Application Auto Scaling

ユーザーガイド

Application Auto Scaling: ユーザーガイド

Table of Contents

Application Auto Scaling とは

Application Auto Scaling は、Amazon EC2 以外の個々の AWS のサービスのスケーラブルリソースを自動的にスケールするソリューションを必要とするデベロッパーとシステム管理者のためのウェブサービスです。Application Auto Scaling では、以下のリソースに対してオートスケーリングを設定できます。

- AppStream 2.0 フリート
- Aurora レプリカ
- Amazon Comprehend ドキュメントの分類とエンティティ認識のエンドポイント
- DynamoDB テーブルとグローバルセカンダリインデックス
- Amazon Elastic Container Service (ECS) サービス
- ElastiCache for Redis クラスター (レプリケーショングループ)
- Amazon EMR クラスター
- Amazon Keyspaces (Apache Cassandra 用) テーブル
- Lambda 関数のプロビジョニングされた同時実行数
- Amazon Managed Streaming for Apache Kafka (MSK) ブローカーストレージ
- Amazon Neptune クラスター
- SageMaker エンドポイントバリアント
- スポットフリートリクエスト
- 独自のアプリケーションまたはサービスにより提供されるカスタムリソース。詳細については、GitHub リポジトリを参照してください。

上記の AWS のサービスを利用できるリージョンを確認するには、「リージョン表」を参照してください。

Auto Scaling グループを使用した Amazon EC2 インスタンスフリートのスケーリングについては、Amazon EC2 Auto Scaling ユーザーガイドを参照してください。

AWS Auto Scaling を使用して、複数のサービス全体でリソースをスケールするスケーリングプランを作成することも可能です。詳細については、AWS Auto Scaling ユーザーガイドを参照してください。

Application Auto Scaling の特徴

Application Auto Scaling では、ユーザー定義の条件に従ってスケーラブルリソースを自動的にスケールすることができます。

- ターゲット追跡スケーリング – 特定の CloudWatch メトリクスのターゲット値に基づいてリソースをスケールします。
- ステップスケーリング – 超過アラームのサイズによって異なる一連のスケーリング調整値に基づいてリソースをスケーリングします。
- スケジュールに基づくスケーリング – 1 回のみ、または定期的なスケジュールでリソースをスケールします。

Application Auto Scaling と連携

スケーリングするリソースに応じて、次のインターフェイスを使用してスケーリングを設定できます。

- AWS Management Console – スケーリングを設定する際に使用するウェブインターフェイスを提供します。AWS アカウントへのサインアップが済んでいる場合は、AWS Management Consoleにサインインすることによって Application Auto Scaling にアクセスします。次に、概要に一覧表示されているリソースの 1 つのサービスコンソールを開きます。作業するリソースと同じ AWS リージョン でコンソールを開いていることを確認します。

  Note

  リソースにはコンソールアクセスを利用できないものもあります。詳細については、「Application Auto Scaling を使用できる AWS のサービス (p. 7)」を参照してください。
- AWS Command Line Interface (AWS CLI) – さまざまな AWS のサービス のコマンドを提供し、Windows、macOS、Linux でサポートされています。開始するには、「AWS Command Line Interface ユーザーガイド」を参照してください。詳細については、AWS CLI コマンドリファレンスの「application-autoscaling」を参照してください。
- AWS Tools for Windows PowerShell – PowerShell 環境でスクリプトを作成するユーザー向けに、さまざまな AWS 製品用のコマンドが用意されています。使用を開始するには、「 AWS Tools for Windows PowerShellユーザーガイド」を参照してください。詳細については、「AWS Tools for PowerShell Cmdlet Reference」を参照してください。
- AWS SDK – 言語固有の API オペレーションを提供し、署名の計算、リクエストの再試行処理、エラー処理など、接続のさまざまな詳細に対処します。詳細については、AWSSDK をご参照ください。
- クエリ API – HTTPS リクエストを使用して呼び出す低レベル API アクションを提供します。クエリ API の使用は、AWS のサービス にアクセスする最も直接的な方法です。ただし、リクエストに署名するハッシュの生成やエラー処理など、アプリケーションが低レベルな作業を処理することを要求します。詳細については、Application Auto Scaling API リファレンス を参照してください。
- AWS CloudFormation – CloudFormation テンプレートを使用したスケーリングプランの設定をサポートします。詳細については、「AWS CloudFormation を使用した Application Auto Scaling リソースの作成 (p. 114)」を参照してください。

AWS のサービス にプログラムで接続するため、エンドポイントを使用します。Application Auto Scaling へのコールのエンドポイントについては、AWS 全般リファレンスの「Application Auto Scaling エンドポイントとクォータ」、AWS シークレットリージョンユーザーガイドの、、。

設定する

Application Auto Scaling を使用してオートスケーリングを設定する前に、AWS アカウントを作成し、アクセス許可を設定して、AWS Command Line Interface (AWS CLI) をセットアップします。

トピック

AWS アカウントにサインアップする

Amazon Web Service でアカウントにサインアップすると、Application Auto Scaling を含めたすべての AWS サービスにアカウントが自動的にサインアップされます。料金は、使用するサービスの料金のみが請求されます。

AWS アカウントをお持ちでない場合は、作成する必要があります。AWS アカウント をすでにお持ちの場合は、以下の手順の AWS アカウント 作成ステップをスキップすることができます。

AWS アカウント にサインアップするには

1. https://aws.amazon.com/ を開いて、[Sign Up] (サインアップ) をクリックします。
2. オンラインの指示に従ってください。サインアップ手順の一環として、通話呼び出しを受け取り、電話のキーパッドを用いて確認コードを入力することが求められます。サインアッププロセスが完了すると、AWS より確認メールが送信されます。

AWS リージョンでの Application Auto Scaling の使用

Application Auto Scaling は、複数の AWS リージョンで利用できます。グローバル AWS アカウントでは、ほとんどのリージョン内にあるリソースを使用できます。中国リージョン内のリソースで Application Auto Scaling を使用するときは、個別の Amazon Web Services (中国) アカウントが必要になることに留意してください。また、Application Auto Scaling の実装方法にもいくつかの違いがあります。中国リージョンでの Application Auto Scaling の使用に関する詳細ついては、「Application Auto Scaling in China」を参照してください。

AWS CLI をセットアップする

AWS Command Line Interface (AWS CLI) は、Application Auto Scaling を含めた AWS のサービスを管理するための統合デベロッパーツールです。これらのステップに従って、AWS CLI をダウンロードして設定します。

AWS CLI をセットアップする

1. AWS CLI をダウンロードして設定します。手順については、「AWS Command Line Interface ユーザーガイド」の次のトピックを参照してください。
   - AWS CLI の最新バージョンのインストールまたは更新
   - 高速セットアップ

Note

AWS アカウントが作成(これをAWS アカウントのルートユーザーと読んでいます) された際に提供された E メールアドレスとパスワードに関連付けられた認証情報を使用する AWS CLI を設定できますが、これは AWS のセキュリティの面でベストプラクティスではありません。代わりに、AWS アカウントで IAM 管理者ユーザーの認証情報を使用する AWS CLI の設定をお勧めします。IAM 管理者ユーザーは、AWS アカウントのルートユーザーと同様の AWS アクセス許可を持ち、関連するセキュリティリスクの一部を回避します。IAM 管理者ユーザーとして AWS CLI を設定できない場合は、AWS アカウントの管理者にチェックしてください。詳細については、「IAM ユーザーガイド」の「最初の IAM 管理者のユーザーおよびグループの作成」を参照してください。

2. AWS CLI プロファイルが適切に設定されていることを確認するには、コマンドウィンドウで次のコマンドを実行します。

```
aws configure
```

プロファイルが正しく設定されている場合は、以下のような出力が表示されます。

```
AWS Access Key ID [****************52FQ]:
AWS Secret Access Key [****************xgyZ]:
Default region name [us-east-1]:
Default output format [json]:
```

3. 以下のコマンドを実行して、AWS CLI 用の Application Auto Scaling コマンドがインストールされていることを確認します。

```
aws application-autoscaling help
```

Application Auto Scaling の使用開始方法

このトピックでは、Application Auto Scaling について学習し、使用を開始するために役立つ主な概念について説明します。

スケーラブルターゲット

スケールするリソースを指定するために作成するエンティティです。各スケーラブルターゲットは、サービス名前空間、リソース ID、およびスケーラブルディメンションによって一意に識別されます。これは、基盤となるサービスの容量ディメンションを表します。例えば、Amazon ECS サービスはそのタスク数のオートスケーリングをサポートし、DynamoDB テーブルはテーブルとそのグローバルセカンダリインデックスの読み込みキャパシティーと書き込みキャパシティーのオートスケーリングをサポートし、Aurora クラスターはそのレプリカ数のスケーリングをサポートします。

Tip

各スケーラブルターゲットには、最小容量と最大容量もあります。スケーリングポリシーが、最小容量から最大容量までの範囲を超える、または下回ることはありません。Application Auto Scaling が認識しない、この範囲外の帯域外変更を基盤となるリソースに直接行うことができます。ただし、スケーリングポリシー、または `RegisterScalableTarget` API が呼び出されるときは常に、Application Auto Scaling が現在の容量を取得して、それを最小容量および最大容量と比較します。それが最小容量から最大容量までの範囲内に当てはまらない場合、設定された最小容量と最大容量に適合するように容量が更新されます。

スケールイン

Application Auto Scaling がスケーラブルターゲットの容量を自動的に減少させると、スケーラブルターゲットがスケールインします。スケーラブルターゲットの最小容量は、スケーリングポリシーがスケールインできる最小容量です。

スケールアウト

Application Auto Scaling がスケーラブルターゲットの容量を自動的に増加させると、スケーラブルターゲットがスケールアウトします。スケーラブルターゲットの最大容量は、スケーリングポリシーがスケールアウトできる最大容量です。

スケーリングポリシー

スケーリングポリシーは、Application Auto Scaling に対して、特定の CloudWatch メトリクスを追跡するように指示します。その後、メトリクスが特定のしきい値よりも高い、または低いときに実行するスケーリングアクションを決定します。例えば、クラスター全体の CPU 使用率が上昇し始めた場合はスケールアウトし、再び低下した場合はスケールインすることができます。

オートスケーリングに使用されるメトリクスはターゲットサービスによって発行されますが、独自のメトリクスを CloudWatch に発行して、それをスケーリングポリシーで使用することもできます。

スケーリングアクティビティ間のクールダウン期間は、別のスケーリングアクティビティが開始される前にリソースを安定させます。Application Auto Scaling は、クールダウン期間中も引き続きメトリクスを評価します。クールダウン期間が終了すると、スケーリングポリシーが、必要に応じて別のスケーリングアクティビティを開始します。クールダウン期間の実施中、現行のメトリクス値に基づいてより大きなスケールアウトが必要になった場合は、スケーリングポリシーが直ちにスケールアウトします。

スケジュールされたアクション

スケジュールされたアクションは、特定の日付けと時刻にリソースを自動的にスケールします。これらは、スケーラブルターゲットの最小容量と最大容量を変更することによって機能するため、最小容量を高く、または最大容量を低く設定することで、スケジュールに従ってスケールインおよびスケールアウトするために使用できます。例えば、スケジュールされたアクションを使用して、金曜日の容量を減らし、翌週月曜日の容量を増やすことによって、週末にリソースを消費しないアプリケーションをスケールすることができます。

また、最小値と最大値を経時的に最適化するスケジュールされたアクションを使用して、マーケティングキャンペーンや季節的な変動など、通常よりも多いトラフィックが予想される状況に適応することも可能です。そうすることにより、使用量の増加に合わせてスケールアウトする必要があるときにはパフォーマンスを向上させ、使用するリソースが少ないときにはコストを削減することができます。

詳細情報

Application Auto Scaling を使用できる AWS のサービス (p. 7) – このセクションは、スケール可能なサービスについて紹介し、スケーラブルターゲットを登録することによるオートスケーリングのセットアップに役立ちます。また、ターゲットサービス内のリソースにアクセスするために Application Auto Scaling が作成する、各 IAM サービスリンクロールについても説明します。

Application Auto Scaling のターゲット追跡スケーリングポリシー (p. 46) – Application Auto Scaling の主な機能の 1 つは、ターゲット追跡スケーリングポリシーです。設定されたメトリクスと目標値に基づいて使用量を一定のレベルに保つために、ターゲット追跡ポリシーが望ましい容量を自動的に調整する方法について学びます。例えば、スポットフリートの平均 CPU 使用率を 50% に維持するようにターゲット追跡を設定できます。これが設定されると、Application Auto Scaling は、すべてのサーバー全体で集約された CPU 使用率を 50% に維持するために、必要に応じて EC2 インスタンスを起動または終了します。

Application Auto Scaling を使用できる AWS のサービス

Application Auto Scaling は AWS の他のサービスと統合するため、スケーリング機能を追加してアプリケーションの需要を満たすことができます。オートスケーリングは、ほとんどすべての場合にデフォルトで無効になっているサービスのオプション機能です。

以下の表には、Application Auto Scaling を使用できる AWS のサービスがリストされており、オートスケーリングの設定にサポートされている手法に関する情報が含まれています。Application Auto Scaling は、カスタムリソースで使用することも可能です。

コンソールアクセス – AWS の互換性があるサービスのコンソールでスケーリングポリシーを設定することによってターゲットサービスを設定し、オートスケーリングを開始できます。現在、スケジュールされたスケーリングに対するコンソールサポートを提供しているのは Amazon AppStream 2.0、ElastiCache とスポットフリートのみです。サービスがコンソールアクセスをサポートする場合は、詳細はこちらのリンク先を参照して、そのサービスのコンソールからオートスケーリングを設定する方法を学んでください。

CLI アクセス – AWS CLI を使用して AWS の互換性があるサービスを設定し、オートスケーリングを開始できます。

SDK アクセス – AWS SDK を使用して AWS の互換性があるサービスを設定し、オートスケーリングを開始できます。

CloudFormation アクセス – AWS CloudFormation スタックテンプレートを使用して AWS の互換性があるサービスを設定し、オートスケーリングを開始できます。詳細については、「AWS CloudFormation を使用した Application Auto Scaling リソースの作成 (p. 114)」を参照してください。

AWS のサービス	コンソールアクセス	CLI アクセス	SDK アクセス	CloudFormation アクセス
AppStream 2.0 (p. 8)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Aurora (p. 9)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Amazon Comprehend (p. 11)	✗いいえ	✓はい	✓はい	✓はい
Amazon DynamoDB (p. 12)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Amazon ECS (p. 14)	✓はい 詳細はこちら	✓はい	✓はい	✓はい

AWS のサービス	コンソールアクセス	CLI アクセス	SDK アクセス	CloudFormation アクセス
Amazon ElastiCache (p. 15)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Amazon EMR	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Amazon Keyspaces (p. 17)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Lambda (p. 18)	✗いいえ	✓はい	✓はい	✓はい
Amazon MSK (p. 20)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
Amazon Neptune (p. 21)	✗いいえ	✓はい	✓はい	✓はい
SageMaker (p. 22)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
スポットフリート (p. 23)	✓はい 詳細はこちら	✓はい	✓はい	✓はい
カスタムリソース (p. 25)	✗いいえ	✓はい	✓はい	✓はい

Amazon AppStream 2.0 と Application Auto Scaling

AppStream 2.0 フリートは、ターゲット追跡スケーリングポリシー、ステップスケーリングポリシー、およびスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、AppStream 2.0 の Application Auto Scaling との統合に役立ててください。

AppStream 2.0 フリートのスケーリングを始めたばかりの場合は、以下のドキュメントで、AppStream 2.0 での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon AppStream 2.0 管理ガイドの「AppStream 2.0 向け Fleet Auto Scaling」

AppStream 2.0 用に作成されたサービスリンクロール

AppStream 2.0 リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `appstream.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての AppStream 2.0 フリートの Application Auto Scaling への登録

Application Auto Scaling では、AppStream 2.0 フリートのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

AppStream 2.0 コンソールを使用してオートスケーリングを設定すると、AppStream 2.0 がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  AppStream 2.0 フリートの登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 1 個のフリートインスタンス、および最大容量を 5 個のフリートインスタンスとして、`sample-fleet` という名前のフリートの希望容量を登録します。

```
aws application-autoscaling register-scalable-target \
   --service-namespace appstream \
   --scalable-dimension appstream:fleet:DesiredCapacity \
   --resource-id fleet/sample-fleet \
   --min-capacity 1 \
   --max-capacity 5
```

- AWS SDK:

  RegisterScalableTarget オペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

Amazon Aurora と Application Auto Scaling

Aurora DB クラスターは、ターゲット追跡スケーリングポリシー、ステップスケーリングポリシー、およびスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、Aurora の Application Auto Scaling との統合に役立ててください。

Aurora DB クラスターのスケーリングを始めたばかりの場合は、以下のドキュメントで、Aurora での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon RDS ユーザーガイドの「Aurora レプリカでの Amazon Aurora Auto Scaling の使用」

Aurora 用に作成されたサービスリンクロール

Aurora リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_RDSCluster`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `rds.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての Aurora DB クラスターの Application Auto Scaling への登録

Application Auto Scaling では、Aurora DB クラスターのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

Aurora コンソールを使用してオートスケーリングを設定すると、Aurora がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  Aurora クラスターの登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 1 個の Aurora レプリカ、および最大容量を 8 個の Aurora レプリカとして、`my-db-cluster` という名前のクラスター内の Aurora レプリカの数を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount \
  --resource-id cluster:my-db-cluster \
  --min-capacity 1 \
  --max-capacity 8
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

Amazon Comprehend と Application Auto Scaling

Amazon Comprehend のドキュメント分類とエンティティ認識器の各エンドポイントは、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、Amazon Comprehend の Application Auto Scaling との統合に役立ててください。

Amazon Comprehend のドキュメント分類とエンティティ認識器のエンドポイントのスケーリングを始めたばかりの場合は、以下のドキュメントで、Amazon Comprehend での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon Comprehend デベロッパーガイドの「Auto scaling with endpoints」

Amazon Comprehend 用に作成されたサービスリンクロール

Amazon Comprehend リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `comprehend.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての Amazon Comprehend リソースの Application Auto Scaling への登録

Application Auto Scaling では、Amazon Comprehend のドキュメント分類とエンティティ認識器の各エンドポイントのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  ドキュメント分類エンドポイントに対して 登録登録-スケーラブル-ターゲット コマンドを呼び出します。以下の例は、最小容量を 1 個の推論単位、および最大容量を 3 個の推論単位とし、ドキュメント分

類器エンドポイントの ARN を使用してそのエンドポイントのモデルによって使用される推論単位の希望数を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-
endpoint/EXAMPLE \
  --min-capacity 1 \
  --max-capacity 3
```

エンティティ認識器エンドポイントに対して `register-scalable-target` コマンドを呼び出します。以下の例は、最小容量を 1 個の推論単位、および最大容量を 3 個の推論単位とし、エンティティ認識器エンドポイントの ARN を使用してそのエンドポイントのモデルによって使用される推論単位の希望数を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace comprehend \
  --scalable-dimension comprehend:entity-recognizer-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:entity-recognizer-
endpoint/EXAMPLE \
  --min-capacity 1 \
  --max-capacity 3
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

Amazon DynamoDB と Application Auto Scaling

DynamoDB のテーブルとグローバルセカンダリインデックスは、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、DynamoDB の Application Auto Scaling との統合に役立ててください。

DynamoDB のテーブルとグローバルセカンダリインデックスのスケーリングを始めたばかりの場合は、以下のドキュメントで、DynamoDB での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon DynamoDB デベロッパーガイドの「DynamoDB Auto Scaling によるスループット容量の管理」

> **Tip**
>
> 「チュートリアル:AWS CLI を使用したスケジュールに基づくスケーリングの開始方法 (p. 38)」には、スケジュールされたスケーリングのためのチュートリアルも記載されています。このチュートリアルでは、DynamoDB テーブルがスケジュールされた時刻にスケールされるようにスケーリングを設定するための基本的な手順について学びます。

DynamoDB 用に作成されたサービスリンクロール

DynamoDB リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_DynamoDBTable`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `dynamodb.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての DynamoDB リソースの Application Auto Scaling への登録

Application Auto Scaling では、DynamoDB のテーブルとグローバルセカンダリインデックスのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

DynamoDB コンソールを使用してオートスケーリングを設定すると、DynamoDB がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定する場合は、以下のオプションを使用できます。

- AWS CLI:

  テーブルの書き込み容量に対して register-scalable-target コマンドを呼び出します。以下の例は、最小容量を 5 個の書き込みキャパシティーユニット、最大容量を 10 個の書き込みキャパシティーユニットとして、`my-table` と呼ばれるテーブルのプロビジョニングされた書き込みキャパシティーを登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/my-table \
  --min-capacity 5 \
  --max-capacity 10
```

  テーブルの読み取り容量に対して register-scalable-target コマンドを呼び出します。以下の例は、最小容量を 5 個の読み取り容量ユニット、最大容量を 10 個の読み取り容量ユニットとして、`my-table` と呼ばれるテーブルのプロビジョニングされた読み取り容量を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits \
  --resource-id table/my-table \
  --min-capacity 5 \
  --max-capacity 10
```

  グローバルセカンダリインデックスの書き込み容量に対して register-scalable-target コマンドを呼び出します。以下の例は、最小容量を 5 個の書き込みキャパシティーユニット、最大容量を 10 個の書き込

みキャパシティーユニットとして、my-table-index と呼ばれるグローバルセカンダリインデックスのプロビジョニングされた書き込みキャパシティーを登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:WriteCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

グローバルセカンダリインデックスの読み取り容量に対して register-scalable-target コマンドを呼び出します。以下の例は、最小容量を 5 個の読み取り容量ユニット、最大容量を 10 個の読み取り容量ユニットとして、my-table-index と呼ばれるグローバルセカンダリインデックスのプロビジョニングされた読み取り容量を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:index:ReadCapacityUnits \
  --resource-id table/my-table/index/my-table-index \
  --min-capacity 5 \
  --max-capacity 10
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、ResourceId、ScalableDimension、ServiceNamespace、MinCapacity、および MaxCapacity をパラメータとして指定します。

Amazon ECS と Application Auto Scaling

ECS サービスは、ターゲット追跡スケーリングポリシー、ステップスケーリングポリシー、およびスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、Amazon ECS の Application Auto Scaling との統合に役立ててください。

ECS サービスのスケーリングをスケーリングを始めたばかりの場合は、以下のドキュメントで、Amazon ECS での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon Elastic Container Service デベロッパーガイドの「サービスのオートスケーリング」

Amazon ECS 用に作成されたサービスリンクロール

Amazon ECS リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- AWSServiceRoleForApplicationAutoScaling_ECSService

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `ecs.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての ECS サービスの Application Auto Scaling への登録

Application Auto Scaling では、Amazon ECS サービスのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

Amazon ECS コンソールを使用してオートスケーリングを設定すると、Amazon ECS がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  Amazon ECS サービス用の登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小タスク数を 1 個のタスク、最大タスク数を 10 個のタスクとして、`default` クラスターで実行される `sample-app-service` と呼ばれるサービスのスケーラブルターゲットを登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --min-capacity 1 \
  --max-capacity 10
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` および `MaxCapacity` をパラメータとして指定します。

ElastiCache for Redis と Application Auto Scaling

ElastiCache for Redis のレプリケーショングループは、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、ElastiCache の Application Auto Scaling との統合に役立ててください。

ElastiCache for Redis レプリケーショングループのスケーリングをスケーリングを始めたばかりの場合は、以下のドキュメントで、ElastiCache での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon ElastiCache for Redis ユーザーガイドの「Auto Scaling ElastiCache for Redis clusters」

ElastiCache 用に作成されたサービスリンクロール

ElastiCache リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `elasticache.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての ElastiCache for Redis レプリケーショングループの Application Auto Scaling への登録

Application Auto Scaling では、ElastiCache レプリケーショングループのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

ElastiCache コンソールを使用してオートスケーリングを設定すると、ElastiCache がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  ElastiCache レプリケーショングループに対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 1、最大容量を 5 として、`mycluster` という名前のレプリケーショングループの希望ノードグループ数を登録します。

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:NodeGroups \
   --resource-id replication-group/mycluster \
   --min-capacity 1 \
   --max-capacity 5
```

  以下の例は、最小容量を 1、最大容量を 5 として、`mycluster` という名前のレプリケーショングループのノードグループあたりの希望レプリカ数を登録します。

```
aws application-autoscaling register-scalable-target \
   --service-namespace elasticache \
   --scalable-dimension elasticache:replication-group:Replicas \
   --resource-id replication-group/mycluster \
   --min-capacity 1 \
   --max-capacity 5
```

- AWS SDK:

  RegisterScalableTarget オペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity` および `MaxCapacity` をパラメータとして指定します。

Amazon Keyspaces (Apache Cassandra 向け) と Application Auto Scaling

Amazon Keyspaces のテーブルは、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、Amazon Keyspaces の Application Auto Scaling との統合に役立ててください。

Amazon Keyspaces テーブルのスケーリングをスケーリングを始めたばかりの場合は、以下のドキュメントで、Amazon Keyspaces での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon Keyspaces (Apache Cassandra 向け) デベロッパーガイドの「Managing Amazon Keyspaces throughput capacity with Application Auto Scaling」

Amazon Keyspaces 用に作成されたサービスリンクロール

Amazon Keyspaces リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_CassandraTable`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `cassandra.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての Amazon Keyspaces テーブルの Application Auto Scaling への登録

Application Auto Scaling では、Amazon Keyspaces テーブルのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

Amazon Keyspaces コンソールを使用してオートスケーリングを設定すると、Amazon Keyspaces がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS CLI または AWS SDK の 1 つを使用してオートスケーリングを設定する場合は、次のオプションを使用できます。

- AWS CLI:

  Amazon Keyspaces テーブルに対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 5 個の書き込みキャパシティーユニット、最大容量を 10 個の書き込みキャパシティーユニットとして、mytable と呼ばれるテーブルのプロビジョニングされた書き込みキャパシティーを登録します。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:WriteCapacityUnits \
    --resource-id keyspace/mykeyspace/table/mytable \
    --min-capacity 5 \
    --max-capacity 10
  ```

  以下の例は、最小容量を 5 個の読み取りキャパシティーユニット、最大容量を 10 個の読み取りキャパシティーユニットとして、mytable と呼ばれるテーブルのプロビジョニングされた読み取りキャパシティーを登録します。

  ```
  aws application-autoscaling register-scalable-target \
    --service-namespace cassandra \
    --scalable-dimension cassandra:table:ReadCapacityUnits \
    --resource-id keyspace/mykeyspace/table/mytable \
    --min-capacity 5 \
    --max-capacity 10
  ```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、ResourceId、ScalableDimension、ServiceNamespace、MinCapacity および MaxCapacity をパラメータとして指定します。

AWS Lambda と Application Auto Scaling

ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用して、AWS Lambda プロビジョニングされた同時実行数をスケールできます。

以下の情報を使用して、Lambda の Application Auto Scaling との統合に役立ててください。

Lambda 関数のスケーリングを始めたばかりの場合は、以下のドキュメントで Lambda での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- 詳細については、AWS Lambda デベロッパーガイドの Lambda プロビジョン同時実行数の管理を参照してください。

Lambda 用に作成されたサービスリンクロール

Lambda リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `lambda.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての Lambda 関数の Application Auto Scaling への登録

Application Auto Scaling では、Lambda 関数のスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  Lambda 関数に対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 0、最大容量を 100 として、`my-function` と呼ばれる関数の `BLUE` というエイリアスに対するプロビジョニングされた同時実行数を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace lambda \
  --scalable-dimension lambda:function:ProvisionedConcurrency \
  --resource-id function:my-function:BLUE \
  --min-capacity 0 \
  --max-capacity 100
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

Amazon Managed Streaming for Apache Kafka (MSK) と Application Auto Scaling

Amazon MSK クラスターストレージは、ターゲット追跡スケーリングポリシーを使用してスケールアウトできます。ターゲット追跡ポリシーによるスケールインが無効になっています。

以下の情報を使用して、Amazon MSK の Application Auto Scaling との統合に役立ててください。

Amazon MSK クラスターストレージのスケーリングを始めたばかりの場合は、以下のドキュメントで Amazon MSK での Application Auto Scaling の使用に関する詳細を確認できます。

- Amazon Managed Streaming for Apache Kafka デベロッパーガイドの「Auto-expanding storage for an Amazon MSK cluster」

Amazon MSK 用に作成されたサービスリンクロール

Amazon MSK リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_KafkaCluster`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `kafka.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての Amazon MSK クラスターストレージの Application Auto Scaling への登録

Application Auto Scaling では、Amazon MSK クラスターのブローカーごとのストレージボリュームサイズに対するスケーリングポリシーを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

Amazon MSK コンソールを使用してオートスケーリングを設定すると、Amazon MSK がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

Amazon MSK クラスターに対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 100 GiB、最大容量を 800 GiB として、Amazon MSK クラスターのブローカーあたりのストレージボリュームサイズを登録します。

```
aws application-autoscaling register-scalable-target \
   --service-namespace kafka \
   --scalable-dimension kafka:broker-storage:VolumeSize \
   --resource-id arn:aws:kafka:us-east-1:123456789012:cluster/demo-
cluster-1/6357e0b2-0e6a-4b86-a0b4-70df934c2e31-5 \
   --min-capacity 100 \
   --max-capacity 800
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

  Note

  Amazon MSK クラスターがスケーラブルターゲットである場合は、スケールインが無効化されており、有効にすることはできません。

Amazon Neptune と Application Auto Scaling

Neptune 関数は、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、Neptune の Application Auto Scaling との統合に役立ててください。

Neptune クラスターのスケーリングを始めたばかりの場合は、以下のドキュメントで、Neptune での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Neptune ユーザーガイドの Amazon Neptune DB クラスター内のレプリカの数の Auto Scaling

Neptune 用に作成されたサービスリンクロール

Neptune リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウント に以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_NeptuneCluster`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `neptune.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての Neptune クラスターの Application Auto Scaling への登録

Application Auto Scaling では、Neptune クラスターのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  Neptune クラスターに対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 1 個、および最大容量を 8 個のフリートインスタンスとして、`mycluster` という名前のクラスターの希望容量を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace neptune \
  --scalable-dimension neptune:cluster:ReadReplicaCount \
  --resource-id cluster:mycluster \
  --min-capacity 1 \
  --max-capacity 8
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

Amazon SageMaker と Application Auto Scaling

SageMaker エンドポイントバリアントは、ターゲット追跡スケーリングポリシー、ステップスケーリングポリシー、およびスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、SageMaker の Application Auto Scaling との統合に役立ててください。

SageMaker エンドポイントバリアントのスケーリングを始めたばかりの場合は、以下のドキュメントで SageMaker での Application Auto Scaling の使用に関するサンプル設定と詳細を確認できます。

- Amazon SageMaker デベロッパーガイドの「Automatically scale Amazon SageMaker models」

SageMaker 用に作成されたサービスリンクロール

SageMaker リソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `sagemaker.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしての SageMaker エンドポイントバリアントの Application Auto Scaling への登録

Application Auto Scaling では、SageMaker xxx のスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

SageMaker コンソールを使用してオートスケーリングを設定すると、SageMaker がユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  SageMaker エンドポイントバリアントに対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 1 個のインスタンス、最大容量を 8 個のインスタンスとして、`my-endpoint` エンドポイントで実行される `my-variant` と呼ばれる製品バリアントに対する EC2 インスタンスの希望数を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace sagemaker \
  --scalable-dimension sagemaker:variant:DesiredInstanceCount \
  --resource-id endpoint/my-endpoint/variant/my-variant \
  --min-capacity 1 \
  --max-capacity 8
```

- AWS SDK:

  RegisterScalableTarget オペレーションを呼び出し、`ResourceId`、`ScalableDimension`、`ServiceNamespace`、`MinCapacity`、および `MaxCapacity` をパラメータとして指定します。

Amazon EC2 スポットフリートと Application Auto Scaling

スポットフリートは、ターゲット追跡スケーリングポリシー、ステップスケーリングポリシー、およびスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、スポットフリートの Application Auto Scaling との統合に役立ててください。

スポットフリートのスケーリングを始めたばかりの場合は、以下のドキュメントでスポットフリートでの Application Auto Scaling の使用に関する詳細を確認できます。

- Amazon EC2 ユーザーガイドの「スポットフリートの自動スケーリング」

スポットフリート用に作成されたサービスリンクロール

スポットフリートリソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- `AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest`

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- `ec2.application-autoscaling.amazonaws.com`

スケーラブルターゲットとしてのスポットフリートの Application Auto Scaling への登録

Application Auto Scaling では、スポットフリートのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

スポットフリートコンソールを使用してオートスケーリングを設定すると、スポットフリートがユーザーに代わってスケーラブルターゲットを自動的に登録します。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

  スポットフリートに対して登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小容量を 2 個のインスタンス、および最大容量を 10 個のインスタンスとし、スポットフリートのリクエスト ID を使用してそのターゲット容量を登録します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 \
```

```
  --max-capacity 10
```

- AWS SDK:

  RegisterScalableTargetオペレーションを呼び出し、ResourceId、ScalableDimension、ServiceNamespace、MinCapacity、および MaxCapacity をパラメータとして指定します。

カスタムリソースと Application Auto Scaling

カスタムリソースは、ターゲット追跡スケーリングポリシー、ステップスケーリングポリシー、およびスケジュールされたスケーリングを使用してスケールできます。

以下の情報を使用して、カスタムリソースの Application Auto Scaling との統合に役立ててください。

カスタムリソースのスケーリングを始めたばかりの場合は、カスタムリソースが Application Auto Scaling と統合される方法について詳しく説明する GitHub リポジトリを参照することができます。

カスタムリソース用に作成されたサービスリンクロール

カスタムリソースをスケーラブルターゲットとして Application Auto Scaling に登録すると、AWS アカウントに以下のサービスリンクロールが自動的に作成されます。このロールは、アカウント内でサポートされている操作を実行することを Application Auto Scaling に許可します。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

- AWSServiceRoleForApplicationAutoScaling_CustomResource

サービスリンクロールが使用するサービスプリンシパル

前のセクションで説明したサービスリンクロールを引き受けることができるのは、ロールに定義された信頼関係によって認可されるサービスプリンシパルのみです。Application Auto Scaling が使用するサービスリンクロールは、以下のサービスプリンシパルに対するアクセス権を付与します。

- custom-resource.application-autoscaling.amazonaws.com

スケーラブルターゲットとしてのカスタムリソースの Application Auto Scaling への登録

Application Auto Scaling では、カスタムリソースのスケーリングポリシーまたはスケジュールされたアクションを作成する前に、スケーラブルターゲットが必要になります。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。スケーラブルターゲットは、リソース ID、スケーラブルディメンション、および名前空間の組み合わせによって一意に識別されます。

AWS SDK のいずれか、または AWS CLI を使用してオートスケーリングを設定するには、以下のオプションを使用できます。

- AWS CLI:

カスタムリソース用の登録-スケーラブル-ターゲットコマンドを呼び出します。以下の例は、最小希望数を 1 個のキャパシティーユニット、最大希望数を 10 個のキャパシティーユニットとして、カスタムリソースをスケーラブルターゲットとして登録します。custom-resource-id.txt ファイルにはリソース ID を識別する文字列が含まれており、これは Amazon API Gateway エンドポイント経由でのカスタムリソースへのパスを表します。

```
aws application-autoscaling register-scalable-target \
  --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --min-capacity 1 \
  --max-capacity 10
```

custom-resource-id.txt の内容:

```
https://example.execute-api.us-west-2.amazonaws.com/prod/
scalableTargetDimensions/1-23456789
```

- AWS SDK:

  RegisterScalableTarget オペレーションを呼び出し、ResourceId、ScalableDimension、ServiceNamespace、MinCapacityおよび MaxCapacity をパラメータとして指定します。

Application Auto Scaling のスケジュールされたスケーリング

スケジュールに基づいたスケーリングにより、予想可能な負荷の変化に従って独自のスケーリングスケジュールを設定できます。例えば、毎週、ウェブアプリケーションへのトラフィックが水曜日に増え始め、木曜日は高いままで、金曜日に減り始めるとします。この場合は、水曜日に容量を増やし、金曜日に容量を減らすように Application Auto Scaling のスケジュールを設定できます。

スケジュールされたスケーリングを使用するには、スケジュールされたアクションを作成します。これは、特定の時間にスケーリングアクティビティを実行するよう Application Auto Scaling に指示します。スケジュールされたアクションを作成するときは、スケーラブルターゲット、スケーリングアクティビティを実行するタイミング、最小容量、および最大容量を指定します。スケジュールされたアクションは、1 度だけスケールする、または定期的なスケジュールに従ってスケールするものを作成できます。

指定された時間がくると、Application Auto Scaling は、現行の容量を指定された最小容量および最大容量と比較することによって、新しい容量値に基づいたスケーリングを実行します。

- 現行の容量が指定された最小容量を下回る場合、Application Auto Scaling は指定された最小容量までスケールアウト (容量を増加) します。
- 現行の容量が指定された最大容量を上回る場合、Application Auto Scaling は指定された最大容量までスケールイン (容量を低減) します。

同じリソース上でスケジュールされたスケーリングとスケーリングポリシーを併用して、両方のメリットを得ることができます。スケジュールされたアクションの実行後、スケーリングポリシーは容量をさらにスケールするかどうかの判断を引き続き行うことができます。これは、アプリケーションの負荷を処理するために十分な容量を確保する上で役立ちます。アプリケーションは需要に合わせてスケールしますが、現行の容量は、スケジュールされたアクションによって設定された最小容量と最大容量内に収まる必要があります。

スケジュールされたスケーリングの使用に関する詳しい例については、AWS コンピューティングブログ「Scheduling AWS Lambda Provisioned Concurrency for recurring peak usage」を参照してください。サンプル AWS リソースを使用してスケジュールされたアクションを作成する方法を詳しく説明するチュートリアルについては、「チュートリアル:AWS CLI を使用したスケジュールに基づくスケーリングの開始方法 (p. 38)」を参照してください。

考慮事項

スケジュールされたアクションを作成する場合、次の点に注意してください。

- スケジュールされたアクションにより、指定された日時に、`MinCapacity` と `MaxCapacity` がスケジュールされたアクションで指定した容量に設定されます。リクエストには、オプションで、これらのサイズの 1 つだけを含めることができます。例えば、最小容量のみを指定してスケジュールされたアクションを作成できます。ただし、場合によっては、新しい最小容量が最大容量を上回らない、または新しい最大容量が最小容量を下回らないように、両方のサイズを含める必要があります。
- デフォルトでは、設定した定期的なスケジュールは協定世界時 (UTC) です。ローカルタイムゾーンまたはネットワークの他の部分のタイムゾーンに対応するタイムゾーンに変更できます。夏時間を実施するタイムゾーンを指定すると、夏時間 (DST) に合わせて、アクションが自動的に調整されます。詳細については、「cron 式を使用して、定期的なスケーリングアクションをスケジュールする (p. 28)」を参照してください。
- スケーラブルターゲットに対してスケジュールされたスケーリングをオフにできます。これにより、スケジュールされたアクションを削除せずにアクティブになるのを防ぐことができます。スケジュールさ

れたスケーリングを再度使用する場合は、スケジュールされたスケーリングを再開できます。詳細については、「Application Auto Scaling のスケーリングの一時停止と再開 (p. 70)」を参照してください。

- スケジュールされたアクションが実行される順序は、同一のスケーラブルターゲットに対して保証されますが、複数のスケーラブルターゲットにまたがってスケジュールされたアクションに対しては保証されません。
- スケジュールされたアクションが正常に完了するには、指定されたリソースがターゲットサービスでスケーラブルな状態になっている必要があります。その状態になっていない場合、リクエストは失敗し、エラーメッセージ (`Resource Id [ActualResourceId] is not scalable. Reason: The status of all DB instances must be 'available' or 'incompatible-parameters'` など) が返されます。
- Application Auto Scaling とターゲットサービスには分散的な性質があるため、スケジュールされたアクションがトリガーされてからターゲットサービスがスケーリングアクションを引き受けるまでの遅延が数秒におよぶ可能性があります。スケジュールされたアクションは指定された順序で実行されるため、開始時刻が近いスケジュールされたアクションの実行にはより長い時間がかかる場合があります。

スケジュールされたアクションの作成、管理、および削除用によく使用されるコマンド

スケジュールされたスケーリングの操作用によく使用されるコマンドには以下が含まれます。

- register-scalable-target AWS リソースまたはカスタムリソースをスケーラブルターゲット (Application Auto Scaling がスケールできるリソース) として登録し、スケーリングを一時停止および再開します。
- put-scheduled-action 既存のスケーラブルターゲットに対するスケジュールされたアクションを追加または変更します。
- describe-scaling-activities AWS リージョン内でのスケーリングアクティビティに関する情報を返します。
- describe-scheduled-actions AWS リージョン内でのスケジュールされたアクションに関する情報を返します。
- delete-scheduled-action スケジュールされたアクションを削除します。

制約事項

以下は、スケジュールされたスケーリングの使用時における制限事項です。

- スケジュールされたアクションの名前は、スケーラブルターゲットごとに一意である必要があります。
- Application Auto Scaling は、スケジュール式で秒レベルの精度を提供しません。Cron 式を使用した場合の最も細かい粒度は 1 分です。
- スケーラブルターゲットを Amazon MSK クラスターにすることはできません。Amazon MSK はスケジュールされたスケーリングをサポートしません。

cron 式を使用して、定期的なスケーリングアクションをスケジュールする

cron 式を使用して、定期的なスケジュールで実行されるスケジュールされたアクションを作成できます。

定期的なスケジュールを作成する場合は、cron 式とタイムゾーンを指定して、スケジュールされたアクションがいつ繰り返されるのかを記述します。サポートされているタイムゾーン値は、Joda-Time でサポートされている IANA タイムゾーンの正規名です (`Etc/GMT+9`、`Pacific/Tahiti` など)。必要に応じ

て、開始時刻、終了時刻、またはその両方の日付と時刻を指定できます。AWS CLI を使用してスケジュールされたアクションを作成するコマンドの例については、「タイムゾーンを指定する定期的なスケジュールされたアクションを作成する (p. 33)」を参照してください。

サポートされている cron 式の形式は、スペースで区切られた [Minutes] [Hours] [Day_of_Month] [Month] [Day_of_Week] [Year] の 6 つのフィールドで構成されます。例えば、cron 式 `30 6 * * 2 *` は毎週火曜日の午前 6:30 に繰り返されるスケジュールされたアクションを設定します。アスタリスクは、フィールドのすべての値を照合するワイルドカードとして使用されます。Cron 式の記述に関する詳細については、Amazon CloudWatch Events ユーザーガイドの「Cron 式」を参照してください。

定期的なスケジュールを作成するときは、開始時刻と終了時刻を慎重に選択します。以下に留意してください。

- 開始時刻を指定すると、Application Auto Scaling はこの時刻にアクションを実行し、その後は指定された反復周期に基づいてアクションを実行します。
- 終了時刻を指定すると、その時刻以降はアクションが反復されなくなります。Application Auto Scaling は以前の値を記録せず、終了時刻後に以前の値に戻ることはありません。
- AWS CLI または AWS SDK を使用して、スケジュールされたアクションを作成または更新する場合、開始時刻と終了時刻を UTC で設定する必要があります。

例

Application Auto Scaling のスケーラブルターゲットに対して定期的なスケジュールを作成する場合は、次の表を参照してください。次は、Application Auto Scaling を使用して、スケジュールされたアクションを作成または更新するための正しい構文の例です。

分	時間	日	月	曜日	年	意味
0	10	*	*	?	*	毎日午前 10:00 (UTC) に実行
15	12	*	*	?	*	毎日午後 12:15 (UTC) に実行
0	18	?	*	MON-FRI	*	毎週月曜日から金曜日まで午後 6:00 (UTC) に実行
0	8	1	*	?	*	毎月 1 日の午前 8:00 (UTC) に実行
0/15	*	*	*	?	*	15 分ごとに実行
0/10	*	?	*	MON-FRI	*	月曜日から金曜日まで 10 分ごとに実行
0/5	8-17	?	*	MON-FRI	*	毎週月曜日から金曜日まで午前

分	時間	日	月	曜日	年	意味
						8:00 から午後 5:55 (UTC) の間に 5 分ごとに実行

Exception

7 つのフィールドを含む文字列値を使用して cron 式を作成することもできます。この場合、最初の 3 つのフィールドを使用して、スケジュールされたアクションを実行する時間を秒単位で指定できます。完全な cron 式には、スペースで区切られた [Seconds] [Minutes] [Hours] [Day_of_Month] [Month] [Day_of_Week] [Year] のフィールドが含まれます。ただし、この方法は、スケジュールされたアクションが指定した秒に正確に実行されることを保証するものではありません。また、一部のサービスコンソールでは、cron 式の 2 番目のフィールドがサポートされていない場合があります。

Application Auto Scaling のスケジュールされたアクションの例

以下の例は、AWS CLI put-scheduled-action コマンドを使用してスケジュールされたアクションを作成する方法を説明しています。新しい容量を指定するときは、最小容量、最大容量、またはその両方を指定できます。

> **Note**
>
> 簡略化のため、このトピックの例では、Application Auto Scaling と統合されている一部のサービス用の CLI コマンドを例示しています。別のスケーラブルターゲットを指定するには、`--service-namespace` でその名前空間、`--scalable-dimension` でそのスケーラブルディメンション、`--resource-id` でそのリソース ID を指定します。

目次

1 回だけ実行される、スケジュールされたアクションを作成する

指定した日時にスケーラブルターゲットを 1 度だけ自動的にスケーリングするには、`--schedule "at(yyyy-mm-ddThh:mm:ss)"` オプションを使用します。

Example 例: 1 回限りのスケールアウト

以下は、特定の日時に容量をスケールアウトするためのスケジュールされたアクションを作成する例です。

`--schedule` に指定された日時 (2021 年 3 月 31 日の午後 10:00 (UTC)) の時点で、`MinCapacity` に指定された値が現行の容量を超えている場合、Application Auto Scaling が `MinCapacity` にスケールアウトします。

Linux、macOS、または Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-out \
  --schedule "at(2021-03-31T22:00:00)" \
  --scalable-target-action MinCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource --
scalable-dimension custom-resource:ResourceType:Property --resource-id file://~/custom-
resource-id.txt --scheduled-action-name scale-out --schedule "at(2021-03-31T22:00:00)" --
scalable-target-action MinCapacity=3
```

> **Note**
>
> このスケジュールされたアクションの実行時に、最大容量が最小容量に指定された値を下回る場合は、新しい最小容量だけではなく、新しい最小容量と最大容量を指定する必要があります。

Example 例: 1 回限りのスケールイン

以下は、特定の日時に容量をスケールインするためのスケジュールされたアクションを作成する例です。

`--schedule` に指定された日時 (2021 年 3 月 31 日の午後 10:30 (UTC)) の時点で、`MaxCapacity` に指定された値が現行の容量を下回る場合、Application Auto Scaling が `MaxCapacity` にスケールインします。

Linux、macOS、または Unix

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource \
  --scalable-dimension custom-resource:ResourceType:Property \
  --resource-id file://~/custom-resource-id.txt \
  --scheduled-action-name scale-in \
  --schedule "at(2021-03-31T22:30:00)" \
  --scalable-target-action MinCapacity=0,MaxCapacity=0
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace custom-resource --
scalable-dimension custom-resource:ResourceType:Property --resource-id file://~/custom-
resource-id.txt --scheduled-action-name scale-in --schedule "at(2021-03-31T22:30:00)" --
scalable-target-action MinCapacity=0,MaxCapacity=0
```

定期的な間隔で実行されるスケジュールされたアクションを作成する

定期的な間隔でスケーリングをスケジュールするには、`--schedule "rate(value unit)"` オプションを使用します。値は正の整数である必要があります。単位は、`minute`、`minutes`、`hour`、`hours`、`day`、または `days` にすることができます。詳細については、Amazon CloudWatch Events ユーザーガイドの「rate 式」を参照してください。

以下は、rate 式を使用するスケジュールされたアクションの例です。

指定されたスケジュール (2021 年 1 月 30 日の午後 12:00 (UTC) から 5 時間ごとに実行され、2021 年 1 月 31 日の午後 10:00 (UTC) に終了) で、`MinCapacity` に指定された値が現行の容量を超えている場

合、Application Auto Scaling が MinCapacity にスケールアウトします。MaxCapacity に指定された値が現行の容量を下回る場合は、Application Auto Scaling が MaxCapacity にスケールインします。

Linux、macOS、または Unix

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/web-app \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace ecs --scalable-
dimension ecs:service:DesiredCount --resource-id service/default/web-app --scheduled-
action-name my-recurring-action --schedule "rate(5 hours)" --start-time 2021-01-30T12:00:00
 --end-time 2021-01-31T22:00:00 --scalable-target-action MinCapacity=3,MaxCapacity=10
```

定期的なスケジュールで実行されるスケジュールされたアクションを作成する

定期的なスケーリングをスケジュールするには、--schedule "cron(*fields*)" オプションを使用します。詳細については、「cron 式を使用して、定期的なスケーリングアクションをスケジュールする (p. 28)」を参照してください。

以下は、Cron 式を使用するスケジュールされたアクションの例です。

指定されたスケジュール (毎日午前 9:00 (UTC)) で、MinCapacity に指定された値が現行の容量を超えている場合、Application Auto Scaling が MinCapacity にスケールアウトします。MaxCapacity に指定された値が現行の容量を下回る場合は、Application Auto Scaling が MaxCapacity にスケールインします。

Linux、macOS、または Unix

```
aws application-autoscaling put-scheduled-action --service-namespace appstream \
  --scalable-dimension appstream:fleet:DesiredCapacity \
  --resource-id fleet/sample-fleet \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 9 * * ? *)" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace appstream --scalable-
dimension appstream:fleet:DesiredCapacity --resource-id fleet/sample-fleet --scheduled-
action-name my-recurring-action --schedule "cron(0 9 * * ? *)" --scalable-target-action
 MinCapacity=10,MaxCapacity=50
```

タイムゾーンを指定する 1 回限りのスケジュールされたアクションを作成する

スケジュールされたアクションは、デフォルトで UTC タイムゾーンに設定されます。別のタイムゾーンを指定するには、--timezone オプションを含めて、タイムゾーンの正規名 (America/New_York など) を

指定します。詳細については、https://www.joda.org/joda-time/timezones.html を参照してください。このページには、put-scheduled-action を呼び出すときにサポートされる IANA タイムゾーンに関する情報が記載されています。

以下は、特定の日時に容量をスケールするためのスケジュールされたアクションの作成時に `--timezone` オプションを使用する例です。

`--schedule` に指定された日時 (2021 年 1 月 31 日の午後 5:00 (ローカルタイム)) の時点で、`MinCapacity` に指定された値が現行の容量を超えている場合、Application Auto Scaling が `MinCapacity` にスケールアウトします。`MaxCapacity` に指定された値が現行の容量を下回る場合は、Application Auto Scaling が `MaxCapacity` にスケールインします。

Linux、macOS、または Unix

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend \
  --scalable-dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits \
  --resource-id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/
EXAMPLE \
  --scheduled-action-name  my-one-time-action \
  --schedule "at(2021-01-31T17:00:00)" --timezone "America/New_York" \
  --scalable-target-action MinCapacity=1,MaxCapacity=3
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace comprehend --scalable-
dimension comprehend:document-classifier-endpoint:DesiredInferenceUnits --resource-
id arn:aws:comprehend:us-west-2:123456789012:document-classifier-endpoint/EXAMPLE --
scheduled-action-name  my-one-time-action --schedule "at(2021-01-31T17:00:00)" --timezone
 "America/New_York" --scalable-target-action MinCapacity=1,MaxCapacity=3
```

タイムゾーンを指定する定期的なスケジュールされたアクションを作成する

以下は、キャパシティーを拡張するための定期的なスケジュール済みアクションを作成できる `--timezone` オプションの使用例です。詳細については、「cron 式を使用して、定期的なスケーリングアクションをスケジュールする (p. 28)」を参照してください。

指定されたスケジュール (毎週月曜日から金曜日までの午後 6:00 (ローカルタイム)) で、`MinCapacity` に指定された値が現行の容量を超えている場合、Application Auto Scaling が `MinCapacity` にスケールアウトします。`MaxCapacity` に指定された値が現行の容量を下回る場合は、Application Auto Scaling が `MaxCapacity` にスケールインします。

Linux、macOS、または Unix

```
aws application-autoscaling put-scheduled-action --service-namespace lambda \
  --scalable-dimension lambda:function:ProvisionedConcurrency \
  --resource-id function:my-function:BLUE \
  --scheduled-action-name my-recurring-action \
  --schedule "cron(0 18 ? * MON-FRI *)" --timezone "Etc/GMT+9" \
  --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace lambda --scalable-
dimension lambda:function:ProvisionedConcurrency --resource-id function:my-function:BLUE --
scheduled-action-name my-recurring-action --schedule "cron(0 18 ? * MON-FRI *)" --timezone
 "Etc/GMT+9" --scalable-target-action MinCapacity=10,MaxCapacity=50
```

Application Auto Scaling のスケジュールされたスケーリングを管理する

AWS CLI には、スケジュールされたアクションの管理に役立つ他のコマンドがいくつか含まれています。

> **Note**
>
> 簡略化のため、このトピックの例では、Application Auto Scaling と統合されている一部のサービス用の CLI コマンドを例示しています。別のスケーラブルターゲットを指定するには、`--service-namespace` でその名前空間、`--scalable-dimension` でそのスケーラブルディメンション、`--resource-id` でそのリソース ID を指定します。

目次

指定されたサービスのスケーリングアクティビティを表示する

指定されたサービス名前空間にあるすべてのスケーラブルターゲットに対するスケーリングアクティビティを表示するには、describe-scaling-activities コマンドを使用します。

以下の例は、`dynamodb` サービス名前空間に関連付けられているスケーリングアクティビティを取得します。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
```

コマンドが正常に完了した場合は、以下のような出力が表示されます。

```
{
    "ScalingActivities": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/my-table",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
 successfully fulfilled by dynamodb.",
```

```
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/my-table",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to 10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/my-table",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
 successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/my-table",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to 20",
            "StatusCode": "Successful"
        }
    ]
}
```

このコマンドを変更して、スケーラブルターゲットのうち 1 つのターゲットのみに関するスケーリングアクティビティを取得するようにするには、`--resource-id` オプションを追加します。

指定されたサービスに対するすべてのスケジュールされたアクションの記述

指定されたサービス名前空間にあるすべてのスケーラブルターゲットに対するスケジュールされたアクションを記述するには、describe-scheduled-actions コマンドを使用します。

以下の例は、`ec2` サービス名前空間に関連付けられているスケジュールされたアクションを取得します。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2
```

正常に完了した場合、このコマンドは以下のような出力を返します。

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/spot-
fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-one-time-
action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2021-01-31T17:00:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MaxCapacity": 1
            },
            "CreationTime": 1607454792.331
        },
        {
            "ScheduledActionName": "my-recurring-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:493a6261-fbb9-432d-855d-3c302c14bdb9:resource/ec2/spot-
fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE:scheduledActionName/my-recurring-
action",
            "ServiceNamespace": "ec2",
            "Schedule": "rate(5 minutes)",
            "ResourceId": "spot-fleet-request/sfr-107dc873-0802-4402-a901-37294EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "StartTime": 1604059200.0,
            "EndTime": 1612130400.0,
            "ScalableTargetAction": {
                "MinCapacity": 3,
                "MaxCapacity": 10
            },
            "CreationTime": 1607454949.719
        },
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/spot-
fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-time-
action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```


スケーラブルターゲットに対する 1 つまたは複数のスケジュールされたアクションを記述する

指定されたスケーラブルターゲットに対するスケジュールされたアクションの情報を取得するには、describe-scheduled-actions コマンドを使用してスケジュールされたアクションを記述するときに `--resource-id` オプションを追加します。

以下の例にあるように、--scheduled-action-names オプションを含めて、スケジュールされたアクションの名前をその値として指定すると、コマンドは名前が一致するスケジュールされたアクションのみを返します。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 \
  --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE \
  --scheduled-action-names my-one-time-action
```

Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace ec2 --resource-id spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE --scheduled-action-names my-one-time-action
```

以下は出力例です。

```
{
    "ScheduledActions": [
        {
            "ScheduledActionName": "my-one-time-action",
            "ScheduledActionARN": "arn:aws:autoscaling:us-
west-2:123456789012:scheduledAction:4bce34c7-bb81-4ecf-b776-5c726efb1567:resource/ec2/spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE:scheduledActionName/my-one-time-action",
            "ServiceNamespace": "ec2",
            "Schedule": "at(2020-12-08T9:36:00)",
            "Timezone": "America/New_York",
            "ResourceId": "spot-fleet-request/sfr-40edeb7b-9ae7-44be-bef2-5c4c8EXAMPLE",
            "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
            "ScalableTargetAction": {
                "MinCapacity": 1,
                "MaxCapacity": 3
            },
            "CreationTime": 1607456031.391
        }
    ]
}
```

--scheduled-action-names オプションに複数の値が指定されている場合、名前が一致するスケジュールされたアクションのすべてが出力に含まれます。

スケーラブルターゲットに対するスケジュールされたスケーリングをオフにする

スケジュールされたスケーリングは、スケジュールされたアクションを削除せずに一時的に無効化することができます。詳細については、「Application Auto Scaling のスケーリングの一時停止と再開 (p. 70)」を参照してください。

以下の例にあるように、--suspended-state オプションがある register-scalable-target コマンドを使用し、ScheduledScalingSuspended 属性の値として true を指定することによって、スケーラブルターゲットでスケジュールされたスケーリングを一時停止します。

Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target --service-namespace rds \
  --scalable-dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster \
  --suspended-state '{"ScheduledScalingSuspended": true}'
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace rds --scalable-
dimension rds:cluster:ReadReplicaCount --resource-id cluster:my-db-cluster --suspended-
state "{\"ScheduledScalingSuspended\": true}"
```

正常に完了すると、このコマンドはプロンプトに戻ります。

スケジュールされたスケーリングを再開するには、`ScheduledScalingSuspended` 属性の値として `false` を指定して、このコマンドを再度実行します。

スケジュールされたアクションの削除

スケジュールされたアクションが不要になったら、delete-scheduled-action コマンドを使用してアクションを削除することができます。

Linux、macOS、または Unix

```
aws application-autoscaling delete-scheduled-action --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE \
  --scheduled-action-name my-recurring-action
```

Windows

```
aws application-autoscaling delete-scheduled-action --service-namespace ec2 --scalable-
dimension ec2:spot-fleet-request:TargetCapacity --resource-id spot-fleet-request/
sfr-73fbd2ce-aa30-494c-8788-37294EXAMPLE --scheduled-action-name my-recurring-action
```

正常に完了すると、このコマンドはプロンプトに戻ります。

チュートリアル:AWS CLI を使用したスケジュールに基づくスケーリングの開始方法

次のチュートリアルでは、AWS CLI を使用して、`TestTable` と呼ばれるサンプル DynamoDB テーブルをスケーリングするスケジュールされたアクションの作成を支援することにより、スケジュールされたスケーリングを開始する方法を示します。テストに使用する`TestTable` テーブルが DynamoDB にまだない場合は、Amazon DynamoDB デベロッパーガイドのステップ 1 : DynamoDB テーブルを作成するに示されている create-table コマンドを実行してテーブルを作成できます。

AWS CLI を使用するときは、プロファイルに設定した AWS リージョンでコマンドが実行されることを覚えておいてください。別のリージョンでコマンドを実行する場合は、プロファイルのデフォルトのリージョンを変更するか、コマンドに `--region` パラメータを使用します。

> **Note**
>
> このチュートリアルの一環として AWS 料金が発生する場合があります。無料利用枠の使用状況をモニタリングするとともに、DynamoDB データベースが使用する読み取りおよび書き込み容量のユニット数に関連付けられたコストを理解しておくようにしてください。

目次

ステップ 1: スケーラブルターゲットを登録する

スケーラブルターゲットとしてDynamoDB テーブルを Application Auto Scaling に登録することから始めます。

Application Auto Scaling にスケーラブルなターゲットを登録する

1. まず、describe-scalable-targets コマンドを使用して、DynamoDB リソースが既に登録されているかどうかをチェックします。これにより、新しいテーブルではない場合に備えて、`TestTable` テーブルは登録解除されているか確認します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling describe-scalable-targets \
     --service-namespace dynamodb
   ```

   Windows

   ```
   aws application-autoscaling describe-scalable-targets --service-namespace dynamodb
   ```

   既存のスケーラブルなターゲットがない場合、レスポンスは次のようになります。

   ```
   {
       "ScalableTargets": []
   }
   ```

2. 以下の register-scalable-target コマンドを使用して、`TestTable` と呼ばれる DynamoDB テーブルの書き込み容量を登録します。希望する最小容量を 5 個の書き込みキャパシティーユニット、希望する最大容量を 10 個の書き込みキャパシティーユニットに設定します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling register-scalable-target \
     --service-namespace dynamodb \
     --scalable-dimension dynamodb:table:WriteCapacityUnits \
     --resource-id table/TestTable \
     --min-capacity 5 --max-capacity 10
   ```

   Windows

   ```
   aws application-autoscaling register-scalable-target --service-namespace dynamodb --
   scalable-dimension dynamodb:table:WriteCapacityUnits --resource-id table/TestTable --
   min-capacity 5 --max-capacity 10
   ```

   このコマンドが正常に完了した場合は、出力が返されません。

ステップ 2: 2 つのスケジュールされたアクションを作成する

Application Auto Scaling では、スケーリングアクションが実行される時刻をスケジュールすることができます。スケーラブルなターゲット、スケジュール、最小キャパシティー、および最大キャパシティーを指

定します。指定された時間がくると、Application Auto Scaling がスケーラブルターゲットの最小値と最大値を更新します。現在のキャパシティーが範囲外の場合はスケーリングアクティビティにつながります。

最小キャパシティーと最大キャパシティーへの更新のスケジュールは、スケーリングポリシーを作成する場合にも役立ちます。スケーリングポリシーを使用すると、現在のリソース使用率に基づいてリソースを動的にスケーリングできます。スケーリングポリシーの一般的なガードレールとして、最小キャパシティーと最大キャパシティーに適切な値を設定します。

この演習では、1 回限りのアクションを 2 つ作成し、スケールアウトまたはスケールインできます。

スケジュールされたアクションを作成および表示する

1. 最初のスケジュールされたアクションを作成するには、以下の put-scheduled-action コマンドを使用します。

   `--schedule` の at コマンドは、指定された将来の日時に 1 回実行されるアクションをスケジュールします。時間は 24 時間形式 (UTC) です。今から約 5 分後にアクションが発生するようにスケジュールします。

   指定された日時がくると、Application Auto Scaling が `MinCapacity` および `MaxCapacity` の値を更新します。テーブルに現在 5 個の書き込みキャパシティーユニットがあるとすると、Application Auto Scaling は `MinCapacity` にスケールアウトして、そのテーブルが 15 ~ 20 書き込みキャパシティーユニットの新しい希望範囲内に収まるようにします。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace dynamodb \
     --scalable-dimension dynamodb:table:WriteCapacityUnits \
     --resource-id table/TestTable \
     --scheduled-action-name my-first-scheduled-action \
     --schedule "at(2019-05-20T17:05:00)" \
     --scalable-target-action MinCapacity=15,MaxCapacity=20
   ```

   Windows

   ```
   aws application-autoscaling put-scheduled-action --service-namespace dynamodb --
   scalable-dimension dynamodb:table:WriteCapacityUnits --resource-id table/TestTable --
   scheduled-action-name my-first-scheduled-action --schedule "at(2019-05-20T17:05:00)" --
   scalable-target-action MinCapacity=15,MaxCapacity=20
   ```

   このコマンドが正常に完了した場合は、出力が返されません。

2. Application Auto Scaling がスケールインするために使用する 2 番目のスケジュールされたアクションを作成するには、以下の put-scheduled-action コマンドを使用します。

   今から約 10 分後にアクションが発生するようにスケジュールします。

   指定された日時がくると、Application Auto Scaling がテーブルの `MinCapacity` と `MaxCapacity` を更新して `MaxCapacity` にスケールインし、テーブルを元の希望範囲である 5 ~ 10 書き込みキャパシティーユニットに戻します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace dynamodb \
     --scalable-dimension dynamodb:table:WriteCapacityUnits \
     --resource-id table/TestTable \
     --scheduled-action-name my-second-scheduled-action \
     --schedule "at(2019-05-20T17:10:00)" \
   ```

```
  --scalable-target-action MinCapacity=5,MaxCapacity=10
```

Windows

```
aws application-autoscaling put-scheduled-action --service-namespace dynamodb --
scalable-dimension dynamodb:table:WriteCapacityUnits --resource-id table/TestTable --
scheduled-action-name my-second-scheduled-action --schedule "at(2019-05-20T17:10:00)"
 --scalable-target-action MinCapacity=5,MaxCapacity=10
```

3. (オプション) 以下の describe-scheduled-actions コマンドを使用して、指定されたサービス名前空間に対するスケジュールされたアクションのリストを取得します。

   Linux、macOS、または Unix

```
aws application-autoscaling describe-scheduled-actions \
  --service-namespace dynamodb
```

   Windows

```
aws application-autoscaling describe-scheduled-actions --service-namespace dynamodb
```

   以下は出力例です。

```
{
    "ScheduledActions": [
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:35:00)",
            "ResourceId": "table/TestTable",
            "CreationTime": 1561571888.361,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/TestTable:scheduledActionName/my-first-scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 15,
                "MaxCapacity": 20
            },
            "ScheduledActionName": "my-first-scheduled-action",
            "ServiceNamespace": "dynamodb"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Schedule": "at(2019-05-20T18:40:00)",
            "ResourceId": "table/TestTable",
            "CreationTime": 1561571946.021,
            "ScheduledActionARN": "arn:aws:autoscaling:us-
east-1:123456789012:scheduledAction:2d36aa3b-cdf9-4565-b290-81db519b227d:resource/
dynamodb/table/TestTable:scheduledActionName/my-second-scheduled-action",
            "ScalableTargetAction": {
                "MinCapacity": 5,
                "MaxCapacity": 10
            },
            "ScheduledActionName": "my-second-scheduled-action",
            "ServiceNamespace": "dynamodb"
        }
    ]
}
```

ステップ 3: スケーリングアクティビティを表示する

このステップでは、スケジュールされたアクションによってトリガーされたスケーリングアクティビティを表示してから、DynamoDB がテーブルの書き込みキャパシティーを変更したことを確認します。

スケーリングアクティビティを表示する

1. 選択した時間まで待機し、以下の describe-scaling-activities コマンドを使用して、スケジュールされたアクションが機能していることを確認します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling describe-scaling-activities \
     --service-namespace dynamodb
   ```

   Windows

   ```
   aws application-autoscaling describe-scaling-activities --service-namespace dynamodb
   ```

   以下は、スケジュールされたアクションの進行中に、スケジュールされた最初のアクションの出力例です。

   スケーリングアクティビティは作成日順に並べられ、最新のスケーリングアクティビティが最初に返ります。

   ```
   {
       "ScalingActivities": [
           {
               "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
               "Description": "Setting write capacity units to 15.",
               "ResourceId": "table/TestTable",
               "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
               "StartTime": 1561574108.904,
               "ServiceNamespace": "dynamodb",
               "Cause": "minimum capacity was set to 15",
               "StatusMessage": "Successfully set write capacity units to 15. Waiting for
    change to be fulfilled by dynamodb.",
               "StatusCode": "InProgress"
           },
           {
               "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
               "Description": "Setting min capacity to 15 and max capacity to 20",
               "ResourceId": "table/TestTable",
               "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
               "StartTime": 1561574108.512,
               "ServiceNamespace": "dynamodb",
               "Cause": "scheduled action name my-first-scheduled-action was triggered",
               "StatusMessage": "Successfully set min capacity to 15 and max capacity to
    20",
               "StatusCode": "Successful"
           }
       ]
   }
   ```

   以下は、スケジュールされたアクションがいずれも実行された後の出力例です。

   ```
   {
       "ScalingActivities": [
           {
   ```

```
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 10.",
            "ResourceId": "table/TestTable",
            "ActivityId": "4d1308c0-bbcf-4514-a673-b0220ae38547",
            "StartTime": 1561574415.086,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574449.51,
            "Cause": "maximum capacity was set to 10",
            "StatusMessage": "Successfully set write capacity units to 10. Change
successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 5 and max capacity to 10",
            "ResourceId": "table/TestTable",
            "ActivityId": "f2b7847b-721d-4e01-8ef0-0c8d3bacc1c7",
            "StartTime": 1561574414.644,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-second-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 5 and max capacity to
10",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting write capacity units to 15.",
            "ResourceId": "table/TestTable",
            "ActivityId": "d8ea4de6-9eaa-499f-b466-2cc5e681ba8b",
            "StartTime": 1561574108.904,
            "ServiceNamespace": "dynamodb",
            "EndTime": 1561574140.255,
            "Cause": "minimum capacity was set to 15",
            "StatusMessage": "Successfully set write capacity units to 15. Change
successfully fulfilled by dynamodb.",
            "StatusCode": "Successful"
        },
        {
            "ScalableDimension": "dynamodb:table:WriteCapacityUnits",
            "Description": "Setting min capacity to 15 and max capacity to 20",
            "ResourceId": "table/TestTable",
            "ActivityId": "3250fd06-6940-4e8e-bb1f-d494db7554d2",
            "StartTime": 1561574108.512,
            "ServiceNamespace": "dynamodb",
            "Cause": "scheduled action name my-first-scheduled-action was triggered",
            "StatusMessage": "Successfully set min capacity to 15 and max capacity to
20",
            "StatusCode": "Successful"
        }
    ]
}
```

2. スケジュールされたアクションが正常に実行されたら、DynamoDB コンソールに移動して、使用するテーブルを選択します。[Capacity] (キャパシティー) タブで、[Write capacity units] (書き込みキャパシティーユニット) を確認します。2 番目のスケーリングアクションが実行されると、書き込みキャパシティーユニットは 15 から 10 にスケーリングされます。

   describe-table コマンドを使用して、テーブルの現在の書き込み容量も確認します。出力をフィルタリングするには、`--query` オプションを含めます。AWS CLI の出力フィルタリング機能の詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLI からのコマンド出力の制御」を参照してください。

   Linux、macOS、または Unix

```
aws dynamodb describe-table --table-name TestTable \
  --query 'Table.[TableName,TableStatus,ProvisionedThroughput]'
```

Windows

```
aws dynamodb describe-table --table-name TestTable --query "Table.
[TableName,TableStatus,ProvisionedThroughput]"
```

以下は出力例です。

```
[
    "TestTable",
    "ACTIVE",
    {
        "NumberOfDecreasesToday": 1,
        "WriteCapacityUnits": 10,
        "LastIncreaseDateTime": 1561574133.264,
        "ReadCapacityUnits": 5,
        "LastDecreaseDateTime": 1561574435.607
    }
]
```

ステップ 4: 次のステップ

スケジュールされたスケーリングポリシーとスケーリングポリシーの両方でスケーリングを試す場合は、チュートリアル: 大量のワークロードを処理するためのオートスケーリングの設定 (p. 62) の手順に従います。

ステップ 5 : クリーンアップ

「使用開始」演習が終了したら、関連付けられているリソースを以下のようにクリーンアップすることができます。

スケジュールされたアクションを削除する

以下の delete-scheduled-action コマンドは、指定されているスケジュールされたアクションを削除します。後で使用できるように、スケジュールされたアクションを保持する場合は、このステップをスキップできます。

Linux、macOS、または Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/TestTable \
  --scheduled-action-name my-second-scheduled-action
```

Windows

```
aws application-autoscaling delete-scheduled-action --service-namespace dynamodb --
scalable-dimension dynamodb:table:WriteCapacityUnits --resource-id table/TestTable --
scheduled-action-name my-second-scheduled-action
```

スケーラブルなターゲットを登録解除する

以下の deregister-scalable-target コマンドを使用して、スケーラブルターゲットの登録を解除します。自分で作成したスケーリングポリシーや、まだ削除されていないスケジュールされたアクションがある場合は、このコマンドによって削除されます。後で使用できるように、登録されたスケーラブルなターゲットを保持する場合は、このステップをスキップできます。

Linux、macOS、または Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:WriteCapacityUnits \
  --resource-id table/TestTable
```

Windows

```
aws application-autoscaling deregister-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:WriteCapacityUnits --resource-id table/TestTable
```

DynamoDB テーブルを削除する

以下の delete-table コマンドを使用して、このチュートリアルで使用したテーブルを削除します。後で使用できるように、テーブルを保持する場合は、このステップをスキップできます。

Linux、macOS、または Unix

```
aws dynamodb delete-table --table-name TestTable
```

Windows

```
aws dynamodb delete-table --table-name TestTable
```

Application Auto Scaling のターゲット追跡スケーリングポリシー

ターゲット追跡スケーリングポリシーでは、ユーザーがスケーリングメトリクスと一連のターゲット値を選択します。Application Auto Scaling は、スケーリングポリシーをトリガーする CloudWatch アラームを作成および管理し、メトリクスとターゲット値に基づいてスケーリング調整値を計算します。スケーリングポリシーは、指定されたターゲット値、またはそれに近い値にメトリクスを維持するため、必要に応じてキャパシティーを追加または削除します。ターゲット追跡スケーリングポリシーは、メトリクスをターゲット値に近い値に維持することに加えて、変化するロードパターンによるメトリクスの変化に適応します。

メトリクスの選択

ターゲット追跡スケーリングポリシーを作成するときは、以下の事前定義されたメトリクスを使用できます。オプションで、カスタムメトリクス仕様を使用することによって、ターゲット追跡スケーリングポリシーで監視および使用するメトリクスを定義することも可能です。

AppStream 2.0

- `AppStreamAverageCapacityUtilization`

Aurora

- `RDSReaderAverageCPUUtilization`
- `RDSReaderAverageDatabaseConnections`

Amazon Comprehend

- `ComprehendInferenceUtilization`

DynamoDB

- `DynamoDBReadCapacityUtilization`
- `DynamoDBWriteCapacityUtilization`

Amazon ECS

- `ALBRequestCountPerTarget` (ロードバランサーメトリクス)
- `ECSServiceAverageCPUUtilization`
- `ECSServiceAverageMemoryUtilization`

ElastiCache

- `ElastiCachePrimaryEngineCPUUtilization`
- `ElastiCacheReplicaEngineCPUUtilization`
- `ElastiCacheDatabaseMemoryUsageCountedForEvictPercentage`

Amazon Keyspaces (Apache Cassandra 向け)

- `CassandraReadCapacityUtilization`
- `CassandraWriteCapacityUtilization`

Lambda

- `LambdaProvisionedConcurrencyUtilization`

Amazon Managed Streaming for Apache Kafka (MSK)

- `KafkaBrokerStorageUtilization`

Neptune

- `NeptuneReaderAverageCPUUtilization`

スポットフリート (Amazon EC2)

- `ALBRequestCountPerTarget` (ロードバランサーメトリクス)
- `EC2SpotFleetRequestAverageCPUUtilization`
- `EC2SpotFleetRequestAverageNetworkIn`
- `EC2SpotFleetRequestAverageNetworkOut`

SageMaker

- `SageMakerVariantInvocationsPerInstance`

各メトリクスは、Amazon CloudWatch に保存されているデータポイントの時間順のセットを表します。AWS のメトリクスの多くはデフォルトで毎分報告されますが、Amazon EC2 メトリクスはデフォルトで 5 分ごとに報告されます。追加料金をお支払いいただくことで、詳細モニタリングを有効にして、1 分間隔でインスタンスのメトリクスデータを取得することができます。使用率の変化に対するより迅速な対応を確実にするためにも、詳細モニタリングの有効化をお勧めします。詳細については、Amazon EC2 – Linux インスタンス用ユーザーガイドの「インスタンスの詳細モニタリングを有効または無効にする」を参照してください。

メトリクスを選択するときは、以下の点に注意してください。

- メトリクスにはターゲット追跡に使用できないものもあります。これは、カスタムメトリクスを指定するときに重要になる場合があります。メトリクスは、有効な使用率メトリクスで、スケーラブルなターゲットの使用頻度を示す必要があります。メトリクス値は、スケーラブルなターゲットを比例的にスケールするためにメトリクスデータを使用できるようにするため、スケーラブルなターゲットの容量に対して比例的に増減する必要があります。
- `ALBRequestCountPerTarget` メトリクスを使用するには、`ResourceLabel` パラメータを指定して、メトリクスに関連付けられているターゲットグループを識別する必要があります。
- メトリクスが CloudWatch に実数 0 の値を出力する場合 (`ALBRequestCountPerTarget` など)、Application Auto Scaling は、アプリケーションへのトラフィックがない場合に 0 にスケールインできます。スケーラブルターゲットにリクエストがルーティングされないときにターゲットを 0 にスケールインするには、スケーラブルターゲットの最小容量が 0 に設定されている必要があります。
- サービスの中には、ターゲットサービスのコンソールを通じてカスタムメトリクスを管理できないものもあります。ターゲットサービスがコンソールでカスタムメトリクスをサポートするかどうかを確認するには、そのサービスのドキュメントを参照してください。

考慮事項

次の考慮事項に注意が必要です。

- ターゲットの追跡スケーリングポリシーでは、指定されたメトリクスがターゲット値を超えている場合、スケールアウトする必要があると見なされます。指定されたメトリクスがターゲット値を下回っている場合、ターゲットの追跡スケーリングポリシーを使用してスケールアウトすることはできません。
- ターゲット値と実際のメトリクスデータポイント間にギャップが発生する場合があります。これは、Application Auto Scaling が追加または削除する容量を判断するときに、その数を切り上げまたは切り捨てることによって、常に控えめに動作するためです。これにより、不十分な容量を追加したり、必要以上に容量を削除することを防ぎます。ただし、小容量のスケーラブルなターゲットの場合、実際のメトリクスデータポイントがターゲット値からかなり離れているように見えることがあります。
- 容量が大きいスケーラブルなターゲットでは、容量を追加または削除することにより、ターゲット値と実際のメトリクスデータポイントの間のギャップが少なくなります。
- アプリケーションの可用性を確保するため、Application Auto Scaling はスケールアウトをメトリクスに比例して可能な限り早急に実行しますが、スケールインはより段階的に行います。
- それぞれが異なるメトリクスを使用していれば、スケーラブルなターゲットに対して複数のターゲットの追跡スケーリングポリシーを設定できます。Application Auto Scaling の目的は常に可用性を優先することであるため、その動作は、スケールアウトまたはスケールインに対するターゲット追跡ポリシーの準備が整っているかどうかに応じて異なります。ターゲット追跡ポリシーのいずれかでスケールアウトする準備ができると、スケーラブルなターゲットがスケールアウトされますが、すべてのターゲット追跡ポリシー (スケールイン部分が有効) でスケールインする準備ができている場合のみスケールインされます。
- 複数のポリシーが、スケーラブルターゲットに対してスケールアウトまたはスケールインする指示を同時に出す場合、Application Auto Scaling はスケールインとスケールアウトのどちらについても、最大の容量を提供するポリシーに基づいてスケールします。これにより、複数のシナリオに対応する柔軟性が高まり、アプリケーションワークロードを処理するのに十分な容量が常に確保されます。
- ターゲット追跡スケーリングポリシーのスケールイン部分を無効にすることもできます。この機能には、スケールアウトに使用するのとは異なるメソッドをスケーリングに使用できる柔軟性があります。たとえば、スケールアウトにはターゲットの追跡スケーリングポリシーを使用しながら、スケールインにはステップスケーリングポリシーを使用できます。
- ただし、ターゲット追跡スケーリングポリシーをステップスケーリングポリシーとともに使用する場合、ポリシー間の競合によって望ましくない動作が生じる可能性があるため、注意することをお勧めします。たとえば、ターゲット追跡ポリシーがスケールインする準備が整う前に、ステップスケーリングポリシーがスケールインアクティビティを開始した場合、スケールインアクティビティはブロックされません。スケールインアクティビティが完了した後で、ターゲット追跡ポリシーにより、スケーラブルなターゲットに再びスケールアウトするよう指示できます。
- ターゲット追跡スケーリングポリシーは、アラームの状態が `INSUFFICIENT_DATA` になっているときにはスケールインしません。詳細については、「CloudWatch アラームでのモニタリング (p. 75)」を参照してください。
- ターゲット追跡スケーリングポリシー用に設定された CloudWatch アラームを編集または削除しないでください。ターゲット追跡スケーリングポリシーに関連付けられている CloudWatch アラームは AWS によって管理されており、不要になると自動的に削除されます。

クールダウン期間

前回のスケーリングアクティビティが有効になるまで待機する時間をクールダウン期間と呼びます。

ターゲット追跡スケーリングポリシーでは、次の 2 種類のクールダウン期間があります。

- スケールアウトクールダウン期間では、スケールアウトが継続的に (ただし過剰になることなく) 行われます。ターゲット追跡スケーリングポリシーを使用して Application Auto Scaling が正常にスケールア

ウトすると、クールダウン時間の計算が開始されます。スケーリングポリシーは、より大きなスケールアウトがトリガーされるか、クールダウン期間が終了しない限り、必要な容量を再度増加させません。このスケールアウトクールダウン期間が有効な間は、スケールアウトアクティビティを開始することで追加された容量は、次のスケールアウトアクティビティに予定される容量の一部として繰り入れられます。

- スケールインクールダウン期間では、スケールインを控え目に行ってアプリケーションの可用性を保護することを目的としているため、スケールインアクティビティはクールダウン期間が終了するまでブロックされます。ただし、スケールインクールダウン期間中に別のアラームがスケールアウトアクティビティをトリガーした場合、Application Auto Scaling scale によってターゲットが即座にスケールアウトされます。この場合、スケールインクールダウン期間は停止し、完了しません。

各クールダウン期間は秒単位で測定され、スケーリングポリシー関連のスケーリングアクティビティにのみ適用されます。クールダウン期間中、スケジュールされたアクションがスケジュールされた時間に開始されると、クールダウン期間の期限が切れるのを待たずにスケーリングアクティビティを即座にトリガーできます。

デフォルト値で開始し、値を後で微調整できます。たとえば、ターゲット追跡スケーリングポリシーが短期間に発生する変更に対して積極的になりすぎないように、場合によってはクールダウン期間を延長する必要があります。デフォルト値については、Application Auto Scaling API リファレンスの「TargetTrackingScalingPolicyConfiguration」を参照してください。

使用率の高い期間中のアプリケーションの可用性のサポート

ターゲット追跡スケーリングポリシーは、使用率が低下したときの容量の削除よりも、使用率が増加したときの容量の追加の方が積極的です。例えば、ポリシーの指定されたメトリクスがターゲット値に到達した場合、ポリシーはアプリケーションの負荷がすでに高くなっていると見なします。したがって、できるだけ早くメトリクス値に比例した容量を追加することで対応します。メトリクスが大きいほど、より多くの容量が追加されます。

メトリクスがターゲット値を下回ると、ポリシーは使用率が最終的には再び増加することを期待します。このため、ポリシーが容量を削除することによってスケーリングの速度を落とすのは、使用率がターゲット値を下回るしきい値未満になり (通常は 10% 以上低下)、そのレベルが使用率が減速したと見なされるに十分である場合のみになります。この保守的な動作の意図は、アプリケーションの需要が以前ほど高いレベルでなくなった場合にのみ、容量が削除されるようにすることです。これは現在、すべてのターゲット追跡スケーリングポリシーのデフォルトの動作です (ただし、動作は将来変更される可能性があります)。

周期的な性質のワークロードの場合、スケジュールされたスケーリングを使用してスケジュールに従って容量の変更を自動化することもできます。スケジュールされたアクションごとに、新しい最小容量値と新しい最大容量値を定義できます。これらの値は、スケーリングポリシーの境界を形成します。

スケジュールされたスケーリングとターゲットトラッキングスケーリングの組み合わせにより、容量がすぐに必要になったときに、使用率レベルの急激な増加による影響を軽減できます。

スケーリングポリシーの作成、管理、および削除用によく使用されるコマンド

スケーリングポリシーの操作用によく使用されるコマンドには以下が含まれます。

- register-scalable-target AWS リソースまたはカスタムリソースをスケーラブルターゲット (Application Auto Scaling がスケールできるリソース) として登録し、スケーリングを一時停止および再開します。

- put-scaling-policy 既存のスケーラブルターゲットのスケーリングポリシーを追加または変更します。
- describe-scaling-activities AWS リージョン内でのスケーリングアクティビティに関する情報を返します。
- describe-scaling-policies AWS リージョン内のスケーリングポリシーに関する情報を返します。
- delete-scaling-policy スケーリングポリシーを削除します。

機能制限

以下は、ターゲット追跡スケーリングポリシーの使用時における制限事項です。

- スケーラブルターゲットを Amazon EMR クラスターにすることはできません。Amazon EMR はターゲット追跡スケーリングポリシーをサポートしません。
- Amazon MSK クラスターがスケーラブルターゲットである場合は、スケールインが無効化されており、有効にすることはできません。
- `RegisterScalableTarget` または `PutScalingPolicy` API オペレーションを使用して、AWS Auto Scaling スケーリングプランを更新することはできません。スケーリングプランの使用については、AWS Auto Scaling ドキュメントを参照してください。

AWS CLI を使用したターゲット追跡スケーリングポリシーの作成

Application Auto Scaling のターゲット追跡スケーリングポリシーは、以下の設定タスクに AWS CLI を使用して作成することができます。

[Tasks] (タスク)

- スケーラブルターゲットを登録する (p. 50)
- ターゲット追跡スケーリングポリシーを作成する (p. 51)

スケーラブルターゲットを登録する

まだ登録していない場合は、スケーラブルターゲットを登録します。register-scalable-target コマンドを使用して、ターゲットサービス内の特定のリソースをスケーラブルターゲットとして登録します。以下の例は、スポットフリートリクエストを Application Auto Scaling に登録します。Application Auto Scaling は、スポットフリート内のインスタンスの数を最小 2 インスタンス、および最大 10 インスタンスにスケールできます。

Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ec2 --scalable-
dimension ec2:spot-fleet-request:TargetCapacity --resource-id spot-fleet-request/
sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --min-capacity 2 --max-capacity 10
```

Note

簡略化のため、このトピックの例では、Amazon EC2 スポットフリート用の CLI コマンドを例示しています。別のスケーラブルターゲットを指定するには、`--service-namespace` でその名前空間、`--scalable-dimension` でそのスケーラブルディメンション、`--resource-id` でそのリソース ID を指定します。

ターゲット追跡スケーリングポリシーを作成する

例: ターゲットの追跡設定ファイル

平均 CPU 使用率を 40 パーセントに維持するターゲット追跡設定の例を次に示します。この設定を `config.json` という名前のファイルに保存してください。

```
{
  "TargetValue": 40.0,
  "PredefinedMetricSpecification":
    {
      "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
    }
}
```

詳細については、Application Auto Scaling API リファレンスの「PredefinedMetricSpecification」を参照してください。

または、CloudWatch でカスタマイズされたメトリクス仕様を作成し、各パラメータの値を追加することによって、スケーリング用のカスタムメトリクスを使用することもできます。指定されたメトリクスの平均使用率を 40 パーセントに維持するターゲット追跡設定の例を以下に示します。

```
{
   "TargetValue":40.0,
   "CustomizedMetricSpecification":{
      "MetricName":"MyUtilizationMetric",
      "Namespace":"MyNamespace",
      "Dimensions":[
         {
            "Name":"MyOptionalMetricDimensionName",
            "Value":"MyOptionalMetricDimensionValue"
         }
      ],
      "Statistic":"Average",
      "Unit":"Percent"
   }
}
```

詳細については、Application Auto Scaling API リファレンスの「CustomizedMetricSpecification」を参照してください。

例: cpu40-target-tracking-scaling-policy

作成した `config.json` ファイルと共に以下の put-scaling-policy コマンドを使用して、`cpu40-target-tracking-scaling-policy` という名前のスケーリングポリシーを作成します。

Linux、macOS、または Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu40-target-tracking-scaling-policy --policy-type TargetTrackingScaling \
```

```
  --target-tracking-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ec2 --scalable-
dimension ec2:spot-fleet-request:TargetCapacity --resource-id spot-fleet-request/
sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --policy-name cpu40-target-tracking-scaling-policy
 --policy-type TargetTrackingScaling --target-tracking-scaling-policy-configuration file://
config.json
```

正常に完了した場合、このコマンドは、ユーザーに代わって作成された 2 つの CloudWatch アラームの ARN と名前を返します。

```
{
    "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:policy-id:resource/
ec2/spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE:policyName/cpu40-target-
tracking-scaling-policy",
    "Alarms": [
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-spot-
fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-
a60f-3b36d653feca",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
        },
        {
            "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-
a812-6c67aaf2910d",
            "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
        }
    ]
}
```

ターゲット追跡スケーリングポリシーを記述する

以下の describe-scaling-policies コマンドを使用して、指定したサービス名前空間に対するすべてのスケーリングポリシーを記述することができます。

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2
```

結果をフィルタリングし、--query パラメータを使用してターゲット追跡スケーリングポリシーのみに制限することができます。query 用の構文の詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLI からのコマンド出力の制御」を参照してください。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 \
  --query 'ScalingPolicies[?PolicyType==`TargetTrackingScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ec2 --query
 "ScalingPolicies[?PolicyType==`TargetTrackingScaling`]"
```

以下は出力例です。

```
[
    {
        "PolicyARN": "PolicyARN",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "EC2SpotFleetRequestAverageCPUUtilization"
            },
            "TargetValue": 40.0
        },
        "PolicyName": "cpu40-target-tracking-scaling-policy",
        "ScalableDimension": "ec2:spot-fleet-request:TargetCapacity",
        "ServiceNamespace": "ec2",
        "PolicyType": "TargetTrackingScaling",
        "ResourceId": "spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE",
        "Alarms": [
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-
a60f-3b36d653feca",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmHigh-d4f0770c-b46e-434a-a60f-3b36d653feca"
            },
            {
                "AlarmARN": "arn:aws:cloudwatch:region:account-id:alarm:TargetTracking-
spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-
a812-6c67aaf2910d",
                "AlarmName": "TargetTracking-spot-fleet-request/sfr-73fbd2ce-
aa30-494c-8788-1cee4EXAMPLE-AlarmLow-1b437334-d19b-4a63-a812-6c67aaf2910d"
            }
        ],
        "CreationTime": 1515021724.807
    }
]
```

ターゲット追跡スケーリングポリシーを削除する

ターゲット追跡スケーリングポリシーが不要になったら、delete-scaling-policy コマンドを使用してポリシーを削除することができます。

次のコマンドは、指定したスポットフリートリクエストに対して指定したターゲット追跡スケーリングポリシーを削除します。これは、Application Auto Scaling がユーザーに代わって作成した CloudWatch アラームも削除します。

Linux、macOS、または Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 \
  --scalable-dimension ec2:spot-fleet-request:TargetCapacity \
  --resource-id spot-fleet-request/sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE \
  --policy-name cpu40-target-tracking-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ec2 --scalable-
dimension ec2:spot-fleet-request:TargetCapacity --resource-id spot-fleet-request/
sfr-73fbd2ce-aa30-494c-8788-1cee4EXAMPLE --policy-name cpu40-target-tracking-scaling-policy
```

Application Auto Scaling のステップスケーリングポリシー

ステップスケーリングでは、ユーザーがスケーリングメトリクス、およびスケーリングプロセスをトリガーする CloudWatch アラームのしきい値を選択するとともに、指定された数の評価期間にわたってしきい値違反が発生している場合にスケーラブルターゲットをどのようにスケールするかを定義します。

スケーリングメトリクスが、スケーラブルなターゲットのキャパシティに比例してスケールする使用率メトリクスである場合は、ターゲット追跡スケーリングポリシーを使用することをお勧めします。詳細については、「Application Auto Scaling のターゲット追跡スケーリングポリシー (p. 46)」を参照してください。より高度なスケーリングポリシーの設定には、ステップスケーリングによりターゲット追跡スケーリングを使用するオプションが引き続き用意されています。たとえば、必要に応じて、使用率が一定のレベルに達したときにより積極的な応答を設定できます。

ステップスケーリングポリシーは、一連のスケーリング調整値 (ステップ調整値) に基づいて、スケーラブルなターゲットの現在の容量を増減させます。調整値は、アラーム違反の大きさに応じて異なります。すべてのアラーム違反は、Application Auto Scaling がアラームメッセージを受け取るときに評価されます。

ステップ調整値

ステップスケーリングポリシーを作成するときは、アラーム超過のサイズに基づいてスケールできるようにする 1 つ以上のステップ調整値を追加します。各ステップ調整値は、次のように指定します。

- メトリクス値の下限
- メトリクス値の上限
- スケーリング調整タイプに基づいてスケールする量

CloudWatch は、CloudWatch アラームに関連付けられたメトリクスの統計に基づいて、メトリクスデータポイントを集計します。アラーム違反が発生すると、適切なスケーリングポリシーがトリガーされます。Application Auto Scaling は、raw メトリクスデータではなく、CloudWatch からの最新のメトリクスデータポイントに指定された集計タイプを適用します。ステップ調整によって定義された上限と下限に対して、この集約メトリクス値を比較することにより、実行するステップ調整が決定されます。

違反しきい値に比例して上限と下限を指定します。たとえば、現在の容量と希望する容量が 10 のスケーラブルなターゲットがあるとします。違反しきい値が 50% の CloudWatch アラームがあります。`PercentChangeInCapacity` の調整タイプがあり、スケールアウトおよびスケールインポリシーに以下のステップ調整値が含まれるとします。

例: スケールアウトポリシーのステップ調整値

下限	上限	調整
0	10	0
10	20	10
20	null	30

例: スケールインポリシーのステップ調整値

下限	上限	調整
-10	0	0
-20	-10	-10
null	-20	-30

これにより、次のスケーリング設定が作成されます。

```
Metric value

-infinity           30%     40%           60%      70%              infinity
-------------------------------------------------------------------------
          -30%       | -10% | Unchanged   | +10%   |        +30%
-------------------------------------------------------------------------
```

以下の点は、スケーラブルターゲットの希望容量と現在の容量に関連してスケーリング設定の動作をまとめたものです。

- 集合メトリクス値が 40 より大きく 60 未満である間は、現在の容量と必要な容量が維持されます。
- メトリクス値が 60 に到達すると、Application Auto Scaling はスケーラブルターゲットの希望容量に 1 を足して 11 にします。これはスケールアウトポリシーの 2 番目のステップ調整値に基づきます (10 の 10% を追加)。新しい容量が追加されると、Application Auto Scaling は現行の容量を 11 に増やします。この容量の増加後にメトリクス値が 70 に上昇すると、Application Auto Scaling はターゲット容量に 3 を足して 14 にします。これはスケールアウトポリシーの 3 番目のステップ調整値に基づきます (11 の 30% である 3.3 を、3 に切り捨てて追加)。
- メトリクス値が 40 になると、Application Auto Scaling はスケールインポリシーの 2 番目のステップ調整値 (14 の 10%、つまり 1.4 を四捨五入した 1 を削除) に基づき、ターゲット容量から 1 を引いて 13 にします。この容量の減少後にメトリクス値がさらに 30 まで減った場合、Application Auto Scaling はスケールインポリシーの 3 番目のステップ調整 (13 の 30%、つまり 3.9 を四捨五入した 3 を削除) に基づき、ターゲット容量から 3 を引いて 10 にします。

スケーリングポリシーのステップ調整を指定するときは、次の点に注意してください。

- ステップ調整値の範囲に重複や間隔があってはなりません。
- 1 つのステップ調整値のみ、下限を null (負の無限大) にすることができます。下限が負のステップ調整値がある場合は、下限が null のステップ調整値が必要です。
- 1 つのステップ調整値のみ、上限を null (正の無限大) にすることができます。上限が正のステップ調整値がある場合は、上限が null のステップ調整値が必要です。
- 同じステップ調整値で上限と下限を null にすることはできません。
- メトリクス値が超過しきい値を上回っている場合、下限にその値を含み、上限には含みません。メトリクス値が超過しきい値を下回っている場合、下限にその値を含まず、上限に含みます。

スケーリング調整タイプ

選択したスケーリング調整タイプに基づいて、最適なスケーリングアクションを実行するスケーリングポリシーを定義できます。調整タイプは、スケーラブルターゲットの現在の容量に対する割合、または絶対数で指定できます。

Application Auto Scaling は、ステップスケーリングポリシーに対して以下の調整タイプをサポートします。

- ChangeInCapacity – スケーラブルターゲットの現行容量を、指定された値に基づいて増減させます。正の値はキャパシティーを増やし、負の値はキャパシティーを減らします。例えば、現行容量が 3 で調整値が 5 の場合、Application Auto Scaling は容量に 5 を追加して合計を 8 にします。
- ExactCapacity – スケーラブルターゲットの現行容量を、指定された値に変更します。この調整タイプには正の値を指定します。例えば、現行容量が 3 で調整値が 5 の場合、Application Auto Scaling は容量を 5 に変更します。
- PercentChangeInCapacity – スケーラブルターゲットの現行容量を、指定された割合 (%) に基づいて増減させます。正の値はキャパシティーを増やし、負の値はキャパシティーを減らします。例えば、現行容量が 10 で調整値が 10 パーセントの場合、Application Auto Scaling は容量に 1 を追加して合計を 11 にします。

  Note

  調整後の値が整数ではない場合、Application Auto Scaling はその値を以下のように四捨五入します。

  - 1 より大きい値は小数点以下が切り捨てられます。たとえば、`12.7` は `12` に丸められます。
  - 0 と 1 の間の値は 1 に丸められます。たとえば、`.67` は `1` に丸められます。
  - 0 と -1 の間の値は -1 に丸められます。たとえば、`-.58` は `-1` に丸められます。
  - -1 未満の値は小数点以下が切り捨てられます。たとえば、`-6.67` は `-6` に丸められます。

  PercentChangeInCapacity では、`MinAdjustmentMagnitude` パラメータを使用してスケーリングする最小の数量を指定できます。たとえば、25% 追加するポリシーを作成して、最小数量を 2 に指定するとします。スケーラブルなターゲットの容量が 4 の時にスケーリングポリシーを実行すると、4 の 25% は 1 です。しかし、最小増分が 2 に指定されていることから、Application Auto Scaling は 2 を追加します。

クールダウン期間

前回のスケーリングアクティビティが有効になるまで待機する時間をクールダウン期間と呼びます。

- スケールアウトポリシーでは、スケールアウトが継続的に (ただし過剰になることなく) 行われます。ステップスケーリングスケーリングポリシーを使用して Application Auto Scaling が正常にスケールアウトすると、クールダウン時間の計算が開始されます。スケーリングポリシーは、より大きなスケールアウトがトリガーされるか、クールダウン期間が終了しない限り、必要な容量を再度増加させません。このクールダウン期間が有効な間は、スケールアウトアクティビティを開始することで追加された容量は、次のスケールアウトアクティビティに予定される容量の一部として繰り入れられます。たとえば、あるアラームが容量を 2 増加させるステップスケーリングポリシーをトリガーする場合、スケーリングアクティビティは正常に完了し、クールダウン期間が始まります。クールダウン期間中にアラームが再度トリガーし、さらに進んだステップ調整を行う場合 (3 の増加)、以前の 2 の増加は現在の容量の一部とみなされます。したがって、容量に追加されるのは 1 だけです。
- スケールインポリシーでは、スケールインを控え目に行ってアプリケーションの可用性を保護することを目的としているため、スケールインアクティビティはクールダウン期間が終了するまでブロックされます。ただし、スケールインアクティビティの後、クールダウン期間中に別のアラームがスケールアウトアクティビティをトリガーした場合、Application Auto Scaling scale によってターゲットが即座にスケールアウトされます。この場合、スケールインアクティビティのクールダウン期間は停止し、完了しません。

クールダウン期間は秒単位で測定され、スケーリングポリシー関連のスケーリングアクティビティにのみ適用されます。クールダウン期間中、スケジュールされたアクションがスケジュールされた時間に開始されると、クールダウン期間の期限が切れるのを待たずにスケーリングアクティビティを即座にトリガーできます。

スケーリングポリシーの作成、管理、および削除用によく使用されるコマンド

スケーリングポリシーの操作用によく使用されるコマンドには以下が含まれます。

- register-scalable-target AWS リソースまたはカスタムリソースをスケーラブルターゲット (Application Auto Scaling がスケールできるリソース) として登録し、スケーリングを一時停止および再開します。
- put-scaling-policy 既存のスケーラブルターゲットのスケーリングポリシーを追加または変更します。
- describe-scaling-activities AWS リージョン内でのスケーリングアクティビティに関する情報を返します。
- describe-scaling-policies AWS リージョン内のスケーリングポリシーに関する情報を返します。
- delete-scaling-policy スケーリングポリシーを削除します。

機能制限

以下は、ステップスケーリングポリシーの使用時における制限事項です。

- 特定のサービスにはステップスケーリングポリシーを作成できません。ステップスケーリングポリシーは、DynamoDB、Amazon Comprehend、Lambda、Amazon Keyspaces、Amazon MSK、ElastiCache、または Neptune 向けにサポートされていません。

AWS CLI を使用してステップスケーリングポリシーを作成する

Application Auto Scaling のステップスケーリングポリシーは、以下の設定タスクに AWS CLI を使用して作成することができます。

[Tasks] (タスク)

- スケーラブルターゲットを登録する (p. 57)
- ステップスケーリングポリシーを作成する (p. 58)
- スケーリングポリシーをトリガーするアラームを作成する (p. 59)

スケーラブルターゲットを登録する

まだ登録していない場合は、スケーラブルターゲットを登録します。register-scalable-target コマンドを使用して、ターゲットサービス内の特定のリソースをスケーラブルターゲットとして登録します。以下の例は、Amazon ECS サービスを Application Auto Scaling に登録します。Application Auto Scaling は、タスクの数を最小 2 タスク、および最大 10 タスクにスケールできます。

Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --min-capacity 2 --max-capacity 10
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace ecs --scalable-
dimension ecs:service:DesiredCount --resource-id service/default/sample-app-service --min-
capacity 2 --max-capacity 10
```

Note

簡略化のため、このトピックの例では、Amazon ECS サービス用の CLI コマンドを例示しています。別のスケーラブルターゲットを指定するには、`--service-namespace` でその名前空間、`--scalable-dimension` でそのスケーラブルディメンション、`--resource-id` でそのリソース ID を指定します。

ステップスケーリングポリシーを作成する

以下は、調整タイプが `ChangeInCapacity` のサンプルステップ設定で、以下のステップ調整値に基づいてスケーラブルターゲットの容量を増加させます (CloudWatch アラームしきい値を 70 パーセントとした場合)。

- メトリクスの値が 70 パーセント以上、85 パーセント未満の場合は容量を 1 増やします。
- メトリクスの値が 85 パーセント以上、95 パーセント未満の場合は容量を 2 増やします。
- メトリクスの値が 95 パーセント以上の場合は容量を 3 増やします。

この設定を `config.json` という名前のファイルに保存してください。

```
{
  "AdjustmentType": "ChangeInCapacity",
  "MetricAggregationType": "Average",
  "Cooldown": 60,
  "StepAdjustments": [
    {
      "MetricIntervalLowerBound": 0,
      "MetricIntervalUpperBound": 15,
      "ScalingAdjustment": 1
    },
    {
      "MetricIntervalLowerBound": 15,
      "MetricIntervalUpperBound": 25,
      "ScalingAdjustment": 2
    },
    {
      "MetricIntervalLowerBound": 25,
      "ScalingAdjustment": 3
    }
  ]
}
```

作成した `config.json` ファイルと共に以下の put-scaling-policy コマンドを使用して、`my-step-scaling-policy` という名前のスケーリングポリシーを作成します。

Linux、macOS、または Unix

```
aws application-autoscaling put-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --policy-name my-step-scaling-policy --policy-type StepScaling \
  --step-scaling-policy-configuration file://config.json
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace ecs --scalable-
dimension ecs:service:DesiredCount --resource-id service/default/sample-app-service
 --policy-name my-step-scaling-policy --policy-type StepScaling --step-scaling-policy-
configuration file://config.json
```

出力には、ポリシーの一意の名前となる ARN が含まれます。これは、CloudWatch アラームを作成するために必要です。

```
{
    "PolicyARN": "arn:aws:autoscaling:region:123456789012:scalingPolicy:ac542982-
cbeb-4294-891c-a5a941dfa787:resource/ecs/service/default/sample-app-service:policyName/my-
step-scaling-policy"
}
```

スケーリングポリシーをトリガーするアラームを作成する

最後に、以下の CloudWatch put-metric-alarm コマンドを使用して、ステップスケーリングポリシーで使用するアラームを作成します。この例では、CPU の平均利用率に基づくアラームもあります。アラームは、少なくとも 2 つの連続する 60 秒の評価期間に 70 パーセントのしきい値に達した場合に、ALARM 状態となるよう設定されます。別の CloudWatch メトリクスを指定する、または独自のカスタムメトリクスを使用するには、`--metric-name` でその名前を指定し、`--namespace` でその名前空間を指定します。

Linux、macOS、または Unix

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/default/
sample-app-service \
  --metric-name CPUUtilization --namespace AWS/ECS --statistic Average \
  --period 60 --evaluation-periods 2 --threshold 70 \
  --comparison-operator GreaterThanOrEqualToThreshold \
  --dimensions Name=ClusterName,Value=default Name=ServiceName,Value=sample-app-service \
  --alarm-actions PolicyARN
```

Windows

```
aws cloudwatch put-metric-alarm --alarm-name Step-Scaling-AlarmHigh-ECS:service/
default/sample-app-service --metric-name CPUUtilization --namespace AWS/ECS --
statistic Average --period 60 --evaluation-periods 2 --threshold 70 --comparison-
operator GreaterThanOrEqualToThreshold --dimensions Name=ClusterName,Value=default
 Name=ServiceName,Value=sample-app-service --alarm-actions PolicyARN
```

ステップスケーリングポリシーを記述する

以下の describe-scaling-policies コマンドを使用して、指定したサービス名前空間に対するすべてのスケーリングポリシーを記述することができます。

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs
```

`--query` パラメータを使用して、結果をステップスケーリングポリシーのみにフィルタリングすることができます。`query` 用の構文の詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLI からのコマンド出力の制御」を参照してください。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs \
  --query 'ScalingPolicies[?PolicyType==`StepScaling`]'
```

Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace ecs --query
 "ScalingPolicies[?PolicyType==`StepScaling`]"
```

以下は出力例です。

```
[
    {
        "PolicyARN": "PolicyARN",
        "StepScalingPolicyConfiguration": {
            "MetricAggregationType": "Average",
            "Cooldown": 60,
            "StepAdjustments": [
                {
                    "MetricIntervalLowerBound": 0.0,
                    "MetricIntervalUpperBound": 15.0,
                    "ScalingAdjustment": 1
                },
                {
                    "MetricIntervalLowerBound": 15.0,
                    "MetricIntervalUpperBound": 25.0,
                    "ScalingAdjustment": 2
                },
                {
                    "MetricIntervalLowerBound": 25.0,
                    "ScalingAdjustment": 3
                }
            ],
            "AdjustmentType": "ChangeInCapacity"
        },
        "PolicyType": "StepScaling",
        "ResourceId": "service/default/sample-app-service",
        "ServiceNamespace": "ecs",
        "Alarms": [
            {
                "AlarmName": "Step-Scaling-AlarmHigh-ECS:service/default/sample-app-
service",
                "AlarmARN": "arn:aws:cloudwatch:region:012345678910:alarm:Step-Scaling-
AlarmHigh-ECS:service/default/sample-app-service"
            }
        ],
        "PolicyName": "my-step-scaling-policy",
        "ScalableDimension": "ecs:service:DesiredCount",
        "CreationTime": 1515024099.901
    }
]
```

ステップスケーリングポリシーを削除する

不要になったステップのスケーリングポリシーは削除できます。スケーリングポリシーと CloudWatch アラームの両方を削除するには、以下のタスクを完了します。

スケーリングポリシーを削除する

以下の delete-scaling-policy コマンドを使用します。

Linux、macOS、または Unix

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs \
  --scalable-dimension ecs:service:DesiredCount \
  --resource-id service/default/sample-app-service \
  --policy-name my-step-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace ecs --scalable-
dimension ecs:service:DesiredCount --resource-id service/default/sample-app-service --
policy-name my-step-scaling-policy
```

CloudWatch アラームを削除する

delete-alarms コマンドを使用します。1 つ以上のアラームを一度に削除することができます。たとえば、次のコマンドを使用して Step-Scaling-AlarmHigh-ECS:service/default/sample-app-service アラームおよび Step-Scaling-AlarmLow-ECS:service/default/sample-app-service アラームを削除します。

```
aws cloudwatch delete-alarms --alarm-name Step-Scaling-AlarmHigh-ECS:service/default/
sample-app-service Step-Scaling-AlarmLow-ECS:service/default/sample-app-service
```

チュートリアル: 大量のワークロードを処理するためのオートスケーリングの設定

Important

このチュートリアルに進む前に、以下の概要チュートリアルを確認することをお勧めします。チュートリアル:AWS CLI を使用したスケジュールに基づくスケーリングの開始方法 (p. 38)

このチュートリアルでは、アプリケーションのワークロードが通常よりも多くなる時間枠に基づいてスケールアウトおよびスケールインする方法を学びます。これは、定期的に、または季節に応じて訪問者の数が急増する可能性があるアプリケーションが存在する場合に役立ちます。

追加の負荷を処理するには、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを併用できます。スケジュールされたスケーリングは、ユーザー指定のスケジュールに基づいて、`MinCapacity` および `MaxCapacity` への変更をユーザーに代って自動的に開始します。ターゲット追跡スケーリングポリシーがリソースでアクティブになっていると、新しい最小容量と最大容量の範囲内で、現行のリソース使用率に基づいて動的にスケールすることができます。

このチュートリアルを完了すると、以下を行う方法を理解できます。

- スケジュールされたスケーリングを使用して、高負荷状態になる前にそれらに対応するための容量を追加し、容量が必要なくなったときに削除する。
- ターゲット追跡スケーリングポリシーを使用して、現行のリソース使用率に基づいてアプリケーションをスケールする。

目次

前提条件

このチュートリアルでは、以下を実行済みであることを前提としています。

- AWS アカウントが作成されている。詳細については、「AWS アカウントにサインアップする (p. 3)」を参照してください。
- AWS CLI がインストールされ、設定されている。詳細については、「AWS CLI をセットアップする (p. 3)」を参照してください。
- アカウントに、スケーラブルターゲットとしてリソースを Application Auto Scaling に登録または登録解除するために必要な許可がある。また、スケーリングポリシーとスケジュールされたアクションを作成するために必要なすべての許可もそろっている。詳細については、「Application Auto Scaling の Identity and Access Management (p. 85)」を参照してください。
- 非実稼働環境にこのチュートリアルで使用するためのサポート対象リソースがある。まだ作成していない場合は、新しく作成してください。Application Auto Scaling と連携する AWS サービスとリソースの

詳細については、Application Auto Scaling を使用できる AWS のサービス (p. 7) セクションを参照してください。

Note

このチュートリアルの実行中、 リソースの最小容量と最大容量の値を 0 に設定して、現在の容量を 0 にリセットする 2 つのステップがあります。Application Auto Scaling で使用しているリソースによっては、これらの手順で現在の容量を 0 にリセットできない場合があります。この問題に対処できるように、出力内のメッセージによって最小容量を指定された値未満にすることはできないことが指摘され、AWS リソースによる受け入れが可能な最小容量値が提供されます。

ステップ 1: スケーラブルターゲットを登録する

スケーラブルターゲットとしてリソースを Application Auto Scaling に登録することから始めます。スケーラブルターゲットとは、Application Auto Scaling がスケールアウトおよびスケールインできるリソースです。

Application Auto Scaling にスケーラブルなターゲットを登録する

- 以下の register-scalable-target コマンドを使用して、新しいスケーラブルターゲットを登録します。`--min-capacity` および `--max-capacity` の値を 0 に設定して、現行容量を 0 にリセットします。

  `--service-namespace` のサンプルテキストを、Application Auto Scaling で使用している AWS サービスの名前空間、`--scalable-dimension` を登録しているリソースに関連付けられているスケーラブルディメンション、`--resource-id` をリソースの識別子に置き換えます。これらの値は、使用されるリソースとリソース ID の構築方法によって異なります。詳細については、 Application Auto Scaling を使用できる AWS のサービス (p. 7) セクションのトピックを参照してください。これらのトピックには、スケーラブルなターゲットをApplication Auto Scaling に登録する方法を示すコマンド例が含まれています。

  Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --min-capacity 0 --max-capacity 0
```

  Windows

```
aws application-autoscaling register-scalable-target --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --min-capacity 0 --max-capacity 0
```

  このコマンドが正常に完了した場合は、出力が返されません。

ステップ 2: 要件に従ってスケジュールされたアクションをセットアップする

put-scheduled-action コマンドを使用して、ビジネスニーズを満たすように設定されたスケジュールされたアクションを作成することができます。このチュートリアルでは、容量を 0 に減らすことによって、就業時間外におけるリソースの消費を停止する設定に焦点を当てます。

午前中にスケールアウトするスケジュールされたアクションを作成する

1. スケーラブルターゲットをスケールアウトするには、以下の put-scheduled-action コマンドを使用します。Cron 式を使用して、UTC での定期的なスケジュールが設定された `--schedule` パラメータを含めます。

   Application Auto Scaling は、指定されたスケジュール (毎日午前 9:00 (UTC)) に従って、`MinCapacity` および `MaxCapacity` の値を希望範囲の 1～5 キャパシティーユニットに更新します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
     --scheduled-action-name my-first-scheduled-action \
     --schedule "cron(0 9 * * ? *)" \
     --scalable-target-action MinCapacity=1,MaxCapacity=5
   ```

   Windows

   ```
   aws application-autoscaling put-scheduled-action --service-namespace namespace --
   scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
   first-scheduled-action --schedule "cron(0 9 * * ? *)" --scalable-target-action
    MinCapacity=1,MaxCapacity=5
   ```

   このコマンドが正常に完了した場合は、出力が返されません。

2. スケジュールされたアクションが存在することを確認するには、以下の describe-scheduled-actions コマンドを使用します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace namespace \
     --query 'ScheduledActions[?ResourceId==`identifier`]'
   ```

   Windows

   ```
   aws application-autoscaling describe-scheduled-actions --service-namespace namespace --
   query "ScheduledActions[?ResourceId==`identifier`]"
   ```

   以下は出力例です。

   ```
   [
       {
           "ScheduledActionName": "my-first-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 9 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 1,
               "MaxCapacity": 5
           },
           ...
       }
   ]
   ```

夜間にスケールインするスケジュールされたアクションを作成する

1. 上記の手順を繰り返して、Application Auto Scaling が 1 日の終わりにスケールインするために使用する、別のスケジュールされたアクションを作成します。

   Application Auto Scaling は、以下の put-scheduled-action コマンドの指示通りに、指定されたスケジュール (毎日午後 8:00 (UTC)) に従ってターゲットの `MinCapacity` および `MaxCapacity` を 0 に更新します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling put-scheduled-action \
     --service-namespace namespace \
     --scalable-dimension dimension \
     --resource-id identifier \
     --scheduled-action-name my-second-scheduled-action \
     --schedule "cron(0 20 * * ? *)" \
     --scalable-target-action MinCapacity=0,MaxCapacity=0
   ```

   Windows

   ```
   aws application-autoscaling put-scheduled-action --service-namespace namespace --
   scalable-dimension dimension --resource-id identifier --scheduled-action-name my-
   second-scheduled-action --schedule "cron(0 20 * * ? *)" --scalable-target-action
    MinCapacity=0,MaxCapacity=0
   ```

2. スケジュールされたアクションが存在することを確認するには、以下の describe-scheduled-actions コマンドを使用します。

   Linux、macOS、または Unix

   ```
   aws application-autoscaling describe-scheduled-actions \
     --service-namespace namespace \
     --query 'ScheduledActions[?ResourceId==`identifier`]'
   ```

   Windows

   ```
   aws application-autoscaling describe-scheduled-actions --service-namespace namespace --
   query "ScheduledActions[?ResourceId==`identifier`]"
   ```

   以下は出力例です。

   ```
   [
       {
           "ScheduledActionName": "my-first-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 9 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 1,
               "MaxCapacity": 5
           },
           ...
       },
       {
           "ScheduledActionName": "my-second-scheduled-action",
           "ScheduledActionARN": "arn",
           "Schedule": "cron(0 20 * * ? *)",
           "ScalableTargetAction": {
               "MinCapacity": 0,
   ```

```
            "MaxCapacity": 0
        },
        ...
    }
]
```

ステップ 3: ターゲット追跡スケーリングポリシーを追加する

基本的なスケジュールが設定されたところで、現行のリソース使用率に基づいてスケールするためのターゲット追跡スケーリングポリシーを追加します。

ターゲット追跡では、Application Auto Scaling がポリシーのターゲット値を指定されたメトリクスの現行値と比較します。それらが一定期間同等でなかった場合は、Application Auto Scaling が容量を追加または削除して、安定したパフォーマンスを維持します。アプリケーションに対する負荷とメトリクス値の増加に伴い、Application Auto Scaling は、`MaxCapacity` を超過することなく、可能な限り早急に容量を追加します。負荷が最小限であることを理由に Application Auto Scaling が容量を削除するときは、`MinCapacity` を下回らないように削除します。使用量に基づいて容量を調整することで、料金の支払いがアプリケーションに必要な容量分のみになります。詳細については、「使用率の高い期間中のアプリケーションの可用性のサポート (p. 49)」を参照してください。

アプリケーションに負荷がないことが原因でメトリクスに十分なデータがない場合、Application Auto Scaling は容量の追加または削除を行いません。この動作は、十分な情報が利用できない状況で可用性を優先することを目的とするものです。

スケーリングポリシーは複数追加できますが、競合するステップスケーリングポリシーは追加しないようにしてください。これらは望ましくない動作の原因となる可能性があります。たとえば、ターゲット追跡ポリシーがスケールインする準備が整う前に、ステップスケーリングポリシーがスケールインアクティビティを開始した場合、スケールインアクティビティはブロックされません。ターゲット追跡ポリシーは、スケールインアクティビティの完了後、再度スケールアウトするように Application Auto Scaling に指示できます。

ターゲット追跡スケーリングポリシーを作成する

1. 以下の put-scaling-p コマンドを使用して、ポリシーを作成します。

   ターゲット追跡のために最も頻繁に使用されるメトリクスは事前定義されており、これらは CloudWatch から完全なメトリクス仕様を提供しなくても使用できます。利用可能な事前定義されたメトリクスの詳細については、「Application Auto Scaling のターゲット追跡スケーリングポリシー (p. 46)」を参照してください。。

   このコマンドを実行する前に、事前定義されたメトリクスがターゲット値を期待していることを確認してください。例えば、CPU 使用率が 50% に達したときにスケールアウトするには、50.0 のターゲット値を指定します。または、使用量が 70% に達したときに Lambda のプロビジョニングされた同時実行数をスケールアウトするには、0.7 のターゲット値を指定します。特定のリソースのターゲット値に関する情報は、ターゲット追跡の設定方法について、サービス提供のドキュメントを参照してください。詳細については、「Application Auto Scaling を使用できる AWS のサービス (p. 7)」を参照してください。

   Linux、macOS、または Unix

```
aws application-autoscaling put-scaling-policy \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy --policy-type TargetTrackingScaling \
```

```
  --target-tracking-scaling-policy-configuration '{ "TargetValue": 50.0,
 "PredefinedMetricSpecification": { "PredefinedMetricType": "predefinedmetric" }}'
```

Windows

```
aws application-autoscaling put-scaling-policy --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --policy-name my-scaling-policy
 --policy-type TargetTrackingScaling --target-tracking-scaling-policy-configuration
 "{ \"TargetValue\": 50.0, \"PredefinedMetricSpecification\": { \"PredefinedMetricType
\": \"predefinedmetric\" }}"
```

正常に完了した場合、このコマンドは、ユーザーに代わって作成された 2 つの CloudWatch アラームの ARN と名前を返します。

2. スケジュールされたアクションが存在することを確認するには、以下の describe-scaling-policies コマンドを使用します。

   Linux、macOS、または Unix

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace \
  --query 'ScalingPolicies[?ResourceId==`identifier`]'
```

   Windows

```
aws application-autoscaling describe-scaling-policies --service-namespace namespace --
query "ScalingPolicies[?ResourceId==`identifier`]"
```

   以下は出力例です。

```
[
    {
        "PolicyARN": "arn",
        "TargetTrackingScalingPolicyConfiguration": {
            "PredefinedMetricSpecification": {
                "PredefinedMetricType": "predefinedmetric"
            },
            "TargetValue": 50.0
        },
        "PolicyName": "my-scaling-policy",
        "PolicyType": "TargetTrackingScaling",
        "Alarms": [],
        ...
    }
]
```

ステップ 4: 次のステップ

スケーリングアクティビティが発生すると、スケーラブルターゲットのスケーリングアクティビティの出力にそのレコードが表示されます。次に例を示します。

```
Successfully set desired count to 1. Change successfully fulfilled by ecs.
```

Application Auto Scaling を使用してスケーリングアクティビティを監視するには、describe-scaling-activities コマンドを使用できます。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scaling-activities
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling describe-scaling-activities --service-namespace namespace --
scalable-dimension dimension --resource-id identifier
```

ステップ 5：クリーンアップ

アカウントでアクティブにスケーリングしている最中に作成されたリソースに対する料金が発生しないようにするために、関連付けられたスケーリング設定を以下のようにクリーンアップすることができます。

スケーリング設定を削除しても、基盤となる AWS リソースは削除されません。また、リソースが元の容量に戻されることもありません。リソースの削除、またはその容量の調整は、そのリソースを作成したサービスのコンソールを使用して行うことができます。

スケジュールされたアクションを削除する

以下の delete-scheduled-action コマンドは、指定されているスケールされたアクションを削除します。作成したスケジュールされたアクションを保持したい場合は、このステップをスキップできます。

Linux、macOS、または Unix

```
aws application-autoscaling delete-scheduled-action \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --scheduled-action-name my-second-scheduled-action
```

Windows

```
aws application-autoscaling delete-scheduled-action --service-namespace namespace --
scalable-dimension dimension --resource-id identifier --scheduled-action-name my-second-
scheduled-action
```

スケーリングポリシーを削除する

以下の delete-scheduled-action コマンドは、指定されたターゲット追跡スケーリングポリシーを削除します。作成したスケーリングポリシーを保持したい場合は、このステップをスキップできます。

Linux、macOS、または Unix

```
aws application-autoscaling delete-scaling-policy \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier \
  --policy-name my-scaling-policy
```

Windows

```
aws application-autoscaling delete-scaling-policy --service-namespace namespace --scalable-
dimension dimension --resource-id identifier --policy-name my-scaling-policy
```

スケーラブルなターゲットを登録解除する

以下の deregister-scalable-target コマンドを使用して、スケーラブルターゲットの登録を解除します。自分で作成したスケーリングポリシーや、まだ削除されていないスケジュールされたアクションがある場合は、このコマンドによって削除されます。後で使用できるように、登録されたスケーラブルなターゲットを保持する場合は、このステップをスキップできます。

Linux、macOS、または Unix

```
aws application-autoscaling deregister-scalable-target \
  --service-namespace namespace \
  --scalable-dimension dimension \
  --resource-id identifier
```

Windows

```
aws application-autoscaling deregister-scalable-target --service-namespace namespace --
scalable-dimension dimension --resource-id identifier
```

Application Auto Scaling のスケーリングの一時停止と再開

このトピックでは、アプリケーションでスケーラブルターゲットのスケーリングアクティビティの 1 つ、または複数を一時停止し、その後再開する方法について説明します。一時停止/再開機能は、スケーリングポリシーとスケジュールされたアクションによってトリガーされたスケーリングアクティビティを一時的に停止するために使用されます。これは、たとえば、変更を行っている間や設定の問題を調査しているときに、自動スケーリングに干渉されないようにする場合などに便利です。スケーリングポリシーとスケジュールされたアクションは保持し、準備が整ったら、スケーリングアクティビティを再開できます。

以下のサンプル CLI コマンドでは、config.json ファイル で JSON 形式のパラメータを渡します。これらのパラメータは、引用符を使用して JSON データ構造を囲むことによって、コマンドラインで渡すこともできます。詳細については、AWS Command Line Interface ユーザーガイドの「AWS CLI での文字列への引用符の使用」を参照してください。

目次

スケーリングアクティビティ

Application Auto Scaling では、以下のスケーリングアクティビティを一時停止状態にすることができます。

- スケーリングポリシーによってトリガーされるすべてのスケールインアクティビティ。
- スケーリングポリシーによってトリガーされるすべてのスケールアウトアクティビティ。
- スケジュールされたアクションに関係するすべてのスケーリングアクティビティ。

以下の説明では、個々のスケーリングアクティビティが停止されると何が起こるかについて説明しています。それぞれ個別に停止および再開できます。スケーリングアクティビティを停止する理由によっては、複数のスケーリングアクティビティをまとめて停止する必要がある場合があります。

`DynamicScalingInSuspended`

- Application Auto Scaling は、ターゲット追跡スケーリングポリシーまたはステップスケーリングポリシーがトリガーされたときに容量を削除しません。これは、スケーリングポリシー、またはそれらに関連する CloudWatch アラームを削除することなく、スケーリングポリシーに関連付けられたスケールインアクティビティを一時的に無効化することを可能にします。スケールインを再開するときは、Application Auto Scaling が違反状態のアラームしきい値があるポリシーを評価します。

`DynamicScalingOutSuspended`

- Application Auto Scaling は、ターゲット追跡スケーリングポリシーまたはステップスケーリングポリシーがトリガーされたときに容量を追加しません。これは、スケーリングポリシー、またはそれらに関連する CloudWatch アラームを削除することなく、スケーリングポリシーに関連付けられたスケールアウトアクティビティを一時的に無効化することを可能にします。スケールアウトを再開するときは、Application Auto Scaling が違反状態のアラームしきい値があるポリシーを評価します。

`ScheduledScalingSuspended`

- Application Auto Scaling は、一時停止期間中に実行がスケジュールされているスケーリングアクションを開始しません。スケジュールされたスケーリングを再開するとき、Application Auto Scaling は、実行時刻がまだ過ぎていないスケジュールされたアクションのみを評価します。

AWS CLI を使用したスケーリングアクティビティの一時停止と再開

Application Auto Scaling のスケーラブルターゲットに対するスケーリングアクティビティは、個別に、またはすべてを一時停止して再開することができます。

> **Note**
>
> 簡略化のため、これらの例では、DynamoDB テーブルのスケーリングを一時停止して再開する方法を例示しています。別のスケーラブルターゲットを指定するには、`--service-namespace` でその名前空間、`--scalable-dimension` でそのスケーラブルディメンション、`--resource-id` でそのリソース ID を指定します。

スケーリングアクティビティを停止するには

コマンドラインウィンドウを開き、`--suspended-state` オプションがある register-scalable-target コマンドを以下のように使用します。

Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

スケーリングポリシーによってトリガーされたスケールインアクティビティのみを停止するには、config.json で次のように指定します。

```
{
    "DynamicScalingInSuspended":true
}
```

スケーリングポリシーによってトリガーされたスケールアウトアクティビティのみを停止するには、config.json で次のように指定します。

```
{
    "DynamicScalingOutSuspended":true
}
```

スケジュールされたアクションに関連するスケーリングアクティビティのみを停止するには、config.json で以下を指定します。

```
{
```

```
    "ScheduledScalingSuspended":true
}
```

すべてのスケーリングアクティビティを停止するには

`--suspended-state` オプションがある register-scalable-target コマンドを以下のように使用します。

Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

この例では、ファイル config.json に以下の JSON 形式パラメータが含まれていると仮定しています。

```
{
    "DynamicScalingInSuspended":true,
    "DynamicScalingOutSuspended":true,
    "ScheduledScalingSuspended":true
}
```

一時停止されたスケーリングアクティビティを表示する

describe-scalable-targets コマンドを使用して、スケーラブルターゲットに対するスケーリングアクティビティのどれが一時停止状態になっているかを判断します。

Linux、macOS、または Unix

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

Windows

```
aws application-autoscaling describe-scalable-targets --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table
```

以下は出力例です。

```
{
    "ScalableTargets": [
        {
            "ServiceNamespace": "dynamodb",
            "ScalableDimension": "dynamodb:table:ReadCapacityUnits",
            "ResourceId": "table/my-table",
            "MinCapacity": 1,
            "MaxCapacity": 20,
            "SuspendedState": {
                "DynamicScalingOutSuspended": true,
```

```
                "DynamicScalingInSuspended": true,
                "ScheduledScalingSuspended": true
            },
            "CreationTime": 1558125758.957,
            "RoleARN": "arn:aws:iam::123456789012:role/aws-
service-role/dynamodb.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_DynamoDBTable"
        }
    ]
}
```

スケーリングアクティビティを再開する

スケーリングアクティビティを再開する準備が整ったら、register-scalable-target コマンドを使用してそれらを再開できます。

次のコマンド例では、指定されたスケーラブルなターゲットのすべてのスケーリングアクティビティを再開します。

Linux、macOS、または Unix

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb \
  --scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table \
  --suspended-state file://config.json
```

Windows

```
aws application-autoscaling register-scalable-target --service-namespace dynamodb --
scalable-dimension dynamodb:table:ReadCapacityUnits --resource-id table/my-table --
suspended-state file://config.json
```

この例では、ファイル config.json に以下の JSON 形式パラメータが含まれていると仮定しています。

```
{
    "DynamicScalingInSuspended":false,
    "DynamicScalingOutSuspended":false,
    "ScheduledScalingSuspended":false
}
```

Application Auto Scaling のモニタリング

モニタリングは、Application Auto Scaling とその他 AWS ソリューションの信頼性、可用性、およびパフォーマンスを維持する上で重要な要素です。マルチポイント障害が発生した場合のデバッグをより簡単に実行できるように、AWS ソリューションのあらゆる部分からモニタリングデータを収集する必要があります。AWS は、Application Auto Scaling を監視し、問題の発生時に報告して、適切な場合は自動アクションを実行するためのモニタリングツールを提供します。

以下の機能を使用して、AWS リソースの管理に役立てることができます。

Amazon CloudWatch アラーム

異常なアプリケーション動作を検出するには、AWS リソースに関する特定のメトリクスを自動的にモニタリングする CloudWatch が役立ちます。CloudWatch アラームを設定して、メトリクスの値が期待どおりでない場合、または特定の異常が検出された場合に E メールを送信する Amazon SNS 通知をセットアップできます。たとえば、ネットワークアクティビティがメトリクスの期待値よりも急激に高くなった、または低くなったときに、通知を受け取ることができます。詳細については、「CloudWatch アラームでのモニタリング (p. 75)」および Amazon CloudWatch Logs ユーザーガイドを参照してください。

Amazon CloudWatch ダッシュボード

Amazon CloudWatch は、AWS リソースと、AWS で実行しているアプリケーションをリアルタイムでモニタリングします。CloudWatch ダッシュボードは、CloudWatch コンソールにあるカスタマイズ可能なホームページです。これらのページを使用して、異なるリージョンにまたがるリソースも含めて、単一のビューでリソースをモニタリングできます。CloudWatch ダッシュボードを使用して、AWS のリソースのメトリクスおよびアラームのカスタマイズされたビューを作成できます。詳細については、「CloudWatch を使用したダッシュボードの構築 (p. 76)」を参照してください。

Amazon EventBridge

EventBridge は、AWS リソースにおける変化を説明するシステムイベントの、ほぼリアルタイムのストリームを配信します。EventBridge は、自動化されたイベント駆動型のコンピューティングを可能にします。特定のイベントを監視し、これらのイベントが発生したときに AWS の他のサービスで自動化されたアクションをトリガーするルールを記述できます。詳細については、「EventBridge 経由でスケーリングを妨げているイベントの通知を受け取る (p. 79)」を参照してください。

AWS CloudTrail

AWS CloudTrail は、AWS アカウントによって行われた、またはアカウントに代わって実行された API コールと関連イベントをキャプチャします。次に、ユーザー指定の Amazon S3 バケットにログファイルを配信します。AWS を呼び出したユーザーとアカウント、呼び出し元のソース IP アドレス、および呼び出しの発生日時を特定できます。詳細については、AWS CloudTrail ユーザーガイドを参照してください。CloudTrail によってログに記録される Application Auto Scaling API コールについては、「Logging Application Auto Scaling API calls with CloudTrail」を参照してください。

Amazon CloudWatch Logs

Amazon CloudWatch Logs は、Amazon EC2 インスタンス、CloudTrail、およびその他ソースからのログファイルを監視および保存し、アクセスすることを可能にします。CloudWatch Logs は、ログファイル内の情報を監視し、特定のしきい値が満たされたときに通知します。また、耐久性の高いストレージにログデータをアーカイブすることもできます。詳細については、 Amazon CloudWatch Logs ユーザーガイドを参照してください。

AWS Health Dashboard

AWS Health Dashboard (PHD) は情報を表示し、AWS リソースの健全性の変更によってトリガーされる通知も提供します。情報は 2 つの方法で表示されます。ダッシュボードには、最近のイベントおよび予定されているイベントがカテゴリ別に分類されて表示されます。詳細なイベントログには、過去 90 日間のすべてのイベントが表示されます。詳細については、「Application Auto Scaling 向けの AWS Health Dashboard 通知 (p. 80)」を参照してください。

CloudWatch アラームでのモニタリング

注意が必要となる可能性がある問題を Amazon CloudWatch が検出した場合に通知を行うアラームを作成できます。CloudWatch は、AWS に関する特定のメトリクスを自動的にモニタリングすることによってユーザーを支援します。

CloudWatch アラームは、単一のメトリクスをモニタリングします。CloudWatch は、アラームの状態が変化し、その状態が指定された期間継続した場合に限り、1 つ、または複数のアクションを呼び出します。例えば、メトリクス値が特定のレベルまで低下した、またはそれを超過した場合に通知するアラームを設定して、潜在的な問題が発生する前に通知を受けることを確実にできます。

CloudWatch では、メトリクスが `INSUFFICIENT_DATA` 状態になったときに通知するアラームも設定できます。AWS のどのサービスのどのメトリクスでも、`INSUFFICIENT_DATA` についてアラームを発行することができます。これは新しいアラームの初期状態ですが、アラーム状態は、CloudWatch メトリクスが利用できなくなった場合、またはメトリクスがアラーム状態を判断するために十分なデータがない場合も `INSUFFICIENT_DATA` に変化します。例えば、AWS Lambda が 1 分ごとに `ProvisionedConcurrencyUtilization` メトリクスを CloudWatch に出力するのは、Lambda 関数がアクティブになっている場合のみです。関数がアクティブではない場合、メトリクスを待機する間にアラームが `INSUFFICIENT_DATA` 状態になる原因になります。これは正常な動作で、問題があることを意味するとは限りませんが、一定の期間内にアクティビティを期待していたが、アクティビティがなかったという場合は、問題を示唆している可能性があります。

このトピックでは、メトリクスがユーザー定義のしきい値内またはしきい値外の場合、またはデータが不足している場合に通知を送信するアラームを作成する方法を説明します。アラームの詳細については、Amazon CloudWatch ユーザーガイドの「Amazon CloudWatch アラームの使用」を参照してください。

E メールを送信するアラームを作成する

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで、[Alarms]、[Create Alarm] の順に選択します。
3. [メトリクスの選択] を選択します。

   すべてのメトリクスを確認できるページが表示されます。利用できるメトリクスのタイプは、使用するサービスと機能に応じて異なります。メトリクスは、まずサービス名前空間ごとにグループ化されてから、各名前空間内のさまざまなディメンションの組み合わせでグループ化されます。
4. メトリクスの名前空間 ([Lambda] など]) を選択してから、メトリクスディメンション ([By Function Name] (関数名別) など) を選択します。

   [All metrics] (すべてのメトリクス) タブには、選択したディメンションと名前空間のすべてのメトリクスが表示されます。
5. アラームを作成するメトリクスの横にあるチェックボックスをオンにしてから、[Select metric] (メトリクスの選択) をクリックします。
6. アラームを以下のように設定して、[Next] (次へ) をクリックします。

   - [Metric] (メトリクス) で、`1 minute` または `5 minutes` の集計期間を選択します。メトリクスの集計期間として 1 分を使用する場合、1 分ごとに 1 つのデータポイントができます。期間が短いほど、作成されるアラームの感度が高くなります。

- [Conditions] (条件) でしきい値を設定します。例えば、通知が生成される前にメトリクスが超過する必要がある値などです。
- [Additional configuration] (その他の設定) の [Datapoints to alarm] (アラームを実行するデータポイント) に、データポイントの数 (評価期間) を入力します。アラームをトリガーするには、メトリクス値がこの期間中にしきい値条件を満たす必要があります。例えば、2 つの連続する 5 分期間では、アラームがトリガーされるまで 10 分かかります。
- [Missing data treatment] (欠落データの処理) については、デフォルトをそのまま使用して、欠落データポイントを欠落しているとして処理します。

  メトリクスには、アクティビティが発生しているときだけ報告されるものもあります。これは、メトリクスがまばらにしか報告されない原因となる可能性があります。メトリクスのデータポイントが頻繁に欠落する仕様の場合は、これらの期間中、アラームの状態が `INSUFFICIENT_DATA` になります。アラームに以前の `ALARM` または `OK` 状態を強制的に維持させて、アラートがフラッピングしないようにするには、欠落データを無視することを代わりに選択できます。

7. [Notification] (通知) で、アラームが `ALARM` 状態、`OK` 状態、または `INSUFFICIENT_DATA` 状態のときに通知する SNS トピックを選択または作成します。同じアラーム状態または複数の異なるアラーム状態について複数の通知を送信するには、[Add notification (通知の追加)] を選択します。
8. 完了したら、[次へ] を選択します。
9. 名前を入力し、必要に応じてアラームの説明を入力して [次へ] を選択します。
10. [アラームの作成] を選択します。

アラームの状態をチェックする

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで [Alarms] (アラーム) をクリックしてアラームのリストを表示します。
3. アラームをフィルタリングするには、検索フィールドの横にあるドロップダウンフィルターを使用して、適用するフィルターオプションを選択します。
4. アラームを編集または削除するには、アラームを選択してから [Actions] (アクション)、[Edit] (編集)、または [Actions] (アクション)、[Delete] (削除) の順にクリックします。

CloudWatch を使用したダッシュボードの構築

使用状況とパフォーマンスに関するメトリクスを生成する Amazon CloudWatch を使用して、アプリケーションがリソースをどのように使用しているかをモニタリングできます。CloudWatch は、AWS リソースと AWS で実行しているアプリケーションから raw データを収集し、それを読み取り可能なほぼリアルタイムのメトリクスに処理します。メトリクスは 15 か月間保持されるため、履歴情報にアクセスして、アプリケーションのパフォーマンスをよりよく把握できます。詳細については、Amazon CloudWatch ユーザーガイドを参照してください。

CloudWatch ダッシュボードは、CloudWatch コンソールにあるカスタマイズ可能なホームページで、リソースが異なるリージョン全体に散在している場合でも、リソースを単一のビューでモニタリングするために使用できます。CloudWatch ダッシュボードを使用して、AWS リソース用に選択したメトリクスのカスタマイズされたビューを作成できます。各グラフのメトリクスごとに使用する色を選択できるため、複数のグラフで同一のメトリクスを追跡しやすくなります。

CloudWatch ダッシュボードを作成する

1. CloudWatch コンソール (https://console.aws.amazon.com/cloudwatch/) を開きます。
2. ナビゲーションペインで、[ダッシュボード] を選択し、[新しいダッシュボードの作成] を選択します。
3. ダッシュボードの名前を入力します。これは、CloudWatch データを表示するサービスの名前などにすることができます。

4. [ダッシュボードの作成] を選択します。
5. 折れ線グラフなど、ダッシュボードに追加するウィジェットのタイプを選択します。次に、[設定] を選択し、ダッシュボードに追加するメトリクスを選択します。詳細については、Amazon CloudWatch ユーザーガイドの「CloudWatch ダッシュボードからグラフを追加または削除する」を参照してください。

デフォルトで、CloudWatch ダッシュボードで作成するメトリクスは平均値です。

メトリクスとディメンション

Application Auto Scaling と統合するサービスとやり取りするときは、サービスが以下の表にあるメトリクスを CloudWatch に送信します。CloudWatch では、メトリクスがまずサービス名前空間ごとにグループ化され、次に各名前空間内のさまざまなディメンションの組み合わせでグループ化されます。

これらのメトリクスは、アプリケーションのキャパシティ要件の検出に役立ちます。この情報を使用して、容量を静的に設定したり、自動スケーリングを設定したりできます。アプリケーションのワークロードが一定ではない場合は、オートスケーリングの使用を検討する必要があることを示しています。

メトリクス名	名前空間	ディメンション	Applies to	
AvailableCapacity	AWS/AppStream	フリート	AppStream	
CapacityUtilization	AWS/AppStream	フリート	AppStream	
CPUUtilization	AWS/RDS	DBClusterIdentifier、Role (READER)	Aurora	
DatabaseConnections	AWS/RDS	DBClusterIdentifier、Role (READER)	Aurora	
InferenceUtilization	AWS/Comprehend	EndpointArn	Comprehend	
ProvisionedReadCapacityUnits	AWS/DynamoDB	TableName、GlobalSecondaryIndexName	DynamoDB	
ProvisionedWriteCapacityUnits	AWS/DynamoDB	TableName、GlobalSecondaryIndexName	DynamoDB	
ConsumedReadCapacityUnits	AWS/DynamoDB	TableName、GlobalSecondaryIndexName	DynamoDB	
ConsumedWriteCapacityUnits	AWS/DynamoDB	TableName、GlobalSecondaryIndexName	DynamoDB	
CPUUtilization	AWS/ECS	ClusterName, ServiceName	ECS	
MemoryUtilization	AWS/ECS	ClusterName, ServiceName	ECS	
RequestCountPerTarget	AWS/ApplicationELB	TargetGroup	ECS	
YARNMemoryAvailablePercentage	AWS/ElasticMapReduce	ClusterId	EMR	

メトリクス名	名前空間	ディメンション	Applies to	
ProvisionedReadCapacityUnits	AWS/Cassandra	Keyspace、TableName	Amazon Keyspaces	
ProvisionedWriteCapacityUnits	AWS/Cassandra	Keyspace、TableName	Amazon Keyspaces	
ConsumedReadCapacityUnits	AWS/Cassandra	Keyspace、TableName	Amazon Keyspaces	
ConsumedWriteCapacityUnits	AWS/Cassandra	Keyspace、TableName	Amazon Keyspaces	
ProvisionedConcurrencyUtilization	AWS/Lambda	FunctionName、Resource	Lambda	
KafkaDataLogsDiskUsed	AWS/Kafka	クラスター名	Amazon MSK	
KafkaDataLogsDiskUsed	AWS/Kafka	クラスター名、ブローカー ID	Amazon MSK	
InvocationsPerInstance	AWS/SageMaker	EndpointName、VariantName	SageMaker	
CPUUtilization	AWS/EC2Spot	FleetRequestId	スポットフリート	
NetworkIn	AWS/EC2Spot	FleetRequestId	スポットフリート	
NetworkOut	AWS/EC2Spot	FleetRequestId	スポットフリート	
RequestCountPerTarget	AWS/ApplicationELB	TargetGroup	スポットフリート	

CloudWatch では、各メトリクスについて任意の統計と期間を選択できますが、すべての組み合わせが役に立つわけではありません。たとえば、CPU 使用率の Average、Minimum、Maximum 統計は役に立ちますが、Sum 統計は役に立ちません。詳細については、前述の表に示されているリンクをたどって、サービスのドキュメントを参照してください。

アプリケーションのパフォーマンスで一般的に使用される指標は、平均 CPU 使用率です。CPU 使用率が増加し、処理できる容量が不足している場合、アプリケーションが応答しなくなる可能性があります。一方、容量が多すぎて、使用率が低いときにリソースが実行されている場合は、そのサービスを使用するためのコストが増加します。

サービスによっては、利用可能なプロビジョニングされたスループットの量を追跡するメトリクスもあります。例えば、プロビジョニングされた同時実行数が設定されている関数のエイリアスまたはバージョンで処理されている呼び出しの数について、Lambda は `ProvisionedConcurrencyUtilization` メトリクスを出力します。大きなジョブを開始し、同じ関数を何度も同時に呼び出した場合、使用可能なプロビジョニングされた同時実行量を超えると、ジョブにレイテンシーが発生する可能性があります。一方、プロビジョニングされた同時実行性が必要以上に高い場合は、コストが必要以上に高くなる可能性があります。

CloudWatch コンソールにこれらのメトリクスが表示されない場合は、リソースのセットアップが完了していることを確認してください。リソースが完全にセットアップされるまでは、メトリクスは表示されません。また、メトリクスが過去 14 日間データを発行していない場合は、CloudWatch ダッシュボードでグ

ラフに追加するメトリクスを検索するときに、そのメトリクスを見つけることはできません。メトリクスを手動で追加する方法については、Amazon CloudWatch ユーザーガイドの「CloudWatch ダッシュボードで手動でメトリクスをグラフ化する」を参照してください。

EventBridge 経由でスケーリングを妨げているイベントの通知を受け取る

Application Auto Scaling は Amazon EventBridge と統合して、スケーリングに影響を及ぼす特定のイベントについて通知します。AWS サービスからのイベントは、ほぼリアルタイムに EventBridge に提供されます。簡単なルールを記述して、注目するイベントと、イベントがルールに一致した場合に自動的に実行するアクションを指定できます。自動的にトリガーできるオペレーションには、以下が含まれます。

- AWS Lambda 関数の呼び出し
- Amazon EC2 Run Command の呼び出し
- Amazon Kinesis Data Streams へのイベントの中継
- AWS Step Functions ステートマシンのアクティブ化
- Amazon SNS トピックまたは Amazon SQS キューの通知

Application Auto Scaling API コールでトリガーされるルールを作成することもできます。詳細については、Amazon EventBridge ユーザーガイドの「Creating an EventBridge rule that triggers on an AWS API call using AWS CloudTrail」を参照してください。

詳細については、Amazon EventBridge ユーザーガイドの「Getting started with Amazon EventBridge」を参照してください。

Application Auto Scaling イベント

以下は、Application Auto Scaling からのサンプルイベントです。イベントは、ベストエフォートベースで出力されます。

現在使用できるのは、`scaledToMax` 固有のイベントのみです。

状態変化のイベント: 最大までスケーリング

次のイベントは、そのリソースのスケーリング設定で指定した最大容量制限に達したときに送信されます。

```
{
  "version": "0",
  "id": "7bf73129-1428-4cd3-a780-95db273d1602",
  "detail-type": "Application Auto Scaling Scaling Activity State Change",
  "source": "aws.application-autoscaling",
  "account": "123456789012",
  "time": "2019-06-12T10:23:40Z",
  "region": "us-west-2",
  "resources": [],
  "detail": {
    "startTime": "2019-06-12T10:20:43Z",
    "endTime": "2019-06-12T10:23:40Z",
    "newDesiredCapacity": 8,
    "oldDesiredCapacity": 5,
    "minCapacity": 2,
    "maxCapacity": 8,
    "resourceId": "table/my-table",
    "scalableDimension": "dynamodb:table:WriteCapacityUnits",
```

```
    "serviceNamespace": "dynamodb",
    "statusCode": "Successful",
    "scaledToMax": true,
    "direction": "scale-out",
}
```

最大までスケールされたイベントのサンプルイベントパターン

ルールでは、イベントパターンを使用してイベントを選択し、ターゲットに振り分けます。以下は、Application Auto Scaling のサンプルイベントパターンです。

```
{
  "source": [
    "aws.application-autoscaling"
  ],
  "detail-type": [
    "Application Auto Scaling Scaling Activity State Change"
  ],
  "detail": {
    "scaledToMax": [
      true
    ]
  }
}
```

Application Auto Scaling 向けの AWS Health Dashboard 通知

失敗したスケーリングイベントを管理しやすくするために、AWS Health Dashboard は Application Auto Scaling が出力する通知のサポートを提供します。現在利用できるのは、DynamoDB リソースに固有のスケールアウトイベントのみです。

AWS Health Dashboard は AWS Health サービスの一部です。セットアップを行う必要はなく、アカウントで認証されているすべてのユーザーが表示できます。詳細については、「Getting started with the AWS Health Dashboard」を参照してください。

DynamoDB の Service Quotas 制限が原因で DynamoDB リソースがスケールアウトされない場合は、以下のようなメッセージを受け取ります。このメッセージが表示された場合は、アクションを実行するためのアラームとして処理する必要があります。

```
  Hello,

  A scaling action has attempted to scale out your DynamoDB resources in the
  eu-west-1 region. This operation has been prevented because it would have exceeded a
  table-level write throughput limit (Provisioned mode). This limit restricts the
  provisioned write capacity of the table and all of its associated global secondary
  indexes. To address the issue, refer to the Amazon DynamoDB Developer Guide for
  current limits and how to request higher limits [1].

  To identify your DynamoDB resources that are impacted, use the
  describe-scaling-activities command or the DescribeScalingActivities operation [2][3].
  Look for a scaling activity with StatusCode "Failed" and a StatusMessage similar to
  "Failed to set write capacity units to 45000. Reason: The requested WriteCapacityUnits,
  45000, is above the per table maximum for the account in eu-west-1. Per table maximum:
  40000." You can also view these scaling activities from the Capacity tab of your tables
  in the AWS Management Console for DynamoDB.

  We strongly recommend that you address this issue to ensure that your tables
```

```
  are prepared to handle increases in traffic. This notification is sent only once in
  each 12 hour period, even if another failed scaling action occurs.

  [1] https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Limits.html#default-
limits-throughput-capacity-modes
  [2] https://docs.aws.amazon.com/cli/latest/reference/application-autoscaling/describe-
scaling-activities.html
  [3] https://docs.aws.amazon.com/autoscaling/application/APIReference/
API_DescribeScalingActivities.html

  Sincerely,
  Amazon Web Services
```

Application Auto Scaling のセキュリティ

AWS では、クラウドのセキュリティが最優先事項です。AWS のお客様は、セキュリティを最も重視する組織の要件を満たすように構築されたデータセンターとネットワークアーキテクチャから利点を得られます。

セキュリティは、AWS とお客様の間の責任共有です。責任共有モデルでは、これをクラウドのセキュリティおよびクラウド内のセキュリティと説明しています。

- クラウドのセキュリティ - AWS は、AWS クラウドで AWS のサービスを実行するインフラストラクチャを保護する責任を負います。また、AWS は、使用するサービスを安全に提供します。AWS コンプライアンスプログラムの一環として、サードパーティーの監査が定期的にセキュリティの有効性をテストおよび検証しています。Application Auto Scaling に適用されるコンプライアンスプログラムの詳細については、コンプライアンスプログラムによる AWS 対象範囲内のサービスを参照してください。
- クラウド内のセキュリティ – お客様の責任は使用する AWS のサービスによって決まります。また、お客様は、お客様のデータの機密性、企業の要件、および適用可能な法律および規制などの他の要因についても責任を担います。

このドキュメントは、Application Auto Scaling の使用時に責任共有モデルがどのように適用されるかを理解するために役立ちます。以下のトピックでは、セキュリティおよびコンプライアンス上の目的を達成するために Application Auto Scaling を設定する方法について説明します。また、Application Auto Scaling リソースのモニタリングとセキュア化に役立つ AWS のその他のサービスを使用する方法も学びます。

トピック

Application Auto Scaling とインターフェイス VPC エンドポイント

インターフェイス VPC エンドポイントを使用するように Application Auto Scaling を設定することで、VPC のセキュリティ体制を強化できます。インターフェイスエンドポイントは、VPC と Application Auto Scaling の間のすべてのネットワークトラフィックを Amazon ネットワークに制限することにより、プライベートで AApplication Auto Scaling API にアクセスすることを可能にする、AWS PrivateLink を使用した技術です。インターフェイスエンドポイントでは、インターネットゲートウェイ、NAT デバイス、または仮想プライベートゲートウェイも必要ありません。

AWS PrivateLink の設定は要件ではありませんが、推奨されます。AWS PrivateLink および VPC エンドポイントの詳細については、「AWS PrivateLink ガイド」の「What is AWS PrivateLink?」(AWS PrivateLink とは?) を参照してください。

トピック

インターフェイス VPC エンドポイントを作成する

Application Auto Scaling 用のエンドポイントは、以下のサービス名を使用して作成します。

```
com.amazonaws.region.application-autoscaling
```

詳細については、「AWS PrivateLink ガイド」の「Access an AWS service using an interface VPC endpoint」(インターフェイス VPC エンドポイントを使用して AWS のサービスにアクセスする) を参照してください。

その他の設定を変更する必要はありません。Application Auto Scaling は、サービスエンドポイントまたはプライベートインターフェイス VPC エンドポイントのうち、使用されている方のエンドポイントを使用して、AWS の他のサービスを呼び出します。

VPC エンドポイントポリシーを作成する

Application Auto Scaling API へのアクセスを制御するために、VPC エンドポイントにポリシーをアタッチすることができます。このポリシーでは以下の内容を指定します。

- アクションを実行できるプリンシパル。
- 実行可能なアクション。
- このアクションを実行できるリソース。

以下の例では、エンドポイントを介してスケーリングポリシーを削除するためのアクセス許可を全員に対して拒否する VPC エンドポイントポリシーを示しています。このポリシー例では、他のすべてのアクションを実行するアクセス許可も全員に付与しています。

```
{
   "Statement": [
        {
            "Action": "*",
            "Effect": "Allow",
            "Resource": "*",
            "Principal": "*"
        },
        {
            "Action": "application-autoscaling:DeleteScalingPolicy",
            "Effect": "Deny",
            "Resource": "*",
            "Principal": "*"
        }
    ]
}
```

詳細については、「AWS PrivateLink ガイド」の「VPC endpoint policies」(VPC エンドポイントポリシー) を参照してください。

エンドポイントの移行

Application Auto Scaling は最近、Application Auto Scaling API に対するコールのための新しいデフォルトの DNS ホスト名とエンドポイントとして、application-autoscaling.*region*.amazonaws.com

を導入しました。新しいエンドポイントには、AWS CLI および SDK の最新リリースとの互換性があります。まだインストールしていない場合は、最新の AWS CLI と SDK をインストールして、新しいエンドポイントを使用してください。AWS CLI を更新するには、AWS Command Line Interface ユーザーガイドで pip を使用した AWS CLI のインストールについて参照してください。AWS SDK については、Amazon Web Services のツールを参照してください。

Note

下位互換性のため、`autoscaling.region.amazonaws.com` エンドポイントは、Application Auto Scaling API に対するコール用に引き続きサポートされます。`autoscaling.region.amazonaws.com` エンドポイントをプライベートインターフェイス VPC エンドポイントとしてセットアップするには、Amazon EC2 Auto Scaling ユーザーガイドの「Amazon EC2 Auto Scaling and interface VPC endpoints」を参照してください。

CLI または AWS API の使用時に呼び出すエンドポイント

Application Auto Scaling の現在のリリースでは、Application Auto Scaling API に対するコールが、`autoscaling.region.amazonaws.com` ではなく、`application-autoscaling.region.amazonaws.com` エンドポイントに自動送信されます。

各コマンドでパラメータ `--endpoint-url https://application-autoscaling.region.amazonaws.com` を使用してエンドポイントを指定することにより、CLI で新しいエンドポイントを呼び出すことができます。

推奨されませんが、各コマンドでパラメータ `--endpoint-url https://autoscaling.region.amazonaws.com` を使用してエンドポイントを指定することにより、CLI で古いエンドポイントを呼び出すこともできます。

API コールに使用されるさまざまな SDK については、目的の SDK のドキュメントで特定のエンドポイントへのリクエストの送信方法を参照してください。詳細については、Tools for Amazon Web Servicesを参照してください。

Application Auto Scaling とデータ保護

AWS の責任共有モデルは、Application Auto Scaling でのデータ保護に適用されます。このモデルで説明されているように、AWS は、AWS クラウド のすべてを実行するグローバルインフラストラクチャを保護する責任を負います。お客様は、このインフラストラクチャでホストされているコンテンツに対する管理を維持する責任があります。このコンテンツには、使用される AWS のサービス のセキュリティ設定と管理タスクが含まれます。データプライバシーの詳細については、「データプライバシーのよくある質問」を参照してください。欧州でのデータ保護の詳細については、AWS セキュリティブログ に投稿された「AWS 責任共有モデルおよび GDPR」のブログ記事を参照してください。

データを保護するため、AWS アカウント の認証情報を保護し、AWS Identity and Access Management (IAM)を使用して個々のユーザーアカウントをセットアップすることをお勧めします。この方法により、それぞれのジョブを遂行するために必要な許可のみを各ユーザーに付与できます。また、次の方法でデータを保護することをお勧めします。

- 各アカウントで多要素認証 (MFA) を使用します。
- SSL/TLS を使用して AWS リソースと通信します。TLS 1.2 以降が推奨されています。
- AWS CloudTrail で API とユーザーアクティビティログをセットアップします。
- AWS 暗号化ソリューションを AWS のサービス内のすべてのデフォルトのセキュリティ管理と一緒に使用します。
- Amazon Macie などのアドバンストマネージドセキュリティサービスを使用します。これは、Amazon S3 に保存されている個人データの検出と保護を支援します。

- コマンドラインインターフェイスまたは API を使用して AWS にアクセスするときに FIPS 140-2 検証済みの暗号化モジュールが必要な場合は、FIPS エンドポイントを使用します。使用可能な FIPS エンドポイントの詳細については、「連邦情報処理規格 (FIPS) 140-2」を参照してください。

顧客のメールアドレスなどの機密または注意を要する情報は、タグや [Name] (名前) フィールドなど自由形式のフィールドに配置しないことを強くお勧めします。これには、コンソール、API、AWS CLI、または AWS SDK を用いた Application Auto Scaling または AWS のその他サービスの使用時が含まれます。タグまたは名前に使用する自由記入欄に入力したデータは、課金や診断ログに使用される場合があります。外部サーバーへの URL を提供する場合は、そのサーバーへのリクエストを検証するための認証情報を URL に含めないように強くお勧めします。

Application Auto Scaling の Identity and Access Management

AWS Identity and Access Management (IAM) は、管理者が AWS リソースへのアクセスをセキュアに制御するために役立つ Amazon Web Services (AWS) のサービスです。IAM 管理者は、誰を認証 (サインイン) し、誰に AWS リソースの使用を許可する (アクセス許可を持たせる) かを制御します。IAM では、AWS のサービスで追加料金は発生しません。

Application Auto Scaling を使用するには、AWS アカウントと認証情報が必要です。AWS アカウントのセキュリティを強化するため、AWS アカウントルートユーザーの認証情報を使用するのではなく、IAM ユーザーを使用して認証されたリクエストを実行することが推奨されます。IAM ユーザーを作成して、そのユーザーに完全なアクセス権を付与できます。このようなユーザーを管理者ユーザーと呼びます。AWS とやり取りし、スケーリングポリシーの設定などのタスクを実行するには、AWS アカウントルートユーザーの認証情報ではなく、管理者ユーザーの認証情報を使用できます。詳細については、AWS アカウント 全般のリファレンスの 「AWS アカウントルートユーザーの認証情報と IAM ユーザーの認証情報」、および IAM ユーザーガイドの「IAM でのセキュリティのベストプラクティス」を参照してください。

IAM ユーザーの作成後、クエリ (HTTPS) インターフェイスを使用して直接行う、または SDK、AWS Command Line Interface、もしくは AWS Tools for Windows PowerShell を使用して間接的に行うかにかかわらず、Application Auto Scaling API 経由で Application Auto Scaling にアクセスする場合は、AWS アクセスキーを取得する必要があります。AWS アクセスキーは、アクセスキー ID とシークレットアクセスキーで構成されています。AWS アクセスキーの取得に関する詳細については、AWS 全般のリファレンスの「AWS セキュリティ認証情報」を参照してください。

Application Auto Scaling の使用をすばやく開始するには、「設定する (p. 3)」の手順を実行します。

アクセスコントロール

リクエストを認証するための有効な認証情報があっても、許可がなければ Application Auto Scaling リソースを作成、またはそれらにアクセスすることはできません。例えば、スケーリングポリシーの作成、スケジュールされたスケーリングの設定などのアクセス権限が必要です。

以下のセクションでは、IAM 管理者が、Application Auto Scaling API を実行できるユーザーを制御することによって AWS リソースをセキュア化するために IAM を使用する方法を詳しく説明します。

トピック

Application Auto Scaling で IAM が機能する仕組み

> Note
>
> 2017 年 12 月、Application Auto Scaling の更新が行われ、Application Auto Scaling 統合サービスのために複数のサービスリンクロールが有効化されました。ユーザーがスケーリングを設定できるようにするには、特定の IAM 許可、および Application Auto Scaling サービスリンクロール (または Amazon EMR オートスケーリング用のサービスロール) が必要です。

IAM を使用して Application Auto Scaling へのアクセスを管理する前に、Application Auto Scaling で使用できる IAM 機能を理解しておく必要があります。Application Auto Scaling と AWS のその他サービスで IAM が機能する仕組みの概要については、IAM ユーザーガイドの「IAM と連携する AWS のサービス」を参照してください。

トピック

Application Auto Scaling のアイデンティティベースポリシー

IAM アイデンティティベースポリシーでは、許可または拒否されるアクションとリソース、およびアクションが許可または拒否される条件を指定できます。Application Auto Scaling は、特定のアクション、リソース、および条件キーをサポートします。JSON ポリシーで使用するすべての要素については、IAM ユーザーガイド のIAM JSON ポリシーエレメントのリファレンスを参照してください。

アクション

管理者は AWS JSON ポリシーを使用して、誰が何にアクセスできるかを指定できます。つまり、どのプリンシパルがどのリソースに対してどのような条件下でアクションを実行できるかということです。

JSON ポリシーの `Action` 要素には、ポリシー内のアクセスを許可または拒否するために使用できるアクションが記述されます。ポリシーアクションの名前は通常、関連する AWS API オペレーションと同じです。一致する API オペレーションのない許可のみのアクションなど、いくつかの例外があります。また、ポリシーに複数のアクションが必要なオペレーションもあります。これらの追加アクションは、依存アクションと呼ばれます。

このアクションは、関連付けられたオペレーションを実行するための許可を付与するポリシーで使用されます。

IAM ポリシーステートメントの Application Auto Scaling API は、アクションの前に次のプレフィックスを使用します：`application-autoscaling:`。ポリシーステートメントには、`Action` または `NotAction` 要素を含める必要があります。Application Auto Scaling は、このサービスで実行できるタスクを記述する、独自の一連のアクションを定義します。

1 つのステートメントで複数のアクションを指定するには、次の例のようにカンマで区切ります。

```
"Action": [
      "application-autoscaling:DescribeScalingPolicies",
```

```
    "application-autoscaling:DescribeScalingActivities"
```

ワイルドカード (*) を使用して複数のアクションを指定することができます。例えば、`Describe` という単語で始まるすべてのアクションを指定するには、次のアクションを含めます。

```
"Action": "application-autoscaling:Describe*"
```

Application Auto Scaling のポリシーアクションの完全なリストを確認するには、「サービス認証リファレンス」の「Application Auto Scaling のアクション、リソース、および条件キー」を参照してください。

リソース

`Resource` 要素は、アクションが適用されるオブジェクトを指定します。

Application Auto Scaling には、IAM ポリシーステートメントの `Resource` 要素として使用できるサービス定義のリソースがありません。このため、IAM ポリシーで使用するための Application Auto Scaling の Amazon リソースネーム (ARN) もありません。Application Auto Scaling API アクションへのアクセスを制御するため、IAM ポリシーを記述するときは、常に * (アスタリスク) をリソースとして使用してください。

条件キー

`Condition` 要素 (または`Condition` ブロック) を使用すると、ステートメントが有効な条件を指定できます。例えば、特定の日付の後にのみ適用されるポリシーが必要になる場合があります。条件を表すには、あらかじめ定義された条件キーを使用します。

Application Auto Scaling はサービス固有の条件キーを提供しませんが、いくつかのグローバル条件キーの使用がサポートされています。すべての AWS グローバル条件キーを確認するには、「IAM ユーザーガイド」の「AWS グローバル条件コンテキストキー」を参照してください。

`Condition` 要素はオプションです。

例

Application Auto Scaling のアイデンティティベースポリシーの例については、「Application Auto Scaling のアイデンティティベースポリシー例 (p. 99)」を参照してください。

Application Auto Scaling のリソースベースポリシー

Amazon Simple Storage Service などの AWS のその他サービスでは、リソースベースの許可ポリシーがサポートされています。例えば、ポリシーを S3 バケットにアタッチして、そのバケットに対するアクセス許可を管理できます。

Application Auto Scaling は、リソースベースポリシーをサポートしません。

アクセスコントロールリスト (ACL)

Application Auto Scaling は、アクセスコントロールリスト (ACL) をサポートしません。

Application Auto Scaling タグに基づく認可

Application Auto Scaling には、タグ付けできるサービス定義のリソースがありません。したがって、タグに基づくアクセスの制御はサポートされていません。

Application Auto Scaling の IAM ロール

IAM ロール は、特定のアクセス許可を持つ、AWS アカウント 内のエンティティです。

Application Auto Scaling での一時的な認証情報の使用

一時的な認証情報を使用して、フェデレーションでサインインする、IAM ロールを引き受ける、またはクロスアカウントロールを引き受けることができます。一時的なセキュリティ認証情報を取得するには、AssumeRole または GetFederationToken などの AWS STS API オペレーションを呼び出します。

Application Auto Scaling は、一時的な認証情報の使用をサポートします。

サービスリンクロール

サービスリンクロールは、Application Auto Scaling がユーザーに代わって AWS のその他サービスに特定の呼び出しを行うことができるようにする許可を付与します。サービスリンクロールは、IAM アカウント内に表示され、サービスによって所有されます。IAM 管理者はサービスリンクロールの許可を表示できますが、編集することはできません。

Application Auto Scaling は、サービスリンクロールをサポートします。詳細については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

サービスロール

Amazon EMR クラスターがオートスケーリングを使用する場合、この機能は、Application Auto Scaling がユーザーに代わってサービスロールを引き受けることを許可します。サービスリンクロールと同様に、サービスロールは、サービスがユーザーに代わって他のサービスのリソースにアクセスし、アクションを完了することを許可します。サービスロールは、IAM アカウントに表示され、アカウントによって所有されます。つまり、IAM 管理者が、このロールの許可を変更することができます。ただし、これを行うことにより、サービスの機能が損なわれる場合があります。

Application Auto Scaling は、Amazon EMR に対してのみサービスロールをサポートします。EMR サービスロールのドキュメントについては、Amazon EMR 管理ガイドの「Using automatic scaling with a custom policy for instance groups」を参照してください。

> Note
>
> サービスにリンクされたロールの導入により、いくつかのレガシーサービスロールは不要になりました。例えば、Amazon ECS やスポットフリートなどです。

Application Auto Scaling 用の AWS マネージドポリシー

ユーザー、グループ、ロールへの許可の追加は、ポリシーを独自に記述するよりも、AWS マネージドポリシーを使用する方が簡単に実行できます。チームに必要な許可のみを提供する IAM カスタマーマネージドポリシーを作成するには、時間と専門知識が必要です。すぐに使用を開始するために、AWS マネージドポリシーを使用できます。これらのポリシーは、一般的なユースケースをターゲット範囲に含めており、AWS アカウント で利用できます。AWS マネージドポリシーの詳細については、「IAM ユーザーガイド」の「AWS マネージドポリシー」を参照してください。

AWS のサービス は、AWS マネージドポリシーを維持し、更新します。AWS マネージドポリシーの許可を変更することはできません。サービスでは、新しい機能を利用できるようにするために、AWS マネージドポリシーに許可が追加されることがあります。この種類の更新は、ポリシーがアタッチされている、すべてのアイデンティティ (ユーザー、グループおよびロール) に影響を与えます。新しい機能が立ち上げられた場合や、新しいオペレーションが使用可能になった場合に、各サービスが AWS マネージドポリシーを更新する可能性が最も高くなります。サービスは、AWS マネージドポリシーから許可を削除しないため、ポリシーの更新によって既存の許可が破棄されることはありません。

さらに、AWS では、複数のサービスにまたがるジョブ機能のためのマネージドポリシーもサポートしています。例えば、`ViewOnlyAccess` AWS マネージドポリシーでは、多くの AWS のサービス およびリソー

スへの読み取り専用アクセスを許可します。あるサービスで新しい機能を立ち上げる場合は、AWS は、追加された演算とリソースに対し、読み取り専用の許可を追加します。職務機能ポリシーのリストと説明については、「IAM ユーザーガイド」の「ジョブ機能の AWS マネージドポリシー」を参照してください。

目次

AppStream 2.0 と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingAppStreamFleetPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingAppStreamFleetPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Amazon AppStream と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_AppStreamFleet` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `appstream:DescribeFleets`
- アクション: `appstream:UpdateFleet`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

Aurora と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingRDSClusterPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingRDSClusterPolicy` をアタッチすることはできません。このポリシー

は、Application Auto Scaling がユーザーに代わって Aurora と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_RDSCluster` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `rds:AddTagsToResource`
- アクション: `rds:CreateDBInstance`
- アクション: `rds:DeleteDBInstance`
- アクション: `rds:DescribeDBClusters`
- アクション: `rds:DescribeDBInstance`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

Amazon Comprehend と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingComprehendEndpointPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingComprehendEndpointPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Amazon Comprehend と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `comprehend:UpdateEndpoint`
- アクション: `comprehend:DescribeEndpoint`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

DynamoDB と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingDynamoDBTablePolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingDynamoDBTablePolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって DynamoDB と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_DynamoDBTable` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `dynamodb:DescribeTable`
- アクション: `dynamodb:UpdateTable`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

Amazon ECS と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingECSServicePolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingECSServicePolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Amazon ECS と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_ECSService` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `ecs:DescribeServices`
- アクション: `ecs:UpdateService`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

ElastiCache と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingElastiCacheRGPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingElastiCacheRGPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって ElastiCache と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG` サービスリンクロール許可ポリシーは、Application Auto Scaling が指定されたリソースで以下のアクションを完了することを許可します。

- アクション: すべてのリソースでの `elasticache:DescribeReplicationGroups`
- アクション: すべてのリソースでの `elasticache:ModifyReplicationGroupShardConfiguration`
- アクション: すべてのリソースでの `elasticache:IncreaseReplicaCount`

- アクション: すべてのリソースでの `elasticache:DecreaseReplicaCount`
- アクション: すべてのリソースでの `elasticache:DescribeCacheClusters`
- アクション: すべてのリソースでの `elasticache:DescribeCacheParameters`
- アクション: すべてのリソースでの `cloudwatch:DescribeAlarms`
- アクション: リソース `arn:*:cloudwatch:*:*:alarm:TargetTracking*` での `cloudwatch:PutMetricAlarm`
- アクション: リソース `arn:*:cloudwatch:*:*:alarm:TargetTracking*` での `cloudwatch:DeleteAlarms`
- アクション: `cloudwatch:DeleteAlarms`

Amazon Keyspaces と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingCassandraTablePolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingCassandraTablePolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Amazon Keyspaces と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_CassandraTable` サービスリンクロール許可ポリシーは、Application Auto Scaling が指定されたリソースで以下のアクションを完了することを許可します。

- アクション: リソース `arn:*:cassandra:*:*:/keyspace/system/table/*` での `cassandra:Select`
- アクション: リソース `arn:*:cassandra:*:*:/keyspace/system_schema/table/*` での `cassandra:Select`
- アクション: リソース `arn:*:cassandra:*:*:/keyspace/system_schema_mcs/table/*` での `cassandra:Select`
- アクション: リソース `arn:*:cassandra:*:*:"*"` での `cassandra:Alter`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

Lambda と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingLambdaConcurrencyPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingLambdaConcurrencyPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Lambda と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `lambda:PutProvisionedConcurrencyConfig`
- アクション: `lambda:GetProvisionedConcurrencyConfig`
- アクション: `lambda:DeleteProvisionedConcurrencyConfig`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

Amazon MSK と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingKafkaClusterPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingKafkaClusterPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Amazon MSK と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_KafkaCluster` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `kafka:DescribeCluster`
- アクション: `kafka:DescribeClusterOperation`
- アクション: `kafka:UpdateBrokerStorage`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

Napture と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingNeptuneClusterPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingNeptuneClusterPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Napture と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_NeptuneCluster` サービスリンクロール許可ポリシーは、Application Auto Scaling が指定されたリソースで以下のアクションを完了することを許可します。

- アクション:Amazon Neptune データベースエンジン (`"Condition":{"StringEquals":{"rds:DatabaseEngine":"neptune"}}`) のプレフィックス Autoscaled 閲覧者が付いたリソースの `rds:AddTagsToResource`
- アクション: すべてのリソースでの `rds:ListTagsForResource`

- アクション:Amazon Neptune データベースエンジン (`"Condition": {"StringEquals":{"rds:DatabaseEngine":"neptune"}}`) のすべての DB クラスター (`"Resource":"arn:*:rds:*:*:db:autoscaled-reader*", "arn:aws:rds:*:*:cluster:*"`) のプレフィックス Autoscaled 閲覧者が付いたリソースの `rds:CreateDBInstance`
- アクション: すべてのリソースでの `rds:DescribeDBInstances`
- アクション: すべてのリソースでの `rds:DescribeDBClusters`
- アクション: すべてのリソースでの `rds:DescribeDBClusterParameters`
- アクション: リソース `arn:*:rds:*:*:db:autoscaled-reader*` での `rds:DeleteDBInstance`
- アクション: すべてのリソースでの `cloudwatch:DescribeAlarms`
- アクション: リソース `arn:*:cloudwatch:*:*:alarm:TargetTracking*` での `cloudwatch:PutMetricAlarm`
- アクション: リソース `arn:*:cloudwatch:*:*:alarm:TargetTracking*` での `cloudwatch:DeleteAlarms`
- アクション: `cloudwatch:DeleteAlarms`

SageMaker と CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingSageMakerEndpointPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingSageMakerEndpointPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって SageMaker と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `sagemaker:DescribeEndpoint`
- アクション: `sagemaker:DescribeEndpointConfig`
- アクション: `sagemaker:UpdateEndpointWeightsAndCapacities`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

EC2 スポットフリートと CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoscalingEC2SpotFleetRequestPolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoscalingEC2SpotFleetRequestPolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって Amazon EC2 と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `ec2:DescribeSpotFleetRequests`
- アクション: `ec2:ModifySpotFleetRequest`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

カスタムリソースと CloudWatch へのアクセス権を付与する AWS マネージドポリシー

ポリシー名: AWSApplicationAutoScalingCustomResourcePolicy

AWS Identity and Access Management (IAM) エンティティに `AWSApplicationAutoScalingCustomResourcePolicy` をアタッチすることはできません。このポリシーは、Application Auto Scaling がユーザーに代わって API Gateway 経由で利用できるカスタムリソースと CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。

許可の詳細

`AWSServiceRoleForApplicationAutoScaling_CustomResource` サービスリンクロール許可ポリシーは、Application Auto Scaling がすべての関連リソース (「リソース」:「*」) で以下のアクションを完了することを許可します。

- アクション: `execute-api:Invoke`
- アクション: `cloudwatch:DescribeAlarms`
- アクション: `cloudwatch:PutMetricAlarm`
- アクション: `cloudwatch:DeleteAlarms`

AWS マネージドポリシーに対する Application Auto Scaling 更新

Application Auto Scaling がこれらの変更の追跡を開始してからの、このサービス用の AWS マネージドポリシーに対する更新の詳細を確認します。このページへの変更に関する自動アラートを受け取るには、Application Auto Scaling のドキュメント履歴ページで RSS フィードにサブスクライブしてください。

変更	説明	Date
Application Auto Scaling は Neptune ポリシーを追加します	Application Auto Scaling が、Nepture 向けの新しいマネージドポリシーを追加しました。このポリシーは、Application Auto Scaling がユーザーに代わって Nepture と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。	2021 年 10 月 6 日

変更	説明	Date
Application Auto Scaling が ElastiCache for Redis ポリシーを追加	Application Auto Scaling が、ElastiCache 向けの新しいマネージドポリシーを追加しました。このポリシーは、Application Auto Scaling がユーザーに代わって ElastiCache と CloudWatch を呼び出し、スケーリングを実行することを許可するサービスリンクロールにアタッチされます。	2021 年 8 月 19 日
Application Auto Scaling が変更の追跡を開始	Application Auto Scaling が、その AWS マネージドポリシーに対する変更の追跡を開始しました。	2021 年 8 月 19 日

Application Auto Scaling 用のサービスリンクロール

Application Auto Scaling は、ユーザーに代わって AWS のその他サービスを呼び出す上で必要な許可のために、サービスリンクロールを使用します。サービスリンクロールは、AWS のサービスに直接リンクされた特殊なタイプの AWS Identity and Access Management (IAM) ロールです。サービスリンクロールは、AWS のサービスに許可を委任するためのセキュアな方法を提供します。これは、リンクされたサービスのみが、サービスリンクロールを引き受けることができるためです。

目次

概要

Application Auto Scaling と統合されるサービスについては、Application Auto Scaling がユーザーのためにサービスリンクロールを作成します。サービスリンクロールはサービスごとに 1 つあります。サービスリンクロールはそれぞれ、指定されたサービスプリンシパルを信頼してそのロールを継承します。詳細については、「Application Auto Scaling を使用できる AWS のサービス (p. 7)」を参照してください。

Application Auto Scaling は、各サービスリンクロールに必要な許可のすべてを含めます。これらのマネージド許可は、Application Auto Scaling によって作成および管理され、各リソースタイプに対して許可されるアクションを定義します。各ロールが付与する許可の詳細については、「Application Auto Scaling 用の AWS マネージドポリシー (p. 88)」を参照してください。

以下のセクションでは、Application Auto Scaling サービスリンクロールを作成し、管理する方法について説明します。これは、IAM エンティティ (ユーザー、グループ、またはロールなど) がサービスリンクロールの作成、編集、または削除を行うための許可を設定することから始めます。

サービスリンクロールの作成に必要な許可

Application Auto Scaling には、AWS アカウントのユーザーが所定のサービスの `RegisterScalableTarget` を初めて呼び出すときにサービスリンクロールを作成するための許可が必要です。Application Auto Scaling は、アカウントにターゲットサービス用のサービスリンクロールが既に存在しない場合、そのロールを作成します。サービスリンクロールは Application Auto Scaling に許可を付与して、ユーザーに代わってターゲットサービスを呼び出すことができるようにします。

この自動ロール作成が正常に行われるには、ユーザーが `iam:CreateServiceLinkedRole` アクションに対する許可を持っている必要があります。

```
"Action": "iam:CreateServiceLinkedRole"
```

以下は、スポットフリート用のサービスリンクロールを作成することを IAM ユーザーまたはロールに許可する許可ポリシーです。以下にあるように、サービスリンクロールは ARN としてポリシーの `Resource` フィールドに指定し、サービスリンクロールのサービスプリンシパルは条件として指定できます。各サービスの ARN については、「サービスリンクロールの ARN リファレンス (p. 98)」を参照してください。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/ec2.application-
autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-autoscaling.amazonaws.com"
                }
            }
        }
    ]
}
```

> **Note**
>
> `iam:AWSServiceName` IAM 条件キーは、ロールがアタッチされるサービスプリンシパルを指定するもので、このポリシー例では *`ec2.application-autoscaling`*`.amazonaws.com` として記述されています。サービスプリンシパルを推測しようとしないでください。サービスのサービスプリンシパルを確認するには、「Application Auto Scaling を使用できる AWS のサービス (p. 7)」を参照してください。

サービスリンクロールを作成する (自動)

サービスリンクロールを手動で作成する必要はありません。Application Auto Scaling は、ユーザーが `RegisterScalableTarget` を呼び出す時に、適切なサービスリンクロールを作成します。例えば、Amazon ECS サービスのオートスケーリングをセットアップする場合は、Application Auto Scaling が `AWSServiceRoleForApplicationAutoScaling_ECSService` ロールを作成します。

サービスリンクロールを作成する (手動)

サービスリンクロールを作成するには、IAM コンソール、AWS CLI、または IAM API を使用できます。詳細については、IAM ユーザーガイドの「サービスリンクロールの作成」を参照してください。

サービスリンクロールの作成 (AWS CLI)

以下の create-service-linked-role CLI コマンドを使用して、Application Auto Scaling サービスリンクロールを作成します。リクエストでは、サービス名の「prefix」を指定します。

サービス名のプレフィックスを確認するには、「Application Auto Scaling を使用できる AWS のサービス (p. 7)」セクションで、各サービス用のサービスリンクロールのサービスプリンシパルに関する情報を参照してください。サービス名とサービスプリンシパルは同じプレフィックスを共有します。例えば、AWS Lambda サービスリンクロールを作成するには、`lambda.application-autoscaling.amazonaws.com` を使用します。

```
aws iam create-service-linked-role --aws-service-name prefix.application-autoscaling.amazonaws.com
```

サービスリンクロールを編集する

Application Auto Scaling によって作成されたサービスリンクロールで編集できるのは、それらの説明のみです。詳細については、IAM ユーザーガイドの「サービスリンクロールの編集」を参照してください。

サービスリンクロールを削除する

サポートされているサービスで Application Auto Scaling を使用しなくなった場合は、対応するサービスリンクロールを削除することをお勧めします。

サービスリンクロールは、関連する AWS リソースを削除した後でしか削除できません。これは、リソースに対する Application Auto Scaling 許可を誤って取り消すことがないようにします。詳細については、スケーラブルリソースのドキュメントを参照してください。例えば、Amazon ECS サービスを削除するには、Amazon Elastic Container Service デベロッパーガイドの「サービスの削除」を参照してください。

サービスリンクロールは、IAM を使用して削除できます。詳細については、IAM ユーザーガイド の「サービスにリンクされたロールの削除」を参照してください。

サービスリンクロールの削除後に `RegisterScalableTarget` を呼び出すと、Application Auto Scaling がそのロールを再度作成します。

Application Auto Scaling サービスリンクロールがサポートされるリージョン

Application Auto Scaling は、このサービスを利用できるすべての AWS リージョンでサービスリンクロールの使用をサポートします。

サービスリンクロールの ARN リファレンス

サービス	ARN
AppStream 2.0	`arn:aws:iam::012345678910:role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`
Aurora	`arn:aws:iam::012345678910:role/aws-service-role/rds.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_RDSCluster`
Comprehend	`arn:aws:iam::012345678910:role/aws-service-role/comprehend.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ComprehendEndpoint`

サービス	ARN
DynamoDB	`arn:aws:iam::012345678910:role/aws-service-role/dynamodb.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_DynamoDBTable`
ECS	`arn:aws:iam::012345678910:role/aws-service-role/ecs.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ECSService`
ElastiCache	`arn:aws:iam::012345678910:role/aws-service-role/elasticache.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_ElastiCacheRG`
Keyspaces	`arn:aws:iam::012345678910:role/aws-service-role/cassandra.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CassandraTable`
Lambda	`arn:aws:iam::012345678910:role/aws-service-role/lambda.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_LambdaConcurrency`
MSK	`arn:aws:iam::012345678910:role/aws-service-role/kafka.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_KafkaCluster`
Neptune	`arn:aws:iam::012345678910:role/aws-service-role/neptune.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_NeptuneCluster`
SageMaker	`arn:aws:iam::012345678910:role/aws-service-role/sagemaker.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_SageMakerEndpoint`
Spot Fleets	`arn:aws:iam::012345678910:role/aws-service-role/ec2.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest`
カスタムリソース	`arn:aws:iam::012345678910:role/aws-service-role/custom-resource.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_CustomResource`

Note

AWS CloudFormation スタックテンプレートにある AWS::ApplicationAutoScaling::ScalableTarget リソースの `RoleARN` プロパティには、指定するサービスリンクロールがまだ存在しない場合でも、サービスリンクロールの ARN を指定できます。Application Auto Scaling が、そのロールを自動的に作成します。

Application Auto Scaling のアイデンティティベースポリシー例

デフォルトで、新しい IAM ユーザーには、何かを実行する許可が一切ありません。IAM 管理者は、スケーリングポリシーの設定などの Application Auto Scaling API アクションを実行するための許可を、ユーザーとロールに付与する IAM ポリシーを作成する必要があります。作成後、管理者は、許可を必要とする IAM ユーザーまたはロールにこれらのポリシーをアタッチする必要があります。

以下のサンプル JSON ポリシードキュメントを使用して IAM ポリシーを作成する方法については、IAM ユーザーガイドの「[JSON] タブでのポリシーの作成」を参照してください。

目次

Application Auto Scaling API アクションに必要な許可

以下のポリシーは、Application Auto Scaling API の呼び出し時に、一般的なユースケースに対して許可を付与します。アクセスコントロール (p. 85) をセットアップする時、および IAM ユーザーまたはロールにアタッチできる許可ポリシーを記述する時は、このセクションを参照してください。各ポリシーは、Application Auto Scaling API アクションのすべて、または一部に対するアクセス権を付与します。IAM ユーザーまたはロールに、ターゲットサービスと CloudWatch に対する許可ポリシーがあることを確認する必要もあります (詳細については、次のセクションを参照してください)。

以下の許可ポリシーは、すべての Application Auto Scaling API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:*"
            ],
            "Resource": "*"
        }
    ]
}
```

以下の許可ポリシーは、スケジュールされたアクションではなく、スケーリングポリシーを設定するために必要なすべての Application Auto Scaling API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScalingPolicy",
              "application-autoscaling:DescribeScalingPolicies",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScalingPolicy"
            ],
            "Resource": "*"
        }
    ]
}
```

以下の許可ポリシーは、スケーリングポリシーではなく、スケジュールされたアクションを設定するために必要なすべての Application Auto Scaling API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "application-autoscaling:RegisterScalableTarget",
              "application-autoscaling:DescribeScalableTargets",
              "application-autoscaling:DeregisterScalableTarget",
              "application-autoscaling:PutScheduledAction",
              "application-autoscaling:DescribeScheduledActions",
              "application-autoscaling:DescribeScalingActivities",
              "application-autoscaling:DeleteScheduledAction"
            ],
            "Resource": "*"
        }
    ]
}
```

ターゲットサービスと CloudWatch での API アクションに必要な許可

ターゲットサービスで Application Auto Scaling を正常に設定して使用するには、Amazon CloudWatch、およびスケーリングを設定する各ターゲットサービスに対する必要な許可を IAM ユーザーに付与する必要があります。以下のポリシーを使用して、ターゲットサービスと CloudWatch での作業に必要な最小限の許可をユーザーに付与します。

目次

AppStream 2.0 フリート

以下の許可ポリシーは、必要とされるすべての AppStream 2.0 および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "appstream:DescribeFleets",
              "appstream:UpdateFleet",
              "cloudwatch:DescribeAlarms",
```

```
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Aurora レプリカ

以下の許可ポリシーは、必要とされるすべての Aurora および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DeleteDBInstance",
              "rds:DescribeDBClusters",
              "rds:DescribeDBInstances",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Comprehend ドキュメントの分類とエンティティ認識のエンドポイント

以下の許可ポリシーは、必要とされるすべての Amazon Comprehend および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "comprehend:UpdateEndpoint",
              "comprehend:DescribeEndpoint",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

DynamoDB テーブルとグローバルセカンダリインデックス

以下の許可ポリシーは、必要とされるすべての DynamoDB および CloudWatch API アクションへのアクセス権を付与します。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "dynamodb:DescribeTable",
              "dynamodb:UpdateTable",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

ECS サービス

以下の許可ポリシーは、必要とされるすべての ECS および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "ecs:DescribeServices",
              "ecs:UpdateService",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

ElastiCache レプリケーショングループ

以下の許可ポリシーは、必要とされるすべての ElastiCache および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "elasticache:ModifyReplicationGroupShardConfiguration",
              "elasticache:IncreaseReplicaCount",
              "elasticache:DecreaseReplicaCount",
              "elasticache:DescribeReplicationGroups",
              "elasticache:DescribeCacheClusters",
              "elasticache:DescribeCacheParameters",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

Amazon EMR クラスター

以下の許可ポリシーは、必要とされるすべての Amazon EMR および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "elasticmapreduce:ModifyInstanceGroups",
              "elasticmapreduce:ListInstanceGroups",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Keyspaces テーブル

以下の許可ポリシーは、必要とされるすべての Amazon Keyspaces および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "cassandra:Select",
              "cassandra:Alter",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Lambda 関数

以下の許可ポリシーは、必要とされるすべての Lambda および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "lambda:PutProvisionedConcurrencyConfig",
              "lambda:GetProvisionedConcurrencyConfig",
              "lambda:DeleteProvisionedConcurrencyConfig",
              "cloudwatch:DescribeAlarms",
```

```
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Amazon Managed Streaming for Apache Kafka (MSK) ブローカーストレージ

以下の許可ポリシーは、必要とされるすべての Amazon MSK および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "kafka:DescribeCluster",
              "kafka:DescribeClusterOperation",
              "kafka:UpdateBrokerStorage",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

Neptune クラスター

以下の許可ポリシーは、必要とされるすべての Nepture および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "rds:AddTagsToResource",
              "rds:CreateDBInstance",
              "rds:DescribeDBInstances",
              "rds:DescribeDBClusters",
              "rds:DescribeDBClusterParameters",
              "rds:DeleteDBInstance",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

SageMaker エンドポイント

以下の許可ポリシーは、必要とされるすべての SageMaker および CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "sagemaker:DescribeEndpoint",
              "sagemaker:DescribeEndpointConfig",
              "sagemaker:UpdateEndpointWeightsAndCapacities",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

スポットフリート (Amazon EC2)

以下の許可ポリシーは、必要とされるすべてのスポットフリートおよび CloudWatch API アクションへのアクセス権を付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "ec2:DescribeSpotFleetRequests",
              "ec2:ModifySpotFleetRequest",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

カスタムリソース

以下の許可ポリシーは、API Gateway API 実行アクションに対する必要な許可をユーザーに付与します。このポリシーは、必要とされるすべての CloudWatch アクションへのアクセス権も付与します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
              "execute-api:Invoke",
              "cloudwatch:DescribeAlarms",
              "cloudwatch:PutMetricAlarm",
              "cloudwatch:DeleteAlarms"
            ],
            "Resource": "*"
        }
    ]
}
```

AWS Management Consoleで作業するための許可

Application Auto Scaling にスタンドアロンコンソールはありません。Application Auto Scaling と統合するほとんどのサービスには、それらのコンソールでスケーリングを設定することを目的とした機能があります。

多くの場合、各サービスは、そのコンソールへのアクセス権を定義する事前定義された AWS マネージド IAM ポリシーを提供し、これには Application Auto Scaling API アクションに対する許可が含まれます。詳細については、コンソールを使用するサービスのドキュメントを参照してください。

AWS Management Consoleで特定の Application Auto Scaling アクションを表示して使用するためのきめ細かな許可をユーザーに付与する、独自のカスタム IAM ポリシーを作成することもできます。前のセクションのサンプルポリシーを使用することが可能ですが、これらは AWS CLI または SDK を使用して行われるリクエスト向けに設計されています。コンソールではこの機能を実行するために追加の API アクションを使用するので、これらのポリシーは正常に動作しない可能性があります。例えば、ステップスケーリングを設定するには、CloudWatch アラームを作成して管理するための追加の許可がユーザーに必要となる場合があります。

> **Tip**
>
> コンソールでタスクを実行するために必要な API アクションを探すには、AWS CloudTrail などのサービスを使用できます。詳細については、AWS CloudTrail ユーザーガイドを参照してください。

以下は、ユーザーがスポットフリートのスケーリングポリシーを設定することを許可する許可ポリシーの例です。コンソールからフリートスケーリング設定にアクセスする IAM ユーザーには、スポットフリートに対する IAM 許可に加えて、動的スケーリングをサポートするサービスに対する適切な許可も必要です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:*",
                "ec2:DescribeSpotFleetRequests",
                "ec2:ModifySpotFleetRequest",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarmHistory",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:DescribeAlarmsForMetric",
                "cloudwatch:GetMetricStatistics",
                "cloudwatch:ListMetrics",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DisableAlarmActions",
                "cloudwatch:EnableAlarmActions",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Get*",
                "sns:List*"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:CreateServiceLinkedRole",
            "Resource": "arn:aws:iam::*:role/aws-service-role/ec2.application-
autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_EC2SpotFleetRequest",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName":"ec2.application-autoscaling.amazonaws.com"
                }
```

```
            }
        }
    ]
}
```

このポリシーは、ユーザーが Amazon EC2 コンソールでスケーリングポリシーを表示して変更し、CloudWatch コンソールで CloudWatch アラームを作成して管理することを可能にします。

API アクションを調整して、ユーザーアクセスを制限できます。例えば、`application-autoscaling:*` を `application-autoscaling:Describe*` に置き換えると、ユーザーには読み取り専用アクセスが与えられます。

また、必要に応じて CloudWatch 許可を調整して、CloudWatch 機能へのユーザーアクセスを制限することもできます。詳細については、Amazon CloudWatch ユーザーガイドの「CloudWatch コンソールの使用に必要な許可」を参照してください。

Application Auto Scaling へのアクセスのトラブルシューティング

Application Auto Scaling の使用時に `AccessDeniedException` または同様の問題が発生する場合は、このセクションの情報を参考にしてください。

Application Auto Scaling でアクションを実行する権限がありません

AWS API オペレーションの呼び出し時に `AccessDeniedException` を受け取った場合は、使用している AWS Identity and Access Management (IAM) ユーザーまたはロールの認証情報に、その呼び出しを行うために必要な許可がないことを意味します。

以下のサンプルエラーは、`mateojackson` IAM ユーザーがスケーラブルターゲットに関する詳細を表示しようとしているが、`application-autoscaling:DescribeScalableTargets` 許可を持っていないという場合に発生します。

```
An error occurred (AccessDeniedException) when calling the DescribeScalableTargets
 operation: User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 application-autoscaling:DescribeScalableTargets
```

このエラー、または同様のエラーが発生する場合は、管理者に問い合わせてサポートを受ける必要があります。

アカウントの管理者は、ユーザーの IAM ユーザーまたはロールに、Application Auto Scaling がターゲットサービスと CloudWatch のリソースへのアクセスに使用するすべての API アクションに対する許可があることを確認する必要があります。どのリソースで作業を行っているかに応じて、異なる許可が必要になります。Application Auto Scaling には、ユーザーが所定のリソースに対するスケーリングを初めて設定するときにサービスリンクロールを作成するための許可も必要です。

管理者として Application Auto Scaling へのアクセスを他のユーザーに許可したいと考えています

Application Auto Scaling へのアクセスを他のユーザーに許可するには、アクセス権が必要なユーザーまたはアプリケーション用の IMA エンティティ (ユーザーまたはロール) を作成する必要があります。ユーザーまたはアプリケーションは、このエンティティの認証情報を使用して AWS にアクセスします。次に、Application Auto Scaling での適切な許可を付与するポリシーをエンティティにアタッチする必要があります。

これを開始するには、IAM ユーザーガイドの「IAM が委任した最初のユーザーおよびユーザーグループの作成」を参照してください。

管理者ですが、IAM ポリシーからエラーが返される、またはポリシーが期待どおりに動作しません

IAM 許可ポリシーは、Application Auto Scaling API アクションに必要な IAM 許可に加えて、ターゲットサービスと CloudWatch を呼び出すためのアクセス権も付与する必要があります。

ユーザーまたはアプリケーションに適切な IAM ポリシー許可がない場合は、それらのアクセスが予期せず拒否される可能性があります。アカウントのユーザーとアプリケーション用の許可ポリシーを記述するには、「Application Auto Scaling のアイデンティティベースポリシー例 (p. 99)」の情報を参考にしてください。

検証の実行方法については、「ターゲットリソースでの API コールに対する許可の検証 (p. 109)」を参照してください。

一部の許可問題は、Application Auto Scaling が使用するサービスリンクロールの作成に関する問題に起因する可能性があることに注意してください。これらのサービスリンクロールの作成については、「Application Auto Scaling 用のサービスリンクロール (p. 96)」を参照してください。

ターゲットリソースでの API コールに対する許可の検証

Application Auto Scaling API アクションに対して認可されたリクエストを実行するには、API の呼び出し元が、ターゲットサービスと CloudWatch 内の AWS リソースにアクセスする許可を持っている必要があります。Application Auto Scaling は、リクエストを続行する前に、ターゲットサービスと CloudWatch の両方に関連付けられているリクエストに対する許可を検証します。これを行うには、一連のコールを発行してターゲットリソースに対する IAM 許可を検証します。レスポンスが返されると、Application Auto Scaling がそのレスポンスを読み取ります。IAM 許可が所定のアクションが許可しない場合、Application Auto Scaling はリクエストを失敗させ、欠落している許可に関する情報が含まれたエラーをユーザーに返します。これは、ユーザーがデプロイするスケーリング設定が意図したとおりに機能することと、リクエストが失敗した場合に有用なエラーが返されることを確実にします。

以下の情報は、この仕組みの例として、Application Auto Scaling が Aurora と CloudWatch で許可の検証を実行する方法を詳しく説明します。

IAM ユーザーが Aurora DB クラスターに対して `RegisterScalableTarget` API を呼び出すと、Application Auto Scaling は以下のすべてのチェックを実行して、IAM ユーザーに必要な許可 (太字) があることを確認します。

- rds:CreateDBInstance: ユーザーにこの許可があるかどうかを判断するため、`CreateDBInstance` API オペレーションにリクエストを送信して、ユーザーが指定した Aurora DB クラスターで無効なパラメータ (空のインスタンス ID) を使った DB インスタンスの作成を試みます。許可があるユーザーの場合、API は、リクエストを監査した後で `InvalidParameterValue` エラーコードレスポンスを返します。しかし、許可がないユーザーの場合は、`AccessDenied` エラーが発生し、欠落している許可がリストされた、ユーザーへの `ValidationException` エラーを伴って Application Auto Scaling リクエストが失敗します。
- rds:DeleteDBInstance: `DeleteDBInstance` API オペレーションに空のインスタンス ID を送信します。許可があるユーザーの場合、このリクエストの結果は `InvalidParameterValue` エラーになります。許可がないユーザーの場合は、結果が `AccessDenied` になり、ユーザーに検証例外が送信されます (最初の箇条書きで説明されているものと同じ対応)。
- rds:AddTagsToResource: `AddTagsToResource` API オペレーションには Amazon リソースネーム (ARN) が必要であるため、無効なアカウント ID (12345) とダミーインスタンス ID (non-existing-db) を使用した「ダミー」リソースを指定して ARN (`arn:aws:rds:us-east-1:12345:db:non-`

existing-db) を作成する必要があります。許可があるユーザーの場合、このリクエストの結果は InvalidParameterValue エラーになります。許可がないユーザーの場合は、結果が AccessDenied になり、ユーザーに検証例外が送信されます

- rds:DescribeDBCluster: オートスケーリングに登録されているリソースのクラスター名を記述します。許可があるユーザーの場合、有効な記述結果が得られます。許可がないユーザーの場合は、結果が AccessDenied になり、ユーザーに検証例外が送信されます
- rds:DescribeDBInstance。スケーラブルターゲットを登録するためにユーザーが提供したクラスター名をフィルタリングする db-cluster-id フィルターを使って、DescribeDBInstance API を呼び出します。許可があるユーザーの場合、DB クラスター内のすべての DB インスタンスを記述することが許可されます。許可がないユーザーの場合は、この呼び出しの結果が AccessDenied になり、ユーザーに検証例外が送信されます
- cloudwatch:PutMetricAlarm: パラメータなしで PutMetricAlarm API を呼び出します。アラーム名が欠落してるため、リクエストの結果は、許可があるユーザーに対する ValidationError になります。許可がないユーザーの場合は、結果が AccessDenied になり、ユーザーに検証例外が送信されます
- cloudwatch:DescribeAlarms: 最大レコード数の値を 1 に設定して DescribeAlarms API を呼び出します。許可があるユーザーの場合、レスポンスに 1 つのアラームに関する情報があることを期待できます。許可がないユーザーの場合は、この呼び出しの結果が AccessDenied になり、ユーザーに検証例外が送信されます
- cloudwatch:DeleteAlarms: 上記の PutMetricAlarm と同じく、DeleteAlarms リクエストにパラメータを指定しません。リクエストにアラーム名がないため、この呼び出しは、許可があるユーザーに対する ValidationError を伴って失敗します。許可がないユーザーの場合は、結果が AccessDenied になり、ユーザーに検証例外が送信されます

これらの検証例外は、そのうちのどれかが発生するたびにログに記録されます。AWS CloudTrail を使用して、どのコールが検証に失敗したかを手動で特定する手順を実行できます。詳細については、AWS CloudTrail ユーザーガイドを参照してください。

> Note
>
> CloudTrail を使用して Application Auto Scaling イベントのアラートを受信した場合、これらのアラートには、ユーザーアクセス許可を検証するための Application Auto Scaling に対するコールがデフォルトで含まれています。これらのアラートを除外する場合は、これらの検証チェックのための application-autoscaling.amazonaws.com が含まれている invokedBy フィールドを使用します。

Application Auto Scaling のコンプライアンス検証

サードパーティーの監査人は、SOC、PCI、FedRAMP、HIPAA など複数の AWS コンプライアンス プログラムのパートとして、AWS のサービス のセキュリティとコンプライアンスを評価します。

Application Auto Scaling やその他の AWS のサービス が特定のコンプライアンスプログラムの対象であるかどうかを確認するには、「 のコンプライアンスプログラムによる対象範囲内の AWS サービス」を参照してください。一般的な情報については、「AWS コンプライアンスプログラム」を参照してください。

AWS Artifact を使用して、サードパーティーの監査レポートをダウンロードできます。詳細については、「AWS Artifact におけるダウンロードレポート」を参照してください。

AWS のサービス を使用する際のユーザーのコンプライアンス責任は、ユーザーのデータの機密性や貴社のコンプライアンス目的、適用される法律および規制によって決まります。AWS では、コンプライアンスに役立つ次のリソースを提供しています。

- 「Security and Compliance Quick Start Guides」(セキュリティおよびコンプライアンスのクイックスタートガイド) - これらのデプロイガイドでは、アーキテクチャ上の考慮事項について説明し、セキュリティとコンプライアンスを重視したベースライン環境を AWS でデプロイするステップを説明します。

- Architecting for HIPAA Security and Compliance Whitepaper (HIPAA のセキュリティとコンプライアンスのためのアーキテクチャの設計ホワイトペーパー) – このホワイトペーパーは、企業が AWS を使用して HIPAA 対象アプリケーションを作成する方法を説明します。

  Note

  すべての AWS のサービス が HIPAA 適格であるわけではありません。詳細については、「HIPAA 対応サービスのリファレンス」を参照してください。
- AWS コンプライアンスのリソース– このワークブックおよびガイドのコレクションは、お客様の業界と拠点に適用されるものである場合があります。
- 「AWS Config デベロッパーガイド」の「ルールでのリソースの評価」 — AWS Config のサービスでは、自社のプラクティス、業界ガイドライン、および規制に対するリソースの設定の準拠状態を評価します。
- AWS Security Hub — この AWS のサービス では、AWS 内のセキュリティ状態が包括的に示されており、セキュリティ業界の標準およびベストプラクティスへの準拠の確認に役立ちます。
- AWS Audit Manager - この AWS のサービス は AWS の使用状況を継続的に監査し、リスクの管理方法やコンプライアンスを業界スタンダードへの準拠を簡素化するために役立ちます。

Application Auto Scaling の耐障害性

AWS のグローバルインフラストラクチャは AWS リージョンとアベイラビリティーゾーンを中心として構築されます。

AWS リージョンには、低レイテンシー、高いスループット、そして高度の冗長ネットワークで接続されている複数の物理的に独立・隔離されたアベイラビリティーゾーンがあります。

アベイラビリティーゾーンを使用すると、中断することなくゾーン間で自動的にフェイルオーバーするアプリケーションとデータベースを設計および運用できます。アベイラビリティーゾーンは、従来の単一または複数のデータセンターインフラストラクチャよりも可用性、耐障害性、および拡張性が優れています。

AWS リージョンとアベイラビリティーゾーンの詳細については、「AWS グローバルインフラストラクチャ」を参照してください。

Application Auto Scaling のインフラストラクチャセキュリティ

マネージドサービスである Application Auto Scaling は、Amazon Web Services: セキュリティプロセスの概要ホワイトペーパーで説明されている AWS グローバルネットワークセキュリティ手順で保護されています。

ユーザーは、AWS が公開した API コールを使用して、ネットワーク経由で Application Auto Scaling にアクセスします。クライアントで Transport Layer Security (TLS) 1.0 以降がサポートされている必要があります。TLS 1.2 以降が推奨されています。また、Ephemeral Diffie-Hellman (DHE)や Elliptic Curve Ephemeral Diffie-Hellman (ECDHE) などの Perfect Forward Secrecy (PFS)を使用した暗号スイートもクライアントでサポートされている必要があります。これらのモードは、Java 7 以降など、最近のほとんどのシステムでサポートされています。

また、リクエストは、アクセスキー ID と、IAM プリンシパルに関連付けられているシークレットアクセスキーを使用して署名する必要があります。または、AWS Security Token Service (AWS STS) を使用して、一時的なセキュリティ認証情報を生成し、リクエストに署名することもできます。

Application Auto Scaling のクォータ

AWS アカウント アカウントには、AWS のサービスごとにデフォルトのクォータ (以前は制限と呼ばれていました) があります。特に明記されていない限り、クォータはリージョンごとに存在します。一部のクォータについては引き上げをリクエストできますが、その他のクォータについてはリクエストできません。

Application Auto Scaling のクォータを表示するには、Service Quotas コンソールを開きます。ナビゲーションペインで、[AWS services] (AWS のサービス)、[Application Auto Scaling] の順に選択します。

クォータの引き上げをリクエストするには、「Service Quotas ユーザーガイド」の「クォータ引き上げリクエスト」を参照してください。Service Quotas でクォータがまだ利用できない場合は、Application Auto Scaling の制限フォームを使用してください。引き上げのリクエストには、リソースのタイプ (Amazon ECS または DynamoDB など) を指定するようにしてください。

AWS アカウント には、Application Auto Scaling に関連する次のクォータがあります。

各アカウントのリージョンあたりのデフォルトクォータ

項目	デフォルト	調整可能
リソースタイプごとにスケーラブルなターゲットの最大数	デフォルトのクォータは、リソースタイプに応じて異なります。 Amazon DynamoDB スケーラブルターゲットは最大 5,000 個、ECS スケーラブルターゲットは最大 3,000 個、その他すべてのリソースタイプのスケーラブルターゲットはそれぞれ最大 500 個です。	あり
スケーラブルなターゲットあたりのスケーリングポリシーの最大数	50 これには、ステップスケーリングポリシーとターゲット追跡ポリシーの両方が含まれます。	なし
スケーラブルなターゲットあたりのスケジュールされたアクションの最大数	200	なし
ステップスケーリングポリシーあたりのステップ調整値の最大数	20	なし

ワークロードをスケールアウトする際は、サービスのクォータを念頭に置いてください。例えば、サービスで許可されるキャパシティーユニットの最大数に達すると、スケールアウトは停止します。需要が低下し、現行の容量が減少すると、Application Auto Scaling が再びスケールアウトできるようになります。この容量制限に再度到達しないようにするために、引き上げをリクエストします。各サービスには、リソースの最大容量に対する独自のデフォルトのクォータがあります。AWS のその他サービスのデフォル

トクォータについては、Amazon Web Services 全般のリファレンスの「サービスエンドポイントとクォータ」を参照してください。

AWS CloudFormation を使用した Application Auto Scaling リソースの作成

Application Auto Scaling は、リソースとインフラストラクチャの作成と管理に費やす時間を削減できるように、AWS リソースのモデル化とセットアップを支援するサービスである AWS CloudFormation と統合されています。必要な AWS リソースのすべてを記述するテンプレートを作成すると、AWS CloudFormation がユーザーに代わってこれらのリソースのプロビジョンと設定を行います。

AWS CloudFormation を使用するときは、テンプレートを再利用して、Application Auto Scaling リソースを一貫的に繰り返しセットアップできます。リソースを一度記述するだけで、同じリソースを複数の AWS アカウント とリージョンで何度でもプロビジョニングできます。

Application Auto Scaling と AWS CloudFormation テンプレート

Application Auto Scaling と関連サービスのリソースをプロビジョニングして設定するには、AWS CloudFormation テンプレートを理解しておく必要があります。テンプレートは、JSON または YAML 形式のテキストファイルです。これらのテンプレートには、AWS CloudFormation スタックにプロビジョニングしたいリソースを記述します。JSONやYAMLに不慣れな方は、AWS CloudFormation Designer を使えば、 AWS CloudFormation テンプレートを使いこなすことができます。詳細については、AWS CloudFormation ユーザーガイドの「AWS CloudFormationDesigner とは」を参照してください。

Application Auto Scaling リソースのスタックテンプレートを作成するときは、以下を指定する必要があります。

- ターゲットサービスの名前空間 (**`appstream`** など)。サービス名前空間を入手するには、「AWS::ApplicationAutoScaling::ScalableTarget」リファレンスを参照してください。
- ターゲットリソースに関連付けられているスケーラブルディメンション (**`appstream:fleet:DesiredCapacity`** など)。スケーラブルディメンションを入手するには、「AWS::ApplicationAutoScaling::ScalableTarget」リファレンスを参照してください。
- ターゲットリソースのリソース ID (**`fleet/sample-fleet`** など)。特定のリソース ID の構文と例については、「AWS::ApplicationAutoScaling::ScalableTarget」リファレンスを参照してください。
- ターゲットリソース用のサービスリンクロール (**`arn:aws:iam::012345678910:role/aws-service-role/appstream.application-autoscaling.amazonaws.com/AWSServiceRoleForApplicationAutoScaling_AppStreamFleet`** など)。ロール ARN を入手するには、「サービスリンクロールの ARN リファレンス (p. 98)」の表を参照してください。

Application Auto Scaling リソースの詳細については、AWS CloudFormation ユーザーガイドの Application Auto Scaling リファレンスを参照してください。

サンプルテンプレートスニペット

AWS では、スタックテンプレートでさまざまなスケーリングポリシーとスケジュールされたアクションを宣言する方法を理解するために使用できる、JSON および YAML のテンプレートスニペットをいくつかご

用意しています。詳細については、AWS CloudFormation ユーザーガイドの Application Auto Scaling テンプレートの例のセクションを参照してください。その他の例については、AWS CloudFormation ユーザーガイドのApplication Auto Scalingの「例」のセクションを参照してください。

AWS CloudFormation の詳細はこちら

AWS CloudFormation の詳細については、以下のリソースを参照してください。

- AWS CloudFormation
- AWS CloudFormation ユーザーガイド
- AWS CloudFormation API リファレンス
- AWS CloudFormation コマンドラインインターフェイスユーザーガイド

ドキュメント履歴

以下の表は、2018 年 1 月以降の Application Auto Scaling ドキュメントへの重要な追加項目をまとめたものです。このドキュメントの更新に関する通知については、RSS フィードにサブスクライブできます。

update-history-change	update-history-description	update-history-date
ガイドの変更点 (p. 116)	クォータに関するドキュメントのリソースタイプエントリあたりのスケーラブルターゲットの最大数が更新されました。「Quotas for Application Auto Scaling」(Application Auto Scaling のクォータ) を参照してください。	2022 年 5 月 6 日
Amazon Neptune クラスターのサポートを追加 (p. 116)	アプリケーションの Auto Scaling を使用して、Amazon Neptune DB クラスター内のレプリカ数をスケーリングします。詳細については、「Amazon Neptune とApplication Auto Scaling」を参照してください。。トピック Application Auto Scaling が更新されるAWSマネージドポリシーが更新され、Neptune との統合に関する新しい管理ポリシーが一覧表示されました。	2021 年 10 月 6 日
Application Auto Scaling がその AWS マネージドポリシーへの変更の報告を開始 (p. 116)	2021 年 8 月 19 日より、「AWS マネージドポリシーに対する Application Auto Scaling 更新」トピックでマネージドポリシーへの変更が報告されるようになります。リストされている最初の変更は、ElastiCache for Redis に必要な許可の追加です。	2021 年 8 月 19 日
ElastiCache for Redis replication レプリケーショングループのサポートを追加 (p. 116)	Application Auto Scaling を使用して、ElastiCache for Redis レプリケーショングループ (クラスター) のノードグループ数と、ノードグループあたりのレプリカ数をスケールします。詳細については、「ElastiCache for Redis と Application Auto Scaling」を参照してください。	2021 年 8 月 19 日
ガイドの変更点 (p. 116)	Application Auto Scaling ユーザーガイドの新しい IAM トピックは、Application Auto Scaling へのアクセスのトラブルシューティングに役立ちます。詳細については、「Application Auto Scaling の Identity and Access Management」を参照してくださ	2021 年 2 月 23 日

	い。また、ターゲットサービスと Amazon CloudWatch でのアクションに対する新しい IAM 許可ポリシーの例も追加されました。詳細については、「AWS CLI または SDK を使用するためのサンプルポリシー」を参照してください。	
ローカルタイムゾーンのサポートを追加 (p. 116)	ローカルタイムゾーンでスケジュールされたアクションを作成できるようになりました。タイムゾーンが夏時間を実施する場合は、夏時間 (DST) に合わせて自動的に調整されます。詳細については、「スケジュールされたスケーリング」を参照してください。	2021 年 2 月 2 日
ガイドの変更点 (p. 116)	Application Auto Scaling ユーザーガイドの新しいチュートリアルは、Application Auto Scaling の使用時に、ターゲット追跡スケーリングポリシーとスケジュールされたスケーリングを使用してアプリケーションの可用性を向上させる方法を理解するために役立ちます。また、新しいトピックでは、注意が必要となる可能性がある問題を Amazon CloudWatch が検出したときに通知をトリガーする方法も説明されています。	2020 年 10 月 15 日
Amazon Managed Streaming for Apache Kafka クラスターストレージのサポートを追加 (p. 116)	ターゲット追跡スケーリングポリシーを使用して、Amazon MSK クラスターに関連付けられているブローカーストレージの量をスケールアウトします。	2020 年 9 月 30 日
Amazon Comprehend エンティティ認識器エンドポイントのサポートを追加 (p. 116)	Application Auto Scaling を使用して、Amazon Comprehend エンティティ認識器エンドポイントにプロビジョニングされた推論単位の数をスケールします。	2020 年 9 月 28 日
Amazon Keyspaces (Apache Cassandra 用) テーブルのサポートを追加 (p. 116)	Application Auto Scaling を使用して、Amazon Keyspaces テーブルのプロビジョニングされたスループット (読み込みキャパシティーと書き込みキャパシティー) をスケールします。	2020 年 4 月 23 日

新しい「セキュリティ」章 (p. 116)	Application Auto Scaling ユーザーガイドの新しい「セキュリティ」章は、Application Auto Scaling の使用時に責任共有モデルを適用する方法を理解するために役立ちます。この更新の一環として、ユーザーガイドの「認証とアクセスコントロール」章が、新しいより有益な「Application Auto Scaling の Identity and Access Management」セクションに置き換えられました。	2020 年 1 月 16 日
マイナーな更新 (p. 116)	さまざまな改善と修正。	2020 年 1 月 15 日
通知機能の追加 (p. 116)	Application Auto Scaling が、特定のアクションが発生したときにイベントを Amazon EventBridge に送信し、通知を AWS Health Dashboard に通知を送信するようになりました。詳細については、「Application Auto Scaling のモニタリング」を参照してください。	2019 年 12 月 20 日
AWS Lambda 関数のサポートを追加 (p. 116)	Application Auto Scaling を使用して、Lambda 関数のプロビジョニングされた同時実行数をスケールします。	2019 年 12 月 3 日
Amazon Comprehend ドキュメント分類エンドポイントのサポートを追加 (p. 116)	Application Auto Scaling を使用して、Amazon Comprehend ドキュメント分類エンドポイントのスループット容量をスケールします。	2019 年 11 月 25 日
ターゲット追跡スケーリングポリシーに AppStream 2.0 サポートを追加 (p. 116)	ターゲット追跡スケーリングポリシーを使用して、AppStream 2.0 フリートのサイズをスケールします。	2019 年 11 月 25 日
Amazon VPC エンドポイントのサポート (p. 116)	VPC と Application Auto Scaling の間でプライベート接続を確立できるようになりました。移行の考慮事項と手順については、「Application Auto Scaling とインターフェイス VPC エンドポイント」を参照してください。	2019 年 11 月 22 日
スケーリングの一時停止と再開 (p. 116)	スケーリングの中断と再開のサポートが追加されました。詳細については、「Application Auto Scaling のスケーリングの一時停止と再開」を参照してください。	2019 年 8 月 29 日

新規セクション (p. 116)	「セットアップ」セクションが Application Auto Scaling ドキュメントに追加されました。マイナー改善と修正がユーザーガイドに加えられました。	2019 年 6 月 28 日
ガイドの変更点 (p. 116)	Application Auto Scaling ドキュメントの「スケジュールされたスケーリング」、「ステップスケーリングポリシー」、および「ターゲット追跡スケーリングポリシー」の各セクションを改善しました。	2019 年 3 月 11 日
カスタムリソースのサポートを追加 (p. 116)	Application Auto Scaling を使用して、独自のアプリケーションまたはサービスによって提供されるカスタムリソースをスケールします。詳細については、GitHub リポジトリを参照してください。	2018 年 7 月 9 日
SageMaker エンドポイントバリアントのサポートを追加 (p. 116)	Application Auto Scaling を使用して、バリアントに対してプロビジョニングされたエンドポイントインスタンスの数をスケールします。	2018 年 2 月 28 日

以下の表は、2018 年 1 月までの Application Auto Scaling ドキュメントへの重要な変更をまとめたものです。

変更	説明	Date
Aurora レプリカのサポートを追加	Application Auto Scaling を使用して、希望数をスケールします。詳細については、Amazon RDS ユーザーガイドの「Aurora レプリカでの Amazon Aurora Auto Scaling の使用」を参照してください。	2017 年 11 月 17 日
スケジュールに基づくスケーリングのサポートを追加	スケジュールに基づくスケーリングを使用して、事前設定された特定の日時または間隔でリソースをスケールします。詳細については、「Application Auto Scaling のスケジュールされたスケーリング」を参照してください。	2017 年 11 月 8 日
ターゲット追跡スケーリングポリシーのサポートを追加	ターゲットの追跡スケーリングポリシーを使用して、いくつかの簡単なステップでアプリケーションの動的スケーリングをセットアップします。詳細については、「Application Auto Scaling のターゲット追跡スケー	2017 年 7 月 12 日

変更	説明	Date
	リングポリシー」を参照してください。	
DynamoDB のテーブルとグローバルセカンダリインデックスのプロビジョニングされた読み込みキャパシティーと書き込みキャパシティーのサポートを追加	Application Auto Scaling を使用して、プロビジョニングされたスループット (読み込みキャパシティーと書き込みキャパシティー) をスケールします。詳細については、Amazon DynamoDB デベロッパーガイドの「DynamoDB Auto Scaling によるスループット容量の管理」を参照してください。	2017 年 6 月 14 日
AppStream 2.0 フリートのサポートを追加	Application Auto Scaling を使用して、フリートのサイズをスケールします。詳細については、Amazon AppStream 2.0 管理ガイドの「AppStream 2.0 向け Fleet Auto Scaling」を参照してください。	2017 年 3 月 23 日
Amazon EMR クラスターのサポートを追加	Application Auto Scaling を使用して、コアノードとタスクノードをスケールします。詳細については、Amazon EMR 管理ガイドの「Using automatic scaling in Amazon Neptune」を参照してください。	2016 年 11 月 18 日
スポットフリートのサポートを追加	Application Auto Scaling を使用して、ターゲット容量をスケールします。詳細については、Amazon EC2 – Linux インスタンス用ユーザーガイドの「スポットフリートの自動スケーリング」を参照してください。	2016 年 9 月 1 日
Amazon ECS サービスのサポートを追加	Application Auto Scaling を使用して、希望数をスケールします。詳細については、Amazon Elastic Container Service デベロッパーガイドの「サービスのオートスケーリング」を参照してください。	2016 年 8 月 9 日