aws

ユーザーガイド

AWS PCS

AWS PCS: ユーザーガイド

Table of Contents

AWS Parallel Computing Service とは

AWS Parallel Computing Service (AWS PCS) は、ハイパフォーマンスコンピューティング (HPC) ワークロードの実行とスケーリングを容易にし、Slurm AWS を使用して で科学モデルとエンジニアリングモデルを構築できるマネージドサービスです。を使用して AWS PCS、クラス最高のコンピューティング、ストレージ、ネットワーク、視覚化を統合する AWS コンピューティングクラスターを構築します。シミュレーションを実行するか、科学モデルとエンジニアリングモデルを構築します。組み込みの管理機能とオブザーバビリティ機能を使用して、クラスターオペレーションを合理化および簡素化します。使い慣れた環境でアプリケーションやジョブを実行できるようにすることで、ユーザーが研究とイノベーションに集中できるようにします。

主要なコンセプト

の AWS PCSクラスターには 1 つ以上のキューがあり、少なくとも 1 つのコンピューティングノードグループに関連付けられています。ジョブはキューに送信され、コンピューティングノードグループで定義されたEC2インスタンスで実行されます。これらの基盤を使用して、高度なHPCアーキテクチャを実装できます。

クラスター

クラスターは、リソースを管理し、ワークロードを実行するためのリソースです。クラスターは、コンピューティング、ネットワーク、ストレージ、アイデンティティ、ジョブスケジューラ設定のアセンブリを定義する AWS PCSリソースです。クラスターを作成するには、使用するジョブスケジューラ (Slurm の現在) 、使用するスケジューラ設定、クラスターを管理するサービスコントローラー、VPCクラスターリソースを起動する場所を指定します。スケジューラはジョブを受け入れてスケジュールし、それらのジョブを処理するコンピューティングノード (EC2 インスタンス) も起動します。

コンピューティングノードグループ

コンピューティングノードグループは、 が AWS PCSジョブを実行したり、クラスターへのインタラクティブなアクセスを提供したりするコンピューティングノードのコレクションです。コンピューティングノードグループを定義するときは、Amazon EC2インスタンスタイプ、最小インスタンス数と最大インスタンス数、ターゲットVPCサブネット、Amazon マシンイメージ (AMI) 、購入オプション、カスタム起動設定などの一般的な特性を指定します。 AWS PCS は、これらの設定を使用して、コンピューティングノードグループ内のコンピューティングノードを効率的に起動、管理、および終了します。

キュー

特定のクラスターでジョブを実行する場合は、特定のキュー (パーティション とも呼ばれます) に送信します。ジョブは、 がコンピューティングノードグループで実行するように AWS PCSスケジュールするまでキューに残ります。1 つ以上のコンピューティングノードグループを各キューに関連付けます。キューは、ジョブスケジューラが提供するさまざまなスケジューリングポリシーを使用して、基盤となるコンピューティングノードグループリソースでジョブをスケジュールして実行するために必要です。ユーザーは、コンピューティングノードまたはコンピューティングノードグループに直接ジョブを送信しません。

システム管理者

システム管理者は、クラスターをデプロイ、維持、運用します。、 AWS Management Console、AWS PCSAPIおよび AWS PCSを介して にアクセスできます AWS SDK。SSH または を介して特定のクラスターにアクセスできます。このクラスターでは AWS Systems Manager、管理タスクの実行、ジョブの実行、データの管理、その他のシェルベースのアクティビティを実行できます。詳細については、AWS Systems Manager ドキュメント を参照してください。

エンドユーザー

エンドユーザーには、クラスターをデプロイまたは運用する day-to-day 責任はありません。ターミナルインターフェイス (などSSH) を使用して、クラスターリソースへのアクセス、ジョブの実行、データの管理、その他のシェルベースのアクティビティを実行します。

AWS Parallel Computing Service のセットアップ

Parallel Computing Service () AWS を設定するには、次のタスクを実行しますAWS PCS。

トピック

にサインアップする AWS アカウント

がない場合は AWS アカウント、次の手順を実行して作成します。

にサインアップするには AWS アカウント

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

   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、電話キーパッドで検証コードを入力するように求められます。

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

AWS サインアッププロセスが完了すると、 から確認メールが送信されます。https://aws.amazon.com/ に移動し、マイアカウント を選択すると、いつでも現在のアカウントアクティビティを表示し、アカウントを管理できます。

管理アクセスを持つユーザーを作成する

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

のセキュリティ保護 AWS アカウントのルートユーザー

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

   ルートユーザーを使用してサインインする方法については、AWS サインイン ユーザーガイドのルートユーザーとしてサインインするを参照してください。

2. ルートユーザーの多要素認証 (MFA) を有効にします。

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

管理アクセスを持つユーザーを作成する

1. IAM Identity Center を有効にします。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「AWS IAM Identity Centerの有効化」を参照してください。

2. IAM Identity Center で、ユーザーに管理アクセス権を付与します。

   を ID ソース IAM アイデンティティセンターディレクトリ として使用する方法のチュートリアルについては、「 ユーザーガイド」の「デフォルト でユーザーアクセスを設定する IAM アイデンティティセンターディレクトリAWS IAM Identity Center 」を参照してください。

管理アクセス権を持つユーザーとしてサインインする

- IAM Identity Center ユーザーでサインインするには、IAMIdentity Center ユーザーの作成時に E メールアドレスにURL送信されたサインインを使用します。

  IAM Identity Center ユーザーを使用してサインインする方法については、「 ユーザーガイド」の AWS 「 アクセスポータルにサインインする」を参照してください。 AWS サインイン

追加のユーザーにアクセス権を割り当てる

1. IAM Identity Center で、最小特権のアクセス許可を適用するベストプラクティスに従うアクセス許可セットを作成します。

手順については、「AWS IAM Identity Center ユーザーガイド」の「権限設定を作成する」を参照してください。

2. グループにユーザーを割り当て、そのグループにシングルサインオンアクセス権を割り当てます。

   手順については、「AWS IAM Identity Center ユーザーガイド」の「グループの参加」を参照してください。

のインストール AWS CLI

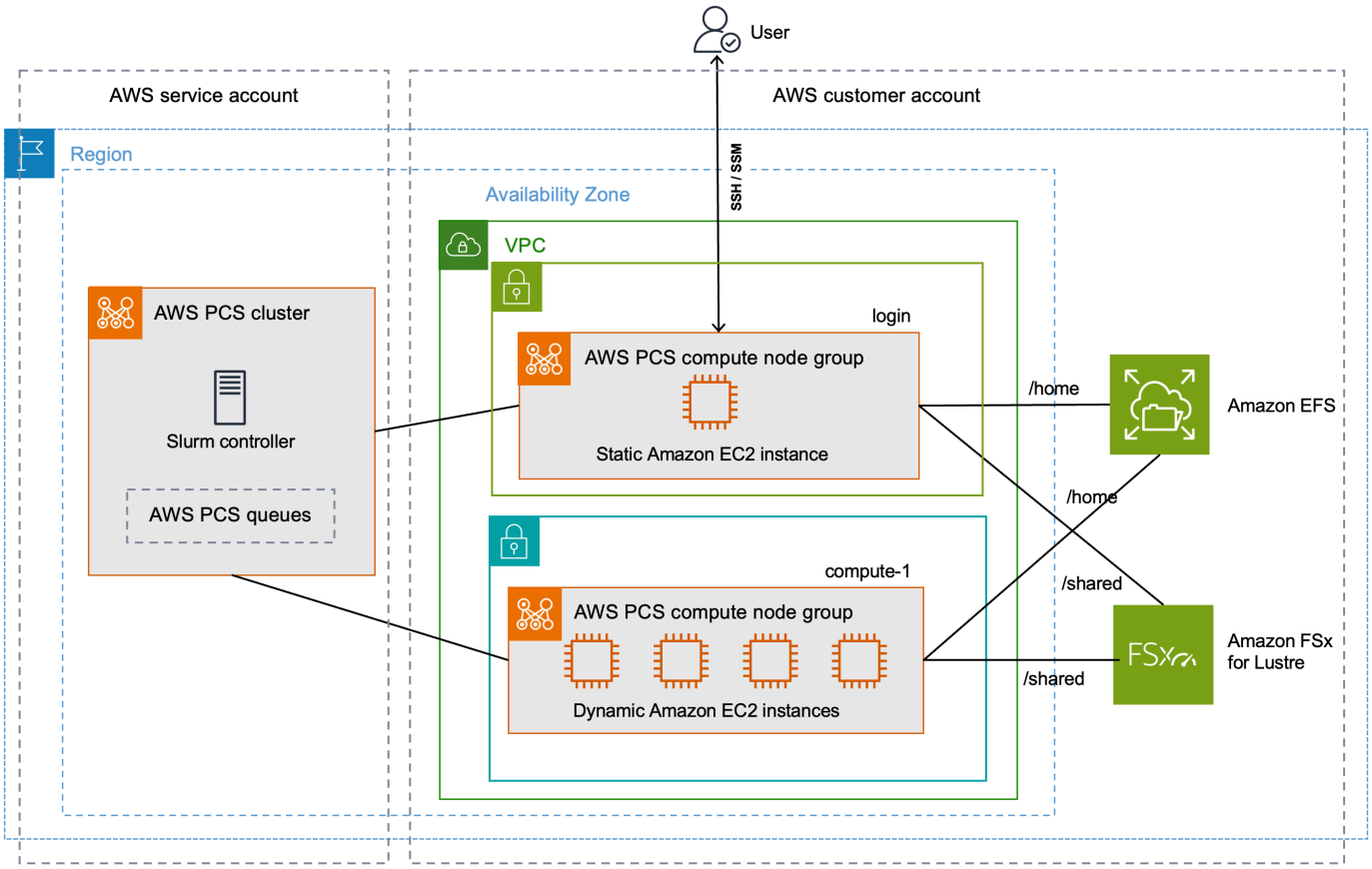
の最新バージョンを使用する必要があります AWS CLI。詳細については、「 バージョン 2 のユーザーガイド」の「 の最新バージョンのインストールまたは更新 AWS CLIAWS Command Line Interface 」を参照してください。

コマンドプロンプトで次のコマンドを入力して、 を確認します AWS CLI。ヘルプ情報が表示されます。

```
aws pcs help
```

の開始方法 AWS PCS

これは、 を試すために使用できるシンプルなクラスターを作成するためのチュートリアルです AWS PCS。次の図は、クラスターの設計を示しています。



チュートリアルクラスター設計には、次の主要コンポーネントがあります。

- AWS PCS ネットワーク要件を満たす VPCおよび サブネット。
- 共有ホームディレクトリとして使用される Amazon EFS ファイルシステム。
- 共有ハイパフォーマンスディレクトリを提供する Amazon FSx for Lustre ファイルシステム。
- Slurm コントローラーを提供する AWS PCSクラスター。
- 2 つのコンピューティングノードグループ。
  - システムへのシェルベースのインタラクティブアクセスを提供するloginノードグループ。
  - compute-1 ノードグループは、ジョブを実行するために伸縮自在にスケーリングするインスタンスを提供します。
- compute-1 ノードグループのEC2インスタンスにジョブを送信する 1 つのキュー。

クラスターには、セキュリティグループ、IAMロール、EC2起動テンプレートなどの追加の AWS リソースが必要ですが、図には示されていません。

トピック

の使用を開始するための前提条件 AWS PCS

このチュートリアルを開始する前に、 クラスターの作成と管理 AWS PCSに必要な以下のツールとリソースをインストールして設定します。

- AWS CLI – を含む のサービスを操作する AWS ためのコマンドラインツール AWS PCS。詳細については、「 バージョン 2 のユーザーガイド」の「 の最新バージョンのインストールまたは更新 AWS CLI」を参照してください。 AWS Command Line Interface をインストールしたら AWS CLI、設定することもお勧めします。詳細については、「 バージョン 2 のユーザーガイド AWS CLI」の「 の設定」を参照してください。 AWS Command Line Interface
- 必要なIAMアクセス許可 – 使用しているIAMセキュリティプリンシパルには、IAMロール、サービスにリンクされたロール、 AWS CloudFormation、VPCおよび関連リソースを操作する AWS PCSためのアクセス許可が必要です。詳細については、「 ユーザーガイド」のAWS Parallel Computing Service の Identity and Access Management「」および「サービスにリンクされたロールの作成AWS Identity and Access Management 」を参照してください。 https://

docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create-service-linked-role.htmlこのガイドのすべての手順は、1 つのユーザーとして実行する必要があります。現在のユーザーを確認するには、次のコマンドを実行します。

```
aws sts get-caller-identity
```

- このトピックのコマンドラインステップは、Bash シェルで完了することをお勧めします。Bash シェルを使用していない場合、行継続文字や、変数の設定と使用に関する方法など、一部のスクリプトコマンドのためにシェルの調整が必要となります。さらに、シェルの引用規則とエスケープ規則は異なる場合があります。詳細については、「 バージョン 2 ユーザーガイド」の「 の文字列を含む引用符とリテラル AWS CLI」を参照してください。 AWS Command Line Interface

の VPCおよび サブネットを作成する AWS PCS

CloudFormation テンプレートを使用して VPCおよび サブネットを作成できます。以下を使用して CloudFormation テンプレートURLをダウンロードし、AWS CloudFormation コンソールでテンプレートをアップロードして新しい CloudFormationスタックを作成します。詳細については、「 ユーザーガイド」の「 AWS CloudFormation コンソールの使用AWS CloudFormation 」を参照してください。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/net/hpc_large_scale/assets/
main.yaml
```

AWS CloudFormation コンソールでテンプレートを開いた状態で、次のオプションを入力します。テンプレートで指定されたデフォルト値を使用できます。

- スタック名 を指定します。
  - スタック名 で、次のように入力します。

    ```
    hpc-networking
    ```
- パラメータ の下：
  - の下VPC：
    - でCidrBlock、次のように入力します。

      ```
      10.3.0.0/16
      ```
  - サブネット A の下：

- CidrPublicSubnetで、次のように入力します。

```
10.3.0.0/20
```

- CidrPrivateSubnetで、次のように入力します。

```
10.3.128.0/20
```

- サブネット B の下：
  - CidrPublicSubnetB で、次のように入力します。

```
10.3.16.0/20
```

  - CidrPrivateSubnetB で、次のように入力します。

```
10.3.144.0/20
```

- サブネット C の下：
  - ProvisionSubnetsC の場合は、True を選択します。
  - CidrPublicSubnetC で、次のように入力します。

```
10.3.32.0/20
```

  - CidrPrivateSubnetC で、次のように入力します。

```
10.3.160.0/20
```

- 「 の機能」で：
  - がIAMリソース を作成する AWS CloudFormation 可能性があることを了承します。

CloudFormation スタックのステータスをモニタリングします。に達したらCREATE_COMPLETE、新しい でデフォルトのセキュリティグループの ID を見つけますVPC。ID はチュートリアルの後半で使用します。

クラスターのデフォルトのセキュリティグループを検索する VPC

新しい でデフォルトのセキュリティグループの ID を検索するにはVPC、次の手順に従います。

- Amazon VPCコンソール に移動します。

- VPC ダッシュボード で、 でフィルタリング VPCを選択します。
  - 名前VPCが で始まる を選択しますhpc-networking。
  - セキュリティ で、セキュリティグループ を選択します。
- という名前のグループのセキュリティグループ ID を見つけますdefault。説明 がありますdefault VPC security group。後で ID を使用してEC2起動テンプレートを設定します。

のセキュリティグループを作成する AWS PCS

AWS PCS は、セキュリティグループに依存して、クラスターとそのコンピューティングノードグループとの間で送受信されるネットワークトラフィックを管理します。このトピックの詳細については、「」を参照してくださいセキュリティグループの要件と考慮事項。

このステップでは、2 つのセキュリティグループに CloudFormation テンプレートを使用します。

- コントローラー、コンピューティングノード、ログインノード間の AWS PCS通信を可能にするクラスターセキュリティグループ。
- アクセスをサポートするためにログインノードにオプションで追加できるインバウンドSSHセキュリティグループ SSH

のセキュリティグループを作成する AWS PCS

この CloudFormation テンプレートを使用して VPCおよび サブネットを作成できます。以下を使用して CloudFormation テンプレートURLをダウンロードし、AWS CloudFormation コンソールでテンプレートをアップロードして新しい CloudFormation スタックを作成します。詳細については、「 ユーザーガイド」の AWS CloudFormation 「 コンソールの使用AWS CloudFormation 」を参照してください。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/getting_started/assets/pcs-cluster-sg.yaml
```

AWS CloudFormation コンソールでテンプレートを開いた状態で、次のオプションを入力します。一部のオプションはテンプレートに事前入力されることに注意してください。デフォルト値のままにしておくだけで済みます。

- スタック名を指定する
  - スタック名 で、次のように入力します。

```
getstarted-sg
```

- パラメータの下
  - でVpcId、名前VPCが で始まる を選択しますhpc-networking。
  - （ オプション) でClientIpCidr、インバウンドSSHセキュリティグループのより制限の厳しい IP 範囲を入力します。これを独自の IP/サブネット (独自の ip の場合は x.x.x.x/32、範囲の場合は x.x.x/24 に制限することをお勧めします。 x.x.x.x を独自の PUBLIC IP に置き換えます。 https://ifconfig.co/ などのツールを使用してパブリック IP を取得できます ）

CloudFormation スタックのステータスをモニタリングします。セキュリティグループリソースに到達するCREATE_COMPLETEと、リソースの準備が整います。

次の 2 つのセキュリティグループが作成され、名前が付けられます。

- cluster-getstarted-sg — これはクラスターセキュリティグループです
- inbound-ssh-getstarted-sg — これはインバウンドSSHアクセスを許可するセキュリティグループです。

でクラスターを作成する AWS PCS

では AWS PCS、クラスターはリソースを管理し、ワークロードを実行するための永続的なリソースです。新規または既存の のサブネットで、特定のスケジューラ (AWS PCS現在 Slurm をサポート) のクラスターを作成しますVPC。クラスターはジョブを受け入れてスケジュールし、それらのジョブを処理するコンピューティングノード (EC2 インスタンス) を起動します。

クラスターを作成するには

1. AWS PCS コンソールを開き、クラスターの作成 を選択します。
2. クラスター設定セクションで、次のフィールドに入力します。
   - クラスター名 – 入力 get-started
   - コントローラーサイズ – 小さいを選択
3. ネットワークセクションで、次のフィールドの値を選択します。
   - VPC – VPC名前付き を選択します。 hpc-networking:Large-Scale-HPC

- サブネット — 名前の先頭のサブネットを選択します。 `hpc-networking:PrivateSubnetA`
- セキュリティグループ – という名前のクラスターセキュリティグループを選択します。 `cluster-getstarted-sg`

4. [クラスターを作成] を選択します。

> **Note**
>
> ステータスフィールドには、クラスターのプロビジョニング中に を作成 と表示されます。クラスターの作成には数分かかる場合があります。

Amazon Elastic File System で の共有ストレージを作成する AWS PCS

Amazon Elastic File System (Amazon EFS) は、サーバーレスで完全に伸縮自在なファイルストレージを提供する AWS サービスです。ストレージ容量とパフォーマンスをプロビジョニングまたは管理することなく、ファイルデータを共有できます。詳細については、Amazon Elastic File System ユーザーガイドの「Amazon Elastic File System とは」を参照してください。

AWS PCS デモンストレーションクラスターは、EFSファイルシステムを使用して、クラスターノード間で共有ホームディレクトリを提供します。クラスターVPCと同じ にEFSファイルシステムを作成します。

Amazon EFS ファイルシステムを作成するには

1. Amazon EFSコンソール に移動します。
2. を試す AWS リージョン のと同じ に設定されていることを確認します AWS PCS。
3. ファイルシステムを作成する を選択します。
4. ファイルシステムの作成ページで、次のパラメータを設定します。

   - [名前] に「`getstarted-efs`」と入力します。
   - Virtual Private Cloud (VPC) で、 VPC という名前の を選択します。 `hpc-networking:Large-Scale-HPC`
   - [Create] (作成) を選択します。これにより、ファイルシステムページに戻ります。

5. `getstarted-efs` ファイルシステムのファイルシステム ID を書き留めます。この情報は後で使用します。

Amazon FSx for Lustre で の共有ストレージを作成する AWS PCS

Amazon FSx for Lustre を使用すると、人気のある高性能 Lustre ファイルシステムの起動と実行を簡単かつ費用対効果の高い方法で行うことができます。Lustre は、機械学習、ハイパフォーマンスコンピューティング (HPC) 、ビデオ処理、財務モデリングなど、スピードが重要なワークロードに使用します。詳細については、「Amazon FSx for Lustre ユーザーガイド」の「Amazon FSx for Lustre とは」を参照してください。

AWS PCS デモンストレーションクラスターはFSx、 for Lustre ファイルシステムを使用して、クラスターノード間で高性能な共有ディレクトリを提供できます。クラスターVPCと同じ に FSx for Lustre ファイルシステムを作成します。

FSx for Lustre ファイルシステムを作成するには

1. Amazon FSxコンソール に移動します。
2. コンソールがクラスター AWS リージョン と同じ を使用するように設定されていることを確認します。
3. ファイルシステムを作成する を選択します。
   - ファイルシステムタイプの選択 で、Amazon FSx for Lustre を選択し、次へ を選択します。
4. ファイルシステムの詳細の指定ページで、次のパラメータを設定します。
   - ファイルシステムの詳細の下
     - [名前] に「`getstarted-fsx`」と入力します。
     - デプロイとストレージタイプ で、永続、 SSD
     - ストレージ単位あたりのスループット で、125 MB/秒/TiB を選択します。
     - ストレージ容量 には、1.2 TiB と入力します。
     - メタデータ設定 で、自動 を選択します。
     - データ圧縮タイプ で、 を選択します。 LZ4
   - ネットワークとセキュリティの下
     - Virtual Private Cloud (VPC) では、VPC名前付き を選択します。 `hpc-networking:Large-Scale-HPC`

- VPC セキュリティグループ の場合、セキュリティグループは のままにします。 `default`
- サブネット で、名前が で始まるサブネットを選択します。 `hpc-networking:PrivateSubnetA`

- 他のオプションはデフォルト値のままにします。
- [Next (次へ)] を選択します。

5. 確認と作成 ページで、ファイルシステムの作成 を選択します。これにより、ファイルシステムページに戻ります。
6. 作成した FSx for Lustre ファイルシステムの詳細ページに移動します。
7. ファイルシステム ID とマウント名 を書き留めます。この情報は後で使用します。

> **Note**
>
> ステータスフィールドには、ファイルシステムのプロビジョニング中に を作成 と表示されます。ファイルシステムの作成には数分かかる場合があります。完了するまで待ってから、チュートリアルの残りの部分に進みます。

でコンピューティングノードグループを作成する AWS PCS

コンピューティングノードグループは、 が AWS PCS起動して管理するコンピューティングノード (EC2 インスタンス) の仮想コレクションです。コンピューティングノードグループを定義するときは、EC2インスタンスタイプ、最小インスタンス数と最大インスタンス数、ターゲットVPCサブネット、優先購入オプション、カスタム起動設定などの一般的な特性を指定します。 AWS PCS は、これらの設定に従って、コンピューティングノードグループ内のコンピューティングノードを効率的に起動、管理、および終了します。デモンストレーションクラスターは、コンピューティングノードグループを使用してユーザーアクセス用のログインノードを提供し、別のコンピューティングノードグループを使用してジョブを処理します。以下のトピックでは、クラスターでこれらのコンピューティングノードグループを設定する手順について説明します。

トピック

- のインスタンスプロファイルを作成する AWS PCS
- の起動テンプレートを作成する AWS PCS
- でログインノード用のコンピューティングノードグループを作成する AWS PCS

- でコンピューティングジョブを実行するためのコンピューティングノードグループを作成する AWS PCS

のインスタンスプロファイルを作成する AWS PCS

コンピューティングノードグループの作成時にインスタンスプロファイルが必要です。を使用して Amazon のロール AWS Management Console を作成する場合EC2、コンソールは自動的にインスタンスプロファイルを作成し、ロールと同じ名前を付けます。詳細については、「 ユーザーガイド」の「インスタンスプロファイルの使用AWS Identity and Access Management 」を参照してください。

次の手順では、 を使用して Amazon のロール AWS Management Console を作成します。これによりEC2、コンピューティングノードグループのインスタンスプロファイルも作成されます。

ロールとインスタンスプロファイルを作成するには

- IAM コンソールに移動します。
- [アクセス管理] で、[ポリシー] を選択します。
  - [Create policy] を選択します。
  - アクセス許可の指定 で、ポリシーエディタ で を選択しますJSON。
  - テキストエディタの内容を以下に置き換えます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "pcs:RegisterComputeNodeGroupInstance"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

  - [Next (次へ)] を選択します。
  - ポリシー名 の「確認して作成する」に「」と入力しますAWSPCS-getstarted-policy。
  - [Create policy] を選択します。

- [Access management] (アクセス管理) で、[Roles] (ロール) を選択します。
- [ロールの作成] を選択します。
- 「信頼されたエンティティの選択」で、次のようにします。
  - 信頼できるエンティティタイプ で、AWS サービスを選択します。
  - ユースケース で、 を選択しますEC2。
    - 次に、指定されたサービスのユースケースを選択するで、 を選択しますEC2。
  - [Next (次へ)] を選択します。
- 「アクセス許可の追加」で、次のようにします。
  - アクセス許可ポリシー で、AWSPCS-getstarted-policy を検索します。
  - ロールに追加するには、AWSPCS-getstarted-policy の横にあるチェックボックスをオンにします。
  - アクセス許可ポリシー で、 mazonSSMManagedInstanceCoreを検索します。
  - AmazonSSMManagedInstanceCore の横にあるチェックボックスをオンにして、ロールに追加します。
  - [Next (次へ)] を選択します。
- 名前、レビュー、および作成で、以下を実行します。
  - 「ロールの詳細」で：
    - [Role name] (ロール名) に`AWSPCS-getstarted-role`と入力します。
  - [ロールの作成] を選択します。

の起動テンプレートを作成する AWS PCS

コンピューティングノードグループを作成するときは、 がEC2起動する AWS PCSEC2インスタンスの設定に使用する起動テンプレートを指定します。これには、インスタンスの起動時に実行されるセキュリティグループやスクリプトなどの設定が含まれます。

このステップでは、1 つの CloudFormation テンプレートを使用して 2 つのEC2起動テンプレートを作成します。1 つのテンプレートはログインノードの作成に使用され、もう 1 つのテンプレートはコンピューティングノードの作成に使用されます。両者の主な違いは、ログインノードをインバウンドSSHアクセスを許可するように設定できることです。

CloudFormation テンプレートにアクセスする

以下を使用して CloudFormation テンプレートURLをダウンロードし、AWS CloudFormation コンソールでテンプレートをアップロードして新しい CloudFormation スタックを作成します。詳細については、「 ユーザーガイド」の「 AWS CloudFormation コンソールの使用AWS CloudFormation 」を参照してください。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/getting_started/assets/pcs-lt-efs-fsxl.yaml
```

CloudFormation テンプレートを使用してEC2起動テンプレートを作成する

AWS CloudFormation コンソールで CloudFormation テンプレートを完了するには、次の手順を使用します。

- スタック名 を指定します。
  - スタック名 に と入力しますgetstarted-lt。
- パラメータ の下：
  - セキュリティの下
    - にはVpcSecurityGroupId、クラスター default で という名前のセキュリティグループを選択しますVPC。
    - にはClusterSecurityGroupId、 という名前のグループを選択します。 cluster-getstarted-sg
    - にはSshSecurityGroupId、 という名前のグループを選択します。 inbound-ssh-getstarted-sg
    - でSshKeyName、任意のSSHキーペアを選択します。
  - ファイルシステムの下
    - にはEfsFilesystemId、チュートリアルの前半で作成したファイルシステムのEFSファイルシステム ID を入力します。
    - にはFSxLustreFilesystemId、チュートリアルの前半で作成した FSx for Lustre ファイルシステムのファイルシステム ID を入力します。
    - にはFSxLustreFilesystemMountName、Lustre ファイルシステムと同じ FSx のマウント名を入力します。
- 次へ を選択し、もう一度次へ を選択します。
- [送信] を選択します。

CloudFormation スタックのステータスをモニタリングします。起動テンプレートに達するCREATE_COMPLETEと、使用する準備が整います。

> **Note**
>
> CloudFormation テンプレートが作成したすべてのリソースを表示するには、AWS CloudFormation コンソール を開きます。getstarted-lt スタックを選択し、[リソース] タブを選択します。

でログインノード用のコンピューティングノードグループを作成する AWS PCS

コンピューティングノードグループは、 が AWS PCS起動して管理するコンピューティングノード (EC2 インスタンス) の仮想コレクションです。コンピューティングノードグループを定義するときは、EC2インスタンスタイプ、最小インスタンス数と最大インスタンス数、ターゲットVPCサブネット、優先購入オプション、カスタム起動設定などの一般的な特性を指定します。 AWS PCS は、これらの設定に従って、コンピューティングノードグループ内のコンピューティングノードを効率的に起動、管理、および終了します。

このステップでは、クラスターへのインタラクティブなアクセスを提供する静的コンピューティングノードグループを起動します。SSH または Amazon EC2 Systems Manager (SSM) を使用してログインし、シェルコマンドを実行して Slurm ジョブを管理できます。

コンピューティングノードグループを作成するには

- AWS PCS コンソールを開き、クラスター に移動します。
- という名前のクラスターを選択します。 get-started
- Compute node groups に移動し、Create を選択します。
- 「コンピューティングノードグループのセットアップ」セクションで、以下を指定します。
  - コンピューティングノードグループ名 – と入力しますlogin。
- 「コンピューティング設定」で、次の値を入力または選択します。
  - EC2 起動テンプレート — 名前が である起動テンプレートを選択します。 login-getstarted-lt
  - IAM インスタンスプロファイル — という名前のインスタンスプロファイルを選択します。 AWSPCS-getstarted-role

  - サブネット – 名前が で始まるサブネットを選択しますhpc-networking:PublicSubnetA。
  - インスタンス – を選択しますc6i.xlarge。
  - スケーリング設定 – 最小インスタンス数 には、 と入力します1。最大インスタンス数 には、 と入力します1。
- 「追加設定」で、以下を指定します。
  - AMI ID — 名前の先頭AMIが である を選択します。 aws-pcs-sample_ami-amzn2-x86_64-slurm-23.11
- コンピューティングノードグループの作成 を選択します。

ステータスフィールドには、コンピューティングノードグループのプロビジョニング中に を作成 と表示されます。チュートリアルの進行中は、次のステップに進むことができます。

でコンピューティングジョブを実行するためのコンピューティングノードグループを作成する AWS PCS

このステップでは、クラスターに送信されたジョブを実行するために伸縮自在にスケーリングするコンピューティングノードグループを起動します。

コンピューティングノードグループを作成するには

- AWS PCS コンソールを開き、クラスター に移動します。
- という名前のクラスターを選択する get-started
- Compute node groups に移動し、Create を選択します。
- 「コンピューティングノードグループのセットアップ」セクションで、以下を指定します。
  - コンピューティングノードグループ名 – と入力しますcompute-1。
- 「コンピューティング設定」で、次の値を入力または選択します。
  - EC2 起動テンプレート — 名前が である起動テンプレートを選択します。 compute-getstarted-lt
  - IAM インスタンスプロファイル — という名前のインスタンスプロファイルを選択します。 AWSPCS-getstarted-role
  - サブネット – 名前が で始まるサブネットを選択しますhpc-networking:PrivateSubnetA。
  - インスタンス – を選択しますc6i.xlarge。
  - スケーリング設定 — 最小インスタンス数 には、 と入力します0。最大インスタンス数 には、 と入力します4。

- 「追加設定」で、以下を指定します。
  - AMI ID – 名前AMIが で始まる を選択しますaws-pcs-sample_ami-amzn2-x86_64-slurm-23.11。
- コンピューティングノードグループの作成 を選択します。

ステータス フィールドには、コンピューティングノードグループのプロビジョニング中に を作成 と表示されます。

> **Important**
>
> このチュートリアルの次のステップに進む前に、ステータスフィールドがアクティブと表示されるのを待ちます。

でジョブを管理するキューを作成する AWS PCS

ジョブをキューに送信して実行します。ジョブは、 がコンピューティングノードグループで実行するようにスケジュールするまで AWS PCSキューに残ります。各キューは 1 つ以上のコンピューティングノードグループに関連付けられ、処理を実行するために必要なEC2インスタンスを提供します。

このステップでは、コンピューティングノードグループを使用してジョブを処理するキューを作成します。

キューを作成するには

- AWS PCS コンソール を開きます。
- get-started という名前のクラスターを選択します。
- コンピューティングノードグループに移動し、compute-1グループのステータスがアクティブ であることを確認します。

> **Important**
>
> 次のステップに進む前に、compute-1グループのステータスがアクティブである必要があります。

- キュー に移動し、キューの作成 を選択します。
  - 「キュー設定」セクションで、次の値を指定します。

    - キュー名 – 次のように入力します。 demo
    - コンピューティングノードグループ — という名前のコンピューティングノードグループを選択しますcompute-1。
- [キューの作成]を選択します。

ステータス フィールドには、キューの作成中に を作成 と表示されます。

> **Important**
>
> このチュートリアルの次のステップに進む前に、ステータス フィールドがアクティブと表示されるのを待ちます。

クラスターに接続する AWS PCS

login コンピューティングノードグループのステータスがアクティブ になったら、作成したEC2インスタンスに接続できます。

ログインノードに接続するには

- AWS PCS コンソールを開き、クラスター に移動します。
- get-started という名前のクラスターを選択します。
- Compute node groups を選択します。
- という名前のコンピューティングノードグループに移動しますlogin。
- コンピューティングノードグループ ID を見つけます。
- 別のブラウザウィンドウまたはタブで、Amazon EC2コンソール を開きます。
    - [Instances] を選択します。
    - 次のタグを持つEC2インスタンスを検索します。置換 *node-group-id* を前のステップのコンピューティングノードグループ ID の値で指定します。インスタンスが 1 つあるはずです。

      ```
      aws:pcs:compute-node-group-id=node-group-id
      ```

    - EC2 インスタンスに接続します。Session Manager または を使用できますSSH。

      Session Manager

        - インスタンスを選択します。

- [接続]を選択します。
- 「インスタンスに接続」で、「セッションマネージャー」を選択します。
- [接続]を選択します。
- [接続]を選択します。インタラクティブターミナルがブラウザで起動します。

SSH

- インスタンスを選択します。
- [接続]を選択します。
- 「インスタンスに接続」で、SSHクライアント を選択します。
- コンソールの指示に従ってください。

> **Note**
>
> インスタンスのユーザー名は **ec2-user**ではありませんroot。

でクラスター環境を調べる AWS PCS

クラスターにログインしたら、シェルコマンドを実行できます。例えば、ユーザーを変更したり、共有ファイルシステムでデータを操作したり、Slurm とやり取りしたりできます。

ユーザーの変更

Session Manager を使用してクラスターにログインした場合は、 として接続されている可能性がありますssm-user。これは、Session Manager 用に作成された特別なユーザーです。次のコマンドを使用して、Amazon Linux 2 のデフォルトユーザーに切り替えます。を使用して接続した場合、これを行う必要はありませんSSH。

```
sudo su - ec2-user
```

共有ファイルシステムの使用

コマンド を使用して、EFSファイルシステムと FSx for Lustre ファイルシステムが使用可能であることを確認できますdf -h。クラスターの出力は次のようになります。

```
[ec2-user@ip-10-3-6-103 ~]$ df -h
Filesystem                Size  Used Avail Use% Mounted on
```

```
devtmpfs                    3.8G     0  3.8G   0% /dev
tmpfs                       3.9G     0  3.9G   0% /dev/shm
tmpfs                       3.9G  556K  3.9G   1% /run
tmpfs                       3.9G     0  3.9G   0% /sys/fs/cgroup
/dev/nvme0n1p1               24G   18G  6.6G  73% /
127.0.0.1:/                 8.0E     0  8.0E   0% /home
10.3.132.79@tcp:/zlshxbev   1.2T  7.5M  1.2T   1% /shared
tmpfs                       780M     0  780M   0% /run/user/0
tmpfs                       780M     0  780M   0% /run/user/1000
```

/home ファイルシステムは 127.0.0.1 をマウントし、非常に大容量です。これは、チュートリアルの前半で作成したEFSファイルシステムです。ここで書き込まれたファイルは、クラスター内のすべてのノード/homeで で使用できます。

/shared ファイルシステムはプライベート IP をマウントし、容量は 1.2 TB です。これは、チュートリアルFSxの前半で作成した for Lustre ファイルシステムです。ここで書き込まれたファイルは、クラスター内のすべてのノード/sharedで で使用できます。

Slurm を操作する

トピック

- キューとノードを一覧表示する
- ジョブの表示

キューとノードを一覧表示する

を使用して、キューとそれらが関連付けられているノードを一覧表示できますsinfo。クラスターからの出力は次のようになります。

```
[ec2-user@ip-10-3-6-103 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
demo         up   infinite      4  idle~ compute-1-[1-4]
[ec2-user@ip-10-3-6-103 ~]$
```

という名前のパーティションを書き留めますdemo。ステータスは upで、最大 4 つのノードがあります。これは、ノードグループのcompute-1ノードに関連付けられます。コンピューティングノードグループを編集し、インスタンスの最大数を 8 に増やす8と、ノードの数が を読み取り、ノードリストが を読み取りますcompute-1-[1-8]。4 つのノードtestを持つ という名前の 2 番目のコン

ピューティングノードグループを作成し、demoキューに追加した場合、それらのノードもノードリストに表示されます。

ジョブの表示

を使用して、システム上のすべてのジョブを任意の状態で一覧表示できますsqueue。クラスターからの出力は次のようになります。

```
[ec2-user@ip-10-3-6-103 ~]$ squeue
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)
```

Slurm ジョブが保留中または実行中の場合は、後で実行squeueを再試行してください。

で単一ノードジョブを実行する AWS PCS

Slurm を使用してジョブを実行するには、ジョブ要件を指定する送信スクリプトを準備し、 sbatch コマンドを使用してキューに送信します。通常、これは共有ディレクトリから行われるため、ログインノードとコンピューティングノードにはファイルにアクセスするための共通のスペースがあります。

クラスターのログインノードに接続し、シェルプロンプトで次のコマンドを実行します。

- デフォルトユーザーになる。共有ディレクトリに変更します。

  ```
  sudo su - ec2-user
  cd /shared
  ```

- 次のコマンドを使用して、サンプルジョブスクリプトを作成します。

  ```
  cat << EOF > job.sh
  #!/bin/bash
  #SBATCH -J single
  #SBATCH -o single.%j.out
  #SBATCH -e single.%j.err

  echo "This is job \${SLURM_JOB_NAME} [\${SLURM_JOB_ID}] running on \
  ${SLURMD_NODENAME}, submitted from \${SLURM_SUBMIT_HOST}" && sleep 60 && echo "Job
   complete"
  EOF
  ```

- ジョブスクリプトを Slurm スケジューラに送信します。

```
sbatch -p demo job.sh
```

- ジョブが送信されると、ジョブ ID が数値として返されます。その ID を使用してジョブのステータスを確認します。置換 *job-id* 次のコマンドの を、 から返された番号で指定しますsbatch。

```
squeue --job job-id
```

Example

```
squeue --job 1
```

squeue コマンドは次のような出力を返します。

```
JOBID PARTITION NAME USER     ST TIME NODES NODELIST(REASON)
1     demo      test ec2-user CF 0:47 1     compute-1
```

- (R実行中) ステータスに達するまで、ジョブのステータスを引き続き確認します。ジョブは、squeueが何も返さないときに行われます。
- /shared ディレクトリの内容を確認します。

```
ls -alth /shared
```

コマンド出力は、次のようになります。

```
-rw-rw-r- 1 ec2-user ec2-user 107 Mar 19 18:33 single.1.out
-rw-rw-r- 1 ec2-user ec2-user 0 Mar 19 18:32 single.1.err
-rw-rw-r- 1 ec2-user ec2-user 381 Mar 19 18:29 job.sh
```

single.1.out および という名前のファイルは、クラスターのコンピューティングノードの 1 つによってsingle.1.err書き込まれました。ジョブは共有ディレクトリ (/shared) で実行されたため、ログインノードでも利用できます。このため、このクラスターFSxに for Lustre ファイルシステムを設定しました。

- single.1.out ファイルの内容を確認します。

```
cat /shared/single.1.out
```

出力は次の例のようになります。

```
This is job test [1] running on compute-1, submitted from ip-10-3-13-181
Job complete
```

で Slurm を使用してマルチノードMPIジョブを実行する AWS PCS

これらの手順は、Slurm を使用して でメッセージパスインターフェイス (MPI) ジョブを実行する方法を示しています AWS PCS。

ログインノードのシェルプロンプトで次のコマンドを実行します。

- デフォルトユーザーになる。をホームディレクトリに変更します。

```
sudo su - ec2-user
cd ~/
```

- C プログラミング言語でソースコードを作成します。

```
cat > hello.c << EOF
// * mpi-hello-world - https://www.mpitutorial.com
// Released under MIT License
//
// Copyright (c) 2014 MPI Tutorial.
//
// Permission is hereby granted, free of charge, to any person obtaining a copy
// of this software and associated documentation files (the "Software"), to
// deal in the Software without restriction, including without limitation the
// rights to use, copy, modify, merge, publish, distribute, sublicense, and/or
// sell copies of the Software, and to permit persons to whom the Software is
// furnished to do so, subject to the following conditions:
// The above copyright notice and this permission notice shall be included in
// all copies or substantial portions of the Software.
//
// THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
// IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
// FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
// AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
// LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING
// FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER
// DEALINGS IN THE SOFTWARE.
```

```
#include <mpi.h>
#include <stdio.h>
#include <stddef.h>

int main(int argc, char** argv) {
  // Initialize the MPI environment. The two arguments to MPI Init are not
  // currently used by MPI implementations, but are there in case future
  // implementations might need the arguments.
  MPI_Init(NULL, NULL);

  // Get the number of processes
  int world_size;
  MPI_Comm_size(MPI_COMM_WORLD, &world_size);

  // Get the rank of the process
  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

  // Get the name of the processor
  char processor_name[MPI_MAX_PROCESSOR_NAME];
  int name_len;
  MPI_Get_processor_name(processor_name, &name_len);

  // Print off a hello world message
  printf("Hello world from processor %s, rank %d out of %d processors\n",
         processor_name, world_rank, world_size);

  // Finalize the MPI environment. No more MPI calls can be made after this
  MPI_Finalize();
}
EOF
```

- OpenMPI モジュールをロードします。

```
module load openmpi
```

- C プログラムをコンパイルします。

```
mpicc -o hello hello.c
```

- Slurm ジョブ送信スクリプトを記述します。

```
cat > hello.sh << EOF
#!/bin/bash
#SBATCH -J multi
#SBATCH -o multi.out
#SBATCH -e multi.err
#SBATCH --exclusive
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=1

srun $HOME/hello
EOF
```

- 共有ディレクトリに変更します。

```
cd /shared
```

- ジョブスクリプトを送信します。

```
sbatch -p demo ~/hello.sh
```

- ジョブが完了するまでジョブをモニタリングsqueueするには、 を使用します。
- の内容を確認しますmulti.out。

```
cat multi.out
```

出力は以下のようになります。各ランクには、異なるノードで実行されたため、独自の IP アドレスがあることに注意してください。

```
Hello world from processor ip-10-3-133-204, rank 0 out of 4 processors
Hello world from processor ip-10-3-128-219, rank 2 out of 4 processors
Hello world from processor ip-10-3-141-26, rank 3 out of 4 processors
Hello world from processor ip-10-3-143-52, rank 1 out of 4 processor
```

の AWS リソースを削除する AWS PCS

このチュートリアル用に作成したクラスターとノードグループが完了したら、作成したリソースを削除する必要があります。

> **Important**
>
> で実行されているすべてのリソースに対して料金が発生します。 AWS アカウント

このチュートリアル用に作成したリソースを削除するには AWS PCS

- AWS PCS コンソール を開きます。
- get-started という名前のクラスターに移動します。
- キュー セクションに移動します。
- デモ という名前のキューを選択します。
- [削除] を選択します。

> **Important**
>
> キューが削除されるまで待ってから続行します。

- コンピューティングノードグループセクションに移動します。
- compute-1 という名前のコンピューティングノードグループを選択します。
- [削除] を選択します。
- ログイン という名前のコンピューティングノードグループを選択します。
- [削除] を選択します。

> **Important**
>
> 両方のコンピューティングノードグループが削除されるまで待ってから続行します。

- 開始方法 のクラスターの詳細ページで、「 の削除」を選択します。

> **Important**
>
> クラスターが削除されるまで待ってから、以降のステップに進みます。

このチュートリアル用に作成した他の AWS リソースを削除するには

- IAM コンソール を開きます。

- [ロール] を選択します。
- AWSPCS-getstarted-role という名前のロールを選択し、「削除」を選択します。
- ロールが削除されたら、ポリシー を選択します。
- AWSPCS-getstarted-policy という名前のポリシーを選択し、「削除」を選択します。

- AWS CloudFormation コンソールを開きます。
  - getstarted-lt という名前のスタックを選択します。
  - [削除] を選択します。

> **Important**
> スタックが削除されるまで待ってから次に進みます。

- Amazon EFSコンソール を開きます。
  - [File Systems (ファイルシステム)] を選択します。
  - getstarted-efs という名前のファイルシステムを選択します。
  - [削除] を選択します。

> **Important**
> ファイルシステムが削除されるのを待ってから次に進みます。

- Amazon FSxコンソール を開きます。
  - [File Systems (ファイルシステム)] を選択します。
  - getstarted-fsx という名前のファイルシステムを選択します。
  - [削除] を選択します。

> **Important**
> ファイルシステムが削除されるのを待ってから次に進みます。

- AWS CloudFormation コンソールを開きます。
  - getstarted-sg という名前のスタックを選択します。
  - [削除] を選択します。
- AWS CloudFormation コンソールを開きます。

- hpc-networking という名前のスタックを選択します。
- [削除] を選択します。

の使用 AWS PCS

この章では、 の使用に役立つ情報とガイダンスを提供します AWS PCS。

トピック

AWS PCS クラスター

AWS PCS クラスターは以下のコンポーネントで構成されます。

- Slurm コントロールデーモン () などのHPCシステムスケジューラソフトウェアのマネージドインスタンス`slurmctld`。
- Amazon EC2インスタンスをプロビジョニングおよび管理するためにHPCシステムスケジューラと統合されるコンポーネント。
- HPC ログとメトリクスを Amazon に送信するシステムスケジューラと統合するコンポーネント CloudWatch。

これらのコンポーネントは、 によって管理されるアカウントで実行されます AWS。これらは連携して、顧客アカウントの Amazon EC2インスタンスを管理します。 AWS PCS は、Amazon VPC サブネット内の Elastic Network Interface をプロビジョニングして、スケジューラソフトウェアから Amazon EC2インスタンスへの接続を提供します (たとえば、スケジューラソフトウェアでバッチジョブをスケジュールし、ユーザーがスケジューラコマンドを実行してそれらのジョブを一覧表示および管理できるようにします) 。

トピック

AWS Parallel Computing Service でのクラスターの作成

このトピックでは、使用可能なオプションの概要と、 AWS Parallel Computing Service () でクラスターを作成するときに考慮すべき事項について説明しますAWS PCS。クラスターを AWS PCS初めて作成する場合は、 に従うことをお勧めしますの開始方法 AWS PCS。このチュートリアルは、使用可能なすべてのオプションとHPCシステムアーキテクチャを拡張することなく、作業システムを作成するのに役立ちます。

前提条件

- AWS PCS ネットワーク 要件を満たす既存の VPCとサブネット。本番稼働用のクラスターをデプロイする前に、 VPCおよび サブネットの要件を十分に理解しておくことをお勧めします。VPC および サブネットを作成するには、「」を参照してくださいクラスターVPCの AWS PCSの作成。
- リソースを作成および管理 AWS PCSするためのアクセス許可を持つIAMプリンシパル。詳細については、「AWS Parallel Computing Service の Identity and Access Management」を参照してください。

クラスターを作成する AWS PCS

AWS Management Console または を使用してクラスター AWS CLI を作成できます。

AWS Management Console

クラスターを作成するには

1. home AWS PCShttps://console.aws.amazon.com/pcs/#/clusters でコンソールを開き、クラスターの作成 を選択します。
2. クラスターセットアップセクションで、次のフィールドに入力します。
   - クラスター名 – クラスターの名前。この名前には、英数字 (大文字と小文字が区別されます) とハイフンのみを使用できます。アルファベット文字で始まり、40 文字を超えること

はできません。名前は、クラスターを作成する AWS リージョン および AWS アカウント内で一意である必要があります。

- スケジューラ — スケジューラとバージョンを選択します。 AWS PCS は現在、Slurm 23.11 をサポートしています。詳細については、「の Slurm バージョン AWS PCS」を参照してください。
- コントローラーサイズ – コントローラーのサイズを選択します。これにより、 AWS PCS クラスターで管理できる同時ジョブとコンピューティングノードの数が決まります。コントローラーのサイズは、クラスターの作成時にのみ設定できます。サイジングの詳細については、「」を参照してくださいクラスターサイズの選択 AWS PCS 。

3. ネットワークセクションで、次のフィールドの値を選択します。

- VPC – VPC要件を満たす AWS PCS既存の を選択します。詳細については、「AWS PCS VPC およびサブネットの要件と考慮事項」を参照してください。クラスターを作成した後は、その を変更することはできませんVPC。リストVPCsにない場合は、最初に作成する必要があります。
- サブネット — 選択した で使用可能なすべてのサブネットVPCが一覧表示されます。異なるアベイラビリティーゾーンで 2 つを選択します。各サブネットは AWS PCS、サブネットの要件を満たしている必要があります。詳細については、「AWS PCS VPC およびサブネットの要件と考慮事項」を参照してください。スケジューラエンドポイントがパブリックインターネットに公開されないように、プライベートサブネットを選択することをお勧めします。
- セキュリティグループ – クラスター用に作成するネットワークインターフェイスに関連付けるセキュリティグループを指定します (複数可) AWS PCS。クラスターとそのコンピューティングノード間の通信を許可するセキュリティグループを少なくとも 1 つ選択する必要があります。詳細については、「セキュリティグループの要件と考慮事項」を参照してください。

4. (オプション) 暗号化 では、以下のフィールドを設定することで、コントローラーデータを暗号化するカスタムキーを定義できます。

- KMS キー ID – がPCS作成するKMSキーを使用するにはaws/pcs、 のままにします。カスタムKMSキーを使用する既存のKMSキーエイリアスを選択します。クラスターの作成に使用するアカウントには、カスタムKMSキーに対するkms:Decrypt権限が必要であることに注意してください。

5. (オプション) Slurm 設定セクションで、 によって設定されたデフォルトを上書きする Slurm 設定オプションを指定できます AWS PCS。

- スケールダウンアイドル時間 – 動的にプロビジョニングされたコンピューティングノードが、ジョブが完了または終了した後もアクティブのままになる時間を制御します。これをより長い値に設定すると、後続のジョブをノードで実行できる可能性が高くなりますが、コストが増加する可能性があります。値を短くするとコストは削減されますが、ノードでジョブを実行するのではなく、HPCシステムがノードのプロビジョニングに費やす時間の割合が増加する可能性があります。
- Prolog – これは、コンピューティングノードグループインスタンスの Prolog スクリプトディレクトリへの完全修飾パスです。これは Slurm の Prolog 設定に対応します。これは、特定の実行可能ファイルへのパスではなく、ディレクトリである必要があることに注意してください。
- Epilog – これは、コンピューティングノードグループインスタンスの epilog スクリプトディレクトリへの完全修飾パスです。これは Slurm の Epilog 設定に対応します。これは、特定の実行可能ファイルへのパスではなく、ディレクトリである必要があることに注意してください。
- Select type パラメータ — Slurm で使用されるリソース選択アルゴリズムを制御するのに役立ちます。この値を に設定するCR_CPU_Memoryとメモリ対応スケジューリングがアクティブになり、 に設定すると CPUのみのスケジューリングCR_CPUがアクティブになります。このパラメータは、 が によって に設定されている Slurm SelectType select/cons_tres SelectTypeParametersの設定に対応します AWS PCS。

6. (オプション) タグ で、クラスターに AWS PCSタグを追加します。
7. [クラスターを作成] を選択します。ステータスフィールドは、 が AWS PCSクラスターを作成するCreating間に表示されます。この処理には数分かかることもあります。

> **Important**
>
> ごとに AWS リージョン 1 つの Creating 状態になるクラスターは 1 つだけです AWS アカウント。 AWS PCS は、クラスターの作成時に Creating状態のクラスターがすでに存在する場合にエラーを返します。

AWS CLI

クラスターを作成するには

1. 下記のコマンドを使用して、クラスターを作成します。コマンドを実行する前に、次の置き換えを行います。

   - 置換 *region* など、クラスター AWS リージョン を作成する の ID を持つ us-east-1。
   - 置換 *my-cluster* クラスターの名前。この名前には、英数字 (大文字と小文字が区別されます) とハイフンのみを使用できます。アルファベット文字で始まり、40 文字を超えることはできません。名前は、クラスターを作成する AWS リージョン および AWS アカウント 内で一意である必要があります。
   - 置換 *23.11* サポートされているバージョンの Slurm を使用する。

     > **Note**
     >
     > AWS PCS は現在、Slurm 23.11 をサポートしています。

   - 置換 *SMALL* サポートされているクラスターサイズ。これにより、クラスターで管理できる同時ジョブとコンピューティングノードの数を決定します AWS PCS。クラスターが作成された場合にのみ設定できます。サイジングの詳細については、「」を参照してくださいクラスターサイズの選択 AWS PCS 。
   - の値を独自の値subnetIdsに置き換えます。スケジューラエンドポイントがパブリックインターネットに公開されないように、プライベートサブネットを選択することをお勧めします。
   - クラスター用に作成するネットワークインターフェイスsecurityGroupIdsに関連付ける を指定します AWS PCS。セキュリティグループは、クラスターVPCと同じ に存在する必要があります。クラスターとそのコンピューティングノード間の通信を許可するセキュリティグループを少なくとも 1 つ選択する必要があります。詳細については、「セキュリティグループの要件と考慮事項」を参照してください。
   - オプションで、--slurm-configrationオプションを追加して Slurm の動作を微調整できます。例えば、 を使用してスケールダウンアイドル時間を 60 分 (3600 秒) に設定できます--slurm configuration scaleDownIdeTime=3600。
   - オプションで、 を使用してコントローラーのデータを暗号化するカスタムKMSキーを指定できます--kms-key-id *kms-key*。を既存の KMS ARN、キー ID、またはエイリアス*kms-key*に置き換えます。クラスターの作成に使用するアカウントには、カスタムKMSキーに対するkms:Decrypt権限が必要であることに注意してください。

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM,version=23.11 \
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1
```

2. クラスターのプロビジョニングには数分かかる場合があります。クラスターのステータスのクエリを実行するには、次のコマンドを使用します。クラスターのステータスフィールドが になるまで、キューまたはコンピューティングノードグループの作成に進まないでくださいACTIVE。

```
aws pcs get-cluster --region region --cluster-identifier my-cluster
```

> **Important**
>
> ごとに AWS リージョン 1 つの Creating 状態になるクラスターは 1 つだけです AWS アカウント。 AWS PCS は、クラスターの作成時に Creating状態のクラスターがすでに存在する場合にエラーを返します。

クラスターに推奨される次のステップ

- コンピューティングノードグループを追加します。
- キューを追加します。
- ログ作成を有効化します。

でのクラスターの削除 AWS PCS

このトピックでは、 AWSPCSクラスターを削除する方法の概要を説明します。

AWS PCS クラスターを削除する際の考慮事項

- クラスターを削除する前に、クラスターに関連付けられているすべてのキューを削除する必要があります。詳細については、「でのキューの削除 AWS PCS」を参照してください。

- クラスターを削除する前に、クラスターに関連付けられているすべてのコンピューティングノードグループを削除する必要があります。詳細については、「でのコンピューティングノードグループの削除 AWS PCS」を参照してください。

クラスターを削除する

クラスター AWS CLI を削除するには、 AWS Management Console または を使用できます。

AWS Management Console

クラスターを削除するには

1. AWS PCS コンソール を開きます。
2. 削除するクラスターを選択します。
3. [削除] を選択します。
4. クラスターのステータスフィールドには が表示されますDeleting。完了までに数分かかることがあります。

AWS CLI

クラスターを削除するには

1. 次のコマンドを使用して、これらの置き換えでクラスターを削除します。
   - 置換 *region-code* クラスター AWS リージョン がある 。
   - 置換 *my-cluster* クラスターの名前または ID を入力します。

```
aws pcs delete-cluster --region region-code --cluster-identifier my-cluster
```

2. クラスターの削除には数分かかる場合があります。次のコマンドを使用して、クラスターのステータスを確認できます。

```
aws pcs get-cluster --region region-code --cluster-identifier my-cluster
```

クラスターサイズの選択 AWS PCS

AWS PCS は、パッチ適用、ノードプロビジョニング、更新などの主要なタスクを自動化しながら、可用性と安全性の高いクラスターを提供します。

クラスターを作成するときは、次の 2 つの要素に基づいてクラスターのサイズを選択します。

- 管理するコンピューティングノードの数
- クラスターで実行される予定のアクティブなジョブとキュージョブの数

Slurm クラスターサイズ	マネージドインスタンスの数	アクティブなジョブとキューに入っているジョブの数
小	最大 32	最大 256
中程度	最大 512	最大 8192
ラージ	最大 2048	最大 16384

例

- クラスターに最大 24 個のマネージドインスタンスがあり、最大 100 個のジョブを実行する場合は、スモール を選択します。
- クラスターに最大 24 個のマネージドインスタンスがあり、最大 1000 個のジョブを実行する場合は、中 を選択します。
- クラスターに最大 1000 個のマネージドインスタンスがあり、最大 100 個のジョブを実行する場合は、ラージ を選択します。
- クラスターに最大 1000 個のマネージドインスタンスがあり、最大 10,000 個のジョブを実行する場合は、ラージ を選択します。

でのクラスターシークレットの使用 AWS PCS

クラスターの作成の一環として、 AWS PCS は、クラスター上のジョブスケジューラに接続するために必要なクラスターシークレットを作成します。また、スケーリングイベントに応じて起動するインスタンスのセットを定義するコンピューティングノードグループを作成します AWS PCS。 AWS

PCS は、それらのコンピューティングノードグループによって起動されるインスタンスをクラスターシークレットで設定し、ジョブスケジューラに接続できるようにします。Slurm クライアントを手動で設定する場合があります。例としては、永続的なログインノードの構築や、ジョブ管理機能を備えたワークフローマネージャーの設定などがあります。

AWS PCS は、クラスターシークレットを のプレフィックスが付いたマネージドシークレットとして保存しますpcs! AWS Secrets Manager。シークレットのコストは、 の使用料金に含まれています AWS PCS。

> **Warning**
>
> クラスターシークレットを変更しないでください。 AWS PCSクラスターシークレットを変更すると、 はクラスターと通信できません。 AWS PCS はクラスターシークレットのローテーションをサポートしていません。クラスターシークレットを変更する必要がある場合は、新しいクラスターを作成する必要があります。

目次

Slurm クラスターシークレットの検索

AWS PCSマネージドシークレットは、 AWS Secrets Manager コンソールまたは を使用するか API、 から直接 AWS PCS、またはタグを使用して見つけることができます。

AWS Secrets Manager を使用してクラスターシークレットを検索する

AWS Management Console

1. [Secrets Manager console] (Secrets Managerのコンソール) に移動します。
2. シークレット を選択し、pcs!プレフィックスを検索します。

> **Note**
>
> クラスターシークレットには、 AWS PCSがクラスター ID `pcs!slurm-secret-cluster-id` *cluster-id* である形式で名前が付けられます AWS PCS。

AWS CLI

各 AWS PCSクラスターシークレットにも のタグが付けられます`aws:pcs:`*cluster-id*。クラスターのシークレット ID は、次のコマンドで取得できます。コマンドを実行する前に、次の置換を行います。

- を *region*に置き換え AWS リージョン て、 などの でクラスターを作成しますus-east-1。
- を、 AWS PCSクラスターシークレットを検索するクラスターの ID *cluster-id*に置き換えます。

```
aws secretsmanager list-secrets \
    --region region  \
    --filters Key=tag-key,Values=aws:pcs:cluster-id \
            Key=tag-value,Values=cluster-id
```

を使用して AWS PCSクラスターシークレットを検索する

を使用して AWS CLI 、 クラスターシークレットARNの AWS PCSを検索できます。次のコマンドを入力して、次の置換を行います。

- を *region*に置き換え AWS リージョン て、 などの でクラスターを作成しますus-east-1。
- をクラスターの名前または識別子*my-cluster*に置き換えます。

```
aws pcs get-cluster --region region --cluster-identifier my-cluster
```

次の出力例は、 `get-cluster` コマンドからのものです。シークレットを取得するには、`secretArn`と `secretVersion`一緒に使用できます。

```
{
    "cluster": {
        "name": "pcsdemo",
```

```
        "id": "s3431v9rx2",
        "arn": "arn:aws:pcs:us-east-1:012345678901:cluster/s3431v9rx2",
        "status": "ACTIVE",
        "createdAt": "2024-07-12T15:32:27.225136+00:00",
        "modifiedAt": "2024-07-12T15:32:27.225136+00:00",
        "scheduler": {
            "type": "SLURM",
            "version": "23.11"
        },
        "size": "SMALL",
        "networking": {
            "subnetIds": [
                "subnet-0123456789abcdef"
            ],
            "securityGroupIds": [
                "sg-0123456789abcde"
            ]
        },
        "endpoints": [
            {
                "type": "SLURMCTLD",
                "privateIpAddress": "127.0.0.1",
                "port": "6817"
            }
        ],
        "secretArn": "arn:aws:secretsmanager:us-east-1:012345678901:secret:pcs!slurm-
secret-s3431v9rx2-FN7tJF",
        "secretVersion": "ff58d1fd-070e-4bbc-98a0-64ef967cebcc"
    }
}
```

Slurm クラスターシークレットを取得する

Secrets Manager を使用して、現在の base64 でエンコードされたバージョンの Slurm クラスターシークレットを取得できます。次の例では、 を使用します AWS CLI。コマンドを実行する前に、次の置換を行います。

- を *region*に置き換え AWS リージョン て、 などの でクラスターを作成しますus-east-1。
- を クラスターsecretArnの *secret-arn* AWS PCSに置き換えます。

```
aws secretsmanager get-secret-value \
```

```
    --region region \
    --secret-id 'secret-arn' \
    --version-stage AWSCURRENT \
    --query 'SecretString' \
    --output text
```

Slurm クラスターシークレットの使用方法については、「」を参照してくださいスタンドアロンインスタンスをログインノードとして使用する AWS PCS 。

アクセス許可

プリンIAMシパルを使用して Slurm クラスターシークレットを取得します。IAM プリンシパルには、シークレットを読み取るアクセス許可が必要です。詳細については、「 AWS Identity and Access Management ユーザーガイド」の「ロールの用語と概念」を参照してください。

次のサンプルIAMポリシーでは、サンプルクラスターシークレットへのアクセスを許可します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSecretValueRetrievalAndVersionListing",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-east-1:012345678901:secret:pcs!
slurm-secret-s3431v9rx2-FN7tJF"
        }
    ]
}
```

AWS PCS コンピューティングノードグループ

AWS PCS コンピューティングノードグループは、ノード (Amazon EC2インスタンス) の論理コレクションです。これらのノードは、コンピューティングジョブの実行や、 HPCシステムへのインタラクティブなシェルベースのアクセスの提供に使用できます。コンピューティングノードグループは、使用する Amazon EC2インスタンスタイプ、実行するインスタンスの数、スポットインスタンスとオンデマンドインスタンスのどちらを使用するか、使用するサブネットとセキュリティグループ、

起動時に各インスタンスを設定する方法など、ノードを作成するためのルールで構成されます。これらのルールが更新されると、 AWS PCS はコンピューティングノードグループに関連付けられたリソースを一致させるように更新します。

トピック

- でのコンピューティングノードグループの作成 AWS PCS
- コンピューティングノードグループの更新 AWS PCS
- でのコンピューティングノードグループの削除 AWS PCS
- でのコンピューティングノードグループインスタンスの検索 AWS PCS

でのコンピューティングノードグループの作成 AWS PCS

このトピックでは、使用可能なオプションの概要と、 AWS Parallel Computing Service () でコンピューティングノードグループを作成するときに考慮すべき点について説明しますAWS PCS。でコンピューティングノードグループを初めて作成する場合は AWS PCS、「」のチュートリアルに従うことをお勧めしますの開始方法 AWS PCS。このチュートリアルは、使用可能なすべてのオプションとHPCシステムアーキテクチャを拡張することなく、作業システムを作成するのに役立ちます。

前提条件

- で必要な数のEC2インスタンスを起動するのに十分なサービスクォータ AWS リージョン。を使用してAWS Management Console、サービスクォータの引き上げを確認およびリクエストできます。
- ネットワーク要件を満たす既存の VPCとサブネット (複数可 AWS PCS) 。本番環境で使用するクラスターをデプロイする前に、これらの要件を十分に理解することをお勧めします。詳細については、「AWS PCS VPC およびサブネットの要件と考慮事項」を参照してください。テンプレートを使用して、 CloudFormation VPCおよび subnets. AWS provides テンプレートのHPCレシピを作成することもできます CloudFormation 。詳細については、「」の aws-hpc-recipes「」を参照してください GitHub。
- `RegisterComputeNodeGroupInstance` API アクションを AWS PCS呼び出すアクセス許可と、ノードグループIAMインスタンスに必要な他の AWS リソースへのアクセス許可を持つインスタンスプロファイル。詳細については、「IAM AWS Parallel Computing Service の インスタンスプロファイル」を参照してください。
- ノードグループインスタンスの起動テンプレート。詳細については、「での Amazon EC2起動テンプレートの使用 AWS PCS」を参照してください。

- Amazon Spot EC2 インスタンスを使用するコンピューティングノードグループを作成するには、にAWSServiceRoleForEC2Spotサービスにリンクされたロールが必要です AWS アカウント。詳細については、「の Amazon EC2 スポットロール AWS PCS」を参照してください。

でコンピューティングノードグループを作成する AWS PCS

コンピューティングノードグループは、 AWS Management Console または を使用して作成できます AWS CLI。

AWS Management Console

コンソールを使用してコンピューティングノードグループを作成するには

1. AWS PCS コンソール を開きます。
2. コンピューティングノードグループを作成するクラスターを選択します。Compute node groups に移動し、Create を選択します。
3. 「コンピューティングノードグループのセットアップ」セクションで、ノードグループの名前を指定します。名前には、大文字と小文字を区別する英数字とハイフンのみを使用できます。アルファベット文字で始まり、25 文字を超えることはできません。名前はクラスター内で一意である必要があります。
4. 「コンピューティング設定」で、次の値を入力または選択します。

    a. EC2 起動テンプレート — このノードグループに使用するカスタム起動テンプレートを選択します。起動テンプレートを使用して、サブネット、セキュリティグループ、モニタリング設定、インスタンスレベルのストレージなどのネットワーク設定をカスタマイズできます。起動テンプレートを準備していない場合は、での Amazon EC2起動テンプレートの使用 AWS PCS「」を参照して作成方法を確認してください。

    > **Important**
    >
    > AWS PCS は、コンピューティングノードグループごとにマネージド起動テンプレートを作成します。これらは という名前ですpcs-*identifier*-do-not-delete。コンピューティングノードグループを作成または更新するときにこれらを選択しないでください。選択しないと、ノードグループが正しく機能しません。

    b. EC2 起動テンプレートのバージョン — カスタム起動テンプレートのバージョンを選択します。特定のバージョンを選択できるため、再現性を高めることができます。後

でバージョンを変更する場合は、コンピューティングノードグループを更新して起動テンプレートの変更を検出する必要があります。詳細については、「コンピューティングノードグループの更新 AWS PCS 」を参照してください。

c. AMI ID – 起動テンプレートに AMI ID が含まれていない場合、または起動テンプレートの値を上書きする場合は、ここで AMI ID を指定します。ノードグループAMIに使用される は、 と互換性がある必要があることに注意してください AWS PCS。AMI が提供するサンプルを選択することもできます AWS。このトピックの詳細については、「」を参照してくださいの Amazon マシンイメージ (AMIs) AWS PCS。

d. IAM インスタンスプロファイル — ノードグループのインスタンスプロファイルを選択します。インスタンスプロファイルは、リソースとサービスに安全にアクセスする AWS アクセス許可をインスタンスに付与します。準備が整っていない場合は、IAM AWS Parallel Computing Service の インスタンスプロファイル「」を参照して作成方法を確認してください。

e. サブネット – クラスターVPCがデプロイされている で 1 つ以上のサブネットを選択します AWS PCS。複数のサブネットを選択すると、ノード間のEFA通信は利用できなくなり、異なるサブネットのノード間の通信のレイテンシーが増加する可能性があります。ここで指定するサブネットが、EC2起動テンプレートで定義したサブネットと一致することを確認してください。

f. インスタンス - ノードグループ内のスケーリングリクエストを満たすには、1 つ以上のインスタンスタイプを選択します。すべてのインスタンスタイプは、同じプロセッサアーキテクチャ (x864_64 または arm64) と の数を持つ必要がありますvCPUs。インスタンスに がある場合GPUs、すべてのインスタンスタイプに同じ数の が必要です GPUs。

g. スケーリング設定 — ノードグループのインスタンスの最小数と最大数を指定します。実行中のノード数が固定されている静的設定、またはノードの最大数まで実行できる動的設定のいずれかを定義できます。静的な設定の場合は、最小値と最大値をゼロより大きい同じ値に設定します。動的設定の場合、最小インスタンスを 0 に設定し、最大インスタンスを 0 より大きい数に設定します。 AWS PCS は、静的インスタンスと動的インスタンスが混在するコンピューティングノードグループをサポートしていません。

5. (オプション) 追加設定 で、以下を指定します。

a. 購入オプション – スポットインスタンスとオンデマンドインスタンスのどちらかを選択します。

b. 配分戦略 — スポット購入オプションを選択した場合、ノードグループでインスタンスを起動するときにスポットキャパシティープールを選択する方法を指定できます。詳細

については、「Amazon Elastic Compute Cloud ユーザーガイド」の「スポットインスタンスの配分戦略」を参照してください。このオプションは、オンデマンド購入オプションを選択した場合は効果がありません。

6. （オプション) Slurmカスタム設定セクションで、次の値を指定します。

   a. 重み — この値は、スケジューリングの目的でグループ内のノードの優先度を設定します。重みの低いノードは優先度が高く、単位は任意です。詳細については、 Slurmドキュメントの「重み」を参照してください。

   b. 実メモリ – この値は、ノードグループ内のノードの実メモリのサイズ (GB) を設定します。これは、 のクラスターSlurm設定の `CR_CPU_Memory`オプションと組み合わせて使用することを目的としています AWS PCS。詳細については、 SlurmドキュメントRealMemoryの「」を参照してください。

7. （オプション) タグ で、コンピューティングノードグループにタグを追加します。

8. コンピューティングノードグループの作成 を選択します。ステータスフィールドには、 がノードグループをプロビジョニング`Creating`している間 AWS PCSに表示されます。これには数分間かかる場合があります。

推奨される次のステップ

- ノードグループを の AWS PCSキューに追加して、ジョブを処理できるようにします。

AWS CLI

を使用してコンピューティングノードグループを作成するには AWS CLI

次のコマンドを使用してキューを作成します。コマンドを実行する前に、次の置き換えを行います。

1. 置換 *`region`*など、クラスター AWS リージョン を作成する の ID を持つ `us-east-1`。
2. 置換 *`my-cluster`* クラスターの名前または `clusterId`を指定します。
3. 置換 *`my-node-group`*コンピューティングノードグループの名前。この名前には、英数字 (大文字と小文字が区別されます) とハイフンのみを使用できます。アルファベット文字で始まり、25 文字を超えることはできません。名前はクラスター内で一意である必要があります。
4. 置換 *`subnet-ExampleID1`* クラスター IDsの 1 つ以上のサブネットを持つ VPC。

5. 置換 *lt-ExampleID1* カスタム起動テンプレートの ID を指定します。準備が整っていない場合は、での Amazon EC2起動テンプレートの使用 AWS PCS「」を参照して作成方法を確認してください。

> **Important**
>
> AWS PCS は、コンピューティングノードグループごとにマネージド起動テンプレートを作成します。これらは という名前ですpcs-*identifier*-do-not-delete。コンピューティングノードグループを作成または更新するときにこれらを選択しないでください。選択しないと、ノードグループが正しく機能しません。

6. 置換 *launch-template-version* ノードグループを特定のバージョンに関連付ける場合は、特定の起動テンプレートバージョンを使用します。

7. 置換 *arn:InstanceProfile*IAM インスタンスプロファイルARNの を使用します。準備が整っていない場合は、での Amazon EC2起動テンプレートの使用 AWS PCS「」でガイダンスを参照してください。

8. 置換 *min-instances* また、*max-instances* 整数値を持つ 。実行中のノード数が固定されている静的設定、またはノードの最大数まで実行できる動的設定のいずれかを定義できます。静的な設定の場合は、最小値と最大値をゼロより大きい同じ値に設定します。動的設定の場合は、最小インスタンス数を 0 に設定し、最大インスタンス数を 0 より大きい数値に設定します。 AWS PCS は、静的インスタンスと動的インスタンスが混在するコンピューティングノードグループをサポートしていません。

9. 置換 *t3.large* 別のインスタンスタイプを使用する。instanceType 設定のリストを指定することで、インスタンスタイプを追加できます。例えば、 などです *--instance-configs instanceType=c6i.16xlarge,instanceType=c6a.16xlarge*。 すべてのインスタンスタイプは、同じプロセッサアーキテクチャ (x864_64 または arm64) と の数を持つ必要がありますvCPUs。インスタンスに がある場合GPUs、すべてのインスタンスタイプに同じ数の が必要ですGPUs。

```
aws pcs create-compute-node-group --region region \
    --cluster-identifier my-cluster \
    --compute-node-group-name my-node-group \
    --subnet-ids subnet-ExampleID1 \
    --custom-launch-template id=lt-ExampleID1,version='launch-template-version' \
    --iam-instance-profile arn=arn:InstanceProfile \
    --scaling-config minInstanceCount=min-instances,maxInstanceCount=max-instance \
```

```
    --instance-configs instanceType=t3.large
```

create-compute-node-group コマンドに追加できるオプションの設定がいくつかあります。

- カスタム起動テンプレートに への参照が含まれていない--amiIdかAMI、その値を上書きするかを指定できます。ノードグループAMIに使用される は、 と互換性がある必要があることに注意してください AWS PCS。AMI が提供するサンプルを選択することもできます AWS。このトピックの詳細については、「」を参照してくださいの Amazon マシンイメージ (AMIs) AWS PCS。
- を使用して、オンデマンド (ONDEMAND) インスタンスとスポット (SPOT) インスタンスを選択できます--purchase-option。オンデマンドがデフォルトです。スポットインスタンスを選択した場合、 --allocation-strategyを使用して、ノードグループでインスタンスを起動するときに がスポットキャパシティープールを選択する方法 AWS PCSを定義することもできます。詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の「スポットインスタンスの配分戦略」を参照してください。
- を使用して、ノードグループ内のノードSlurmの設定オプションを指定できます--slurm-configuration。重み (スケジューリング優先度) と実際のメモリを設定できます。重みの低いノードは優先度が高く、単位は任意です。詳細については、 Slurmドキュメントの「重み」を参照してください。実メモリは、ノードグループ内のノードの実メモリのサイズ (GB 単位) を設定します。これは、 Slurm設定の で AWS PCSクラスターの CR_CPU_Memoryオプションと組み合わせて使用するためのものです。詳細については、 SlurmドキュメントRealMemoryの「」を参照してください。

> **Important**
>
> コンピューティングノードグループの作成には数分かかる場合があります。

次のコマンドを使用して、ノードグループのステータスをクエリできます。ステータスが になるまで、ノードグループをキューに関連付けることはできませんACTIVE。

```
aws pcs get-compute-node-group --region region \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group
```

コンピューティングノードグループの更新 AWS PCS

このトピックでは、使用可能なオプションの概要と、AWSPCSコンピューティングノードグループを更新するときに考慮すべき点について説明します。

AWS PCS コンピューティングノードグループを更新するためのオプション

AWS PCS コンピューティングノードグループを更新すると、 AWS によって起動されるインスタンスのプロパティとPCS、それらのインスタンスの起動方法のルールを変更できます。例えば、ノードグループインスタンスAMIの を、別のソフトウェアがインストールされている別のインスタンスに置き換えることができます。または、セキュリティグループを更新して、インバウンドまたはアウトバウンドのネットワーク接続を変更することもできます。スケーリング設定を変更したり、スポットインスタンスとの間で優先購入オプションを変更したりすることもできます。

次のノードグループ設定は、作成後に変更できません。

- 名前
- インスタンス

AWS PCS コンピューティングノードグループを更新する際の考慮事項

コンピューティングノードグループは、ジョブの処理、インタラクティブなシェルアクセスの提供、およびその他のタスクに使用されるEC2インスタンスを定義します。多くの場合、1 つ以上の AWS PCSキューに関連付けられます。コンピューティングノードグループを更新して動作 (またはそのノードの動作) を変更するときは、次の点を考慮してください。

- コンピューティングノードグループプロパティの変更は、コンピューティングノードグループのステータスが更新からアクティブに変わると有効になります。新しいインスタンスは、更新されたプロパティで起動します。
- 特定のノードの設定に影響を与えない更新は、実行中のノードには影響しません。例えば、サブネットの追加や配分戦略の変更などです。
- コンピューティングノードグループの起動テンプレートを更新する場合は、新しいバージョンを使用するようにコンピューティングノードグループを更新する必要があります。
- コンピューティングノードグループのノードからセキュリティグループを追加または削除するには、その起動テンプレートを編集し、コンピューティングノードグループを更新します。新しいインスタンスは、更新されたセキュリティグループのセットで起動します。

- コンピューティングノードグループが使用するセキュリティグループを直接編集すると、実行中のインスタンスと将来のインスタンスにすぐに反映されます。
- コンピューティングノードグループで使用されるIAMインスタンスプロファイルからアクセス許可を追加または削除すると、実行中のインスタンスと将来のインスタンスにすぐに反映されます。
- コンピューティングノードグループのインスタンスAMIで使用される を変更するには、コンピューティングノードグループ (またはその起動テンプレート) を更新して新しい を使用しAMI、がインスタンスを置き換えるのを待ち AWS PCSます。
- AWS PCS は、ノードグループの更新オペレーション後に、ノードグループ内の既存のインスタンスを置き換えます。ノードで実行中のジョブがある場合、それらのジョブは がノードを AWS PCS置き換える前に完了できます。インタラクティブユーザープロセス (ログインノードインスタンスなど) は終了します。ノードグループのステータスは に戻りActive、 は AWS PCSインスタンスを置換対象としてマークしますが、実際の置換はインスタンスがアイドル状態のときに発生します。
- コンピューティングノードグループで許可されるインスタンスの最大数を減らすと、 は新しい最大値を満たすために Slurm からノード AWS PCSを削除します。 AWS PCS は、削除された Slurm ノードに関連付けられた実行中のインスタンスを終了します。削除されたノードで実行中のジョブは失敗し、キューに戻ります。
- AWS PCS は、コンピューティングノードグループごとにマネージド起動テンプレートを作成します。これらは という名前でpcs-*identifier*-do-not-delete。コンピューティングノードグループを作成または更新するときに選択しないでください。選択しないと、ノードグループが正しく機能しません。
- 購入オプションに Spot を使用するようにコンピューティングノードグループを更新する場合は、アカウントにAWSServiceRoleForEC2Spotサービスにリンクされたロールが必要です。詳細については、「の Amazon EC2 スポットロール AWS PCS」を参照してください。

AWS PCS コンピューティングノードグループを更新するには

ノードグループは、 AWSマネジメントコンソールまたは AWS を使用して更新できますCLI。

AWS Management Console

コンピューティングノードグループを更新するには

1. でAWSPCSコンソールを開きます。 https://console.aws.amazon.com/pcs/home#/clusters
2. コンピューティングノードグループを更新するクラスターを選択します。

3. コンピューティングノードグループ に移動し、更新するノードグループに移動し、編集 を選択します。
4. 「コンピューティング設定」、「追加設定」、およびSlurm「カスタマイズ設定」セクションで、以下以外の値を更新します。
   - インスタンス – コンピューティングノードグループのインスタンスを変更することはできません。
5. [Update] (更新) を選択します。ステータスフィールドには、変更の適用中に更新中と表示されます。

> **Important**
> コンピューティングノードグループの更新には数分かかる場合があります。

AWS CLI

コンピューティングノードグループを更新するには

1. 次のコマンドを使用して、コンピューティングノードグループを更新します。コマンドを実行する前に、次の置き換えを行います。
   a. 置換 *region-code* クラスターを作成するAWSリージョンを指定します。
   b. 置換 *my-node-group* をコンピューティングノードグループの名前または computeNodeGroupId に置き換えます。
   c. 置換 *my-cluster* クラスターの名前または clusterId を指定します。

```
aws pcs update-compute-node-group --region region-code \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group
```

2. 以外のノードグループパラメータを更新します--instance-configs。例えば、新しい AMI ID を設定するには、 を渡--amiId my-custom-ami-idします。*my-custom-ami-id* は、選択した AMI に置き換えられます。

> **Important**
>
> コンピューティングノードグループの更新には数分かかる場合があります。

次のコマンドを使用して、ノードグループのステータスをクエリできます。

```
aws pcs get-compute-node-group --region region-code \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group
```

でのコンピューティングノードグループの削除 AWS PCS

このトピックでは、使用可能なオプションの概要と、 でコンピューティングノードグループを削除するときに考慮すべき事項について説明します AWS PCS。

コンピューティングノードグループを削除する際の考慮事項

コンピューティングノードグループは、ジョブの処理、インタラクティブなシェルアクセスの提供、およびその他のタスクに使用されるEC2インスタンスを定義します。多くの場合、1 つ以上の AWS PCSキューに関連付けられます。コンピューティングノードグループを削除する前に、次の点を考慮してください。

- コンピューティングノードグループによって起動されたEC2インスタンスはすべて終了します。これにより、これらのインスタンスで実行されているジョブがキャンセルされ、実行中のインタラクティブプロセスが終了します。
- 削除する前に、コンピューティングノードグループとすべてのキューの関連付けを解除する必要があります。詳細については、「キューの更新 AWS PCS 」を参照してください。

コンピューティングノードグループを削除する

AWS Management Console または を使用して、コンピューティングノードグループ AWS CLI を削除できます。

AWS Management Console

コンピューティングノードグループを削除するには

1. AWS PCS コンソール を開きます。
2. コンピューティングノードグループのクラスターを選択します。
3. コンピューティングノードグループに移動し、削除するコンピューティングノードグループを選択します。
4. [削除] を選択します。
5. ステータス フィールドには が表示されますDeleting。完了までに数分かかることがあります。

> **Note**
>
> スケジューラにネイティブなコマンドを使用して、コンピューティングノードグループが削除されたことを確認できます。例えば、Slurm squeueには sinfoまたは を使用します。

AWS CLI

コンピューティングノードグループを削除するには

- 次のコマンドを使用して、コンピューティングノードグループを削除し、これらの置き換えを行います。
  - 置換 *region-code* クラスター AWS リージョン がある 。
  - 置換 *my-node-group* コンピューティングノードグループの名前または ID を入力します。
  - 置換 *my-cluster* クラスターの名前または ID を入力します。

```
aws pcs delete-compute-node-group --region region-code \
       --compute-node-group-identifier my-node-group \
       --cluster-identifier my-cluster
```

コンピューティングノードグループの削除には数分かかる場合があります。

> **Note**
>
> スケジューラにネイティブなコマンドを使用して、コンピューティングノードグループが削除されたことを確認できます。例えば、Slurm squeueには sinfoまたは を使用します。

でのコンピューティングノードグループインスタンスの検索 AWS PCS

各 AWS PCSコンピューティングノードグループは、共有設定でEC2インスタンスを起動できます。EC2 タグを使用して、 AWS Management Console または でコンピューティングノードグループのインスタンスを検索できます AWS CLI。

AWS Management Console

コンピューティングノードグループインスタンスを検索するには

1. AWS PCS コンソール を開きます。
2. クラスターを選択します。
3. ノードグループの計算 を選択します。
4. 作成したログインノードグループの ID を見つけます。
5. EC2 コンソールに移動し、インスタンス を選択します。
6. 次のタグでインスタンスを検索します。置換 *node-group-id* コンピューティングノードグループの ID (名前ではない) を指定します。

```
aws:pcs:compute-node-group-id=node-group-id
```

7. (オプション) 検索フィールドでインスタンス状態の値を変更して、設定中または最近終了したインスタンスを検索できます。
8. タグ付けされたインスタンスのリストで、各インスタンスのインスタンス ID と IP アドレスを見つけます。

AWS CLI

ノードグループインスタンスを検索するには、次のコマンドを使用します。コマンドを実行する前に、次の置換を行います。

- をクラスター AWS リージョン の *region-code*に置き換えます。例: us-east-1
- をコンピューティングノードグループの ID (名前ではない) *node-group-id*に置き換えます。
- running を pendingや などの他のインスタンス状態に置き換えterminatedて、他の状態の EC2インスタンスを検索します。

```
aws ec2 describe-instances \
     --region region-code --filters \
     "Name=tag:aws:pcs:compute-node-group-id,Values=node-group-id" \
     "Name=instance-state-name,Values=running" \
     --query 'Reservations[*].Instances[*].
{InstanceID:InstanceId,State:State.Name,PublicIP:PublicIpAddress,PrivateIP:PrivateIpAddress}
```

このコマンドにより、以下のような出力が返されます。インスタンスがプライベートサブネットにあるnull場合、 の値は PublicIPです。

```
[
    [
        {
            "InstanceID": "i-0123456789abcdefa",
            "State": "running",
            "PublicIP": "18.189.32.188",
            "PrivateIP": "10.0.0.1"
        }
    ]
]
```

> **Note**
>
> 多数のインスタンスを返describe-instancesす場合は、複数のページにオプションを使用する必要があります。詳細については、DescribeInstances「Amazon Elastic Compute Cloud APIリファレンス」の「」を参照してください。

での Amazon EC2起動テンプレートの使用 AWS PCS

Amazon ではEC2、起動テンプレートに一連の設定を保存できるため、インスタンスの起動時に個別に指定する必要はありません。 AWS PCS では、コンピューティングノードグループを設定する柔軟な方法として起動テンプレートが組み込まれています。ノードグループを作成するときは、起動テ

ンプレートを指定します。 AWS PCS は、 サービスで確実に機能するように、変換を含む派生起動テンプレートを作成します。

カスタム起動テンプレートを作成する際のオプションと考慮事項を理解すると、 で使用するテンプレートを作成するのに役立ちます AWS PCS。起動テンプレートの詳細については、「Amazon EC2 ユーザーガイド」の「起動テンプレートからのインスタンスの起動」を参照してください。

トピック

概要

EC2 起動テンプレートに含めることができるパラメータは 30 個以上あり、インスタンスの設定方法の多くの側面を制御します。ほとんどの は と完全に互換性がありますが AWS PCS、いくつかの例外があります。

EC2 起動テンプレートの以下のパラメータは、 によって AWS PCS無視されます。これらのプロパティは サービスによって直接管理される必要があるためです。

- インスタンスタイプ/インスタンスタイプの属性を指定する (`InstanceRequirements`) — AWS PCS 属性ベースのインスタンス選択はサポートされていません。
- インスタンスタイプ (`InstanceType`) — ノードグループを作成するときにインスタンスタイプを指定します。
- 詳細/IAMインスタンスプロファイル (`IamInstanceProfile`) – ノードグループを作成または更新するときに指定します。
- 詳細/API終了の無効化 (`DisableApiTermination`) — AWS PCSは、起動するノードグループインスタンスのライフサイクルを制御する必要があります。
- 詳細/API停止の無効化 (`DisableApiStop`) — AWS PCSは、起動するノードグループインスタンスのライフサイクルを制御する必要があります。
- 詳細/停止 – 休止動作 (`HibernationOptions`) – AWS PCS はインスタンスの休止をサポートしていません。

- 詳細/Elastic GPU (ElasticGpuSpecifications) — Amazon Elastic Graphics は 2024 年 1 月 8 日にサポート終了しました。
- 高度な詳細/Elastic Inference (ElasticInferenceAccelerators) — Amazon Elastic Inference は、新規のお客様は利用できなくなりました。
- AAdvanced details/Specify CPU options/Threads per core (ThreadsPerCore) – AWS PCSコアあたりのスレッド数を 1 に設定します。

これらのパラメータには、 との互換性をサポートする特別な要件があります AWS PCS。

- ユーザーデータ (UserData) – これはマルチパートエンコードされている必要があります。「Amazon EC2 ユーザーデータの使用」を参照してください。
- アプリケーションおよび OS イメージ (ImageId) – これを含めることができます。ただし、ノードグループを作成または更新するときに AMI ID を指定すると、起動テンプレートの値が上書きされます。AMI 指定する は、 と互換性がある必要があります AWS PCS。詳細については、「」を参照してくださいの Amazon マシンイメージ (AMIs ） AWS PCS。
- ネットワーク設定/ファイアウォール (セキュリティグループ ）(SecurityGroups) — セキュリティグループ名のリストは AWS PCS起動テンプレートで設定できません。起動テンプレートでネットワークインターフェイスを定義しない限り、セキュリティグループ IDs (SecurityGroupIds) のリストを設定できます。次に、インターフェイスIDsごとにセキュリティグループを指定する必要があります。詳細については、「のセキュリティグループ AWS PCS」を参照してください。
- ネットワーク設定/高度なネットワーク設定 (NetworkInterfaces) – 単一のネットワークカードでEC2インスタンスを使用し、特別なネットワーク設定を必要としない場合、 AWS PCSはインスタンスネットワークを設定できます。複数のネットワークカードを設定するか、インスタンスで Elastic Fabric Adapter を有効にするには、 を使用しますNetworkInterfaces。各ネットワークインターフェイスには、 IDsのセキュリティグループのリストが必要ですGroups。詳細については、「の複数のネットワークインターフェイス AWS PCS」を参照してください。
- 詳細/キャパシティ予約 (CapacityReservationSpecification) – これは設定できますが、 の使用CapacityReservationId時に特定の を参照することはできません AWS PCS。ただし、キャパシティ予約グループを参照できます。このグループには 1 つ以上のキャパシティ予約が含まれます。詳細については、「でのキャパシティ予約 AWS PCS」を参照してください。

基本的な起動テンプレートを作成する

AWS Management Console または を使用して起動テンプレートを作成できます AWS CLI。

AWS Management Console

起動テンプレートを作成するには

1. Amazon EC2コンソールを開き、起動テンプレート を選択します。
2. [起動テンプレートの作成] を選択します。
3. 起動テンプレート名と説明に、起動テンプレート名に一意で固有の名前を入力します。
4. 「キーペア名」の「キーペア (ログイン) 」で、 によって管理されるEC2インスタンスへのログインに使用するSSHキーペアを選択します AWS PCS。これはオプションですが推奨されます。
5. ネットワーク設定 で、次にファイアウォール (セキュリティグループ) で、ネットワークインターフェイスにアタッチするセキュリティグループを選択します。起動テンプレート内のすべてのセキュリティグループは、クラスター からの AWS PCSものである必要がありますVPC。少なくとも、以下を選択します。
   - クラスターとの AWS PCS通信を許可するセキュリティグループ
   - によって起動されたEC2インスタンス間の通信を許可するセキュリティグループ AWS PCS
   - (オプション) インタラクティブインスタンスへのインバウンドSSHアクセスを許可するセキュリティグループ
   - (オプション) コンピューティングノードがインターネットへの送信接続を行うことを許可するセキュリティグループ
   - (オプション) 共有ファイルシステムやデータベースサーバーなどのネットワークリソースへのアクセスを許可するセキュリティグループ (複数可) 。
6. 新しい起動テンプレート ID は、Amazon EC2コンソールの起動テンプレート からアクセスできます。起動テンプレート ID には という形式がありますlt-0123456789abcdef01。

推奨される次のステップ

- 新しい起動テンプレートを使用して、コンピューティングノードグループを作成または更新 AWS PCSします。

AWS CLI

起動テンプレートを作成するには

次のコマンドを使用して起動テンプレートを作成します。

- コマンドを実行する前に、次の置き換えを行います。
  a. 置換 *region-code* で作業している AWS リージョン を使用する AWS PCS
  b. 置換 *my-launch-template-name* テンプレートの名前を入力します。 AWS リージョン 使用している AWS アカウント と に一意である必要があります。
  c. 置換 *my-ssh-key-name* 任意のSSHキーの名前。
  d. 置換 *sg-ExampleID1* また、*sg-ExampleID2* インスタンスとスケジューラ間の通信とEC2インスタンス間の通信IDsを許可するセキュリティグループEC2。このトラフィックをすべて有効にするセキュリティグループが 1 つしかない場合は、 sg-ExampleID2 とその前のカンマ文字を削除できます。セキュリティグループ を追加することもできますIDs。起動テンプレートに含めるすべてのセキュリティグループは、クラスター からのAWS PCSものである必要がありますVPC。

```
aws ec2 create-launch-template --region region-code \
    --launch-template-name my-template-name \
    --launch-template-data '{"KeyName":"my-ssh-key-name","SecurityGroupIds":
 ["sg-ExampleID1","sg-ExampleID2"]}'
```

AWS CLI は次のようなテキストを出力します。起動テンプレート ID は にありますLaunchTemplateId。

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0123456789abcdef01",
        "LaunchTemplateName": "my-launch-template-name",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-04-30T18:16:06.000Z"
    }
}
```

推奨される次のステップ

- 新しい起動テンプレートを使用して、コンピューティングノードグループを作成または更新 AWS PCSします。

Amazon EC2 ユーザーデータの使用

インスタンスの起動時にcloud-init実行される起動テンプレートにEC2ユーザーデータを指定できます。コンテンツタイプのユーザーデータブロックは、インスタンスが に登録 AWS PCSされる前にcloud-config実行されますがAPI、コンテンツタイプのユーザーデータブロックは、登録完了後に実行されますが、Slurm デーモンが起動する前にtext/x-shellscript実行されます。コンテンツタイプの詳細については、cloud-init のドキュメントを参照してください。

ユーザーデータは、次のような一般的な設定シナリオを実行できます。

- ユーザーまたはグループを含める
- パッケージのインストール
- パーティションとファイルシステムの作成
- ネットワークファイルシステムのマウント

起動テンプレートのユーザーデータはMIME、マルチパートアーカイブ形式である必要があります。これは、ユーザーデータが、ノードグループ内のノードの設定に必要な他の AWS PCSユーザーデータとマージされるためです。複数のユーザーデータブロックを 1 つのMIMEマルチパートファイルにまとめることができます。

MIME マルチパートファイルは、次のコンポーネントで構成されます。

- コンテンツの種類およびパート境界の宣言: `Content-Type: multipart/mixed; boundary="==BOUNDARY=="`
- MIME バージョン宣言: `MIME-Version: 1.0`
- 次のコンポーネントを含む 1 つ以上のユーザーデータブロック:
  - ユーザーデータブロックの始まりを示す開始境界: `--==BOUNDARY==`。この境界の前の行は空白にしておく必要があります。
  - ブロックのコンテンツタイプ宣言: `Content-Type: text/cloud-config; charset="us-ascii"`または `Content-Type: text/x-shellscript; charset="us-ascii"`。コンテンツタイプ宣言の後の行は空白にしておく必要があります。

- ユーザーデータのコンテンツ (例えば、シェルコマンドや cloud-config ディレクティブのリスト)。
- MIME マルチパートファイルの終了を通知する終了境界: --==BOUNDARY==--。この境界の前の行は空白にしておく必要があります。

> **Note**
>
> Amazon EC2コンソールで起動テンプレートにユーザーデータを追加する場合は、プレーンテキストとして貼り付けることができます。または、ファイルからアップロードすることもできます。 AWS CLI または を使用する場合 AWS SDK、このJSONファイルに示すように、を呼び出すときに、まず base64 でユーザーデータをエンコードしCreateLaunchTemplate、その文字列を UserDataパラメータの値として送信する必要があります。

```
{
    "LaunchTemplateName": "base64-user-data",
    "LaunchTemplateData": {
        "UserData":
 "ewogICAgIkxhdW5jaFRlbXBsYXRlTmFtZSI6ICJpbmNyZWFzZS1jb250YWluZXItdm9sdW..."
    }
}
```

例

- 例: パッケージリポジトリからソフトウェアをインストールする
- 例: S3 バケットからスクリプトを実行する
- 例: グローバル環境変数の設定
- でのネットワークファイルシステムの使用 AWS PCS
- 例: EFS ファイルシステムを共有ホームディレクトリとして使用する

例: パッケージリポジトリから 用の AWS PCSソフトウェアをインストールする

このスクリプトを起動テンプレート"userData"の の値として指定します。詳細については、「Amazon EC2 ユーザーデータの使用」を参照してください。

このスクリプトは、クラウド設定を使用して、起動時にノードグループインスタンスにソフトウェアパッケージをインストールします。詳細については、cloud-init ドキュメントの「ユーザーデータ形式」を参照してください。この例では、 curlと をインストールしますllvm。

> **Note**
>
> インスタンスは、設定されたパッケージリポジトリに接続できる必要があります。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- python3-devel
- rust
- golang

--==MYBOUNDARY==--
```

例: S3 バケットから の追加スクリプト AWS PCSを実行する

このスクリプトを起動テンプレート"userData"の の値として指定します。詳細については、「Amazon EC2 ユーザーデータの使用」を参照してください。

このスクリプトは、cloud-config を使用して S3 バケットからスクリプトをインポートし、起動時にノードグループインスタンスで実行します。詳細については、cloud-init ドキュメントの「ユーザーデータ形式」を参照してください。

このスクリプトの次の値を独自の詳細に置き換えます。

- *my-bucket-name* – アカウントが読み取ることができる S3 バケットの名前。
- *path* – S3 バケットルートへの相対パス。
- *shell* – スクリプトの実行に使用する Linux シェル。 などbash。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="
```

```
--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- aws s3 cp s3://my-bucket-name/path /tmp/script.sh
- /usr/bin/shell /tmp/script.sh

--==MYBOUNDARY==--
```

ノードグループのIAMインスタンスプロファイルには、バケットへのアクセス権が必要です。次のIAMポリシーは、上記のユーザーデータスクリプトのバケットの例です。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::my-bucket-name",
                "arn:aws:s3:::my-bucket-name/path/*"
            ]
        }
    ]
}
```

例: のグローバル環境変数の設定 AWS PCS

このスクリプトを起動テンプレート"userData"の の値として指定します。詳細については、「Amazon EC2 ユーザーデータの使用」を参照してください。

次の例では、 を使用して/etc/profile.dノードグループインスタンスにグローバル変数を設定します。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
```

```
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
touch /etc/profile.d/awspcs-userdata-vars.sh
echo MY_GLOBAL_VAR1=100 >> /etc/profile.d/awspcs-userdata-vars.sh
echo MY_GLOBAL_VAR2=abc >> /etc/profile.d/awspcs-userdata-vars.sh

--==MYBOUNDARY==--
```

例: EFS ファイルシステムを の共有ホームディレクトリとして使用する AWS PCS

このスクリプトを起動テンプレート"userData"の の値として指定します。詳細については、「Amazon EC2 ユーザーデータの使用」を参照してください。

この例では、 でEFSマウント例を拡張でのネットワークファイルシステムの使用 AWS PCSして、共有ホームディレクトリを実装します。/home の内容は、EFSファイルシステムがマウントされる前にバックアップされます。その後、マウントが完了すると、コンテンツは共有ストレージ上の所定の場所にすばやくコピーされます。

このスクリプトの次の値を独自の詳細に置き換えます。

- *`/mount-point-directory`* - EFS ファイルシステムをマウントするインスタンスのパス。
- *`filesystem-id`* – ファイルシステムのEFSファイルシステム ID。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
  - amazon-efs-utils

runcmd:
  - mkdir -p /tmp/home
  - rsync -a /home/ /tmp/home
  - echo "filesystem-id:/ /mount-point-directory efs tls,_netdev" >> /etc/fstab
  - mount -a -t efs defaults
  - rsync -a --ignore-existing /tmp/home/ /home
  - rm -rf /tmp/home/
```

```
--==MYBOUNDARY==--
```

パスワードレスの有効化 SSH

共有ホームディレクトリの例に基づいて構築し、 SSHキーを使用してクラスターインスタンス間のSSH接続を実装できます。共有ホームファイルシステムを使用するユーザーごとに、次のようなスクリプトを実行します。

```
#!/bin/bash

mkdir -p $HOME/.ssh && chmod 700 $HOME/.ssh
touch $HOME/.ssh/authorized_keys
chmod 600 $HOME/.ssh/authorized_keys

if [ ! -f "$HOME/.ssh/id_rsa" ]; then
    ssh-keygen -t rsa -b 4096 -f $HOME/.ssh/id_rsa -N ""
    cat ~/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
fi
```

> **Note**
>
> インスタンスは、クラスターノード間のSSH接続を許可するセキュリティグループを使用する必要があります。

でのキャパシティ予約 AWS PCS

オンデマンドEC2キャパシティ予約またはキャパシティブロックを使用して、特定のアベイラビリティーゾーンおよび特定の期間に Amazon EC2キャパシティを予約し、必要なコンピューティングキャパシティーを必要なときに利用できるようにすることができます。

> **Note**
>
> AWS PCS はオンデマンドキャパシティ予約 (ODCR) をサポートしていますが、現在 ML のキャパシティブロックはサポートされていません。

ODCRs で を使用する AWS PCS

がリザーブドインスタンスをどのように消費するか AWS PCSを選択できます。オープン を作成するとODCR、 アカウントで AWS PCSまたは他のプロセスによって起動された一致するインスタンスは、予約に対してカウントされます。ターゲット ではODCR、特定の予約 ID で起動されたインスタンスのみが予約に対してカウントされます。時間的制約のあるワークロードでは、ターゲットが一般的ODCRsです。

ターゲット AWS PCSを使用するようにコンピューティングノードグループを設定するには、起動テンプレートに追加ODCRします。これを行う手順は次のとおりです。

1. ターゲットのオンデマンドキャパシティ予約 () を作成しますODCR。
2. ODCR をキャパシティーの予約グループに追加します。
3. キャパシティ予約グループを起動テンプレートに関連付けます。
4. 起動テンプレートを使用するようにコンピューティングノードグループを作成 AWS PCSまたは更新します。

例: ターゲットの で hpc6a.48xlarge インスタンスを予約して使用する ODCR

このコマンド例では、32 個の hpc6a.48xlarge インスタンスODCRのターゲット を作成します。プレイスメントグループでリザーブドインスタンスを起動するには、 コマンド`--placement-group-arn`に を追加します。`--end-date` および を使用して終了日を定義できます。そうしないと`--end-date-type`、予約は手動で終了するまで続行されます。

```
aws ec2 create-capacity-reservation \
    --instance-type hpc6a.48xlarge \
    --instance-platform Linux/UNIX \
    --availability-zone us-east-2a \
    --instance-count 32 \
    --instance-match-criteria targeted
```

このコマンドの結果は、新しい ARNの になりますODCR。ODCR で を使用するには AWS PCS、キャパシティ予約グループに追加する必要があります。これは、 が個々の をサポートしていないためです AWS PCSODCRs。詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の「キャパシティ予約グループ」を参照してください。

という名前のODCRキャパシティ予約グループに を追加する方法は次のとおりです`EXAMPLE-CR-GROUP`。

```
aws resource-groups group-resources --group EXAMPLE-CR-GROUP \
    --resource-arns arn:aws:ec2:sa-east-1:123456789012:capacity-reservation/
cr-1234567890abcdef1
```

ODCR を作成してキャパシティ予約グループに追加すると、起動テンプレートに追加 AWS PCSしてコンピューティングノードグループに接続できるようになりました。キャパシティ予約グループを参照する起動テンプレートの例を次に示します。

```
{
  "CapacityReservationSpecification": {
    "CapacityReservationResourceGroupArn": "arn:aws:resource-groups:us-
east-2:123456789012:group/EXAMPLE-CR-GROUP"
  }
}
```

最後に、コンピューティングノードグループを作成または更新 AWS PCSして hpc6a.48xlarge インスタンスを使用し、ODCRキャパシティー予約グループの を参照する起動テンプレートを使用します。静的ノードグループの場合、最小インスタンス数と最大インスタンス数を予約のサイズ (32) に設定します。動的ノードグループの場合、最小インスタンス数を 0 に設定し、最大インスタンス数を予約サイズに設定します。

この例では、1 つのコンピューティングノードグループにODCRプロビジョニングされた単一の の簡単な実装を示します。ただし、 AWS PCS は他の多くの設計をサポートしています。例えば、複数のコンピューティングノードグループ間で大きな ODCRまたは キャパシティーの予約 グループを分割できます。または、別のAWSアカウントが作成して共有ODCRsした を使用することもできます。重要な制約は、 ODCRsが常にキャパシティーの予約グループに含まれている必要があることです。

詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の「ML のオンデマンドキャパシティ予約とキャパシティブロック」を参照してください。

便利な起動テンプレートパラメータ

このセクションでは、 で広く役立つ起動テンプレートパラメータについて説明します AWS PCS。

詳細 CloudWatchモニタリングを有効にする

起動テンプレートパラメータを使用して、短い間隔で CloudWatch メトリクスの収集を有効にできます。

AWS Management Console

起動テンプレートを作成または編集するためのコンソールページで、このオプションは詳細セクションにあります。詳細 CloudWatch モニタリングを有効にに設定します。

YAML

```
Monitoring:
    Enabled: True
```

JSON

```
{"Monitoring": {"Enabled": "True"}}
```

詳細については、「Linux インスタンス用 Amazon Elastic Compute Cloud ユーザーガイド」の「インスタンスの詳細な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。

インスタンスメタデータサービスバージョン 2 (IMDS v2)

EC2 インスタンスで IMDS v2 を使用すると、セキュリティが大幅に強化され、 AWS 環境内のインスタンスメタデータへのアクセスに関連する潜在的なリスクを軽減できます。

AWS Management Console

起動テンプレートを作成または編集するためのコンソールページで、このオプションは詳細セクションにあります。メタデータへのアクセス可能を有効、メタデータバージョンを V2 のみ (トークンが必要) 、メタデータレスポンスホップ制限を 4 に設定します。

YAML

```
MetadataOptions:
  HttpEndpoint: enabled
  HttpTokens: required
  HttpPutResponseHopLimit: 4
```

JSON

```
{
    "MetadataOptions": {
        "HttpEndpoint": "enabled",
        "HttpPutResponseHopLimit": 4,
```

```
        "HttpTokens": "required"
    }
}
```

.

AWS PCS キュー

AWS PCS キューは、スケジューラのワークキューのネイティブ実装に対する軽量な抽象化です。Slurm の場合、 AWS PCSキューは Slurm パーティションと同等です。

ユーザーは、1 つ以上のコンピューティングノードグループによって提供されるノードで実行するようにスケジュールされるまで、ジョブが存在するキューに送信します。 AWS PCS クラスターには複数のジョブキューを含めることができます。例えば、優先度の高いジョブに Amazon EC2 オンデマンドインスタンスを使用するキューと、優先度の低いジョブに Amazon EC2スポットインスタンスを使用する別のキューを作成できます。

トピック

- でのキューの作成 AWS PCS
- キューの更新 AWS PCS
- でのキューの削除 AWS PCS

でのキューの作成 AWS PCS

このトピックでは、使用可能なオプションの概要と、 でキューを作成するときに考慮すべき事項について説明します AWS PCS。

前提条件

- AWS PCS クラスター - キューは、特定のPCSクラスターと関連付けてのみ作成できます。
- 1 つ以上のAWSPCSコンピューティングノードグループ - キューは少なくとも 1 つのPCSコンピューティングノードグループに関連付けられている必要があります。

で キューを作成するには AWS PCS

キューは、 AWS Management Console または を使用して作成できます AWS CLI。

AWS Management Console

コンソールを使用してキューを作成するには

1. でAWSPCSコンソールを開きます。 `https://console.aws.amazon.com/pcs/home#/clusters`
2. キューを作成するクラスターを選択します。キュー に移動し、キューの作成 を選択します。
3. 「キュー設定」セクションで、次の値を指定します。
   a. キュー名 – キューの名前。この名前には、英数字 (大文字と小文字が区別されます) とハイフンのみを使用できます。アルファベット文字で始まり、25 文字を超えることはできません。名前はクラスター内で一意である必要があります。
   b. コンピューティングノードグループ — このキューを処理するコンピューティングノードグループを 1 つ以上選択します。コンピューティングノードグループは、複数のキューに関連付けることができます。
4. (オプション) タグ で、AWSPCSキューにタグを追加します。
5. [キューの作成]を選択します。ステータスフィールドには、キューの設定中に の作成 と表示されます。キューの作成には数分かかる場合があります。

推奨される次のステップ

- 新しいキューにジョブを送信する

AWS CLI

を使用してキューを作成するには AWS CLI

次のコマンドを使用してキューを作成します。コマンドを実行する前に、次の置き換えを行います。

1. 置換 *region-code* クラスターを作成するAWSリージョンを指定します。
2. 置換 *my-queue* キューの名前を入力します。この名前には、英数字 (大文字と小文字が区別されます) とハイフンのみを使用できます。アルファベット文字で始まり、25 文字を超えることはできません。名前はクラスター内で一意である必要があります。
3. 置換 *my-cluster* クラスターの名前または clusterId を指定します。

4. の値を独自のコンピューティングノードグループ識別子computeNodeGroupIdに置き換えます。キューの作成時にコンピューティングノードグループ名を指定することはできません。

```
aws pcs create-queue --region region-code \
    --queue-name my-queue \
    --cluster-identifier my-cluster \
    --compute-node-group-configurations \
    computeNodeGroupId=computeNodeGroupExampleID1
```

キューの作成には数分かかる場合があります。次のコマンドを使用して、キューのステータスをクエリできます。ステータスが に達するまで、キューにジョブを送信することはできませんACTIVE。

```
aws pcs get-queue --region region-code \
    --cluster-identifier my-cluster \
    --queue-identifier my-queue
```

推奨される次のステップ

- 新しいキューにジョブを送信する

キューの更新 AWS PCS

このトピックでは、使用可能なオプションの概要と、AWSPCSキューを更新するときに考慮すべき事項について説明します。

AWS PCS キューを更新する際の考慮事項

キューの更新は実行中のジョブには影響しませんが、キューの更新中にクラスターが新しいジョブを受け入れることができない場合があります。

AWS PCS コンピューティングノードグループを更新するには

ノードグループは、 AWSマネジメントコンソールまたは AWS を使用して更新できますCLI。

AWS Management Console

キューを更新するには

1. でAWSPCSコンソールを開きます。 `https://console.aws.amazon.com/pcs/home#/clusters`
2. キューを更新するクラスターを選択します。
3. キュー に移動し、更新するキューに移動し、編集 を選択します。
4. キュー設定セクションで、次のいずれかの値を更新します。
   - ノードグループ – キューとの関連付けからコンピューティングノードグループを追加または削除します。
   - タグ — キューのタグを追加または削除します。
5. [Update] (更新) を選択します。ステータスフィールドには、変更の適用中に更新中と表示されます。

> **Important**
> キューの更新には数分かかる場合があります。

AWS CLI

キューを更新するには

1. 次のコマンドを使用してキューを更新します。コマンドを実行する前に、次の置き換えを行います。
   a. 置換 *region-code* クラスター AWS リージョン を作成する を使用します。
   b. 置換 *my-queue* をキューの名前または `computeNodeGroupId` に指定します。
   c. 置換 *my-cluster* クラスターの名前または `clusterId` を指定します。
   d. コンピューティングノードグループの関連付けを変更するには、 の更新リストを指定します`--compute-node-group-configurations`。
      - 例えば、2 番目のコンピューティングノードグループ を追加するには、次のようにします`computeNodeGroupExampleID2`。

```
--compute-node-group-configurations
 computeNodeGroupId=computeNodeGroupExampleID1,computeNodeGroupId=computeNodeGro
```

```
aws pcs update-queue --region region-code \
    --queue-identifier my-queue \
    --cluster-identifier my-cluster \
    --compute-node-group-configurations \
    computeNodeGroupId=computeNodeGroupExampleID1
```

2. キューの更新には数分かかる場合があります。次のコマンドを使用して、キューのステータスをクエリできます。ステータスが に達するまで、ジョブをキューに送信することはできませんACTIVE。

```
aws pcs get-queue --region region-code \
    --cluster-identifier my-cluster \
    --queue-identifier my-queue
```

推奨される次のステップ

- 更新されたキューにジョブを送信します。

でのキューの削除 AWS PCS

このトピックでは、 で キューを削除する方法の概要を説明します AWS PCS。

キューを削除する際の考慮事項

- キューで実行中のジョブがある場合、キューが削除されるとスケジューラによってジョブが終了します。キュー内の保留中のジョブはキャンセルされます。キュー内のジョブが完了するのを待つか、スケジューラのネイティブコマンド (Slurm scancelの など) を使用して手動でジョブを停止/キャンセルすることを検討してください。

キューの削除

AWS Management Console または を使用してキュー AWS CLI を削除できます。

AWS Management Console

キューを削除するには

1. AWS PCS コンソール を開きます。
2. キューのクラスターを選択します。
3. キューに移動し、削除するキューを選択します。
4. [削除] を選択します。
5. ステータス フィールドには が表示されますDeleting。完了までに数分かかることがあります。

> **Note**
>
> スケジューラにネイティブなコマンドを使用して、キューが削除されたことを確認できます。例えば、Slurm squeueには sinfoまたは を使用します。

AWS CLI

キューを削除するには

- 次のコマンドを使用して、これらの置換でキューを削除します。
  - 置換 *region-code* クラスター AWS リージョン がある 。
  - 置換 *my-queue* キューの名前または ID を入力します。
  - 置換 *my-cluster* クラスターの名前または ID を入力します。

```
aws pcs delete-queue --region region-code \
       --queue-identifier my-queue \
       --cluster-identifier my-cluster
```

キューの削除には数分かかる場合があります。

> **Note**
>
> スケジューラにネイティブなコマンドを使用して、キューが削除されたことを確認できます。例えば、Slurm squeueには sinfoまたは を使用します。

AWS PCS ログインノード

クラスターでは通常、 AWS PCSインタラクティブなアクセスとジョブ管理をサポートするために、少なくとも 1 つのログインノードが必要です。これを実現する方法は、ログインノード機能用に設定された静的 AWS PCSコンピューティングノードグループを使用することです。ログインノードとして機能するスタンドアロンEC2インスタンスを設定することもできます。

トピック

- AWS PCS コンピューティングノードグループを使用してログインノードを提供する
- スタンドアロンインスタンスをログインノードとして使用する AWS PCS

AWS PCS コンピューティングノードグループを使用してログインノードを提供する

このトピックでは、推奨される設定オプションの概要と、AWSPCSコンピューティングノードグループを使用してクラスターへの永続的でインタラクティブなアクセスを提供するときに考慮すべき点について説明します。

ログインノード用の AWS PCSコンピューティングノードグループの作成

運用上、これは通常のコンピューティングノードグループの作成と大きく変わりません。ただし、いくつかのキー設定の選択肢があります。

- コンピューティングノードグループ内の少なくとも 1 つのEC2インスタンスの静的スケーリング設定を設定します。
- インスタンスが再利用されないように、オンデマンド購入オプションを選択します。
- ログインなど、コンピューティングノードグループの情報名を選択します。
- ログインノードインスタンスに の外部からアクセスできるようにする場合はVPC、パブリックサブネットの使用を検討してください。

- SSH アクセスを許可する場合、起動テンプレートには、選択した IP アドレスにSSHポートを公開するセキュリティグループが必要です。
- IAM インスタンスプロファイルには、エンドユーザーに付与するAWSアクセス許可のみが必要です。詳細については、「IAM AWS Parallel Computing Service の インスタンスプロファイル」を参照してください。
- AWS Systems Manager Session Manager がログインインスタンスを管理できるようにすることを検討してください。
- インスタンスAWS認証情報へのアクセスを管理者ユーザーのみに制限することを検討してください。
- ログインノードは継続的に実行されるため、通常のコンピューティングノードグループよりも安価なインスタンスタイプを選択します。
- すべてのインスタンスに同じソフトウェアがインストールされていることを確認するには、他のコンピューティングノードグループAMIと同じ (または派生) を使用します。のカスタマイズの詳細についてはAMIs、「」を参照してください。 の Amazon マシンイメージ (AMIs ） AWS PCS
- ログインノードにはEFS、コンピューティングインスタンスと同じネットワークファイルシステム (Amazon 、Amazon FSx for Lustre など) マウントを設定します。詳細については、「でのネットワークファイルシステムの使用 AWS PCS」を参照してください。

ログインノードにアクセスする

新しいコンピューティングノードグループのステータスに達するとACTIVE、作成したEC2インスタンスを見つけてログインできます (複数可 ）。詳細については、「でのコンピューティングノードグループインスタンスの検索 AWS PCS」を参照してください。

ログインノードの AWS PCSコンピューティングノードグループの更新

を使用してログインノードグループを更新できます UpdateComputeNodeGroup。ノードグループの更新プロセスの一環として、実行中のインスタンスが置き換えられます。これにより、インスタンス上のアクティブなユーザーセッションまたはプロセスが中断されることに注意してください。実行中またはキューに入れられた Slurm ジョブは影響を受けません。詳細については、「コンピューティングノードグループの更新 AWS PCS 」を参照してください。

コンピューティングノードグループで使用される起動テンプレートを編集することもできます。を使用して UpdateComputeNodeGroup 、更新された起動テンプレートをコンピューティングノードグループに適用する必要があります。コンピューティングノードグループで起動された新しいEC2インスタンスは、更新された起動テンプレートを使用します。詳細については、「での Amazon EC2起動テンプレートの使用 AWS PCS」を参照してください。

ログインノードの AWS PCSコンピューティングノードグループの削除

のコンピューティングノードグループの削除メカニズムを使用して、ログインノードグループを更新できます AWS PCS。実行中のインスタンスは、ノードグループの削除の一環として終了します。これにより、インスタンス上のアクティブなユーザーセッションまたはプロセスが中断されることに注意してください。実行中またはキューに入れられた Slurm ジョブは影響を受けません。詳細については、「でのコンピューティングノードグループの削除 AWS PCS」を参照してください。

スタンドアロンインスタンスをログインノードとして使用する AWS PCS

クラスターの Slurm スケジューラと AWS PCSやり取りするように独立したEC2インスタンスを設定できます。これは、ログインノード、ワークステーション、またはクラスターと AWS PCS連携するが、管理外 AWS PCSで動作する専用のワークフロー管理ホストを作成するのに役立ちます。これを行うには、各スタンドアロンインスタンスが以下を実行する必要があります。

1. 互換性のある Slurm ソフトウェアバージョンがインストールされていること。
2. クラスターの AWS PCS Slurmctld エンドポイントに接続できる。
3. AWS PCS クラスターのエンドポイントとシークレットを使用して、Slurm Auth デーモンと Cred Kiosk デーモン (sackd) を適切に設定します。詳細については、Slurm ドキュメントの「sackd」を参照してください。

このチュートリアルは、 クラスターに接続する AWS PCS独立したインスタンスを設定するのに役立ちます。

目次

ステップ 1 – ターゲット AWS PCSクラスターのアドレスとシークレットを取得する

次のコマンド AWS CLI で を使用して、ターゲット AWS PCSクラスターの詳細を取得します。コマンドを実行する前に、次の置き換えを行います。

- 置換 *region-code* ターゲットクラスター AWS リージョン が実行中の 。
- 置換 *cluster-ident* ターゲットクラスターの名前または識別子を含む

```
aws pcs get-cluster --region region-code --cluster-identifier cluster-ident
```

コマンドは、この例のような出力を返します。

```
{
    "cluster": {
        "name": "independent-instance-demo",
        "id": "s3431v9rx2",
        "arn": "arn:aws:pcs:us-east-1:012345678901:cluster/s3431v9rx2",
        "status": "ACTIVE",
        "createdAt": "2024-07-12T15:32:27.225136+00:00",
        "modifiedAt": "2024-07-12T15:32:27.225136+00:00",
        "scheduler": {
            "type": "SLURM",
            "version": "23.11"
        },
        "size": "SMALL",
        "networking": {
            "subnetIds": [
                "subnet-0123456789abdef"
            ],
            "securityGroupIds": [
                "sg-0123456789abdef"
            ]
        },
        "endpoints": [
            {
                "type": "SLURMCTLD",
                "privateIpAddress": "10.3.149.220",
                "port": "6817"
            }
        ],
        "authKey": {
            "secretArn": "arn:aws:secretsmanager:us-east-1:123456789012:secret:pcs!
slurm-secret-s3431v9rx2-FN7tJFf",
            "secretVersion": "ff58d1fd-070e-4bbc-98a0-64ef967cebcc"
        }
    }
```

```
}
```

このサンプルでは、クラスター Slurm コントローラーエンドポイントの IP アドレスは で`10.3.149.220`、ポート で実行されています`6817`。`secretArn` は、後のステップでクラスターシークレットを取得するために使用します。IP アドレスとポートは、後のステップで`sackd`サービスの設定に使用されます。

ステップ 2 – EC2インスタンスを起動する

EC2 インスタンスを起動するには

1. Amazon EC2コンソール を開きます。
2. ナビゲーションペインで、[Instances] (インスタンス) を選択し、[Launch Instances] (インスタンスの起動) を選択して、新しいインスタンス起動ウィザードを開きます。
3. (オプション) 名前とタグ セクションで、 などのインスタンスの名前を指定しますPCS-`LoginNode`。指定した名前は、リソースタグとしてインスタンスに割り当てられます (`Name=PCS-LoginNode`)。
4. アプリケーションと OS イメージ セクションAMIで、 でサポートされているオペレーティングシステムの を選択します AWS PCS。詳細については、「サポートされるオペレーティングシステム」を参照してください。
5. インスタンスタイプ セクションで、サポートされているインスタンスタイプを選択します。詳細については、「サポートされるインスタンスタイプ」を参照してください。
6. 「キーペア」セクションで、インスタンスに使用するSSHキーペアを選択します。
7. ネットワーク設定 セクションで :

    - [編集] を選択します。

        i. クラスターVPCの を選択します AWS PCS。

        ii. [ファイアウォール (セキュリティグループ)] で、[既存のセキュリティグループを選択する] を選択します。

            A. インスタンスとターゲット AWS PCSクラスターの Slurm コントローラー間のトラフィックを許可するセキュリティグループを選択します。詳細については、「セキュリティグループの要件と考慮事項」を参照してください。

            B. (オプション) インスタンスへのインバウンドSSHアクセスを許可するセキュリティグループを選択します。

8. ストレージセクションで、必要に応じてストレージボリュームを設定します。アプリケーションとライブラリをインストールしてユースケースを有効にするには、必ず十分なスペースを設定してください。
9. 詳細 で、クラスターシークレットへのアクセスを許可する IAMロールを選択します。詳細については、「Slurm クラスターシークレットを取得する」を参照してください。
10. 概要ペインで、インスタンスの起動 を選択します。

ステップ 3 – インスタンスに Slurm をインストールする

インスタンスが起動してアクティブになったら、任意のメカニズムを使用してインスタンスに接続します。が提供する Slurm インストーラー AWS を使用して、インスタンスに Slurm をインストールします。詳細については、「Slurm インストーラ」を参照してください。

Slurm インストーラをダウンロードして解凍し、`installer.sh`スクリプトを使用して Slurm をインストールします。詳細については、「ステップ 3 – Slurm をインストールする」を参照してください。

ステップ 4 – クラスターシークレットを取得して保存する

これらの手順には が必要です AWS CLI。詳細については、「 バージョン 2 のユーザーガイド」の「 の最新バージョンのインストールまたは更新 AWS CLI」を参照してください。 AWS Command Line Interface

次のコマンドを使用してクラスターシークレットを保存します。

- Slurm の設定ディレクトリを作成します。

```
sudo mkdir -p /etc/slurm
```

- クラスターシークレットを取得、デコード、保存します。このコマンドを実行する前に、*region-code* をターゲットクラスターが実行されているリージョンに置き換え、*secret-arn* ステップ 1 でsecretArn取得した の値を持つ 。 ???

```
sudo aws secretsmanager get-secret-value \
    --region region-code \
    --secret-id 'secret-arn' \
    --version-stage AWSCURRENT \
    --query 'SecretString' \
    --output text | base64 -d > /etc/slurm/slurm.key
```

> **Warning**
>
> マルチユーザー環境では、インスタンスにアクセスできるすべてのユーザーがインスタンスメタデータサービス () にアクセスできる場合、クラスターシークレットを取得できる可能性がありますIMDS。これにより、他のユーザーを偽装できるようになります。へのアクセスをルートユーザーまたは管理ユーザーのみに制限IMDSすることを検討してください。または、インスタンスプロファイルに依存しない別のメカニズムを使用してシークレットを取得して設定することを検討してください。

- Slurm キーファイルに所有権とアクセス許可を設定します。

```
sudo chmod 0600 /etc/slurm/slurm.key
sudo chown slurm:slurm /etc/slurm/slurm.key
```

> **Note**
>
> Slurm キーは、sackdサービスが実行されるユーザーとグループによって所有されている必要があります。

ステップ 5 – クラスターへの接続を設定する AWS PCS

クラスターへの接続を確立するには、以下の手順に従ってシステムサービスsackdとして を起動します AWS PCS。

1. 次のコマンドを使用して、sackdサービスの環境ファイルを設定します。コマンドを実行する前に、*ip-address* また、*port* ステップ 1 でエンドポイントから取得した値を含む 。

```
sudo echo "SACKD_OPTIONS='--conf-server=ip-address:port'" > /etc/sysconfig/sackd
```

2. sackd プロセスを管理するためのsystemdサービスファイルを作成します。

```
sudo cat << EOF > /etc/systemd/system/sackd.service
[Unit]
Description=Slurm auth and cred kiosk daemon
After=network-online.target remote-fs.target
Wants=network-online.target
ConditionPathExists=/etc/sysconfig/sackd
```

```
[Service]
Type=notify
EnvironmentFile=/etc/sysconfig/sackd
User=slurm
Group=slurm
RuntimeDirectory=slurm
RuntimeDirectoryMode=0755
ExecStart=/opt/aws/pcs/scheduler/slurm-23.11/sbin/sackd --systemd \$SACKD_OPTIONS
ExecReload=/bin/kill -HUP \$MAINPID
KillMode=process
LimitNOFILE=131072
LimitMEMLOCK=infinity
LimitSTACK=infinity

[Install]
WantedBy=multi-user.target
EOF
```

3. sackd サービスファイルの所有権を設定します。

```
sudo chown root:root /etc/systemd/system/sackd.service && \
    sudo chmod 0644 /etc/systemd/system/sackd.service
```

4. sackd サービスを有効にします。

```
sudo systemctl daemon-reload && sudo systemctl enable sackd
```

5. sackd サービスを開始します。

```
sudo systemctl start sackd
```

ステップ 6 – (オプション) 接続をテストする

sackd サービスが実行されていることを確認します。サンプル出力を次に示します。エラーがある場合は、一般的にここに表示されます。

```
[root@ip-10-3-27-112 ~]# systemctl status sackd
[x] sackd.service - Slurm auth and cred kiosk daemon
   Loaded: loaded (/etc/systemd/system/sackd.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2024-07-16 16:34:55 UTC; 8s ago
 Main PID: 9985 (sackd)
```

```
   CGroup: /system.slice/sackd.service
           ##9985 /opt/aws/pcs/scheduler/slurm-23.11/sbin/sackd --systemd --conf-
server=10.3.149.220:6817

Jul 16 16:34:55 ip-10-3-27-112.ec2.internal systemd[1]: Starting Slurm auth and cred
 kiosk daemon...
Jul 16 16:34:55 ip-10-3-27-112.ec2.internal systemd[1]: Started Slurm auth and cred
 kiosk daemon.
Jul 16 16:34:55 ip-10-3-27-112.ec2.internal sackd[9985]: sackd: running
```

クラスターへの接続が、 sinfoや などの Slurm クライアントコマンドを使用して動作していることを確認しますsqueue。からの出力例を次に示しますsinfo。

```
[root@ip-10-3-27-112 ~]# /opt/aws/pcs/scheduler/slurm-23.11/bin/sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
all up infinite 4 idle~ compute-[1-4]
```

ジョブを送信することもできます。例えば、この例のようなコマンドは、クラスター内の 1 つのノードでインタラクティブジョブを起動します。

```
/opt/aws/pcs/scheduler/slurm-23.11/bin/srun --nodes=1 -p all --pty bash -i
```

AWS PCS ネットワーク

AWS PCS クラスターは Amazon で作成されますVPC。この章では、クラスターのスケジューラとノードのネットワークに関する以下のトピックについて説明します。

インスタンスを起動するサブネットを選択する場合を除き、EC2起動テンプレートを使用してコンピューティングノードグループのネットワーク AWS PCSを設定する必要があります。起動テンプレートの詳細については、「での Amazon EC2起動テンプレートの使用 AWS PCS」を参照してください。

トピック

- AWS PCS VPC およびサブネットの要件と考慮事項
- クラスターVPCの AWS PCSの作成
- のセキュリティグループ AWS PCS
- の複数のネットワークインターフェイス AWS PCS

- のEC2インスタンスのプレイスメントグループ AWS PCS
- での Elastic Fabric Adapter (EFA) の使用 AWS PCS

AWS PCS VPC およびサブネットの要件と考慮事項

クラスターを作成する AWS PCSときは、その にVPCサブネットを指定しますVPC。このトピックでは、クラスターで使用する およびサブネット (複数可) に関する特定の要件VPCと考慮事項の概要 AWS PCSを説明します。VPC で使用する がない場合は AWS PCS、 AWSが提供する AWS CloudFormation テンプレートを使用して作成できます。の詳細についてはVPCs、「Amazon VPC ユーザーガイド」の「仮想プライベートクラウド (VPC) 」を参照してください。

VPC の要件と考慮事項

クラスターを作成する場合、指定する VPCは、以下の要件と考慮事項を満たす必要があります。

- には、作成するクラスター、ノード、およびその他のクラスターリソースで使用できる十分な数の IP アドレスVPCが必要です。詳細については、「Amazon VPCユーザーガイド」の「 VPCsおよび サブネットの IP アドレス指定」を参照してください。
- にはホスト名DNSとDNS解決のサポートVPCが必要です。そうしないと、ノードはカスタマークラスターを登録できません。詳細については、「Amazon ユーザーガイドDNS」の「 の属性 VPC」を参照してください。 VPC
- では、 を使用してVPCエンドポイントが AWS PrivateLink に接続できるようにする必要がある VPC場合があります AWS PCSAPI。詳細については、「Amazon ユーザーガイド」の「 を使用して を サービスVPCに接続する AWS PrivateLink」を参照してください。 VPC

サブネットの要件と考慮事項

Slurm クラスターを作成すると、 AWS PCS は指定したサブネットに Elastic Network Interface(ENI) を作成します。このネットワークインターフェイスにより、スケジューラコントローラーと顧客 間の通信が可能になりますVPC。ネットワークインターフェイスにより、Slurm はお客様のアカウントにデプロイされたコンポーネントと通信することもできます。クラスターのサブネットは、作成時にのみ指定できます。

クラスターのサブネットの要件

クラスターの作成時に指定するサブネットは、次の要件を満たしている必要があります。

- サブネットには、 が使用する IP アドレスが少なくとも 1 つ必要です AWS PCS。

- サブネットは、 AWS Outposts、 AWS Wavelength、または AWS ローカルゾーンに存在できません。
- サブネットはパブリックでもプライベートでもかまいません。可能であれば、プライベートサブネットを指定することをお勧めします。パブリックサブネットは、インターネットゲートウェイへのルートを含むルートテーブルを持つサブネットです。プライベートサブネットは、インターネットゲートウェイへのルートを含まないルートテーブルを持つサブネットです。

ノードのサブネットの要件

ノードやその他のクラスターリソースは、クラスターの作成 AWS PCS時に指定したサブネット、および同じ 内の他のサブネットにデプロイできますVPC。

ノードとクラスターリソースをデプロイするサブネットは、次の要件を満たしている必要があります。

- サブネットに、すべてのノードとクラスターリソースをデプロイするのに十分な IP アドレスがあることを確認する必要があります。
- ノードをパブリックサブネットにデプロイする場合、そのサブネットはIPv4パブリックアドレスを自動的に割り当てる必要があります。
- ノードをデプロイするサブネットがプライベートサブネットで、そのルートテーブルにネットワークアドレス変換 (NAT) デバイス () へのルートが含まれていない場合はIPv4、 を使用してVPCエンドポイント AWS PrivateLink をカスタマー に追加しますVPC。VPC エンドポイントは、ノードが接続するすべての AWS サービスに必要です。必要なエンドポイントは、ノードが `registerNodeGroupInstances`APIアクションを呼び出せるようにする のみです AWS PCS。
- パブリックサブネットまたはプライベートサブネットのステータスは には影響しません AWS PCS。必要なエンドポイントにアクセスできる必要があります。

クラスターVPCの AWS PCSの作成

AWS Parallel Computing Service (VPC) 内のクラスター用に Amazon Virtual Private Cloud (Amazon) を作成できますAWS PCS。

Amazon VPCを使用して、定義した仮想ネットワークでVPCリソースを起動します。この仮想ネットワークは、お客様自身のデータセンターで運用されている従来のネットワークによく似ています。ただし、Amazon Web Services のスケーラブルなインフラストラクチャを使用できるというメリットがあります。本番VPCクラスターをデプロイする前に、Amazon VPCサービスを十分に理解して

おくことをお勧めします。詳細については、「作成者ビジュアルモード」の「Amazon VPCとは」を参照してください。Amazon VPC ユーザーガイド 。

PCS クラスター、ノード、およびサポートリソース (ファイルシステムやディレクトリサービスなど) は、Amazon 内にデプロイされますVPC。で既存の Amazon VPC を使用する場合はPCS、「」で説明されている要件を満たしている必要がありますAWS PCS VPC およびサブネットの要件と考慮事項。このトピックでは、 が提供する AWS CloudFormation テンプレートを使用してVPCPCS要件を満たす AWSを作成する方法について説明します。テンプレートをデプロイすると、テンプレートによって作成されたリソースを表示して、作成されたリソースとそれらのリソースの設定を正確に把握できます。

前提条件

Amazon VPC for を作成するにはPCS、Amazon VPCリソースを作成するために必要なIAMアクセス許可が必要です。これらのリソースはVPCs、、サブネット、セキュリティグループ、ルートテーブルとルート、インターネットとNATゲートウェイです。詳細については、「Amazon ユーザーガイド」の「パブリックサブネットVPCを使用して を作成する」を参照してください。 VPC Amazon の完全なリストを確認するにはEC2、「サービス認証リファレンス」の「Amazon のアクション、リソース、および条件キーEC2」を参照してください。

Amazon を作成する VPC

AWS リージョン を使用する URLに適した をコピーして貼り付けVPCて、 を作成しますPCS。テンプレートをダウンロード AWS CloudFormation して、自分でAWS CloudFormation コンソールにアップロードすることもできます。

- 米国東部 (バージニア北部) (us-east-1)

```
https://console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 米国東部 (オハイオ) (us-east-2)

```
https://console.aws.amazon.com/cloudformation/home?region=us-east-2#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 米国西部 (オレゴン) (us-west-2)

```
https://console.aws.amazon.com/cloudformation/home?region=us-west-2#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- テンプレートのみ

```
https://aws-hpc-recipes.s3.us-east-1.amazonaws.com/main/recipes/net/hpc_large_scale/
assets/main.yaml
```

VPC の Amazon を作成するには PCS

1. AWS CloudFormation コンソール で テンプレートを開きます。

   > **Note**
   >
   > これらはテンプレートにあらかじめ入力されているため、デフォルト値のままにしておくだけで済みます。

2. 「スタック名 を指定」、次に「スタック名 」で、「」と入力しますhpc-networking。
3. パラメータ に、次の詳細を入力します。

   a. VPCで、次に CidrBlockと入力します。 10.3.0.0/16

   b. サブネット A の下：

      i. 次にCidrPublicSubnet、「」と入力します。 10.3.0.0/20

      ii. 次にCidrPrivateSubnet、「」と入力します。 10.3.128.0/20

   c. サブネット B の下：

      i. 次に、CidrPublicSubnetB と入力します。 10.3.16.0/20

      ii. 次にCidrPrivateSubnet、「」と入力します。 10.3.144.0/20

   d. サブネット C の下：

      i. ProvisionSubnetsC の場合は、 を選択しますTrue。

> **Note**
>
> アベイラビリティーゾーンが 3 つ未満のリージョンVPCで を作成する場合、このオプションは に設定されている場合無視されますTrue。

ii. 次にCidrPublicSubnet、B と入力します。 `10.3.32.0/20`

iii. 次にCidrPrivateSubnet、「」と入力します。 `10.3.160.0/20`

4. 「機能」で、 がIAMリソース を作成するAWS CloudFormation 可能性があることを承認するチェックボックスをオンにします。

AWS CloudFormation スタックのステータスをモニタリングします。に達するとCREATE_COMPLETE、VPCリソースを使用できるようになります。

> **Note**
>
> AWS CloudFormation テンプレートが作成したすべてのリソースを表示するには、AWS CloudFormation コンソール を開きます。hpc-networking スタックを選択し、[リソース] タブを選択します。

のセキュリティグループ AWS PCS

Amazon のセキュリティグループは、インスタンスへのインバウンドトラフィックとアウトバウンドトラフィックを制御する仮想ファイアウォールEC2として機能します。コンピューティングノードグループの起動テンプレートを使用して、 AWS PCSインスタンスにセキュリティグループを追加または削除します。起動テンプレートにネットワークインターフェイスが含まれていない場合は、SecurityGroupIdsを使用してセキュリティグループのリストを指定します。起動テンプレートでネットワークインターフェイスを定義する場合は、 Groupsパラメータを使用して各ネットワークインターフェイスにセキュリティグループを割り当てる必要があります。起動テンプレートの詳細については、「での Amazon EC2起動テンプレートの使用 AWS PCS」を参照してください。

> **Note**
>
> 起動テンプレートのセキュリティグループ設定の変更は、コンピューティングノードグループの更新後に起動された新しいインスタンスにのみ影響します。

セキュリティグループの要件と考慮事項

AWS PCS は、クラスターの作成時に指定したサブネットにクロスアカウント Elastic Network Interface (ENI) を作成します。これにより、 によって管理されるアカウントで実行されているHPCスケジューラが提供されます。これは AWS、 によって起動されたEC2インスタンスと通信するためのパスです AWS PCS。スケジューラENIとクラスターEC2インスタンス間の双方向通信ENIを許可する のセキュリティグループを指定する必要があります。

これを実現する簡単な方法は、グループ内のすべてのメンバー間のすべてのポートで TCP/IP トラフィックを許可する、許容される自己参照セキュリティグループを作成することです。これは、クラスターとノードグループEC2インスタンスの両方にアタッチできます。

許容セキュリティグループ設定の例

[Rule type] (ルールタイプ)	プロトコル	ポート	送信元	デスティネーション
インバウンド	すべて	すべて	自分	
アウトバウンド	すべて	すべて		0.0.0.0/0
アウトバウンド	すべて	すべて		自分

これらのルールにより、すべてのトラフィックが Slurm コントローラーとノード間で自由に流れ、任意の宛先へのすべてのアウトバウンドトラフィックが許可され、EFAトラフィック が有効になります。

制限付きセキュリティグループ設定の例

クラスターとそのコンピューティングノード間の開いているポートを制限することもできます。Slurm スケジューラの場合、クラスターにアタッチされたセキュリティグループは、次のポートを許可する必要があります。

- 6817 – EC2インスタンス`slurmctld`から へのインバウンド接続を有効にする
- 6818 – EC2インスタンスで から `slurmd` の実行`slurmctld`へのアウトバウンド接続を有効にする

コンピューティングノードにアタッチされたセキュリティグループは、次のポートを許可する必要があります。

- 6817 – EC2インスタンスslurmctldからの へのアウトバウンド接続を有効にします。
- 6818 — ノードグループインスタンスslurmdの slurmctldと slurmd との間のインバウンドおよびアウトバウンド接続を有効にします
- 60001 ~ 63000 - サポートするノードグループインスタンス間のインバウンド接続とアウトバウンド接続 srun
- EFA ノードグループインスタンス間の トラフィック。詳細については、「Linux インスタンス用ユーザーガイド」のEFA「 が有効なセキュリティグループを準備する」を参照してください。
- ワークロードに必要なその他のノード間トラフィック

の複数のネットワークインターフェイス AWS PCS

一部のEC2インスタンスには複数のネットワークカードがあります。これにより、100 Gbps を超える帯域幅機能やパケット処理の改善など、より高いネットワークパフォーマンスを実現できます。複数のネットワークカードを持つインスタンスの詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の「Elastic Network Interface」を参照してください。

EC2 起動テンプレートにネットワークインターフェイスを追加して、 AWS PCSコンピューティングノードグループのインスタンス用に追加のネットワークカードを設定します。以下は、hpc7a.96xlarge インスタンスにある など、2 つのネットワークカードを有効にする起動テンプレートの例です。以下の詳細に注意してください。

- 各ネットワークインターフェイスのサブネットは、起動テンプレートを使用するコンピューティングノードグループを設定する AWS PCSときに選択したサブネットと同じである必要があります。
- プライマリネットワークデバイスでは、 SSHや HTTPSトラフィックなどの定期的なネットワーク通信が行われるため、 DeviceIndexを に設定することで確立されます0。他のネットワークインターフェイスの は DeviceIndexです1。プライマリネットワークインターフェイスは 1 つだけです。他のすべてのインターフェイスはセカンダリです。
- すべてのネットワークインターフェイスには一意の が必要ですNetworkCardIndex。起動テンプレートで定義されているように、順番に番号を付けることをお勧めします。
- 各ネットワークインターフェイスのセキュリティグループは、 を使用して設定されますGroups。この例では、インバウンドSSHセキュリティグループ (sg-*SshSecurityGroupId*) がプライマリネットワークインターフェイスに追加され、セキュリティグループがクラスター内通信 () を有効にしますsg-*ClusterSecurityGroupId*。最後に、インターネット (sg-*InternetOutboundSecurityGroupId*) へのアウトバウンド接続を許可するセキュリティグループが、プライマリインターフェイスとセカンダリインターフェイスの両方に追加されます。

```
{
    "NetworkInterfaces": [
        {
            "DeviceIndex": 0,
            "NetworkCardIndex": 0,
            "SubnetId": "subnet-SubnetId",
            "Groups": [
               "sg-SshSecurityGroupId",
               "sg-ClusterSecurityGroupId",
               "sg-InternetOutboundSecurityGroupId"
            ]
        },
        {
            "DeviceIndex": 1,
            "NetworkCardIndex": 1,
            "SubnetId": "subnet-SubnetId",
            "Groups": ["sg-InternetOutboundSecurityGroupId"]
        }
    ]
}
```

のEC2インスタンスのプレイスメントグループ AWS PCS

プレイスメントグループを使用して、EC2インスタンスのプレイスメントに影響を与え、インスタンスで実行されるワークロードのニーズに合わせてインスタンスを配置できます。

プレイスメントグループタイプ

- クラスター – 低レイテンシーの通信を最適化するために、インスタンスをアベイラビリティーゾーンで密接にパックします。
- パーティション — 論理パーティションにインスタンスを分散させ、回復力を最大化します。
- スプレッド — 少数のインスタンスが個別のハードウェアで起動することを厳密に強制します。これは回復性にも役立ちます。

詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の「Amazon EC2インスタンスのプレイスメントグループ」を参照してください。

Elastic Fabric Adapter () を使用するようにコンピューティングノードグループを設定する場合は、AWS PCSクラスタープレイスメントグループを含めることをお勧めしますEFA。

で動作するクラスタープレイスメントグループを作成するには EFA

1. コンピューティングノードグループのタイプクラスターを持つプレイスメントグループを作成します。
   - 次の AWS CLI コマンドを使用します。

     ```
     aws ec2 create-placement-group --strategy cluster --group-name PLACEMENT-GROUP-NAME
     ```

   - CloudFormation テンプレートを使用してプレイスメントグループを作成することもできます。詳細については、「 ユーザーガイド」の CloudFormation「 テンプレートの使用AWS CloudFormation 」を参照してください。以下からテンプレートをダウンロードしURL、CloudFormation コンソール にアップロードします。

     ```
     https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/enable_efa/assets/efa-placement-group.yaml
     ```

2. コンピューティングノードグループのEC2起動テンプレートにプレイスメントグループを含めます AWS PCS。

での Elastic Fabric Adapter (EFA) の使用 AWS PCS

Elastic Fabric Adapter (EFA) は、ハイパフォーマンスコンピューティング (HPC) と機械学習アプリケーションを高速化するためにEC2インスタンスにアタッチできる、 からの AWS 高性能なアドバンストネットワーク相互接続です。で クラスターで実行されているアプリケーションを有効にするEFAには、 AWS PCSEFA次のように を使用するようにコンピューティングノードグループインスタンスを設定 AWS PCSする必要があります。

目次

- （オプション）CloudFormation テンプレートを使用して EFAが有効な起動テンプレートを作成する

AWS PCS互換の EFAに をインストールする AMI

コンピューティングノードグループAMIで使用される AWS PCS には、EFAドライバーがインストールされ、ロードされている必要があります。EFA ソフトウェアがインストールされAMIたカスタムを構築する方法については、「」を参照してくださいのカスタム Amazon マシンイメージ (AMIs) AWS PCS。

EFAが有効なEC2インスタンスを特定する

を使用するにはEFA、コンピューティンググループで AWS PCS許可されるすべてのインスタンスタイプが をサポートしている必要がありEFA、 の数が同じである必要があります vCPUs（GPUs必要に応じて も）。EFA対応インスタンスのリストについては、「Amazon Elastic Compute Cloud ユーザーガイド」の「Amazon の HPCおよび ML ワークロード用の Elastic Fabric AdapterEC2」を参照してください。 を使用して、 をサポートするインスタンスタイプのリスト AWS CLI を表示することもできますEFA。置換 *region-code* は、 など AWS PCS、 AWS リージョン を使用する で使用しますus-east-1。

```
aws ec2 describe-instance-types \
   --region region-code \
   --filters Name=network-info.efa-supported,Values=true \
   --query "InstanceTypes[*].[InstanceType]" \
   --output text | sort
```

使用可能なネットワークインターフェイスの数を決定する

一部のEC2インスタンスには複数のネットワークカードがあります。これにより、複数の を持つことができますEFAs。詳細については、「の複数のネットワークインターフェイス AWS PCS」を参照してください。

EFA 通信をサポートするセキュリティグループを作成する

AWS CLI

次の AWS CLI コマンドを使用して、 をサポートするセキュリティグループを作成できますEFA。コマンドはセキュリティグループ ID を出力します。次の置換を行います。

- *region-code* – など AWS PCS、 を使用する を指定します AWS リージョン us-east-1。
- *vpc-id* – VPCに使用する の ID を指定します AWS PCS。
- *efa-group-name* – 選択したセキュリティグループの名前を指定します。

```
aws ec2 create-security-group \
    --group-name efa-group-name \
    --description "Security group to enable EFA traffic" \
    --vpc-id vpc-id \
    --region region-code
```

次のコマンドを使用して、インバウンドおよびアウトバウンドのセキュリティグループルールをアタッチします。次の置き換えを行います。

- *efa-secgroup-id* – 先ほど作成したEFAセキュリティグループの ID を指定します。

```
aws ec2 authorize-security-group-ingress \
    --group-id efa-secgroup-id \
    --protocol -1 \
    --source-group efa-secgroup-id

aws ec2 authorize-security-group-egress \
    --group-id efa-secgroup-id \
    --protocol -1 \
    --source-group efa-secgroup-id
```

CloudFormation template

CloudFormation テンプレートを使用して、 をサポートするセキュリティグループを作成できますEFA。次の からテンプレートをダウンロードしURL、AWS CloudFormation コンソール にアップロードします。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/enable_efa/assets/efa-sg.yaml
```

AWS CloudFormation コンソールでテンプレートを開いた状態で、次のオプションを入力します。

- スタック名を指定する

    - スタック名 に、 などの名前を入力しますefa-sg-stack。
- パラメータの下
    - にSecurityGroupName、 などの名前を入力しますefa-sg。
    - でVPC、 VPC を使用する を選択します AWS PCS。

CloudFormation スタックの作成を完了し、そのステータスをモニタリングします。セキュリティEFAグループが に達するCREATE_COMPLETEと、使用できるようになります。

(オプション) プレイスメントグループを作成する

クラスタープレイスメントグループEFAで を使用するすべてのインスタンスを起動して、それらの間の物理的な距離を最小限に抑えることをお勧めします。を使用するコンピューティングノードグループごとにプレイスメントグループを作成することをお勧めしますEFA。コンピューティングノードグループのプレイスメントグループのEC2インスタンスのプレイスメントグループ AWS PCSを作成するには、「」を参照してください。

EC2 起動テンプレートを作成または更新する

EFA ネットワークインターフェイスは、コンピューティングノードグループのEC2起動テンプレート AWS PCSで設定されます。ネットワークカードが複数ある場合は、複数EFAsを設定できます。EFA セキュリティグループとオプションのプレイスメントグループも起動テンプレートに含まれます。

以下は、hpc7a.96xlarge など、2 つのネットワークカードを持つインスタンスの起動テンプレートの例です。インスタンスは、クラスタープレイスメントグループ subnet-*SubnetID1*の で起動されますpg-*PlacementGroupId1*。

セキュリティグループは、特に各EFAインターフェイスに追加する必要があります。すべての には、EFAトラフィックを有効にするセキュリティグループ () EFAが必要ですsg-*EfaSecGroupId*。他のセキュリティグループ、特に SSHや などの通常のトラフィックを処理するセキュリティグループはHTTPS、プライマリネットワークインターフェイス (DeviceIndexの で指定) にのみアタッチする必要があります0。ネットワークインターフェイスが定義されている起動テンプレートは、SecurityGroupIdsパラメータを使用したセキュリティグループの設定をサポートしていません。設定するネットワークインターフェイスGroupsごとに の値を設定する必要があります。

```
{
    "Placement": {
```

```
        "GroupId": "pg-PlacementGroupId1"
    },
    "NetworkInterfaces": [
        {
            "DeviceIndex": 0,
            "InterfaceType": "efa",
            "NetworkCardIndex": 0,
            "SubnetId": "subnet-SubnetId1",
            "Groups": [
                "sg-SecurityGroupId1",
                "sg-EfaSecGroupId"
            ]
        },
        {
            "DeviceIndex": 1,
            "InterfaceType": "efa",
            "NetworkCardIndex": 1,
            "SubnetId": "subnet-SubnetId1"
            "Groups": ["sg-EfaSecGroupId"]
        }
    ]
}
```

コンピューティングノードグループを作成または更新する

同じ数の 、同じプロセッサアーキテクチャvCPUs、すべての が をサポートするインスタンスを使用して、コンピューティングノードグループを作成または更新 AWS PCSしますEFA。AMI EFA ソフトウェアがインストールされている を使用し、 EFA対応ネットワークインターフェイスを設定する起動テンプレートを使用するようにコンピューティングノードグループを設定します。

(オプション) テスト EFA

EFA ソフトウェアのインストールに含まれる `fi_pingpong` プログラムを実行することで、コンピューティングノードグループ内の 2 つのノード間の EFA対応通信をデモンストレーションできます。このテストが成功すると、 が正しく設定EFAされている可能性があります。

開始するには、コンピューティングノードグループに 2 つの実行中のインスタンスが必要です。コンピューティングノードグループが静的容量を使用している場合、使用可能なインスタンスが既にあるはずです。動的容量を使用するコンピューティングノードグループの場合、 `salloc` コマンドを使用して 2 つのノードを起動できます。という名前のキューhpc7gに関連付けられた という名前の動的ノードグループを持つクラスターの例を次に示しますall。

```
% salloc --nodes 2 -p all
salloc: Granted job allocation 6
salloc: Waiting for resource configuration
... a few minutes pass ...
salloc: Nodes hpc7g-[1-2] are ready for job
```

を使用して、割り当てられた 2 つのノードの IP アドレスを確認しますscontrol。次の例では、アドレスは 10.3.140.69用hpc7g-1、 10.3.132.211用ですhpc7g-2。

```
% scontrol show nodes hpc7g-[1-2]
NodeName=hpc7g-1 Arch=aarch64 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=64 CPUTot=64 CPULoad=0.00
   AvailableFeatures=hpc7g
   ActiveFeatures=hpc7g
   Gres=(null)
   NodeAddr=10.3.140.69 NodeHostName=ip-10-3-140-69 Version=23.11.8
   OS=Linux 5.10.218-208.862.amzn2.aarch64 #1 SMP Tue Jun 4 16:52:10 UTC 2024
   RealMemory=124518 AllocMem=0 FreeMem=110763 Sockets=64 Boards=1
   State=IDLE+CLOUD ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A
   Partitions=efa
   BootTime=2024-07-02T19:00:09 SlurmdStartTime=2024-07-08T19:33:25
   LastBusyTime=2024-07-08T19:33:25 ResumeAfterTime=None
   CfgTRES=cpu=64,mem=124518M,billing=64
   AllocTRES=
   CapWatts=n/a
   CurrentWatts=0 AveWatts=0
   ExtSensorsJoules=n/a ExtSensorsWatts=0 ExtSensorsTemp=n/a
   Reason=Maintain Minimum Number Of Instances [root@2024-07-02T18:59:00]
   InstanceId=i-04927897a9ce3c143 InstanceType=hpc7g.16xlarge

NodeName=hpc7g-2 Arch=aarch64 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=64 CPUTot=64 CPULoad=0.00
   AvailableFeatures=hpc7g
   ActiveFeatures=hpc7g
   Gres=(null)
   NodeAddr=10.3.132.211 NodeHostName=ip-10-3-132-211 Version=23.11.8
   OS=Linux 5.10.218-208.862.amzn2.aarch64 #1 SMP Tue Jun 4 16:52:10 UTC 2024
   RealMemory=124518 AllocMem=0 FreeMem=110759 Sockets=64 Boards=1
   State=IDLE+CLOUD ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A
   Partitions=efa
   BootTime=2024-07-02T19:00:09 SlurmdStartTime=2024-07-08T19:33:25
   LastBusyTime=2024-07-08T19:33:25 ResumeAfterTime=None
   CfgTRES=cpu=64,mem=124518M,billing=64
```

```
   AllocTRES=
   CapWatts=n/a
   CurrentWatts=0 AveWatts=0
   ExtSensorsJoules=n/a ExtSensorsWatts=0 ExtSensorsTemp=n/a
   Reason=Maintain Minimum Number Of Instances [root@2024-07-02T18:59:00]
   InstanceId=i-0a2c82623cb1393a7 InstanceType=hpc7g.16xlarge
```

SSH (または hpc7g-1) を使用して、いずれかのノード (この例では) に接続しますSSM。これは内部 IP アドレスであるため、 を使用する場合は、ログインノードの 1 つから接続する必要がある場合がありますSSH。また、インスタンスは、コンピューティングノードグループの起動テンプレートを使用して SSHキーで設定する必要があることに注意してください。

```
% ssh ec2-user@10.3.140.69
```

次に、サーバーfi_pingpongモードで を起動します。

```
/opt/amazon/efa/bin/fi_pingpong -p efa
```

2 番目のインスタンス () に接続しますhpc7g-2。

```
% ssh ec2-user@10.3.132.211
```

fi_pingpong クライアントモードで を実行し、 上のサーバーに接続しますhpc7g-1。以下の例のような出力が表示されます。

```
% /opt/amazon/efa/bin/fi_pingpong -p efa 10.3.140.69

bytes   #sent   #ack     total       time     MB/sec    usec/xfer   Mxfers/sec
64      10      =10      1.2k        0.00s      3.08      20.75       0.05
256     10      =10      5k          0.00s     21.24      12.05       0.08
1k      10      =10      20k         0.00s     82.91      12.35       0.08
4k      10      =10      80k         0.00s    311.48      13.15       0.08
[error] util/pingpong.c:1876: fi_close (-22) fid 0
```

（オプション）CloudFormation テンプレートを使用して EFAが有効な起動テンプレートを作成する

の設定にはいくつかの依存関係があるためEFA、コンピューティングノードグループの設定に使用できる CloudFormation テンプレートが用意されています。最大 4 つのネットワークカードを持つインスタンスをサポートします。複数のネットワークカードを持つインスタンスの詳細については、

「Amazon Elastic Compute Cloud ユーザーガイド」の「Elastic Network Interface」を参照してください。

次の から CloudFormation テンプレートをダウンロードしURL、 AWS リージョン を使用する の CloudFormation コンソールにアップロードします AWS PCS。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/enable_efa/assets/pcs-lt-efa.yaml
```

AWS CloudFormation コンソールでテンプレートを開いた状態で、次の値を入力します。テンプレートにはいくつかのデフォルトのパラメータ値が用意されるため、デフォルト値のままにしておくことができます。

- スタック名を指定する
  - スタック名 にわかりやすい名前を入力します。など、コンピューティングノードグループに選択する AWS PCS名前を組み込むことをお勧めします*NODEGROUPNAME*-efa-lt。
- パラメータの下
  - でNumberOfNetworkCards、ノードグループに含まれるインスタンス内のネットワークカードの数を選択します。
  - でVpcId、クラスターVPCが AWS PCSデプロイされている を選択します。
  - でNodeGroupSubnetId、 EFA対応インスタンスが起動VPCされるクラスター内のサブネットを選択します。
  - でPlacementGroupName、 フィールドを空白のままにして、ノードグループの新しいクラスタープレイスメントグループを作成します。使用する既存のプレイスメントグループがある場合は、ここにその名前を入力します。
  - でClusterSecurityGroupId、クラスター内の他のインスタンスと AWS PCS へのアクセスを許可するために使用するセキュリティグループを選択しますAPI。多くのお客様は、クラスター からデフォルトのセキュリティグループを選択しますVPC。
  - でSshSecurityGroupId、クラスター内のノードへのインバウンドSSHアクセスを許可するために使用しているセキュリティグループの ID を指定します。
  - でSshKeyName、クラスター内のノードにアクセスするためのSSHキーペアを選択します。
  - にはLaunchTemplateName、 などの起動テンプレートのわかりやすい名前を入力しますNODEGROUPNAME-efa-lt。名前は、 AWS リージョン を使用する AWS アカウント の に一意である必要があります AWS PCS。
- 機能の下

- がIAMリソース を作成する AWS CloudFormation 可能性があることを了承します。

CloudFormation スタックのステータスをモニタリングします。起動テンプレートに到達するCREATE_COMPLETEと、使用する準備が整います。で前述したように、コンピューティングノードグループで AWS PCS使用しますコンピューティングノードグループを作成または更新する。

でのネットワークファイルシステムの使用 AWS PCS

AWS Parallel Computing Service (AWS PCS) コンピューティングノードグループで起動されたノードにネットワークストレージボリュームをアタッチして、データとファイルを書き込んでアクセスできる永続的な場所を提供できます。 AWS サービスが提供するボリュームを使用できます。ボリュームには、Amazon Elastic File System (Amazon EFS) 、Amazon FSx for NetApp ONTAP、Amazon FSx for Open ZFS、Amazon FSx for Lustre 、および Amazon File Cache が含まれます。NFS サーバーなどの自己管理型ボリュームを使用することもできます。

このトピックでは、 でネットワーク化されたファイルシステムを使用するための考慮事項と例について説明します AWS PCS。

ネットワークファイルシステムの使用に関する考慮事項

さまざまなファイルシステムの実装の詳細は異なりますが、いくつかの一般的な考慮事項があります。

- 関連するファイルシステムソフトウェアをインスタンスにインストールする必要があります。例えば、Amazon FSx for Lustre を使用するには、適切なLustreパッケージが存在する必要があります。これは、コンピューティングノードグループに含めるか、インスタンスの起動時に実行されるスクリプトAMIを使用することで実現できます。
- 共有ストレージボリュームとコンピューティングノードグループインスタンスの間にネットワークルートが必要です。
- 共有ストレージボリュームとコンピューティングノードグループインスタンスの両方のセキュリティグループルールでは、関連するポートへの接続を許可する必要があります。
- ファイルシステムにアクセスするリソース全体で一貫したPOSIXユーザーおよびグループ名前空間を維持する必要があります。そうしないと、PCSクラスターで実行されるジョブやインタラクティブプロセスでアクセス許可エラーが発生する可能性があります。
- ファイルシステムのマウントは、EC2起動テンプレートを使用して行われます。ネットワークファイルシステムのマウント時にエラーやタイムアウトが発生すると、インスタンスがジョブを実行で

きなくなる可能性があります。これにより、予期しないコストが発生する可能性があります。起動テンプレートのデバッグの詳細については、「」を参照してください[での Amazon EC2起動テンプレートの使用 AWS PCS]。

ネットワークマウントの例

Amazon 、Amazon FSx for LustreEFS、Amazon FSx for Open 、ZFSおよび Amazon File Cache を使用してファイルシステムを作成できます。以下の関連セクションを展開して、各ネットワークマウントの例を確認してください。

Amazon EFS

ファイルシステムのセットアップ

Amazon EFS ファイルシステムを作成します。PCS コンピューティングノードグループインスタンスを起動する各アベイラビリティーゾーンにマウントターゲットがあることを確認します。また、各マウントターゲットが、PCSコンピューティングノードグループインスタンスからのインバウンドアクセスとアウトバウンドアクセスを許可するセキュリティグループに関連付けられていることを確認します。詳細については、「Amazon Elastic File System [ユーザーガイド」の「マウントターゲットとセキュリティグループ]」を参照してください。 Amazon Elastic File System

起動テンプレート

ファイルシステムのセットアップから、コンピューティングノードグループに使用する起動テンプレートにセキュリティグループ (複数可) を追加します。

Amazon EFS ファイルシステムをマウントするcloud-configメカニズムを使用するユーザーデータを含めます。このスクリプトの次の値を独自の詳細に置き換えます。

- *mount-point-directory* – Amazon をマウントする各インスタンスのパス EFS
- *filesystem-id* – ファイルシステムのEFSファイルシステム ID

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
```

```
  - amazon-efs-utils

runcmd:
  - mkdir -p /mount-point-directory
  - echo "filesystem-id:/ /mount-point-directory efs tls,_netdev" >> /etc/fstab
  - mount -a -t efs defaults

--==MYBOUNDARY==--
```

Amazon FSx for Lustre

ファイルシステムのセットアップ

VPC を使用する で FSx for Lustre ファイルシステムを作成します AWS PCS。ゾーン間転送を最小限に抑えるには、PCSコンピューティングノードグループインスタンスの大部分を起動するのと同じアベイラビリティーゾーンのサブネットに をデプロイします。ファイルシステムが、PCSコンピューティングノードグループインスタンスからのインバウンドアクセスとアウトバウンドアクセスを許可するセキュリティグループに関連付けられていることを確認します。セキュリティグループの詳細については、「Amazon for Lustre ユーザーガイド」の「Amazon によるファイルシステムのアクセスコントロールVPCFSx」を参照してください。

起動テンプレート

FSx for Lustre ファイルシステムのマウントcloud-configに を使用するユーザーデータを含めます。このスクリプトの次の値を独自の詳細に置き換えます。

- *mount-point-directory* — Lustre にマウントするインスタンスFSxのパス
- *filesystem-id* – FSx for Lustre ファイルシステムのファイルシステム ID
- *mount-name* – FSx for Lustre ファイルシステムのマウント名
- *region-code* – AWS リージョン FSx for Lustre ファイルシステムがデプロイされている (システムと同じ AWS PCSである必要があります)
- （オプション) *latest* – FSx for Lustre でLustreサポートされている の任意のバージョン

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"
```

```
runcmd:
- amazon-linux-extras install -y lustre=latest
- mkdir -p /mount-point-directory
- mount -t lustre filesystem-id.fsx.region-code.amazonaws.com@tcp:/mount-name /mount-
point-directory

--==MYBOUNDARY==
```

Amazon FSx for OpenZFS

ファイルシステムのセットアップ

VPC を使用する で FSx for OpenZFS ファイルシステムを作成します AWS PCS。ゾーン間転送を最小限に抑えるには、コンピューティングノードグループインスタンスの大部分を起動するのと同じアベイラビリティーゾーンのサブネットに をデプロイします AWS PCS。ファイルシステムが、コンピューティングノードグループインスタンスからの AWS PCSインバウンドアクセスとアウトバウンドアクセスを許可するセキュリティグループに関連付けられていることを確認します。セキュリティグループの詳細については、「 for Open User Guide」の「Managing file system access with AmazonVPC」を参照してください。 FSx ZFS

起動テンプレート

FSx for OpenZFS ファイルシステムのルートボリュームをマウントcloud-configするために が使用するユーザーデータを含めます。このスクリプトの次の値を独自の詳細に置き換えます。

- *mount-point-directory* – for OpenZFS Share をマウントするインスタンスFSxのパス
- *filesystem-id* – for FSx OpenZFS ファイルシステムのファイルシステム ID
- *region-code* – AWS リージョン FSx for OpenZFS ファイルシステムがデプロイされている (システムと同じ AWS PCSである必要があります)

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- mkdir -p /mount-point-directory
```

```
- mount -t nfs -o noatime,nfsvers=4.2,sync,rsize=1048576,wsize=1048576 filesystem-
id.fsx.region-code.amazonaws.com:/fsx/ /mount-point-directory

--==MYBOUNDARY==
```

Amazon File Cache

ファイルシステムのセットアップ

VPC を使用する に Amazon File Cache を作成します AWS PCS。ゾーン間転送を最小限に抑えるには、PCSコンピューティングノードグループインスタンスの大部分を起動するのと同じアベイラビリティーゾーン内のサブネットを選択します。ファイルキャッシュが、PCSインスタンスとファイルキャッシュ間のポート 988 でのインバウンドトラフィックとアウトバウンドトラフィックを許可するセキュリティグループに関連付けられていることを確認します。セキュリティグループの詳細については、「Amazon File Cache ユーザーガイド」の「Amazon によるキャッシュアクセスコントロールVPC」を参照してください。

起動テンプレート

ファイルシステムのセットアップから、コンピューティングノードグループに使用する起動テンプレートにセキュリティグループ (複数可) を追加します。

が Amazon File Cache のマウントcloud-configに使用するユーザーデータを含めます。このスクリプトの次の値を独自の詳細に置き換えます。

- *mount-point-directory* — Lustre にマウントするインスタンスFSxのパス
- *cache-dns-name* – ファイルキャッシュのドメインネームシステム (DNS) 名
- *mount-name* – ファイルキャッシュのマウント名

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- amazon-linux-extras install -y lustre=2.12
- mkdir -p /mount-point-directory
- mount -t lustre -o relatime,flock cache-dns-name@tcp:/mount-name /mount-point-
directory
```

```
--==MYBOUNDARY==
```

の Amazon マシンイメージ (AMIs ） AWS PCS

AWS PCS は、 が提供する AMIs と連携するため、クラスター内のノードにあるソフトウェアと設定に高い柔軟性がもたらされます。を試す場合は AWS PCS、 によってAMI提供され、 によって管理されているサンプルを使用できます AWS。本番環境で を使用している場合 AWS PCSは、独自の を構築することをお勧めしますAMIs。このトピックでは、サンプル を検出して使用する方法とAMIs、カスタマイズした独自の を構築して使用する方法について説明しますAMIs。

トピック

- でのサンプル Amazon マシンイメージ (AMIs) の使用 AWS PCS
- のカスタム Amazon マシンイメージ (AMIs) AWS PCS
- AMIs のカスタムを構築するためのソフトウェアインストーラー AWS PCS

でのサンプル Amazon マシンイメージ (AMIs) の使用 AWS PCS

AWS は、 を操作するための出発点として使用できるサンプルAMIsを提供します AWS PCS。

> **Important**
> サンプルAMIsはデモンストレーション用であり、本番ワークロードにはお勧めしません。

現在の AWS PCSサンプルを検索する AMIs

AWS Management Console

AWS PCS サンプルAMIsには次の命名規則があります。

```
aws-pcs-sample_ami-OS-architecture-schdeulder-scheduler-major-version
```

使用できる値

- *OS* – amzn2
- *architecture* – x86_64または arm64

- *scheduler* – slurm
- *scheduler-major-version* – 23.11

サンプルを検索する AWS PCSには AMIs

1. Amazon EC2コンソール を開きます。
2. に移動しますAMIs。
3. 「パブリックイメージ」を選択します。
4. 属性またはタグAMIで検索 で、テンプレート化された名前AMIを使用して を検索します。

   例

   - Graviton AMIをサポートする Slurm 23.11

     ```
     aws-pcs-sample_ami-amzn2-arm64-slurm-23.11
     ```

   - x86 インスタンスAMIのサンプル

     ```
     aws-pcs-sample_ami-amzn2-x86_64-slurm-23.11
     ```

   > **Note**
   >
   > が複数ある場合はAMIs、最新のタイムスタンプAMIで を使用します。

5. コンピューティングノードグループを作成または更新するときに AMI ID を使用します。

AWS CLI

最新の AWS PCSサンプルは、以下のコマンドAMIで確認できます。置換 *region-code* など AWS PCS、 AWS リージョン を使用する で使用しますus-east-1。

- x86_64

```
aws ec2 describe-images --region region-code --owners amazon 533267220047
 654654292779 654654317195 975050324343 \
--filters 'Name=name,Values=aws-pcs-sample_ami-amzn2-x86_64-slurm-23.11*' \
            'Name=state,Values=available' \
```

```
--query 'sort_by(Images, &CreationDate)[-1].[Name,ImageId]' --output text
```

- Arm64

```
aws ec2 describe-images --region region-code --owners amazon 533267220047
 654654292779 654654317195 975050324343 \
--filters 'Name=name,Values=aws-pcs-sample_ami-amzn2-arm64-slurm-23.11*' \
            'Name=state,Values=available' \
--query 'sort_by(Images, &CreationDate)[-1].[Name,ImageId]' --output text
```

コンピューティングノードグループを作成または更新するときに AMI ID を使用します。

サンプルの詳細 AWS PCS AMIs

コンテンツ、 AWS PCSサンプル の現在および以前のリリースの設定の詳細を表示するにはAMIs、「」を参照してくださいサンプルのリリースノート AWS PCS AMIs。

とAMIs互換性のある独自の を構築する AWS PCS

とAMIs連携する独自の を構築する方法については AWS PCS、「」を参照してくださいのカスタム Amazon マシンイメージ (AMIs) AWS PCS。

のカスタム Amazon マシンイメージ (AMIs) AWS PCS

AWS PCS は、サービスに持ち込む Amazon マシンイメージ (AMI) と連携するように設計されています。エージェントと互換性のあるバージョンの Slurm が正しくインストールおよび設定されている限り、これら AWS PCSには任意のソフトウェアと設定をインストールAMIsできます。 AWSが提供するインストーラを使用して、カスタム にソフトウェアをインストール AWS PCSする必要がありますAMI。 AWSが提供するインストーラを使用してカスタムに Slurm をインストールすることをお勧めしますAMIが、必要に応じて Slurm を自分でインストールできます (推奨されません)。

> **Note**
>
> カスタム を構築せずに試 AWS PCSす場合はAMI、 AMIが提供するサンプルを使用できます AWS。詳細については、「でのサンプル Amazon マシンイメージ (AMIs) の使用 AWS PCS」を参照してください。

このチュートリアルでは、PCSコンピューティングノードグループAMIで使用できる を作成して、HPCおよび AI/ML ワークロードを強化します。

トピック

ステップ 1 – 一時インスタンスを起動する

ソフトウェアと Slurm スケジューラのインストールと設定に使用できる一時インスタンスを起動します AWS PCS。このインスタンスを使用して、 とAMI互換性のある を作成します AWS PCS。

一時インスタンスを起動するには

1. Amazon EC2コンソール を開きます。
2. ナビゲーションペインで、インスタンス を選択し、インスタンスを起動 を選択して新しいインスタンス起動ウィザードを開きます。
3. (オプション) 名前とタグ セクションで、 などのインスタンスの名前を指定しますPCS-AMI-instance。指定した名前は、リソースタグとしてインスタンスに割り当てられます (Name=PCS-AMI-instance)。
4. アプリケーションと OS イメージ セクションAMIで、サポートされているオペレーティングシステム の 1 つに対して を選択します。
5. [Instance type] (インスタンスタイプ) セクションで、サポートされているインスタンスタイプを選択します。
6. [Key pair] (キーペア) セクションで、インスタンスに使用するキーペアを選択します。
7. ネットワーク設定 セクションで :

- ファイアウォール (セキュリティグループ) で、既存のセキュリティグループを選択 を選択し、インスタンスへのインバウンドSSHアクセスを許可するセキュリティグループを選択します。

8. [Storage] (ストレージ) セクションで、必要に応じてボリュームを設定します。独自のアプリケーションとライブラリをインストールするのに十分なスペースを設定してください。
9. [Summary] (サマリー) パネルで、[Launch instance] (インスタンスの起動) を選択します。

ステップ 2 – エージェントをインストールする AWS PCS

Slurm で使用するために によって AWS PCS起動されたインスタンスを設定する エージェントをインストールします。

AWS PCS エージェントをインストールするには

1. 起動したインスタンスに接続します。詳細については、「Linux インスタンスへの接続」を参照してください。
2. (オプション) すべてのソフトウェアパッケージが最新であることを確認するには、インスタンスでソフトウェアをすばやく更新します。このプロセスには数分かかることがあります。

   - Amazon Linux 2、RHEL9、Rocky Linux 9

     ```
     sudo yum update -y
     ```

   - Ubuntu 22.04

     ```
     sudo apt-get update && sudo apt-get upgrade -y
     ```

3. インスタンスを再起動して、そのインスタンスに再接続します。
4. エージェントのインストールファイルをダウンロードします AWS PCS。インストールファイルは圧縮された tarball (`.tar.gz`) ファイルにパッケージ化されます。次のコマンドを使用して、安定している最新バージョンをダウンロードします。置換 *`region`* など、一時インスタンスを起動 AWS リージョン した を持つ `us-east-1`。

   ```
   curl https://aws-pcs-repo-region.s3.amazonaws.com/aws-pcs-agent/aws-pcs-agent-
   v1.0.0-1.tar.gz -o aws-pcs-agent-v1.0.0-1.tar.gz
   ```

前のコマンドでバージョン番号を に置き換えることでlatest、最新バージョンを取得することもできます (例: aws-pcs-agent-v1-latest.tar.gz)。

> **Note**
>
> これは、エージェントソフトウェアの今後のリリース AWS PCSで変更される可能性があります。

5. (オプション) ソフトウェア tarball の信頼性と整合性を検証します AWS PCS。ソフトウェア発行元の ID を検証し、発行後にファイルの改変や破損がないことを確認するために、これを行うことをお勧めします。

   a. のパブリックGPGキー AWS PCSをダウンロードし、キーリングにインポートします。置換 *region* 一時インスタンスを起動 AWS リージョン した を使用します。コマンドはキーの値を返します。キー値を記録します。次のステップで使用します。

   ```
   wget https://aws-pcs-repo-public-keys-region.s3.amazonaws.com/aws-pcs-public-key.pub && \
       gpg --import aws-pcs-public-key.pub
   ```

   b. 次のコマンドを実行して、GPGキーのフィンガープリントを確認します。

   ```
   gpg --fingerprint 7EEF030EDDF5C21C
   ```

   コマンドは、以下と同じフィンガープリントを返す必要があります。

   ```
   1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
   ```

   > **Important**
   >
   > フィンガープリントが一致し AWS PCSない場合は、エージェントインストールスクリプトを実行しないでください。AWS Support にお問い合わせください。

   c. 署名ファイルをダウンロードし、ソフトウェア tarball ファイルの署名 AWS PCSを確認します。置換 *region* など、一時インスタンスを起動 AWS リージョン した を持つ us-east-1。

```
wget https://aws-pcs-repo-region.s3.amazonaws.com/aws-pcs-agent/aws-pcs-agent-
v1.0.0-1.tar.gz.sig && \
    gpg --verify ./aws-pcs-agent-v1.0.0-1.tar.gz.sig
```

出力は次の例のようになります:

```
gpg: assuming signed data in './aws-pcs-agent-v1.0.0-1.tar.gz'
gpg: Signature made Thu Aug  8 18:50:19 2024 CEST
gpg:                using RSA key 4BAA531875430EB0739E6D961BA7F0AF6E34C496
gpg: Good signature from "AWS PCS Packages (AWS PCS Packages)" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
   Subkey fingerprint: 4BAA 5318 7543 0EB0 739E  6D96 1BA7 F0AF 6E34 C496
```

結果に が含まれGood signature、フィンガープリントが前のステップで返されたフィンガープリントと一致する場合は、次のステップに進みます。

> **Important**
>
> フィンガープリントが一致し AWS PCSない場合は、ソフトウェアインストールスクリプトを実行しないでください。AWS Support にお問い合わせください。

6. 圧縮ファイルから.tar.gzファイルを抽出し、抽出されたディレクトリに移動します。

```
tar -xf aws-pcs-agent-v1.0.0-1.tar.gz && \
    cd aws-pcs-agent
```

7. AWS PCS ソフトウェアをインストールします。

```
sudo ./installer.sh
```

8. ソフトウェアバージョンファイルをチェック AWS PCSして、正常にインストールされたことを確認します。

```
cat /opt/aws/pcs/version
```

出力は次の例のようになります:

```
AGENT_INSTALL_DATE='Mon Aug 12 12:28:43 UTC 2024'
AGENT_VERSION='1.0.0'
AGENT_RELEASE='1'
```

ステップ 3 – Slurm をインストールする

と互換性のあるバージョンの Slurm をインストールします AWS PCS。

Slurm をインストールするには

1. ソフトウェアをインストール AWS PCSしたのと同じ一時インスタンスに接続します。
2. Slurm インストーラソフトウェアをダウンロードします。Slurm インストーラは、圧縮された tarball (`.tar.gz`) ファイルにパッケージ化されています。次のコマンドを使用して、安定している最新バージョンをダウンロードします。置換 *region* などの一時インスタンス AWS リージョン の を使用しますus-east-1。

   ```
   curl https://aws-pcs-repo-region.s3.amazonaws.com/aws-pcs-slurm/aws-pcs-
   slurm-23.11-installer-23.11.9-1.tar.gz \
       -o aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz
   ```

   前のコマンドでバージョン番号を に置き換えることでlatest、最新バージョンを取得することもできます (例: `aws-pcs-slurm-23.11-installer-latest.tar.gz`)。

   > **Note**
   >
   > これは、Slurm インストーラソフトウェアの今後のリリースで変更される可能性があります。

3. (オプション) Slurm インストーラ tarball の信頼性と整合性を検証します。ソフトウェア発行元の ID を検証し、発行後にファイルの改変や破損がないことを確認するために、これを行うことをお勧めします。

   a. のパブリックGPGキー AWS PCSをダウンロードし、キーリングにインポートします。置換 *region* 一時インスタンスを起動 AWS リージョン した を使用します。コマンドはキーの値を返します。キー値を記録します。次のステップで使用します。

```
wget https://aws-pcs-repo-public-keys-region.s3.amazonaws.com/aws-pcs-public-
key.pub && \
    gpg --import aws-pcs-public-key.pub
```

b. 次のコマンドを実行して、GPGキーのフィンガープリントを確認します。

```
gpg --fingerprint 7EEF030EDDF5C21C
```

コマンドは、以下と同じフィンガープリントを返す必要があります。

```
1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
```

> **Important**
>
> フィンガープリントが一致しない場合は、Slurm インストールスクリプトを実行しないでください。AWS Support にお問い合わせください。

c. 署名ファイルをダウンロードし、Slurm インストーラ tarball ファイルの署名を確認します。置換 *region* など、一時インスタンスを起動 AWS リージョン した を持つ us-east-1。

```
wget https://aws-pcs-repo-region.s3.amazonaws.com/aws-pcs-slurm/aws-pcs-
slurm-23.11-installer-23.11.9-1.tar.gz.sig && \
    gpg --verify ./aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz.sig
```

出力は次の例のようになります:

```
gpg: assuming signed data in './aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz'
gpg: Signature made Thu Aug  8 14:23:38 2024 CEST
gpg:                using RSA key 4BAA531875430EB0739E6D961BA7F0AF6E34C496
gpg: Good signature from "AWS PCS Packages (AWS PCS Packages)" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
   Subkey fingerprint: 4BAA 5318 7543 0EB0 739E  6D96 1BA7 F0AF 6E34 C496
```

結果に が含まれGood signature、フィンガープリントが前のステップで返されたフィンガープリントと一致する場合は、次のステップに進みます。

> **Important**
>
> フィンガープリントが一致しない場合は、Slurm インストールスクリプトを実行しないでください。AWS Support にお問い合わせください。

4. 圧縮された .tar.gz ファイルからファイルを展開し、展開されたディレクトリに移動します。

```
tar -xf aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz && \
    cd aws-pcs-slurm-23.11-installer
```

5. Slurm をインストールします。インストーラは、Slurm とその依存関係をダウンロード、コンパイル、インストールします。選択した一時インスタンスの仕様に応じて、数分かかります。

```
sudo ./installer.sh -y
```

6. スケジューラのバージョンファイルをチェックして、インストールを確認します。

```
cat /opt/aws/pcs/scheduler/slurm-23.11/version
```

出力は次の例のようになります:

```
SLURM_INSTALL_DATE='Mon Aug 12 12:38:56 UTC 2024'
SLURM_VERSION='23.11.9'
PCS_SLURM_RELEASE='1'
```

ステップ 4 – (オプション) 追加のドライバー、ライブラリ、アプリケーションソフトウェアをインストールする

一時インスタンスに追加のドライバー、ライブラリ、アプリケーションソフトウェアをインストールします。インストール手順は、特定のアプリケーションとライブラリによって異なります。AMI の AWS PCSカスタムを以前に構築したことがない場合は、まずソフトウェアと Slurm がインストールされているAMIだけで AWS PCS を構築してテストし、最初の成功を確認したら独自のソフトウェアと設定を段階的に追加することをお勧めします。

例

- Elastic Fabric Adapter (EFA) ソフトウェア。詳細については、「Amazon Elastic Compute Cloud ユーザーガイドMPI」の「Amazon でのHPCワークロードの EFAおよび の開始EC2方法」を参照してください。
- Amazon Elastic File System (Amazon EFS) クライアント。詳細については、「Amazon Elastic File System ユーザーガイド」の「Amazon EFSクライアントを手動でインストールする」を参照してください。 Amazon Elastic File System
- Amazon FSx for Lustre と Amazon File Cache を使用する Lustre クライアント。詳細については、 for Lustre ユーザーガイドの「Lustre クライアントFSxのインストール」を参照してください。
- CloudWatch ログとメトリクスを使用する Amazon CloudWatch エージェント。詳細については、「Amazon ユーザーガイド」の CloudWatch 「 エージェントのインストール」を参照してください。 CloudWatch
- AWS Neuron、trn* および inf* インスタンスタイプを使用します。詳細については、AWS 「 Neuron ドキュメント」を参照してください。
- NVIDIA p* または g* インスタンスタイプを使用するドライバーDCGM、、CUDAおよび 。

ステップ 5 – とAMI互換性のある を作成する AWS PCS

必要なソフトウェアコンポーネントをインストールしたら、コンピューティングノードグループで AWS PCSインスタンスを起動するために再利用AMIできる を作成します。

一時インスタンスAMIから を作成するには

1. Amazon EC2コンソール を開きます。
2. ナビゲーションペインで、[インスタンス] を選択します。
3. 作成した一時インスタンスを選択します。アクション、イメージ、イメージの作成 を選択します。
4. [イメージの作成] で、次を行います。
   a. イメージ名 には、 のわかりやすい名前を入力しますAMI。
   b. (オプション) イメージの説明 に、 の目的の簡単な説明を入力しますAMI。
   c. [イメージを作成] を選択します。
5. ナビゲーションペインで、 を選択しますAMIs。

6. リストで作成した AMI を見つけます。ステータスが「保留中」から「利用可能」に変わるのを待ってから、コンピューティングノードグループで AWS PCS使用します。

ステップ 6 – AWS PCSコンピューティングノードグループAMIでカスタム を使用する

カスタム は、新規または既存の AWS PCSコンピューティングノードグループAMIで使用できます。

New compute node group

カスタム を使用するには AMI

1. AWS PCS コンソール を開きます。
2. ナビゲーションペインで [Clusters] (クラスター) を選択します。
3. カスタム を使用するクラスターを選択しAMI、コンピューティングノードグループ を選択します。
4. 新しいコンピューティングノードグループを作成します。詳細については、「でのコンピューティングノードグループの作成 AWS PCS」を参照してください。AMI ID で、AMI使用するカスタムの名前または ID を検索します。コンピューティングノードグループの設定を終了し、コンピューティングノードグループの作成 を選択します。
5. (オプション) がインスタンスの起動AMIをサポートしていることを確認します。コンピューティングノードグループでインスタンスを起動します。これを行うには、コンピューティングノードグループを単一の静的インスタンスを持つように設定するか、コンピューティングノードグループを使用するキューにジョブを送信できます。

   a. 新しいコンピューティングノードグループ ID でタグ付けされたインスタンスが表示されるまで、Amazon EC2コンソールを確認します。詳細については、「」を参照してくださいでのコンピューティングノードグループインスタンスの検索 AWS PCS。

   b. インスタンスの起動が表示され、ブートストラッププロセスが完了したら、想定されるを使用していることを確認しますAMI。これを行うには、インスタンスを選択し、詳細で AMI ID を検査します。これは、コンピューティングノードグループ設定でAMI設定した と一致する必要があります。

   c. (オプション) コンピューティングノードグループのスケーリング設定を任意の値に更新します。

Existing compute node group

カスタム を使用するには AMI

1. AWS PCS コンソール を開きます。
2. ナビゲーションペインで [Clusters] (クラスター) を選択します。
3. カスタム を使用するクラスターを選択しAMI、コンピューティングノードグループ を選択します。
4. 設定するノードグループを選択し、編集 を選択します。AMI ID で、AMI使用するカスタムの名前または ID を検索します。コンピューティングノードグループの設定を終了し、 の更新 を選択します。コンピューティングノードグループで起動された新しいインスタンスは、更新された AMI ID を使用します。既存のインスタンスは、 によって置き換えられるAMIまで AWS PCS古い を引き続き使用します。詳細については、「コンピューティングノードグループの更新 AWS PCS 」を参照してください。
5. (オプション) がインスタンスの起動AMIをサポートしていることを確認します。コンピューティングノードグループでインスタンスを起動します。これを行うには、コンピューティングノードグループを単一の静的インスタンスを持つように設定するか、コンピューティングノードグループを使用するキューにジョブを送信できます。

   a. 新しいコンピューティングノードグループ ID でタグ付けされたインスタンスが表示されるまで、Amazon EC2コンソールを確認します。詳細については、「」を参照してくださいでのコンピューティングノードグループインスタンスの検索 AWS PCS。
   b. インスタンスの起動が表示され、ブートストラッププロセスが完了したら、想定されるを使用していることを確認しますAMI。これを行うには、インスタンスを選択し、詳細で AMI ID を検査します。これは、コンピューティングノードグループ設定でAMI設定した と一致する必要があります。
   c. (オプション) コンピューティングノードグループのスケーリング設定を任意の値に更新します。

ステップ 7 – 一時インスタンスを終了する

が で意図したとおりにAMI動作することを確認したら AWS PCS、一時インスタンスを終了して料金の発生を停止できます。

一時インスタンスを終了するには

1. Amazon EC2コンソール を開きます。

2. ナビゲーションペインで、[インスタンス] を選択します。
3. 作成した一時インスタンスを選択し、アクション 、インスタンスの状態 、インスタンスの終了 を選択します。
4. 確認を求められたら、 の終了 を選択します。

AMIs のカスタムを構築するためのソフトウェアインストーラー AWS PCS

AWS は、ソフトウェアを AWS PCSインスタンスにインストールできるダウンロード可能なファイルを提供します。 は、関連するバージョンの Slurm とその依存関係をダウンロード、コンパイル、およびインストールできるソフトウェア AWS も提供します。これらの手順を使用して、 AMIsで使用する AWS PCSカスタム を構築することも、独自の方法を使用することもできます。

目次

AWS PCS ソフトウェアインストーラー

AWS PCS ソフトウェアインストーラは、インスタンスブートストラッププロセス中に と AWS PCS連携するようにインスタンスを設定します。 AWSが提供するインストーラを使用して、カスタム にソフトウェアをインストール AWS PCSする必要がありますAMI。

Slurm インストーラ

Slurm インストーラは、関連するバージョンの Slurm とその依存関係をダウンロード、コンパイル、インストールします。Slurm インストーラを使用して、 AMIsのカスタムを構築できます AWS PCS。Slurm インストーラが提供するソフトウェア設定と整合性がある場合は、独自のメカニズムを使用することもできます。

AWSが提供するソフトウェアは、以下をインストールします。

- リクエストされたメジャーバージョンとメンテナンスバージョン (現在のバージョン 23.11.8) の Slurm - ライセンス GPL 2

  - Slurm は を --sysconfdirに設定して構築されます。 /etc/slurm
  - Slurm は、 オプション--enable-pamと を使用して構築されています。 --without-munge
  - Slurm は オプションで構築されています --sharedstatedir=/run/slurm/
  - Slurm は PMIX と JWT サポートで構築されています
  - Slurm が にインストールされている /opt/aws/pcs/schedulers/slurm-23.11
- OpenPMIX (バージョン 4.2.6) – ライセンス
  - Open PMIXは のサブディレクトリとしてインストールされます。 /opt/aws/pcs/scheduler/
- libjwt (バージョン 1.15.3) – ライセンス MPL-2.0
  - libjwt が のサブディレクトリとしてインストールされている /opt/aws/pcs/scheduler/

AWSが提供するソフトウェアは、システム設定を次のように変更します。

- ビルドによって作成された Slurm systemd ファイルは、ファイル名 /etc/systemd/system/でにコピーされますslurmd-23.11.service。
- 存在しない場合、Slurm ユーザーとグループ (slurm:slurm) は の UID/GID で作成されます401。
- Amazon Linux 2 および Rocky Linux 9 では、インストールによってEPELリポジトリが追加され、Slurm またはその依存関係を構築するために必要なソフトウェアがインストールされます。
- インストールRHEL9では、 codeready-builder-for-rhel-9-rhui-rpmsと epel-release-latest-9が Slurm またはその依存関係を構築するために必要なソフトウェアfedoraprojectをインストールできるようになります。

サポートされるオペレーティングシステム

ソフトウェアと AWS PCSSlurm インストーラは、次のオペレーティングシステムをサポートしています。

- Amazon Linux 2
- RedHat Enterprise Linux 9
- Rocky Linux 9
- Ubuntu 22.04

> **Note**
>
> AWS Deep Learning AMIs Amazon Linux 2 および Ubuntu 22.04 に基づく (DLAMI) バージョンは、ソフトウェアおよび Slurm インストーラと AWS PCS互換性がある必要があります。詳細については、「 AWS Deep Learning AMIs デベロッパーガイド」の「 の選択DLAMI」を参照してください。

サポートされるインスタンスタイプ

AWS PCS ソフトウェアおよび Slurm インストーラは、サポートされているオペレーティングシステムのいずれかを実行できる よりも、任意の x86_64 または arm64 インスタンスタイプをサポートします。

サポートされている Slurm バージョン

Slurm では、次のメジャーバージョンがサポートされています。

- Slurm 23.11

チェックサムを使用してインストーラを検証する

SHA256 チェックサムを使用して、インストーラの tarball (.tar.gz) ファイルを確認できます。ソフトウェア発行元の ID を確認し、発行後にアプリケーションの変更または破損がないことを確認するために、この操作を行うことをお勧めします。

tarball を検証するには

SHA256 チェックサムに sha256sum ユーティリティを使用し、tarball ファイル名を指定します。tarball ファイルを保存したディレクトリから コマンドを実行する必要があります。

- SHA256

```
$ sha256sum tarball_filename.tar.gz
```

コマンドは、次の形式でチェックサム値を返す必要があります。

```
checksum_value tarball_filename.tar.gz
```

コマンドによって返されるチェックサム値を、次の表に示すチェックサム値と比較します。チェックサムが一致した場合は、インストールスクリプトを安全に実行できます。

> **Important**
>
> チェックサムが一致しない場合は、インストールスクリプトを実行しないでください。AWS Support に連絡する。

例えば、次のコマンドは Slurm 23.11.9 tarball のSHA256チェックサムを生成します。

```
$ sha256sum aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz
```

出力例:

```
1de7d919c8632fe8e2806611bed4fde1005a4fadc795412456e935c7bba2a9b8 aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz
```

次の表に、最新バージョンのインストーラのチェックサムを示します。置換 *us-east-1* を、 AWS リージョン を使用する で使用します AWS PCS。

Installer (インストーラ)	ダウンロード URL	SHA256 チェックサム
スラム 23.11.9	`https://aws-pcs-repo- us-east-1 .s3.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz`	`1de7d919c8632fe8e2806611bed4fde1005a4fadc795412456e935c7bba2a9b8`
AWS PCSエージェント 1.0.0	`https://aws-pcs-repo- us-east-1 .s3.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.0.0-1.tar.gz`	`d2d3d68d00c685435c38af471d7e2492dde5ce9eb222d7b6ef0042144b134ce0`

の Slurm バージョン AWS PCS

SchedMD は、新機能、最適化、セキュリティパッチを使用して Slurm を継続的に強化します。SchedMD は定期的に新しいメジャーバージョンをリリースし、いつでも最大 3 つのバージョンをサポートする予定です。 AWS PCS は当初、Slurm 23.11 をサポートしています。新しいバージョンがリリースされたら、Slurm メジャーバージョンをアップグレードできます。 AWS PCS は、Slurm コントローラーをパッチバージョンで自動的に更新するように設計されています。

SchedMD が特定のメジャーバージョンのサポートを終了すると、 AWS PCS はそのメジャーバージョンのサポートも終了します。 AWS PCS は、Slurm メジャーバージョンがサポート終了に近づいた場合に事前通知を送信し、お客様がクラスターを新しいサポート対象バージョンにアップグレードするタイミングを把握できるようにします。

サポートされている最新の Slurm バージョンを使用してクラスターをデプロイし、最新の進歩と改善点にアクセスすることをお勧めします。

Slurm バージョンに関するよくある質問

は AWS PCS Slurm バージョンをどのくらいの期間サポートしていますか？

AWS PCS は、メジャーバージョンの SchedMD サポートサイクルに従います。 AWS PCS は、常に最大 3 つのメジャーバージョンをサポートします。SchedMD が新しいメジャーバージョンをリリースすると、 AWS PCS はサポートされている最も古いバージョンを廃止します。 AWS PCS は、Slurm の新しいメジャーバージョンをできるだけ早くリリースしますが、SchedMD リリースとでの可用性の間に遅延が生じる可能性があります AWS PCS。

Slurm バージョンのサポート終了 (EOSL) はいつ AWS PCS通知されますか？

AWS PCS は、EOSL日付の前に、事前に決められた頻度で複数回通知します。

Slurm バージョンが に近づいたらどうすればよいですかEOSL？

安全でサポートされている環境を維持するためにEOSL、 より前に Slurm バージョンを更新する必要があります。

新しいメジャーバージョンの Slurm を使用するようにクラスターを更新するにはどうすればよいですか？

Slurm バージョンを更新するには、新しいクラスターを作成する必要があります。また、 内の同等の AWS PCSソフトウェアにアップグレードAMIし、それを使用して新しいクラスターのコンピューティングノードグループを作成する必要があります。

クラスターで新しい Slurm パッチバージョンのリリースを取得する方法

AWS PCS は、Slurm Common Vulnerabilities and Exposures () に対応するためにパッチを自動的に適用するように設計されていますCVEs。 AWS PCS は、内部サービス所有アカウントで実行されるクラスターコントローラーにパッチを適用します。のEC2インスタンスにパッチをインストールするには、 AWS Management Console または AWS PCS APIアクションを使用する必要がありますAWS アカウント。

EOSL 日付までに Slurm を更新しない場合はどうなりますか？

AWS PCS は、サポートされていない Slurm バージョンのクラスターを停止するように設計されています。クラスターコントローラーの Slurm メジャーバージョンと、コンピューティングノードグループにインストールされている AWS PCSソフトウェアを更新する必要があります。

がサポートしている AWS PCS Slurm のバージョンはいくつですか？

AWS PCS は、現在のメジャーバージョンと 2 つの以前のメジャーバージョンを含め、常に最大 3 つのメジャー Slurm バージョンをサポートします。

どの Slurm バージョン更新を適用すればよいですか？

クラスター内のすべてのコンポーネントで同じメジャーバージョンを使用し、最新のパッチがリリースされたらすぐにインストールすることを強くお勧めします。AMIs コンピューティングノードグループの は、クラスターコントローラーの Slurm バージョンと互換性のあるバージョンの Slurm ソフトウェアを使用する必要があります。の Slurm メジャーバージョンは、クラスターコントローラーの Slurm メジャーバージョンの 2 つのバージョン内にあるAMIs必要があります。クラスター内の実行中のEC2インスタンスAMIにインストールされている Slurm バージョンを、クラスターコントローラーの Slurm バージョンより新しいものにすることはできません。クラスターのサポートを維持するには、 でサポートされている AWS PCSソフトウェアバージョンAMIsを使用する必要があります。

Slurm メジャーバージョンを更新しても、コンピューティングノードグループAMI用に で古い Slurm ソフトウェアを使用する場合はどうなりますか？

新しい Slurm 機能を使用するには、ソフトウェアを同じバージョンに更新 AWS PCSする必要があります。完全な AWS PCSサポートを受けるには、すべての Slurm コンポーネントでサポートされているバージョンを使用する必要があります。要約は、以下のとおりです。

- クラスターコントローラーと のすべてのコンポーネント (AWS PCS パッケージ) AWS アカウントの両方がサポートされているバージョンを使用する場合、フルサポートを提供できます。

- AWS PCS は、コントローラーの Slurm バージョンが に達した場合にクラスターを停止するように設計されていますEOSL。
- 内のコンポーネントの Slurm バージョンが AWS アカウント に到達した場合EOSL、クラスターはサポートされません。

クラスター内のコンポーネントはどの順序で更新すればよいですか？

新しい Slurm バージョンAMIで を使用する前に、クラスターコントローラーの Slurm バージョンを更新する必要があります。を使用するようにコンピューティングノードグループを更新しますAMI。AWS PCS は AMIを使用してコンピューティングノードグループで新しいEC2インスタンスを起動します。 AWS PCS は実行中のジョブがある既存のEC2インスタンスを更新しません。 AWS PCS はジョブの完了後にそれらのインスタンスを終了するように設計されています。

は AWS PCSSlurm バージョンの拡張サポートを提供していますか？

いいえ。追加コストや提供される特定のサポートカバレッジなど、拡張サポートオプションに関する詳細情報をお知らせします。

AWS Parallel Computing Service のセキュリティ

のクラウドセキュリティが最優先事項 AWS です。お客様は AWS 、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メリットを得られます。

セキュリティは、 AWS とユーザーの間で共有される責任です。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、 で AWS 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担います AWS クラウド。また、 は、安全に使用できるサービス AWS も提供します。コンプライアンスAWS プログラム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者は定期的にセキュリティの有効性をテストおよび検証。 AWS Parallel Computing Service 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AWS による対象範囲内のサービスコンプライアンスプログラム」を参照してください。
- クラウドのセキュリティ — お客様の責任は、使用する AWS サービスによって決まります。また、お客様は、データの機密性、会社の要件、適用される法律や規制など、その他の要因についても責任を負います。

このドキュメントは、 の使用時に責任共有モデルを適用する方法を理解するのに役立ちます AWS PCS。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の目的を達成するために を設定する AWS PCS方法を示します。また、 リソースのモニタリングや保護に役立つ他の AWS のサービスの使用方法についても AWS PCS説明します。

トピック

- AWS Parallel Computing Service でのデータ保護
- インターフェイスエンドポイント (AWS PrivateLink) を使用して AWS Parallel Computing Service にアクセスする
- AWS Parallel Computing Service の Identity and Access Management
- AWS Parallel Computing Service のコンプライアンス検証
- AWS Parallel Computing Service の耐障害性
- AWS Parallel Computing Service のインフラストラクチャセキュリティ
- AWS Parallel Computing Service での脆弱性の分析と管理
- サービス間での不分別な代理処理の防止

- AWS Parallel Computing Service のセキュリティのベストプラクティス

AWS Parallel Computing Service でのデータ保護

責任 AWS 共有モデル、 AWS Parallel Computing Service 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 はすべての 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があります AWS クラウド。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。また、使用する AWS のサービス のセキュリティ設定と管理タスクもユーザーの責任となります。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FAQ」を参照してください。欧州におけるデータ保護の詳細については、AWS 「 セキュリティブログ」のAWS 「責任共有モデル」とGDPRブログ記事を参照してください。

データ保護の目的で、認証情報を保護し AWS アカウント 、 AWS IAM Identity Center または AWS Identity and Access Management () を使用して個々のユーザーを設定することをお勧めしますIAM。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権限のみが各ユーザーに付与されます。また、次の方法でデータを保護することもお勧めします:

- 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。1TLS.2 が必要で、1.3 TLS をお勧めします。
- を使用して APIとユーザーアクティビティのログ記録を設定します AWS CloudTrail。
- AWS 暗号化ソリューションと、 内のすべてのデフォルトのセキュリティコントロールを使用します AWS のサービス。
- Amazon Macie などの高度なマネージドセキュリティサービスを使用します。これらは、Amazon S3 に保存されている機密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 AWS を介して にアクセスするときに FIPS 140-3 検証済みの暗号化モジュールが必要な場合はAPI、FIPSエンドポイントを使用します。利用可能な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) 140-3」を参照してください。

お客様の E メールアドレスなどの極秘または機密情報は、タグ、または名前フィールドなどの自由形式のテキストフィールドに配置しないことを強くお勧めします。これは、コンソール、、 AWS PCSまたは を使用して または他の AWS のサービス を操作する場合API AWS CLIも同様です AWS SDKs。名前に使用する自由記述のテキストフィールドやタグ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。URL を外部サーバーに提供する場合は、そのサーバーへのリクエストを検証URLするために認証情報を に含めないことを強くお勧めします。

保管中の暗号化

、、、APIまたは を使用して AWS Parallel Computing Service (AWS PCS) クラスターを作成すると AWS Management Console AWS CLI、 AWS PCS保管中のデータに対して暗号化がデフォルトで有効になります AWS SDKs。 AWS PCS は、 AWS 所有KMSキーを使用して保管中のデータを暗号化します。詳細については、「 AWS KMS デベロッパーガイド」の「カスタマーキーと AWS キー」を参照してください。クラスターシークレットは に保存 AWS Secrets Manager され、Secrets Manager マネージドKMSキーで暗号化されます。詳細については、「でのクラスターシークレットの使用 AWS PCS」を参照してください。

クラスターでは AWS PCS、次のデータは保管中です。

- スケジューラの状態 — 実行中のジョブとクラスター内のプロビジョニングされたノードに関するデータが含まれます。これは、Slurm が でStateSaveLocation定義されている に保持するデータですslurm.conf。詳細については、Slurm ドキュメントStateSaveLocationの の説明を参照してください。 AWS PCS はジョブの完了後にジョブデータを削除します。
- スケジューラ認証シークレット — これを使用して、クラスター内のすべてのスケジューラ通信を認証します。 AWS PCS

スケジューラの状態情報については、 AWS PCS はデータとメタデータをファイルシステムに書き込む前に自動的に暗号化します。暗号化されたファイルシステムは、保管中のデータに業界標準の AES-256 暗号化アルゴリズムを使用します。

転送中の暗号化

への接続では、 AWS PCS AWS Command Line Interface （ AWS CLI) と のどちらを使用するかに関係なく、署名バージョン 4 の署名プロセスによるTLS暗号化APIが使用されます AWS SDKs。詳細については、「 AWS Identity and Access Management ユーザーガイドAWS API」の「リクエストの署名」を参照してください。 は、接続に使用するセキュリティ認証情報のIAMポリシーAPIを使用して、 を通じてアクセスコントロール AWS を管理します。

AWS PCS は TLS を使用して他の AWS サービスに接続します。

Slurm クラスター内では、スケジューラはすべてのスケジューラ通信にauth/slurm認証を提供する認証プラグインで設定されます。Slurm は、通信のアプリケーションレベルで暗号化を提供しません。クラスターインスタンスをまたいで流れるすべてのデータは、 にローカルに留まるため EC2VPC、それらのインスタンスが転送中のVPC暗号化をサポートしている場合、暗号化の対象となります。詳細については、「Amazon Elastic Compute Cloud ユーザーガイド」の「転送中の暗号

化」を参照してください。 通信は、アカウント内のクラスターノードのコントローラー (サービスアカウントでプロビジョニング) 間で暗号化されます。

キー管理

AWS PCS は、 AWS 所有KMSキーを使用してデータを暗号化します。詳細については、「 AWS KMS デベロッパーガイド」の「カスタマーキーと AWS キー」を参照してください。クラスターシークレットは に保存 AWS Secrets Manager され、Secrets Manager マネージドKMSキーで暗号化されます。詳細については、「でのクラスターシークレットの使用 AWS PCS」を参照してください。

ネットワーク間トラフィックのプライバシー

AWS PCS クラスターの コンピューティングリソースは、お客様のアカウントVPCの 1 内にあります。したがって、クラスター内のすべての内部 AWS PCSサービストラフィックは AWS ネットワーク内にとどまり、インターネットを経由しません。ユーザーと AWS PCSノード間の通信はインターネットを経由できるため、 SSHまたは Systems Manager を使用してノードに接続することをお勧めします。詳細については、「 ユーザーガイド」の「 とは AWS Systems ManagerAWS Systems Manager 」を参照してください。

以下のサービスを使用して、オンプレミスネットワークを に接続することもできます AWS。

- AWS Site-to-Site VPN。 詳細については、「 ユーザーガイド」の AWS Site-to-Site VPN「 とはAWS Site-to-Site VPN 」を参照してください。
- AWS Direct Connect。詳細については、「 ユーザーガイド」の AWS Direct Connect「 とはAWS Direct Connect 」を参照してください。

にアクセスしてAPI、 AWS PCSサービスの管理タスクを実行します。ユーザーとユーザーは Slurm エンドポイントポートにアクセスして、スケジューラと直接やり取りします。

API トラフィックの暗号化

にアクセスするにはAPI、クライアントが AWS PCS Transport Layer Security (TLS) 1.2 以降をサポートしている必要があります。1TLS.2 が必要で、1.3 TLS をお勧めします。クライアントは、エフェメラル Diffie-Hellman (PFS) や楕円曲線 Diffie-Hellman Ephemeral () など、Perfect Forward Secrecy (DHE) を使用する暗号スイートもサポートする必要がありますECDHE。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。さらに、リクエストは、プリンIAMシパルに関連付けられたアクセスキー ID と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。 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。

データトラフィックの暗号化

転送中のデータの暗号化は、スケジューラエンドポイントにアクセスするサポートされているEC2インスタンスから、および 内から ComputeNodeGroup インスタンス間で有効になります AWS クラウド。詳細については、「転送中の暗号化」を参照してください。

インターフェイスエンドポイント (AWS PrivateLink) を使用して AWS Parallel Computing Service にアクセスする

を使用して AWS PrivateLink 、 VPCと AWS Parallel Computing Service () の間にプライベート接続を作成できますAWS PCS。インターネットゲートウェイ、NATデバイスVPC、VPN接続、または AWS Direct Connect 接続を使用せずに、 にあるかのように にアクセスできます AWS PCS。のインスタンスは、パブリック IP アドレスがVPCなくても にアクセスできます AWS PCS。

このプライベート接続を確立するには、 AWS PrivateLinkを利用したインターフェイスエンドポイントを作成します。インターフェイスエンドポイントに対して有効にする各サブネットにエンドポイントネットワークインターフェイスを作成します。これらは、 宛てのトラフィックのエントリポイントとして機能するリクエスタ管理のネットワークインターフェイスです AWS PCS。

詳細については、「 AWS PrivateLink ガイド」の「 AWS のサービス によるアクセス AWS PrivateLink」を参照してください。

に関する考慮事項 AWS PCS

のインターフェイスエンドポイントを設定する前に AWS PCS、「 AWS PrivateLink ガイド」の「インターフェイスVPCエンドポイントを使用して AWSサービスにアクセスする」を参照してください。

AWS PCS は、インターフェイスエンドポイントを介したすべてのAPIアクションの呼び出しをサポートします。

にインターネットへの直接アクセスVPCがない場合は、コンピューティングノードグループインスタンスが `RegisterComputeNodeGroupInstance`APIアクションを呼び AWS PCS出せるようにVPCエンドポイントを設定する必要があります。

のインターフェイスエンドポイントを作成する AWS PCS

Amazon VPCコンソールまたは AWS Command Line Interface () を使用して、 用の AWS PCSインターフェイスエンドポイントを作成できますAWS CLI。詳細については、「AWS PrivateLink ガイド」の「インターフェイスエンドポイントを作成」を参照してください。

次のサービス名を使用して、 用の AWS PCSインターフェイスエンドポイントを作成します。

```
com.amazonaws.region.pcs
```

置換 *region* は、 など、エンドポイント AWS リージョン を作成する の ID で指定しますus-east-1。

インターフェイスエンドポイントDNSのプライベートを有効にすると、デフォルトのリージョンDNS名 AWS PCSを使用して にAPIリクエストを行うことができます。例えば、pcs.us-east-1.amazonaws.com と指定します。

インターフェイスエンドポイントのエンドポイントポリシーを作成する

エンドポイントポリシーは、インターフェイスエンドポイントにアタッチできる IAMリソースです。デフォルトのエンドポイントポリシーでは、インターフェイスエンドポイントを介した へのAWS PCSフルアクセスが許可されます。から に AWS PCS許可されるアクセスを制御するにはVPC、インターフェイスエンドポイントにカスタムエンドポイントポリシーをアタッチします。

エンドポイントポリシーは、以下の情報を指定します。

- アクションを実行できるプリンシパル (AWS アカウント、IAMユーザー、IAMロール) 。
- 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。

詳細については、AWS PrivateLink ガイドのControl access to services using endpoint policies (エンドポイントポリシーを使用してサービスへのアクセスをコントロールする)を参照してください。

例: アクションのVPC AWS PCSエンドポイントポリシー

以下は、カスタムエンドポイントポリシーの例です。このポリシーをインターフェイスエンドポイントにアタッチすると、指定された を持つクラスターに、すべてのプリンシパルにリストされた AWS PCSアクションへのアクセスが付与されます。*cluster-id*。 置き換え *region* など、クラスター

の AWS リージョン の ID を持つ us-east-1。置換 *account-id* クラスター AWS アカウント の数。

```
{
    "Statement": [
            {
                "Action": [
                "pcs:CreateCluster",
                "pcs:ListClusters",
                "pcs:DeleteCluster",
                "pcs:GetCluster",
                ],
                "Effect": "Allow",
                "Principal": "*",
                "Resource": [
                    "arn:aws:pcs:region:account-id:cluster/cluster-id*"
                ]
            }
        ]
}
```

AWS Parallel Computing Service 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安全に制御 AWS のサービス するのに役立つ です。IAM 管理者は、誰を認証 (サインイン) し、誰に リソースの使用 AWS PCSを承認する (アクセス許可を付与する) かを制御します。IAM は追加料金なしで AWS のサービス 使用できる です。

トピック

- 対象者
- アイデンティティを使用した認証
- ポリシーを使用したアクセスの管理
- AWS Parallel Computing Service と の連携方法 IAM
- AWS Parallel Computing Service のアイデンティティベースのポリシーの例
- AWSAWS Parallel Computing Service の マネージドポリシー

- AWS PCS のサービスリンクロール
- の Amazon EC2 スポットロール AWS PCS
- の最小アクセス許可 AWS PCS
- IAM AWS Parallel Computing Service の インスタンスプロファイル
- AWS Parallel Computing Service のアイデンティティとアクセスのトラブルシューティング

対象者

AWS Identity and Access Management （ IAM) の使用方法は、 で行う作業によって異なります AWS PCS。

サービスユーザー – サービスを使用して AWS PCSジョブを実行する場合、管理者から必要な認証情報とアクセス許可が与えられます。さらに多くの AWS PCS機能を使用して作業を行う場合は、追加のアクセス許可が必要になることがあります。アクセスの管理方法を理解しておくと、管理者に適切な許可をリクエストするうえで役立ちます。の機能にアクセスできない場合は、 AWS PCS「」を参照してくださいAWS Parallel Computing Service のアイデンティティとアクセスのトラブルシューティング。

サービス管理者 – 社内のリソースを担当 AWS PCSしている場合は、通常、 へのフルアクセスがあります AWS PCS。サービスユーザーがどの AWS PCS機能やリソースにアクセスするかを決めるのは管理者の仕事です。次に、サービスユーザーのアクセス許可を変更するリクエストをIAM管理者に送信する必要があります。このページの情報を確認して、 の基本概念を理解してくださいIAM。会社で を と使用する方法の詳細については、IAM AWS PCS「」を参照してくださいAWS Parallel Computing Service と の連携方法 IAM。

IAM 管理者 - IAM管理者は、 へのアクセスを管理するポリシーの作成方法の詳細について確認する場合があります AWS PCS。で使用できるアイデンティティベースのポリシーの例 AWS PCSを表示するにはIAM、「」を参照してくださいAWS Parallel Computing Service のアイデンティティベースのポリシーの例。

アイデンティティを使用した認証

認証とは、ID 認証情報 AWS を使用して にサインインする方法です。として、IAMユーザーとしてAWS アカウントのルートユーザー、または IAMロールを引き受けることによって認証 （ にサインイン AWS) される必要があります。

ID ソースを介して提供された認証情報を使用して、フェデレーティッド ID AWS として にサインインできます。 AWS IAM Identity Center （ IAM Identity Center) ユーザー、会社のシングルサインオン

認証、Google または Facebook の認証情報は、フェデレーティッド ID の例です。フェデレーティッド ID としてサインインすると、管理者は以前に IAMロールを使用して ID フェデレーションをセットアップしていました。フェデレーション AWS を使用して にアクセスすると、間接的にロールを引き受けることになります。

ユーザーのタイプに応じて、 AWS Management Console または AWS アクセスポータルにサインインできます。へのサインインの詳細については AWS、「 ユーザーガイド」の「 へのサインイン AWS アカウント方法AWS サインイン 」を参照してください。

AWS プログラムで にアクセスする場合、 はソフトウェア開発キット (SDK) とコマンドラインインターフェイス (CLI) AWS を提供し、認証情報を使用してリクエストに暗号で署名します。 AWS ツールを使用しない場合は、リクエストに自分で署名する必要があります。推奨される方法を使用してリクエストを自分で署名する方法の詳細については、「 IAMユーザーガイド」のAWS API「リクエストの署名」を参照してください。

使用する認証方法を問わず、追加セキュリティ情報の提供をリクエストされる場合もあります。例えば、 AWS では、アカウントのセキュリティを高めるために多要素認証 (MFA) を使用することをお勧めします。詳細については、「 ユーザーガイド」の「多要素認証」および「 ユーザーガイド」の「 での多要素認証 (MFA) AWS IAM の使用」を参照してください。 AWS IAM Identity Center

AWS アカウント ルートユーザー

を作成するときは AWS アカウント、アカウント内のすべての およびリソースへの AWS のサービス 完全なアクセス権を持つ 1 つのサインインアイデンティティから始めます。この ID は AWS アカウント ルートユーザーと呼ばれ、アカウントの作成に使用した E メールアドレスとパスワードでサインインすることでアクセスできます。日常的なタスクには、ルートユーザーを使用しないことを強くお勧めします。ルートユーザーの認証情報は保護し、ルートユーザーでしか実行できないタスクを実行するときに使用します。ルートユーザーとしてサインインする必要があるタスクの完全なリストについては、「 IAMユーザーガイド」の「ルートユーザーの認証情報を必要とするタスク」を参照してください。

フェデレーティッドアイデンティティ

ベストプラクティスとして、管理者アクセスを必要とするユーザーを含む人間のユーザーに、一時的な認証情報を使用して にアクセスするための ID プロバイダーとのフェデレーションの使用を要求 AWS のサービス します。

フェデレーション ID は、エンタープライズユーザーディレクトリ、ウェブ ID プロバイダー、 AWS Directory Service、Identity Center ディレクトリ、または ID ソースを通じて提供された認証情報

AWS のサービス を使用して にアクセスするユーザーです。フェデレーティッド ID が にアクセスすると AWS アカウント、ロールを引き受け、ロールは一時的な認証情報を提供します。

アクセスを一元管理する場合は、 AWS IAM Identity Centerを使用することをお勧めします。IAM Identity Center でユーザーとグループを作成することも、独自の ID ソース内のユーザーとグループのセットに接続して同期して、すべての AWS アカウント とアプリケーションで使用することもできます。IAM Identity Center の詳細については、「 ユーザーガイド」のIAM「Identity Center とはAWS IAM Identity Center 」を参照してください。

IAM ユーザーとグループ

IAM ユーザーは、単一のユーザーまたはアプリケーションに対して特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。可能であれば、パスワードやアクセスキーなどの長期的な認証情報を持つIAMユーザーを作成する代わりに、一時的な認証情報を使用することをお勧めします。ただし、IAMユーザーとの長期的な認証情報を必要とする特定のユースケースがある場合は、アクセスキーをローテーションすることをお勧めします。詳細については、「 ユーザーガイド」の「長期的な認証情報を必要とするユースケースでアクセスキーを定期的にローテーションするIAM」を参照してください。

IAM グループは、IAMユーザーのコレクションを指定する ID です。グループとしてサインインすることはできません。グループを使用して、複数のユーザーに対して一度に権限を指定できます。多数のユーザーグループがある場合、グループを使用することで権限の管理が容易になります。例えば、という名前のグループIAMAdminsを作成し、そのグループにIAMリソースを管理するアクセス許可を付与できます。

ユーザーは、ロールとは異なります。ユーザーは 1 人の人または 1 つのアプリケーションに一意に関連付けられますが、ロールはそれを必要とする任意の人が引き受けるようになっています。ユーザーには永続的な長期の認証情報がありますが、ロールでは一時認証情報が提供されます。詳細については、「 ユーザーガイド」のIAM「 (ロールではなく) ユーザーを作成する場合IAM」を参照してください。

IAM ロール

IAM ロールは、特定のアクセス許可 AWS アカウント を持つ 内のアイデンティティです。これは IAM ユーザーと似ていますが、特定のユーザーに関連付けられていません。IAM ロール を切り替える AWS Management Console ことで、 で ロールを一時的に引き受けることができます。ロールを引き受けるには、 または AWS API オペレーションを AWS CLI 呼び出すか、カスタム を使用しますURL。ロールの使用方法の詳細については、「 ユーザーガイド」のIAM「ロールの使用IAM」を参照してください。

IAM 一時的な認証情報を持つ ロールは、以下の状況で役立ちます。

- フェデレーションユーザーアクセス – フェデレーティッド ID に許可を割り当てるには、ロールを作成してそのロールの許可を定義します。フェデレーティッド ID が認証されると、その ID はロールに関連付けられ、ロールで定義されている許可が付与されます。フェデレーションのロールの詳細については、「 ユーザーガイド」の「サードパーティー ID プロバイダーのロールの作成IAM」を参照してください。IAM Identity Center を使用する場合は、アクセス許可セットを設定します。ID が認証後にアクセスできる内容を制御するために、IAMIdentity Center はアクセス許可セットを のロールに関連付けますIAM。アクセス許可セットの詳細については、「AWS IAM Identity Center ユーザーガイド」の「アクセス許可セット」を参照してください。
- 一時的なIAMユーザーアクセス許可 – IAM ユーザーまたはロールは、 IAMロールを引き受けて、特定のタスクに対して異なるアクセス許可を一時的に引き受けることができます。
- クロスアカウントアクセス – IAMロールを使用して、別のアカウントのユーザー (信頼されたプリンシパル) がアカウントのリソースにアクセスすることを許可できます。クロスアカウントアクセスを許可する主な方法は、ロールを使用することです。ただし、一部の では AWS のサービス、(プロキシとしてロールを使用する代わりに) リソースにポリシーを直接アタッチできます。クロスアカウントアクセスのロールとリソースベースのポリシーの違いについては、「 ユーザーガイド」の「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。
- クロスサービスアクセス — 一部の は、他の の機能 AWS のサービス を使用します AWS のサービス。例えば、サービスで呼び出しを行うと、そのサービスが Amazon でアプリケーションを実行EC2したり、Amazon S3 にオブジェクトを保存したりするのが一般的です。サービスでは、呼び出し元プリンシパルの許可、サービスロール、またはサービスリンクロールを使用してこれを行う場合があります。
  - 転送アクセスセッション (FAS) – IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するための権限が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。
  - サービスロール – サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受ける IAMロールです。IAM 管理者は、 内からサービスロールを作成、変更、削除できますIAM。詳細については、「 ユーザーガイド」の「 にアクセス許可を委任するロールの作成 AWS のサービスIAM」を参照してください。

- サービスにリンクされたロール – サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。
- Amazon で実行されているアプリケーション EC2 – IAMロールを使用して、EC2インスタンスで実行され、 AWS CLI または AWS API リクエストを行うアプリケーションの一時的な認証情報を管理できます。これは、EC2インスタンス内にアクセスキーを保存するよりも望ましいです。 AWS ロールをEC2インスタンスに割り当て、そのすべてのアプリケーションで使用できるようにするには、インスタンスにアタッチされたインスタンスプロファイルを作成します。インスタンスプロファイルには ロールが含まれており、EC2インスタンスで実行されているプログラムが一時的な認証情報を取得できるようにします。詳細については、「 ユーザーガイド」の「 IAMロールを使用して Amazon EC2インスタンスで実行されているアプリケーションにアクセス許可を付与するIAM」を参照してください。

IAM ロールとIAMユーザーのどちらを使用するかについては、「 ユーザーガイド」の「 IAMロールを作成する場合 (ユーザーではなく) IAM」を参照してください。

ポリシーを使用したアクセスの管理

でアクセスを制御する AWS には、ポリシーを作成し、 AWS ID またはリソースにアタッチします。ポリシーは、アイデンティティまたはリソースに関連付けられているときにアクセス許可を定義するのオブジェクトです。プリンシパル (ユーザー、ルートユーザー、またはロールセッション) AWS がリクエストを行うと、 はこれらのポリシー AWS を評価します。ポリシーでの権限により、リクエストが許可されるか拒否されるかが決まります。ほとんどのポリシーはJSONドキュメント AWS として に保存されます。JSON ポリシードキュメントの構造と内容の詳細については、「 ユーザーガイド」のJSON「ポリシーの概要IAM」を参照してください。

管理者はポリシーを使用して AWS JSON、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。

デフォルトでは、ユーザーやロールに権限はありません。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するために、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、ユーザーはロールを引き受けることができます。

IAM ポリシーは、オペレーションの実行に使用する方法に関係なく、アクションのアクセス許可を定義します。例えば、`iam:GetRole`アクションを許可するポリシーがあるとします。そのポリシー

を持つユーザーは、 AWS Management Console、、 AWS CLIまたは AWS からロール情報を取得できますAPI。

アイデンティティベースのポリシー

ID ベースのポリシーは、IAMユーザー、ユーザーのグループ、ロールなどの ID にアタッチできるJSONアクセス許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

アイデンティティベースのポリシーは、さらにインラインポリシーまたはマネージドポリシーに分類できます。インラインポリシーは、単一のユーザー、グループ、またはロールに直接埋め込まれています。管理ポリシーは、 の複数のユーザー、グループ、ロールにアタッチできるスタンドアロンポリシーです AWS アカウント。管理ポリシーには、 AWS 管理ポリシーとカスタマー管理ポリシーが含まれます。管理ポリシーとインラインポリシーのどちらかを選択する方法については、 IAM ユーザーガイドの「管理ポリシーとインラインポリシーの選択」を参照してください。

リソースベースのポリシー

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースのポリシーの例としては、IAMロールの信頼ポリシー や Amazon S3 バケットポリシー などがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、または を含めることができます AWS のサービス。

リソースベースのポリシーは、そのサービス内にあるインラインポリシーです。リソースベースのポリシーIAMでは、 の AWS 管理ポリシーを使用できません。

アクセスコントロールリスト (ACLs)

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。ACLs はリソースベースのポリシーに似ていますが、JSONポリシードキュメント形式を使用しません。

Amazon S3、 AWS WAF、および Amazon VPCは、 をサポートするサービスの例ですACLs。の詳細についてはACLs、Amazon Simple Storage Service デベロッパーガイドの「アクセスコントロールリスト (ACL) の概要」を参照してください。

その他のポリシータイプ

AWS は、一般的ではない追加のポリシータイプをサポートします。これらのポリシータイプでは、より一般的なポリシータイプで付与された最大の権限を設定できます。

- アクセス許可の境界 – アクセス許可の境界は、アイデンティティベースのポリシーがIAMエンティティ (IAMユーザーまたはロール) に付与できるアクセス許可の上限を設定する高度な機能です。エンティティにアクセス許可の境界を設定できます。結果として得られる権限は、エンティティのアイデンティティベースポリシーとそのアクセス許可の境界の共通部分になります。`Principal` フィールドでユーザーまたはロールを指定するリソースベースのポリシーでは、アクセス許可の境界は制限されません。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。アクセス許可の境界の詳細については、「 IAMユーザーガイド」の「 IAMエンティティのアクセス許可の境界」を参照してください。
- サービスコントロールポリシー (SCPs) – SCPsは、 の組織または組織単位 (OU) に対する最大アクセス許可を指定するJSONポリシーです AWS Organizations。 AWS Organizations は、 AWS アカウント ビジネスが所有する複数の をグループ化して一元管理するためのサービスです。組織内のすべての機能を有効にすると、サービスコントロールポリシー (SCPs) をアカウントの一部またはすべてに適用できます。は、各 を含むメンバーアカウントのエンティティのアクセス許可SCP を制限します AWS アカウントのルートユーザー。Organizations と の詳細についてはSCPs、「 AWS Organizations ユーザーガイド」の「サービスコントロールポリシー」を参照してください。
- セッションポリシー - セッションポリシーは、ロールまたはフェデレーションユーザーの一時的なセッションをプログラムで作成する際にパラメータとして渡す高度なポリシーです。結果としてセッションの権限は、ユーザーまたはロールのアイデンティティベースポリシーとセッションポリシーの共通部分になります。また、リソースベースのポリシーから権限が派生する場合もあります。これらのポリシーのいずれかを明示的に拒否した場合、権限は無効になります。詳細については、「 ユーザーガイド」の「セッションポリシーIAM」を参照してください。

複数のポリシータイプ

1 つのリクエストに複数のタイプのポリシーが適用されると、結果として作成される権限を理解するのがさらに難しくなります。複数のポリシータイプが関与する場合にリクエストを許可するかどうか AWS を決定する方法については、「 ユーザーガイド」の「ポリシー評価ロジックIAM」を参照してください。

AWS Parallel Computing Service と の連携方法 IAM

IAM を使用して へのアクセスを管理する前に AWS PCS、 で使用できるIAM機能を確認してください AWS PCS。

IAM AWS Parallel Computing Service で使用できる の機能

IAM 機能	AWS PCS サポート
アイデンティティベースのポリシー	あり
リソースベースのポリシー	なし
ポリシーアクション	あり
ポリシーリソース	はい
ポリシー条件キー (サービス固有)	あり
ACLs	なし
ABAC (ポリシー内のタグ)	あり
一時的な認証情報	あり
プリンシパル権限	あり
サービスロール	いいえ
サービスリンクロール	あり

およびその他の AWS のサービスがほとんどの IAM 機能とどのように連携するか AWS PCSの概要を把握するには、IAM「 ユーザーガイド」のAWS 「 と連携する のサービスIAM」を参照してください。

のアイデンティティベースのポリシー AWS PCS

アイデンティティベースのポリシーのサポート: あり

ID ベースのポリシーは、IAMユーザー、ユーザーのグループ、ロールなどの ID にアタッチできるJSONアクセス許可ポリシードキュメントです。これらのポリシーは、ユーザーとロールが実行できるアクション、リソース、および条件をコントロールします。アイデンティティベースのポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

IAM アイデンティティベースのポリシーでは、許可または拒否された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。プリンシパルは、それが添付されているユーザーまたはロールに適用されるため、アイデンティティベースのポリシーでは指定できません。JSON ポリシーで使用できるすべての要素については、「 ユーザーガイド」の「 IAMJSONポリシー要素のリファレンスIAM」を参照してください。

のアイデンティティベースのポリシーの例 AWS PCS

アイデンティティベースのポリシーの例 AWS PCSを表示するには、「」を参照してくださいAWS Parallel Computing Service のアイデンティティベースのポリシーの例。

内のリソースベースのポリシー AWS PCS

リソースベースのポリシーのサポート: なし

リソースベースのポリシーは、リソースにアタッチするJSONポリシードキュメントです。リソースベースのポリシーの例としては、IAMロールの信頼ポリシー や Amazon S3 バケットポリシー などがあります。リソースベースのポリシーをサポートするサービスでは、サービス管理者はポリシーを使用して特定のリソースへのアクセスをコントロールできます。ポリシーがアタッチされているリソースの場合、指定されたプリンシパルがそのリソースに対して実行できるアクションと条件は、ポリシーによって定義されます。リソースベースのポリシーでは、プリンシパルを指定する必要があります。プリンシパルには、アカウント、ユーザー、ロール、フェデレーティッドユーザー、またはを含めることができます AWS のサービス。

クロスアカウントアクセスを有効にするには、リソースベースのポリシーで、アカウント全体または別のアカウントのIAMエンティティをプリンシパルとして指定できます。リソースベースのポリシーにクロスアカウントのプリンシパルを追加しても、信頼関係は半分しか確立されない点に注意してください。プリンシパルとリソースが異なる にある場合 AWS アカウント、信頼されたアカウントのIAM管理者は、プリンシパルエンティティ (ユーザーまたはロール) にリソースへのアクセス許可も付与する必要があります。IAM 管理者は、アイデンティティベースのポリシーをエンティティにアタッチすることで権限を付与します。ただし、リソースベースのポリシーで、同じアカウントのプリンシパルへのアクセス権が付与されている場合は、アイデンティティベースのポリシーをさらに付与する必要はありません。詳細については、「 ユーザーガイド」の「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。

のポリシーアクション AWS PCS

ポリシーアクションのサポート: あり

管理者はポリシーを使用して AWS JSON、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`要素は、ポリシーで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を記述します。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オペレーションがないアクセス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権限を付与するポリシーで使用されます。

アクションのリスト AWS PCSを確認するには、「サービス認証リファレンス」の AWS 「Parallel Computing Service で定義されるアクション」を参照してください。

の AWS PCSポリシーアクションは、アクションの前に次のプレフィックスを使用します。

```
pcs
```

単一のステートメントで複数のアクションを指定するには、アクションをカンマで区切ります。

```
"Action": [
      "pcs:action1",
      "pcs:action2"
         ]
```

アイデンティティベースのポリシーの例 AWS PCSを表示するには、「」を参照してくださいAWS Parallel Computing Service のアイデンティティベースのポリシーの例。

のポリシーリソース AWS PCS

ポリシーリソースのサポート: あり

管理者はポリシーを使用して AWS JSON、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。

Policy ResourceJSON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。ステートメントには、Resource または NotResource 要素を含める必要があります。ベストプラクティスとして、Amazon リソースネーム (ARN) を使用してリソースを指定します。これは、リソースレベルの許可と呼ばれる特定のリソースタイプをサポートするアクションに対して実行できます。

オペレーションのリスト化など、リソースレベルの権限をサポートしないアクションの場合は、ステートメントがすべてのリソースに適用されることを示すために、ワイルドカード (*) を使用します。

```
"Resource": "*"
```

リソースタイプとその のリスト AWS PCSを確認するにはARNs、「サービス認証リファレンス」の AWS 「Parallel Computing Service で定義されるリソース」を参照してください。各リソースARNのを指定できるアクションについては、AWS 「Parallel Computing Service で定義されるアクション」を参照してください。

アイデンティティベースのポリシーの例 AWS PCSを表示するには、「」を参照してくださいAWS Parallel Computing Service のアイデンティティベースのポリシーの例。

のポリシー条件キー AWS PCS

サービス固有のポリシー条件キーのサポート: あり

管理者はポリシーを使用して AWS JSON、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。

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。Condition 要素はオプションです。イコールや未満などの 条件演算子 を使用して条件式を作成することで、ポリシーの条件とリクエスト内の値を一致させることができます。

1 つのステートメントに複数の Condition 要素を指定する場合、または 1 つの Condition 要素に複数のキーを指定する場合、 AWS では AND 論理演算子を使用してそれらを評価します。1 つの条件キーに複数の値を指定すると、 は論理ORオペレーションを使用して条件 AWS を評価します。ステートメントの権限が付与される前にすべての条件が満たされる必要があります。

条件を指定する際にプレースホルダー変数も使用できます。例えば、リソースにIAMユーザー名でタグ付けされている場合にのみ、リソースにアクセスするアクセス許可をIAMユーザーに付与できます。詳細については、「 ユーザーガイド」のIAM「ポリシー要素: 変数とタグIAM」を参照してください。

AWS は、グローバル条件キーとサービス固有の条件キーをサポートします。すべての AWS グローバル条件キーを確認するには、「 ユーザーガイド」のAWS 「 グローバル条件コンテキストキーIAM」を参照してください。

条件キーのリスト AWS PCSを確認するには、「サービス認証リファレンス」の AWS 「並列コンピューティングサービスの条件キー」を参照してください。 条件キーを使用できるアクションとリソースについては、「AWS Parallel Computing Service で定義されるアクション」を参照してください。

アイデンティティベースのポリシーの例 AWS PCSを表示するには、「」を参照してくださいAWS Parallel Computing Service のアイデンティティベースのポリシーの例。

ACLs の AWS PCS

をサポートACLs： いいえ

アクセスコントロールリスト (ACLs) は、リソースへのアクセス許可を持つプリンシパル (アカウントメンバー、ユーザー、またはロール) を制御します。ACLs はリソースベースのポリシーに似ていますが、JSONポリシードキュメント形式を使用しません。

ABAC で AWS PCS

サポート ABAC (ポリシー内のタグ): はい

属性ベースのアクセスコントロール (ABAC) は、属性に基づいてアクセス許可を定義する認可戦略です。では AWS、これらの属性はタグ と呼ばれます。タグは、IAMエンティティ (ユーザーまたはロール) および多くの AWS リソースにアタッチできます。エンティティとリソースのタグ付けは、の最初のステップですABAC。次に、プリンシパルのタグが、アクセスしようとしているリソースのタグと一致する場合に、オペレーションを許可するABACポリシーを設計します。

ABAC は、急速に成長している環境や、ポリシー管理が煩雑になる状況に役立ちます。

タグに基づいてアクセスを管理するには、`aws:ResourceTag/key-name`、`aws:RequestTag/key-name`、または `aws:TagKeys` の条件キーを使用して、ポリシーの条件要素でタグ情報を提供します。

サービスがすべてのリソースタイプに対して 3 つの条件キーすべてをサポートする場合、そのサービスの値はありです。サービスが一部のリソースタイプに対してのみ 3 つの条件キーのすべてをサポートする場合、値は「部分的」になります。

の詳細についてはABAC、「 IAMユーザーガイド」の「 とはABAC」を参照してください。のセットアップ手順を含むチュートリアルを表示するにはABAC、「 ユーザーガイド」の「属性ベースのアクセスコントロール (ABAC) を使用するIAM」を参照してください。

での一時的な認証情報の使用 AWS PCS

一時的な認証情報のサポート: あり

一部の は、一時的な認証情報を使用してサインインすると機能 AWS のサービス しません。一時的な認証情報 AWS のサービス を使用する などの詳細については、「 ユーザーガイドAWS のサービス 」の「 と連携IAMする IAM 」を参照してください。

ユーザー名とパスワード以外の AWS Management Console 方法で にサインインする場合、一時的な認証情報を使用します。例えば、会社のシングルサインオン (SSO) リンク AWS を使用して にアクセスすると、そのプロセスによって一時的な認証情報が自動的に作成されます。また、ユーザーとしてコンソールにサインインしてからロールを切り替える場合も、一時的な認証情報が自動的に作成されます。ロールの切り替えの詳細については、「 IAMユーザーガイド」の「ロールへの切り替え (コンソール) 」を参照してください。

一時的な認証情報は、 AWS CLI または を使用して手動で作成できます AWS API。その後、これらの一時的な認証情報を使用して . AWS recommends にアクセスできます AWS。これは、長期的なアクセスキーを使用する代わりに、一時的な認証情報を動的に生成することを推奨しています。詳細については、「」の「一時的なセキュリティ認証情報IAM」を参照してください。

のクロスサービスプリンシパル許可 AWS PCS

転送アクセスセッションをサポート (FAS): はい

IAM ユーザーまたはロールを使用して でアクションを実行すると AWS、プリンシパルと見なされます。一部のサービスを使用する際に、アクションを実行することで、別のサービスの別のアクションがトリガーされることがあります。FAS は、 を呼び出すプリンシパルのアクセス許可を AWS のサービス、ダウンストリームサービス AWS のサービス へのリクエストのリクエストと組み合わせて使用します。FAS リクエストは、サービスが他の AWS のサービス またはリソースとのやり取りを完了する必要があるリクエストを受け取った場合にのみ行われます。この場合、両方のアクションを実行するための権限が必要です。FAS リクエストを行う際のポリシーの詳細については、「転送アクセスセッション」を参照してください。

AWS PCS のサービスロール

サービスロールのサポート: なし

サービスロールは、ユーザーに代わってアクションを実行するためにサービスが引き受ける IAM ロールです。IAM 管理者は、 内からサービスロールを作成、変更、削除できますIAM。詳細については、「 ユーザーガイド」の「 にアクセス許可を委任するロールの作成 AWS のサービスIAM」を参照してください。

> **Warning**
>
> サービスロールのアクセス許可を変更すると、機能が破損 AWS PCSする可能性があります。が指示する場合 AWS PCS以外は、サービスロールを編集しないでください。

のサービスにリンクされたロール AWS PCS

サービスリンクロールのサポート: あり

サービスにリンクされたロールは、 にリンクされたサービスロールの一種です AWS のサービス。サービスは、ユーザーに代わってアクションを実行するロールを引き受けることができます。サービスにリンクされたロールは に表示され AWS アカウント 、サービスによって所有されます。IAM 管理者は、サービスにリンクされたロールのアクセス許可を表示できますが、編集することはできません。

サービスにリンクされたロールの作成または管理の詳細については、「 AWS と連携する のサービスIAM」を参照してください。表の中から、[Service-linked role] (サービスにリンクされたロール) 列に Yes と記載されたサービスを見つけます。サービスリンクロールに関するドキュメントをサービスで表示するには、はい リンクを選択します。

AWS Parallel Computing Service のアイデンティティベースのポリシーの例

デフォルトでは、ユーザーとロールにはリソースを作成または変更 AWS PCSするアクセス許可はありません。また、、 AWS Command Line Interface （ AWS CLI ） AWS Management Console、または を使用してタスクを実行することはできません AWS API。必要なリソースに対してアクションを実行するアクセス許可をユーザーに付与するために、IAM管理者はIAMポリシーを作成できます。その後、管理者はIAMポリシーをロールに追加し、ユーザーはロールを引き受けることができます。

これらのポリシードキュメント例を使用してIAMアイデンティティベースのJSONポリシーを作成する方法については、「 ユーザーガイド」のIAM「ポリシーの作成IAM」を参照してください。

ARNs 各リソースタイプの の形式など AWS PCS、 で定義されるアクションとリソースタイプの詳細については、「サービス認証リファレンス」のAWS 「Parallel Computing Service 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。

トピック

- ポリシーのベストプラクティス
- コンソール AWS PCSの使用
- 自分の権限の表示をユーザーに許可する

ポリシーのベストプラクティス

ID ベースのポリシーは、ユーザーのアカウントで誰かがリソースを作成、アクセス、または削除AWS PCSできるどうかを決定します。これらのアクションを実行すると、 AWS アカウントに料金が発生する可能性があります。アイデンティティベースポリシーを作成したり編集したりする際には、以下のガイドラインと推奨事項に従ってください:

- AWS 管理ポリシーを開始し、最小特権のアクセス許可に移行する – ユーザーとワークロードにアクセス許可を付与するには、多くの一般的なユースケースにアクセス許可を付与する AWS 管理ポリシーを使用します。これらは で使用できます AWS アカウント。ユースケースに固有の AWS カスタマー管理ポリシーを定義して、アクセス許可をさらに減らすことをお勧めします。詳細については、「 ユーザーガイド」の「 AWS 管理ポリシーAWS 」または「 ジョブ機能の 管理ポリシーIAM」を参照してください。
- 最小特権のアクセス許可を適用する – IAMポリシーでアクセス許可を設定する場合は、タスクの実行に必要なアクセス許可のみを付与します。これを行うには、特定の条件下で特定のリソースに対して実行できるアクションを定義します。これは、最小特権アクセス許可とも呼ばれています。IAM を使用してアクセス許可を適用する方法の詳細については、「 ユーザーガイド」の「 のポリシーとアクセス許可IAMIAM」を参照してください。
- IAM ポリシーの条件を使用してアクセスをさらに制限する – ポリシーに条件を追加して、アクションとリソースへのアクセスを制限できます。例えば、ポリシー条件を記述して、すべてのリクエストを を使用して送信する必要があることを指定できますSSL。条件を使用して、 などの特定の を介してサービスアクションが使用される場合に AWS のサービス、サービスアクションへのアクセスを許可することもできます AWS CloudFormation。詳細については、「 ユーザーガイド」のIAMJSON「ポリシー要素: 条件IAM」を参照してください。

- IAM Access Analyzer を使用してIAMポリシーを検証し、安全で機能的なアクセス許可を確保する – IAM Access Analyzer は、ポリシーがポリシー言語 (JSON) とIAMベストプラクティスに準拠するように、新規および既存のIAMポリシーを検証します。IAM Access Analyzer には、安全で機能的なポリシーの作成に役立つ 100 を超えるポリシーチェックと実用的な推奨事項が用意されています。詳細については、「 ユーザーガイド」のIAM「Access Analyzer ポリシーの検証IAM」を参照してください。
- 多要素認証を要求する (MFA) – でIAMユーザーまたはルートユーザーを必要とするシナリオがある場合は AWS アカウント、セキュリティを強化MFAするために をオンにします。API オペレーションが呼び出されるMFAタイミングを要求するには、ポリシーにMFA条件を追加します。詳細については、「 IAMユーザーガイド」のMFA「 で保護されたAPIアクセスの設定」を参照してください。

のベストプラクティスの詳細についてはIAM、「 ユーザーガイド」の「 のセキュリティのベストプラクティスIAMIAM」を参照してください。

コンソール AWS PCSの使用

AWS Parallel Computing Service コンソールにアクセスするには、最小限のアクセス許可のセットが必要です。これらのアクセス許可により、 のリソースの詳細を AWS PCS一覧表示および表示できます AWS アカウント。最小限必要な許可よりも制限が厳しいアイデンティティベースのポリシーを作成すると、そのポリシーを持つエンティティ (ユーザーまたはロール) に対してコンソールが意図したとおりに機能しません。

AWS CLI または のみを呼び出すユーザーには、最小限のコンソールアクセス許可を付与する必要はありません AWS API。代わりに、実行しようとしているAPIオペレーションに一致するアクションのみへのアクセスを許可します。

コンソールを使用するために必要な最小限のアクセス許可の詳細については、 AWS PCS「」を参照してくださいの最小アクセス許可 AWS PCS。

自分の権限の表示をユーザーに許可する

この例では、IAMユーザーがユーザー ID にアタッチされているインラインポリシーと管理ポリシーを表示できるようにするポリシーを作成する方法を示します。このポリシーには、コンソールで、または AWS CLI または を使用してプログラムでこのアクションを実行するアクセス許可が含まれています AWS API。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

AWSAWS Parallel Computing Service の マネージドポリシー

AWS 管理ポリシーは、 によって作成および管理されるスタンドアロンポリシーです AWS。 AWS 管理ポリシーは、多くの一般的なユースケースにアクセス許可を付与するように設計されているため、ユーザー、グループ、ロールにアクセス許可の割り当てを開始できます。

AWS 管理ポリシーは、すべての AWS お客様が使用できるため、特定のユースケースに対して最小特権のアクセス許可を付与しない場合があることに注意してください。ユースケース別にカスタマーマネージドポリシーを定義して、マネージドポリシーを絞り込むことをお勧めします。

AWS 管理ポリシーで定義されているアクセス許可は変更できません。が AWS 管理ポリシーで定義されたアクセス許可 AWS を更新すると、ポリシーがアタッチされているすべてのプリンシパル ID (ユーザー、グループ、ロール) が更新されます。 AWS のサービス は、新しい AWS が起動されるか、既存のサービスで新しいAPIオペレーションが使用可能になると、 AWS 管理ポリシーを更新する可能性が最も高くなります。

詳細については、「 ユーザーガイド」の「 AWS 管理ポリシーIAM」を参照してください。

AWS マネージドポリシー： AWSPCSServiceRolePolicy

IAM エンティティ AWSPCSServiceRolePolicy に をアタッチすることはできません。このポリシーは、 がユーザーに代わってアクションを実行できるようにする AWS PCSサービスにリンクされたロールにアタッチされます。詳細については、「AWS PCS のサービスリンクロール」を参照してください。

アクセス許可の詳細

このポリシーには、以下のアクセス許可が含まれています。

- `ec2` — が Amazon AWS PCSEC2リソースを作成および管理できるようにします。
- `iam` — Amazon AWS PCSEC2フリートのサービスにリンクされたロールを作成し、そのロールを Amazon に渡すことを に許可しますEC2。
- `cloudwatch` — AWS PCSがサービスメトリクスを Amazon に発行できるようにします CloudWatch。
- `secretsmanager` – AWS PCSがクラスターリソースのシークレットを管理できるようにします AWS PCS。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PermissionsToCreatePCSNetworkInterfaces",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSPCSManaged": "false"
                }
            }
        },
        {
            "Sid": "PermissionsToCreatePCSNetworkInterfacesInSubnet",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:subnet/*",
                "arn:aws:ec2:*:*:security-group/*"
            ]
        },
        {
            "Sid": "PermissionsToManagePCSNetworkInterfaces",
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteNetworkInterface",
                "ec2:CreateNetworkInterfacePermission"
            ],
            "Resource": "arn:aws:ec2:*:*:network-interface/*",
            "Condition": {
                "Null": {
                        "aws:ResourceTag/AWSPCSManaged": "false"
                }
            }
        },
        {
            "Sid": "PermissionsToDescribePCSResources",
```

```
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeKeyPairs",
                "ec2:DescribeImages",
                "ec2:DescribeImageAttribute"
            ],
            "Resource": "*"
        },
        {
            "Sid": "PermissionsToCreatePCSLaunchTemplates",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateLaunchTemplate"
            ],
            "Resource": "arn:aws:ec2:*:*:launch-template/*",
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSPCSManaged": "false"
                }
            }
        },
        {
            "Sid": "PermissionsToManagePCSLaunchTemplates",
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteLaunchTemplate",
                "ec2:DeleteLaunchTemplateVersions",
                "ec2:CreateLaunchTemplateVersion"
            ],
            "Resource": "arn:aws:ec2:*:*:launch-template/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AWSPCSManaged": "false"
                }
```

```
            }
        },
        {
            "Sid": "PermissionsToTerminatePCSManagedInstances",
            "Effect": "Allow",
            "Action": [
                "ec2:TerminateInstances"
            ],
            "Resource": "arn:aws:ec2:*:*:instance/*",
            "Condition": {
                "Null": {
                    "aws:ResourceTag/AWSPCSManaged": "false"
                }
            }
        },
        {
            "Sid": "PermissionsToPassRoleToEC2",
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::*:role/*/AWSPCS*",
                "arn:aws:iam::*:role/AWSPCS*",
                "arn:aws:iam::*:role/aws-pcs/*",
                "arn:aws:iam::*:role/*/aws-pcs/*"
            ],
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Sid": "PermissionsToControlClusterInstanceAttributes",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances",
                "ec2:CreateFleet"
            ],
            "Resource": [
                "arn:aws:ec2:*::image/*",
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*:*:subnet/*",
```

```
                "arn:aws:ec2:*:*:network-interface/*",
                "arn:aws:ec2:*:*:security-group/*",
                "arn:aws:ec2:*:*:volume/*",
                "arn:aws:ec2:*:*:key-pair/*",
                "arn:aws:ec2:*:*:launch-template/*",
                "arn:aws:ec2:*:*:placement-group/*",
                "arn:aws:ec2:*:*:capacity-reservation/*",
                "arn:aws:resource-groups:*:*:group/*",
                "arn:aws:ec2:*:*:fleet/*"
            ]
        },
        {
            "Sid": "PermissionsToProvisionClusterInstances",
            "Effect": "Allow",
            "Action": [
                "ec2:RunInstances",
                "ec2:CreateFleet"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "Null": {
                    "aws:RequestTag/AWSPCSManaged": "false"
                }
            }
        },
        {
            "Sid": "PermissionsToTagPCSResources",
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": [
                        "RunInstances",
                        "CreateLaunchTemplate",
                        "CreateFleet",
                        "CreateNetworkInterface"
                    ]
```

```
                }
            }
        },
        {
            "Sid": "PermissionsToPublishMetrics",
            "Effect": "Allow",
            "Action": "cloudwatch:PutMetricData",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "cloudwatch:namespace": "AWS/PCS"
                }
            }
        },
        {
            "Sid": "PermissionsToManageSecret",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:DescribeSecret",
                "secretsmanager:GetSecretValue",
                "secretsmanager:PutSecretValue",
                "secretsmanager:UpdateSecretVersionStage",
                "secretsmanager:DeleteSecret"
            ],
            "Resource": "arn:aws:secretsmanager:*:*:secret:pcs!*",
            "Condition": {
                "StringEquals": {
                    "secretsmanager:ResourceTag/aws:secretsmanager:owningService":
 "pcs",
                    "aws:ResourceAccount": "${aws:PrincipalAccount}"
                }
           }
        }
    ]
}
```

AWS PCS AWS 管理ポリシーの更新

このサービスがこれらの変更の追跡を開始した以降の の AWS PCS AWS マネージドポリシーの更新に関する詳細を表示します。このページの変更に関する自動アラートを受け取るには、ドキュメント履歴ページのRSSフィードにサブスクライブします AWS PCS。

変更	説明	日付
AWS PCS は変更の追跡を開始しました	AWS PCS が AWS マネージドポリシーの変更の追跡を開始しました。	2024 年 8 月 28 日

AWS PCS のサービスリンクロール

AWS Parallel Computing Service は AWS Identity and Access Management 、 (IAM) サービスにリンクされたロール を使用します。サービスにリンクされたロールは、 に直接リンクされた一意のタイプのIAMロールです AWS PCS。サービスにリンクされたロールは によって AWS PCS事前定義されており、ユーザーに代わってサービスから他の AWS のサービスを呼び出すために必要なすべてのアクセス許可が含まれています。

サービスにリンクされたロールを使用すると、必要なアクセス許可を手動で追加する必要がないため、 の設定 AWS PCSが簡単になります。 AWS PCS は、サービスにリンクされたロールのアクセス許可を定義し、特に定義されている場合を除き、 のみが AWS PCSそのロールを引き受けることができます。定義されたアクセス許可には、信頼ポリシーとアクセス許可ポリシーが含まれ、そのアクセス許可ポリシーを他のIAMエンティティにアタッチすることはできません。

サービスリンクロールを削除するには、まずその関連リソースを削除します。これにより、リソースへのアクセス許可を誤って削除することがなくなるため、リソースが AWS PCS保護されます。

サービスにリンクされたロールをサポートする他のサービスの詳細については、「 AWS と連携するのサービスIAM」を参照し、「サービスにリンクされたロール」列で「はい」があるサービスを探します。 サービスにリンクされたロールに関するドキュメントをサービスで表示するには、[Yes] (はい) リンクを選択します。

のサービスにリンクされたロールのアクセス許可 AWS PCS

AWS PCS は、 という名前のサービスにリンクされたロールを使用しますAWSServiceRoleForPCS。 — が Amazon AWS PCSEC2リソースを管理できるようにします。

AWSServiceRoleForPCS サービスにリンクされたロールは、次のサービスを信頼してロールを引き受けます。

- `pcs.amazonaws.com`

という名前のロールアクセス許可ポリシーAWSPCSServiceRolePolicyは AWS PCS、 が特定のリソースに対するアクションを完了することを許可します。

ユーザー、グループ、ロールなどがサービスにリンクされたロールを作成、編集、削除できるようにするには、アクセス権限を設定する必要があります。詳細については、「 ユーザーガイド」の「サービスにリンクされたロールのアクセス許可IAM」を参照してください。

のサービスにリンクされたロールの作成 AWS PCS

サービスにリンクされたロールを手動で作成する必要はありません。 AWS PCS は、クラスターの作成時にサービスにリンクされたロールを作成します。

のサービスにリンクされたロールの編集 AWS PCS

AWS PCS では、 AWSServiceRoleForPCS サービスにリンクされたロールを編集することはできません。サービスリンクロールを作成した後は、多くのエンティティによってロールが参照される可能性があるため、ロール名を変更することはできません。ただし、 を使用してロールの説明を編集することはできますIAM。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編集」を参照してください。

のサービスにリンクされたロールの削除 AWS PCS

サービスリンクロールが必要な機能またはサービスが不要になった場合には、そのロールを削除することをお勧めします。そうすることで、積極的にモニタリングまたは保守されていない未使用のエンティティを排除できます。ただし、手動で削除する前に、サービスリンクロールのリソースをクリーンアップする必要があります。

> **Note**
>
> リソースを削除しようとしたときに AWS PCSサービスがロールを使用している場合、削除が失敗する可能性があります。失敗した場合は、数分待ってから操作を再試行してください。

が使用するリソースを削除するには AWS PCS AWSServiceRoleForPCS

AWSServiceRoleForPCS サービスにリンクされたロールを削除するには、すべてのクラスターを削除する必要があります。詳細については、「クラスターの削除」を参照してください。

を使用してサービスにリンクされたロールを手動で削除するには IAM

IAM コンソール、、または AWS API を使用して AWS CLI、 AWSServiceRoleForPCS サービスにリンクされたロールを削除します。詳細については、「 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IAM」を参照してください。

AWS PCS のサービスにリンクされたロールをサポートするリージョン

AWS PCS は、サービスが利用可能なすべてのリージョンでサービスにリンクされたロールの使用をサポートします。詳細については、「AWS リージョンとエンドポイント」を参照してください。

の Amazon EC2 スポットロール AWS PCS

スポットを購入オプションとして使用するコンピューティングノードグループを作成する AWS PCS 場合は、 にAWSServiceRoleForEC2Spotサービスにリンクされたロールも必要です AWS アカウント。次の AWS CLI コマンドを使用してロールを作成できます。詳細については、「 ユーザーガイド」の「サービスにリンクされたロールを作成する」および AWS 「 のサービスにアクセス許可を委任するロールを作成するAWS Identity and Access Management 」を参照してください。

```
aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

> **Note**
>
> に AWS アカウント 既に AWSServiceRoleForEC2SpotIAMロールがある場合、次のエラーが表示されます。
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
> operation: Service role name AWSServiceRoleForEC2Spot has been taken in this
> account, please try a different suffix.
> ```

の最小アクセス許可 AWS PCS

このセクションでは、IAMID (ユーザー、グループ、またはロール) がサービスを使用するために必要な最小限のIAMアクセス許可について説明します。

目次

- タグを使用するために必要な最小限のアクセス許可
- ログのサポートに必要な最小限のアクセス許可
- サービス管理者の最小アクセス許可

API アクションを使用するための最小限のアクセス許可

API アクション	最小アクセス許可	コンソールに対する追加のアクセス許可
CreateCluster	`ec2:CreateNetworkInterface, ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:GetSecurityGroupsForVpc, iam:CreateServiceLinkedRole, secretsmanager:CreateSecret, secretsmanager:TagResource, pcs:CreateCluster`	
ListClusters	`pcs:ListClusters`	
GetCluster	`pcs:GetCluster`	`ec2:DescribeSubnets`
DeleteCluster	`pcs:DeleteCluster`	
CreateComputeNodeGroup	`ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:DescribeLaunchTemplates,`	`iam:ListInstanceProfiles, ec2:DescribeImages, pcs:GetCluster`

API アクション	最小アクセス許可	コンソールに対する追加のアクセス許可
	`ec2:DescribeLaunchTemplateVersions, ec2:DescribeInstanceTypes, ec2:RunInstances, ec2:CreateFleet, ec2:CreateTags, iam:PassRole, iam:GetInstanceProfile, pcs:CreateComputeNodeGroup`	
ListComputerNodeGroups	`pcs:ListComputeNodeGroups`	`pcs:GetCluster`
GetComputeNodeGroup	`pcs:GetComputeNodeGroup`	`ec2:DescribeSubnets`
UpdateComputeNodeGroup	`ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:DescribeLaunchTemplates, ec2:DescribeLaunchTemplateVersions, ec2:DescribeInstanceTypes, ec2:RunInstances, ec2:CreateFleet, ec2:CreateTags, iam:PassRole, iam:GetInstanceProfile, pcs:UpdateComputeNodeGroup`	`pcs:GetComputeNodeGroup, iam:ListInstanceProfiles, ec2:DescribeImages, pcs:GetCluster`

API アクション	最小アクセス許可	コンソールに対する追加のアクセス許可
DeleteComputeNodeGroup	`pcs:DeleteComputeNodeGroup`	
CreateQueue	`pcs:CreateQueue`	`pcs:ListComputeNodeGroups, pcs:GetCluster`
ListQueues	`pcs:ListQueues`	`pcs:GetCluster`
GetQueue	`pcs:GetQueue`	
UpdateQueue	`pcs:UpdateQueue`	`pcs:ListComputeNodeGroups, pcs:GetQueue`
DeleteQueue	`pcs:DeleteQueue`	

タグを使用するために必要な最小限のアクセス許可

のリソースでタグを使用するには、次のアクセス許可が必要です AWS PCS。

```
pcs:ListTagsForResource
pcs:TagResource
pcs:UntagResource
```

ログのサポートに必要な最小限のアクセス許可

AWS PCS はログデータを Amazon CloudWatch Logs (CloudWatch Logs) に送信します。ID に CloudWatch ログを使用するための最小限のアクセス許可があることを確認する必要があります。詳細については、「Amazon CloudWatch Logs ユーザーガイド」の「ログリソースへのアクセス許可の管理の概要」を参照してください。 CloudWatch

サービスが CloudWatch ログにログを送信するために必要なアクセス許可については、「Amazon Logs ユーザーガイド」の AWS 「 のサービスからのログ記録の有効化」を参照してください。 CloudWatch

サービス管理者の最小アクセス許可

次のIAMポリシーは、IAMID (ユーザー、グループ、またはロール) がサービスを設定および管理 AWS PCSするために必要な最小限のアクセス許可を指定します。

> **Note**
>
> サービスを設定および管理しないユーザーは、これらのアクセス許可を必要としません。 ジョブのみを実行するユーザーは、セキュアシェル (SSH) を使用してクラスターに接続します。 AWS Identity and Access Management （ IAM) は の認証または認可を処理しません SSH。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateNetworkInterface",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeSubnets",
                "ec2:DescribeVpcs",
                "ec2:GetSecurityGroupsForVpc",
                "firehose:*",
                "iam:GetInstanceProfile",
                "iam:ListInstanceProfiles",
                "iam:PassRole",
                "kms:*",
                "logs:*",
                "pcs:*",
                "s3:*"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

ポリシーから次のアクセス許可を除外し、代わりに で対応する マネージドポリシーを使用できますIAM。

- `"firehose:*"`

  ```
  AmazonKinesisFirehoseFullAccess
  ```

- `"kms:*"`

  ```
  AWSKeyManagementServicePowerUser
  ```

- `"logs:*"`

  ```
  CloudWatchLogsFullAccess
  ```

- `"s3:*"`

  ```
  AmazonS3FullAccess
  ```

IAM AWS Parallel Computing Service の インスタンスプロファイル

EC2 インスタンスで実行されるアプリケーションは、リクエスト AWS APIに AWS 認証情報を含める必要があります。EC2 インスタンスの一時的な認証情報を管理するには、 IAMロールを使用することをお勧めします。これを行うインスタンスプロファイルを定義し、インスタンスにアタッチできます。詳細については、IAM「Amazon Elastic Compute Cloud ユーザーガイドEC2」の「Amazon のロール」を参照してください。

> **Note**
>
> を使用して Amazon の IAMロール AWS Management Console を作成するとEC2、コンソールによってインスタンスプロファイルが自動的に作成され、IAMロールと同じ名前が付けられます。 AWS CLI、 AWS API アクション、または AWS SDK を使用してIAMロールを作成する場合は、別のアクションとしてインスタンスプロファイルを作成します。詳細について

は、「Amazon Elastic Compute Cloud ユーザーガイド」の「インスタンスプロファイル」を参照してください。

コンピューティングノードグループを作成するときは、インスタンスプロファイルARNの を指定する必要があります。コンピューティングノードグループの一部またはすべてに異なるインスタンスプロファイルを選択できます。

インスタンスプロファイルの要件

インスタンスプロファイル名

IAM インスタンスプロファイルは、 で始まるかAWSPCS、パス/aws-pcs/に含めるARN必要があります。

Example

- `arn:aws:iam::*:instance-profile/AWSPCS-example-role-1` および
- `arn:aws:iam::*:instance-profile/aws-pcs/example-role-2`.

アクセス許可

少なくとも、 のインスタンスプロファイル AWS PCSには次のポリシーを含める必要があります。これにより、コンピューティングノードは、サービスが AWS PCS動作可能になったときにサービスに通知できます。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "pcs:RegisterComputeNodeGroupInstance"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

追加ポリシー

インスタンスプロファイルに管理ポリシーを追加することを検討してください。例:

- AmazonS3ReadOnlyAccess は、すべての S3 バケットへの読み取り専用アクセスを提供します。
- は、Amazon mazonSSMManagedInstanceCoreマネジメントコンソールから直接のリモートアクセスなど、AWSSystems Manager サービスのコア機能を有効にします。
- CloudWatchAgentServerPolicy には、 サーバー AmazonCloudWatchAgent で を使用するために必要なアクセス許可が含まれています。

特定のユースケースをサポートする独自のIAMポリシーを含めることもできます。

インスタンスプロファイルの作成

インスタンスプロファイルは、Amazon EC2コンソールから直接作成できます。詳細については、「 ユーザーガイド」の「インスタンスプロファイルの使用AWS Identity and Access Management 」を参照してください。

AWS Parallel Computing Service のアイデンティティとアクセスのトラブルシューティング

次の情報は、 および の使用 AWS PCS時に発生する可能性がある一般的な問題の診断と修正に役立ちますIAM。

トピック

- でアクションを実行する権限がない AWS PCS
- iam を実行する権限がありません。PassRole
- 自分の 以外のユーザーに自分のリソース AWS アカウント へのアクセス AWS PCSを許可したい

でアクションを実行する権限がない AWS PCS

「I am not authorized to perform an action in Amazon Bedrock」というエラーが表示された場合、そのアクションを実行できるようにポリシーを更新する必要があります。

次の例のエラーは、`mateojackson`IAMユーザーが コンソールを使用して架空の*`my-example-widget`*リソースの詳細を表示しようとしているが、架空の`pcs:`*`GetWidget`*アクセス許可がない場合に発生します。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 pcs:GetWidget on resource: my-example-widget
```

この場合、pcs:*GetWidget* アクションを使用して *my-example-widget*リソースへのアクセスを許可するように、mateojackson ユーザーのポリシーを更新する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

iam を実行する権限がありません。PassRole

iam:PassRole アクションを実行する権限がないというエラーが表示された場合は、ポリシーを更新して にロールを渡すことができるようにする必要があります AWS PCS。

一部の AWS のサービス では、新しいサービスロールまたはサービスにリンクされたロールを作成する代わりに、そのサービスに既存のロールを渡すことができます。そのためには、サービスにロールを渡す権限が必要です。

次の例のエラーは、 というIAMユーザーがコンソールを使用して marymajor でアクションを実行しようする場合に発生します AWS PCS。ただし、このアクションをサービスが実行するには、サービスロールから付与された権限が必要です。メアリーには、ロールをサービスに渡す許可がありません。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

この場合、Mary のポリシーを更新してメアリーに iam:PassRole アクションの実行を許可する必要があります。

サポートが必要な場合は、 AWS 管理者にお問い合わせください。サインイン認証情報を提供した担当者が管理者です。

自分の 以外のユーザーに自分のリソース AWS アカウント へのアクセス AWS PCSを許可したい

他のアカウントのユーザーや組織外の人が、リソースにアクセスするために使用できるロールを作成できます。ロールの引き受けを委託するユーザーを指定できます。リソースベースのポリシーまたはアクセスコントロールリスト (ACLs) をサポートするサービスでは、これらのポリシーを使用して、ユーザーにリソースへのアクセスを許可できます。

詳細については、以下を参照してください。

- がこれらの機能をサポートしているかどうか AWS PCSを確認するには、「」を参照してくださいAWS Parallel Computing Service と の連携方法 IAM。
- 所有している のリソースへのアクセスを提供する方法については、 AWS アカウント 「 ユーザーガイド」の「所有 AWS アカウント している別の のIAMユーザーへのアクセスを提供するIAM」を参照してください。
- リソースへのアクセスをサードパーティー に提供する方法については AWS アカウント、「 ユーザーガイド」の「サードパーティー AWS アカウント が所有する へのアクセスを提供するIAM」を参照してください。
- ID フェデレーションを通じてアクセスを提供する方法については、 IAMユーザーガイドの「外部認証されたユーザーへのアクセスの提供 (ID フェデレーション)」を参照してください。
- クロスアカウントアクセスでのロールとリソースベースのポリシーの使用の違いについては、「 ユーザーガイド」の「 でのクロスアカウントリソースアクセスIAMIAM」を参照してください。

AWS Parallel Computing Service のコンプライアンス検証

AWS のサービス が特定のコンプライアンスプログラムの範囲内にあるかどうかを確認するには、コンプライアンスプログラムAWS のサービス による対象範囲内のコンプライアンスプログラムを参照し、関心のあるコンプライアンスプログラムを選択します。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。

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 AWS Artifact。詳細については、「 でのレポートのダウンロード AWS Artifact」の」を参照してください。

を使用する際のお客様のコンプライアンス責任 AWS のサービス は、お客様のデータの機密性、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。 は、コンプライアンスに役立つ以下のリソース AWS を提供しています。

- セキュリティと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に重点を置いたベースライン環境 AWS を にデプロイする手順について説明します。
- アマゾン ウェブ サービス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 – このホワイトペーパーでは、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について説明します。

> **Note**
>
> すべての AWS のサービス がHIPAA対象となるわけではありません。詳細については、HIPAA「対象サービスリファレンス」を参照してください。

- AWS コンプライアンスリソース – このワークブックとガイドのコレクションは、お客様の業界や地域に適用される場合があります。
- AWS カスタマーコンプライアンスガイド — コンプライアンスの観点から責任共有モデルを理解します。このガイドでは、ガイダンスを保護し AWS のサービス 、複数のフレームワーク (米国国立標準技術研究所 (NIST) 、Payment Card Industry Security Standards Council ()、PCI国際標準化機構 (ISO) など) のセキュリティコントロールにマッピングするためのベストプラクティスをまとめています。
- 「 デベロッパーガイド」の「 ルールによるリソースの評価」 – この AWS Config サービスは、リソース設定が社内プラクティス、業界ガイドライン、および規制にどの程度準拠しているかを評価します。 AWS Config
- AWS Security Hub – これにより AWS のサービス 、 内のセキュリティ状態を包括的に確認できます AWS。Security Hub では、セキュリティコントロールを使用して AWS リソースを評価し、セキュリティ業界標準とベストプラクティスに対するコンプライアンスをチェックします。サポートされているサービスとコントロールのリストについては、Security Hub のコントロールリファレンスを参照してください。
- Amazon GuardDuty – これにより AWS アカウント、疑わしいアクティビティや悪意のあるアクティビティがないか環境を監視することで、、、ワークロード、コンテナ、データに対する潜在的な脅威 AWS のサービス を検出します。 GuardDuty は、特定のコンプライアンスフレームワークで義務付けられている侵入検知要件を満たすことでDSS、 PCI などのさまざまなコンプライアンス要件に対応するのに役立ちます。
- AWS Audit Manager – これにより AWS のサービス 、 AWS 使用状況を継続的に監査し、リスクの管理方法と規制や業界標準への準拠を簡素化できます。

AWS Parallel Computing Service の耐障害性

AWS グローバルインフラストラクチャは AWS リージョン およびアベイラビリティーゾーンを中心に構築されています。物理的に分離および分離された複数のアベイラビリティーゾーン AWS リージョン を提供し、低レイテンシー、高スループット、高冗長ネットワークで接続されます。アベイラビリティーゾーンでは、ゾーン間で中断することなく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することができます。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が高く、フォールトトレラントで、スケーラブルです。

AWS リージョン および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。

AWS Parallel Computing Service 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 AWS パラレルコンピューティングサービスは、 AWS グローバルネットワークセキュリティで保護されています。 AWS セキュリティサービスと がインフラストラクチャ AWS を保護する方法については、AWS 「 クラウドセキュリティ」を参照してください。インフラストラクチャセキュリティのベストプラクティスを使用して AWS 環境を設計するには、「 Security Pillar AWS Well-Architected Framework」の「Infrastructure Protection」を参照してください。

が AWS 公開したAPI呼び出しを使用して、ネットワーク経由で にアクセスします AWS PCS。クライアントは以下をサポートする必要があります:

- Transport Layer Security (TLS) 。1TLS.2 が必要で、1.3 TLS をお勧めします。
- (Ephemeral Diffie-HellmanPFS) や DHE (Elliptic Curve Ephemeral Diffie-Hellman) などの完全前方秘匿性 ECDHE () を備えた暗号スイート。これらのモードは、Java 7 以降など、ほとんどの最新システムでサポートされています。

さらに、リクエストは、 IAMプリンシパルに関連付けられたアクセスキー ID と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。

がクラスター AWS PCSを作成すると、サービスは、アカウント内のコンピューティングノードとは別に、サービス所有のアカウントで Slurm コントローラーを起動します。コントローラーとコンピューティングノード間の通信をブリッジするために、 AWS PCS はクロスアカウントの Elastic Network Interface (ENI) を に作成しますVPC。Slurm コントローラーは ENIを使用して、さまざまなのコンピューティングノードを管理および通信し AWS アカウント、リソースのセキュリティと分離を維持しながら、効率HPC的で AI/ML オペレーションを促進します。

AWS Parallel Computing Service での脆弱性の分析と管理

設定と IT コントロールは、 AWS とユーザーの間で共有される責任です。詳細については、AWS 「 責任共有モデル」を参照してください。 は、コントローラーインスタンスのオペレーティングシ

ステムへのパッチ適用、ファイアウォール設定、インフラストラクチャディザスタリカバリなど、サービスアカウントの基盤となる AWS インフラストラクチャの基本的なセキュリティタスク AWS を処理します。これらの手順は適切なサードパーティーによって確認され、認証されています。詳細については、「セキュリティ、アイデンティティ、コンプライアンスのベストプラクティス」を参照してください。

お客様は、 の基盤となるインフラストラクチャのセキュリティに責任を負います AWS アカウント。

- 更新やセキュリティパッチなど、コードを維持します。
- ノードグループインスタンスのオペレーティングシステムにパッチを適用して更新します。
- スケジューラを更新して、サポートされているバージョン内に保持します。
- ユーザークライアントと接続先のノード間の通信を認証および暗号化します。

サービス間での不分別な代理処理の防止

混乱した代理問題は、アクションを実行するためのアクセス許可を持たないエンティティが、より特権のあるエンティティにアクションの実行を強制できてしまう場合に生じる、セキュリティ上の問題です。では AWS、サービス間のなりすましにより、混乱した代理問題が発生する可能性があります。サービス間でのなりすましは、1 つのサービス (呼び出し元サービス) が、別のサービス (呼び出し対象サービス) を呼び出すときに発生する可能性があります。呼び出し元サービスは、本来ならアクセスすることが許可されるべきではない方法でその許可を使用して、別のお客様のリソースに対する処理を実行するように操作される場合があります。これを防ぐため、 AWS では、アカウント内のリソースへのアクセス権が付与されたサービスプリンシパルですべてのサービスのデータを保護するために役立つツールを提供しています。

リソースポリシーで `aws:SourceArn`および `aws:SourceAccount` グローバル条件コンテキストキーを使用して、 AWS Parallel Computing Service (AWS PCS) が別のサービスに付与するアクセス許可をリソースに制限することをお勧めします。クロスサービスアクセスにリソースを 1 つだけ関連付けたい場合は、`aws:SourceArn` を使用します。そのアカウント内のリソースをクロスサービスの使用に関連付けることを許可する場合は、`aws:SourceAccount` を使用します。

混乱した代理問題から保護する最も効果的な方法は、リソースARNがいっぱいになった `aws:SourceArn` グローバル条件コンテキストキーを使用することです。リソースARNの完全版がわからない場合、または複数のリソースを指定する場合は、 の不明な部分にワイルドカード文字 (*) を含むaws:SourceArnグローバルコンテキスト条件キーを使用しますARN。例えば、`arn:aws:servicename:*:123456789012:*` と指定します。

aws:SourceArn 値に Amazon S3 バケット などのアカウント ID が含まれていない場合はARN、両方のグローバル条件コンテキストキーを使用してアクセス許可を制限する必要があります。

の値はクラスター aws:SourceArnである必要がありますARN。

次の例は、 で aws:SourceArnおよび aws:SourceAccount グローバル条件コンテキストキーAWS PCSを使用して、混乱した代理問題を回避する方法を示しています。

```
{
"Version": "2012-10-17",
  "Statement": {
"Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "pcs.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:pcs:us-east-1:123456789012:cluster/*"
        ]
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

IAM コンピューティングノードグループの一部としてプロビジョニングされた Amazon EC2インスタンスの ロール

AWS PCS は、クラスター内の設定された各コンピューティングノードグループの Amazon EC2 容量を自動的にオーケストレーションします。コンピューティングノードグループを作成する場合、ユーザーは iamInstanceProfileArnフィールドを通じてIAMインスタンスプロファイルを指定する必要があります。インスタンスプロファイルは、プロビジョニングされたEC2インスタンスに関連付けられたアクセス許可を指定します。 AWS PCS は、 をロール名のプレフィックスAWSPCSとして、またはロールパス/aws-pcs/の一部として持つすべてのロールを受け入れます。アクセスiam:PassRole許可は、コンピューティングノードグループを作成または更新する IAM ID (ユーザーまたはロール) に必要です。ユーザーが CreateComputeNodeGroupま

たは UpdateComputeNodeGroupAPIアクションを呼び出すと、 AWS PCS はユーザーがiam:PassRoleアクションを実行できるかどうかを確認します。

次のポリシー例では、名前が で始まるIAMロールのみを渡すアクセス許可を付与しますAWSPCS。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/AWSPCS*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

AWS Parallel Computing Service のセキュリティのベストプラクティス

このセクションでは、 AWS Parallel Computing Service () に固有のセキュリティのベストプラクティスについて説明しますAWS PCS。のセキュリティのベストプラクティスの詳細についてはAWS、「セキュリティ、アイデンティティ、コンプライアンスのベストプラクティス」を参照してください。

AMI関連のセキュリティ

- 本番環境のワークロードAMIsにはサンプルを使用しないでください AWS PCS。サンプルAMIsはサポートされておらず、テストのみを目的としています。
- 脆弱性を軽減するために、インスタンスの AWS PCSオペレーティングシステムとソフトウェアを定期的に更新します。
- AWS Systems Manager を使用してパッチ適用を自動化し、セキュリティポリシーへの準拠を維持します。

- 公式 AWS PCS AWS ソースからダウンロードした認証済みの公式パッケージのみを使用してください。
- コンピューティングノードのパッケージを定期的に更新 AWS PCSして、セキュリティパッチと改善点を受け取ります。脆弱性を最小限に抑えるために、このプロセスを自動化することを検討してください。

Slurm Workload Manager のセキュリティ

- アクセスコントロールとネットワーク制限を実装して、Slurm コントロールノードとコンピューティングノードを保護します。信頼できるユーザーとシステムのみがジョブを送信し、Slurm 管理コマンドにアクセスできるようにします。
- Slurm 認証などの Slurm の組み込みセキュリティ機能を使用して、ジョブの送信と通信が認証されるようにします。
- Slurm バージョンを更新して、スムーズなオペレーションとクラスターサポートを維持します。

> **Important**
>
> サポート終了 (EOSL) に達したバージョンの Slurm を使用するクラスターは、直ちに停止します。ユーザーガイドページの上部にあるリンクを使用してドキュメントRSSフィードを AWS PCSサブスクライブし、Slurm バージョンが に近づいたときに通知を受信しますEOSL。

モニタリングとログ記録

- Amazon CloudWatch Logs と AWS CloudTrail を使用して、クラスターと のアクションをモニタリングおよび記録します AWS アカウント。トラブルシューティングと監査にデータを使用します。

ネットワークセキュリティ

- クラスターを AWS PCS別の にデプロイVPCして、HPC環境を他のネットワークトラフィックから分離します。

- セキュリティグループとネットワークアクセスコントロールリスト (ACLs) を使用して、インスタンスとサブネットへの AWS PCSインバウンドトラフィックとアウトバウンドトラフィックを制御します。
- AWS PrivateLink または VPCエンドポイントを使用して、クラスターとネットワーク内の他の AWS サービス間の AWS ネットワークトラフィックを維持します。

のログ記録とモニタリング AWS PCS

モニタリングは、 およびその他の AWSリソースの信頼性、可用性、パフォーマンス AWS PCSを維持する上で重要な部分です。AWS には、 を監視し AWS PCS、問題が発生したときに報告し、必要に応じて自動アクションを実行するための以下のモニタリングツールが用意されています。

- Amazon CloudWatch は、 AWS リソースと、 で実行しているアプリケーションを AWS リアルタイムでモニタリングします。メトリクスの収集と追跡、カスタマイズしたダッシュボードの作成、および指定したメトリクスが指定したしきい値に達したときに通知またはアクションを実行するアラームの設定を行うことができます。例えば、 で Amazon EC2インスタンスのCPU使用状況やその他のメトリクス CloudWatch を追跡し、必要に応じて新しいインスタンスを自動的に起動できます。詳細については、「Amazon ユーザーガイド CloudWatch 」を参照してください。
- Amazon CloudWatch Logs を使用すると、Amazon EC2インスタンスやその他のソースからログファイルをモニタリング、保存 CloudTrail、およびアクセスできます。 CloudWatch Logs はログファイル内の情報をモニタリングし、特定のしきい値に達したときに通知できます。高い耐久性を備えた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 CloudTrail は、 AWS アカウントによって、またはアカウントに代わって行われたAPI呼び出しおよび関連イベントをキャプチャし、指定した Amazon S3 バケットにログファイルを配信します。を呼び出したユーザーとアカウント AWS、呼び出し元の IP アドレス、呼び出しが発生した日時を特定できます。詳細については、AWS CloudTrail ユーザーガイドをご参照ください。

AWS PCS スケジューラログ

クラススケジューラから Amazon CloudWatch Logs、Amazon Simple Storage Service (Amazon S3)、および Amazon Data Firehose に詳細なログデータを送信するように を設定できます AWS PCS。これは、モニタリングとトラブルシューティングに役立ちます。スケジューラログは、 AWS PCS コンソールを使用して設定することも、 AWS CLI または を使用してプログラムで設定 AWS PCSすることもできますSDK。

目次

  - クラスターを AWS PCS配信ソースとして有効にする
  - クラスター配信ソースを配信先に接続する
- スケジューラログストリームのパスと名前
- スケジューラログレコードの例 AWS PCS

前提条件

AWS PCS クラスターの管理に使用されるIAMプリンシパルは、 を許可する必要がありますpcs:AllowVendedLogDeliveryForResource。有効にするAWSIAMポリシーの例を次に示します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "PcsAllowVendedLogsDelivery",
            "Effect": "Allow",
            "Action": ["pcs:AllowVendedLogDeliveryForResource"],
            "Resource": [
                "arn:aws:pcs:::cluster/*"
            ]
        }
    ]
}
```

AWS PCS コンソールを使用したスケジューラログの設定

コンソールでスAWSPCSケジューラログを設定するには、次の手順に従います。

1. AWS PCS コンソール を開きます。
2. クラスターを選択し、ログ記録を有効にする AWS PCSクラスターの詳細ページに移動します。
3. [Logs] (ログ) を選択します。
4. ログ配信 – スケジューラログ – オプション
   a. 最大 3 つのログ配信先を追加します。選択には、 CloudWatch ログ、Amazon S3、または Firehose が含まれます。
   b. ログ配信の更新 を選択します。

このページを再確認することで、ログ配信を再構成、追加、または削除できます。

を使用したスケジューラログの設定 AWS CLI

これを実現するには、少なくとも 1 つの配信先、1 つの配信元 (PCSクラスター)、1 つの配信が必要です。これは、送信元を宛先に接続する関係です。

配信先を作成する

AWS PCS クラスターからスケジューラログを受信するには、少なくとも 1 つの配信先が必要です。このトピックの詳細については、 ユーザーガイドの CloudWatch API PutDeliveryDestinationセクションを参照してください。

を使用して配信先を作成するには AWS CLI

- 次のコマンドを使用して送信先を作成します。コマンドを実行する前に、次の置き換えを行います。
  - 置換 *region-code* 送信先を作成する AWS リージョン を使用します。これは通常、クラスターがデプロイされている AWS PCSリージョンと同じリージョンになります。
  - 置換 *pcs-logs-destination* 任意の名前で。アカウント内のすべての配信先に対して一意である必要があります。
  - 置換 *resource-arn* を CloudWatch 、ログの既存のロググループの 、S3 バケット、または Firehose の配信ストリームARNで使用します。その例を以下に示します。
    - CloudWatch ロググループ

      ```
      arn:aws:logs:region-code:account-id:log-group:/log-group-name:*
      ```

    - S3 バケット

      ```
      arn:aws:s3:::bucket-name
      ```

    - Firehose 配信ストリーム

      ```
      arn:aws:firehose:region-code:account-id:deliverystream/stream-name
      ```

```
aws logs put-delivery-destination --region region-code \
    --name pcs-logs-destination \
    --delivery-destination-configuration destinationResourceArn=resource-arn
```

新しい配信先の ARNを書き留めておきます。これは、配信を設定するために必要になるためです。

クラスターを AWS PCS配信ソースとして有効にする

からスケジューラログを収集するにはAWSPCS、クラスターを配信ソースとして設定します。詳細については、「Amazon Logs リファレンスPutDeliverySource」の「」を参照してください。 CloudWatch API

を使用してクラスターを配信ソースとして設定するには AWS CLI

- 次のコマンドを使用して、クラスターからのログ配信を有効にします。コマンドを実行する前に、次の置き換えを行います。
  - 置換 *region-code* クラスター AWS リージョン がデプロイされている を使用します。
  - 置換 *cluster-logs-source-name* このソースの名前を持つ 。これは、 内のすべての配信ソースで一意である必要があります AWS アカウント。 AWS PCS クラスターの名前または ID を組み込むことを検討してください。
  - 置換 *cluster-arn* クラスターARNの AWS PCSで

```
aws logs put-delivery-source \
    --region region-code \
    --name cluster-logs-source-name \
    --resource-arn cluster-arn \
    --log-type PCS_SCHEDULER_LOGS
```

クラスター配信ソースを配信先に接続する

スケジューラログデータをクラスターから送信先にフローするには、それらを接続するための配信を設定する必要があります。詳細については、「Amazon Logs リファレンスCreateDelivery」の「」を参照してください。 CloudWatch API

を使用して配信を作成するには AWS CLI

- 次のコマンドを使用して配信を作成します。コマンドを実行する前に、次の置き換えを行います。
  - 置換 *region-code* 送信元と送信先が存在する AWS リージョン を持つ 。
  - 置換 *cluster-logs-source-name* は、上記の配信ソースの名前を使用します。
  - 置換 *destination-arn* ログを配信する配信先ARNからの 。

```
aws logs create-delivery \
    --region region-code \
    --delivery-source-name cluster-logs-source \
    --delivery-destination-arn destination-arn
```

スケジューラログストリームのパスと名前

スAWSPCSケジューラログのパスと名前は、送信先タイプによって異なります。

- CloudWatch ログ
  - CloudWatch Logs ストリームはこの命名規則に従います。

    ```
    AWSLogs/PCS/${cluster_id}/${log_name}_${scheduler_major_version}.log
    ```

    Example

    ```
    AWSLogs/PCS/abcdef0123/slurmctld_24.05.log
    ```
- S3 バケット
  - S3 バケットの出力パスは次の命名規則に従います。

    ```
    AWSLogs/${account-id}/PCS/${region}/${cluster_id}/${log_name}/
    ${scheduler_major_version}/yyyy/MM/dd/HH/
    ```

    Example

    ```
    AWSLogs/111111111111/PCS/us-east-2/abcdef0123/slurmctld/24.05/2024/09/01/00.
    ```
  - S3 オブジェクト名は次の規則に従います。

    ```
    PCS_${log_name}_${scheduler_major_version}_#{expr date 'event_timestamp', format:
     "yyyy-MM-dd-HH"}_${cluster_id}_${hash}.log
    ```

    Example

    ```
    PCS_slurmctld_24.05_2024-09-01-00_abcdef0123_0123abcdef.log
    ```

スケジューラログレコードの例 AWS PCS

AWS PCS スケジューラログは構造化されています。これには、Slurm コントローラープロセスから出力されるログメッセージに加えて、クラスター識別子、スケジューラタイプ、メジャーバージョン、パッチバージョンなどのフィールドが含まれます。以下はその例です。

```
{
    "resource_id": "s3431v9rx2",
    "resource_type": "PCS_CLUSTER",
    "event_timestamp": 1721230979,
    "log_level": "info",
    "log_name": "slurmctld",
    "scheduler_type": "slurm",
    "scheduler_major_version": "23.11",
    "scheduler_patch_version": "8",
    "node_type": "controller_primary",
    "message": "[2024-07-17T15:42:58.614+00:00] Running as primary controller\n"
}
```

Amazon による AWS 並列コンピューティングサービスのモニタリング CloudWatch

Amazon CloudWatch は、クラスターからメトリクスを一定の間隔で収集することで、 AWS Parallel Computing Service (AWS PCS) クラスターのヘルスとパフォーマンスをモニタリングします。これらのメトリクスは保持されるため、履歴データにアクセスし、時間の経過とともにクラスターのパフォーマンスに関するインサイトを得ることができます。

CloudWatch では、スケーリング要件を満たすために によって AWS PCS起動されたEC2インスタンスをモニタリングすることもできます。実行中のインスタンスのログを検査できますが、通常、CloudWatch メトリクスとログデータはインスタンスが終了すると削除されます。ただし、インスタンスの終了後もメトリクスとログを保持するようにEC2起動テンプレートを使用してインスタンスで CloudWatch エージェントを設定できるため、長期的なモニタリングと分析が可能になります。

を使用したモニタリング AWS PCSの詳細については、このセクションのトピックを参照してください CloudWatch。

トピック

- を使用したメトリクスのモニタリング AWS PCS CloudWatch

- Amazon を使用したインスタンスのモニタリング AWS PCS CloudWatch

を使用したメトリクスのモニタリング AWS PCS CloudWatch

Amazon を使用してクラスターの状態をモニタリング AWS PCSできます。Amazon は CloudWatch クラスターからデータを収集し、ほぼリアルタイムのメトリクスに変換します。これらの統計は 15 か月間保持されるため、履歴情報にアクセスしてクラスターの動作をより的確に把握できます。クラスターメトリクスは 1 分 CloudWatch 間隔で に送信されます。の詳細については CloudWatch、「Amazon ユーザーガイド」の「Amazon CloudWatchとは」を参照してください。CloudWatch

AWS PCS は、次のメトリクスを の AWS/PCS 名前空間に発行します CloudWatch。1 つのディメンション があります`ClusterId`。

名前	説明	単位
ActualCapacity	IdleCapacity + UtilizedCapacity	カウント
CapacityUtilization	UtilizedCapacity / ActualCapacity	カウント
DesiredCapacity	ActualCapacity + PendingCapacity	カウント
IdleCapacity	実行中のがジョブに割り当てられていないインスタンスの数	カウント
UtilizedCapacity	実行中でジョブに割り当てられたインスタンスの数	Count (カウント)

Amazon を使用したインスタンスのモニタリング AWS PCS CloudWatch

AWS PCS は、PCSコンピューティングノードグループで定義されているスケーリング要件を満たすために、必要に応じて Amazon EC2インスタンスを起動します。これらのインスタンスは、Amazon を使用して実行中にモニタリングできます CloudWatch。実行中のインスタンスのログは、ログインしてインタラクティブなコマンドラインツールを使用して検査できます。ただし、デフォルトでは、

CloudWatch メトリクスデータはインスタンスが終了すると一定期間のみ保持され、通常、インスタンスログはインスタンスをバックアップするEBSボリュームとともに削除されます。によって起動されたインスタンスのメトリクスまたはログ記録データが終了したPCS後に保持するには、EC2起動テンプレートを使用してインスタンスで CloudWatch エージェントを設定できます。このトピックでは、実行中のインスタンスのモニタリングの概要と、永続インスタンスのメトリクスとログを設定する方法の例を示します。

実行中のインスタンスのモニタリング

AWS PCS インスタンスの検索

によって起動されたインスタンスをモニタリングするにはPCS、クラスターまたはコンピューティングノードグループに関連付けられている実行中のインスタンスを見つけます。次に、特定のインスタンスのEC2コンソールで、ステータスとアラーム、モニタリングセクションを検査します。これらのインスタンスのログインアクセスが設定されている場合は、インスタンスに接続して、インスタンス上のさまざまなログファイルを検査できます。によって管理されるインスタンスを特定する方法の詳細については、PCS「」を参照してくださいでのコンピューティングノードグループインスタンスの検索 AWS PCS。

詳細メトリクスの有効化

デフォルトでは、インスタンスメトリクスは 5 分間隔で収集されます。1 分間隔でメトリクスを収集するには、コンピューティングノードグループの起動テンプレートで詳細 CloudWatch モニタリングを有効にします。詳細については、「詳細 CloudWatchモニタリングを有効にする」を参照してください。

永続インスタンスのメトリクスとログの設定

インスタンスに Amazon CloudWatch エージェントをインストールして設定することで、インスタンスのメトリクスとログを保持できます。これは主に 3 つのステップで構成されています。

1. CloudWatch エージェント設定を作成します。
2. PCS インスタンスで取得できる設定を保存します。
3. CloudWatch エージェントソフトウェアをインストールし、設定を取得し、設定を使用して CloudWatch エージェントを起動するEC2起動テンプレートを作成します。

詳細については、「Amazon ユーザーガイド」の CloudWatch 「 エージェントを使用してメトリクス、ログ、トレースを収集する」、および「」を参照してくださいでの Amazon EC2起動テンプレートの使用 AWS PCS。 CloudWatch

CloudWatch エージェント設定を作成する

インスタンスに CloudWatch エージェントをデプロイする前に、収集するメトリクス、ログ、トレースを指定するJSON設定ファイルを生成する必要があります。設定ファイルは、ウィザードを使用して作成することも、テキストエディタを使用して手動で作成することもできます。設定ファイルは、このデモンストレーション用に手動で作成されます。

AWS CLI がインストールされているコンピュータで、config.json という名前 CloudWatch の設定ファイルを以下の内容で作成します。以下を使用して、 ファイルのコピーURLをダウンロードすることもできます。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/cloudwatch/assets/config.json
```

メモ

- サンプルファイルのログパスは Amazon Linux 2 用です。インスタンスが別の基本オペレーティングシステムを使用する場合は、必要に応じてパスを変更します。
- 他のログをキャプチャするには、 の下にエントリを追加しますcollect_list。
- の値はテンプレート変数{brackets}です。サポートされている変数の完全なリストについては、「Amazon CloudWatch ユーザーガイド」の CloudWatch 「エージェント設定ファイルを手動で作成または編集する」を参照してください。
- これらの情報タイプを収集metricsしない場合は、 logsを省略するか、 を省略するかを選択できます。

```
{
    "agent": {
        "metrics_collection_interval": 60
    },
    "logs": {
        "logs_collected": {
            "files": {
                "collect_list": [
                    {
                        "file_path": "/var/log/cloud-init.log",
                        "log_group_class": "STANDARD",
                        "log_group_name": "/PCSLogs/instances",
                        "log_stream_name": "{instance_id}.cloud-init.log",
                        "retention_in_days": 30
                    },
```

```
                    {
                        "file_path": "/var/log/cloud-init-output.log",
                        "log_group_class": "STANDARD",
                        "log_stream_name": "{instance_id}.cloud-init-output.log",
                        "log_group_name": "/PCSLogs/instances",
                        "retention_in_days": 30
                    },
                    {
                        "file_path": "/var/log/amazon/pcs/bootstrap.log",
                        "log_group_class": "STANDARD",
                        "log_stream_name": "{instance_id}.bootstrap.log",
                        "log_group_name": "/PCSLogs/instances",
                        "retention_in_days": 30
                    },
                    {
                        "file_path": "/var/log/slurmd.log",
                        "log_group_class": "STANDARD",
                        "log_stream_name": "{instance_id}.slurmd.log",
                        "log_group_name": "/PCSLogs/instances",
                        "retention_in_days": 30
                    },
                    {
                        "file_path": "/var/log/messages",
                        "log_group_class": "STANDARD",
                        "log_stream_name": "{instance_id}.messages",
                        "log_group_name": "/PCSLogs/instances",
                        "retention_in_days": 30
                    },
                    {
                        "file_path": "/var/log/secure",
                        "log_group_class": "STANDARD",
                        "log_stream_name": "{instance_id}.secure",
                        "log_group_name": "/PCSLogs/instances",
                        "retention_in_days": 30
                    }
                ]
            }
        }
    },
    "metrics": {
        "aggregation_dimensions": [
            [
                "InstanceId"
            ]
```

```
        ],
        "append_dimensions": {
            "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
            "ImageId": "${aws:ImageId}",
            "InstanceId": "${aws:InstanceId}",
            "InstanceType": "${aws:InstanceType}"
        },
        "metrics_collected": {
            "cpu": {
                "measurement": [
                    "cpu_usage_idle",
                    "cpu_usage_iowait",
                    "cpu_usage_user",
                    "cpu_usage_system"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ],
                "totalcpu": false
            },
            "disk": {
                "measurement": [
                    "used_percent",
                    "inodes_free"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ]
            },
            "diskio": {
                "measurement": [
                    "io_time"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ]
            },
            "mem": {
                "measurement": [
                    "mem_used_percent"
                ],
```

```
                "metrics_collection_interval": 60
            },
            "swap": {
                "measurement": [
                    "swap_used_percent"
                ],
                "metrics_collection_interval": 60
            }
        }
    }
}
```

このファイルは、インスタンスのブートストラップ、認証とログイン、およびその他のトラブルシューティングドメインのエラーの診断に役立つ複数のファイルをモニタリングするように CloudWatch エージェントに指示します。具体的には次のとおりです。

- `/var/log/cloud-init.log` - インスタンス設定の初期段階からの出力
- `/var/log/cloud-init-output.log` – インスタンス設定中に実行されるコマンドからの出力
- `/var/log/amazon/pcs/bootstrap.log` – インスタンス設定中に実行される PCS固有のオペレーションからの出力
- `/var/log/slurmd.log` – Slurm ワークロードマネージャーのデーモン slurmd からの出力
- `/var/log/messages` – カーネル、システムサービス、アプリケーションからのシステムメッセージ
- `/var/log/secure` – SSH、sudo、その他のセキュリティイベントなどの認証試行に関連するログ

ログファイルは、 という名前の CloudWatch ロググループに送信されます`/PCSLogs/instances`。ログストリームは、インスタンス ID とログファイルのベース名の組み合わせです。ロググループの保持期間は 30 日です。

さらに、 ファイルは、いくつかの一般的なメトリクスを収集し、インスタンス ID 別に集約するように CloudWatch エージェントに指示します。

設定を保存する

CloudWatch エージェント設定ファイルは、PCSコンピューティングノードインスタンスがアクセスできる場所に保存する必要があります。これを行うには、2 つの一般的な方法があります。コンピューティングノードグループインスタンスがインスタンスプロファイルを介してアクセスできる

Amazon S3 バケットにアップロードできます。または、Amazon Systems Manager パラメータストアにSSMパラメータとして保存することもできます。

S3 バケットへのアップロード

ファイルを S3 に保存するには、次のAWSCLIコマンドを使用します。コマンドを実行する前に、次の置換を行います。

- 置換 *DOC-EXAMPLE-BUCKET* 独自の S3 バケット名を使用する

まず、 (既存のバケットがある場合はオプション) 設定ファイルを保持するバケットを作成します。

```
aws s3 mb s3://DOC-EXAMPLE-BUCKET
```

次に、ファイルをバケットにアップロードします。

```
aws s3 cp ./config.json s3://DOC-EXAMPLE-BUCKET/
```

SSM パラメータとして保存する

ファイルをSSMパラメータとして保存するには、次のコマンドを使用します。コマンドを実行する前に、次の置換を行います。

- 置換 *region-code* を、 で作業している AWS AWSリージョンで使用しますPCS。
- (オプション) を置き換える *AmazonCloudWatch-PCS* パラメータに独自の名前を付けます。名前のプレフィックスを から変更する場合AmazonCloudWatch-、ノードグループインスタンスプロファイルの SSMパラメータに読み取りアクセスを具体的に追加する必要があることに注意してください。

```
aws ssm put-parameter \
   --region region-code \
   --name "AmazonCloudWatch-PCS" \
   --type String \
   --value file://config.json
```

EC2 起動テンプレートを記述する

起動テンプレートの具体的な詳細は、設定ファイルが S3 に保存されているか に保存されているかによって異なりますSSM。

S3 に保存されている設定を使用する

このスクリプトは CloudWatch エージェントをインストールし、S3 バケットから設定ファイルをインポートして、 CloudWatch エージェントを起動します。このスクリプトの次の値を独自の詳細に置き換えます。

- *DOC-EXAMPLE-BUCKET* – アカウントが読み取ることができる S3 バケットの名前
- */config.json* – 設定が保存されている S3 バケットルートへの相対パス

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-cloudwatch-agent

runcmd:
- aws s3 cp s3://DOC-EXAMPLE-BUCKET/config.json /etc/s3-cw-config.json
- /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m
 ec2 -s -c file://etc/s3-cw-config.json

--==MYBOUNDARY==--
```

ノードグループのIAMインスタンスプロファイルには、バケットへのアクセス権が必要です。上記のユーザーデータスクリプトのバケットのIAMポリシーの例を次に示します。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET",
                "arn:aws:s3:::DOC-EXAMPLE-BUCKET/*"
            ]
```

```
        }
    ]
}
```

また、インスタンスは S3 および CloudWatch エンドポイントへのアウトバウンドトラフィックを許可する必要があることに注意してください。これは、クラスターアーキテクチャに応じて、セキュリティグループまたはVPCエンドポイントを使用して実現できます。

に保存されている設定を使用する SSM

このスクリプトは、 CloudWatch エージェントをインストールし、 SSMパラメータから設定ファイルをインポートして、 CloudWatch エージェントを起動します。このスクリプトの次の値を独自の詳細に置き換えます。

- (オプション) を置き換える *AmazonCloudWatch-PCS* パラメータに独自の名前を付けます。

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-cloudwatch-agent

runcmd:
- /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m
 ec2 -s -c ssm:AmazonCloudWatch-PCS

--==MYBOUNDARY==--
```

ノードグループのIAMインスタンスポリシーには、 がアCloudWatchAgentServerPolicyタッチされている必要があります。

パラメータ名が で始まらないAmazonCloudWatch-場合は、ノードグループインスタンスプロファイルの SSMパラメータに読み取りアクセスを具体的に追加する必要があります。プレフィックスとしてこれを示すIAMポリシーの例を次に示します。*DOC-EXAMPLE-PREFIX*.

```
{
  "Version" : "2012-10-17",
```

```
  "Statement" : [
    {
      "Sid" : "CustomCwSsmMParamReadOnly",
      "Effect" : "Allow",
      "Action" : [
        "ssm:GetParameter"
      ],
      "Resource" : "arn:aws:ssm:*:*:parameter/DOC-EXAMPLE-PREFIX*"
    }
  ]
}
```

また、インスタンスは SSMおよび CloudWatch エンドポイントへのアウトバウンドトラフィックを許可する必要があります。これは、クラスターアーキテクチャに応じて、セキュリティグループまたはVPCエンドポイントを使用して実現できます。

を使用した AWS Parallel Computing Service API呼び出しのログ記録 AWS CloudTrail

AWS PCS は、 と統合されています。これは AWS CloudTrail、ユーザー、ロール、または のサービスによって実行されたアクションを記録する AWS サービスです AWS PCS。 は、 のすべてのAPI呼び出しをイベント AWS PCSとして CloudTrail キャプチャします。キャプチャされた呼び出しには、 AWS PCSコンソールからの呼び出しと、 APIオペレーションへのコード呼び出しが含まれます AWS PCS。証跡を作成する場合は、 の CloudTrail イベントなど、Amazon S3 バケットへのイベントの継続的な配信を有効にすることができます AWS PCS。証跡を設定しない場合でも、 CloudTrail コンソールのイベント履歴 で最新のイベントを表示できます。によって収集された情報を使用して CloudTrail、 に対して行われたリクエスト AWS PCS、リクエスト元の IP アドレス、リクエスト者、リクエスト日時などの詳細を確認できます。

の詳細については CloudTrail、「 AWS CloudTrail ユーザーガイド」を参照してください。

AWS PCS の情報 CloudTrail

CloudTrail アカウントを作成する AWS アカウント と、 で が有効になります。でアクティビティが発生すると AWS PCS、そのアクティビティは CloudTrail イベント履歴 の他の AWS サービスイベントとともにイベントに記録されます。で最近のイベントを表示、検索、ダウンロードできます AWS アカウント。詳細については、「イベント履歴で CloudTrail イベントを表示する」を参照してください。

のイベントなど AWS アカウント、 のイベントの継続的な記録については AWS PCS、証跡を作成します。証跡により CloudTrail 、 はログファイルを Amazon S3 バケットに配信できます。デフォルトでは、コンソールで証跡を作成するときに、証跡がすべての AWS リージョンに適用されます。証跡は、 AWS パーティション内のすべてのリージョンからのイベントをログに記録し、指定した Amazon S3 バケットにログファイルを配信します。さらに、 CloudTrail ログで収集されたイベントデータをより詳細に分析し、それに基づいて行動するように、他の AWS サービスを設定できます。詳細については、次を参照してください:

- 追跡を作成するための概要
- CloudTrail がサポートするサービスと統合
- の Amazon SNS通知の設定 CloudTrail
- 複数のリージョンからの CloudTrail ログファイルの受信と複数のアカウントからの CloudTrail ログファイルの受信

すべての AWS PCSアクションは によってログに記録 CloudTrail され、AWS Parallel Computing Service APIリファレンス に記載されています。例えば、`CreateComputeNodeGroup`、、および `DeleteCluster`アクションを呼び出すと`UpdateQueue`、 CloudTrail ログファイルにエントリが生成されます。

各イベントまたはログエントリには、誰がリクエストを生成したかという情報が含まれます。アイデンティティ情報は、以下を判別するのに役立ちます:

- リクエストがルート認証情報または AWS Identity and Access Management （ IAM) ユーザー認証情報のどちらを使用して行われたか。
- リクエストがロールまたはフェデレーションユーザーのテンポラリなセキュリティ認証情報を使用して行われたかどうか。
- リクエストが別の AWS サービスによって行われたかどうか。

詳細については、CloudTrail userIdentity「」要素を参照してください。

からの CloudTrail ログファイルエントリについて AWS PCS

証跡は、指定した S3 バケットにイベントをログファイルとして配信できるようにする設定です。CloudTrail ログファイルには 1 つ以上のログエントリが含まれます。イベントは任意のソースからの単一のリクエストを表し、リクエストされたアクション、アクションの日時、リクエストパラメータなどに関する情報が含まれます。 CloudTrail ログファイルはパブリックAPIコールの順序付けられたスタックトレースではないため、特定の順序では表示されません。

次の例は、 CreateQueueアクションの CloudTrail ログエントリを示しています。


```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:admin",
        "arn": "arn:aws:sts::012345678910:assumed-role/Admin/admin",
        "accountId": "012345678910",
        "accessKeyId": "ASIAY36PTPIEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAY36PTPIEEXAMPLE",
                "arn": "arn:aws:iam::012345678910:role/Admin",
                "accountId": "012345678910",
                "userName": "Admin"
            },
            "attributes": {
                "creationDate": "2024-07-16T17:05:51Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-07-16T17:13:09Z",
    "eventSource": "pcs.amazonaws.com",
    "eventName": "CreateQueue",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/126.0.0.0 Safari/537.36",
    "requestParameters": {
        "clientToken": "c13b7baf-2894-42e8-acec-example",
        "clusterIdentifier": "abcdef0123",
        "computeNodeGroupConfigurations": [
            {
                "computeNodeGroupId": "abcdef0123"
            }
        ],
        "queueName": "all"
    },
    "responseElements": {
        "queue": {
```

```
            "arn": "arn:aws:pcs:us-east-1:609783872011:cluster/abcdef0123/queue/
abcdef0123",
            "clusterId": "abcdef0123",
            "computeNodeGroupConfigurations": [
                {
                    "computeNodeGroupId": "abcdef0123"
                }
            ],
            "createdAt": "2024-07-16T17:13:09.276069393Z",
            "id": "abcdef0123",
            "modifiedAt": "2024-07-16T17:13:09.276069393Z",
            "name": "all",
            "status": "CREATING"
        }
    },
    "requestID": "a9df46d7-3f6d-43a0-9e3f-example",
    "eventID": "7ab18f88-0040-47f5-8388-example",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "012345678910",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "pcs.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

のエンドポイントとサービスクォータ AWS PCS

以下のセクションでは、 AWS Parallel Computing Service () のエンドポイントとサービスクォータについて説明しますAWS PCS。以前は制限と呼ばれていたサービスクォータは、 のサービスリソースまたはオペレーションの最大数です AWS アカウント。

AWS アカウント には、 AWS サービスごとにデフォルトのクォータがあります。特に明記されていない限り、クォータは地域固有です。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引き上げることはできません。

詳細については、「AWS 全般のリファレンス」の「AWS Service Quotas」を参照してください。

目次

サービスエンドポイント

リージョン名	リージョン	エンドポイント	プロトコル
米国東部 (バージニア北部)	us-east-1	pcs.us-east-1.amazonaws.com	HTTPS
米国東部 (オハイオ)	us-east-2	pcs.us-east-2.amazonaws.com	HTTPS
米国西部 (オレゴン)	us-west-2	pcs.us-west-2.amazonaws.com	HTTPS
アジアパシフィック (シンガポール)	ap-southeast-1	pcs.ap-southeast-1.amazonaws.com	HTTPS
アジアパシフィック (シドニー)	ap-southeast-2	pcs.ap-southeast-2.amazonaws.com	HTTPS

リージョン名	リージョン	エンドポイント	プロトコル
アジアパシフィック (東京)	ap-northeast-1	pcs.ap-northeast-1.amazonaws.com	HTTPS
欧州 (フランクフルト)	eu-central-1	pcs.eu-central-1.amazonaws.com	HTTPS
欧州 (アイルランド)	eu-west-1	pcs.eu-west-1.amazonaws.com	HTTPS
欧州 (ストックホルム)	eu-north-1	pcs.eu-north-1.amazonaws.com	HTTPS

Service Quotas

名前	デフォルト	調整可能	説明
クラスター	5	あり	あたりのクラスターの最大数 AWS リージョン。

> **Note**
>
> デフォルト値は、 によって設定された初期クォータです AWS。これらのデフォルト値は、実際に適用されたクォータ値および適用可能な最大 Service Quotas とは異なります。詳細については、「Service Quotas ユーザーガイド」の「Service Quotas の用語」を参照してください。

これらのサービスクォータは、「」のAWS 「Parallel Computing Service (PCS) 」に記載されていますAWS Management Console。調整可能として表示される値のクォータ引き上げをリクエストするには、「Service Quotas ユーザーガイド」の「クォータ引き上げのリクエスト」を参照してください。 Service Quotas

> **Important**
>
> の現在の AWS リージョン 設定を必ず確認してください AWS Management Console。

内部クォータ

次のクォータは内部クォータであり、調整できません。

名前	デフォルト	調整可能	説明
クラスターの同時作成	1	なし	あたりの `Creating`状態のクラスターの最大数 AWS リージョン。

他の AWS サービスの関連クォータ

AWS PCS は他の AWS サービスを使用します。これらのサービスのサービスクォータは、 の使用に影響します AWS PCS。

に影響する Amazon EC2サービスクォータ AWS PCS

- スポットインスタンスリクエスト
- オンデマンドインスタンスの実行
- 起動テンプレート
- 起動テンプレートのバージョン
- Amazon EC2APIリクエスト

詳細については、「Amazon Elastic Compute Cloud EC2 ユーザーガイド」の「Amazon Service Quotas」を参照してください。

サンプルのリリースノート AWS PCS AMIs

AWS PCS サンプルAMIsには、セキュリティパッチの夜間リリース頻度があります。これらの増分セキュリティパッチは公式リリースノートには含まれていません。

> **Important**
>
> サンプルAMIsはデモンストレーション用であり、本番ワークロードにはお勧めしません。

目次

AWS PCS Slurm 23.11 (Amazon Linux 2) AMIのサンプル x86_64

このドキュメントでは、 AWS PCSサンプル x86_64 AMI (Amazon Linux 2) の最新の変更、追加、既知の問題、修正について説明します。

- 作成日: 2024 年 7 月 15 日
- リリース日: 2024 年 8 月 22 日
- 最終更新日: 2024 年 8 月 22 日

AMI 名前

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-23.11`

サポートされているEC2インスタンス

- 64 ビット x86 プロセッサを搭載したすべてのインスタンス。互換性のあるインスタンスを検索するには、Amazon EC2コンソール に移動します。インスタンスタイプ を選択し、 を検索しますArchitectures=x86_64。

AMI コンテンツ

- サポートされている AWS サービス： AWS PCS

- オペレーティングシステム: Amazon Linux 2
- コンピューティングアーキテクチャ: x86_64
- Linux カーネル: 5.10.220-209.867.amzn2.x86_64
- EBS ボリュームタイプ: gp2
- AWS PCS Slurm 23.11 インストーラー: 23.11.9-1
- AWS PCS ソフトウェアインストーラ: 1.0.0-1
- EFA インストーラ: 1.33.0
- GDRCopy: 2.4
- NVIDIA ドライバー: 535.154.05
- NVIDIA CUDA: 12.2.2_535.104.05

注意

- なし

リリース日: 2024-08-22

更新

- なし。初回リリース。

追加

- なし。初回リリース。

削除済み

- なし。初回リリース。

AWS PCS Slurm 23.11 (Amazon Linux 2) AMI用の サンプル Arm64

このドキュメントでは、 AWS PCSSample Arm64 AMI (Amazon Linux 2) の最新の変更、追加、既知の問題、修正について説明します。

- 作成日: 2024 年 7 月 15 日

- リリース日: 2024 年 8 月 22 日
- 最終更新日: 2024 年 8 月 22 日

AMI 名前

- `aws-pcs-sample_ami-amzn2-arm64-slurm-23.11`

サポートされているEC2インスタンス

- 64 ビット Arm プロセッサを使用するすべてのインスタンス。互換性のあるインスタンスを検索するには、Amazon EC2コンソール に移動します。インスタンスタイプ を選択し、 を検索しますArchitectures=arm64。

AMI コンテンツ

- サポートされている AWS サービス： AWS PCS
- オペレーティングシステム: Amazon Linux 2
- コンピューティングアーキテクチャ: arm64
- Linux カーネル: 5.10.220-209.867.amzn2.aarch64
- EBS ボリュームタイプ: gp2
- AWS PCS Slurm 23.11 インストーラー: 23.11.9-1
- AWS PCS ソフトウェアインストーラ: 1.0.0-1
- EFA インストーラ: 1.33.0
- GDRCopy: 2.4
- NVIDIA ドライバー: 535.154.05
- NVIDIA CUDA: 12.2.2_535.104.05

注意

- なし

リリース日: 2024-08-22

更新

- なし。初回リリース。

追加

- なし。初回リリース。

削除済み

- なし。初回リリース。

AWS PCS ユーザーガイドのドキュメント履歴

次の表に、 のドキュメントリリースを示します AWS PCS。

日付	変更	ドキュメントの更新	API バージョンの更新
2024 年 8 月 28 日	マネージドポリシーページを追加	詳細については、「AWSAWS Parallel Computing Service の マネージドポリシー」を参照してください。	該当なし
2024 年 8 月 28 日	AWS PCS リリース	ユーザーガイドの AWS PCS初回リリース。	AWS SDK: 2024-08-28

AWS 用語集

最新の AWS 用語については、「 AWS の用語集 リファレンス」のAWS 「 用語集」を参照してください。

翻訳は機械翻訳により提供されています。提供された翻訳内容と英語版の間で齟齬、不一致または矛盾がある場合、英語版が優先します。