aws

用户指南

AWS PC

AWS PC: 用户指南

Table of Contents

什么是 AWS 并行计算服务？

AWS 并行计算服务 (AWS PCS) 是一项托管服务，可以更轻松地运行和扩展高性能计算 (HPC) 工作负载，以及 AWS 使用 Slurm 构建科学和工程模型。使用 AWS PCS 构建集成了同类最佳 AWS 计算、存储、联网和可视化的计算集群。运行仿真或构建科学和工程模型。使用内置的管理和可观察性功能简化和简化集群操作。让您的用户能够在熟悉的环境中运行应用程序和作业，从而使他们能够专注于研究和创新。

主题

AWS PCS 中的概念

AWS PCS 中的一个集群有 1 个或多个队列，这些队列与至少 1 个计算节点组相关联。作业提交到队列并在计算节点组定义的 EC2 实例上运行。您可以使用这些基础来实现复杂的 HPC 架构。

集群

集群是一种用于管理资源和运行工作负载的资源。集群是一种 AWS PCS 资源，它定义了计算、网络、存储、身份和作业调度器配置的组合。您可以通过指定要使用的任务计划程序（当前为 Slurm）、想要的调度器配置、要管理集群的服务控制器以及要在哪个 VPC 中启动集群资源来创建集群。调度器接受和调度作业，还启动处理这些任务的计算节点（EC2 实例）。

计算节点组

计算节点组是 AWS PCS 用来运行作业或提供对集群的交互式访问的计算节点的集合。在定义计算节点组时，您需要指定常见特征， EC2 例如 Amazon 实例类型、最小和最大实例数、目标 VPC 子网、Amazon 系统映像 (AMI)、购买选项和自定义启动配置。 AWS PCS 使用这些设置来高效启动、管理和终止计算节点组中的计算节点。

队列

当你想在特定集群上运行作业时，可以将其提交到特定的队列（有时也称为分区）。在 AWS PCS 安排作业在计算节点组上运行之前，该作业将一直保留在队列中。您可以将一个或多个计算节点组与每个队列相关联。需要一个队列才能使用作业调度器提供的各种调度策略在底层计算节点组资源上调度和执行作业。用户不会直接向计算节点或计算节点组提交作业。

系统管理员

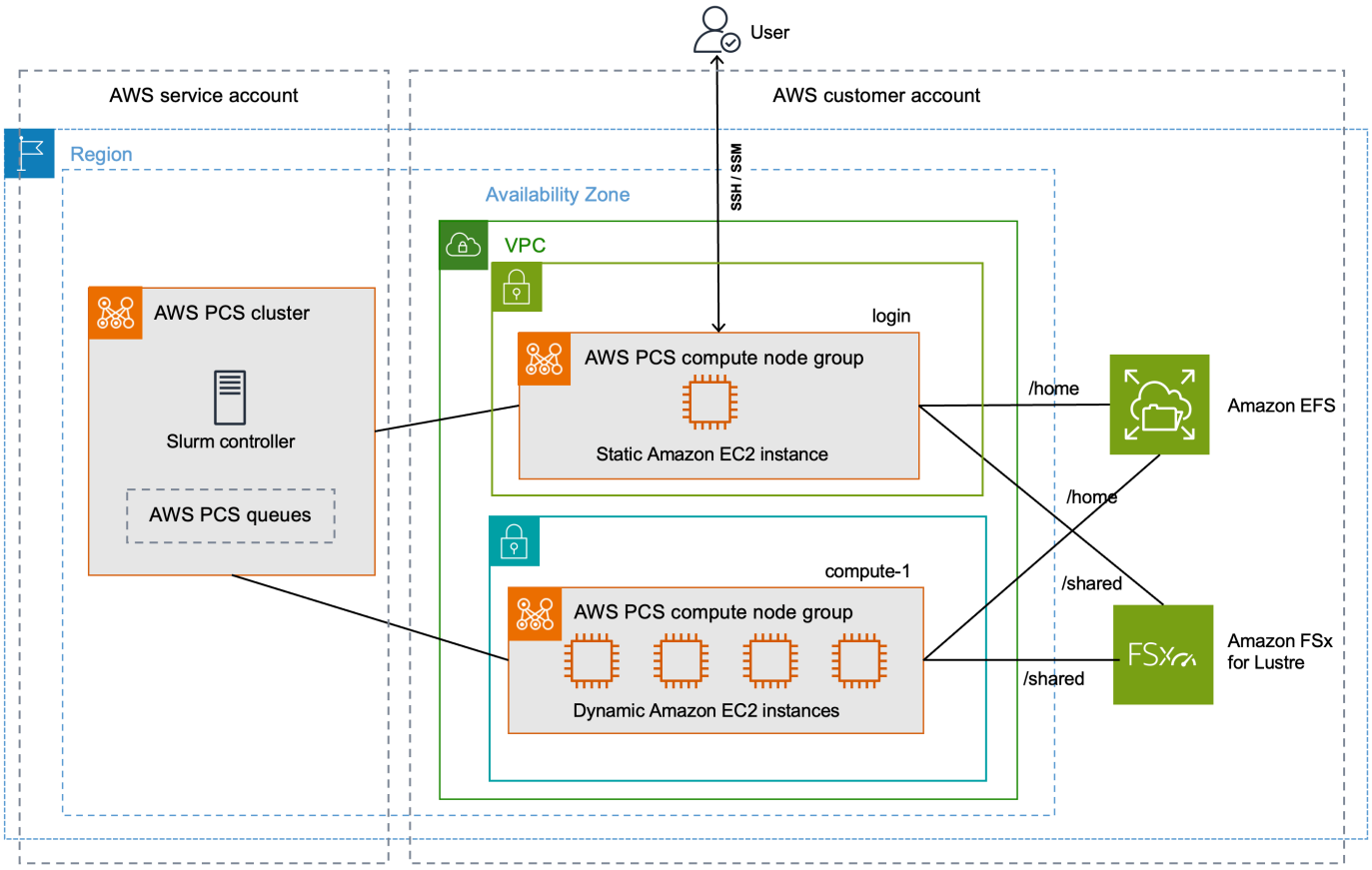
系统管理员部署、维护和操作集群。他们可以通过 AWS 管理控制台、 AWS PCS API 和 AWS SDK 访问 AWS PCS。他们可以通过 SSH 访问特定的集群 AWS Systems Manager，或者在那里他们可以运行管理任务、运行作业、管理数据和执行其他基于 shell 的活动。有关更多信息，请参阅 AWS Systems Manager 文档。

最终用户

最终用户没有 day-to-day责任部署或操作集群。他们使用终端接口（例如 SSH）来访问群集资源、运行作业、管理数据和执行其他基于 shell 的活动。

开始使用并 AWS 行计算服务

这是一个创建简单集群的教程，你可以用它来试用 AWS PCS。下图显示了集群的设计。



集群设计教程包含以下关键组件：

- 满足 PCS 联网要求的 V AWS PC 和子网。
- Amazon EFS 文件系统，将用作共享的主目录。
- Amazon FSx for Lustre 文件系统，它提供共享的高性能目录。
- 一个 AWS PCS 集群，它提供 Slurm 控制器。
- 2 AWS 个 PCS 计算节点组。
  - `login`节点组，它提供对系统的基于 shell 的交互式访问。
  - `compute-1`节点组提供弹性伸缩实例来运行作业。
- 1 个向`compute-1`节点组中的 EC2 实例发送任务的队列。

集群需要其他 AWS 资源，例如安全组、IAM 角色和 EC2 启动模板，这些资源未显示在图表中。

> **Note**
>
> 我们建议您在 Bash shell 中完成本主题中的命令行步骤。如果您没有使用 Bash shell，则某些脚本命令（例如行延续字符以及变量的设置和使用方式）需要调整 shell。此外，您的 Shell 的引用和转义规则可能有所不同。有关更多信息，请参阅《版本 2 AWS Command Line Interface 用户指南》 AWS CLI中的 “引号和带字符串的文字”。

主题

开始使用 AWS PCS 的先决条件

请参阅以下主题，为你 AWS 账户 和本地的 AWS PCS 开发环境做好准备。

主题

注册 AWS 并创建管理员用户

完成以下任务以设置 AWS 并行计算服务 (AWS PCS)。

主题

注册获取 AWS 账户

如果您没有 AWS 账户，请完成以下步骤来创建一个。

报名参加 AWS 账户

1. 打开https://portal.aws.amazon.com/billing/注册。
2. 按照屏幕上的说明操作。

   在注册时，将接到电话或收到短信，要求使用电话键盘输入一个验证码。

   当您注册时 AWS 账户，就会创建AWS 账户根用户一个。根用户有权访问该账户中的所有 AWS 服务 和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行需要根用户访问权限的任务。

AWS 注册过程完成后会向您发送一封确认电子邮件。您可以随时前往 https://aws.amazon.com/并选择“我的账户”，查看您当前的账户活动并管理您的账户。

创建具有管理访问权限的用户

注册后，请保护您的安全 AWS 账户 AWS 账户根用户 AWS IAM Identity Center，启用并创建管理用户，这样您就不会使用 root 用户执行日常任务。

保护你的 AWS 账户根用户

1. 选择 Root 用户并输入您的 AWS 账户 电子邮件地址，以账户所有者的身份登录。AWS 管理控制台在下一页上，输入您的密码。

   要获取使用根用户登录方面的帮助，请参阅《AWS 登录 用户指南》中的 Signing in as the root user。
2. 为您的根用户启用多重身份验证（MFA）。

有关说明，请参阅 I A M 用户指南中的为 AWS 账户 根用户启用虚拟 MFA 设备（控制台）。

创建具有管理访问权限的用户

1. 启用 IAM Identity Center。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Enabling AWS IAM Identity Center。

2. 在 IAM Identity Center 中，为用户授予管理访问权限。

   有关使用 IAM Identity Center 目录 作为身份源的教程，请参阅《用户指南》 IAM Identity Center 目录中的使用默认设置配置AWS IAM Identity Center 用户访问权限。

以具有管理访问权限的用户身份登录

- 要使用您的 IAM Identity Center 用户身份登录，请使用您在创建 IAM Identity Center 用户时发送到您的电子邮件地址的登录网址。

  有关使用 IAM Identity Center 用户登录的帮助，请参阅AWS 登录 用户指南中的登录 AWS 访问门户。

将访问权限分配给其他用户

1. 在 IAM Identity Center 中，创建一个权限集，该权限集遵循应用最低权限的最佳做法。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Create a permission set。

2. 将用户分配到一个组，然后为该组分配单点登录访问权限。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Add groups。

安装 AWS CLI 适用于 AWS PCS 的

您必须使用最新版本的 AWS CLI。有关信息，请参阅《版本 2 AWS Command Line Interface 用户指南》 AWS CLI中的 “安装或更新至最新版本”。

您必须配置 AWS CLI。有关更多信息，请参阅版本 2 AWS Command Line Interface 用户指南 AWS CLI中的配置。

在命令提示符下输入以下命令进行检查 AWS CLI；它应该会显示帮助信息。

```
aws pcs help
```

AWS PCS 所需的 IAM 权限

您正在使用的 IAM 安全委托人必须有权使用 AWS PCS IAM 角色、服务关联角色 AWS CloudFormation、VPC 和相关资源。有关更多信息并 AWS 行计算服务的 Identity and Access 管理，请参阅AWS Identity and Access Management 用户指南中的和创建服务相关角色。您必须以同一用户身份完成本指南中的所有步骤。要查看当前用户，请运行以下命令：

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

在 AWS PCS 教程中使用 AWS CloudFormation

AWS PCS 教程包含许多步骤，旨在帮助您了解 AWS PCS 集群的各个部分以及创建集群所需的步骤。我们建议您至少 1 次完成教程步骤。在您对所涉及的内容有了很好的了解之后 AWS CloudFormation ，您可以使用自动快速创建示例集群。

CloudFormation 是一项 AWS 服务，使您能够以可预测的方式重复创建和配置 AWS 基础架构部署。您可以使用 CloudFormation 模板将示例集群的 AWS 资源自动配置为单个单元，称为堆栈。使用完堆栈后，您可以将其删除。

有关更多信息，请参阅 开始使用 CloudFormation 和 AWS PCS。

为 PCS 创建 VPC 和子网 AWS

您可以使用 CloudFormation 模板创建 VPC 和子网。使用以下 URL 下载 CloudFormation 模板，然后在CloudFormation 控制台中上传模板以创建新 CloudFormation堆栈。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用 CloudFormation 控制台。

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

在 CloudFormation 控制台中打开模板后，输入以下选项。您可以使用模板中提供的默认值。

- 在 “提供堆栈名称” 下：

- 在堆栈名称下，输入：

```
hpc-networking
```

- 在 “参数” 下：
  - 在 VPC 下：
    - 在下CidrBlock方输入：

```
10.3.0.0/16
```

  - 在子网 A 下：
    - 在 CidrPublicSubnetA 下，输入：

```
10.3.0.0/20
```

    - 在 CidrPrivateSubnetA 下，输入：

```
10.3.128.0/20
```

  - 在子网 B 下：
    - 在 CidrPublicSubnetB 下，输入：

```
10.3.16.0/20
```

    - 在 CidrPrivateSubnetB 下，输入：

```
10.3.144.0/20
```

  - 在子网 C 下：
    - 对于 ProvisionSubnetsC，选择 True
    - 在 CidrPublicSubnetC 下，输入：

```
10.3.32.0/20
```

    - 在 CidrPrivateSubnetC 下，输入：

```
10.3.160.0/20
```

  - 在 “能力” 下：

- 选中 “我确认 AWS CloudFormation 可能会创建 IAM 资源” 复选框。

监控 CloudFormation 堆栈的状态。当它到达时`CREATE_COMPLETE`，在新 VPC 中找到默认安全组的 ID。您将在本教程的后面部分使用该 ID。

查找集群 VPC 的默认安全组

要在新 VPC 中查找默认安全组的 ID，请按照以下步骤操作：

- 导航到 Amazon VPC 控制台。
- 在 VPC 控制面板下，选择按 VPC 筛选。
  - 选择名称开头的 VPC `hpc-networking`。
  - 在 “安全” 下，选择 “安全组”。
- 查找名为的组的安全组 ID `default`。它有描述`default VPC security group`。您可以稍后使用该 ID 来配置 EC2 启动模板。

为 AWS PCS 创建安全组

AWS PCS 依靠安全组来管理进出集群及其计算节点组的网络流量。有关此主题的详细信息，请参阅安全组要求和注意事项。

在此步骤中，您将使用 CloudFormation 模板创建两个安全组。

- 集群安全组，它支持 AWS PCS 控制器、计算节点和登录节点之间的通信。
- 入站 SSH 安全组，您可以选择将其添加到登录节点以支持 SSH 访问

为 AWS PCS 创建安全组

您可以使用 CloudFormation 模板来创建安全组。使用以下 URL 下载 CloudFormation 模板，然后在CloudFormation 控制台中上传模板以创建新 CloudFormation 堆栈。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用 CloudFormation 控制台。

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

在 AWS CloudFormation 控制台中打开模板后，输入以下选项。请注意，某些选项将在模板中预先填充，您只需将其保留为默认值即可。

- 在 “提供堆栈名称” 下
  - 在堆栈名称下，输入：

```
getstarted-sg
```

- 在 “参数” 下
  - 在下 VpcId，选择名称以开头的 VPC `hpc-networking`。
  - （可选）在下方 ClientIpCidr，为入站 SSH 安全组输入限制性更强的 IP 范围。我们建议您使用自己的 IP/子网对此进行限制（x.x.x.x/32 代表您自己的 IP，x.x.x.x/24 表示范围。 将 x.x.x.x 替换为你自己的公有 IP。 你可以使用诸如 https://ifconfig.co/ 之类的工具获取你的公有 IP)

监控 CloudFormation 堆栈的状态。当它到达安全`CREATE_COMPLETE`组时，资源就准备好了。

创建了两个安全组，名称为：

- `cluster-getstarted-sg`— 这是群集安全组
- `inbound-ssh-getstarted-sg`— 这是一个允许入站 SSH 访问的安全组

在 AWS PCS 中创建集群

在 AWS PCS 中，集群是一种用于管理资源和运行工作负载的永久资源。您可以在新的或现有 VPC 的子网中为特定的调度程序（AWS PCS 当前支持 Slurm）创建集群。集群接受和调度作业，还会启动处理这些任务的计算节点（EC2 实例）。

创建集群的步骤

1. 打开 AWS PCS 控制台并选择创建集群。
2. 在集群详细信息部分，输入以下字段：
   - 集群名称-输入 `get-started`
   - 调度程序 — 选择 Slur m 版本 25.05
   - 控制器大小-选择 “小”
3. 在 “网络” 部分中，为以下字段选择值：
   - VPC — 选择名为的 VPC `hpc-networking:Large-Scale-HPC`
   - 子网-选择名称开头的子网 `hpc-networking:PrivateSubnetA`

- 安全组-选择名为的集群安全组 `cluster-getstarted-sg`

4. 选择创建集群。

> **Note**
>
> 置备集群时，状态字段显示正在创建。创建集群可能需要几分钟。

在亚马逊 Elastic File System 中为 AWS PCS 创建共享存储

Amazon Elastic File System (Amazon EFS) 是一项提供无服务器、完全弹性的文件存储的 AWS 服务，因此您无需预置或管理存储容量和性能即可共享文件数据。有关更多信息，请参阅 Amazon Elastic File System 用户指南中的什么是 Amazon Elastic File System？。

AWS PCS 演示群集使用 EFS 文件系统在群集节点之间提供共享的主目录。在集群所在的 VPC 中创建 EFS 文件系统。

创建 Amazon EFS 文件系统

1. 前往 Amazon EFS 控制台。
2. 确保将其设置为与试用 AWS PCS 的 AWS 区域 位置相同。
3. 选择创建文件系统。
4. 在创建文件系统页面上，设置以下参数：
   - 对于名称，输入 `getstarted-efs`。
   - 在虚拟私有云 (VPC) Pri vate Cloud 下，选择名为的 VPC `hpc-networking:Large-Scale-HPC`
   - 选择创建。这会将您返回到 “文件系统” 页面。
5. 记下文件系统的**`getstarted-efs`**文件系统 ID。稍后您将使用此信息。

在 Amazon for Lustre 中 FSx 为 AWS PCS 创建共享存储

Amazon f FSx or Lustre 可以轻松且经济高效地启动和运行流行的高性能 Lustre 文件系统。您可以将 Lustre 用于速度至关重要的工作负载，例如机器学习、高性能计算（HPC）、视频处理和财务建模。有关更多信息，请参阅 Amazon FSx for Lustre 是什么？ 在《Amazon f FSx or Lustre 用户指南》中。

AWS PCS 演示集群可以使用 fo FSx r Lustre 文件系统在群集节点之间提供高性能共享目录。在与您的集群相同 FSx 的 VPC 中创建 for Lustre 文件系统。

创建 for L FSx ustre 文件系统

1. 前往 Amazon FSx 控制台。
2. 确保将控制台设置为使用与您的集群 AWS 区域 相同的控制台。
3. 选择创建文件系统。
   - 在 “选择文件系统类型” 中，选择 Amazon f FSx or Lustre，然后选择 “下一步”。
4. 在指定文件系统详细信息页面上，设置以下参数：
   - 在 “文件系统详细信息” 下
     - 对于名称，输入 `getstarted-fsx`。
     - 对于部署和存储类型，请选择持久、SSD
     - 对于每单位存储的吞吐量，请选择 125 MB /s/TiB
     - 对于存储容量，请输入 1.2 TiB
     - 在 “元数据配置” 中，选择 “自动”
     - 对于数据压缩类型，选择 LZ4
   - 在 “网络与安全” 下
     - 对于虚拟私有云 (VPC) Pri vate Cloud，请选择名为的 VPC `hpc-networking:Large-Scale-HPC`
     - 对于 VPC 安全组，请将安全组命名为 `default`
     - 对于子网，请选择名称开头的子网 `hpc-networking:PrivateSubnetA`
   - 将其他选项设置为其默认值。
   - 选择下一步。
5. 在 “查看并创建” 页面上，选择 “创建文件系统”。这会将您返回到 “文件系统” 页面。
6. 导航到您创建的 for Lustre 文件系统的详细信息页面。 FSx
7. 记下文件系统 ID 和装载名称。稍后您将使用此信息。

> **Note**
>
> “状态” 字段显示在置备文件系统时正在创建。创建文件系统可能需要几分钟。等到它完成后再继续本教程的其余部分。

在 AWS PCS 中创建计算节点组

计算节点组是 AWS PCS 启动和管理的计算节点（EC2 实例）的虚拟集合。在定义计算节点组时，您需要指定常见特征，例如 EC2 实例类型、最小和最大实例数、目标 VPC 子网、首选购买选项和自定义启动配置。 AWS 根据这些设置，PCS 可以有效地启动、管理和终止计算节点组中的计算节点。演示集群使用计算节点组为用户访问提供登录节点，使用单独的计算节点组来处理作业。以下主题描述了在集群中设置这些计算节点组的过程。

主题

为 AWS PCS 创建实例配置文件

计算节点组在创建时需要实例配置文件。如果使用 AWS 管理控制台 创建 Amazon EC2 的角色，则控制台自动创建实例配置文件，将其命名为与角色相同的名称。有关更多信息，请参阅AWS Identity and Access Management 用户指南中的使用实例配置文件。

在以下步骤中，您可以使用为 Amazon EC2 创建角色，该角色还会为您的计算节点组创建实例配置文件。 AWS 管理控制台

创建角色和实例配置文件

- 导航到 IAM 控制台。
- 在访问管理下，选择策略。
  - 选择 Create policy (创建策略)。
  - 在 “指定权限” 下的 “策略编辑器” 中，选择 JSON。
  - 将文本编辑器的内容替换为以下内容：

JSON

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

- 选择下一步。
- 在 “查看并创建” 下，在 “策略名称” 中输入AWSPCS-getstarted-policy。
- 选择创建策略。

- 在 Access management（访问管理）下，请选择 Roles（角色）。
- 选择创建角色。
- 在 “选择可信实体” 下：
  - 对于 “可信实体类型”，选择 “AWS 服务”
  - 在 “用例” 下，选择 EC2。
    - 然后，在 “为指定服务选择用例” 下，选择 EC2。
  - 选择下一步。
- 在 “添加权限” 下：
  - 在权限策略中，搜索 AWSPCS-getstarted-policy。
  - 选中 AWSPCS-getstarted-policy 旁边的复选框将其添加到角色中。
  - 在权限策略中，搜索 Amazon SSMManaged InstanceCore。
  - 选中 Amazon SSMManaged InstanceCore 旁边的复选框将其添加到角色中。
  - 选择下一步。
- 在 “名称” 下，查看并创建：
  - 在 “角色详情” 下：
    - 对于 Role name（角色名称），输入 AWSPCS-getstarted-role。
  - 请选择 Create role（创建角色）。

为 AWS PCS 创建启动模板

创建计算节点组时，您需要提供一个 EC2 启动模板， AWS PCS 使用该模板来配置其启动的 EC2 实例。这包括实例启动时运行的安全组和脚本等设置。

在此步骤中，将使用一个 CloudFormation 模板来创建两个 EC2 启动模板。一个模板将用于创建登录节点，另一个模板将用于创建计算节点。它们之间的关键区别在于，可以将登录节点配置为允许入站 SSH 访问。

访问 CloudFormation 模板

使用以下 URL 下载 CloudFormation 模板，然后在CloudFormation 控制台中上传模板以创建新 CloudFormation 堆栈。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用 CloudFormation 控制台。

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

使用 CloudFormation 模板创建 EC2 启动模板

使用以下步骤在 CloudFormation 控制台中完成 CloudFormation 模板

- 在 “提供堆栈名称” 下：
  - 在堆栈名称下，输入`getstarted-lt`。
- 在 “参数” 下：
  - 在安全之下
    - 对于 VpcSecurityGroupId，选择在您的集群 VPC `default` 中命名的安全组。
    - 对于 ClusterSecurityGroupId，请选择名为的群组 `cluster-getstarted-sg`
    - 对于 SshSecurityGroupId，请选择名为的群组 `inbound-ssh-getstarted-sg`
    - 对于 SshKeyName，请选择您的首选 SSH 密钥对。
  - 在 “文件系统” 下
    - 对于 EfsFilesystemId，请输入您在本教程前面创建的 EFS 文件系统的文件系统 ID。
    - 对于 FSxLustreFilesystemId，请输入您在本教程前面创建 FSx 的 for Lustre 文件系统的文件系统 ID。
    - 对于 FSxLustreFilesystemMountName，输入 Lustre 文件 FSx 系统的相同装载名称。
- 选择 “下一步”，然后再次选择 “下一步”。

- 选择提交。

监控 CloudFormation 堆栈的状态。当它到达CREATE_COMPLETE时，启动模板就可以使用了。

> **Note**
>
> 要查看 CloudFormation 模板创建的所有资源，请打开CloudFormation 控制台。选择 getstarted-lt 堆栈，然后选择 Resources（资源）选项卡。

在 AWS PCS 中为登录节点创建计算节点组

计算节点组是 AWS PCS 启动和管理的计算节点（EC2 实例）的虚拟集合。在定义计算节点组时，您需要指定常见特征，例如 EC2 实例类型、最小和最大实例数、目标 VPC 子网、首选购买选项和自定义启动配置。 AWS 根据这些设置，PCS 可以有效地启动、管理和终止计算节点组中的计算节点。

在此步骤中，您将启动一个提供集群交互式访问权限的静态计算节点组。你可以使用 SSH 或 Amazon EC2 Systems Manager (SSM) 登录它，然后运行 shell 命令并管理 Slurm 任务。

创建计算节点组

- 打开 AWS PCS 控制台并导航到集群。
- 选择名为的集群 get-started
- 导航到 “计算节点组”，然后选择 “创建”。
- 在计算节点组设置部分，提供以下内容：
  - 计算节点组名称-输入login。
- 在 “计算配置” 下，输入或选择以下值：
  - EC2 启动模板-选择名称所在的启动模板 login-getstarted-lt
  - IAM 实例配置文件-选择名为的实例配置文件 AWSPCS-getstarted-role
  - 子网-选择名称开头的子网。hpc-networking:PublicSubnetA
  - 实例-选择c6i.xlarge。
  - 扩展配置-对于最小实例数，请输入1。在 “最大实例数” 中，输入**1**。
- 在 “其他设置” 下，指定以下内容：
  - AMI ID — 选择要使用的 AMI，其名称格式如下：

```
aws-pcs-sample_ami-amzn2-platform-slurm-version
```

有关该示例的更多信息 AMIs，请参阅在 AWS PCS 上使用示例亚马逊系统映像 (AMIs)。

- 选择创建计算节点组。

在配置计算节点组时，状态字段显示正在创建。在本教程的下一步中，您可以继续进行下一步。

创建用于在 AWS PCS 中运行计算任务的计算节点组

在此步骤中，您将启动一个计算节点组，该节点组可以弹性扩展以运行提交到集群的作业。

创建计算节点组

- 打开 AWS PCS 控制台并导航到集群。
- 选择名为的集群 `get-started`
- 导航到 “计算节点组”，然后选择 “创建”。
- 在计算节点组设置部分，提供以下内容：
  - 计算节点组名称-输入`compute-1`。
- 在 “计算配置” 下，输入或选择以下值：
  - EC2 启动模板-选择名称所在的启动模板 `compute-getstarted-lt`
  - IAM 实例配置文件-选择名为的实例配置文件 `AWSPCS-getstarted-role`
  - 子网-选择名称开头的子网。`hpc-networking:PrivateSubnetA`
  - 实例-选择`c6i.xlarge`。
  - 扩展配置-对于最小实例数，请输入`0`。在 “最大实例数” 中，输入**4**。
- 在 “其他设置” 下，指定以下内容：
  - AMI ID — 选择要使用的 AMI，其名称格式如下：

```
aws-pcs-sample_ami-amzn2-platform-slurm-version
```

有关该示例的更多信息 AMIs，请参阅在 AWS PCS 上使用示例亚马逊系统映像 (AMIs)。

- 选择创建计算节点组。

在配置计算节点组时，状态字段显示正在创建。

> **Important**
>
> 等待 “状态” 字段显示为 “活动”，然后再继续本教程的下一步。

创建队列来管理 AWS PCS 中的作业

您将作业提交到队列以运行它。在 AWS PCS 安排作业在计算节点组上运行之前，该作业将一直保留在队列中。每个队列都与一个或多个计算节点组相关联，这些节点组提供了执行处理所需的 EC2 实例。

在此步骤中，您将创建一个使用计算节点组处理作业的队列。

创建队列

- 打开 AWS PCS 控制台。
- 选择名为 `get-started` 的集群。
- 导航到 “计算节点组”，并确保该`compute-1`组的状态为 “活动”。

> **Important**
>
> 在继续下一步之前，`compute-1`群组的状态必须为 “活动”。

- 导航到队列并选择创建队列。
  - 在队列配置部分，提供以下值：
    - 队列名称-输入以下内容：`demo`
    - 计算节点组-选择名为的计算节点组`compute-1`。
- 选择创建队列。

创建队列时，状态字段显示正在创建。

> **Important**
>
> 等待 “状态” 字段显示为 “活动”，然后再继续本教程的下一步。

Connect 连接到你的 AWS PCS 集群

`login`计算节点组的状态变为 “活动” 后，您可以连接到它创建的 EC2 实例。

连接到登录节点

- 打开 AWS PCS 控制台并导航到集群。
- 选择名为 `get-started` 的集群。
- 选择计算节点组。
- 导航到名为的计算节点组`login`。
- 找到计算节点组 ID。
- 在另一个浏览器窗口或选项卡中，打开 Amazon EC2 控制台。
  - 选择实例。
  - 搜索带有以下标签的 EC2 实例。*`node-group-id`*替换为上一步中的计算节点组 ID 的值。应该有 1 个实例。

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

  - Connect 连接到 EC2 实例。您可以使用会话管理器或 SSH。

    Session Manager

    - 选择实例。
    - 选择连接。
    - 在 Connect to 实例下，选择会话管理器。
    - 选择连接。
    - 选择连接。交互式终端将在您的浏览器中启动。

    SSH

    - 选择实例。
    - 选择连接。
    - 在 “连接到实例” 下，选择 SSH 客户端。
    - 按照控制台提供的说明进行操作。

      > **Note**
      >
      > 该实例的用户名**`ec2-user`**不是`root`。

探索 AWS PCS 中的集群环境

登录到集群后，您可以运行 shell 命令。例如，您可以更改用户、处理共享文件系统上的数据以及与 Slurm 交互。

更改用户

如果您使用会话管理器登录到集群，则可能以身份进行连接`ssm-user`。这是为会话管理器创建的特殊用户。使用以下命令在 Amazon Linux 2 上切换到默认用户。如果您使用 SSH 连接，则无需执行此操作。

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

使用共享文件系统

您可以使用命令确认 EFS 文件系统和 FSx Lustre 文件系统是否可用。`df -h`集群上的输出应类似于以下内容：

```
[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`文件系统装载了 127.0.0.1，容量非常大。这是您在本教程前面部分创建的 EFS 文件系统。此处写入的所有文件都将在集群中的所有节点`/home`上都可用。

`/shared`文件系统挂载一个私有 IP，容量为 1.2 TB。这是您在本教程前面 FSx 部分创建的 for Lustre 文件系统。此处写入的所有文件都将在集群中的所有节点`/shared`上都可用。

与 Slurm 互动

主题

- 列出队列和节点
- 显示职位

列出队列和节点

您可以使用列出队列及其关联的节点sinfo。集群的输出应类似于以下内容：

```
[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 ~]$
```

记下名为的分区demo。它的状态为up，最多有 4 个节点。它与节点组中的compute-1节点相关联。如果您编辑计算节点组并将最大实例数增加到 8，则会读取节点数8并读取节点列表compute-1-[1-8]。如果您创建了第二个名为 4 个节点test的计算节点组，并将其添加到demo队列中，则这些节点也将显示在节点列表中。

显示职位

您可以使用列出系统上所有处于任何状态的作业squeue。集群的输出应类似于以下内容：

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

当你有 Slurm 任务待处理或正在运行时，请稍后squeue再试运行。

在 AWS PCS 中运行单节点作业

要使用 Slurm 运行作业，您需要准备一个指定作业要求的提交脚本，然后使用命令将其提交到队列。sbatch通常，这是在共享目录中完成的，因此登录和计算节点有一个用于访问文件的公共空间。

连接到集群的登录节点，并在其 shell 提示符下运行以下命令。

- 成为默认用户。切换到共享目录。

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

- 使用以下命令创建示例作业脚本：

```
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
```

- 将作业脚本提交给 Slurm 调度器：

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

- 提交作业后，它将以数字形式返回作业 ID。使用该 ID 来检查任务状态。用返回*job-id*的数字替换以下命令中的数字sbatch。

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

Example

```
squeue --job 1
```

该squeue命令返回的输出类似于以下内容：

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

- 继续检查作业的状态，直到它达到R（正在运行）状态。当squeue没有返回任何东西时，工作就完成了。
- 检查/shared目录的内容。

```
ls -alth /shared
```

命令输出类似于以下内容：

```
-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.err这些文件名为single.1.out、由您的集群的一个计算节点写入。由于作业是在共享目录 (/shared) 中运行的，因此它们也可以在您的登录节点上使用。这就是您为该集群配置 fo FSx r Lustre 文件系统的原因。

- 检查single.1.out文件内容。

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

输出类似于以下内容：

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

在 PCS 中使用 Slurm 运行多节点 MPI 作业 AWS

这些说明演示了如何使用 Slurm 在 PCS 中运行消息传递接口 (MPI) 作业。 AWS

在登录节点的 shell 提示符下运行以下命令。

- 成为默认用户。切换到其主目录。

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

- 使用 C 编程语言创建源代码。

```
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
```

- 加载 OpenMPI 模块。

```
module load openmpi
```

- 编译 C 程序。

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

- 编写 Slurm 作业提交脚本。

```
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
```

- 切换到共享目录。

```
cd /shared
```

- 提交作业脚本。

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

- squeue用于监视作业直至其完成。
- 检查以下内容multi.out：

```
cat multi.out
```

输出类似于以下内容。请注意，每个等级都有自己的 IP 地址，因为它运行在不同的节点上。

```
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
```

删除你的 AWS PCS AWS 资源

完成为本教程创建的集群和节点组后，应删除已创建的资源。

> **Important**
>
> 你需要为你运行的所有资源收取账单费用 AWS 账户

删除您为本教程创建的 AWS PCS 资源

- 打开 AWS PCS 控制台。
- 导航到名为 get-starte d 的集群。
- 导航到 “队列” 部分。
- 选择名为 demo 的队列。
- 选择删除。

> **Important**
>
> 等到队列被删除后再继续。

- 导航至 “计算节点组” 部分。
- 选择名为 compute-1 的计算节点组。
- 选择删除。
- 选择名为 log in 的计算节点组。
- 选择删除。

> **Important**
>
> 等到两个计算节点组都被删除后再继续。

- 在用于入门的集群详细信息页面中，选择删除。

> **Important**
>
> 等到集群被删除后再继续执行后续步骤。

删除您为本教程创建的其他 AWS 资源

- 打开 IAM 管理控制台。
  - 选择角色。
  - 选择名为 AWSPCS-getstarted-role 的角色，然后选择删除。
  - 删除角色后，选择策略。
  - 选择名为 AWSPCS-getstarted-policy 的策略，然后选择删除。
- 打开 CloudFormation 管理控制台。
  - 选择名为 getstart ed-lt 的堆栈。
  - 选择删除。

> **Important**
> 等待堆栈删除后再继续。

- 打开 Amazon EFS 控制台。
  - 选择文件系统。
  - 选择名为 getstart ed-efs 的文件系统。
  - 选择删除。

> **Important**
> 等待文件系统删除后再继续。

- 打开亚马逊 FSx 控制台。
  - 选择文件系统。
  - 选择名为 getstart ed-fsx 的文件系统。
  - 选择删除。

> **Important**
> 等待文件系统删除后再继续。

- 打开 CloudFormation 管理控制台。
  - 选择名为 getstart ed-sg 的堆栈。

- 选择删除。
- 打开 CloudFormation 管理控制台。
  - 选择名为 hpc-networking 的堆栈。
  - 选择 Delete（删除）。

开始使用 CloudFormation 和 AWS PCS

您可以使用 AWS CloudFormation 创建 AWS PCS 集群。 CloudFormation 使您能够以可预测的方式重复创建和配置 AWS 基础架构部署。您可以使用自动配置来自许多 AWS 服务的资源，CloudFormation 以便在中构建高度可靠、可扩展且经济实惠的应用程序， AWS Cloud 而无需创建和配置底层 AWS 基础架构。 CloudFormation 允许您使用模板文件将资源集合作为一个单元（称为堆栈）一起创建和删除。有关的更多信息 CloudFormation，请参阅什么是 CloudFormation？ 在《AWS CloudFormation 用户指南》中。有关中 AWS PCS 资源类型的更多信息 CloudFormation，请参阅《AWS CloudFormation 用户指南》中的 AWS PCS 资源类型参考。

主题

- CloudFormation 用于创建示例 AWS PCS 集群
- Connect 连接到使用创建的 AWS PCS 集群 CloudFormation
- 清理 AWS PCS 集群 CloudFormation
- AWS PCS CloudFormation 模板的一部分
- CloudFormation 用于创建示例 AWS PCS 集群的模板

CloudFormation 用于创建示例 AWS PCS 集群

以下过程使用中的 CloudFormation 模板创建示例 AWS PCS 集群。 AWS 管理控制台 有关的更多信息 CloudFormation，请参阅什么是 CloudFormation？ 在《AWS CloudFormation 用户指南》中。有关中 AWS PCS 资源类型的更多信息 CloudFormation，请参阅《AWS CloudFormation 用户指南》中的 AWS PCS 资源类型参考。

创建示例集群

1. 选择 AWS 区域 要在其中创建集群的（该链接将打开带有模板的 CloudFormation 控制台）：
   - 美国东部（弗吉尼亚北部）(us-east-1)
   - 美国东部（俄亥俄）(us-east-2)
   - 美国西部（俄勒冈）(us-west-2)
   - 亚太地区（孟买）(ap-south-1)
   - 亚太地区（新加坡）(ap-southeast-1)
   - 亚太地区（悉尼）(ap-southeast-2)

- 亚太地区（东京）(ap-northeast-1)
- 欧洲（法兰克福）（eu-central-1）
- 欧洲（爱尔兰）(eu-west-1)
- 欧洲（伦敦）(eu-west-2)
- 欧洲（巴黎）（eu-west-3）
- 欧洲（米兰）(eu-south-1)
- 欧洲（斯德哥尔摩）（eu-north-1）
- AWS GovCloud (美国东部) (us-gov-east-1)
- AWS GovCloud (美国西部) (us-gov-west-1)

2. 在“提供堆栈名称”下，输入描述性名称。这是您的 CloudFormation 堆栈的名称。模板使用此值作为 AWS PCS 集群的名称。
3. 在“参数”下：
   a. 在下方 SlurmVersion，选择您希望集群使用的 Slurm 版本。
   b. 在下方 NodeArchitecture，选择 x86 部署使用 x86_64 兼容实例的集群，或者选择 Graviton 以使用 Ar m64 实例。
   c. 对于 KeyName，选择 SSH 密钥对来访问集群登录节点。确保您拥有所选密钥对的 PEM 文件。
   d. 对于 ClientIpCidr，以 CIDR 格式输入 IP 范围以控制对登录节点的访问。

      > ⚠ Warning
      > 默认值为`0.0.0.0/0`允许从所有 IP 地址进行访问。

   e. 将 HpcRecipesS3Bucket 和的值保留HpcRecipesBranch为其默认值。
4. 在“能力和转换”下：
   a. 选中复选框以确认 CloudFormation 将创建 IAM 资源。
   b. 选中复选框以确认 CloudFormation 将创建具有自定义名称的 IAM 资源。
   c. 选中该复选框`CAPABILITY_AUTO_EXPAND`以确认新堆栈。有关更多信息，请参阅《AWS CloudFormation API Reference》中的 CreateStack。
5. 选择创建堆栈。
6. 监控堆栈的状态。在堆栈的状态为之后，您就可以连接到集群`CREATE_COMPLETE`。

Connect 连接到使用创建的 AWS PCS 集群 CloudFormation

根据 CloudFormation 模板创建 AWS PCS 集群后，您可以使用 AWS PCS 控制台（在 AWS 管理控制台）管理集群。您还可以连接到集群的 1 个登录节点来管理集群、运行作业和管理数据。CloudFormation 堆栈提供了可用于连接到集群的链接。

连接到您的集群

1. 打开 CloudFormation 控制台
2. 选择您创建的堆栈。
3. 选择堆栈的 “输出” 选项卡。

   堆栈提供以下链接：

   - PcsConsoleUrl— 选择此链接以打开选定集群的 AWS PCS 控制台。您可以使用它来浏览集群、节点组和队列配置。
   - Ec@@ 2 ConsoleUrl — 选择此链接可打开 Amazon EC2 控制台，该控制台经过筛选后显示集群的登录节点组管理的实例。

     在此视图中，您可以选择一个实例并选择 Connect。示例集群的实例支持入站 SSH 和 Web 浏览器中的 AWS Systems Manager 连接。有关更多信息，请参阅 Connect 连接到你的 AWS PCS 集群。

     连接到登录实例后，您可以按照中的教程进行操作探索 AWS PCS 中的集群环境。

清理 AWS PCS 集群 CloudFormation

如果您曾经 CloudFormation 创建 AWS PCS 集群，则可以打开CloudFormation 控制台并删除堆栈，以删除集群及其所有关联资源。

> **Important**
>
> 对于示例集群，如果您在集群中创建了其他计算节点组或队列（除了示例 CloudFormation 模板创建的`login`和`compute-1`组之外），则必须使用 AWS PCS 控制台或 AWS CLI 删除这些资源，然后才能删除 CloudFormation 堆栈。有关更多信息，请参阅 在 AWS PCS 中删除集群。

AWS PCS CloudFormation 模板的一部分

一个 CloudFormation 模板有 1 个或多个部分，每个部分都有特定的用途。 CloudFormation 在模板中定义标准格式、语法和语言。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用 CloudFormation 模板。

CloudFormation 模板是高度可定制的，因此它们的格式可能会有所不同。要了解创建 AWS PCS 集群所需的 CloudFormation 模板部分，我们建议您查看我们为创建示例集群而提供的示例模板。本主题简要介绍了该示例模板的各个部分。

> **Important**
>
> 本主题中的代码示例不完整。省略号 (`[...]`) 的存在表示还有其他未显示的代码。要下载完整的 YAML 格式 CloudFormation 模板，请参阅。CloudFormation 用于创建示例 AWS PCS 集群的模板

目录

标题

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: AWS Parallel Computing Service "getting started" cluster
```

`AWSTemplateFormatVersion`标识模板符合的模板格式版本。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的CloudFormation模板格式版本语法。

`Transform`指定用于处理模板的宏。 CloudFormation 有关更多信息，请参阅《AWS CloudFormation 用户指南》中的CloudFormation模板转换部分。`AWS::Serverless-2016-10-31`转换

CloudFormation 允许处理用 AWS Serverless Application Model (AWS SAM) 语法编写的模板。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的AWS::Serverless转换。

元数据

```
### Stack metadata
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: PCS Cluster configuration
        Parameters:
          - SlurmVersion
          - ManagedAccounting
          - AccountingPolicyEnforcement
      - Label:
          default: PCS ComputeNodeGroups configuration
        Parameters:
          - NodeArchitecture
          - KeyName
          - ClientIpCidr
      - Label:
          default: HPC Recipes configuration
        Parameters:
          - HpcRecipesS3Bucket
          - HpcRecipesBranch
```

CloudFormation 模板的metadata部分提供有关模板本身的信息。示例模板创建了一个使用 PC AWS S 的完整高性能计算 (HPC) 集群。示例模板的元数据部分声明了控制如何 CloudFormation 启动（配置）相应堆栈的参数。有控制架构选择 (NodeArchitecture)、Slurm 版本 (SlurmVersion) 和访问控制 (KeyName和ClientIpCidr) 的参数。

Parameters

该Parameters部分定义了模板的自定义参数。 CloudFormation 使用这些参数定义来构造和验证从此模板启动堆栈时与之交互的表单。

```
Parameters:

  NodeArchitecture:
    Type: String
    Default: x86
```

```
    AllowedValues:
      - x86
      - Graviton
    Description: Processor architecture for the login and compute node instances

  SlurmVersion:
    Type: String
    Default: 25.05
    Description: Version of Slurm to use
    AllowedValues:
         - 24.11
         - 25.05

  ManagedAccounting:
    Type: String
    Default: 'disabled'
    AllowedValues:
      - 'enabled'
      - 'disabled'
    Description: Monitor cluster usage, manage access control, and enforce resource
limits with Slurm accounting. Requires Slurm 24.11 or newer.

  AccountingPolicyEnforcement:
    Description: Specify which Slurm accounting policies to enforce
    Type: String
    Default: none
    AllowedValues:
      - none
      - 'associations,limits,safe'

  KeyName:
    Description: SSH keypair to log in to the head node
    Type: AWS::EC2::KeyPair::KeyName
    AllowedPattern: ".+"  # Required

  ClientIpCidr:
    Description: IP(s) allowed to access the login node over SSH. We recommend that
you restrict it with your own IP/subnet (x.x.x.x/32 for your own ip or x.x.x.x/24 for
range. Replace x.x.x.x with your own PUBLIC IP. You can get your public IP using tools
such as https://ifconfig.co/)
    Default: 127.0.0.1/32
    Type: String
    AllowedPattern: (\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})
```

```
    ConstraintDescription: Value must be a valid IP or network range of the form
 x.x.x.x/x.

  HpcRecipesS3Bucket:
    Type: String
    Default: aws-hpc-recipes
    Description: HPC Recipes for AWS S3 bucket
    AllowedValues:
         - aws-hpc-recipes
         - aws-hpc-recipes-dev
  HpcRecipesBranch:
    Type: String
    Default: main
    Description: HPC Recipes for AWS release branch
    AllowedPattern: '^(?!.*/\.git$)(?!.*/\.)(?!.*\\.\.)[a-zA-Z0-9-_\.]+$'
```

映像

本Mappings节定义了根据特定条件或依赖关系指定值的键值对。

```
Mappings:

  Architecture:
    AmiArchParameter:
      Graviton: arm64
      x86: x86_64
    LoginNodeInstances:
      Graviton: c7g.xlarge
      x86: c6i.xlarge
    ComputeNodeInstances:
      Graviton: c7g.xlarge
      x86: c6i.xlarge
```

资源

该Resources部分将要配置和配置的 AWS 资源声明为堆栈的一部分。

```
Resources:

  [...]
```

该模板分层配置示例集群基础架构。它以 V Networking PC 配置开头。存储由双系统提供：EfsStorage用于共享存储和FSxLStorage高性能存储。核心集群是通过建立的PCSCluster。

```
  Networking:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        ProvisionSubnetsC: "False"
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/net/hpc_large_scale/assets/main.yaml'

  EfsStorage:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        SubnetIds: !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
        SubnetCount: 1
        VpcId: !GetAtt [ Networking, Outputs.VPC ]
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/storage/efs_simple/assets/main.yaml'

  FSxLStorage:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        PerUnitStorageThroughput: 125
        SubnetId: !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
        VpcId: !GetAtt [ Networking, Outputs.VPC ]
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/storage/fsx_lustre/assets/persistent.yaml'

  [...]

  # Cluster
  PCSCluster:
    Type: AWS::PCS::Cluster
    Properties:
      Name: !Sub '${AWS::StackName}'
      Size: SMALL
      Scheduler:
        Type: SLURM
        Version: !Ref SlurmVersion
```

```
      Networking:
        SubnetIds:
          - !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
        SecurityGroupIds:
          - !GetAtt [ PCSSecurityGroup, Outputs.ClusterSecurityGroupId ]
```

对于计算资源，该模板创建两个节点组：PCSNodeGroupLogin用于单个登录节点和PCSNodeGroupCompute最多四个计算节点。这些节点组受权限PCSInstanceProfile和实例配置PCSLaunchTemplate的支持。

```
  # Compute Node groups
  PCSInstanceProfile:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        # We have to regionalize this in case CX use the template in more than one
 region. Otherwise,
        # the create action will fail since instance-role-${AWS::StackName} already
 exists!
        RoleName: !Sub '${AWS::StackName}-${AWS::Region}'
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/pcs/getting_started/assets/pcs-iip-minimal.yaml'

  PCSLaunchTemplate:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        VpcDefaultSecurityGroupId: !GetAtt [ Networking, Outputs.SecurityGroup ]
        ClusterSecurityGroupId: !GetAtt [ PCSSecurityGroup,
 Outputs.ClusterSecurityGroupId ]
        SshSecurityGroupId: !GetAtt [ PCSSecurityGroup,
 Outputs.InboundSshSecurityGroupId ]
        EfsFilesystemSecurityGroupId: !GetAtt [ EfsStorage, Outputs.SecurityGroupId ]
        FSxLustreFilesystemSecurityGroupId: !GetAtt [ FSxLStorage,
 Outputs.FSxLustreSecurityGroupId ]
        SshKeyName: !Ref KeyName
        EfsFilesystemId: !GetAtt [ EfsStorage, Outputs.EFSFilesystemId ]
        FSxLustreFilesystemId: !GetAtt [ FSxLStorage, Outputs.FSxLustreFilesystemId ]
        FSxLustreFilesystemMountName: !GetAtt [ FSxLStorage,
 Outputs.FSxLustreMountName ]
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/pcs/getting_started/assets/cfn-pcs-lt-efs-fsxl.yaml'
```

```
  # Compute Node groups - Login Nodes
  PCSNodeGroupLogin:
    Type: AWS::PCS::ComputeNodeGroup
    Properties:
      ClusterId: !GetAtt [PCSCluster, Id]
      Name: login
      ScalingConfiguration:
        MinInstanceCount: 1
        MaxInstanceCount: 1
      IamInstanceProfileArn: !GetAtt [ PCSInstanceProfile, Outputs.InstanceProfileArn ]
      CustomLaunchTemplate:
        TemplateId: !GetAtt [ PCSLaunchTemplate, Outputs.LoginLaunchTemplateId ]
        Version: 1
      SubnetIds:
        - !GetAtt [ Networking, Outputs.DefaultPublicSubnet ]
      AmiId: !GetAtt [PcsSampleAmi, AmiId]
      InstanceConfigs:
        - InstanceType: !FindInMap [ Architecture, LoginNodeInstances, !Ref
NodeArchitecture ]

  # Compute Node groups - Compute Nodes
  PCSNodeGroupCompute:
    Type: AWS::PCS::ComputeNodeGroup
    Properties:
      ClusterId: !GetAtt [PCSCluster, Id]
      Name: compute-1
      ScalingConfiguration:
        MinInstanceCount: 0
        MaxInstanceCount: 4
      IamInstanceProfileArn: !GetAtt [ PCSInstanceProfile, Outputs.InstanceProfileArn ]
      CustomLaunchTemplate:
        TemplateId: !GetAtt [ PCSLaunchTemplate, Outputs.ComputeLaunchTemplateId ]
        Version: 1
      SubnetIds:
        - !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
      AmiId: !GetAtt [PcsSampleAmi, AmiId]
      InstanceConfigs:
        - InstanceType: !FindInMap [ Architecture, ComputeNodeInstances, !Ref
NodeArchitecture ]
```

Job 调度是通过处理的PCSQueueCompute。

```
  PCSQueueCompute:
    Type: AWS::PCS::Queue
    Properties:
      ClusterId: !GetAtt [PCSCluster, Id]
      Name: demo
      ComputeNodeGroupConfigurations:
        - ComputeNodeGroupId: !GetAtt [PCSNodeGroupCompute, Id]
```

AMI 选择通过 Pcs AMILookup Fn Lambda 函数和相关资源自动进行。

```
  PcsAMILookupRole:
    Type: AWS::IAM::Role
    [...]

  PcsAMILookupFn:
    Type: AWS::Lambda::Function
    Properties:
      Runtime: python3.12
      Handler: index.handler
      Role: !GetAtt PcsAMILookupRole.Arn
      Code:
        [...]
      Timeout: 30
      MemorySize: 128

  # Example of using the custom resource to look up an AMI
  PcsSampleAmi:
    Type: Custom::AMILookup
    Properties:
      ServiceToken: !GetAtt PcsAMILookupFn.Arn
      OperatingSystem: 'amzn2'
      Architecture: !FindInMap [ Architecture, AmiArchParameter, !Ref
 NodeArchitecture ]
      SlurmVersion: !Ref SlurmVersion
```

输出

该模板 URLs 通过ClusterId、PcsConsoleUrl、和输出集群识别和管理Ec2ConsoleUrl。

```
Outputs:
```

```
  ClusterId:
    Description: The Id of the PCS cluster
    Value: !GetAtt [ PCSCluster, Id ]

  PcsConsoleUrl:
    Description: URL to access the cluster in the PCS console
    Value: !Sub
      - https://${ConsoleDomain}/pcs/home?region=${AWS::Region}#/clusters/${ClusterId}
      - { ConsoleDomain: !If [ GovCloud, 'console.amazonaws-us-gov.com', !If [ China,
 'console.amazonaws.cn', !Sub '${AWS::Region}.console.aws.amazon.com']],
          ClusterId: !GetAtt [ PCSCluster, Id ]
        }
    Export:
      Name: !Sub ${AWS::StackName}-PcsConsoleUrl

  Ec2ConsoleUrl:
    Description: URL to access instance(s) in the login node group via Session Manager
    Value: !Sub
      - https://${ConsoleDomain}/ec2/home?region=
${AWS::Region}#Instances:instanceState=running;tag:aws:pcs:compute-node-group-id=
${NodeGroupLoginId}
      - { ConsoleDomain: !If [ GovCloud, 'console.amazonaws-us-gov.com', !If [ China,
 'console.amazonaws.cn', !Sub '${AWS::Region}.console.aws.amazon.com']],
          NodeGroupLoginId: !GetAtt [ PCSNodeGroupLogin, Id ]
        }
    Export:
      Name: !Sub ${AWS::StackName}-Ec2ConsoleUrl
```

CloudFormation 用于创建示例 AWS PCS 集群的模板

AWS 区域 名字	AWS 区域	查看源代码	启动堆栈
美国东部（弗吉尼亚州北部）	us-east-1	下载 YAML	Launch Stack
美国东部（俄亥俄州）	us-east-2	下载 YAML	Launch Stack
美国西部（俄勒冈州）	us-west-2	下载 YAML	Launch Stack

AWS 区域 名字	AWS 区域	查看源代码	启动堆栈
亚太地区（孟买）	ap-south-1	下载 YAML	Launch Stack
亚太地区（新加坡）	ap-southeast-1	下载 YAML	Launch Stack
亚太地区（悉尼）	ap-southeast-2	下载 YAML	Launch Stack
亚太地区（东京）	ap-northeast-1	下载 YAML	Launch Stack
欧洲地区（法兰克福）	eu-central-1	下载 YAML	Launch Stack
欧洲地区（爱尔兰）	eu-west-1	下载 YAML	Launch Stack
欧洲（伦敦）	eu-west-2	下载 YAML	Launch Stack
欧洲地区（巴黎）	eu-west-3	下载 YAML	Launch Stack
欧洲地区（米兰）	eu-south-1	下载 YAML	Launch Stack
欧洲地区（斯德哥尔摩）	eu-north-1	下载 YAML	Launch Stack
AWS GovCloud（美国东部）	us-gov-east-1	下载 YAML	Launch Stack
AWS GovCloud（美国西部）	us-gov-west-1	下载 YAML	Launch Stack

AWS PCS 集群

AWS PCS 集群由以下组件组成：

- HPC 系统调度程序软件的托管实例，例如 Slurm 控制守护程序 ()。`slurmctld`
- 与 HPC 系统调度程序集成的组件，用于配置和管理 Ama EC2 zon 实例。
- 与 HPC 系统调度程序集成的组件，用于向 Amazon 传输日志和指标。 CloudWatch

这些组件在由管理的账户中运行 AWS。它们共同管理您的客户账户中的 Amazon EC2 实例。 AWS PCS 在您的 Amazon VPC 子网中配置弹性网络接口，以提供从计划程序软件到 Amazon EC2 实例的连接（例如，支持在这些实例上安排批处理作业，并允许用户运行计划程序命令来列出和管理这些任务）。

主题

在 AWS PCS 中创建集群

本主题概述了可用选项，并介绍了在并 AWS 行计算服务 (AWS PCS) 中创建集群时应考虑的事项。如果这是您第一次创建 AWS PCS 集群，我们建议您遵循此操作开始使用并 AWS 行计算服务。本教程可以帮助您创建可运行的 HPC 系统，而无需扩展到所有可能的可用选项和系统架构。

> **Note**
>
> 创建集群后，您可以修改许多配置设置，而无需重建基础架构。有关更多信息，请参阅 在 AWS PCS 中更新集群。

> **Note**
>
> 您可以配置自定义 Slurm 设置以实现高级调度策略和资源管理。有关更多信息，请参阅 在 PCS 中配置自定义 Slurm 设置 AWS。

先决条件

- 符合AWS PCS 联网要求的现有 VPC 和子网。在部署集群用于生产用途前，我们建议您彻底了解 VPC 和子网要求。要创建 VPC 和子网，请参阅为您的 PC AWS S 集群创建 VPC。
- 有权创建和管理 AWS PCS 资源的 IAM 委托人。有关更多信息，请参阅 并 AWS 行计算服务的 Identity and Access 管理。

创建 AWS PCS 集群

您可以使用 AWS 管理控制台 或 AWS CLI 来创建集群。

AWS 管理控制台

创建集群

1. 在 https://console.aws.amazon.com/pcs/home#/clusters 上打开 AWS PCS 控制台，然后选择创建集群。
2. 在集群设置部分中，输入以下字段：
   - 集群名称-您的集群的名称。名称只能包含字母数字字符（区分大小写）和连字符。它必须以字母字符开头，长度不能超过 40 个字符。该名称在创建集群时 AWS 区域 AWS 账户 使用的名称必须是唯一的。
   - 调度程序-选择调度程序和版本。有关更多信息，请参阅 PCS 中的 Slurm 版本 AWS。
   - 控制器大小-选择控制器的大小。这决定了 AWS PCS 集群可以管理多少并发任务和计算节点。您只能在创建集群时设置控制器的大小。有关尺码的更多信息，请参阅AWS PCS 中的集群大小。
3. 在“网络”部分中，为以下字段选择值：
   - 网络类型-为您的集群选择 IP 地址类型。您的集群可以同时使用 IPv4 或 IPv6，但不能同时使用两者。VPC 和子网必须使用相同的网络地址类型。每个子网使用的 IP 地址块必须至

少有 1 个可用地址。 AWS 在每个子网中保留一些地址。有关更多信息，请参阅《Amazon VPC 用户指南》中的子网 CIDR 块。

- VPC — 选择一个满足 AWS PCS 要求的现有 VPC。有关更多信息，请参阅 AWS PCS VPC 和子网要求和注意事项。创建集群后，您无法更改其 VPC。如果未 VPCs 列出，则必须先创建一个。
- 子网-列出了所选 VPC 中的所有可用子网。选择符合 AWS PCS 子网要求的子网。有关更多信息，请参阅 AWS PCS VPC 和子网要求和注意事项。我们建议您选择私有子网，以避免将您的调度程序端点暴露给公共 Internet。
- 安全组-指定您希望 AWS PCS 与其为集群创建的网络接口关联的安全组。您必须至少选择一个允许集群与其计算节点之间通信的安全组。您可以选择 “快速创建安全组”，让 AWS PCS 在选定的 VPC 中创建具有必要配置的安全组，也可以选择现有安全组。有关更多信息，请参阅 安全组要求和注意事项。

4. （可选）在 Slurm 记账配置部分，您可以启用 Slurm 记账并设置记账参数。有关更多信息，请参阅 PCS 中的 Slurm 会计 AWS。
5. （可选）在 Slurm 配置部分，您可以添加参数名称和值对来配置其他 Slurm 设置。有关支持的参数的完整列表，请参阅PCS 集群的自定义 Slurm 设置 AWS。
6. （可选）在 “标签” 下，将所有标签添加到您的 AWS PCS 集群。
7. 选择创建集群。 AWS PCS 创建集群`Creating`时会显示状态字段。此过程可能耗时数分钟。

> **Important**
>
> AWS 区域 每个集群只能有 1 个处于同一个`Creating`状态的集群 AWS 账户。 AWS 如果您在尝试创建集群时已有处于`Creating`状态的集群，PCS 会返回错误。

AWS CLI

创建集群

1. 使用以下命令创建集群。在运行命令之前，进行以下替换：
   - *`region`*替换为您要 AWS 区域 在其中创建集群的 ID，例如`us-east-1`。
   - 将 *`my-cluster`* 替换为您的集群名称。名称只能包含字母数字字符（区分大小写）和连字符。它必须以字母字符开头，长度不能超过 40 个字符。该名称在创建集群的 AWS 账户 位置 AWS 区域 和创建集群的地方必须是唯一的。

- *25.05*替换为任何支持的 Slurm 版本。

  > **Note**
  >
  > AWS PCS 目前支持 Slurm 25.05 和 24.11。

- *SMALL*替换为任何支持的群集大小。这决定了 AWS PCS 集群可以管理多少并发任务和计算节点。它只能在创建集群时进行设置。有关尺码的更多信息，请参阅AWS PCS 中的集群大小。
- 的值替换为您自己的值。subnetIds我们建议您选择私有子网，以避免将您的调度程序端点暴露给公共 Internet。
- 指定您希望 AWS PCS 与它为集群创建的网络接口关联的。securityGroupIds安全组必须与集群位于同一 VPC 中。您必须至少选择一个允许集群与其计算节点之间通信的安全组。有关更多信息，请参阅 安全组要求和注意事项。

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM,version=25.05 \
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1
```

- 要使用 IPv6，请networkType=IPV6添加到--networking配置中。

```
  --networking networkType=IPV6,subnetIds=subnet-
ExampleId1,securityGroupIds=sg-ExampleId1
```

- 或者，您可以添加自定义 Slurm 行为的--slurm-configration选项并指定 Slurm 配置选项。以下示例将缩减空闲时间设置为 60 分钟（3600 秒），启用 Slurm 记账，并将slurm.conf设置指定为的值。slurmCustomSettings有关更多信息，请参阅 PCS 中的 Slurm 会计 AWS。

  > **Note**
  >
  > Slurm 24.11 或更高版本支持记账。

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM,version=25.05 \
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1
    --slurm-configuration
 scaleDownIdleTimeInSeconds=3600,accounting='{mode=STANDARD}',slurmCustomSettings='[{p
```

2. 配置集群可能需要几分钟。可使用以下命令查询集群的状态。在集群的状态字段变为之前，请勿继续创建队列或计算节点组ACTIVE。

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

> **Important**
>
> AWS 区域 每个集群只能有 1 个处于同一个Creating状态的集群 AWS 账户。 AWS 如果您在尝试创建集群时已有处于Creating状态的集群，PCS 会返回错误。

为您的集群推荐的后续步骤

- 添加计算节点组。
- 添加队列。
- 启用日志记录。

在 AWS PCS 中更新集群

AWS PCS 允许您在创建集群配置后通过 UpdateCluster API 或控制台更新集群配置。无需重建基础架构即可修改集群设置，这样可以减少运营开销并最大限度地减少中断。

集群更新的好处

更新 AWS PCS 集群可以让您在不中断服务的情况下调整 HPC 基础架构以适应新的需求。配置更改需要几分钟，而不是重建集群所需的小时或更长时间。对于需要最少停机时间的生产环境以及需要在工作负载模式变化时调整群集设置的团队来说，此功能非常重要。

支持的配置更改

您可以修改三个主要类别的设置：

- 记账配置-启用或禁用托管记账并配置保留设置。
- 缩小行为-调整`scaleDownIdleTime`参数，该参数控制动态实例在 AWS PCS 自动终止之前保持空闲状态的时间。
- Slurm 自定义设置-修改适用于集群级别的任何支持的 Slurm 设置，包括 Prolog、Epilog 和。SelectTypeParameters

限制

集群创建后，您无法修改某些配置。这些指令包括：

- 安全组配置
- VPC 子网选择
- 集群大小
- Slurm 版本
- 集群名称

这些设置是集群架构的基础，需要创建新的集群才能对其进行修改。

集群更新的先决条件

在更新集群之前，请确保满足以下条件：

- 集群必须处于`ACTIVE`、`UPDATE_FAILED`或`SUSPENDED`状态
- 所有关联的资源（队列、计算节点组）都必须处于`ACTIVE`状态
- 您必须拥有相应的 IAM 权限才能 UpdateCluster 执行该操作
- 无法进行其他更新操作

更新流程和任务影响

在更新操作期间，即使集群控制器短暂无法访问，计算节点仍会继续运行现有作业。但是，在此期间，系统无法接受新的工作提交或做出日程安排决定。

您可以通过控制台和 API 接口监控集群更新。更新期间，集群将进入以下状态：

- UPDATING-更新正在进行中
- ACTIVE-更新已成功完成
- UPDATE_FAILED-更新遇到错误

更新期间的账单

更新操作期间，您的 AWS PCS 集群将继续按标准小时收费。当您更新集群以禁用记账功能时，一旦集群进入UPDATING状态，记账功能的计费就会停止。启用记账功能时，直到集群成功完成更新并恢复到ACTIVE状态后才会开始计费。

主题

- 更新 AWS PCS 集群
- 有关在 AWS PCS 中更新集群的常见问题
- AWS PCS 集群更新疑难解答

更新 AWS PCS 集群

使用这些步骤修改集群上的调度程序设置、记账配置和 Slurm 自定义设置。有关更多信息，请参阅 PCS 集群的自定义 Slurm 设置 AWS。

先决条件

- 集群必须处于ACTIVEUPDATE_FAILED、或SUSPENDED状态
- 所有关联的资源（队列、计算节点组）都必须处于ACTIVE状态
- 无法进行其他更新操作

过程

AWS 管理控制台

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 在导航窗格中，选择集群。
3. 选择要更新的集群。

4. 选择编辑。
5. 在编辑集群页面上，修改所需的设置：
   - 在 S cheduler 配置下，更新 Scale-down 空闲时间，以控制动态实例在自动终止之前保持空闲状态的时间。
   - 根据需要修改 Prolog、Epilog 和 Select-Type 参数设置。
   - 启用、禁用或配置托管记账的保留时间。
   - 在 “其他调度程序设置” 下，添加、编辑或删除 Slurm 自定义设置。有关支持的参数的更多信息，请参阅PCS 集群的自定义 Slurm 设置 AWS。

   > **Note**
   >
   > 无法编辑的字段显示为只读并显示其当前值。

6. 选择 “更新” 以提交更改。
7. 监控集群状态，在此过程中显示为 “正在更新”。更新成功完成后，状态会发生变化。

AWS CLI

1. 打开终端或命令提示符。
2. 使用以下命令验证集群状态：

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

3. 使用以下示例之一提交更新请求：
   - 要启用管理记账，请执行以下操作：

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration 'accounting={mode=STANDARD}'
```

   - 要更新 Slurm Prolog 设置，请执行以下操作：

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'SlurmCustomSettings=[{parameterName=Prolog,parameterValue="/path/to/
prolog.sh"}]'
```

- 要更新缩减空闲时间，请执行以下操作：

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration 'scaleDownIdleTimeInSeconds=300'
```

4. 通过检查集群状态来监控更新进度：

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

更新请求成功后，该命令将返回包含所有更改的 Cluster 对象。完成ACTIVE后，集群状态UPDATING将从变为。

有关在 AWS PCS 中更新集群的常见问题

获取有关在 AWS PCS 中更新集群配置的常见问题的答案。

我可以修改哪些设置？

您可以修改记账配置（启用/禁用托管会计）、缩减行为（scaleDownIdle时间参数）以及适用于集群级别的任何受支持的 Slurm 自定义设置。您无法修改安全组、VPC 子网、集群大小、Slurm 版本或集群名称。

我可以排队多个更新吗？

不是。 在提交其他更新之前，必须等待集群恢复到ACTIVE状态。所有关联的资源（队列、计算节点组）也必须处于ACTIVE状态。

我可以取消集群更新操作吗？

不可以，您无法取消正在进行的集群更新操作。

我可以在集群更新时提交任务吗？

我们建议您避免在集群更新期间提交作业。更新过程中，Slurm 控制器可能不可用。

集群更新期间，我的作业会继续运行吗？

是的，即使在更新过程中群集控制器短暂无法访问，正在运行的作业也会继续在计算节点上执行。但是，在控制器再次可用之前，任务状态可能不会更新。

更新期间账单会受到什么影响？

更新操作期间，按小时收取标准费用。禁用记账时，当集群进入UPDATING状态时，计费将停止。启用记账功能时，从集群成功恢复到ACTIVE状态时开始计费。

AWS PCS 集群更新疑难解答

本主题可帮助您识别和解决更新集群配置时可能出现的常见问题。

更新因记账配置错误而失败

常见原因

集群进入UPDATE_FAILED状态，错误消息表明存在记账配置问题。当记账配置与当前 Slurm 版本不兼容或包含无效设置时，通常会发生这种情况。

解决方案

检查您的记账设置是否与集群的 Slurm 版本兼容，并使用有效的配置参数提交更正后的更新请求。

更新失败，出现自定义设置错误

常见原因

集群进入UPDATE_FAILED状态，错误消息表明 Slurm 自定义设置存在问题。当您提供无效的 Slurm 参数值或不支持的参数组合时，就会发生这种情况。

解决方案

根据支持的参数验证您的 Slurm 自定义设置，并使用有效的参数值和组合提交更正后的更新请求。

无法提交更新请求

常见原因

控制台中的更新按钮被禁用或者 API 返回 400 级错误。当群集未处于适当的状态、关联的资源未处于活动状态或您的配置中存在验证失败时，就会发生这种情况。

解决方案

等待集群和所有关联资源达到ACTIVE状态，然后检查您的配置是否存在验证错误，然后再重新提交更新请求。

验证错误

常见原因

该命令立即返回，并带有 400 级 HTTP 错误和描述性消息。出现这种情况的原因是群集状态、资源状态或配置参数无效。

解决方案

请解决响应中提到的特定验证错误，然后重试更新操作。

在 AWS PCS 中删除集群

本主题概述了如何删除 AWS PCS 集群。

删除 AWS PCS 集群时的注意事项

- 必须先删除与集群关联的所有队列，然后才能删除集群。有关更多信息，请参阅 在 AWS PCS 中删除队列。
- 必须先删除与集群关联的所有计算节点组，然后才能删除集群。有关更多信息，请参阅 删除 AWS PCS 中的计算节点组。

删除集群

您可以使用 AWS 管理控制台 或 AWS CLI 来删除集群。

AWS 管理控制台

删除集群

1. 打开 AWS PCS 控制台。
2. 选择要删除的集群。
3. 选择删除。
4. 将显示集群状态字段`Deleting`。可能需要几分钟的时间才能完成。

AWS CLI

删除集群

1. 使用以下命令删除集群，并使用以下替换命令：

   - *region-code*替换为 AWS 区域 您的集群所在的。
   - *my-cluster*替换为集群的名称或 ID。

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

2. 删除集群可能需要几分钟。您可以使用以下命令检查集群的状态。

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

AWS PCS 中的集群大小

AWS PCS 提供高度可用且安全的集群，同时自动执行修补、节点配置和更新等关键任务。

创建集群时，您可以根据两个因素为其选择大小：

- 它将管理的计算节点数量
- 控制器在任何给定时间跟踪的作业数量

> **Note**
>
> 任务计数包括正在运行的作业、待处理的作业和最近完成的作业。控制器会在短时间内跟踪已完成的作业，然后才会被清除。在高作业吞吐量期间，这可能会导致跟踪的任务总数超过您观察到的活跃作业数。

> **Important**
>
> 创建集群后，您无法更改集群大小。如果需要更改大小，则必须创建一个新集群。

Slurm 集群大小	托管的实例数量	控制器跟踪的作业数量
Small	最多 32	最多 256
中	最多 512	最高 8192
大型	直到 2048	最高 16384

示例

- 如果您的集群将有多达 24 个托管实例并运行多达 100 个作业，请选择 Sm all。
- 如果您的集群将有多达 24 个托管实例并运行多达 1000 个作业，请选择 “中”。
- 如果您的集群将有多达 1000 个托管实例并运行多达 100 个作业，请选择大型。
- 如果您的集群将有多达 1000 个托管实例并运行多达 10,000 个作业，请选择大型。

在 AWS PCS 中使用集群密钥

作为创建集群的一部分， AWS PCS 会创建连接到集群上的作业调度程序所需的集群密钥。您还可以创建 AWS PCS 计算节点组，这些节点组定义了为响应扩展事件而启动的实例集。 AWS PCS 使用集群密钥配置由这些计算节点组启动的实例，以便它们可以连接到作业调度器。在某些情况下，您可能需要手动配置 Slurm 客户端。示例包括构建永久登录节点或设置具有作业管理功能的工作流管理器。

AWS PCS 将集群密钥存储为托管密钥，前缀pcs!为 AWS Secrets Manager。使用 AWS PCS的费用已包含在使用密钥的费用中。您可以轮换集群密钥 AWS Secrets Manager ，以维护安全合规性并修复潜在的安全漏洞。

主题

- 用于 AWS Secrets Manager 查找集群密钥
- 使用 AWS PCS 查找集群密钥
- 获取 Slurm 集群的秘密
- 在 AWS PCS 中轮换集群密钥

用于 AWS Secrets Manager 查找集群密钥

AWS 管理控制台

1. 导航到 Secrets Manager 控制台。
2. 选择 “密钥”，然后搜索前pcs!缀。

> **Note**
>
> AWS PCS 集群密钥的名称形式*cluster-id*为 AWS PCS 集群 ID，pcs!slurm-secret-*cluster-id*其中。

AWS CLI

每 AWS 个 PCS 集群密钥也都标有aws:pcs:*cluster-id*。您可以使用以下命令获取集群的密钥 ID。在运行命令之前进行以下替换：

- *region*替换 AWS 区域 为可在其中创建集群，例如us-east-1。
- *cluster-id*替换为 AWS PCS 集群的 ID 以查找其集群密钥。

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

使用 AWS PCS 查找集群密钥

您可以使用查找 AWS PCS 集群密钥的 ARN。 AWS CLI 输入以下命令，进行以下替换：

- *region*替换 AWS 区域 为可在其中创建集群，例如us-east-1。
- *my-cluster*替换为您的集群的名称或标识符。

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

以下示例输出来自该get-cluster命令。你可以secretVersion一起使用secretArn和来获得秘密。

```
{
    "cluster": {
        "name": "get-started",
        "id": "pcs_123456abcd",
        "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_123456abcd",
        "status": "ACTIVE",
        "createdAt": "2024-12-17T21:03:52+00:00",
        "modifiedAt": "2024-12-17T21:03:52+00:00",
        "scheduler": {
            "type": "SLURM",
            "version": "25.05"
        },
        "size": "SMALL",
        "slurmConfiguration": {
            "authKey": {
                "secretArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:pcs!
slurm-secret-pcs_123456abcd-a12ABC",
                "secretVersion": "ef232370-d3e7-434c-9a87-ec35c1987f75"
            }
        },
        "networking": {
            "subnetIds": [
                "subnet-0123456789abcdef0"
            ],
            "securityGroupIds": [
                "sg-0123456789abcdef0"
            ]
        },
        "endpoints": [
            {
                "type": "SLURMCTLD",
                "privateIpAddress": "10.3.149.220",
                "port": "6817"
            }
        ]
    }
}
```

获取 Slurm 集群的秘密

你可以使用 Secrets Manager 获取 Slurm 集群密钥的当前 base64 编码版本。以下示例使用了。 AWS CLI在运行命令之前，请进行以下替换。

- *region*替换 AWS 区域 为可在其中创建集群，例如us-east-1。
- *secret-arn*替换为 AWS PCS 集群中的。secretArn

```
aws secretsmanager get-secret-value \
    --region region \
    --secret-id 'secret-arn' \
    --version-stage AWSCURRENT \
    --query 'SecretString' \
    --output text
```

有关如何使用 Slurm 集群密钥的信息，请参阅。使用独立实例作为 AWS PCS 登录节点

Permissions

您使用 IAM 委托人获取 Slurm 集群密钥。IAM 委托人必须拥有读取密钥的权限。有关更多信息，请参阅AWS Identity and Access Management 用户指南中的角色术语和概念。

以下 IAM 策略示例允许访问示例集群密钥。

```
{
    "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"
        }
    ]
}
```

在 AWS PCS 中轮换集群密钥

使用 AWS Secrets Manager 托管轮换在 AWS PCS 中轮换集群密钥。定期轮换密钥是在 HPC 环境中保持强大安全态势的最佳安全实践。此功能使您能够满足行业合规标准，包括要求定期轮换证书的 HIPAA 和 FedRAMP。

集群密钥有两个用途：对加入集群的计算节点进行身份验证，以及作为 Slurm REST API 身份验证的 JWT 密钥。旋转时，两个方面会同时受到影响。

集群密钥轮换的工作原理

手动准备以在密钥轮换期间保持集群的稳定性：

1. 准备 — 将所有计算节点组的容量扩展到 0 并确保没有作业在运行
2. 轮换 — 通过 Secrets Manager 控制台或 API 启动轮换
3. 监控-通过 CloudTrail 事件跟踪进度
4. 恢复-将计算节点组扩展回所需的容量

在轮换期间，您的集群将保持`ACTIVE`状态，并且继续正常计费。该过程通常需要几分钟。

要求和限制

在轮换集群密钥之前，请完成以下要求：

- 集群必须处于`ACTIVE`或`UPDATE_FAILED`状态
- IAM 角色必须拥有`secretsmanager:RotateSecret`权限
- 所有计算节点组的容量都必须扩展到 0
- 轮换前停止所有作业

限制：

- 每次轮换都需要手动准备
- 现有的 JWT 代币失效，需要重新发行
- BYO 登录节点需要在轮换后手动更新密钥

主题

在 AWS PCS 中轮换集群密钥

轮换您的集群密钥以符合安全要求并解决潜在的漏洞。此过程需要将您的集群置于维护模式。

先决条件

- 具有`secretsmanager:RotateSecret`权限的 IAM 角色
- 集群处于`ACTIVE`或`UPDATE_FAILED`状态

过程

1. 通知集群用户即将到来的维护时段。
2. 通过将所有计算节点组缩放到 0 容量，将集群置于维护模式。
   a. 使用 UpdateComputeNodeGroup API 将所有计算节点组 maxInstanceCount 的 minInstanceCount 和设置为 0。
   b. 等到所有节点停止。
   c. 可选：在终止容量以优雅地处理任务之前，使用 Slurm 命令耗尽调度器队列。
3. 通过 Secrets Manager 启动轮换。
   - 控制台方法：
     - 导航到 Secrets Manager，选择您的集群密钥，然后选择轮换密钥。
   - API 方法：
     - 使用 Secrets Manager `rotate-secret` API。
4. 监控轮换进度。
   a. 通过 CloudTrail 事件跟踪进度。
   b. `lastRotatedDate`通过 Secrets Manager 控制台或 `secretsmanager:describeSecret` API 进行查看。
   c. 等待我们的`RotationSucceeded`或`RotationFailed` CloudTrail 活动。
5. 成功轮换后，恢复集群容量。

a. 使用 UpdateComputeNodeGroup API 将节点组重置为所需 min/max 容量。

b. 对于 AWS PC 管理的登录节点：无需执行其他操作。

c. 对于 BYO 登录节点：

i. Connect 连接到登录节点。

ii. 使用 Sec `/etc/slurm/slurm.key` rets Manager 中的新密钥进行更新。

iii. 重启 Slurm Auth and Cred Kiosk 守护程序（sackd）。

有关 AWS PCS 中集群密钥轮换的常见问题

查找有关 AWS PCS 中集群密钥轮换的常见问题的答案。

什么是集群密钥？

集群密钥是一种安全凭证，可实现 Slurm 控制器和 AWS PCS 计算节点之间的安全通信。它还用作 Slurm REST API 身份验证的 JSON 网络令牌 (JWT) 密钥。

集群密钥和 JWT 密钥有什么区别？

在 AWS PCS 中，集群密钥和 JWT 密钥是相同的资源，用于不同的用途。集群密钥对 Slurm 内部通信进行身份验证，而 JWT 密钥对 REST API 身份验证的令牌进行签名。旋转时，两个方面会同时受到影响。

轮换需要多长时间？

轮换过程通常需要几分钟。您的集群仍处于 ACTIVE 状态，并且在轮换期间继续正常计费。

我可以安排自动轮换吗？

你可以在 Secrets Manager 中启用定时轮换。但是，初始版本需要在每次轮换之前进行手动准备（将节点组缩放到 0）。

我现有的 JWT 代币在轮换后还能使用吗？

不，现有的 JWT 代币在轮换后会失效。为 REST API 客户端发行新令牌。

我在哪里可以找到我的集群密钥？

你可以在 Secrets Manager 控制台或 AWS PCS 控制台中找到你的集群密钥。有关详细说明，请参阅用于 AWS Secrets Manager 查找集群密钥和使用 AWS PCS 查找集群密钥。

为什么轮换需要将节点组缩放到 0？

轮换不需要正在运行的实例，以确保密钥更新过程中的集群稳定性。这样可以防止新旧密钥之间的身份验证冲突。

此功能支持哪些合规性要求？

此功能使 AWS PCS能够满足行业合规标准，包括HIPAA和FedRAMP，这两个标准要求定期轮换凭证，这是其安全控制的一部分。

排除 AWS PCS 中集群密钥轮换的问题

如果环境准备不当，集群密钥轮换将失败。最常见的原因是集群中的实例处于活动状态。为防止失败：

1. 将所有节点组的容量设置为 0。
2. 等待节点停止。
3. 验证您的集群是否处于以下状态：`CREATE_FAILED`、`DELETE_FAILED`、`RESUMING`、`SUSPENDING`、或`SUSPENDED`。

如果轮换失败：

- 出现一个 RotationFailed CloudTrail 事件
- 集群密钥保持不变
- 查看 RotationFailed 活动了解 CloudTrail 详情
- 完成成功轮换的所有准备步骤

AWS PCS 计算节点组

AWS PCS 计算节点组是节点的逻辑集合（Amazon EC2 实例）。这些节点可用于运行计算作业，以及提供对 HPC 系统的交互式、基于 shell 的访问。计算节点组由创建节点的规则组成，包括要使用的 Amazon EC2 实例类型、要运行的实例数量、使用竞价型实例还是按需实例、要使用哪些子网和安全组，以及如何在每个实例启动时对其进行配置。更新这些规则后， AWS PCS 会更新与计算节点组关联的资源以使其匹配。

主题

在 AWS PCS 中创建计算节点组

本主题概述了可用选项，并介绍了在并 AWS 行计算服务 (AWS PCS) 中创建计算节点组时应考虑的事项。如果这是您第一次在 AWS PCS 中创建计算节点组，我们建议您按照中的教程进行操作开始使用并 AWS 行计算服务。本教程可以帮助您创建可运行的 HPC 系统，而无需扩展到所有可能的可用选项和系统架构。

> **Note**
>
> 您可以在计算节点组上配置自定义 Slurm 设置，以控制资源利用率和节点级行为。有关更多信息，请参阅 在 PCS 中配置自定义 Slurm 设置 AWS。

> **Important**
>
> AWS PCS 目前需要 IPv4 支持本地节点通信的内核，即使您在 IPv6仅限网络中使用 AWS PCS 也是如此。有关更多信息，请参阅 适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

先决条件

- 足够的服务配额可以在您的中启动所需数量的 EC2 实例 AWS 区域。您可以使用AWS 管理控制台来检查和请求增加服务配额。
- 满足 PCS 联网要求的现有 V AWS PC 和子网。我们建议您在部署用于生产的集群之前，充分了解这些要求。有关更多信息，请参阅 AWS PCS VPC 和子网要求和注意事项。您也可以使用 CloudFormation 模板创建 VPC 和子网。 AWS 提供了 CloudFormation 模板的 HPC 配方。有关更多信息，请参阅 aws-hpc-recipes上的 GitHub。
- 一个 IAM 实例配置文件，有权调用 AWS PCS `RegisterComputeNodeGroupInstance` API 操作并访问您的节点组实例所需的任何其他 AWS 资源。有关更多信息，请参阅 AWS 并行计算服务的 IAM 实例配置文件。
- 您的节点组实例的启动模板。有关更多信息，请参阅 在 AWS PCS 上使用亚马逊 EC2 启动模板。
- 要创建使用 Amazon EC2 竞价型实例的计算节点组，您必须拥有AWSServiceRoleForEC2竞价服务相关角色。 AWS 账户有关更多信息，请参阅 适用于 AWS PCS 的 Amazon EC2 竞价角色。

在 AWS PCS 中创建计算节点组

您可以使用 AWS 管理控制台 或创建计算节点组 AWS CLI。

AWS 管理控制台

使用控制台创建计算节点组

1. 打开 AWS PCS 控制台。
2. 选择要在其中创建计算节点组的集群。导航到 “计算节点组”，然后选择 “创建”。
3. 在计算节点组设置部分，为您的节点组提供一个名称。名称只能包含区分大小写的字母数字字符和连字符。它必须以字母字符开头，长度不能超过 25 个字符。该名称在集群中必须是唯一的。
4. 在 “计算配置” 下，输入或选择以下值：

    a. EC2 启动模板-选择用于此节点组的自定义启动模板。启动模板可用于自定义网络设置，例如子网、安全组、监控配置和实例级存储。如果您尚未准备好启动模板，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板以了解如何创建启动模板。

> **Important**
>
> AWS PCS 为每个计算节点组创建托管启动模板。这些都被命名pcs-*identifier*-do-not-delete了。创建或更新计算节点组时请勿选择这些，否则节点组将无法正常运行。

b. EC2 启动模板版本-您必须选择自定义启动模板的版本。如果稍后更改版本，则必须更新计算节点组以检测启动模板中的更改。有关更多信息，请参阅 更新 AWS PCS 计算节点组。

c. AMI ID — 如果您的启动模板不包含 AMI ID，或者您想覆盖启动模板中的值，请在此处提供 AMI ID。请注意，用于节点组的 AMI 必须与 AWS PCS 兼容。您也可以选择由提供的示例 AMI AWS。有关此主题的更多信息，请参阅适用于 AWS PCS 的亚马逊机器映像 (AMIs)。

d. IAM 实例配置文件-为节点组选择实例配置文件。实例配置文件授予实例安全访问 AWS 资源和服务的权限。如果您还没有准备好基本配置文件，则可以选择 “创建基本配置文件”，让 AWS PCS 使用最低策略为您创建一个基本配置文件，或者参见AWS 并行计算服务的 IAM 实例配置文件。

e. 子网-在部署您的 PC AWS S 集群的 VPC 中选择一个或多个子网。如果您选择多个子网，则节点之间将无法使用 EFA 通信，并且不同子网中的节点之间的通信可能会增加延迟。确保您在此处指定的子网与您在 EC2 启动模板中定义的任何子网相匹配。

f. 实例-选择一个或多个实例类型来满足节点组中的扩展请求。所有实例类型都必须具有相同的处理器架构（x86_64 或 arm64），编号必须为 v。CPUs 如果实例有 GPUs，则所有实例类型必须具有相同数量的 GPUs。

g. 扩展配置-指定节点组的最小和最大实例数。您可以定义静态配置（其中有固定数量的节点在运行），也可以定义动态配置，其中最多可以运行最大数量的节点。对于静态配置，请将最小值和最大值设置为相同的、大于零的数字。对于动态配置，请将最小实例数设置为零，将最大实例数设置为大于零的数字。 AWS PCS 不支持混合使用静态和动态实例的计算节点组。

5. （可选）在 “其他设置” 下，指定以下内容：

a. 购买选项-选择按需实例、竞价型实例或现有容量块。如果您计划使用按需容量预留 (ODCR)，也请选择按需。有关更多信息，请参阅 ODCRs 与 AWS PCS 一起使用。选择容量块以使用现有的 Amazon EC2 容量块进行机器学习预留。有关更多信息，请参阅 将 Amazon EC2 容量块用于带有 AWS PCS 的机器学习。

b. 分配策略 — 如果您选择了竞价购买选项，则可以指定在启动节点组中的实例时如何选择竞价容量池。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的竞价型实例分配策略。如果您选择了按需购买选项，则此选项无效。

6. （可选）在Slurm自定义设置部分，您可以添加参数名称和值对来配置其他 Slurm 设置。有关支持的参数的完整列表，请参阅AWS PCS 计算节点组的自定义 Slurm 设置。
7. （可选）在标签下，将所有标签添加到您的计算节点组。
8. 选择创建计算节点组。当 AWS PCS 配置节点组`Creating`时，会显示 “状态” 字段。这个过程可能需要几分钟。

建议采取下一步行动

- 将您的节点组添加到 AWS PCS 中的队列中，使其能够处理作业。

AWS CLI

使用创建计算节点组 AWS CLI

使用以下命令创建队列。在运行命令之前，进行以下替换：

1. *`region`*替换为 AWS 区域 要在其中创建集群的 ID，例如`us-east-1`。
2. *`my-cluster`*替换为集群`clusterId`的名称或。
3. *`my-node-group`*替换为计算节点组的名称。名称只能包含字母数字字符（区分大小写）和连字符。它必须以字母字符开头，长度不能超过 25 个字符。该名称在集群中必须是唯一的。
4. *`subnet-ExampleID1`*替换为集群 VPC IDs 中的一个或多个子网。
5. *`lt-ExampleID1`*替换为自定义启动模板的 ID。如果您还没有准备好，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板以了解如何创建一个。

> **Important**
>
> AWS PCS 为每个计算节点组创建托管启动模板。这些都被命名`pcs-`*`identifier`*`-do-not-delete`了。创建或更新计算节点组时请勿选择这些，否则节点组将无法正常运行。

6. *`launch-template-version`*替换为特定的启动模板版本。 AWS PCS 将您的节点组与该特定版本的启动模板相关联。

7. *arn:InstanceProfile*替换为您的 IAM 实例配置文件的 ARN。如果您还没有准备好，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板获取指导。
8. 用整数值替换*min-instances*和*max-instances*。您可以定义静态配置（其中有固定数量的节点在运行），也可以定义动态配置，其中最多可以运行最大数量的节点。对于静态配置，请将最小值和最大值设置为相同的、大于零的数字。对于动态配置，请将最小实例数设置为零，将最大实例数设置为大于零的数字。 AWS PCS 不支持混合使用静态和动态实例的计算节点组。
9. *t3.large*替换为其他实例类型。您可以通过指定instanceType设置列表来添加更多实例类型。例如 *--instance-configs instanceType=c6i.16xlarge instanceType=c6a.16xlarge*。所有实例类型都必须具有相同的处理器架构（x86_64 或 arm64），编号必须为 v。CPUs 如果实例有 GPUs，则所有实例类型必须具有相同数量的 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
```

Example— 使用自定义 Slurm 设置创建计算节点组

```
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 \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Features,parameterValue="gpu,nvme"}]'
```

有关更多信息，请参阅 AWS PCS 计算节点组的自定义 Slurm 设置。

您可以将几个可选的配置设置添加到create-compute-node-group命令中。

- 您可以指定您的自定义启动模板`--amiId`是否不包含对 AMI 的引用，或者您是否希望覆盖该值。请注意，用于节点组的 AMI 必须与 AWS PCS 兼容。您也可以选择由提供的示例 AMI AWS。有关此主题的更多信息，请参阅适用于 AWS PCS 的亚马逊机器映像 (AMIs)。
- `--purchase-option`用于选择 AWS PCS 为您的计算节点组购买 EC2 实例的方式。按需是默认设置。
  - `ONDEMAND`— 使用按需实例。如果您计划使用按需容量预留 (ODCR)，也可以选择此选项。有关更多信息，请参阅 ODCRs 与 AWS PCS 一起使用。
  - `SPOT`— 使用竞价型实例。如果您选择竞价型实例，则还可以使用`--allocation-strategy`来定义 AWS PCS 在启动节点组中的实例时如何选择竞价型容量池。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的竞价型实例分配策略。
  - `CAPACITY_BLOCK`— 使用现有的 Amazon EC2 容量块进行机器学习预留。有关更多信息，请参阅 将 Amazon EC2 容量块用于带有 AWS PCS 的机器学习。
- 可以使用为节点组中的节点提供Slurm配置选项`--slurm-configuration`。您可以设置权重（调度优先级）和实际内存。权重较低的节点具有更高的优先级，并且单位是任意的。有关更多信息，请参阅Slurm文档中的重量。实际内存设置节点组中节点上实际内存的大小（以 GB 为单位）。在您的Slurm配置中，它应与 AWS PCS 中的集群`CR_CPU_Memory`选项结合使用。有关更多信息，请参阅 Slurm文档中的 RealMemory。

> **Important**
>
> 创建计算节点组可能需要几分钟。

您可以使用以下命令查询节点组的状态。在节点组的状态达到之前，您将无法将其与队列关联`ACTIVE`。

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

更新 AWS PCS 计算节点组

本主题概述了可用选项，并介绍了更新 AWS PCS 计算节点组时应考虑的事项。有关 Slurm 自定义设置的信息，请参阅。AWS PCS 计算节点组的自定义 Slurm 设置

更新 AWS PCS 计算节点组的选项

更新 AWS PCS 计算节点组使您能够更改 AWS PCS 启动的实例的属性以及这些实例的启动规则。例如，您可以将节点组实例的 AMI 替换为另一个安装了不同软件的 AMI。或者，您可以更新安全组以更改入站或出站网络连接。您还可以更改扩展配置和首选购买选项。

以下节点组设置在创建后无法更改：

- Name
- 实例

更新 AWS PCS 计算节点组时的注意事项

计算节点组定义了用于处理任务、提供交互式外壳访问权限和其他任务的 EC2 实例。它们通常与一个或多 AWS 个 PCS 队列相关联。在更新计算节点组以更改其行为（或其节点的行为）时，请考虑以下几点：

- 当计算节点组状态从 “更新” 变为 “活动” 时，对计算节点组属性的更改就会生效。使用更新的属性启动新实例。
- 不影响特定节点配置的更新不会影响正在运行的节点。例如，添加子网和更改分配策略。
- 如果您更新计算节点组的启动模板，则必须更新计算节点组才能使用新版本。
- 要在计算节点组的节点中添加或移除安全组，请编辑其启动模板并更新计算节点组。使用更新的安全组集启动新实例。
- 如果您直接编辑计算节点组使用的安全组，则该安全组会立即对正在运行的实例和将来的实例生效。
- 如果您在计算节点组使用的 IAM 实例配置文件中添加或删除权限，它将立即对正在运行的实例和 future 实例生效。
- 要更改计算节点组实例使用的 AMI，请更新计算节点组（或其启动模板）以使用新的 AMI，然后等待 AWS PCS 替换实例。
- AWS 在节点组更新操作后，PCS 会替换节点组中的现有实例。如果某个节点上正在运行作业，则允许在 AWS PCS 替换该节点之前完成这些作业。交互式用户进程（例如在登录节点实例上）终止。`Active`当 AWS PCS 将实例标记为替换时，节点组状态会恢复为，但实际替换发生在实例空闲时。
- 如果您减少计算节点组中允许的最大实例数， AWS PCS 会从 Slurm 中移除节点以达到新的最大值。 AWS PCS 会终止与已移除的 Slurm 节点关联的正在运行的实例。已移除的节点上正在运行的作业失败并返回其队列。

- AWS PCS 为每个计算节点组创建托管启动模板。他们被命名pcs-*identifier*-do-not-delete了。创建或更新计算节点组时请勿选择它们，否则节点组将无法正常运行。
- 如果您更新计算节点组以使用竞价作为其购买选项，则您的账户中必须具有AWSServiceRoleForEC2竞价服务相关角色。有关更多信息，请参阅 适用于 AWS PCS 的 Amazon EC2 竞价角色。

更新 AWS PCS 计算节点组

您可以使用 AWS 管理控制台或 AWS CLI 更新节点组。

AWS 管理控制台

更新计算节点组

1. 打开 AWS PCS 控制台，网址为 `https://console.aws.amazon.com/pcs/home#/clusters`
2. 选择要更新计算节点组的集群。
3. 导航到计算节点组，转到要更新的节点组，然后选择编辑。
4. 在 “计算配置”、“其他设置” 和 “Slurm自定义设置” 部分中，更新除以下值之外的所有值：
   - 实例-您无法更改计算节点组中的实例。

   有关 Slurm 自定义设置的更多信息，请参阅。AWS PCS 计算节点组的自定义 Slurm 设置
5. 选择更新。应用更改时，“状态” 字段将显示 “正在更新”。

> **Important**
> 计算节点组更新可能需要几分钟时间。

AWS CLI

更新计算节点组

1. 使用以下命令更新您的计算节点组。在运行命令之前，进行以下替换：
   a. *region-code*替换为您要在其中创建集群的 AWS 区域。

b. *my-node-group*替换computeNodeGroupId为计算节点组的名称或。

c. *my-cluster*替换为集群clusterId的名称或。

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

Example— 使用自定义 Slurm 设置更新计算节点组

```
aws pcs update-compute-node-group --region region-code \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Features,parameterValue="gpu,nvme"}]'
```

有关更多信息，请参阅 AWS PCS 计算节点组的自定义 Slurm 设置。

2. 更新除之外的所有节点组参数--instance-configs。例如，要设置新的 AMI ID，请传递用您选择*my-custom-ami-id*的 AMI 替换--amiId my-custom-ami-id的位置。

> **Important**
>
> 更新计算节点组可能需要几分钟。

您可以使用以下命令查询节点组的状态。

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

删除 AWS PCS 中的计算节点组

本主题概述了可用选项，并介绍了在 AWS PCS 中删除计算节点组时应考虑的事项。

删除计算节点组时的注意事项

计算节点组定义了用于处理任务、提供交互式外壳访问权限和其他任务的 EC2 实例。它们通常与一个或多 AWS 个 PCS 队列相关联。在删除计算节点组之前，请考虑以下事项：

- 计算节点组启动的任何 EC2 实例都将被终止。这将取消在这些实例上运行的作业，并终止正在运行的交互式进程。
- 必须先取消计算节点组与所有队列的关联，然后才能将其删除。有关更多信息，请参阅 更新 AWS PCS 队列。

删除计算节点组

您可以使用 AWS 管理控制台 或 AWS CLI 来删除计算节点组。

AWS 管理控制台

删除计算节点组

1. 打开 AWS PCS 控制台。
2. 选择计算节点组的集群。
3. 导航到计算节点组，然后选择要删除的计算节点组。
4. 选择删除。
5. 将显示 “状态” 字段`Deleting`。可能需要几分钟的时间才能完成。

> **Note**
> 您可以使用调度程序原生的命令来确认计算节点组已删除。例如，对于 Slur `squeue m` 使用`sinfo`或。

AWS CLI

删除计算节点组

- 使用以下命令删除具有以下替换内容的计算节点组：
  - *`region-code`*替换为 AWS 区域 您的集群所在的。

- *my-node-group*替换为计算节点组的名称或 ID。
- *my-cluster*替换为集群的名称或 ID。

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

删除计算节点组可能需要几分钟。

> **Note**
>
> 您可以使用调度程序原生的命令来确认计算节点组已删除。例如，对于 Slur squeue m 使用sinfo或。

在 AWS PCS 中获取计算节点组的详细信息

您可以使用 AWS 管理控制台 或 AWS CLI 来获取有关计算节点组的详细信息，例如其计算节点组 ID、亚马逊资源名称 (ARN) 和亚马逊系统映像 (AMI) ID。这些详细信息通常是 AWS PCS API 操作和配置的必填值。

AWS 管理控制台

获取计算节点组的详细信息

1. 打开 AWS PCS 控制台。
2. 选择 集群。
3. 选择计算节点组。
4. 从列表窗格中选择计算节点组。

AWS CLI

获取计算节点组的详细信息

1. 使用 ListClustersAPI 操作查找您的集群名称或 ID。

```
aws pcs list-clusters
```

输出示例：

```
{
    "clusters": [
        {
            "name": "get-started-cfn",
            "id": "pcs_abc1234567",
            "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567",
            "createdAt": "2025-04-01T20:11:22+00:00",
            "modifiedAt": "2025-04-01T20:11:22+00:00",
            "status": "ACTIVE"
        }
    ]
}
```

2. 使用 ListComputeNodeGroupsAPI 操作列出集群中的计算节点组。

```
aws pcs list-compute-node-groups --cluster-identifier cluster-name-or-id
```

示例调用：

```
aws pcs list-compute-node-groups --cluster-identifier get-started-cfn
```

输出示例：

```
{
    "computeNodeGroups": [
        {
            "name": "compute-1",
            "id": "pcs_abc123abc1",
            "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567/
computenodegroup/pcs_abc123abc1",
            "clusterId": "pcs_abc1234567",
            "createdAt": "2025-04-01T20:19:25+00:00",
            "modifiedAt": "2025-04-01T20:19:25+00:00",
            "status": "ACTIVE"
        },
        {
```

```
            "name": "login",
            "id": "pcs_abc456abc7",
            "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567/
computenodegroup/pcs_abc456abc7",
            "clusterId": "pcs_abc1234567",
            "createdAt": "2025-04-01T20:19:31+00:00",
            "modifiedAt": "2025-04-01T20:19:31+00:00",
            "status": "ACTIVE"
        }
    ]
}
```

3. 使用 GetComputeNodeGroupAPI 操作获取计算节点组的更多详细信息。

```
aws pcs get-compute-node-group --cluster-identifier cluster-name-or-id --
compute-node-group-identifier compute-node-group-name-or-id
```

示例调用：

```
aws pcs get-compute-node-group --cluster-identifier get-started-cfn --compute-
node-group-identifier compute-1
```

输出示例：

```
{
    "computeNodeGroup": {
        "name": "compute-1",
        "id": "pcs_abc123abc1",
        "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567/
computenodegroup/pcs_abc123abc1",
        "clusterId": "pcs_abc1234567",
        "createdAt": "2025-04-01T20:19:25+00:00",
        "modifiedAt": "2025-04-01T20:19:25+00:00",
        "status": "ACTIVE",
        "amiId": "ami-0123456789abcdef0",
        "subnetIds": [
            "subnet-abc012345789abc12"
        ],
        "purchaseOption": "ONDEMAND",
        "customLaunchTemplate": {
            "id": "lt-012345abcdef01234",
            "version": "1"
```

```
        },
        "iamInstanceProfileArn": "arn:aws:iam::111122223333:instance-profile/
AWSPCS-get-started-cfn-us-east-1",
        "scalingConfiguration": {
            "minInstanceCount": 0,
            "maxInstanceCount": 4
        },
        "instanceConfigs": [
            {
                "instanceType": "c6i.xlarge"
            }
        ]
    }
}
```

在 AWS PCS 中查找计算节点组实例

每 AWS 个 PCS 计算节点组都可以使用共享配置启动 EC2 实例。您可以使用 EC2 标签在 AWS 管理控制台 或的计算节点组中查找实例 AWS CLI。

AWS 管理控制台

查找您的计算节点组实例

1. 打开 AWS PCS 控制台。
2. 选择 集群。
3. 选择计算节点组。
4. 查找您创建的登录节点组的 ID。
5. 导航到 EC2 控制台并选择实例。
6. 搜索带有以下标签的实例。*node-group-id*替换为计算节点组的 ID（不是名称）。

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

7. （可选）您可以在搜索字段中更改实例状态的值，以查找正在配置或最近终止的实例。
8. 在已标记的实例列表中查找每个实例的实例 ID 和 IP 地址。

AWS CLI

要查找您的节点组实例，请使用以下命令。在运行命令之前，请进行以下替换：

- *region-code*替换为您的 AWS 区域 集群的。示例：us-east-1
- *node-group-id*替换为计算节点组的 ID（不是名称）。要查找计算节点组的 ID，请参阅在 AWS PCS 中获取计算节点组的详细信息。
- running替换为其他实例状态（例如pending或）terminated以查找处于其他状态的 EC2 实例。

```
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}
```

该命令返回的输出类似于下方内容。null如果实例PublicIP位于私有子网中，则值为。

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

> **Note**
>
> 如果您希望describe-instances返回大量实例，则必须对多个页面使用选项。有关更多信息，请参阅DescribeInstances《亚马逊弹性计算云 API 参考》。

在 AWS PCS 上使用亚马逊 EC2 启动模板

在 Amazon EC2 中，启动模板可以存储一组首选项，这样您就不必在启动实例时单独指定它们。 AWS PCS 采用启动模板作为配置计算节点组的灵活方式。创建节点组时，您需要提供启动模板。 AWS PCS 从中创建派生的启动模板，其中包含转换，以帮助确保其与服务配合使用。

了解编写自定义启动模板时的选项和注意事项可以帮助您编写一个用于 AWS PCS 的模板。有关启动模板的更多信息，请参阅 Amazon EC2 用户指南中的通过启动模板启动实例。

主题

AWS PCS 中的启动模板概述

您可以在 EC2 启动模板中包含 30 多个参数，控制实例配置方式的许多方面。大多数都与 AWS PCS 完全兼容，但也有一些例外。

AWS PCS 将忽略 EC2 启动模板的以下参数，因为这些属性必须由服务直接管理：

- 实例 type/Specify 实例类型属性 (`InstanceRequirements`)- AWS PCS 不支持基于属性的实例选择。
- 实例类型 (`InstanceType`)-在创建节点组时指定实例类型。
- 高级 details/IAM 实例配置文件 (`IamInstanceProfile`)-您在创建或更新节点组时提供此信息。
- 高级 details/Disable API 终止 (`DisableApiTermination`) — AWS PCS 必须控制其启动的节点组实例的生命周期。
- 高级 details/Disable API stop (`DisableApiStop`) — AWS PCS 必须控制其启动的节点组实例的生命周期。
- 高级 details/Stop — 休眠行为 (`HibernationOptions`) — AWS PCS 不支持实例休眠。
- 高级 details/Elastic GPU (`ElasticGpuSpecifications`) — 亚马逊 Elastic Graphics 于 2024 年 1 月 8 日停产。

- 高级 details/Elastic 推理 (ElasticInferenceAccelerators) — Amazon Elastic Inference 不再向新客户开放。
- AAdvanced details/Specify CPU options/Threads每核 (ThreadsPerCore) — AWS PCS 将每个内核的线程数设置为 1。

这些参数具有支持与 AWS PCS 兼容的特殊要求：

- 用户数据 (UserData)-必须进行多部分编码。请参阅处理适用于 AWS PCS 的 Amazon EC2 用户数据。
- 应用程序和操作系统映像 (ImageId)-您可以将其包括在内。但是，如果您在创建或更新节点组时指定 AMI ID，它将覆盖启动模板中的值。您提供的 AMI 必须与 AWS PCS 兼容。有关更多信息，请参阅 “适用于 AWS PCS 的亚马逊机器映像 (AMIs)。
- 网络 settings/Firewall （ 安全组 ） (SecurityGroups)-无法在 AWS PCS 启动模板中设置安全组名称列表。除非您在启动模板中定义网络接口，否则您可以设置安全组列表 IDs (SecurityGroupIds)。然后，必须 IDs 为每个接口指定安全组。有关更多信息，请参阅 AWS PCS 中的安全组。
- 网络 settings/Advanced 网络配置 (NetworkInterfaces) — 如果您使用带有单个网卡的 EC2 实例，并且不需要任何专门的联网配置，则 AWS PCS 可以为您配置实例联网。要配置多个网卡或在您的实例上启用弹性结构适配器，请使用NetworkInterfaces。每个网络接口 IDs 下都必须有一个安全组列表Groups。有关更多信息，请参阅 AWS PCS 中有多个网络接口。
- 高级详细信息/容量预留 (CapacityReservationSpecification)-可以设置，但在使用 PCS CapacityReservationId 时不能引用具体内容。 AWS 但是，您可以引用容量预留组，该组包含一个或多个容量预留。有关更多信息，请参阅 AWS PCS 中的容量预留。

创建基本的启动模板

您可以使用 AWS 管理控制台 或创建启动模板 AWS CLI。

AWS 管理控制台

创建启动模板

1. 打开 Amazon EC2 控制台，然后选择 “启动模板”。
2. 选择Create launch template（ 创建启动模板 ）。
3. 在 Launch 模板名称和描述下，为 La unch 模板名称输入一个唯一且独特的名称

4. 在 “密钥对名称” 的 “密钥对（登录）” 下，选择将用于登录由 AWS PCS 管理的 EC2 实例的 SSH 密钥对。您可以自由选择，但我们建议您这样做。
5. 在网络设置下，然后选择防火墙（安全组），选择要连接到网络接口的安全组。启动模板中的所有安全组都必须来自您的 AWS PCS 集群 VPC。至少选择：

   - 允许与 AWS PCS 集群通信的安全组
   - 允许 AWS PCS 启动的 EC2 实例之间进行通信的安全组
   - （可选）允许对交互式实例进行入站 SSH 访问的安全组
   - （可选）允许计算节点与 Internet 建立传出连接的安全组
   - （可选）允许访问网络资源（例如共享文件系统或数据库服务器）的安全组。
6. 您可以在亚马逊 EC2 控制台的 “启动模板” 下方访问您的新启动模板 ID。启动模板 ID 将包含表单`lt-0123456789abcdef01`。

建议采取下一步行动

- 使用新的启动模板创建或更新 AWS PCS 计算节点组。

AWS CLI

创建启动模板

使用以下命令创建启动模板。

- 在运行命令之前，进行以下替换：

  a. *region-code*替换为你正在使用 AWS PCS AWS 区域 的地方

  b. *my-launch-template-name*替换为模板的名称。它必须是 AWS 区域 您正在使用的唯一的。 AWS 账户

  c. *my-ssh-key-name*替换为首选 SSH 密钥的名称。

  d. 将*sg-ExampleID1*和替换为 IDs 允许您的 EC2 实例*sg-ExampleID2*与计划程序之间通信以及 EC2 实例之间通信的安全组。如果您只有一个安全组可以启用所有这些流量，则可以删除`sg-ExampleID2`其前面的逗号字符。您还可以添加更多安全组 IDs。您在启动模板中包含的所有安全组都必须来自您的 AWS PCS 集群 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 将输出类似于以下内容的文本。启动模板 ID 可在中找到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"
    }
}
```

建议采取下一步行动

- 使用新的启动模板创建或更新 AWS PCS 计算节点组。

处理适用于 AWS PCS 的 Amazon EC2 用户数据

您可以在实例启动时cloud-init运行的启动模板中提供 EC2 用户数据。内容类型的用户数据块在实例向 AWS PCS API 注册之前cloud-config运行，而内容类型的用户数据块在注册完成后但在 Slurm 守护程序启动之前text/x-shellscript运行。有关内容类型的更多信息，请参阅 cloud-init 文档。

我们的用户数据可以执行常见的配置场景，包括但不限于以下情况：

- 包括用户或群组
- 安装软件包
- 创建分区和文件系统
- 挂载网络文件系统

启动模板中的用户数据必须采用 MIME 多部分存档格式。这是因为您的用户数据与在节点组中配置节点所需的其他 AWS PCS 用户数据合并。您可以将多个用户数据块合并到一个 MIME 分段文件中。

MIME 分段文件包含以下组成部分：

- 内容类型和段边界声明：`Content-Type: multipart/mixed; boundary="==BOUNDARY=="`
- MIME 版本声明：`MIME-Version: 1.0`
- 一个或多个用户数据块，其包含以下组成部分：
  - 开口边界，表示用户数据块的开头：`--==BOUNDARY==`必须将此边界之前的行留空。
  - 区块的内容类型声明：`Content-Type: text/cloud-config; charset="us-ascii"`或`Content-Type: text/x-shellscript; charset="us-ascii"`。必须将内容类型声明之后的行留空。
  - 用户数据的内容，例如，Shell 命令或 `cloud-config` 指令的列表。
- 封闭边界，表示 MIME 分段文件的结尾：`--==BOUNDARY==--`必须将此闭合边界之前的行留空。

> **Note**
>
> 如果您在 Amazon EC2 控制台的启动模板中添加用户数据，则可以将其粘贴为纯文本。或者，您可以从文件上传它。如果您使用 AWS CLI 或 AWS SDK，则必须先对用户数据进行 base64 编码，并在调用时将该字符串作为`UserData`参数值提交 CreateLaunchTemplate，如此 JSON 文件所示。

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

示例

- 示例：从软件包存储库安装软件
- 示例：从 S3 存储桶运行脚本
- 示例：设置全局环境变量
- 在 AWS PCS 上使用网络文件系统
- 示例：使用 EFS 文件系统作为共享主目录

示例：从软件包存储库中安装适用于 AWS PCS 的软件

在启动模板"userData"中提供此脚本作为的值。有关更多信息，请参阅 处理适用于 AWS PCS 的 Amazon EC2 用户数据。

此脚本使用 cloud-config 在启动时在节点组实例上安装软件包。有关更多信息，请参阅 cloud-in it 文档中的用户数据格式。此示例安装curl和llvm。

> **Note**
>
> 您的实例必须能够连接到其配置的软件包存储库。

```
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==--
```

示例：从 S3 存储桶为 AWS PCS 运行其他脚本

在启动模板"userData"中提供此脚本作为的值。有关更多信息，请参阅 处理适用于 AWS PCS 的 Amazon EC2 用户数据。

以下用户数据脚本使用 cloud-config 从 S3 存储桶导入脚本，并在启动时在节点组实例上运行该脚本。有关更多信息，请参阅 cloud-in it 文档中的用户数据格式。

用您自己的详细信息替换以下值：

- *amzn-s3-demo-bucket*— 您的账户可以读取的 S3 存储桶的名称。
- *object-key*— 要导入的脚本的 S3 对象密钥。这包括脚本的名称及其在存储桶文件夹结构中的位置。例如 scripts/script.sh。有关更多信息，请参阅《亚马逊简单存储服务用户指南》中的使用文件夹在 Amazon S3 控制台中组织对象。

- *shell*— 用于运行脚本的 Linux 外壳，例如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://amzn-s3-demo-bucket/object-key /tmp/script.sh
- /usr/bin/shell /tmp/script.sh

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

节点组的 IAM 实例配置文件必须有权访问存储桶。以下 IAM 策略是上述用户数据脚本中存储桶的示例。

JSON

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

示例：为 AWS PCS 设置全局环境变量

在启动模板"userData"中提供此脚本作为的值。有关更多信息，请参阅 处理适用于 AWS PCS 的 Amazon EC2 用户数据。

以下示例用于/etc/profile.d在节点组实例上设置全局变量。

```
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==--
```

示例：使用 EFS 文件系统作为 AWS PCS 的共享主目录

在启动模板"userData"中提供此脚本作为的值。有关更多信息，请参阅 处理适用于 AWS PCS 的 Amazon EC2 用户数据。

此示例扩展了 EFS 装载的示例，在 AWS PCS 上使用网络文件系统以实现共享的主目录。/home 的内容会在装载 EFS 文件系统之前进行备份。然后在装载完成后将内容快速复制到共享存储器上。

用您自己的详细信息替换此脚本中的以下值：

- *`/mount-point-directory`*— 您要在其中装载 EFS 文件系统的实例上的路径。
- *`filesystem-id`*— EFS 文件系统的文件系统 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==--
```

示例：启用无密码 SSH

您可以在共享主目录示例的基础上构建，使用 SSH 密钥在集群实例之间实现 SSH 连接。对于使用共享主文件系统的每个用户，请运行类似于以下内容的脚本：

```
#!/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**
>
> 这些实例必须使用允许在群集节点之间进行 SSH 连接的安全组。

AWS PCS 中的容量预留

您可以使用按需 EC2 容量预留或 Amazon Capacity Blocks for ML 在特定可用区域中预留特定时间段的 Amazon EC2 容量，以确保在需要时有必要的计算容量可用。

按需容量预留 (ODCRs) 允许您在特定可用区内为 Amazon EC2 实例预留任意持续时间的计算容量。您可以随时创建和取消预订，无需长期承诺或预付款。 ODCRs 当您需要灵活的容量预留时，这是理想的选择，可以根据需求的变化进行修改。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的按需容量预留。

Amazon EC2 Capacity Blocks for ML 允许您最多提前 8 周预留基于 GPU 的加速计算实例以备将来使用。您可以预留 1-64 个实例的区块，持续时间从 1 天到 6 个月不等。Capacity Blocks 非常适合需要在特定时间保证访问 GPU 容量的机器学习工作负载。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的机器学习容量块。

主题

- ODCRs 与 AWS PCS 一起使用
- 将 Amazon EC2 容量块用于带有 AWS PCS 的机器学习

ODCRs 与 AWS PCS 一起使用

您可以选择 AWS PCS 如何使用您的预留实例。如果您创建了开放的 ODCR，则您的账户中 AWS PCS 或其他进程启动的任何匹配实例都将计入预留中。对于定向 ODCR，只有使用特定预留 ID 启动的实例才会计入预留。对于时间敏感型工作负载，定向 ODCRs 更为常见。

您可以将 AWS PCS 计算节点组配置为使用目标 ODCR，方法是将其添加到启动模板中。以下是执行此操作的步骤：

1. 使用 Amazon EC2 创建容量预留用户指南创建有针对性的按需容量预留 (ODCR)。
2. 将 ODCR 与启动模板关联。有两种方法可以做到这一点：
    a. 直接关联 ODCR：直接在启动模板中引用 ODCR ID。这种方法提供了严格的容量控制并且不支持实例回填（如果计算节点组请求的实例数超过 ODCR 中的可用实例，则不会启动其他实例）。
    b. 容量预留组关联：将 ODCR 添加到容量预留组并在启动模板中引用该组。此方法支持实例回填，允许 AWS PCS 在超过预留容量时启动其他按需实例。
3. 创建或更新 AWS PCS 计算节点组以使用启动模板。有关更多信息，请参阅 AWS PCS 计算节点组用户指南。
    - 将计算节点组`purchaseOption`的设置为`ONDEMAND`。

示例：预留并使用带有目标 ODCR 的 hpc6a.48xlarge 实例

此示例命令为 32 个 hpc6a.48xlarge 实例创建目标 ODCR。要在置放群组中启动预留实例，请`--placement-group-arn`向命令中添加。您可以使用`--end-date`和定义停止日期`--end-date-type`，否则预留将一直持续到手动终止。

```
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
```

此命令的结果将是新 ODCR 的 ARN。ODCR ID 可以从 "arn:aws:ec2:us-east-2:123456789012:capacity-reservation/ODCR-ID" ARN 中检索，也可以使用 Amazon EC2 检索。 DescribeCapacityReservations

直接 ODCR 关联：将 ODCR ID 添加到启动模板中。以下是引用 ODCR ID 的启动模板示例。

```
{
  "CapacityReservationSpecification": {
    "CapacityReservationTarget": {
      "CapacityReservationId": "cr-1234567890abcdef1"
    }
  }
}
```

容量预留组关联：创建容量预留组并将该组添加到启动模板中。以下命令创建名为的容量预留组EXAMPLE-CR-GROUP。

```
aws resource-groups create-group \
    --name EXAMPLE-CR-GROUP \
    --configuration \
        '{"Type": "AWS::EC2::CapacityReservationPool"}' \
        '{"Type": "AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-
resource-types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

以下命令将 ODCR 添加到容量预留组。

```
aws resource-groups group-resources --group EXAMPLE-CR-GROUP \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-reservation/
cr-1234567890abcdef1
```

创建 ODCR 并将其添加到容量预留组后，现在可以通过将其添加到启动模板来将其连接到 AWS PCS 计算节点组。以下是引用容量预留组的启动模板示例。

```
{
```

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

最后，创建或更新 AWS PCS 计算节点组以使用 hpc6a.48xlarge 实例，并使用引用 ODCR 的启动模板。对于静态节点组，将最小和最大实例数设置为预留的大小 (32)。对于动态节点组，请将最小实例数设置为 0，将最大实例设置为所需的实例大小。

此示例是为一个计算节点组配置的单个 ODCR 的简单实现。但是， AWS PCS 支持许多其他设计。例如，您可以将一个大型 ODCR 或容量预留组细分为多个计算节点组。或者 ODCRs ，您可以使用其他 AWS 账户创建并与您共享的账户。

有关更多信息，请参阅 Amazon 弹性计算云用户指南中的机器学习按需容量预留和容量块。

将 Amazon EC2 容量块用于带有 AWS PCS 的机器学习

适用于 ML 的 Amazon EC2 容量块是一种 Amazon EC2 购买选项，允许您提前付费在特定日期和时间范围内预留基于 GPU 的加速计算实例，以支持短期工作负载。在容量块内运行的实例会自动放置在 Amazon EC2 中 UltraClusters，以实现低延迟、PB 级的无阻塞联网。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的机器学习容量块。

您可以使用启动模板让 AWS PCS 在为计算节点组启动实例时使用容量块。

> **Note**
>
> AWS 自 Slurm 版本 24.05 以来，PCS 引入了对容量块的支持。

限制

- AWS PCS 仅支持 P6-B300、P6-B200、p5en、p5e、P5 和 p4d 实例系列的容量块。
- 一次只能将一个计算节点组与 1 个容量块相关联。
- 您无法将计算节点组与组合了多个容量块的容量预留组相关联。
- 容量块必须处于`scheduled`或`active`状态才能与 AWS PCS 配合使用。您不能在其他州使用容量块，例如`payment-failed`。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的查看容量块。

- 有关 P6 和 P5 实例类型，请参阅相关的 AWS 文档：P6 实例的软件要求，使用多张网卡最大化 Amazon EC2 实例的网络带宽

容量块到期

容量区块仅限于特定的日期和时间范围。当容量块过期时：

- 与该容量块关联的计算节点组继续存在并保持与相同队列的关联。
- 根据您的 Slurm 设置，计算节点组中的所有实例都将终止，活动作业可能会失败。
- AWS PCS 无法在计算节点组中启动新实例。
- 在将另一个计算节点组附加到队列或您更新计算节点组以使用指定新容量块的新启动模板之前，所有已排队或新提交的作业都将保持待处理状态。

将 AWS PCS 计算节点组配置为使用容量块

将容量块与计算节点组关联

1. 为 AWS PCS 创建一个 Amazon EC2 启动模板，指定您的容量区块。有关为 AWS PCS 创建启动模板的更多信息，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板。

   您的启动模板必须包括：

   - `MarketType`的值`InstanceMarketOptions`必须设置为`capacity-block`。
   - `CapacityReservationSpecification`具有有效值的 A `CapacityReservationId`
   - 与您购买的容量块的实例类型相匹配的有效`InstanceType`值。
2. 创建使用启动模板的计算节点组。有关更多信息，请参阅 在 AWS PCS 中创建计算节点组。您也可以更新现有的计算节点组以使用启动模板。有关更多信息，请参阅 更新 AWS PCS 计算节点组。

   创建或更新计算节点组时：

   - 您用于创建或更新计算节点组的 IAM 身份必须具有以下权限：

     ```
     ec2:DescribeCapacityReservations
     ```

     有关更多信息，请参阅 AWS PCS 的最低权限。
   - 容量块必须处于`scheduled`或`active`状态。

- 将计算节点组purchaseOption的设置为CAPACITY_BLOCK。
- 计算节点组maxInstanceCount的不能超过容量块的大小。
- 计算节点组的可用区必须与计算节点组的 1 个子网可用区相匹配。

> **Important**
>
> 更新计算节点组时，您无法更改其实例类型。您只能使用与计算节点组具有相同实例类型的容量块。如果要将容量块与不同的实例类型一起使用，则必须创建一个新的计算节点组。

有关在 AWS PCS 中使用容量块的常见问题

我刚刚支付了容量块的费用，并立即尝试将其与 AWS PCS 一起使用，但计算节点组创建失败了。发生了什么？

您的容量块可能未处于scheduled或active状态。在容量块为scheduled或之后再试一次active。

我在 AWS PCS 中使用的是容量块，我在扩展到期之前就购买了扩展。如何继续在 AWS PCS 中使用它？

您无需执行任何操作即可继续使用 AWS PCS 中的容量块。延期付款成功后，容量区块的结束日期会更新。只要您的容量块未过期，计算节点组就会继续运行。如果您的延期付款失败，则您的容量块将保留active，计算节点组将一直运行，直到容量块在其原始结束日期到期。

如果我的容量块过期，我的排队和正在运行的作业会怎样？

在将另一个计算节点组附加到队列或使用新的容量块更新计算节点组之前，在容量块到期之前未启动的排队作业仍处于待处理状态。您仍然可以向队列提交作业。你的 Slurm 设置会影响活跃的作业。默认情况下，活动作业会自动重新排队，但可能会出现错误或失败。

我的容量限制已过期。我应该做点什么吗？

你不必做任何事情。您可以在 Amazon EC2 控制台上查看 EC2 容量预留的状态。当容量块过期时，与该容量块关联的计算节点组将继续存在并处理相同的队列。计算节点组没有任何用于运行作业的实例。您可以删除计算节点组或将其与队列取消关联，以防止用户提交无法运行的作业。

我想在我的 AWS PCS 计算节点组中使用新的容量块。我应该怎么办？

我们建议您创建一个新的计算节点组以使用新的容量块。有关更多信息，请参阅 [将 AWS PCS 计算节点组配置为使用容量块]。

如何在集群和服务之间共享 1 个容量块？

您可以将容量块拆分到多个集群和服务中。例如，要将容量块拆分为 64 个`p5.48xlarge`实例，其中 20 个节点在 PCS-Cluster-1 上，16 个节点在 PCS-Cluster-2 上，其余节点用于其他服务，请将 PCS-Cluster-1 的和设置为 20，PCS-Cluster-2 的`minInstanceCount`和`maxInstanceCount`设置为 16。

我能否将 1 个以上的容量块或组合容量与 1 个计算节点组一起使用？

不是。 只有 1 个容量块可以与单个计算节点组相关联。 AWS PCS 不支持将多个容量块组合在一起的容量预留组。

我怎么知道我的容量块何时开始或过期？

与 AWS PCS 无关，Amazon EC2 会在容量块预留开始 EventBridge 时发送一个`Capacity Block Reservation DeliveredCapacity Block Reservation Expiration Warning`事件，在容量块预留到期前 40 分钟发送一个事件。有关更多信息，请参阅 Amazon 弹性计算云用户指南 EventBridge中的使用监控容量块。

Slurm 如何跟踪我的容量区块的状态？

你可以跑`sinfo`去了解 AWS PCS 是如何使用容量块的。在以下示例输出中，队列与从`active`容量块运行 4 个实例的计算节点组相关联。节点处于 `idle` Slurm 状态（可供使用但尚未分配给任何作业）。

```
$ sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
fanout up infinite 4 idle node-fanout-[1-4]
```

如果节点处于`maint`状态，则可以运行`scontrol show res`查看有关控制此状态的 Slurm 预留的详细信息。在以下示例输出中，Capacity Block `scheduled` 的开始日期为 future。

```
$ scontrol show res

ReservationName=node-fanout-scheduled StartTime=2025-10-14T13:09:17
 EndTime=2025-10-14T13:11:17 Duration=00:02:00
   Nodes=node-fanout-[1-4] NodeCnt=4 CoreCnt=16 Features=(null) PartitionName=(null)
 Flags=MAINT,SPEC_NODES
   TRES=cpu=16

   Users=root Groups=(null) Accounts=(null) Licenses=(null) State=ACTIVE
 BurstBuffer=(null)
```

```
   MaxStartDelay=(null)

   Comment=node-fanout Scheduled
```

如何判断我在启动容量时遇到的错误是否是因为我的容量块已共享？

在 Amazon EC2 控制台中查看容量预留，以了解容量块中有多少实例处于活动预置状态。检查每个实例的标签，找出使用它的服务或集群。例如，所有 PCS 实例都有 AWS AWS PCS 标签`aws:pcs:cluster-id = pcs_l0mizqyk5o | aws:pcs:compute-node-group-id = pcs_ic7onkmfqk`，例如用于指示该实例属于哪些集群和计算节点组。然后，您可以检查容量块是否已达到最大容量。

您可以`scontrol show nodes`用来检查 AWS PCS 集群中的容量块节点是否正在触发`ReservationCapacityExceeded`：

```
[root@ip-172-16-10-54 ~]# scontrol show nodes test-node-8-gamma-cb-2
NodeName=test-8-gamma-cb-2 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=8 CPUTot=8 CPULoad=0.00
   AvailableFeatures=test-8-gamma-cb,gpu
   ActiveFeatures=test-8-gamma-cb,gpu
   Gres=gpu:H100:1
   NodeAddr=test-8-gamma-cb-2 NodeHostName=test-8-gamma-cb-2
   RealMemory=249036 AllocMem=0 FreeMem=N/A Sockets=8 Boards=1
   State=IDLE+CLOUD+POWERING_DOWN ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A
 MCS_label=N/A
   Partitions=my-q
   BootTime=None SlurmdStartTime=None
   LastBusyTime=Unknown ResumeAfterTime=None
   CfgTRES=cpu=8,mem=249036M,billing=8
   AllocTRES=
   CurrentWatts=0 AveWatts=0
   Reason=Failed to launch backing instance (Error Code:
 ReservationCapacityExceeded) [root@2025-08-28T15:15:33]
```

当多个计算节点组连接到同一个队列时，如何强制作业在 Capacity Block 支持的实例上运行？

你可以使用 Slurm 的功能和约束将任务锁定到一组特定的节点。我们建议您不要为每个计算节点组设置 Slurm 权重，因为这仅适用于未处于该`maint`状态的节点。

有用的启动模板参数

本节介绍一些可能对 AWS PCS 有广泛用处的启动模板参数。

开启详细 CloudWatch 监控

您可以使用启动模板参数在较短的时间间隔内启用 CloudWatch 指标收集。

AWS 管理控制台

在用于创建或编辑启动模板的控制台页面上，可以在高级详细信息部分下找到此选项。将 “详细 CloudWatch 监控” 设置为 “启用”。

YAML

```
Monitoring:
    Enabled: True
```

JSON

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

有关更多信息，请参阅适用于 Linux 实例的 Amazon 弹性计算云用户指南中的启用或关闭实例的详细监控。

实例元数据服务版本 2 (IMDS v2)

将 IMDS v2 与 EC2 实例配合使用可显著增强安全性，并有助于降低与在环境中 AWS 访问实例元数据相关的潜在风险。

AWS 管理控制台

在用于创建或编辑启动模板的控制台页面上，可以在高级详细信息部分下找到此选项。将可访问的元数据设置为已启用，将元数据版本设置为仅限 V2（需要令牌），将元数据响应跳跃限制设置为 4。

YAML

```
MetadataOptions:
  HttpEndpoint: enabled
```

```
  HttpTokens: required
  HttpPutResponseHopLimit: 4
```

JSON

```
{
    "MetadataOptions": {
        "HttpEndpoint": "enabled",
        "HttpPutResponseHopLimit": 4,
        "HttpTokens": "required"
    }
}
```

.

AWS PCS 队列

AWS PCS 队列是对调度器对工作队列的本机实现的轻量级抽象。就 Slurm 而言，AWS PCS 队列等同于 Slurm 分区。

用户将作业提交到他们所在的队列，直到可以安排作业在一个或多个计算节点组提供的节点上运行。一个 AWS PCS 集群可以有多个任务队列。例如，您可以创建一个使用亚马逊 EC2 按需实例处理高优先级任务的队列和另一个使用亚马逊 EC2 竞价型实例处理低优先级任务的队列。

主题

在 AWS PCS 中创建队列

本主题概述了可用选项，并介绍了在 AWS PCS 中创建队列时应考虑的事项。

> **Note**
>
> 您可以在队列上配置自定义 Slurm 设置，以实现特定分区的调度策略和资源管理。有关更多信息，请参阅 在 PCS 中配置自定义 Slurm 设置 AWS。

先决条件

- AWS PCS 集群-队列只能与特定 AWS PCS 集群关联创建。
- 一个或多 AWS 个 PCS 计算节点组-队列必须与至少一个 AWS PCS 计算节点组相关联。

在 AWS PCS 中创建队列

您可以使用 AWS 管理控制台 或创建队列 AWS CLI。

AWS 管理控制台

使用控制台创建队列

1. 打开 AWS PCS 控制台。
2. 为队列选择集群。导航到 “队列”，然后选择 “创建队列”。
3. 在队列配置部分中，提供以下值：
   a. 队列名称-队列的名称。名称只能包含字母数字字符（区分大小写）和连字符。它必须以字母字符开头，长度不能超过 25 个字符。该名称在集群中必须是唯一的。
   b. 计算节点组-选择 1 个或多个计算节点组来为该队列提供服务。一个计算节点组可以与多个队列关联。
4. （可选）在其他调度程序设置部分，您可以添加参数名称和值对来配置其他 Slurm 设置。有关支持的参数的完整列表，请参阅PCS 队列的自定义 Slurm 设置 AWS。
5. （可选）在 “标签” 下，将所有标签添加到您的 AWS PCS 队列中
6. 选择创建队列。当 AWS PCS 创建队列时，状态字段将显示正在创建。创建队列可能需要几分钟。

建议采取下一步行动

- 向您的新队列提交任务。

AWS CLI

使用创建队列 AWS CLI

使用以下命令创建队列。进行以下替换：

1. *region-code*替换为集群的 AWS 区域。例如 `us-east-1`。
2. *my-queue*替换为队列的名称。名称只能包含字母数字字符（区分大小写）和连字符。它必须以字母字符开头，长度不能超过 25 个字符。该名称在集群中必须是唯一的。
3. *my-cluster*替换为集群的名称或 ID。
4. *compute-node-group-id*替换为为队列提供服务的计算节点组的 ID。例如 `pcs_abcdef12345`。

> **Note**
>
> 创建队列时，必须提供计算节点组的 ID，而不是其名称。

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

Example— 使用自定义 Slurm 设置创建队列

```
aws pcs create-queue --region region-code \
    --queue-name my-queue \
    --cluster-identifier my-cluster \
    --compute-node-group-configurations \
    computeNodeGroupId=compute-node-group-id \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Default,parameterValue=YES}]'
```

有关更多信息，请参阅 PCS 队列的自定义 Slurm 设置 AWS。

创建队列可能需要几分钟。您可以使用以下命令查询队列的状态。在队列状态达到之前，您将无法向队列提交作业ACTIVE。

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

建议采取下一步行动

- 向新队列提交作业

更新 AWS PCS 队列

本主题概述了可用选项，并介绍了更新 AWS PCS 队列时应考虑的事项。有关 Slurm 自定义设置的信息，请参阅。PCS 队列的自定义 Slurm 设置 AWS

更新 AWS PCS 队列时的注意事项

队列更新不会影响正在运行的作业，但是在队列更新期间，集群可能无法接受新作业。

更新 AWS PCS 队列

您可以使用 AWS 管理控制台 或 AWS CLI 更新队列。

AWS 管理控制台

更新队列

1. 在以下位置打开 AWS PCS 控制台 `https://console.aws.amazon.com/pcs/home#/clusters`
2. 选择要在其中更新队列的集群。
3. 导航至 “队列”，转至要更新的队列，然后选择 “编辑”。
4. 在队列配置部分，更新以下任意值：
   - 节点组-添加或移除计算节点组与队列的关联。
   - 其他调度程序设置-添加、修改或删除队列的自定义 Slurm 设置。有关更多信息，请参阅 PCS 队列的自定义 Slurm 设置 AWS。
   - 标签-为队列添加或移除标签。
5. 选择更新。应用更改时，“状态” 字段将显示 “正在更新”。

> **Important**
> 队列更新可能需要几分钟。

AWS CLI

更新队列

1. 使用以下命令更新您的队列。在运行命令之前，进行以下替换：
   a. *region-code*替换为 AWS 区域 要在其中创建集群的。
   b. *my-queue*替换为队列`computeNodeGroupId`的名称或。
   c. *my-cluster*替换为集群`clusterId`的名称或。

d. 要更改计算节点组关联，请提供更新后的列表`--compute-node-group-configurations`。

   - 例如，要添加第二个计算节点组，请执行`computeNodeGroupExampleID2`以下操作：

```
--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
```

Example— 使用自定义 Slurm 设置更新队列

```
aws pcs update-queue --region region-code \
    --queue-identifier my-queue \
    --cluster-identifier my-cluster \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Default,parameterValue=YES}]'
```

有关更多信息，请参阅 PCS 队列的自定义 Slurm 设置 AWS。

2. 更新队列可能需要几分钟。您可以使用以下命令查询队列的状态。在队列状态达到之前，您将无法向队列提交作业`ACTIVE`。

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

建议的后续步骤

- 向更新后的队列提交任务。

在 AWS PCS 中删除队列

本主题概述了如何在 AWS PCS 中删除队列。

删除队列时的注意事项

- 如果队列中有作业在运行，则在删除队列时，调度程序将终止这些作业。队列中的待处理任务将被取消。可以考虑等待队列中的任务完成， stop/cancel 或者使用调度程序的本机命令（`scancel`例如 Slurm）手动完成任务。

删除队列

您可以使用 AWS 管理控制台 或 AWS CLI 删除队列。

AWS 管理控制台

删除队列

1. 打开 AWS PCS 控制台。
2. 选择队列的集群。
3. 导航到 “队列”，然后选择要删除的队列。
4. 选择删除。
5. 将显示 “状态” 字段`Deleting`。可能需要几分钟的时间才能完成。

> **Note**
> 您可以使用调度程序原生的命令来确认队列已删除。例如，对于 Slur `squeue m` 使用`sinfo`或。

AWS CLI

删除队列

- 使用以下命令删除队列，并使用以下替换命令：
  - *`region-code`*替换为 AWS 区域 您的集群所在的。

- *my-queue*替换为队列的名称或 ID。
- *my-cluster*替换为集群的名称或 ID。

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

删除队列可能需要几分钟。

> **Note**
>
> 您可以使用调度程序原生的命令来确认队列已删除。例如，对于 Slur squeue m 使用sinfo或。

AWS PCS 登录节点

AWS PCS 集群通常需要至少 1 个登录节点来支持交互式访问和任务管理。实现这一目标的一种方法是使用为登录节点功能配置静态 AWS PCS 计算节点组。您也可以将独立的 EC2 实例配置为登录节点。

主题

使用 AWS PCS 计算节点组提供登录节点

本主题概述了建议的配置选项，并介绍了在使用 AWS PCS 计算节点组为集群提供持久的交互式访问时应考虑的事项。

为登录节点创建 AWS PCS 计算节点组

从操作上讲，这与创建常规计算节点组没有太大区别。但是，可以做出一些关键的配置选择：

- 为计算节点组中至少一个 EC2 实例设置静态扩展配置。
- 选择按需购买选项，以避免回收您的实例。
- 为计算节点组选择一个信息性名称，例如登录。
- 如果您希望登录节点实例可在您的 VPC 之外访问，请考虑使用公有子网。
- 如果您打算允许 SSH 访问，则启动模板需要有一个安全组，用于向您选择的 IP 地址公开 SSH 端口。
- IAM 实例配置文件应仅具有您希望最终用户拥有的 AWS 权限。有关详细信息，请参阅 AWS 并行计算服务的 IAM 实例配置文件。
- 考虑允许 AWS Systems Manager 会话管理器来管理您的登录实例。
- 考虑仅限管理员用户访问实例 AWS 证书
- 选择比普通计算节点组更便宜的实例类型，因为登录节点将持续运行。
- 使用与其他计算节点组相同的（或衍生的）AMI，以帮助确保所有实例都安装了相同的软件。有关自定义的更多信息 AMIs，请参阅 适用于 AWS PCS 的亚马逊机器映像 (AMIs)

- 在登录节点上配置与计算实例相同的网络文件系统（Amazon FSx EFS、Amazon for Lustre 等）的挂载。有关更多信息，请参阅 在 AWS PCS 上使用网络文件系统。

访问您的登录节点

当您的新计算节点组变为 ACTIVE 状态后，您可以找到它创建的 EC2 实例并登录到这些实例中。有关更多信息，请参阅 在 AWS PCS 中查找计算节点组实例。

更新登录节点的 AWS PCS 计算节点组

您可以使用更新登录节点组 UpdateComputeNodeGroup。作为节点组更新过程的一部分，将替换正在运行的实例。请注意，这将中断实例上所有活跃的用户会话或进程。正在运行或排队的 Slurm 作业不会受到影响。有关更多信息，请参阅 更新 AWS PCS 计算节点组。

您也可以编辑计算节点组使用的启动模板。必须使用 UpdateComputeNodeGroup 将更新的启动模板应用于计算节点组。在计算节点组中启动的新 EC2 实例使用更新的启动模板。有关更多信息，请参阅 在 AWS PCS 上使用亚马逊 EC2 启动模板。

删除登录节点的 AWS PCS 计算节点组

您可以使用 AWS PCS 中的删除计算节点组机制更新登录节点组。作为删除节点组的一部分，正在运行的实例将被终止。请注意，这将中断实例上所有活跃的用户会话或进程。正在运行或排队的 Slurm 作业不会受到影响。有关更多信息，请参阅 删除 AWS PCS 中的计算节点组。

使用独立实例作为 AWS PCS 登录节点

您可以设置独立的 EC2 实例来与 AWS PCS 集群的 Slurm 调度器进行交互。这对于创建登录节点、工作站或专用工作流管理主机非常有用，这些主机可以与 AWS PCS 集群配合使用，但在 AWS PCS 管理之外运行。为此，每个独立实例必须：

1. 安装兼容的 Slurm 软件版本。
2. 能够连接到 AWS PCS 集群的 Slurmctld 终端节点。
3. 使用 AWS PCS 集群的终端节点和密钥正确配置 Slurm Auth 和 Cred Kiosk Daemon (`sackd`)。有关更多信息，请参阅 Slurm 文档中的 sackd。

本教程帮助您配置连接到 AWS PCS 集群的独立实例。

目录

步骤 1-检索目标 AWS PCS 集群的地址和密码

使用以下命令检索有关目标 AWS PCS 集群的详细信息。 AWS CLI 在运行命令之前，进行以下替换：

- *region-code*替换为目标 AWS 区域 集群的运行位置。
- *cluster-ident*替换为目标集群的名称或标识符

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

该命令将返回类似于此示例的输出。

```
{
    "cluster": {
        "name": "get-started",
        "id": "pcs_123456abcd",
        "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_123456abcd",
        "status": "ACTIVE",
        "createdAt": "2024-12-17T21:03:52+00:00",
        "modifiedAt": "2024-12-17T21:03:52+00:00",
        "scheduler": {
            "type": "SLURM",
            "version": "25.05"
        },
        "size": "SMALL",
        "slurmConfiguration": {
            "authKey": {
                "secretArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:pcs!
slurm-secret-pcs_123456abcd-a12ABC",
                "secretVersion": "ef232370-d3e7-434c-9a87-ec35c1987f75"
            }
```

```
        },
        "networking": {
            "subnetIds": [
                "subnet-0123456789abcdef0"
            ],
            "securityGroupIds": [
                "sg-0123456789abcdef0"
            ]
        },
        "endpoints": [
            {
                "type": "SLURMCTLD",
                "privateIpAddress": "10.3.149.220",
                "port": "6817"
            }
        ]
    }
}
```

在此示例中，集群 Slurm 控制器端点的 IP 地址为，`10.3.149.220`并且正在端口上运行。`6817``secretArn`将在后面的步骤中使用来检索集群密钥。IP 地址和端口将在后续步骤中用于配置`sackd`服务。

步骤 2-启动 EC2 实例

启动 EC2 实例

1. 打开 Amazon EC2 控制台。
2. 在导航窗格中，请选择 Instances（实例），然后选择 Launch Instances（启动实例）以打开新的启动实例向导。
3. （可选）在名称和标签部分中，提供实例的名称，例如`PCS-LoginNode`。名称作为资源标签（`Name=PCS-LoginNode`）分配给实例。
4. 在“应用程序和操作系统映像”部分，为 AWS PCS 支持的操作系统选择一个 AMI。有关更多信息，请参阅 支持的操作系统。
5. 在实例类型部分中，选择支持的实例类型。有关更多信息，请参阅 支持的实例类型。
6. 在密钥对部分，选择要用于实例的 SSH 密钥对。
7. 在网络设置 部分中：
   - 选择编辑。

i. 选择您的 PC AWS S 集群的 VPC。

ii. 对于防火墙（安全组），请选择选择现有安全组。

A. 选择一个允许在实例和目标 AWS PCS 集群的 Slurm 控制器之间进行流量的安全组。有关更多信息，请参阅 安全组要求和注意事项。

B. （可选）选择允许对您的实例进行入站 SSH 访问的安全组。

8. 在“存储”部分，根据需要配置存储卷。确保配置足够的空间来安装应用程序和库，以启用您的用例。
9. 在“高级”下，选择允许访问集群密钥的 IAM 角色。有关更多信息，请参阅 获取 Slurm 集群的秘密。
10. 在摘要窗格中，选择启动实例。

步骤 3-在实例上安装 Slurm

当实例启动并变为活动状态时，请使用您的首选机制连接到该实例。使用提供的 Slurm 安装程序将 Slurm 安装 AWS 到实例上。有关更多信息，请参阅 Slurm 安装程序。

下载 Slurm 安装程序，将其解压缩，然后使用`installer.sh`脚本安装 Slurm。有关更多信息，请参阅 第 3 步 — 安装 Slurm。

步骤 4-检索和存储集群密钥

这些说明需要 AWS CLI. 有关更多信息，请参阅版本 2 AWS Command Line Interface 用户指南 AWS CLI中的安装或更新到最新版本的。

使用以下命令存储集群密钥。

- 为 Slurm 创建配置目录。

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

- 检索、解码和存储集群密钥。在运行此命令之前，请*region-code*替换为运行目标集群的区域，并*secret-arn*替换为步骤 1 中`secretArn`检索到的值。

```
aws secretsmanager get-secret-value \
 --region region-code \
 --secret-id 'secret-arn' \
```

```
--version-stage AWSCURRENT \
--query 'SecretString' \
--output text | base64 -d | sudo tee /etc/slurm/slurm.key
```

> **Warning**
>
> 在多用户环境中，任何有权访问实例的用户只要能够访问实例元数据服务 (IMDS)，都可能能够获取集群密钥。反过来，这可能允许他们冒充其他用户。考虑仅限根用户或管理员用户访问 IMDS。或者，可以考虑使用另一种不依赖实例配置文件来获取和配置密钥的机制。

- 设置 Slurm 密钥文件的所有权和权限。

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

> **Note**
>
> Slurm 密钥必须归运行sackd服务的用户和群组所有。

步骤 5-配置与 AWS PCS 集群的连接

要建立与 AWS PCS 集群的连接，请按照以下步骤sackd作为系统服务启动。

> **Note**
>
> 如果您使用 Slurm 25.05 或更高版本，则可以使用脚本将登录节点设置为连接到多个集群。有关更多信息，请参阅 将独立登录节点连接到 AWS PCS 中的多个集群。

1. 使用以下命令为sackd服务设置环境文件。在运行命令之前，请将*ip-address*和*port*替换为步骤 1 中从端点检索到的值。

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

2. 创建用于管理sackd流程的systemd服务文件。

   ```
   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-25.05/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. 设置sackd服务文件的所有权。

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

4. 启用该sackd服务。

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

5. 启动 sackd 服务。

```
sudo systemctl start sackd
```

步骤 6-（可选）测试连接

确认sackd服务正在运行。示例输出如下。如果有错误，它们通常会出现在这里。

```
[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 2024-12-17 16:34:55 UTC; 8s ago
 Main PID: 9985 (sackd)
   CGroup: /system.slice/sackd.service
           ##9985 /opt/aws/pcs/scheduler/slurm-25.05/sbin/sackd --systemd --conf-
server=10.3.149.220:6817

Dec 17 16:34:55 ip-10-3-27-112.ec2.internal systemd[1]: Starting Slurm auth and cred
 kiosk daemon...
Dec 17 16:34:55 ip-10-3-27-112.ec2.internal systemd[1]: Started Slurm auth and cred
 kiosk daemon.
Dec 17 16:34:55 ip-10-3-27-112.ec2.internal sackd[9985]: sackd: running
```

使用 Slurm 客户端命令（例如sinfo和）确认与集群的连接是否正常运行。squeue以下是的输出示例sinfo。

```
[root@ip-10-3-27-112 ~]# /opt/aws/pcs/scheduler/slurm-25.05/bin/sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
all up infinite 4 idle~ compute-[1-4]
```

您还应该能够提交工作。例如，类似于此示例的命令将在集群中的 1 个节点上启动交互式作业。

```
/opt/aws/pcs/scheduler/slurm-25.05/bin/srun --nodes=1 -p all --pty bash -i
```

将独立登录节点连接到 AWS PCS 中的多个集群

该pcs-multi-cluster-login-configure.sh脚本提供了一种在单个独立登录节点上配置多个 Slurm sackd 守护程序的自动方法。它使登录节点能够与多个集群通信。该脚本可自动执行以下操作：

- 使用 AWS PCS API 操作来获取集群信息
- 提示输入 base64 编码的 Slurm 身份验证密钥
- 使用集群身份验证密钥创建 Slurm JWKS 文件
- 使用集群终端节点和端口配置sackd服务
- 为特定于集群sackd的守护程序创建systemd服务文件
- 为集群环境设置生成激活脚本
- 启用和启动sackd服务

> **Note**
>
> 此脚本需要 Slurm 版本 25.05 或更高版本。

Slurm 必须已经安装在实例上（相当于手动过程中的步骤 3）。该实例必须能够到达目标集群的终端节点。在手动配置过程中，该脚本执行的操作与步骤 4 和步骤 5 的操作相同。它会自动获取集群信息、配置服务、创建必要的`sackd`和`systemd`服务文件并创建激活脚本，用户可以使用该脚本配置其 shell 环境以进行集群交互。

主题

AWS PCS 多集群登录节点配置脚本的先决条件

系统要求

- `systemd`支持的 Linux 操作系统
- 系统配置的 root 权限

必需的命令和软件包

- `bash`— 外壳解释器（版本 4.0+）
- `curl`— 用于 AWS IMDS v2 元数据检索
- `jq`— 用于解析 AWS API 响应的 JSON 处理器
- `aws`— AWS CLI v2 用于运行 AWS PCS API 操作和访问 Secrets Manager
- `systemctl`— `systemd` 服务管理
- `find`— 文件系统搜索实用程序
- `grep`— 文本模式匹配
- `sed`— 用于文本操作的流编辑器
- `sort`— 文本排序工具
- `tail`— 显示文件的最后几行

- `mkdir`— 目录创建
- `chmod`— 更改文件权限
- `chown`— 更改文件所有权
- `ldconfig`— 动态链接器配置

AWS 要求

- 运行 Slurm 版本 25.05 或更高版本的 AWS PCS 集群
- AWS 配置的证书（通过 IAM 角色、证书文件或环境变量）
- 以下各项的权限：
  - `pcs:GetCluster`
  - `secretsmanager:GetSecretValue`（如果您使用备用密钥）

系统用户和群组

- `slurm`用户和组必须存在于系统中

Slurm 安装

- Slurm 必须与 AWS PCS Slurm 安装程序包安装在相同的位置：

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

AWS PCS 多集群登录节点配置脚本代码

将以下源代码保存到具有以下名称的文件中：

```
pcs-multi-cluster-login-configure.sh
```

脚本源代码

```
#!/bin/bash
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

# AWS PCS Multi-Cluster Standalone Login Node Configuration Script
```

```
#
# This script configures AWS Parallel Computing Service (PCS) multi-cluster stand alone
 login nodes
# by setting up the Slurm authentication and credential kiosk daemon (sackd)
# for connecting to remote PCS clusters.
#
# Prerequisites:
# - AWS CLI configured with appropriate permissions
# - Slurm version 25.05 or later
# - Root privileges for system configuration
# - Network connectivity to AWS PCS endpoints


set -eo pipefail

# Function to display usage
usage() {
    echo "Usage: $0 --cluster-identifier <cluster-identifier> [--endpoint-url
 <endpoint-url>]"
    echo "       $0 -h|--help"
}

# Function to display help
help() {
    echo "AWS PCS Multi-Cluster Standalone Login Node Configuration Script"
    echo "================================================"
    echo
    echo "This script configures multi-cluster standalone login node for AWS Parallel
 Computing Service (PCS)"
    echo "by setting up the Slurm authentication and credential kiosk daemon (sackd)."
    echo
    usage
    echo
    echo "Options:"
    echo "  --cluster-identifier <id>    AWS PCS cluster identifier (required)"
    echo "  --endpoint-url <url>         Custom PCS endpoint URL (optional)"
    echo "  -h, --help                   Show this help message"
    echo
    echo "Examples:"
    echo "  $0 --cluster-identifier my-pcs-cluster"
    echo
    echo "Note: This script requires root privileges and Slurm version 25.05 or later."
}
```

```
# Function to retrieve authentication key
get_auth_key() {
    if [ "$ALTERNATE_SECRET_RETRIEVAL" = "true" ]; then
        echo "Retrieving authentication key from AWS Secrets Manager..." >&2
        local auth_key_arn=$(echo "$CLUSTER_INFO" | jq -r
 '.cluster.slurmConfiguration.authKey.secretArn')
        local auth_key_version=$(echo "$CLUSTER_INFO" | jq -r
 '.cluster.slurmConfiguration.authKey.secretVersion')

        if [ "$auth_key_arn" = "null" ] || [ "$auth_key_version" = "null" ]; then
            echo "Error: Auth key information not found in cluster configuration" >&2
            exit 1
        fi

        if ! aws secretsmanager get-secret-value --secret-id "$auth_key_arn" --version-
id "$auth_key_version" --query SecretString --output text --region "$REGION" 2>/dev/
null; then
            echo "Error: Failed to retrieve auth key from Secrets Manager" >&2
            exit 1
        fi
    else
        echo "Please enter the base64-encoded Slurm authentication key:" >&2
        echo -n "Base64 of the Slurm secret key: " >&2
        local key
        read -rs key
        echo >&2
        echo "$key"
    fi
}

# Function to get next available SACKD port
get_next_sackd_port() {
    local exclude_file="$1"
    local port=6918
    local used_ports=()

    # Get all currently used SACKD ports into an array
    while IFS= read -r line; do
        used_ports+=("$line")
    done < <(find /etc/sysconfig -name "sackd-pcs-*" ! -path "$exclude_file" \
             -exec grep SACKD_PORT= '{}' ';' 2>/dev/null | \
             sed 's/.*SACKD_PORT=//' | sort -n)

    # Loop through used ports to find first available port
```

```
    for used_port in "${used_ports[@]}"; do
        if [ "$port" -lt "$used_port" ]; then
            break
        elif [ "$port" -eq "$used_port" ]; then
            ((port++))
        fi
    done

    echo "$port"
}

# Function to configure cluster
configure_cluster() {
    mkdir -p /etc/slurm
    SLURM_JWKS_FILE="/etc/slurm/slurm-${CLUSTER_NAME}.jwks"
    echo '{"keys":
[{"alg":"HS256","kty":"oct","kid":"key-'"${CLUSTER_ID}"'","k":"'"${BASE64_SLURM_KEY}"'"}]}'
 | jq -c '.' > "${SLURM_JWKS_FILE}"

    chmod 0600 "$SLURM_JWKS_FILE"
    chown slurm:slurm "$SLURM_JWKS_FILE"

    SLURM_INSTALL_PATH="/opt/aws/pcs/scheduler/slurm-${SLURM_VERSION}"

    SACKD_RUNTIME_DIRECTORY="/run/slurm-${CLUSTER_NAME}"
    mkdir -p "${SACKD_RUNTIME_DIRECTORY}"
    chown slurm:slurm "${SACKD_RUNTIME_DIRECTORY}"

    mkdir -p /etc/sysconfig
    SACKD_SERVICE_NAME="sackd-pcs-${CLUSTER_NAME}"
    SACKD_SERVICE_ENV="/etc/sysconfig/${SACKD_SERVICE_NAME}"
    SACKD_PORT=$(get_next_sackd_port "$SACKD_SERVICE_ENV")
    cat > "${SACKD_SERVICE_ENV}" << EOF
SACKD_OPTIONS='--conf-server=$ENDPOINTS'
SLURM_SACK_JWKS='$SLURM_JWKS_FILE'
RUNTIME_DIRECTORY='$SACKD_RUNTIME_DIRECTORY'
SACKD_PORT=$SACKD_PORT
EOF

    SACKD_SERVICE_PATH="/etc/systemd/system/${SACKD_SERVICE_NAME}.service"

    cat << EOF > "$SACKD_SERVICE_PATH"
[Unit]
Description=Slurm auth and cred kiosk daemon
```

```
After=network-online.target remote-fs.target
Wants=network-online.target
ConditionPathExists=${SACKD_SERVICE_ENV}

[Service]
Type=notify
EnvironmentFile=${SACKD_SERVICE_ENV}
User=slurm
Group=slurm
RuntimeDirectory=slurm-${CLUSTER_NAME}
RuntimeDirectoryMode=0755
ExecStart=${SLURM_INSTALL_PATH}/sbin/sackd --systemd \$SACKD_OPTIONS
ExecReload=/bin/kill -HUP \$MAINPID
KillMode=process
LimitNOFILE=131072
LimitMEMLOCK=infinity
LimitSTACK=infinity

[Install]
WantedBy=multi-user.target
EOF

    chown root:root "$SACKD_SERVICE_PATH"
    chmod 0644 "$SACKD_SERVICE_PATH"
    systemctl daemon-reload && systemctl enable "$SACKD_SERVICE_NAME"
    systemctl restart "$SACKD_SERVICE_NAME"

    ACTIVATE_SCRIPT="activate-pcs-${CLUSTER_NAME}"
    cat > "$ACTIVATE_SCRIPT" << EOF
# Activate script for Slurm cluster ${CLUSTER_NAME}

# Add Slurm paths
export PATH="${SLURM_INSTALL_PATH}/bin:\$PATH"
export MANPATH="${SLURM_INSTALL_PATH}/share/man:\$MANPATH"
export LD_LIBRARY_PATH="${SLURM_INSTALL_PATH}/lib:\$LD_LIBRARY_PATH"
ldconfig

# Set Slurm configuration
export SLURM_CONF="/run/slurm-${CLUSTER_NAME}/conf/slurm.conf"
export PCS_CLUSTER_NAME="${CLUSTER_NAME}"
export PCS_CLUSTER_IDENTIFIER="${CLUSTER_IDENTIFIER}"
export PCS_CLUSTER_ID="${CLUSTER_ID}"

echo "Activated PCS cluster environment: ${CLUSTER_NAME}"
```

```
# Deactivate function
function deactivate-pcs-${CLUSTER_NAME}() {
    export PATH="\$(echo "\$PATH" | sed -e "s|${SLURM_INSTALL_PATH}/bin:||g" -e "s|:
${SLURM_INSTALL_PATH}/bin||g" -e "s|^${SLURM_INSTALL_PATH}/bin\$||")"
    export MANPATH="\$(echo "\$MANPATH" | sed -e "s|${SLURM_INSTALL_PATH}/share/man:||
g" -e "s|:${SLURM_INSTALL_PATH}/share/man||g" -e "s|^${SLURM_INSTALL_PATH}/share/man\
$||")"
    export LD_LIBRARY_PATH="\$(echo "\$LD_LIBRARY_PATH" | sed -e "s|
${SLURM_INSTALL_PATH}/lib:||g" -e "s|:${SLURM_INSTALL_PATH}/lib||g" -e "s|^
${SLURM_INSTALL_PATH}/lib\$||")"
    unset SLURM_CONF
    unset PCS_CLUSTER_NAME
    unset PCS_CLUSTER_IDENTIFIER
    unset PCS_CLUSTER_ID
    unset -f deactivate-pcs-${CLUSTER_NAME}
    ldconfig
    echo "Deactivated PCS cluster environment: ${CLUSTER_NAME}"
}

export -f deactivate-pcs-${CLUSTER_NAME}

EOF
}

# Main function
main() {
    # Parse arguments
    CLUSTER_IDENTIFIER=""
    PCS_ENDPOINT_URL=""

    while [ "$1" != "" ]; do
        case $1 in
            --cluster-identifier)
                shift
                CLUSTER_IDENTIFIER="$1"
                ;;
            --endpoint-url)
                shift
                PCS_ENDPOINT_URL="--endpoint-url $1"
                ;;
            -h|--help)
                help
                exit 0
```

```
                ;;
            *)
                echo "Invalid argument: $1" >&2
                usage >&2
                exit 1
                ;;
        esac
        shift
    done

    # Validate required arguments
    if [ -z "$CLUSTER_IDENTIFIER" ]; then
        echo "Error: --cluster-identifier is required" >&2
        usage >&2
        exit 1
    fi

    # Validate running as root
    if [ "$EUID" -ne 0 ]; then
        echo "Error: This script must be run as root" >&2
        exit 1
    fi

    # Validate required commands are available
    for cmd in aws jq curl; do
        if ! command -v "$cmd" &> /dev/null; then
            echo "Error: Required command '$cmd' not found" >&2
            exit 1
        fi
    done

    # Get the region name from IMDS v2 with error handling (try IPv6 first, fallback to
 IPv4)
    echo "Retrieving AWS region from instance metadata..."
    # Try IPv6 IMDS endpoint first (fd00:ec2::254) with fast timeout (1s connect, 2s
 total)
    # If IPv6 fails, fallback to IPv4 IMDS endpoint (169.254.169.254)
    IMDS_ENDPOINT="http://[fd00:ec2::254]"
    if ! TOKEN=$(curl -s -X PUT "${IMDS_ENDPOINT}/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600" --connect-timeout 1 --max-time 2 2>/dev/null); then
        IMDS_ENDPOINT="http://169.254.169.254"
        if ! TOKEN=$(curl -s -X PUT "${IMDS_ENDPOINT}/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600" --max-time 5); then
```

```
            echo "Error: Failed to retrieve IMDS token. Ensure this script is running
 on an EC2 instance." >&2
            exit 1
        fi
    fi

    if ! REGION=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" "${IMDS_ENDPOINT}/
latest/dynamic/instance-identity/document" --max-time 5 | jq -r '.region'); then
        echo "Error: Failed to retrieve AWS region from instance metadata" >&2
        exit 1
    fi

    echo "Detected AWS region: $REGION"

    # Retrieve cluster information from AWS PCS
    echo "Retrieving cluster information for: $CLUSTER_IDENTIFIER"
    # shellcheck disable=SC2086
    if ! CLUSTER_INFO=$(aws pcs get-cluster --region "$REGION" --cluster-identifier
 "$CLUSTER_IDENTIFIER" $PCS_ENDPOINT_URL 2>/dev/null); then
        echo "Error: Failed to retrieve cluster information. Check cluster identifier
 and AWS permissions." >&2
        exit 1
    fi

    CLUSTER_ID=$(echo "$CLUSTER_INFO" | jq -r '.cluster.id')
    CLUSTER_NAME="$(echo "$CLUSTER_INFO" | jq -r '.cluster.name')"
    SLURM_VERSION=$(echo "$CLUSTER_INFO" | jq -r '.cluster.scheduler.version')
    SLURM_VERSION=${SLURM_VERSION#Slurm_}

    # Check if Slurm version is >= 25.05
    # shellcheck disable=SC2072
    if [[ "$SLURM_VERSION" < "25.05" ]]; then
        echo "Error: This script requires Slurm version 25.05 or later. Found version:
 $SLURM_VERSION" >&2
        exit 1
    fi

    ENDPOINTS=$(echo "$CLUSTER_INFO" | jq -r '.cluster.endpoints[] | select(.type
 == "SLURMCTLD") | (if .privateIpAddress != "" then .privateIpAddress else "["
 + .ipv6Address + "]" end) + ":" + .port' | tr '\n' ',' | sed 's/,$//')

    # Get BASE64_SLURM_KEY
    BASE64_SLURM_KEY=$(get_auth_key)
```

```
    if [ -z "$BASE64_SLURM_KEY" ]; then
        echo "Error: base64 Slurm key cannot be empty" >&2
        exit 1
    fi

    configure_cluster

    # Final configuration summary
    echo "========================================"
    echo "Configuration completed successfully!"
    echo "========================================"
    echo "Cluster Name: $CLUSTER_NAME"
    echo "Cluster ID: $CLUSTER_ID"
    echo "Slurm Version: $SLURM_VERSION"
    echo "Service Name: $SACKD_SERVICE_NAME"
    echo "SACKD Port: $SACKD_PORT"
    echo
    echo "To activate this cluster environment, run:"
    echo "  source ./$ACTIVATE_SCRIPT"
    echo
    echo "To deactivate this cluster environment, run:"
    echo "  deactivate-pcs-${CLUSTER_NAME}"
    echo
    echo "To check service status:"
    echo "  systemctl status $SACKD_SERVICE_NAME"
    echo
    echo "To view service logs:"
    echo "  journalctl -u $SACKD_SERVICE_NAME -f"
}

# Exit if being sourced for testing
[[ "${BASH_SOURCE[0]}" != "${0}" ]] && return

# Execute main function
main "$@"
```

使用 AWS PCS 多集群登录节点配置脚本

运行脚本

运行配置脚本

1. 将脚本内容保存在名为：的文件中

```
pcs-multi-cluster-login-configure.sh
```

2. 使其可执行：

```
chmod +x pcs-multi-cluster-login-configure.sh
```

3. 运行脚本：

```
./pcs-multi-cluster-login-configure.sh --cluster-identifier cluster-name
```

集群交互环境

成功配置后，该脚本将在当前目录中生成特定于群集的激活脚本。脚本有名字activate-pcs-*cluster-name*。激活脚本配置必要的环境变量和路径以与目标集群进行交互。

激活群集环境

- 使用source命令运行激活脚本

```
source ./activate-pcs-cluster-name
```

Example

```
# Activate cluster environment for cluster 'my-cluster'
source ./activate-pcs-my-cluster

# Now you can use Slurm commands
sinfo
squeue
sbatch my-job.sh
```

激活脚本的作用

- 将SLURM_CONF环境变量设置为指向群集的配置。
- 更新PATH以包含集群的 Slurm 二进制文件。
- 配置其他必要的 Slurm 环境变量 (MANPATH,)。LD_LIBRARY_PATH
- 设置 AWS PCS 集群标识变量。

- 实现与目标 AWS PCS 集群的无缝交互。

停用群集环境

- 运行停用命令。

```
deactivate-pcs-cluster-name
```

Example

```
# After activating a cluster
source ./activate-pcs-my-cluster

# Work with the cluster
sinfo

# Deactivate when done
deactivate-pcs-my-cluster
```

停用命令的作用

- 恢复原始PATH环境变量。
- 取消设置特定于集群的 Slurm 环境变量。
- 将 shell 环境恢复到其激活前的状态。

> **Note**
>
> 激活是特定于会话的，并且必须源自您要与集群交互的 shell 会话。

AWS PCS 联网

您的 AWS PCS 集群是在亚马逊 VPC 中创建的。本章包括以下有关集群调度器和节点网络的主题。

除了选择用于启动实例的子网外，您还必须使用 EC2 启动模板为 AWS PCS 计算节点组配置网络。有关启动模板的更多信息，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板。

主题

AWS PCS VPC 和子网要求和注意事项

创建 AWS PCS 集群时，您需要将 VPC 指定为该 VPC 中的子网。本主题概述了您在集群中使用的 VPC 和子网的 AWS PCS 特定要求和注意事项。如果您没有可用于 PC AWS S 的 VPC，则可以使用 AWS提供的 CloudFormation 模板创建 VPC。有关更多信息 VPCs，请参阅 Amazon VPC 用户指南中的虚拟私有云 (VPC)。

VPC 要求和注意事项

在创建集群时，您指定的 VPC 必须满足以下要求和注意事项：

- VPC 必须有足够数量的 IP 地址可用于您要创建的集群、任何节点和其他集群资源。有关更多信息，请参阅 Amazon VPC 用户指南中的您的 VPCs 和子网的 IP 地址。
- 如果您的集群使用 IPv6：
  - 将 IPv6 CIDR 块与您的 VPC 关联。有关更多信息，请参阅 Amazon VPC 用户指南中的创建 VPC。

> **Important**
>
> 尽管您可以同时使用 IPv4 和配置您的 VPC IPv6，但您只能为集群选择 1 种网络类型。

- 为您的子网启用自动分配 IPv6 地址。
- 有关更多信息，请参阅:
  - IPv6 on AWS
  - 了解 AWS 上的 IPv6 寻址并设计可扩展的寻址计划
- VPC 必须具有 DNS 主机名和 DNS 解析支持。否则，节点无法注册客户集群。有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 的 DNS 属性。
- VPC 可能需要使用的 VPC 终端节点 AWS PrivateLink 才能联系 AWS PCS API。有关更多信息，请参阅 Amazon VPC 用户指南 AWS PrivateLink中的使用将您的 VPC 连接到服务。

> **Important**
>
> AWS PCS 不支持具有专用实例租期的 VPC。您用于 AWS PCS 的 VPC 必须使用`default`实例租期。您可以更改现有 VPC 的实例租期。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的更改 VPC 的实例租期。

子网要求和注意事项

创建 Slurm 集群时， AWS PCS 会在您指定的子网中创建一个弹性网络接口 (ENI)。此网络接口支持调度器控制器和客户 VPC 之间的通信。网络接口还使 Slurm 能够与您账户中部署的组件进行通信。您只能在创建集群时为集群指定子网。

集群的子网要求

您在创建集群时指定的子网必须满足以下要求：

- 子网必须至少有 1 个 IP 地址才能供 AWS PCS 使用。
- 如果您的集群使用 IPv6，则集群中的所有子网都必须使用 IPv6。

> **Important**
>
> 如果子网仅配置为使用，则使用 AWS IPv6 PCS 示例 AMIs 和多个网络接口配置的计算节点组目前将无法运行。改用双栈子网（IPv4 和 IPv6）或 IPv4仅使用双栈子网。有关更多信息，请参阅 在 AWS PCS 上使用示例亚马逊系统映像 (AMIs)。

- 子网不能位于 AWS Outposts AWS Wavelength、或 AWS 本地区域中。
- 子网可以是公有子网或私有子网。如果可能，我们建议您指定私有子网。公有子网是带有路由表的子网，其中包含通往互联网网关的路由；私有子网是带有路由表的子网，其中不包括通往互联网网关的路由。

节点的子网要求

您可以将节点和其他集群资源部署到创建 AWS PCS 集群时指定的子网，也可以部署到同一 VPC 中的其他子网。

将节点和群集资源部署到的任何子网都必须满足以下要求：

- 您必须确保子网有足够的可用的 IP 地址来部署所有节点和群集资源。
- 如果您的集群使用 IPv4 并且您计划将节点部署到公有子网，则该子网必须自动分配 IPv4 公有地址。

> **Note**
>
> 公有子网中的实例必须使用具有允许来自公有 IP 地址的流量的入站规则的安全组。除非您有特定的源地址限制，否则这意味着 IPv4 源地址为 0.0.0.0/0 或 IPv6 源地址为:: /0。

- 如果您部署节点的子网是私有子网，并且其路由表不包括通往网络地址转换 (NAT) 设备的路由 (IPv4)，请使用 AWS PrivateLink 向客户 VPC 添加 VPC 终端节点。节点联系的所有 AWS 服务都需要 VPC 终端节点。 AWS PCS 唯一需要的端点是允许节点调用 `RegisterComputeNodeGroupInstance` API 操作。有关更多信息，请参阅 AWS PCS API 参考RegisterComputeNodeGroupInstance中的。
- 公有或私有子网状态不会影响 AWS PCS；所需的端点必须可以访问。

为您的 PC AWS S 集群创建 VPC

您可以在 AWS 并行计算服务 (PCS) 中为集群创建亚马逊虚拟私有云 (Amazon V AWS PC)。

使用 Amazon VPC 将 VPC 资源启动到您定义的虚拟网络中。此虚拟网络与您在自己的数据中心中运行的传统网络极为相似。但是，它带有使用 Amazon Web Services 的可扩展基础设施的优势。我们建议您在部署生产 VPC 集群之前全面了解 Amazon VPC 服务。有关更多信息，请参阅什么是 Amazon VPC？ 在作者视觉模式下。 亚马逊 VPC 用户指南。

PCS 集群、节点和支持资源（例如文件系统和目录服务）部署在您的 Amazon VPC 中。如果您想将现有 Amazon VPC 与 PCS 配合使用，则必须满足中所述的要求AWS PCS VPC 和子网要求和注意事项。本主题介绍如何使用 AWS提供的 CloudFormation 模板创建满足 PCS 要求的 VPC。部署模板后，您可以查看该模板创建的资源，以确切了解它创建了哪些资源以及这些资源的配置。

先决条件

要创建适用于 PCS 的亚马逊 VPC，您必须拥有必要的 IAM 权限才能创建亚马逊 VPC 资源。这些资源包括子网 VPCs、安全组、路由表和路由，以及 Internet 和 NAT 网关。有关更多信息，请参阅 Amazon VPC 用户指南中的使用公有子网创建 VPC。要查看 Amazon EC2 的完整列表，请参阅《服务授权参考》中的 Amazon EC2 的操作、资源和条件密钥。

创建亚马逊 VPC

通过复制并粘贴您将在 AWS 区域 哪里使用 PCS 的相应网址来创建 VPC。您也可以下载 CloudFormation 模板并自己将其上传到CloudFormation 控制台。

- 美国东部（弗吉尼亚北部）(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
```

- 美国东部（俄亥俄）(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
```

- 美国西部（俄勒冈）(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
```

- 亚太地区（孟买）(ap-south-1)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-south-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
```

- 亚太地区（新加坡）(ap-southeast-1)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-southeast-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
```

- 亚太地区（悉尼）(ap-southeast-2)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-southeast-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
```

- 亚太地区（东京）(ap-northeast-1)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-northeast-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
```

- 欧洲（法兰克福）（eu-central-1）

```
https://console.aws.amazon.com/cloudformation/home?region=eu-central-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
```

- 欧洲（爱尔兰）(eu-west-1)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-west-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
```

- 欧洲（伦敦）(eu-west-2)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-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
```

- 欧洲（巴黎）（eu-west-3）

```
https://console.aws.amazon.com/cloudformation/home?region=eu-west-3#/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
```

- 欧洲（米兰）(eu-south-1)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-south-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
```

- 欧洲（斯德哥尔摩）（eu-north-1）

```
https://console.aws.amazon.com/cloudformation/home?region=eu-north-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
```

- AWS GovCloud (美国东部) (us-gov-east-1)

```
https://console.aws.amazon.com/cloudformation/home?region=us-gov-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
```

- AWS GovCloud (美国西部) (us-gov-west-1)

```
https://console.aws.amazon.com/cloudformation/home?region=us-gov-west-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
```

- 仅限模板

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

为 PCS 创建亚马逊 VPC

1. 在CloudFormation 控制台中打开模板。

   > **Note**
   >
   > 它们已在模板中预先填充，因此您只需将其保留为默认值即可。

2. 在 “提供堆栈名称” 下，然后输入 “堆栈名称” `hpc-networking`。
3. 在参数下，输入以下详细信息：

a. 然后 CidrBlock，在 VPC 下输入 `10.3.0.0/16`
b. 在子网 A 下：
    i. 然后输入 CidrPublicSubnetA `10.3.0.0/20`
    ii. 然后输入 CidrPrivateSubnetA `10.3.128.0/20`
c. 在子网 B 下：
    i. 然后输入 CidrPublicSubnetB `10.3.16.0/20`
    ii. 然后输入 CidrPrivateSubnetA `10.3.144.0/20`
d. 在子网 C 下：
    i. 对于 ProvisionSubnetsC，选择`True`。

> **Note**
> 如果您在可用区少于三个的区域中创建 VPC，则如果设置为，则此选项将被忽略`True`。

    ii. 然后输入 CidrPublicSubnetB `10.3.32.0/20`
    iii. 然后输入 CidrPrivateSubnetA `10.3.160.0/20`
4. 在“能力”下，选中“我确认 AWS CloudFormation 可能会创建 IAM 资源”复选框。

监控 CloudFormation 堆栈的状态。当它到达时`CREATE_COMPLETE`，VPC 资源已准备就绪，可供您使用。

> **Note**
> 要查看 CloudFormation 模板创建的所有资源，请打开CloudFormation 控制台。选择 `hpc-networking` 堆栈，然后选择 Resources（资源）选项卡。

AWS PCS 中的安全组

Amazon EC2 中的安全组充当虚拟防火墙，用于控制实例的入站和出站流量。使用 AWS PCS 计算节点组的启动模板向其实例添加或移除安全组。如果您的启动模板不包含任何网络接口，`SecurityGroupIds`请使用提供安全组列表。如果您的启动模板定义了网络接口，则必须使

用Groups参数为每个网络接口分配安全组。有关启动模板的更多信息，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板。

> **Note**
>
> 对启动模板中安全组配置的更改仅影响在更新计算节点组后启动的新实例。

安全组要求和注意事项

AWS PCS 在创建集群时指定的子网中创建跨账户弹性网络接口 (ENI)。这为在由 AWS管理的账户中运行的 HPC 调度程序提供了与 PCS 启动的 EC2 实例通信的路径。 AWS 您必须为该 ENI 提供一个安全组，允许调度程序 ENI 和您的集群 EC2 实例之间进行双向通信。

实现这一目标的一种直接方法是创建一个允许的自引用安全组，允许该组所有成员之间的所有端口上的 TCP/IP 流量。您可以将其连接到集群和节点组 EC2 实例。

许可安全组配置示例

IPv4

规则类型	协议	端口	来源	目标位置
入站	全部	全部	自身	
出站	全部	全部		0.0.0.0/0
出站	全部	全部		自身

IPv6

规则类型	协议	端口	来源	目标位置
入站	全部	全部	自身	
出站	全部	All		::/0
出站	全部	全部		自身

这些规则允许所有流量在 Slurm 控制器和节点之间自由流动，允许所有出站流量到达任何目的地，并启用 EFA 流量。

限制性安全组配置示例

您还可以限制集群与其计算节点之间的开放端口。对于 Slurm 调度程序，连接到集群的安全组必须允许以下端口：

- 6817 — 启用`slurmctld`从 EC2 实例到的入站连接
- 6818 — 启用从 EC2 实例`slurmctld`到`slurmd`运行的出站连接

连接到您的计算节点的安全组必须允许以下端口：

- 6817 — 启用`slurmctld`从 EC2 实例到的出站连接。
- 6818 — 启用与节点组实例之间的入`slurmd`站`slurmctld`和`slurmd`出站连接
- 60001—63000 — 要支持的节点组实例之间的入站和出站连接 `srun`
- 节点组实例之间的 EFA 流量。有关更多信息，请参阅 Linux 实例用户指南中的准备启用 EFA 的安全组
- 您的工作负载所需的任何其他节点间流量

AWS PCS 中有多个网络接口

某些 EC2 实例有多张网卡。这使他们能够提供更高的网络性能，包括超过 100 Gbps 的带宽能力和改进的数据包处理。有关带有多个网卡的实例的更多信息，请参阅 Amazon 弹性计算云用户指南中的弹性网络接口。

通过在 PCS 计算节点组的 EC2 启动模板中添加网络接口，为 AWS PCS 计算节点组中的实例配置其他网卡。以下是启用两个网卡的启动模板示例，例如可以在`hpc7a.96xlarge`实例上找到。请注意以下详细信息：

- 每个网络接口的子网必须与您在配置将使用启动模板的 AWS PCS 计算节点组时选择的子网相同。
- 通过将设置为`0`来建立主网络设备，用于进行例行网络通信（例如 SSH 和 HTTPS 流量）。`DeviceIndex`其他网络接口`DeviceIndex`有`1`。只能有一个主网络接口，所有其他接口都是辅助接口。
- 所有网络接口都必须具有唯一性`NetworkCardIndex`。建议的做法是按照启动模板中定义的顺序对它们进行编号。

- 每个网络接口的安全组都是使用设置的Groups。在此示例中，将入站 SSH 安全组 (sg-*SshSecurityGroupId*) 添加到主网络接口，以及启用集群内通信的安全组 () sg-*ClusterSecurityGroupId*。最后，在主接口和辅助接口上都添加了一个允许出站连接到 Internet (sg-*InternetOutboundSecurityGroupId*) 的安全组。

```
{
    "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"]
        }
    ]
}
```

AWS PCS 中 EC2 实例的置放群组

您可以使用置放群组来影响 EC2 实例的放置，以适应在其上运行的工作负载的需求。

置放群组类型

- 集群 — 将实例紧密地打包在可用区中，以优化低延迟通信。
- 分区-跨逻辑分区分布实例，以帮助最大限度地提高弹性。
- Sp read — 严格要求少量实例在不同的硬件上启动，这也有助于提高弹性。

有关更多信息，请参阅《亚马逊弹性计算云用户指南》中的 Amazon EC2 实例的置放群组。

我们建议您在将 AWS PCS 计算节点组配置为使用 Elastic Fabric Adapt er (EFA) Fabric Adapter 时加入集群置放群组。

创建可使用 EFA 的集群置放群组

1. 为计算节点组创建集群类型的置放群组。
   - 使用以下 AWS CLI 命令：

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

   - 您也可以使用 CloudFormation 模板来创建置放群组。有关更多信息，请参阅《AWS CloudFormation 用户指南》中的使用 CloudFormation模板。从以下 URL 下载模板并将其上传到CloudFormation 控制台。

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

2. 将置放群组包含在 AWS PCS 计算节点组的 EC2 启动模板中。

在 PCS 上使用弹性结构适配器 (EFA) Fabric Adapt AWS er

Elastic Fabric Adapter (EFA) Fabric Adapter 是一种高性能的高级网络互连，您可以将其连接到 AWS EC2 实例，以加速高性能计算 (HPC) 和机器学习应用程序。要使您的应用程序在 AWS PCS 集群上运行 EFA，需要将 AWS PCS 计算节点组实例配置为使用 EFA，如下所示。

> **Note**
>
> 在@@ 兼容 AWS PCS 的 AMI 上安装 EFA — AWS PCS 计算节点组中使用的 AMI 必须安装并加载 EFA 驱动程序。有关如何在安装了 EFA 软件的情况下构建自定义 AMI 的信息，请参阅适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

目录

- 为 EFA 创建或更新计算节点组
- （可选）测试 EFA
- （可选）使用 CloudFormation 模板创建启用 EFA 的启动模板

识别支持 EFA 的 EC2 实例

要使用 EFA，AWS PCS 计算组允许的所有实例类型都必须支持 EFA，并且必须具有相同数量的 vCPUs（GPUs 如果适用）。有关启用 EFA 的实例列表，请参阅《亚马逊弹性计算云用户指南》中的 Amazon EC2 上适用于 HPC 和 ML 工作负载的弹性结构适配器。您还可以使用 AWS CLI 查看支持 EFA 的实例类型列表。*region-code*替换为使用 AWS PCS AWS 区域 的地方，例如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
```

> **Note**
>
> 确定有多少网络接口可用 — 某些 EC2 实例有多个网卡。这允许他们有多个 EFAs。有关更多信息，请参阅 AWS PCS 中有多个网络接口。

创建支持 EFA 通信的安全组

AWS CLI

您可以使用以下 AWS CLI 命令创建支持 EFA 的安全组。该命令输出一个安全组 ID。进行以下替换：

- *region-code*— 指定在 AWS 区域 哪里使用 AWS PCS，例如us-east-1。
- *vpc-id*— 指定用于 PCS 的 VP AWS C 的 ID。
- *efa-group-name*— 提供您为安全组选择的名称。

```
aws ec2 create-security-group \
    --group-name efa-group-name \
```

```
    --description "Security group to enable EFA traffic" \
    --vpc-id vpc-id \
    --region region-code
```

使用以下命令附加入站和出站安全组规则。进行以下替换：

- *efa-secgroup-id*— 提供您刚刚创建的 EFA 安全组的 ID。

```
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

您可以使用 CloudFormation 模板创建支持 EFA 的安全组。从以下 URL 下载模板，然后将其上传到AWS CloudFormation 控制台。

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

在 AWS CloudFormation 控制台中打开模板后，输入以下选项。

- 在 “提供堆栈名称” 下
  - 在堆栈名称下，输入一个名称，例如`efa-sg-stack`。
- 在 “参数” 下
  - 在下方 SecurityGroupName，输入一个名称，例如`efa-sg`。
  - 在 VPC 下，选择您将使用 PCS 的 V AWS PC。

完成 CloudFormation 堆栈的创建并监控其状态。当它到达时`CREATE_COMPLETE`，EFA 安全组就可以使用了。

（可选）创建置放群组

我们建议您在集群置放群组中启动所有使用 EFA 的实例，以最大限度地缩短它们之间的物理距离。为计划使用 EFA 的每个计算节点组创建一个置放群组。AWS PCS 中 EC2 实例的置放群组要为您的计算节点组创建置放群组，请参阅。

创建或更新 EC2 启动模板

EFA 网络接口是在 AWS PCS 计算节点组的 EC2 启动模板中设置的。如果有多个网卡，则 EFAs 可以配置多个网卡。EFA 安全组和可选置放群组也包含在启动模板中。

以下是带有两张网卡的实例的启动模板示例，例如 hpc 7a.96xlarge。实例将在集群置放群组subnet-*SubnetID1*中启动pg-*PlacementGroupId1*。

必须专门向每个 EFA 接口添加安全组。每个 EFA 都需要启用 EFA 流量的安全组(sg-*EfaSecGroupId*)。其他安全组，尤其是处理常规流量（如 SSH 或 HTTPS）的安全组，只需连接到主网络接口（由 a DeviceIndex of 指定0）即可。定义网络接口的启动模板不支持使用SecurityGroupIds参数设置安全组，您必须在配置的每个网络接口Groups中为设置一个值。

```
{
    "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"]
        }
```

```
    ]
}
```

为 EFA 创建或更新计算节点组

您的 AWS PCS 计算节点组必须包含具有相同数量的 v CPUs、处理器架构和 EFA 支持的实例。将计算节点组配置为使用安装了 EFA 软件的 AMI，并使用配置启用 EFA 的网络接口的启动模板。

（可选）测试 EFA

您可以通过运行程序来演示计算节点组中两个节点之间启用 EFA 的通信，该`fi_pingpong`程序包含在 EFA 软件安装中。如果此测试成功，则很可能已正确配置 EFA。

要启动，您需要在计算节点组中运行两个实例。如果您的计算节点组使用静态容量，则应该已经有可用的实例。对于使用动态容量的计算节点组，您可以使用`salloc`命令启动两个节点。以下是一个集群的示例，该群集的动态节点组名为`hpc7g`与名为的队列相关联`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
```

使用查找两个已分配节点的 IP 地址`scontrol`。在以下示例中，地址分别是 for `hpc7g-1` 和 `10.3.140.69` f `10.3.132.211` or `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=25.05.5
   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=25.05.5
   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
```

使用 SSH（或 SSMhpc7g-1）连接到其中一个节点（在本例中为）。请注意，这是一个内部 IP 地址，因此如果您使用 SSH，则可能需要从其中一个登录节点进行连接。另请注意，需要通过计算节点组启动模板使用 SSH 密钥配置实例。

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

现在，fi_pingpong以服务器模式启动。

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

连接到第二个实例 (hpc7g-2)。

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

fi_pingpong在客户端模式下运行，连接到服务器hpc7g-1。您应该看到类似于以下示例的输出。

```
% /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
```

（可选）使用 CloudFormation 模板创建启用 EFA 的启动模板

由于设置 EFA 有多种依赖关系，因此提供了一个可用于配置计算节点组的 CloudFormation 模板。它支持最多带有四个网卡的实例。要详细了解带有多个网卡的实例，请参阅 Amazon 弹性计算云用户指南中的弹性网络接口。

从以下 URL 下载 CloudFormation 模板，然后将其上传到您使用 AWS PCS AWS 区域 的 CloudFormation 控制台。

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

在 CloudFormation 控制台中打开模板后，输入以下值。请注意，模板将提供一些默认参数值，您可以将其保留为默认值。

- 在 “提供堆栈名称” 下
  - 在堆栈名称下，输入描述性名称。我们建议使用您将为 AWS PCS 计算节点组选择的名称，例如*NODEGROUPNAME*-efa-lt。
- 在 “参数” 下
  - 在下方 NumberOfNetworkCards，选择您的节点组中实例中的网卡数量。
  - 在下方 VpcId，选择部署您的 AWS PCS 集群的 VPC。
  - 在下方 NodeGroupSubnetId，选择集群 VPC 中将在其中启动启用 EFA 的实例的子网。
  - 在下方 PlacementGroupName，将该字段留空，为该节点组创建新的集群置放群组。如果您要使用现有的置放群组，请在此处输入其名称。
  - 在下方 ClusterSecurityGroupId，选择您用于允许访问集群中的其他实例和 AWS PCS API 的安全组。许多客户从其集群 VPC 中选择默认安全组。
  - 在下方 SshSecurityGroupId，提供您用于允许对集群中节点进行入站 SSH 访问的安全组的 ID。
  - 对于 SshKeyName，选择用于访问集群中节点的 SSH 密钥对。

  - 对于 LaunchTemplateName，输入启动模板的描述性名称，例如*NODEGROUPNAME*`-efa-lt`。在您使用 AWS PCS 的 AWS 区域 位置 AWS 账户 中，该名称必须是唯一的。
- 能力不足
  - 选中 “我确认 AWS CloudFormation 可能会创建 IAM 资源” 复选框。

监控 CloudFormation 堆栈的状态。当它到达`CREATE_COMPLETE`时，启动模板就可以使用了。将其与 AWS PCS 计算节点组配合使用，如上所述为 EFA 创建或更新计算节点组。

在 AWS PCS 上使用网络文件系统

您可以将网络文件系统附加到在 AWS 并行计算服务 (AWS PCS) 计算节点组中启动的节点，以提供写入和访问数据和文件的永久位置。您可以使用 AWS 服务提供的文件系统，包括亚马逊弹性文件系统（亚马逊 EFS）、亚马逊 for Lustre、亚马逊 FSx for NetApp ONTAP、Amazon FSx for OpenZ FS 和亚马逊文件缓存。 FSx 您也可以使用自我管理的文件系统，例如 NFS 服务器。

本主题介绍在 AWS PCS 中使用网络文件系统的注意事项和示例。

使用网络文件系统的注意事项

各种文件系统的实现细节各不相同，但有一些常见的注意事项。

- 必须在实例上安装相关的文件系统软件。例如，要使用 Amazon f FSx or Lustre，则应提供相应的Lustre包装。这可以通过将其包含在计算节点组 AMI 中或使用在实例启动时运行的脚本来实现。
- 共享网络文件系统和计算节点组实例之间必须有网络路由。
- 共享网络文件系统和计算节点组实例的安全组规则必须允许连接到相关端口。
- 您必须在访问文件系统的资源之间保持一致的POSIX用户和组命名空间。否则，在 PCS 集群上运行的作业和交互式进程可能会遇到权限错误。
- 文件系统装载是使用EC2启动模板完成的。挂载网络文件系统时出现错误或超时可能会使实例无法运行作业。反过来，这可能会导致意想不到的成本。有关调试启动模板的更多信息，请参阅在 AWS PCS 上使用亚马逊 EC2 启动模板。

网络挂载示例

你可以使用 Amazon EFS、Amazon for Lustre、Amaz FSx on for NetApp ONTAP、Ama FSx zon for OpenZFS 和亚马逊 FSx 文件缓存创建文件系统。展开下面的相关部分，查看每个网络挂载的示例。

Amazon EFS

文件系统设置

创建 Amazon EFS 文件系统。确保它在每个可用区中都有一个挂载目标，您将在其中启动 PCS 计算节点组实例。还要确保每个挂载目标都与一个安全组相关联，该安全组允许从 PCS 计算节点组实例进行入站和出站访问。有关更多信息，请参阅 Amazon Elastic File System 用户指南中的挂载目标和安全组。

启动模板

将文件系统设置中的安全组添加到将用于计算节点组的启动模板中。

包括使用挂载 Amazon EFS 文件系统的`cloud-config`机制的用户数据。用您自己的详细信息替换此脚本中的以下值：

- *`mount-point-directory`*— 每个实例上要挂载 Amazon EFS 的路径
- *`filesystem-id`*— EFS 文件系统的文件系统 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==--
```

亚马逊 f FSx or Lustre

文件系统设置

在要使用 AWS PCS FSx 的 VPC 中创建一个 for Lustre 文件系统。为了最大限度地减少区域间传输，请在同一个可用区的子网中部署，您将在那里启动大多数 PCS 计算节点组实例。确保文件系统与允许从 PCS 计算节点组实例进行入站和出站访问的安全组相关联。有关安全组的更多信息，请参阅 Amazon for Lustre 用户指南中的使用 Amazon VPC FSx 进行文件系统访问控制。

启动模板

包括 FSx 用于装载 for `cloud-config` Lustre 文件系统的用户数据。用您自己的详细信息替换此脚本中的以下值：

- *`mount-point-directory`*— 你要为 Lustre 挂载 FSx 的实例上的路径

- *filesystem-id*— Lustre 文件系统的文件系统 ID FSx
- *mount-name*— 适用于 Lustre 文件 FSx 系统的装载名称
- *region-code*— for Lustre 文件系统的部署 AWS 区域 位置（必须与您的 AWS PCS 系统相同） FSx
- （可选）*latest*— for Lustre Lustre FSx 支持的任何版本的

```
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==--
```

FSx 适用于 NetApp ONTAP 的亚马逊

文件系统设置

在要使用 AWS PCS FSx 的 VPC 中创建适用于 NetApp ONTAP 的 Amazon 文件系统。为了最大限度地减少区域间传输，请在同一个可用区的子网中部署，您将在那里启动大多数 AWS PCS 计算节点组实例。确保文件系统与允许从 AWS PCS 计算节点组实例进行入站和出站访问的安全组相关联。有关安全组的更多信息，请参阅《适用于 ONTAP 的用户指南》中的 Amazon VPC 文件系统访问控制。FSx

启动模板

包括用于挂载 for ONTAP 文件系统的根卷 FSx 的用户数据。`cloud-config`用您自己的详细信息替换此脚本中的以下值：

- *mount-point-directory*— 您要在实例上挂载 for ONTAP 卷 FSx 的路径
- *svm-id*— 适用于 ONTAP 文件系统 FSx 的 SVM ID
- *filesystem-id*— 适用于 ONTAP 文件系统的文件系统 ID FSx

- *region-code*— ONTAP 文件系统的部署 AWS 区域 位置（必须与您的 AWS PCS 系统相同） FSx
- *volume-name*— ONTAP FSx 的卷名

```
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 svm-id.filesystem-id.fsx.region-code.amazonaws.com:/volume-name /mount-
point-directory

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

FSx 适用于 OpenZFS 的亚马逊

文件系统设置

在要使用 AWS PCS FSx 的 VPC 中创建一个适用于 OpenZFS 的文件系统。为了最大限度地减少区域间传输，请在同一个可用区的子网中部署，您将在那里启动大多数 AWS PCS 计算节点组实例。确保文件系统与允许从 AWS PCS 计算节点组实例进行入站和出站访问的安全组相关联。有关安全组的更多信息，请参阅 OpenZFS 用户指南中的使用 Amazon VPC 管理文件系统访问权限。FSx

启动模板

包括用于挂载 fo `cloud-config` r OpenZFS 文件系统的根卷 FSx 的用户数据。用您自己的详细信息替换此脚本中的以下值：

- *mount-point-directory*— 要在实例上挂载 for OpenZFS FSx 共享的路径
- *filesystem-id*— 适用于 OpenZFS 文件系统的文件系统 ID FSx
- *region-code*— OpenZFS 文件系统的部署 AWS 区域 位置（必须与您的 AWS PCS 系统相同） FSx

```
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

文件系统设置

在您将使用 PC AWS S 的 VPC 中创建亚马逊文件缓存。要最大限度地减少区域间传输，请在启动大多数 PCS 计算节点组实例的同一个可用区中选择一个子网。确保文件缓存与安全组关联，该安全组允许您的 PCS 实例和文件缓存之间通过端口 988 进行入站和出站流量。有关安全组的更多信息，请参阅《亚马逊文件缓存用户指南》中的 Amazon VPC 缓存访问控制。

启动模板

将文件系统设置中的安全组添加到将用于计算节点组的启动模板中。

包括用于`cloud-config`挂载 Amazon 文件缓存的用户数据。用您自己的详细信息替换此脚本中的以下值：

- *mount-point-directory*— 你要为 Lustre 挂载 FSx 的实例上的路径
- *cache-dns-name*— 文件缓存的域名系统 (DNS) 名称
- *mount-name*— 文件缓存的挂载名称

```
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==--
```

适用于 AWS PCS 的亚马逊机器映像 (AMIs)

AWS PCS AMIs 可与您提供的软件配合使用，为集群中节点上的软件和配置提供了极大的灵活性。如果您正在试用 AWS PCS，则可以使用由提供并由维护的示例 AMI AWS。如果您在生产环境中使用 AWS PCS，我们建议您自己构建 AMIs。本主题介绍如何发现和使用示例 AMIs，以及如何构建和使用自己的自定义示例 AMIs。

主题

在 AWS PCS 上使用示例亚马逊系统映像 (AMIs)

AWS 提供了示例 AMIs，您可以将其用作使用 AWS PCS 的起点。

> **Important**
>
> 示例 AMIs 仅用于演示目的，不建议用于生产工作负载。

> **Important**
>
> 如果子网仅配置为使用，则使用 AWS IPv6 PCS 示例 AMIs 和多个网络接口配置的计算节点组目前将无法运行。改用双栈子网（IPv4 和 IPv6）或 IPv4仅使用双栈子网。

查找当前的 AWS PCS 样本 AMIs

AWS 管理控制台

AWS PCS 示例 AMIs 具有以下命名约定：

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

接受的值

- *OS* – amzn2
- *architecture* – x86_64 或 arm64
- *scheduler* – slurm
- *scheduler-major-version* – 25.05

查找 AWS PCS 样本 AMIs

1. 打开 Amazon EC2 控制台。
2. 导航到 AMIs。
3. 选择公有映像。
4. 在按属性或标签查找 AM I 中，使用模板名称搜索 AMI。

   示例

   - Arm64 实例上适用于 Slurm 25.05 的 AMI 示例

     ```
     aws-pcs-sample_ami-amzn2-arm64-slurm-25.05
     ```

   - x86 实例上适用于 Slurm 25.05 的 AMI 示例

     ```
     aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05
     ```

   > **Note**
   >
   > 如果有多个 AMIs，请使用带有最新时间戳的 AMI。

5. 创建或更新计算节点组时使用 AMI ID。

AWS CLI

您可以通过以下命令找到最新的 AWS PCS 示例 AMI。*region-code*替换为使用 AWS PCS AWS 区域 的地方，例如us-east-1。

- x86_64

```
aws ec2 describe-images --region region-code --owners amazon \
--filters 'Name=name,Values=aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05*' \
            'Name=state,Values=available' \
--query 'sort_by(Images, &CreationDate)[-1].[Name,ImageId]' --output text
```

- Arm64

```
aws ec2 describe-images --region region-code --owners amazon \
--filters 'Name=name,Values=aws-pcs-sample_ami-amzn2-arm64-slurm-25.05*' \
            'Name=state,Values=available' \
--query 'sort_by(Images, &CreationDate)[-1].[Name,ImageId]' --output text
```

创建或更新计算节点组时使用 AMI ID。

了解有关 AWS PCS 示例的更多信息 AMIs

要查看 AWS PCS 示例当前版本和先前版本的内容和配置详细信息 AMIs，请参见AWS PCS 示例的发行说明 AMIs。

自己开发与 AWS PCS AMIs 兼容

要了解如何自己 AMIs 构建可与 AWS PCS 配合使用，请参阅适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)

AWS PCS 旨在与您为该服务提供的亚马逊系统映像 (AMI) 配合使用。它们上面 AMIs 可以安装任意软件和配置，前提是它们安装并正确配置了 AWS PCS 代理和兼容版本的 Slurm。您必须使用 AWS提供的安装程序在自定义 AMI 上安装 AWS PCS 软件。我们建议您使用 AWS提供的安装程序在自定义 AMI 上安装 Slurm，但如果您愿意，也可以自己安装 Slurm（不推荐）。

> **Note**
>
> 如果您想在不构建自定义 AMI 的情况下试用 AWS PCS，则可以使用提供的示例 AMI AWS。有关更多信息，请参阅 在 AWS PCS 上使用示例亚马逊系统映像 (AMIs)。

> **Important**
>
> AWS PCS 目前需要 IPv4 支持本地节点通信的内核，即使您在 IPv6仅限网络中使用 AWS PCS 也是如此。

本教程帮助您创建可与 PCS 计算节点组配合使用的 AMI，从而为您的 HPC 和 AI/ML 工作负载提供支持。

主题

步骤 1-启动临时实例

启动一个临时实例，您可以使用该实例来安装和配置 AWS PCS 软件和 Slurm 调度程序。您可以使用此实例创建与 AWS PCS 兼容的 AMI。

启动临时实例

1. 打开 Amazon EC2 控制台。
2. 在导航窗格中，选择实例，然后选择启动实例以打开新的启动实例向导。
3. （可选）在名称和标签部分中，提供实例的名称，例如`PCS-AMI-instance`。名称作为资源标签（`Name=PCS-AMI-instance`）分配给实例。
4. 在 Application and OS Images（应用程序和操作系统映像）部分中，为其中一个支持的操作系统选择 AMI。
5. 在 Instance type（实例类型）部分中，选择 supported instance type（支持的实例类型）。
6. 在 Key pair（密钥对）部分中，选择要用于实例的密钥对。
7. 在网络设置 部分中：

- 对于防火墙（安全组），选择选择现有安全组，然后选择允许对您的实例进行入站 SSH 访问的安全组。

8. 在 Storage（存储）部分中，根据需要配置卷。确保配置足够的空间来安装您自己的应用程序和库。
9. 在 Summary（摘要）面板中，选择 Launch instance（启动实例）。

步骤 2-安装 AWS PCS 代理

安装用于配置 AWS PCS 启动的实例的代理，以便与 Slurm 配合使用。有关 AWS PCS 代理的更多信息，请参阅AWS PCS 代理版本。

安装 AWS PCS 代理

1. 连接到您启动的实例。有关更多信息，请参阅连接到您的 Linux 实例。
2. （可选）为确保您的所有软件包都是最新的，请对您的实例执行快速软件更新。此过程可能需要几分钟时间。

   - 亚马逊 Linux 2、亚马逊 Linux 2023、RHEL 9、RHEL 8、Rocky Linux 9 和 Rocky Linux 8

     ```
     sudo yum update -y
     ```

   - Ubuntu 22.04 和 Ubuntu 24.04

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

3. 重启实例并重新连接到它。
4. 下载 AWS PCS 代理安装文件。安装文件被打包成压缩的 tarball (`.tar.gz`) 文件。要下载最新的稳定 版本，请使用以下命令。*region*替换为启动临时实例 AWS 区域 的位置，例如`us-east-1`。

   ```
   curl https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.2-1.tar.gz -o aws-pcs-agent-v1.3.2-1.tar.gz
   ```

   您也可以通过将前面的命令`latest`中的版本号替换为来获取最新版本（例如：`aws-pcs-agent-v1-latest.tar.gz`）。

> **Note**
>
> 在 AWS PCS 代理软件的未来版本中，这种情况可能会发生变化。

5. （可选）验证 AWS PCS 软件压缩包的真实性和完整性。建议您执行此操作以验证软件发布者的身份，并检查该文件自发布以来是否已被更改或损坏。

   a. 下载适用于 AWS PCS 的 GPG 公钥并将其导入您的密钥环。*region*替换为启动临时实例 AWS 区域 的位置。该命令应返回一个密钥值。记录密钥值；您可以在下一步中使用它。

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

   b. 运行以下命令验证 GPG 密钥的指纹。

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

   该命令应返回与以下内容相同的指纹：

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

   > **Important**
   >
   > 如果指纹不匹配，请不要运行 AWS PCS 代理安装脚本。请联系 AWS Support。

   c. 下载签名文件并验证 AWS PCS 软件压缩包文件的签名。*region*替换为您启动临时实例 AWS 区域 的位置，例如us-east-1。

   ```
   wget https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.2-1.tar.gz.sig && \
       gpg --verify ./aws-pcs-agent-v1.3.2-1.tar.gz.sig
   ```

   此输出应当类似于如下所示：

   ```
   gpg: assuming signed data in './aws-pcs-agent-v1.3.2-1.tar.gz'
   gpg: Signature made Thu 06 Nov 2025 11:10:36 AM CET using RSA key ID ECC0AE5C
   gpg: Good signature from "AWS PCS Packages (AWS PCS Packages)"
   ```

```
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: B7E1 8788 3517 6A74 C3D5  EAF5 6088 136D ECC0 AE5C
```

如果结果包含Good signature并且指纹与上一步返回的指纹相匹配，则继续下一步。

> **Important**
>
> 如果指纹不匹配，请不要运行 AWS PCS 软件安装脚本。请联系 AWS Support。

6. 从压缩文件中提取.tar.gz文件并导航到解压缩的目录。

```
tar -xf aws-pcs-agent-v1.3.2-1.tar.gz && \
    cd aws-pcs-agent
```

7. 安装 AWS PCS 软件。

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

8. 检查 AWS PCS 软件版本文件以确认安装成功。

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

此输出应当类似于如下所示：

```
AGENT_INSTALL_DATE='Fri Dec 13 12:28:43 UTC 2024'
AGENT_VERSION='1.3.2'
AGENT_RELEASE='1'
```

第 3 步 — 安装 Slurm

安装与 PCS 兼容的 Slurm 版本。 AWS 有关更多信息，请参阅 PCS 中的 Slurm 版本 AWS。

> **Note**
>
> 如果您的 AMI 上安装了 Slurm 软件的先前版本，则必须执行以下步骤才能安装新版本的 Slurm。根据创建集群时配置的 Slurm 版本， AWS PCS 代理在运行时启用 Slurm 二进制文件的正确版本。

要安装 Slurm

1. Connect 连接到安装了 AWS PCS 软件的同一个临时实例。
2. 下载 Slurm 安装程序软件。Slurm 安装程序被打包成压缩的 tarball () `.tar.gz` 文件。要下载最新的稳定 版本，请使用以下命令。*`region`*替换 AWS 区域 为临时实例的，例如`us-east-1`。

   ```
   curl https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz \
        -o aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz
   ```

   您也可以通过将前面的命令`latest`中的版本号替换为来获取最新版本（例如：`aws-pcs-slurm-25.05-installer-latest.tar.gz`）。有关带有校验和的可用版本的完整列表，请参见PCS 中的 Slurm 版本 AWS。

   > **Note**
   >
   > 在 Slurm 安装程序软件的未来版本中，这种情况可能会发生变化。

3. （可选）验证 Slurm 安装程序压缩包的真实性和完整性。建议您执行此操作以验证软件发布者的身份，并检查该文件自发布以来是否已被更改或损坏。

   a. 下载适用于 AWS PCS 的 GPG 公钥并将其导入您的密钥环。*`region`*替换为启动临时实例 AWS 区域 的位置。该命令应返回一个密钥值。记录密钥值；您可以在下一步中使用它。

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

   b. 运行以下命令验证 GPG 密钥的指纹。

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

该命令应返回与以下内容相同的指纹：

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

> **Important**
>
> 如果指纹不匹配，请不要运行 Slurm 安装脚本。请联系 AWS Support。

c. 下载签名文件并验证 Slurm 安装程序压缩包文件的签名。*region*替换为您启动临时实例 AWS 区域 的位置，例如us-east-1。

```
wget https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-slurm/aws-pcs-
slurm-25.05-installer-25.05.5-1.tar.gz.sig && \
    gpg --verify ./aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz.sig
```

此输出应当类似于如下所示：

```
gpg: assuming signed data in './aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz'
gpg: Signature made Fri 14 Nov 2025 11:35:15 AM UTC using RSA key ID ECC0AE5C
gpg: Good signature from "AWS PCS Packages (AWS PCS Packages)"
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: B7E1 8788 3517 6A74 C3D5  EAF5 6088 136D ECC0 AE5C
```

如果结果包含Good signature并且指纹与上一步返回的指纹相匹配，则继续下一步。

> **Important**
>
> 如果指纹不匹配，请不要运行 Slurm 安装脚本。请联系 AWS Support。

4. 从压缩的 .tar.gz 文件中提取文件，并导航到提取的目录。

```
tar -xf aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz && \
    cd aws-pcs-slurm-25.05-installer
```

5. 安装 Slurm。安装程序会下载、编译和安装 Slurm 及其依赖项。这需要几分钟，具体取决于您选择的临时实例的规格。

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

6. 检查调度程序版本文件以确认安装。

```
cat /opt/aws/pcs/scheduler/slurm-25.05/version
```

此输出应当类似于如下所示：

```
SLURM_INSTALL_DATE='Fri Nov 14 15:15:37 UTC 2025'
SLURM_VERSION='25.05.5'
PCS_SLURM_RELEASE='1'
```

步骤 4-（可选）安装其他驱动程序、库和应用程序软件

在临时实例上安装其他驱动程序、库和应用程序软件。安装过程将因特定的应用程序和库而异。如果您之前没有为 AWS PCS 构建过自定义 AMI，我们建议您先构建和测试 AMI，只安装了 AWS PCS 软件和 Slurm，然后在确认初步成功后逐步添加自己的软件和配置。

示例

- Elastic Fabric Adapter (EFA) 软件。有关更多信息，请参阅《亚马逊弹性计算云用户指南》中的 Amazon EC2 上的 HPC 工作负载的 EFA 和 MPI 入门。
- 亚马逊 Elastic File System（亚马逊 EFS）客户端。有关更多信息，请参阅 Amazon Elastic F ile System 用户指南中的手动安装 Amazon EFS 客户端。
- Lustre 客户端，使用亚马逊获取 FSx Lustre 和亚马逊文件缓存。有关更多信息，请参阅 for Lustre 用户指南中的安装 Lustre 客户端。FSx
- Amazon CloudWatch 代理，用于使用 CloudWatch 日志和指标。有关更多信息，请参阅 Amazon CloudWatch 用户指南中的安装 CloudWatch 代理。
- AWS 神经元，使用 t rn* 和 inf * 实例类型。有关更多信息，请参阅 AWS Neuron 文档。
- NVIDIA 驱动程序、CUDA 和 DCGM，用于使用 p* 或 g * 实例类型。

第 5 步 — 创建与 AWS PCS 兼容的 AMI

安装所需的软件组件后，您可以创建一个 AMI，您可以重复使用它来启动 AWS PCS 计算节点组中的实例。

> **Important**
>
> AWS PCS 目前需要 IPv4 支持本地节点通信的内核，即使您在 IPv6仅限网络中使用 AWS PCS 也是如此。

从临时实例创建 AMI

1. 打开 Amazon EC2 控制台。
2. 在导航窗格中，选择 Instances (实例)。
3. 选择您创建的临时实例。选择 “操作”、“图像”、“创建图像”。
4. 对于 Create image (创建映像)，请执行以下操作：
   a. 对于 Image name (映像名称)，为 AMI 输入一个描述性名称。
   b. （可选）对于 Image description (映像描述)，输入 AMI 用途的简要描述。
   c. 选择创建映像。
5. 在导航窗格中，请选择 AMIs。
6. 在列表中找到您创建的 AMI。等待其状态从 “待定” 变为 “可用”，然后将其与 AWS PCS 计算节点组配合使用。

步骤 6-将自定义 AMI 与 AWS PCS 计算节点组配合使用

您可以将自定义 AMI 与新的或现有的 AWS PCS 计算节点组配合使用。

> **Important**
>
> AWS PCS 目前需要 IPv4 支持本地节点通信的内核，即使您在 IPv6仅限网络中使用 AWS PCS 也是如此。

New compute node group

使用自定义 AMI

1. 打开 AWS PCS 控制台。
2. 在导航窗格中，选择集群。
3. 选择要在其中使用自定义 AMI 的集群，然后选择计算节点组。

4. 创建新的计算节点组。有关更多信息，请参阅 在 AWS PCS 中创建计算节点组。在 AMI ID 下，搜索要使用的自定义 AMI 的名称或 ID。完成计算节点组的配置，然后选择创建计算节点组。
5. （可选）确认 AMI 支持实例启动。在计算节点组中启动实例。为此，您可以将计算节点组配置为具有单个静态实例，也可以向使用该计算节点组的队列提交作业。

   a. 检查 Amazon EC2 控制台，直到出现带有新计算节点组 ID 的实例。有关这方面的更多信息，请参阅在 AWS PCS 中查找计算节点组实例...

   b. 当您看到实例启动并完成其引导过程时，请确认它正在使用预期的 AMI。为此，请选择实例，然后在详细信息下检查 AMI ID。它应与您在计算节点组设置中配置的 AMI 相匹配。

   c. （可选）将计算节点组扩展配置更新为您的首选值。

Existing compute node group

使用自定义 AMI

1. 打开 AWS PCS 控制台。
2. 在导航窗格中，选择集群。
3. 选择要在其中使用自定义 AMI 的集群，然后选择计算节点组。
4. 选择要配置的节点组，然后选择编辑。在 AMI ID 下，搜索要使用的自定义 AMI 的名称或 ID。完成计算节点组的配置，然后选择更新。在计算节点组中启动的新实例将使用更新后的 AMI ID。现有实例将继续使用旧的 AMI，直到 AWS PCS 取代它们。有关更多信息，请参阅 更新 AWS PCS 计算节点组。
5. （可选）确认 AMI 支持实例启动。在计算节点组中启动实例。为此，您可以将计算节点组配置为具有单个静态实例，也可以向使用该计算节点组的队列提交作业。

   a. 检查 Amazon EC2 控制台，直到出现带有新计算节点组 ID 的实例。有关这方面的更多信息，请参阅在 AWS PCS 中查找计算节点组实例...

   b. 当您看到实例启动并完成其引导过程时，请确认它正在使用预期的 AMI。为此，请选择实例，然后在详细信息下检查 AMI ID。它应与您在计算节点组设置中配置的 AMI 相匹配。

   c. （可选）将计算节点组扩展配置更新为您的首选值。

步骤 7-终止临时实例

确认您的 AMI 在 AWS PCS 上按预期运行后，您可以终止临时实例以停止为此产生费用。

终止临时实例

1. 打开 Amazon EC2 控制台。
2. 在导航窗格中，选择 Instances (实例)。
3. 选择您创建的临时实例，然后选择操作、实例状态、终止实例。
4. 当系统提示您确认时，选择终止。

要为 PCS 定制 AMIs 版本的软件安装程序 AWS

AWS 提供了可在实例上安装 AWS PCS 软件的可下载文件。 AWS 还提供了可以下载、编译和安装相关版本的 Slurm 及其依赖项的软件。您可以使用这些说明来构建用 AMIs 于 AWS PCS 的自定义版本，也可以使用自己的方法。

目录

AWS PCS 代理软件安装程序

AWS PCS 代理软件安装程序将实例配置为在实例引导过程中与 AWS PCS 配合使用。您必须使用AWS提供的安装程序在自定义 AMI 上安装 AWS PCS 代理。

有关 AWS PCS 代理软件的更多信息，请参见AWS PCS 代理版本。

Slurm 安装程序

Slurm 安装程序下载、编译和安装 Slurm 及其依赖项的相关版本。你可以使用 Slurm 安装程序为 PCS 构建自定义版本 AMIs 。 AWS 您也可以使用自己的机制，前提是它们与 Slurm 安装程序提供的软件配置一致。有关 AWS PCS 对 Slurm 的支持的更多信息，请参阅。PCS 中的 Slurm 版本 AWS

AWS提供的软件安装以下内容：

- Slurm 处于所要求的主版本和维护版本（当前版本 25.05.x）——许可证 GPL 2
  - Slurm 的构建设置为 `--sysconfdir /etc/slurm`
  - Slurm 是用以下选项构建的 `--enable-pam --without-munge`
  - Slurm 是用选项构建的 `--sharedstatedir=/run/slurm/`
  - Slurm 是在 PMIX 和 JWT 支持下构建的
  - Slurm 安装在 `/opt/aws/pcs/schedulers/slurm-25.05`
- OpenpMix（版本 4.2.6）— 许可证
  - OpenpMix 是作为子目录安装的 `/opt/aws/pcs/scheduler/`
- libjwt（版本 1.17.0）— 许可证 MPL-2.0
  - libjwt 是作为子目录安装的 `/opt/aws/pcs/scheduler/`

AWS提供的软件按如下方式更改系统配置：

- 将版本创建的 Slurm `systemd` 文件复制到文件`/etc/systemd/system/`名中。`slurmd-25.05.service`
- 如果不存在，则使用 of 创建 Slurm 用户和群组 (`slurm:slurm`)。 UID/GID `401`
- 该文件夹`/etc/aws/pcs/scheduler/slurm-25.05/plugstack.conf.d/`已创建，用于存储您的使用 SPANK 插 AWS 件在 PCS 上扩展 Slurm 功能配置。
- 在 Amazon Linux 2 和 Rocky Linux 9 上，安装会添加 EPEL 存储库，用于安装构建 Slurm 或其依赖项所需的软件。
- 安装 RHEL9 时将启用`codeready-builder-for-rhel-9-rhui-rpms`并`epel-release-latest-9`从中`fedoraproject`安装构建 Slurm 或其依赖项所需的软件。

支持的操作系统

请参阅AWS PCS 中支持的操作系统。

> **Note**
>
> AWS Deep Learning AMIs 基于亚马逊 Linux 2 和 Ubuntu 22.04 的 (DLAMI) 版本应与 AWS PCS 软件和 Slurm 安装程序兼容。有关更多信息，请参阅《开发者指南》中的 “选择您的DLAMI”。AWS Deep Learning AMIs

支持的实例类型

AWS PCS 软件和 Slurm 安装程序支持任何可以运行支持的操作系统之一的 x86_64 或 arm64 实例类型。

支持的 Slurm 版本

请参阅PCS 中的 Slurm 版本 AWS。

使用校验和验证安装程序

您可以使用 SHA256 校验和来验证安装程序压缩包 (.tar.gz) 文件。建议您执行此操作以验证软件发布者的身份，并检查该应用程序自发布以来是否已被更改或损坏。

验证压缩包

使用 s ha256sum 实用程序获取 SHA256 校验和并指定压缩包文件名。您必须从保存 tarball 文件的目录中运行该命令。

- SHA256

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

该命令应返回以下格式的校验和值。

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

将命令返回的校验和值与下表中提供的校验和值进行比较。如果校验和匹配，则可以安全地运行安装脚本。

> **Important**
>
> 如果校验和不匹配，请不要运行安装脚本。联系 支持。

例如，以下命令生成 Slurm 25.0 SHA256 5.5-1 压缩包的校验和。

```
$ sha256sum aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz
```

输出示例：

```
3b0f93bce441d4f4f6935175f2c1e81cd961cb923adb416fa6689f5592047a7d aws-pcs-slurm-25.05-
installer-25.05.5-1.tar.gz
```

下表列出了最新版本安装程序的校验和。*us-east-1*替换为使用 AWS PCS AWS 区域 的地方。

AWS PCS 代理

Installer（安装程序）	下载 URL	SHA256 校验和
AWS PCS 代理 1.3.2-1	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.2-1.tar.gz`	`06b32a952a1c849e3442e35c28ac2e4d6962b09286cad748f3c83d561b52ec6f`
AWS PCS 代理 1.3.1-1	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.1-1.tar.gz`	`5b7f1eb7b3a86bd2d331b5cb0138d868dc9452da34b480becd86af892c7e8d19`
AWS PCS 代理 1.3.0-1	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.0-1.tar.gz`	`eadc9b65c3db248bdde2a6c41814dfb1b97239f24ad55e03d8526dd9ab4a8d16`
AWS PCS 代理 1.2.2-1	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.2.2-1.tar.gz`	`fd7b6ea5442db75d723fc4971781ce6ae511baa21b87c4286fc1df8127b282b8`
AWS PCS 代理 1.2.1-1	`https://aws-pcs-repo- us-east-1 .s3.us-`	`2b784643ca01ccca1baa64fbfb34bb41efe8`

Installer（安装程序）	下载 URL	SHA256 校验和
	east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.2.1-1.tar.gz	bdca69470998b74ce3962bc271d4
AWS PCS 代理 1.2.0-1	https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.2.0-1.tar.gz	470db8c4fc9e50277b6317f98584b6b547e73523043e34f018eecae767846805
AWS PCS 代理 1.1.1-1	https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.1.1-1.tar.gz	bef078bf60a6d8ecde2e6c49cd34d088703f02550279e3bf483d57a235334dc6
AWS PCS 代理 1.1.0-1	https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.1.0-1.tar.gz	594c32194c71bccc5d66e5213213ae38dd2c6d2f9a950bb01accea0bbab0873a
AWS PCS 代理 1.0.1-1	https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.0.1-1.tar.gz	04e22264019837e3f42d8346daf5886eaacecd21571742eb505ea8911786bcb2

Installer（安装程序）	下载 URL	SHA256 校验和
AWS PCS 代理 1.0.0-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.0.0-1.tar.gz	d2d3d68d00c685435c38af471d7e2492dde5ce9eb222d7b6ef0042144b134ce0

Slurm 安装程序

Installer（安装程序）	下载 URL	SHA256 校验和
Slurm 25.05.5-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz	e7bc84db4e71b8c7174e2f581a31233f839affb5306c76a8adba23204dcc703b
Slurm 25.05.4-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.4-1.tar.gz	3b0f93bce441d4f4f6935175f2c1e81cd961cb923adb416fa6689f5592047a7d
Slurm 25.05.3-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.3-1.tar.gz	851bb5815b6700ceb30cc4a3fda204ca8ce362c14528c339908983255a936cf0
Slurm 24.11.7-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs	73d75be82c6f88f6e248fd0cc779a5630c62

Installer（安装程序）	下载 URL	SHA256 校验和
	-slurm-24.11-insta ller-24.11.7-1.tar.gz	d91ebabdd9cf0f61b1 943b6d7d09
Slurm 24.11.6-2	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.11-insta ller-24.11.6-2.tar.gz	f17cd78e0bc6b9c818 b794d9d2685cceabdc 73f4fbb12f7566ae5b 86a5abc32b
Slurm 24.11.6-1	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.11-insta ller-24.11.6-1.tar.gz	225de9fc18206f5f65 f412effe1fd457614a c97ee9822b3ff804a4 52b0fae522
Slurm 24.11.5-1	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.11-insta ller-24.11.5-1.tar.gz	593efe4d66bef2f3e4 6d5a382fb5a32f7a3c a2510bcf1b3c85739f 4f951810d5
Slurm 24.05.8-2	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.05-insta ller-24.05.8-2.tar.gz	c494b0b55c319a4c2f 3faf668c759d46c32c 4c7aa94ae97d941283 28fe95364b

Installer（安装程序）	下载 URL	SHA256 校验和
Slurm 24.05.8-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-24.05-installer-24.05.8-1.tar.gz`	`210a43b376af082bbad640b2032655885790c5dab0e6489cc327c7310a375849`
Slurm 24.05.7-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-24.05-installer-24.05.7-1.tar.gz`	`0b5ed7c81195de2628c78f37c79e63fc4ae99132ca6b019b53a0d68792ee82c5`
Slurm 24.05.5-2	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-24.05-installer-24.05.5-2.tar.gz`	`7cc8d8294f2fbff95fe0602cf9e21e02003b5d96c0730e0a18c6aa04c7a4967b`
Slurm 23.11.10-4（已弃用）	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-4.tar.gz`	`bb2d8c919c69dba38d14358f49c7f0427564c5dd4af85a1c9eca2c57ceeae29a`
Slurm 23.11.10-3（已弃用）	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-3.tar.gz`	`488a10ee0fbd57ec0e0ff7ea708a9e3038fafdc025c6bb391c75c2e2a7852a00`

Installer（安装程序）	下载 URL	SHA256 校验和
Slurm 23.11.10-2（已弃用）	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-2.tar.gz	0bbe85423305c05987931168caf98da08a34c25f9eec0690e8e74de0b7bc8752
Slurm 23.11.10-1（已弃用）	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-1.tar.gz	27e8faa9980e92cdfd8cfdc71f937777f0934552ce61e33dac4ecf5a20321e44
Slurm 23.11.9-1（已弃用）	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz	1de7d919c8632fe8e2806611bed4fde1005a4fadc795412456e935c7bba2a9b8

AWS PCS 示例的发行说明 AMIs

AMIs 对于支持的最新计划程序主要版本，请接收安全更新和严重错误修复。这些增量安全补丁未包含在官方发行说明中。

> **Important**
>
> 不支持与旧调度程序版本 AMIs 相关的示例，也不会收到更新。

> **Important**
>
> 示例 AMIs 仅用于演示目的，不建议用于生产工作负载。

目录

AWS x86_6 AMIs 4 的 PCS 样本（亚马逊 Linux 2）

Slurm 25.05

AMI 名称

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05`

支持的 EC2 实例

- 所有采用 64 位 x86 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索架构 =x86_64。

AMI 内容

- 支持的 AWS 服务：AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：x86_64
- EBS 卷类型：gp2
- EFA 安装程序：1.43.1
- GDRCopy: 2.5.1
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

Slurm 24.11

> **Note**
>
> AWS PCS 支持 Slurm 24.11 及更高版本的账目。有关更多信息，请参阅 PCS 中的 Slurm 会计 AWS。

AMI 名称

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-24.11`

支持的 EC2 实例

- 所有采用 64 位 x86 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索`Architectures=x86_64`。

AMI 内容

- 支持的 AWS 服务： AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：x86_64
- EBS 卷类型：gp2
- EFA 安装程序：1.33.0
- GDRCopy: 2.4
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

Slurm 24.05

AMI 名称

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-24.05`

支持的 EC2 实例

- 所有采用 64 位 x86 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索Architectures=x86_64。

AMI 内容

- 支持的 AWS 服务： AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：x86_64
- EBS 卷类型：gp2
- EFA 安装程序：1.33.0
- GDRCopy: 2.4
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

Slurm 23.11

AMI 名称

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

支持的 EC2 实例

- 所有采用 64 位 x86 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索Architectures=x86_64。

AMI 内容

- 支持的 AWS 服务： AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：x86_64
- EBS 卷类型：gp2
- EFA 安装程序：1.33.0

- GDRCopy: 2.4
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

AWS Arm64 AMIs 的 PCS 样本（亚马逊 Linux 2）

Slurm 25.05

AMI 名称

- `aws-pcs-sample_ami-amzn2-arm64-slurm-25.05`

支持的 EC2 实例

- 所有带有 64 位 Arm 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索架构 =arm64。

AMI 内容

- 支持的 AWS 服务：AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：arm64
- EBS 卷类型：gp2
- EFA 安装程序：1.43.1
- GDRCopy: 2.5.1
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

Slurm 24.11

> **Note**
>
> AWS PCS 支持 Slurm 24.11 及更高版本的账目。有关更多信息，请参阅 PCS 中的 Slurm 会计 AWS。

AMI 名称

- `aws-pcs-sample_ami-amzn2-arm64-slurm-24.11`

支持的 EC2 实例

- 所有带有 64 位 Arm 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索`Architectures=arm64`。

AMI 内容

- 支持的 AWS 服务： AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：arm64
- EBS 卷类型：gp2
- EFA 安装程序：1.33.0
- GDRCopy: 2.4
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

Slurm 24.05

AMI 名称

- `aws-pcs-sample_ami-amzn2-arm64-slurm-24.05`

支持的 EC2 实例

- 所有带有 64 位 Arm 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索`Architectures=arm64`。

AMI 内容

- 支持的 AWS 服务： AWS PCS
- 操作系统：Amazon Linux 2

- 计算架构：arm64
- EBS 卷类型：gp2
- EFA 安装程序：1.33.0
- GDRCopy: 2.4
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

Slurm 23.11

AMI 名称

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

支持的 EC2 实例

- 所有带有 64 位 Arm 处理器的实例。要查找兼容的实例，请导航至 Amazon EC2 控制台。选择实例类型，然后搜索`Architectures=arm64`。

AMI 内容

- 支持的 AWS 服务： AWS PCS
- 操作系统：Amazon Linux 2
- 计算架构：arm64
- EBS 卷类型：gp2
- EFA 安装程序：1.33.0
- GDRCopy: 2.4
- NVIDIA 驱动程序：550.127.08
- NVIDIA CUDA：12.4.1_550.54.15

AWS PCS 中支持的操作系统

AWS PCS 使用为计算节点组配置的 Amazon 系统映像 (AMI) 来启动该计算节点组中的 EC2 实例。AMI 决定了 EC2 实例使用的操作系统。您无法在 AWS PCS 示例中更改操作系统 AMIs。如果要使用其他操作系统，则必须创建自定义 AMI。有关更多信息，请参阅 适用于 AWS PCS 的亚马逊机器映像 (AMIs)。

支持的操作系统

- Amazon Linux 2

  这是 AWS PCS 示例中的操作系统 AMIs。

  > **Important**
  >
  > 示例 AMIs 仅用于演示目的，不建议用于生产工作负载。即使您打算使用 Amazon Linux 2，您也应该为生产工作负载创建和使用自定义 AMI。

- Amazon Linux 2023
- RedHat 企业 Linux 9 (RHEL 9)

  任何实例类型的 RHEL 的按需成本都高于其他支持的操作系统。有关定价的更多信息，请参阅按需定价和 Amazon Elastic Compute Cloud 上如何提供和定价 Red Hat Enterprise Linux？。

- RedHat 企业 Linux 8 (RHEL 8)
- Rocky Linux 9

  你可以使用官方的 Rocky Linux 9 AMIs 作为自定义 AMI 的基础。如果基本 AMI 没有最新的内核，则您的自定义 AMI 构建可能会失败。

  升级内核

  1. 通过 https://rockylinux.org/cloud-images/，使用 rocky9 AMI id 启动实例
  2. 通过 ssh 登录实例并运行以下命令：

     ```
     sudo yum -y update
     ```

  3. 从实例创建镜像。您可以将此图像指定`ParentImage`为自定义 AMI。

- 洛基 Linux 8

- Ubuntu 22.04

  Ubuntu 22.04 需要更安全的密钥才能使用 SSH，并且默认情况下不支持 RSA 密钥。我们建议您改为生成和使用 ED25519 密钥。

- Ubuntu 24.04

AWS PCS 代理版本

AWS PCS 代理软件将 AWS PCS 启动的 EC2 实例配置为与 Slurm 配合使用。您将代理包含在为集群创建计算节点组时指定的亚马逊系统映像 (AMI) 中。在这些计算节点组中启动的 EC2 实例使用指定的 AMI 及其随附的 AWS PCS 代理软件。 AWS PCS 代理使 EC2 实例能够将自己注册为集群的一部分。要使用最新的 AWS PCS 代理软件，必须更新您的自定义软件 AMIs。有关更多信息，请参阅适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)中的步骤 2-安装 AWS PCS 代理。

AWS PCS 代理版本	发行日期	发行说明
v1.3.2-1	2026年3月10日	• 修复了运行 RHEL 8.10 或 Rocky Linux 8.10 的计算节点由于这些操作系统中的 `curl` SigV4 向后移植错误而无法启动的问题。
v1.3.1-1	2025 年 11 月 7 日	• 通过使用 “smt/control” sysfs 参数（如果有）改进了对超线程的禁用。 • 修复了 PCS Agent 尝试禁用超线程时 CPU 在启动期间锁定时可能出现的争用情况。 • 修复了导致 Slurm 计算节点的`InstanceId` 和`InstanceType` 字段分别填充时间戳和连字符的问题。
v1.3.0-1	2025 年 11 月 3 日	• 增加了对新操作系统的支持：亚马逊 Linux 2023、Ubuntu 24、RHEL 8、Rocky 8。
v1.2.2-1	2025 年 10 月 16 日	• 如果 IPv6 终端节点不可用，则允许向 IPv4 终端节点查询实例元数据。

AWS PCS 代理版本	发行日期	发行说明
		• 修复了如果内核返回兄弟线程作为 CPU ID 范围，则无法禁用超线程的问题。 • 修复了成功禁用超线程后在日志中生成错误失败消息的问题。
v1.2.1-1	2025 年 6 月 19 日	• 如果控制器不可用，AWS PCS 代理现在会尝试启动 slurmd 长达 30 分钟。 • 修复了如果对的响应包 RegisterComputeNodeGroupInstance 含 SLURMDBD 端点，则会产生错误的 slurmd 配置的问题。
v1.2.0-1	2025 年 3 月 7 日	• 已启用对 in IPv6 的支持`slurmd.conf` 。
v1.1.1-1	2024 年 12 月 13 日	• 修复了在调用时报告的 Slurm 版本不正确的问题。 RegisterComputeNodeGroupInstance • 修复了在中执行自定义脚本时无法正确获取实例元数据`/opt/aws/pcs/etc/bootstrap_hooks/` 的问题。
v1.1.0-1	2024 年 12 月 6 日	• 允许自定义脚本在`/opt/aws/pcs/etc/bootstrap_hooks/` 引导步骤之前运行。

AWS PCS 代理版本	发行日期	发行说明
v1.0.1-1	2024 年 10 月 22 日	• 修复了 NVIDIA 设备在支持 GPU 的实例上`slurmd`启动时无法运行的问题。
v1.0.0-1	2024 年 8 月 28 日	• 初始版本。

PCS 中的 Slurm 调度器 AWS

Slurm 是一款专为 Linux 集群设计的开源工作负载管理器，可为 HPC 工作负载提供作业调度、资源分配和作业监控功能。 AWS PCS 支持 Slurm 调度器来管理您的集群工作负载。

主题

PCS 中的 Slurm 版本 AWS

SchedMD 通过新功能、优化和安全补丁不断增强 Slurm。SchedMD 定期发布新的主要版本，并计划在任何给定时间最多支持 3 个版本。 AWS PCS 旨在使用补丁版本自动更新 Slurm 控制器。

当 SchedMD 终止对特定主要版本的支持时， AWS PCS 会将该版本指定为生命周期终止 (EOL)。在 EOL 之后，无法使用该版本创建新集群，但现有集群可以在没有保证支持的情况下持续运行长达 12 个月。 AWS 如果 Slurm 主版本接近 EOL，PCS 会提前发出通知，以帮助客户知道何时将其集群升级到更新的支持版本。

我们建议您使用最新支持的 Slurm 版本来部署集群，以访问最新的改进和改进。

PCS 中支持的 Slurm 版本 AWS

下表显示了支持的 Slurm 版本以及每个版本的重要日期和信息。

Slurm 版本	schedMD 发布日期	AWS PCS 发布日期	AWS PCS EOL 日期	兼容 AWS PCS 代理程序的最低版本	支持的 AWS PCS 示例 AMIs
25.05	5/29/2025	2025 年 10 月 16 日	11/30/2026	1.0.0-1	• aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05 • aws-pcs-sample_ami-amzn2-arm64-slurm-25.05
24.11	11/29/2024	2025 年 5 月 14 日	5/31/2026	1.0.0-1	• aws-pcs-sample_ami-amzn2-x86_64-slurm-24.11 • aws-pcs-sample_ami-amzn2-arm64-slurm-24.11

PCS 中不支持的 Slurm 版本 AWS

下表显示了 PCS 中 AWS 不支持的 Slurm 版本。

Slurm 版本	schedMD 发布日期	AWS PCS 发布日期	AWS PCS EOL 日期		
24.05	2024 年 5 月 30 日	12/18/2024	11/30/2025		
23.11	11/21/2023	8/28/2024	5/31/2025		

PCS 中 Slurm 版本的发行说明 AWS

本主题介绍了 PCS 当前支持的每个 Slurm 版本的重要更改。 AWS 我们建议您在升级集群时查看新旧版本之间的变化。

Slurm 25.05

在 AWS PCS 中实施的更改

- 现在，默认情况下，Slurm requeue_on_resume_f SchedulerParameter ailure 处于启用状态。
- “stderr” 作为选项已被删除 LogTimeFormat，因为它在 Slurm 25.05 中被禁用。
- AWS PCS 支持多集群 sackd 配置：登录节点可以访问多个集群。

有关 Slurm 25.05 的更多信息，请参阅以下出版物：

- schedMD 发布公告：https://www.schedmd.com/slurm-version-25-05-0-is-now-available/
- schedMD 发行说明：https://github.com/SchedMD/slurm/blob/slurm-25-05-0-1/RELEASE_notes.md

Slurm 24.11

在 AWS PCS 中实施的更改

- AWS PCS 支持 Slurm 记账。有关更多信息，请参阅 PCS 中的 Slurm 会计 AWS。

有关 Slurm 24.11 的更多信息，请参阅以下出版物：

- schedMD 发布公告
- schedMD 发行说明

Slurm 24.05

在 AWS PCS 中实施的更改

- 现在，新的 Slurm Step Manager 模块在 PCS 中 AWS 已默认启用。该模块通过将步骤管理从中央控制器转移到计算节点来提供显著的好处，从而大大提高了步进使用量大的环境中的系统并发性。为了支持此配置以及更好的隔离Prolog和Epilog流程执行，启用了新的 prolog 标志 (Contain,Alloc)。
- 支持从控制器到计算节点的分层通信，以优化 Slurm 节点内通信，从而提高可扩展性和性能。此外，路由配置现在使用分区节点列表进行来自控制器的通信，而不是插件的默认路由算法，从而增强了系统的弹性。
- 新的哈希插件HashPlugin=hash/sha3取代了以前的哈希插件hash/k12 plugin。现在，在 AWS PCS 集群中，此功能已默认启用。
- Slurm 控制器日志现在包括针对所有入站远程过程调用 (RPC) 的增强审计功能。slurmctld日志包括源地址、经过身份验证的用户和连接处理之前的 RPC 类型。

有关 Slurm 24.05 的更多信息，请参阅以下出版物：

- schedMD 发布公告
- schedMD 发行说明

Slurm 23.11

你可以在 PCS 中更改 Slurm 设置 AWS

- SuspendTime默认为60。使用 AWS PCS scaleDownIdleTimeInSeconds 配置参数进行设置。有关更多信息，请参阅《AWS PCS API 参考》中ClusterSlurmConfiguration数据类型的scaleDownIdleTimeInSeconds参数。
- MaxJobCount和MaxArraySize基于您为集群选择的大小。有关更多信息，请参阅《AWS PCS *CreateCluster* API 参考》中的 API 操作size参数。
- S SelectTypeParameters lurm 设置默认为。CR_CPU您可以将其作为值提供，slurmCustomSettings以便在创建集群时对其进行设置。有关更多信息，请参

阅 CreateCluster API 操作的slurmCustomSettings参数和 AWS PCS API 参考SlurmCustomSetting中。

- 可以在集群级别设置Prolog和Epilog。您可以将其作为值提供，slurmCustomSettings以便在创建集群时对其进行设置。有关更多信息，请参阅 AWS PCS API 参考SlurmCustomSetting中的CreateCluster和。
- 可以在计算节点组级别设置Weight和RealMemory。在创建计算节点组时slurmCustomSettings，可以将其作为值提供给进行设置。有关更多信息，请参阅 AWS PCS API 参考SlurmCustomSetting中的CreateComputeNodeGroup和。

有关 PCS 中 Slurm 版本的常见问题 AWS

AWS PCS 保持对多个 Slurm 版本的支持。推出新的 Slurm 版本时， AWS PCS 会提供技术支持和安全补丁，直到该版本终止 SchedMD 的支持 (EOS)。 AWS 为了与术语保持一致，PCS 将 Slurm 版本的 EOS 日期称为生命周期结束 (EOL)。 AWS

AWS PCS 支持 Slurm 版本多长时间？

AWS PCS 对 Slurm 版本的支持与 SchedMD 对主要版本的支持周期一致。 AWS PCS 支持当前版本和 2 个最新的先前主要版本。当 SchedMD 发布新的主要版本时， AWS PCS 将终止对支持的最旧版本的支持。 AWS PCS 会尽快发布 Slurm 的新主要版本，但在 SchedMD 的发布和在 PCS 中的上市之间可能会有延迟。 AWS

我的集群如何获得新的 Slurm 补丁版本？

为了解决错误和安全修复， AWS PCS 旨在自动将补丁应用于在内部服务拥有的帐户中运行的集群控制器。要在您的 EC2 实例上安装补丁 AWS 账户，请更新计算节点组的 Amazon 系统映像 (AMI)，并更新计算节点组以使用更新后的 AMI。有关更多信息，请参阅 适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

> **Note**
>
> 当我们更新 Slurm 控制器时，它们不可用。正在运行的作业不受影响。在集群的控制器变为不可用之前提交的任务将一直保留，直到控制器可用为止。

我如何获悉即将推出的 Slurm 版本 EOL 活动？

我们会在 EOL 日期前 6 个月向您发送一封电子邮件。我们每个月都会在 EOL 之前向您发送一封电子邮件，最后一封电子邮件将在 EOL 日期前 1 周发送给您。在 EOL 日期之后，我们会每月向运行 EOL

Slurm 版本的 AWS PCS 集群的客户发送 12 个月的电子邮件。如果发现某个 EOL Slurm 版本存在安全漏洞，我们可能会暂停该版本的集群。

如何确定我的集群使用的 Slurm 版本是否正在运行 EOL Slurm 版本？

我们会向您发送一封电子邮件，通知您您的集群正在运行 EOL Slurm 版本。我们会针对警报发布警 AWS Health Dashboard 报，其中包含使用 EOL Slurm 版本的集群的详细信息。您还可以使用 AWS PCS 控制台识别具有 EOL Slurm 版本的集群。

如果我的 Slurm 版本接近或超过 EOL，我该怎么做？

使用支持的最新版本的 Slurm 创建新集群，并在计算节点组 AMI 中更新 Slurm 版本。您的 AMI 和正在运行的 EC2 实例中的 Slurm 版本不能比集群的 Slurm 版本落后超过 2 个版本。有关更多信息，请参阅适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

如果我没有在 EOL 日期之前切换到较新版本的 Slurm 会发生什么？

您无法使用 EOL Slurm 版本创建新集群。在没有 AWS 支持的情况下，现有集群可以运行长达 12 个月，无需立即采取行动即可维持其运行。停产日期过后，将无法保证支持、安全更新和可用性。出于安全原因，我们可能会暂停集群。我们强烈建议您使用支持的 Slurm 版本来维护您的 AWS PCS 集群的安全性和支持。

使用 EOL Slurm 版本运行集群有哪些风险？

采用 EOL Slurm 版本的集群存在严重的安全和运营风险。如果没有 SchedMD 的主动监控，安全漏洞可能仍未被发现或未得到解决。如果发现严重漏洞，我们可能会立即暂停您的集群。

集群暂停后，我的任务、集群计算、存储和网络资源会怎样？

AWS PCS 管理的所有资源都将终止。这包括 Slurm 控制器、计算节点组和 EC2 实例。在计算实例上运行的任何作业都将立即终止，集群进入暂停状态。客户管理的资源（例如外部文件系统）保持不变。您可以使用 AWS PCS 控制台和 API 操作来访问集群的配置。

我能否重启已暂停的集群以恢复其剩余任务？

不，您无法重启已暂停的集群。您可以使用暂停集群的配置来创建支持的 Slurm 版本的新集群。如果将剩余的作业保存在外部文件系统中，则可以运行它们。

我能否在 12 个月的宽限期之后申请延期？

不可以，在 12 个月的宽限期之后，您不能请求延期以运行您的集群。我们延长了时间，帮助您切换到支持的 Slurm 版本。为避免集群操作中断，我们建议您在 Slurm 版本到期 OL 之前进行切换。

PCS 中的 Slurm 会计 AWS

您可以在新 AWS PCS 集群上启用记账功能，以监控集群使用情况、强制执行资源限制以及管理对特定队列或计算节点组的精细访问控制。 AWS PCS 为您的集群创建和管理会计数据库，您无需创建和管理自己的独立会计数据库。 AWS PCS 使用 Slurm 中的记账功能。有关 Slurm 中记账功能的更多信息，请参阅 SchedMD 上的 Slurm 文档。

要使用记账，请在创建新集群时将其启用，并可选择设置记账参数。在集群状态为Active并且具有计算节点组后，您可以连接到登录节点的 Linux 外壳以执行记账功能，例如使用 Slurm sacct 命令查看作业数据。

> **Note**
>
> Slurm 24.11 或更高版本支持记账。

AWS PCS console

在创建集群页面上，必须选择有效的 Slurm 版本（版本 24.11 或更高版本）。在“日程安排器设置”下，启用“会计”。

AWS PCS API

在调用 CreateCluster API 操作时提供accounting配置。在accounting对象中，将设置mode为STANDARD。有关更多信息，请参阅 AWS PCS API 参考中的CreateCluster和会计。

以下示例使用调 AWS CLI 用 CreateCluster API 操作。参数值子字符串accounting='{mode=STANDARD}'启用记账。

```
aws pcs create-cluster --cluster-name cluster-name \
                       --scheduler type=SLURM,version=24.11 \
                       --size SMALL \
                       --networking subnetIds=cluster-subnet-
id,securityGroupIds=cluster-security-group-id \
                       --slurm-configuration
 scaleDownIdleTimeInSeconds=180,accounting='{mode=STANDARD}',slurmCustomSettings='[{paramete
```

> **Important**
>
> 如果您启用会计，则会收取额外的账单费用。有关更多信息，请参阅 AWS PCS 定价页面。

修改会计设置

无需重建基础架构，即可在现有集群上启用或禁用记账。有关更多信息，请参阅 在 AWS PCS 中更新集群。

禁用记账功能后，一旦集群进入UPDATING状态，记账功能的计费就会停止。启用记账功能后，从集群成功恢复到ACTIVE状态时开始计费。

PCS 中 Slurm 会计的关键概念 AWS

以下概念特定于 PCS，用于控制 AWS PCS 如何 AWS 实现 Slurm 记账。

会计数据库

AWS PCS 将您的会计数据存储在 AWS 拥有 AWS 账户 的数据库中。您无权访问 `slurmdbd.conf`.

默认清除时间

此 AWS PCS 设置指定所有会计记录类型（作业、事件、预留、步骤、暂停、交易、使用数据）的保留期（以天为单位）。例如，如果值为 30，则 AWS PCS 会将会计记录保留 30 天。您在创建集群时提供此值。如果您不提供值， AWS PCS 会无限期地在数据库中保留会计记录。

AWS PCS console

在创建集群的步骤中，您可以指定默认的清除时间。在创建集群页面上，必须选择有效的 Slurm 版本（版本 24.11 或更高版本）并启用记账。在“计划程序设置”下，为“默认清除时间（天）”提供一个整数值。

AWS PCS API

将`defaultPurgeTimeInDays`指定为您在调用 `CreateCluster` API 操作时提供的`accounting`信息的一部分。有关更多信息，请参阅 AWS PCS API 参考中的CreateCluster和会计。

> **Note**
>
> 使用 AWS PCS API 创建集群时，的默认值为`-1`且`defaultPurgeTimeInDays`0不是有效值。

会计政策的执行

此设置决定了 Slurm 对您的集群执行任务提交规则、资源限制和会计策略的严格程度。此设置与集群`slurm.conf`文件中的`AccountingStorageEnforce`参数相对应。您可以选择执法选项的任意组合。如果您未选择任何选项，则不会对集群上的作业施加任何记账限制。 AWS PCS 支持以下选项：

- 关联- job-to-account 映射
- 限制-资源限制
- QoS — 服务质量要求
- 安全模式 — 保证在限制范围内完成
- nosteps — 禁用步骤记账
- nojobs — 禁用工作记账

有关这些选项的更多信息，请参阅 SchedMD 上的 Slurm 文档。

AWS PCS console

您可以在创建集群的步骤中设置这些选项。在创建集群页面上，必须选择有效的 Slurm 版本（版本 24.11 或更高版本）并启用记账。从 “计划程序设置” 下的 “会计政策实施” 下拉列表中选择所需的选项。

AWS PCS API

在 Slurm 中，这些选项是在集群文件中设置的。`slurm.conf`您无法直接访问您`slurm.conf`的 AWS PCS 集群。相反，您可以在创建集群时提供 `SlurmCustomSettings` `CreateCluster` API 操作。有关更多信息，请参阅 AWS PCS API 参考CreateCluster中的。

获取现有 AWS PCS 集群的记账配置

Slurm 记账配置包含在您的集群的 Slurm 配置中。

AWS PCS console

1. 从导航窗格中选择 “集群”。
2. 从列表中选择集群名称。
3. 在 “配置” 选项卡上，在 Slur m 配置下找到记账配置

AWS PCS API

使用 `GetCluster` API 操作获取集群配置。您可以在中找到记账配置`slurmConfiguration`。的设置`mode`和的值`defaultPurgeTimeInDays`都低于`accounting`。选定的会计政策执行选项如下`slurmCustomSettings`。有关更多信息，请参阅 AWS PCS API 参考GetCluster中的。

PCS 中的 Slurm REST API AWS

AWS PCS 通过为 Slurm 的原生 REST API 提供托管支持`slurmrestd`，为编程集群交互提供一个 HTTP 接口。您可以通过标准 HTTP 请求提交作业、监控集群状态和管理资源，而无需直接 shell 访问您的集群。

常见使用案例

Slurm REST API 支持各种集成场景：

- Web 应用程序集成：构建可直接提交和管理作业的自定义前端和 Web 应用程序。
- Jupyter 笔记本集成：允许研究人员在不离开开发工作流程的情况下从笔记本环境中提交作业。
- 合作伙伴解决方案集成：将第三方 HPC 工具和工作流程管理器连接到您的 AWS PCS 集群。
- 编程集群管理：自动执行作业提交、监控和资源管理工作流程。
- 研究计算工作流程：支持需要 API 驱动的作业管理的学术和企业研究环境。

要求和限制

在使用 Slurm REST API 之前，请查看以下详细信息：

- 您的集群必须使用 Slurm 版本 25.05 或更高版本。
- API 终端节点只能通过集群 VPC 内的私有 IP 地址进行访问。
- 您的集群安全组必须允许端口 6820 上的 HTTP 流量。
- 身份验证需要带有特定用户身份声明的 JWT 令牌。

目前的限制包括：

- 不支持`scontrol token`生成的令牌。
- `X-SLURM-USER-NAME`标题模拟不可用。
- 某些功能需要启用 Slurm 记账。
- 与 Slurm CLI 过滤器插件机制不兼容。
- 与 REST API 端点的连接未使用 TLS 进行加密。

主题

在 PCS 中启用 Slurm REST API AWS

启用 Slurm REST API 以访问集群的 HTTP 接口，以进行编程作业管理和监控。您可以在创建集群时启用此功能，也可以更新符合要求的现有集群。

先决条件

在启用 Slurm REST API 之前，请确保您已具备以下条件：

- 集群版本：Slurm 版本 25.05 或更高版本。
- 安全组：允许端口 6820 上的 HTTP 流量来自所需来源的规则。

过程

在新集群上启用 Slurm REST API

AWS 管理控制台

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 选择创建集群。
3. 在 “集群详细信息” 下，选择 Slurm 版本 25.05 或更高版本。

4. 根据需要配置其他群集设置。
5. 在计划程序配置部分中，将 REST API 设置为已启用。
6. 将您的集群安全组配置为允许来自所需来源的端口 6820 上的 HTTP 流量。
7. 完成集群创建过程。

AWS CLI

1. 在创建集群时添加 Slurm REST 配置。

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM, version=25.05 \
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1 \
    --slurm-configuration slurmRest='{mode=STANDARD}'
```

2. 将您的集群安全组配置为允许来自所需来源的端口 6820 上的 HTTP 流量。

在现有集群上启用 Slurm REST API

AWS 管理控制台

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 从列表中选择您的集群。
3. 在集群详细信息中确认您的集群使用 Slurm 版本 25.05 或更高版本。
4. 选择编辑集群。
5. 在计划程序配置部分中，将 REST API 设置为已启用。
6. 选择更新集群以应用更改。
7. 将您的集群安全组配置为允许来自所需来源的端口 6820 上的 HTTP 流量。

AWS CLI

1. 使用 Slurm REST 配置更新您的集群，如本示例所示。

```
aws pcs update-cluster --cluster-identifier my-cluster \
    --slurm-configuration 'slurmRest={mode=STANDARD}'
```

2. 将您的集群安全组配置为允许来自所需来源的端口 6820 上的 HTTP 流量。

启用后会发生什么

当你启用 REST API 时，AWS PCS 会自动：

- 生成 JWT 签名密钥并将其存储在 S AWS ecrets Manager 中。
- 在您的 VPC `https://<clusterPrivateIpAddress>:6820` 内公开 API 终端节点。
- 更新您的集群配置以显示 REST API 终端节点的详细信息。

现在，您可以进行身份验证并使用 REST API 进行任务管理和集群操作。

在 PCS 中使用 Slurm REST API 进行身份验证 AWS

AWS PCS 中的 Slurm REST API 使用 JSON 网络令牌 (JWT) 身份验证来确保对集群资源的安全访问。 AWS PCS 提供存储在 Secrets Manager 中的托管签名 AWS 密钥，您可以使用该密钥生成包含特定用户身份声明的 JWT 令牌。

先决条件

在使用 Slurm REST API 进行身份验证之前，请确保您已具备以下条件：

- 集群配置：启用 Slurm 25.05+ 且启用 REST API 的 AWS PCS 集群。
- AWS 权限：访问 S AWS ecrets Manager 以获取 JWT 签名密钥。
- 用户信息：用户名、POSIX 用户 ID 以及您的集群账户的一个或多个 POSIX 组 IDs 。
- 网络访问：通过允许端口 6820 的安全组在集群的 VPC 内进行连接。

过程

检索 Slurm REST API 端点地址

AWS 管理控制台

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 从列表中选择您的集群。
3. 在集群配置详细信息中，找到 “终端节点” 部分。

4. 注意 Slurm REST API (slurm restd) 的私有 IP 地址和端口。
5. 您可以通过向该地址发送格式正确的 HTTP 请求来进行 API 调用。

AWS CLI

1. 使用查询您的集群状态`aws pcs get-cluster`。在响应的`endpoints`字段中查找`SLURMRESTD`端点。示例如下：

```
"endpoints": [
      {
          "type": "SLURMCTLD",
          "privateIpAddress": "192.0.2.1",
          "port": "6817"
      },
      {
          "type": "SLURMRESTD",
          "privateIpAddress": "192.0.2.1",
          "port": "6820"
      }
  ]
```

2. 您可以通过向发送格式正确的 HTTP 请求来进行 API 调用 `http://<privateIpAddress>:<port>/`

检索 JWT 签名密钥

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 从列表中选择您的集群。
3. 在集群配置详细信息中，找到 “调度程序身份验证” 部分。
4. 注意 JSON 网络令牌 (JWT) 密钥 ARN 和版本。
5. 使用从 Sec AWS CLI rets Manager 中检索签名密钥：

```
aws secretsmanager get-secret-value --secret-
id arn:aws:secretsmanager:region:account:secret:name --version-id version
```

生成 JWT 令牌

1. 使用以下必填声明创建 JWT：

- exp— 自 1970 年以来 JWT 的到期时间（以秒为单位）
- iat— 自 1970 年以来的当前时间（以秒为单位）
- sun— 用于身份验证的用户名
- uid— POSIX 用户 ID
- gid— POSIX 群组 ID
- id— 其他 POSIX 身份属性
  - gecos— 用户评论字段，通常用于存储人类可读的名称
  - dir— 用户的主目录
  - shell— 用户的默认外壳
  - gids— 用户所在的其他 POSIX 组 IDs 的列表

2. 使用从 Secrets Manager 中检索到的签名密钥对 JWT 进行签名。
3. 为令牌设置适当的到期时间。

> **Note**
>
> 作为sun索赔的替代方案，您可以提供以下任一信息：
>
> - username
> - 您通过userclaimfield中的定义的自定义字段名称 AuthAltParameters Slurm custom settings
> - id索赔中的一个name字段

对 API 请求进行身份验证

1. 使用以下方法之一将 JWT 令牌包含在您的 HTTP 请求中：
   - 不记名代币-添加Authorization: Bearer *<jwt>*标题
   - Slurm 标题 — 添加标题 X-SLURM-USER-TOKEN: *<jwt>*
2. 向 REST API 端点发出 HTTP 请求：

   以下是使用 curl 和Authorized: Bearer标头访问 /ping API 的示例。

```
curl -X GET -H "Authorization: Bearer <jwt>" \
```

```
        http://<privateIpAddress>:6820/slurm/v0.0.43/ping
```

JWT 生成示例

获取 AWS PCS 集群 JWT 签名密钥并将其存储为本地文件。将 aw s-region 、 secret-arn 和密钥版本的值替换为适合您的集群的值。

```
#!/bin/bash
SECRET_KEY=$(aws secretsmanager get-secret-value \
  --region aws-region \
  --secret-id secret-arn \
  --version-stage secret-version \
  --query 'SecretString' \
  --output text)
echo "$SECRET_KEY" | base64 --decode > jwt.key
```

此 Python 示例说明了如何使用签名密钥生成 JWT 令牌：

```
#!/usr/bin/env python3

import sys
import os
import pprint
import json
import time
from datetime import datetime, timedelta, timezone
from jwt import JWT
from jwt.jwa import HS256
from jwt.jwk import jwk_from_dict
from jwt.utils import b64decode,b64encode
if len(sys.argv) != 3:
    sys.exit("Usage: gen_jwt.py [jwt_key_file] [expiration_time_seconds]")
SIGNING_KEY = sys.argv[1]
EXPIRATION_TIME = int(sys.argv[2])
with open(SIGNING_KEY, "rb") as f:
    priv_key = f.read()
signing_key = jwk_from_dict({
    'kty': 'oct',
    'k': b64encode(priv_key)
})
message = {
    "exp": int(time.time() + EXPIRATION_TIME),
```

```
    "iat": int(time.time()),
    "sun": "ec2-user",
    "uid": 1000,
    "gid": 1000,
    "id": {
        "gecos": "EC2 User",
        "dir": "/home/ec2-user",
        "gids": [1000],
        "shell": "/bin/bash"
    }
}
a = JWT()
compact_jws = a.encode(message, signing_key, alg='HS256')
print(compact_jws)
```

该脚本会将 JWT 打印到屏幕上。

```
abcdefgtjwttoken...
```

在 PCS 中使用 Slurm REST API 进行作业管理 AWS

Slurm REST API 概述

Slurm REST API 通过 HTTP 请求提供对集群管理功能的编程访问。了解这些关键特征将有助于您有效地将 API 与 AWS PCS 配合使用：

- 访问协议：API 使用 HTTP（不是 HTTPS）在集群的私有网络内进行通信。
- 连接详情：使用集群的私有 IP 地址和`slurmrestd`端口（通常为 6820）访问 API。完整的基本 URL 格式为`http://<privateIpAddress>:6820`。
- API 版本控制：API 版本与你的 Slurm 安装相对应。对于 Slurm 25.05，请使用 v0.0.43 版本。版本号会随着每个 Slurm 版本而变化。您可以在 Slurm 发行说明中找到当前支持的 API 版本。
- 网址结构：Slurm REST API 的网址结构是。`http://<privateIpAddress>:<port>/<api-version>/<endpoint>`REST API 端点的详细使用信息可以在 Slurm 文档中找到。

有关使用 Slurm REST API 的具体信息，请参阅 REST 客户端 Sl urm 文档。

先决条件

在使用 Slurm REST API 之前，请确保您已具备以下条件：

- 集群配置：启用 Slurm 25.05+ 且启用 REST API 的 AWS PCS 集群。
- 身份验证：具有正确用户身份声明的有效 JWT 令牌。
- 网络访问：通过允许端口 6820 的安全组在集群的 VPC 内进行连接。

过程

使用 REST API 提交作业

1. 使用所需参数创建任务提交请求：

```
{
  "job": {
    "name": "my-job",
    "partition": "compute",
    "nodes": 1,
    "tasks": 1,
    "script": "#!/bin/bash\necho 'Hello from Slurm REST API'",
    "environment": ["PATH=/usr/local/bin:/usr/bin:/bin"]
  }
}
```

2. 使用 HTTP POST 请求提交任务：

```
curl -X POST \
  -H "Authorization: Bearer <jwt>" \
  -H "Content-Type: application/json" \
  -d '<job-json>' \
  https://<privateIpAddress>:6820/slurm/v0.0.43/job/submit
```

3. 出于监控目的，请记下响应中返回的任务 ID。

监控作业状态

1. 获取有关特定工作的信息：

```
curl -X GET -H "Authorization: Bearer <jwt>" \
    https://<privateIpAddress>:6820/slurm/v0.0.43/job/<job-id>
```

2. 列出经过身份验证的用户的所有作业：

```
curl -X GET -H "Authorization: Bearer <jwt>" \
    https://<privateIpAddress>:6820/slurm/v0.0.43/jobs
```

取消作业

- 发送 DELETE 请求以取消特定任务：

```
curl -X DELETE -H "Authorization: Bearer <jwt>" \
    https://<privateIpAddress>:6820/slurm/v0.0.43/job/<job-id>
```

PCS 中的 Slurm REST API 常见问题解答 AWS

本节回答了有关 PCS 中的 Slurm REST API 的 AWS 常见问题。

什么是 Slurm REST API？

Slurm REST API 是一个 HTTP 接口，允许你以编程方式与 Slurm 工作负载管理器进行交互。您可以使用标准 HTTP 方法（例如 GET、POST 和 DELETE）来提交作业、监控集群状态和管理资源，而无需命令行访问集群。

我可以使用由生成的代币**`scontrol token`**吗？

不是，标准`scontrol token`输出与 AWS PCS 不兼容。PCS Slurm REST API 需要包含包含特定身份声明的丰富的 JWT 令牌，包括用户名 (sun)、POSIX 用户 ID (uid) 和群组 ()。 IDs `gids`标准 Slurm 代币缺少这些必需的声明，因此会被 API 拒绝。

我能否从我的 VPC 外部访问 API？

不可以，只有在您的 VPC 中使用 Slurm 控制器的私有 IP 地址才能访问 REST API 终端节点。要启用外部访问，请实施诸如带有 VPC Link 的 Application Load Balancer、API Gateway 之类的 AWS 服务，或者建立 VPC 对等或 VPN 连接以实现安全连接。

为什么 API 使用 HTTP 而不是 HTTPS？

Slurm REST API 旨在成为集群专用网络中的内部终端节点。对于需要加密的生产部署，您可以在架构中更高级别实现 SSL/TLS 终止，例如通过 API 网关、负载均衡器或反向代理。

如何控制对 REST API 的访问权限？

配置集群的安全组规则，以限制对 Slurm 控制器上的 6820 端口的访问。将入站规则设置为仅允许来自可信 IP 范围或 VPC 内特定来源的连接，从而阻止对 API 终端节点的未经授权的访问。

如何轮换 JWT 签名密钥？

将您的集群置于没有活动实例的维护模式，然后通过 Secrets Manager 启动 AWS 密钥轮换。轮换完成后，重新启用队列。所有现有 JWT 令牌都将失效，必须使用 Secrets Manager 中的新签名密钥重新生成。

我需要启用 Slurm 记账功能才能使用 REST API 吗？

不，作业提交和监控等基本 REST API 操作不需要 Slurm 记账。但是，整个`/slurmdb`端点都需要记账才能处于活动状态。

哪些第三方工具可以与 AWS PCS REST API 配合使用？

许多现有的 Slurm REST API 客户端都应使用 AWS PCS，包括适用于 Prometheus 的 Slurm Exporter，以及遵循标准 Slurm REST API SlurmWeb 格式的自定义应用程序。但是，依赖身份验证`scontrol token`的工具需要修改才能满足 AWS PCS JWT 的要求。

使用 REST API 会产生任何额外费用吗？

不，启用或使用 Slurm REST API 功能无需支付额外费用。您只需像往常一样为底层群集资源付费。

如何对 REST API 进行故障排除？

- 网络连接问题

  如果您无法访问 API 终端节点，则在向集群控制器发出 HTTP 请求时，您将看到连接超时或 “连接被拒绝” 错误。

  操作方法：验证您的客户端位于同一 VPC 中或网络路由正确，并确认您的安全组允许来自您的源 IP 或子网的 6820 端口上的 HTTP 流量。

- Slurm REST 身份验证问题

  如果您的 JWT 令牌无效、过期或签名不正确，API 请求将在响应的错误字段中返回 “协议身份验证错误”。

  错误消息示例：

```
{
"errors": [
    {
    "description": "Batch job submission failed",
    "error_number": 1007,
    "error": "Protocol authentication error",
    "source": "slurm_submit_batch_job()"
```

```
    }
  ]
}
```

操作方法：检查您的 JWT 令牌的格式是否正确、未过期，并使用来自 Secrets Manager 的正确密钥进行签名。验证令牌的格式是否正确且包含所需的声明，以及您使用的身份验证标头格式是否正确。

- 提交后 Job 无法运行

  如果您的 JWT 令牌有效但包含不正确的内部结构或内容，则任务可能已进入带有原因代码的 pause PD d () 状态。`JobAdminHeadscontrol show job <job-id>`用于检查作业 — 你会看到`JobState=PENDING, Reason=JobHeldAdmin`，和`SystemComment=slurm_cred_create failure, holding job`。

  怎么做：根本原因可能是 JWT 中的值错误。根据PCS文档，验证令牌的结构是否正确，并包含所需的声明。

- 工作目录权限问题

  如果您的 JWT 中指定的用户身份缺少对作业工作目录的写入权限，则作业将因权限错误而失败，类似于`sbatch --chdir`使用无法访问的目录。

  操作方法：确保您的 JWT 令牌中指定的用户对作业的工作目录具有相应的权限。

- 还在遇到问题吗？

  1. 查看 SchedMD 关于 REST API 规范的文档。
  2. 查看 Slurm 控制器日志，了解有关错误的更多详细信息（有关更多详细信息，计划程序在 PCS 中 AWS 登录请参阅）。

在 PCS 中使用 Slurm 重启计算节点 AWS

AWS PCS 支持 Slurm 的原`scontrol reboot`生命令。使用此命令可在不更换 EC2 实例的情况下重启计算节点。其他重启方法（Amazon EC2 控制台 AWS CLI、自动补丁或系统维护）会导致 AWS PCS 认为 EC2 实例运行状况不佳并替换它。

重启 Slurm 的好处

Slurm 重启为集群维护提供了多种优势：

- 保留容量-避免将容量受限的 EC2 实例丢给其他客户。

- 降低成本 — 消除不必要的实例更换周期和对闲置节点的持续计费。
- 更快的恢复 — 与更换实例相比，没有配置延迟。
- 操作灵活性 — 清除内存泄漏、删除临时文件以及将节点从降级状态中恢复。

何时使用 Slurm 重启

在常见的操作维护场景中使用 Slurm 重启：

- 故障排除-解决性能问题或进程无响应，尤其是 GPU 节点。
- 资源清理-清除影响作业性能的内存泄漏/tmp、临时文件或卡住的进程。
- 恢复-在要求更换完整节点之前，将节点从挂起或降级状态中恢复。

限制

- 只有 Slurm 管理员用户（root 用户）可以执行重启命令。
- 重启支持`scontrol reboot`仅限于。
- RebootProgram 不支持配置。
- 没有控制台界面 — 仅限命令行。

主题

在 PCS 中使用 Slurm 重启计算节点 AWS

使用 Slurm 的本机重启命令来解决性能问题、清除资源问题或从降级状态中恢复，而不会损失 EC2 实例容量。

先决条件

- Slurm 管理员权限（根用户访问权限）
- 访问 AWS PCS 集群中的登录节点

过程

1. 通过 EC2 控制台连接到登录节点。

   a. 在 EC2 控制台中，选择 Instances (实例)。

   b. 选择您的登录节点实例。

   c. 选择连接。

2. 使用`sinfo`或标识目标计算节点的名称`scontrol show node`。

```
sinfo
# or
scontrol show node
```

3. 使用以下选项之一执行重启命令：

> **Warning**
>
> 不要与`scontrol reboot`命令`nextstate=DOWN`一起使用。此参数将节点标记为运行状况不佳并触发实例替换。

   - 基本重启（等待节点空闲）：

```
scontrol reboot nodename
```

   - 立即重启（耗尽节点并在任务完成后重新启动）：

```
scontrol reboot ASAP nodename
```

   - 重启的原因是：

```
scontrol reboot ASAP reason="troubleshooting" nodename
```

   - 在恢复状态下重新启动：

```
scontrol reboot ASAP nextstate=RESUME nodename
```

4. 使用监控重启进度`scontrol show node`。

```
scontrol show node nodename
```

5. 验证节点在重启完成后是否恢复服务。

取消 AWS PCS 中待重启的操作

取消待定重启，以避免在问题得到解决或不再需要重启时出现不必要的停机。

先决条件

- Slurm 管理员权限
- 节点必须处于待重启状态（显示 “重启已发出” 状态）
- 访问登录节点以执行命令

过程

1. Connect 连接到登录节点。
2. 使用验证节点是否处于待重启状态`scontrol show node`。

```
scontrol show node nodename
```

在节点状态中查找 “已发出重启通知”。

3. 执行取消命令。

```
scontrol cancel_reboot nodename
```

4. 验证是否取消重启并且节点状态恢复正常。

```
scontrol show node nodename
```

PCS 中的 Slurm 重启常见问题解答 AWS

查找有关在 PCS 中 AWS 使用 Slurm 重启的常见问题的答案。

什么是 Slurm 重启支持？

支持原生 Slurm 命令`scontrol reboot`。使用此命令无需自动更换实例即可重启计算节点，这样可以保留 EC2 实例容量并降低运营成本。

谁可以使用 Slurm 重启命令？

只有 Slurm 管理员用户（root 用户）可以执行重启命令。尝试使用的普通用户`scontrol reboot`将收到来自 Slurm 的权限被拒绝的错误，而不会影响节点。

在重启期间运行作业会怎样？

默认情况下，任务会在重新启动之前正常完成。使用 ASAP 选项，节点会被耗尽以防止新作业，并且在当前任务完成后重新启动。可以取消任务或重新排队以便立即重启。

这与 EC2 控制台重启有何不同？

Slurm 重启可保留 EC2 实例并避免替换，而 EC2 控制台重启会触发 PCS 更换实例，因为在重启过程中运行状况检查失败。

我可以配置自定义重启脚本吗？

不是，初始版本不支持 RebootProgram 配置。该功能使用标准的 Slurm 重启行为，不支持自定义脚本。

Slurm 重启需要多长时间？

重启时间因实例类型、客户启动流程、AMI 配置以及是否需要先完成任务而异。该过程包括等待任务完成、物理重启、运行状况检查和 slurmd 守护程序注册。

我能看到重启的历史吗？

重启事件记录在 Slurm 日志（slurmctld 和 slurmd）中，可以通过这些日志进行监控。 CloudWatch 节点状态中的原因字段显示了该过程中的重启原因。

如果节点在重启期间卡住了怎么办？

如果某个节点未完成其中的重启过程 ResumeTimeout，则该节点将被标记为 DOWN。检查 CloudWatch 日志中是否存在错误，验证网络连接，并检查 slurmd 日志。如果问题仍然存在，请联系 AWS Support。

我能否同时重启多个节点？

是的，你可以在 reboot 命令中指定多个节点：

```
scontrol reboot ASAP node1,node2,node3
```

如何在不等待任务完成的情况下重启节点？

要在遇到问题节点影响多节点作业、性能严重下降或 GPU 行为不稳定等问题时立即重启节点，您有两种选择：

- 取消并重新启动-首先，使用取消受影响的作业`scancel <job_id>`，然后使用启动立即重启`scontrol reboot ASAP <nodename>`。正在运行的作业将被终止，需要在节点恢复后重新提交。
- D@@ rain and Requeue（影响较小）— 首先使用启动引流并重启`scontrol reboot ASAP <nodename>`，然后使用重新排队受影响的作业。`scontrol requeue <job_id>`这会使任务恢复到待处理状态，而不是取消它们。

如果我指定 nextState=down 会发生什么？

如果您指定`nextstate=DOWN`，则在重启并触发实例替换后，该节点将被标记为运行状况不佳。为避免替换实例，请不要指定 nextstate 或使用`nextstate=RESUME`。

其他资源

- 有关基本的重新启动过程，请参见在 PCS 中使用 Slurm 重启计算节点 AWS。
- 有关重新启动问题的疑难解答，请参阅对 PCS 中的 Slurm 重启问题进行故障排除 AWS。
- 有关 Slurm 重启文档，请参阅 S lur m scontrol 文档。

对 PCS 中的 Slurm 重启问题进行故障排除 AWS

遇到节点重启问题时，请先使用检查节点状态`scontrol show node nodename`。然后检查 Slurm（slurmctld 和 slurmd）和 CloudWatch 系统日志的日志，以确定潜在的错误。

要进行基本的故障排除，请验证网络连接，检查安全组设置，并确保重启后所有必需的服务都在运行。如果完成基本故障排除步骤后问题仍然存在，请联系 Supp AWS ort。联系支持人员时，请提供相关的日志摘录、节点状态信息和重启尝试的时间表，以帮助加快解决过程。

其他资源

- 有关使用监控 AWS PCS 实例的信息 CloudWatch，请参阅使用 Amazon 监控 AWS PCS 实例 CloudWatch。
- 有关一般故障排除，请参阅对 AWS 并行计算服务中的问题进行故障排除。
- 有关 Slurm 文档，请参阅 Slur m 故障排除指南。

在 PCS 中配置自定义 Slurm 设置 AWS

使用自定义 Slurm 设置跨集群、队列和计算节点组资源配置其他 Slurm 参数。此版本增加了对队列资源上的 Slurm 设置的支持，从而提供了对特定分区的行为的精细控制。

自定义 Slurm 设置的好处

自定义 Slurm 设置可对 AWS 基于 PC 的 HPC 环境进行复杂的控制。您可以通过 quality-of-service配置和抢占策略实施详细记账、实施访问控制并优化工作负载执行。这些功能可确保关键作业获得必要的资源，同时保持集群的高效利用率。无论您是管理 GPU 加速的工作负载、实施公平共享调度还是控制作业生命周期，自定义设置都能帮助您的 HPC 基础架构与运营要求和研究目标保持一致。

配置自定义设置

自定义 Slurm 设置可以通过 AWS 控制台、CLI 进行配置，也可以在资源创建 SDKs 期间进行配置，也可以稍后通过更新操作进行修改。

AWS 管理控制台

在创建或编辑页面中导航到任何资源类型（群集、队列或计算节点组）的其他调度器设置。

添加新设置

1. 选择 “添加新设置”。
2. 从下拉列表中选择一个参数名称（其中包括简短的参数描述）。
3. 提供相应的值。

取消设置自定义设置

1. 选择相关 parameter/value 配对旁边的 “移除”。
2. 创建或更新资源。

AWS CLI

要对自定义设置进行编程管理，请在创建或更新操作中使用该`SlurmCustomSettings`字段。

Example— 更新群集上的Prolog参数

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
```

```
'SlurmCustomSettings=[{parameterName=Prolog,parameterValue="/path/to/prolog.sh"}]'
```

Example— 将队列设置为集群Default上的队列

```
aws pcs update-queue \
    --cluster-identifier my-cluster \
    --queue-identifier my-queue \
    --slurm-configuration
 'SlurmCustomSettings=[{parameterName=Default,parameterValue=YES}]'
```

Example— 在计算节点组Features上设置自定义

```
aws pcs update-compute-node-group \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-cng-1 \
    --slurm-configuration \
    'SlurmCustomSettings=[{parameterName=Features,parameterValue="gpu,nvme"}]'
```

验证和错误处理

AWS PCS 为自定义 Slurm 设置实施了多层验证流程。在创建和更新操作期间，我们会执行同步验证，包括：

- 字段级检查：我们会验证各个设置是否正确的数据类型、允许的值和格式要求。例如，我们确保时间值采用正确的 Slurm 格式，布尔值使用公认的 Slurm 布尔表示形式。
- 上下文感知验证：某些设置是根据更广泛的配置上下文进行检查的。例如，某些参数仅在启用 Slurm 记账时才有效。
- 设置间的一致性：我们验证互斥选项没有设置在一起，并且相互依赖的设置配置正确。

如果验证失败，您将收到`ValidationException`包含特定错误代码（例如 InvalidInput）、描述问题的清晰错误消息以及无效字段及其相应错误详细信息的列表。

虽然在初始验证过程中发现了许多问题，但设置之间的一些复杂交互只有在应用配置时才会显现出来。在这种情况下，操作将失败并显示一条提示性错误消息，并且所有部分更改都将被回退。

限制

AWS PCS 采用允许名单方法来保护服务安全和操作稳定性。可能危及服务帐户安全或干扰托管服务功能的设置受到限制。但是，我们会持续评估客户需求，并可以根据客户反馈增加对其他设置的支持。

主题

PCS 集群的自定义 Slurm 设置 AWS

集群级别支持以下自定义 Slurm 设置：

- AccountingStorageEnforce
- AccountingStorageTRES
- AccountingStoreFlags
- DefMemPerCPU
- Epilog
- EnforcePartLimits
- FairShareDampeningFactor
- FirstJobId
- HealthCheckInterval
- HealthCheckNodeState
- HealthCheckProgram
- JobRequeue
- LaunchParameters
- Licenses
- MinJobAge

> **Note**
>
> AWS PCS 支持的最小值为 5 秒`MinJobAge`。

- OverTimeLimit
- PreemptExemptTime
- PreemptMode

- PreemptParameters
- PreemptType
- PriorityCalcPeriod
- PriorityDecayHalfLife
- PriorityFavorSmall
- PriorityFlags
- PriorityMaxAge
- PriorityUsageResetPeriod
- PriorityWeightAge
- PriorityWeightAssoc
- PriorityWeightFairshare
- PriorityWeightJobSize
- PriorityWeightPartition
- PriorityWeightQOS
- PriorityWeightTRES
- Prolog
- PrologFlags
- PropagatePrioProcess
- PropagateResourceLimits
- PropagateResourceLimitsExcept
- RequeueExit
- RequeueExitHold
- SchedulerParameters
- SelectTypeParameters
- SrunPortRange
- TaskEpilog
- TaskPluginParam
- TaskProlog
- TrackWCKey
- UnkillableStepProgram

- UnkillableStepTimeout

AWS PCS 计算节点组的自定义 Slurm 设置

计算节点组级别支持以下自定义 Slurm 设置：

- CpuSpecList
- Features
- MemSpecLimit
- RealMemory
- Weight

PCS 队列的自定义 Slurm 设置 AWS

队列级别支持以下自定义 Slurm 设置：

- AllowAccounts
- AllowQoS
- Default
- DefaultTime
- DenyAccounts
- DenyQoS
- ExclusiveUser
- GraceTime
- MaxTime
- OverSubscribe
- OverTimeLimit
- PreemptMode
- PriorityJobFactor
- PriorityTier
- QOS
- TRESBillingWeights

对 PCS 中的自定义 Slurm 设置进行故障排除 AWS

如果您在使用 Slurm 自定义设置创建或更新 AWS PCS 资源时遇到错误，则可以使用日志记录来诊断和解决问题。

对不兼容的 Slurm 自定义设置进行故障排除

问题：在执行群集、计算节点组或队列操作时，您会收到类似以下内容的错误消息：

```
{OPERATION} failed. The Slurm custom settings of the cluster might be incompatible.
 Check the settings and try again.
```

以下操作可能会出现此错误：

- CreateCluster
- CreateComputeNodeGroup
- UpdateComputeNodeGroup
- CreateQueue
- UpdateQueue

解决方案：启用日志记录功能以了解具体问题并对不兼容的设置进行故障排除。

要解决不兼容的 Slurm 自定义设置问题

1. 如果集群尚不存在，请创建该集群，或者确保您的现有集群处于可以启用日志记录的状态。
2. 为您的集群启用日志记录。有关详细说明，请参阅AWS PCS 的日志记录和监控。

   > **Note**
   >
   > 创建集群后，即可启用日志记录。

3. 查看日志，找出导致不兼容的特定 Slurm 配置问题。
4. 根据日志信息更正不兼容的自定义设置，然后重试该操作。

有关支持的 Slurm 自定义设置的信息，请参阅：

- PCS 集群的自定义 Slurm 设置 AWS
- AWS PCS 计算节点组的自定义 Slurm 设置

- PCS 队列的自定义 Slurm 设置 AWS

在 PCS 中 AWS 配置自定义 cgroup 设置

Slurm 使用 Linux cgroup 子系统来管理和限制作业资源，包括内存、CPU 内核、设备和交换空间。AWS PCS 允许您在集群创建或更新`SlurmConfiguration`期间通过的`CgroupCustomSettings`属性自定义集群级别的`cgroup.conf`设置。

配置 cgroup 设置

Cgroup 自定义设置可以通过 AWS 控制台、CLI 进行配置，也可以在集群创建 SDKs 期间进行配置，也可以稍后通过更新操作进行修改。

AWS 管理控制台

在群集资源的创建或编辑页面中导航到其他调度器设置。

添加新设置

1. 选择 “添加新设置”。
2. 从下拉列表中选择一个参数名称（其中包括简短的参数描述）。
3. 提供相应的值。

取消设置自定义设置

1. 选择相关 parameter/value 配对旁边的 “移除”。
2. 创建或更新资源。

AWS CLI

要对 cgroup 设置进行编程管理，请使用创建或更新集群操作中的`CgroupCustomSettings`字段。

Example— 在集群ConstrainRAMSpace上设置

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'CgroupCustomSettings=[{parameterName=ConstrainRAMSpace,parameterValue="yes"}]'
```

集群支持的 cgroup 设置

集群级别支持以下自定义 cgroup 设置：

- AllowedRAMSpace
- AllowedSwapSpace
- ConstrainCores
- ConstrainDevices
- ConstrainRAMSpace
- ConstrainSwapSpace
- IgnoreSystemd
- MaxRAMPercent
- MaxSwapPercent
- MinRAMSpace
- SignalChildrenProcesses

在 PCS 中配置自定义 slurmdBD 设置 AWS

Slurm 的数据库守护程序 (slurmdbd) 管理会计数据、数据保留策略和隐私控制。 AWS PCS 允许您在集群创建或更新`SlurmConfiguration`期间通过的`SlurmdbdCustomSettings`属性自定义集群级别的`slurmdbd.conf`设置。

配置 slurmdbd 设置

Slurmdbd 自定义设置可以通过控制台 AWS 、CLI 或 SDKs 在集群创建期间进行配置，也可以稍后通过更新操作进行修改。

AWS 管理控制台

在群集资源的创建或编辑页面中导航到其他调度器设置。

添加新设置

1. 选择 “添加新设置”。
2. 从下拉列表中选择一个参数名称（其中包括简短的参数描述）。
3. 提供相应的值。

取消设置自定义设置

1. 选择相关 parameter/value 配对旁边的 “移除”。
2. 创建或更新资源。

AWS CLI

要对 slurmdbd 设置进行编程管理，请使用创建或更新`SlurmdbdCustomSettings`集群操作中的字段。

Example— 在集群TrackWCKey上设置

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'SlurmdbdCustomSettings=[{parameterName=TrackWCKey,parameterValue="yes"}]'
```

集群支持的 slurmdbd 设置

集群级别支持以下自定义 slurmdbd 设置：

- AllowNoDefAcct
- AllResourcesAbsolute
- CommitDelay
- DefaultQOS
- MaxQueryTimeRange
- Parameters
- PrivateData
- PurgeEventAfter
- PurgeJobAfter
- PurgeResvAfter
- PurgeStepAfter
- PurgeSuspendAfter
- PurgeTXNAfter
- PurgeUsageAfter

- TrackWCKey

使用 SPANK 插 AWS 件在 PCS 上扩展 Slurm 功能

使用 SPANK（适用于节点和作业 Kontrol 的 Slurm 插件架构）插件在 PCS 集群上启动和执行任务期间扩展和修改 Slurm 的行为。 AWS SPANK 插件提供了一个用于拦截和修改任务启动阶段的通用接口。

在您的计算节点 AMI 上安装 SPANK 插件，然后对其进行配置，以根据您的工作负载要求自定义 Slurm 集群的行为。有关 SPANK 的更多信息，请参阅 SchedMD 网站上的 SPANK 文档。

目录

在 PCS 上 AWS 安装 SPANK 插件

按照插件的文档在您的 AMI 上安装 SPANK 插件。

在你的集群上为特定 Slurm 版本编译 SPANK 插件。 AWS PCS 提供的 Slurm 安装程序将 Slurm 存储在中。`/opt/aws/pcs/scheduler/slurm-version`编译插件时，请指定 Slurm 版本。

以下示例说明如何为某些插件指定 Slurm 版本：

```
export CFLAGS="-I/opt/aws/pcs/scheduler/slurm-version/include"
```

如果您的 AMI 中有多个 Slurm 版本，请为每个版本编译插件。将已编译的插件存储在版本控制的文件夹中。

以下示例说明如何为某些插件指定目标文件夹：

```
export DESTDIR="your-preferred-versioned-path"
```

> **Important**
>
> 插件可能需要不同的变量。请参阅您正在安装的插件的官方文档。

在 PC 上 AWS 配置 SPANK 插件

默认情况下，将配置文件存储在中/etc/aws/pcs/scheduler/slurm-*version*/plugstack.conf.d/。

要将 SPANK 配置存储在其他位置，请将您的位置添加到默认目录中的配置文件中。

以下示例说明如何包含来自其他目录的配置文件：

```
# content of /etc/aws/pcs/scheduler/slurm-version/any-filename.conf
include path-to-your-configuration-folder/*.conf
include path-to-a-second-configuration-folder/*.conf
```

将每个配置存储在专用文件或公共文件中。您可以使用多个配置文件。

以下示例显示了示例配置文件：

```
# content of path-to-your-or-default-config-folder/filename-1.conf
required path-to-plugin-1 arguments
optional path-to-plugin-2 arguments
```

```
# content of path-to-your-or-default-config-folder/filename-2.conf
required path-to-plugin-3 arguments
```

有关如何配置插件的更多信息，请参阅 SchedMD 网站上的 SPANK 配置文档。

Important

设置文件夹权限以防止未经授权更改您的插件配置。

Note

AWS PCS 无法管理你的 SPANK 插件。如果您遇到与插件相关的错误，请查看计算节点上的错误日志。

Note

Slurm 在加载你的 SPANK 配置时错误地记录了类似于以下内容的错误：

```
error: "Include" failed in file /etc/slurm/plugstack.conf line 3
```

您可以忽略该错误。它不会影响 SPANK 插件的工作方式。

有关 PCS 上的 SPANK 插件的 AWS 常见问题

本节介绍有关在 AWS PCS 集群上安装和配置 SPANK 插件的常见问题。

我需要在登录节点和计算节点上都安装 SPANK 插件吗？

某些 SPANK 插件不需要在所有节点上安装；但为了获得更好的兼容性，我们建议您在每个节点上安装所有 SPANK 插件。

在生产环境中使用 SPANK 插件还需要什么额外的配置？

除了示例中显示的基本安装和配置外，生产部署通常还需要额外的设置。基于容器的插件（例如 Pyxis）可能需要您为 Enroot 设置环境变量、启用 PMI（流程管理接口）以及配置容器运行时的权限。有关详细的生产部署要求，请参阅特定插件的文档。

如何解决 SPANK 插件问题？

AWS PCS 不管理 SPANK 插件。检查计算节点上的错误日志以解决问题。

使用 Slurm CLI 过滤器插件在 PCS 中自定义作业提交 AWS

AWS PCS 支持 Slurm CLI 过滤器插件来运行自定义 Lua 脚本，用于验证和修改登录和计算节点上的作业提交参数。有关 CLI 过滤器插件的详细信息，请参阅 SchedMD 网站上的 cli_filter 插件 API 文档。

要求

CLI 过滤器插件需要 Slurm 版本 24.11 或更高版本，并且需要在所有登录和计算节点上部署 Lua 脚本。

> **Important**
>
> 对于 Slurm 版本 24.11 和 25.05，CLI 过滤器插件需要 AWS 使用 PCS Slurm 安装程序（版本 24.11.6-2+ 或 25.05.4-1+）安装 Slurm。有关安装 Slurm 的更多信息，请参阅。第 3 步 — 安装 Slurm

限制和安全注意事项

- 安全实施 — 任何用户都可以轻松绕过 CLI 过滤器插件，并且不得用于安全关键型策略。用户可以通过提供在提交作业时禁用的自定义配置来`CLIFilterPlugins`禁用 CLI 过滤器插件。
- 仅限 Lua 实现 — 支持 Lua 脚本实现。不支持 C 实现。

主题

- 在 PCS 集群上配置 Slurm CLI 过滤器插件 AWS
- 使用 Amazon S3 在 AWS PCS 中部署 CLI 筛选插件脚本
- 翻译 Slurm Job 提交插件脚本以便在 PCS 中使用 CLI 过滤器插件 AWS
- 有关 PCS 中的 Slurm CLI 过滤器插件的常见问题 AWS
- 排除 PCS 中的 Slurm CLI 过滤器插件问题 AWS

在 PCS 集群上配置 Slurm CLI 过滤器插件 AWS

在创建新的 AWS PCS 集群时配置 CLI 筛选插件。您可以使用更新 API 或控制台在现有集群上启用或禁用 CLI 筛选器插件，而无需重新创建集群。

先决条件

在配置 CLI 筛选器插件之前，请完成以下任务：

- 编写并测试实现 CLI 过滤器插件 API 的 Lua 脚本
- 准确命名你的 Lua 脚本 `cli_filter.lua`
- 选择一种将脚本部署到所有集群实例（AMI、S3 或文件系统）的方法
- 确认你使用的是 Slurm 版本 24.11 或更高版本

在新集群上启用 CLI 筛选插件

AWS PCS console

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 在导航窗格中，选择集群。
3. 选择创建集群。
4. 选择 Slurm 的有效版本（版本 24.11 或更高版本）。
5. 在 “日程安排器设置” 下，展开 “其他调度程序设置”。
6. 添加新的 Slurm 自定义设置，参数名称设置为`CliFilterPlugins`，参数值设置为。`cli_filter/lua`
7. 完成剩余的集群配置，然后选择创建集群。

AWS PCS API

在调用 `CreateCluster` API 操作时提供`slurmCustomSettings`配置。将 to `CliFilterPlugins` 和 `parameterName` to 设置`parameterValue`为`cli_filter/lua`。有关更多信息，请参阅 AWS PCS API 参考CreateCluster中的。

以下示例使用调 AWS CLI 用 `CreateCluster` API 操作。自定义设置`CliFilterPlugins=cli_filter/lua`启用 CLI 过滤器插件。

```
aws pcs create-cluster --cluster-name cluster-name \
--scheduler type=SLURM,version=24.11 \
--size SMALL \
--networking subnetIds=cluster-subnet-id,securityGroupIds=cluster-security-group-id
 \
--slurm-configuration \
'slurmCustomSettings=[{parameterName=CliFilterPlugins,parameterValue="cli_filter/
lua"}]'
```

部署 CLI 过滤器插件脚本

将 CLI 筛选插件脚本部署到您的集群

1. 确保计算节点组中所有 AMIs 使用的 Slurm 都通过 AWS PCS Slurm 安装程序安装 Slurm。

> **Note**
>
> 如果您对所有计算节点组使用 AWS PCS 示例 AMI，请跳过此步骤。Slurm 已经安装好了。

2. 将您的`cli_filter.lua`脚本部署到`/etc/aws/pcs/scheduler/slurm-<version>/cli_filter.lua`集群中的所有实例。

   例如，对于 Slurm 版本 24.11：

   ```
   /etc/aws/pcs/scheduler/slurm-24.11/cli_filter.lua
   ```

3. 使用准备好的启动所有登录和计算节点 AMIs。
4. 测试作业提交以验证 CLI 过滤器插件是否正确执行。

在现有集群上启用或禁用 CLI 筛选插件

无需重建基础架构，即可在现有集群上启用或禁用 CLI 筛选器插件。有关更多信息，请参阅 在 AWS PCS 中更新集群。

AWS PCS console

1. 打开 AWS PCS 控制台，网址为https://console.aws.amazon.com/pcs/。
2. 在导航窗格中，选择集群。
3. 选择要更新的集群。
4. 选择编辑操作。
5. 在 “编辑集群” 页面的 “其他调度程序设置” 下：
   - 启用 CLI 过滤器插件：添加新的 Slurm 自定义设置，参数名称设置为`CliFilterPlugins`，参数值设置为。`cli_filter/lua`
   - 要禁用 CLI 过滤器插件，请执行以下操作：删除现有`CliFilterPlugins`设置。
6. 选择更新集群以提交更改。
7. 监控集群状态，在更新过程中显示为 “正在更新”，更新完成后显示为 “活动”。

AWS PCS API

使用 `UpdateCluster` API 操作启用或禁用 CLI 过滤器插件。有关更多信息，请参阅 AWS PCS API 参考UpdateCluster中的。

要在现有集群上启用 CLI 筛选插件，请执行以下操作：

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'slurmCustomSettings=[{parameterName=CliFilterPlugins,parameterValue="cli_filter/
lua"}]'
```

要在现有集群上禁用 CLI 筛选插件，请执行以下操作：

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'slurmCustomSettings=[]'
```

预期结果

完成配置后：

- 您的集群是在开启 CLI 筛选插件的情况下创建的
- Job 提交会在到达 Slurm 控制器之前触发你的自定义验证逻辑
- 不合规的任务会被拒绝，并显示您的自定义错误消息
- 合规作业通过 Slurm 调度器正常进行

问题排查

任何节点上都缺少 CLI 过滤器插件脚本

症状：由于插件加载错误，Job 提交立即失败。

可能的原因：脚本未部署到所有实例，或者文件路径或名称不正确。

解决方案：验证脚本是否存在于所有登录和计算节点上的正确路径上，并具有确切的文件名`cli_filter.lua`。

CLI 筛选插件配置无效

症状：集群创建失败并出现验证错误。

可能的原因：`CliFilterPlugins`参数未设置为`cli_filter/lua`格式。

分辨率：在中使用精确`cli_filter/lua`的参数值`slurmCustomSettings`。

使用 Amazon S3 在 AWS PCS 中部署 CLI 筛选插件脚本

如果您想在不进行重建的情况下更新实时集群上的作业提交逻辑，请使用 S3 部署 CLI 筛选器插件脚本 AMIs。此方法在实例启动期间使用用户数据从 S3 下载脚本。

先决条件

在使用 S3 部署脚本之前，请完成以下任务：

- 使用 CLI 过滤器插件 Lua 脚本创建 S3 存储桶
- 配置 IAM 实例配置文件，使其具有对 S3 存储桶的读取权限
- 设置 S3 VPC 网关终端节点，无需互联网即可直接访问
- 准备要从 S3 下载的用户数据脚本

使用 S3 部署 CLI 过滤器插件脚本

1. 将您的`cli_filter.lua`脚本上传到 S3 存储桶。
2. 使用存储桶的 S3 读取权限配置您的 IAM 实例配置文件。
3. 在启动模板用户数据中添加 shell 代码以下载脚本：

```
aws s3 cp s3://my-bucket/cli_filter.lua /etc/aws/pcs/scheduler/slurm-24.11/
cli_filter.lua
chmod 644 /etc/aws/pcs/scheduler/slurm-24.11/cli_filter.lua
```

4. 使用更新的启动模板部署计算节点组。
5. 测试作业提交以验证脚本功能。

预期结果

完成 S3 部署后：

- CLI 筛选插件脚本会在启动期间自动下载到所有实例
- S3 中的脚本更新会反映在新启动的实例上
- Job 提交策略在整个集群中得到一致执行

问题排查

S3 访问被拒

症状：实例启动失败或脚本未下载。

可能的原因：缺少 IAM 权限或 S3 VPC 终端节点。

解决方案：验证 IAM 实例配置文件是否具有`s3:GetObject`权限并且已配置 S3 VPC 终端节点。

翻译 Slurm Job 提交插件脚本以便在 PCS 中使用 CLI 过滤器插件 AWS

当你从其他 Slurm 环境迁移时，将你现有的 Job Submit Plugin Lua 脚本翻译为 CLI 过滤器插件。翻译过程包括更新函数名称和字段访问模式以使用 CLI 过滤器插件 API。

先决条件

在翻译脚本之前，请完成以下任务：

- 查看你现有的 Job Submit 插件 Lua 脚本
- 了解 Job Submit 和 CLI 筛选插件之间的区别 APIs
- 访问 Slurm CLI 过滤器插件文档

将 Job Submit 插件脚本翻译为 CLI 过滤器插件

1. 查看您现有的 Job Submit 插件脚本函数 (`slurm_job_submit`,`slurm_job_modify`)。
2. 确定等效的 CLI 过滤器插件函数：
   - `slurm_job_submit` 变为 `slurm_cli_pre_submit`
   - `slurm_cli_setup_defaults`为默认参数设置添加
   - `slurm_cli_post_submit`为提交后的操作添加
3. 将作业验证逻辑从`job_desc`字段转换为`options`数组访问权限：

- `job_desc.account` 变为 `options["account"]`
- `job_desc.partition` 变为 `options["partition"]`
- `job_desc.features` 变为 `options["constraint"]`

4. 将记录呼叫从更新`slurm.log_user()`为`slurm.log_error()`。
5. 在开发集群上测试已翻译的脚本。
6. 按照标准的 CLI 过滤器插件部署流程部署到您的生产集群。

预期结果

完成翻译后：

- 您的翻译脚本提供了同等的作业提交验证
- 用户看到的错误消息和提示与你最初的 Job Submit 插件类似
- 在迁移到 AWS PCS 期间，将维护作业提交策略

问题排查

脚本翻译错误

症状：Job 提交失败，导致 Lua 执行错误。

可能的原因：翻译后的脚本中的字段访问或函数调用不正确。

解决方案：查看 CLI 筛选插件 API 文档，比较 Job Submit 和 CLI 筛选器接口之间的字段映射。

有关 PCS 中的 Slurm CLI 过滤器插件的常见问题 AWS

查看这些有关 CLI 过滤器插件的常见问题。

CLI 筛选插件和 Job Submit 插件有什么区别？

在作业提交到达控制器之前，CLI 筛选器插件在登录和计算节点上在客户端运行，而 Job Submit Plugin 在作业提交后在控制器上运行服务器端。用户可以绕过 CLI 过滤器插件，但不锁定控制器锁，而 Job Submit 是安全的，但在执行过程中可能会影响集群性能。

AWS PCS 是否支持 Slurm Job 提交插件？

不是，PCS 不支持 Job 提交插 AWS 件。改用 CLI 筛选插件进行作业提交验证和修改。

我可以使用 CLI 过滤器插件进行安全执法吗？

不，CLI 过滤器插件可以被确定的用户绕过，因此不应依赖它来强制执行安全措施。将其用于用户体验改进、默认参数设置和策略指导，而不是安全关键型策略。

为什么脚本必须放在所有计算节点上，而不仅仅是登录节点？

像 Slurm 这样的命令`srun`可以在计算节点上的作业脚本中执行，这也会触发 CLI 过滤器插件的执行。无论在何处执行 Slurm 命令，该脚本都必须可用。

我能否在实时集群上修改 CLI 过滤器插件脚本？

可以，如果您使用 S3 或文件系统部署方法。新实例将获得更新的脚本，但现有实例需要手动或通过您选择的部署方法更新脚本。

我能否在不同的计算节点组上使用不同的 CLI 筛选插件脚本？

是的，但不建议这样做。您可以为不同的计算节点组提供具有不同逻辑的脚本，但您负责管理相互依赖关系并防止逻辑重叠。大多数客户在整个集群中提供一组逻辑。

我能否使用带有 C 实现的 CLI 过滤器插件来代替 Lua？

不支持 C 实现。 AWS PCS 中仅支持 Lua 脚本实现。SchedMD 建议客户在实现 CLI 过滤器插件时使用 Lua 而不是 C，以便于使用。

我能否在现有集群上打开或关闭 CLI 过滤器插件？

是的，您可以使用更新 API 在现有集群上启用或禁用 CLI 筛选器插件，而无需重新创建集群。

排除 PCS 中的 Slurm CLI 过滤器插件问题 AWS

使用此疑难解答信息来解决常见的 CLI 过滤器插件问题。

由于插件加载错误，Job 提交立即失败

症状：提交作业时，用户会收到有关 CLI 过滤器插件丢失或失败的错误消息。

可能的原因：

- 一个或多个节点缺少 CLI 筛选插件脚本
- 脚本文件名不正确（必须完全正确`cli_filter.lua`）
- 脚本部署到错误的目录路径
- 脚本的文件权限不正确

解决方法：

- 验证所有登录节点和计算节点/etc/aws/pcs/scheduler/slurm-<version>/cli_filter.lua上是否存在脚本
- 检查脚本文件名是否正确 cli_filter.lua
- 确保脚本具有可读权限（644 或类似权限）
- 在部署到完整集群之前，在单个登录节点上测试脚本部署

集群创建失败，出现 CLI 筛选插件验证错误

症状：集群创建失败，错误提示CliFilterPlugins参数无效。

可能的原因：

- 中的参数值格式不正确 slurmCustomSettings
- 参数名称或值中有错字

解决方法：

- 使用确切的参数名称：CliFilterPlugins
- 使用精确的参数值：cli_filter/lua
- 验证slurmCustomSettings数组中的 JSON 语法

CLI 过滤器插件脚本已执行，但作业验证无法按预期运行

症状：作业成功提交，但自定义验证逻辑不会触发或产生意外结果。

可能的原因：

- Lua 脚本语法错误
- 字段访问模式不正确（使用 Job Submit 插件语法而不是 CLI 筛选插件）
- 验证条件中的逻辑错误

解决方法：

- 查看 Lua 脚本是否存在语法错误
- 验证字段访问权限使用options["field_name"]格式而不是 job_desc.field_name
- 向调试脚本执行流程添加日志语句
- 先用简单的验证案例测试脚本逻辑

S3 脚本部署失败

症状：实例启动但未从 S3 下载 CLI 筛选器插件脚本。

可能的原因：

- IAM 实例配置文件缺少 S3 读取权限
- 未配置 S3 VPC 终端节点
- 用户数据中的 S3 存储桶或对象路径不正确

解决方法：

- 验证 IAM 实例配置文件是否`s3:GetObject`有权访问您的存储桶
- 配置 S3 VPC 网关终端节点以便直接访问
- 在用户数据脚本中检查 S3 存储桶名称和对象路径
- 查看实例用户数据日志，查看 S3 下载错误

AWS 并行计算服务中的安全性

云安全 AWS 是重中之重。作为 AWS 客户，您可以从专为满足最安全敏感的组织的要求而构建的数据中心和网络架构中受益。

安全是双方共同承担 AWS 的责任。责任共担模式将其描述为云的安全性和云中的安全性：

- 云安全 — AWS 负责保护在云中运行 AWS 服务的基础架构 AWS Cloud。 AWS 还为您提供可以安全使用的服务。作为AWS 合规计划合规计划合规计划合的一部分，第三方审计师定期测试和验证我们安全的有效性。要了解适用于 AWS 并行计算AWS 服务的合规性计划，请参阅按合规计划划分的范围内的合规性计划AWS 服务按合规计划。
- 云端安全-您的责任由您使用的 AWS 服务决定。您还需要对其他因素负责，包括您的数据的敏感性、您公司的要求以及适用的法律法规。

本文档可帮助您了解在使用 AWS PCS 时如何应用分担责任模型。以下主题向您介绍如何配置 AWS PCS 以满足您的安全和合规性目标。您还将学习如何使用其他 AWS 服务来帮助您监控和保护您的 AWS PCS 资源。

主题

AWS 并行计算服务中的数据保护

分 AWS 担责任模型适用于 AWS 并行计算服务中的数据保护。如本模型所述 AWS ，负责保护运行所有内容的全球基础架构 AWS Cloud。您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 AWS 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问

题。有关欧洲数据保护的信息，请参阅 AWS Security Blog 上的 AWS Shared Responsibility Model and GDPR 博客文章。

出于数据保护目的，我们建议您保护 AWS 账户 凭证并使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 用于 SSL/TLS 与 AWS 资源通信。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用设置 API 和用户活动日志 AWS CloudTrail。有关使用 CloudTrail 跟踪捕获 AWS 活动的信息，请参阅《AWS CloudTrail 用户指南》中的使用跟 CloudTrail 踪。
- 使用 AWS 加密解决方案以及其中的所有默认安全控件 AWS 服务。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。
- 如果您在 AWS 通过命令行界面或 API 进行访问时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》https://aws.amazon.com/compliance/fips/。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您 AWS 服务 使用控制台、API 或与 AWS PCS 或其他人合作时 AWS SDKs。 AWS CLI在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供 URL，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

静态加密

使用、、PCS API 或创建 AWS 并行计算服务 (AWS PCS) 集群时 AWS 管理控制台 AWS CLI， AWS 默认情况下会对静态数据启用加密 AWS SDKs。 AWS PCS 使用AWS 拥有的 KMS 密钥对静态数据进行加密。有关更多信息，请参阅《AWS KMS 开发人员指南》中的客户 AWS 密钥和密钥。您也可以使用客户管理的密钥。有关更多信息，请参阅 在 PCS 中使用加密 EBS 卷所必需的 KMS 密钥策略 AWS。

集群密钥存储在 Sec rets Manager 托管的 KMS 密钥中， AWS Secrets Manager 并使用 KMS 密钥进行加密。有关更多信息，请参阅 在 AWS PCS 中使用集群密钥。

在 AWS PCS 集群中，以下数据处于静止状态：

- 调度器状态 — 它包括集群中正在运行的作业和已配置节点的数据。这是 Slurm 保留在你的`StateSaveLocation`定义中的数据。`slurm.conf`有关更多信息，请参阅 Slurm 文档StateSaveLocation中的描述。 AWS 任务完成后，PCS 会删除作业数据。
- 调度程序身份验证密钥 — AWS PCS 使用它来验证集群中的所有调度程序通信。

对于调度程序状态信息， AWS PCS 会在将数据和元数据写入文件系统之前自动对其进行加密。加密文件系统对静态数据使用行业标准的 AES-256 加密算法。

传输中加密

无论您使用 AWS Command Line Interface (AWS CLI) 还是，您与 AWS PCS API 的连接都使用签名版本 4 签名过程的 TLS 加密 AWS SDKs。有关更多信息，请参阅AWS Identity and Access Management 用户指南中的签署 AWS API 请求。 AWS 通过 API 管理您用于连接的安全证书的 IAM 策略的访问控制。

AWS PCS 使用 TLS 来连接其他 AWS 服务。

在 Slurm 集群中，调度器配置了`auth/slurm`身份验证插件，该插件可为所有调度程序通信提供身份验证。Slurm 不为其通信提供应用程序级别的加密，所有流经集群实例的数据都在 EC2 VPC 本地，因此，如果这些实例支持传输中的加密，则需要进行 VPC 加密。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的传输中加密。控制器（在服务帐户中配置）与您账户中的集群节点之间的通信是加密的。

密钥管理

AWS PCS 使用AWS 拥有的 KMS 密钥对数据进行加密。有关更多信息，请参阅《AWS KMS 开发人员指南》中的客户 AWS 密钥和密钥。您也可以使用客户管理的密钥。有关更多信息，请参阅 在 PCS 中使用加密 EBS 卷所必需的 KMS 密钥策略 AWS。

集群密钥存储在 Sec rets Manager 托管的 KMS 密钥中， AWS Secrets Manager 并使用 KMS 密钥进行加密。有关更多信息，请参阅 在 AWS PCS 中使用集群密钥。

互联网络流量隐私

AWS 集群的 PCS 计算资源位于客户账户中的 1 个 VPC 内。因此，集群内的所有内部 AWS PCS 服务流量都留在 AWS 网络内，不会通过互联网传输。用户和 AWS PCS 节点之间的通信可以通过互联网传输，我们建议使用 SSH 或 Systems Manager 连接到节点。有关更多信息，请参阅什么是 AWS Systems Manager？ 在《AWS Systems Manager 用户指南》中。

您还可以使用以下产品将本地网络连接到 AWS：

- AWS Site-to-Site VPN。 有关更多信息，请参阅什么是 AWS Site-to-Site VPN？ 在《AWS Site-to-Site VPN 用户指南》中。
- 一个 AWS Direct Connect。有关更多信息，请参阅什么是 AWS Direct Connect？ 在《AWS Direct Connect 用户指南》中。

您可以访问 AWS PCS API 来执行服务的管理任务。您和您的用户访问 Slurm 端点端口，直接与调度程序进行交互。

加密 API 流量

要访问 AWS PCS API，客户端必须支持传输层安全 (TLS) 1.2 或更高版本。我们要求使用 TLS 1.2，建议使用 TLS 1.3。客户端还必须支持具有完全向前保密（PFS）的密码套件，例如 Ephemeral Diffie-Hellman（DHE）或 Elliptic Curve Diffie-Hellman Ephemeral（ECDHE）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。此外，必须使用访问密钥 ID 和与 IAM 主体关联的秘密访问密钥来对请求进行签名。您还可以使用 AWS Security Token Service (AWS STS) 生成临时安全证书来签署请求。

加密数据流量

通过支持的 EC2 实例访问调度程序终端节点，以及从内部的 ComputeNodeGroup 实例之间，可以对传输中的数据进行 AWS Cloud加密。有关更多信息，请参阅 传输中加密。

在 PCS 中使用加密 EBS 卷所必需的 KMS 密钥策略 AWS

AWS PCS 使用与服务相关的角色将权限委托给其他 AWS 服务人。 AWS PCS 服务相关角色是预定义的，包括 AWS PCS 代表您呼叫他人所需的权限。 AWS 服务 预定义的权限还包括对您的客户托管密钥的访问权限 AWS 托管式密钥 ，但不包括对您的客户托管密钥的访问权限。

本主题介绍在您为 Amazon EBS 加密指定客户托管密钥时，如何设置启动实例所需的密钥策略。

> **Note**
>
> AWS PCS 不需要额外的授权即可使用默认权限 AWS 托管式密钥 来保护您账户中的加密卷。

内容

- 示例 1：允许访问客户托管密钥的关键策略部分
- 示例 2：允许跨账户访问客户托管密钥的关键策略部分
- 在 AWS KMS 控制台中编辑密钥策略

概述

当 AWS PCS 启动实例时，您可以使用以下 AWS KMS keys 方法进行 Amazon EBS 加密：

- AWS 托管式密钥：您的账户中 Amazon EBS 创建、拥有和管理的加密密钥。这是新账户的默认加密密钥。除非您指定客户托管密钥，AWS 托管式密钥 否则 Amazon EBS 将使用进行加密。
- 客户托管密钥：您创建、拥有和管理的自定义加密密钥。有关更多信息，请参阅《AWS Key Management Service 开发人员指南》中的创建 KMS 密钥。

> **Note**
>
> 密钥必须是对称的。Amazon EBS 不支持非对称的客户托管密钥。

在创建加密快照或指定加密卷的启动模板时，或者选择默认启用加密时，您可以配置客户托管密钥。

配置密钥策略

您的 KMS 密钥必须具有密钥策略，允许 AWS PCS 启动使用客户托管密钥加密的 Amazon EBS 卷的实例。

使用本页上的示例配置密钥策略，让 AWS PCS 可以访问您的客户托管密钥。您可以在创建密钥时或以后修改客户托管密钥的密钥策略。

密钥策略必须包含以下声明：

- 一种声明，允许`Principal`元素中指定的 IAM 身份直接使用客户托管密钥。它包括对密钥执行 AWS KMS `Encrypt`、`Decrypt`、`ReEncrypt*`、`GenerateDataKey*`、和`DescribeKey`操作的权限。
- 一种语句，允许`Principal`元素中指定的 IAM 身份使用该`CreateGrant`操作生成授权，这些授权将自己的一部分权限委托给与 AWS KMS 或其他委托人集成的权限。 AWS 服务 这样，他们可以使用密钥代表您创建加密的资源。

在向密钥策略中添加新的政策声明时，请勿更改策略中的任何现有声明。

有关更多信息，请参阅:

- 命令参考中的 create-keyAWS CLI
- 《AWS CLI Command Reference》中的 put-key-policy。
- 在开发者指南中找到密钥 ID 和密钥 ARN AWS Key Management Service
- PCS 的服务相关角色 AWS
- 亚马逊 EBS 用户指南中的亚马逊 EBS 加密
- 《AWS Key Management Service 开发人员指南》中的AWS Key Management Service

示例 1：允许访问客户托管密钥的关键策略部分

将以下政策声明添加到客户托管密钥的密钥策略中。将示例 ARN 替换为服务相关角色的 ARN。AWSServiceRoleForPCS此示例策略授予 AWS PCS 服务相关角色 (AWSServiceRoleForPCS) 使用客户托管密钥的权限。

```
{
   "Sid": "Allow service-linked role use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::account-id:role/aws-service-role/pcs.amazonaws.com/
AWSServiceRoleForPCS"
       ]
   },
   "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
       "kms:DescribeKey"
   ],
   "Resource": "*"
}
```

```
{
   "Sid": "Allow attachment of persistent resources",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
```

```
            "arn:aws:iam::account-id:role/aws-service-role/pcs.amazonaws.com/
AWSServiceRoleForPCS"
        ]
    },
    "Action": [
        "kms:CreateGrant"
    ],
    "Resource": "*",
    "Condition": {
        "Bool": {
            "kms:GrantIsForAWSResource": true
        }
     }
}
```

示例 2：允许跨账户访问客户托管密钥的关键策略部分

如果您在与 AWS PCS 集群不同的账户中创建客户托管密钥，则必须将授权与密钥策略结合使用，以允许跨账户访问该密钥。

授予对密钥的访问权限

1. 将以下政策声明添加到客户托管密钥的密钥策略中。将示例 ARN 替换为其他账户的 ARN。*111122223333*替换为要在中 AWS 账户 创建 AWS PCS 集群的的实际账户 ID。这将允许您向指定账户中的 IAM 用户或角色授予使用下面的 CLI 命令为密钥创建授权的权限。默认情况下，用户无权访问密钥。

```
{.
    "Sid": "Allow external account 111122223333 use of the customer managed key",
    "Effect": "Allow",
    "Principal": {
        "AWS": [
            "arn:aws:iam::111122223333:root"
        ]
    },
    "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
```

```
   "Resource": "*"
}
```

```
{
   "Sid": "Allow attachment of persistent resources in external
 account 111122223333",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::111122223333:root"
       ]
   },
   "Action": [
       "kms:CreateGrant"
   ],
   "Resource": "*"
}
```

2. 使用您要在其中创建 AWS PCS 集群的账户创建授权，将相关权限委托给 AWS PCS 服务相关角色。的值`grantee-principal`是服务相关角色的 ARN。的值`key-id`是密钥的 ARN。

   以下示例 create-grant CLI 命令为账户`AWSServiceRoleForPCS`中指定的服务相关角色提供了在账户中使用客户托管密钥的*111122223333*权限。*444455556666*

```
aws kms create-grant \
  --region us-west-2 \
  --key-id arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d \
  --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
pcs.amazonaws.com/AWSServiceRoleForPCS \
  --operations "Encrypt" "Decrypt" "ReEncryptFrom" "ReEncryptTo" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "DescribeKey" "CreateGrant"
```

> **Note**
>
> 发出请求的用户必须拥有使用该`kms:CreateGrant`操作的权限。

   以下示例 IAM 策略允许账户中的 IAM 身份（用户或角色）*111122223333*为账户中的客户托管密钥创建授权*444455556666*。

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowCreationOfGrantForTheKMSKeyinExternalAccount444455556666",
      "Effect": "Allow",
      "Action": "kms:CreateGrant",
      "Resource": "arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    }
  ]
}
```

有关为不同 AWS 账户中的 KMS 密钥创建授权的更多信息，请参阅《AWS Key Management Service 开发人员指南》中的 AWS KMS中的授权。

> **Important**
>
> 指定为被授权者委托人的服务相关角色名称必须是现有角色的名称。创建授权后，为确保该授权允许 AWS PCS 使用指定的 KMS 密钥，请勿删除和重新创建服务相关角色。

在 AWS KMS 控制台中编辑密钥策略

之前部分中的示例仅显示如何向密钥策略添加语句，这只是更改密钥策略的一种方法。更改密钥策略的最简单方法是使用 AWS KMS 控制台的默认密钥策略视图，并将 IAM 身份（用户或角色）设为相应密钥策略的关键用户之一。有关更多信息，请参阅《AWS Key Management Service 开发人员指南》中的使用 AWS 管理控制台 默认视图。

> **Warning**
>
> 控制台的默认视图策略声明包括对客户托管密钥执行 AWS KMS Revoke操作的权限。如果您撤销授予您账户中客户托管密钥 AWS 账户 访问权限的授权，则该用户将 AWS 账户 失去对加密数据和密钥的访问权限。

AWS 并行计算服务 使用接口端点进行访问 (AWS PrivateLink)

您可以使用 AWS PrivateLink 在您的 VPC 和 AWS 并行计算服务 (AWS PCS) 之间创建私有连接。您可以像在 VPC 中 AWS PCS 一样进行访问，无需使用互联网网关、NAT 设备、VPN 连接或 Direct Connect 连接。VPC 中的实例不需要公有 IP 地址即可访问 AWS PCS。

您可以通过创建由 AWS PrivateLink提供支持的接口端点来建立此私有连接。我们将在您为接口端点启用的每个子网中创建一个端点网络接口。这些是请求者托管的网络接口，用作发往 AWS PCS的流量的入口点。

有关更多信息，请参阅AWS PrivateLink 指南 AWS PrivateLink中的AWS 服务 直通访问。

的注意事项 AWS PCS

在为设置接口终端节点之前 AWS PCS，请查看AWS PrivateLink 指南中的使用接口 VPC 终端节点访问 AWS 服务。

AWS PCS 支持通过接口端点调用其所有 API 操作。

如果您的 VPC 无法直接访问互联网，则必须配置 VPC 终端节点以使您的计算节点组实例能够调用 AWS PCS `RegisterComputeNodeGroupInstance`API 操作。

为创建接口终端节点 AWS PCS

您可以创建用于 AWS PCS 使用 Amazon VPC 控制台或 AWS Command Line Interface (AWS CLI) 的接口终端节点。有关更多信息，请参阅《AWS PrivateLink 指南》中的创建接口端点。

AWS PCS 使用以下服务名称创建接口终端节点：

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

*region*替换为 AWS 区域 用于创建终端节点的 ID，例如`us-east-1`。

如果为接口端点启用私有 DNS，则可使用其默认区域 DNS 名称向 AWS PCS 发出 API 请求。例如 `pcs.us-east-1.amazonaws.com`。

为 VPC 端点创建端点策略

端点策略是一种 IAM 资源，您可以将其附加到接口端点。默认终端节点策略允许 AWS PCS 通过接口终端节点进行完全访问。要控制允许 AWS PCS 从您的 VPC 访问权限，请将自定义终端节点策略附加到接口终端节点。

端点策略指定以下信息：

- 可执行操作的主体（AWS 账户、IAM 用户和 IAM 角色）。
- 可执行的操作。
- 可对其执行操作的资源。

有关更多信息，请参阅《AWS PrivateLink 指南》中的使用端点策略控制对服务的访问权限。

示例：用于 AWS PCS 操作的 VPC 终端节点策略

以下是自定义端点策略的示例。当您将此策略附加到接口终端节点时，它会向具有指定*cluster-id*权限的集群的所有委托人授予对所列 AWS PCS 操作的访问权限。*region*替换为集 AWS 区域 群的 ID，例如us-east-1。*account-id*替换为集群的 AWS 账户 编号。

```
{
    "Statement": [
            {
                "Action": [
                "pcs:CreateCluster",
                "pcs:ListClusters",
                "pcs:DeleteCluster",
                "pcs:GetCluster",
                ],
                "Effect": "Allow",
                "Principal": "*",
                "Resource": [
                    "arn:aws:pcs:region:account-id:cluster/cluster-id*"
                ]
            }
        ]
}
```

并 AWS 行计算服务的 Identity and Access 管理

AWS Identity and Access Management (IAM) AWS 服务 可帮助管理员安全地控制对 AWS 资源的访问权限。IAM 管理员控制谁可以进行身份验证（登录）和授权（有权限）使用 AWS PCS 资源。您可以使用 IAM AWS 服务 ，无需支付额外费用。

主题

受众

您的使用方式 AWS Identity and Access Management (IAM) 因您的角色而异：

- 服务用户：如果您无法访问功能，请从管理员处请求权限（请参阅排查 AWS 并行计算服务身份和访问权限）
- 服务管理员：确定用户访问权限并提交权限请求（请参阅AWS 并行计算服务如何与 IAM 配合使用）
- IAM 管理员：编写用于管理访问权限的策略（请参阅AWS 并行计算服务的基于身份的策略示例）

使用身份进行身份验证

身份验证是您 AWS 使用身份凭证登录的方式。您必须以 IAM 用户身份进行身份验证 AWS 账户根用户，或者通过担任 IAM 角色进行身份验证。

您可以使用来自身份源的证书 AWS IAM Identity Center（例如（IAM Identity Center）、单点登录身份验证或 Google/Facebook 证书，以联合身份登录。有关登录的更多信息，请参阅《AWS 登录 用户指南》中的如何登录您的 AWS 账户。

对于编程访问， AWS 提供 SDK 和 CLI 来对请求进行加密签名。有关更多信息，请参阅《IAM 用户指南》中的适用于 API 请求的AWS 签名版本 4。

AWS 账户 root 用户

创建时 AWS 账户，首先会有一个名为 AWS 账户 root 用户的登录身份，该身份可以完全访问所有资源 AWS 服务 和资源。我们强烈建议不要使用根用户进行日常任务。有关需要根用户凭证的任务，请参阅《IAM 用户指南》中的需要根用户凭证的任务。

联合身份

作为最佳实践，要求人类用户使用与身份提供商的联合身份验证才能 AWS 服务 使用临时证书进行访问。

联合身份是指来自您的企业目录、Web 身份提供商的用户 Directory Service ，或者 AWS 服务 使用来自身份源的凭据进行访问的用户。联合身份代入可提供临时凭证的角色。

要集中管理访问权限，建议使用。 AWS IAM Identity Center有关更多信息，请参阅《AWS IAM Identity Center 用户指南》中的什么是 IAM Identity Center？。

IAM 用户和群组

IAM 用户是对某个人员或应用程序具有特定权限的一个身份。建议使用临时凭证，而非具有长期凭证的 IAM 用户。有关更多信息，请参阅 IAM 用户指南中的要求人类用户使用身份提供商的联合身份验证才能 AWS 使用临时证书进行访问。

IAM 组指定一组 IAM 用户，便于更轻松地对大量用户进行权限管理。有关更多信息，请参阅《IAM 用户指南》中的 IAM 用户使用案例。

IAM 角色

IAM 角色是具有特定权限的身份，可提供临时凭证。您可以通过从用户切换到 IAM 角色（控制台）或调用 AWS CLI 或 AWS API 操作来代入角色。有关更多信息，请参阅《IAM 用户指南》中的担任角色的方法。

IAM 角色对于联合用户访问、临时 IAM 用户权限、跨账户访问、跨服务访问以及在 Amazon EC2 上运行的应用程序非常有用。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

使用策略管理访问

您可以 AWS 通过创建策略并将其附加到 AWS 身份或资源来控制中的访问权限。策略定义了与身份或资源关联时的权限。 AWS 在委托人提出请求时评估这些政策。大多数策略都以 JSON 文档的 AWS 形式存储在中。有关 JSON 策略文档的更多信息，请参阅《IAM 用户指南》中的 JSON 策略概述。

管理员使用策略，通过定义哪个主体可以在什么条件下对哪些资源执行哪些操作来指定谁有权访问什么。

默认情况下，用户和角色没有权限。IAM 管理员创建 IAM 策略并将其添加到角色中，然后用户可以担任这些角色。IAM 策略定义权限，与执行操作所用的方法无关。

基于身份的策略

基于身份的策略是您附加到身份（用户、组或角色）的 JSON 权限策略文档。这些策略控制身份可以执行什么操作、对哪些资源执行以及在什么条件下执行。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

基于身份的策略可以是内联策略（直接嵌入到单个身份中）或托管策略（附加到多个身份的独立策略）。要了解如何在托管策略和内联策略之间进行选择，请参阅《IAM 用户指南》中的在托管策略与内联策略之间进行选择。

基于资源的策略

基于资源的策略是附加到资源的 JSON 策略文档。示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。您必须在基于资源的策略中指定主体。

基于资源的策略是位于该服务中的内联策略。您不能在基于资源的策略中使用 IAM 中的 AWS 托管策略。

其他策略类型

AWS 支持其他策略类型，这些策略类型可以设置更常见的策略类型授予的最大权限：

- 权限边界 – 设置基于身份的策略可以授予 IAM 实体的最大权限。有关更多信息，请参阅《 IAM 用户指南》中的 IAM 实体的权限边界。
- 服务控制策略 (SCPs)-在中指定组织或组织单位的最大权限 AWS Organizations。有关更多信息，请参阅《AWS Organizations 用户指南》中的服务控制策略。
- 资源控制策略 (RCPs)-设置账户中资源的最大可用权限。有关更多信息，请参阅《AWS Organizations 用户指南》中的资源控制策略 (RCPs)。
- 会话策略 – 在为角色或联合用户创建临时会话时，作为参数传递的高级策略。有关更多信息，请参阅《IAM 用户指南》中的会话策略。

多个策略类型

当多个类型的策略应用于一个请求时，生成的权限更加复杂和难以理解。要了解在涉及多种策略类型时如何 AWS 确定是否允许请求，请参阅 IAM 用户指南中的策略评估逻辑。

AWS 并行计算服务如何与 IAM 配合使用

在使用 IAM 管理对 AWS PCS 的访问权限之前，请先了解有哪些 IAM 功能可用于 AWS PCS。

您可以与 AWS 并行计算服务一起使用的 IAM 功能

IAM 功能	AWS 电脑支持
基于身份的策略	是
基于资源的策略	否
策略操作	是
策略资源	是
策略条件键（特定于服务）	是
ACLs	否
ABAC（策略中的标签）	是
临时凭证	是
主体权限	是
服务角色	否
服务关联角色	是

要全面了解 AWS PCS 和其他 AWS 服务如何与大多数 IAM 功能配合使用，请参阅 IAM 用户指南中的与 IAM 配合使用的AWS 服务。

PCS 基于身份的策略 AWS

支持基于身份的策略：是

基于身份的策略是可附加到身份（如 IAM 用户、用户组或角色）的 JSON 权限策略文档。这些策略控制用户和角色可在何种条件下对哪些资源执行哪些操作。要了解如何创建基于身份的策略，请参阅《IAM 用户指南》中的使用客户管理型策略定义自定义 IAM 权限。

通过使用 IAM 基于身份的策略，您可以指定允许或拒绝的操作和资源以及允许或拒绝操作的条件。要了解可在 JSON 策略中使用的所有元素，请参阅《IAM 用户指南》中的 IAM JSON 策略元素引用。

PCS 基于身份的策略示例 AWS

要查看 AWS PCS 基于身份的策略的示例，请参阅。AWS 并行计算服务的基于身份的策略示例

PCS 中 AWS 基于资源的策略

支持基于资源的策略：否

基于资源的策略是附加到资源的 JSON 策略文档。基于资源的策略的示例包括 IAM 角色信任策略和 Amazon S3 存储桶策略。在支持基于资源的策略的服务中，服务管理员可以使用它们来控制对特定资源的访问。对于在其中附加策略的资源，策略定义指定主体可以对该资源执行哪些操作以及在什么条件下执行。您必须在基于资源的策略中指定主体。委托人可以包括账户、用户、角色、联合用户或 AWS 服务。

要启用跨账户访问，您可以将整个账户或其他账户中的 IAM 实体指定为基于资源的策略中的主体。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

AWS PCS 的政策措施

支持策略操作：是

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

JSON 策略的 `Action` 元素描述可用于在策略中允许或拒绝访问的操作。在策略中包含操作以授予执行关联操作的权限。

要查看 AWS PCS 操作列表，请参阅《服务授权参考》中的 “ AWS 并行计算服务定义的操作”。

AWS PCS 中的策略操作在操作前使用以下前缀：

```
pcs
```

要在单个语句中指定多项操作，请使用逗号将它们隔开。

```
"Action": [
      "pcs:action1",
      "pcs:action2"
         ]
```

AWS PCS 的策略资源

支持策略资源：是

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

`Resource` JSON 策略元素指定要向其应用操作的一个或多个对象。作为最佳实践，请使用其 Amazon 资源名称（ARN）指定资源。对于不支持资源级权限的操作，请使用通配符 (*) 指示语句应用于所有资源。

```
"Resource": "*"
```

要查看 AWS PCS 资源类型及其列表 ARNs，请参阅《服务授权参考》中的“ AWS 并行计算服务定义的资源”。要了解您可以使用哪些操作来指定每种资源的 ARN，请参阅AWS 并行计算服务定义的操作。

要查看 AWS PCS 基于身份的策略的示例，请参阅。AWS 并行计算服务的基于身份的策略示例

AWS PCS 的策略条件密钥

支持特定于服务的策略条件键：是

管理员可以使用 AWS JSON 策略来指定谁有权访问什么。也就是说，哪个主体可以对什么资源执行操作，以及在什么条件下执行。

`Condition` 元素根据定义的条件指定语句何时执行。您可以创建使用条件运算符（例如，等于或小于）的条件表达式，以使策略中的条件与请求中的值相匹配。要查看所有 AWS 全局条件键，请参阅 IAM 用户指南中的AWS 全局条件上下文密钥。

要查看 AWS PCS 条件密钥列表，请参阅《服务授权参考》中的“ AWS 并行计算服务的条件密钥”。要了解可以使用条件键的操作和资源，请参阅AWS 并行计算服务定义的操作。

要查看 AWS PCS 基于身份的策略的示例，请参阅。AWS 并行计算服务的基于身份的策略示例

ACLs 在 AWS PCS 中

支持 ACLs：否

访问控制列表 (ACLs) 控制哪些委托人（账户成员、用户或角色）有权访问资源。 ACLs 与基于资源的策略类似，尽管它们不使用 JSON 策略文档格式。

带 PCS 的 ABA AWS C

支持 ABAC（策略中的标签）：是

基于属性的访问权限控制（ABAC）是一种授权策略，该策略基于称为标签的属性来定义权限。您可以将标签附加到 IAM 实体和 AWS 资源，然后设计 ABAC 策略以允许在委托人的标签与资源上的标签匹配时进行操作。

要基于标签控制访问，您需要使用 `aws:ResourceTag/`*`key-name`*`aws:RequestTag/`*`key-name`* 或 `aws:TagKeys` 条件键在策略的条件元素中提供标签信息。

如果某个服务对于每种资源类型都支持所有这三个条件键，则对于该服务，该值为是。如果某个服务仅对于部分资源类型支持所有这三个条件键，则该值为部分。

有关 ABAC 的更多信息，请参阅《IAM 用户指南》中的使用 ABAC 授权定义权限。要查看设置 ABAC 步骤的教程，请参阅《IAM 用户指南》中的使用基于属性的访问权限控制（ABAC）。

在 AWS PCS 中使用临时证书

支持临时凭证：是

临时证书提供对 AWS 资源的短期访问权限，并且是在您使用联合身份或切换角色时自动创建的。AWS 建议您动态生成临时证书，而不是使用长期访问密钥。有关更多信息，请参阅《IAM 用户指南》中的 IAM 中的临时安全凭证和使用 IAM 的。AWS 服务

PCS 的跨服务主体 AWS 权限

支持转发访问会话（FAS）：是

转发访问会话 (FAS) 使用调用主体的权限 AWS 服务，再加上 AWS 服务 向下游服务发出请求的请求。有关发出 FAS 请求时的策略详情，请参阅转发访问会话。

AWS PCS 的服务角色

支持服务角色：否

服务角色是由一项服务担任、代表您执行操作的 IAM 角色。IAM 管理员可以在 IAM 中创建、修改和删除服务角色。有关更多信息，请参阅《IAM 用户指南》中的创建向 AWS 服务委派权限的角色。

> **Warning**
>
> 更改服务角色的权限可能会中断 AWS PCS 的功能。仅当 AWS PCS 提供相关指导时才编辑服务角色。

PCS 的服务相关角色 AWS

支持服务关联角色：是

服务相关角色是一种与服务相关联的 AWS 服务服务角色。服务可以代入代表您执行操作的角色。服务相关角色出现在您的中 AWS 账户 ，并且归服务所有。IAM 管理员可以查看但不能编辑服务关联角色的权限。

有关创建或管理 AWS PCS 服务相关角色的详细信息，请参阅PCS 的服务相关角色 AWS。

AWS 并行计算服务的基于身份的策略示例

默认情况下，用户和角色无权创建或修改 AWS PCS 资源。要授予用户对所需资源执行操作的权限，IAM 管理员可以创建 IAM 策略。

要了解如何使用这些示例 JSON 策略文档创建基于 IAM 身份的策略，请参阅《IAM 用户指南》中的创建 IAM 策略（控制台）。

有关 AWS PCS 定义的操作和资源类型（包括每种资源类型的格式）的详细信息，请参阅《服务授权参考》中的 “AWS 并行计算服务的操作、资源和条件密钥”。 ARNs

主题

- 策略最佳实践
- 使用 AWS PCS 控制台
- 允许用户查看他们自己的权限

策略最佳实践

基于身份的策略决定了某人是否可以在您的账户中创建、访问或删除 AWS PCS 资源。这些操作可能会使 AWS 账户产生成本。创建或编辑基于身份的策略时，请遵循以下指南和建议：

- 开始使用 AWS 托管策略并转向最低权限权限 — 要开始向用户和工作负载授予权限，请使用为许多常见用例授予权限的AWS 托管策略。它们在你的版本中可用 AWS 账户。我们建议您通过定义针对您的用例的 AWS 客户托管策略来进一步减少权限。有关更多信息，请参阅《IAM 用户指南》中的AWS 托管策略或工作职能的AWS 托管策略。
- 应用最低权限：在使用 IAM 策略设置权限时，请仅授予执行任务所需的权限。为此，您可以定义在特定条件下可以对特定资源执行的操作，也称为最低权限许可。有关使用 IAM 应用权限的更多信息，请参阅《IAM 用户指南》中的 IAM 中的策略和权限。
- 使用 IAM 策略中的条件进一步限制访问权限：您可以向策略添加条件来限制对操作和资源的访问。例如，您可以编写策略条件来指定必须使用 SSL 发送所有请求。如果服务操作是通过特定的方式使用的，则也可以使用条件来授予对服务操作的访问权限 AWS 服务，例如 CloudFormation。有关更多信息，请参阅《IAM 用户指南》中的 IAM JSON 策略元素：条件。
- 使用 IAM Access Analyzer 验证您的 IAM 策略，以确保权限的安全性和功能性：IAM Access Analyzer 会验证新策略和现有策略，以确保策略符合 IAM 策略语言（JSON）和 IAM 最佳实践。IAM Access Analyzer 提供 100 多项策略检查和可操作的建议，以帮助您制定安全且功能性强的策略。有关更多信息，请参阅《IAM 用户指南》中的使用 IAM Access Analyzer 验证策略。
- 需要多重身份验证 (MFA)-如果 AWS 账户您的场景需要 IAM 用户或根用户，请启用 MFA 以提高安全性。若要在调用 API 操作时需要 MFA，请将 MFA 条件添加到您的策略中。有关更多信息，请参阅《IAM 用户指南》中的使用 MFA 保护 API 访问。

有关 IAM 中的最佳实操的更多信息，请参阅《IAM 用户指南》中的 IAM 中的安全最佳实践。

使用 AWS PCS 控制台

要访问 AWS 并行计算服务控制台，您必须拥有一组最低权限。这些权限必须允许您列出和查看有关您的 AWS PCS 资源的详细信息 AWS 账户。如果创建比必需的最低权限更为严格的基于身份的策略，对于附加了该策略的实体（用户或角色），控制台将无法按预期正常运行。

对于仅调用 AWS CLI 或 AWS API 的用户，您无需为其设置最低控制台权限。相反，只允许访问与其尝试执行的 API 操作相匹配的操作。

有关使用 AWS PCS 控制台所需的最低权限的更多信息，请参阅AWS PCS 的最低权限。

允许用户查看他们自己的权限

该示例说明了您如何创建策略，以允许 IAM 用户查看附加到其用户身份的内联和托管式策略。此策略包括在控制台上或使用 AWS CLI 或 AWS API 以编程方式完成此操作的权限。

```
{
```

```
    "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": "*"
        }
    ]
}
```

AWS AWS 并行计算服务的托管策略

AWS 托管策略是由创建和管理的独立策略 AWS。 AWS 托管策略旨在为许多常见用例提供权限，以便您可以开始为用户、组和角色分配权限。

请记住， AWS 托管策略可能不会为您的特定用例授予最低权限权限，因为它们可供所有 AWS 客户使用。我们建议通过定义特定于使用案例的客户管理型策略来进一步减少权限。

您无法更改 AWS 托管策略中定义的权限。如果 AWS 更新 AWS 托管策略中定义的权限，则更新会影响该策略所关联的所有委托人身份（用户、组和角色）。 AWS 最有可能在启动新的 API 或现有服务可以使用新 AWS 服务 的 API 操作时更新 AWS 托管策略。

有关更多信息，请参阅《IAM 用户指南》中的 AWS 托管式策略。

AWS 托管策略： AWSPCSComputeNodePolicy

您可以附加 AWSPCSComputeNodePolicy 到您的 IAM 实体。您可以将此策略附加到您指定的 AWS PCS 计算节点 IAM 角色，以允许使用该角色的节点连接到 AWS PCS 集群。

AWS 当您使用控制台创建计算节点组时，PCS 会将此策略附加到计算节点组角色。

权限详细信息

该策略包含以下权限。

- `pcs:RegisterComputeNodeGroupInstance`— 允许 AWS PCS 计算节点（EC2 实例）向 AWS PCS 集群注册。

要查看此策略的权限，请参阅《AWS 托管式策略参考》中的 AWSPCSComputeNodePolicy。

AWS 托管策略： AWSPCSServiceRolePolicy

您无法附加 AWSPCSServiceRolePolicy 到您的 IAM 实体。此策略附加到服务相关角色，允许 AWS PCS 代表您执行操作。有关更多信息，请参阅 PCS 的服务相关角色 AWS。

权限详细信息

该策略包含以下权限。

- `ec2`— 允许 AWS PCS 创建和管理 Amazon EC2 资源。
- `iam`— 允许 AWS PCS 为亚马逊 EC2 队列创建服务相关角色并将该角色传递给亚马逊 EC2。

- `cloudwatch`— 允许 AWS PCS 向亚马逊发布服务指标 CloudWatch。
- `secretsmanager`— 允许 AWS PCS 管理 AWS PCS 群集资源的密钥。

要查看此策略的权限，请参阅《AWS 托管式策略参考》中的 AWSPCSServiceRolePolicy。

AWS AWS 托管策略的 PCS 更新

查看自该服务开始跟踪这些更改以来对 AWS PCS AWS 托管策略的更新的详细信息。要获得有关此页面变更的自动提醒，请订阅 AWS PCS 文档历史记录页面上的 RSS 提要。

更改	描述	日期
AWSPCSServiceRolePolicy：对现有策略的更新	AWS PCS 添加了新的权限来支持容量块以实现可预测的计算容量。 增加了`ec2:DescribeCapacityReservations` 允许 AWS PCS 发现和使用计算节点组的容量块预留的权限。	2025 年 9 月 11 日
AWSPCSComputeNodePolicy：新策略	AWS PCS 添加了一项新策略，向 AWS PCS 计算节点授予连接到 AWS PCS 集群的权限。 AWS 当您在 PCS 控制台中创建计算节点组时， AWS PCS 会将此策略附加到 IAM 角色。	2025 年 6 月 23 日
更新了本文档中的 JSON	更正了本文档中的 JSON 以包含在内`"arn:aws:ec2:*:*:spot-instances-request/*"` 。	2024 年 9 月 5 日
AWS PCS 开始追踪变更	AWS PCS 开始跟踪其 AWS 托管策略的更改。	2024 年 8 月 28 日

PCS 的服务相关角色 AWS

AWS 并行计算服务使用 AWS Identity and Access Management (IAM) 服务相关角色。服务相关角色是一种独特的 IAM 角色，直接关联到 AWS PCS。服务相关角色由 AWS PCS 预定义，包括该服务代表您调用其他 AWS 服务所需的所有权限。

服务相关角色可以更轻松地设置 AWS PCS，因为您不必手动添加必要的权限。 AWS PCS 定义其服务相关角色的权限，除非另有定义，否则只有 AWS PCS 可以担任其角色。定义的权限包括信任策略和权限策略，而且权限策略不能附加到任何其它 IAM 实体。

只有在首先删除服务相关角色的相关资源后，才能删除该角色。这样可以保护您的 AWS PCS 资源，因为您不会意外删除访问这些资源的权限。

有关支持服务相关角色的其他服务的信息，请参阅与 IAM 配合使用的AWS 服务，并在服务相关角色列中查找标有 “是” 的服务。选择是和链接，查看该服务的服务关联角色文档。

PCS 的服务相关角色权限 AWS

AWS PCS 使用名为 PCS 的服务相关角色 — 向 AWSServiceRoleForPCS 授予管理 Amazon EC2 资源的权限。 AWS

AWSServiceRoleForPCS 服务相关角色信任以下服务来代入该角色：

- `pcs.amazonaws.com`

名为的角色权限策略AWSPCSServiceRolePolicy允许 AWS PCS 完成对特定资源的操作。

您必须配置使用户、组或角色能够创建、编辑或删除服务相关角色的权限。有关更多信息，请参阅《IAM 用户指南》中的服务相关角色权限。

为 PCS 创建服务相关角色 AWS

您无需手动创建服务相关角色。 AWS 创建集群时，PCS 会为您创建一个服务相关角色。

编辑 PCS 的服务相关角色 AWS

AWS PCS 不允许您编辑 AWSService RoleFor PCS 服务相关角色。创建服务关联角色后，您将无法更改角色的名称，因为可能有多种实体引用该角色。但是可以使用 IAM 编辑角色描述。有关更多信息，请参阅《IAM 用户指南》中的编辑服务关联角色。

删除 PCS 的服务相关角色 AWS

如果不再需要使用某个需要服务关联角色的功能或服务，我们建议您删除该角色。这样就没有未被主动监控或维护的未使用实体。但是，必须先清除服务相关角色的资源，然后才能手动删除它。

> **Note**
>
> 如果您尝试删除资源时 AWS PCS 服务正在使用该角色，则删除可能会失败。如果发生这种情况，请等待几分钟后重试。

移除 AWS PCS 使用的 AWSService RoleFor PCS 资源

您必须删除所有集群才能删除 AWSService RoleFor PCS 服务相关角色。有关更多信息，请参阅删除集群。

使用 IAM 手动删除服务关联角色

使用 IAM 控制台 AWS CLI、或 AWS API 删除 AWSService RoleFor PCS 服务相关角色。有关更多信息，请参阅《IAM 用户指南》中的删除服务关联角色。

AWS PCS 服务相关角色支持的区域

AWS PCS 支持在提供服务的所有地区使用服务相关角色。有关更多信息，请参阅AWS 区域和端点。

适用于 AWS PCS 的 Amazon EC2 竞价角色

如果要创建使用 Spot 作为购买选项的 AWS PCS 计算节点组，则还必须具有AWSServiceRoleForEC2竞价服务相关角色。 AWS 账户您可以使用以下 AWS CLI 命令来创建角色。有关更多信息，请参阅《AWS Identity and Access Management 用户指南》中的创建服务相关角色和创建向 AWS 服务委派权限的角色。

```
aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

> **Note**
>
> 如果您 AWS 账户 已经拥有 `AWSServiceRoleForEC2Spot` IAM 角色，则会收到以下错误。

```
 An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
operation: Service role name AWSServiceRoleForEC2Spot has been taken in this
account, please try a different suffix.
```

AWS PCS 的最低权限

本节介绍了 IAM 身份（用户、群组或角色）使用该服务所需的最低 IAM 权限。

目录

使用 API 操作的最低权限

API 操作	最小权限	控制台的其他权限
CreateCluster	ec2:CreateNetworkInterface, ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:GetSecurityGroupsForVpc, iam:CreateServiceLinkedRole, secretsmanager:CreateSecret, secretsmanager:TagResource, secretsmanager:RotateSecret,	

API 操作	最小权限	控制台的其他权限
	pcs:CreateCluster	
ListClusters	pcs:ListClusters	
GetCluster	pcs:GetCluster	ec2:DescribeSubnets
DeleteCluster	pcs:DeleteCluster	
CreateComputeNodeGroup	ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:DescribeLaunchTemplates, ec2:DescribeLaunchTemplateVersions, ec2:DescribeInstanceTypes, ec2:DescribeInstanceTypeOfferings, ec2:RunInstances, ec2:CreateFleet, ec2:CreateTags, iam:PassRole, iam:GetInstanceProfile, pcs:CreateComputeNodeGroup	iam:ListInstanceProfiles, ec2:DescribeImages, pcs:GetCluster
ListComputerNodeGroups	pcs:ListComputeNodeGroups	pcs:GetCluster
GetComputeNodeGroup	pcs:GetComputeNodeGroup	ec2:DescribeSubnets

API 操作	最小权限	控制台的其他权限
UpdateComputeNodeGroup	`ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:DescribeLaunchTemplates, ec2:DescribeLaunchTemplateVersions, ec2:DescribeInstanceTypes, ec2:DescribeInstanceTypeOfferings, ec2:RunInstances, ec2:CreateFleet, ec2:CreateTags, iam:PassRole, iam:GetInstanceProfile, pcs:UpdateComputeNodeGroup`	`pcs:GetComputeNodeGroup, iam:ListInstanceProfiles, ec2:DescribeImages, pcs:GetCluster`
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`

API 操作	最小权限	控制台的其他权限
DeleteQueue	`pcs:DeleteQueue`	

使用标签的最低权限

在 AWS PCS 中对资源使用标签需要以下权限。

```
pcs:ListTagsForResource,
pcs:TagResource,
pcs:UntagResource
```

支持日志的最低权限

AWS PCS 将日志数据发送到 Amazon CloudWatch 日志（CloudWatch 日志）。您必须确保您的身份具有使用 CloudWatch 日志的最低权限。有关更多信息，请参阅 Amazon Logs 用户指南中的管理 CloudWatch CloudWatch 日志资源访问权限概述。

有关服务向日志发送日志所需的权限的信息，请参阅 Amazon CloudWatch Lo g CloudWatch s 用户指南中的启用 AWS 服务日志记录。

使用容量块的最低权限

适用于 ML 的 Amazon EC2 容量块是一种 Amazon EC2 购买选项，允许您提前付费在特定日期和时间范围内预留基于 GPU 的加速计算实例，以支持短期工作负载。有关更多信息，请参阅 将 Amazon EC2 容量块用于带有 AWS PCS 的机器学习。

创建或更新计算节点组时，您可以选择使用容量块。您用于创建或更新计算节点组的 IAM 身份必须具有以下权限：

```
ec2:DescribeCapacityReservations
```

服务管理员的最低权限

以下 IAM 策略指定了 IAM 身份（用户、群组或角色）配置和管理 AWS PCS 服务所需的最低权限。

> **Note**
>
> 不配置和管理服务的用户不需要这些权限。仅运行作业的用户使用安全外壳 (SSH) 连接到集群。 AWS Identity and Access Management (IAM) 不处理 SSH 的身份验证或授权。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PCSAccess",
      "Effect": "Allow",
      "Action": [
        "pcs:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "EC2Access",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeImages",
        "ec2:GetSecurityGroupsForVpc",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeVpcs",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ec2:DescribeInstanceTypes",
        "ec2:DescribeInstanceTypeOfferings",
        "ec2:RunInstances",
        "ec2:CreateFleet",
        "ec2:CreateTags",
        "ec2:DescribeCapacityReservations"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamInstanceProfile",
      "Effect": "Allow",
      "Action": [
```

```
        "iam:GetInstanceProfile"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamPassRole",
      "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": "SLRAccess",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/pcs.amazonaws.com/AWSServiceRoleFor*",
        "arn:aws:iam::*:role/aws-service-role/spot.amazonaws.com/AWSServiceRoleFor*"
      ],
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": [
            "pcs.amazonaws.com",
            "spot.amazonaws.com"
          ]
        }
      }
    },
    {
```

```
      "Sid": "AccessKMSKey",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:GenerateDataKey",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SecretManagementAccess",
      "Effect": "Allow",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:TagResource",
        "secretsmanager:UpdateSecret",
        "secretsmanager:RotateSecret"
      ],
      "Resource": "*"
    },
    {
       "Sid": "ServiceLogsDelivery",
       "Effect": "Allow",
       "Action": [
         "pcs:AllowVendedLogDeliveryForResource",
         "logs:PutDeliverySource",
         "logs:PutDeliveryDestination",
         "logs:CreateDelivery"
       ],
       "Resource": "*"
    }
  ]
}
```

AWS 并行计算服务的 IAM 实例配置文件

在 EC2 实例上运行的应用程序必须在其发出的任何 AWS API 请求中包含 AWS 证书。我们建议您使用 IAM 角色来管理 EC2 实例上的临时证书。您可以定义实例配置文件来执行此操作，并将其附加到您的实例。有关更多信息，请参阅《亚马逊弹性计算云用户指南》中的 Amazon EC2 的 IAM 角色。

> **Note**
>
> 当您使用为 Amazon EC2 创建 IAM 角色时，控制台会自动创建实例配置文件并将其命名为 IAM 角色。 AWS 管理控制台 如果您使用 AWS CLI、 AWS API 操作或 AWS 软件开发工具包创建 IAM 角色，则可以将实例配置文件作为单独的操作创建。有关更多信息，请参阅 Amazon 弹性计算云用户指南中的实例配置文件。

创建计算节点组时，必须指定实例配置文件的 Amazon 资源名称 (ARN)。您可以为部分或所有计算节点组选择不同的实例配置文件。

要求

实例配置文件的 IAM 角色

与实例配置文件关联的 IAM 角色的路径`/aws-pcs/`中必须有，或者其名称必须以开头`AWSPCS`。

IAM 角色示例 ARNs

- `arn:aws:iam::*:role/AWSPCS-example-role-1`
- `arn:aws:iam::*:role/aws-pcs/example-role-2`

Permissions

与 AWS PCS 实例配置文件关联的 IAM 角色必须包含以下策略。

JSON

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

其他政策

考虑向实例配置文件添加托管策略。例如：

- AmazonS3 ReadOnlyAccess 提供对所有 S3 存储桶的只读访问权限。
- 亚马逊SSMManagedInstanceCore启用 AWS Systems Manager 服务的核心功能，例如直接从亚马逊管理控制台进行远程访问。
- CloudWatchAgentServerPolicy包含 AmazonCloudWatchAgent 在服务器上使用所需的权限。

您也可以加入自己的 IAM 策略来支持您的特定使用案例。

为 AWS PCS 创建实例配置文件

AWS PCS console

在创建计算节点组时选择“创建基本配置文件”，让 AWS PCS 使用最低要求的策略为您创建一个基本配置文件。

Amazon EC2 console

您可以直接从 Amazon EC2 控制台创建实例配置文件。有关更多信息，请参阅AWS Identity and Access Management 用户指南中的使用实例配置文件。

> **Important**
>
> 确保在 IAM 角色名称AWSPCS中使用所需的前缀。

AWS CLI

使用 AWS CLI 设置基本实例配置文件

> **Note**
>
> 将以下示例*example-role*中的名称替换为您的 IAM 角色的名称。

1. 以路径属性或/aws-pcs/以开头的名称创建 IAM 角色AWSPCS。

   a. 将以下内容复制并粘贴到名为的新文本文件中trust_policy.json。

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
             "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            },
            "Action": [
                "sts:AssumeRole"
            ]
        }
    ]
}
```

b. 使用以下命令之一，创建 IAM 角色。

```
aws iam create-role --path /aws-pcs/ --role-name example-role --assume-role-policy-document file://trust_policy.json
```

或者

```
aws iam create-role --role-name AWSPCS-example-role --assume-role-policy-document file://trust_policy.json
```

2. 附加权限。

a. 将以下内容复制并粘贴到名为的新文本文件中policy_document.json。

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

```
        }
    ]
}
```

b. 将策略文档附加到该角色。此命令将策略作为内联策略附加。

```
aws iam put-role-policy \
    --role-name example-role \
    --policy-name pcsRegisterInstancePolicy \
    --policy-document file://policy_document.json
```

3. 创建实例配置文件。***example-profile***替换为您的实例配置文件的名称。

```
aws iam create-instance-profile --instance-profile-name example-profile
```

4. 将 IAM 角色与实例配置文件关联。

```
aws iam add-role-to-instance-profile \
   --instance-profile-name example-profile \
   --role-name example-role
```

查找与 AWS PCS 一起使用的实例配置文件

1. 如果您不知道 AWS PCS 的 IAM 角色的确切名称，请使用以下 AWS CLI 命令列出符合 AWS PCS 名称要求的 IAM 角色。

```
aws iam list-roles --query "Roles[?starts_with(RoleName, 'AWSPCS') ||
 contains(Path, '/aws-pcs/')].[RoleName]" --output text
```

2. 使用以下 AWS CLI 命令列出与特定 IAM 角色关联的实例配置文件。替换*role-name*为符合 AWS PCS 名称要求的 IAM 角色的名称。

```
aws iam list-instance-profiles-for-role --role-name role-name
```

排查 AWS 并行计算服务身份和访问权限

使用以下信息来帮助您诊断和修复在使用 AWS PCS 和 IAM 时可能遇到的常见问题。

主题

- 我无权在 AWS PCS 中执行操作
- 我无权执行 iam：PassRole
- 我想允许我以外的人 AWS 账户 访问我的 AWS PCS 资源

我无权在 AWS PCS 中执行操作

如果您收到错误提示，指明您无权执行某个操作，则必须更新策略以允许执行该操作。

当 mateojackson IAM 用户尝试使用控制台查看有关虚构 *my-example-widget* 资源的详细信息，但不拥有虚构 pcs:*GetWidget* 权限时，会发生以下示例错误。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 pcs:GetWidget on resource: my-example-widget
```

在此情况下，必须更新 mateojackson 用户的策略，以允许使用 pcs:*GetWidget* 操作访问 *my-example-widget* 资源。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

我无权执行 iam：PassRole

如果您收到错误消息，提示您无权执行iam:PassRole操作，则必须更新您的策略以允许您将角色传递给 AWS PCS。

有些 AWS 服务 允许您将现有角色传递给该服务，而不是创建新的服务角色或服务相关角色。为此，您必须具有将角色传递到服务的权限。

当名为的 IAM 用户marymajor尝试使用控制台在 AWS PCS 中执行操作时，会出现以下示例错误。但是，服务必须具有服务角色所授予的权限才可执行此操作。Mary 不具有将角色传递到服务的权限。

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

在这种情况下，必须更新 Mary 的策略以允许她执行 iam:PassRole 操作。

如果您需要帮助，请联系您的 AWS 管理员。您的管理员是提供登录凭证的人。

我想允许我以外的人 AWS 账户 访问我的 AWS PCS 资源

您可以创建一个角色，以便其他账户中的用户或您组织外的人员可以使用该角色来访问您的资源。您可以指定谁值得信赖，可以代入角色。对于支持基于资源的策略或访问控制列表 (ACLs) 的服务，您可以使用这些策略向人们授予访问您的资源的权限。

要了解更多信息，请参阅以下内容：

- 要了解 AWS PCS 是否支持这些功能，请参阅AWS 并行计算服务如何与 IAM 配合使用。
- 要了解如何提供对您拥有的资源的访问权限 AWS 账户 ，请参阅 IAM 用户指南中的向您拥有 AWS 账户 的另一个 IAM 用户提供访问权限。
- 要了解如何向第三方提供对您的资源的访问权限 AWS 账户，请参阅 IAM 用户指南中的向第三方提供访问权限。 AWS 账户
- 要了解如何通过身份联合验证提供访问权限，请参阅《IAM 用户指南》中的为经过外部身份验证的用户（身份联合验证）提供访问权限。
- 要了解使用角色和基于资源的策略进行跨账户访问之间的差别，请参阅《IAM 用户指南》中的 IAM 中的跨账户资源访问。

AWS 并行计算服务的合规性验证

要了解是否属于特定合规计划的范围，请参阅AWS 服务 “按合规计划划分的范围” ”，然后选择您感兴趣的合规计划。 AWS 服务 有关一般信息，请参阅AWS 合规计划AWS。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅中的 “下载报告” 中的 “ AWS Artifact。

您在使用 AWS 服务 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。有关您在使用时的合规责任的更多信息 AWS 服务，请参阅AWS 安全文档。

AWS 并行计算服务中的弹性

AWS 全球基础设施是围绕 AWS 区域 可用区构建的。 AWS 区域 提供多个物理分隔和隔离的可用区，这些可用区通过低延迟、高吞吐量和高度冗余的网络连接。利用可用区，您可以设计和操作在可用区之间无中断地自动实现失效转移的应用程序和数据库。与传统的单个或多个数据中心基础设施相比，可用区具有更高的可用性、容错能力和可扩展性。

有关 AWS 区域 和可用区的更多信息，请参阅AWS 全球基础设施。

AWS 并行计算服务中的基础设施安全

作为一项托管服务，AWS 并行计算服务受 AWS 全球网络安全的保护。有关 AWS 安全服务以及如何 AWS 保护基础设施的信息，请参阅AWS 云安全。要使用基础设施安全的最佳实践来设计您的 AWS 环境，请参阅 S AWS ecurity Pillar Well-Architected Fram ework 中的基础设施保护。

您可以使用 AWS 已发布的 API 调用通过网络访问 AWS PCS。客户端必须支持以下内容：

- 传输层安全性协议（TLS）。我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 具有完全向前保密（PFS）的密码套件，例如 DHE（临时 Diffie-Hellman）或 ECDHE（临时椭圆曲线 Diffie-Hellman）。大多数现代系统（如 Java 7 及更高版本）都支持这些模式。

当 AWS PCS 创建集群时，该服务将在服务拥有的账户中启动 Slurm 控制器，该账户与您账户中的计算节点分开。为了桥接控制器和计算节点之间的通信，AWS PCS 在您的 VPC 中创建一个跨账户弹性网络接口 (ENI)。Slurm 控制器使用 ENI 来管理不同计算节点并与之通信 AWS 账户，维护资源的安全和隔离，同时促进高效的 HPC 和操作。 AI/ML

并 AWS 行计算服务中的漏洞分析和管理

配置和 IT 控制是您 AWS 和您的共同责任。有关更多信息，请参阅责任AWS 共担模型。 AWS 处理服务帐户中底层基础设施的基本安全任务，例如在控制器实例上修补操作系统、配置防火墙和 AWS 基础设施灾难恢复。这些流程已通过相应第三方审核和认证。有关更多详细信息，请参阅安全性、身份和合规性最佳实践。

> **Note**
>
> 当我们更新 Slurm 控制器时，它们不可用。正在运行的作业不受影响。当集群的控制器不可用时提交的作业将一直保留，直到控制器可用为止。

您应对以下基础架构的安全性负责 AWS 账户：

- 维护您的代码，包括更新和安全补丁。
- 修补和更新计算节点组的 Amazon 系统映像 (AMI) 中的操作系统，更新计算节点组以使用更新后的 AMI。
- 更新调度程序以使其保持在支持的版本内。更新计算节点组的 AMI，更新计算节点组以使用更新后的 AMI。

- 对用户客户端与其连接的节点之间的通信进行身份验证和加密。

有关更新计算节点组的 AMI 的更多信息，请参阅适用于 AWS PCS 的亚马逊机器映像 (AMIs)。

防止跨服务混淆代理

混淆代理问题是一个安全问题，即没有执行操作权限的实体可能会迫使更具权限的实体执行该操作。在中 AWS，跨服务模仿可能会导致混乱的副手问题。一个服务（呼叫服务）调用另一项服务（所谓的服务）时，可能会发生跨服务模拟。可以操纵调用服务，使用其权限以在其他情况下该服务不应有访问权限的方式对另一个客户的资源进行操作。为防止这种情况， AWS 提供可帮助您保护所有服务的数据的工具，而这些服务中的服务主体有权限访问账户中的资源。

我们建议在资源策略中使用aws:SourceArn和aws:SourceAccount全局条件上下文密钥来限制 AWS 并行计算服务 (AWS PCS) 向该资源提供的其他服务的权限。如果您只希望将一个资源与跨服务访问相关联，请使用。aws:SourceArn如果您想允许该账户中的任何资源与跨服务使用操作相关联，请使用。aws:SourceAccount

防范混淆代理问题最有效的方法是使用 aws:SourceArn 全局条件上下文键和资源的完整 ARN。如果不知道资源的完整 ARN，或者正在指定多个资源，请针对 ARN 未知部分使用带有通配符字符（*）的 aws:SourceArn 全局上下文条件键。例如 arn:aws:*servicename*:*:*123456789012*:*。

如果 aws:SourceArn 值不包含账户 ID，例如 Amazon S3 存储桶 ARN，您必须使用两个全局条件上下文键来限制权限。

的值aws:SourceArn必须是集群 ARN。

以下示例显示了如何在 AWS PCS 中使用aws:SourceArn和aws:SourceAccount全局条件上下文键来防止出现混淆的副手问题。

```
{
"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"
      }
    }
  }
}
```

作为计算节点组一部分配置的 Amazon EC2 实例的 IAM 角色

AWS PCS 会自动为集群中每个已配置的计算节点组编排 Amazon EC2 容量。创建计算节点组时，用户必须通过`iamInstanceProfileArn`字段提供 IAM 实例配置文件。实例配置文件指定了与预配置的 EC2 实例相关的权限。 AWS PCS 接受任何具有`AWSPCS`角色名称前缀或`/aws-pcs/`角色路径一部分的角色。创建或更新计算节点组的 IAM 身份（用户或角色）需要`iam:PassRole`获得权限。当用户调用`CreateComputeNodeGroup`或 UpdateComputeNodeGroup API 操作时， AWS PCS 会检查是否允许该用户执行`iam:PassRole`操作。

下面的策略示例授予仅传递名称以 AWSPCS 开头的 IAM 角色的权限。

```
{
    "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 并行计算服务的安全最佳实践

本节介绍特定于 AWS 并行计算服务 (AWS PCS) 的安全最佳实践。要详细了解中的安全最佳实践 AWS，请参阅安全、身份和合规性最佳实践。

与 AMI 相关的安全性

- 不要将 AWS PCS 示例 AMIs 用于生产工作负载。该样本 AMIs 不受支持，仅用于测试。
- 定期更新计算节点组的 AMI 中的操作系统和软件，以缓解漏洞。
- 仅使用从官方 AWS 来源下载的经过身份验证的官方 AWS PCS 软件包。
- 定期更新 AMI 中计算节点组的 AWS PCS 包，并更新计算节点以使用更新后的 AMI。考虑将此过程自动化，以最大限度地减少漏洞。

有关更多信息，请参阅 适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

Slurm 工作负载管理器安全

- 实施访问控制和网络限制，以保护 Slurm 控制和计算节点。仅允许受信任的用户和系统提交作业和访问 Slurm 管理命令。
- 使用 Slurm 的内置安全功能（例如 Slurm 身份验证）来确保提交的工作和通信经过身份验证。
- 更新 Slurm 版本以保持流畅的操作和集群支持。

> **Important**
>
> 任何使用已达到支持寿命 (EOSL) 的 Slurm 版本的集群都将立即停止。使用用户指南页面顶部的链接订阅 AWS PCS 文档 RSS 提要，以便在 Slurm 版本接近 EOSL 时收到通知。

  有关更多信息，请参阅 PCS 中的 Slurm 版本 AWS。
- 定期轮换集群密钥，以维护安全合规性并修复潜在的安全漏洞。这是 HIPAA 和 FedRAMP 合规所必需的。

  有关更多信息，请参阅 在 AWS PCS 中轮换集群密钥。

监控和日志记录

- 使用 Amazon CloudWatch Logs and AWS CloudTrail 来监控和记录集群中的操作，以及 AWS 账户。使用这些数据进行故障排除和审计。

网络安全

- 将 AWS PCS 集群部署在单独的 VPC 中，将您的 HPC 环境与其他网络流量隔离开来。
- 使用安全组和网络访问控制列表 (ACLs) 来控制 AWS PCS 实例和子网的入站和出站流量。
- 使用 AWS PrivateLink 我们的 VPC 终端节点将网络流量保持在您的集群和 AWS 网络内的其他 AWS 服务之间。有关更多信息，请参阅 AWS 并行计算服务 使用接口端点进行访问 (AWS PrivateLink)。

AWS PCS 的日志记录和监控

监控是维护 AWS PCS 和您的其他 AWS 资源的可靠性、可用性和性能的重要组成部分。AWS 提供以下监控工具，用于监视 AWS PCS、报告问题并在适当时自动采取措施：

- Amazon 会实时 CloudWatch监控您的 AWS 资源和您运行 AWS 的应用程序。您可以收集和跟踪指标，创建自定义的控制面板，以及设置警报以在指定的指标达到您指定的阈值时通知您或采取措施。例如，您可以 CloudWatch 跟踪您的 Amazon EC2 实例的 CPU 使用率或其他指标，并在需要时自动启动新实例。有关更多信息，请参阅 Amazon CloudWatch 用户指南。
- Amazon CloudWatch Logs 允许您监控、存储和访问来自 Amazon EC2 实例和其他来源的日志文件。 CloudTrail CloudWatch 日志可以监视日志文件中的信息，并在达到特定阈值时通知您。您还可以在高持久性存储中检索您的日志数据。有关更多信息，请参阅 Amazon CloudWatch 日志用户指南。
- AWS CloudTrail捕获由您的账户或代表您的 AWS 账户进行的 API 调用和相关事件，并将日志文件传输到您指定的 Amazon S3 存储桶。您可以识别哪些用户和帐户拨打了电话 AWS、发出呼叫的源 IP 地址以及呼叫发生的时间。有关更多信息，请参阅 AWS CloudTrail 《用户指南》。

AWS PCS 中的 Job 完成日志

任务完成日志可在 AWS 并行计算服务 (AWS PCS) 任务完成时为您提供有关这些任务的关键细节，无需支付额外费用。您可以使用其他 AWS 服务来访问和处理您的日志数据，例如亚马逊 CloudWatch 日志、亚马逊简单存储服务 (Amazon S3) Service 和 Amazon Data Firehose AWS ；PCS 会记录有关您的任务的元数据，例如以下内容。

- Job ID 和名称
- 用户和群组信息
- Job 状态（例如`COMPLETED`、`FAILED`、`CANCELLED`）
- 使用的分区
- 时间限制
- 开始、结束、提交和符合条件的时间
- 节点列表和计数
- 处理器数量
- 工作目录

- 资源使用情况（CPU、内存）
- 退出代码
- 节点详情（名称、实例 IDs、实例类型）

目录

先决条件

管理 AWS PCS 集群的 IAM 委托人必须允许该`pcs:AllowVendedLogDeliveryForResource`操作。

以下示例 IAM 策略授予所需的权限。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "PcsAllowVendedLogsDelivery",
         "Effect": "Allow",
         "Action": ["pcs:AllowVendedLogDeliveryForResource"],
         "Resource": [
            "arn:aws:pcs:*::cluster/*"
         ]
      }
   ]
}
```

设置任务完成日志

您可以使用 AWS 管理控制台 或为 AWS PCS 集群设置任务完成日志 AWS CLI。

AWS 管理控制台

使用控制台设置任务完成日志

1. 打开 AWS PCS 控制台。
2. 在导航窗格中，选择集群。
3. 选择要在其中添加任务完成日志的集群。
4. 在集群详细信息页面上，选择日志选项卡。
5. 在 “任务完成日志” 下，选择 “添加”，从日志、Amazon S3 和 Firehose 中添加最多 3 个 CloudWatch 日志传输目标。
6. 选择 “更新日志传送”。

AWS CLI

要使用设置任务完成日志 AWS CLI

1. 创建日志传输目标：

```
aws logs put-delivery-destination --region region \
  --name pcs-logs-destination \
  --delivery-destination-configuration \
  destinationResourceArn=resource-arn
```

进行如下替换：

- *region*— 您要在 AWS 区域 哪里创建目的地，例如 us-east-1
- *pcs-logs-destination*— 目的地的名称
- *resource-arn*— CloudWatch 日志组、S3 存储桶或 Firehose 传输流的亚马逊资源名称 (ARN)。

有关更多信息，请参阅 Amazon CloudWatch 日志 API 参考PutDeliveryDestination中的。

2. 将 PCS 集群设置为日志传输源：

```
aws logs put-delivery-source --region region \
  --name cluster-logs-source-name \
  --resource-arn cluster-arn \
  --log-type PCS_JOBCOMP_LOGS
```

进行如下替换：

- *region*— 您的 AWS 区域 集群的，例如 us-east-1
- *cluster-logs-source-name*— 来源的名称
- *cluster-arn*— 您的 PCS 集群的 AWS ARN

有关更多信息，请参阅 Amazon CloudWatch 日志 API 参考PutDeliverySource中的。

3. 将传送源连接到传送目的地：

```
aws logs create-delivery --region region \
  --delivery-source-name cluster-logs-source \
  --delivery-destination-arn destination-arn
```

进行如下替换：

- *region*— 那个 AWS 区域，比如 us-east-1
- *cluster-logs-source*— 您的配送来源的名称
- *destination-arn*— 您的配送目的地的 ARN

有关更多信息，请参阅 Amazon CloudWatch 日志 API 参考CreateDelivery中的。

如何查找任务完成日志

您可以在日志和 Amazon S3 中配置 CloudWatch 日志目标。 AWS PCS 使用以下结构化路径名和文件名。

CloudWatch 日志

AWS PCS 对 CloudWatch 日志流使用以下名称格式：

```
AWSLogs/PCS/cluster-id/jobcomp.log
```

例如：AWSLogs/PCS/pcs_abc123de45/jobcomp.log

Amazon S3

AWS PCS 对 S3 路径使用以下名称格式：

```
AWSLogs/account-id/PCS/region/cluster-id/jobcomp/year/month/day/hour/
```

例如：AWSLogs/111122223333/PCS/us-east-1/pcs_abc123de45/jobcomp/2025/06/19/11/

AWS PCS 对日志文件使用以下名称格式：

```
PCS_jobcomp_year-month-day-hour_cluster-id_random-id.log.gz
```

例如：PCS_jobcomp_2025-06-19-11_pcs_abc123de45_04be080b.log.gz

Job 完成日志字段

AWS PCS 将任务完成日志数据写为 JSON 对象。JSON jobcomp 容器包含任务详细信息。下表描述了jobcomp容器内的字段。有些字段仅在特定情况下才会出现，例如用于数组作业或异构作业。

Job 完成日志字段

Name	示例值	必需	注意
job_id	11	是	始终以价值为本
user	"root"	是	始终以价值为本
user_id	0	是	始终以价值为本
group	"root"	是	始终以价值为本
group_id	0	是	始终以价值为本
name	"wrap"	是	始终以价值为本
job_state	"COMPLETED"	是	始终以价值为本

Name	示例值	必需	注意
partition	"Hydra-MpiQueue-abcdef01-7"	是	始终以价值为本
time_limit	"UNLIMITED"	是	永远在场，但可能是 "UNLIMITED"
start_time	"2025-06-19T10:58:57"	是	永远在场，但可能是 "Unknown"
end_time	"2025-06-19T10:58:57"	是	永远在场，但可能是 "Unknown"
node_list	"Hydra-MpiNG-abcdef01-2345-1"	是	始终以价值为本
node_cnt	1	是	始终以价值为本
proc_cnt	1	是	始终以价值为本
work_dir	"/root"	是	永远在场，但可能是 "Unknown"
reservation_name	"weekly_maintenance"	是	始终存在，但可能是一个空字符串 ""
tres.cpu	1	是	始终以价值为本
tres.mem.val	600	是	始终以价值为本
tres.mem.unit	"M"	是	可以是"M"或 "bb"
tres.node	1	是	始终以价值为本

Name	示例值	必需	注意
tres.billing	1	是	始终以价值为本
account	"finance"	是	始终存在，但可能是一个空字符串 ""
qos	"normal"	是	始终存在，但可能是一个空字符串 ""
wc_key	"project_1"	是	始终存在，但可能是一个空字符串 ""
cluster	"unknown"	是	永远在场，但可能是 "unknown"
submit_time	"2025-06-19T10:55:46"	是	永远在场，但可能是 "Unknown"
eligible_time	"2025-06-19T10:55:46"	是	永远在场，但可能是 "Unknown"
array_job_id	12	否	仅当作业是阵列作业时才会出现
array_task_id	1	否	仅当作业是阵列作业时才会出现
het_job_id	10	否	仅当作业是异构作业时才会出现
het_job_offset	0	否	仅当作业是异构作业时才会出现
derived_exit_code_status	0	是	始终以价值为本
derived_exit_code_signal	0	是	始终以价值为本

Name	示例值	必需	注意
exit_code_status	0	是	始终以价值为本
exit_code_signal	0	是	始终以价值为本
node_details[0].name	"Hydra-MpiNG-abcdef01-2345-1"	否	永远在场，但node_details 可能是 "[]"
node_details[0].instance_id	"i-0abcdef01234567a"	否	永远在场，但node_details 可能是 "[]"
node_details[0].instance_type	"t4g.micro"	否	永远在场，但node_details 可能是 "[]"

作业完成日志示例

以下示例显示了各种作业类型和状态的任务完成日志：

```
{ "jobcomp": { "job_id": 1, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T16:32:57",
 "end_time": "2025-06-19T16:33:03", "node_list": "Hydra-MpiNG-abcdef01-2345-
[1-2]", "node_cnt": 2, "proc_cnt": 2, "work_dir": "/usr/bin", "reservation_name":
 "", "tres": { "cpu": 2, "mem": { "val": 1944, "unit": "M" }, "node": 2,
 "billing": 2 }, "account": "", "qos": "", "wc_key": "", "cluster": "unknown",
 "submit_time": "2025-06-19T16:29:40", "eligible_time": "2025-06-19T16:29:41",
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc123def45678", "instance_type": "t4g.micro" }, { "name":
 "Hydra-MpiNG-abcdef01-2345-2", "instance_id": "i-0def456abc78901", "instance_type":
 "t4g.micro" } ] } }
```

```
{ "jobcomp": { "job_id": 2, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T16:33:13",
 "end_time": "2025-06-19T16:33:14", "node_list": "Hydra-MpiNG-abcdef01-2345-
[1-2]", "node_cnt": 2, "proc_cnt": 2, "work_dir": "/usr/bin", "reservation_name":
 "", "tres": { "cpu": 2, "mem": { "val": 1944, "unit": "M" }, "node": 2,
 "billing": 2 }, "account": "", "qos": "", "wc_key": "", "cluster": "unknown",
 "submit_time": "2025-06-19T16:33:13", "eligible_time": "2025-06-19T16:33:13",
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc123def45678", "instance_type": "t4g.micro" }, { "name":
 "Hydra-MpiNG-abcdef01-2345-2", "instance_id": "i-0def456abc78901", "instance_type":
 "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 3, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T22:58:57", "end_time":
 "2025-06-19T22:58:57", "node_list": "Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 972, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T22:55:46",
 "eligible_time": "2025-06-19T22:55:46", "derived_exit_code_status": 0,
 "derived_exit_code_signal": 0, "exit_code_status": 0, "exit_code_signal":
 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1", "instance_id":
 "i-0abc234def56789", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 4, "user": "root", "user_id": 0, "group": "root",
 "group_id": 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-
MpiQueue-abcdef01-7", "time_limit": "525600", "start_time": "2025-06-19T23:04:27",
 "end_time": "2025-06-19T23:04:27", "node_list": "Hydra-MpiNG-abcdef01-2345-
[1-2]", "node_cnt": 2, "proc_cnt": 2, "work_dir": "/root", "reservation_name":
 "", "tres": { "cpu": 2, "mem": { "val": 1944, "unit": "M" }, "node": 2,
 "billing": 2 }, "account": "", "qos": "", "wc_key": "", "cluster": "unknown",
 "submit_time": "2025-06-19T23:01:38", "eligible_time": "2025-06-19T23:01:38",
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc234def56789", "instance_type": "t4g.micro" }, { "name":
 "Hydra-MpiNG-abcdef01-2345-2", "instance_id": "i-0def345abc67890", "instance_type":
 "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 5, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "FAILED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:09:00", "end_time":
 "2025-06-19T23:09:00", "node_list": "(null)", "node_cnt": 0, "proc_cnt": 0,
 "work_dir": "/root", "reservation_name": "", "tres": { "cpu": 1, "mem": { "val":
 1, "unit": "G" }, "node": 1, "billing": 1 }, "account": "", "qos": "", "wc_key":
 "", "cluster": "unknown", "submit_time": "2025-06-19T23:09:00", "eligible_time":
```

```
 "2025-06-19T23:09:00", "derived_exit_code_status": 0, "derived_exit_code_signal": 0,
 "exit_code_status": 0, "exit_code_signal": 1, "node_details": [] } }
{ "jobcomp": { "job_id": 6, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "CANCELLED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:09:36",
 "end_time": "2025-06-19T23:09:36", "node_list": "(null)", "node_cnt": 0, "proc_cnt":
 0, "work_dir": "/root", "reservation_name": "", "tres": { "cpu": 1, "mem":
 { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "", "qos":
 "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:09:35",
 "eligible_time": "2025-06-19T23:09:36", "het_job_id": 6, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status": 0,
 "exit_code_signal": 1, "node_details": [] } }
{ "jobcomp": { "job_id": 7, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "CANCELLED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:10:03",
 "end_time": "2025-06-19T23:10:03", "node_list": "(null)", "node_cnt": 0, "proc_cnt":
 0, "work_dir": "/root", "reservation_name": "", "tres": { "cpu": 1, "mem":
 { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "", "qos":
 "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:10:03",
 "eligible_time": "2025-06-19T23:10:03", "het_job_id": 7, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status": 0,
 "exit_code_signal": 1, "node_details": [] } }
{ "jobcomp": { "job_id": 8, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:11:24", "end_time":
 "2025-06-19T23:11:24", "node_list": "Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:11:23",
 "eligible_time": "2025-06-19T23:11:23", "het_job_id": 8, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc234def56789", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 9, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:11:24", "end_time":
 "2025-06-19T23:11:24", "node_list": "Hydra-MpiNG-abcdef01-2345-2", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:11:23",
 "eligible_time": "2025-06-19T23:11:23", "het_job_id": 8, "het_job_offset": 1,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-2",
 "instance_id": "i-0def345abc67890", "instance_type": "t4g.micro" } ] } }
```

```
{ "jobcomp": { "job_id": 10, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:12:24", "end_time":
 "2025-06-19T23:12:24", "node_list":"Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:12:14",
 "eligible_time": "2025-06-19T23:12:14", "het_job_id": 10, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc234def56789", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 11, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:12:24", "end_time":
 "2025-06-19T23:12:24", "node_list":"Hydra-MpiNG-abcdef01-2345-2", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 600, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:12:14",
 "eligible_time": "2025-06-19T23:12:14", "het_job_id": 10, "het_job_offset": 1,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-2",
 "instance_id": "i-0def345abc67890", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 13, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:47:57", "end_time":
 "2025-06-19T23:47:58", "node_list":"Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 972, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:43:56",
 "eligible_time": "2025-06-19T23:43:56" , "array_job_id": 12, "array_task_id": 1,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc345def67890", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 12, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:47:58", "end_time":
 "2025-06-19T23:47:58", "node_list":"Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 972, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:43:56",
 "eligible_time": "2025-06-19T23:43:56" , "array_job_id": 12, "array_task_id": 2,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
```

```
0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
"instance_id": "i-0abc345def67890", "instance_type": "t4g.micro" } ] } }
```

计划程序在 PCS 中 AWS 登录

您可以将 AWS PCS 配置为将详细的日志数据从集群计划程序发送到亚马逊 CloudWatch 日志、亚马逊简单存储服务 (Amazon S3) Service 和 Amazon Data Firehose。这可以帮助进行监控和故障排除。

目录

先决条件

管理 AWS PCS 集群的 IAM 委托人必须允许该`pcs:AllowVendedLogDeliveryForResource`操作。

以下示例 IAM 策略授予所需的权限。

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "PcsAllowVendedLogsDelivery",
         "Effect": "Allow",
         "Action": ["pcs:AllowVendedLogDeliveryForResource"],
         "Resource": [
            "arn:aws:pcs:*::cluster/*"
         ]
      }
   ]
}
```

设置调度日志

您可以使用 AWS 管理控制台 或 AWS CLI为 AWS PCS 集群设置调度程序日志。

AWS 管理控制台

使用控制台设置调度程序日志

1. 打开 AWS PCS 控制台。
2. 在导航窗格中，选择集群。
3. 选择要在其中添加调度程序日志的集群。
4. 在集群详细信息页面上，选择日志选项卡。
5. 在 “计划程序日志” 下，选择 “添加”，从日志、Amazon S3 和 Firehose 中添加最多 3 个 CloudWatch 日志传输目标。
6. 选择 “更新日志传送”。

AWS CLI

要使用设置调度程序日志 AWS CLI

1. 创建日志传输目标：

```
aws logs put-delivery-destination --region region \
  --name pcs-logs-destination \
  --delivery-destination-configuration \
  destinationResourceArn=resource-arn
```

进行如下替换：

- *region*— 您要在 AWS 区域 哪里创建目的地，例如 us-east-1
- *pcs-logs-destination*— 目的地的名称
- *resource-arn*— CloudWatch 日志组、S3 存储桶或 Firehose 传输流的亚马逊资源名称 (ARN)。

有关更多信息，请参阅 Amazon CloudWatch 日志 API 参考PutDeliveryDestination中的。

2. 将 PCS 集群设置为日志传输源：

```
aws logs put-delivery-source --region region \
  --name cluster-logs-source-name \
  --resource-arn cluster-arn \
  --log-type PCS_SCHEDULER_LOGS
```

进行如下替换：

- *region*— 您的 AWS 区域 集群的，例如 us-east-1
- *cluster-logs-source-name*— 来源的名称
- *cluster-arn*— 您的 PCS 集群的 AWS ARN

有关更多信息，请参阅 Amazon CloudWatch 日志 API 参考PutDeliverySource中的。

3. 将传送源连接到传送目的地：

```
aws logs create-delivery --region region \
  --delivery-source-name cluster-logs-source \
  --delivery-destination-arn destination-arn
```

进行如下替换：

- *region*— 那个 AWS 区域，比如 us-east-1
- *cluster-logs-source*— 您的配送来源的名称
- *destination-arn*— 您的配送目的地的 ARN

有关更多信息，请参阅 Amazon CloudWatch 日志 API 参考CreateDelivery中的。

调度器日志流路径和名称

AWS PCS 计划程序日志的路径和名称取决于目标类型。

- CloudWatch 日志
  - CloudWatch 日志流遵循此命名约定。

```
AWSLogs/PCS/${cluster_id}/${log_name}_${scheduler_major_version}.log
```

Example

```
AWSLogs/PCS/abcdef0123/slurmctld_24.05.log
```

- S3 存储桶
  - S3 存储桶输出路径遵循以下命名约定：

```
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 对象名称遵循以下约定：

```
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
```

调度器日志记录示例

AWS PCS 调度程序日志是结构化的。除了 Slurm 控制器进程发出的日志消息外，它们还包括集群标识符、调度器类型、主要版本和补丁版本等字段。见下列。

```
{
    "resource_id": "s3431v9rx2",
    "resource_type": "PCS_CLUSTER",
    "event_timestamp": 1721230979,
    "log_level": "info",
    "log_name": "slurmctld",
    "scheduler_type": "slurm",
    "scheduler_major_version": "25.05",
    "scheduler_patch_version": "3",
    "node_type": "controller_primary",
```

```
    "message": "[2024-07-17T15:42:58.614+00:00] Running as primary controller\n"
}
```

使用 Amazon 监控 AWS 并行计算服务 CloudWatch

Amaz CloudWatch on 通过定期从集群收集指标，监控您的 AWS 并行计算服务 (AWS PCS) 集群的运行状况和性能。这些指标将被保留，使您可以访问历史数据并深入了解集群在一段时间内的性能。

CloudWatch 还允许您监控 AWS PCS 启动的 EC2 实例，以满足您的扩展要求。虽然您可以检查正在运行的实例的日志，但 CloudWatch 指标和日志数据通常会在实例终止后删除。但是，您可以使用 EC2 启动模板在实例上配置 CloudWatch 代理，使其即使在实例终止后也能保留指标和日志，从而实现长期监控和分析。

浏览本节的主题，详细了解如何使用监控 AWS PCS CloudWatch。

主题

- 使用监控 AWS PCS 指标 CloudWatch
- 使用亚马逊监控 AWS PCS 实例 CloudWatch

使用监控 AWS PCS 指标 CloudWatch

您可以使用 Amazon 监控 AWS PCS 集群的运行状况 CloudWatch，它会从您的集群中收集数据并将其转换为近乎实时的指标。这些统计数据会保留 15 个月，因此您可以访问历史信息并更好地了解集群的性能。集群指标以 1 分钟 CloudWatch 为周期发送到。有关的更多信息 CloudWatch，请参阅 Amazon 是什么 CloudWatch？ 在《亚马逊 CloudWatch 用户指南》中。

AWS PCS 将以下指标发布到中的 AWS/PCS 命名空间。 CloudWatch它们只有一个维度，`ClusterId`。

Name	说明	单位
ActualCapacity	IdleCapacity + UtilizedCapacity	计数
CapacityUtilization	UtilizedCapacity / ActualCapacity	计数

Name	说明	单位
DesiredCapacity	ActualCapacity + PendingCapacity	计数
IdleCapacity	正在运行但未分配给任务的实例数	计数
UtilizedCapacity	正在运行并分配给任务的实例数	计数

使用亚马逊监控 AWS PCS 实例 CloudWatch

AWS PCS 会根据需要启动 Amazon EC2 实例，以满足您的 PCS 计算节点组中定义的扩展要求。您可以使用 Amazon 在这些实例运行时对其进行监控 CloudWatch。您可以通过登录实例并使用交互式命令行工具来检查正在运行的实例的日志。但是，默认情况下， CloudWatch 指标数据仅在实例终止后保留一段有限的时间，并且实例日志通常会与支持该实例的 EBS 卷一起删除。要保留终止后由 PCS 启动的实例的指标或日志数据，您可以使用 EC2 启动模板在您的实例上配置 CloudWatch 代理。本主题概述了监控正在运行的实例，并提供了如何配置永久性实例指标和日志的示例。

监控正在运行的实例

查找 AWS PCS 实例

要监控由 PCS 启动的实例，请查找与集群或计算节点组关联的正在运行的实例。然后，在给定实例的 EC2 控制台中，检查状态和警报以及监控部分。如果为这些实例配置了登录访问权限，则可以连接到这些实例并检查这些实例上的各种日志文件。有关识别哪些实例由 PCS 管理的更多信息，请参阅在 AWS PCS 中查找计算节点组实例。

启用详细指标

默认情况下，每隔 5 分钟收集一次实例指标。要每隔一分钟收集指标，请在计算节点组启动模板中启用详细 CloudWatch 监控。有关更多信息，请参阅 开启详细 CloudWatch 监控。

配置永久性实例指标和日志

您可以通过在实例上安装和配置 Amazon CloudWatch 代理，保留实例中的指标和日志。这包括三个主要步骤：

1. 创建 CloudWatch 代理配置。

2. 将配置存储在 PCS 实例可以检索的地方。
3. 编写一个 EC2 启动模板，用于安装 CloudWatch 代理软件、获取您的配置并使用配置启动 CloudWatch 代理。

有关更多信息，请参阅 Amazon CloudWatch 用户指南中的使用 CloudWatch 代理收集指标、日志和跟踪，以及在 AWS PCS 上使用亚马逊 EC2 启动模板。

创建 CloudWatch 代理配置

在您的实例上部署 CloudWatch 代理之前，您必须生成一个 JSON 配置文件，该文件指定要收集的指标、日志和跟踪。可以使用向导创建配置文件，也可以使用文本编辑器手动创建配置文件。将为本演示手动创建配置文件。

在安装了 AWS CLI 的计算机上，创建一个名为 config.json 的 CloudWatch 配置文件，其中包含以下内容。您也可以使用以下 URL 下载该文件的副本。

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/cloudwatch/assets/config.json
```

注意

- 示例文件中的日志路径适用于亚马逊 Linux 2。如果您的实例将使用不同的基础操作系统，请根据需要更改路径。
- 要捕获其他日志，请在下添加其他条目`collect_list`。
- 中的值`{brackets}`是模板化变量。有关支持变量的完整列表，请参阅 Amazon CloudWatch 用户指南中的手动创建或编辑 CloudWatch 代理配置文件。
- 您可以选择省略`logs`或`metrics`不想收集这些信息类型。

```
{
    "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
            }
        }
    }
}
```

此文件指示 CloudWatch 代理监视多个文件，这些文件可能有助于诊断引导、身份验证和登录以及其他疑难解答域中的错误。这些方法包括：

- `/var/log/cloud-init.log`— 实例配置初始阶段的输出
- `/var/log/cloud-init-output.log`— 实例配置期间运行的命令的输出
- `/var/log/amazon/pcs/bootstrap.log`— 在实例配置期间运行的 PC 特定操作的输出
- `/var/log/slurmd.log`— 来自 Slurm 工作负载管理器的守护程序 slurmd 的输出
- `/var/log/messages`— 来自内核、系统服务和应用程序的系统消息
- `/var/log/secure`— 与身份验证尝试相关的日志，例如 SSH、sudo 和其他安全事件

日志文件将发送到名为的 CloudWatch 日志组`/PCSLogs/instances`。日志流是实例 ID 和日志文件基本名称的组合。该日志组的保留时间为 30 天。

此外，该文件还指示 CloudWatch 代理收集几个常用指标，按实例 ID 汇总这些指标。

存储配置

CloudWatch 代理配置文件必须存储在 PCS 计算节点实例可以访问的地方。有两种常见的方法可以做到这一点。您可以将其上传到您的计算节点组实例可以通过其实例配置文件访问的 Amazon S3 存储桶，或者，您可以将其作为 SSM 参数存储在 Amazon Systems Manager 参数存储中。

上传到 S3 存储桶

要将文件存储在 S3 中，请使用下面的 AWS CLI 命令。在运行命令之前，请进行以下替换：

- *amzn-s3-demo-bucket*替换为您自己的 S3 存储桶名称

首先，（如果您已有存储桶，则这是可选的），创建一个存储桶来存放您的配置文件。

```
aws s3 mb s3://amzn-s3-demo-bucket
```

接下来，将文件上传到存储桶。

```
aws s3 cp ./config.json s3://amzn-s3-demo-bucket/
```

作为 SSM 参数存储

要将文件存储为 SSM 参数，请使用以下命令。在运行命令之前，请进行以下替换：

- *region-code*替换为您正在使用 AWS PCS 的 AWS 区域。
- （可选）将参数*AmazonCloudWatch-PCS*替换为您自己的名称。请注意，如果您更改名称的前缀，则需要在节点组实例配置文件中专门添加对 SSM 参数的读取权限。AmazonCloudWatch-

```
aws ssm put-parameter \
   --region region-code \
   --name "AmazonCloudWatch-PCS" \
   --type String \
   --value file://config.json
```

编写 EC2 启动模板

启动模板的具体细节取决于您的配置文件存储在 S3 还是 SSM 中。

使用存储在 S3 中的配置

此脚本安装 CloudWatch 代理，从 S3 存储桶导入配置文件，然后使用该文件启动 CloudWatch 代理。用您自己的详细信息替换此脚本中的以下值：

- *amzn-s3-demo-bucket*— 您的账户可以读取的 S3 存储桶的名称
- */config.json*— 相对于存储配置的 S3 存储桶根目录的路径

```
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://amzn-s3-demo-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==--
```

节点组的 IAM 实例配置文件必须有权访问存储桶。以下是上述用户数据脚本中存储桶的 IAM 策略示例。

JSON

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

另请注意，这些实例必须允许流向 S3 和 CloudWatch终端节点的出站流量。这可以使用安全组或 VPC 终端节点来实现，具体取决于您的集群架构。

使用存储在 SSM 中的配置

此脚本安装 CloudWatch 代理，从 SSM 参数导入配置文件，然后使用该文件启动 CloudWatch 代理。用您自己的详细信息替换此脚本中的以下值：

- （可选）将参数*AmazonCloudWatch-PCS*替换为您自己的名称。

```
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==--
```

节点组的 IAM 实例策略必须CloudWatchAgentServerPolicy附加。

如果您的参数名称不是以开头，AmazonCloudWatch-则需要在节点组实例配置文件中专门添加对 SSM 参数的读取权限。以下是一个 IAM 策略示例，它说明了前缀的相关内容*DOC-EXAMPLE-PREFIX*。

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "CustomCwSsmMParamReadOnly",
      "Effect" : "Allow",
      "Action" : [
        "ssm:GetParameter"
      ],
      "Resource" : "arn:aws:ssm:*:*:parameter/DOC-EXAMPLE-PREFIX*"
    }
```

```
  ]
}
```

另请注意，这些实例必须允许流向 SSM 和 CloudWatch终端节点的出站流量。这可以使用安全组或 VPC 终端节点来实现，具体取决于您的集群架构。

使用记录 AWS 并行计算服务 API 调用 AWS CloudTrail

AWS PCS 与 AWS CloudTrail一项服务集成，该服务提供用户、角色或 AWS 服务在 AWS PCS 中执行的操作的记录。 CloudTrail 将 AWS PCS 的所有 API 调用捕获为事件。捕获的调用包括来自 AWS PCS 控制台的调用和对 AWS PCS API 操作的代码调用。如果您创建了跟踪，则可以允许将 CloudTrail 事件持续传输到 Amazon S3 存储桶，包括 AWS PCS 的事件。如果您未配置跟踪，您仍然可以在 CloudTrail 控制台的 “事件历史记录” 中查看最新的事件。使用收集的信息 CloudTrail，您可以确定向 AWS PCS 发出的请求、发出请求的 IP 地址、谁发出了请求、何时发出请求以及其他详细信息。

要了解更多信息 CloudTrail，请参阅《AWS CloudTrail 用户指南》。

AWS PCS 信息在 CloudTrail

CloudTrail 在您创建账户 AWS 账户 时已在您的账户上启用。当 AWS PCS 中发生活动时，该活动会与其他 AWS 服务 CloudTrail 事件一起记录在事件历史记录中。您可以在中查看、搜索和下载最近发生的事件 AWS 账户。有关更多信息，请参阅使用事件历史记录查看 CloudTrail 事件。

要持续记录您的 AWS 账户事件（包括 AWS PCS 的事件），请创建跟踪。跟踪允许 CloudTrail 将日志文件传输到 Amazon S3 存储桶。预设情况下，在控制台中创建跟踪记录时，此跟踪记录应用于所有 AWS 区域。跟踪记录 AWS 分区中所有区域的事件，并将日志文件传送到您指定的 Amazon S3 存储桶。此外，您可以配置其他 AWS 服务，以进一步分析和处理 CloudTrail 日志中收集的事件数据。有关更多信息，请参阅下列内容：

- 创建跟踪记录概述
- CloudTrail 支持的服务和集成
- 配置 Amazon SNS 通知 CloudTrail
- 接收来自多个区域的 CloudTrail 日志文件和接收来自多个账户的 CloudTrail 日志文件

所有 AWS PCS 操作均由《 CloudTrail 并行计算服务 API 参考》记录并记录在《AWS 并行计算服务 API 参考》中。例如，对CreateComputeNodeGroupUpdateQueue、和DeleteCluster操作的调用会在 CloudTrail 日志文件中生成条目。

每个事件或日志条目都包含有关生成请求的人员信息。身份信息有助于您确定以下内容：

- 请求是使用根证书还是 AWS Identity and Access Management (IAM) 用户凭证发出。
- 请求是使用角色还是联合用户的临时安全凭证发出的。
- 请求是否由其他 AWS 服务发出。

有关更多信息，请参阅 CloudTrail userIdentity 元素。

了解来自 AWS PCS 的 CloudTrail 日志文件条目

跟踪是一种配置，允许将事件作为日志文件传输到您指定的 S3 存储桶。 CloudTrail 日志文件包含一个或多个日志条目。事件代表来自任何来源的单个请求，包括有关请求的操作、操作的日期和时间、请求参数等的信息。 CloudTrail 日志文件不是公共 API 调用的有序堆栈跟踪，因此它们不会按任何特定的顺序出现。

以下示例显示了CreateQueue操作的 CloudTrail 日志条目。

```
{
    "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"
}
```

AWS PCS 的终端节点和服务配额

以下各节介绍 AWS 并行计算服务 (AWS PCS) 的终端节点和服务配额。服务配额（以前称为限制）是您的服务资源或操作的最大数量 AWS 账户。

您的每项 AWS 服务 AWS 账户 都有默认配额。除非另有说明，否则，每个限额是区域特定的。您可以请求增加某些配额，但其他一些配额无法增加。

有关更多信息，请参阅《AWS 一般参考》中的 AWS 服务配额。

目录

服务端点

区域名称	区域	端点	协议
美国东部（俄亥俄州）	us-east-2	pcs.us-east-2.amazonaws.com pcs-fips.us-east-2.amazonaws.com pcs-fips.us-east-2.api.aws pcs.us-east-2.api.aws	HTTPS
美国东部（弗吉尼亚州北部）	us-east-1	pcs.us-east-1.amazonaws.com pcs-fips.us-east-1.amazonaws.com	HTTPS

区域名称	区域	端点	协议
		pcs-fips.us-east-1.api.aws pcs.us-east-1.api.aws	
美国西部（俄勒冈州）	us-west-2	pcs.us-west-2.amazonaws.com pcs-fips.us-west-2.amazonaws.com pcs-fips.us-west-2.api.aws pcs.us-west-2.api.aws	HTTPS
亚太地区（孟买）	ap-south-1	pcs.ap-south-1.amazonaws.com pcs.ap-south-1.api.aws	HTTPS
亚太地区（新加坡）	ap-southeast-1	pcs.ap-southeast-1.amazonaws.com pcs.ap-southeast-1.api.aws	HTTPS
亚太地区（悉尼）	ap-southeast-2	pcs.ap-southeast-2.amazonaws.com pcs.ap-southeast-2.api.aws	HTTPS
亚太地区（东京）	ap-northeast-1	pcs.ap-northeast-1.amazonaws.com pcs.ap-northeast-1.api.aws	HTTPS

区域名称	区域	端点	协议
欧洲地区（法兰克福）	eu-central-1	pcs.eu-central-1.amazonaws.com pcs.eu-central-1.api.aws	HTTPS
欧洲地区（爱尔兰）	eu-west-1	pcs.eu-west-1.amazonaws.com pcs.eu-west-1.api.aws	HTTPS
欧洲地区（伦敦）	eu-west-2	pcs.eu-west-2.amazonaws.com pcs.eu-west-2.api.aws	HTTPS
欧洲地区（巴黎）	eu-west-3	pcs.eu-west-3.amazonaws.com pcs-eu-west-3.api.aws	HTTPS
欧洲地区（米兰）	eu-south-1	pcs.eu-south-1.amazonaws.com pcs-eu-south-1.api.aws	HTTPS
欧洲地区（斯德哥尔摩）	eu-north-1	pcs.eu-north-1.amazonaws.com pcs.eu-north-1.api.aws	HTTPS

区域名称	区域	端点	协议
AWS GovCloud（美国东部）	us-gov-east-1	pcs。 us-gov-east-1.amazonaws.com pcs-fips。 us-gov-east-1.amazonaws.com pcs-fips。 us-gov-east-1.api.aws pcs。 us-gov-east-1.api.aws	HTTPS
AWS GovCloud（美国西部）	us-gov-west-1	pcs。 us-gov-west-1.amazonaws.com pcs-fips。 us-gov-west-1.amazonaws.com pcs-fips。 us-gov-west-1.api.aws pcs。 us-gov-west-1.api.aws	HTTPS

服务限额

名称	默认	可调整	描述
集群	5	是	每个集群的最大数量 AWS 区域。

> **Note**
>
> 默认值为由设置的初始配额 AWS。这些默认值与实际应用的限额值和最大可能的服务限额是分开的。有关更多信息，请参阅《服务限额用户指南》中的服务限额中的术语。

这些服务配额列在中的AWS 并行计算服务 (PCS) 下AWS 管理控制台。要为显示为可调整的值申请增加配额，请参阅 Servic e Quotas 用户指南中的申请增加配额。

> **Important**
>
> 记得在中查看当前 AWS 区域 设置 AWS 管理控制台。

内部配额

以下配额属于内部配额且不可调整。

名称	默认	可调整	描述
并发集群创建	1	否	每个 AWS 区域处于 `Creating` 状态的最大集群数。
计算每个集群的节点组	10	否	每个集群的最大计算节点组数。
每个集群的队列	10	否	每个集群的最大队列数。

其他 AWS 服务的相关配额

AWS PCS 使用其他 AWS 服务。这些服务的服务配额会影响您对 AWS PCS 的使用。

影响 AWS PCS 的 Amazon EC2 服务配额

- 竞价型实例请求
- 正在运行的按需型实例

- 启动模板
- 启动模板版本
- Amazon EC2 API 请求

有关更多信息，请参阅《亚马逊弹性计算云用户指南》中的 Amazon EC2 服务配额。

对 AWS 并行计算服务中的问题进行故障排除

以下主题提供了解决您在 AWS PCS 中可能遇到的一些问题的指导。

AWS PCS 中的 EC2 实例在重启后终止并被替换

问题概述

计算节点组中的 EC2 实例重启后，AWS PCS 会自动终止并替换该实例。

为什么会发生这种情况

AWS PCS 不支持实例重启。如果 EC2 实例重启，AWS PCS 会认为该实例运行状况不佳并替换它。如果 AWS PCS 持续终止并替换您的实例，则可能是因为您的实例启动后某些东西会重新启动。一些示例包括 EC2 实例上的自动化重启（例如修补后自动重启）、EC2 实例外部的自动化（例如网络管理应用程序）、其他 AWS 服务（例如 AWS Systems Manager），或者由个人手动重启。

操作

您可以查看`slurmctld`或`slurmd`日志，查看您的实例是否已重启。有关更多信息，请参阅计划程序在 PCS 中 AWS 登录和使用亚马逊监控 AWS PCS 实例 CloudWatch。以下示例`slurmctld`日志条目表示实例已重启：

Example

```
[2024-09-12T06:42:50.393+00:00] validate_node_specs: Node Login-1 unexpectedly rebooted
 boot_time=1726123354 last response=1726123285
```

由于正在进行补丁而重新启动

应用补丁后，通常需要重新启动。不要将补丁直接应用于属于 AWS PCS 计算节点组的 EC2 实例。如果您必须修补 EC2 实例，则应将补丁应用于更新的亚马逊系统映像 (AMI)，并更新您的计算节点组以使用更新后的 AMI。 AWS PCS 为这些计算节点组启动的新 EC2 实例将使用更新的（已修补的）AMI。有关更多信息，请参阅 适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

对 PCS 中的计算节点引导和注册问题进行故障排除 AWS

当计算节点无法引导或无法正确注册到您的 AWS PCS 集群时，您可能会遇到以下症状：

- 工作还没开始
- 您无法连接到中的实例 AWS Systems Manager
- 实例意外关闭
- 实例会不断被替换

这些故障可能是由 EC2 实例启动期间或 AWS PCS 计算节点引导过程中的问题引起的。本主题介绍帮助您解决 AWS PCS 节点引导过程中出现的问题的步骤。有关排除 EC2 实例启动故障的更多信息，请参阅《亚马逊弹性计算云用户指南》中的 Amazon EC2 实例启动问题疑难解答。

当 EC2 实例成功启动但在加入 AWS PCS 集群的过程中失败时，就会发生引导失败。引导过程包括两个主要阶段：

- 节点注册 — EC2 实例调用 RegisterComputeNodeGroupInstance AWS PCS API 操作向 AWS PCS 服务注册。失败可能是由于以下问题造成的：
  - Permissions
    - 错误的实例配置文件
  - Networking
    - 无法连接到 AWS PCS 端点
    - AWS PCS 端点配置错误
    - 没有公有 IP 的公有子网中的实例
    - 公有子网中的多 NIC 实例
  - 集群密钥
    - 集群密钥已被删除或标记为删除
- Slurm 集成 — 实例运行`slurmd`并加入 Slurm 集群。失败可能是由于以下问题造成的：

- Permissions
  - 安全组配置
  - Slurmctld 无法 ping 计算节点
- 自定义 AMI 设置
  - 缺少英伟达驱动程序
  - ResumeTimeout 到达

Slurm 在 PCS 上的工作原理 AWS

将 Slurm 的标准工作方式与 Slurm 在 PCS 上的工作方式进行比较可能会有所帮助。 AWS

标准 Slurm 作业处理

在标准 Slurm 作业处理中会执行以下步骤：

1. 当您提交任务时，会slurmctld验证该任务并对其进行排队。
2. 当资源可用时，slurmctld分配现有节点。
3. slurmd守护程序在分配的节点上运行作业。

在 PCS 上处理 Slurm 作业 AWS

AWS PCS 作业处理中将执行以下步骤：

1. 当您提交任务时，会slurmctld验证该任务并对其进行排队。
2. 当需要更多容量时， AWS PCS 会使用计算节点组的启动模板来启动新的 EC2 实例。
3. 新实例引导进入集群：
   a. 实例向 AWS PCS 注册。
   b. 实例加入 Slurm 集群。
4. 资源准备就绪后，slurmctld分配节点（包括新引导的节点）。
5. slurmd守护程序在分配的节点上运行作业。

检索实例日志

解决计算节点引导问题的第一步是检索实例日志。您可以使用以下方法之一：

AWS CLI

使用以下命令从计算节点检索控制台输出：

```
aws ec2 get-console-output --region us-east-1 --instance-id i-1234567890abcdef0 --
output text
```

*us-east-1*替换为您所在的 AWS 地区*i-1234567890abcdef0*和您的实例 ID。

AWS Systems Manager

如果您可以使用 Systems Manager 连接到实例，则可以直接查看引导日志文件：

1. 使用 Systems Manager 连接到实例。有关更多信息，请参阅《S ystems Manager 用户指南》中的启动会话。
2. 查看引导日志文件：

```
sudo cat /var/log/amazon/pcs/bootstrap.log
```

> **Note**
>
> 如果在初始化阶段出现问题，则可能需要等待大约 20 分钟才能连接到实例。Systems Manager 和 SSH 服务仅在初始化完成后启动，或者在出现故障时引导执行达到超时时时启动。

从实例 ID 中检索VPC/Subnet/Security组

要解决计算节点的问题，您可能需要检索与您的实例关联的 VPC、子网和安全组的信息。如果您不知道自己的实例 IDs，请参阅在 AWS PCS 中查找计算节点组实例。

AWS 管理控制台

获取 VPC、子网和安全组

1. 打开 Amazon EC2 控制台。
2. 选择实例。
3. 在实例表中，选择实例 ID。
4. 在显示的实例摘要中找到该实例的 VPC ID 和子网 ID。

5. 在实例摘要中，选择安全选项卡。
6. 在 “安全” 选项卡中找到 “安全组”。

AWS CLI

使用以下命令检索您的实例的 VPC、子网和安全组信息：

```
aws ec2 describe-instances --instance-ids i-1234567890abcdef0 --query
 'Reservations[*].Instances[*].
{InstanceId:InstanceId,VpcId:VpcId,SubnetId:SubnetId,SecurityGroups:SecurityGroups[*].GroupI
 --output table
```

节点注册问题

节点注册是计算节点在引导期间执行的第一个操作。该节点调用 AWS PCS API 端点向 AWS PCS 注册自己。注册失败通常会显示类似于以下内容的错误消息：

```
<13>Nov 13 16:23:50 user-data: [2025-11-13T16:23:50.510+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Registering node to cluster <clusterId>
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.192+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Retriable exception detected.
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.193+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Response is [specific error message]
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.194+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Retrying in 31 seconds...
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.192+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Retriable exception detected.
...
<13>Nov 13 16:25:18 user-data: [2025-11-13T16:25:18.195+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Registration timeout (600 seconds) reached. Exiting.
<13>Nov 13 16:25:18 user-data: [2025-11-13T16:25:18.200+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: ERROR: Error: (2) occurred on line 1 when running /opt/aws/pcs/
bin/pcs_bootstrap_init.sh. Shutting down instance.
```

错误的实例配置文件

如果节点由于错误的实例配置文件而无法注册，您将看到以下错误：

```
<13>Nov 13 18:43:08 user-data: [2025-11-13T18:43:08.268+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Response is {
```

```
<13>Nov 13 18:43:08 user-data:   "__type":
 "com.amazon.coral.service#AccessDeniedException",
<13>Nov 13 18:43:08 user-data:   "Message": "User: arn:aws:sts::<accountId>:assumed-
role/<roleName>/<instanceId> is not authorized to perform:
 pcs:RegisterComputeNodeGroupInstance on resource:
 arn:aws:pcs:<regionCode>:<accountId>:cluster/<clusterId> as either the resource does
 not exist, some policy explicitly denies access, or no policy grants access",
<13>Nov 13 18:43:08 user-data:   "nodeID": null
<13>Nov 13 18:43:08 user-data: }
```

验证与计算节点关联的实例配置文件是否具有`pcs:RegisterComputeNodeGroupInstance`权限。有关如何创建有效实例配置文件的更多信息，请参阅为 AWS PCS 创建实例配置文件。

无法连接到 AWS PCS 端点

如果您的计算节点位于私有子网中，请确保您已为 PC AWS S 配置了 VPC 终端节点，或者您的子网具有通往 NAT 网关的路由以进行互联网访问。有关更多信息，请参阅下列内容：

- 使用 Amazon Virtual Private Cloud AWS PrivateLink 指南中的接口 VPC 终端节点访问 AWS 服务。
- AWS PCS 的终端节点和服务配额.
- 在《亚马逊虚拟私有云用户指南》中将您的 VPC 连接到其他网络
- AWS PCS 联网

AWS PCS 端点配置错误

如果您看到类似于以下内容的错误消息，请验证与您的 AWS PCS VPC 终端节点关联的策略：

```
com.amazon.coral.security.AccessDeniedException: User: arn:aws:sts::xxx:assumed-
role/<roleName>/<instanceId> is not authorized to perform:
 pcs:RegisterComputeNodeGroupInstance on resource:
 arn:aws:pcs:<regionCode>:<accountId>:cluster/<clusterId> as either the resource does
 not exist, some policy explicitly denies access, or no policy grants access
```

有关如何为 PC AWS S 配置 VPC 接口终端节点的更多信息，请参阅AWS 并行计算服务 使用接口端点进行访问 (AWS PrivateLink)。

没有公有 IP 的公有子网中的实例

如果您的子网未启用自动分配公有 IP，并且您的路由配置使用互联网网关，则实例将无法与 AWS PCS API 通信。

带有互联网网关的子网中的实例必须具有公有 IP 地址。要解决此问题，请选择以下选项之一：

- 将 PCS 的 VP AWS C 终端节点添加到您的集群 VPC。这使实例能够与 AWS PCS 通信，而无需公有 IP 地址通过互联网网关。
- 使用带有 NAT 网关的私有子网，这样就不需要公有 IP 地址。
- 通过您的子网或启动模板启用自动公有 IP 地址分配，以便实例可以通过互联网网关联系 API。请注意，此选项对多网络接口实例无效。

公有子网中的多 NIC 实例

如果您使用的实例类型具有多个网络接口，则必须使用私有子网（NICs）。

AWS 公有 IP 地址只能分配给使用单个网络接口启动的实例。有关 IP 地址的更多信息，请参阅《适用于 Linux 实例的 Amazon EC2 用户指南》中的 “在实例启动期间分配公有 IPv4 地址”。

多 NIC 实例类型需要子网中的 NAT 网关或内部代理才能访问 AWS PCS 终端节点。或者，您可以将用于 PC AWS S 的 VPC 终端节点添加到您的集群 VPC。

集群密钥已被删除或标记为删除

如果 Secr AWS ets Manager 中的 Slurm 共享密钥已被删除或标记为删除，则计算节点将无法注册，您的集群将受到损害。

AWS 当你创建集群时，PCS 会自动在 Secr AWS ets Manager 中创建 Slurm 共享密钥（名称格式:`pcs!slurm-secret-<cluster-id>`）。此密钥是集群中安全通信所必需的。有关更多信息，请参阅 在 AWS PCS 中使用集群密钥。

如果此密钥被删除或标记为删除，则新节点将无法加入集群，并且控制器或其他集群守护程序（例如`slurmd`和`slurmdbd`）在重新启动后可能无法重新加入集群。

要解决此问题，如果已删除的密钥仍在恢复窗口内，则可以将其恢复。有关详细说明，请参阅恢复 S AWS ecrets Manager 密钥。

如果恢复窗口过期，则无法恢复密钥，也无法恢复受影响的 AWS PCS 集群。您需要使用相同的配置创建一个新集群。 AWS PCS 会自动创建新的调度程序密钥。

Slurm 集群加入问题

成功注册节点后，计算节点将尝试加入 Slurm 集群。节点`slurmd`上的守护程序与 Slurm 控制器联系，以便在集群中注册。Slurm 加入失败通常会显示类似于以下内容的错误消息：

```
<13>Nov  5 17:20:29 user-data: [2024-11-05T17:20:28+00:00] FATAL:
 Mixlib::ShellOut::ShellCommandFailed: service[slurmd] (aws-pcs-slurm::finalize_slurm
 line 18) had an error: Mixlib::ShellOut::ShellCommandFailed: Expected process to exit
 with [0], but received '1'
<13>Nov  5 17:20:29 user-data: ---- Begin output of ["/usr/bin/systemctl", "--system",
 "start", "slurmd"] ----
<13>Nov  5 17:20:29 user-data: STDOUT:
<13>Nov  5 17:20:29 user-data: STDERR: Job for slurmd.service failed because the
 control process exited with error code. See "systemctl status slurmd.service" and
 "journalctl -xe" for details.
<13>Nov  5 17:20:29 user-data: ---- End output of ["/usr/bin/systemctl", "--system",
 "start", "slurmd"] ----
```

安全组配置

确认您的安全组配置正确，允许计算节点和 Slurm 控制器之间进行通信。安全组必须允许以下流量：

- 用于`slurmd`与之通信的端口 6817 `slurmctld`
- 端口 6818 用于 pin `slurmctld` g `slurmd`

有关安全组要求的更多信息，请参阅以下主题：

- 为 AWS PCS 创建安全组
- 为 AWS PCS 创建启动模板
- 安全组要求和注意事项

> **Important**
>
> 在创建集群时与集群关联的集群安全组也必须在计算节点组安全组中进行配置，以允许计算节点与控制器通信。

缺少英伟达驱动程序

如果实例启动正确，但作业未启动，并且您在实例日志中看到类似于以下内容的错误消息，则可能缺少 NVIDIA 驱动程序：

```
<13>Dec  2 13:52:00 user-data: [2024-12-02T13:52:00.094+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_config_always.sh: INFO: nvidia-smi not found!
```

```
...
<13>Dec  2 13:54:10 user-data: Job for slurmd.service failed because the control
 process exited with error code. See "systemctl status slurmd.service" and "journalctl
 -xe" for details.
<13>Dec  2 13:54:12 user-data: [2024-12-02T13:54:12.718+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_finalize.sh: INFO: systemctl could not start slurmd!
```

如果您连接到实例并检查slurmd守护程序状态，则可能会看到类似于以下内容的错误：

```
$ systemctl status slurmd
...
fatal: can't stat gres.conf file /dev/nvidia0: No such file or directory
```

要解决此问题，请在您的自定义 AMI 上安装 NVIDIA 驱动程序。有关更多信息，请参阅 步骤 4-（可选）安装其他驱动程序、库和应用程序软件。

ResumeTimeout 到达

如果计算节点及其 EC2 实例因节点运行状况不佳而终止，则 AWS PCS 可能不支持 AMI 或可能存在网络问题。EC2 实例运行大约 30 分钟，直到到达 Slurm 并将该节点标记 ResumeTimeout 为。DOWN

如果实例无法正确引导，也未在 AWS PCS 中注册（未RegisterComputeNodeGroupInstance调用 EC2 实例），请检查您的实例日志中是否有类似于以下内容的错误消息：

```
/opt/aws/pcs/bin/pcs_bootstrap_init.sh: No such file or directory
```

此错误表明 AWS PCS 引导软件不是 AMI 的一部分。要解决此问题，请确保您的自定义 AMI 包含 AWS PCS 引导软件。有关更多信息，请参阅 适用于 AWS PCS 的自定义 Amazon 机器映像 (AMIs)。

Slurmctld 无法 ping 计算节点

如果实例正确执行了引导程序并已在 AWS PCS 中注册，但slurmctld无法看到它并向其提交作业，则该实例将在一段时间DOWN后设置为，然后终止。

这可能是由于安全组配置错误造成的。例如，如果已启用端口 6817 slurmd 以允许与之通信slurmctld，但缺少端口 6818 slurmctld 以允许 ping 通。slurmd

验证您的安全组是否包含所有必需的规则，如中所述安全组要求和注意事项。

解决由于 MaxJobCount 限制导致的作业提交失败的问题

问题：Job 提交失败，并显示以下错误消息：

```
sbatch: error: Slurm temporarily unable to accept job, sleeping and retrying
```

即使正在运行和待处理的任务数量似乎远低于集群的任务限制，也会发生此错误。

原因：该`MaxJobCount`限制包括 Slurm 跟踪的所有作业，而不仅仅是正在运行或待处理的作业。已完成的任务会在 Slurm 的内存中保留一段时间（默认为 5 分钟），然后才会被清除。在高作业吞吐量期间，活动任务加上最近完成的任务的总数可能会超过限制。

您可以通过在群集节点上运行以下命令来验证任务总数：

```
scontrol show jobs | grep -c JobId
```

这显示了 Slurm 正在跟踪的任务总数，包括等待清除的已完成任务。

解决方案：考虑以下方法之一：

- 创建更大的集群-如果您的工作负载一直需要更多的并发作业，请创建一个规模更大的新集群。有关集群大小及其限制的更多信息，请参阅AWS PCS 中的集群大小。
- 降低作业提交率 — 调整作业提交脚本，以较慢的速度提交作业，从而使已完成的作业时间从 Slurm 的跟踪中清除。

AWS PCS 用户指南的文档历史记录

下表描述了 AWS PCS 文档的重要更改。

日期	更改	文档更新	更新的 API 版本
2026年4月16日	AWS PCS 在欧洲（米兰）上市	AWS PCS 现已在欧洲（米兰）（eu-south-1）上市。 CloudFormation 欧洲（米兰）有模板可供入门 AWS 区域。有关更多信息，请参阅CloudFormation 用于创建示例 AWS PCS 集群和CloudFormation 用于创建示例 AWS PCS 集群的模板。	不适用
2026 年 3 月 10 日	更新了 PCS 代理	更新了 AWS PCS 代理 1.3.2-1 的 AMI 主题。修复了影响 RHEL 8.10 和 Rocky Linux 8.10 计算节点引导程序的问题。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和AWS PCS 代理版本。	不适用
2026 年 2 月 11 日	AWS PCS 在亚太地区（孟买）和欧洲（巴黎）上市	AWS PCS 现已在亚太地区（孟买）（ap-south-1）和欧洲（巴黎）（eu-west-3）上市。 CloudFormation 亚太地区（孟买） AWS 区域 和欧洲（巴黎）均有	不适用

日期	更改	文档更新	更新的 API 版本
		模板可供入门 AWS 区域。有关更多信息，请参阅CloudFormation 用于创建示例 AWS PCS 集群和CloudFormation 用于创建示例 AWS PCS 集群的模板。	
2025 年 11 月 18 日	新功能：Slurm REST API	Slurm 25.05 或更高版本现在支持 Slurm REST API。有关更多信息，请参阅 PCS 中的 Slurm REST API AWS。	AWS 软件开发工具包：2025-11-18
2025 年 11 月 17 日	更新了 Slurm 安装程序	更新了 AWS PCS Slurm 安装程序 24.11.7-1 和 25.05.5-1 的 AMI 主题。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和AWS PCS 代理版本。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 11 月 10 日	新功能：支持 Slurm CLI 过滤器插件	AWS PCS 现在支持 Slurm CLI 过滤器插件来运行自定义 Lua 脚本，这些脚本可以在作业提交参数到达 Slurm 控制器之前对其进行验证和修改。使用 CLI 筛选器强制执行自定义策略，设置默认参数，并在作业提交期间提供用户指导。此功能需要 Slurm 版本 25.05 或更高版本。有关更多信息，请参阅 使用 Slurm CLI 过滤器插件在 PCS 中自定义作业提交 AWS。	不适用
2025 年 11 月 7 日	更新了 PCS 代理	更新了 AWS PCS 代理 1.3.1-1 的 AMI 主题。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和AWS PCS 代理版本。	不适用
2025 年 11 月 3 日	更新了 PCS 代理和 Slurm 安装程序	更新了 AWS PCS 代理 1.3.0-1 和 Slurm 安装程序 24.11.6-2、24.05.8-2 和 23.11.10-4 的 AMI 主题。更新了支持的操作系统列表。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和AWS PCS 代理版本。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 10 月 23 日	更新内容： pcs-multi-cluster-login-configure.sh	修复了多集群登录节点配置脚本中的一些错误。有关更多信息，请参阅 AWS PCS 多集群登录节点配置脚本代码。	不适用
2025 年 10 月 21 日	新功能：集群密钥轮换	AWS PCS 现在支持集群密钥轮换，以增强安全性。有关更多信息，请参阅 在 AWS PCS 中轮换集群密钥。 更新了最低管理员权限以支持集群密钥轮换。有关更多信息，请参阅 AWS PCS 的最低权限。	不适用
2025 年 10 月 17 日	新主题：多集群登录节点配置脚本	添加了一个新主题，该主题提供了配置独立登录节点以连接到多 AWS 个 PCS 集群的脚本。该脚本可自动配置多个 Slurm `sackd` 守护程序，并为集群交互创建激活脚本。 有关更多信息，请参阅 将独立登录节点连接到 AWS PCS 中的多个集群。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 10 月 16 日	已针对 Slurm 25.05 进行了更新	更新了 Slurm 25.05 支持的用户指南。Slurm 25.05 现在是默认版本。有关更多信息，请参阅下列内容： • PCS 中的 Slurm 版本 AWS • 要为 PCS 定制 AMIs 版本的软件安装程序 AWS • AWS PCS 示例的发行说明 AMIs	不适用
2025 年 10 月 16 日	更新了 PCS 代理	更新了 AWS PCS 代理 1.2.2-1 的 AMI 主题。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和AWS PCS 代理版本。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 10 月 2 日	新功能：Slurm 节点重启、集群更新和自定义 Slurm 设置	AWS PCS 增加了对多项新功能的支持： • Slurm 节点重启 — 使用 Slurm 的本机`scontrol reboot`命令重启计算节点，无需更换实例。有关更多信息，请参阅在 PCS 中使用 Slurm 重启计算节点 AWS。 • 集群更新-在创建后修改集群配置，无需重新构建。有关更多信息，请参阅 在 AWS PCS 中更新集群。 • Slurm 自定义设置 — 跨集群、队列和计算节点组资源配置高级 Slurm 参数。有关更多信息，请参阅 在 PCS 中配置自定义 Slurm 设置 AWS。	2025-10-01
2025 年 9 月 23 日	新的疑难解答主题：计算节点引导问题	添加了用于诊断和解决计算节点引导问题的故障排除指南。有关更多信息，请参阅 对 PCS 中的计算节点引导和注册问题进行故障排除 AWS。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 9 月 17 日	新功能：机器学习的容量块	AWS PCS 现在支持适用于 ML 的 Amazon EC2 容量块，这使您能够为集群预留基于 GPU 的加速计算实例。有关更多信息，请参阅 将 Amazon EC2 容量块用于带有 AWS PCS 的机器学习。 现在，支持容量块的最低权限已成为服务管理员最低权限的一部分。有关更多信息，请参阅 AWS PCS 的最低权限。	2025-09-17
2025 年 9 月 11 日	AWS 托管政策更新	AWS PCS 更新了 AWSPCSServiceRolePolicy 以支持容量块。有关更多信息，请参阅 AWS AWS 并行计算服务的托管策略。	不适用
2025 年 8 月 14 日	更新了实例配置文件文档	使用有关创建 IAM 角色和实例配置文件的全面 CLI 说明增强了实例配置文件文档。添加了使用设置实例配置文件的 step-by-step过程 AWS CLI 和改进的查找与 AWS PCS 一起使用的实例配置文件的指南。 有关更多信息，请参阅 AWS 并行计算服务的 IAM 实例配置文件。	2025-08-14

日期	更改	文档更新	更新的 API 版本
2025 年 8 月 1 日	新话题：SPANK 插件	添加了 SPANK（适用于节点和作业 Kontrol 的 Slurm 插件架构）插件的文档，您可以使用这些插件在 PCS 集群上启动和执行任务期间扩展和修改 Slurm 的行为。 AWS 有关更多信息，请参阅 使用 SPANK 插 AWS 件在 PCS 上扩展 Slurm 功能。	不适用
2025 年 8 月 1 日	IPv6 网络支持	在创建 AWS PCS 集群时添加了对 IPv6 联网的支持。现在，您可以选择 IPv6 集群的网络类型，并对 VPC 要求、子网配置、安全组设置和集群创建过程进行相应的更新。 有关更多信息，请参阅AWS PCS VPC 和子网要求和注意事项和在 AWS PCS 中创建集群。	2025-08-01

日期	更改	文档更新	更新的 API 版本
2025 年 7 月 3 日	AWS PCS 在欧洲（伦敦）上市	AWS PCS 现已在欧洲（伦敦）（eu-west-2）上市。 CloudFormation 欧洲（伦敦）有模板可供入门 AWS 区域。有关更多信息，请参阅CloudFormation 用于创建示例 AWS PCS 集群和CloudFormation 用于创建示例 AWS PCS 集群的模板。	不适用
2025 年 7 月 1 日	更新了控制台说明	现在，当您在控制台中创建集群和计算节点组时，您可以让 AWS PCS 为您创建基本的实例配置文件和安全组。有关更多信息，请参阅: • 在 AWS PCS 中创建集群 • 在 AWS PCS 中创建计算节点组 • AWS 并行计算服务的 IAM 实例配置文件	不适用
2025 年 6 月 23 日	新的托管策略：AWSPCSComputeNodePolicy	添加了一个新的托管策略，该策略向 AWS PCS 计算节点授予连接到 AWS PCS 集群的权限。有关更多信息，请参阅 AWS 托管策略： AWSPCSComputeNodePolicy。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 6 月 19 日	新主题：任务完成日志	使用任务完成日志记录任务完成时的详细信息，无需支付额外费用。有关更多信息，请参阅 AWS PCS 中的 Job 完成日志。	不适用
2025 年 6 月 18 日	AWS PCS 发布于 AWS GovCloud (US)	AWS PCS 现已在 AWS GovCloud （美国东部）(-us-gov-east 1) 和（美国西部） AWS GovCloud (-us-gov-west 1) 中推出。 CloudFormation 模板可在中开始使用 AWS GovCloud (US) Regions。有关更多信息，请参阅CloudFormation 用于创建示例 AWS PCS 集群和CloudFormation 用于创建示例 AWS PCS 集群的模板。 有关中 AWS PCS 服务终端节点的更多信息 AWS GovCloud (US) Regions，请参阅AWS PCS 的终端节点和服务配额。 有关差异的更多信息 AWS GovCloud (US) Regions，请参阅《AWS GovCloud (US) 用户指南》 AWS GovCloud (US)中的AWS PCS。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 6 月 18 日	更新了 PCS 代理	更新了 AWS PCS 代理 1.2.1-1 的 AMI 主题。有关更多信息，请参阅 要为 PCS 定制 AMIs 版本的软件安装程序 AWS。	不适用
2025 年 5 月 15 日	新功能：会计	Slurm 24.11 或更高版本现在支持 Slurm 记账。有关更多信息，请参阅 PCS 中的 Slurm 会计 AWS。	AWS 软件开发工具包：2025-05-15
2025 年 5 月 15 日	已针对 Slurm 24.11 进行了更新	更新了 Slurm 24.11.5 支持的用户指南。有关更多信息，请参阅下列内容： • PCS 中的 Slurm 版本 AWS • 要为 PCS 定制 AMIs 版本的软件安装程序 AWS • AWS PCS 示例的发行说明 AMIs	不适用
2025 年 5 月 5 日	更新的 Slurm 版本常见问题解答	更新了 Slurm 版本关于 Slurm 版本接近或即将到期 (EOL) 的常见问题 (FAQ)。有关更多信息，请参阅 有关 PCS 中 Slurm 版本的常见问题 AWS。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 4 月 17 日	新主题：如何获取计算节点组详细信息	了解如何获取 AWS PCS 计算节点组的详细信息，例如其 ID、ARN 和 AMI ID。有关更多信息，请参阅 在 AWS PCS 中获取计算节点组的详细信息。	不适用
2025 年 4 月 2 日	更新了 Slurm 安装程序	更新了 Slurm 安装程序 24.05.7-1 的 AMI 主题。有关更多信息，请参阅 要为 PCS 定制 AMIs 版本的软件安装程序 AWS。	不适用
2025 年 3 月 28 日	增加了计算节点组和队列的最大数量的配额	为每个集群的最大计算节点组数和每个集群的最大队列数添加了不可调整的内部配额。有关更多信息，请参阅 内部配额。	不适用
2025 年 3 月 14 日	更改了 CloudFormation 模板中的属性密钥	`Id`现在是`TemplateId` 针对 CloudFormation 模板中的 `CustomLaunchTemplate` 属性的。有关更多信息，请参阅AWS PCS CloudFormation 模板的一部分中的资源。	不适用

日期	更改	文档更新	更新的 API 版本
2025 年 3 月 13 日	添加了 AWS PCS 代理和 Slurm 的版本信息	添加了一个新主题，描述了每个版本的 AWS PCS 代理的更改。有关更多信息，请参阅 AWS PCS 代理版本。 在 Slurm 版本主题中添加了更多信息，该主题描述了对 Slurm 的 AWS PCS 支持的重要支持日期和详细版本说明。有关更多信息，请参阅 PCS 中的 Slurm 版本 AWS。	不适用
2025 年 3 月 7 日	更新了 PCS 代理	更新了 AWS PCS 代理 1.2.0-1 的 AMI 主题。有关更多信息，请参阅 要为 PCS 定制 AMIs 版本的软件安装程序 AWS。	不适用
2025 年 2 月 3 日	添加了有关 AWS CloudFormation 与 AWS PCS 配合使用的主题	在用户指南中添加了一个主题，其中提供了如何 CloudFormation 与 AWS PCS 配合使用的示例。本主题提供了使用示例 CloudFormation 模板创建示例 AWS PCS 集群的过程，并简要介绍了该模板的各个部分。有关更多信息，请参阅 开始使用 CloudFormation 和 AWS PCS。	不适用

日期	更改	文档更新	更新的 API 版本
2024 年 12 月 18 日	已针对 Slurm 24.05 进行了更新	更新了 Slurm 24.05 支持的用户指南。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和AWS PCS 示例的发行说明 AMIs。	不适用
2024 年 12 月 18 日	更新了 Slurm 23.11 示例的 NVIDIA 版本 AMIs	在 Slurm 23.11 示例中更新了 NVIDIA 驱动程序和 CUDA 版本。 AMIs 有关更多信息，请参阅 AWS PCS 示例的发行说明 AMIs。	不适用
2024 年 12 月 17 日	更新了 Slurm 安装程序	更新了 Slurm 安装程序 23.11.10-3 的 AMI 主题。有关更多信息，请参阅 要为 PCS 定制 AMIs 版本的软件安装程序 AWS。	不适用
2024 年 12 月 13 日	更新了 PCS 代理	更新了 AWS PCS 代理 1.1.1-1 的 AMI 主题。有关更多信息，请参阅 要为 PCS 定制 AMIs 版本的软件安装程序 AWS。	不适用
2024 年 12 月 6 日	更新了 PCS 代理和 Slurm 安装程序	更新了 AWS PCS 代理 1.1.0-1 和 Slurm 安装程序 23.11.10-2 的 AMI 主题。有关更多信息，请参阅 要为 PCS 定制 AMIs 版本的软件安装程序 AWS。	不适用
2024 年 12 月 6 日	添加了有关操作系统支持的主题	有关更多信息，请参阅 AWS PCS 中支持的操作系统。	不适用

日期	更改	文档更新	更新的 API 版本
2024 年 11 月 8 日	重新整理的用户指南	我们重新整理了用户指南，将主题置于顶层，将一些主题移到了自己的页面，并将相似的主题组合在一起。	不适用
2024 年 11 月 7 日	更新了 AMI 主题	更新了 Slurm 23.11.10 和 libjwt 17.0 的 AMI 主题。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和第 3 步 — 安装 Slurm。 简化并更正了的发行说明 AMIs。有关更多信息，请参阅 AWS PCS 示例的发行说明 AMIs。	不适用
2024 年 11 月 7 日	添加了有关在 PCS 中使用加密 EBS 卷的新 AWS 主题	添加了一个主题，描述了 AWS PCS 中加密 EBS 卷所需的 KMS 密钥策略。有关更多信息，请参阅 在 PCS 中使用加密 EBS 卷所必需的 KMS 密钥策略 AWS。	不适用
2024 年 10 月 18 日	AWS PCS Agent 1.0.1-1 已发布	更新了 AMI 相关文档以参考 AWS PCS 代理版本 1.0.1-1。有关更多信息，请参阅要为 PCS 定制 AMIs 版本的软件安装程序 AWS和步骤 2-安装 AWS PCS 代理。	不适用

日期	更改	文档更新	更新的 API 版本
2024 年 10 月 10 日	添加了疑难解答章节	添加了故障排除章节，主题涉及重启后自动替换 EC2 实例。有关更多信息，请参阅 对 AWS 并行计算服务中的问题进行故障排除。	不适用
2024 年 9 月 23 日	更新了使用 API 操作和服务管理员的最低权限	现在，`CreateComputeNodeGroup` 和 `UpdateComputeNodeGroup` API 操作需要该`ec2:DescribeInstanceTypeOfferings` 权限。有关更多信息，请参阅 AWS PCS 的最低权限。	不适用
2024 年 9 月 5 日	更新了服务管理员最低权限的 IAM 策略示例	有关更多信息，请参阅 服务管理员的最低权限。	不适用
2024 年 9 月 5 日	在托管策略页面中为 JSON 添加了缺少的权限	这只是对文档的更正。实际的托管策略没有改变。有关更多信息，请参阅 AWS AWS 并行计算服务的托管策略。	不适用
2024 年 8 月 28 日	已添加托管策略页面	有关更多信息，请参阅 AWS AWS 并行计算服务的托管策略。	不适用
2024 年 8 月 28 日	AWS PCS 发布	AWS PCS 用户指南的初始版本。	AWS SDK：2024-08-28

AWS 词汇表

有关最新 AWS 术语，请参阅《AWS 词汇表 参考资料》中的AWS 词汇表。

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。