMI で使用されていないことを確認します。次のコマンドを実行して、AMI によって使用されるデバイス名を表示します。

```
$ aws ec2 describe-images --image-id ami_id --query
 'Images[*].BlockDeviceMappings[].DeviceName'
```

- ルートボリューム用に予約されているデバイス名を使用していないことを確認します。詳細については、「使用できるデバイス名」を参照してください。
- リクエストで指定されている各ボリュームのデバイス名が一意であることを確認します。
- 指定したデバイス名が正しい形式となっていることを確認します。詳細については、「使用できるデバイス名」を参照してください。

インスタンス制限の超過

説明

新しいインスタンスを起動するか、停止したインスタンスを再起動しようとすると、InstanceLimitExceeded エラーが発生する。

原因

新しいインスタンスを起動するか、停止したインスタンスを再起動しようとすると InstanceLimitExceeded エラーが発生する場合、リージョンで起動できるインスタンス数の制限に達しています。AWS アカウントを作成するときに、リージョンごとに、実行できるインスタンスの数についてデフォルトの制限が設定されます。

解決策

インスタンス制限の引き上げは、リージョンごとにリクエストできます。詳細については、「Amazon EC2 の Service Quotas」を参照してください。

インスタンス容量の不足

説明

新しいインスタンスを起動するか、停止したインスタンスを再起動しようとすると、InsufficientInstanceCapacity エラーが発生する。

原因

インスタンスを起動したり、停止したインスタンスを再起動したりする際にこのエラーが発生する場合、現在 AWS にはリクエストに対応するために必要とされる十分なオンデマンドキャパシティーがありません。

解決策

この問題を解決するには、以下の手順を実行します。

- 数分間待ってからリクエストを再度送信します。容量は頻繁に変化します。
- インスタンス数を減らして新しいリクエストを送信します。たとえば、15 インスタンスを起動する 1 つのリクエストを行っている場合、代わりに 5 つのインスタンスに対する 3 つのリクエストを作成するか、1 つのインスタンスに対する 15 のリクエストを作成してみてください。

- インスタンスを起動する場合は、アベイラビリティーゾーンを指定しないで新しいリクエストを送信します。
- インスタンスを起動する場合は、別のインスタンスタイプを使用して新しいリクエストを送信します (これは後でサイズを変更できます)。詳細については、「インスタンスタイプを変更する」を参照してください。
- クラスタープレイスメントグループにインスタンスを起動すると、容量不足エラーが発生する場合があります。詳細については、「プレイスメントグループの操作」を参照してください。

リクエストされた設定は現在サポートされていません。サポートされている設定については、ドキュメントを参照してください。

説明

インスタンス設定がサポートされていないため、新しいインスタンスを起動しようとすると、Unsupported エラーが表示されます。

原因

エラーメッセージには、詳細が記載されています。例えば、インスタンスタイプまたはインスタンス購入オプションは、指定されたリージョンまたはアベイラビリティーゾーンではサポートされていない可能性があります。

解決策

別のインスタンス設定を試してください。要件を満たすインスタンスタイプを検索するには、「Amazon EC2 インスタンスタイプの検索」を参照してください。

インスタンスがすぐに終了する

説明

インスタンスは pending 状態から terminated 状態に移行します。

原因

インスタンスがすぐに終了する理由を次にいくつか示します。

- EBS ボリュームの制限を超えた。詳細については、「インスタンスボリューム数の制限」を参照してください。

- EBS スナップショットが破損している。
- ルート EBS ボリュームは暗号化されていて、復号用の KMS キー にアクセスする権限がない。
- AMI のブロックデバイスマッピングで指定されたスナップショットは暗号化されていて、復号するための KMS キー へのアクセス権限がないか、復元されたボリュームを暗号化するための KMS キー へのアクセス権限がありません。
- インスタンスを起動するために使用した Instance Store-Backed AMI で、必要な部分 (image.part.xx ファイル)。

詳細については、次のいずれかの方法を使用して、削除された理由を確認します。

Amazon EC2 コンソールを使用して、削除された理由を確認するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. 最初のタブで、[State transition reason (状態遷移の理由)] の横にある理由を見つけます。

AWS Command Line Interface を使用して、削除された理由を確認するには

1. describe-instances コマンドを使用して、インスタンス ID を指定します。

   ```
   aws ec2 describe-instances --instance-id instance_id
   ```

2. コマンドによって返された JSON レスポンスで、`StateReason` レスポンス要素の値を確認します。

   次のコードブロックは `StateReason` レスポンス要素の例を示しています。

   ```
   "StateReason": {
     "Message": "Client.VolumeLimitExceeded: Volume limit exceeded",
     "Code": "Server.InternalError"
   },
   ```

AWS CloudTrail を使用して、削除された理由を確認するには

詳細については、AWS CloudTrail ユーザーガイドの「CloudTrail イベント履歴でのイベントの表示」を参照してください。

解決策

削除の理由に応じて、次のアクションの 1 つを実行します。

- **`Client.VolumeLimitExceeded: Volume limit exceeded`** — 未使用のボリュームを削除します。ボリューム制限を引き上げるリクエストを送信できます。
- **`Client.InternalError: Client error on launch`** - ボリュームの復号と暗号化に使用する AWS KMS keys への必要なアクセス権限があることを確認します。詳細については、AWS Key Management Service デベロッパーガイドの「AWS KMS でのキーポリシーの使用」を参照してください。

アクセス権限の不足

説明

新しいインスタンスを起動しようとすると、`"errorMessage": "You are not authorized to perform this operation."` エラーが発生し、起動が失敗します。

原因

インスタンスを起動しようとしたときにこのエラーが表示される場合は、インスタンスを起動するために必要な IAM 権限が不足している可能性があります。

不足している可能性のある権限には次のものがあります。

- `ec2:RunInstances`
- `iam:PassRole`

その他のアクセス許可が不足している可能性もあります。インスタンスの起動に必要な権限のリストについては、例: EC2 起動インスタンスウィザードの使用 および インスタンスの起動 (RunInstances) の下にある IAM ポリシーの例を参照してください。

解決策

問題を解決するには:

- IAM ユーザーとしてリクエストを発行する場合は、以下の条件が満たされていることを確認します。
  - `ec2:RunInstances` でリソースがワイルドカード (*) で定義されている

  - `iam:PassRole` でロールの ARN (`arn:aws:iam::999999999999:role/ExampleRoleName` など) に一致するリソースが定義されている
- 上記の権限がない場合は、IAM ロールまたはユーザーに関連付けられた IAM ポリシーを編集して、不足している必要な権限を追加してください。

問題が解決されず、起動失敗エラーが引き続き表示される場合は、エラーに含まれる認証失敗メッセージをデコードすることができます。デコードされたメッセージには、IAM ポリシーにない権限が含まれています。詳細については、「EC2 インスタンスの起動中に 「UnauthorizedOperation」というエラーを受け取った後、認証失敗のメッセージをデコードするには」を参照してください。

インスタンスへの接続に関するトラブルシューティング

以下の情報と一般的なエラーは、Linux インスタンスへの接続に関するトラブルシューティングに役立ちます。

Windows インスタンスの接続のトラブルシューティングについては、「Windows インスタンスの Amazon EC2 ユーザーガイド」の「Windows インスタンスのトラブルシューティング」を参照してください。

接続の問題とエラー

- 接続の問題の一般的な原因
- インスタンスへの接続エラー: 接続タイムアウト
- エラー: キーを読み込めません..。期待: 任意のプライベートキー
- エラー: ユーザーキーがサーバーによって認識されない
- エラー: アクセス許可が拒否されたか、[インスタンス] ポート 22 によって接続が閉じられました。
- エラー: Unprotected Private Key File (保護されていないプライベートキーファイル)
- エラー: プライベートキーの先頭は「-----BEGIN RSA PRIVATE KEY-----」、末尾は「-----END RSA PRIVATE KEY-----」にする必要があります
- エラー: Server refused our key または No supported authentication methods available (サーバーはキーを拒否しましたまたは利用可能なサポートされる認証方法はありません)
- インスタンスに対して ping を実行できない
- エラー: サーバーによる予期しないネットワーク接続の閉鎖
- エラー: EC2 Instance Connect のホストキーの検証に失敗しました

- EC2 Instance·Connect を使用して Unbntu インスタンスに接続できない
- プライベートキーを紛失しました。Linux インスタンスに接続するにはどうすればよいですか?

接続の問題の一般的な原因

次のタスクを正確に実行したことを確認して、インスタンス接続の問題のトラブルシューティングを開始することをお勧めします。

インスタンスのユーザー名を確認する

インスタンスに接続するには、ユーザーアカウントのユーザー名、またはインスタンスの起動に使用した AMI のデフォルトのユーザー名を使用します。

- ユーザーアカウントのユーザー名を取得します。

  ユーザーアカウントの作成方法については、「Linux インスタンスのユーザーアカウントを管理する」を参照してください。
- インスタンスの起動に使用した AMI のデフォルトのユーザー名を取得します。

インスタンスの起動に使用される AMI	デフォルトユーザー名
AL2023 Amazon Linux 2 Amazon Linux	`ec2-user`
CentOS	`centos` または `ec2-user`
Debian	`admin`
Fedora	`fedora`、または `ec2-user`
RHEL	`ec2-user`、または `root`
SUSE	`ec2-user`、または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`

インスタンスの起動に使用される AMI	デフォルトユーザー名
Bitnami	`bitnami`
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

セキュリティグループルールでトラフィックが許可されていることを確認する

インスタンスに関連付けられているセキュリティグループで、IP アドレスからの受信 SSH トラフィックが許可されていることを確認します。VPC のデフォルトのセキュリティグループでは、着信 SSH トラフィックはデフォルトでは許可されません。インスタンス起動ウィザードで作成されたセキュリティグループでは、デフォルトで SSH トラフィックが許可されます。Linux インスタンスにインバウンド SSH トラフィックのルールを追加する手順については、「Linux インスタンス用のインバウンドトラフィックの承認」を参照してください。確認する手順については、「インスタンスへの接続エラー: 接続タイムアウト」を参照してください。

インスタンスが準備ができていることを確認する

インスタンスを起動してから接続できるようになるまでには、数分かかる場合があります。インスタンスをチェックして、それが実行中であり、ステータスチェックに合格していることを確認します。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. 以下について確認してください。

   a. [インスタンスの状態] 列で、インスタンスの状態が `running` であることを確認します。
   b. [ステータスチェック] 列で、インスタンスが 2 つのステータスチェックに合格したことを確認します。

接続の前提条件をすべて満たしていることを確認する

接続に必要な情報がすべて揃っていることを確認します。詳細については、「Linux インスタンスへの接続」を参照してください。

SSH、EC2 Instance Connect、OpenSSH、PuTTY などの接続タイプに固有の前提条件については、以下のオプションを参照してください。

Linux または macOS X

ローカルコンピュータのオペレーティングシステムが Linux または macOS X の場合、次の接続オプション固有の前提条件を確認してください。

- SSH クライアント
- EC2 Instance Connect
- AWS Systems Manager Session Manager

Windows

ローカルコンピュータのオペレーティングシステムが Windows の場合、次の接続オプション固有の前提条件を確認してください。

- OpenSSH
- PuTTY
- AWS Systems Manager Session Manager
- Windows Subsystem for Linux

インスタンスへの接続エラー: 接続タイムアウト

インスタンスへ接続しようとして、エラーメッセージ `Network error: Connection timed out` または `Error connecting to [instance], reason: -> Connection timed out: connect` が表示される場合、次を実行します。

セキュリティグループルールを調べます。

ローカルコンピュータの適切なポートのパブリック IPv4 アドレスからのインバウンドトラフィックがセキュリティグループルールで許可されている必要があります。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. コンソールページの下部にある [セキュリティ] タブの [インバウンドルール] で、選択したインスタンスで有効なルールのリストを確認します。

   - Linux インスタンスの場合: ローカルコンピュータからポート 22 (SSH) へのトラフィックを許可するルールがあることを確認します。
   - Windows インスタンスの場合: ローカルコンピュータからポート 3389 (RDP) へのトラフィックを許可するルールがあることを確認します。

セキュリティグループに、ローカルコンピュータからのインバウンドトラフィックを許可するルールがない場合は、セキュリティグループにルールを追加します。詳細については、「Linux インスタンス用のインバウンドトラフィックの承認」を参照してください。

4. インバウンドトラフィックを許可するルールについては、[Source] (ソース) フィールドを確認します。値が単一の IP アドレスで、IP アドレスが静的でない場合、コンピュータを再起動するたびに新しい IP アドレスが割り当てられます。これにより、ルールにはコンピュータの IP アドレストラフィックが含まれなくなります。コンピュータが企業ネットワークにある場合またはインターネットサービスプロバイダー (ISP) を通じて接続する場合は、コンピュータの IP アドレスが静的ではない可能性があります。つまり、コンピュータの IP アドレスは動的で、コンピュータを再起動するたびに変化します。セキュリティグループルールで、ローカルコンピュータからのインバウンドトラフィックが許可されるようにするには、[Source] (ソース) に単一の IP アドレスを指定するのではなく、クライアントコンピュータで使用されている IP アドレスの範囲を指定します。

   セキュリティグループのルールの詳細については、Amazon VPCユーザーガイドの「セキュリティグループのルール」を参照してください。

サブネットのルートテーブルを確認します。

VPC の外部あてのすべてのトラフィックを VPC のインターネットゲートウェイに送信するには、ルートが必要です。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. [ネットワーキング] タブで、VPC ID とサブネット ID の値を書き留めます。
4. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
5. ナビゲーションペインで、[Internet Gateways] を選択します。ご使用の VPC にアタッチされているインターネットゲートウェイがあることを確認します。または、[インターネットゲートウェイの作成] を選択し、インターネットゲートウェイの名前を入力して [インターネットゲートウェイの作成] を選択します。次に、作成したインターネットゲートウェイで、[アクション]、[VPC にアタッチ]、[VPC] の順に選択し、[インターネットゲートウェイのアタッチ] をクリックして VPC にアタッチします。
6. ナビゲーションペインで [Subnets] を選択し、サブネットを選択します。
7. [ルートテーブル] タブで、送信先として `0.0.0.0/0`、ターゲットとして VPC のインターネットゲートウェイが指定されたルートがあることを確認します。IPv6 アドレスを使用してインス

タンスに接続する場合は、インターネットゲートウェイを指しているすべての IPv6 トラフィック (`::/0`) 用のルートがあることを確認します。それ以外の場合は、以下の作業を行います。

a. ルートテーブルの ID (rtb-xxxxxxxx) を選択して、ルートテーブルに移動します。

b. [Routes] タブで、[Edit routes] を選択します。[Add route] を選択して、`0.0.0.0/0` を送信先として追加し、インターネットゲートウェイをターゲットとして使用します。IPv6 の場合は、[Add route] を選択して、`::/0` を送信先として追加し、インターネットゲートウェイをターゲットとして使用します。

c. [Save routes] を選択します。

サブネットのネットワークアクセスコントロールリスト (ACL) を確認します。

ネットワーク ACL では、ローカル IP アドレスからのインバウンドトラフィックが、ポート 22 (Linux インスタンスの場合)、またはポート 3389 (Windows インスタンスの場合) で許可されている必要があります。また、一時ポート (1024-65535) へのアウトバウンドトラフィックも許可する必要があります。

1. Amazon VPC コンソール (https://console.aws.amazon.com/vpc/) を開きます。
2. ナビゲーションペインで、[Subnets (サブネット)] を選択します。
3. サブネットを選択する
4. [ネットワーク ACL] タブの [インバウンドルール] で、必要なポートでコンピュータからのインバウンドトラフィックを許可しているルールがあることを確認します。それが見つからない場合は、コンピュータへのトラフィックをブロックしているルールを削除または変更します。
5. [アウトバウンドルール] で、一時ポートでコンピュータへのトラフィックを許可しているルールがあることを確認します。存在しない場合は、コンピュータへのトラフィックをブロックしているルールを削除または変更します。

コンピュータが社内ネットワークに接続されている場合

社内ファイアウォールで、ご使用のコンピュータのインバウンドおよびアウトバウンドのトラフィックがポート 22 (Linux インスタンスの場合) またはポート 3389 (Windows インスタンスの場合) で許可されているかどうか、ネットワーク管理者に問い合わせてください。

ご使用のコンピュータにファイアウォールが設定されている場合、そのファイアウォールでコンピュータのインバウンドおよびアウトバウンドのトラフィックがポート 22 (Linux インスタンスの場合) またはポート 3389 (Windows インスタンスの場合) で許可されているかどうか確認します。

インスタンスにパブリック IPv4 アドレスがあることを確認します。

そうでない場合は、Elastic IP アドレスをインスタンスに関連付けることができます。詳細については、「Elastic IP アドレス」を参照してください。

サーバーが過負荷になっている可能性のあるインスタンスの CPU 負荷を確認します。

AWS は自動的に Amazon CloudWatch メトリクスおよびインスタンスステータスなどのデータを提供します。これらを使用することでインスタンスの CPU 負荷を確認でき、必要に応じて負荷の処理方法を調整できます。詳細については、「CloudWatch を使用したインスタンスのモニタリング」を参照してください。

- 負荷が変化する場合、Auto Scaling および Elastic Load Balancing を使用して、インスタンスの増減を自動的に縮小・拡張できます。
- 負荷が常に増加している場合、大きなインスタンスタイプに移動できます。詳細については、「インスタンスタイプを変更する」を参照してください。

IPv6 アドレスを使用してインスタンスに接続するには、以下のことを確認します。

- サブネットはインターネットゲートウェイへの IPv6 トラフィック (`::/0`) のルートを持つルートテーブルに関連付けられている必要があります。
- セキュリティグループルールでは、適切なポート (Linux の場合は 22、Windows の場合は 3389) のローカル IPv6 アドレスからの着信トラフィックを許可する必要があります。
- ネットワーク ACL ルールでは、インバウンドおよびアウトバウンドの IPv6 トラフィックを許可する必要があります。
- 古い AMI からインスタンスを起動した場合、DHCPv6 用に設定されていない可能性があります (IPv6 アドレスはネットワークインターフェイスでは自動的に認識されません)。詳細については、「Amazon VPC ユーザーガイド」の「インスタンスに IPv6 を設定する」を参照してください。
- ローカルコンピュータに IPv6 アドレスがあり、IPv6 を使用するように設定されている必要があります。

エラー: キーを読み込めません...。期待: 任意のプライベートキー

インスタンスに接続しようとして、エラーメッセージ、`unable to load key ... Expecting: ANY PRIVATE KEY` が表示される場合、プライベートキーが保存されているファイルが正しく設定されていません。プライベートキーファイルが `.pem` で終わる場合でも、正しく設定されていない可

能性があります。プライベートキーファイルが正しく設定されていない原因として考えられるのは、証明書がないことです。

プライベートキーファイルが正しく設定されていない場合は、以下の手順に従って エラーを解決する

1. 新しいキーペアを作成します。詳細については、「Amazon EC2 を使用してキーペアを作成する」を参照してください。

   > **Note**
   >
   > サードパーティー製のツールを使用して、新しいキーペアを作成することもできます。詳細については、「サードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートする」を参照してください。

2. 新しいキーペアをインスタンスに追加します。詳細については、「プライベートキーを紛失しました。Linux インスタンスに接続するにはどうすればよいですか?」を参照してください。
3. 新しいキーペアを使用してインスタンスに接続します。

エラー: ユーザーキーがサーバーによって認識されない

SSH を使用してインスタンスに接続している場合

- ssh -vvv を使用して、接続中に 3 倍詳細デバッグ情報を取得します。

```
ssh -vvv -i path/key-pair-name.pem instance-user-
name@ec2-203-0-113-25.compute-1.amazonaws.com
```

次のサンプル出力は、サーバーが認識しないキーを使用してインスタンスに接続しようとした場合に表示される可能性があります。

```
open/ANT/myusername/.ssh/known_hosts).
debug2: bits set: 504/1024
debug1: ssh_rsa_verify: signature correct
debug2: kex_derive_keys
debug2: set_newkeys: mode 1
debug1: SSH2_MSG_NEWKEYS sent
debug1: expecting SSH2_MSG_NEWKEYS
debug2: set_newkeys: mode 0
debug1: SSH2_MSG_NEWKEYS received
```

```
debug1: Roaming not allowed by server
debug1: SSH2_MSG_SERVICE_REQUEST sent
debug2: service_accept: ssh-userauth
debug1: SSH2_MSG_SERVICE_ACCEPT received
debug2: key: boguspem.pem ((nil))
debug1: Authentications that can continue: publickey
debug3: start over, passed a different list publickey
debug3: preferred gssapi-keyex,gssapi-with-mic,publickey,keyboard-
interactive,password
debug3: authmethod_lookup publickey
debug3: remaining preferred: keyboard-interactive,password
debug3: authmethod_is_enabled publickey
debug1: Next authentication method: publickey
debug1: Trying private key: boguspem.pem
debug1: read PEM private key done: type RSA
debug3: sign_and_send_pubkey: RSA 9c:4c:bc:0c:d0:5c:c7:92:6c:8e:9b:16:e4:43:d8:b2
debug2: we sent a publickey packet, wait for reply
debug1: Authentications that can continue: publickey
debug2: we did not send a packet, disable method
debug1: No more authentication methods to try.
Permission denied (publickey).
```

PuTTY を使用してインスタンスに接続している場合

- 秘密キー (.pem) ファイルが PuTTY によって認識される形式 (.ppk) に変換されていることを確認します。プライベートキーの変換の詳細については、「PuTTY を使用して Windows から Linux インスタンスに接続する」を参照してください。

> **Note**
>
> PuTTYgen でプライベートキーファイルをロードし、[Generate] ではなく [Save Private Key] を選択します。

- AMI 用の適切なユーザー名で接続していることを確認します。[PuTTY Configuration] ウィンドウの [Host name] ボックスにユーザー名を入力します。

インスタンスの起動に使用される AMI	デフォルトユーザー名
AL2023	`ec2-user`

インスタンスの起動に使用される AMI	デフォルトユーザー名
Amazon Linux 2 Amazon Linux	
CentOS	centos または ec2-user
Debian	admin
Fedora	fedora、または ec2-user
RHEL	ec2-user、または root
SUSE	ec2-user、または root
Ubuntu	ubuntu
Oracle	ec2-user
Bitnami	bitnami
Rocky Linux	rocky
その他	AMI プロバイダーに確認してください。

- 適切なポートへのインバウンドトラフィックを許可しているインバウンドセキュリティグループがあることを確認します。詳細については、「インスタンスへのネットワークアクセスの許可」を参照してください。

エラー: アクセス許可が拒否されたか、[インスタンス] ポート 22 によって接続が閉じられました。

SSH を使用してインスタンスに接続し、Host key not found in [directory]、Permission denied (publickey)、Authentication failed, permission denied、または Connection closed by [instance] port 22 のいずれかのエラーが発生した場合は、AMI 用の適切なユーザー名で接続しており、かつ、インスタンス用の適切なプライベートキー (.pem) ファイル) を指定していることを確認します。

適切なユーザー名は以下のとおりです。

インスタンスの起動に使用される AMI	デフォルトユーザー名
AL2023 Amazon Linux 2 Amazon Linux	`ec2-user`
CentOS	`centos` または `ec2-user`
Debian	`admin`
Fedora	`fedora`、または `ec2-user`
RHEL	`ec2-user`、または `root`
SUSE	`ec2-user`、または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

例えば、SSH クライアントを使用して Amazon Linux インスタンスに接続するには、次のコマンドを使用します。

```
ssh -i /path/key-pair-name.pem instance-user-
name@ec2-203-0-113-25.compute-1.amazonaws.com
```

使用しているプライベートキーファイルが、インスタンスの起動時に選択したキーペアに対応していることを確認します。

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. [詳細] タブの [インスタンスの詳細] で、[キーペア名] の値を確認します。

4. インスタンスを起動したときにキーペアを指定しなかった場合は、キーペアを確実に指定するために、インスタンスを終了してから新しいインスタンスを起動します。それまで使用していたインスタンスで、キーペアに対する .pem ファイルがもう存在しない場合は、そのキーペアを新しいキーペアで置き換えることができます。詳細については、「プライベートキーを紛失しました。Linux インスタンスに接続するにはどうすればよいですか?」を参照してください。

独自のキーペアを生成した場合は、キージェネレータが RSA キーを作成するように設定されていることを確認します。DSA キーは受け入れられません。

Permission denied (publickey) エラーが表示され、上のいずれも当てはまらない場合 (例えば、以前は接続できていたなど)、インスタンスのホームディレクトリのアクセス権限が変更された可能性があります。/home/*instance-user-name*/.ssh/authorized_keys のアクセス権限は、所有者のみに制限する必要があります。

インスタンスのアクセス権限を検証するには

1. インスタンスを停止し、ルートボリュームをデタッチします。詳細については、「インスタンスの停止と起動」を参照してください。
2. 同じアベイラビリティーゾーンにある一時インスタンスを現在のインスタンスとして起動し (現在のインスタンスに使用したのと同様または同じ AMI を使用)、ルートボリュームを一時インスタンスにアタッチします。
3. 一時インスタンスに接続してマウントポイントを作成し、アタッチしたボリュームをマウントします。
4. 一時インスタンスから、アタッチされたボリュームの /home/*instance-user-name*/ ディレクトリのアクセス権限をチェックします。必要に応じて、次のようにアクセス権限を調整します。

```
[ec2-user ~]$ chmod 600 mount_point/home/instance-user-name/.ssh/authorized_keys
```

```
[ec2-user ~]$ chmod 700 mount_point/home/instance-user-name/.ssh
```

```
[ec2-user ~]$ chmod 700 mount_point/home/instance-user-name
```

5. ボリュームをアンマウントして一時インスタンスからデタッチし、元のインスタンスに再アタッチします。ルートボリュームに適切なデバイス名を指定したことを確認します (/dev/xvda など)。

6. インスタンスを起動します。一時インスタンスが必要なくなった場合は、終了できます。

エラー: Unprotected Private Key File (保護されていないプライベートキーファイル)

プライベートキーファイルはその他のすべてのユーザーの読み取りおよび書き込み操作から保護されている必要があります。プライベートキーがお客様以外のユーザーによって読み取りまたは書き込みできる場合、SSH はキーを無視し、次の警告メッセージが表示されます。

```
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@         WARNING: UNPROTECTED PRIVATE KEY FILE!          @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
Permissions 0777 for '.ssh/my_private_key.pem' are too open.
It is required that your private key files are NOT accessible by others.
This private key will be ignored.
bad permissions: ignore key: .ssh/my_private_key.pem
Permission denied (publickey).
```

インスタンスへのログインを試みたときに同様のメッセージが表示された場合は、エラーメッセージの最初の行を調べて、インスタンスに対して正しいパブリックキーを使用していることを確認します。上記の例では、プライベートキー .ssh/my_private_key.pem をファイル権限 0777 とともに使用します。これにより、任意のユーザーがこのファイルの読み取りまたは書き込みを行うことができます。この権限レベルの安全性は非常に低いので、SSH はこのキーを無視します。

macOS または Linux から接続する場合にエラーを修正するには、プライベートキーファイルのパスを置き換えて次のコマンドを実行します。

```
[ec2-user ~]$ chmod 0400 .ssh/my_private_key.pem
```

Windows から接続する場合は、ローカルコンピュータに対して次の手順を実行します。

1. .pem ファイルに移動します。
2. .pem ファイルを右クリックし、[プロパティ] を選択します。
3. [セキュリティ] タブを選択します。
4. [詳細] を選択します。
5. 自分がファイルの所有者であることを確認します。そうでない場合は、所有者を自分のユーザー名に変更します。

6. [継承の無効化] および [このオブジェクトから継承されたすべてのアクセス許可を削除] を選択します。
7. [追加]、[プリンシパルの選択] を選択し、ユーザー名を入力して、[OK] をクリックします。
8. [許可エントリ] ウィンドウから、読み取りアクセス許可を付与して、[OK] をクリックします。
9. [Apply] (適用) をクリックして、すべての設定が保存されているようにします。
10. [OK] をクリックして、[セキュリティの詳細設定] ウィンドウを閉じます。
11. [OK] をクリックして、[プロパティ] ウィンドウを閉じます。
12. Windows から SSH 経由で Linux インスタンスに接続できるはずです。

Windows のコマンドプロンプトから次のコマンドを実行します。

1. コマンドプロンプトから、.pem ファイルのファイルパスの場所に移動します。
2. 明示的なアクセス許可をリセットおよび削除するには、次のコマンドを実行します。

```
icacls.exe $path /reset
```

3. 現在のユーザーに読み取りアクセス許可を付与するには、次のコマンドを実行します。

```
icacls.exe $path /GRANT:R "$($env:USERNAME):(R)"
```

4. 継承を無効にし、継承されたアクセス許可を削除するには、次のコマンドを実行します。

```
icacls.exe $path /inheritance:r
```

5. Windows から SSH 経由で Linux インスタンスに接続できるはずです。

エラー: プライベートキーの先頭は「-----BEGIN RSA PRIVATE KEY-----」、末尾は「-----END RSA PRIVATE KEY-----」にする必要があります

サードパーティーツール (例: ssh-keygen) を使用して RSA キーペアを作成すると、プライベートキーが OpenSSH キー形式で生成されます。インスタンスに接続する際、OpenSSH 形式のプライベートキーを使用してパスワードを復号すると、エラー (`Private key must begin with "-----BEGIN RSA PRIVATE KEY-----" and end with "-----END RSA PRIVATE KEY-----"`) が発生する場合があります。

エラーを解消するには、プライベートキーは PEM 形式である必要があります。PEM 形式でプライベートキーを作成するには、次のコマンドを使用します。

```
ssh-keygen -m PEM
```

エラー: Server refused our key または No supported authentication methods available (サーバーはキーを拒否しましたまたは利用可能なサポートされる認証方法はありません)

PuTTY を使用してインスタンスに接続し、[Error: Server refused our key] または [Error: No supported authentication methods available] エラーが発生した場合は、AMI の適切なユーザー名で接続していることを確認します。[PuTTY 設定] ウィンドウの [ユーザー名] にユーザー名を入力します。

適切なユーザー名は以下のとおりです。

インスタンスの起動に使用される AMI	デフォルトユーザー名
AL2023 Amazon Linux 2 Amazon Linux	`ec2-user`
CentOS	`centos` または `ec2-user`
Debian	`admin`
Fedora	`fedora`、または `ec2-user`
RHEL	`ec2-user`、または `root`
SUSE	`ec2-user`、または `root`
Ubuntu	`ubuntu`
Oracle	`ec2-user`
Bitnami	`bitnami`

インスタンスの起動に使用される AMI	デフォルトユーザー名
Rocky Linux	`rocky`
その他	AMI プロバイダーに確認してください。

次の点も確認する必要があります。

- 最新バージョンの PuTTY を使用している。詳細については、PuTTY のウェブページを参照してください。
- プライベートキー (.pem) ファイルが PuTTY によって認識される形式 (.ppk) に正しく変換されている。プライベートキーの変換の詳細については、「PuTTY を使用して Windows から Linux インスタンスに接続する」を参照してください。

インスタンスに対して ping を実行できない

`ping` コマンドは、ICMP トラフィックの一種です。インスタンスに対して Ping を実行できない場合は、インバウンドセキュリティグループのルールで、すべてのソースからの、あるいはコマンドを発行しているコンピュータまたはインスタンスからの `Echo Request` メッセージについて、ICMP トラフィックが許可されていることを確認します。

インスタンスから `ping` コマンドを発行できない場合は、アウトバウンドセキュリティグループのルールで、すべての宛先への、または Ping の対象であるホストへの `Echo Request` メッセージについて、ICMP トラフィックが許可されていることを確認します。

`Ping` コマンドは、ファイアウォールによってブロックされたり、ネットワークのレイテンシーやハードウェアの問題によってタイムアウトしたりすることもあります。詳しいトラブルシューティングについては、ローカルネットワークまたはシステム管理者に問い合わせてください。

エラー: サーバーによる予期しないネットワーク接続の閉鎖

PuTTY を使用してインスタンスに接続中に「サーバーによる予期しないネットワーク接続の閉鎖」エラーを受け取った場合、PuTTY 設定の接続ページでキープアライブを有効化して、切断を回避していることを確認してください。一部のサーバーは、指定された時間内にデータが一切受信されない場合に、クライアントを切断します。キープアライブ間の秒数を 59 秒に設定します。

キープアライブを有効後にも問題が依然として発生する場合には、PuTTY 設定の接続ページで Nagle のアルゴリズムを無効にすることを試してください。

エラー: EC2 Instance Connect のホストキーの検証に失敗しました

インスタンスホストキーをローテーションしても、新しいホストキーは AWS の信頼されたホストキーデータベースに自動的にアップロードされません。これにより、EC2 Instance Connect ブラウザベースのクライアントを使用してインスタンスに接続しようとすると、ホストキーの検証が失敗し、インスタンスに接続できなくなります。

エラーを解決するには、`eic_harvest_hostkeys` スクリプトをインスタンスで実行する必要があります。これにより、新しいホストキーが EC2 Instance Connect にアップロードされます。スクリプトは Amazon Linux 2 インスタンス上の `/opt/aws/bin/` にあり、Ubuntu インスタンスでは `/usr/share/ec2-instance-connect/` にあります。

Amazon Linux 2

Amazon Linux 2 インスタンスでホストキーの検証に失敗したエラーを解決するには

1. SSH を使用してインスタンスに接続します。

   EC2 Instance Connect CLI を使用するか、インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスの起動に使用した AMI のデフォルトユーザー名を使用して接続できます。Amazon Linux 2 の場合、デフォルトのユーザー名は `ec2-user` です。

   例えば、Amazon Linux 2 を使用してインスタンスを起動した場合、インスタンスのパブリック DNS 名は `ec2-a-b-c-d.us-west-2.compute.amazonaws.com`、キーペアは `my_ec2_private_key.pem` です。次のコマンドを使用して SSH 経由でインスタンスに接続します。

   ```
   $ ssh -i my_ec2_private_key.pem ec2-user@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。
2. 次のフォルダに移動します。

   ```
   [ec2-user ~]$ cd /opt/aws/bin/
   ```
3. インスタンスで次のコマンドを実行します。

   ```
   [ec2-user ~]$ ./eic_harvest_hostkeys
   ```

呼び出しが成功しても、出力されないことに注意してください。

これで、EC2 Instance Connect ブラウザベースのクライアントを使用してインスタンスに接続できます。

Ubuntu

Ubuntuインスタンスでホストキーの検証に失敗したエラーを解決するには

1. SSH を使用してインスタンスに接続します。

   EC2 Instance Connect CLI を使用するか、インスタンスの起動時にインスタンスに割り当てた SSH キーペアと、インスタンスの起動に使用した AMI のデフォルトユーザー名を使用して接続できます。Ubuntu の場合は、デフォルトのユーザー名は ubuntu です。

   例えば、Ubuntu を使用してインスタンスを起動した場合、インスタンスのパブリック DNS 名は ec2-a-b-c-d.us-west-2.compute.amazonaws.com、キーペアは my_ec2_private_key.pem です。次のコマンドを使用して SSH 経由でインスタンスに接続します。

   ```
   $ ssh -i my_ec2_private_key.pem ubuntu@ec2-a-b-c-d.us-
   west-2.compute.amazonaws.com
   ```

   インスタンスへの接続の詳細については、SSH を使用して Linux または macOS から Linux インスタンスに接続します。を参照してください。

2. 次のフォルダに移動します。

   ```
   [ec2-user ~]$ cd /usr/share/ec2-instance-connect/
   ```

3. インスタンスで次のコマンドを実行します。

   ```
   [ec2-user ~]$ ./eic_harvest_hostkeys
   ```

   呼び出しが成功しても、出力されないことに注意してください。

   これで、EC2 Instance Connect ブラウザベースのクライアントを使用してインスタンスに接続できます。

EC2 Instance·Connect を使用して Unbntu インスタンスに接続できない

EC2 Instance Connect を使用して Ubuntu インスタンスへの接続を試行中にエラーが発生した場合、次の情報を使用して問題の解決を試みることができます。

考えられる原因

インスタンス上のec2-instance-connectパッケージは最新バージョンではありません。

解決策

次のように、インスタンス上のec2-instance-connectパッケージを最新バージョンにアップデートします。

1. EC2 Instance Connect 以外の方法でインスタンスに接続します。
2. インスタンス上で次のコマンドを実行してec2-instance-connectパッケージを最新バージョンにアップデートします。

```
apt update && apt upgrade
```

プライベートキーを紛失しました。Linux インスタンスに接続するにはどうすればよいですか?

EBS-Backed インスタンスのプライベートキーを失った場合は、インスタンスへのアクセス権を回復することができます。インスタンスを停止し、そのルートボリュームをデタッチし、データボリュームとして別のインスタンスにアタッチし、新しいパブリックキーで authorized_keys ファイルを変更して、ボリュームを元のインスタンスに戻し、インスタンスを再起動する必要があります。インスタンスの起動、接続、および停止の詳細については、インスタンスのライフサイクルを参照してください。

この手順は、EBS ルートボリュームを持つインスタンスでのみサポートされます。ルートデバイスがインスタンスストアボリュームの場合、この手順を使用してインスタンスへのアクセスを回復することはできません。インスタンスに接続するには、プライベートキーが必要です。インスタンスのルート· デバイス· タイプを決定するには、Amazon EC2コンソールを開き、[インスタンス] を選択し、[ストレージ] タブを選択し、[ルートデバイス詳細] セクションで、[ルートデバイスタイプ] の値をチェックします。

この値は EBS または INSTANCE-STORE のどちらかです。

プライベートキーを紛失した場合、以下の手順以外にも Linux インスタンスに接続する方法があります。詳細については、最初の起動後に SSH キーペアを紛失した場合、Amazon EC2 インスタンスに接続するにはどうすればよいですか?を参照してください。

別のキーペアを使用して EBS-Backed インスタンスに接続するためのステップ

- ステップ 1: 新しいキーペアを作成する
- ステップ 2: 元のインスタンスとそのルートボリュームに関する情報を取得する
- ステップ 3: 元のインスタンスを停止する
- ステップ 4: 一時インスタンスを起動する
- ステップ 5: 元のインスタンスからルートボリュームをデタッチし、一時インスタンスにアタッチする
- ステップ 6: 一時インスタンスにマウントされた元のボリュームの authorized_keys に、新しいパブリックキーを追加する
- ステップ 7: 一時インスタンスから元のボリュームをアンマウントしてデタッチし、元のインスタンスに再アタッチする
- ステップ 8: 新しいキーペアを使用して元のインスタンスに接続する
- ステップ 9: クリーンアップする

ステップ 1: 新しいキーペアを作成する

Amazon EC2 コンソールまたはサードパーティ製のツールで、新しいキーペアを作成します。新しいキーペアの名前として、紛失したプライベートキーと同じ名前を指定するには、まず既存のキーペアを削除する必要があります。新しいキーペアの作成の詳細については、Amazon EC2 を使用してキーペアを作成するまたはサードパーティー製のツールを使用してキーペアを作成し、Amazon EC2 にパブリックキーをインポートするを参照してください。

ステップ 2: 元のインスタンスとそのルートボリュームに関する情報を取得する

この手順を完了するために必要になるので、次の情報を書き留めます。

元のインスタンスに関する情報を取得するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、接続先にするインスタンスを選択します (このインスタンスを元のインスタンスと呼びます)。

3. [Details] タブで、インスタンス ID と AMI ID を書き留めます。
4. [ネットワーキング] タブで、アベイラビリティーゾーンを書き留めます。
5. [Storage] タブの [Root device name] で、ルートボリュームのデバイス名 (/dev/xvda など) を書き留めます。次に、[Block devices] で、このデバイス名を見つけ、ボリューム ID (vol-0a1234b5678c910de など) を書き留めます。

ステップ 3: 元のインスタンスを停止する

[Instance state (インスタンスの状態)]、[Stop instance (インスタンスの停止)] の順に選択します。このオプションが無効になっている場合は、インスタンスが既に停止しているか、またはルートボリュームがインスタンスストアボリュームです。

> **Warning**
>
> インスタンスを停止すると、インスタンスストアボリューム上のデータは消去されます。インスタンスストアボリュームのデータを保持するには、データを永続的ストレージに必ずバックアップします。

ステップ 4: 一時インスタンスを起動する

New console

一時インスタンスを起動するには

1. ナビゲーションペインで [Instances] (インスタンス)、[Launch instances] (インスタンスの起動) の順に選択します。
2. [Name and tags] (名前とタグ) セクションの [Name] (名前) に「Temporary (一時)」と入力します。
3. [Application and OS Images] (アプリケーションと OS イメージ) セクションで、元のインスタンスの起動に使用したものと同じ AMI を選択します。その AMI を使用できない場合は、停止したインスタンスから使用可能な AMI を作成できます。詳細については、「Amazon EBS-backed Linux AMI を作成する」、「」を参照してください。
4. [Instance type] (インスタンスタイプ) セクションでは、デフォルトのインスタンスタイプを保持します。
5. [Key pair] (キーペア) セクションの [Key pair name] (キーペア名) で、使用する既存のキーペアを選択するか、新しいキーペアを作成します。

6. [ネットワーク設定] セクションで [編集] を選択し、次に [サブネット] で、元のインスタンスと同じアベイラビリティーゾーンのサブネットを選択します。
7. [Summary] (サマリー) パネルで、[Launch] (起動) を選択します。

Old console

[Launch instances] を選択し、launch wizardを使用して、以下のオプションで一時インスタンスを起動します。

- [Choose an AMI] ページで、元のインスタンスを起動するのに使用したのと同じ AMI を選択します。その AMI を使用できない場合は、停止したインスタンスから使用可能な AMI を作成できます。詳細については、Amazon EBS-backed Linux AMI を作成するを参照してください。
- [Choose an Instance Type] ページで、ウィザードによって自動的に選択されたデフォルトのインスタンスタイプをそのままにします。
- [インスタンス詳細を設定する] ページで、元のインスタンスと同じアベイラビリティーゾーンを指定します。VPC のインスタンスを起動する場合、このアベイラビリティーゾーンのサブネットを選択します。
- [Add Tags] ページで、一時インスタンスであることを示すために、インスタンスに `Name=Temporary` タグを追加します。
- [Review] ページで、[Launch] を選択します。ステップ 1 で作成したキーペアを選択し、インスタンスの起動を選択します。

ステップ 5: 元のインスタンスからルートボリュームをデタッチし、一時インスタンスにアタッチする

1. ナビゲーションペインで [Volumes] を選択し、元のインスタンスのルートデバイスボリュームを選択します (前のステップでそのボリューム ID を書き留めました)。[Actions] (アクション)、[Detach Volume] (ボリュームのデタッチ)、[Detach] (デタッチする) の順に選択します。ボリュームの状態が `available` になるまで待ちます ([Refresh] アイコンを選択しなければならない場合があります)。
2. ボリュームを選択したまま [Actions] (アクション) を選択し、次に [Attach Volume] (ボリュームをアタッチ) を選択します。一時インスタンスのインスタンス ID を選択し、[Device name] (デバイス名) で指定されたデバイス名 (例: `/dev/sdf`) を書き留めて、[Attach volume] (ボリュームをアタッチ) を選択します。

> **Note**
>
> 元のインスタンスを AWS Marketplace AMI から起動して、ボリュームに AWS Marketplace のコードが含まれている場合は、ボリュームをアタッチする前に一時インスタンスを停止する必要があります。

ステップ 6: 一時インスタンスにマウントされた元のボリュームの **authorized_keys** に、新しいパブリックキーを追加する

1. 一時インスタンスに接続します。
2. 一時インスタンスから、そのファイルシステムにアクセスできるように、インスタンスにアタッチしたボリュームをマウントします。例えば、デバイス名が /dev/sdf の場合、次のコマンドを使用してボリュームを /mnt/tempvol としてマウントします。

> **Note**
>
> デバイス名の表示がインスタンスでは異なる場合があります。例えば、/dev/sdf としてマウントされているデバイスが、インスタンスでは /dev/xvdf として表示される場合があります。Red Hat の一部のバージョン (または CentOS などのバリアント) では、さらに末尾の文字が 4 文字インクリメントされる場合があります。例えば、/dev/sd*f* は /dev/xvd*k* になります。

   a. lsblkコマンドを使用して、ボリュームがパーティション分割されているかどうかを判断します。

```
[ec2-user ~]$ lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda    202:0    0    8G  0 disk
##xvda1 202:1    0    8G  0 part /
xvdf    202:80   0  101G  0 disk
##xvdf1 202:81   0  101G  0 part
xvdg    202:96   0   30G  0 disk
```

   前述の例では、/dev/xvda と /dev/xvdf は、パーティション分割されたボリュームで、/dev/xvdg はパーティション分割されていません。ボリュームがパーティション分割

されている場合は、次のステップで raw デバイス (/dev/xvdf1)) の代わりにパーティション(/dev/xvdf) をマウントします。

b. ボリュームをマウントするための一時ディレクトリを作成します。

```
[ec2-user ~]$ sudo mkdir /mnt/tempvol
```

c. 以前に特定したデバイス名またはボリューム名を使用して、一時マウントポイントにボリューム (またはパーティション) をマウントします。必要なコマンドは、オペレーティングシステムのファイルシステムによって異なります。注意事項デバイス名の表示がインスタンスでは異なる場合があります。詳細については、ステップ 6 の「note」を参照してください。

- Amazon Linux、Ubuntu、Debian

```
[ec2-user ~]$ sudo mount /dev/xvdf1 /mnt/tempvol
```

- Amazon Linux 2、CentOS、SUSE Linux 12、RHEL 7.x

```
[ec2-user ~]$ sudo mount -o nouuid /dev/xvdf1 /mnt/tempvol
```

> **Note**
>
> ファイルシステムが破損していることを示すエラーが表示された場合は、次のコマンドを実行して fsck ユーティリティを使用してファイルシステムをチェックし、問題を修復します。
>
> ```
> [ec2-user ~]$ sudo fsck /dev/xvdf1
> ```

3. 一時インスタンスから、次のコマンドを使用して、一時インスタンスの authorized_keys からの新しいパブリックキーを使用し、マウントされたボリューム上で authorized_keys を更新します。

> **Important**
>
> 以下の例では、Amazon Linux ユーザー名 ec2-user を使用します。Ubuntu インスタンスの場合は ubuntu など、別のユーザー名への置き換えが必要になる場合があります。

```
[ec2-user ~]$ cp .ssh/authorized_keys /mnt/tempvol/home/ec2-user/.ssh/
authorized_keys
```

このコピーが正常に終了すると、次のステップに進むことができます。

(オプション) または、/mnt/tempvol のファイルを編集するアクセス許可がない場合、sudo を使用してファイルを更新してから、ファイルに対するアクセス許可を確認して、元のインスタンスにログインできるかどうかを確認する必要があります。次のコマンドを使用して、ファイルに対するアクセス許可を確認します。

```
[ec2-user ~]$ sudo ls -l /mnt/tempvol/home/ec2-user/.ssh
total 4
-rw------- 1 222 500 398 Sep 13 22:54 authorized_keys
```

この出力例では、*222* はユーザー ID、*500* はグループ ID です。次に、sudo を使用して失敗したコピーコマンドを再実行します。

```
[ec2-user ~]$ sudo cp .ssh/authorized_keys /mnt/tempvol/home/ec2-user/.ssh/
authorized_keys
```

次のコマンドを再度実行して、アクセス許可が変更されているかどうかを判断します。

```
[ec2-user ~]$ sudo ls -l /mnt/tempvol/home/ec2-user/.ssh
```

ユーザー ID とグループ ID が変更されている場合は、次のコマンドを実行して復元します。

```
[ec2-user ~]$ sudo chown 222:500 /mnt/tempvol/home/ec2-user/.ssh/authorized_keys
```

ステップ 7: 一時インスタンスから元のボリュームをアンマウントしてデタッチし、元のインスタンスに再アタッチする

1. 一時インスタンスから、元のインスタンスに再アタッチできるように、アタッチしたボリュームをアンマウントします。例えば、/mnt/tempvol のボリュームをアンマウントするには、次のコマンドを使用します。

```
[ec2-user ~]$ sudo umount /mnt/tempvol
```

2. (前のステップでアンマウントした) 一時インスタンスからボリュームをデタッチする: Amazon EC2 コンソールのナビゲーションペインで [Volumes] (ボリューム) を選択し、(前のステップでボリューム ID を書き留めた) 元のインスタンスのルートデバイスボリュームを選択し、[Actions] (アクション)、[Detach Volume] (ボリュームのデタッチ) の順に選択します。次に、[Detach] (デタッチ) を選択します。ボリュームの状態が `available` になるまで待ちます ([Refresh] アイコンを選択しなければならない場合があります)。
3. ボリュームを元のインスタンスに再アタッチする: ボリュームを選択した状態で、[Action] (アクション)、[Attach Volume] (ボリュームをアタッチ) の順に選択します。元のインスタンスのインスタンス ID を選択し、元のルートデバイスのアタッチメントについて先程の ステップ 2 で記録したデバイス名 (`/dev/sda1` または `/dev/xvda`) を指定してから、[Attach Volume] (ボリュームをアタッチ) を選択します。

> **Important**
>
> 元のアタッチと同じデバイス名を指定しない場合、元のインスタンスを起動することはできません。Amazon EC2 は、ルートデバイスボリュームが `sda1` または `/dev/xvda` であることを想定しています。

ステップ 8: 新しいキーペアを使用して元のインスタンスに接続する

元のインスタンスを選択し、[Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択します。インスタンスが `running` 状態になったら、新しいキーペアのプライベートキーファイルを使用して、そのインスタンスに接続できます。

> **Note**
>
> 新しいキーペアおよび対応するプライベートキーファイルの名前が元のキーペアの名前と異なる場合は、インスタンスに接続するときに新しいプライベートキーファイルの名前を必ず指定します。

ステップ 9: クリーンアップする

(オプション) 一時インスタンスをそれ以上使用しない場合は、終了できます。一時インスタンスを選択し、[Instance state (インスタンスの状態)]、[Terminate instance (インスタンスの終了)] の順に選択します。

インスタンスの停止に関するトラブルシューティング

Amazon EBS-Backed インスタンスを停止して `stopping` 状態のままスタックしているように見える場合、基になるホストコンピュータに問題がある可能性があります。

インスタンスが `stopping` 状態または `running` 以外の状態にある間は、インスタンスの使用にコストがかかりません。インスタンスが `running` 状態のときのみ、インスタンスの使用量に対して課金されます。

インスタンスの強制停止

コンソールまたは AWS CLI を使用してインスタンスを強制的に停止できます。

> **Note**
>
> インスタンスが `stopping` 状態にある間のみ、コンソールを使用してインスタンスを強制的に停止できます インスタンスが `shutting-down` および `terminated` 以外の状態にある場合、AWS CLIを使用してインスタンスを強制的に停止できます。

Console

コンソールを使用してインスタンスを強制的に停止するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Instances (インスタンス)] を選択し、処理が止まってしまったインスタンスを選択します。
3. [Instance state (インスタンスの状態)]、[Force stop instance (インスタンスの強制停止)]、[Stop (停止)] の順に選択します。

   [Force stop instance] (インスタンスの強制停止) は、インスタンスが`stopping`状態である場合のみコンソールで利用できることに注意してください。インスタンスが別の状態の場合

(shutting-down と terminated を除く)は、AWS CLI を使用してインスタンスを強制停止します。

AWS CLI

AWS CLI を使用してインスタンスを強制的に停止するには

stop-instances コマンドと --force オプションを次のように使用します。

```
aws ec2 stop-instances --instance-ids i-0123ab456c789d01e --force
```

10 分経過してもインスタンスが停止しない場合、AWS re:Post にヘルプリクのエストを投稿してください。迅速な解決のために、インスタンス ID を含めて、既に行った手順について説明してください。また、サポートプランを契約している場合は、サポートセンターでサポートケースを作成できます。

代替インスタンスの作成

AWS re:Post または [Support Center] (サポートセンター) からの支援を待っている間に問題解決を試みるには、代替えのインスタンスを作成してください。処理が止まってしまったインスタンスの AMI を作成し、新しい AMI を使用して新しいインスタンスを起動します。

> **Important**
>
> インスタンスのステータスチェックを行うと、壊れた OS の完全なレプリカが AMI にコピーされることになるため、システムのステータスチェックのみを登録する場合は、代替インスタンスを作成することをお勧めします。ステータスメッセージを確認したら、AMI を作成し、新しい AMI を使用して新しいインスタンスを起動します。

Console

コンソールを使用して代わりのインスタンスを作成するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインの [Instances (インスタンス)] を選択し、処理が止まってしまったインスタンスを選択します。

3. [アクション]、[Image and templates (イメージとテンプレート)]、[イメージの作成] の順に選択します。
4. [イメージの作成] ページで、次の操作を行います。

   a. AMI の名前と説明を入力します。

   b. [No reboot] を選択します。

   c. [イメージを作成] を選択します。

   詳細については、「インスタンスからの Linux AMI の作成 」を参照してください。
5. AMI から新しいインスタンスを起動し、その新しいインスタンスが動作していることを確認します。
6. 処理が止まってしまったインスタンスを選択し、[Actions (アクション)]、[Instance state (インスタンスの状態)]、[Terminate instance (インスタンスの終了)] の順に選択します。インスタンスの終了処理も止まってしまう場合は、Amazon EC2 は数時間以内に自動的にそのインスタンスを強制終了します。

AWS CLI

CLI を使用して代わりのインスタンスを作成するには

1. create-image (AWS CLI) コマンドと `--no-reboot` オプションを次のように使用して、処理が止まってしまったインスタンスから AMI を作成します。

```
aws ec2 create-image --instance-id i-0123ab456c789d01e --name "AMI" --
description "AMI for replacement instance" --no-reboot
```

2. run-instances (AWS CLI) コマンドを次のように使用し、作成した AMI から新しいインスタンスを起動します。

```
aws ec2 run-instances --image-id ami-1a2b3c4d --count 1 --instance-type c3.large
 --key-name MyKeyPair --security-groups MySecurityGroup
```

3. 新しいインスタンスが動作していることを確認します。
4. 次のように terminate-instances (AWS CLI) コマンドを使用し、処理が止まってしまったインスタンスを終了します。

```
aws ec2 terminate-instances --instance-ids i-1234567890abcdef0
```

前の手順で説明されたように、インスタンスから AMI を作成できない場合は、次のようにして代わりのインスタンスを設定できます。

(代替方法) コンソールを使用して代わりのインスタンスを作成するには

1. インスタンスを選択し、[Description (説明)]、[Block devices (ブロックデバイス)] の順に選択します。各ボリュームを選択し、そのボリューム ID を書き留めます。必ずどのボリュームがルートボリュームであるかメモしておきます。
2. ナビゲーションペインの [Volumes] を選択します。インスタンスの各ボリュームを選択し、[Actions]、[Create Snapshot] の順に選択します。
3. ナビゲーションペインで、[Snapshots] を選択します。作成したスナップショットを選択し、[Actions]、[Create Volume] の順に選択します。
4. 処理が止まってしまったインスタンスと同じオペレーティングシステムのインスタンスを起動します。そのルートボリュームのボリューム ID とデバイス名をメモしておきます。
5. ナビゲーションペインで、[Instances] を選択し、起動したインスタンスを選択した後で、[Instance state]、[Stop instance] の順に選択します。
6. ナビゲーションペインで [Volumes] を選択し、停止したインスタンスのルートボリュームを選択した後で、[Actions]、[Detach Volume] の順に選択します。
7. 処理が停止してしまったインスタンスから作成したルートボリュームを選択し、[Actions]、[Attach Volume] の順に選択して、そのルートボリュームとして新しいインスタンスにアタッチします (書き留めたデバイス名を使用)。その他の非ルートボリュームをインスタンスにアタッチします。
8. ナビゲーションペインで、[Instances] を選択し、代わりのインスタンスを選択します。[Instance state (インスタンスの状態)]、[Start instance (インスタンスの開始)] の順に選択します。インスタンスが動作していることを確認します。
9. 処理が止まったインスタンスを選択し、[Instance state (インスタンスの状態)]、[Terminate instance (インスタンスの終了)] の順に選択します。インスタンスの終了処理も止まってしまう場合は、Amazon EC2 は数時間以内に自動的にそのインスタンスを強制終了します。

インスタンスの終了 (シャットダウン) のトラブルシューティング

インスタンスが `running` 状態ではない場合は、インスタンスの使用に対して課金されません。つまり、インスタンスを終了させると、そのステータスが `shutting-down` に変わるとすぐに、そのインスタンスへの課金は停止します。

インスタンスがすぐに終了する

複数の問題により、起動時にインスタンスがすぐに終了する可能性があります。詳細については、「インスタンスがすぐに終了する」を参照してください。

インスタンスの削除の遅延

インスタンスの shutting-down 状態が数分以上続く場合は、インスタンスによって実行されるシャットダウンスクリプトが原因で遅れている可能性があります。

もう 1 つ考えられる原因として、基盤となるホストコンピュータの問題があります。インスタンスの shutting-down 状態が数時間以上続く場合、Amazon EC2 はそれを停止したインスタンスとして扱い、強制終了します。

インスタンスの終了処理が停止していると考えられ、すでに数時間以上経過している場合は、AWS re:Post にヘルプリクエストを投稿してください。迅速な解決のために、インスタンス ID を含めて、既に行った手順について説明してください。また、サポートプランを契約している場合は、サポートセンターでサポートケースを作成できます。

表示されているインスタンスを削除する

インスタンスの削除後、インスタンスはしばらくの間削除されずに表示されたままとなります。状態は terminated となります。このエントリが数時間経過しても削除されない場合には、サポートに連絡してください。

エラー: インスタンスは終了できない可能性があります。その「disableApiTermination」インスタンス属性を変更します

インスタンスを終了しようとしたときに `The instance instance_id may not be terminated. Modify its 'disableApiTermination' instance attribute` エラーメッセージが表示される場合は、そのインスタンスの終了保護が有効になっていることを示します。削除保護はインスタンスが誤って削除されないようにします。詳細については、「終了保護を有効化する」を参照してください。

インスタンスを終了する前に、終了保護を無効にする必要があります。

Amazon EC2 コンソールを使用して終了保護を無効にするには、インスタンスを選択してから、[アクション]、[インスタンス設定]、[終了保護の変更] を選択します。

AWS CLI を使用してクラスターの削除保護を無効にするには、次のコマンドを実行します。

```
$ aws ec2 modify-instance-attribute --instance-id instance_id --no-disable-api-
termination
```

インスタンスが自動的に起動または終了される

通常、以下の動作は、定義した基準に基づいて自動的にコンピューティングリソースをスケールするため、Amazon EC2 Auto Scaling、EC2 フリート、またはスポットフリートを使用していることを意味します。

- インスタンスを終了すると、別のインスタンスが自動的に起動します。
- インスタンスを起動すると、いずれかのインスタンスが自動的に終了します。
- インスタンスを停止すると、そのインスタンスが終了し、別のインスタンスが自動的に起動します。

自動スケーリングを停止するには、「Amazon EC2 Auto Scaling ユーザーガイド」、「EC2 Fleet」、または「スポットフリートリクエストを作成します。」を参照してください。

ステータスチェックに失敗したインスタンスのトラブルシューティング

以下の情報は、インスタンスでステータスチェックに失敗した場合の問題のトラブルシューティングに役立ちます。まず、アプリケーションで問題が発生しているかどうかを確認します。インスタンスでアプリケーションが正常に実行されていないことを確認した場合は、ステータスチェック情報とシステムログを確認します。

ステータスチェックの失敗の原因となる問題の例については、「インスタンスのステータスチェック」を参照してください。

コンテンツ

- エラー: mmu_update failed (メモリ管理の更新に失敗しました)
- I/O エラー (ブロックデバイス障害)
- I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス)
- request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている)
- 「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」 (カーネルと AMI の不一致)
- 「FATAL: Could not load /lib/modules」または「BusyBox」 (カーネルモジュールの欠如)
- エラー: 無効のカーネル (EC2 と互換性のないカーネル)
- fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。)
- General error mounting filesystems (マウント失敗)
- VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致)
- Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致)
- XENBUS: Device with no driver..。
- ... days without being checked, check forced (ファイルシステムのチェックが必要です)
- fsck died with exit status..。(デバイスが見つからない)
- GRUB プロンプト (grubdom>)
- Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス)
- SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定)
- XENBUS: Timeout connecting to devices (Xenbus タイムアウト)

ステータスチェック情報の確認

Amazon EC2 コンソールを使用して、問題のあるインスタンスを調査するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [インスタンス] を選択し、インスタンスを選択します。
3. 詳細ペインの [ステータスとアラーム] タブを選択して、すべての [システムステータスのチェック] と [インスタンスステータスのチェック] に関する個々の結果を表示します。

システムのステータスチェックに失敗した場合、次のいずれかの方法を試すことができます。

- インスタンスの復旧アラームを作成します。詳細については、「インスタンスを停止、終了、再起動、または復旧するアラームを作成する」を参照してください。
- インスタンスタイプを Nitro システムで構築されたインスタンスに変更した場合、必要な ENA と NVMe ドライバがないインスタンスから移行するとステータスチェックは失敗します。詳細については、「インスタンスタイプ変更の互換性」を参照してください。
- Amazon EBS-Backed AMI を使用するインスタンスの場合、いったんインスタンスを停止してから再開します。
- instance-store backed AMI を使用するインスタンスの場合、インスタンスを終了し、代わりのインスタンスを起動します。
- Amazon EC2 が問題を解決するのを待ちます。
- 問題を AWSre: Post に投稿してください。
- インスタンスが Auto Scaling グループにある場合は、Amazon EC2 Auto Scaling サービスによって、代わりのインスタンスが自動的に起動されます。詳細については、『Amazon EC2 Auto Scaling ユーザーガイド』の「Auto Scaling インスタンスのヘルスチェック」を参照してください。
- システムログを取得し、エラーを探します。

システムログの取得

インスタンスのステータスチェックに失敗した場合は、インスタンスを再起動してシステムログを取得できます。ログから判明したエラーが問題のトラブルシューティングに役立つ場合があります。再起動すると、ログから不要な情報が消去されます。

インスタンスを再起動してシステムログを取得するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで [Instances] を選択し、インスタンスを選択します。
3. [インスタンスの状態]、[インスタンスの再起動] の順に選択します。インスタンスが再起動するまでには数分かかることがあります。
4. 問題がまだ存在することを確認します。再起動によって、問題が解決することがあります。
5. インスタンスが `running` 状態になったら、[アクション]、[モニタリングとトラブルシューティング]、[システムログの取得] の順に選択します。

6. 画面に表示されるログを確認し、下記の既知のシステムログエラー文のリストを使用して、問題のトラブルシューティングを行います。
7. 問題が解決されない場合は、問題を AWS re:Post に投稿できます。

Linux ベースのインスタンスに関するシステムログエラーのトラブルシューティング

インスタンスの接続性チェックなど、インスタンスのステータスチェックに失敗した Linux ベースのインスタンスの場合、上記の手順に従ってシステムログを取得したことを確認します。次のリストは、一般的なシステムログエラー、および各エラーの問題解決に対して推奨する対処を示しています。

メモリエラー

- メモリ不足: プロセスの終了
- エラー: mmu_update failed (メモリ管理の更新に失敗しました)

デバイスエラー

- I/O エラー (ブロックデバイス障害)
- I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス)

カーネルエラー

- request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている)
- 「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」 (カーネルと AMI の不一致)
- 「FATAL: Could not load /lib/modules」または「BusyBox」 (カーネルモジュールの欠如)
- エラー: 無効のカーネル (EC2 と互換性のないカーネル)

ファイルシステムエラー

- fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。)
- General error mounting filesystems (マウント失敗)

- VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致)
- Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致)
- XENBUS: Device with no driver..。
- ... days without being checked, check forced (ファイルシステムのチェックが必要です)
- fsck died with exit status..。(デバイスが見つからない)

[オペレーティングシステムエラー]

- GRUB プロンプト (grubdom>)
- Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス)
- SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定)
- XENBUS: Timeout connecting to devices (Xenbus タイムアウト)

メモリ不足: プロセスの終了

メモリ不足エラーは、下記のようなシステムログで示されます。

```
[115879.769795] Out of memory: kill process 20273 (httpd) score 1285879
or a child
[115879.769795] Killed process 1917 (php-cgi) vsz:467184kB, anon-
rss:101196kB, file-rss:204kB
```

可能性のある原因

メモリの枯渇

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • インスタンスを停止し、異なるインスタンスタイプを使用するようにインスタンスを変更

インスタンスタイプ	操作
	した後、インスタンスを再び起動します。たとえば、大きいインスタンスタイプやメモリ最適化インスタンスタイプです。 • インスタンスを再起動して、障害のないステータスに戻します。インスタンスタイプを変更しない限り、問題が再び発生する可能性があります。
Instance store-Backed	次のいずれかを行ってください。 • インスタンスを終了し、別のインスタンスタイプを指定して、新しいインスタンスを起動します。たとえば、大きいインスタンスタイプやメモリ最適化インスタンスタイプです。 • インスタンスを再起動して、障害のないステータスに戻します。インスタンスタイプを変更しない限り、問題が再び発生する可能性があります。

エラー: mmu_update failed (メモリ管理の更新に失敗しました)

メモリ管理更新失敗は、下記のようなシステムログで示されます。

```
...
Press `ESC' to enter the menu... 0   [H[J  Booting 'Amazon Linux 2011.09
 (2.6.35.14-95.38.amzn1.i686)'


root (hd0)

 Filesystem type is ext2fs, using whole disk

kernel /boot/vmlinuz-2.6.35.14-95.38.amzn1.i686 root=LABEL=/ console=hvc0 LANG=

en_US.UTF-8 KEYTABLE=us

initrd /boot/initramfs-2.6.35.14-95.38.amzn1.i686.img
```

```
ERROR: mmu_update failed with rc=-22
```

可能性のある原因

Amazon Linux に関する問題

推奨する対処

問題を デベロッパーフォーラムに投稿するか、AWS Support にお問い合わせください。

I/O エラー (ブロックデバイス障害)

入力/出力エラーは、次の例のようなシステムログで示されます。

```
[9943662.053217] end_request: I/O error, dev sde, sector 52428288
[9943664.191262] end_request: I/O error, dev sde, sector 52428168
[9943664.191285] Buffer I/O error on device md0, logical block 209713024
[9943664.191297] Buffer I/O error on device md0, logical block 209713025
[9943664.191304] Buffer I/O error on device md0, logical block 209713026
[9943664.191310] Buffer I/O error on device md0, logical block 209713027
[9943664.191317] Buffer I/O error on device md0, logical block 209713028
[9943664.191324] Buffer I/O error on device md0, logical block 209713029
[9943664.191332] Buffer I/O error on device md0, logical block 209713030
[9943664.191339] Buffer I/O error on device md0, logical block 209713031
[9943664.191581] end_request: I/O error, dev sde, sector 52428280
[9943664.191590] Buffer I/O error on device md0, logical block 209713136
[9943664.191597] Buffer I/O error on device md0, logical block 209713137
[9943664.191767] end_request: I/O error, dev sde, sector 52428288
[9943664.191970] end_request: I/O error, dev sde, sector 52428288
[9943664.192143] end_request: I/O error, dev sde, sector 52428288
[9943664.192949] end_request: I/O error, dev sde, sector 52428288
[9943664.193112] end_request: I/O error, dev sde, sector 52428288
[9943664.193266] end_request: I/O error, dev sde, sector 52428288
...
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	障害が発生した Amazon EBS ボリューム

インスタンスタイプ	可能性のある原因
Instance store-Backed	障害が発生した物理ドライブ

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ボリュームをデタッチします。 3. ボリュームの回復を試みます。 **Note** Amazon EBS ボリュームのスナップショットを頻繁に作成することをお勧めします。これによって、エラーのためにデータを損失する危険性が大幅に減少します。 4. ボリュームをインスタンスに再アタッチします。 5. インスタンスを起動します。
Instance store-Backed	インスタンスを終了し、新しいインスタンスを起動します。 **Note** データを復旧できない。バックアップから復旧します。

インスタンスタイプ	操作
	Note Amazon S3 または Amazon EBS をバックアップに使用することをお勧めします。インスタンスストアボリュームは、単一のホストと単一のディスクエラーに直接結びついています。

I/O エラー: ローカルでもリモートディスクでもありません (破損した分散ブロックデバイス)

デバイスでの入力/出力エラーは、次の例のようなシステムログで示されます。

```
...
block drbd1: Local IO failed in request_timer_fn. Detaching...

Aborting journal on device drbd1-8.

block drbd1: IO ERROR: neither local nor remote disk

Buffer I/O error on device drbd1, logical block 557056

lost page write due to I/O error on drbd1

JBD2: I/O error detected when updating journal superblock for drbd1-8.
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	障害が発生した Amazon EBS ボリューム
Instance store-Backed	障害が発生した物理ドライブ

推奨する対処

インスタンスを終了し、新しいインスタンスを起動します。

Amazon EBS-Backed インスタンスの場合、最新スナップショットからイメージを作成して、データを回復できます。スナップショットを作成した後に追加されたデータは回復できません。

request_module: runaway loop modprobe (古い Linux バージョンでレガシーカーネル modprobe がループしている)

下記のようなシステムログで、この状態が示されます。不安定であるか古い Linux カーネル (例: 2.6.16-xenU) を使用すると、起動時に無限ループが発生することがあります。

```
Linux version 2.6.16-xenU (builder@xenbat.amazonsa) (gcc version 4.0.1
20050727 (Red Hat 4.0.1-5)) #1 SMP Mon May 28 03:41:49 SAST 2007

BIOS-provided physical RAM map:

 Xen: 0000000000000000 - 0000000026700000 (usable)

0MB HIGHMEM available.
...

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c

request_module: runaway loop modprobe binfmt-464c
```

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかのオプションを使用して、GRUB ベースまたは静的な新しいカーネルを使用します。

インスタンスタイプ	操作
	オプション 1: インスタンスを終了し、-kernel および -ramdisk パラメータを指定して新しいインスタンスを起動します。 オプション 2: 1. インスタンスを停止します。 2. 新しいカーネルを使用するようカーネルとラムディスク属性を変更します。 3. インスタンスを起動します。
Instance store-Backed	インスタンスを終了し、-kernel および -ramdisk パラメータを指定して新しいインスタンスを起動します。

「FATAL: kernel too old」および「fsck: No such file or directory while trying to open /dev」 (カーネルと AMI の不一致)

下記のようなシステムログで、この状態が示されます。

```
Linux version 2.6.16.33-xenU (root@dom0-0-50-45-1-a4-ee.z-2.aes0.internal)
(gcc version 4.1.1 20070105 (Red Hat 4.1.1-52)) #2 SMP Wed Aug 15 17:27:36 SAST 2007
...
FATAL: kernel too old
Kernel panic - not syncing: Attempted to kill init!
```

可能性のある原因

互換性のないカーネルとユーザーランド

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。

インスタンスタイプ	操作
	1. インスタンスを停止します。 2. 新しいカーネルを使用するよう設定を変更します。 3. インスタンスを起動します。
Instance store-Backed	次の手順に従ってください。 1. より新しいカーネルを使用する AMI を作成します。 2. インスタンスを終了します。 3. 作成した AMI から新しいインスタンスを起動します。

「FATAL: Could not load /lib/modules」または「BusyBox」 (カーネルモジュールの欠如)

下記のようなシステムログで、この状態が示されます。

```
[    0.370415] Freeing unused kernel memory: 1716k freed
Loading, please wait...
WARNING: Couldn't open directory /lib/modules/2.6.34-4-virtual: No such file or
 directory
FATAL: Could not open /lib/modules/2.6.34-4-virtual/modules.dep.temp for writing: No
 such file or directory
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
Couldn't get a file descriptor referring to the console
Begin: Loading essential drivers... ...
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
Done.
Begin: Running /scripts/init-premount ...
Done.
Begin: Mounting root file system... ...
Begin: Running /scripts/local-top ...
```

```
Done.
Begin: Waiting for root file system... ...
Done.
Gave up waiting for root device.  Common problems:
 - Boot args (cat /proc/cmdline)
   - Check rootdelay= (did the system wait long enough?)
   - Check root= (did the system wait for the right device?)
 - Missing modules (cat /proc/modules; ls /dev)
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
FATAL: Could not load /lib/modules/2.6.34-4-virtual/modules.dep: No such file or
 directory
ALERT! /dev/sda1 does not exist. Dropping to a shell!


BusyBox v1.13.3 (Ubuntu 1:1.13.3-1ubuntu5) built-in shell (ash)
Enter 'help' for a list of built-in commands.

(initramfs)
```

可能性のある原因

次の 1 つ以上の条件によって、この問題が発生する可能性があります。

- ラムディスクが見つからない
- ラムディスクに正しいモジュールが見つからない
- Amazon EBS ルートボリュームが /dev/sda1 として正しくアタッチされていない

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. Amazon EBS ボリュームに対して正しいラムディスクを選択します。 2. インスタンスを停止します。 3. ボリュームをデタッチし、修復します。

インスタンスタイプ	操作
	4. ボリュームをインスタンスにアタッチします。 5. インスタンスを起動します。 6. 正しいラムディスクを使用するよう AMI を変更します。
Instance store-Backed	次の手順に従ってください。 1. インスタンスを終了してから、正しいラムディスクを使って新たなインスタンスを起動します。 2. 正しいラムディスクを使って新たな AMI を作成します。

エラー: 無効のカーネル (EC2 と互換性のないカーネル)

下記のようなシステムログで、この状態が示されます。

```
...
root (hd0)

 Filesystem type is ext2fs, using whole disk

kernel /vmlinuz root=/dev/sda1 ro

initrd /initrd.img

ERROR Invalid kernel: elf_xen_note_check: ERROR: Will only load images
built for the generic loader or Linux images
xc_dom_parse_image returned -1

Error 9: Unknown boot failure

  Booting 'Fallback'

root (hd0)

 Filesystem type is ext2fs, using whole disk
```

```
kernel /vmlinuz.old root=/dev/sda1 ro

Error 15: File not found
```

可能性のある原因

次の一方または両方の条件によって、この問題が発生する可能性があります。

- 指定されたカーネルは GRUB でサポートされていません
- フォールバックカーネルが存在しません

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. 機能するカーネルに変更します。 3. フォールバックカーネルをインストールします。 4. カーネルを訂正して AMI を変更します。
Instance store-Backed	次の手順に従ってください。 1. インスタンスを終了してから、正しいカーネルを使って新たなインスタンスを起動します。 2. 正しいカーネルを使って AMI を作成します。 3. (オプション) データ復旧の技術サポートについては、AWS Support にお問い合わせください。

fsck: No such file or directory while trying to open..。(ファイルシステムが見つからない。)

下記のようなシステムログで、この状態が示されます。

```
  Welcome to Fedora
  Press 'I' to enter interactive startup.
Setting clock : Wed Oct 26 05:52:05 EDT 2011 [  OK  ]

Starting udev: [  OK  ]

Setting hostname localhost:  [  OK  ]

No devices found
Setting up Logical Volume Management: File descriptor 7 left open
  No volume groups found
[  OK  ]

Checking filesystems
Checking all file systems.
[/sbin/fsck.ext3 (1) -- /] fsck.ext3 -a /dev/sda1
/dev/sda1: clean, 82081/1310720 files, 2141116/2621440 blocks
[/sbin/fsck.ext3 (1) -- /mnt/dbbackups] fsck.ext3 -a /dev/sdh
fsck.ext3: No such file or directory while trying to open /dev/sdh

/dev/sdh:
The superblock could not be read or does not describe a correct ext2
filesystem.  If the device is valid and it really contains an ext2
filesystem (and not swap or ufs or something else), then the superblock
is corrupt, and you might try running e2fsck with an alternate superblock:
    e2fsck -b 8193 <device>

[FAILED]


*** An error occurred during the file system check.
*** Dropping you to a shell; the system will reboot
*** when you leave the shell.
Give root password for maintenance
(or type Control-D to continue):
```

可能性のある原因

- ラムディスクファイルシステム定義 /etc/fstab にバグがある
- /etc/fstab のファイルシステム定義の設定が不適切
- ドライブが見つからないかドライブにエラーがある

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止し、ルートボリュームをデタッチし、/etc/fstab を修復または変更し、ボリュームをインスタンスにアタッチし、インスタンスを起動します。 2. ラムディスクを修正して、変更した /etc/fstab を含めます (適用可能な場合)。 3. 新しいラムディスクを使用するよう AMI を変更します。 fstab の 6 番目のフィールドではマウントの可用性要件を定義します。0 以外の値を指定した場合、そのボリュームに対して fsck を実行して成功しなければならないことを意味します。Amazon EC2 でこのフィールドを使用すると、問題が発生することがあります。これは、実行に失敗すると通常は対話的なコンソールプロンプトが表示されますが、このコンソールプロンプトは現在 Amazon EC2 で使用できないためです。この機能は慎重に使用してください。また、fstab の Linux マニュアルページを参照してください。
Instance store-Backed	次の手順に従ってください。

インスタンスタイプ	操作
	1. インスタンスを終了し、新しいインスタンスを起動します。 2. 障害のある Amazon EBS ボリュームをデタッチし、インスタンスを再起動します。 3. (オプション) データ復旧の技術サポートについては、AWS Support にお問い合わせください。

General error mounting filesystems (マウント失敗)

下記のようなシステムログで、この状態が示されます。

```
Loading xenblk.ko module
xen-vbd: registered block device major 8

Loading ehci-hcd.ko module
Loading ohci-hcd.ko module
Loading uhci-hcd.ko module
USB Universal Host Controller Interface driver v3.0

Loading mbcache.ko module
Loading jbd.ko module
Loading ext3.ko module
Creating root device.
Mounting root filesystem.
kjournald starting.  Commit interval 5 seconds

EXT3-fs: mounted filesystem with ordered data mode.

Setting up other filesystems.
Setting up new root fs
no fstab.sys, mounting internal defaults
Switching to new root and running init.
unmounting old /dev
unmounting old /proc
unmounting old /sys
mountall:/proc: unable to mount: Device or resource busy
mountall:/proc/self/mountinfo: No such file or directory
```

```
mountall: root filesystem isn't mounted
init: mountall main process (221) terminated with status 1

General error mounting filesystems.
A maintenance shell will now be started.
CONTROL-D will terminate this shell and re-try.
Press enter for maintenance
(or type Control-D to continue):
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	• Amazon EBS ボリュームがデタッチされているか、ボリュームにエラーがあります。 • ファイルシステムが破損している。 • ラムディスクと AMI の組み合わせが一致していません (例: Debian ラムディスクと SUSE AMI)。
Instance store-Backed	• ドライブにエラーがあります。 • ファイルシステムが破損しています。 • ラムディスクと AMI の組み合わせが一致していません (例: Debian ラムディスクと SUSE AMI) 。

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ルートボリュームをデタッチする。

インスタンスタイプ	操作
	3. ルートボリュームを既知の動作しているインスタンスにアタッチします。 4. ファイルシステムチェック (fsck -a /dev/...) を実行します 5. エラーを修正します。 6. ボリュームを既知の動作しているインスタンスからデタッチします。 7. 停止したインスタンスにボリュームをアタッチします。 8. インスタンスを起動します。 9. インスタンスのステータスを再確認します。
Instance store-Backed	以下のいずれかを行ってください。 • 新しいインスタンスを起動します。 • (オプション) データ復旧の技術サポートについては、AWS Support にお問い合わせください。

VFS: Unable to mount root fs on unknown-block (ルートファイルシステム不一致)

下記のようなシステムログで、この状態が示されます。

```
Linux version 2.6.16-xenU (builder@xenbat.amazonsa) (gcc version 4.0.1
 20050727 (Red Hat 4.0.1-5)) #1 SMP Mon May 28 03:41:49 SAST 2007
...
Kernel command line:  root=/dev/sda1 ro 4
...
Registering block device major 8
...
Kernel panic - not syncing: VFS: Unable to mount root fs on unknown-block(8,1)
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	• デバイスが正しくアタッチされていない。 • ルートデバイスが正しいデバイスポイントにアタッチされていない。 • ファイルシステムのフォーマットが正しくありません。 • レガシーカーネルを使用している (たとえば、2.6.16-XenU)。 • インスタンスの最新のカーネル更新 (更新エラーまたは更新バグ)
Instance store-Backed	ハードウェアデバイスのエラー。

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • インスタンスを停止し、再起動します。 • 正しいデバイスポイントでアタッチするようルートボリュームを変更します。たとえば、/dev/sda の代わりに /dev/sda1 を使用します。 • 停止し、新しいカーネルを使用するように変更します。 • 既知の更新バグを確認するには、Linux ディストリビューションのドキュメントを参照してください。カーネルを変更または再インストールします。

インスタンスタイプ	操作
Instance store-Backed	インスタンスを終了し、新しいカーネルを使用して、新しいインスタンスを起動します。

Error: Unable to determine major/minor number of root device..。(ルートファイルシステム/デバイス不一致)

下記のようなシステムログで、この状態が示されます。

```
...
XENBUS: Device with no driver: device/vif/0
XENBUS: Device with no driver: device/vbd/2048
drivers/rtc/hctosys.c: unable to open rtc device (rtc0)
Initializing network drop monitor service
Freeing unused kernel memory: 508k freed
:: Starting udevd...
done.
:: Running Hook [udev]
:: Triggering uevents...<30>udevd[65]: starting version 173
done.
Waiting 10 seconds for device /dev/xvda1 ...
Root device '/dev/xvda1' doesn't exist. Attempting to create it.
ERROR: Unable to determine major/minor number of root device '/dev/xvda1'.
You are being dropped to a recovery shell
    Type 'exit' to try and continue booting
sh: can't access tty; job control turned off
[ramfs /]#
```

可能性のある原因

- 仮想ブロックデバイスドライバーが見つからないか、設定が間違っている
- デバイス列挙が競合している (sda と xvda または sda1 の代わりに sda)
- インスタンスカーネルが正しく選択されていない

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ボリュームをデタッチします。 3. デバイスのマッピングの問題を解決します。 4. インスタンスを起動します。 5. AMI を変更して、デバイスのマッピングの問題に対処します。
Instance store-Backed	次の手順に従ってください。 1. 適切な修正を使用して新しい AMI を作成します (ブロックデバイスを正しくマッピングします)。 2. インスタンスを終了し、作成した AMI から新しいインスタンスを起動します。

XENBUS: Device with no driver..。

下記のようなシステムログで、この状態が示されます。

```
XENBUS: Device with no driver: device/vbd/2048
drivers/rtc/hctosys.c: unable to open rtc device (rtc0)
Initializing network drop monitor service
Freeing unused kernel memory: 508k freed
:: Starting udevd...
done.
:: Running Hook [udev]
:: Triggering uevents...<30>udevd[65]: starting version 173
done.
Waiting 10 seconds for device /dev/xvda1 ...
Root device '/dev/xvda1' doesn't exist. Attempting to create it.
ERROR: Unable to determine major/minor number of root device '/dev/xvda1'.
You are being dropped to a recovery shell
```

```
    Type 'exit' to try and continue booting
sh: can't access tty; job control turned off
[ramfs /]#
```

可能性のある原因

- 仮想ブロックデバイスドライバーが見つからないか、設定が間違っている
- デバイス列挙が競合している (sda と xvda)。
- インスタンスカーネルが正しく選択されていない

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. インスタンスを停止します。 2. ボリュームをデタッチします。 3. デバイスのマッピングの問題を解決します。 4. インスタンスを起動します。 5. AMI を変更して、デバイスのマッピングの問題に対処します。
Instance store-Backed	次の手順に従ってください。 1. 適切な修正を使用して AMI を作成します (ブロックデバイスを正しくマッピングします)。 2. インスタンスを終了し、作成した AMI を使用して新しいインスタンスを起動します。

... days without being checked, check forced (ファイルシステムのチェックが必要です)

下記のようなシステムログで、この状態が示されます。

```
...
Checking filesystems
Checking all file systems.
[/sbin/fsck.ext3 (1) -- /] fsck.ext3 -a /dev/sda1
/dev/sda1 has gone 361 days without being checked, check forced
```

可能性のある原因

ファイルシステムのチェック期間が経過したため、ファイルシステムチェックが強制実行されている。

推奨する対処

- ファイルシステムチェックが完了するまで待ちます。ルートファイルシステムのサイズによっては、ファイルシステムチェックに時間がかかることがあります。
- tune2fs またはファイルシステムに適したツールを使用してファイルシステムを変更し、ファイルシステムチェック (fsck) の実行を削除します。

fsck died with exit status..。(デバイスが見つからない)

下記のようなシステムログで、この状態が示されます。

```
Cleaning up ifupdown....
Loading kernel modules...done.
...
Activating lvm and md swap...done.
Checking file systems...fsck from util-linux-ng 2.16.2
/sbin/fsck.xfs: /dev/sdh does not exist
fsck died with exit status 8
[31mfailed (code 8).[39;49m
```

可能性のある原因

- 存在しないドライブをラムディスクが検索している
- ファイルシステムの整合性チェックが強制実行されている
- ドライブにエラーがあるか、デタッチされている

推奨する対処

インスタンスタイプ	操作
Amazon EBS-backed	問題を解決するため、次のいずれかを試します。 • インスタンスを停止し、ボリュームを既存の実行中インスタンスにアタッチします。 • 整合性チェックを手動で実行します。 • ラムディスクを修正して、関連するユーティリティを含めます。 • ファイルシステム調整パラメータを変更して、整合性要件を削除します (お勧めしません)。
Instance store-Backed	問題を解決するため、次のいずれかを試します。 • ラムディスクに正しいツールをリバンドリングします。 • ファイルシステム調整パラメータを変更して、整合性要件を削除します (お勧めしません)。 • インスタンスを終了し、新しいインスタンスを起動します。 • (オプション) データ復旧の技術サポートについては、AWS Support にお問い合わせください。

GRUB プロンプト (grubdom>)

下記のようなシステムログで、この状態が示されます。

```
    GNU GRUB  version 0.97  (629760K lower / 0K upper memory)
```

```
       [ Minimal BASH-like line editing is supported.   For

         the   first   word,  TAB  lists  possible  command

         completions.  Anywhere else TAB lists the possible

         completions of a device/filename. ]

grubdom>
```

可能性のある原因

インスタンスタイプ	可能性のある原因
Amazon EBS-Backed	• GRUB 設定ファイルがありません。 • 正しくない GRUB イメージが使用されています。別の場所に GRUB 設定ファイルが必要です。 • GRUB 設定ファイルを保存するために使用されているファイルシステムがサポートされていません (たとえば、ルートファイルシステムを GRUB の以前のバージョンでサポートされていないタイプに変換した)
Instance store-Backed	• GRUB 設定ファイルがありません。 • 正しくない GRUB イメージが使用されています。別の場所に GRUB 設定ファイルが必要です。 • GRUB 設定ファイルを保存するために使用されているファイルシステムがサポートされていません (たとえば、ルートファイルシステムを GRUB の以前のバージョンでサポートされていないタイプに変換した)

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	オプション 1: AMI を変更しインスタンスを再作成します。 1. ソース AMI を変更して、標準の場所に GRUB 設定ファイルを作成します (/boot/grub/menu.lst)。 2. GRUB のバージョンが、基になるファイルシステムのタイプをサポートしていることを確認し、必要に応じて GRUB をアップグレードします。 3. 適切な GRUB イメージを選択します (hd0 – 第 1 ドライブまたは hd00 – 第 1 ドライブ、第 1 パーティション)。 4. インスタンスを終了し、作成した AMI を使用して新しいインスタンスを起動します。 オプション2: 既存のインスタンスの修正: 1. インスタンスを停止します。 2. ルートファイルシステムをデタッチします。 3. ルートファイルシステムを既知の動作しているインスタンスにアタッチします。 4. ファイルシステムをマウントします。 5. GRUB 設定ファイルを作成します。 6. GRUB のバージョンが、基になるファイルシステムのタイプをサポートしていることを確認し、必要に応じて GRUB をアップグレードします。 7. ファイルシステムをデタッチします。 8. 元のインスタンスにアタッチします。

インスタンスタイプ	操作
	9. カーネル属性を変更して、適切な GRUB イメージを使用します (第 1 ディスクまたは第 1 ディスクの第 1 パーティション)。 10.インスタンスを起動します。
Instance store-Backed	オプション 1: AMI を変更しインスタンスを再作成します。 1. GRUB 設定ファイルを使用して、標準の場所に新しい AMI を作成します (/boot/grub/menu.lst)。 2. 適切な GRUB イメージを選択します (hd0 – 第 1 ドライブまたは hd00 – 第 1 ドライブ、第 1 パーティション)。 3. GRUB のバージョンが、基になるファイルシステムのタイプをサポートしていることを確認し、必要に応じて GRUB をアップグレードします。 4. インスタンスを終了し、作成した AMI を使用して新しいインスタンスを起動します。 オプション 2: インスタンスを終了し、正しいカーネルを指定して新しいインスタンスを起動します。 **Note** 既存のインスタンスからデータを復旧するには、AWS Support にお問い合わせください。

Bringing up interface eth0: Device eth0 has different MAC address than expected, ignoring。(ハードコードされた MAC アドレス)

下記のようなシステムログで、この状態が示されます。

```
...
Bringing up loopback interface:  [  OK  ]

Bringing up interface eth0:  Device eth0 has different MAC address than expected,
 ignoring.
[FAILED]

Starting auditd: [  OK  ]
```

可能性のある原因

AMI 設定にハードコードされたインターフェイス MAC がある

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • AMI を変更してハードコードを削除し、インスタンスを再作成します。 • ハードコードされた MAC アドレスを削除するようにインスタンスを変更します。 または 次の手順に従ってください。 1. インスタンスを停止します。 2. ルートボリュームをデタッチする。 3. ボリュームを別のインスタンスにアタッチし、ハードコードされた MAC アドレスを削除するようにボリュームを変更します。

インスタンスタイプ	操作
	4. 初期インスタンスにボリュームをアタッチします。 5. インスタンスを起動します。
Instance store-Backed	次のいずれかを行ってください。 • ハードコードされた MAC アドレスを削除するようにインスタンスを変更します。 • インスタンスを終了し、新しいインスタンスを起動します。

SELinux ポリシーを読み込めません。Machine is in enforcing mode。Halting now。(SELinux の誤設定)

下記のようなシステムログで、この状態が示されます。

```
audit(1313445102.626:2): enforcing=1 old_enforcing=0 auid=4294967295
Unable to load SELinux Policy. Machine is in enforcing mode. Halting now.
Kernel panic - not syncing: Attempted to kill init!
```

可能性のある原因

SELinux が誤って有効にされた。

- 指定されたカーネルは GRUB でサポートされていません
- フォールバックカーネルが存在しません

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次の手順に従ってください。 1. 障害のあるインスタンスを停止します。

インスタンスタイプ	操作
	2. 障害の起きたインスタンスのルートボリュームをデタッチします。 3. 別の実行中の Linux インスタンスにルートボリュームをアタッチします (リカバリインスタンスとして扱われます)。 4. リカバリインスタンスを接続し、障害の起きたインスタンスのルートボリュームをマウントします。 5. マウントしたルートボリュームの SELinux を無効にします。このプロセスは Linux ディストリビューションによって異なります。詳細については各 OS のドキュメントを参照してください。 **Note** 一部のシステムでは、`/mount_point /etc/sysconfig/selinux` ファイルで `SELINUX=disabled` と設定することで SELinux を無効にできます。*mount_point* 、 は、リカバリインスタンス上の、ボリュームをマウントした場所です。 6. リカバリインスタンスからルートボリュームをアンマウントしてデタッチし、元のインスタンスに再アタッチします。 7. インスタンスを起動します。

インスタンスタイプ	操作
Instance store-Backed	次の手順に従ってください。 1. インスタンスを終了し、新しいインスタンスを起動します。 2. (オプション) データ復旧の技術サポートについては、AWS Support にお問い合わせください。

XENBUS: Timeout connecting to devices (Xenbus タイムアウト)

下記のようなシステムログで、この状態が示されます。

```
Linux version 2.6.16-xenU (builder@xenbat.amazonsa) (gcc version 4.0.1
20050727 (Red Hat 4.0.1-5)) #1 SMP Mon May 28 03:41:49 SAST 2007
...
XENBUS: Timeout connecting to devices!
...
Kernel panic - not syncing: No init found.  Try passing init= option to kernel.
```

可能性のある原因

- ブロックデバイスがインスタンスに接続されていない
- このインスタンスは古いインスタンスカーネルを使用している

推奨する対処

インスタンスタイプ	操作
Amazon EBS-Backed	次のいずれかを行ってください。 • AMI とインスタンスを変更して新しいカーネルを使用し、インスタンスを再作成します。 • インスタンスを再起動します。
Instance store-Backed	次のいずれかを行ってください。

インスタンスタイプ	操作
	• インスタンスを終了します。 • AMI を変更して新しいカーネルを使用し、その AMI を使用して新しいインスタンスを起動します。

接続できないインスタンスのトラブルシューティング

Linux インスタンスに接続できない場合は、次の方法によりトラブルシューティングできます。到達不能な Windows インスタンスのトラブルシューティングについては、「Windows 用 EC2 ユーザーガイド」の「Troubleshoot an unreachable instance」を参照してください。

コンテンツ

インスタンスの再起動

トラブルシューティングにも一般的なインスタンス管理にも、到達できないインスタンスを再起動する方法が重要です。

リセットボタンを押してコンピュータをリセットするように、Amazon EC2 コンソール、CLI、または API を使用して EC2 インスタンスをリセットできます。詳細については、「インスタンスの再起動」を参照してください。

インスタンスコンソール出力

コンソール出力は問題を診断する際に役立つツールで、特に、カーネルの問題やサービス設定の問題のトラブルシューティングを行うときに便利です。これらの問題が発生すると、SSH デーモンの開始前にインスタンスが停止したり、インスタンスに到達不能になったりする可能性があります。

Linux/Unix の場合、マシンに接続されている物理的なモニターに通常表示されるようなコンソール出力がインスタンスコンソール出力に表示されます。コンソール出力は、インスタンス遷移状態 (開

始、停止、再起動、終了) の直後に投稿されたバッファされた情報を返します。表示される出力は、継続的には更新されず、更新する価値があると思われる場合にのみ更新されます。

オプションで、インスタンスのライフサイクル中に最新のシリアルコンソールの出力をいつでも取得できます。このオプションは Nitro System 上に構築されたインスタンス でのみサポートされています。Amazon EC2 コンソールではサポートされていません。

> **Note**
>
> 表示される出力のうち、保存されるのは最新の64 KB のみです。この出力は、出力の送信から少なくとも1時間使用可能です。

インスタンスの所有者のみがコンソール出力にアクセスできます。

コンソール出力を取得するには、以下のいずれかの方法を使用します。

Console

コンソール出力を取得するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインで、[Instances] (インスタンス) をクリックします。
3. インスタンスを選択してから、[アクション]、[モニタリングとトラブルシューティング]、[システムログを取得] を選択します。

Command line

コンソール出力を取得するには

次のいずれかのコマンドを使用できます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- get-console-output (AWS CLI)
- Get-EC2ConsoleOutput (AWS Tools for Windows PowerShell)

一般的なシステムログエラーの詳細については、Linux ベースのインスタンスに関するシステムログエラーのトラブルシューティング を参照してください。

接続できないインスタンスのスクリーンショットの取得

SSH を介して Windows インスタンスにアクセスできない場合は、インスタンスのスクリーンショットをキャプチャして、それをイメージとして表示することができます。イメージによってインスタンスのステータスが可視化されるため、迅速にトラブルシューティングを行うことができます。

インスタンスの実行中またはクラッシュ後にスクリーンショットを生成できます。イメージは JPG 形式で生成され、100 KB 未満です。スクリーンショットにはデータ転送コストがかかりません。

制約事項

この機能は、以下の場合はサポートされません。

- ベアメタルインスタンス (タイプ `*.metal` のインスタンス)
- インスタンスで NVIDIA GRID ドライバーが使用されている
- Arm ベースの Graviton プロセッサを搭載したインスタンス

サポートされるリージョン

この機能は以下の リージョンで利用できます。

- US East (N. Virginia) Region
- US East (Ohio) Region
- 米国西部 (北カリフォルニア) リージョン
- 米国西部 (オレゴン) リージョン
- アフリカ (ケープタウン) リージョン
- アジアパシフィック (香港) リージョン
- アジアパシフィック (ハイデラバード) リージョン
- アジアパシフィック (ジャカルタ) リージョン
- アジアパシフィック (メルボルン) リージョン
- アジアパシフィック (ムンバイ) リージョン
- アジアパシフィック (大阪) リージョン
- Asia Pacific (Seoul) Region
- アジアパシフィック (シンガポール) リージョン

- アジアパシフィック (シドニー) リージョン
- アジアパシフィック (東京) リージョン
- カナダ (中部) リージョン
- カナダ西部 (カルガリー) リージョン
- 中国 (北京) リージョン
- 中国 (寧夏) リージョン
- 欧州 (フランクフルト) リージョン
- 欧州 (アイルランド) リージョン
- 欧州 (ロンドン) リージョン
- 欧州 (ミラノ) リージョン
- 欧州 (パリ) リージョン
- 欧州 (スペイン) リージョン
- 欧州 (ストックホルム) リージョン
- 欧州 (チューリッヒ) リージョン
- イスラエル (テルアビブ) リージョン
- 南米 (サンパウロ) リージョン
- 中東 (バーレーン) リージョン
- 中東 (アラブ首長国連邦) リージョン

Console

インスタンスのスクリーンショットを取得するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左のナビゲーションペインの [インスタンス] を選択します。
3. キャプチャするインスタンスを選択します。
4. [アクション]、[モニタリングとトラブルシューティン]、[インスタンスのスクリーンショットの取得] の順に選択します。
5. [ダウンロード] を選択するか、イメージを右クリックしてダウンロードして保存します。

Command line

インスタンスのスクリーンショットをキャプチャするには

次のいずれかのコマンドを使用できます。返されるコンテンツは base64 でエンコードされます。これらのコマンドラインインターフェイスの詳細については、Amazon EC2 へのアクセスを参照してください。

- get-console-screenshot (AWS CLI)
- GetConsoleScreenshot (Amazon EC2 クエリ API)

ホストコンピュータに障害が発生した場合のインスタンスの復旧

基になるホストコンピュータのハードウェアで復旧不可能な問題が発生した場合、AWS はインスタンスの停止イベントをスケジュールすることがあります。このようなイベントは事前に E メールで通知されます。

障害が発生したホストコンピュータで実行されている Amazon EBS-backed インスタンスを復旧するには

1. インスタンスストアボリュームの重要なデータを Amazon EBS または Amazon S3 にバックアップします。
2. インスタンスを停止します。
3. インスタンスを起動します。
4. 重要なデータを復元します。

詳細については、「インスタンスの停止と起動」を参照してください。

障害が発生したホストコンピュータで実行されている Instance-store Backed インスタンスを復旧するには

1. インスタンスから AMI を作成します。
2. イメージを Amazon S3 にアップロードします。
3. 重要なデータを Amazon EBS または Amazon S3 にバックアップします。
4. インスタンスを終了します。
5. AMI から新しいインスタンスを起動します。
6. 重要なデータを新しいインスタンスに復元します。

詳細については、「instance store-backed Linux AMI を作成する」を参照してください。

間違ったボリュームからの起動

状況によっては、/dev/xvda または /dev/sda にアタッチしたボリューム以外のボリュームが、インスタンスのルートボリュームになっている場合があります。これは、別のインスタンスのルートボリュームや、ルートボリュームのスナップショットから作成されたボリュームを、既存のルートボリュームのインスタンスにアタッチした場合に起こります。

これは Linux の初期ラムディスクの挙動です。/ で /etc/fstab として定義されたボリュームを選択した場合でも、一部のディストリビューションでは、ボリュームパーティションにアタッチされたラベルによってボリュームが決定されます。たとえば、/etc/fstab の内容が次のとおりであったとします。

```
LABEL=/ / ext4 defaults,noatime 1 1
tmpfs /dev/shm tmpfs defaults 0 0
devpts /dev/pts devpts gid=5,mode=620 0 0
sysfs /sys sysfs defaults 0 0
proc /proc proc defaults 0 0
```

両方のボリュームのラベルを確認すれば、両方に / ラベルが含まれることが判ります。

```
[ec2-user ~]$ sudo e2label /dev/xvda1
/
[ec2-user ~]$ sudo e2label /dev/xvdf1
/
```

この例では、初期ラムディスクの実行後、意図していた /dev/xvdf1 ボリュームからの起動ではなく、/dev/xvda1 がインスタンスを起動するルートデバイスになる結果となりました。これを解決するために、同じ e2label コマンドを使用して、起動ボリュームではないアタッチ済みのボリュームのラベルを変更できます。

場合によっては、/etc/fstab で UUID を指定することで、この問題を解決できます。ただし、両方のボリュームが同じスナップショットから作成された場合、またはセカンダリボリュームがプライマリボリュームのスナップショットから作成されている場合は、両ボリュームは UUID を共有します。

```
[ec2-user ~]$ sudo blkid
/dev/xvda1: LABEL="/" UUID=73947a77-ddbe-4dc7-bd8f-3fe0bc840778 TYPE="ext4"
 PARTLABEL="Linux" PARTUUID=d55925ee-72c8-41e7-b514-7084e28f7334
```

```
/dev/xvdf1: LABEL="old/" UUID=73947a77-ddbe-4dc7-bd8f-3fe0bc840778 TYPE="ext4"
 PARTLABEL="Linux" PARTUUID=d55925ee-72c8-41e7-b514-7084e28f7334
```

アタッチされた ext4 ボリュームのラベルを変更するには

1. e2label コマンドを使用して、ボリュームのラベルを / 以外のものに変更します。

```
[ec2-user ~]$ sudo e2label /dev/xvdf1 old/
```

2. ボリュームに新しいラベルがあることを確認します。

```
[ec2-user ~]$ sudo e2label /dev/xvdf1
old/
```

アタッチされた xfs ボリュームのラベルを変更するには

- xfs_admin コマンドを使用して、ボリュームのラベルを / 以外のものに変更します。

```
[ec2-user ~]$ sudo xfs_admin -L old/ /dev/xvdf1
writing all SBs
new label = "old/"
```

図のようにボリュームラベルを変更した後で、インスタンスを再起動すると、インスタンスの起動時にラムディスクが正しいボリュームを選択するはずです。

> **Important**
>
> 新しいラベルを持つボリュームをデタッチし、別のインスタンスに戻してルートボリュームとして使用する場合は、上記の手順をもう一度実行してラベルを元の値に戻す必要があります。これを行わない場合、ラムディスクがラベル / を持つボリュームを見つけることができないため、別のインスタンスが起動しません。

使用アイテム Linux 用 EC2Rescue

Linux 用 EC2Rescue は、使いやすいオープンソースのツールであり、Amazon EC2 Linux インスタンスで実行し、100 を超えるモジュールのライブラリを使用して一般的な問題を診断およびトラブルシューティングできます。Linux 用 EC2Rescue の汎用ユースケースには、syslog およびパッケージ

マネージャーログの収集、リソース使用状況データの収集、問題のある既知のカーネルパラメーターと一般的な OpenSSH の問題の診断および修復などがあります。

AWSSupport-TroubleshootSSH ランブックは Linux 用 EC2Rescue をインストールし、そのツールを使用して、Linux マシンへの SSH 経由でのリモートからの接続を妨げる、一般的な問題を確認しその修正を試みます。より詳しい情報を確認し、この自動化を実行するには、AWS Support-TroubleshootSSH をご参照ください。

Windows インスタンスを使用する場合には、「EC2Rescue for Windows Server」を参照してください。

目次

Linux 用 EC2Rescue のインストール

Linux 用 EC2Rescue ツールは、次の前提要件を満たす Amazon EC2 Linux インスタンスにインストールできます。

前提条件

- サポートされるオペレーティングシステム
  - Amazon Linux 2
  - Amazon Linux 2016.09+
  - SUSE Linux Enterprise Server 12+
  - RHEL 7+
  - Ubuntu 16.04+
- ソフトウェア要件
  - Python 2.7.9+ または 3.2+

AWSSupport-TroubleshootSSH ランブックは Linux 用 EC2Rescue をインストールし、そのツールを使用して、Linux マシンへの SSH 経由でのリモートからの接続を妨げる、一般的な問題を確認しその修正を試みます。より詳しい情報を確認し、この自動化を実行するには、AWS Support-TroubleshootSSH をご参照ください。

システムに必要な Python バージョンがある場合は、標準ビルドをインストールできます。それ以外の場合は、Python の最小のコピーを含むバンドル済みのビルドをインストールできます。

標準ビルドをインストールするには

1. 作動している Linux インスタンスから、Linux 用 EC2Rescue ツールをダウンロードします。

```
curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.tgz
```

2. (オプション) 先に進む前に、オプションで Linux 用 EC2Rescue インストールファイルの署名を検証できます。詳細については、「(オプション) Linux 用 EC2Rescue の署名を検証する」を参照してください。
3. sha256 ハッシュファイルをダウンロードします。

```
curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.tgz.sha256
```

4. tarball の整合性を確認します。

```
sha256sum -c ec2rl.tgz.sha256
```

5. Tarball を解凍します。

```
tar -xzvf ec2rl.tgz
```

6. ヘルプファイルを表示してインストールを検証します。

```
cd ec2rl-<version_number>
./ec2rl help
```

バンドル済みのビルドをインストールするには

ダウンロードのリンクと制限については、github の「Linux 用 EC2Rescue」を参照してください。

(オプション) Linux 用 EC2Rescue の署名を検証する

以下に、Linux ベースのオペレーティングシステム用の Linux 用 EC2Rescue パッケージの有効性を検証するための推奨されるプロセスを示します。

インターネットからアプリケーションをダウンロードする場合は、ソフトウェア発行元のアイデンティティを認証し、アプリケーションの発行後に改ざん、あるいは破損がないか確認することをお

勧めします。これにより、ウイルスやマルウェアに感染したバージョンのアプリケーションをインストールせずに済みます。

このトピックのステップを実行した後に Linux 用 EC2Rescue のソフトウェアが変更または破損していることが判明した場合は、インストールファイルを実行しないでください。このような場合は Amazon Web Services にご連絡ください。

Linux ベースのオペレーティングシステム用の Linux 用 EC2Rescue ファイルの署名には、GnuPG が使用されています。これは安全なデジタル署名のための、オープンソース実装のプリティグッドプライバシー (OpenPGP) 標準です。GnuPG (GPG とも呼ばれます) では、デジタル署名を通じて認証と整合性のチェックが行われます。ダウンロードした Linux 用 EC2Rescue パッケージの検証に使用できるパブリックキーと署名は、AWS により公開されています。PGP と GnuPG (GPG) の詳細については、「http://www.gnupg.org」を参照してください。

まず、ソフトウェア発行元との信頼を確立します。ソフトウェア発行元のパブリックキーをダウンロードし、キー所有者が一致していることを確認してから、キーリングに追加します。キーリングとは、既知のパブリックキーの集合です。真正性が確立されたパブリック キーは、アプリケーションの署名を確認するために使用できます。

タスク

- GPG ツールをインストールする
- パブリック キーを認証およびインポートする
- パッケージの署名の確認

GPG ツールをインストールする

お使いのオペレーティングシステムが Linux または Unix の場合、GPG ツールが既にインストールされている場合があります。システムにツールがインストール済みかどうかをテストするには、コマンドラインプロンプトで gpg2 と入力します。GPG ツールがインストールされている場合、GPG のコマンドプロンプトが表示されます。GPG ツールがインストールされていない場合、コマンドが見つからないというエラーが表示されます。GnuPG パッケージはリポジトリからインストールできます。

Debian ベースの Linux に GPG ツールをインストールするには

- ターミナルから、次のコマンドを実行します。

```
apt-get install gnupg2
```

Red Hat ベースの Linux に GPG ツールをインストールするには

- ターミナルから、次のコマンドを実行します。

```
yum install gnupg2
```

パブリック キーを認証およびインポートする

次の手順では、Linux 用 EC2Rescue のパブリックキーを認証し、信頼されたキーとして GPG キーリングへ追加します。

Linux 用 EC2Rescue のパブリックキーを認証してインポートするには

1. コマンドプロンプトで、次のコマンドを使用して当社のパブリック GPG ビルドキーのコピーを取得します。

```
curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.key
```

2. `ec2rl.key` を保存したディレクトリのコマンドプロンプトで、次のコマンドを使用して Linux 用 EC2Rescue のパブリックキーをキーリングにインポートします。

```
gpg2 --import ec2rl.key
```

コマンドで次のような結果が返されます。

```
gpg: /home/ec2-user/.gnupg/trustdb.gpg: trustdb created
gpg: key 2FAE2A1C: public key "ec2autodiag@amazon.com <EC2 Rescue for Linux>"
 imported
gpg: Total number processed: 1
gpg:               imported: 1  (RSA: 1)
```

パッケージの署名の確認

GPG ツールをインストール後、Linux 用 EC2Rescue パブリックキーを認証してインポートし、Linux 用 EC2Rescue パブリック キーが信頼済みであることを確認すると、Linux 用 EC2Rescue インストールスクリプトの署名を確認できるようになります。

Linux 用 EC2Rescue インストールスクリプトの署名を確認するには

1. コマンド プロンプトで次のコマンドを実行し、インストール スクリプトの署名ファイルをダウンロードします。

```
curl -O https://s3.amazonaws.com/ec2rescuelinux/ec2rl.tgz.sig
```

2. ec2rl.tgz.sig と Linux 用 EC2Rescue インストールファイルを保存したディレクトリのコマンドプロンプトで次のコマンドを実行し、署名を確認します。ファイルが2つとも存在している必要があります。

```
gpg2 --verify ./ec2rl.tgz.sig
```

出力は次のようになります。

```
gpg: Signature made Thu 12 Jul 2018 01:57:51 AM UTC using RSA key ID 6991ED45
gpg: Good signature from "ec2autodiag@amazon.com <EC2 Rescue for Linux>"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: E528 BCC9 0DBF 5AFA 0F6C  C36A F780 4843 2FAE 2A1C
     Subkey fingerprint: 966B 0D27 85E9 AEEC 1146  7A9D 8851 1153 6991 ED45
```

出力に「Good signature from "ec2autodiag@amazon.com <EC2 Rescue for Linux>"」という句が含まれる場合は、署名が正常に確認されており、Linux 用 EC2Rescue のインストールスクリプトを実行できることを意味しています。

出力結果に「BAD signature」という句が含まれる場合、手順が正しいことをもう一度確認してください。この応答が続く場合は、Amazon Web Services に連絡してください。以前にダウンロードしたインストール ファイルを実行しないでください。

以下は、表示される可能性のある警告の詳細です。

- WARNING: This key is not certified with a trusted signature! There is no indication that the signature belongs to the owner. これは、Linux 用 EC2Rescue の認証済みパブリック キーを所有していると考えるユーザーの個人レベルの信頼を参照します。本来は、ユーザーが Amazon Web Services オフィスを訪問してキーを受け取ることが理想的です。しかし、キーは多くの場合 ウェブ サイトからダウンロードされます。この場合、ウェブサイトは Amazon Web Services ウェブサイトです。

- gpg2: no ultimately trusted keys found. これは、特定のキーがユーザー (またはユーザーが信頼する他のユーザー) によって「最終的に信頼された」キーでないことを意味します。

詳細については、「http://www.gnupg.org」を参照してください。

Linux 用 EC2Rescue の操作

ここでは、このツールを使い始めるために実行できる一般的なタスクについて説明します。

タスク

- Run EC2Rescue for Linux
- 結果のアップロード
- バックアップの作成
- ヘルプの表示

Run EC2Rescue for Linux

次の例に示すように Linux 用 EC2Rescue を実行できます。

Example 例: すべてのモジュールを実行します

すべてのモジュールを実行するには、Linux 用 EC2Rescue をオプションを指定せずに実行します。

```
./ec2rl run
```

一部のモジュールには、ルートアクセスが必要です。ルートユーザーではない場合は、以下のように sudo を使用してこれらのモジュールを実行します。

```
sudo ./ec2rl run
```

Example 例: 特定のモジュールの実行

特定のモジュールのみ実行するには、--only-modules パラメータを使用します。

```
./ec2rl run --only-modules=module_name --arguments
```

たとえば、このコマンドは、dig モジュールを実行して、amazon.com ドメインに対してクエリを実行します。

```
./ec2rl run --only-modules=dig --domain=amazon.com
```

Example 例: 結果の表示

結果を /var/tmp/ec2rl に表示できます。

```
cat /var/tmp/ec2rl/logfile_location
```

例: dig モジュールのログファイルを表示する場合

```
cat /var/tmp/ec2rl/2017-05-11T15_39_21.893145/mod_out/run/dig.log
```

結果のアップロード

S3 バケットからの結果あるいはその共有を、AWS Support によりリクエストされている場合には、Linux 用 EC2Rescue CLI ツールを使用してそれらをアップロードします。Linux 用 EC2Rescue コマンドの出力によって、使用する必要があるコマンドが提供されます。

Example 例: 結果を AWS Support にアップロードする

```
./ec2rl upload --upload-directory=/var/tmp/ec2rl/2017-05-11T15_39_21.893145 --support-
url="URLProvidedByAWSSupport"
```

Example 例: S3 バケットに結果をアップロードする

```
./ec2rl upload --upload-directory=/var/tmp/ec2rl/2017-05-11T15_39_21.893145 --
presigned-url="YourPresignedS3URL"
```

Amazon S3 に署名付きの URL を生成するための詳細については、「署名付き URL を使用したオブジェクトのアップロード」を参照してください。

バックアップの作成

次のコマンドを使用して、インスタンス、1 つ以上のボリューム、または特定のデバイス ID のバックアップを作成します。

Example 例: Amazon マシンイメージ (AMI) を使用してインスタンスをバックアップする

```
./ec2rl run --backup=ami
```

Example 例: インスタンスに関連付けられるすべてのボリュームのバックアップを作成する

```
./ec2rl run --backup=allvolumes
```

Example 例: 特定のボリュームをバックアップする

```
./ec2rl run --backup=volumeID
```

ヘルプの表示

Linux 用 EC2Rescue には、詳細を説明したヘルプファイルと利用できる各コマンドの構文が含まれています。

Example 例: 全般的なヘルプの表示

```
./ec2rl help
```

Example 例: 利用できるモジュールを一覧表示する

```
./ec2rl list
```

Example 例: 特定のモジュールのヘルプを表示する

```
./ec2rl help module_name
```

たとえば、dig モジュールのヘルプファイルを表示するには、以下のコマンドを使用します。

```
./ec2rl help dig
```

EC2Rescue モジュールの開発

モジュールは、データシリアル化スタンダードである YAML デ書き込まれます。モジュールの YAML ファイルは、モジュールとその属性を示す単一のドキュメントで構成されます。

モジュール属性の追加

次の表には、利用できるモジュールの属性が一覧表示されます。

属性	説明
name	モジュールの名前。この名前は、長さが 18 文字以下である必要があります。
version	モジュールのバージョン番号。
タイトル	モジュールの短い説明タイトルです。この値は、長さが 50 文字以下である必要があります。

helptext	モジュールの拡張された説明。各列は、長さが 75 文字以下である必要があります。必須あるいはオプションでモジュールが引数を消費する場合、helptext 値にこの引数を含めます。
	
	次に例を示します。
	
	`helptext: !!str |`
	`  Collect output from ps for system analysis`
	`  Consumes --times= for number of times to repeat`
	`  Consumes --period= for time period between repetition`

placement	モジュールが実行されるべきステージ。サポートされる値。 • prediagnostic • run • postdiagnostic
language	モジュールコードが書き込まれている言語。サポートされる値。 • bash • python

属性	説明
	Note Python コードは、Python 2.7.9+ および Python 3.2+ の両方と互換性がある必要があります。
修復	モジュールが修復をサポートするかどうかを示します。サポートされている値は `True` または `False` です。 この値がない場合、モジュールのデフォルトは `False` です。修復をサポートしないそれらのモジュールのオプション属性となります。
コンテンツ	全スクリプトコード。
制約	制約値を含むオブジェクトの名前。
ドメイン	モジュールがどのようにグループ化または分類されているかの説明。含まれているモジュール一連は次のドメインを使用します。 • 同時接続の • net • os • パフォーマンス
class	モジュールによって実行されるタスクの種類の説明。含まれているモジュール一連は次のクラスを使用します。 • 回収 (プログラムからの出力を回収します) • 診断 (一連の基準の達成/未達成) • 収集 (ファイルのコピーと特定のファイルへの書き込み)

属性	説明
distro	このモジュールがサポートする Linux ディストリビューションの一覧。含まれているモジュール一連は次のディストリビューションを使用します。 • alami (Amazon Linux) • rhel • Ubuntu • suse
必須	CLI オプションからモジュールが消費する必要な引数。
optional	モジュールが使用できるオプションの引数。
ソフトウェア	モジュールで使用される実行可能なソフトウェア。この属性は、デフォルトでインストールされないソフトウェアの特定を行います。Linux 用 EC2Rescue ロジックは、モジュールを実行する前に、このプログラムが存在し、実行可能であることを確認します。
package	実行ファイル用のソースソフトウェアパッケージ。この属性は、ソフトウェアのパッケージにダウンロード用 URL やそのほかの詳細などの詳しい情報を提供するためのものです。
sudo	ルートアクセスがモジュールの実行に必要であるかどうかを示します。 モジュールスクリプトで sudo チェックを行う必要はありません。値が true になると、Linux 用 EC2Rescue ロジックは実行しているユーザーがルートアクセスを所持している場合にのみモジュールを実行します。

属性	説明
perfimpact	モジュールが実行している環境に重要な影響を及ぼす可能性があるかどうかを示します。値が true であり、`--perfimpact=true` 引数が存在しない場合、モジュールはスキップされます。
parallelexclusive	相互占有を必要とするプログラムを特定します。たとえば、「bpf」を指定するすべてのモジュールはシリアル方法で実行します。

環境変数の追加

次の表には、利用できるモジュールの属性が一覧表示されます。

環境変数	説明
`EC2RL_CALLPATH`	`ec2rl.py` へのパス。このパスを使用すると、lib ディレクトリを見つけて、ベンダーの Python モジュールを使用できます。
`EC2RL_WORKDIR`	診断ツールの主要な tmp ディレクトリ。 デフォルト値: `/var/tmp/ec2rl` 。
`EC2RL_RUNDIR`	すべての出力が保存されているディレクトリ。 デフォルト値: `/var/tmp/ec2rl/<date&timestamp>` 。
`EC2RL_GATHEREDDIR`	収集されたモジュールデータを配置するルートディレクトリ。 デフォルト値:`/var/tmp/ec2rl/<date&timestamp>/mod_out/gathered/` 。

環境変数	説明
EC2RL_NET_DRIVER	初めて使用されるドライバーが、インスタンスの非仮想ネットワークインターフェースでアルファベット順に順序付けされます。 例: • xen_netfront • ixgbevf • ena
EC2RL_SUDO	Linux 用 EC2Rescue がルートとして実行されている場合には true、そうでない場合には false。
EC2RL_VIRT_TYPE	インスタンスメタデータから提供される仮想化タイプ。 例: • default-hvm • default-paravirtual
EC2RL_INTERFACES	システム上のインターフェースの列挙一覧。この値は、eth0 や eth1 などの名前が含まれる文字列です。これは functions.bash を介して生成され、これをソースとするモジュールのみで利用できます。

YAML 構文の使用

モジュール YAML ファイルを構築する際、以下に注意してください。

- 3 つのハイフン (---) は、ドキュメントの明示的な開始を示します。
- !ec2rlcore.module.Module タグは、データストリームからオブジェクトを作成する際にどのコンストラクタを呼び出すかを YAML パーサーに伝えます。module.py ファイル内コンストラクタを検索できます。

- !!str タグは、データの種類を決定する試行を行なわず、代わりにコンテンツを文字列リテラルとして解釈するように YAML パーサーに伝えます。
- パイプ文字 (|) は、値がリテラル形式のスカラーであることを YAML パーサーに伝えます。この場合、パーサーにはすべての空白が含まれます。インデントと改行文字が保持されるため、これはモジュールにとって重要です。
- YAML スタンダードインデントは 2 つのスペースとなり、次の例で示されます。スクリプトでスタンダードインデント (たとえば、Python では 4 つの空白) を維持して、モジュールファイル内で全コンテンツを 2 つのスペースでインデントすることを確認します。

モジュールの例

例 1 (mod.d/ps.yaml):

```
--- !ec2rlcore.module.Module
# Module document. Translates directly into an almost-complete Module object
name: !!str ps
path: !!str
version: !!str 1.0
title: !!str Collect output from ps for system analysis
helptext: !!str |
  Collect output from ps for system analysis
  Requires --times= for number of times to repeat
  Requires --period= for time period between repetition
placement: !!str run
package:
  - !!str
language: !!str bash
content: !!str |
  #!/bin/bash
  error_trap()
  {
      printf "%0.s=" {1..80}
      echo -e "\nERROR: "$BASH_COMMAND" exited with an error on line ${BASH_LINENO[0]}"
      exit 0
  }
  trap error_trap ERR

  # read-in shared function
  source functions.bash
  echo "I will collect ps output from this $EC2RL_DISTRO box for $times times every
 $period seconds."
```

```
  for i in $(seq 1 $times); do
      ps auxww
      sleep $period
  done
constraint:
  requires_ec2: !!str False
  domain: !!str performance
  class: !!str collect
  distro: !!str alami ubuntu rhel suse
  required: !!str period times
  optional: !!str
  software: !!str
  sudo: !!str False
  perfimpact: !!str False
  parallelexclusive: !!str
```

Linux インスタンス用 EC2 シリアルコンソール

EC2 シリアルコンソールを使用すると、Amazon EC2 インスタンスのシリアルポートにアクセスできます。このシリアルポートを使用して、起動、ネットワーク設定、およびその他の問題をトラブルシューティングできます。シリアルコンソールでは、インスタンスにネットワーク機能を持たせる必要はありません。シリアルコンソールを使用すると、キーボードとモニターがインスタンスのシリアルポートに直接接続されているかのように、インスタンスにコマンドを入力できます。シリアルコンソールセッションは、インスタンスの再起動中および停止中も継続します。再起動中は、起動時のすべてのブートメッセージを表示できます。

デフォルトでは、シリアルコンソールにアクセスできません。組織は、アカウントにシリアルコンソールへのアクセスを許可し、IAM ポリシーを設定して、ユーザーにシリアルコンソールへのアクセスを許可する必要があります。シリアルコンソールへのアクセスは、インスタンス ID、リソースタグ、その他の IAM レバーを使用して、きめ細かいレベルで制御できます。詳細については、「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。

シリアルコンソールには、EC2 コンソールまたは AWS CLI を使用してアクセスできます。

シリアルコンソールは追加料金なしで利用できます。

Windows インスタンスを使用している場合は、Windows インスタンスの Amazon EC2 ユーザーガイド の「Windows インスタンス用の EC2 シリアルコンソール」を参照してください。

トピック

- 前提条件

- EC2 シリアルコンソールへのアクセスを設定する
- EC2 シリアルコンソールに接続する
- EC2 シリアルコンソールからの切断
- EC2 シリアルコンソールを使用して Linux インスタンスをトラブルシューティングする

前提条件

EC2 シリアルコンソールに接続し、選択したツールでトラブルシューティングを行うには、以下の前提条件が満たされている必要があります。

- AWS リージョン
- Wavelength ゾーンと AWS Outposts
- ローカルゾーン
- インスタンスのタイプ
- アクセス権を付与する
- ブラウザベースのクライアントのサポート
- インスタンスの状態
- Amazon EC2 Systems Manager
- sshd サーバー
- トラブルシューティングツールを選択

AWS リージョン

カナダ西部 (カルガリー) を除くすべての AWS リージョン でサポートされています。

Wavelength ゾーンと AWS Outposts

サポート外。

ローカルゾーン

すべての Local Zones ではサポートされています。

インスタンスのタイプ

サポートされるインスタンスタイプ:

- Nitro システム上に構築されたすべての仮想化インスタンス。
- 以下を除くすべてのベアメタルインスタンス:
  - 汎用: a1.metal, mac1.metal, mac2.metal
  - 高速コンピューティング: g5g.metal
  - メモリ最適化: u-6tb1.metal, u-9tb1.metal, u-12tb1.metal, u-18tb1.metal, u-24tb1.metal

アクセス権を付与する

EC2 シリアルコンソールへのアクセス権を付与する設定タスクを完了する必要があります。詳細については、「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。

ブラウザベースのクライアントのサポート

ブラウザベースのクライアントを使用してシリアルコンソールに接続するには、そのブラウザで WebSocket をサポートしている必要があります。お使いのブラウザが WebSocket をサポートしていない場合は、独自のキーとSSH クライアントを使用してシリアルコンソールに接続します。

インスタンスの状態

running を指定してください。

インスタンスが pending、stopping、stopped、shutting-down、または terminated 状態の場合、シリアルコンソールに接続できません。

インスタンスステータスの詳細については、「インスタンスのライフサイクル」を参照してください。

Amazon EC2 Systems Manager

インスタンスで Amazon EC2 Systems Manager を使用する場合は、SSM Agent のバージョン 3.0.854.0 以降を、そのインスタンスにインストールする必要があります。SSM Agent の詳細については、AWS Systems Manager ユーザーガイドの「SSM Agentを使用する」を参照してください。

sshd サーバー

インスタンスで、sshd サーバーをインストールまたは実行する必要はありません。

トラブルシューティングツールを選択

シリアルコンソールから Linux インスタンスのトラブルシューティングを行うには、GRUB または SysRq を使用します。これらのツールを使用できるようにするには、ツールを使用するすべてのインスタンスで設定手順を実行する必要があります。

選択したトラブルシューティングツールを Windows で設定する手順については、「Amazon EC2 Windows インスタンス用ユーザーガイド」の「選択したトラブルシューティングツールを設定する」を参照してください。

ツール

- GRUB を設定する
- SysRq を設定する

GRUB を設定する

シリアルコンソールで GRUB を使用する前に、シリアルコンソールから GRUB を使用するようにインスタンスを設定する必要があります。

GRUB を設定するには、インスタンスの起動に使用された AMI に基づいて、次のいずれかの手順を選択します。

Amazon Linux 2

Amazon Linux 2 インスタンスで GRUB を設定するには

1. Linux インスタンスへの接続
2. `/etc/default/grub` で、次のオプションを追加または変更します。
   - `GRUB_TIMEOUT=1` を設定します。
   - `GRUB_TERMINAL="console serial"` を追加する。
   - `GRUB_SERIAL_COMMAND="serial --speed=115200"` を追加する。

   `/etc/default/grub` の例を次に示します。場合によっては、システム設定に基づいて設定を変更することが必要な場合があります。

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8 net.ifnames=0
 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff rd.shell=0"
GRUB_TIMEOUT=1
```

```
GRUB_DISABLE_RECOVERY="true"
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed=115200"
```

3. 次のコマンドを実行して、更新された設定を適用します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

Ubuntu

Ubuntu インスタンスで GRUB を設定するには

1. インスタンスに接続します。
2. /etc/default/grub.d/50-cloudimg-settings.cfg で、次のオプションを追加または変更します。

   - GRUB_TIMEOUT=1 を設定します。
   - GRUB_TIMEOUT_STYLE=menu を追加する。
   - GRUB_TERMINAL="console serial" を追加する。
   - GRUB_HIDDEN_TIMEOUT を削除します。
   - GRUB_SERIAL_COMMAND="serial --speed=115200" を追加する。

   /etc/default/grub.d/50-cloudimg-settings.cfg の例を次に示します。場合によっては、システム設定に基づいて設定を変更することが必要な場合があります。

```
# Cloud Image specific Grub settings for Generic Cloud Images
# CLOUD_IMG: This file was created/modified by the Cloud Image build process

# Set the recordfail timeout
GRUB_RECORDFAIL_TIMEOUT=0

# Do not wait on grub prompt
GRUB_TIMEOUT=1
GRUB_TIMEOUT_STYLE=menu

# Set the default commandline
GRUB_CMDLINE_LINUX_DEFAULT="console=tty1 console=ttyS0
 nvme_core.io_timeout=4294967295"
```

```
# Set the grub console type
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed 115200"
```

3. 次のコマンドを実行して、更新された設定を適用します。

```
[ec2-user ~]$ sudo update-grub
```

RHEL

RHEL インスタンスで GRUB を設定するには

1. インスタンスに接続します。
2. /etc/default/grub で、次のオプションを追加または変更します。
   - GRUB_TERMINAL_OUTPUT を削除します。
   - GRUB_TERMINAL="console serial" を追加する。
   - GRUB_SERIAL_COMMAND="serial --speed=115200" を追加する。

   /etc/default/grub の例を次に示します。場合によっては、システム設定に基づいて設定を変更することが必要な場合があります。

```
GRUB_TIMEOUT=1
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_CMDLINE_LINUX="console=tty0 console=ttyS0,115200n8 net.ifnames=0
 rd.blacklist=nouveau nvme_core.io_timeout=4294967295 crashkernel=auto"
GRUB_DISABLE_RECOVERY="true"
GRUB_ENABLE_BLSCFG=true
GRUB_TERMINAL="console serial"
GRUB_SERIAL_COMMAND="serial --speed=115200"
```

3. 次のコマンドを実行して、更新された設定を適用します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

CentOS

CentOS AMI を使用して起動されるインスタンスの場合、GRUB はデフォルトでシリアルコンソール用に設定されます。

`/etc/default/grub` の例を次に示します。構成は、システム設定によって異なる場合があります。

```
GRUB_TIMEOUT=1
GRUB_DISTRIBUTOR="$(sed 's, release .*$,,g' /etc/system-release)"
GRUB_DEFAULT=saved
GRUB_DISABLE_SUBMENU=true
GRUB_TERMINAL="serial console"
GRUB_SERIAL_COMMAND="serial --speed=115200"
GRUB_CMDLINE_LINUX="console=tty0 crashkernel=auto console=ttyS0,115200"
GRUB_DISABLE_RECOVERY="true"
```

SysRq を設定する

SysRq を設定するには、現在のブートサイクルで SysRq コマンドを有効にします。設定を永続化するために、後続の起動で SysRq コマンドを有効にすることもできます。

現在のブートサイクルについてすべての SysRq コマンドを有効にするには

1. インスタンスに接続します。
2. 次のコマンドを実行します。

```
[ec2-user ~]$ sudo sysctl -w kernel.sysrq=1
```

> **Note**
>
> この設定は、次回の再起動時にクリアされます。

後続の起動についてすべての SysRq コマンドを有効にするには

1. `/etc/sysctl.d/99-sysrq.conf` ファイルを作成し、お気に入りのエディタで開きます。

```
[ec2-user ~]$ sudo vi /etc/sysctl.d/99-sysrq.conf
```

2. 次の行を追加します。

```
kernel.sysrq=1
```

3. インスタンスを再起動して、変更を適用します。

```
[ec2-user ~]$ sudo reboot
```

4. login プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
5. Password プロンプトで、パスワードを入力し、Enter キーを押します。

EC2 シリアルコンソールへのアクセスを設定する

シリアルコンソールへのアクセスを設定するには、アカウントレベルでシリアルコンソールへのアクセスを許可し、ユーザーにアクセス権を付与するように IAM ポリシーを設定する必要があります。また、ユーザーがトラブルシューティングのためにシリアルコンソールを使用できるように、すべてのインスタンスでパスワードベースのユーザーを設定する必要があります。

開始する前に、前提条件を必ず確認してください。

トピック

- EC2 シリアルコンソールへのアクセスのレベル
- EC2 シリアルコンソールへのアカウントアクセスを管理する
- EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する
- OS のユーザーパスワードを設定する

EC2 シリアルコンソールへのアクセスのレベル

デフォルトでは、アカウントレベルでシリアルコンソールにアクセスすることはできません。アカウントレベルでシリアルコンソールへのアクセスを明示的に許可する必要があります。詳細については、「EC2 シリアルコンソールへのアカウントアクセスを管理する」を参照してください。

サービスコントロールポリシー (SCP) を使用して、組織内でシリアルコンソールへのアクセスを許可できます。その後、IAM ポリシーを使用してアクセスをコントロールすることで、ユーザーレベルできめ細かいアクセスコントロールを行うことができます。SCP ポリシーと IAM ポリシーを組み

合わせて使用することで、シリアルコンソールに対するさまざまなレベルのアクセス制御が可能になります。

組織レベル

サービスコントロールポリシー (SCP) を使用して、組織内のメンバーアカウントのためにシリアルコンソールへのアクセスを許可できます。SCP の詳細については、AWS Organizations ユーザーガイド の「サービスコントロールポリシー」を参照してください。

インスタンスレベル

IAM PrincipalTag および ResourceTag 構造を使用し、ID でインスタンスを指定することで、シリアルコンソールのアクセスポリシーを設定できます。詳細については、「EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する」を参照してください。

ユーザーレベル

特定のインスタンスのシリアルコンソールサービスに SSH パブリックキーをプッシュするためのアクセス権限を指定ユーザーに許可または拒否するように IAM ポリシーを設定することで、ユーザーレベルでアクセスを設定できます。詳細については、「EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する」を参照してください。

OS レベル

ユーザーパスワードは、ゲスト OS レベルで設定できます。これにより、一部のユースケースのためにシリアルコンソールにアクセス権を付与します。ただし、ログをモニタリングするには、パスワードベースのユーザーは必要ありません。詳細については、「OS のユーザーパスワードを設定する」を参照してください。

EC2 シリアルコンソールへのアカウントアクセスを管理する

デフォルトでは、アカウントレベルでシリアルコンソールにアクセスすることはできません。アカウントレベルでシリアルコンソールへのアクセスを明示的に許可する必要があります。

トピック

- ユーザーにアカウントアクセスを管理するための許可を付与する
- シリアルコンソールへのアカウントアクセスのステータスを表示する
- シリアルコンソールへのアカウントアクセスを許可する
- シリアルコンソールへのアカウントアクセスを拒否する

ユーザーにアカウントアクセスを管理するための許可を付与する

ユーザーが EC2 シリアルコンソールへのアカウントアクセスを管理できるようにするには、必要な IAM 許可をユーザーに付与する必要があります。

次のポリシーは、アカウントステータスを表示するためのアクセス権限、ならびに EC2 シリアルコンソールへのアカウントアクセスを許可および禁止するためのアクセス権限を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:GetSerialConsoleAccessStatus",
                "ec2:EnableSerialConsoleAccess",
                "ec2:DisableSerialConsoleAccess"
            ],
            "Resource": "*"
        }
    ]
}
```

詳細については、IAM ユーザーガイド の「IAM ポリシーの作成」を参照してください。

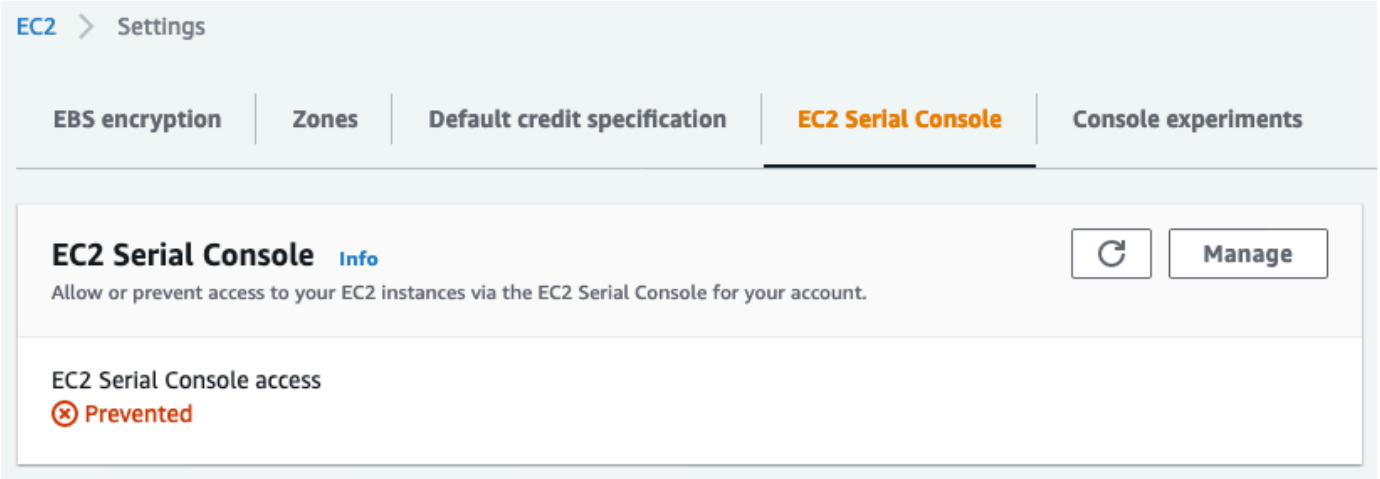
シリアルコンソールへのアカウントアクセスのステータスを表示する

シリアルコンソール (コンソール) へのアカウントアクセスのステータスを表示するには

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側ナビゲーションペインで、[EC2 ダッシュボード] をクリックします。
3. [Account attributes] (アカウントの属性) から、[EC2 Serial Console] (EC2 シリアルコンソール) を選択します。

   [EC2 Serial Console access] (EC2 シリアルコンソールアクセス) フィールドには、アカウントアクセスが [Allowed] (許可) されているか、[Prevented] (禁止) されているかが示されます。

   次のスクリーンショットは、アカウントが EC2 シリアルコンソールを使用できないことを示しています。



シリアルコンソールへのアカウントアクセスのステータスを表示するには (AWS CLI)

シリアルコンソールへのアカウントアクセスのステータスを表示するには、get-serial-console-access-status コマンドを使用します。

```
aws ec2 get-serial-console-access-status --region us-east-1
```

次の出力では、true は、アカウントがシリアルコンソールへのアクセスを許可されていることを示しています。

```
{
    "SerialConsoleAccessEnabled": true
}
```

シリアルコンソールへのアカウントアクセスを許可する

シリアルコンソールへのアカウントアクセスを許可するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側ナビゲーションペインで、[EC2 ダッシュボード] をクリックします。
3. [Account attributes] (アカウントの属性) から、[EC2 Serial Console] (EC2 シリアルコンソール) を選択します。
4. [管理] をクリックします。
5. アカウント内のすべてのインスタンスの EC2 シリアルコンソールへのアクセスを許可するには、[Allow] (許可) チェックボックスをオンにします。
6. [更新] を選択します。

シリアルコンソールへのアカウントアクセスを許可するには (AWS CLI)

enable-serial-console-access コマンドを使用して、シリアルコンソールへのアカウントアクセスを許可します。

```
aws ec2 enable-serial-console-access --region us-east-1
```

次の出力では、true は、アカウントがシリアルコンソールへのアクセスを許可されていることを示しています。

```
{
    "SerialConsoleAccessEnabled": true
}
```

シリアルコンソールへのアカウントアクセスを拒否する

シリアルコンソールへのアカウントアクセスを拒否するには (コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. 左側ナビゲーションペインで、[EC2 ダッシュボード] をクリックします。
3. [Account attributes] (アカウントの属性) から、[EC2 Serial Console] (EC2 シリアルコンソール) を選択します。
4. [管理] をクリックします。
5. アカウント内のすべてのインスタンスの EC2 シリアルコンソールへのアクセスを禁止するには、[Allow] (許可) チェックボックスをオフにします。
6. [更新] を選択します。

シリアルコンソールへのアカウントアクセスを拒否するには (AWS CLI)

disable-serial-console-access コマンドを使用して、シリアルコンソールへのアカウントアクセスを禁止します。

```
aws ec2 disable-serial-console-access --region us-east-1
```

次の出力では、false は、アカウントがシリアルコンソールへのアクセスを拒否されていることを示しています。

```
{
```

```
    "SerialConsoleAccessEnabled": false
}
```

EC2 シリアルコンソールのアクセスについての IAM ポリシーを設定する

デフォルトでは、ユーザーはシリアルコンソールにアクセスできません。組織は IAM ポリシーを設定して、ユーザーに必要なアクセスを許可する必要があります。詳細については、IAM ユーザーガイド の「IAM ポリシーの作成」を参照してください。

シリアルコンソールのアクセスについて、ec2-instance-connect:SendSerialConsoleSSHPublicKey アクションを含む JSON ポリシードキュメントを作成します。このアクションは、シリアルコンソールセッションを開始するシリアルコンソールサービスにパブリックキーをプッシュするための許可をユーザーに付与します。特定の EC2 インスタンスへのアクセスを制限することをお勧めします。それ以外の場合、この許可を持つすべてのユーザーは、すべての EC2 インスタンスのシリアルコンソールに接続できます。

IAM ポリシーの例

- シリアルコンソールへのアクセスを明示的に許可する
- シリアルコンソールへのアクセスを明示的に拒否する
- リソースタグを使用してシリアルコンソールへのアクセスを制御する

シリアルコンソールへのアクセスを明示的に許可する

デフォルトでは、誰もシリアルコンソールにアクセスできません。シリアルコンソールへのアクセスを許可するには、明示的にアクセスを許可するようにポリシーを設定する必要があります。特定のインスタンスへのアクセスを制限するポリシーを設定することをお勧めします。

次のポリシーは、インスタンス ID によって識別される特定のインスタンスのシリアルコンソールへのアクセスを許可します。

DescribeInstances、DescribeInstanceTypes、GetSerialConsoleAccessStatus アクションはリソースレベルの権限をサポートしていないため、これらのアクションには * (アスタリスク) で示されるすべてのリソースを指定する必要があることに注意してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDescribeInstances",
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
                "ec2:GetSerialConsoleAccessStatus"
            ],
             "Resource": "*"
        },
        {
            "Sid": "AllowinstanceBasedSerialConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/i-0598c7d356eba48d7"
        }
    ]
}
```

シリアルコンソールへのアクセスを明示的に拒否する

次の IAM ポリシーは、* (アスタリスク) で示されるすべてのインスタンスのシリアルコンソールへのアクセスを許可し、ID によって識別される特定のインスタンスのシリアルコンソールへのアクセスを明示的に拒否します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSerialConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
                "ec2:GetSerialConsoleAccessStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "DenySerialConsoleAccess",
            "Effect": "Deny",
            "Action": [
```

```
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey"
            ],
            "Resource": "arn:aws:ec2:region:account-id:instance/i-0598c7d356eba48d7"
        }
    ]
}
```

リソースタグを使用してシリアルコンソールへのアクセスを制御する

リソースタグを使用して、インスタンスのシリアルコンソールへのアクセスを制御できます。

属性ベースアクセス制御は、ユーザーおよび AWS リソースにアタッチできるタグに基づいてアクセス権限を定義する認証戦略です。例えば、次のポリシーは、インスタンスのリソースタグとプリンシパルのタグがタグキーについて同じ SerialConsole の値を持っている場合に限り、ユーザーがインスタンスのシリアルコンソール接続を開始することを許可します。

AWS リソースへのアクセスを制御するタグの使用の詳細については、IAM ユーザーガイドの「AWS リソースへのアクセス制御」を参照してください。

DescribeInstances、DescribeInstanceTypes、GetSerialConsoleAccessStatus アクションはリソースレベルの権限をサポートしていないため、これらのアクションには * (アスタリスク) で示されるすべてのリソースを指定する必要があることに注意してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowDescribeInstances",
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
                "ec2:GetSerialConsoleAccessStatus"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowTagBasedSerialConsoleAccess",
            "Effect": "Allow",
            "Action": [
                "ec2-instance-connect:SendSerialConsoleSSHPublicKey"
            ],
```

```
            "Resource": "arn:aws:ec2:region:account-id:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/SerialConsole":
 "${aws:PrincipalTag/SerialConsole}"
                }
            }
        }
    ]
}
```

OS のユーザーパスワードを設定する

パスワードなしでシリアルコンソールに接続できます。ただし、インスタンスのトラブルシューティングのためにシリアルコンソールを使用するには、インスタンスにパスワードベースの OS ユーザーが必要です。

root ユーザーを含む任意の OS ユーザーに対し、パスワードを設定できます。root ユーザーはすべてのファイルを変更できますが、それ以外の OS ユーザーは、権限が制限されていることに注意してください。

シリアルコンソールを使用するすべてのインスタンスについて、ユーザーパスワードを設定する必要があります。これは、インスタンスごとに 1 回のみ必要なセットアップです。

> **Note**
>
> 以下の手順は、AWS が提供する AMI を使用してインスタンスを起動した場合にのみ適用されます。これは、AWS が提供する AMI はデフォルトでは、パスワードベースのユーザーを使用して設定されていないためです。既にルートユーザーパスワードが設定されている AMI を使用してインスタンスを起動した場合は、これらの手順を省略できます。

OS ユーザーのパスワードを設定するには

1. インスタンスに接続します。EC2 シリアルコンソールの接続方法を除き、インスタンスへの接続には任意の方法を使用できます。
2. ユーザーのパスワードを設定するには、passwd コマンドを使用します。次の例では、ユーザーは `root` です。

```
[ec2-user ~]$ sudo passwd root
```

出力例を次に示します。

```
Changing password for user root.
New password:
```

3. New password のプロンプトに従って、新しいパスワードを入力します。
4. プロンプトに従って、パスワードを再入力します。

EC2 シリアルコンソールに接続する

Amazon EC2 コンソールまたは SSH を使用して EC2 インスタンスのシリアルコンソールに接続できます。シリアルコンソールに接続したら、起動、ネットワーク設定、およびその他の問題のトラブルシューティングに使用できます。トラブルシューティングの詳細については、「EC2 シリアルコンソールを使用して Linux インスタンスをトラブルシューティングする」を参照してください。

考慮事項

- インスタンスごとにサポートされるアクティブなシリアルコンソール接続は 1 つだけです。
- シリアルコンソール接続は、ユーザーが解除しない限り、通常 1 時間継続します。ただし、システムメンテナンス中は、Amazon EC2 によりシリアルコンソールのセッションが切断されます。
- シリアルコンソールから切断した後、新しいセッションを許可するためにセッションを終了処理するには、30 秒かかります。
- Linux: ttyS0 用にサポートされるシリアルコンソールポート
- シリアルコンソールに接続すると、インスタンスのスループットがわずかに低下することがあります。

トピック

- ブラウザベースのクライアントを使用した接続
- 独自のキーと SSH クライアントを使用して接続する
- EC2 シリアルコンソールエンドポイントとフィンガープリント

ブラウザベースのクライアントを使用した接続

ブラウザベースのクライアントを使用して、EC2 インスタンスのシリアルコンソールに接続できます。これを行うには、Amazon EC2 コンソールでインスタンスを選択し、シリアルコンソールへの

接続を選択します。ブラウザベースのクライアントは、アクセス権限を処理し、正常な接続を提供します。

EC2 シリアルコンソールは、ほとんどのブラウザで動作し、キーボードとマウスの入力をサポートしています。

接続する前に、前提条件を満たしていることを確認してください。

ブラウザベースのクライアントを使用してインスタンスのシリアルポートに接続するには (Amazon EC2 コンソール)

1. Amazon EC2 コンソール (https://console.aws.amazon.com/ec2/) を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. インスタンスを選択し、[Actions] (アクション)、[Monitor and troubleshoot] (モニタリングとトラブルシューティング)、[EC2 Serial Console] (EC2 シリアルコンソール)、[Connect] (接続) の順に選択します。

   または、インスタンスを選択し、[Connect] (接続)、[EC2 Serial Console] (EC2 シリアルコンソール)、[Connect] (接続) の順に選択します。

   ブラウザ内ターミナルウィンドウが開きます。
4. Enter キーを押します。ログインプロンプトが返された場合は、シリアルコンソールに接続されています。

   画面が黒いままの場合は、シリアルコンソールへの接続に関する問題の解決に役立てるために次の情報を使用できます。

   - シリアルコンソールへのアクセスが設定されていることを確認します。詳細については、「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。
   - SysRq を使用してシリアルコンソールに接続します。SysRq では、ブラウザベースのクライアント経由で接続する必要はありません。詳細については、「SysRq を使用して Linux インスタンスをトラブルシューティングする」を参照してください。
   - getty を再起動します。インスタンスへの SSH アクセスがある場合は、SSH を使用してインスタンスに接続し、次のコマンドを使用して getty を再起動します。

     ```
     [ec2-user ~]$ sudo systemctl restart serial-getty@ttyS0
     ```

- インスタンスを再起動します。SysRq、EC2 コンソール、または AWS CLI を使用して、インスタンスを再起動できます。詳細については、「SysRq を使用して Linux インスタンスをトラブルシューティングする」または「インスタンスの再起動」を参照してください。

5. `login` プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
6. `Password` プロンプトで、パスワードを入力し、Enter キーを押します。

   これでインスタンスにログオンし、トラブルシューティングにシリアルコンソールを使用できるようになりました。

独自のキーと SSH クライアントを使用して接続する

シリアルコンソール API の使用中に、独自の SSH キーを使用して、選択した SSH クライアントからインスタンスに接続できます。これにより、インスタンスにパブリックキーをプッシュするシリアルコンソール機能を活用できます。

接続する前に、前提条件を満たしていることを確認してください。

SSH を使用してインスタンスのシリアルコンソールに接続するには

1. SSH パブリックキーをインスタンスにプッシュして、シリアルコンソールのセッションを開始する

   send-serial-console-ssh-public-key コマンドを使用して、SSH パブリックキーをインスタンスにプッシュします。これにより、シリアルコンソールのセッションが開始されます。

   このインスタンスについてシリアルコンソールのセッションが既に開始されている場合、一度に 1 つのセッションしか開くことができないため、コマンドは失敗します。シリアルコンソールから切断した後、新しいセッションを許可するためにセッションを終了処理するには、30 秒かかります。

```
$ aws ec2-instance-connect send-serial-console-ssh-public-key \
    --instance-id i-001234a4bf70dec41EXAMPLE \
    --serial-port 0 \
    --ssh-public-key file://my_key.pub \
    --region us-east-1
```

2. プライベートキーを使用してシリアルコンソールに接続する

   パブリックキーをシリアルコンソールサービスから削除する前に、ssh コマンドを使用してシリアルコンソールに接続します。削除されるまで 60 秒かかります。

   パブリックキーに対応するプライベートキーを使用します。

   ユーザー名の形式は `instance-id.port0` であり、インスタンス ID とポート 0 で構成されます。次の例では、ユーザー名は `i-001234a4bf70dec41EXAMPLE.port0` です。

   シリアルコンソールサービスのエンドポイントはリージョンごとに異なります。各リージョンのエンドポイントについては、EC2 シリアルコンソールエンドポイントとフィンガープリント の表を参照してください。次の例では、シリアルコンソールサービスが *us-east-1* リージョンにあります。

   ```
   $ ssh -i my_key i-001234a4bf70dec41EXAMPLE.port0@serial-console.ec2-instance-connect.us-east-1.aws
   ```

3. (オプション) フィンガープリントを検証する

   シリアルコンソールの初回接続時に、フィンガープリントを検証するように求めるメッセージが表示されます。シリアルコンソールのフィンガープリントと、検証のために表示されるフィンガープリントを比較できます。これらのフィンガープリントが一致しない場合、「中間者 (MITM) 」攻撃を受けている可能性があります。一致する場合は、確信をもってシリアルコンソールに接続できます。

   次のフィンガープリントは、us-east-1 リージョンのシリアルコンソールサービス用です。各リージョンのフィンガープリントについては、EC2 シリアルコンソールエンドポイントとフィンガープリント をご参照ください。

   ```
   SHA256:dXwn5ma/xadVMeBZGEru5l2gx+yI5LDiJaLUcz0FMmw
   ```

   > **Note**
   >
   > フィンガープリントは、シリアルコンソールへの初回接続時のみ表示されます。

4. Enter キーを押します。プロンプトが返された場合は、シリアルコンソールに接続されています。

画面が黒いままの場合は、シリアルコンソールへの接続に関する問題の解決に役立てるために次の情報を使用できます。

- シリアルコンソールへのアクセスが設定されていることを確認します。詳細については、「EC2 シリアルコンソールへのアクセスを設定する」を参照してください。
- SysRq を使用してシリアルコンソールに接続します。SysRq では、SSH 経由で接続する必要はありません。詳細については、「SysRq を使用して Linux インスタンスをトラブルシューティングする」を参照してください。
- getty を再起動します。インスタンスへの SSH アクセスがある場合は、SSH を使用してインスタンスに接続し、次のコマンドを使用して getty を再起動します。

```
[ec2-user ~]$ sudo systemctl restart serial-getty@ttyS0
```

- インスタンスを再起動します。SysRq、EC2 コンソール、または AWS CLI を使用して、インスタンスを再起動できます。詳細については、「SysRq を使用して Linux インスタンスをトラブルシューティングする」または「インスタンスの再起動」を参照してください。

5. `login` プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
6. `Password` プロンプトで、パスワードを入力し、Enter キーを押します。

   これでインスタンスにログオンし、トラブルシューティングにシリアルコンソールを使用できるようになりました。

EC2 シリアルコンソールエンドポイントとフィンガープリント

EC2 シリアルコンソールのサービスエンドポイントとフィンガープリントは次のとおりです。インスタンスのシリアルコンソールにプログラムで接続するには、EC2 シリアルコンソールエンドポイントを使用します。EC2 シリアルコンソールエンドポイントとフィンガープリントは、AWS リージョンごとに一意です。

リージョン名	リージョン	エンドポイント	フィンガープリント
米国東部 (オハイオ)	us-east-2	serial-console.ec2-instance-connect.us-east-2.aws	SHA256:EhwPkTzRtTY7TRSzz26XbB0/

リージョン名	リージョン	エンドポイント	フィンガープリント
			HvV9jRM7mCZN0xw/d/0
米国東部 (バージニア北部)	us-east-1	serial-console.ec2-instance-connect.us-east-1.aws	SHA256:dXwn5ma/xadVMeBZGEru5l2gx+yI5LDiJaLUcz0FMmw
米国西部 (北カリフォルニア)	us-west-1	serial-console.ec2-instance-connect.us-west-1.aws	SHA256:OHldlcMET8u7QLSX3jmRTRAPFHVtqbyoLZBMUCqiH3Y
米国西部 (オレゴン)	us-west-2	serial-console.ec2-instance-connect.us-west-2.aws	SHA256:EMCIe23TqKaBI6yGHainqZcMwqNkDhhAVHa1O2JxVUc
アフリカ (ケープタウン)	af-south-1	ec2-serial-console.af-south-1.api.aws	SHA256:RMWWZ2fVePeJUqzjO5jL2KIgXsczoHlz21Ed00biiWI
アジアパシフィック (香港)	ap-east-1	ec2-serial-console.ap-east-1.api.aws	SHA256:T0Q1lpiXxChoZHplnAkjbP7tkm2xXViC9bJFsjYnifk
アジアパシフィック (ハイデラバード)	ap-south-2	ec2-serial-console.ap-south-2.api.aws	SHA256:WJgPBSwV4/shN+OPITValoewAuYj15DVW845JEhDKRs
アジアパシフィック (ジャカルタ)	ap-southeast-3	ec2-serial-console.ap-southeast-3.api.aws	SHA256:5ZwgrCh+lfns32XITqL/4O0zIfbx4bZgsYFqy3o8mIk

リージョン名	リージョン	エンドポイント	フィンガープリント
アジアパシフィック (メルボルン)	ap-southeast-4	ec2-serial-console.ap-southeast-4.api.aws	SHA256:Av aq27hFgLv jn5gTSShZ 0oV7h90p0 GG46wfOeT6ZJvM
アジアパシフィック（ムンバイ）	ap-south-1	serial-console.ec2-instance-connect.ap-south-1.aws	SHA256:oB LXcYmklqH HEbliARxEgH8IsO51r ezTPiSM35BsU40
アジアパシフィック (大阪)	ap-northeast-3	ec2-serial-console.ap-northeast-3.api.aws	SHA256:Am0/ jiBKBnBuFnHr9aXs gEV3G8Tu/ vVHFXE/3UcyjsQ
アジアパシフィック（ソウル）	ap-northeast-2	serial-console.ec2-instance-connect.ap-northeast-2.aws	SHA256:FoqWXNX +DZ++GuNTztg9 PK49WYMqBX +FrcZM2dSrqrI
アジアパシフィック（シンガポール）	ap-southeast-1	serial-console.ec2-instance-connect.ap-southeast-1.aws	SHA256:PL FNn7WnCQD Hx3qmwLu1Gy/ O8TUX7LQgZuaC6L 45CoY
アジアパシフィック（シドニー）	ap-southeast-2	serial-console.ec2-instance-connect.ap-southeast-2.aws	SHA256:yF vMwUK9lEU QjQTRoXXzuN+cW9/ VSe9W984Cf5Tgzo4

リージョン名	リージョン	エンドポイント	フィンガープリント
アジアパシフィック（東京）	ap-northeast-1	serial-console.ec2-instance-connect.ap-northeast-1.aws	SHA256:RQfsDCZTOfQawewTRDV1t9Em/HMrFQe+CRlIOT5um4k
カナダ（中部）	ca-central-1	serial-console.ec2-instance-connect.ca-central-1.aws	SHA256:P2O2jOZwmpMwkpO6YW738FIOTHdUTyEv2gczYMMO7s4
中国 (北京)	cn-north-1	ec2-serial-console.cn-north-1.api.amazonwebservices.com.cn	SHA256:2gHVFy4H7uU3+WaFUxD28v/ggMeqjvSlgngpgLgGT+Y
中国 (寧夏)	cn-northwest-1	ec2-serial-console.cn-northwest-1.api.amazonwebservices.com.cn	SHA256:TdgrNZkiQOdVfYEBUhO4SzUA09VWI5rYOZGTogpwmiM
欧州 (フランクフルト)	eu-central-1	serial-console.ec2-instance-connect.eu-central-1.aws	SHA256:aCMFS/yIcOdOlkXvOl8AmZ1Toe+bBnrJJ3Fy0k0De2c
欧州 (アイルランド)	eu-west-1	serial-console.ec2-instance-connect.eu-west-1.aws	SHA256:h2AaGAWO4Hathhtm6ezs3Bj7udgUxi2qTrHjZAwCW6E

リージョン名	リージョン	エンドポイント	フィンガープリント
欧州 (ロンドン)	eu-west-2	serial-console.ec2-instance-connect.eu-west-2.aws	SHA256:a69rd5CE/AEG4Amm53I6lkD1ZPvS/BCV3tTPW2RnJg8
欧州 (ミラノ)	eu-south-1	ec2-serial-console.eu-south-1.api.aws	SHA256:IC0kOVJnpgFyBVrxn0A7n99ecLbXSX95cuuS7X7QK30
欧州 (パリ)	eu-west-3	serial-console.ec2-instance-connect.eu-west-3.aws	SHA256:q8ldnAf9pymeNe8BnFVngY3RPAr/kxswJUzfrlxeEWs
欧州 (スペイン)	eu-south-2	ec2-serial-console.eu-south-2.api.aws	SHA256:GoCW2DFRlu669QNxqFxEcsR6fZUz/4F4n7T45ZcwoEc
欧州 (ストックホルム)	eu-north-1	serial-console.ec2-instance-connect.eu-north-1.aws	SHA256:tkGFFUVUDvocDiGSS3Cu8Gdl6w2ul32EPNpKFKLwX84
欧州 (チューリッヒ)	eu-central-2	ec2-serial-console.eu-central-2.api.aws	SHA256:8Ppx2mBMf6WdCw0NUlzKfwM4/lfRz4OaXFutQXWp6mk

リージョン名	リージョン	エンドポイント	フィンガープリント
イスラエル (テルアビブ)	il-central-1	ec2-serial-console.il-central-1.api.aws	SHA256:JR 6q8v6kNNP i8+QSFQ4d j5dimNmZP TgwgsM1SNvtYyU
中東 (バーレーン)	me-south-1	ec2-serial-console .me-south-1.api.aws	SHA256:nP jLLKHu2Qn LdUq2kVAr soK5xvPJO MRJKCBzCDqC3k8
中東 (アラブ首長国連邦)	me-central-1	ec2-serial-console .me-central-1.api.aws	SHA256:zpb5duKiBZ +l0dFwPeyy kB4MPBYhl/ XzXNeFSDKBvLE
南米 (サンパウロ)	sa-east-1	ec2-serial-console.sa-east-1.api.aws	SHA256:rd2+/32Ognj ew1yVIemENaQzC +Botbih62OqAPDq1dI
AWS GovCloud (米国東部)	us-gov-east-1	serial-console.ec2 -instance-connect. us-gov-east-1.amaz onaws.com	SHA256:tI we19GWsoy LClrtvu38YEEh+DHIk qnDcZnmtebvF28
AWS GovCloud (米国西部)	us-gov-west-1	serial-console.ec2 -instance-connect. us-gov-west-1.amaz onaws.com	SHA256:kf OFRWLaOZfB +utbd3bRf8OlPf8nG O2YZLqXZiIw5DQ

EC2 シリアルコンソールからの切断

インスタンスの EC2 シリアルコンソール に接続する必要がなくなった場合は、接続を切断できます。シリアルコンソールとの接続を切断しても、インスタンスで実行中のすべてのシェルセッション

は引き続き実行されます。シェルセッションを終了したい場合は、シリアルコンソールとの接続を切断する前に、そのセッションを終了しておく必要があります。

考慮事項

- シリアルコンソール接続は、ユーザーが解除しない限り、通常 1 時間継続します。ただし、システムメンテナンス中は、Amazon EC2 によりシリアルコンソールのセッションが切断されます。
- シリアルコンソールから切断した後、新しいセッションを許可するためにセッションを終了処理するには、30 秒かかります。

シリアルコンソールとの接続解除の方法は、クライアントによって異なります。

ブラウザベースのクライアント

シリアルコンソールから切断するには、シリアルコンソールのブラウザ内ターミナルウィンドウを閉じます。

標準 OpenSSH クライアント

シリアルコンソールから切断するには、次のコマンドを使用して SSH 接続を閉じます。このコマンドは、新しい行の直後に実行する必要があります。

```
$ ~.
```

SSH 接続を閉じるために使用するコマンドは、使用している SSH クライアントによって異なる場合があります。

EC2 シリアルコンソールを使用して Linux インスタンスをトラブルシューティングする

EC2 シリアルコンソールを使用して、インスタンスのシリアルポートに接続することで、起動、ネットワーク設定、およびその他の問題をトラブルシューティングできます。

開始する前に、前提条件を必ず確認してください。

トピック

- GRUB を使用して Linux インスタンスをトラブルシューティングする
- SysRq を使用して Linux インスタンスをトラブルシューティングする

Windows インスタンスのトラブルシューティングの詳細については、Windows インスタンスの Amazon EC2 ユーザーガイド の「EC2 シリアルコンソールを使用した Windows インスタンスのトラブルシューティング」を参照してください。

GRUB を使用して Linux インスタンスをトラブルシューティングする

GNU GRUB (GNU GRand Unified Bootloader の略。一般に GRUB と呼ばれます) は、ほとんどの Linux オペレーティングシステムのデフォルトのブートローダーです。GRUB メニューから、起動先のカーネルを選択したり、メニューエントリを変更してカーネルの起動方法を変更したりできます。これは、障害が発生したインスタンスをトラブルシューティングする際に役立ちます。

GRUB メニューは、ブートプロセス中に表示されます。通常の SSH ではメニューにアクセスできませんが、EC2 シリアルコンソールからアクセスできます。

GRUB を使用できるようにするには、シリアルコンソールへのアクセス権付与と GRUB の設定を含む、前提条件を完了する必要があります。

GRUB を使用する

GRUB を設定したら、シリアルコンソールに接続し、reboot コマンドを使用してインスタンスを再起動します。再起動中は、GRUB メニューが表示されます。GRUB メニューが表示されたら、任意のキーを押してブートプロセスを停止し、GRUB メニューを操作できるようにします。

トピック

- シングルユーザーモード
- 緊急モード

シングルユーザーモード

シングルユーザーモードでは、カーネルを低めの実行レベルで起動します。例えば、ファイルシステムをマウントしても、ネットワークをアクティブ化しない場合があり、インスタンスの修正に必要なメンテナンスを実行することができます。

シングルユーザーモードで起動するには

1. インスタンスのシリアルコンソールに接続します。
2. 次のコマンドを実行して、インスタンスを再起動します。

```
[ec2-user ~]$ sudo reboot
```

3. 再起動時に GRUB メニューが表示されたら、任意のキーを押してブートプロセスを停止します。
4. GRUB メニューで、矢印キーを使用して起動先のカーネルを選択し、キーボードの e を押します。
5. 矢印キーを使用して、カーネルを含む行にカーソルを置きます。行は、インスタンスの起動に使用された AMI に応じて、`linux` または `linux16` のいずれかで始まります。Ubuntu の場合、2 つの行は `linux` で始まります。どちらも次のステップで変更する必要があります。
6. 行の最後に、単語 `single` を追加します。

   Amazon Linux 2 の例を次に示します。

   ```
   linux /boot/vmlinuz-4.14.193-149.317.amzn2.aarch64 root=UUID=d33f9c9a-\
   dadd-4499-938d-ebbf42c3e499 ro  console=tty0 console=ttyS0,115200n8 net.ifname\
   s=0 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff rd.she\
   ll=0 single
   ```
7. シングルユーザーモードで起動するには、Ctrl+X キーを押します。
8. `login` プロンプトで、前に設定したパスワードベースのユーザーのユーザー名を入力し、Enter キーを押します。
9. `Password` プロンプトで、パスワードを入力し、Enter キーを押します。

緊急モード

緊急モードはシングルユーザーモードと似ていますが、カーネルは可能な限り低い実行レベルで実行される点が異なります。

緊急モードで起動するには、前のセクションの シングルユーザーモード の手順に従います。ただし、ステップ 6 では、`single` の代わりに `emergency` という単語を追加します。

SysRq を使用して Linux インスタンスをトラブルシューティングする

システムリクエスト (SysRq) キーは、「マジック SysRq」とも呼ばれ、シェルの外部でカーネルにコマンドを直接送信するために使用でき、カーネルが何をしているかにかかわらず、カーネルは応答します。例えば、インスタンスが応答を停止した場合、SysRq キーを使用して、カーネルにクラッシュまたは再起動するように指示できます。詳細については、Wikipedia の「マジック SysRq キー」を参照してください。

SysRq を使用できるようにするには、シリアルコンソールへのアクセス権付与と SysRq の設定を含む、前提条件を完了する必要があります。

SysRq を使用する

SysRq コマンドは、EC2 シリアルコンソールブラウザベースのクライアントまたは SSH クライアントで使用できます。中断リクエストを送信するコマンドは、クライアントごとに異なります。

SysRq を使用するには、使用しているクライアントに基づいて、次のいずれかの手順を選択します。

Browser-based client

シリアルコンソールのブラウザベースのクライアントで SysRq を使用するには

1. インスタンスのシリアルコンソールに接続します。
2. 中断リクエストを送信するには、CTRL+0 (ゼロ) を押します。キーボードがサポートしている場合は、Pause キーまたは Break キーを使用して中断リクエストを送信することもできます。

```
[ec2-user ~]$ CTRL+0
```

3. SysRq コマンドを発行するには、必要なコマンドに対応するキーボードのキーを押します。例えば、SysRq コマンドのリストを表示するには、h を押します。

```
[ec2-user ~]$ h
```

h コマンドは、次のような内容を出力します。

```
[ 1169.389495] sysrq: HELP : loglevel(0-9) reboot(b) crash(c) terminate-all-
tasks(e) memory-full-oom-kill(f) kill-all-tasks(i) thaw-filesystems
(j) sak(k) show-backtrace-all-active-cpus(l) show-memory-usage(m) nice-all-RT-
tasks(n) poweroff(o) show-registers(p) show-all-timers(q) unraw(r
) sync(s) show-task-states(t) unmount(u) show-blocked-tasks(w) dump-ftrace-
buffer(z)
```

SSH client

SSH クライアントで SysRq を使用するには

1. インスタンスのシリアルコンソールに接続します。
2. 中断リクエストを送信するには、~B (チルダ、その後に大文字の B) を押します。

```
[ec2-user ~]$ ~B
```

3. SysRq コマンドを発行するには、必要なコマンドに対応するキーボードのキーを押します。例えば、SysRq コマンドのリストを表示するには、h を押します。

```
[ec2-user ~]$ h
```

h コマンドは、次のような内容を出力します。

```
[ 1169.389495] sysrq: HELP : loglevel(0-9) reboot(b) crash(c) terminate-all-
tasks(e) memory-full-oom-kill(f) kill-all-tasks(i) thaw-filesystems
(j) sak(k) show-backtrace-all-active-cpus(l) show-memory-usage(m) nice-all-RT-
tasks(n) poweroff(o) show-registers(p) show-all-timers(q) unraw(r
) sync(s) show-task-states(t) unmount(u) show-blocked-tasks(w) dump-ftrace-
buffer(z)
```

> **Note**
>
> 中断リクエストの送信に使用するコマンドは、使用している SSH クライアントによって異なる場合があります。

診断割り込みの送信 (上級ユーザーのみ)

> **Warning**
>
> 診断割り込みは、上級ユーザーが使用することを目的としています。不適切な使用は、インスタンスに悪影響を与える可能性があります。診断割り込みをインスタンスに送信すると、インスタンスがクラッシュして再起動し、データが失われる可能性があります。

到達できないまたは応答しない Linux インスタンスに診断割り込みを送信して、カーネルパニックを手動でトリガーできます。

Linux オペレーティングシステムは一般的に、カーネルパニックが発生するとクラッシュして再起動されます。ただし、オペレーティングシステムの具体的な動作は設定によって異なります。カーネルパニックは、インスタンスのオペレーティングシステムカーネルでクラッシュダンプファイルの生成

などのタスクを実行するためにも使用できます。このクラッシュダンプファイル内の情報を使用すると、根本原因解析を実施してインスタンスのデバッグを行うことができます。

クラッシュダンプデータは、インスタンスの代わりにオペレーティングシステムによってローカルで生成されます。

インスタンスに診断割り込みを送信する前に、オペレーティングシステムのドキュメントを参照し、必要な設定を変更することをお勧めします。

コンテンツ

サポートされるインスタンスタイプ

診断割り込みは、Nitro ベースのすべてのインスタンスタイプでサポートされます。ただし、AWS Graviton プロセッサで動作するものを除きます。詳細については、「Nitro System 上に構築されたインスタンス および AWS Graviton」を参照してください。

前提条件

診断割り込みを使用する前に、インスタンスのオペレーティングシステムを設定する必要があります。これにより、カーネルパニックの発生時に必要なアクションをオペレーティングシステムで実行できます。

カーネルパニックの発生時にクラッシュダンプが生成されるように Amazon Linux 2 を設定するには

1. インスタンスに接続します。
2. kexec と kdump をインストールします。

```
[ec2-user ~]$ sudo yum install kexec-tools -y
```

3. セカンダリカーネル用に適切な量のメモリが予約されるようにカーネルを設定します。予約するメモリの量は、インスタンスで使用可能な合計メモリによって異なります。適切なテキストエディタを使用して /etc/default/grub ファイルを開き、GRUB_CMDLINE_LINUX_DEFAULT から始まる行を見つけて、crashkernel という形式で crashkernel=*memory_to_reserve*

パラメータを追加します。たとえば、160MB を予約するには、grub ファイルを次のように変更します。

```
GRUB_CMDLINE_LINUX_DEFAULT="crashkernel=160M console=tty0 console=ttyS0,115200n8
 net.ifnames=0 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff
 rd.shell=0"
GRUB_TIMEOUT=0
GRUB_DISABLE_RECOVERY="true"
```

4. 変更内容を保存し、grub ファイルを閉じます。
5. GRUB2 設定ファイルを再構築します。

```
[ec2-user ~]$ sudo grub2-mkconfig -o /boot/grub2/grub.cfg
```

6. Intel および AMD プロセッサをベースとしたインスタンスの場合、send-diagnostic-interrupt コマンドを実行すると 不明なマスク不可割り込み (NMI、unknown non-maskable interrupt) がインスタンスに送信されます。不明な NMI を受信した際にはクラッシュするようにカーネルを設定しておく必要があります。適切なテキストエディタを使用して /etc/sysctl.conf ファイルを開き、以下を追加します。

```
kernel.unknown_nmi_panic=1
```

7. インスタンスを再起動して再接続します。
8. 正しいcrashkernel パラメータを使用してカーネルが起動されていることを確認します。

```
$ grep crashkernel /proc/cmdline
```

次の出力例は、適切な設定を示しています。

```
BOOT_IMAGE=/boot/vmlinuz-4.14.128-112.105.amzn2.x86_64 root=UUID=a1e1011e-
e38f-408e-878b-fed395b47ad6 ro crashkernel=160M console=tty0 console=ttyS0,115200n8
 net.ifnames=0 biosdevname=0 nvme_core.io_timeout=4294967295 rd.emergency=poweroff
 rd.shell=0
```

9. kdump サービスが実行中であることを確認します。

```
[ec2-user ~]$ systemctl status kdump.service
```

次の出力例は、kdump サービスが実行中である場合の結果を示しています。

```
kdump.service - Crash recovery kernel arming
   Loaded: loaded (/usr/lib/systemd/system/kdump.service; enabled; vendor preset:
 enabled)
   Active: active (exited) since Fri 2019-05-24 23:29:13 UTC; 22s ago
  Process: 2503 ExecStart=/usr/bin/kdumpctl start (code=exited, status=0/SUCCESS)
 Main PID: 2503 (code=exited, status=0/SUCCESS)
```

> **Note**
>
> デフォルトでは、クラッシュダンプファイルは /var/crash/ に保存されます。保存先を変更するには、適切なテキストエディタを使用して /etc/kdump.conf ファイルを変更します。

カーネルパニックの発生時にクラッシュダンプが生成されるように Amazon Linux を設定するには

1. インスタンスに接続します。
2. kexec と kdump をインストールします。

```
[ec2-user ~]$ sudo yum install kexec-tools -y
```

3. セカンダリカーネル用に適切な量のメモリが予約されるようにカーネルを設定します。予約するメモリの量は、インスタンスで使用可能な合計メモリによって異なります。

```
$ sudo grubby --args="crashkernel=memory_to_reserve" --update-kernel=ALL
```

   たとえば、クラッシュカーネル用に 160MB を予約するには、次のコマンドを使用します。

```
$ sudo grubby --args="crashkernel=160M" --update-kernel=ALL
```

4. Intel および AMD プロセッサをベースとしたインスタンスの場合、send-diagnostic-interrupt コマンドを実行すると 不明なマスク不可割り込み (NMI、unknown non-maskable interrupt) がインスタンスに送信されます。不明な NMI を受信した際にはクラッシュするようにカーネルを設定しておく必要があります。適切なテキストエディタを使用して /etc/sysctl.conf ファイルを開き、以下を追加します。

```
kernel.unknown_nmi_panic=1
```

5. インスタンスを再起動して再接続します。
6. 正しいcrashkernel パラメータを使用してカーネルが起動されていることを確認します。

```
$ grep crashkernel /proc/cmdline
```

次の出力例は、適切な設定を示しています。

```
root=LABEL=/ console=tty1 console=ttyS0 selinux=0 nvme_core.io_timeout=4294967295
 LANG=en_US.UTF-8 KEYTABLE=us crashkernel=160M
```

7. kdump サービスが実行中であることを確認します。

```
[ec2-user ~]$ sudo service kdump status
```

サービスが実行中であれば、コマンドから Kdump is operational 応答が返されます。

> **Note**
>
> デフォルトでは、クラッシュダンプファイルは /var/crash/ に保存されます。保存先を変更するには、適切なテキストエディタを使用して /etc/kdump.conf ファイルを変更します。

SUSE Linux Enterprise、Ubuntu、または Red Hat Enterprise Linux を設定するには

Intel および AMD プロセッサをベースとしたインスタンスの場合、send-diagnostic-interrupt コマンドを実行すると 不明なマスク不可割り込み (NMI、unknown non-maskable interrupt) がインスタンスに送信されます。オペレーティングシステムの設定ファイルを調整して、不明な NMI を受信したときにカーネルがクラッシュするように設定する必要があります。カーネルをクラッシュするように設定する方法については、オペレーティングシステムのドキュメントを参照してください。

- SUSE Linux Enterprise
- Ubuntu
- Red Hat Enterprise Linux (RHEL)

診断割り込みの送信

必要な設定変更を完了したら、AWS CLI または Amazon EC2 API を使用して、診断割り込みをインスタンスに送信できます。

診断割り込みをインスタンス (AWS CLI) に送信するには

send-diagnostic-interrupt コマンドを使用し、インスタンス ID を指定します。

```
aws ec2 send-diagnostic-interrupt --instance-id i-1234567890abcdef0
```

関連情報

このサービスを利用する際に役立つ関連リソースは次のとおりです。

AWS での Linux

- AWS からの Linux — AWS の提供する最新の Linux ベースオペレーティングシステムポートフォリオ。
- EC2 Image Builder - カスタマイズされたセキュアな最新のサーバーイメージの作成、管理、デプロイを自動化します。サーバーイメージには、特定の IT 標準を満たすソフトウェア設定が事前にインストールおよび定義されています。

開発者向けチュートリアル

- Amazon EC2 にウェブアプリケーションをデプロイする — AWS CDK を使用して Amazon EC2 インスタンスを作成し、そのインスタンスにウェブアプリケーションをデプロイします。
- AWS Backup を使用して Amazon EC2 のバックアップと復元 — Amazon EC2 インスタンスのオンデマンドバックアップを作成してから、Amazon EC2 インスタンスをバックアップするためのバックアップ計画を作成する方法を学びます。
- Amazon Elastic Container Service、Docker、Amazon EC2 を使用してモノリスアプリケーションをマイクロサービスに分割 - モノリシックな node.js アプリケーションを Docker コンテナにデプロイし、ダウンタイムなしでアプリケーションをマイクロサービスに分離します。

AWS re:Post

AWS re:Post — AWS が運営する質疑応答 (Q & A)。技術的な質問に対して、専門家がレビューしたクラウドソーシングによる回答を提供します。

料金

Amazon EC2 の料金 - Amazon EC2 の料金情報です。

一般的な AWS リソース

AWS を利用する際に役立つ一般的なリソースは以下の通りです。

- クラスとワークショップ – AWS のスキルを磨き、実践的な経験が得るために役立つセルフペースラボに加えて、ロールベースのコースと特別コースへのリンクです。

- AWS デベロッパーセンター – チュートリアルの検索、ツールのダウンロード、AWS デベロッパーイベントの確認を行います。
- AWS デベロッパーツール - AWS アプリケーションを開発および管理するためのデベロッパーツール、SDK、IDE ツールキット、およびコマンドラインツールへのリンクです。
- ご利用開始のためのリソースセンター — AWS アカウント をセットアップする方法、AWS コミュニティに参加する方法、最初のアプリケーションを起動する方法を説明します。
- ハンズオンチュートリアル - ステップ バイ ステップのチュートリアルに従って、最初のアプリケーションを AWS で起動します。
- AWS ホワイトペーパー – アーキテクチャ、セキュリティ、エコノミクスなどのトピックについて、AWS のソリューションアーキテクトや他の技術エキスパートが記述した AWS の技術ホワイトペーパーの包括的なリストへのリンクです。
- AWS Support Center – AWS Support のケースを作成して管理するためのハブです。フォーラム、技術上のよくある質問、サービスヘルスステータス、AWS Trusted Advisor など、他の役立つリソースへのリンクも含まれています。
- AWS Support – AWS Support に関する情報のメインウェブページです。クラウド内でのアプリケーションの構築および実行を支援するために 1 対 1 での迅速な対応を行うサポートチャネルとして機能します。
- お問い合わせ - AWS の請求、アカウント、イベント、不正使用、その他の問題などに関するお問い合わせの受付窓口です。
- AWS サイトの利用規約 – 当社の著作権、商標、お客様のアカウント、ライセンス、サイトへのアクセス、その他のトピックに関する詳細情報。

ドキュメント履歴

次の表は、2019 年以降の Amazon EC2 ドキュメントへの重要な追加項目をまとめたものです。また、お客様からいただいたフィードバックに対応するために、ドキュメントを頻繁に更新しています。

変更	説明	日付
IMDSv2 をアカウントのデフォルトとして設定する	デフォルトでインスタンスメタデータサービスバージョン 2 (IMDSv2) を使用するように、アカウント内のすべての新しい EC2 インスタンスの起動を設定できます。	2024 年 3 月 25 日
スナップショットから作成された新しい Linux AMI にタグを付ける	スナップショットから Linux AMI を作成する場合、新しい AMI にタグを付けることができます。	2024 年 3 月 7 日
コピー時に新しい AMI とスナップショットにタグを付ける	AMI をコピーするときに、新しい AMI とスナップショットに同じタグを付けるか、異なるタグを付けることができます。	2024 年 3 月 7 日
新規ベアメタルインスタンス	C7gd、M7gd、および R7gd のベアメタルインスタンス。	2024 年 3 月 6 日
EC2 Instance Connect は macOS AMI にプリインストールされています	EC2 Instance Connect が macOS Sonoma 14.2.1 以降、macOS Ventura 13.6.3 以降、および macOS Monterey 12.7.2 以降の AMI にプリインストールされるようになりました。	2024 年 1 月 26 日

CentOS、macOS、RHEL の EC2 Instance Connect サポート	サポートされている CentOS、macOS、RHEL AMI に対して EC2 Instance Connect をインストールできるようになりました。	2023 年 12 月 6 日
C7a、C7i、R7a、R7i、および R7iz の休止状態のサポート	C7a、C7i、R7a、R7i、および R7iz インスタンスタイプで実行する、新しく起動したインスタンスを休止状態にします。	2023 年 12 月 1 日
Amazon Q EC2 インスタンスタイプセレクター	Amazon Q EC2 インスタンスタイプセレクターは、優先するユースケース、ワークロードタイプ、CPU メーカーだけではなく、価格とパフォーマンスの優先順位も考慮します。次に、このデータを使用して、新しいワークロードに最適な Amazon EC2 インスタンスタイプのガイダンスと提案を提供します。	2023 年 11 月 28 日
EC2 無料利用枠	EC2 無料利用枠の使用状況は EC2 ダッシュボードから追跡できます。	2023 年 11 月 26 日

Console-to-Code	Console-to-Code は、自動化コードの使用を開始する際に役立ちます。Console-to-Code はコンソールのアクションを記録し、生成 AI を使用して優先する infrastructure-as code 形式のコードを提案します。このコードを出発点として使用し、特定のユースケースに合わせて本番環境に対応できるようにカスタマイズできます。	2023 年 11 月 26 日
設定可能なアイドル接続追跡タイムアウト	セキュリティグループの接続がアイドル状態のままになると、接続追跡が使い果たされることで接続が追跡されなくなり、また、パケットがドロップされる原因となります。今回、Elastic Network Interface で、セキュリティグループの接続追跡のタイムアウトを秒単位で設定できるようになりました。	2023 年 11 月 17 日
AWSDataLifecycleManagerSSMFullAccess AWS マネージドポリシー	`AWSSystemsManagerSAP-CreateDLMSnapshotForSAPHANA` SSM ドキュメントを使用した SAP HANA で、アプリケーション整合性のあるスナップショットをサポートするようにポリシーを更新しました。	2023 年 11 月 17 日

VolumeStalledIOCheck メトリクス	過去 1 分間にストールした IO チェックに、ボリュームが合格していたか不合格であったかは、VolumeStalledIOCheck メトリクスを使用して確認できます。	2023 年 11 月 16 日
PTP ハードウェアクロック	サポート対象のインスタンスに対し、プレシジョンタイムプロトコル (PTP) のハードウェアクロックが搭載されました。PTP ハードウェアクロックでは、NTP または直接 PTP 接続のいずれかがサポートされています。	2023 年 11 月 16 日
休止が有効になっているインスタンスのインスタンスタイプ変更	休止状態が有効になっていて stopped 状態にあるインスタンスで、インスタンスのタイプを変更できるようになりました。	2023 年 11 月 16 日
Amazon Data Lifecycle Manager のデフォルトポリシー	Amazon EBS スナップショットと EBS で動作する AMI 用に、Amazon Data Lifecycle Manager のデフォルトポリシーを作成し、リージョン内のすべてのボリュームとインスタンスをバックアップできるようになりました。	2023 年 11 月 16 日

DL2q インスタンス	DL2q インスタンスは、第 7 世代の Qualcomm エッジ AI コアを搭載した Qualcomm AI100 推論アクセラレータを使用します。これらを使用すると、深層学習 (DL) ワークロードをコスト効率よくクラウドにデプロイしたり、Qualcomm のエッジデバイスにデプロイされる DL ワークロードのパフォーマンスと精度を検証したりできます。	2023 年 11 月 15 日
Amazon EBS スナップショットのロック	Amazon EBS スナップショットをロックして、偶発的または悪意のある削除から保護したり、WORM 形式で一定期間保存したりできます。	2023 年 11 月 15 日
インスタンストポロジー	DescribeInstanceTopology API を使用してインスタンスの場所を検出できます。この情報は、HPC ジョブと ML ジョブを相互に物理的に近いインスタンスで実行することで、それらのジョブを最適化するために使用できます。	2023 年 11 月 13 日
スナップショットのパブリックアクセスのブロック	パブリックに共有されることを防止するために、スナップショットのパブリックアクセスをブロックできるようになりました。	2023 年 11 月 9 日

Amazon Data Lifecycle Manager の事前および事後スクリプト	Amazon Data Lifecycle Manager のスナップショットポリシーで、事前スクリプトと事後スクリプトを使用して、アプリケーション整合性のあるスナップショットのライフサイクルを自動化できるようになりました。	2023 年 11 月 7 日
AWSDataLifecycleManagerSSMFullAccess AWS マネージドポリシー	Amazon Data Lifecycle Manager に、AWS 管理ポリシー AWSDataLifecycleManagerSSMFullAccess を追加しました。	2023 年 11 月 7 日
機械学習用のキャパシティブロック	短期間の機械学習 (ML) ワークロードをサポートするために、未来の日付で GPU インスタンスを予約できるようになりました。	2023 年 10 月 31 日
新規ベアメタルインスタンス	R7iz の `.metal-16xl` および `.metal-32xl` ベアメタルインスタンス。	2023 年 10 月 30 日
新規ベアメタルインスタンス	M7i、R7i、C7i の `.metal-24xl` および `.metal-48xl` ベアメタルインスタンス。	2023 年 10 月 30 日
新規 I4i インスタンス	`i4i.12xlarge` および `i4i.24xlarge` インスタンスが利用可能になりました。	2023 年 10 月 26 日
Mac2-m2 インスタンス	Apple M2 プロセッサを搭載した新しい汎用インスタンスタイプ。	2023 年 10 月 25 日

スポットインスタンスの休止状態	オンデマンドインスタンスで現在利用できる休止機能およびインスタンスファミリーと同様のものを使用して、スポットインスタンスを休止できるようになりました。	2023 年 10 月 24 日
AMI のパブリックアクセスのブロックをデフォルトに設定	すべての新規アカウントとパブリック AMI を持たない既存のアカウントで、AMI のパブリックアクセスのブロックがデフォルトで有効になりました。	2023 年 10 月 20 日
Amazon EC2 グローバルビュー	Amazon EC2 Global View が、追加のリソースタイプとカスタマイズ可能な表示オプションをサポート。	2023 年 10 月 18 日
R7i インスタンス	第 4 世代 Intel Xeon スケーラブルプロセッサを搭載した新しいメモリ最適化インスタンスタイプ。	2023 年 10 月 16 日
Ubuntu 22.04.2 LTS (Jammy Jellyfish) 休止のサポート	Ubuntu 22.04.2 LTS (Jammy Jellyfish) AMI から新たに起動したインスタンスを休止します。	2023 年 10 月 16 日
AMI の無効化	AMI を無効にして、インスタンスの起動に使用されないようにできます。	2023 年 10 月 12 日

アタッチ済みの EBS ステータスチェック	アタッチ済みの EBS ステータスチェックを使用して、インスタンスにアタッチされている Amazon EBS ボリュームが到達可能かどうかをモニタリングできます。	2023 年 10 月 11 日
新規ベアメタルインスタンス	`r7a.metal-48xl` R7a のベアメタルインスタンス。ベアメタルインスタンスは、ホストサーバーの物理リソースにアプリケーションが直接アクセスできるようにします。	2023 年 10 月 4 日
C7a インスタンス	第 4 世代 AMD EPYC プロセッサが機能する新しいコンピューティング最適化インスタンス。	2023 年 10 月 4 日
Red Hat Enterprise Linux 9 の休止状態サポート	新しく Red Hat Enterprise Linux 9 AMI から起動されたインスタンスでは休止状態が利用可能です。	2023 年 10 月 2 日
AL2023 の休止状態サポート	新しく AL2023 AMI から起動されたインスタンスでは休止状態が利用可能です。	2023 年 10 月 2 日
スポットフリート内のスポットインスタンスの中断を開始する	Amazon EC2 コンソールでスポットフリートを選択してフリート内のスポットインスタンスの中断を実行すると、スポットインスタンス上のアプリケーションでの中断に関する処理をテストできます。	2023 年 9 月 21 日

NVMe 予約	マルチアタッチ対応の io2 ボリュームは、業界標準のストレージフェンシングプロトコルのセットである NVMe 予約をサポートします。	2023 年 9 月 18 日
Mac2-m2pro インスタンス	Apple M2 Pro プロセッサを搭載した新しい汎用インスタンスタイプです。	2023 年 9 月 18 日
C7i インスタンス	第 4 世代インテル Xeon スケーラブルプロセッサを搭載した新しいコンピューティング最適化インスタンス。	2023 年 9 月 14 日
AMI へのパブリックアクセスをブロックする	AMI のパブリックアクセスのブロックをアカウントレベルで有効にして、AMI を公開しようとするあらゆる試みをブロックできます。	2023 年 9 月 12 日
R7a インスタンス	第 4 世代 AMD EPYC 9R14 プロセッサと最大 1536 GiB のシステムメモリを搭載した、新しいメモリ最適化インスタンスタイプです。	2023 年 9 月 11 日
R7iz インスタンス	第 4 世代 Intel Xeon プロセッサを搭載した、新しい高周波数の大容量メモリインスタンス。	2023 年 9 月 7 日
M7i および M7i-flex の休止サポート	M7i および M7i-flex インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2023 年 8 月 22 日

Hpc7a インスタンス	第 4 世代 AMD EPYC プロセッサが機能する新しいコンピューティング最適化インスタンス。これらのインスタンスタイプは、最大 300 Gbps のネットワーク帯域幅と最大 192 の CPU コア、最大 768 GB のシステムメモリをサポートします。	2023 年 8 月 17 日
M7a インスタンス	第 4 世代 AMD EPYC プロセッサを搭載した新しい汎用インスタンス。	2023 年 8 月 15 日
EC2-Classic は廃止されました	EC2-Classic では、EC2 インスタンスが他のお客様と共有される単一のフラットネットワーク内で実行されました。Amazon VPC が EC2 クラシックに取って代わります。Amazon VPC では、インスタンスは AWS アカウントから論理的に独立した仮想プライベートクラウド (VPC) で稼働します。	2023 年 8 月
M7i-flex インスタンス	コンピューティング、メモリ、ネットワークリソースがバランスよく構成され、幅広い汎用アプリケーションに対応する新しい汎用インスタンス。ベースラインの CPU パフォーマンスは 40% で、24 時間の 95% の時間に最大 100% の CPU パフォーマンスを実現できます。	2023 年 8 月 2 日

M7i インスタンス	第 4 世代 Intel Xeon スケーラブルプロセッサーを搭載した、新しい汎用インスタンス。	2023 年 8 月 2 日
R7gd インスタンス	最新の AWS Graviton3 プロセッサを搭載したあたら曽於メモリ最適化インスタンス。	2023 年 7 月 28 日
M7gd インスタンス	最新の AWS Graviton3 プロセッサを搭載した新しい汎用インスタンス。	2023 年 7 月 28 日
C7gd インスタンス	最新の AWS Graviton3 プロセッサを搭載した新しいコンピューティング最適化インスタンス。	2023 年 7 月 28 日
P5 インスタンス	640 GB の高帯域幅 GPU メモリを搭載した 8 つの NVIDIA H100 GPU、第 3 世代 AMD EPYC プロセッサー、2 TB システムメモリを搭載した新しい高速コンピューティングインスタンス。	2023 年 7 月 26 日
Amazon EBS パフォーマンスの更新	R6a インスタンスの Amazon EBS パフォーマンスを更新しました。	2023 年 6 月 29 日
Hpc7g インスタンス	AWS Graviton3E プロセッサを搭載した新しい高性能のコンピューティングインスタンスは、Graviton3 プロセッサに比べて最大 35 パーセント向上したベクトル命令処理パフォーマンスを提供します。	2023 年 6 月 20 日

Dedicated Hosts	Dedicated Hosts は、Outpost の特定のハードウェアアセットに割り当てることができます。	2023 年 6 月 20 日
C7gn インスタンス	最新世代の AWS Graviton3E プロセッサと新しい AWS Nitro Card を搭載した、新しいコンピューティング最適化インスタンスです。このインスタンスは、最大 200 Gbps のネットワーク帯域幅を提供します。	2023 年 6 月 20 日
EC2 Instance Connect Endpoint	インスタンスにパブリック IPv4 アドレスがなくても、SSH または RDP 経由でインスタンスに接続できるようになりました。	2023 年 6 月 13 日
IMDS パッケージアナライザー	IMDS パケットアナライザーを使用して、EC2 インスタンスの IMDSv1 呼び出しのソースを特定できるようになりました。	2023 年 6 月 1 日
新しいストレージ最適化インスタンス	I4g は、AWS Graviton2 プロセッサと AWS Nitro SSD を備えた、新しいストレージ最適化インスタンスです。	2023 年 5 月 9 日
Trn1n インスタンス	AWS Trainium アクセラレータを搭載し、深層学習向けに最適化した新しい高速コンピューティングインスタンス。	2023 年 4 月 13 日

Inf2 インスタンス	AWS によって設計された最新の機械学習チップである AWS Inferentia2 アクセラレータを搭載した新しいインスタンス。	2023 年 4 月 13 日
EC2 シリアルコンソールのベアメタルインスタンス	EC2 シリアルコンソールは、選択したベアメタルインスタンスのシリアルポートへの接続をサポートするようになりました。	2023 年 4 月 11 日
起動テンプレートのクォータ	起動テンプレートのクォータと起動テンプレートバージョンのクォータを、Service Quotas コンソールと Service Quotas CLI を使用して表示できるようになりました。	2023 年 4 月 3 日
キャパシティ予約使用率の通知	AWS Health は、アカウントのキャパシティ予約のキャパシティ使用率が 20% を下回ったときに通知を送信するようになりました。	2023 年 4 月 3 日
Amazon EBS パフォーマンスの更新	M6a インスタンスおよび C6a インスタンスの Amazon EBS パフォーマンスを更新しました。	2023 年 4 月 3 日
キャパシティ予約グループ	自身が所有するキャパシティ予約グループに、共有を受けるキャパシティ予約を追加できるようになりました。	2023 年 3 月 30 日
新規ベアメタルインスタンス	C6in、M6iDn、M6in、R6iDn、および R6in 用ベアメタルインスタンス。	2023 年 3 月 21 日

インスタンスメタデータオプションの変更	Amazon EC2 コンソールを使用して、インスタンスメタデータオプションを変更することができるようになりました。	2023 年 3 月 20 日
macOS オペレーティングシステムのインプレースアップデート	Apple MacOS オペレーティングシステムのインプレース更新を、M1 Mac インスタンスで実行できるようになりました。	2023 年 3 月 14 日
UEFI Preferred	統合拡張ファームウェアインターフェイス (UEFI) とレガシー BIOS ブートモードの両方をサポートする単一の AMI を作成できるようになりました。	2023 年 3 月 3 日
IMDSv2 用に AMI を変更する	既存の AMI から起動されるインスタンスがデフォルトで IMDSv2 を必須とするように AMI を変更します。	2023 年 2 月 28 日
ENA Express でサポートされるインスタンスを追加	ENA Express でサポートされる新規および既存のインスタンスタイプを示す表を追加しました。	2023 年 2 月 13 日
Amazon EBS での障害テスト	AWS FIS を使用して EBS ボリュームとそれがアタッチされているインスタンスとの間の I/O を一時的に停止し、ワークロードが I/O 中断をどのように処理するかをテストします。	2023 年 1 月 27 日

起動テンプレートの AMI エイリアス	起動テンプレートで AMI ID の代わりに AWS Systems Manager パラメータを指定すると、AMI ID が変更されるたびにテンプレートを更新する必要がなくなります。	2023 年 1 月 19 日
C6i、i3en、および M6i の休止サポート	C6i、I3en、M6i インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2022 年 12 月 19 日
Torn Write Prevention	ブロックストレージ機能である Torn Write Prevention 機能により、データの回復力に悪影響を及ぼすことなく、I/O 負荷の高いリレーショナルデータベースワークロードのパフォーマンスを向上させ、レイテンシーを削減します。	2022 年 11 月 29 日
Hpc6id インスタンス	第 3 世代 Intel Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しいメモリ最適化インスタンス。	2022 年 11 月 29 日
R6in および R6idn インスタンス	ネットワーク負荷の大きいワークロード向けの新しいメモリ最適化インスタンス。	2022 年 11 月 28 日
M6in および M6idn インスタンス	新しい汎用コンピューティングインスタンスタイプ。	2022 年 11 月 28 日

ENA Express	ENA Express を使用することで EC2 インスタンス間のネットワークトラフィックのスループットを向上させ、テールレイテンシーを最小限に抑えます。	2022 年 11 月 28 日
C6in インスタンス	ハイパフォーマンスコンピューティングの実行に最適化された、新しいコンピューティング最適化インスタンス。	2022 年 11 月 28 日
ごみ箱の保持ルールのロック	保持ルールをロックすることで、偶発的な、あるいは悪意のある変更や削除から保護できます。	2022 年 11 月 23 日
AMI タグのコピー	AMI をコピーすると、ユーザー定義の AMI タグも同時にコピーできます。	2022 年 11 月 18 日
保存と復元用の AMI サイズ	Amazon S3 バケットに保存、復元できる AMI のサイズ (圧縮前) が、最大 5,000GB まで可能になりました。	2022 年 11 月 16 日
スポットインスタンスの priceCapacityOptimized 配分戦略	`priceCapacityOptimized` 配分戦略を使用するスポットフリートは、価格と容量の両方を考慮し、中断する可能性が最も低く、価格が最も低いスポットインスタンスプールを選択します。	2022 年 11 月 10 日

スポットインスタンスの price-capacity-optimized 配分戦略	`price-capacity-optimized` 配分戦略を使用する EC2 フリートは、価格と容量の両方を考慮し、中断する可能性が最も低く、価格が最も低いスポットインスタンスプールを選択します。	2022 年 11 月 10 日
アカウントと AMI の共有をキャンセルする	AMI が AWS アカウント と共有されていて、そのアカウントとの共有が不要になった場合は、AMI の起動許可からアカウントを削除できます。	2022 年 11 月 4 日
Elastic IP アドレスを移管する	Elastic IP アドレスを 1 つの AWS アカウントから別のアカウントに移管できるようになりました。	2022 年 10 月 31 日
ルートボリュームを置き換える	AMI を使用して実行中のインスタンスのルート Amazon EBS ボリュームを置き換えることができます。	2022 年 10 月 27 日
Trn1 インスタンス	AWS Trainium チップを搭載し、深層学習向けに最適化した新しい高速コンピューティングインスタンス。	2022 年 10 月 10 日
インスタンスをデータベースに自動接続する	自動接続機能を使用して、1 つ以上の EC2 インスタンスを RDS データベースにすばやく接続し、これらの間のトラフィックを許可することができます。	2022 年 10 月 10 日

AMI クォータ	AMI を作成および共有する際には、クォータが適用されます。	2022 年 10 月 10 日
IMDSv2 用に AMI を設定する	AMI から起動されるインスタンスがデフォルトで IMDSv2 を必須とするように AMI を設定します。	2022 年 10 月 3 日
スポットインスタンスの中断を開始する	Amazon EC2 コンソールでスポットインスタンスを選択して中断を実行すると、スポットインスタンス上のアプリケーションでの中断に関する処理をテストできます。	2022 年 9 月 26 日
検証済み AMI プロバイダー	Amazon EC2 コンソールでは、Amazon または検証済み Amazon パートナーが所有するパブリック AMI には [Verified provider] (検証済みプロバイダー) のマークが付されます。	2022 年 7 月 22 日
R6a インスタンス	第 3 世代 AMD EPYC プロセッサを搭載した新しいメモリ最適化インスタンス。	2022 年 7 月 19 日
AWS Outposts のプレイスメントグループ	Outpost のプレイスメントグループにホストの分散戦略が追加されました。	2022 年 6 月 30 日

[Condition keys for Recycle Bin] (ごみ箱の条件キー)	ごみ箱リクエストのアクセスをフィルタリングするために `rbin:Request/ResourceType` と `rbin:Attribute/ResourceType` の条件キーを使用することができます。	2022 年 6 月 14 日
R6id インスタンス	第 3 世代 Intel Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しいメモリ最適化インスタンス。	2022 年 6 月 9 日
`io2` Block Express ボリューム	`io2` Block Express ボリュームのサイズとプロビジョンド IOPS を変更し、高速スナップショット復元のために有効にできます。	2022 年 5 月 31 日
AWS Outposts での Dedicated Hosts	AWS Outposts に Dedicated Hosts を割り当てることができます。	2022 年 5 月 31 日
M6id インスタンス	第 3 世代 Intel Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しい汎用インスタンス。	2022 年 5 月 26 日
C6id インスタンス	第 3 世代 Intel Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しいコンピューティング最適化インスタンス。	2022 年 5 月 26 日
インスタンス停止の防止	インスタンスが誤って停止するのを防ぐために、インスタンスに対する停止保護を有効にすることができます。	2022 年 5 月 24 日

C7g インスタンス	最新の AWS Graviton3 プロセッサを搭載した新しいコンピューティング最適化インスタンス。	2022 年 5 月 23 日
UEFI Secure Boot	UEFI Secure Boot は、Amazon EC2 の長期にわたって使用されてきたセキュアブートプロセスを基に構築されており、さらなる多重防衛をすることで、再起動後も持続する脅威からソフトウェアを保護します。	2022 年 5 月 10 日
NitroTPM	Nitro Trusted Platform Module (NitroTPM) は、AWS Nitro System によって提供される TPM 2.0 仕様に準拠した仮想デバイスです。	2022 年 5 月 10 日
AMI 状態変更イベント	Amazon EC2 で、AMI の状態が変更したときにイベントが生成されるようになりました。Amazon EventBridge を使用することで、これらのイベントの検出と対応が行えるようになります。	2022 年 5 月 9 日
パブリックキーの説明	パブリックキーと Amazon EC2 キーペアの作成日をクエリできます。	2022 年 4 月 28 日
キーペアを作成する	新しいキーペアを作成するときに、キーの形式 (PEM または PPK) を指定できます。	2022 年 4 月 28 日

I4i インスタンス	第 3 世代インテル Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しいストレージ最適化インスタンス。	2022 年 4 月 27 日
起動時に Amazon FSx ファイルシステムをマウントする	新しいインスタンス起動ウィザードを使用して起動時に新規または既存の Amazon FSx for NetApp ONTAP ファイルシステムまたは Amazon FSx for OpenZFS ファイルシステムをマウントできます。	2022 年 4 月 12 日
新しいインスタンス起動ウィザード	Amazon EC2 コンソールの新たに改善された起動エクスペリエンスにより、EC2 インスタンスをすばやく簡単に起動できます。	2022 年 4 月 5 日
パブリック AMI を自動的に非推奨にする	すべてのパブリック AMI を非推奨にする日をデフォルトで AMI 作成日の 2 年後とします。	2022 年 3 月 31 日
インスタンスメタデータカテゴリ: autoscaling/target-lifecycle-state	Auto Scaling グループを使用しているときは、インスタンスメタデータから インスタンスのターゲットライフサイクル状態を取得できます。	2022 年 3 月 24 日
X2idn インスタンスおよび X2iedn インスタンス	インテル Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しいメモリ最適化インスタンス。	2022 年 3 月 10 日

AMI の最終起動時間	`lastLaunchedTime` は、インスタンスの起動のために AMI が最後に使用された時間を示します。	2022 年 2 月 28 日
C6a インスタンス	第 3 世代 AMD EPYC プロセッサ (ミラノ) を搭載した新しいコンピューティング最適化インスタンス。	2022 年 2 月 14 日
AMI のごみ箱	ごみ箱を使用すると、誤って削除した AMI を復元できます。	2022 年 2 月 3 日
X2iezn インスタンス	インテル Xeon プラチナプロセッサ (Cascade Lake) を搭載した、新しいメモリ最適化インスタンス。	2022 年 1 月 26 日
ED25519 キー	ED25519 キーは EC2 インスタンス Connect および EC2 シリアルコンソールでサポートされるようになりました。	2022 年 1 月 20 日
新しいローカルゾーンを追加	アトランタ、フェニックス、シアトルの Local Zones を追加します。	2022 年 1 月 11 日
キャパシティ予約用の追加の RHEL プラットフォーム	オンデマンドキャパシティ予約用の追加の Red Hat Enterprise Linux プラットフォーム。	2022 年 1 月 11 日
Hpc6a インスタンス	AMD EPYC プロセッサを搭載した新しいコンピューティング最適化インスタンス。	2022 年 1 月 10 日

インスタンスメタデータのインスタンスタグ	インスタンスのメタデータからインスタンスのタグにアクセスできます。	2022 年 1 月 6 日
クラスタープレイスメントグループでのキャパシティ予約	クラスタープレイスメントグループでキャパシティ予約を作成できます。	2022 年 1 月 6 日
Im4gn インスタンスと Is4gen インスタンス	新しいストレージ最適化インスタンス。	2021 年 11 月 30 日
Amazon EBS スナップショットのごみ箱	Amazon EBS スナップショットのごみ箱は、誤って削除したスナップショットを復元できるスナップショット復元機能です。	2021 年 11 月 29 日
M6a インスタンス	AMD 第 3 世代 EPYC プロセッサを搭載した新しい汎用インスタンス。	2021 年 11 月 29 日
G5g インスタンス	64 ビット Arm アーキテクチャベースの AWS Graviton2 プロセッサを搭載した新しい高速コンピューティングインスタンス。	2021 年 11 月 29 日
Amazon EBS Snapshots Archive	Amazon EBS Snapshots Archive は、アクセス頻度の低いスナップショットを低コストで長期保存するために使用できる新しいストレージ階層です。	2021 年 11 月 29 日
R6i インスタンス	新しいメモリ最適化インスタンス。	2021 年 11 月 22 日

G5 インスタンス	最大 8 つの NVIDIA A10G GPU と第 2 世代 AMD EPY プロセッサを搭載した、新しい高速コンピューティングインスタンス。	2021 年 11 月 11 日
スポットフリート launch-before-terminate	スポット群は、新しい置換スポットインスタンスが起動された後に、リバランス通知を受信するスポットインスタンスを終了できます。	2021 年 11 月 4 日
EC2 フリート launch-before-terminate	EC2 フリートは、新しい代替スポットインスタンスが起動された後に、再調整通知を受信したスポットインスタンスを終了できます。	2021 年 11 月 4 日
タイムスタンプの比較	Amazon EC2 Linux インスタンスのタイムスタンプを ClockBound と比較することで、イベントの実際の時刻を判断できます。	2021 年 11 月 2 日
AMI を組織および OU と共有	AMI を次の AWS リソースと共有できるようになりました: 組織と組織単位 (OU)	2021 年 10 月 29 日
C6i インスタンス	Intel Xeon スケーラブルプロセッサ (Ice Lake) を搭載した、新しいコンピューティング最適化インスタンス。	2021 年 10 月 28 日
スポットプレイスメントスコア	スポットキャパシティ要件に基づく AWS リージョンまたはアベイラビリティーゾーンのレコメンデーションを入手します。	2021 年 10 月 27 日

スポットフリートの属性ベースのインスタンスタイプの選択	インスタンスが持つ必要がある属性を指定すると、Amazon EC2 はそれらの属性を持つすべてのインスタンスタイプを識別します。	2021 年 10 月 27 日
EC2 フリートの属性ベースのインスタンスタイプの選択	インスタンスが持つ必要がある属性を指定すると、Amazon EC2 はそれらの属性を持つすべてのインスタンスタイプを識別します。	2021 年 10 月 27 日
新しいローカルゾーンを追加	ラスベガス、ニューヨーク市、ポートランドに Local Zones を追加。	2021 年 10 月 26 日
DL1 インスタンス	Habana Gaudi アクセラレータとインテル Xeon Platinum プロセッサー (Cascade Lake) を搭載した、新しい高速コンピューティングインスタンス。	2021 年 10 月 26 日
オンデマンドのキャパシティー予約フリート	キャパシティー予約フリートを使用して、キャパシティー予約のグループまたはフリートを起動できます。	2021 年 10 月 5 日
Ubuntu 20.04 LTS での休止状態のサポート – Focal	新しく Ubuntu 20.04 LTS から起動されたインスタンスでは休止状態が利用可能です – Focal AMI。	2021 年 10 月 4 日
EC2 フリートとターゲットを絞ったオンデマンドキャパシティ予約	EC2 フリートは、オンデマンドインスタンスを`targeted` Capacity Reservations に起動することができます。	2021 年 9 月 22 日

Dedicated Hosts での T3 インスタンス	Amazon EC2 Dedicated Hosts での T3 インスタンスのサポート。	2021 年 9 月 14 日
VT1 インスタンス	新しい高速コンピューティングインスタンスは Xilinx Alveo U30 メディアアクセラレーターを使用し、ライブビデオトランスコードのワークロード向けに設計されています。	2021 年 9 月 13 日
RHEL、Fedora、および CentOS の休止状態のサポート	RHEL、Fedora、および CentOS AMI から起動された新しく起動されたインスタンスを休止状態にします。	2021 年 9 月 9 日
新しいローカルゾーンを追加	シカゴ、ミネアポリス、カンザスシティにLocal Zones を追加します。	2021 年 9 月 8 日
Amazon EC2 グローバルビュー	Amazon EC2 グローバルビューを使用すると、複数の AWSリージョンの VPC、サブネット、インスタンス、セキュリティグループ、およびボリュームを 1 つのコンソールで表示します。	2021 年 9 月 1 日
Amazon Data Lifecycle Manager の AMI の廃止サポート	Amazon Data Lifecycle Manager EBS-backed AMI ポリシーは、AMI を非推奨にすることができます。AWS DataLifecycleManagerServiceRoleForAMIManagementAWS 管理ポリシーが更新され、この機能がサポートされました。	2021 年 8 月 23 日

C5d、M5d、R5d の休止状態のサポート	C5d、M5d、R5d インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2021 年 8 月 19 日
Amazon EC2 のキーペア	Amazon EC2 は、Linux および Mac インスタンスで ED25519 キーをサポートするようになりました。	2021 年 8 月 17 日
M6i インスタンス	第 3 世代 Intel Xeon スケーラブルプロセッサー (Ice Lake) を搭載した、新しい汎用インスタンス。	2021 年 8 月 16 日
Amazon Data Lifecycle Manager の CloudWatch メトリクス	Amazon CloudWatch を使用して、Amazon Data Lifecycle Manager のポリシーをモニタリングできます。	2021 年 7 月 28 日
新しいローカルゾーンを追加	デンバーのローカルゾーンを追加します。	2021 年 7 月 27 日
EBS ダイレクト API の CloudTrail データイベント	ListSnapshotBlocks、ListChangedBlocks、GetSnapshotBlockおよびPutSnapshotBlock の API は、CloudTrail 内のデータイベントをログに記録することができます。	2021 年 7 月 27 日
ネットワークインターフェイスのプレフィクス	プライベート IPv4 または IPv6 CIDR 範囲は、自動または手動で、ネットワークインターフェイスに割り当てることができます。	2021 年 7 月 22 日

io2 Block Express ボリューム	io2 Block Express ボリュームは、R5b インスタンスをサポートするすべてのリージョンとアベイラビリティーゾーンで一般利用が可能になりました。	2021 年 7 月 19 日
イベント Windows	スケジュールされたイベントに対して、週ごとに繰り返されるカスタムのイベントウィンドウを定義して、Amazon EC2 インスタンスを再起動、停止、終了させることができます。	2021 年 7 月 15 日
セキュリティグループルールのリソース ID とタグ付けについてのサポート	リソース ID により、セキュリティグループルールを参照することができます。また、セキュリティグループにはタグも追加できます。	2021 年 7 月 7 日
新しいローカルゾーンを追加	ダラスおよびフィラデルフィアに Local Zones が追加されます。	2021 年 7 月 7 日
AMI を非推奨にする	AMI を非推奨にするタイミングを指定できるようになりました。	2021 年 6 月 11 日
Windows に対する 1 秒単位の課金	Amazon EC2 では、Windows および SQL Server ベースの使用量に対し、秒単位で課金 (最低料金 1 分間分) されます。	2021 年 6 月 10 日
AWS Outposts でのキャパシティ予約	AWS Outposts で、キャパシティー予約を使用できるようになりました。	2021 年 5 月 24 日

キャパシティ予約の共有	Local Zones および Wavelength Zones で作成されたキャパシティーの予約を共有できるようになりました。	2021 年 5 月 24 日
ハイメモリ仮想インスタンス	大規模メモリ内データベースを実行するように設計された、ハイメモリ仮想インスタンス。新しいタイプは、u-6tb1.56xlarge、u-6tb1.112xlarge、u-9tb1.112xlarge、u-12tb1.112xlarge です。	2021 年 5 月 11 日
ルートボリュームの置換	これで、ルートボリューム置換タスクを使用して、実行中のインスタンスのルート EBS ボリュームを置き換えることができます。	2021 年 4 月 22 日
S3 を使用して AMI を保存および復元する	EBS-backed AMI を S3 に保存し、S3 から復元して AMI のクロスパーティションコピーを有効にします。	2021 年 4 月 6 日
EC2 シリアルコンソール	インスタンスのシリアルポートへの接続を確立することにより、起動およびネットワーク接続の問題をトラブルシューティングします。	2021 年 3 月 30 日
ブートモード	Amazon EC2で、選択した AMD および Intel ベースの EC2 インスタンス上で、UEFI ブートがサポートされるようになりました。	2021 年 3 月 22 日

X2gd インスタンス	メモリに最適化されたこの新しいインスタンスでは、64 ビット Arm アーキテクチャベースの AWS Graviton2 プロセッサを搭載しています。	2021 年 3 月 16 日
Amazon EBS local snapshots on Outposts	Outposts の Amazon EBS ローカルスナップショットを使用して、Outpost のボリュームのスナップショットを、その Outpost に置かれた Amazon S3 内でローカルに保存できるようになりました。	2021 年 2 月 4 日
逆引き DNS レコードを作成する	Elastic IP アドレス用に、逆引き DNS ルックアップを設定できるようになりました。	2021 年 2 月 3 日
io2ボリュームに対するマルチアタッチのサポート	Amazon EBS マルチアタッチのプロビジョンド IOPS SSD (io2) ボリュームを有効にできるようになりました。	2020 年 12 月 18 日
C6gn インスタンス	コンピューティングに最適化されたこの新しいインスタンスは、64 ビット Arm アーキテクチャベースの AWS Graviton2 プロセッサを搭載しています。このインスタンスは、最大 100 Gbps のネットワーク帯域幅を利用します。	2020 年 12 月 18 日
Amazon Data Lifecycle Manager	Amazon Data Lifecycle Manager を使用して、AWS アカウント間でスナップショットを共有しコピーするプロセスを自動化します。	2020 年 12 月 17 日

G4ad インスタンス	新しいインスタンスは、AMD Radeon Pro V520 GPU と AMD 2nd Generation EPYC プロセッサを搭載しています。	2020 年 12 月 9 日
AMI 作成時の AMI とスナップショットのタグ付け	AMI を作成するときに、AMI とスナップショットに同じタグを付けるか、異なるタグでタグを付けることができます。	2020 年 12 月 4 日
io2 Block Express プレビュー	io2 Block Express ボリュームプレビューにオプトインできます。io2Block Express ボリュームは、サブミリ秒のレイテンシーを提供し、io2 ボリュームよりも高い IOPS、高いスループット、より大規模な容量をサポートします。	2020 年 12 月 1 日
gp3 ボリューム	新しい Amazon EBS 汎用 SSD ボリュームタイプ。ボリュームを作成または変更するときに、プロビジョンド IOPS とスループットを指定できます。	2020 年 12 月 1 日
D3、D3en、M5zn、R5b インスタンス	Nitro System 上に構築された新しいインスタンスタイプ。	2020 年 12 月 1 日
スループット最適化 HDD ボリュームサイズと Cold HDD ボリュームサイズ	スループット最適化 HDD (st1) ボリュームおよび Cold HDD (sc1) ボリュームのサイズは 125 GiB から 16 TiB です。	2020 年 11 月 30 日

Mac1 インスタンス	Amazon EC2 での macOS ワークロードの実行をサポートする Apple Mac ミニコンピュータ上に構築された新しいインスタンス。	2020 年 11 月 30 日
Amazon EventBridge を使用したスポットフリートイベントのモニタリング	スポットフリート状態の変更やエラーに応じてプログラムによるアクションをトリガーする EventBridge ルールを作成します。	2020 年 11 月 20 日
Amazon EventBridge を使用した EC2 フリートイベントのモニタリング	EC2 フリート状態の変更やエラーに応じてプログラムによるアクションをトリガーする EventBridge ルールを作成します。	2020 年 11 月 20 日
`instant`フリートの削除	タイプ EC2 フリート の `instant` を削除し、1 回の API 呼び出しでフリート内のすべてのインスタンスを終了します。	2020 年 11 月 18 日
T3 および T3a の休止のサポート	T3 および T3a インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2020 年 11 月 17 日
Amazon EFS の簡易版の作成	Amazon EFS の簡易版を使用して、起動時にインスタンスに Amazon Elastic File System (Amazon EFS) ファイルシステムを作成してマウントできます。	2020 年 11 月 9 日

Amazon Data Lifecycle Manager	Amazon Data Lifecycle Manager を使用して、EBS-backed AMI の作成、保持、削除を自動化できます。	2020 年 11 月 9 日
インスタンスメタデータカテゴリ: イベント/レコメンデーション/再調整	EC2 インスタンスの再調整推奨通知がインスタンスに対して送信されるおおよその時間 (UTC)。	2020 年 11 月 4 日
EC2 インスタンスの再調整に関するレコメンデーション	スポットインスタンスが中断するリスクが高い場合に通知するシグナル	2020 年 11 月 4 日
Wavelength Zone 内のキャパシティ予約	Wavelength Zone で キャパシティーの予約 を作成して使用できるようになりました。	2020 年 11 月 4 日
キャパシティーの再調整	Amazon EC2 がリバランス推奨を発行したときに、代替スポットインスタンスを起動するようにスポットフリートまたは EC2 フリートを設定します。	2020 年 11 月 4 日
P4d インスタンス	新しい高速コンピューティングインスタンスは、機械学習と HPC ワークロードに高性能プラットフォームを提供します。	2020 年 11 月 2 日
I3、M5ad、および R5ad の休止状態のサポート	I3、M5ad、R5ad インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2020 年 10 月 21 日

スポットインスタンスの vCPU 制限	スポットインスタンスの上限は、実行中のスポットインスタンスが使用していている vCPU の数、または未処理のリクエストが完了するまでに使用する vCPU の数に関して、管理されるようになりました。	2020 年 10 月 1 日
ローカルゾーンでのキャパシティ予約	Local Zones でキャパシティーの予約を作成して使用できるようになりました。	2020 年 9 月 30 日
Amazon Data Lifecycle Manager	Amazon Data Lifecycle Manager ポリシーは、最大 4 つのスケジュールで設定できます。	2020 年 9 月 17 日
T4g インスタンス	AWS Graviton2 プロセッサ搭載の新しい汎用インスタンスは、64 ビットの Arm Neoverse コアと AWS が設計したカスタムシリコンをベースとしており、パフォーマンスおよびコストを最適化します。	2020 年 9 月 14 日
M5a および R5a の休止状態のサポート	M5a インスタンスタイプと R5a インスタンスタイプで実行されている新しく起動したインスタンスを休止状態にします。	2020 年 8 月 28 日

Amazon EBS のプロビジョンド IOPS SSD (io2) ボリューム	プロビジョンド IOPS SSD (io2) ボリュームは、0.001% 以下の AFR で 99.999% のボリューム耐久性を提供するように設計されています。	2020 年 8 月 24 日
インスタンスメタデータにより、インスタンスの場所と配置情報を提供	`placement` カテゴリの新しいインスタンスメタデータフィールド: リージョン、プレイスメントグループ名、パーティション番号、ホスト ID、アベイラビリティーゾーン ID。	2020 年 8 月 24 日
C5ad インスタンス	第 2 世代 AMD EPYC プロセッサを搭載した新しいコンピューティング最適化インスタンス。	2020 年 8 月 13 日
Wavelength Zone	Wavelength Zone は、Wavelength インフラストラクチャをデプロイする先のキャリアロケーション内の独立したゾーンです。	2020 年 8 月 6 日
キャパシティ予約グループ	AWS Resource Groups を使用してキャパシティー予約の論理コレクションを作成し、それらのグループ内でターゲットインスタンスを起動できます。	2020 年 7 月 29 日

C6gd、M6gd、および R6gd インスタンス	AWS Graviton2 プロセッサ搭載の新しい汎用インスタンスは、64 ビットの Arm Neoverse コアと AWS が設計したカスタムシリコンをベースとしており、パフォーマンスおよびコストを最適化します。	2020 年 7 月 27 日
高速スナップショット復元	共有しているスナップショットに対して高速スナップショット復元を有効にすることができます。	2020 年 7 月 21 日
C6g インスタンスと R6g インスタンス	AWS Graviton2 プロセッサ搭載の新しい汎用インスタンスは、64 ビットの Arm Neoverse コアと AWS が設計したカスタムシリコンをベースとしており、パフォーマンスおよびコストを最適化します。	2020 年 6 月 10 日
G4dn のベアメタルインスタンス	ホストサーバーの物理リソースにアプリケーションが直接アクセスできるようにする新しいインスタンス。	2020 年 6 月 5 日
C5a インスタンス	第 2 世代 AMD EPYC プロセッサを搭載した新しいコンピューティング最適化インスタンス。	2020 年 6 月 4 日
自分の IPv6 アドレスを使用する	自分の IPv6 アドレス範囲の一部またはすべてを、オンプレミスのネットワークから AWS アカウントに導入できます。	2020 年 5 月 21 日

M6g インスタンス	AWS Graviton2 プロセッサ搭載の新しい汎用インスタンスは、64 ビットの Arm Neoverse コアと AWS が設計したカスタムシリコンをベースとしており、パフォーマンスおよびコストを最適化します。	2020 年 5 月 11 日
Systems Manager パラメータを使用してインスタンスを起動する	インスタンスの起動時に、AMI の代わりに AWS Systems Manager パラメータを指定できます。	2020 年 5 月 5 日
スケジュールされたイベント通知のカスタマイズ	スケジュールされたイベント通知をカスタマイズして、メール通知にタグを含めることができます。	2020 年 5 月 4 日
Amazon Linux 2 カーネルライブパッチ	Amazon Linux 2 のカーネルライブパッチを使用すると、実行中のアプリケーションを再起動や中断せずに、実行中の Linux カーネルにセキュリティの脆弱性や重大なバグのパッチを適用することができます。	2020 年 4 月 28 日
Amazon EBS マルチアタッチ	単一のプロビジョンド IOPS SSD (io1) ボリュームを、同じアベイラビリティーゾーンにある最大 16 の Nitro ベースのインスタンスにアタッチできるようになりました。	2020 年 2 月 14 日

スポットインスタンスの停止と開始	停止中断動作に依存するのではなく、Amazon EBS によってバックアップされたスポットインスタンスを停止し、いつでも開始します。	2020 年 1 月 13 日
リソースへのタグ付け	Egress-only インターネットゲートウェイ、ローカルゲートウェイ、ローカルゲートウェイルートテーブル、ローカルゲートウェイ仮想インターフェイス、ローカルゲートウェイ仮想インターフェイスグループ、ローカルゲートウェイルートテーブル VPC の関連付け、およびローカルゲートウェイルートテーブル仮想インターフェイスグループの関連付けをタグ付けできます。	2020 年 1 月 10 日
Session Manager を使用してインスタンスに接続する	Amazon EC2 コンソールからインスタンスで Session Manager セッションを開始できます。	2019 年 12 月 18 日
Inf1 インスタンス	AWS Inferentia を搭載した新しいインスタンス。AWS Inferentia は、低コストで高いパフォーマンスを実現するように設計された機械学習推論チップです。	2019 年 12 月 3 日
Dedicated Hosts およびホストリソースグループ	Dedicated Hosts がホストリソースグループで使用できるようになりました。	2019 年 12 月 2 日

Dedicated Hosts 共有	AWS アカウント間で、Dedicated Hosts を共有できるようになりました。	2019 年 12 月 2 日
アカウントレベルでのデフォルトのクレジット指定	AWS リージョンごとにアカウントレベルで、バースト可能な各パフォーマンスインスタンスファミリーに対し、デフォルトのクレジット仕様を設定できます。	2019 年 11 月 25 日
インスタンスタイプの検出	ニーズに合ったインスタンスタイプを見つけることができます。	2019 年 11 月 22 日
Dedicated Hosts	インスタンスファミリー内にある複数のインスタンスタイプをサポートするようにDedicated Hostを設定できるようになりました。	2019 年 11 月 21 日
Amazon EBS 高速スナップショット復元	EBS スナップショットに対して高速スナップショット復元を有効にすることができます。これにより、スナップショットから作成された EBS ボリュームは、作成時に完全に初期化された状態になり、プロビジョンドパフォーマンスをすべて即座に提供できます。	2019 年 11 月 20 日
インスタンスメタデータサービスバージョン 2	インスタンスメタデータのリクエストに、セッション志向な方法であるインスタンスメタデータサービスバージョン2を使用できます。	2019 年 11 月 19 日

Elastic Fabric Adapter	Elastic Fabric Adapters を インテル MPI 2019 Update 6 と併用できるようになりました。	2019 年 11 月 15 日
リザーブドインスタンスの購入予約のキュー登録	リザーブドインスタンス の購入予約を最大 3 年先までキューに入れることができます。	2019 年 10 月 4 日
G4dn インスタンス	NVIDIA Tesla GPU の新しいインスタンス	2019 年 9 月 19 日
診断割り込み	到達できないまたは応答しないインスタンスに診断割り込みを送信して、カーネルパニックを手動でトリガーできます。	2019 年 8 月 14 日
容量が最適化された割り当て戦略	EC2 フリートまたはスポットフリートを使用して、起動するインスタンス数に最適な容量で、スポットプールからスポットインスタンスを起動できます。	2019 年 8 月 12 日
オンデマンドによるキャパシティ予約の共有	AWS アカウント間で、キャパシティーの予約を共有できるようになりました。	2019 年 7 月 29 日
Elastic Fabric Adapter	EFA では、Open MPI 3.1.4 および Intel MPI 2019 Update 4 がサポートされるようになりました。	2019 年 7 月 26 日
リソースへのタグ付け	作成時にテンプレートを起動します。	2019 年 7 月 24 日

EC2 Instance Connect	EC2 Instance Connect は、Secure Shell (SSH) を使用してインスタンスに接続するシンプルで安全な方法です。	2019 年 6 月 27 日
ホスト復旧	Dedicated Host で予期しないハードウェア障害が発生した場合にインスタンスを新しいホストで自動的に再起動します。	2019 年 6 月 5 日
Amazon EBS マルチボリュームスナップショット	EC2 インスタンスにアタッチされている複数の EBS ボリューム間で、正確なポイントインタイムで、データ調整済みの Crash-consistent スナップショットを取得できます。	2019 年 5 月 29 日
リソースへのタグ付け	Dedicated Host の予約 にタグ付けできます。	2019 年 5 月 27 日
デフォルトでの Amazon EBS 暗号化	リージョンでデフォルトで暗号化を有効にすると、そのリージョンで作成するすべての新しい EBS ボリュームは EBS 暗号化のデフォルトの KMS キー を使用して暗号化されます。	2019 年 5 月 23 日
リソースへのタグ付け	VPC エンドポイント、エンドポイントサービス、およびエンドポイントサービス設定にタグ付けできます。	2019 年 5 月 13 日

Microsoft SQL Server データベースでの Windows から Linux へのプラットフォーム変更アシスタント	既存の Microsoft SQL Server ワークロードを Windows から Linux オペレーティングシステムに移行します。	2019 年 5 月 8 日
I3en インスタンス	新しい I3en インスタンスは、最大 100 Gbps のネットワーク帯域幅を利用します。	2019 年 5 月 8 日
Elastic Fabric Adapter	High Performance Computing (HPC) アプリケーションを高速化するには、Elastic Fabric Adapter をインスタンスに接続します。	2019 年 4 月 29 日
T3a インスタンス	AMD EPYC プロセッサを搭載した新しいインスタンス。	2019 年 4 月 24 日
M5ad インスタンスと R5ad インスタンス	AMD EPYC プロセッサを搭載した新しいインスタンス。	2019 年 3 月 27 日
リソースへのタグ付け	Dedicated Host 予約に任意のタグを割り当てることで、さまざまな方法で分類できます。	2019 年 3 月 14 日
M5、M5d、R5、R5d、および z1d のベアメタルインスタンス	ホストサーバーの物理リソースにアプリケーションが直接アクセスできるようにする新しいインスタンス。	2019 年 2 月 13 日

前年の履歴

次の表は、2018 年以前の Amazon EC2 ドキュメントへの重要な追加項目をまとめたものです。

機能	API バージョン	説明	リリース日
パーティションプレイスメントグループ	2016-11-15	パーティションプレイスメントグループはインスタンスを複数の論理パーティションに分散させ、基盤となるハードウェアを 1 つのパーティション内のインスタンスが他のパーティション内のインスタンスと共有しないようにします。詳細については、「パーティションプレイスメントグループ」を参照してください。	2018 年 12 月 20 日
p3dn.24xlarge インスタンス	2016-11-15	新しい p3dn.24xlarge インスタンスは 100 Gbps のネットワーク帯域幅を提供します。	2018 年 12 月 07 日
EC2 Linux インスタンスの休止	2016-11-15	休止が有効になっており、前提条件を満たしている場合は、Linux インスタンスを休止状態にすることができます。詳細については、「Amazon EC2 インスタンスの休止」を参照してください。	2018 年 11 月 28 日
Amazon Elastic Inference アクセラレーター	2016-11-15	Amazon EI アクセラレーターをインスタンスにアタッチし、GPU アクセラレーションを追加することで、深層学習の推論を実行するコストを削減できます。詳細については、「Amazon Elastic Inference」を参照してください。	2018 年 11 月 28 日
100 Gbps のネットワーク帯域幅のインスタンス	2016-11-15	新しい C5n インスタンスは、最大 100 Gbps のネットワーク帯域幅を利用します。	2018 年 11 月 26 日
Arm ベースプロセッサのインスタンス	2016-11-15	新しい A1 インスタンスは、コストを大幅に削減し、スケールアウトや Arm ベースのワークロードに最適です。	2018 年 11 月 26 日
スポットコンソールはインスタンス群を推奨します	2016-11-15	スポットコンソールは、アプリケーションのニーズに合わせた最小限のハードウェア仕様 (vCPU、メモリ、およびストレージ) を満たす	2018 年 11 月 20 日

機能	API バージョン	説明	リリース日
		ために、スポットのベストプラクティス (インスタンスの多様化) に基づいたインスタンス群を推奨します。詳細については、「スポットフリートリクエストを作成します。」を参照してください。	
新しい EC2 フリート リクエストタイプ: `instant`	2016-11-15	EC2 フリート は、インスタンスタイプと購入モデル全体で同時にキャパシティーをプロビジョニングするために使用できる新しいリクエストタイプ、`instant` をサポートしています。`instant` リクエストは、インスタンスを起動するかどうかおよびいつ起動するかについて制御する、API レスポンスで起動されたインスタンスを返します。それ以上のアクションは実行しません。詳細については、「EC2 フリートのリクエストタイプ」を参照してください。	2018 年 11 月 14 日
AMD EPYC プロセッサを搭載したインスタンス	2016-11-15	新しい 汎用 (M5a) およびメモリ最適化インスタンス (R5a) は、マイクロサービス、中小規模のデータベース、仮想デスクトップ、開発およびテスト環境、ビジネスアプリケーションなどに低価格オプションを提供します。	2018 年 11 月 06 日
スポット削減額情報	2016-11-15	スポットインスタンスを 1 つのスポットフリートまたはすべてのスポットインスタンスに対して使用することで得られる節約額を確認できます。詳細については、「スポットインスタンス購入による削減額」を参照してください。	2018 年 11 月 05 日
CPU オプションを最適化するためのコンソールでのサポート	2016-11-15	インスタンスを起動するとき、Amazon EC2 を使用して特定のワークロードやビジネスニーズに合うように CPU オプションを最適化できます。詳細については、「CPU オプションの最適化」を参照してください。	2018 年 10 月 31 日

機能	API バージョン	説明	リリース日
インスタンスから起動テンプレートを作成するためのコンソールでのサポート	2016-11-15	Amazon EC2 コンソールを使用した新しい起動テンプレートのためのベースとしてインスタンスを使用して起動テンプレートを作成できます。詳細については、「起動テンプレートの作成」を参照してください。	2018 年 10 月 30 日
On-Demand Capacity Reservations	2016-11-15	特定のアベイラビリティーゾーンの Amazon EC2 インスタンスに対して任意の期間キャパシティーを予約できます。これにより、リザーブドインスタンス (RI) が提供する請求割引とは独立して、キャパシティー予約を登録および管理することができます。詳細については、「On-Demand Capacity Reservations」を参照してください。	2018 年 10 月 25 日
自分の IP アドレスを使用する (BYOIP)	2016-11-15	すべての公開 IPv4 アドレス範囲の一部またはすべてを、オンプレミスのネットワークから AWS アカウントに導入できます。アドレス範囲を AWS に設定すると、そのアドレス範囲はアドレスプールとしてアカウントに表示されます。アドレスプールから Elastic IP アドレスを作成し、それを AWS リソースで使用できます。詳細については、「Amazon EC2 で自分の IP アドレスを使用する (BYOIP)」を参照してください。	2018 年 10 月 23 日
g3s.xlarge インスタンス	2016-11-15	g3s.xlarge インスタンスの導入により、Accelerated Computing G3 インスタンスファミリーの範囲を拡張します。	2018 年 10 月 11 日

機能	API バージョン	説明	リリース日
作成時の Dedicated Host タグとコンソールのサポート	2016-11-15	作成時に Dedicated Hosts タグを付けることができ、Amazon EC2 コンソールを使用して Dedicated Host タグを管理できます。詳細については、「Dedicated Hosts の割り当て」を参照してください。	2018 年 10 月 08 日
ハイメモリインスタンス	2016-11-15	これらのインスタンスは、大きなメモリ内のデータベースを実行するように設計されています。ホストハードウェアに直接アクセスできるベアメタルパフォーマンスを提供します。詳細については、「メモリ最適化インスタンス」を参照してください。	2018 年 9 月 27 日
f1.4xlarge インスタンス	2016-11-15	f1.4xlarge インスタンスの導入により、Accelerated Computing F1 インスタンスファミリーの範囲を拡張します。	2018 年 9 月 25 日
スポットフリートのスケジュールされたスケーリングのコンソールサポート	2016-11-15	日付と時刻に基づいて、フリート現在の容量を増減させます。詳細については、「スケジュールに基づくスケーリングを使用して、スポットフリートをスケーリングします。」を参照してください。	2018 年 9 月 20 日
T3 インスタンス	2016-11-15	T3 インスタンスは、ベースラインレベルの CPU パフォーマンスを実現する、バースト可能な汎用インスタンスタイプです。いつでも必要な時間だけ CPU 使用率をバーストさせる機能を備えています。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2018 年 8 月 21 日

機能	API バージョン	説明	リリース日
EC2 フリート の配分戦略	2016-11-15	オンデマンド容量を料金 (最初に最低価格) または優先度 (最初に最も高い優先度) に従って達成するかどうかを指定できます。ターゲットスポット容量を割り当てる先のスポットプールの数を指定できます。詳細については、「スポットインスタンスの配分戦略」を参照してください。	2018 年 7 月 26 日
スポットフリート の配分戦略	2016-11-15	オンデマンド容量を料金 (最初に最低価格) または優先度 (最初に最も高い優先度) に従って達成するかどうかを指定できます。ターゲットスポット容量を割り当てる先のスポットプールの数を指定できます。詳細については、「スポットインスタンスの配分戦略」を参照してください。	2018 年 7 月 26 日
R5 および R5d インスタンス	2016-11-15	R5 および R5d インスタンスは、ハイパフォーマンスのデータベース、分散型のインメモリキャッシュ、およびインメモリ分析に最適です。R5d インスタンスは、NVMe インスタンスストアボリュームに付属しています。詳細については、「メモリ最適化インスタンス」を参照してください。	2018 年 7 月 25 日
z1d インスタンス	2016-11-15	これらのインスタンスは、Electronic Design Automation (EDA) やリレーショナルデータベースなど、コア単位のハイパフォーマンスと大容量のメモリを必要とする用途向けに設計されています。これらのインスタンスは、NVME インスタンスストアボリュームに付属しています。詳細については、「メモリ最適化インスタンス」を参照してください。	2018 年 7 月 25 日

機能	API バージョン	説明	リリース日
スナップショットライフサイクルの自動化	2016-11-15	Amazon Data Lifecycle Manager を使用して、EBS ボリュームをバックアップするスナップショットの作成と削除を自動化できます。詳細については、「Amazon Data Lifecycle Manager」を参照してください。	2018 年 7 月 12 日
起動テンプレートの CPU オプション	2016-11-15	コマンドラインツールを使用して起動テンプレートを作成すると、特定のワークロードまたはビジネスニーズに合わせて CPU オプションを最適化できます。詳細については、「起動テンプレートの作成」を参照してください。	2018 年 7 月 11 日
Dedicated Hosts のタグ付け	2016-11-15	Dedicated Hosts にタグを付けることができます。詳細については、「Dedicated Hosts のタグ付け」を参照してください。	2018 年 7 月 3 日
`i3.metal` インスタンス	2016-11-15	`i3.metal` インスタンスは、プロセッサとメモリなどのホストサーバーの物理リソースにアプリケーションが直接アクセスできるようにします。詳細については、「ストレージ最適化インスタンス」を参照してください。	2018 年 5 月 17 日
最新のコンソール出力を取得する	2016-11-15	get-console-output AWS CLI コマンドを使用すると、一部のインスタンスタイプの最新のコンソール出力を取得できます。	2018 年 5 月 9 日
CPU オプションの最適化	2016-11-15	インスタンスを起動するとき、特定のワークロードやビジネスニーズに合うように CPU オプションを最適化できます。詳細については、「CPU オプションの最適化」を参照してください。	2018 年 5 月 8 日

機能	API バージョン	説明	リリース日
EC2 Fleet	2016-11-15	EC2 フリートを使用すると、異なる EC2 インスタンスタイプとアベイラビリティーゾーン間、オンデマンドインスタンス、リザーブドインスタンス、スポットインスタンスの購入モデル間でインスタンスのグループを起動できます。詳細については、「EC2 Fleet」を参照してください。	2018 年 5 月 2 日
スポットフリート のオンデマンドインスタンス	2016-11-15	常にインスタンス容量を確保するため、スポットフリートリクエストにオンデマンドキャパシティーのリクエストを含められます。詳細については、「スポットフリート」を参照してください。	2018 年 5 月 2 日
作成中の EBS スナップショットをタグ付けする	2016-11-15	スナップショット作成時にタグを適用できます。	2018 年 4 月 2 日
プレイスメントグループの変更	2016-11-15	インスタンスをプレイスメントグループ内またはプレイスメントグループ外に移動させたり、プレイスメントグループを変更したりできます。詳細については、「インスタンスのプレイスメントグループの変更」を参照してください。	2018 年 3 月 1 日
長いリソース ID	2016-11-15	より多くのリソースタイプに対して長い ID 形式を有効にできます。詳細については、「リソース ID」を参照してください。	2018 年 2 月 9 日
ネットワークパフォーマンスの向上	2016-11-15	クラスタプレイスメントグループ外部のインスタンスにおいて、他のインスタンスまたは Amazon S3 との間でネットワークトラフィックを送信または受信する際に、より大きな帯域幅から利点を得られるようになりました。	2018 年 1 月 24 日

機能	API バージョン	説明	リリース日
Elastic IP アドレスにタグを付ける	2016-11-15	Elastic IP アドレスにタグを付けることができます。詳細については、「Elastic IP アドレスにタグを適用する」を参照してください。	2017 年 12 月 21 日
Amazon Linux 2	2016-11-15	Amazon Linux 2 は、Amazon Linux の新しいバージョンです。アプリケーションの高パフォーマンスで安定した安全な基盤が提供されます。詳細については、「Amazon Linux」を参照してください。	2017 年 12 月 13 日
Amazon Time Sync Service のご紹介	2016-11-15	Amazon Time Sync Service を使用して、インスタンスの正確な時刻を維持できます。詳細については、「Linux インスタンスの時刻の設定」を参照してください。	2017 年 11 月 29 日
T2 無制限	2016-11-15	T2 無制限インスタンスは、ベースラインを超えるレベルで必要なだけバーストさせることができます。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2017 年 11 月 29 日
起動テンプレート	2016-11-15	起動テンプレートにインスタントを起動させるパラメータの全体または一部を含めることで、インスタンス起動のたびに指定する必要がなくなります。詳細については、「起動テンプレートからのインスタンスの起動」を参照してください。	2017 年 11 月 29 日
スプレッドプレイスメント	2016-11-15	スプレッドプレイスメントグループは、少数の重要なインスタンスが互いに分離して保持される必要があるアプリケーションに推奨されます。詳細については、「スプレッドプレイスメントグループ」を参照してください。	2017 年 11 月 29 日

機能	API バージョン	説明	リリース日
H1 インスタンス	2016-11-15	H1 インスタンスは、高パフォーマンスビッグデータワークロードの用に設計されています。詳細については、「ストレージ最適化インスタンス」を参照してください。	2017 年 11 月 28 日
M5 インスタンス	2016-11-15	M5 インスタンスは汎用コンピューティングインスタンスです。コンピューティング、メモリ、ストレージおよびネットワークリソースがバランスよく備わっています。	2017 年 11 月 28 日
スポットインスタンスの休止状態	2016-11-15	スポットサービスは、中断が発生した場合スポットインスタンスを休止させることができます。詳細については、「中断した スポットインスタンス の休止」を参照してください。	2017 年 11 月 28 日
スポットフリートのターゲット追跡	2016-11-15	スポットフリートのターゲット追跡スケーリングポリシーを設定できます。詳細については、「ターゲット追跡ポリシーを使用して、スポットフリートをスケーリングします。」を参照してください。	2017 年 11 月 17 日
スポットフリートは Elastic Load Balancing と統合されます。	2016-11-15	スポットフリートに 1 つ以上のロードバランサーをアタッチできます。	2017 年 11 月 10 日
X1e インスタンス	2016-11-15	X1e インスタンスは、高パフォーマンスのデータベース、メモリ内データベース、またその他のメモリを大量に使用するエンタープライズアプリケーションに最適です。詳細については、「メモリ最適化インスタンス」を参照してください。	2017 年 11 月 28 日

機能	API バージョン	説明	リリース日
C5 インスタンス	2016-11-15	C5 インスタンスは、計算量の多いアプリケーション向けに設計されています。詳細については、「コンピュート最適化インスタンス」を参照してください。	2017 年 11 月 6 日
コンバーティブルリザーブドインスタンス のマージと分割	2016-11-15	2 つ以上の コンバーティブルリザーブドインスタンス を新しい コンバーティブルリザーブドインスタンス に交換 (マージ) することができます。変更プロセスを使って、コンバーティブルリザーブドインスタンス をより小さな予約に分割することもできます。詳細については、「コンバーティブルリザーブドインスタンスの交換」を参照してください。	2017 年 11 月 6 日
P3 インスタンス	2016-11-15	P3 インスタンスは、コンピューティング最適化 GPU インスタンスです。詳細については、「Linux 高速コンピューティングインスタンス」を参照してください。	2017 年 10 月 25 日
VPC のテナント属性を変更する	2016-11-15	VPC インスタンスのテナント属性を `dedicated` から `default` に変更することができます。詳細については、「VPC のテナント属性の変更」を参照してください。	2017 年 10 月 16 日
1 秒単位の請求	2016-11-15	Amazon EC2 は、Linux ベースの秒単位での課金 (最低 1 分間分) を行います。	2017 年 10 月 2 日
中断時に停止	2016-11-15	中断時に Amazon EC2 が スポットインスタンス を停止または終了するかを指定できます。詳細については、「中断動作」を参照してください。	2017 年 9 月 18 日

機能	API バージョン	説明	リリース日
NAT ゲートウェイのタグ付け	2016-11-15	NAT ゲートウェイにタグを付けることができます。詳細については、「リソースのタグ付け」を参照してください。	2017 年 9 月 7 日
セキュリティグループルールの説明	2016-11-15	説明をセキュリティグループに追加できます。詳細については、「セキュリティグループのルール」を参照してください。	2017 年 8 月 31 日
Elastic IP アドレスの復元	2016-11-15	VPC で使用するために Elastic IP アドレスを解放した場合、復元できる可能性があります。詳細については、「Elastic IP アドレスの復元」を参照してください。	2017 年 8 月 11 日
スポットフリートのインスタンスにタグを付けます。	2016-11-15	起動するインスタンスに自動的にタグを付けるように、スポットフリートを設定できます。	2017 年 7 月 24 日
G3 インスタンス	2016-11-15	G3 インスタンスは DirectX または OpenGL を使用してグラフィックアプリケーション向けに費用対効果の高パフォーマンスのプラットフォームを提供します。また、G3 インスタンスは NVIDIA GRID 仮想ワークステーション機能も提供し、最大で 4096x2160 の解像度の 4 つのモニターをサポートします。詳細については、「Linux 高速コンピューティングインスタンス」を参照してください。	2017 年 7 月 13 日
F1 インスタンス	2016-11-15	F1 インスタンスは、高速コンピューティングインスタンスです。詳細については、「Linux 高速コンピューティングインスタンス」を参照してください。	2017 年 4 月 19 日

機能	API バージョン	説明	リリース日
リソース作成時のタグ付け	2016-11-15	インスタンスやボリュームの作成時にタグを適用できます。詳細については、「リソースのタグ付け」を参照してください。さらに、タグベースのリソースレベルアクセス権限を使用して、適用されているタグを制御できます。詳細については、「リソース作成時にタグ付けするアクセス許可の付与」を参照してください。	2017 年 3 月 28 日
I3 インスタンス	2016-11-15	I3 インスタンスは、ストレージ最適化インスタンスです。詳細については、「ストレージ最適化インスタンス」を参照してください。	2017 年 2 月 23 日
アタッチされた EBS ボリュームで変更を行う	2016-11-15	ほとんどの EC2 インスタンスにアタッチされたほとんどの EBS ボリュームでは、ボリュームをデタッチしたりインスタンスを停止したりせずに、ボリュームのサイズ、タイプ、IOPS を変更できます。	2017 年 2 月 13 日
IAM ロールをアタッチする	2016-11-15	既存のインスタンスの IAM ロールをアタッチ、デタッチ、または置換できます。詳細については、「Amazon EC2 の IAM ロール」を参照してください。	2017 年 2 月 9 日
専有 スポットインスタンス	2016-11-15	Virtual Private Cloud (VPC) のシングルテナントハードウェアで、スポットインスタンス を実行できます。詳細については、「スポットインスタンス のテナンシーの指定」を参照してください。	2017 年 1 月 19 日
IPv6 サポート	2016-11-15	VPC とサブネットに IPv6 CIDR を関連付け、VPC のインスタンスに IPv6 アドレスを割り当てることができます。詳細については、「Amazon EC2 インスタンスの IP アドレス指定」を参照してください。	2016 年 12 月 1 日

機能	API バージョン	説明	リリース日
R4 インスタンス	2016-09-15	R4 インスタンスはメモリ最適化インスタンスです。R4 インスタンスは、ビジネスインテリジェンス (BI)、データマイニングや分析、インメモリデータベース、ウェブスケールの分散インメモリキャッシング、構造化されていないビッグデータのリアルタイム処理を実行するアプリケーションのパフォーマンスなど、メモリを大量に消費し、レイテンシーの影響を受けやすいワークロードに最適です。詳細については、「メモリ最適化インスタンス」を参照してください。	2016 年 11 月 30 日
新しい `t2.xlarge` および `t2.2xlarge` インスタンスタイプ	2016-09-15	T2 インスタンスは、適度なパフォーマンスを実現したり、ワークロードの必要に応じて非常に高いパフォーマンスまでバーストする機能を実現できるように設計されています。これらのインスタンスは、応答性、制限された期間における高いパフォーマンス、および低コストを必要とするアプリケーション向けのインスタンスです。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2016 年 11 月 30 日
P2 インスタンス	2016-09-15	P2 インスタンスは NVIDIA Tesla K80 GPU を使用し、CUDA または OpenCL プログラミングモデルを使用する汎用 GPU コンピューティング用に設計されています。詳細については、「Linux 高速コンピューティングインスタンス」を参照してください。	2016 年 29 月 9 日
`m4.16xlarge` インスタンス	2016-04-01	64 個の vCPU と 256 GiB の RAM を備えた `m4.16xlarge` インスタンスの導入に伴い、汎用 M4 ファミリーの範囲を拡張します。	2016 年 6 月 9 日

機能	API バージョン	説明	リリース日
スポットフリートの自動スケーリング		スポットフリートのスケーリングポリシーを設定できるようになりました。詳細については、「スポットフリートの自動スケーリング」を参照してください。	2016 年 9 月 1 日
Elastic Network Adapter (ENA)	2016-04-01	ENA を使用してネットワーキングを強化できるようになりました。詳細については、「拡張ネットワークのサポート」を参照してください。	2016 年 6 月 28 日
長い ID の表示および変更の拡張サポート	2016-04-01	他の IAM ユーザー、IAM ロール、ルートユーザーの長い ID 設定を表示および変更できるようになりました。詳細については、「リソース ID」を参照してください。	2016 年 6 月 23 日
暗号化された Amazon EBS スナップショットと AWS アカウント間のコピー	2016-04-01	AWS アカウント間で、暗号化された EBS スナップショットをコピーできるようになりました。	2016 年 6 月 21 日
インスタンスコンソールのスクリーンショットの取得	2015-10-01	到達不可のインスタンスをデバッグするときに、追加の情報を取得できるようになりました。詳細については、「接続できないインスタンスのスクリーンショットの取得」を参照してください。	2016 年 5 月 24 日
X1 インスタンス	2015-10-01	メモリ内データベース、ビッグデータ処理エンジン、ハイパフォーマンスコンピューティング (HPC) アプリケーションの実行用に設計されたメモリ最適化インスタンス。詳細については、「メモリ最適化インスタンス」を参照してください。	2016 年 5 月 18 日

機能	API バージョン	説明	リリース日
新しい 2 タイプの EBS ボリューム	2015-10-01	スループット最適化 HDD (st1) と Cold HDD (sc1) ボリュームを作成できるようになりました。	2016 年 4 月 19 日
Amazon EC2 用の新しい NetworkPacketsIn と NetworkPacketsOut のメトリクスを追加しました。		Amazon EC2 用の新しい NetworkPacketsIn と NetworkPacketsOut のメトリクスを追加しました。詳細については、「インスタンスメトリクス」を参照してください。	2016 年 3 月 23 日
スポットフリートの CloudWatch メトリクス		スポットフリートの CloudWatch メトリクスを取得できるようになりました。詳細については、「スポットフリートの CloudWatch メトリクス」を参照してください。	2016 年 3 月 21 日
スケジュールされたインスタンス	2015-10-01	スケジュールされたリザーブドインスタンス (スケジュールされたインスタンス) によって、毎日、毎週、または毎月ベースの指定された開始時間および期間で繰り返しキャパシティー予約を購入できます。	2016 年 1 月 13 日
長いリソース ID	2015-10-01	一部の Amazon EC2 および Amazon EBS リソースタイプに、段階的に長い ID を導入しています。オプトイン期間中に、サポートされるリソースタイプに対して長い ID 形式を有効にできます。詳細については、「リソース ID」を参照してください。	2016 年 1 月 13 日
ClassicLink DNS サポート	2015-10-01	VPC の ClassicLink DNS サポートを有効にして、リンクされた EC2-Classic インスタンスと VPC のインスタンス間で対応された DNS ホスト名がプライベート IP アドレスに解決され、パブリック IP アドレスに解決されないようにします。	2016 年 1 月 11 日

機能	API バージョン	説明	リリース日
新しい t2.nano インスタンスタイプ	2015-10-01	T2 インスタンスは、適度なパフォーマンスを実現したり、ワークロードの必要に応じて非常に高いパフォーマンスまでバーストする機能を実現できるように設計されています。これらのインスタンスは、応答性、制限された期間における高いパフォーマンス、および低コストを必要とするアプリケーション向けのインスタンスです。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2015 年 12 月 15 日
Dedicated Host	2015-10-01	Amazon EC2 Dedicated Host は、インスタンス容量を利用したお客様専用の物理サーバーです。詳細については、「Dedicated Hosts」を参照してください。	2015 年 11 月 23 日
スポットインスタンスの継続時間	2015-10-01	スポットインスタンス で実行時間を指定できるようになりました。スポットブロックはサポートされていません (2023 年 1 月)。	2015 年 10 月 6 日
スポットフリートの変更リクエスト	2015-10-01	スポットフリートリクエストのターゲット容量が変更できるようになりました。詳細については、「スポットフリートリクエストを変更します。」を参照してください。	2015 年 9 月 29 日
スポットフリートの分散配分戦略	2015-04-15	単一のスポットフリートリクエストを使用して、複数のスポットプールにスポットインスタンスを分散することができるようになりました。詳細については、「スポットインスタンスの配分戦略」を参照してください。	2015 年 9 月 15 日

機能	API バージョン	説明	リリース日
スポットフリートインスタンスの分量指定	2015-04-15	アプリケーションのパフォーマンスに影響する各インスタンスのキャパシティーユニットを定義し、スポットプールごとに スポットインスタンス の支払料金を調整することができるようになりました。詳細については、「スポットフリートインスタンスの分量指定」を参照してください。	2015 年 8 月 31 日
新しい再起動アラームアクションと、アラームアクションで使用する新しい IAM ロール		再起動アラームアクションと、アラームアクションで使用する新しい IAM ロールが追加されました。詳細については、「インスタンスを停止、終了、再起動、または復旧するアラームを作成する」を参照してください。	2015 年 7 月 23 日
新しい `t2.large` インスタンスタイプ		T2 インスタンスは、適度なパフォーマンスを実現したり、ワークロードの必要に応じて非常に高いパフォーマンスまでバーストする機能を実現できるように設計されています。これらのインスタンスは、応答性、制限された期間における高いパフォーマンス、および低コストを必要とするアプリケーション向けのインスタンスです。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2015 年 6 月 16 日
M4 インスタンス		演算能力、メモリ、およびネットワークリソースを備えた汎用インスタンス。M4 インスタンスは、AVX2 付きのカスタム Intel 2.4 GHz Intel® Xeon® E5 2676v3 (Haswell) プロセッサを使用します。	2015 年 6 月 11 日
Spot Fleets	2015-04-15	個別のスポットインスタンスリクエストを管理する代わりに、スポットインスタンスの集合またはフリートを管理できます。詳細については、「スポットフリート」を参照してください。	2015 年 5 月 18 日

機能	API バージョン	説明	リリース日
Elastic IP アドレスを EC2-Classic に移行する	2015-04-15	EC2-Classic で使用するために割り当てた Elastic IP アドレスを、VPC で使用するために移行できます。	2015 年 5 月 15 日
複数のディスクで構成された VM を AMI としてインポート	2015-03-01	VM Import プロセスにより、複数のディスクで構成された VM を AMI としてインポートできるようになりました。詳細については、VM Import/Export ユーザーガイドの「VM Import/Export を使用してイメージとして VM をインポート」を参照してください。	2015 年 4 月 23 日
新しい g2.8xlarge インスタンスタイプ		新しい g2.8xlarge インスタンスは 4 つの高性能な NVIDIA GPU を利用し、大規模なレンダリング、トランスコーディング、機械学習、および超並列処理能力を必要とするその他のサーバー側ワークロードを含む GPU コンピューティングワークロードに最適です。	2015 年 4 月 7 日

機能	API バージョン	説明	リリース日
D2 インスタンス		直接接続型インスタンスストレージにある大量のデータへのシーケンシャルアクセスを必要とするアプリケーションに最適化された、高密度ストレージインスタンスです。D2 インスタンスは、高密度ストレージファミリーで最適な料金とパフォーマンスを提供するように設計されています。2.4 GHz Intel® Xeon® E5 2676v3 (Haswell) プロセッサーを利用する D2 インスタンスは、追加の処理能力、より多くのメモリ、拡張ネットワーキングを提供して HS1 インスタンスでのパフォーマンスを向上させます。さらに、D2 インスタンスには 4 つのインスタンスサイズがあり、6 TB、12 TB、24 TB、48 TB のストレージオプションを利用できます。 詳細については、「ストレージ最適化インスタンス」を参照してください。	2015 年 3 月 24 日
EC2 インスタンスの自動復旧		Amazon EC2 インスタンスをモニタリングする Amazon CloudWatch アラームを作成できます。この機能により、基盤ハードウェアの障害や、AWS による修復を必要とする問題によりインスタンスが正常に機能しなくなった場合に、自動的にそのインスタンスを復旧できます。復旧されたインスタンスは、インスタンス ID、IP アドレス、すべてのインスタンスメタデータを含め、元のインスタンスと同じです。 詳細については、「インスタンスの復旧」を参照してください。	2015 年 1 月 12 日

機能	API バージョン	説明	リリース日
C4 インスタンス		次世代のコンピューティングが最適化されたインスタンスは、経済的な価格で、非常に高い CPU パフォーマンスを提供します。C4 インスタンスは、独自の 2.9 GHz Intel® Xeon® E5-2666 v3 (Haswell) プロセッサで動作します。さらに、Turbo Boost を使用して、C4 インスタンスのプロセッサのクロック速度を、1 個または 2 個のコアで 3.5GHz まで引き上げることができます。C3 コンピューティング最適化インスタンスの機能を拡張することによって、C4 インスタンスは、EC2 インスタンスで最高のプロセッサパフォーマンスをお客様に提供します。これらのインスタンスは、通信量の多いウェブアプリケーション、広告配信、バッチ処理、動画の暗号化、分散分析、高エネルギー物理学、ゲノム分析、計算流体力学に最適です。 詳細については、「コンピュート最適化インスタンス」を参照してください。	2015 年 1 月 11 日
ClassicLink	2014-10-01	ClassicLink を使用すると、EC2-Classic インスタンスをアカウント内の VPC にリンクできます。これによって、VPC のセキュリティグループを EC2-Classic インスタンスに関連付け、プライベート IP アドレスを使用して EC2-Classic インスタンスと VPC 内のインスタンスが通信できるようになります。	2015 年 1 月 7 日

機能	API バージョン	説明	リリース日
スポットインスタンスの終了通知		スポットインスタンスの中断から保護する最善の方法は、アプリケーションを耐障害性のある設計にすることです。さらに、スポットインスタンスの終了通知機能を活用できます。これは Amazon EC2 がスポットインスタンスを終了する 2 分前に警告を発します。 詳細については、「スポットインスタンスの中断通知」を参照してください。	2015 年 1 月 5 日
DescribeVolumes ページ分割サポート	2014-09-01	DescribeVolumes API 呼び出しで、MaxResults パラメータと NextToken パラメータによる結果のページ分割がサポートされるようになりました。詳細については、Amazon EC2 API Reference の「DescribeVolumes」を参照してください。	2014 年 10 月 23 日
T2 インスタンス	2014-06-15	T2 インスタンスは、適度なパフォーマンスを実現したり、ワークロードの必要に応じて非常に高いパフォーマンスまでバーストする機能を実現できるように設計されています。これらのインスタンスは、応答性、制限された期間における高いパフォーマンス、および低コストを必要とするアプリケーション向けのインスタンスです。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2014 年 6 月 30 日
新しい [EC2 Service Limits] ページ		Amazon EC2 コンソールの [EC2 Service Limits] ページを使用して、Amazon EC2 や Amazon VPC から提供されるリソースに対するリージョンごとの現在の制限を表示できます。	2014 年 6 月 19 日

機能	API バージョン	説明	リリース日
Amazon EBS 汎用 SSD ボリューム	2014-05-01	汎用 SSD ボリュームは、さまざまなワークロードに対応できるコスト効率の高いストレージとして使用できます。これらのボリュームは、1 桁ミリ秒のレイテンシー、長時間にわたる 3,000 IOPS へのバースト機能、最大 3 IOPS/GiB のベースパフォーマンスを実現しています。汎用 SSD ボリュームのサイズ範囲は、1 GiB ~ 1 TiB です。	2014 年 6 月 16 日
Amazon EBS encryption	2014-05-01	Amazon EBS 暗号化 により、EBS データボリュームとスナップショットがシームレスに暗号化されるため、セキュアキー管理インフラストラクチャを構築および維持する必要がなくなります。EBS 暗号化サービスは、AWS マネージドキー を使用してデータを暗号化することにより、保管中のデータのセキュリティを確保します。EC2 インスタンスをホストするサーバーで暗号化が行われるため、EC2 インスタンスと EBS ストレージとの間を移動するデータが暗号化されます。	2014 年 5 月 21 日

機能	API バージョン	説明	リリース日
R3 インスタンス	2014-02-01	RAM の GiB 当たりの料金が最適に設定された高パフォーマンスのメモリ最適化インスタンス。これらのインスタンスは、R3 インスタンスの vCPU ごとの大容量メモリ、優れた処理能力、拡張ネットワーキング機能によって恩恵を受けるリレーショナルおよび NoSQL データベース、インメモリ分析ソリューション、科学計算、およびその他のメモリを大量に使用するアプリケーションに適しています。 インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。	2014 年 4 月 9 日
新しい Amazon Linux AMI リリース		Amazon Linux AMI 2014.03 をリリースしました。	2014 年 3 月 27 日
Amazon EC2 使用状況レポート		Amazon EC2 使用状況レポートは、EC2 の使用コストと使用状況データを示す一連のレポートです。詳細については、「Amazon EC2 使用状況レポート」を参照してください。	2014 年 1 月 28 日
追加された M3 インスタンス	2013-10-15	M3 インスタンスのサイズとして `m3.medium` および `m3.large` がサポートされるようになりました。インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。	2014 年 1 月 20 日

機能	API バージョン	説明	リリース日
I2 インスタンス	2013-10-15	このインスタンスは、非常に高い IOPS を提供し、連続 SSD 書き込みパフォーマンスを向上させるために Linux インスタンスで TRIM をサポートします。また、I2 インスタンスは、インスタンス間のレイテンシーが低いだけでなく、ネットワークのストレスが少なく、パケット毎秒 (PPS) が非常に大きい拡張ネットワーキングもサポートします。詳細については、「ストレージ最適化インスタンス」を参照してください。	2013 年 12 月 19 日
更新された M3 インスタンス	2013-10-15	M3 インスタンスサイズ (`m3.xlarge` と `m3.2xlarge`) は、SSD ボリュームと合わせてインスタンスストアをサポートするようになりました。	2013 年 12 月 19 日
Linux 仮想マシンのインポート	2013-10-15	VM Import プロセスが、Linux インスタンスのインポートをサポートするようになりました。詳細については、「VM Import/Export ユーザーガイド」を参照してください。	2013 年 12 月 16 日
RunInstances に関するリソースレベルの許可	2013-10-15	AWS Identity and Access Management でポリシーを作成して、Amazon EC2 RunInstances API アクションについてリソースレベルでアクセス許可を制御できるようになりました。詳細とポリシー例については、「Amazon EC2 の Identity and Access Management」を参照してください。	2013 年 11 月 20 日

機能	API バージョン	説明	リリース日
C3 インスタンス	2013-10-15	計算能力が最適化されたインスタンスは、経済的な価格で、非常に高い CPU パフォーマンスを提供します。また、C3 インスタンスは、インスタンス間のレイテンシーが低いだけでなく、ネットワークのストレスが少なくパケット毎秒 (PPS) が非常に大きい拡張ネットワーキングもサポートします。これらのインスタンスは、通信量の多いウェブアプリケーション、広告配信、バッチ処理、動画の暗号化、分散分析、高エネルギー物理学、ゲノム分析、計算流体力学に最適です。 インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。	2013 年 11 月 14 日
AWS Marketplace からのインスタンスの起動		Amazon EC2 Launch Wizard を使用して、AWS Marketplace からインスタンスを起動できるようになりました。詳細については、「AWS Marketplace インスタンスの起動」を参照してください。	2013 年 11 月 11 日
G2 インスタンス	2013-10-01	これらのインスタンスはビデオ作成サービスや 3D 表示、グラフィックスを集中的に使用するアプリケーション、大規模なパラレル処理を必要とするサーバー側のワークロードに最適です。詳細については、「Linux 高速コンピューティングインスタンス」を参照してください。	2013 年 11 月 4 日

機能	API バージョン	説明	リリース日
新しいlaunch wizard		設計し直された新しい EC2 Launch Wizard が付属します。詳細については、「古いインスタンス起動ウィザードを使用してインスタンスを起動する」を参照してください。	2013 年 10 月 10 日
Amazon EC2 リザーブドインスタンスのインスタンスタイプの変更	2013-10-01	同一ファミリー内 (例えば、M1、M2、M3、C1) であれば Linux リザーブドインスタンスのインスタンスタイプを変更できるようになりました。詳細については、「リザーブドインスタンス の変更」を参照してください。	2013 年 10 月 09 日
新しい Amazon Linux AMI リリース		Amazon Linux AMI 2013.09 をリリースしました。	2013 年 9 月 30 日
Amazon EC2 リザーブドインスタンスの変更	2013-08-15	リージョンのリザーブドインスタンスを変更できるようになりました。詳細については、「リザーブドインスタンス の変更」を参照してください。	2013 年 9 月 11 日
パブリック IP アドレスの割り当て	2013-07-15	VPC でインスタンスを起動するときに、パブリック IP アドレスを割り当てることができるようになりました。詳細については、「インスタンス起動時のパブリック IPv4 アドレスの割り当て」を参照してください。	2013 年 8 月 20 日
リソースレベルのアクセス許可の付与	2013-06-15	Amazon EC2 は、新しい Amazon Resource Name (ARN) および条件キーをサポートします。詳細については、「Amazon EC2 の IAM ポリシー」を参照してください。	2013 年 7 月 8 日
インクリメンタルスナップショットコピー	2013-02-01	インクリメンタルスナップショットコピーを実行できるようになりました。	2013 年 6 月 11 日

機能	API バージョン	説明	リリース日
新しい [Tags] ページ		Amazon EC2 コンソールに新しい [Tags] ページを追加しました。詳細については、「Amazon EC2 リソースのタグ付け」を参照してください。	2013 年 04 月 4 日
新しい Amazon Linux AMI リリース		Amazon Linux AMI 2013.03 をリリースしました。	2013 年 3 月 27 日
追加された EBS 最適化インスタンスタイプ	2013-02-01	次のインスタンスタイプが、EBS 最適化インスタンスとして起動できるようになりました: `c1.xlarge` 、`m2.2xlarge` 、`m3.xlarge` 、`m3.2xlarge` 。	2013 年 3 月 19 日
リージョン間での AMI のコピー	2013-02-01	AMI をリージョン間でコピーして、整合性のあるインスタンスを、複数の AWS リージョンですばやく簡単に起動できます。 詳細については、「AMI のコピー」を参照してください。	2013 年 3 月 11 日
デフォルトの VPC へのインスタンスの起動	2013-02-01	AWS アカウントは、リージョンごとに、EC2-Classic と VPC のいずれか、または VPC のみにインスタンスを起動できます。インスタンスを起動できるのが VPC だけである場合は、デフォルトの VPC が自動的に作成されます。お客様がインスタンスを起動するときは、デフォルトの VPC で起動されるようになります。ただし、お客様が非デフォルト VPC を作成してその VPC でインスタンスを起動するよう指定した場合を除きます。	2013 年 3 月 11 日

機能	API バージョン	説明	リリース日
ハイメモリクラスター (cr1.8xlarge) インスタンスタイプ	2012-12-01	大容量のメモリと、高パフォーマンスの CPU およびネットワークの組み合わせです。このインスタンスは、インメモリアナリティクス、グラフ分析、科学計算アプリケーションに適しています。	2013 年 1 月 21 日
ハイストレージ (`hs1.8xlarge`) インスタンスタイプ	2012-12-01	ハイストレージインスタンスは、ストレージ密度がきわめて高く、インスタンスあたりの順次読み込み/書き込みのパフォーマンスの高さが特徴です。データウェアハウス、Hadoop/MapReduce、並列ファイルシステムに適しています。	2012 年 12 月 20 日
EBS スナップショットのコピー	2012-12-01	スナップショットのコピーを使用して、データのバックアップを作成したり、新しい Amazon EBS ボリュームを作成したり、Amazon マシンイメージ (AMI) を作成したりすることができます。	2012 年 12 月 17 日
Provisioned IOPS SSD ボリューム用の EBS メトリクスおよびステータスチェックの更新	2012-10-01	Provisioned IOPS SSD ボリュームの 2 つの新しいメトリクスを含むように EBS メトリクスが更新されました。また、Provisioned IOPS SSD ボリューム用の新しいステータスチェックを追加しました。	2012 年 11 月 20 日
Linux カーネル		AKI ID を更新し、ディストリビューションカーネルを再編成し、PVOps セクションを更新しました。	2012 年 11 月 13 日

機能	API バージョン	説明	リリース日
M3 インスタンス	2012-10-01	新しい M3 エクストララージおよび M3 ダブルエクストララージのインスタンスタイプがあります。インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。	2012 年 10 月 31 日
スポットインスタンスリクエストステータス	2012-10-01	スポットインスタンスリクエストステータスにより、スポットリクエストの状態を簡単に確認できます。	2012 年 10 月 14 日
新しい Amazon Linux AMI リリース		Amazon Linux AMI 2012.09 をリリースしました。	2012 年 10 月 11 日
Amazon EC2 リザーブドインスタンス Marketplace	2012-08-15	リザーブドインスタンス Marketplace は、不要になった Amazon EC2 リザーブドインスタンスを持つ販売者と、追加の容量の購入を検討する購入者をマッチングします。リザーブドインスタンス Marketplace を通じて売買されたリザーブドインスタンスは、他のリザーブドインスタンス同様に動作します。ただし、完全な標準期間より残りの期間が短く、販売価格が異なる可能性がある点が違います。	2012 年 9 月 11 日
Amazon EBS のプロビジョンド IOPS SSD	2012-07-20	Provisioned IOPS SSD ボリュームは、整合性と応答時間の短さが重要な、データベースアプリケーションなど I/O を多用する作業負荷に対して、予測可能なハイパフォーマンスを提供します。	2012 年 7 月 31 日

機能	API バージョン	説明	リリース日
Amazon EC2 向けのハイ I/O インスタンス	2012-06-15	ハイ I/O インスタンスは、SSD ベースのローカルインスタンスストレージを使用して低レイテンシーのハイ I/O パフォーマンスを提供します。	2012 年 7 月 18 日
Amazon EC2 インスタンスの IAM ロール	2012-06-01	Amazon EC2 向けの IAM ロールは次の機能を提供します。 • Amazon EC2 インスタンスで実行中のアプリケーション用の AWS アクセスキー。 • Amazon EC2 インスタンスの AWS アクセスキーの自動ローテーション。 • Amazon EC2 インスタンスで実行中の、AWS サービスにリクエストを送信しているアプリケーションに対するきめの細かい権限管理。	2012 年 6 月 11 日

機能	API バージョン	説明	リリース日
スポットインスタンスの特徴は、開始や中断の可能性への対処を容易にすることです。		次のように、スポットインスタンス を管理できるようになりました。 • Auto Scaling 起動設定を使用して スポットインスタンス に支払う金額を指定し、スポットインスタンス に支払う金額を指定するスケジュールを設定します。詳細については、Amazon EC2 Auto Scaling ユーザーガイドの「Auto Scaling グループの スポットインスタンス の起動」を参照してください。 • インスタンスの起動または終了の通知を受け取ることができます。 • AWS CloudFormation テンプレートを使用して、AWS リソースのスタック内のスポットインスタンスを起動します。	2012 年 6 月 7 日
Amazon EC2 のステータスチェックのための EC2 インスタンスのエクスポートとタイムスタンプ	2012-05-01	ステータスチェックが失敗した日時を示す、インスタンスステータスとシステムステータスのタイムスタンプのサポートを追加しました。	2012 年 5 月 25 日
EC2 インスタンスのエクスポート、および Amazon VPC に対するインスタンスとシステムのステータスチェックのタイムスタンプ	2012-05-01	Citrix Xen、Microsoft Hyper-V、および VMware vSphere 向けの EC2 インスタンスのエクスポートのサポートを追加しました。 インスタンスとシステムのステータスチェックのタイムスタンプのサポートを追加しました。	2012 年 5 月 25 日

機能	API バージョン	説明	リリース日
クラスターコンピュートエイトエクストララージインスタンス	2012-04-01	VPC に `cc2.8xlarge` インスタンスのサポートを追加しました。	2012 年 4 月 26 日
AWS Marketplace AMI	2012-04-01	AWS Marketplace AMI のサポートが追加されました。	2012 年 4 月 19 日
新しい Linux AMI リリース		Amazon Linux AMI 2012.03 をリリースしました。	2012 年 3 月 28 日
新しい AKI バージョン		AKI バージョン 1.03 および AWS GovCloud (US) リージョン用の AKI をリリースしました。	2012 年 3 月 28 日
ミディアムインスタンス、すべての AMI での 64 ビットのサポート、および Java ベースの SSH クライアント	2011-12-15	新しいインスタンスタイプと 64 ビット情報のサポートを追加しました。Java ベースの SSH クライアントを使用して Linux インスタンスに接続する手順を追加しました。	2012年3月7日
リザーブドインスタンス料金範囲	2011-12-15	リザーブドインスタンス料金範囲に組み込まれた割引料金の利用方法に関する新しいセクションを追加しました。	2012年3月5日
Amazon Virtual Private Cloud での EC2 インスタンスのための Elastic Network Interfaces (ENI)	2011-12-01	VPC での EC2 インスタンスのための Elastic Network Interfaces (ENI) についての新しいセクションを追加しました。詳細については、「Elastic Network Interface」を参照してください。	2011年12月21日

機能	API バージョン	説明	リリース日
新しい GRU リージョンと AKI		SA-East-1 リージョン用の新しい AKI リリースについての情報を追加しました。このリリースで、AKI バージョン 1.01 は廃止されました。AKI バージョン 1.02 には下位互換性があります。	2011 年 12 月 14 日
新しい Amazon EC2 リザーブドインスタンスの提供タイプ	2011-11-01	インスタンスの使用目的に応じて、さまざまなリザーブドインスタンス提供タイプを選択できるようになりました。	2011年12月1日
Amazon EC2 インスタンスのステータス	2011-11-01	AWS で予定されている、インスタンスに影響を及ぼす可能性のあるイベントなど、インスタンスのステータスについて追加の詳細情報を表示できます。これら運用上の作業には、ソフトウェアアップデートやセキュリティパッチを適用するために必要なインスタンスの再起動や、ハードウェアに問題が生じた場合に必要となるインスタンスの廃棄などが含まれます。詳細については、「インスタンスのステータスのモニタリング」を参照してください。	2011年11月16日
Amazon EC2 クラスタコンピュートインスタンスタイプ		クラスターコンピュートエイトエクストララージ (cc2.8xlarge) のサポートを Amazon EC2 に追加しました。	2011年11月14日
新しい PDX リージョンと AKI		新しい US-West 2 リージョン用の新しい AKI のリリースについての情報を追加しました。	2011年11月8日
Amazon VPC の スポットインスタンス	2011-07-15	Amazon VPC での スポットインスタンス のサポートについての情報を追加しました。この更新により、ユーザーは Virtual Private Cloud (VPC) で スポットインスタンス を起動できます。スポットインスタンス のユーザーは スポットインスタンス を起動することで、Amazon VPC の利点を得ることができます。	2011年10月11日

機能	API バージョン	説明	リリース日
新しい Linux AMI リリース		Amazon Linux AMI 2011.09 のリリースについての情報を追加しました。このアップデートにより、Amazon Linux AMI からベータのタグが削除され、リポジトリを特定のバージョンにロックする機能がサポートされ、インストールされているパッケージについてセキュリティアップデートを含むアップデートが利用可能になったときに通知が表示されます。	2011年9月26日
CLI ツールのユーザーのための VM Import 処理の簡素化	2011-07-15	VM Import 処理を簡素化し、`ImportInstance` と `ImportVolume` の機能を拡張しました。これにより、インポートタスクの作成後に、イメージが Amazon EC2 にアップロードされます。さらに、`ResumeImport` の導入により、アップロードが途中で止まった場合にその個所から再開できるようになりました。	2011年9月15日
VHD ファイル形式でのインポートのサポート		VM Import で、仮想マシンイメージファイルを VHD 形式でインポートできるようになりました。VHD ファイル形式は、Citrix Xen および Microsoft Hyper-V 仮想化プラットフォームと互換性があります。VM Import はこのリリースで、RAW、VHD、および VMDK (VMware ESX 互換) イメージ形式をサポートしています。詳細については、「VM Import/Export ユーザーガイド」を参照してください。	2011 年 8 月 24 日
VMware vCenter 用の Amazon EC2 VM Import Connector のアップデート		Amazon EC2 VM Import Connector for VMware vCenter 仮想アプライアンス (Connector) の1.1バージョンについての情報を追加しました。このアップデートには、インターネットアクセスのためのプロキシサポート、エラー処理の向上、タスクプログレスバーの精度向上、および数件のバグ修正が含まれます。	2011年6月27日

機能	API バージョン	説明	リリース日
ユーザー提供のカーネルを実行するための Linux AMI の有効化		AKI のバージョン 1.01 から 1.02 への変更についての情報を追加しました。このバージョンでは、t 1.micro Linux インスタンスに関連する起動エラーに対処するため、PVGRUB が更新されています。詳細については、「ユーザー提供カーネル」を参照してください。	2011年6月20日
スポットインスタンス のアベイラビリティーゾーンの料金変更	2011-05-15	スポットインスタンス のアベイラビリティーゾーンの料金機能についての情報を追加しました。このリリースでは、スポットインスタンスリクエストおよびスポット料金履歴のクエリを実行したときに返される情報の一部として、新しいアベイラビリティーゾーンの料金オプションが追加されました。これにより、特定のアベイラビリティーゾーンでスポットインスタンスを起動するために必要な料金を判断しやすくなりました。	2011年5月26日
AWS Identity and Access Management		AWS Identity and Access Management (IAM) に関する情報を追加しました。ユーザーが Amazon EC2 リソースで通常使用できる Amazon EC2 アクションを、指定できるようにします。詳細については、「Amazon EC2 の Identity and Access Management」を参照してください。	2011 年 4 月 26 日
ユーザー提供のカーネルを実行するための Linux AMI の有効化		ユーザー提供のカーネルの実行用に PVGRUB Amazon Kernel Image (AKI) を使用するために Linux AMI を有効化する方法についての情報を追加しました。詳細については、「ユーザー提供カーネル」を参照してください。	2011 年 4 月 26 日

機能	API バージョン	説明	リリース日
専用インスタンス		専用インスタンスは、ホストのハードウェアレベルで物理的に隔離されているインスタンスであり、Amazon Virtual Private Cloud (Amazon VPC) 内で起動します。ハードウェア専有インスタンスを使うと、Amazon VPC と AWS クラウドの利点を活かすことができます。このインスタンスでは、Amazon EC2 のコンピューティングインスタンスをハードウェアレベルで隔離しながら、伸縮自在なオンデマンドでのプロビジョニングや、従量課金の料金システムをご利用いただけます。詳細については、「Dedicated Instances」を参照してください。	2011 年 3 月 27 日
リザーブドインスタンスに関する AWS マネジメントコンソールの更新		AWS マネジメントコンソールが更新されました。これにより、リザーブドインスタンスの表示や、ハードウェア専有リザーブドインスタンスを含む追加のリザーブドインスタンスの購入が、より簡単に行えるようになりました。詳細については、「Reserved Instances」を参照してください。	2011 年 3 月 27 日
新しい Amazon Linux 参照 AMI		新しい Amazon Linux 参照 AMI は CentOS 参照 AMI を置き換えます。Correcting Clock Drift for Cluster Instances on CentOS 5.4 AMI セクションなど、CentOS 参照 AMI に関する情報を削除しました。	2011年3月15日
メタデータ情報	2011-01-01	2011-01-01 リリースでの変更を反映するため、メタデータについての情報を追加しました。詳細については、「インスタンスメタデータとユーザーデータ」および「インスタンスメタデータのカテゴリ」を参照してください。	2011 年 3 月 11 日

機能	API バージョン	説明	リリース日
Amazon EC2 VM Import Connector for VMware vCenter		Amazon EC2 VM Import Connector for VMware vCenter 仮想アプライアンス (Connector) についての情報を追加しました。このコネクタは、VMware vSphere Client と統合するための、VMware vCenter のプラグインです。コネクタの GUI を使用して、VMware 仮想マシンを Amazon EC2 にインポートすることができます。	2011年3月3日
ボリュームの強制デタッチ		AWS Management Console を使用して、インスタンスから Amazon EBS ボリュームを強制的にデタッチできるようになりました。	2011 年 2 月 23 日
インスタンス終了の防止		AWS マネジメントコンソール を使用して、インスタンスの削除を防止できるようになりました。詳細については、「終了保護を有効化する」を参照してください。	2011 年 2 月 23 日
CentOS 5.4 AMI のクラスタインスタンスのクロック同期ずれの修正		Amazon の CentOS 5.4 AMI 上で実行されているクラスタインスタンスのクロック同期ずれを修正する方法についての情報を追加しました。	2011 年 1 月 25 日
VM Import	2010-11-15	仮想マシンまたはボリュームを Amazon EC2 にインポートする VM Import についての情報を追加しました。詳細については、「VM Import/Export ユーザーガイド」を参照してください。	2010 年 12 月 15 日
インスタンスの基本モニタリング	2010-08-31	EC2 インスタンスの基本モニタリングについての情報を追加しました。	2010 年 12 月 12 日

機能	API バージョン	説明	リリース日
フィルタとタグ	2010-08-31	リソースの一覧表示、フィルタリング、およびタグ付けについての情報を追加しました。詳細については、「リソースの一覧表示およびフィルタリング」および「Amazon EC2 リソースのタグ付け」を参照してください。	2010年9月19日
インスタンス起動時の多重実行禁止	2010-08-31	インスタンスを実行する際に多重実行を禁止する方法についての情報を追加しました。詳細については、Amazon EC2 API Reference の「べき等性の確保」を参照してください。	2010年9月19日
マイクロインスタンス	2010-06-15	Amazon EC2 は、特定のタイプのアプリケーションに対して、`t1.micro`インスタンスタイプを提供します。詳細については、「バーストパフォーマンスインスタンス」を参照してください。	2010年9月8日
Amazon EC2 用の AWS Identity and Access Management		Amazon EC2 が AWS Identity and Access Management (IAM) と統合されるようになりました。詳細については、「Amazon EC2 の Identity and Access Management」を参照してください。	2010 年 9 月 2 日
クラスターインスタンス	2010-06-15	Amazon EC2 は、ハイパフォーマンスコンピューティング (HPC) アプリケーション用にクラスターコンピューティングインスタンスを提供します。インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。	2010年7月12日
Amazon VPC IP アドレス指定	2010-06-15	Amazon VPC ユーザーは、VPC 内で起動されたインスタンスに割り当てる IP アドレスを指定できるようになりました。	2010年7月12日

機能	API バージョン	説明	リリース日
Amazon CloudWatch Amazon EBS ボリュームのモニタリング		Amazon EBS ボリュームに対する Amazon CloudWatch モニタリングが自動的に利用できるようになりました。	2010 年 6 月 14 日
ハイメモリエクストララージインスタンス	2009-11-30	Amazon EC2 は、ハイメモリエクストララージ (m2.xlarge) インスタンスタイプをサポートするようになりました。インスタンスタイプの詳細な仕様については、「Amazon EC2 Instance Types Guide」を参照してください。料金の詳細については、Amazon EC2 のインスタンスタイプのページを参照してください。	2010 年 2 月 22 日