AWS ParallelCluster 用户指南 (v2)

AWS ParallelCluster

AWS ParallelCluster: AWS ParallelCluster 用户指南 (v2)

Table of Contents

什么是 AWS ParallelCluster

AWS ParallelCluster 是一款受 AWS 支持的开源集群管理工具，可帮助您在 AWS Cloud 中部署和管理高性能计算（HPC）集群。它自动设置所需的计算资源、调度器和共享文件系统。您可以将 AWS ParallelCluster 与 AWS Batch 和 Slurm 调度器一起使用。

借助 AWS ParallelCluster，您可以快速构建和部署概念验证和生产 HPC 计算环境。您也可以在 AWS ParallelCluster 基础之上构建和部署高级别的工作流程，例如可自动完成整个 DNA 测序工作流程的基因组学门户。

定价

使用 AWS ParallelCluster 命令行界面 (CLI) 或 API 时，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 AWS 使用的服务 AWS ParallelCluster。

设置 AWS ParallelCluster

主题

安装 AWS ParallelCluster

AWS ParallelCluster 作为 Python 包分发pip，使用 Python 包管理器进行安装。有关安装 Python 程序包的更多信息，请参阅 Python Packaging User Guide 中的 Installing packages。

安装方式 AWS ParallelCluster:

- 使用虚拟环境（推荐）
- 使用 pip

您可以在发布页面CLI上找到最新版本的版本号 GitHub。

在本指南中，命令示例假定您已安装 Python v3。pip 命令示例使用 pip3 版本。

安装 AWS ParallelCluster 在虚拟环境中（推荐）

我们建议你安装 AWS ParallelCluster 在虚拟环境中。如果您在尝试安装时遇到问题 AWS ParallelCluster 使用pip3，你可以安装 AWS ParallelCluster 在虚拟环境中隔离工具及其依赖关系。或者，您可以使用与通常不同的 Python 版本。

安装 AWS ParallelCluster 在非虚拟环境中使用 pip

的主要分发方法 AWS ParallelCluster 在 Linux、Windows 和 macOS 上是pip，它是 Python 的包管理器。它提供了安装、升级和删除 Python 程序包及其依赖项的方式。

Current AWS ParallelCluster 版本

AWS ParallelCluster 会定期更新。要确定您的版本是否为最新版本，请参阅上的发行版页面 GitHub。

如果你已经有 Python 版本pip并且支持该版本，则可以安装 AWS ParallelCluster 通过使用以下命令。如果您安装了 Python 3+ 版本，我们建议您使用 **pip3** 命令。

```
$ pip3 install "aws-parallelcluster<3.0" --upgrade --user
```

安装后需要执行的步骤

安装之后 AWS ParallelCluster，您可能需要将可执行文件路径添加到PATH变量中。有关特定于平台的说明，请参阅以下主题：

- Linux – 添加 AWS ParallelCluster 可执行到你的命令行路径
- macOS – 添加 AWS ParallelCluster 可执行到你的命令行路径
- Windows – 添加 AWS ParallelCluster 可执行到你的命令行路径

你可以验证一下 AWS ParallelCluster 通过运行正确安装`pcluster version`。

```
$ pcluster version
2.11.9
```

AWS ParallelCluster 会定期更新。要更新到最新版本的 AWS ParallelCluster，再次运行安装命令。有关最新版本的详细信息 AWS ParallelCluster，请参阅 AWS ParallelCluster 发行说明。

```
$ pip3 install "aws-parallelcluster<3.0" --upgrade --user
```

要卸载 AWS ParallelCluster，使用`pip uninstall`。

```
$ pip3 uninstall "aws-parallelcluster<3.0"
```

如果您没有 Python 和 `pip`，则使用适合您的环境的过程。

针对每个环境的详细说明

- 安装 AWS ParallelCluster 在虚拟环境中（推荐）
- 安装 AWS ParallelCluster 在 Linux

- 安装 AWS ParallelCluster 在 macOS 上
- 安装 AWS ParallelCluster 在 Windows 上

安装 AWS ParallelCluster 在虚拟环境中（推荐）

我们建议你安装 AWS ParallelCluster 在虚拟环境中，以避免需求版本与其他pip软件包发生冲突。

先决条件

- 验证是否已安装 pip 和 Python。我们建议使用 pip3 和 Python 3 版本 3.8。如果您正在使用 Python 2， 使用的是pip 而不是pip3 ，使用的是 virtualenv 而不是 venv。

要安装 AWS ParallelCluster 在虚拟环境中

1. 如果未安装 virtualenv，请使用 pip3 安装 virtualenv。如果 python3 -m virtualenv help 显示帮助信息，请转到步骤 2。

   Linux, macOS, or Unix

   ```
   $ python3 -m pip install --upgrade pip
   $ python3 -m pip install --user --upgrade virtualenv
   ```

   运行 exit 以离开当前终端窗口并打开一个新的终端窗口以获取对环境的更改。

   Windows

   ```
   C:\>pip3 install --user --upgrade virtualenv
   ```

   运行 exit 以离开当前命令提示符并打开新的命令提示符以获取对环境的更改。
2. 创建虚拟环境并命名它。

   Linux, macOS, or Unix

   ```
   $ python3 -m virtualenv ~/apc-ve
   ```

   或者，您也可以使用 -p 选项指定特定的 Python 版本。

   ```
   $ python3  -m virtualenv -p $(which python3) ~/apc-ve
   ```

Windows

```
C:\>virtualenv %USERPROFILE%\apc-ve
```

3. 激活新虚拟环境。

   Linux, macOS, or Unix

   ```
   $ source ~/apc-ve/bin/activate
   ```

   Windows

   ```
   C:\>%USERPROFILE%\apc-ve\Scripts\activate
   ```

4. 安装 AWS ParallelCluster 进入您的虚拟环境。

   Linux, macOS, or Unix

   ```
   (apc-ve)~$ python3 -m pip install --upgrade "aws-parallelcluster<3.0"
   ```

   Windows

   ```
   (apc-ve) C:\>pip3 install --upgrade "aws-parallelcluster<3.0"
   ```

5. 验证一下 AWS ParallelCluster 已正确安装。

   Linux, macOS, or Unix

   ```
   $ pcluster version
   2.11.9
   ```

   Windows

   ```
   (apc-ve) C:\>pcluster version
   2.11.9
   ```

您可以使用 `deactivate` 命令退出虚拟环境。每当您启动会话时，您必须重新激活该环境。

要升级到最新版本的 AWS ParallelCluster，再次运行安装命令。

Linux, macOS, or Unix

```
(apc-ve)~$ python3 -m pip install --upgrade "aws-parallelcluster<3.0"
```

Windows

```
(apc-ve) C:\>pip3 install --upgrade "aws-parallelcluster<3.0"
```

安装 AWS ParallelCluster 在 Linux

你可以安装 AWS ParallelCluster 通过使用 Python 的包管理器pip，以及它对大多数 Linux 发行版的依赖性。首先，确定是否已安装 Python 和 pip：

1. 要确定您的 Linux 版本是否包含 Python 和 pip，请运行 pip --version。

   ```
   $ pip --version
   ```

   如果您已pip安装，请继续安装 AWS ParallelCluster 带有 pip 话题。否则，请继续执行步骤 2。
2. 要确定是否已安装 Python，请运行 python --version。

   ```
   $ python --version
   ```

   如果你安装了 Python 3 3.6+ 版本或 Python 2 版本 2.7，请继续安装 AWS ParallelCluster 带有 pip 话题。否则，请安装 Python，然后返回到此过程以安装 pip。
3. 使用 Python 打包权威机构提供的脚本来安装 pip。
4. 使用 curl 命令下载安装脚本。

   ```
   $ curl -O https://bootstrap.pypa.io/get-pip.py
   ```
5. 使用 Python 运行脚本以下载并安装最新版本的 pip 和其他必需的支持包。

   ```
   $ python get-pip.py --user
   ```

   或者

   ```
   $ python3 get-pip.py --user
   ```

当您包含 `--user` 开关时，脚本将 pip 安装到路径 `~/.local/bin`。

6. 要验证包含 pip 的文件夹是否是您的 PATH 变量的一部分，请执行以下操作：

   a. 在您的用户文件夹中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 `basename $SHELL`。

   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```

   - Bash – `.bash_profile`、`.profile` 或 `.bash_login`
   - Zsh – `.zshrc`
   - Tcsh – `.tcshrc`、`.cshrc` 或 `.login`

   b. 在配置文件脚本末尾添加与以下示例类似的导出命令。

   ```
   export PATH=~/.local/bin:$PATH
   ```

   此导出命令将路径（在本示例中为 `~/.local/bin`）插入到现有 PATH 变量的前面。

   c. 要使这些更改生效，请将配置文件重新加载到当前会话中。

   ```
   $ source ~/.bash_profile
   ```

7. 验证 pip 是否已正确安装。

   ```
   $ pip3 --version
   pip 21.3.1 from ~/.local/lib/python3.6/site-packages (python 3.6)
   ```

Sections

安装 AWS ParallelCluster 与 **pip**

pip用于安装 AWS ParallelCluster.

```
$ python3 -m pip install "aws-parallelcluster<3.0" --upgrade --user
```

当你使用`--user`交换机时，`pip`会安装 AWS ParallelCluster 到`~/.local/bin`。

验证一下 AWS ParallelCluster 已正确安装。

```
$ pcluster version
2.11.9
```

要升级到最新版本，请重新运行安装命令。

```
$ python3 -m pip install "aws-parallelcluster<3.0" --upgrade --user
```

添加 AWS ParallelCluster 可执行到你的命令行路径

在使用 `pip` 进行安装后，可能需要将 `pcluster` 可执行文件添加到操作系统的 `PATH` 环境变量中。

验证`pip`安装文件夹 AWS ParallelCluster，运行以下命令。

```
$ which pcluster
/home/username/.local/bin/pcluster
```

如果您在安装时省略了`--user`开关 AWS ParallelCluster，则可执行文件可能位于你的 Python 安装`bin`文件夹中。如果您不知道 Python 的安装位置，请运行此命令。

```
$ which python
/usr/local/bin/python
```

请注意，输出可能是符号链接的路径，而不是实际的可执行文件。要查看符号链接所指向的位置，请运行 `ls -al`。

```
$ ls -al $(which python)
/usr/local/bin/python -> ~/.local/Python/3.6/bin/python3.6
```

如果这是您在安装 AWS ParallelCluster的步骤 3 中添加到路径的相同文件夹，则您已完成安装。否则，您必须再次执行步骤 3a – 3c，并将此额外文件夹添加到路径。

在 Linux 上安装 Python

如果你的发行版没有附带 Python，或者附带了更早的版本，请在安装之前安装 Python `pip` 然后 AWS ParallelCluster.

在 Linux 上安装 Python 3

1. 检查是否已安装 Python。

   ```
   $ python3 --version
   ```

   或者

   ```
   $ python --version
   ```

   > **Note**
   >
   > 如果您的 Linux 分发版附带了 Python，则可能需要安装 Python 开发人员程序包。开发者软件包包括编译扩展和安装所需的头文件和库 AWS ParallelCluster。 使用您的软件包管理器来安装开发人员软件包。它通常名为 `python-dev` 或 `python-devel`。

2. 如果尚未安装 Python 2.7 或更高版本，请使用分发版本的程序包管理器来安装 Python 命令和程序包名称会有所不同：

   - 在 Debian 衍生系统（如 Ubuntu）上，请使用 `apt`。

     ```
     $ sudo apt-get install python3
     ```

   - 在 Red Hat 及其衍生系统上，请使用 `yum`。

     ```
     $ sudo yum install python3
     ```

   - 关于SUSE和衍生物，使用`zypper`。

     ```
     $ sudo zypper install python3
     ```

3. 要验证是否已正确安装 Python，请打开命令提示符或 shell，并运行以下命令。

   ```
   $ python3 --version
   ```

```
Python 3.8.11
```

安装 AWS ParallelCluster 在 macOS 上

Sections

先决条件

- Python 3 版本 3.7+ 或 Python 2 版本 2.7

检查您的 Python 安装。

```
$ python --version
```

如果您的计算机上还没有安装 Python，或者您希望安装 Python 的其他版本，请按照安装 AWS ParallelCluster 在 Linux中的过程执行操作。

安装 AWS ParallelCluster 在 macOS 上使用 pip

你也可以pip直接使用来安装 AWS ParallelCluster。 如果没有pip，请按照主要安装主题中的说明进行操作。运行 `pip3 --version` 可查看您的 macOS 版本是否已包含 Python 和 pip3。

```
$ pip3 --version
```

要安装 AWS ParallelCluster 在 macOS 上

1. 从 Python.org 的下载页面下载并安装最新版本的 Python。
2. 下载并运行 Python 打包权威机构提供的 pip3 安装脚本。

   ```
   $ curl -O https://bootstrap.pypa.io/get-pip.py
   $ python3 get-pip.py --user
   ```

3. 使用新安装的软件pip3进行安装 AWS ParallelCluster。 如果您使用的是 Python 版本 3+，我们建议您使用该pip3命令。

```
$ python3 -m pip install "aws-parallelcluster<3.0" --upgrade --user
```

4. 验证一下 AWS ParallelCluster 已正确安装。

```
$ pcluster version
2.11.9
```

如果未找到该程序，请将它添加到命令行路径。

要升级到最新版本，请重新运行安装命令。

```
$ pip3 install "aws-parallelcluster<3.0" --upgrade --user
```

添加 AWS ParallelCluster 可执行到你的命令行路径

在使用 pip 进行安装后，可能需要将 pcluster 程序添加到操作系统的 PATH 环境变量中。程序的位置取决于 Python 的安装位置。

Example AWS ParallelCluster 安装位置——搭载 Python 3.6 的 macOS 和**pip**（用户模式）

```
~/Library/Python/3.6/bin
```

将上面示例中的版本替换为您的 Python 版本。

如果您不知道 Python 的安装位置，请运行 which python。

```
$ which python3
/usr/local/bin/python3
```

输出可能是符号链接的路径，而不是实际程序的路径。运行 ls -al 以查看所指向的路径。

```
$ ls -al /usr/local/bin/python3
lrwxr-xr-x  1 username  admin  36 Mar 12 12:47 /usr/local/bin/python3 -> ../Cellar/
python/3.6.8/bin/python3
```

pip 将程序安装到 Python 应用程序所在的文件夹中。将此文件夹添加到 PATH 变量。

要修改您的 **PATH** 变量（Linux、macOS 或 Unix），请执行以下操作：

1. 在您的用户文件夹中查找 Shell 的配置文件脚本。如果您不能确定所使用的 Shell，请运行 `echo $SHELL`。

   ```
   $ ls -a ~
   .  ..  .bash_logout  .bash_profile  .bashrc  Desktop  Documents  Downloads
   ```

   - Bash – `.bash_profile`、`.profile` 或 `.bash_login`
   - Zsh – `.zshrc`
   - Tcsh – `.tcshrc`、`.cshrc` 或 `.login`
2. 向配置文件脚本中添加导出命令。

   ```
   export PATH=~/.local/bin:$PATH
   ```

   在本示例中，此命令将路径 `~/.local/bin` 添加到当前 `PATH` 变量中。
3. 将配置文件加载到当前会话。

   ```
   $ source ~/.bash_profile
   ```

安装 AWS ParallelCluster 在 Windows 上

你可以安装 AWS ParallelCluster 在 Windows 上使用`pip`，这是 Python 的包管理器。如果您已有 `pip`，请按照主要安装主题中的说明执行操作。

Sections

- 安装 AWS ParallelCluster 使用 Pyth pip on 和 Windows
- 添加 AWS ParallelCluster 可执行到你的命令行路径

安装 AWS ParallelCluster 使用 Pyth **pip** on 和 Windows

Python Software Foundation 为包含 `pip` 的 Windows 提供了安装程序。

安装 Python 3.6 和 **pip** (Windows)

1. 从 Python.org 的下载页面下载 Python Windows x86-64 安装程序。

2. 运行安装程序。
3. 选择 “将 Python 3 添加到” PATH。
4. 选择 Install Now。

安装程序在您的用户文件夹中安装 Python 并将其程序文件夹添加到您的用户路径。

要安装 AWS ParallelCluster 用**pip3**（Windows）

如果您使用的是 Python 3+ 版本，我们建议您使用 pip3 命令。

1. 从开始菜单打开命令提示符。
2. 使用以下命令验证 Python 和 pip 是否已正确安装。

```
C:\>py --version
Python 3.8.11
C:\>pip3 --version
pip 21.3.1 from c:\python38\lib\site-packages\pip (python 3.8)
```

3. 安装 AWS ParallelCluster 使用pip。

```
C:\>pip3 install "aws-parallelcluster<3.0"
```

4. 验证一下 AWS ParallelCluster 已正确安装。

```
C:\>pcluster version
2.11.9
```

要升级到最新版本，请重新运行安装命令。

```
C:\>pip3 install --user --upgrade "aws-parallelcluster<3.0"
```

添加 AWS ParallelCluster 可执行到你的命令行路径

安装后 AWS ParallelCluster 使用pip，将pcluster程序添加到操作系统的PATH环境变量中。

您可以通过运行以下命令找到安装 pcluster 程序的位置。

```
C:\>where pcluster
```

```
C:\Python38\Scripts\pcluster.exe
```

如果该命令未返回任何结果，则必须手动添加路径。使用命令行或 Windows 资源管理器发现它在计算机上的安装位置。典型路径包括：

- Python 3 和 **pip3** – `C:\Python38\Scripts\`
- Python 3 和 **pip3** --user 选项 – `%APPDATA%\Python\Python38\Scripts`

> **Note**
>
> 包含版本号的文件夹名称可能有所不同。上述示例说明了 Python38。根据需要替换为您使用的版本号。

修改你的PATH变量 (Windows)

1. 按 Windows 键并输入 **environment variables**。
2. 选择 Edit environment variables for your account（编辑您账户的环境变量）。
3. 选择 PATH，然后选择 “编辑”。
4. 将路径添加到 Variable value (变量值) 字段。例如：***C:\new\path***
5. 选择 OK（确定）两次以应用新设置。
6. 关闭任何运行的命令提示符并重新打开命令提示符窗口。

配置 AWS ParallelCluster

安装之后 AWS ParallelCluster，完成以下配置步骤。

确认你的 AWS 账户的角色包含运行所需的权限`pcluster`CLI。有关更多信息，请参阅 AWS ParallelCluster 实例和用户策略示例。

设置你的 AWS 证书。有关更多信息，请参阅配置 AWS CLI中的 AWS CLI 用户指南。

```
$ aws configure
  AWS Access Key ID [None]: AKIAIOSFODNN7EXAMPLE
AWS Secret Access Key [None]: wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY
Default AWS ## name [us-east-1]: us-east-1
```

```
Default output format [None]:
```

这些区域有： AWS 区域 启动集群的位置必须至少有一个 Amazon EC2 密钥对。有关更多信息，请参阅《亚马逊EC2用户指南》中的亚马逊EC2密钥对。

```
$ pcluster configure
```

该配置向导会提示您输入所需的所有信息以创建集群。使用时序列的细节会有所不同 AWS Batch 作为调度器与使用相比 Slurm。 有关集群配置的更多信息，请参阅配置。

> **Note**
>
> 从版本 2.11.5 开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。你可以继续在 2.11.4 及之前的版本中使用它们，但它们没有资格获得 future 的更新或疑难解答支持 AWS 服务和 AWS Support 团队。

Slurm

从有效列表中 AWS 区域 标识符，选择 AWS 区域 你想让你的集群运行在哪里。

> **Note**
>
> 的名单 AWS 区域 显示的基于您账户的分区，仅包括 AWS 区域 已为您的账户启用。有关启用的更多信息 AWS 区域 对于您的账户，请参阅管理 AWS 区域中的 AWS 一般参考。显示的示例来自 AWS 全局分区。如果您的账户位于 AWS GovCloud (US) 分区，仅限 AWS 区域 在该分区中列出（`gov-us-east-1`和`gov-us-west-1`）。同样，如果您的账户位于 AWS 仅显示了`cn-north-1`和的`cn-northwest-1`中国分区。如需查看完整列表 AWS 区域 由 AWS ParallelCluster，请参阅 支持的区域。

```
Allowed values for the AWS ## ID:
1. af-south-1
2. ap-east-1
3. ap-northeast-1
4. ap-northeast-2
5. ap-south-1
6. ap-southeast-1
7. ap-southeast-2
```

```
8. ca-central-1
9. eu-central-1
10. eu-north-1
11. eu-south-1
12. eu-west-1
13. eu-west-2
14. eu-west-3
15. me-south-1
16. sa-east-1
17. us-east-1
18. us-east-2
19. us-west-1
20. us-west-2
AWS ## ID [ap-northeast-1]:
```

选择要用于集群的计划程序。

```
Allowed values for Scheduler:
1. slurm
2. awsbatch
Scheduler [slurm]:
```

选择操作系统。

```
Allowed values for Operating System:
1. alinux2
2. centos7
3. ubuntu1804
4. ubuntu2004
Operating System [alinux2]:
```

> **Note**
>
> 中添加`alinux2`了对 Support 的支持 AWS ParallelCluster 版本 2.6.0。

输入计算节点集群的最小和最大大小。这是用实例数来衡量的。

```
Minimum cluster size (instances) [0]:
Maximum cluster size (instances) [10]:
```

输入头节点和计算节点的实例类型。对于实例类型，您的账户实例限制足够大，足以满足您的要求。有关更多信息，请参阅 Amazon EC2 用户指南中的按需实例限制。

```
Master instance type [t2.micro]:
Compute instance type [t2.micro]:
```

密钥对是从选定的 Amazon EC2 注册的密钥对中选择的 AWS 区域.

```
Allowed values for EC2 Key Pair Name:
1. prod-uswest1-key
2. test-uswest1-key
EC2 Key Pair Name [prod-uswest1-key]:
```

完成前面的步骤后，决定是使用现有VPC还是出租 AWS ParallelCluster VPC为你创建一个。如果你没有正确配置VPC， AWS ParallelCluster 可以创建一个新的。它将使用同一公有子网中的头节点和计算节点，或者仅使用公有子网中的头节点，所有节点都在私有子网中。有可能达到你的数量VPCs上限 AWS 区域。 每个的默认限制VPCs为五个 AWS 区域。 有关此限制以及如何申请提高限制的更多信息，请参阅 Amazon VPC 用户指南中的VPC和子网。

如果你让 AWS ParallelCluster 创建VPC，您必须决定是否所有节点都应位于公有子网中。

> **Important**
>
> VPCs由... 创建 AWS ParallelCluster 默认情况下不要启用VPC流日志。VPC使用流日志，您可以捕获有关进出您的网络接口的 IP 流量的信息VPCs。有关更多信息，请参阅 Amazon VPC 用户指南中的VPC流日志。

> **Note**
>
> 如果你愿意`1. Master in a public subnet and compute fleet in a private subnet`， AWS ParallelCluster 创建的NAT网关会产生额外费用，即使您指定了免费套餐资源也是如此。

```
Automate VPC creation? (y/n) [n]: y
Allowed values for Network Configuration:
1. Master in a public subnet and compute fleet in a private subnet
```

```
2. Master and compute fleet in the same public subnet
Network Configuration [Master in a public subnet and compute fleet in a private
 subnet]: 1
Beginning VPC creation. Please do not leave the terminal until the creation is
 finalized
```

如果不创建新的VPC，则必须选择现有的VPC。

如果你选择拥有 AWS ParallelCluster 创建VPC，记下VPC身份证这样你就可以使用 AWS CLI 稍后再将其删除。

```
Automate VPC creation? (y/n) [n]: n
Allowed values for VPC ID:
  #  id                    name                                 number_of_subnets
---  --------------------  -----------------------------------  -------------------
  1  vpc-0b4ad9c4678d3c7ad  ParallelClusterVPC-20200118031893                    2
  2  vpc-0e87c753286f37eef  ParallelClusterVPC-20191118233938                    5
VPC ID [vpc-0b4ad9c4678d3c7ad]: 1
```

选择之后VPC，您需要决定是使用现有子网还是创建新子网。

```
Automate Subnet creation? (y/n) [y]: y
```

```
Creating CloudFormation stack...
Do not leave the terminal until the process has finished
```

AWS Batch

从有效列表中 AWS 区域 标识符，选择 AWS 区域 你想让你的集群运行在哪里。

```
Allowed values for AWS ## ID:
1. ap-northeast-1
2. ap-northeast-2
3. ap-south-1
4. ap-southeast-1
5. ap-southeast-2
6. ca-central-1
7. eu-central-1
8. eu-north-1
9. eu-west-1
```

```
10. eu-west-2
11. eu-west-3
12. sa-east-1
13. us-east-1
14. us-east-2
15. us-west-1
16. us-west-2
AWS ## ID [ap-northeast-1]:
```

选择要用于集群的计划程序。

```
Allowed values for Scheduler:
1. slurm
2. awsbatch
Scheduler [awsbatch]:
```

当选择 awsbatch 作为计划程序时，alinux2 将用作操作系统。

输入计算节点集群的最小和最大大小。这是用来衡量的vCPUs。

```
Minimum cluster size (vcpus) [0]:
Maximum cluster size (vcpus) [10]:
```

输入头节点实例类型。使用 awsbatch 调度器时，计算节点使用的实例类型为 optimal。

```
Master instance type [t2.micro]:
```

Amazon EC2 密钥对是从选定的 Amazon EC2 注册的密钥对中选择的 AWS 区域.

```
Allowed values for EC2 Key Pair Name:
1. prod-uswest1-key
2. test-uswest1-key
EC2 Key Pair Name [prod-uswest1-key]:
```

决定是使用现有VPCs还是出租 AWS ParallelCluster VPCs为你创造。如果你没有正确配置VPC，AWS ParallelCluster 可以创建一个新的。它将使用同一公有子网中的头节点和计算节点，或者仅使用公有子网中的头节点，所有节点都在私有子网中。有可能达到你的数量VPCs上限 AWS 区域。默认数字VPCs为五。有关此限制以及如何申请提高限制的更多信息，请参阅 Amazon VPC 用户指南中的VPC和子网。

> **Important**
>
> VPCs由... 创建 AWS ParallelCluster 默认情况下不要启用VPC流日志。VPC使用流日志，您可以捕获有关进出您的网络接口的 IP 流量的信息VPCs。有关更多信息，请参阅 Amazon VPC 用户指南中的VPC流日志。

如果你让 AWS ParallelCluster 创建VPC，决定是否所有节点都应位于公有子网中。

> **Note**
>
> 如果你愿意`1. Master in a public subnet and compute fleet in a private subnet`，AWS ParallelCluster 创建的NAT网关会产生额外费用，即使您指定了免费套餐资源也是如此。

```
Automate VPC creation? (y/n) [n]: y
Allowed values for Network Configuration:
1. Master in a public subnet and compute fleet in a private subnet
2. Master and compute fleet in the same public subnet
Network Configuration [Master in a public subnet and compute fleet in a private
 subnet]: 1
Beginning VPC creation. Please do not leave the terminal until the creation is
 finalized
```

如果不创建新的VPC，则必须选择现有的VPC。

如果你选择拥有 AWS ParallelCluster 创建VPC，记下VPC身份证这样你就可以使用 AWS CLI 稍后再将其删除。

```
Automate VPC creation? (y/n) [n]: n
Allowed values for VPC ID:
  #  id                    name                                 number_of_subnets
---  --------------------  -----------------------------------  -------------------
  1  vpc-0b4ad9c4678d3c7ad  ParallelClusterVPC-20200118031893                    2
  2  vpc-0e87c753286f37eef  ParallelClusterVPC-20191118233938                    5
VPC ID [vpc-0b4ad9c4678d3c7ad]: 1
```

选择VPC完毕后，决定是使用现有子网还是创建新子网。

```
Automate Subnet creation? (y/n) [y]: y
```

```
Creating CloudFormation stack...
Do not leave the terminal until the process has finished
```

完成上述步骤后，一个简单的集群将启动到VPC。VPC使用支持公有 IP 地址的现有子网。该子网的路由表为 `0.0.0.0/0 => igw-xxxxxx`。请注意以下条件：

- VPC必备`DNS Resolution = yes`和`DNS Hostnames = yes`。
- 还VPC应该有正确的DHCP`domain-name`选项 AWS 区域。 默认DHCP选项集已经指定了所需的AmazonProvidedDNS。 如果指定多个域名服务器，请参阅《Amazon VPC 用户指南》中的DHCP选项集。使用私有子网时，请使用NAT网关或内部代理为计算节点启用 Web 访问。有关更多信息，请参阅 网络配置。

但所有设置都包含有效值时，您可以通过运行创建命令来启动集群：

```
$ pcluster create mycluster
```

集群达到 “CREATE_COMPLETE” 状态后，您可以使用普通的SSH客户端设置连接到该集群。有关连接亚马逊EC2实例的更多信息，请参阅亚马逊EC2用户指南中的EC2用户指南。

要删除该集群，请运行以下命令。

```
$ pcluster delete --region us-east-1 mycluster
```

要删除中的网络资源VPC，您可以删除 CloudFormation 网络堆栈。堆栈名称以” 开头parallelclusternetworking-“并以" YYYYMMDDHHMMSS "格式包含创建时间。您可以使用 list-stacks 命令列出堆栈。

```
$ aws --region us-east-1 cloudformation list-stacks \
   --stack-status-filter "CREATE_COMPLETE" \
   --query "StackSummaries[].StackName" | \
   grep -e "parallelclusternetworking-"
    "parallelclusternetworking-pubpriv-20191029205804"
```

可以使用 delete-stack 命令删除堆栈。

```
$ aws --region us-east-1 cloudformation delete-stack \
   --stack-name parallelclusternetworking-pubpriv-20191029205804
```

为您pcluster configure创建VPC的不是在 CloudFormation 网络堆栈中创建的。你可以在控制台中VPC手动将其删除，也可以使用 AWS CLI.

```
$ aws --region us-east-1 ec2 delete-vpc --vpc-id vpc-0b4ad9c4678d3c7ad
```

最佳实践

最佳实践：主实例类型选择

尽管主节点不执行任何作业，但其功能和大小对集群的整体性能至关重要。

在为主节点选择要使用的实例类型时，您需要评估以下各项：

- 集群大小：主节点编排集群的扩展逻辑，并负责将新节点附加到调度器。如果您需要纵向扩展和缩减由大量节点组成的集群，则需要为主节点提供一些额外的计算容量。
- 共享文件系统：使用共享文件系统在计算节点和主节点之间共享项目时，请考虑主节点是暴露NFS服务器的节点。因此，您需要选择具有足够网络带宽和足够专用 Amazon EBS 带宽的实例类型来处理您的工作流程。

最佳实践：网络性能

有三个提示涵盖了改善网络通信的所有可能性。

- 置放群组：集群置放群组是单个可用区中的实例的逻辑分组。有关置放群组的更多信息，请参阅 Amazon EC2 用户指南中的置放群组。您可以将集群配置为使用您自己的置放群组，`placement_group = your-placement-group-name`也可以让 AWS ParallelCluster 使用"compute"策略创建置放群组`placement_group = DYNAMIC`。有关更多信息，请参阅 placement_group（多队列模式）和 placement_group（单队列模式）。
- 增强联网：考虑选择支持增强联网的实例类型。有关更多信息，请参阅《Amazon EC2 用户指南》中的 Linux 增强联网功能。
- 弹性结构适配器：要支持高水平的可扩展实例间通信，请考虑为您的EFA网络选择网络接口。EFA 的定制操作系统 (OS) 旁路硬件通过按需弹性和灵活性增强了实例间通信 AWS 云。要配置单个 Slurm 要使用的集群队列EFA，设置`enable_efa = true`。有关EFA与一起使用的更多信息 AWS

ParallelCluster，参见Elastic Fabric Adapter和enable_efa。有关更多信息EFA，请参阅 Amazon Linux 实例EC2用户指南中的弹性结构适配器。

- 实例带宽：带宽随实例大小而扩展，请考虑选择更适合您需求的实例类型，请参阅EBS亚马逊EC2用户指南中的亚马逊优化实例和亚马逊EBS卷类型。

最佳实践：预算提醒

要管理 AWS ParallelCluster 资源成本，我们建议您使用 AWS Budgets 创建预算的操作以及为选定对象定义的预算阈值提醒 AWS 资源的费用。有关更多信息，请参阅中的配置预算行动 AWS Budgets 用户指南。您也可以使用 Amazon CloudWatch 创建账单警报。有关更多信息，请参阅创建账单警报以监控您的估算值 AWS 收费。

最佳实践：将集群移至新集群 AWS ParallelCluster 次要版本或补丁版本

目前每个 AWS ParallelCluster 次要版本与其pclusterCLI一起是独立的。要将集群移至新的次要版本或补丁版本，必须使用新版本重新创建集群。CLI

要优化将集群迁移到新次要版本的过程或出于其他原因保存共享存储数据，我们建议您使用以下最佳实践。

- 将个人数据保存在外部卷中，例如 Amazon EFS 和 FSx For Lustre。这样，您可以轻松地将数据从一个集群迁移到另一个集群。
- 使用创建下列类型的共享存储系统 AWS CLI 或者 AWS Management Console:
  - [ebs] 部分
  - [efs] 部分
  - [fsx] 部分

  将它们作为现有文件系统添加到新集群配置中。这样，当您删除集群时，这些文件系统会被保留下来，并且可以附加到新集群。共享存储系统无论是附加到集群还是与集群分离，通常都会产生费用。

  我们建议您使用 Amazon EFS 或 Amazon FSx for Lustre 文件系统，因为它们可以同时连接到多个集群，并且您可以在删除旧集群之前将它们连接到新集群。有关更多信息，请参阅亚马逊用户指南中的挂载亚马逊EFS文件系统和亚马逊 for Lustre Lustre EFS 用户指南中的 “访问 FSx Lustre 文件系统”。FSx
- 使用自定义引导操作来自定义您的实例，而不是使用自定义AMI操作。这优化了创建过程，因为AMI不需要为每个新版本创建新的自定义内容。
- 推荐的序列。

1. 更新集群配置以使用现有文件系统定义。
2. 验证 `pcluster` 版本并在需要时进行更新。
3. 创建并测试新集群。
   - 确保您的数据在新集群中可用。
   - 确保您的应用程序可以在新集群中正常运行。
4. 如果您的新集群经过全面测试并可正常运行，而且您确定不会使用旧集群，请将其删除。

从移动 CfnCluster 到 AWS ParallelCluster

AWS ParallelCluster 是的增强版 CfnCluster。

如果您目前正在使用 CfnCluster，我们建议您使用 AWS ParallelCluster 而是用它创建新的集群。尽管你可以继续使用 CfnCluster，但它不再开发中，也不会添加任何新的特性或功能。

和之间的 CfnCluster 主要区别 AWS ParallelCluster 将在以下各节中介绍。

AWS ParallelCluster CLI管理一组不同的集群

`cfncluster`CLI无法使用管理使用创建的集群`pcluster`CLI。以下命令不适用于由创建的集群 CfnCluster：

```
pcluster list
pcluster update cluster_name
pcluster start cluster_name
pcluster status cluster_name
```

要管理使用创建的集群 CfnCluster，必须使用`cfncluster`CLI。

如果您需要一个 CfnCluster 包来管理您的旧集群，我们建议您从 Python 虚拟环境中安装和使用它。

AWS ParallelCluster 并 CfnCluster 使用不同的IAM自定义策略

以前用于创建 CfnCluster 集群的自定义IAM策略不能用于 AWS ParallelCluster。 如果您需要自定义策略 AWS ParallelCluster，则必须创建新的。请参阅 AWS ParallelCluster 指南。

AWS ParallelCluster 并 CfnCluster 使用不同的配置文件

这些区域有： AWS ParallelCluster 配置文件位于该`~/.parallelcluster`文件夹中。 CfnCluster配置文件位于该`~/.cfncluster`文件夹中。

如果你想将现有的 CfnCluster 配置文件与 AWS ParallelCluster，则必须完成以下操作：

1. 将配置文件从 `~/.cfncluster/config` 移动到 `~/.parallelcluster/config`。
2. 如果使用 `extra_json` 配置参数，请按如下所示进行更改。

CfnCluster 设置：

```
extra_json = { "cfncluster" : { } }
```

AWS ParallelCluster 设置：

```
extra_json = { "cluster" : { } }
```

在 AWS ParallelCluster，默认情况下，神经节处于禁用状态

In AWS ParallelCluster，默认情况下，神经节处于禁用状态。要启用 ganglia，请完成以下步骤：

1. 按如下所示设置 `extra_json` 参数：

   ```
   extra_json = { "cluster" : { "ganglia_enabled" : "yes" } }
   ```

2. 更改头安全组以允许连接到端口 80。

   必须通过添加新的安全组规则来修改 `parallelcluster-<CLUSTER_NAME>-MasterSecurityGroup-<xxx>` 安全组，以允许从您的公有 IP 到端口 80 的入站连接。有关更多信息，请参阅 Amazon EC2 用户指南中的向安全组添加规则。

支持的区域

AWS ParallelCluster 2.x 版本在以下版本中可用 AWS 区域:

区域名称	区域
美国东部（俄亥俄州）	us-east-2
美国东部（弗吉尼亚州北部）	us-east-1
美国西部（北加利福尼亚）	us-west-1

区域名称	区域
美国西部（俄勒冈州）	us-west-2
非洲（开普敦）	af-south-1
亚太地区（香港）	ap-east-1
亚太地区（孟买）	ap-south-1
亚太地区（首尔）	ap-northeast-2
亚太地区（新加坡）	ap-southeast-1
亚太地区（悉尼）	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
加拿大（中部）	ca-central-1
中国（北京）	cn-north-1
中国（宁夏）	cn-northwest-1
欧洲（法兰克福）	eu-central-1
欧洲地区（爱尔兰）	eu-west-1
欧洲地区（伦敦）	eu-west-2
欧洲地区（米兰）	eu-south-1
欧洲地区（巴黎）	eu-west-3
欧洲地区（斯德哥尔摩）	eu-north-1
中东（巴林）	me-south-1
南美洲（圣保罗）	sa-east-1
AWS GovCloud （美国东部）	us-gov-east-1

区域名称	区域
AWS GovCloud（美国西部）	us-gov-west-1

使用 AWS ParallelCluster

主题

网络配置

AWS ParallelCluster 使用亚马逊 Virtual Private Cloud (VPC) 进行联网。VPC提供了一个灵活且可配置的网络平台，您可以在其中部署集群。

VPC必须有`DNS Resolution = yes`，`DNS Hostnames = yes`以及具有该地区正确域名的DHCP选项。默认的DHCP选项集已经指定了所需的选项AmazonProvidedDNS。如果指定多个域名服务器，请参阅《Amazon VPC 用户指南》中的DHCP选项集。

AWS ParallelCluster 支持以下高级配置：

- 适用于头节点和计算节点的一个子网。

- 两个子网，头节点位于一个公有子网中，计算节点位于私有子网中。子网可以是新的子网，也可以是现有子网。

所有这些配置都可以在有或没有公有 IP 地址的情况下运行。 AWS ParallelCluster 也可以部署为使用HTTP代理 AWS 处理所有请求。这些配置的组合会产生许多部署方案。例如，您可以配置一个公有子网，允许所有人通过 Internet 进行访问。或者，您可以使用 AWS Direct Connect 和HTTP代理来配置一个完全私有的网络，用于所有流量。

有关其中一些情形的说明，请参阅以下架构图：

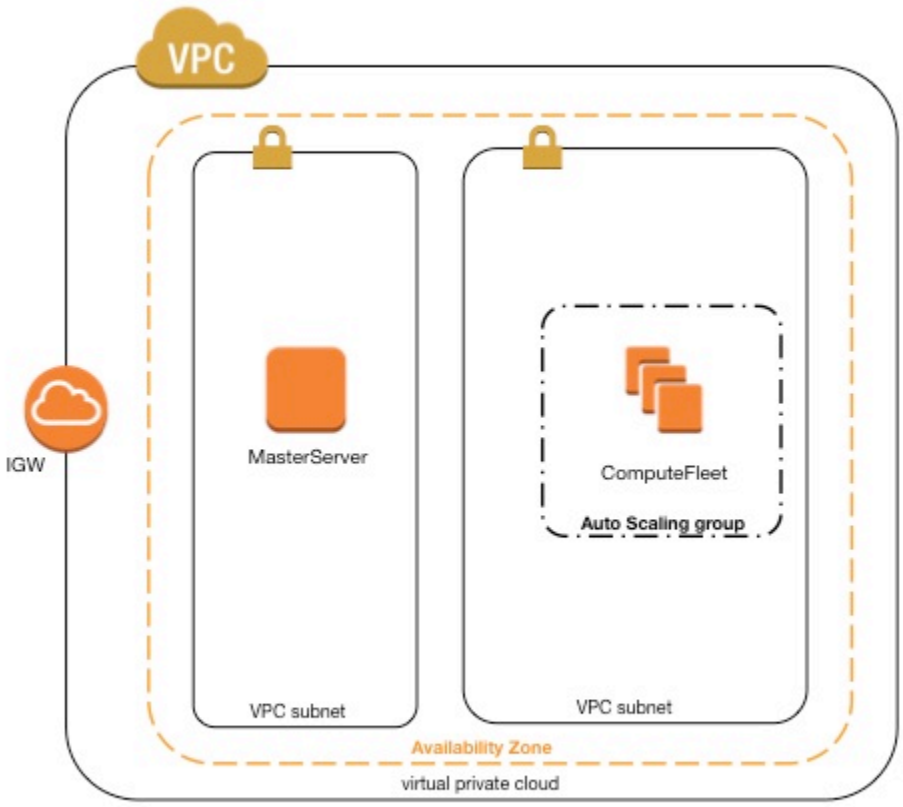
AWS ParallelCluster 在单个公有子网中

此架构的配置需要以下设置：

```
[vpc public]
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-<public>
use_public_ips = true
```

use_public_ips 设置不能设为 false，因为互联网网关要求所有实例都具有全局唯一的 IP 地址。有关更多信息，请参阅 Amazon VPC 用户指南中的启用互联网接入。

AWS ParallelCluster 使用两个子网



为计算实例创建新的私有子网的配置要求使用以下设置：

请注意，所有的值仅作为示例提供。

```
[vpc public-private-new]
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-<public>
compute_subnet_cidr = 10.0.1.0/24
```

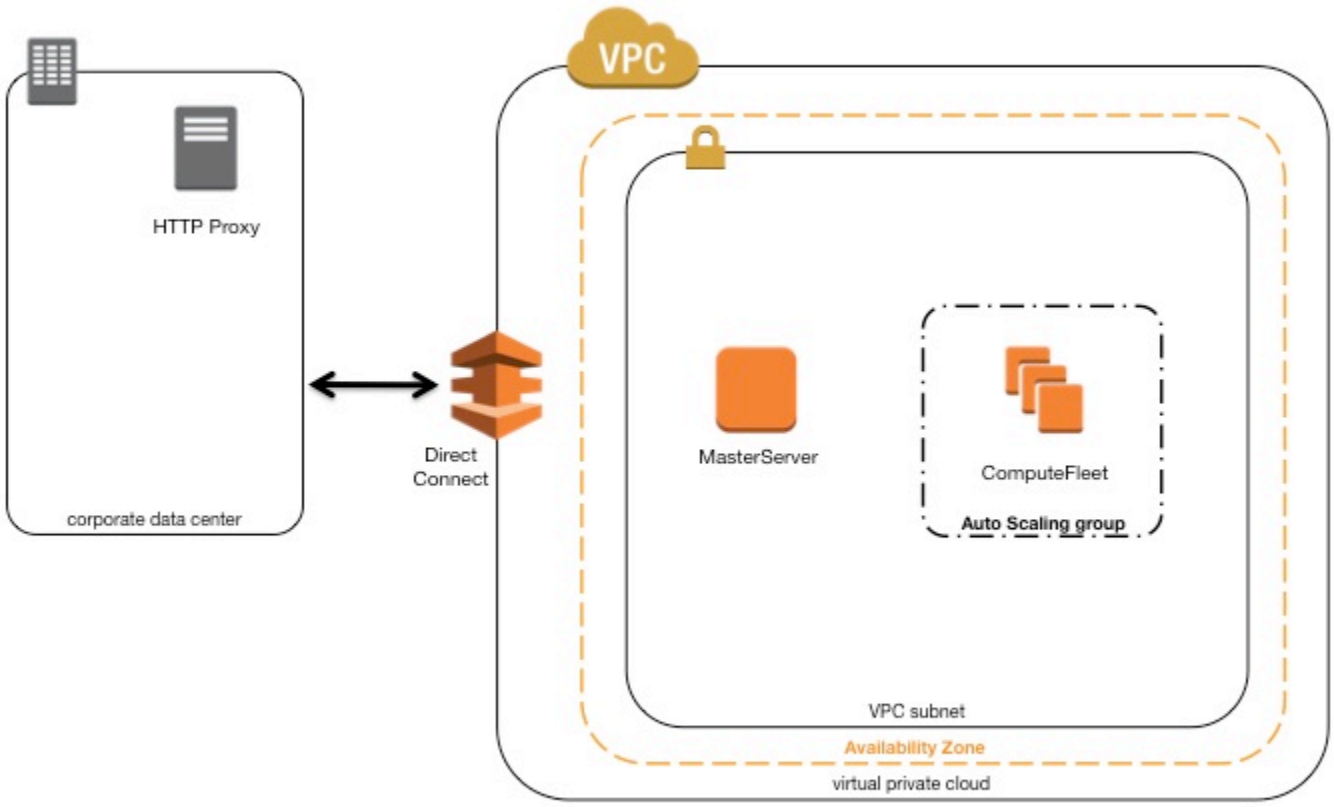
使用现有私有网络的配置要求使用以下设置：

```
[vpc public-private-existing]
vpc_id = vpc-xxxxxx
```

```
master_subnet_id = subnet-<public>
compute_subnet_id = subnet-<private>
```

这两种配置都需要NAT网关或内部代理才能为计算实例启用 Web 访问。

AWS ParallelCluster 在使用连接的单个私有子网中 AWS Direct Connect



此架构的配置需要以下设置：

```
[cluster private-proxy]
proxy_server = http://proxy.corp.net:8080

[vpc private-proxy]
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-<private>
```

```
use_public_ips = false
```

如果设置use_public_ips为false，则VPC必须正确设置才能对所有流量使用代理。头节点和计算节点都需要 Web 访问权限。

AWS ParallelCluster 使用调awsbatch度器

当您使用awsbatch作为调度器类型时， AWS ParallelCluster 会创建一个 AWS Batch 托管计算环境。该 AWS Batch 环境负责管理在中启动的亚马逊弹性容器服务 (AmazonECS) 容器实例compute_subnet。 AWS Batch 为了正常运行，Amazon ECS 容器实例需要访问外部网络才能与亚马逊ECS服务终端节点通信。这会转换为以下情形：

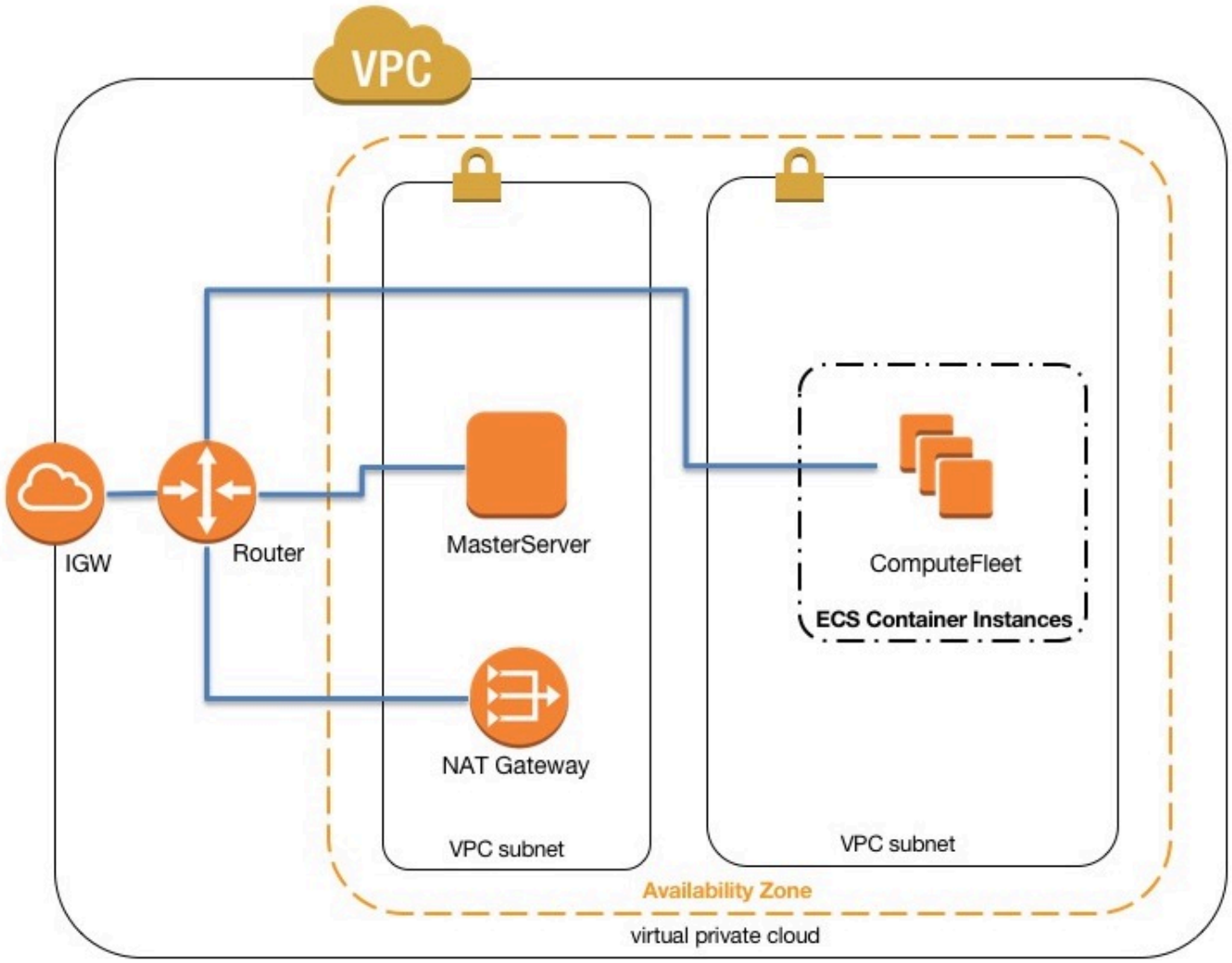
- compute_subnet使用NAT网关访问互联网。（我们建议采用此方法。）
- 在 compute_subnet 中启动的实例具有公有 IP 地址，并可通过互联网网关访问互联网。

此外，如果您对多节点并行作业感兴趣（来自 AWS Batch 文档）：

AWS Batch 多节点并行任务使用 Amazon ECS awsvpc 网络模式，该模式为您的多节点并行任务容器提供了与 Ama EC2 zon 实例相同的网络属性。每个多节点并行作业容器都有自己的弹性网络接口、主私有 IP 地址和内部DNS主机名。网络接口是在与其主机计算资源相同的 Amazon VPC 子网中创建的。适用于计算资源的任何安全组，也适用于该主机计算资源。

使用 Amazon T ECS ask Networking 时，awsvpc网络模式不为使用 Amazon EC2 启动类型的任务提供带有公有 IP 地址的弹性网络接口。要访问互联网，使用 Amazon EC2 启动类型的任务必须在配置为使用网关的私有子NAT网中启动。

必须配置NAT网关才能使集群能够运行多节点 parallel 作业。



有关更多信息，请参阅以下主题：

- AWS Batch 托管计算环境
- AWS Batch 多节点并行作业
- Amazon ECS 任务使用awsvpc网络模式联网

自定义引导操作

AWS ParallelCluster 可以在创建集群时的主引导操作之前（安装前）或之后（安装后）运行任意代码。在大多数情况下，此代码存储在亚马逊简单存储服务 (Amazon S3) 中，并通过连接进行HTTPS访问。此代码将以 root 用户身份运行，可以采用集群操作系统支持的任何脚本语言。代码通常采用 Bash 或 Python 语言。

在启动任何集群部署引导操作之前，会调用预安装操作，例如配置、NAT Amazon Elastic Block Store (AmazonEBS) 或计划程序。某些预安装操作包括修改存储、添加额外的用户和添加程序包。

集群引导过程完成后调用安装后操作。安装后操作充当在实例可视为已完全配置并完成之前执行的最后操作。某些安装后操作包括更改调度器设置、修改存储和修改程序包。

您可以通过在配置期间指定参数，将参数传递到脚本。为此，需要将这些参数包含在双引号中并传递到安装前或安装后操作。

如果安装前或安装后操作失败，则该实例引导也将失败。退出代码为零 (0) 表示成功。任何其他退出代码都表示实例引导失败。

您可以区分正在运行的头节点和计算节点。获取 `/etc/parallelcluster/cfnconfig` 文件并评估对于头节点和计算节点分别具有“`MasterServer`”和“`ComputeFleet`”值的 `cfn_node_type` 环境变量。

```
#!/bin/bash

. "/etc/parallelcluster/cfnconfig"

case "${cfn_node_type}" in
    MasterServer)
        echo "I am the head node" >> /tmp/head.txt
    ;;
    ComputeFleet)
        echo "I am a compute node" >> /tmp/compute.txt
    ;;
    *)
    ;;
esac
```

配置

以下配置设置用于定义安装前与安装后操作和参数。

```
# URL to a preinstall script. This is run before any of the boot_as_* scripts are run
# (no default)
pre_install = https://<bucket-name>.s3.amazonaws.com/my-pre-install-script.sh
# Arguments to be passed to preinstall script
# (no default)
pre_install_args = argument-1 argument-2
# URL to a postinstall script. This is run after any of the boot_as_* scripts are run
```

```
# (no default)
post_install = https://<bucket-name>.s3.amazonaws.com/my-post-install-script.sh
# Arguments to be passed to postinstall script
# (no default)
post_install_args = argument-3 argument-4
```

参数

前两个参数（即 $0 和 $1）是为脚本名称和 URL 预留的。

```
$0 => the script name
$1 => s3 url
$n => args set by pre/post_install_args
```

示例

以下步骤创建一个简单的安装后脚本，此脚本用于在集群中安装 R 程序包。

1. 创建脚本。

```
#!/bin/bash

echo "post-install script has $# arguments"
for arg in "$@"
do
    echo "arg: ${arg}"
done

yum -y install "${@:2}"
```

2. 使用正确的权限将脚本上传到 Amazon S3。如果公共读取权限不适合您，可使用 s3_read_resource 或 s3_read_write_resource 参数来授予访问权限。有关更多信息，请参阅 使用 Amazon S3。

```
$ aws s3 cp --acl public-read /path/to/myscript.sh s3://<bucket-name>/myscript.sh
```

> **Important**
>
> 如果脚本是在 Windows 上编辑的，则在将脚本上传CRLF到 Amazon S3 之前，必须将行尾从更改为 LF。

3. 更新 AWS ParallelCluster 配置以包含新的安装后操作。

```
[cluster default]
...
post_install = https://<bucket-name>.s3.amazonaws.com/myscript.sh
post_install_args = 'R curl wget'
```

如果存储桶没有公共读取权限，请使用s3作为URL协议。

```
[cluster default]
...
post_install = s3://<bucket-name>/myscript.sh
post_install_args = 'R curl wget'
```

4. 启动集群。

```
$ pcluster create mycluster
```

5. 验证输出。

```
$ less /var/log/cfn-init.log
2019-04-11 10:43:54,588 [DEBUG] Command runpostinstall output: post-install script
 has 4 arguments
arg: s3://<bucket-name>/test.sh
arg: R
arg: curl
arg: wget
Loaded plugins: dkms-build-requires, priorities, update-motd, upgrade-helper
Package R-3.4.1-1.52.amzn1.x86_64 already installed and latest version
Package curl-7.61.1-7.91.amzn1.x86_64 already installed and latest version
Package wget-1.18-4.29.amzn1.x86_64 already installed and latest version
Nothing to do
```

使用 Amazon S3

要为集群资源提供访问 Amazon S3 存储桶的权限，请在 AWS ParallelCluster 配置ARNs中的s3_read_resource和s3_read_write_resource参数中指定存储桶。有关使用控制访问权限的更多信息 AWS ParallelCluster，请参阅AWS Identity and Access Management 中的角色 AWS ParallelCluster。

```
# Specify Amazon S3 resource which AWS ParallelCluster nodes will be granted read-only
 access
# (no default)
s3_read_resource = arn:aws:s3:::my_corporate_bucket*
# Specify Amazon S3 resource which AWS ParallelCluster nodes will be granted read-write
 access
# (no default)
s3_read_write_resource = arn:aws:s3:::my_corporate_bucket/*
```

这两个参数都接受其中一个*或一个有效的 Amazon S3 ARN。有关指定 Amazon S3 的信息ARNs，请参阅中的 Amazon S3 ARN 格式AWS 一般参考。

示例

以下示例向您提供对 Amazon S3 存储桶 my_corporate_bucket 中任何对象的读取访问权限。

```
s3_read_resource = arn:aws:s3:::my_corporate_bucket/*
```

以下示例向您提供对存储桶的读取访问权限，但不允许您读取存储桶中的项目。

```
s3_read_resource = arn:aws:s3:::my_corporate_bucket
```

这最后一个示例为您提供了对存储桶以及存储桶中存储项目的读取访问权限。

```
s3_read_resource = arn:aws:s3:::my_corporate_bucket*
```

使用竞价型实例

AWS ParallelCluster 如果集群配置设置为 `cluster_type` = spot，则使用竞价型实例。竞价型实例比按需型实例更具成本效益，但它们可能会中断。中断造成的影响因使用的特定调度器而异。利用竞价型实例中断通知可能会有所帮助，这些通知会在亚马逊EC2必须停止或终止您的竞价型实例之前提供两分钟的警告。有关更多信息，请参阅 Amazon EC2 用户指南中的竞价型实例中断。以下各部分介绍了竞价型实例可能被中断的三种情形。

> **Note**
>
> 使用竞价型实例要求您的账户中存在 AWSServiceRoleForEC2Spot 服务相关角色。要使用在您的账户中创建此角色 AWS CLI，请运行以下命令：

```
aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
```

有关更多信息，请参阅 Amazon EC2 用户指南中的竞价型实例请求的服务相关角色。

情形 1：没有运行作业的竞价型实例被中断

发生这种中断时，如果调度器队列有需要额外实例的待处理任务，或者活动实例的数量少于initial_queue_size设置，则 AWS ParallelCluster 尝试替换实例。如果 AWS ParallelCluster 无法预置新实例，则会定期重复对新实例的请求。

情形 2：运行单节点作业的竞价型实例被中断

此中断的行为取决于正在使用的调度器。

Slurm

作业失败，状态代码为 NODE_FAIL，并且该作业重新排入队列（除非在提交作业时指定了 --no-requeue）。如果节点是静态节点，则会将其替换。如果节点是动态节点，则会终止并重置该节点。有关sbatch更多信息（包括--no-requeue参数），请参见 sbatch在 Slurm 文档中。

> **Note**
>
> 此行为在 2.9.0 AWS ParallelCluster 版本中发生了变化。早期版本在终止作业时会显示状态代码 NODE_FAIL，并从调度器队列中删除该节点。

SGE

> **Note**
>
> 这仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

作业已经终止。如果作业已启用重新运行标志（使用 qsub -r yes 或 qalter -r yes）或队列将 rerun 配置设置为 TRUE，则重新安排作业。计算实例将从调度器队列中删除。此行为来自以下SGE配置参数：

- `reschedule_unknown 00:00:30`
- `ENABLE_FORCED_QDEL_IF_UNKNOWN`
- `ENABLE_RESCHEDULE_KILL=1`

Torque

> **Note**
>
> 这仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

作业将从系统中删除，节点将从调度器中删除。作业不重新运行。如果实例中断时多个作业正在运行，则 Torque 在删除节点时可能会超时。`sqswatcher` 日志文件中可能显示错误。这不会影响缩放逻辑，并且后续重试将执行适当的清理。

情形 3：运行多节点作业的竞价型实例被中断

此中断的行为取决于正在使用的调度器。

Slurm

作业失败，状态代码为 `NODE_FAIL`，并且该作业重新排入队列（除非在提交作业时指定了 `--no-requeue`）。如果节点是静态节点，则会将其替换。如果节点是动态节点，则会终止并重置该节点。运行已终止作业的其他节点可能会被分配给其他待处理作业，或在经过配置的 `scaledown_idletime` 时间后进行缩减。

> **Note**
>
> 此行为在 2.9.0 AWS ParallelCluster 版本中发生了变化。早期版本在终止作业时会显示状态代码 `NODE_FAIL`，并从调度器队列中删除该节点。运行已终止作业的其他节点在经过配置的 `scaledown_idletime` 时间后可能会进行缩减。

SGE

> **Note**
>
> 这仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

作业未终止，并继续在其余节点上运行。计算节点将从调度器队列中移除，但将作为孤立和不可用的节点显示在主机列表中。

发生这种情况时，用户必须删除作业 (`qdel <jobid>`)。节点仍然显示在主机列表 (`qhost`) 中，不过这不会影响 AWS ParallelCluster。要从列表中删除主机，请在替换实例后运行以下命令。

```
sudo -- bash -c 'source /etc/profile.d/sge.sh; qconf -dattr hostgroup
 hostlist <hostname> @allhosts; qconf -de <hostname>'
```

Torque

> **Note**
>
> 这仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

作业将从系统中删除，节点将从调度器中删除。作业不重新运行。如果实例中断时多个作业正在运行，则 Torque 在删除节点时可能会超时。`sqswatcher` 日志文件中可能显示错误。这不会影响缩放逻辑，并且后续重试将执行适当的清理。

有关竞价型实例的更多信息，请参阅 Amazon EC2 用户指南中的竞价型实例。

AWS Identity and Access Management 中的角色 AWS ParallelCluster

AWS ParallelCluster 使用 Amazon 的 AWS Identity and Access Management (IAM) 角色EC2使实例能够访问用于部署和操作集群的 AWS 服务。默认情况下，Amazon 的IAM角色EC2是在创建集群时创建的。这意味着创建集群的用户必须具有适当级别的权限，如以下各节所述。

AWS ParallelCluster 使用多种 AWS 服务来部署和操作集群。请参阅 AWS ParallelCluster中使用的 AWS 服务部分中的完整列表。

您可以在上的AWS ParallelCluster 文档中跟踪示例政策的更改 GitHub。

主题

创建集群的默认设置

当您使用默认设置创建集群时，Amazon EC2 的默认IAM角色将由集群创建。创建集群的用户必须具有适当级别的权限才能创建启动集群所需的所有资源。这包括为 Amazon 创建IAM角色EC2。通常，用户在使用默认设置时必须具有AdministratorAccess托管策略的权限。有关托管策略的信息，请参阅《IAM 用户指南》中的AWS 托管策略。

使用 Amazon 的现有IAM角色 EC2

在创建集群`ec2_iam_role`时，您可以使用现有设置来代替默认设置，但在尝试启动集群之前，必须先定义IAM策略和角色。通常，您可以为 Amazon 选择一个现有IAM角色EC2，以最大限度地减少在用户启动集群时向他们授予的权限。AWS ParallelCluster 实例和用户策略示例包括所需的最低权限 AWS ParallelCluster 及其功能。您必须在中创建策略和角色作为单独的策略，IAM然后将这些角色和策略附加到相应的资源。某些角色策略可能会变得过大并导致配额错误。有关更多信息，请参阅 解决IAM策略大小问题。在策略中，替换 *<REGION>*, *<AWS ACCOUNT ID>*，以及具有适当值的类似字符串。

如果您打算在集群节点的默认设置中添加额外的策略，我们建议您将其他自定义IAM策略与`additional_iam_policies`设置一起传递，而不是使用`ec2_iam_role`设置。

AWS ParallelCluster 实例和用户策略示例

以下示例策略包括资源的亚马逊资源名称 (ARNs)。如果您在 AWS GovCloud (US) 或 AWS 中国分区中工作，则ARNs必须对其进行更改。具体而言，对于分区，必须将其从 “arn: aws” 更改为 “arn:”，对于中国 AWS GovCloud (US) 分区，必须将其从 “arn: aws-aws-us-gov cn” 更改为 “arn: aws-cn”。AWS 有关更多信息，请参阅AWS GovCloud (US) 用户指南中 AWS GovCloud (US) 区域中的 Amazon 资源名称 (ARNs)，以及ARNs中国 AWS 服务入门中的中国 AWS 服务。

这些策略包括当前所需的最低权限 AWS ParallelCluster、其功能和资源。某些角色策略可能会变得过大并导致配额错误。有关更多信息，请参阅 解决IAM策略大小问题。

主题

ParallelClusterInstancePolicy使用 SGE, Slurm，或 Torque

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。您可以在 2.11.4 及之前的版本中继续使用它们，但它们没有资格获得 AWS 服务和支持团队的未来更新或故障排除 AWS 支持。

主题

ParallelClusterInstancePolicy使用 Slurm

以下示例设置了 us `ParallelClusterInstancePolicy` ing Slurm 作为调度器。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeVolumes",
                "ec2:AttachVolume",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeInstanceStatus",
```

```
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInstances",
                "ec2:DescribeRegions",
                "ec2:TerminateInstances",
                "ec2:DescribeLaunchTemplates",
                "ec2:CreateTags"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "EC2"
        },
        {
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:subnet/<COMPUTE SUBNET ID>",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:network-interface/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:instance/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:volume/*",
                "arn:aws:ec2:<REGION>::image/<IMAGE ID>",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:key-pair/<KEY NAME>",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:security-group/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:launch-template/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:placement-group/*"
            ],
            "Effect": "Allow",
            "Sid": "EC2RunInstances"
        },
        {
            "Action": [
                "dynamodb:ListTables"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "DynamoDBList"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:DescribeStackResource",
                "cloudformation:SignalResource"
```

```
            ],
            "Resource": [
                "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/
parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:GetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:DeleteItem",
                "dynamodb:DescribeTable"
            ],
            "Resource": [
                "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "DynamoDBTable"
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "S3GetObj"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "IAMPassRole",
            "Condition": {
                "StringEquals": {
```

```
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::dcv-license.<REGION>/*"
            ],
            "Effect": "Allow",
            "Sid": "DcvLicense"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "GetClusterConfig"
        },
        {
            "Action": [
                "fsx:DescribeFileSystems"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "*"
```

```
            ],
            "Effect": "Allow",
            "Sid": "CWLogs"
        },
        {
            "Action": [
                "route53:ChangeResourceRecordSets"
            ],
            "Resource": [
                "arn:aws:route53:::hostedzone/*"
            ],
            "Effect": "Allow",
            "Sid": "Route53"
        }
    ]
}
```

ParallelClusterInstancePolicy使用 SGE 或者 Torque

以下示例设置了 us ParallelClusterInstancePolicy ing SGE 或者 Torque 作为调度器。

> **Note**
>
> 本政策仅适用于 AWS ParallelCluster 2.11.4 及以下版本。从 2.11.5 版开始，AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeVolumes",
                "ec2:AttachVolume",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInstances",
                "ec2:DescribeRegions",
                "ec2:TerminateInstances",
                "ec2:DescribeLaunchTemplates",
                "ec2:CreateTags"
```

```
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "EC2"
        },
        {
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:subnet/<COMPUTE SUBNET ID>",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:network-interface/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:instance/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:volume/*",
                "arn:aws:ec2:<REGION>::image/<IMAGE ID>",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:key-pair/<KEY NAME>",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:security-group/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:launch-template/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:placement-group/*"
            ],
            "Effect": "Allow",
            "Sid": "EC2RunInstances"
        },
        {
            "Action": [
                "dynamodb:ListTables"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "DynamoDBList"
        },
        {
            "Action": [
                "sqs:SendMessage",
                "sqs:ReceiveMessage",
                "sqs:ChangeMessageVisibility",
                "sqs:DeleteMessage",
                "sqs:GetQueueUrl"
            ],
            "Resource": [
                "arn:aws:sqs:<REGION>:<AWS ACCOUNT ID>:parallelcluster-*"
            ],
```

```
            "Effect": "Allow",
            "Sid": "SQSQueue"
        },
        {
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:TerminateInstanceInAutoScalingGroup",
                "autoscaling:SetDesiredCapacity",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:DescribeTags",
                "autoscaling:SetInstanceHealth"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "Autoscaling"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:DescribeStackResource",
                "cloudformation:SignalResource"
            ],
            "Resource": [
                "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/
parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:GetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:DeleteItem",
                "dynamodb:DescribeTable"
            ],
            "Resource": [
                "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*"
            ],
            "Effect": "Allow",
```

```
            "Sid": "DynamoDBTable"
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "S3GetObj"
        },
        {
            "Action": [
                "sqs:ListQueues"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "SQSList"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "IAMPassRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
```

```
                "arn:aws:s3:::dcv-license.<REGION>/*"
            ],
            "Effect": "Allow",
            "Sid": "DcvLicense"
        },
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "GetClusterConfig"
        },
        {
            "Action": [
                "fsx:DescribeFileSystems"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Action": [
                "logs:CreateLogStream",
                "logs:PutLogEvents"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "CWLogs"
        },
        {
            "Action": [
                "route53:ChangeResourceRecordSets"
            ],
            "Resource": [
                "arn:aws:route53:::hostedzone/*"
            ],
```

```
            "Effect": "Allow",
            "Sid": "Route53"
        }
    ]
}
```

使用 awsbatch 的 ParallelClusterInstancePolicy

以下示例使用 awsbatch 作为调度器设置 ParallelClusterInstancePolicy。您必须包括分配给 AWS Batch AWS CloudFormation 嵌套堆栈中定义的相同策略。BatchUserRoleBatchUserRoleARN以堆栈输出形式提供。在这个例子中，”*<RESOURCES S3 BUCKET>*” 是cluster_resource_bucket设置的值；如果未指定cluster_resource_bucket，那么”*<RESOURCES S3 BUCKET>*” 是 “并行群集-*”。以下示例概述了所需的权限：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "batch:RegisterJobDefinition",
                "logs:GetLogEvents"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "batch:SubmitJob",
                "cloudformation:DescribeStacks",
                "ecs:ListContainerInstances",
                "ecs:DescribeContainerInstances",
                "logs:FilterLogEvents",
                "s3:PutObject",
                "s3:Get*",
                "s3:DeleteObject",
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-
definition/<AWS_BATCH_STACK - JOB_DEFINITION_SERIAL_NAME>:1",
```

```
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-
definition/<AWS_BATCH_STACK - JOB_DEFINITION_MNP_NAME>*",
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-queue/<AWS_BATCH_STACK -
 JOB_QUEUE_NAME>",
                "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/<STACK NAME>/
*",
                "arn:aws:s3:::<RESOURCES S3 BUCKET>/batch/*",
                "arn:aws:iam::<AWS ACCOUNT ID>:role/<AWS_BATCH_STACK - JOB_ROLE>",
                "arn:aws:ecs:<REGION>:<AWS ACCOUNT ID>:cluster/<ECS COMPUTE
 ENVIRONMENT>",
                "arn:aws:ecs:<REGION>:<AWS ACCOUNT ID>:container-instance/*",
                "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:log-group:/aws/batch/job:log-
stream:*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "batch:DescribeJobQueues",
                "batch:TerminateJob",
                "batch:DescribeJobs",
                "batch:CancelJob",
                "batch:DescribeJobDefinitions",
                "batch:ListJobs",
                "batch:DescribeComputeEnvironments"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "ec2:DescribeInstances",
                "ec2:AttachVolume",
                "ec2:DescribeVolumes",
                "ec2:DescribeInstanceAttribute"
```

```
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2"
        },
        {
            "Action": [
                "cloudformation:DescribeStackResource",
                "cloudformation:SignalResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "fsx:DescribeFileSystems"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:TagResource",
                "logs:UntagResource",
                "logs:CreateLogStream"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "CWLogs"
        }
    ]
}
```

`ParallelClusterUserPolicy`使用 Slurm

以下示例使用`ParallelClusterUserPolicy`设置 Slurm 作为调度器。在这个例子中，”*<RESOURCES S3 BUCKET>*” 是`cluster_resource_bucket`设置的值；如果未指定`cluster_resource_bucket`，那么”*<RESOURCES S3 BUCKET>*” 是 “并行群集-*”。

> **Note**
>
> 如果您使用自定义角色 `ec2_iam_role = <role_name>`，则必须将IAM资源更改为包含以下角色的名称：
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"`
> 更改后：
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/<role_name>"`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeKeyPairs",
                "ec2:DescribeRegions",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribePlacementGroups",
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeSnapshots",
                "ec2:DescribeVolumes",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeAddresses",
                "ec2:CreateTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAvailabilityZones"
            ],
            "Resource": "*",
            "Effect": "Allow",
```

```
            "Sid": "EC2Describe"
        },
        {
            "Action": [
                "ec2:CreateVpc",
                "ec2:ModifyVpcAttribute",
                "ec2:DescribeNatGateways",
                "ec2:CreateNatGateway",
                "ec2:DescribeInternetGateways",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:DescribeRouteTables",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:AssociateRouteTable",
                "ec2:CreateSubnet",
                "ec2:ModifySubnetAttribute"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "NetworkingEasyConfig"
        },
        {
            "Action": [
                "ec2:CreateVolume",
                "ec2:RunInstances",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateNetworkInterface",
                "ec2:CreateSecurityGroup",
                "ec2:ModifyVolumeAttribute",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteVolume",
                "ec2:TerminateInstances",
                "ec2:DeleteSecurityGroup",
                "ec2:DisassociateAddress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:ReleaseAddress",
                "ec2:CreatePlacementGroup",
```

```
                "ec2:DeletePlacementGroup"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2Modify"
        },
        {
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate",
                "ec2:DeleteLaunchTemplate",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "ScalingModify"
        },
        {
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:ListTagsOfResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "DynamoDBDescribe"
        },
        {
            "Action": [
                "dynamodb:CreateTable",
                "dynamodb:DeleteTable",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:TagResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "DynamoDBModify"
        },
        {
            "Action": [
```

```
                "route53:ChangeResourceRecordSets",
                "route53:ChangeTagsForResource",
                "route53:CreateHostedZone",
                "route53:DeleteHostedZone",
                "route53:GetChange",
                "route53:GetHostedZone",
                "route53:ListResourceRecordSets",
                "route53:ListQueryLoggingConfigs"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "Route53HostedZones"
        },
        {
            "Action": [
                "cloudformation:DescribeStackEvents",
                "cloudformation:DescribeStackResource",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks",
                "cloudformation:ListStacks",
                "cloudformation:GetTemplate"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudFormationDescribe"
        },
        {
            "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:UpdateStack"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Sid": "CloudFormationModify"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow",
```

```
            "Sid": "S3ResourcesBucket"
        },
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster*"
            ],
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow",
            "Sid": "S3Delete"
        },
        {
            "Action": [
                "iam:PassRole",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/<PARALLELCLUSTER EC2 ROLE NAME>",
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "IAMModify"
        },
        {
          "Condition": {
              "StringEquals": {
```

```
                "iam:AWSServiceName": [
                    "fsx.amazonaws.com",
                    "s3.data-source.lustre.fsx.amazonaws.com"
                ]
            }
        },
        "Action": [
            "iam:CreateServiceLinkedRole"
        ],
        "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/aws-service-role/*",
        "Effect": "Allow",
        "Sid": "IAMServiceLinkedRole"
    },
    {
        "Action": [
            "iam:CreateInstanceProfile",
            "iam:DeleteInstanceProfile"
        ],
        "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/*",
        "Effect": "Allow",
        "Sid": "IAMCreateInstanceProfile"
    },
    {
        "Action": [
            "iam:AddRoleToInstanceProfile",
            "iam:RemoveRoleFromInstanceProfile",
            "iam:GetRolePolicy",
            "iam:GetPolicy",
            "iam:AttachRolePolicy",
            "iam:DetachRolePolicy",
            "iam:PutRolePolicy",
            "iam:DeleteRolePolicy"
        ],
        "Resource": "*",
        "Effect": "Allow",
        "Sid": "IAMInstanceProfile"
    },
    {
        "Action": [
            "elasticfilesystem:DescribeMountTargets",
            "elasticfilesystem:DescribeMountTargetSecurityGroups",
            "ec2:DescribeNetworkInterfaceAttribute"
        ],
        "Resource": "*",
```

```
            "Effect": "Allow",
            "Sid": "EFSDescribe"
        },
        {
            "Action": [
                "ssm:GetParametersByPath"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SSMDescribe"
        },
        {
            "Action": [
                "fsx:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Action": [
                "elasticfilesystem:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EFS"
        },
        {
            "Action": [
                "logs:DeleteLogGroup",
                "logs:PutRetentionPolicy",
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:TagResource",
                "logs:UntagResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudWatchLogs"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
```

```
                "lambda:GetFunctionConfiguration",
                "lambda:GetFunction",
                "lambda:InvokeFunction",
                "lambda:AddPermission",
                "lambda:RemovePermission",
                "lambda:TagResource",
                "lambda:ListTags",
                "lambda:UntagResource"
            ],
            "Resource": [
                "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:parallelcluster-*",
                "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Sid": "CloudWatch",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:PutDashboard",
              "cloudwatch:ListDashboards",
              "cloudwatch:DeleteDashboards",
              "cloudwatch:GetDashboard"
            ],
            "Resource": "*"
        }
    ]
}
```

ParallelClusterUserPolicy使用 SGE 或者 Torque

> **Note**
>
> 本节仅适用于 AWS ParallelCluster 2.11.4 及以下版本。从 2.11.5 版开始，AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

以下示例使用ParallelClusterUserPolicy设置 SGE 或者 Torque 作为调度器。在这个例子中，”*<RESOURCES S3 BUCKET>*” 是cluster_resource_bucket设置的值；如果未指定cluster_resource_bucket，那么”*<RESOURCES S3 BUCKET>*” 是 “并行群集-*”。

> **Note**
>
> 如果您使用自定义角色 ec2_iam_role = *<role_name>*，则必须将IAM资源更改为包含以下角色的名称：
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"`
> 更改后：
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/<role_name>"`

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeKeyPairs",
                "ec2:DescribeRegions",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribePlacementGroups",
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeSnapshots",
                "ec2:DescribeVolumes",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeAddresses",
                "ec2:CreateTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAvailabilityZones"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2Describe"
        },
        {
            "Action": [
                "ec2:CreateVpc",
                "ec2:ModifyVpcAttribute",
                "ec2:DescribeNatGateways",
```

```
                "ec2:CreateNatGateway",
                "ec2:DescribeInternetGateways",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:DescribeRouteTables",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:AssociateRouteTable",
                "ec2:CreateSubnet",
                "ec2:ModifySubnetAttribute"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "NetworkingEasyConfig"
        },
        {
            "Action": [
                "ec2:CreateVolume",
                "ec2:RunInstances",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateNetworkInterface",
                "ec2:CreateSecurityGroup",
                "ec2:ModifyVolumeAttribute",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteVolume",
                "ec2:TerminateInstances",
                "ec2:DeleteSecurityGroup",
                "ec2:DisassociateAddress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:ReleaseAddress",
                "ec2:CreatePlacementGroup",
                "ec2:DeletePlacementGroup"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2Modify"
        },
        {
```

```
            "Action": [
                "autoscaling:DescribeAutoScalingGroups",
                "autoscaling:DescribeAutoScalingInstances"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "AutoScalingDescribe"
        },
        {
            "Action": [
                "autoscaling:CreateAutoScalingGroup",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate",
                "ec2:DeleteLaunchTemplate",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions",
                "autoscaling:PutNotificationConfiguration",
                "autoscaling:UpdateAutoScalingGroup",
                "autoscaling:PutScalingPolicy",
                "autoscaling:DescribeScalingActivities",
                "autoscaling:DeleteAutoScalingGroup",
                "autoscaling:DeletePolicy",
                "autoscaling:DisableMetricsCollection",
                "autoscaling:EnableMetricsCollection"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "AutoScalingModify"
        },
        {
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:ListTagsOfResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "DynamoDBDescribe"
        },
        {
            "Action": [
                "dynamodb:CreateTable",
                "dynamodb:DeleteTable",
                "dynamodb:GetItem",
```

```
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:TagResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "DynamoDBModify"
        },
        {
            "Action": [
                "sqs:GetQueueAttributes"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SQSDescribe"
        },
        {
            "Action": [
                "sqs:CreateQueue",
                "sqs:SetQueueAttributes",
                "sqs:DeleteQueue",
                "sqs:TagQueue"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SQSModify"
        },
        {
            "Action": [
                "sns:ListTopics",
                "sns:GetTopicAttributes"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SNSDescribe"
        },
        {
            "Action": [
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Unsubscribe",
                "sns:DeleteTopic"
            ],
            "Resource": "*",
```

```
            "Effect": "Allow",
            "Sid": "SNSModify"
        },
        {
            "Action": [
                "cloudformation:DescribeStackEvents",
                "cloudformation:DescribeStackResource",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks",
                "cloudformation:ListStacks",
                "cloudformation:GetTemplate"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudFormationDescribe"
        },
        {
            "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:UpdateStack"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Sid": "CloudFormationModify"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow",
            "Sid": "S3ResourcesBucket"
        },
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster*"
            ],
```

```
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow",
            "Sid": "S3Delete"
        },
        {
            "Action": [
                "iam:PassRole",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/<PARALLELCLUSTER EC2 ROLE NAME>",
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "IAMModify"
        },
        {
          "Condition": {
              "StringEquals": {
                  "iam:AWSServiceName": [
                      "fsx.amazonaws.com",
                      "s3.data-source.lustre.fsx.amazonaws.com"
                  ]
              }
          },
          "Action": [
              "iam:CreateServiceLinkedRole"
          ],
          "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/aws-service-role/*",
```

```
            "Effect": "Allow",
            "Sid": "IAMServiceLinkedRole"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/*",
            "Effect": "Allow",
            "Sid": "IAMCreateInstanceProfile"
        },
        {
            "Action": [
                "iam:AddRoleToInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:GetRolePolicy",
                "iam:GetPolicy",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "iam:PutRolePolicy",
                "iam:DeleteRolePolicy"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "IAMInstanceProfile"
        },
        {
            "Action": [
                "elasticfilesystem:DescribeMountTargets",
                "elasticfilesystem:DescribeMountTargetSecurityGroups",
                "ec2:DescribeNetworkInterfaceAttribute"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EFSDescribe"
        },
        {
            "Action": [
                "ssm:GetParametersByPath"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SSMDescribe"
```

```
        },
        {
            "Action": [
                "fsx:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Action": [
                "elasticfilesystem:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EFS"
        },
        {
            "Action": [
                "logs:DeleteLogGroup",
                "logs:PutRetentionPolicy",
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:TagResource",
                "logs:UntagResource"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudWatchLogs"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
                "lambda:GetFunctionConfiguration",
                "lambda:GetFunction",
                "lambda:InvokeFunction",
                "lambda:AddPermission",
                "lambda:RemovePermission",
                "lambda:TagResource",
                "lambda:ListTags",
                "lambda:UntagResource"
            ],
            "Resource": [
```

```
                "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:parallelcluster-*",
                "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Sid": "CloudWatch",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:PutDashboard",
              "cloudwatch:ListDashboards",
              "cloudwatch:DeleteDashboards",
              "cloudwatch:GetDashboard"
            ],
            "Resource": "*"
        }
    ]
}
```

使用 awsbatch 的 ParallelClusterUserPolicy

以下示例使用 awsbatch 作为调度器设置 ParallelClusterUserPolicy。在这个例子中，”*<RESOURCES S3 BUCKET>*” 是cluster_resource_bucket设置的值；如果未指定cluster_resource_bucket，那么”*<RESOURCES S3 BUCKET>*” 是 “并行群集-*”。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeKeyPairs",
                "ec2:DescribeRegions",
                "ec2:DescribeVpcs",
                "ec2:DescribeSubnets",
                "ec2:DescribeSecurityGroups",
                "ec2:DescribePlacementGroups",
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeInstanceTypes",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeSnapshots",
```

```
                "ec2:DescribeVolumes",
                "ec2:DescribeVpcAttribute",
                "ec2:DescribeAddresses",
                "ec2:CreateTags",
                "ec2:DescribeNetworkInterfaces",
                "ec2:DescribeAvailabilityZones"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2Describe"
        },
        {
            "Action": [
                "ec2:CreateLaunchTemplate",
                "ec2:CreateLaunchTemplateVersion",
                "ec2:ModifyLaunchTemplate",
                "ec2:DeleteLaunchTemplate",
                "ec2:DescribeLaunchTemplates",
                "ec2:DescribeLaunchTemplateVersions"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2LaunchTemplate"
        },
        {
            "Action": [
                "ec2:CreateVpc",
                "ec2:ModifyVpcAttribute",
                "ec2:DescribeNatGateways",
                "ec2:CreateNatGateway",
                "ec2:DescribeInternetGateways",
                "ec2:CreateInternetGateway",
                "ec2:AttachInternetGateway",
                "ec2:DescribeRouteTables",
                "ec2:CreateRoute",
                "ec2:CreateRouteTable",
                "ec2:AssociateRouteTable",
                "ec2:CreateSubnet",
                "ec2:ModifySubnetAttribute"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "NetworkingEasyConfig"
        },
```

```
        {
            "Action": [
                "ec2:CreateVolume",
                "ec2:RunInstances",
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateNetworkInterface",
                "ec2:CreateSecurityGroup",
                "ec2:ModifyVolumeAttribute",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:DeleteNetworkInterface",
                "ec2:DeleteVolume",
                "ec2:TerminateInstances",
                "ec2:DeleteSecurityGroup",
                "ec2:DisassociateAddress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:ReleaseAddress",
                "ec2:CreatePlacementGroup",
                "ec2:DeletePlacementGroup"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2Modify"
        },
        {
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:CreateTable",
                "dynamodb:DeleteTable",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:TagResource"
            ],
            "Resource": "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/
parallelcluster-*",
            "Effect": "Allow",
            "Sid": "DynamoDB"
        },
        {
```

```
            "Action": [
                "cloudformation:DescribeStackEvents",
                "cloudformation:DescribeStackResource",
                "cloudformation:DescribeStackResources",
                "cloudformation:DescribeStacks",
                "cloudformation:ListStacks",
                "cloudformation:GetTemplate",
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:UpdateStack"
            ],
            "Resource": "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/
parallelcluster-*",
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "route53:ChangeResourceRecordSets",
                "route53:ChangeTagsForResource",
                "route53:CreateHostedZone",
                "route53:DeleteHostedZone",
                "route53:GetChange",
                "route53:GetHostedZone",
                "route53:ListResourceRecordSets"
            ],
            "Resource": "arn:aws:route53:::hostedzone/*",
            "Effect": "Allow",
            "Sid": "Route53HostedZones"
        },
        {
            "Action": [
                "sqs:GetQueueAttributes",
                "sqs:CreateQueue",
                "sqs:SetQueueAttributes",
                "sqs:DeleteQueue",
                "sqs:TagQueue"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SQS"
        },
        {
            "Action": [
```

```
                "sqs:SendMessage",
                "sqs:ReceiveMessage",
                "sqs:ChangeMessageVisibility",
                "sqs:DeleteMessage",
                "sqs:GetQueueUrl"
            ],
            "Resource": "arn:aws:sqs:<REGION>:<AWS ACCOUNT ID>:parallelcluster-*",
            "Effect": "Allow",
            "Sid": "SQSQueue"
        },
        {
            "Action": [
                "sns:ListTopics",
                "sns:GetTopicAttributes",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Unsubscribe",
                "sns:DeleteTopic"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "SNS"
        },
        {
            "Action": [
                "iam:PassRole",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*",
                "arn:aws:iam::<AWS ACCOUNT ID>:role/<PARALLELCLUSTER EC2 ROLE NAME>"
            ],
            "Effect": "Allow",
            "Sid": "IAMRole"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:GetInstanceProfile",
```

```
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/*",
            "Effect": "Allow",
            "Sid": "IAMInstanceProfile"
        },
        {
            "Action": [
                "iam:AddRoleToInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile",
                "iam:GetRolePolicy",
                "iam:PutRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:GetPolicy",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "IAM"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow",
            "Sid": "S3ResourcesBucket"
        },
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
```

```
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::<RESOURCES S3 BUCKET>"
            ],
            "Effect": "Allow",
            "Sid": "S3Delete"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
                "lambda:GetFunction",
                "lambda:GetFunctionConfiguration",
                "lambda:InvokeFunction",
                "lambda:AddPermission",
                "lambda:RemovePermission",
                "lambda:TagResource",
                "lambda:ListTags",
                "lambda:UntagResource"
            ],
            "Resource": [
                "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:parallelcluster-*",
                "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Action": [
                "logs:*"
            ],
            "Resource": "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:*",
            "Effect": "Allow",
            "Sid": "Logs"
        },
        {
            "Action": [
                "codebuild:*"
            ],
            "Resource": "arn:aws:codebuild:<REGION>:<AWS ACCOUNT ID>:project/
parallelcluster-*",
```

```
            "Effect": "Allow",
            "Sid": "CodeBuild"
        },
        {
            "Action": [
                "ecr:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "ECR"
        },
        {
            "Action": [
                "batch:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "Batch"
        },
        {
            "Action": [
                "events:*"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Sid": "AmazonCloudWatchEvents"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:ListContainerInstances"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "ECS"
        },
        {
            "Action": [
                "elasticfilesystem:CreateFileSystem",
                "elasticfilesystem:CreateMountTarget",
                "elasticfilesystem:DeleteFileSystem",
                "elasticfilesystem:DeleteMountTarget",
                "elasticfilesystem:DescribeFileSystems",
                "elasticfilesystem:DescribeMountTargets"
```

```
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EFS"
        },
        {
            "Action": [
                "fsx:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Sid": "CloudWatch",
            "Effect": "Allow",
            "Action": [
              "cloudwatch:PutDashboard",
              "cloudwatch:ListDashboards",
              "cloudwatch:DeleteDashboards",
              "cloudwatch:GetDashboard"
            ],
            "Resource": "*"
        }
    ]
}
```

`ParallelClusterLambdaPolicy`使用 SGE, Slurm，或 Torque

以下示例使用`ParallelClusterLambdaPolicy`设置 SGE, Slurm，或 Torque 作为调度器。

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogStream",
```

```
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:*",
      "Effect": "Allow",
      "Sid": "CloudWatchLogsPolicy"
    },
    {
      "Action": [
        "s3:DeleteBucket",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:ListBucket",
        "s3:ListBucketVersions"
      ],
      "Resource": [
        "arn:aws:s3:::*"
      ],
      "Effect": "Allow",
      "Sid": "S3BucketPolicy"
    },
    {
      "Action": [
        "ec2:DescribeInstances"
      ],
      "Resource": "*",
      "Effect": "Allow",
      "Sid": "DescribeInstances"
    },
    {
      "Action": [
        "ec2:TerminateInstances"
      ],
      "Resource": "*",
      "Effect": "Allow",
      "Sid": "FleetTerminatePolicy"
    },
    {
      "Action": [
        "dynamodb:GetItem",
        "dynamodb:PutItem"
      ],
      "Resource": "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*",
      "Effect": "Allow",
      "Sid": "DynamoDBTable"
```

```
    },
    {
      "Action": [
        "route53:ListResourceRecordSets",
        "route53:ChangeResourceRecordSets"
      ],
      "Resource": [
        "arn:aws:route53:::hostedzone/*"
      ],
      "Effect": "Allow",
      "Sid": "Route53DeletePolicy"
    }
  ]
}
```

使用 **awsbatch** 的 **ParallelClusterLambdaPolicy**

以下示例使用 awsbatch 作为调度器设置 ParallelClusterLambdaPolicy。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:logs:*:*:*",
      "Sid": "CloudWatchLogsPolicy"
    },
    {
      "Action": [
        "ecr:BatchDeleteImage",
        "ecr:ListImages"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Sid": "ECRPolicy"
    },
    {
      "Action": [
        "codebuild:BatchGetBuilds",
```

```
        "codebuild:StartBuild"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Sid": "CodeBuildPolicy"
    },
    {
      "Action": [
        "s3:DeleteBucket",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:ListBucket",
        "s3:ListBucketVersions"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Sid": "S3BucketPolicy"
    }
  ]
}
```

适用于用户的 **ParallelClusterUserPolicy**

以下示例设置适用于不需要创建或更新集群的用户的 ParallelClusterUserPolicy。支持以下命令。

- pcluster dcv
- pcluster instances
- pcluster list
- pcluster ssh
- pcluster start
- pcluster status
- pcluster stop
- pcluster version

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "MinimumModify",
            "Action": [
               "autoscaling:UpdateAutoScalingGroup",
               "batch:UpdateComputeEnvironment",
               "cloudformation:DescribeStackEvents",
               "cloudformation:DescribeStackResources",
               "cloudformation:GetTemplate",
               "dynamodb:GetItem",
               "dynamodb:PutItem"
            ],
            "Effect": "Allow",
            "Resource": [
                "arn:aws:autoscaling:<REGION>:<AWS ACCOUNT
 ID>:autoScalingGroup:*:autoScalingGroupName/parallelcluster-*",
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:compute-environment/*",
                "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/<CLUSTERNAME>/
*",
                "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/<CLUSTERNAME>"
            ]
        },
        {
           "Sid": "Describe",
           "Action": [
                "cloudformation:DescribeStacks",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus"
           ],
           "Effect": "Allow",
           "Resource": "*"
        }
    ]
}
```

支持的调度器 AWS ParallelCluster

AWS ParallelCluster 支持多个调度程序，使用设置进行scheduler设置。

> **Note**
>
> 从 2.11.5 版开始，AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。您可以在 2.11.4 及之前的版本中继续使用它们，但它们没有资格获得 AWS 服务和支持团队的未来更新或故障排除 AWS 支持。

主题

Son of Grid Engine (`sge`)

> **Note**
>
> 从 2.11.5 版开始，AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。您可以在 2.11.4 及之前的版本中继续使用它们，但它们没有资格获得 AWS 服务和支持团队的未来更新或故障排除 AWS 支持。

AWS ParallelCluster 版本 2.11.4 及更早版本使用 Son of Grid Engine 8.1.9。

Slurm Workload Manager (`slurm`)

AWS ParallelCluster 版本 2.11.9 的使用情况 Slurm 20.11.9。有关信息 Slurm，请参阅 https://slurm.schedmd.com/。有关下载，请参阅 https://github.com/SchedMD/slurm/tags。有关源代码，请参阅 https://github.com/SchedMD/slurm。

> **Important**
>
> AWS ParallelCluster 已通过测试 Slurm 配置参数，这些参数是默认提供的。您对这些内容所做的任何更改 Slurm 配置参数的风险由您自己承担。我们仅尽量为其提供支持。

AWS ParallelCluster 版本	支持 Slurm 版本
2.11.7、2.11.8、2.11.9	20.11.9
2.11.4 至 2.11.6	20.11.8
2.11.0 至 2.11.3	20.11.7
2.10.4	20.02.7
2.9.0 至 2.10.3	20.02.4
2.6 至 2.8.1	19.05.5
2.5.0、2.5.1	19.05.3-2
2.3.1 至 2.4.1	18.08.6-2
2.3.1 之前	16.05.3-1

多队列模式

AWS ParallelCluster 版本 2.9.0 引入了多队列模式。如果 `scheduler` 设置为 `slurm` 且定义了 `queue_settings`，则支持多队列模式 此模式允许不同的实例类型在计算节点中共存。包含不同实例类型的计算资源可以根据需要向上或向下扩展。在队列模式下，最多支持五 (5) 个队列，并且每个 `[queue] 部分`最多可以引用三 (3) 个 `[compute_resource] 部分`。这些`[queue]部分`中的每一个都是一个分区 Slurm Workload Manager。 有关更多信息，请参阅Slurm 多队列模式指南和多队列模式教程。

队列中的每个 `[compute_resource] 部分`都必须具有不同的实例类型，并且这些 `[compute_resource]` 中的每一个都进一步分为静态节点和动态节点。每个 `[compute_resource]` 的静态节点的编号为 1 到 `min_count` 的值。每个 `[compute_resource]` 的动态节点的编号为一 (1) 到 (`max_count` - `min_count`)。例如，如果 `min_count` 为 2 且 `max_count` 为 10，则该 `[compute_resource]` 的动态节点的编号为一 (1) 到八 (8)。在任何时候，节点数可能介于零 (0) 和 `[compute_resource]` 中最大动态节点数之间。

启动到计算实例集中的实例是动态分配的。为了帮助管理此分配过程，将为每个节点生成主机名。主机名的格式如下所示：

```
$HOSTNAME=$QUEUE-$STATDYN-$INSTANCE_TYPE-$NODENUM
```

- $QUEUE 是队列的名称。例如，如果该部分开始，[queue *queue-name*]那么 “$QUEUE” 是”*queue-name*”.
- 对于静态节点，$STATDYN 为 st，对于动态节点则为 dy。
- $INSTANCE_TYPE 是 instance_type 设置中 [compute_resource] 的实例类型。
- $NODENUM 是节点的编号。对于静态节点，$NODENUM 介于一 (1) 和 min_count 的值之间，对于动态节点，则介于一 (1) 和 (max_count - min_count) 之间。

主机名和完全限定域名 (FQDN) 都是使用 Amazon Route 53 托管区域创建的。FQDN是$HOSTNAME.$CLUSTERNAME.pcluster，其中$CLUSTERNAME是用于集群的[cluster]部分的名称。

要将您的配置转换为队列模式，请使用 pcluster-config convert 命令。该命令将使用名为 [queue compute] 的单个 [queue] 部分写入更新的配置。该队列包含一个名为 [compute_resource default] 的 [compute_resource] 部分。[queue compute] 和 [compute_resource default] 的设置迁移自指定的 [cluster] 部分。

Slurm 多队列模式指南

AWS ParallelCluster 版本 2.9.0 引入了多队列模式和新的扩展架构 Slurm Workload Manager (Slurm).

以下各节概述了如何使用 Slurm 使用新引入的扩展架构进行集群。

概述

新的扩展架构基于 Slurm的云调度指南和省电插件。有关省电插件的更多信息，请参阅 Slurm 省电指南。在新架构中，可能可供群集使用的资源通常是在中预定义的 Slurm 配置为云节点。

云节点生命周期

在整个生命周期中，云节点会进入以下几种（如果不是全部）状态：POWER_SAVING、POWER_UP (pow_up)、ALLOCATED (alloc) 和 POWER_DOWN (pow_dn)。在某些情况下，云节点可能会进入 OFFLINE 状态。下面的列表详细介绍了云节点生命周期中这些状态的几个方面。

- 处于 POWER_SAVING 状态的节点在 sinfo 中显示 ~ 后缀（例如 idle~）在这种状态下，没有EC2实例支持该节点。但是，Slurm 仍然可以将任务分配给节点。
- 正在过渡到 POWER_UP 状态的节点将在 sinfo 中显示 # 后缀（例如 idle#）。
- 时间 Slurm 将任务分配给处于某种POWER_SAVING状态的节点，该节点会自动转移到某个POWER_UP状态。除此以外，还可以使用 scontrol update nodename=*nodename* state=power_up 命令手动将节点置于 POWER_UP 状态。在此阶段，将调ResumeProgram用，然后启动EC2实例并将其配置为支持POWER_UP节点。

- 当前可供使用的节点在 sinfo 中不显示任何后缀（例如 idle）。节点设置完毕并加入集群后，即可用于运行作业。在此阶段，节点已正确配置并可供使用。一般而言，我们建议中的EC2实例数量与可用节点的数量相同。在大多数情况下，在创建集群后静态节点始终可用。
- 正在过渡到 POWER_DOWN 状态的节点在 sinfo 中显示 % 后缀（例如 idle%）。经过 scaledown_idletime 之后，动态节点会自动进入 POWER_DOWN 状态。相比之下，静态节点在大多数情况下不会关闭。但可以使用 scontrol update nodename=*nodename* state=powering_down 命令手动将节点置于 POWER_DOWN 状态。在此状态下，与节点关联的实例将会终止，节点将重置回 POWER_SAVING 状态以便在经过 scaledown_idletime 之后供将来使用。该scaledown-idletime设置已保存到 Slurm 配置作为SuspendTimeout设置。
- 离线的节点将在 sinfo 中显示 * 后缀（例如 down*）。如果出现以下情况，则节点将离线 Slurm 控制器无法联系节点，或者如果静态节点被禁用并且后备实例已终止。

现在考虑以下 sinfo 示例中所示的节点状态。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa          up   infinite      4  idle~ efa-dy-c5n18xlarge-[1-4]
efa          up   infinite      1   idle efa-st-c5n18xlarge-1
gpu          up   infinite      1  idle% gpu-dy-g38xlarge-1
gpu          up   infinite      9  idle~ gpu-dy-g38xlarge-[2-10]
ondemand     up   infinite      2   mix# ondemand-dy-c52xlarge-[1-2]
ondemand     up   infinite     18  idle~ ondemand-dy-c52xlarge-[3-10],ondemand-dy-
t2xlarge-[1-10]
spot*        up   infinite     13  idle~ spot-dy-c5xlarge-[1-10],spot-dy-t2large-[1-3]
spot*        up   infinite      2   idle spot-st-t2large-[1-2]
```

spot-st-t2large-[1-2] 和 efa-st-c5n18xlarge-1 节点已经设置了支持实例，并且可供使用。ondemand-dy-c52xlarge-[1-2] 节点处于 POWER_UP 状态，应会在几分钟内可用。gpu-dy-g38xlarge-1 节点处于 POWER_DOWN 状态，它将在 scaledown_idletime（默认为 120 秒）之后过渡到 POWER_SAVING 状态。

所有其他节点都处于POWER_SAVING状态，没有EC2实例支持它们。

使用可用节点

可用节点由EC2实例支持。默认情况下，节点名称可用于直接SSH进入实例（例如ssh efa-st-c5n18xlarge-1）。可以使用 scontrol show nodes *nodename* 命令并检查 NodeAddr 字段来检索实例的私有 IP 地址。对于不可用的节点，该NodeAddr字段不应指向正在运行的EC2实例。而是应与节点名称相同。

作业状态和提交

在大多数情况下，提交的作业会立即分配给系统中的节点，或者如果所有节点都已分配，则将其置于待处理状态。

如果为作业分配的节点包括任何处于 POWER_SAVING 状态的节点，则该作业将以 CF 或 CONFIGURING 状态开始。此时，该作业将会等待处于 POWER_SAVING 状态的节点过渡到 POWER_UP 状态并变为可用。

为作业分配的所有节点都可用后，该作业将进入 RUNNING (R) 状态。

默认情况下，所有作业都提交到默认队列（称为中的分区） Slurm)。 这由队列名称后面的后*缀表示。您可以使用 -p 作业提交选项选择队列。

所有节点都配置了以下特征，这些特征可以在作业提交命令中使用：

- 实例类型（例如 c5.xlarge）。
- 节点类型（dynamic 或 static。）

通过使用 scontrol show nodes *nodename* 命令并查看 AvailableFeatures 列表，您可以查看特定节点的所有可用特征。

另一个考虑因素是作业。首先考虑集群的初始状态，可以通过运行 sinfo 命令来查看该状态。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa          up   infinite      4  idle~ efa-dy-c5n18xlarge-[1-4]
efa          up   infinite      1   idle efa-st-c5n18xlarge-1
gpu          up   infinite     10  idle~ gpu-dy-g38xlarge-[1-10]
ondemand     up   infinite     20  idle~ ondemand-dy-c52xlarge-[1-10],ondemand-dy-
t2xlarge-[1-10]
spot*        up   infinite     13  idle~ spot-dy-c5xlarge-[1-10],spot-dy-t2large-[1-3]
spot*        up   infinite      2   idle spot-st-t2large-[1-2]
```

请注意，spot 是默认队列。它由 * 后缀表示。

向默认队列 (spot) 的一个静态节点提交作业。

```
$ sbatch --wrap "sleep 300" -N 1 -C static
```

向 EFA 队列的一个动态节点提交作业。

```
$ sbatch --wrap "sleep 300" -p efa -C dynamic
```

向 ondemand 队列的八 (8) 个 c5.2xlarge 节点和两 (2) 个 t2.xlarge 节点提交作业。

```
$ sbatch --wrap "sleep 300" -p ondemand -N 10 -C "[c5.2xlarge*8&t2.xlarge*2]"
```

将任务提交到gpu队列GPU中的一个节点。

```
$ sbatch --wrap "sleep 300" -p gpu -G 1
```

现在考虑使用 squeue 命令的作业状态。

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                12  ondemand     wrap   ubuntu CF       0:36     10 ondemand-dy-
c52xlarge-[1-8],ondemand-dy-t2xlarge-[1-2]
                13       gpu     wrap   ubuntu CF       0:05      1 gpu-dy-g38xlarge-1
                 7      spot     wrap   ubuntu  R       2:48      1 spot-st-t2large-1
                 8       efa     wrap   ubuntu  R       0:39      1 efa-dy-
c5n18xlarge-1
```

作业 7 和 8 (在 spot 和 efa 队列中) 已经在运行 (R)。作业 12 和 13 仍在配置 (CF)，可能正在等待实例变为可用。

```
# Nodes states corresponds to state of running jobs
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa          up   infinite      3  idle~ efa-dy-c5n18xlarge-[2-4]
efa          up   infinite      1    mix efa-dy-c5n18xlarge-1
efa          up   infinite      1   idle efa-st-c5n18xlarge-1
gpu          up   infinite      1   mix~ gpu-dy-g38xlarge-1
gpu          up   infinite      9  idle~ gpu-dy-g38xlarge-[2-10]
ondemand     up   infinite     10   mix# ondemand-dy-c52xlarge-[1-8],ondemand-dy-
t2xlarge-[1-2]
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[9-10],ondemand-dy-
t2xlarge-[3-10]
spot*        up   infinite     13  idle~ spot-dy-c5xlarge-[1-10],spot-dy-t2large-[1-3]
spot*        up   infinite      1    mix spot-st-t2large-1
```

```
spot*        up   infinite      1   idle spot-st-t2large-2
```

节点状态和特征

在大多数情况下，节点状态完全由 AWS ParallelCluster 根据本主题前面所述的云节点生命周期中的特定流程进行管理。

但是， AWS ParallelCluster 还会替换或终止处于不健康DRAINED状态的节点DOWN和具有不健康后备实例的节点。有关更多信息，请参阅 clustermgtd。

分区状态

AWS ParallelCluster 支持以下分区状态。A Slurm 分区是一个队列 AWS ParallelCluster。

- UP：表示该分区处于活动状态。这是分区的默认状态。在此状态下，该分区中的所有节点都处于活动状态并且可供使用。
- INACTIVE：表示该分区处于非活动状态。在此状态下，将会终止支持非活动分区的节点的所有实例。不会为非活动分区中的节点启动新实例。

pcluster 启动和停止

运行pcluster stop时，所有分区都将置于INACTIVE状态，并且 AWS ParallelCluster 进程会将分区保持在INACTIVE状态。

运行 pcluster start 时，所有分区最初都处于 UP 状态。但是， AWS ParallelCluster 进程不会使分区保持UP状态。您需要手动更改分区状态。所有静态节点将在几分钟后变为可用。请注意，将分区设置为 UP 不会增加任何动态容量。如果 initial_count 大于 max_count，则在分区状态更改为 UP 状态时，可能无法满足 initial_count。

当 pcluster start 和 pcluster stop 正在运行时，您可以通过运行 pcluster status 命令并检查 ComputeFleetStatus 来查看集群的状态。下面列出了可能的状态：

- STOP_REQUESTED：pcluster stop 请求已发送到集群。
- STOPPING：pcluster 进程当前正在停止集群。
- STOPPED：pcluster 进程已完成停止进程，所有分区都处于 INACTIVE 状态，并且所有计算实例都已终止。
- START_REQUESTED：pcluster start 请求已发送到集群。
- STARTING：pcluster 进程当前正在启动集群

- RUNNING：pcluster 进程已完成启动过程，所有分区都处于 UP 状态，静态节点将在几分钟后可用。

手动控制队列

在某些情况下，您可能需要对节点或队列（称为中的分区）进行一些手动控制 Slurm) 在集群中。您可以通过以下常用过程管理集群中的节点。

- 启动处于POWER_SAVING状态的动态节点：运行scontrol update nodename=*nodename* state=power_up命令或提交占位符sleep 1作业，请求一定数量的节点并依赖 Slurm 为所需数量的节点加电。
- 之前关闭动态节点的电源scaledown_idletime：DOWN使用scontrol update nodename=*nodename* state=down命令将动态节点设置为。 AWS ParallelCluster 自动终止并重置已关闭的动态节点。通常，我们不建议直接使用 scontrol update nodename=*nodename* state=power_down 命令将节点设置为 POWER_DOWN。这是因为 AWS ParallelCluster 会自动处理关闭过程，无需手动干预。因此，我们建议您尽可能将节点设置为 DOWN。
- 禁用队列（分区）或停止特定分区中的所有静态节点：使用 scontrol update partition=*queue name* state=inactive 命令将特定队列设置为 INACTIVE。此操作会终止支持该分区中节点的所有实例。
- 启用队列（分区）：使用 scontrol update partition=*queue name* state=up 命令将特定队列设置为 INACTIVE。

扩展行为和调整

下面是正常扩展工作流程的示例：

- 调度器收到需要两个节点的作业。
- 调度器将两个节点转换为 POWER_UP 状态，并使用节点名称（例如 queue1-dy-c5xlarge-[1-2]）调用 ResumeProgram。
- ResumeProgram启动两个EC2实例并分配的私有 IP 地址和主机名queue1-dy-c5xlarge-[1-2]，等待ResumeTimeout（默认时段为 60 分钟（1 小时）），然后再重置节点。
- 实例配置完成并加入集群。作业开始在实例上运行。
- 作业完成。
- 经过配置的 SuspendTime（设置为 scaledown_idletime）后，调度器将实例置于 POWER_SAVING 状态。调度器将 queue1-dy-c5xlarge-[1-2] 置于 POWER_DOWN 状态并使用节点名称调用 SuspendProgram。

- 为两个节点调用 SuspendProgram。节点保持在 POWER_DOWN 状态，例如通过保持 idle% 状态持续 SuspendTimeout（默认时段为 120 秒（2 分钟））。在 clustermgtd 检测到节点正在关闭后，它会终止支持实例。然后，它将 queue1-dy-c5xlarge-[1-2] 配置为空闲状态并重置私有 IP 地址和主机名，使它们能够启动以供将来的作业使用。

现在，如果出现问题，特定节点的某个实例由于某种原因无法启动，则会发生以下情况。

- 调度器收到需要两个节点的作业。
- 调度器将两个云爆发节点置于 POWER_UP 状态并使用节点名称（例如 queue1-dy-c5xlarge-[1-2]）调用 ResumeProgram。
- ResumeProgram仅启动一个 (1) 个EC2实例并进行配置queue1-dy-c5xlarge-1，但未能启动的实例。queue1-dy-c5xlarge-2
- queue1-dy-c5xlarge-1 将不受影响，并将在进入 POWER_UP 状态后上线。
- queue1-dy-c5xlarge-2处于POWER_DOWN状态，任务会自动重新排队，因为 Slurm 检测到节点故障。
- 经过 SuspendTimeout（默认为 120 秒（2 分钟））之后 queue1-dy-c5xlarge-2 变为可用。在此期间，作业将重新排队，并可以开始在另一个节点上运行。
- 上述过程将会重复，直到作业可以在可用节点上运行而不发生故障。

有两个定时参数可以根据需要进行调整。

- ResumeTimeout（默认为 60 分钟（1 小时））：ResumeTimeout控制时间 Slurm 在将节点置于关闭状态之前会等待。
  - 如果您的安装前/安装后过程几乎需要那么长时间的话，延长此时间可能会很有用。
  - 这也是在出现问题时 AWS ParallelCluster 在更换或重置节点之前等待的最长时间。如果在启动或设置过程中发生任何错误，则计算节点会自行终止。接下来，当 AWS ParallelCluster 进程检测到实例已终止时，它还会替换该节点。
- SuspendTimeout（默认为 120 秒（2 分钟））：SuspendTimeout 控制将节点放回系统并准备好再次使用的速率。
  - 越短SuspendTimeout意味着节点的重置速度会更快，而且 Slurm 能够更频繁地尝试启动实例。
  - SuspendTimeout 越长，故障节点的重置就会越慢。同时，Slurm 轮胎使用其他节点。SuspendTimeout如果超过几分钟 Slurm 尝试循环浏览系统中的所有节点。对于大型系统（超过 1,000 个节点）来说，时间更长SuspendTimeout可能有利于减轻 stress on Slurm 经常将失败的作业重新排队。

- 请注意，SuspendTimeout这并不是指 AWS ParallelCluster 等待终止节点的后备实例的时间。power down 节点的支持实例将会立即终止。终止过程通常在几分钟内完成。但在此期间，节点仍处于关闭状态，无法在调度器中使用。

新架构的日志

以下列表包含多队列架构的关键日志。与 Amazon 日志一起使用的 CloudWatch 日志流名称的格式为*{hostname}*.*{instance_id}*.*{logIdentifier}*，其中 *logIdentifier* 遵循日志名称。有关更多信息，请参阅 与 Amazon CloudWatch 日志集成。

- ResumeProgram:

  /var/log/parallelcluster/slurm_resume.log (slurm_resume)
- SuspendProgram:

  /var/log/parallelcluster/slurm_suspend.log (slurm_suspend)
- clustermgtd:

  /var/log/parallelcluster/clustermgtd.log (clustermgtd)
- computemgtd:

  /var/log/parallelcluster/computemgtd.log (computemgtd)
- slurmctld:

  /var/log/slurmctld.log (slurmctld)
- slurmd:

  /var/log/slurmd.log (slurmd)

常见问题以及调试方法：

无法启动、加电或加入集群的节点：

- 动态节点：
  - 检查 ResumeProgram 日志，查看是否对该节点调用过 ResumeProgram。如果不是，请检查slurmctld日志以确定是否 Slurm 曾经尝试ResumeProgram与该节点通话。请注意，ResumeProgram 上不正确的权限可能会导致它静默失败。

  - 如果调用了 ResumeProgram，请查看是否为该节点启动了实例。如果无法启动实例，则应有明确的错误消息，说明实例启动失败的原因。
  - 如果启动了实例，则可能在引导过程中出现了问题。从ResumeProgram日志中找到相应的私有 IP 地址和实例 ID，然后在 Logs 中查看特定实例的相应引导 CloudWatch 日志。
- 静态节点：
  - 检查 clustermgtd 日志，查看是否为该节点启动了实例。如果没有，则应有明确的错误说明实例启动失败的原因。
  - 如果启动了实例，则在引导过程中出现了问题。从clustermgtd日志中找到相应的私有 IP 和实例 ID，然后在 Logs 中查看特定实例的相应引导 CloudWatch 日志。

节点意外替换或终止、节点故障

- 节点意外替换/终止
  - 在大多数情况下，clustermgtd 会处理所有节点维护操作。要检查 clustermgtd 是否替换或终止了节点，请查看 clustermgtd 日志。
  - 如果 clustermgtd 替换或终止了节点，则应显示一条消息，说明该操作的原因。如果原因与调度器有关（例如，节点处于 DOWN 状态），请查看 slurmctld 日志以获取更多详细信息。如果原因相EC2关，请使用工具检查该实例的状态或日志。例如，您可以检查实例是否有预定事件或运行EC2状况检查失败。
  - 如果clustermgtd没有终止该节点，请检查是否computemgtd终止了该节点，或者是否EC2终止了实例以回收竞价型实例。
- 节点故障
  - 在大多数情况下，如果节点出现故障，作业会自动重新排队。在 slurmctld 日志中查看作业或节点失败的原因，并在其中分析具体情况。

替换或终止实例时出现故障、关闭节点时出现故障

- 通常，clustermgtd 会处理所有预期的实例终止操作。在 clustermgtd 日志中查看其无法替换或终止节点的原因。
- 对于 scaledown_idletime 失败的动态节点，请在 SuspendProgram 日志中查看 slurmctld 运行的程序是否以特定节点作为参数。请注意，SuspendProgram 实际上并不执行任何特定的操作，它只是记录被调用时的时间。所有实例终止和NodeAddr重置均由完成clustermgtd。Slurm将节点放到IDLE之后SuspendTimeout。

其它问题

- AWS ParallelCluster 不会做出工作分配或扩大规模的决策。它简单地尝试根据以下方式启动、终止和维护资源 Slurm的说明。

  对于与作业分配、节点分配和扩展决策有关的问题，请查看 `slurmctld` 日志中是否存在错误。

Torque Resource Manager (`torque`)

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。您可以在 2.11.4 及之前的版本中继续使用它们，但它们没有资格获得 AWS 服务和支持团队的未来更新或故障排除 AWS 支持。

AWS ParallelCluster 版本 2.11.4 及更早版本使用 Torque Resource Manager 6.1.2。有关 Torque Resource Manager 6.1.2，请参阅http://docs.adaptivecomputing.com/torque/6-1-2/releaseNotes/torquerelnote.htm。有关文档，请参阅 http://docs.adaptivecomputing.com/torque/6-1-2/adminGuide/torque.htm。有关源代码，请参阅 https://github.com/adaptivecomputing/torque/tree/6.1.2。

AWS ParallelCluster 版本 2.4.0 及更早版本使用 Torque Resource Manager 6.0.2。有关发行说明，请参阅 http://docs.adaptivecomputing.com/torque/6-0-2/releaseNotes/torqueReleaseNotes6.0.2.pdf。有关文档，请参阅 http://docs.adaptivecomputing.com/torque/6-0-2/adminGuide/help.htm。有关源代码，请参阅 https://github.com/adaptivecomputing/torque/tree/6.0.2。

AWS Batch (`awsbatch`)

有关的信息 AWS Batch，请参见AWS Batch。有关文档，请参阅 AWS Batch User Guide。

AWS ParallelCluster CLI的命令 AWS Batch

使用`awsbatch`调度器时，的 AWS ParallelCluster CLI AWS Batch 命令会自动安装在 AWS ParallelCluster 头节点中。CLI使用 AWS Batch API操作并允许以下操作：

- 提交和管理作业。
- 监控作业、队列和主机。
- 镜像传统调度器命令。

> **Important**
>
> AWS ParallelCluster 不支持以下GPU方面的工作 AWS Batch。有关更多信息，请参阅GPU作业。

主题

- awsbsub
- awsbstat
- awsbout
- awsbkill
- awsbqueues
- awsbhosts

awsbsub

向集群的作业队列提交作业。

```
awsbsub [-h] [-jn JOB_NAME] [-c CLUSTER] [-cf] [-w WORKING_DIR]
        [-pw PARENT_WORKING_DIR] [-if INPUT_FILE] [-p VCPUS] [-m MEMORY]
        [-e ENV] [-eb ENV_DENYLIST] [-r RETRY_ATTEMPTS] [-t TIMEOUT]
        [-n NODES] [-a ARRAY_SIZE] [-d DEPENDS_ON]
        [command] [arguments [arguments ...]]
```

> **Important**
>
> AWS ParallelCluster 不支持以下GPU方面的工作 AWS Batch。有关更多信息，请参阅GPU作业。

定位参数

command

提交作业（指定的命令必须在计算实例上可用），或指定要传输的文件名。另请参阅 `--command-file`。

arguments

（可选）指定命令或命令文件的参数。

命名的参数

-jn *JOB_NAME*, --job-name *JOB_NAME*

为作业命名。第一个字符必须是字母或数字。作业名称可以包含字母（大写和小写）、数字、连字符和下划线，长度不超过 128 个字符。

-c *CLUSTER*, --cluster *CLUSTER*

指定要使用的集群。

-cf, --command-file

指示命令是要传输到计算实例的文件。

默认值：False

-w *WORKING_DIR*, --working-dir *WORKING_DIR*

指定要用作作业的工作目录的文件夹。如果未指定工作目录，则在用户的主目录的 job-*<AWS_BATCH_JOB_ID>* 子文件夹中运行作业。您可以使用此参数或 --parent-working-dir 参数。

-pw *PARENT_WORKING_DIR*, --parent-working-dir *PARENT_WORKING_DIR*

指定作业的工作目录的父文件夹。如果未指定父工作目录，则默认为用户的主目录。在父工作目录中创建名为 job-*<AWS_BATCH_JOB_ID>* 的子文件夹。您可以使用此参数或 --working-dir 参数。

-if *INPUT_FILE*, --input-file *INPUT_FILE*

指定要传输到计算实例的文件（在作业的工作目录中）。您可以指定多个输入文件参数。

-p *VCPUS*, --vcpus *VCPUS*

指定要为容器保留的数量。 vCPUs 与一起使用时–nodes，它会标识每个节点 vCPUs 的数量。

默认值：1

-m *MEMORY*, --memory *MEMORY*

指定要为作业提供的内存的硬限制（以 MiB 为单位）。如果您的作业尝试超出此处指定的内存限制，则该作业将被结束。

默认值：128

-e *ENV*, --env *ENV*

指定要导出到作业环境的环境变量名称的逗号分隔的列表。要导出所有环境变量，请指定“all”。请注意，“all”环境变量列表不包含 `-env-blacklist` 参数中列出的环境变量，或以 `PCLUSTER_*` 或 `AWS_*` 前缀开头的环境变量。

-eb *ENV_DENYLIST*, --env-blacklist *ENV_DENYLIST*

指定不会导出到作业环境的环境变量名称的逗号分隔的列表。默认情况下，不会导出 HOME、PWD、USER、PATH、LD_LIBRARY_PATH、TERM 和 TERMCAP。

-r *RETRY_ATTEMPTS*, --retry-attempts *RETRY_ATTEMPTS*

指定要让作业进入 RUNNABLE 状态的次数。可以指定 1 到 10 之间的尝试次数。如果尝试次数大于 1，则作业在失败后将重试，直到它进入 RUNNABLE 状态的次数达到指定值。

默认值：1

-t *TIMEOUT*, --timeout *TIMEOUT*

指定持续时间（以秒为单位）（根据任务尝试`startedAt`的时间戳衡量），如果任务尚未完成，则该持续时间后 AWS Batch 将终止作业。超时值必须至少为 60 秒。

-n *NODES*, --nodes *NODES*

指定要为作业预留的节点数量。为此参数指定一个值，以启用多节点并行提交。

> **Note**
>
> 当 cluster_type 参数设置为 spot 时，不支持多节点并行作业。

-a *ARRAY_SIZE*, --array-size *ARRAY_SIZE*

指示数组的大小。您可以指定 2 到 10000 之间的值。如果您为一个作业指定数组属性，该作业将变为数组作业。

-d *DEPENDS_ON*, --depends-on *DEPENDS_ON*

指定作业的依赖项的分号分隔的列表。一个作业可依赖于最多 20 个作业。您可以指定 SEQUENTIAL 类型依赖项，而不指定数组作业的作业 ID。顺序依赖项允许每个子数组作业按顺序完成，从索引 0 开始。您也可以使用数组作业的作业 ID 指定 N_TO_N 类型依赖项。N_TO_N 依赖

项意味着此作业的每个子索引必须等待每个依赖项的相应子索引完成后才能开始。此参数的语法为 "jobId=*<string>*，类型=*<string>*;...".

awsbstat

显示集群的作业队列中提交的作业。

```
awsbstat [-h] [-c CLUSTER] [-s STATUS] [-e] [-d] [job_ids [job_ids ...]]
```

定位参数

job_ids

指定要在输出中显示的IDs以空格分隔的作业列表。如果作业是作业数组，则显示所有子作业。如果请求单个作业，则将以详细版本显示该作业。

命名的参数

-c *CLUSTER*, --cluster *CLUSTER*

指示要使用的集群。

-s *STATUS*, --status *STATUS*

指定要包含的作业状态的逗号分隔的列表。默认作业状态为“活动”。接受的值为：SUBMITTED、PENDING、RUNNABLE、STARTING、RUNNING、SUCCEEDED、FAILED 和 ALL。

默认值：“SUBMITTED,PENDING,RUNNABLE,STARTING,RUNNING”

-e, --expand-children

展开具有子作业（数组和多节点并行）的作业。

默认值：False

-d, --details

显示作业详细信息。

默认值：False

awsbout

显示给定作业的输出。

```
awsbout [ - h ] [ - c CLUSTER ] [ - hd HEAD ] [ - t TAIL ] [ - s ] [ - sp STREAM_PERIOD
 ] job_id
```

定位参数

job_id

指定作业 ID。

命名的参数

-c *CLUSTER*, --cluster *CLUSTER*

指示要使用的集群。

-hd *HEAD*, --head *HEAD*

获取第一个 *HEAD* 作业输出的行。

-t *TAIL*, --tail *TAIL*

获取作业输出的最后几个 <tail> 行。

-s, --stream

获取作业输出，然后等待生成其他输出。此参数可与 –tail 一起使用，以从作业输出的最新 <tail> 行开始。

默认值：False

-sp *STREAM_PERIOD*, --stream-period *STREAM_PERIOD*

设置流式传输时段。

默认：5

awsbkill

取消或终止集群中提交的作业。

```
awsbkill [ - h ] [ - c CLUSTER ] [ - r REASON ] job_ids [ job_ids ... ]
```

定位参数

job_ids

指定要取消或终止的IDs以空格分隔的作业列表。

命名的参数

-c *CLUSTER*, --cluster *CLUSTER*

指示要使用的集群的名称。

-r *REASON*, --reason *REASON*

指示要附加到作业的消息，并说明取消作业的原因。

默认：”Terminated by the user”

awsbqueues

显示与集群关联的作业队列。

```
awsbqueues [ - h ] [ - c CLUSTER ] [ - d ] [ job_queues [ job_queues ... ]]
```

定位参数

job_queues

指定要显示的队列的空格分隔的列表。如果请求单个队列，则将以详细版本显示该队列。

命名的参数

-c *CLUSTER*, --cluster *CLUSTER*

指定要使用的集群的名称。

-d, --details

指明是否显示队列的详细信息。

默认值：False

awsbhosts

显示属于集群的计算环境的主机。

```
awsbhosts [ - h ] [ - c CLUSTER ] [ - d ] [ instance_ids [ instance_ids ... ]]
```

定位参数

instance_ids

指定以空格分隔的实例IDs列表。如果请求单个实例，则将以详细版本显示该实例。

命名的参数

-c *CLUSTER*, --cluster *CLUSTER*

指定要使用的集群的名称。

-d, --details

指示是否显示主机的详细信息。

默认值：False

AWS ParallelCluster 资源和标记

AWS ParallelCluster 您可以使用创建标签来跟踪和管理您的 AWS ParallelCluster 资源。您可以在群集配置文件的tags部分中定义 AWS CloudFormation 要创建并传播到所有群集资源的标签。您还可以使用 AWS ParallelCluster 自动生成的标签来跟踪和管理您的资源。

创建集群时，该集群及其资源将使用本节中定义的 AWS ParallelCluster 和 AWS 系统标签进行标记。

AWS ParallelCluster 将标签应用于集群实例、卷和资源。要识别集群堆栈，请 AWS CloudFormation 将 AWS 系统标签应用于集群实例。要识别集群EC2启动模板，请EC2将系统标签应用于实例。您可以使用这些标签来查看和管理您的 AWS ParallelCluster 资源。

您无法修改 AWS 系统标签。为了避免对 AWS ParallelCluster 功能造成影响，请勿修改 AWS ParallelCluster 标签。

以下是 AWS ParallelCluster 资源的 AWS 系统标签示例。您不能修改这些标签。

```
"aws:cloudformation:stack-name"="parallelcluster-clustername-
MasterServerSubstack-ABCD1234EFGH"
```

以下是应用于资源的 AWS ParallelCluster 标签示例。请勿修改这些标签。

```
"aws-parallelcluster-node-type"="Master"
```

```
"Name"="Master"
```

```
"Version"="2.11.9"
```

您可以在的EC2部分中查看这些标签 AWS Management Console。

查看标签

1. 浏览EC2控制台，网址为https://console.aws.amazon.com/ec2/。
2. 要查看所有集群标签，请在导航窗格中选择标签。
3. 要按实例查看集群标签，请在导航窗格中选择实例。
4. 选择一个集群实例。
5. 在实例详细信息中选择管理标签选项卡并查看标签。
6. 在实例详细信息中选择存储选项卡。
7. 选择卷 ID。
8. 在卷中，选择该卷。
9. 在卷详细信息中选择标签选项卡并查看标签。

AWS ParallelCluster 头节点实例标签

键	标签值
ClusterName	*clustername*
Name	Master
Application	parallelcluster- *clustername*

键	标签值
aws:ec2launchtemplate:id	*lt-1234567890abcdef0*
aws:ec2launchtemplate:version	*1*
aws-parallelcluster-node-type	Master
aws:cloudformation:stack-name	parallelcluster- *clustername* -MasterServerSubstack- *ABCD1234EFGH*
aws:cloudformation:logical-id	MasterServer
aws:cloudformation:stack-id	arn:aws:cloudformation: *region-id* :*ACCOUNTID* :stack/parallelcluster- *clustername* -MasterServerSubstack- *ABCD1234EFGH* /*1234abcd-12ab-12ab-12ab-1234567890abcdef0*
Version	*2.11.9*

AWS ParallelCluster 头节点根卷标签

标签密钥	标签值
ClusterName	*clustername*
Application	parallelcluster- *clustername*
aws-parallelcluster-node-type	Master

AWS ParallelCluster 计算节点实例标签

键	标签值
ClusterName	*clustername*

键	标签值
aws-parallelcluster-node-type	Compute
aws:ec2launchtemplate:id	*lt-1234567890abcdef0*
aws:ec2launchtemplate:version	*1*
QueueName	*queue-name*
Version	*2.11.9*

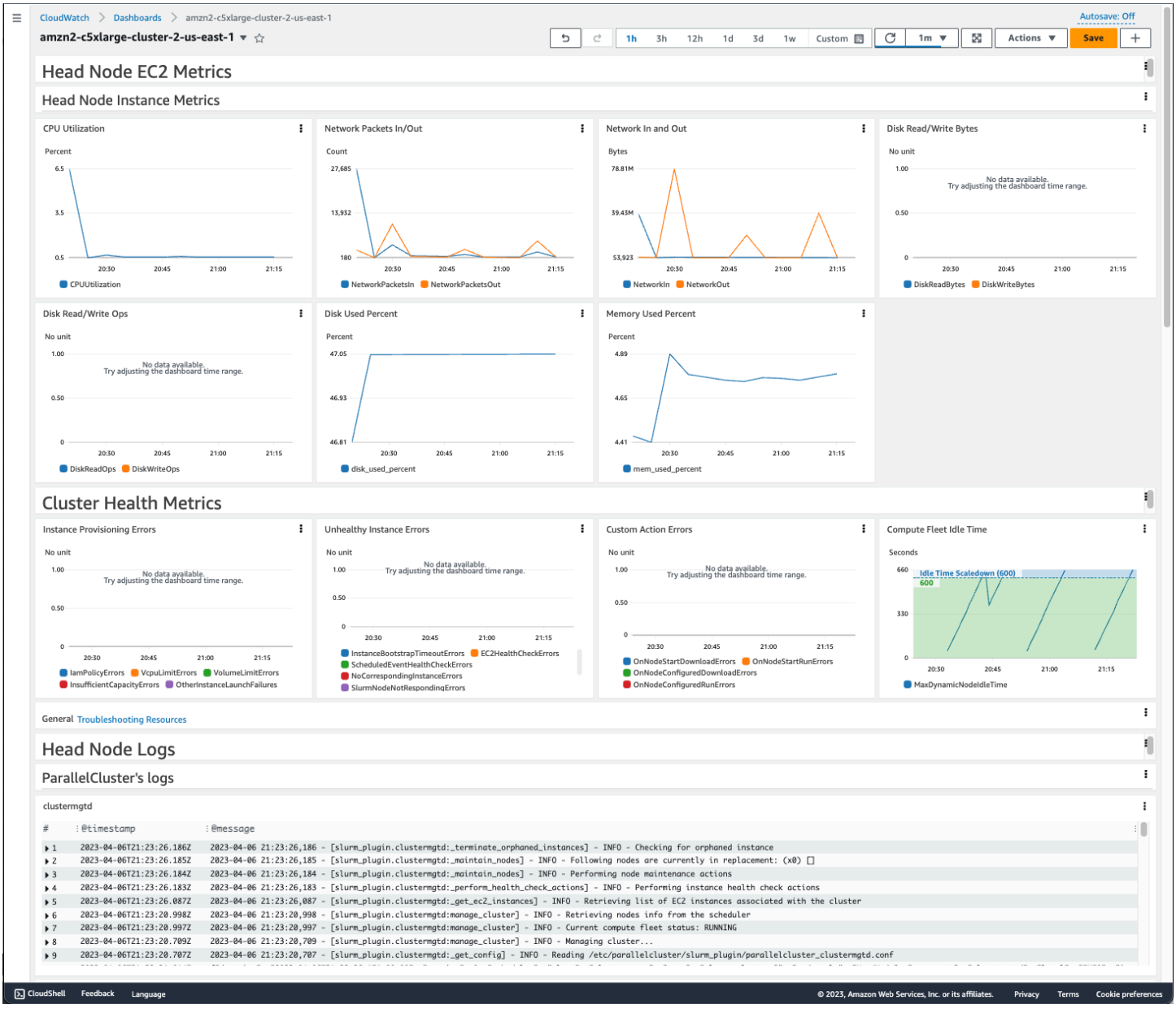
AWS ParallelCluster 计算节点根卷标签

标签密钥	标签值
ClusterName	*clustername*
Application	parallelcluster- *clustername*
aws-parallelcluster-node-type	Compute
QueueName	*queue-name*
Version	*2.11.9*

亚马逊 CloudWatch 控制面板

从 AWS ParallelCluster 版本 2.10.0 开始，在创建集群时会创建一个 Amazon CloudWatch 控制面板。这样可以更轻松地监控集群中的节点和查看 Amazon Logs 中存储的 CloudWatch 日志。控制面板的名称为 parallelcluster-*ClusterName*-*Region*。*ClusterName* 是你的集群的名字而且 *Region* 是集 AWS 区域 群的。您可以在控制台中访问控制面板，也可以通过打开 https://console.aws.amazon.com/cloudwatch/home?region=*Region*#dashboards:name=parallelcluster-*ClusterName* 来访问控制面板。

下图显示了集群的示例 CloudWatch 仪表板。



仪表板的第一部分显示 Head Node EC2 指标的图表。如果您的集群具有共享存储，则下一部分将显示共享存储指标。最后一部分列出了按 ParallelCluster日志、调度程序日志、NICEDCV集成日志和系统日志分组的 Head Node 日志。

有关亚马逊 CloudWatch 控制面板的更多信息，请参阅亚马逊 CloudWatch 用户指南中的使用亚马逊 CloudWatch 控制面板。

如果您不想创建 Amazon CloudWatch 控制面板，则必须完成以下步骤：首先，在配置文件中添加一个[dashboard]部分，然后将该部分的名称作为dashboard_settings设置值添加到您的分[cluster]区中。在 [dashboard] 部分中，设置 enable = false。

例如，如果 [dashboard] 部分命名为 myDashboard，并且 [cluster] 部分命名为 myCluster，则更改类似于下面所示。

```
[cluster MyCluster]
dashboard_settings = MyDashboard
...

[dashboard MyDashboard]
enable = false
```

与 Amazon CloudWatch 日志集成

从 AWS ParallelCluster 版本 2.6.0 开始，默认情况下，常用日志存储在 CloudWatch 日志中。有关 CloudWatch 日志的更多信息，请参阅 Amazon CloudWatch 日志用户指南。要配置 CloudWatch 日志集成，请参阅[cw_log]部分和cw_log_settings设置。

将为每个集群创建一个名为 /aws/parallelcluster/*cluster-name* 的日志组（例如 /aws/parallelcluster/testCluster）。每个节点上的每个日志（如果路径包含 *，则为一组日志）都有一个名为 *{hostname}*.*{instance_id}*.*{logIdentifier}* 的日志流。（例如 ip-172-31-10-46.i-02587cf29cc3048f3.nodewatcher。）日志数据 CloudWatch 由代理发送到，该CloudWatch 代理像root在所有集群实例上一样运行。

从 AWS ParallelCluster 版本 2.10.0 开始，在创建集群时会创建一个 Amazon CloudWatch 控制面板。通过此控制面板，您可以轻松查看存储在日志中的 CloudWatch 日志。有关更多信息，请参阅 亚马逊 CloudWatch 控制面板。

此列表包含 *logIdentifier* 以及可用于平台、调度器和节点的日志流的路径。

适用于平台、调度器和节点的日志流

平台	调度器	Nodes	日志流
amazon centos ubuntu	awsbat slurm	HeadNo	dcv-authenticator：/var/log/parallelcluster/pcluster_dcv_authenticator.log dcv-ext-authenticator: /var/log/parallelcluster/pcluster_dcv_connect.log dcv-agent：/var/log/dcv/agent.*.log

平台	调度器	Nodes	日志流
			dcv-xsession：/var/log/dcv/dcv-xsession.*.log dcv-server：/var/log/dcv/server.log dcv-session-launcher: /var/log/dcv/sessionlauncher.log Xdcv：/var/log/dcv/Xdcv.*.log cfn-init：/var/log/cfn-init.log chef-client：/var/log/chef-client.log
amazon centos ubuntu	awsbatc slurm	Comput eet HeadNc	cloud-init：/var/log/cloud-init.log supervisord：/var/log/supervisord.log
amazon centos ubuntu	slurm	Comput eet	cloud-init-output: /var/log/cloud-init-output.log computemgtd：/var/log/parallelcluster/computemgtd slurmd：/var/log/slurmd.log
amazon centos ubuntu	slurm	HeadNc	clustermgtd：/var/log/parallelcluster/clustermgtd slurm_resume：/var/log/parallelcluster/slurm_resume.log slurm_suspend：/var/log/parallelcluster/slurm_suspend.log slurmctld：/var/log/slurmctld.log
amazon centos	awsbatc slurm	Comput eet HeadNc	system-messages：/var/log/messages

平台	调度器	Nodes	日志流
ubuntu	awsbatc slurm	Comput eet HeadNc	syslog：/var/log/syslog

使用集群中的作业将达到RUNNINGSUCCEEDED、或FAILED状态的任务的输出 AWS Batch 存储在 CloudWatch 日志中。日志组为 /aws/batch/job，日志流名称格式为 *jobDefinitionName*/default/*ecs_task_id*。默认情况下，这些日志设置为永不过期，但您可以修改保留期。有关更多信息，请参阅《Amazon 日志用户指南》中的“ CloudWatch 日志” 中的更改 CloudWatch 日志数据保留期。

> **Note**
>
> chef-client、、cloud-init-output、clustermgtdcomputemgtd、和slurm_resume，slurm_suspend已在 2.9.0 AWS ParallelCluster 版本中添加。对于 AWS ParallelCluster 版本 2.6.0，/var/log/cfn-init-cmd.log(cfn-init-cmd) 和 /var/log/cfn-wire.log (cfn-wire) 也存储在 CloudWatch 日志中。

Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) 是一种具有操作系统旁路功能的网络设备，可与同一子网上的其他实例进行低延迟的网络通信。EFA通过 Libfabric 公开，并且可供使用消息传递接口 (MPI) 的应用程序使用。

要EFA与一起使用 AWS ParallelCluster，请将该行enable_efa = true添加到该[queue]部分。

要查看支持的EC2实例列表EFA，请参阅 Amazon Linux 实例EC2用户指南中的支持的实例类型。

有关 enable_efa 设置的更多信息，请参阅 [queue] 部分中的 enable_efa。

应使用集群置放群组来最大限度地减少实例之间的延迟。有关更多信息，请参阅placement和placement_group。

有关更多信息，请参阅 Amazon EC2 用户指南中的弹性结构适配器以及使用弹性结构适配器扩展HPC工作负载和 AWS ParallelClusterAWS开源博客。

> **Note**
>
> 默认情况下，Ubuntu 启用分发 ptrace （过程跟踪）保护。从 AWS ParallelCluster 2,6.0 开始，ptrace 保护已禁用，这样 Libfabric 才能正常运行。有关更多信息，请参阅《亚马逊EC2用户指南》中的禁用 ptrace 保护。

> **Note**
>
> 2.10.1 版本中增加了EFA对基于 ARM 的 Graviton2 实例的支持。 AWS ParallelCluster

Intel Select Solutions

AWS ParallelCluster 可用作用于仿真和建模的英特尔精选解决方案。配置经过验证，符合英特尔HPC平台规范设定的标准，使用特定的英特尔实例类型，并配置为使用 Elastic Fabric Adapter (EFA) 网络接口。 AWS ParallelCluster 是第一个满足英特尔精选解决方案计划要求的云解决方案。支持的实例类型包括 `c5n.18xlarge`、`m5n.24xlarge` 和 `r5n.24xlarge`。下面提供了与 Intel Select Solutions 标准兼容的配置示例。

Example Intel Select Solutions 配置

```
[global]
update_check = true
sanity_check = true
cluster_template = intel-select-solutions

[aws]
aws_region_name = <Your AWS ##>

[scaling demo]
scaledown_idletime = 5

[cluster intel-select-solutions]
key_name = <Your SSH key name>
base_os = centos7
scheduler = slurm
enable_intel_hpc_platform = true
master_instance_type = c5.xlarge
vpc_settings = <Your VPC section>
```

```
scaling_settings = demo
queue_settings = c5n,m5n,r5n
master_root_volume_size = 200
compute_root_volume_size = 80

[queue c5n]
compute_resource_settings = c5n_i1
enable_efa = true
placement_group = DYNAMIC

[compute_resource c5n_i1]
instance_type = c5n.18xlarge
max_count = 5

[queue m5n]
compute_resource_settings = m5n_i1
enable_efa = true
placement_group = DYNAMIC

[compute_resource m5n_i1]
instance_type = m5n.24xlarge
max_count = 5

[queue r5n]
compute_resource_settings = r5n_i1
enable_efa = true
placement_group = DYNAMIC

[compute_resource r5n_i1]
instance_type = r5n.24xlarge
max_count = 5
```

有关 AWS ParallelCluster 英特尔HPC平台规格的更多信息，请参阅英特尔HPC平台规格。

启用英特尔 MPI

英特MPI尔可在 AWS ParallelCluster AMIs。要使用英特尔MPI，您必须确认并接受英特尔简化软件许可证的条款。默认情况下，路径上MPI会放置 Open。要启用 Intel MPI 而不是 OpenMPI，必须先加载英特尔MPI模块。然后需要使用 `module load intelmpi` 安装最新版本。模块的确切名称随每次更新发生变化。要查看哪些模块可用，请运行 `module avail`。输出如下所示。

```
$ module avail
```

```
--------------------------------------- /usr/share/Modules/modulefiles
 ---------------------------------------
dot                      libfabric-aws/1.8.1amzn1.3 module-info               null
                    use.own
module-git               modules                    openmpi/4.0.2

------------------------------------------------- /etc/modulefiles
 -------------------------------------------------

--------------------------------- /opt/intel/impi/2019.7.217/intel64/modulefiles
 ---------------------------------
intelmpi
```

```
$ module load intelmpi
```

要查看加载了哪些模块，请运行 `module list`。

```
$ module list
Currently Loaded Modulefiles:
  1) intelmpi
```

要验证英特尔MPI是否已启用，请运行`mpirun --version`。

```
$ mpirun --version
Intel(R) MPI Library for Linux* OS, Version 2019 Update 7 Build 20200312 (id:
 5dc2dd3e9)
Copyright 2003-2020, Intel Corporation.
```

加载英特尔MPI模块后，将更改多个路径以使用英特尔MPI工具。要运行英特尔MPI工具编译的代码，请先加载英特尔MPI模块。

> **Note**
>
> 英特尔与基于 AWS Graviton 的实例MPI不兼容。

> **Note**
>
> 在 2.5.0 AWS ParallelCluster 版本之前，MPI英特尔 AWS ParallelCluster AMIs在中国（北京）和中国（宁夏）地区不可用。

英特尔HPC平台规格

AWS ParallelCluster 符合英特尔HPC平台规范。英特尔HPC平台规范提供了一组计算、结构、内存、存储和软件要求，以帮助实现高标准的质量和与HPC工作负载的兼容性。有关更多信息，请参阅经认证的英特尔HPC平台规范和应用程序与英特尔HPC平台规范兼容。

要符合英特尔HPC平台规范，必须满足以下要求：

- 操作系统必须是 CentOS 7 (`base_os = centos7`)。
- 计算节点的实例类型必须具有 Intel CPU 和至少 64 GB 的内存。对于实例类型的 c5 系列，这意味着实例类型必须至少为 `c5.9xlarge` (`compute_instance_type = c5.9xlarge`)。
- 头节点的存储空间必须至少为 200 GB。
- 必须接受 Intel Parallel Studio 的最终用户许可协议 (`enable_intel_hpc_platform = true`)。
- 每个计算节点的存储空间必须至少为 80 GB (`compute_root_volume_size = 80`)。

存储可以是本地的，也可以是网络上的（从头节点、Amazon NFS 共享EBS或FSx用于 Lustre），也可以共享。

Arm Performance Libraries

从 AWS ParallelCluster 版本 2.10.1 开始，Arm Performance Librar AWS ParallelCluster AMIs y 可在该设置的`alinux2centos8ubuntu1804`、、和`ubuntu2004`值上使用。`base_os`Arm Performance Libraries 为 Arm 处理器上的高性能计算应用程序提供优化的标准核心数学库。要使用 Arm Performance Libraries，您必须确认并接受 Arm Performance Libraries（免费版）- 最终用户许可协议的条款。有关 Arm Performance Libraries 的更多信息，请参阅免费 Arm Performance Libraries。

要启用 Arm 性能库，必须先加载 Arm 性能库模块。 `Armpl-21.0.0`需要 GCC -9.3 作为要求，当你加载模块时，`armpl/21.0.0gcc/9.3`模块也会被加载。模块的确切名称随每次更新发生变化。要查看哪些模块可用，请运行 `module avail`。然后，需要使用 `module load armpl` 安装最新版本。输出如下所示。

```
$ module avail

------------------------------ /usr/share/Modules/modulefiles
 --------------------------------
armpl/21.0.0         dot                  libfabric-aws/1.11.1amzn1.0
 module-git
module-info          modules              null                 openmpi/4.1.0
 use.own
```

要加载模块，请运行 `module load` *modulename*。您可以将其添加到用于运行 `mpirun` 的脚本中。

```
$ module load armpl

Use of the free of charge version of Arm Performance Libraries is subject to the terms
 and
conditions of the Arm Performance Libraries (free version) -  End User License
 Agreement
(EULA). A copy of the EULA can be found in the
'/opt/arm/armpl/21.0.0/arm-performance-libraries_21.0_gcc-9.3/license_terms' folder
```

要查看加载了哪些模块，请运行 `module list`。

```
$ module list
Currently Loaded Modulefiles:
1) /opt/arm/armpl/21.0.0/modulefiles/armpl/gcc-9.3
2) /opt/arm/armpl/21.0.0/modulefiles/armpl/21.0.0_gcc-9.3
3) armpl/21.0.0
```

要验证是否已启用 Arm Performance Libraries，请运行示例测试。

```
$ sudo chmod 777 /opt/arm/armpl/21.0.0/armpl_21.0_gcc-9.3/examples
$ cd /opt/arm/armpl/21.0.0/armpl_21.0_gcc-9.3/examples
$ make
...
Testing: no example difference files were generated.
Test passed OK
```

加载 Arm Performance Libraries 模块后，将会更改多个路径以使用 Arm Performance Libraries 工具。要运行由 Arm Performance Libraries 工具编译的代码，请先加载 Arm Performance Libraries 模块。

> **Note**
>
> AWS ParallelCluster 2.10.1 和 2.10.4 之间的版本使用。`armpl/20.2.1`

通过 Amazon 连接到头节点 DCV

Amazon DCV 是一种远程可视化技术，使用户能够安全地连接到托管在远程高性能服务器上的图形密集型 3D 应用程序。有关更多信息，请参阅 Amazon DCV。

使用`base_os = alinux2`、`base_os = centos7``base_os = ubuntu1804`或时，Amazon DCV 软件会自动安装在头节点上`base_os = ubuntu2004`。

如果头节点是ARM实例，则使用`base_os = alinux2`、或时会自动在其上安装 Amazon DCV 软件`base_os = ubuntu1804`。`base_os = centos7`

要在头节点DCV上启用 Amazon，`dcv_settings`必须包含已`enable = master`且`base_os`必须设置为`alinux2`、`centos7``ubuntu1804`、或的`[dcv]`部分的名称`ubuntu2004`。如果头节点是ARM实例，则`base_os`必须设置为`alinux2``centos7`、或`ubuntu1804`。这样， AWS ParallelCluster 将群集配置参数设置`shared_dir`为DCV服务器存储文件夹。

```
[cluster custom-cluster]
...
dcv_settings = custom-dcv
...
[dcv custom-dcv]
enable = master
```

有关 Amazon DCV 配置参数的更多信息，请参阅`dcv_settings`。要连接到 Amazon DCV 会话，请使用`pcluster dcv`命令。

> **Note**
>
> 2.10.4 AWS ParallelCluster 版本中删除`centos8`了对 Amaz DCV on on 的支持。在 2.10.0 AWS ParallelCluster 版本中增加了对 Amaz DCV on `centos8` 的支持。2.9.0 版本中增加了DCV对 AWS 基于 Graviton 的实例上的 Amazon 支持。 AWS ParallelCluster DCV在 2.6.0 AWS ParallelCluster 版本中添加`alinux2``ubuntu1804`了对 Amazon 的支持。在 2.5.0 AWS ParallelCluster 版本中增加了`centos7`对 Amaz DCV on 的支持。

> **Note**
>
> 2.8. DCV 0 AWS ParallelCluster 和 2.8.1 版本中 AWS 基于 Graviton 的实例不支持亚马逊。

亚马逊DCVHTTPS证书

亚马逊会DCV自动生成自签名证书，以保护亚马逊DCV客户端和亚马逊DCV服务器之间的流量。

要将默认的自签名 Amazon DCV 证书替换为其他证书，请先连接到头节点。然后，在运行 `pcluster dcv` 命令之前，将证书和密钥复制到 `/etc/dcv` 文件夹。

有关更多信息，请参阅《Amazon DCV 管理员指南》中的更改TLS证书。

许可 Amazon DCV

在亚马逊EC2实例上运行时，Amazon DCV 服务器不需要许可服务器。但是，Amazon DCV 服务器必须定期连接到 Amazon S3 存储桶，以确定是否有有效的许可证可用。

AWS ParallelCluster 自动将所需的权限添加到`ParallelClusterInstancePolicy`。使用自定义IAM实例策略时，请使用《亚马逊DCV管理员指南》中 Amazon DCV on Amaz on EC2 中描述的权限。

有关故障排除提示，请参阅 对 Amazon 中的问题进行故障排除 DCV。

使用 `pcluster update`

从 2.8.0 AWS ParallelCluster 版开始，`pcluster update`分析用于创建当前集群的设置以及配置文件中的设置以发现问题。如果发现任何问题，就会进行报告并显示修复这些问题所要执行的步骤。例如，如果将 `compute_instance_type` 设置更改为不同的实例类型，则必须先停止计算实例集，然后才能继续更新。此问题发现后即会报告。如果未报告任何阻碍性问题，则会提示您是否要应用更改。

每个设置的文档都定义了该设置的更新策略。

更新策略：可以在更新期间更改这些设置。，更新策略：可以在更新期间更改此设置。

可以更改这些设置，并可使用 `pcluster update` 更新集群。

更新策略：如果更改此设置，则不允许更新。

如果尚未删除现有集群，则无法更改这些设置。必须恢复更改，或者必须删除集群（使用 `pcluster delete`），然后在旧集群的位置创建新集群（使用 `pcluster create`）。

更新策略：在更新期间不分析此设置。

可以更改这些设置，并使用 pcluster update 更新集群。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

当存在计算实例集时，无法更改这些设置。必须恢复更改，或者必须停止计算实例集（使用 pcluster stop），进行更新（使用 pcluster update），然后创建新的计算实例集（使用 pcluster start）。

更新策略：更新期间不能减小此设置。

这些设置可以更改，但不能减小。如果必须减小这些设置，则必须删除集群（使用 pcluster delete），然后创建新集群（使用 pcluster create）。

更新策略：要将队列大小减至当前节点数以下，需要先停止计算实例集。

可以更改这些设置，但如果更改会将队列大小减至当前大小以下，则必须停止计算实例集（使用 pcluster stop），进行更新（使用 pcluster update），然后创建新的计算实例集（使用 pcluster start）。

更新策略：减少队列中静态节点的数量需要先停止计算实例集。

可以更改这些设置，但如果更改会将队列中的静态节点数量减少到当前大小以下，则必须停止计算实例集（使用 pcluster stop），进行更新（使用 pcluster update），然后创建新的计算实例集（使用 pcluster start）。

更新策略：如果更改此设置，则不允许更新。不能强制更新此设置。

如果尚未删除现有集群，则无法更改这些设置。必须恢复更改，或者必须删除集群（使用 pcluster delete），然后在旧集群的位置创建新集群（使用 pcluster create）。

更新政策：如果配置中未指定 AWS ParallelCluster 托管 Amazon FSx for Lustre 文件系统，则可以在更新期间更改此设置。

如果[cluster]fsx_settings未指定此设置，或者如果两者都fsx_settings指定FSx为挂载 Lustre 文件系统的现有外部文件，则可以更改此设置。fsx-fs-id[fsx fs]

下面的示例演示了一个 pcluster update，其中包含一些阻碍更新的更改。

```
$ pcluster update
  Validating configuration file /home/username/.parallelcluster/config...
Retrieving configuration from CloudFormation for cluster test-1...
Found Changes:
```

```
#   section/parameter         old value                 new value
--  ------------------------  ------------------------  ------------------------
    [cluster default]
01* compute_instance_type     t2.micro                  c4.xlarge
02* ebs_settings              ebs2                      -

    [vpc default]
03  additional_sg             sg-0cd61884c4ad16341       sg-0cd61884c4ad11234

    [ebs ebs2]
04* shared_dir                shared                     my/very/very/long/sha...

Validating configuration update...
The requested update cannot be performed. Line numbers with an asterisk indicate
updates requiring additional actions. Please look at the details below:

#01
Compute fleet must be empty to update "compute_instance_type"
How to fix:
Make sure that there are no jobs running, then run the following command:
  pcluster stop -c $CONFIG_FILE $CLUSTER_NAME

#02
Cannot add/remove EBS Sections
How to fix:
Revert "ebs_settings" value to "ebs2"

#04
Cannot change the mount dir of an existing EBS volume
How to fix:
Revert "my/very/very/long/shared/dir" to "shared"

In case you want to override these checks and proceed with the update please
use the --force flag. Note that the cluster could end up in an unrecoverable
state.

Update aborted.
```

AMI修补和EC2实例更换

为确保所有动态启动的集群计算节点的行为方式一致，请 AWS ParallelCluster 禁用集群实例自动操作系统更新。此外，还会为的 AWS ParallelCluster AMIs每个版本 AWS ParallelCluster 及其关联版本构

建一组特定的版本CLI。这组特定的版本AMIs保持不变，只有它们所针对的 AWS ParallelCluster 版本支持。 AWS ParallelCluster AMIs对于已发布的版本未更新。

但是，由于突发的安全问题，客户可能希望向这些补丁添加补丁，AMIs然后使用已修补的补AMI丁更新其集群。这与 AWS ParallelCluster 责任共担模式一致。

要查看您当前使用的 AWS ParallelCluster CLI版本所 AWS ParallelCluster AMIs支持的特定集合，请运行：

```
$ pcluster version
```

然后在 AWS ParallelCluster的 GitHub 存储库中查看 amis.txt。

AWS ParallelCluster 头节点是一个静态实例，你可以手动更新它。如果实例类型没有实例存储，则从 2.11 AWS ParallelCluster 版本开始完全支持重启和重启头节点。有关更多信息，请参阅《Amazon Linux 实例EC2用户指南》中的带有实例存储卷的实例类型。您无法AMI为现有集群更新。

从 3.0.0 AWS ParallelCluster 版开始，完全支持通过AMI更新集群计算实例来重启和重启头节点。要使用这些功能，请考虑升级到最新版本。

头节点实例更新或替换

在某些情况下，可能需要重启或重新引导头节点。例如，当您手动更新操作系统，或者 AWS 实例计划停用强制重启头节点实例时，必须执行此操作。

如果实例没有临时驱动器，则可以随时停止并重新启动该实例。在计划停用的情况下，启动已停止的实例会将其迁移为使用新硬件。

同样，您可以手动停止和启动没有实例存储的实例。对于没有临时卷的实例的这种情况以及其他情况，请参阅停止和启动集群的头节点。

如果实例具有临时驱动器且已停止，则实例存储中的数据将会丢失。您可以根据实例存储卷中的表确定用于头节点的实例类型是否具有实例存储。

以下各节介绍了使用具有实例存储卷的实例时的限制。

实例存储限制

在实例存储中使用 2.11 AWS ParallelCluster 版本和实例类型的限制如下：

- 如果临时驱动器未加密（encrypted_ephemeral参数设置为false或未设置），则 AWS ParallelCluster 实例在实例停止后无法启动。这是因为不存在的旧临时设备上的信息会被写入 fstab，而操作系统会尝试挂载不存在的存储。
- 对临时驱动器进行加密（encrypted_ephemeral参数设置为true）后，可以在停止后启动 AWS ParallelCluster 实例，但新的临时驱动器未设置、已装载或不可用。
- 加密临时驱动器后，可以重新启动 AWS ParallelCluster 实例，但无法访问旧的临时驱动器（在实例重启后仍然存在），因为加密密钥是在重启后丢失的内存中创建的。

唯一支持的情况是临时驱动器未加密时的实例重启。这是因为驱动器在重启后会保留下来，并可通过写入 fstab 的条目而装回。

实例存储限制解决方法

首先，保存您的数据。要检查是否有需要保留的数据，请查看 ephemeral_dir 文件夹（默认情况下为 /scratch）中的内容。您可以将数据传输到根卷或连接到集群的共享存储系统，例如 Amazon FSx、Amazon 或 Ama EFS zon EBS。请注意，将数据传输到远程存储可能会产生额外费用。

限制的根本原因在于 AWS ParallelCluster 用于格式化和装载实例存储卷的逻辑。该逻辑会使用以下格式向 /etc/fstab 中添加一个条目：

```
$ /dev/vg.01/lv_ephemeral ${ephemeral_dir} ext4 noatime,nodiratime 0 0
```

${ephemeral_dir} 是 pcluster 配置文件中 ephemeral_dir 参数的值（默认为 /scratch）。

添加此行是为了在节点重启时自动重新挂载实例存储卷。这是可取的，因为临时驱动器中的数据在重启后仍会保留。但是，在经过启动或停止循环后，临时驱动器上的数据不会保留。这意味着驱动器会被格式化并在没有数据的情况下进行挂载。

唯一支持的情况是临时驱动器未加密时的实例重启。这是因为驱动器在重启后会保留下来，并且因为其写入 fstab 而得以装回。

要在所有其他情况下保留数据，您必须在停止实例之前删除逻辑卷条目。例如，在停止实例之前从 /etc/fstab 中删除 /dev/vg.01/lv_ephemeral。完成此操作后，您可以启动实例而不挂载临时卷。但是，重新挂载的实例存储在停止或启动实例后将不可用。

在保存数据并删除 fstab 条目后，请继续下一节。

停止和启动集群的头节点

> **Note**
>
> 从 AWS ParallelCluster 版本 2.11 开始，只有当实例类型没有实例存储时，才支持头节点停止和启动。

1. 确认集群中没有任何正在运行的作业。

   使用时 Slurm 调度器：

   - 如果未指定 `sbatch --no-requeue` 选项，则对正在运行的作业进行重新排队。
   - 如果指定了 `--no-requeue` 选项，则正在运行的作业将会失败。
2. 请求集群计算实例集停止：

   ```
   $ pcluster stop cluster-name
   Compute fleet status is: RUNNING. Submitting status change request.
   Request submitted successfully. It might take a while for the transition to
    complete.
   Please run 'pcluster status' if you need to check compute fleet status
   ```
3. 等到计算实例集状态变为 STOPPED：

   ```
   $ pcluster status cluster-name
   ...
   ComputeFleetStatus: STOP_REQUESTED
   $ pcluster status cluster-name
   ...
   ComputeFleetStatus: STOPPED
   ```
4. 要通过操作系统重启或实例重启进行手动更新，您可以使用 AWS Management Console 或 AWS CLI。下面是使用 AWS CLI的示例。

   ```
   $ aws ec2 stop-instances --instance-ids 1234567890abcdef0
   {
     "StoppingInstances": [
       {
         "CurrentState": {
           "Name": "stopping"
           ...
   ```

```
      },
      "InstanceId": "i-1234567890abcdef0",
      "PreviousState": {
        "Name": "running"
        ...
      }
    }
  ]
}
$ aws ec2 start-instances --instance-ids 1234567890abcdef0
{
  "StartingInstances": [
    {
      "CurrentState": {
        "Name": "pending"
        ...
      },
      "InstanceId": "i-1234567890abcdef0",
      "PreviousState": {
        "Name": "stopped"
        ...
      }
    }
  ]
}
```

5. 启动集群的计算实例集：

```
$ pcluster start cluster-name
Compute fleet status is: STOPPED. Submitting status change request.
Request submitted successfully. It might take a while for the transition to
 complete.
Please run 'pcluster status' if you need to check compute fleet status
```

AWS ParallelCluster CLI命令

pclusterpcluster-config还有 AWS ParallelCluster CLI命令。您可以使用pcluster在和中启动和管理HPC集群pcluster-config来更新您的配置。 AWS Cloud

要使用pcluster，您必须拥有拥有运行该IAM角色所需权限的角色。

```
pcluster [ -h ] ( create | update | delete | start | stop | status | list |
                  instances | ssh | dcv | createami | configure | version ) ...
 pcluster-config [-h] (convert) ...
```

主题

- pcluster
- pcluster-config

pcluster

pcluster是主要 AWS ParallelCluster CLI命令。您pcluster用于在中启动和管理HPC集群 AWS Cloud。

```
pcluster [ -h ] ( create | update | delete | start | stop | status | list |
                  instances | ssh | dcv | createami | configure | version ) ...
```

参数

pcluster ***command***

可能的选项：configure、create、createami、dcv、delete、instances、list、ssh、start、statu

子命令：

主题

- pcluster configure

- pcluster create
- pcluster createami
- pcluster dcv
- pcluster delete
- pcluster instances
- pcluster list
- pcluster ssh
- pcluster start
- pcluster status
- pcluster stop
- pcluster update
- pcluster version

pcluster configure

开始 AWS ParallelCluster 配置。有关更多信息，请参阅 配置 AWS ParallelCluster。

```
pcluster configure [ -h ] [ -c CONFIG_FILE ] [ -r REGION ]
```

命名的参数

-h, --help

显示 pcluster configure 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件的完整路径。

默认值为 ~/.parallelcluster/config。

有关更多信息，请参阅 配置 AWS ParallelCluster。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。如果指定了此选项，则配置将跳过 AWS 区域 检测。

要删除中的网络资源VPC，您可以删除 CloudFormation 网络堆栈。堆栈名称以" 开头 parallelclusternetworking-"并以" YYYYMMDDHHMMSS "格式包含创建时间。您可以使用 list-stacks 命令列出堆栈。

```
$ aws --region us-east-1 cloudformation list-stacks \
   --stack-status-filter "CREATE_COMPLETE" \
   --query "StackSummaries[].StackName" | \
   grep -e "parallelclusternetworking-"
    "parallelclusternetworking-pubpriv-20191029205804"
```

可以使用 delete-stack 命令删除堆栈。

```
$ aws --region us-east-1 cloudformation delete-stack \
   --stack-name parallelclusternetworking-pubpriv-20191029205804
```

为您pcluster configure创建VPC的不是在 CloudFormation 网络堆栈中创建的。您可以在控制台中VPC手动将其删除，也可以使用 AWS CLI。

```
$ aws --region us-east-1 ec2 delete-vpc --vpc-id vpc-0b4ad9c4678d3c7ad
```

pcluster create

创建新集群。

```
pcluster create [ -h ] [ -c CONFIG_FILE ] [ -r REGION ] [ -nw ] [ -nr ]
                [ -u TEMPLATE_URL ] [ -t CLUSTER_TEMPLATE ]
                [ -p EXTRA_PARAMETERS ] [ -g TAGS ]
                cluster_name
```

定位参数

cluster_name

定义集群的名称。 AWS CloudFormation 堆栈名称是parallelcluster-*cluster_name*。

命名的参数

-h, --help

显示 pcluster create 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。用于为新集群选择 AWS 区域 的优先级顺序如下：

1. pcluster create 的 -r 或 --region 参数。
2. AWS_DEFAULT_REGION 环境变量。
3. aws_region_name在 AWS ParallelCluster 配置文件[aws]部分进行设置（默认位置为~/.parallelcluster/config.）这是由 pcluster configure 命令更新的位置。
4. region在 AWS CLI 配置文件[default]部分进行设置 (~/.aws/config.)

-nw, --nowait

指示在运行堆栈命令后不等待堆栈事件。

默认值为 False。

-nr, --norollback

禁止在出现错误时回滚 堆栈。

默认值为 False。

-u *TEMPLATE_URL*, --template-url *TEMPLATE_URL*

如果创建时使用了自定义 AWS CloudFormation 模板，则URL为该模板指定。

-t *CLUSTER_TEMPLATE*, --cluster-template *CLUSTER_TEMPLATE*

指示要使用的集群模板。

-p *EXTRA_PARAMETERS*, --extra-parameters *EXTRA_PARAMETERS*

向堆栈创建添加额外的参数。

-g *TAGS*, --tags *TAGS*

指定要添加到堆栈的其他标签。

当命令被调用并开始轮询该调用的状态时，可以安全地使用“Ctrl-C”退出。您可以通过调用 pcluster status mycluster 返回以查看当前状态。

使用 2.11.7 AWS ParallelCluster 版本的示例：

```
$ pcluster create mycluster
    Beginning cluster creation for cluster: mycluster
Info: There is a newer version 3.1.4 of AWS ParallelCluster available.
Creating stack named: parallelcluster-mycluster
Status: ComputeFleetHITSubstack - CREATE_IN_PROGRESS
$ pcluster create mycluster --tags '{ "Key1" : "Value1" , "Key2" : "Value2" }'
```

pcluster createami

(Linux/macOS) 创建用于的自定义AMI项。 AWS ParallelCluster

```
pcluster createami [ -h ] -ai BASE_AMI_ID -os BASE_AMI_OS
                   [ -i INSTANCE_TYPE ] [ -ap CUSTOM_AMI_NAME_PREFIX ]
                   [ -cc CUSTOM_AMI_COOKBOOK ] [--no-public-ip]
                   [ -post-install POST_INSTALL_SCRIPT ]
                   [ -c CONFIG_FILE ] [-t CLUSTER_TEMPLATE]
                   [--vpc-id VPC_ID] [--subnet-id SUBNET_ID]
                   [ -r REGION ]
```

必需的依赖项

除了 AWS ParallelCluster CLI，还需要以下依赖关系才能运行`pcluster createami`：

- Packer：从 https://developer.hashicorp.com/packer/downloads 下载最新版本。

> **Note**
>
> 在 2.8.0 AWS ParallelCluster 版本之前，必须使用 Berkshelf（使用安装的`gem install berkshelf`）。`pcluster createami`

命名的参数

`-h, --help`

显示 `pcluster createami` 的帮助文本。

-ai *BASE_AMI_ID*, --ami-id *BASE_AMI_ID*

指定AMI用于构建的基础 AWS ParallelCluster AMI。

-os *BASE_AMI_OS*, --os *BASE_AMI_OS*

指定基础的操作系统AMI。有效的选项为：`alinux2`、`ubuntu1804`、`ubuntu2004` 和 `centos7`。

> **Note**
>
> 操作系统支持不同 AWS ParallelCluster 版本的更改：
>
> - 在 2.10.4 AWS ParallelCluster 版本中删除`centos8`了对 Support 的支持。
> - 在 AWS ParallelCluster 版本 2.10.0 中添加了对 `centos8` 的支持，并删除了对 `centos6` 的支持。
> - 2.6.0 版本中增加了对 alinux2 的 AWS ParallelCluster 支持。
> - 2.5.0 版本中增加了对 ubuntu1804 的支持。 AWS ParallelCluster

-i *INSTANCE_TYPE*, --instance-type *INSTANCE_TYPE*

指定用于创建的实例类型AMI。

默认值为 `t2.xlarge`。

> **Note**
>
> 2.4.1 AWS ParallelCluster 版本中增加了对该`--instance-type`参数的支持。

-ap *CUSTOM_AMI_NAME_PREFIX*, --ami-name-prefix *CUSTOM_AMI_NAME_PREFIX*

指定结果的前缀名称 AWS ParallelCluster AMI。

默认值为 `custom-ami-`。

-cc *CUSTOM_AMI_COOKBOOK*, --custom-cookbook *CUSTOM_AMI_COOKBOOK*

指定用于构建. AWS ParallelCluster AMI

--post-install *POST_INSTALL_SCRIPT*

指定安装后脚本的路径。路径必须使用s3://https://、或file://URL方案。示例包括：

- https://*bucket-name*.s3.*region*.amazonaws.com/*path*/post_install.sh
- s3://*bucket-name*/post_install.sh
- file:///*opt/project*/post_install.sh

> **Note**
>
> 2.10.0 AWS ParallelCluster 版本中增加了对该--post-install参数的支持。

--no-public-ip

请勿将公有 IP 地址与用于创建的实例相关联AMI。默认情况下，公有 IP 地址与该实例关联。

> **Note**
>
> 在 2.5.0 AWS ParallelCluster 版本中添加了对该--no-public-ip参数的支持。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

-t *CLUSTER_TEMPLATE*, --cluster-template *CLUSTER_TEMPLATE*

指定 [cluster] 部分 *CONFIG_FILE* 用于检索VPC和子网设置。

> **Note**
>
> 在 2.4.0 AWS ParallelCluster 版本中添加了对该--cluster-template参数的支持。

--vpc-id *VPC_ID*

指定用于构建的 ID AWS ParallelCluster AMI。VPC

> **Note**
>
> 在 2.5.0 AWS ParallelCluster 版本中添加了对该`--vpc-id`参数的支持。

`--subnet-id` *`SUBNET_ID`*

指定用于构建的子网的 ID AWS ParallelCluster AMI。

> **Note**
>
> 在 2.5.0 AWS ParallelCluster 版本中添加了对该`--vpc-id`参数的支持。

`-r` *`REGION`*`, --region` *`REGION`*

指定 AWS 区域 要使用的。默认为使用`pcluster configure`命令 AWS 区域 指定的值。

pcluster dcv

与在头节点上运行的 Amazon DCV 服务器进行交互。

```
pcluster dcv [ -h ] ( connect )
```

`pcluster dcv` *`command`*

可能的选项：`connect`

> **Note**
>
> 操作系统支持在不同 AWS ParallelCluster 版本中对该`pcluster dcv`命令进行更改：
>
> - 在 AWS ParallelCluster 版本 2.10.0 中添加了对 `centos8` 上 `pcluster dcv` 命令的支持。
> - 2.9.0 版本中增加了对 AWS 基于 Graviton 的实例上的`pcluster dcv`命令的 AWS ParallelCluster 支持。
> - 在 AWS ParallelCluster 版本 2.6.0 中添加了对 `ubuntu1804` 上 `pcluster dcv` 命令的支持。

- 在 AWS ParallelCluster 版本 2.5.0 中添加了对 centos7 上 pcluster dcv 命令的支持。

命名的参数

-h, --help

显示 pcluster dcv 的帮助文本。

子命令

pcluster dcv connect

```
pcluster dcv connect [ -h ] [ -k SSH_KEY_PATH ] [ -r REGION ] cluster_name
```

> **Important**
>
> 将在发布 30 秒后URL过期。如果在URL到期之前没有建立连接，请pcluster dcv connect再次运行以生成一个新的URL。

定位参数

cluster_name

指定要连接到的集群的名称。

命名的参数

-h, --help

显示 pcluster dcv connect 的帮助文本。

-k *SSH_KEY_PATH*, --key-path *SSH_KEY_PATH*

用于连接的SSH密钥的密钥路径。

此键必须是在创建集群时在 key_name 配置参数中指定的键。此参数是可选的，但如果未指定，则默认情况下，该密钥必须可供SSH客户端使用。例如，使用 ssh-add 将其添加到 ssh-agent。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

-s, --show-url

显示一次URL性连接到 Amazon DCV 会话。指定此选项时，不打开默认浏览器。

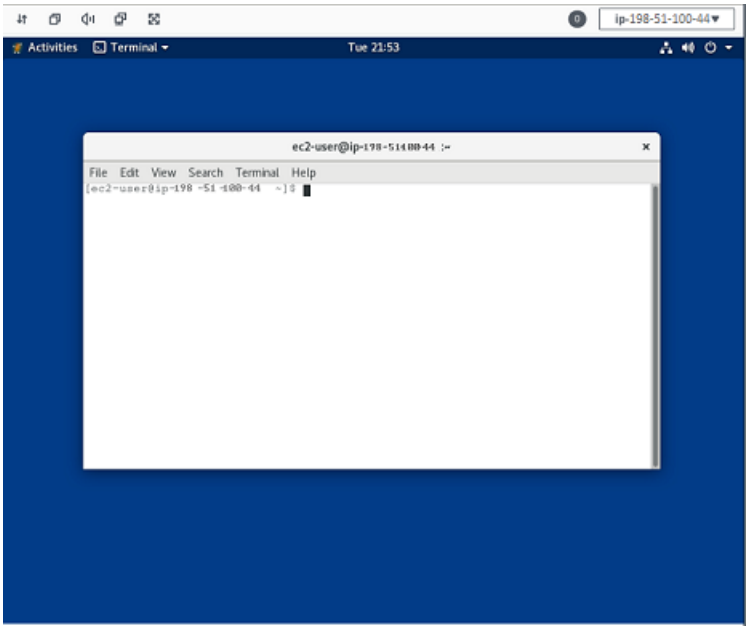
> **Note**
>
> 在 2.5.1 AWS ParallelCluster 版本中添加了对该--show-url参数的支持。

使用 AWS ParallelCluster 版本 2.11.7 的示例：

```
$ pcluster dcv connect -k ~/.ssh/id_rsa mycluster
```

打开默认浏览器以连接到在头节点上运行的 Amazon DCV 会话。

如果尚未启动，则DCV会创建一个新的 Amazon 会话。



pcluster delete

删除集群。

```
pcluster delete [ -h ] [ -c CONFIG_FILE ] [ -r REGION ] [ -nw ] cluster_name
```

定位参数

cluster_name

指定要删除的集群的名称。

命名的参数

-h, --help

显示 pcluster delete 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

--keep-logs

删除集群后保留 CloudWatch 日志数据。日志组将一直保留，直至手动删除，但日志事件将根据 retention_days 设置过期。该设置默认为 14 天。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.6.0 中增加了对 **--keep-logs** 参数的支持。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

当命令被调用并开始轮询该调用的状态时，可以安全地使用“Ctrl-C”退出。您可以通过调用 pcluster status mycluster 返回以查看当前状态。

使用 AWS ParallelCluster 版本 2.11.7 的示例：

```
$ pcluster delete -c path/to/config -r us-east-1 mycluster
Deleting: mycluster
Status: RootRole - DELETE_COMPLETE
Cluster deleted successfully.
```

要删除中的网络资源VPC，您可以删除 CloudFormation 网络堆栈。堆栈名称以" 开头 parallelclusternetworking-"并以" YYYYMMDDHHMMSS "格式包含创建时间。您可以使用 list-stacks 命令列出堆栈。

```
$ aws --region us-east-1 cloudformation list-stacks \
   --stack-status-filter "CREATE_COMPLETE" \
   --query "StackSummaries[].StackName" | \
   grep -e "parallelclusternetworking-"
    "parallelclusternetworking-pubpriv-20191029205804"
```

可以使用 delete-stack 命令删除堆栈。

```
$ aws --region us-east-1 cloudformation delete-stack \
   --stack-name parallelclusternetworking-pubpriv-20191029205804
```

为您pcluster configure创建VPC的不是在 CloudFormation 网络堆栈中创建的。您可以在控制台中VPC手动将其删除，也可以使用 AWS CLI。

```
$ aws --region us-east-1 ec2 delete-vpc --vpc-id vpc-0b4ad9c4678d3c7ad
```

pcluster instances

显示集群中所有实例的列表。

```
pcluster instances [ -h ] [ -c CONFIG_FILE ] [ -r REGION ] cluster_name
```

定位参数

####

显示具有所提供名称的集群的实例。

命名的参数

-h, --help

显示 pcluster instances 的帮助文本。

-c CONFIG_FILE, --config CONFIG_FILE

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

使用 AWS ParallelCluster 版本 2.11.7 的示例：

```
$ pcluster instances -c path/to/config -r us-east-1 mycluster
MasterServer         i-1234567890abcdef0
ComputeFleet         i-abcdef01234567890
```

pcluster list

显示与之关联 AWS ParallelCluster的堆栈列表。

```
pcluster list [ -h ] [ -c CONFIG_FILE ] [ -r REGION ]
```

命名的参数

-h, --help

显示 pcluster list 的帮助文本。

--color

用颜色显示集群状态。

默认值为 False。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 c。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

列出任何名为的 AWS CloudFormation 堆栈的名称。parallelcluster-*

使用 AWS ParallelCluster 版本 2.11.7 的示例：

```
$ pcluster list -c path/to/config -r us-east-1
mycluster            CREATE_IN_PROGRESS  2.11.7
myothercluster       CREATE_IN_PROGRESS  2.11.7
```

pcluster ssh

将 ssh 命令与预先填充的集群用户名和 IP 地址一起运行 。将任意参数附加到 ssh 命令的结尾。可以在配置文件的别名部分中自定义此命令。

```
pcluster ssh [ -h ] [ -d ] [ -r REGION ] cluster_name
```

定位参数

cluster_name

指定要连接到的集群的名称。

命名的参数

-h, --help

显示 pcluster ssh 的帮助文本。

-d, --dryrun

打印命令，该命令将运行并退出。

默认值为 False。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认值为使用 pcluster configure 命令指定的区域。

使用 2.11.7 AWS ParallelCluster 版本的示例：

```
$ pcluster ssh -d mycluster -i ~/.ssh/id_rsa
SSH command: ssh ec2-user@1.1.1.1 -i /home/user/.ssh/id_rsa
```

```
$ pcluster ssh mycluster -i ~/.ssh/id_rsa
```

使用预先填充的集群用户名和 IP 地址运行 ssh 命令：

```
ssh ec2-user@1.1.1.1 -i ~/.ssh/id_rsa
```

在 [aliases] 部分 下的全局配置文件中定义 ssh 命令。它可以是自定义的，如下所示。

```
[ aliases ]
ssh = ssh {CFN_USER}@{MASTER_IP} {ARGS}
```

替换的变量：

CFN_USER

选择的 base_os 的用户名。

MASTER_IP

头节点的 IP 地址。

ARGS

要传递到 ssh 命令的可选参数。

pcluster start

为已停止的集群启动计算队列。

```
pcluster start [ -h ] [ -c CONFIG_FILE ] [ -r REGION ] cluster_name
```

定位参数

cluster_name

启动提供的集群名称的计算队列。

命名的参数

-h, --help

显示 pcluster start 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

使用 AWS ParallelCluster 版本 2.11.7 的示例：

```
$ pcluster start mycluster
Compute fleet status is: RUNNING. Submitting status change request.
Request submitted successfully. It might take a while for the transition to complete.
Please run 'pcluster status' if you need to check compute fleet status
```

此命令将 Auto Scaling 组参数设置为下列项目之一：

- 已用于创建集群的模板中的初始配置值（max_queue_size 和 initial_queue_size）。
- 自集群首次创建以来用于更新集群的配置值。

pcluster status

拉取集群的当前状态。

```
pcluster status [ -h ] [ -c CONFIG_FILE ] [ -r REGION ] [ -nw ] cluster_name
```

定位参数

cluster_name

显示具有所提供名称的集群的状态。

命名的参数

-h, --help

显示 pcluster status 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

-nw, --nowait

指示在处理堆栈命令后不等待堆栈事件。

默认值为 False。

使用 AWS ParallelCluster 版本 2.11.7 的示例：

```
$ pcluster status -c path/to/config -r us-east-1 mycluster
Status: ComputeFleetHITSubstack - CREATE_IN_PROGRESS
```

pcluster stop

停止计算实例集，同时让头节点保持运行。

```
pcluster stop [ -h ] [ -c CONFIG_FILE ] [ -r REGION ] cluster_name
```

定位参数

cluster_name

停止提供的集群名称的计算队列。

使用 AWS ParallelCluster 版本 2.11.7 的示例：

命名的参数

-h, --help

显示 pcluster stop 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

```
$ pcluster stop mycluster
Compute fleet status is: STOPPED. Submitting status change request.
Request submitted successfully. It might take a while for the transition to complete.
Please run 'pcluster status' if you need to check compute fleet status
```

将自动扩缩组参数设置为最小/最大/预期值 = 0/0/0 并终止计算实例集。头节点保持运行。要终止所有EC2资源并避免EC2收费，请考虑删除集群。

pcluster update

分析配置文件以确定是否可以安全地更新集群。如果分析后确定可以更新集群，系统会提示您确认更改。如果分析结果显示无法更新集群，则会枚举导致冲突的配置设置并显示详细信息。有关更多信息，请参阅 使用 pcluster update。

```
pcluster update [ -h ] [ -c CONFIG_FILE ] [ --force ] [ -r REGION ] [ -nr ]
                [ -nw ] [ -t CLUSTER_TEMPLATE ] [ -p EXTRA_PARAMETERS ] [ -rd ]
                [ --yes ] cluster_name
```

定位参数

cluster_name

指定要更新的集群的名称。

命名的参数

-h, --help

显示 pcluster update 的帮助文本。

-c *CONFIG_FILE*, --config *CONFIG_FILE*

指定要使用的替代配置文件。

默认值为 ~/.parallelcluster/config。

--force

即使一个或多个设置存在阻止更新的更改，或者需要执行尚未执行的操作（例如停止计算实例集）才能继续更新，也会启用更新。此参数不应与 --yes 参数结合使用。

-r *REGION*, --region *REGION*

指定 AWS 区域 要使用的。默认为使用pcluster configure命令 AWS 区域 指定的值。

-nr, --norollback

出错时禁用 AWS CloudFormation 堆栈回滚。

默认值为 False。

-nw, --nowait

指示在处理堆栈命令后不等待堆栈事件。

默认值为 False。

-t *CLUSTER_TEMPLATE*, --cluster-template *CLUSTER_TEMPLATE*

指定集群部分使用的模板。

-p *EXTRA_PARAMETERS*, --extra-parameters *EXTRA_PARAMETERS*

向堆栈更新添加额外的参数。

-rd, --reset-desired

将 Auto Scaling 组的当前容量重置为初始配置值。

默认值为 False。

--yes

自动假定所有提示的回答均为肯定回答。此参数不应与 --force 参数结合使用。

```
$ pcluster update -c path/to/config mycluster
Retrieving configuration from CloudFormation for cluster mycluster...
Validating configuration file .parallelcluster/config...
Found Configuration Changes:
```

```
#    parameter                   old value    new value
---  --------------------------  -----------  -----------
     [compute_resource default]
01   min_count                   1            2
02   max_count                   5            12

Validating configuration update...
Congratulations! The new configuration can be safely applied to your cluster.
Do you want to proceed with the update? - Y/N: Y
Updating: mycluster
Calling update_stack
Status: parallelcluster-mycluster - UPDATE_COMPLETE
```

当命令被调用并开始轮询该调用的状态时，可以安全地使用“Ctrl-C”退出。您可以通过调用 `pcluster status mycluster` 返回以查看当前状态。

pcluster version

显示 AWS ParallelCluster 版本。

```
pcluster version [ -h ]
```

对于命令特定的标记，请运行：`pcluster [command] --help`。

命名的参数

-h, --help

显示 `pcluster version` 的帮助文本。

当命令被调用并开始轮询该调用的状态时，可以安全地使用“Ctrl-C”退出。您可以通过调用 `pcluster status mycluster` 返回以查看当前状态。

```
$ pcluster version
2.11.7
```

pcluster-config

更新 AWS ParallelCluster 配置文件。

```
pcluster-config [ -h ] [convert]
```

对于命令特定的标记，请运行：pcluster-config [command] -h。

命名的参数

-h, --help

显示 pcluster-config 的帮助文本。

> **Note**
>
> 该pcluster-config命令是在 2.9.0 AWS ParallelCluster 版本中添加的。

子命令

pcluster-config convert

```
pcluster-config convert [ -h ] [ -c CONFIG_FILE ] [ -t CLUSTER_TEMPLATE ]
                        [ -o OUTPUT_FILE ]
```

命名的参数

-h, --help

显示 pcluster-config convert 的帮助文本。

-c *CONFIG_FILE*, --config-file *CONFIG_FILE*

指定要读取的配置文件的路径。

默认值为 ~/.parallelcluster/config。

有关更多信息，请参阅 配置 AWS ParallelCluster。

-t *CLUSTER_TEMPLATE*, --cluster-template *CLUSTER_TEMPLATE*

指示要使用的 [cluster] 部分。如果未指定此参数，则 pcluster-config convert 将使用 [global] 部分中的 cluster_template 设置。如果未指定该部分，则使用 [cluster default] 部分。

-o *OUTPUT_FILE*, --output *OUTPUT_FILE*

指定要写入的已转换配置文件的路径。默认情况下，输出将写入 STDOUT。

例如：

```
$ pcluster-config convert -t alpha -o ~/.parallelcluster/multiinstance
```

转换 ~/.parallelcluster/config 的 [cluster alpha] 部分中指定的集群配置，将已转换配置文件写入 ~/.parallelcluster/multiinstance。

配置

默认情况下，AWS ParallelCluster 为所有配置参数使用 `~/.parallelcluster/config` 文件。您可以使用 `-c` 或 `--config` 命令行选项或 `AWS_PCLUSTER_CONFIG_FILE` 环境变量指定自定义配置文件。

一个示例配置文件随 AWS ParallelCluster 安装在 Python 目录 `site-packages/aws-parallelcluster/examples/config` 中。GitHub 上也提供了该示例配置文件，网址为：https://github.com/aws/aws-parallelcluster/blob/v2.11.9/cli/src/pcluster/examples/config。

当前 AWS ParallelCluster 2 版本：2.11.9。

主题

布局

在多个部分中定义 AWS ParallelCluster 配置。

以下部分是必需的：[global] 部分和 [aws] 部分。

您还必须包含至少一个 [cluster] 部分和一个 [vpc] 部分。

一个部分以方括号中的部分名称开头，后跟参数和配置。

```
[global]
cluster_template = default
update_check = true
sanity_check = true
```

[global] 部分

指定与 pcluster 相关的全局配置选项。

```
[global]
```

主题

- cluster_template
- update_check
- sanity_check

cluster_template

定义默认用于集群的 cluster 部分的名称。有关 cluster 部分的更多信息，请参阅 [cluster] 部分。集群名称必须以字母开头，不能超过 60 个字符，并且只能包含字母、数字和连字符 (-)。

例如，以下设置指定默认情况下使用 [cluster default] 开始的部分。

```
cluster_template = default
```

更新策略：在更新期间不分析此设置。

update_check

（可选）检查 pcluster 的更新。

默认值为 true。

```
update_check = true
```

更新策略：在更新期间不分析此设置。

sanity_check

（可选）尝试验证集群参数中定义的资源的配置。

默认值为 true。

> **Warning**
>
> 如果 sanity_check 设置为 false，则跳过重要的检查。这可能会导致您的配置无法正常运行。

```
sanity_check = true
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.5.0 之前，sanity_check 默认为 false。

更新策略：在更新期间不分析此设置。

[aws] 部分

（可选）用于选择 AWS 区域。

创建集群时按以下优先级顺序为新集群选择 AWS 区域：

1. pcluster create 的 -r 或 --region 参数。
2. AWS_DEFAULT_REGION 环境变量。

3. AWS ParallelCluster 配置文件（默认位置为 `~/.parallelcluster/config`）的 `[aws]` 部分中的 `aws_region_name` 设置。这是由 `pcluster configure` 命令更新的位置。
4. AWS CLI 配置文件 (`~/.aws/config`) 的 `[default]` 部分中的 `region` 设置。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.0 之前，这些设置是必需的，并且适用于所有集群。

要存储凭证，您可以使用环境、Amazon EC2 的 IAM 角色或 AWS CLI，而不是将凭证保存到 AWS ParallelCluster 配置文件中。

```
[aws]
aws_region_name = Region
```

更新策略：在更新期间不分析此设置。

`[aliases]` 部分

指定别名，并使您能够自定义 `ssh` 命令。

请注意以下默认设置：

- `CFN_USER` 设置为操作系统的默认用户名
- `MASTER_IP` 设置为头节点的 IP 地址
- `ARGS` 设置为用户在 *`pcluster ssh cluster_name`* 之后提供的任何参数

```
[aliases]
# This is the aliases section, you can configure
# ssh alias here
ssh = ssh {CFN_USER}@{MASTER_IP} {ARGS}
```

更新策略：在更新期间不分析此设置。

`[cluster]` 部分

定义可用于创建集群的集群模板。配置文件可以包含多个 `[cluster]` 部分。

可以使用同一个集群模板创建多个集群。

格式为 [cluster *cluster-template-name*]。默认情况下，使用由 [global] 部分中的 cluster_template 设置命名的 [cluster] 部分，但可以在 pcluster 命令行上覆盖。

cluster-template-name 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[cluster default]
```

主题

- additional_cfn_template
- additional_iam_policies
- base_os
- cluster_resource_bucket
- cluster_type
- compute_instance_type
- compute_root_volume_size
- custom_ami
- cw_log_settings
- dashboard_settings
- dcv_settings
- desired_vcpus
- disable_cluster_dns
- disable_hyperthreading
- ebs_settings
- ec2_iam_role
- efs_settings
- enable_efa
- enable_efa_gdr
- enable_intel_hpc_platform

- encrypted_ephemeral
- ephemeral_dir
- extra_json
- fsx_settings
- iam_lambda_role
- initial_queue_size
- key_name
- maintain_initial_size
- master_instance_type
- master_root_volume_size
- max_queue_size
- max_vcpus
- min_vcpus
- placement
- placement_group
- post_install
- post_install_args
- pre_install
- pre_install_args
- proxy_server
- queue_settings
- raid_settings
- s3_read_resource
- s3_read_write_resource
- scaling_settings
- scheduler
- shared_dir
- spot_bid_percentage
- spot_price

- tags
- template_url
- vpc_settings

additional_cfn_template

（可选）定义要与集群一起启动的附加 AWS CloudFormation 模板。此附加模板用于创建存在于集群外部但属于集群生命周期一部分的资源。

该值必须是公共模板HTTPURL的值，并提供所有参数。

没有默认值。

```
additional_cfn_template = https://<bucket-name>.s3.amazonaws.com/my-cfn-template.yaml
```

更新策略：如果更改此设置，则不允许更新。

additional_iam_policies

（可选）为亚马逊指定IAM政策的亚马逊资源名称列表 (ARNs) EC2。除了 AWS ParallelCluster 所需的权限（以逗号分隔）之外，此列表也附加到集群中使用的根角色。IAM策略名称和策略名称ARN是不同的。名称不能用作 additional_iam_policies 的参数。

如果您打算在集群节点的默认设置中添加额外的策略，我们建议您将其他自定义IAM策略与additional_iam_policies设置一起传递，而不是使用ec2_iam_role设置来添加您的特定EC2策略。这是因为additional_iam_policies已添加到 AWS ParallelCluster 所需的默认权限中。现有 ec2_iam_role 必须包含所需的所有权限。但是，随着功能的添加，不同版本之间所需的权限通常会有所不同，因此现有的 ec2_iam_role 可能会过时。

没有默认值。

```
additional_iam_policies = arn:aws:iam::123456789012:policy/CustomEC2Policy
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.5.0 中添加了对 additional_iam_policies 的支持。

更新策略：可以在更新期间更改此设置。

base_os

（必需）指定在集群中使用的操作系统类型。

可用的选项为：

- alinux2
- centos7
- ubuntu1804
- ubuntu2004

> **Note**
>
> 对于 AWS 基于 Graviton 的实例，仅alinux2支持ubuntu1804、或ubuntu2004。

> **Note**
>
> 在 2.11.4 AWS ParallelCluster 版本中删除centos8了对 Support 的支持。在 AWS ParallelCluster 版本 2.11.0 中添加了对 ubuntu2004 的支持，并删除了对 alinux 和 ubuntu1604 的支持。在 2.10.0 AWS ParallelCluster 版本中添加centos8centos6了对的支持，并删除了对的支持。在 AWS ParallelCluster 版本 2.6.0 中添加了对 alinux2 的支持。在 AWS ParallelCluster 版本 2.5.0 中添加了对 ubuntu1804 的支持，并删除了对 ubuntu1404 的支持。

除下表中 AWS 区域 提到的具体内容外，其他不支持centos7。所有其他 AWS 商业区域都支持以下所有操作系统。

分区（AWS 区域）	**alinux2**	**centos7**	**ubuntu1804** 和 **ubuntu2004**
商业（均 AWS 区域 未特别提及）	True	True	True

分区（AWS 区域）	**alinux2**	**centos7**	**ubuntu1804 和 ubuntu2004**
AWS GovCloud （美国东部）(us-gov-east-1)	True	False	True
AWS GovCloud （美国西部）(us-gov-west-1)	True	False	True
中国（北京）(cn-north-1)	True	False	True
中国（宁夏）(cn-northwest-1)	True	False	True

> **Note**
>
> base_os 参数还确定用于登录集群的用户名。

- centos7: centos
- ubuntu1804 和 ubuntu2004：ubuntu
- alinux2: ec2-user

> **Note**
>
> 在 2.7.0 AWS ParallelCluster 版本之前，该base_os参数是可选的，默认值为。alinux从 AWS ParallelCluster 版本 2.7.0 开始，base_os 参数是必需的。

> **Note**
>
> 如果 scheduler 参数为 awsbatch，则仅支持 alinux2。

```
base_os = alinux2
```

更新策略：如果更改此设置，则不允许更新。

cluster_resource_bucket

（可选）指定用于托管创建集群时生成的资源的 Amazon S3 存储桶的名称。桶必须启用版本控制。有关更多信息，请参阅 Amazon Simple Storage Service 用户指南 中的使用版本控制。此存储桶可用于多个集群。桶和集群必须位于同一区域中。

如果未指定此参数，则在创建集群时会创建新桶。新桶的名称为 parallelcluster-*random_string*。用这个名字，*random_string* 是一个由字母数字字符组成的随机字符串。所有集群资源都存储在此存储桶中，路径为：*bucket_name*/*resource_directory* resource_directory有表格*stack_name*-*random_string*，在哪里 *stack_name* 是使用的其中一个 AWS CloudFormation 堆栈的名称。 AWS ParallelCluster的价值 *bucket_name* 可以在parallelcluster-*clustername*堆栈输出的ResourcesS3Bucket值中找到。的价值 *resource_directory* 可以在同一堆栈的ArtifactS3RootDirectory输出值中找到。

默认值为 parallelcluster-*random_string*。

```
cluster_resource_bucket = amzn-s3-demo-bucket
```

> **Note**
>
> 在 2.10.0 AWS ParallelCluster 版本中添加cluster_resource_bucket了对 Support 的支持。

更新策略：如果更改此设置，则不允许更新。不能强制更新此设置。

cluster_type

（可选）定义要启动的集群的类型。如果定义了 queue_settings 设置，则必须在 [queue] 部分中将此设置替换为 compute_type 设置。

有效的选项为：ondemand 和 spot。

默认值为 ondemand。

有关竞价型实例的更多信息，请参阅使用竞价型实例。

> **Note**
>
> 使用竞价型实例要求您的账户中存在 AWSServiceRoleForEC2Spot 服务相关角色。要使用在您的账户中创建此角色 AWS CLI，请运行以下命令：
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> 有关更多信息，请参阅 Amazon EC2 用户指南中的竞价型实例请求的服务相关角色。

```
cluster_type = ondemand
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

compute_instance_type

（可选）定义用于集群计算节点的 Amazon EC2 实例类型。该实例类型的架构必须与用于 master_instance_type 设置的架构相同。如果定义了 queue_settings 设置，则必须在 [compute_resource] 部分中将此设置替换为 instance_type 设置。

如果您使用的是awsbatch调度程序，请参阅 AWS Batch 用户界面中创建的计算环境以获取支持的实例类型列表。

默认值为 t2.micro；当调度器为 awsbatch 时，为 optimal。

```
compute_instance_type = t2.micro
```

> **Note**
>
> 2.8.0 版本中增加了对 AWS 基于 Graviton 的C6g实例（包括A1和实例）的 AWS ParallelCluster 支持。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

compute_root_volume_size

（可选）以千兆字节 (GiB) 为单位指定 ComputeFleet 根卷大小。AMI必须支持growroot。

默认值为 35。

> **Note**
>
> 对于 2.5.0 和 2.10.4 之间的 AWS ParallelCluster 版本，默认值为 25。在 2.5.0 AWS ParallelCluster 版本之前，默认值为 20。

```
compute_root_volume_size = 35
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

custom_ami

（可选）指定用于头部和计算节点的自定义AMI的 ID，而不是默认发布的 ID AMIs。有关更多信息，请参阅修改 AMI 或构建自定义 AWS ParallelCluster AMI。

没有默认值。

```
custom_ami = ami-00d4efc81188687a0
```

如果自定义AMI需要额外的权限才能启动，则必须将这些权限添加到用户和头节点策略中。

例如，如果自定义的快照与其关联AMI了加密快照，则用户和头节点策略中都需要以下其他策略：

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
           "Effect": "Allow",
           "Action": [
               "kms:DescribeKey",
               "kms:ReEncrypt*",
               "kms:CreateGrant",
               "kms:Decrypt"
           ],
           "Resource": [
               "arn:aws:kms:<AWS_REGION>:<AWS_ACCOUNT_ID>:key/<AWS_KMS_KEY_ID>"
           ]
       }
```

```
    ]
}
```

更新策略：如果更改此设置，则不允许更新。

cw_log_settings

（可选）使用 CloudWatch 日志配置标识该[cw_log]部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [cw_log] 部分、亚马逊 CloudWatch 控制面板 和 与 Amazon CloudWatch 日志集成。

例如，以下设置指定启动部分[cw_log custom-cw]用于 CloudWatch 日志配置。

```
cw_log_settings = custom-cw
```

> **Note**
>
> 在 2.6.0 AWS ParallelCluster 版本中添加cw_log_settings了对 Support 的支持。

更新策略：如果更改此设置，则不允许更新。

dashboard_settings

（可选）使用 CloudWatch仪表板配置标识该[dashboard]部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [dashboard] 部分。

例如，以下设置指定启动部分[dashboard custom-dashboard用于 CloudWatch 仪表板配置。

```
dashboard_settings = custom-dashboard
```

> **Note**
>
> 在 2.10.0 AWS ParallelCluster 版本中添加dashboard_settings了对 Support 的支持。

更新策略：可以在更新期间更改此设置。

dcv_settings

（可选）使用 Amazon DCV 配置标识该[dcv]部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [dcv] 部分。

例如，以下设置指定开始部分用[dcv custom-dcv]于 Amazon DCV 配置。

```
dcv_settings = custom-dcv
```

> **Note**
>
> 在 AWS 基于 Graviton 的实例上，DCV仅支持 Amazon。alinux2

> **Note**
>
> 在 2.5.0 AWS ParallelCluster 版本中添加dcv_settings了对 Support 的支持。

更新策略：如果更改此设置，则不允许更新。

desired_vcpus

（可选）在计算环境 vCPUs 中指定所需的数量。仅在调度器为 awsbatch 时使用。

默认值为 4。

```
desired_vcpus = 4
```

更新策略：在更新期间不分析此设置。

disable_cluster_dns

（可选）指定是否不应为集群创建DNS条目。默认情况下， AWS ParallelCluster 会创建 Route 53 托管区域。如果 disable_cluster_dns 设置为 true，则不会创建托管区。

默认值为 false。

```
disable_cluster_dns = true
```

> **Warning**
>
> 集群需要名称解析系统才能正常运行。如果 disable_cluster_dns 设置为 true，则还必须提供其他名称解析系统。

> **Important**
>
> 只有在指定了 queue_settings 设置时才支持 disable_cluster_dns = true。

> **Note**
>
> 在 2.9.1 AWS ParallelCluster 版本中增加了对 Support 的支持。disable_cluster_dns

更新策略：如果更改此设置，则不允许更新。

disable_hyperthreading

（可选）禁用头节点和计算节点上的超线程。并非所有实例类型都可以禁用超线程。有关支持禁用超线程的实例类型列表，请参阅 A mazon EC2 用户指南中每种实例类型的CPU内CPU核和线程。如果定义了 queue_settings 设置，则可以定义此设置，也可以定义 [queue] 部分中的 disable_hyperthreading 设置。

默认值为 false。

```
disable_hyperthreading = true
```

> **Note**
>
> disable_hyperthreading 仅在 scheduler = awsbatch 时影响头节点。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.5.0 中添加了对 disable_hyperthreading 的支持。

更新策略：如果更改此设置，则不允许更新。

ebs_settings

（可选）标识装载在头节点上的 Amazon EBS 卷的[ebs]部分。使用多个 Amazon EBS 卷时，请在列表中输入这些参数，每个参数用逗号分隔。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

最多支持五 (5) 个额外的 Amazon EBS 卷。

有关更多信息，请参阅 [ebs] 部分。

例如，以下设置指定 Amazon 卷的开头部分[ebs custom1]和用[ebs custom2]于 Amazon EBS 卷的部分。

```
ebs_settings = custom1, custom2
```

更新策略：如果更改此设置，则不允许更新。

ec2_iam_role

（可选）定义附加到集群中所有实例EC2的 Amazon 现有IAM角色的名称。IAM角色名称和其 Amazon 资源名称 (ARN) 不同。ARNs不能用作论据ec2_iam_role。

如果指定了此选项，则忽略 additional_iam_policies 设置。如果您打算在集群节点的默认设置中添加额外的策略，我们建议您将其他自定义IAM策略与additional_iam_policies设置一起传递，而不是使用ec2_iam_role设置。

如果未指定此选项，则使用 Amazon EC2 的默认 AWS ParallelCluster IAM角色。有关更多信息，请参阅 AWS Identity and Access Management 中的角色 AWS ParallelCluster。

没有默认值。

```
ec2_iam_role = ParallelClusterInstanceRole
```

更新策略：如果更改此设置，则不允许更新。

efs_settings

（可选）指定与 Amazon EFS 文件系统相关的设置。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [efs] 部分。

例如，以下设置指定启动部分用[efs customfs]于 Amazon EFS 文件系统配置。

```
efs_settings = customfs
```

更新策略：如果更改此设置，则不允许更新。

enable_efa

（可选）如果存在，则指定为计算节点启用 Elastic Fabric Adapter (EFA)。要查看支持的EC2实例列表EFA，请参阅 Amazon Linux 实例EC2用户指南中的支持的实例类型。有关更多信息，请参阅 Elastic Fabric Adapter。如果定义了 queue_settings 设置，则可以定义此设置，也可以定义 [queue] 部分中的 enable_efa 设置。应使用集群置放群组来最大限度地减少实例之间的延迟。有关更多信息，请参阅placement 和placement_group。

```
enable_efa = compute
```

> **Note**
>
> 2.10.1 版本中增加了EFA对基于 ARM 的 Graviton2 实例的支持。 AWS ParallelCluster

更新策略：如果更改此设置，则不允许更新。

enable_efa_gdr

（可选）从 2.11.3 AWS ParallelCluster 版本开始，此设置无效。如果实例类型和操作系统都支持 Elastic Fabric Adapter GPUDirect RDMA ()，则始终启用对（远程直接内存访问）的支持。EFA

> **Note**
>
> AWS ParallelCluster 版本 2.10.0 到 2.11.2：如果compute，则指定为计算节点启用 Elastic Fabric Adapter GPUDirect RDMA (EFA) 支持（远程直接内存访问）。

将此设置设置为 compute 需要先将 enable_efa 设置设为 compute。EFA特定GPUDirectRDMA操作系统上的特定实例类型 (p4d.24xlarge) 支持对的支持（base_os是alinux2centos7、ubuntu1804、或ubuntu2004）。如果定义了 queue_settings 设置，则可以定义此设置，也可以定义 [queue] 部分中的 enable_efa_gdr 设置。应使用集群置放群组来最大限度地减少实例之间的延迟。有关更多信息，请参阅placement 和placement_group。

```
enable_efa_gdr = compute
```

> **Note**
>
> 在 2.10.0 AWS ParallelCluster 版本中添加enable_efa_gdr了对 Support 的支持。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

enable_intel_hpc_platform

（可选）如果存在，则表示接受 Intel Parallel Studio 的最终用户许可协议。这将导致 Intel Parallel Studio 安装在头节点上并与计算节点共享。这使头节点进行引导的时间增加了几分钟。仅支持该enable_intel_hpc_platform设置 CentOS 7 (base_os = centos7)。

默认值为 false。

```
enable_intel_hpc_platform = true
```

> **Note**
>
> 该enable_intel_hpc_platform参数与基于 AWS Graviton 的实例不兼容。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.5.0 中添加了对 enable_intel_hpc_platform 的支持。

更新策略：如果更改此设置，则不允许更新。

encrypted_ephemeral

（可选）使用（Linux Unified Key Setup）使用LUKS不可恢复的内存密钥加密临时实例存储卷。

有关更多信息，请参阅 https://gitlab.com/cryptsetup/cryptsetup/blob/master/README.md。

默认值为 false。

```
encrypted_ephemeral = true
```

更新策略：如果更改此设置，则不允许更新。

ephemeral_dir

（可选）定义实例存储卷（如果使用）的挂载路径。

默认值为 /scratch。

```
ephemeral_dir = /scratch
```

更新策略：如果更改此设置，则不允许更新。

extra_json

（可选）定义合并到JSON的额外内容 Chef dna.json（有关更多信息，请参阅 构建自定义 AWS ParallelCluster AMI。）

默认值为 {}。

```
extra_json = {}
```

> **Note**
>
> 从 AWS ParallelCluster 版本 2.6.1 开始，启动节点时默认会跳过大多数安装配方，以缩短启动时间。要以牺牲启动时间为代价运行所有安装食谱以获得更好的向后兼容性，请将 "skip_install_recipes" : "no" 添加到 extra_json 设置中的 cluster 键。例如：
>
> ```
> extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
> ```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

fsx_settings

（可选）指定定义 for Lustre 配置FSx的部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [fsx] 部分。

例如，以下设置指定启动部分用[fsx fs]于 for Lustre 配置。FSx

```
fsx_settings = fs
```

更新策略：如果更改此设置，则不允许更新。

iam_lambda_role

（可选）定义现有 AWS Lambda 执行角色的名称。此角色附加到集群中所有 Lambda 函数。有关更多信息，请参阅AWS Lambda 开发人员指南 中的 AWS Lambda 执行角色。

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

IAM角色名称和其 Amazon 资源名称 (ARN) 不同。ARNs不能用作论据iam_lambda_role。如果同时定义了 ec2_iam_role 和 iam_lambda_role，并且 scheduler 是 sge、slurm 或 torque，则不会创建任何角色。如果 scheduler 是 awsbatch，则在 pcluster start 期间将创建角色。有关示例策略，请参阅 ParallelClusterLambdaPolicy使用 SGE, Slurm，或 Torque 和 使用 awsbatch 的 ParallelClusterLambdaPolicy。

没有默认值。

```
iam_lambda_role = ParallelClusterLambdaRole
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本中增加了对 Support 的支持。iam_lambda_role

更新策略：可以在更新期间更改此设置。

initial_queue_size

（可选）设置在集群中作为计算节点启动的 Amazon EC2 实例的初始数量。如果定义了 queue_settings 设置，则必须在 [compute_resource] 部分中删除此设置并替换为 initial_count 设置。

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

此设置仅适用于传统调度器 (SGE, Slurm，以及 Torque)。 如果maintain_initial_size设置为true，则该initial_queue_size设置必须至少为一 (1)。

如果调度器是 awsbatch，请改用 min_vcpus。

默认值为 2。

```
initial_queue_size = 2
```

更新策略：可以在更新期间更改此设置。

key_name

（可选）命名用于SSH访问实例的现有 Amazon EC2 密钥对。

```
key_name = mykey
```

> **Note**
>
> 在 2.11.0 AWS ParallelCluster 版本之前，key_name是必需的设置。

更新策略：如果更改此设置，则不允许更新。

maintain_initial_size

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

（可选）保持传统调度器的 Auto Scaling 组的初始大小（SGE, Slurm，以及 Torque).

如果调度器是 awsbatch，请改用 desired_vcpus。

此设置是一个布尔标记。如果设置为 true，则自动扩缩组的成员数永远不会少于 initial_queue_size 的值，并且 initial_queue_size 的值必须为一 (1) 或更大。集群仍可以扩展到 max_queue_size 的值。如果为 cluster_type = spot，则自动扩缩组的实例可能会中断，并且大小可能降至 initial_queue_size 以下。

如果设置为 false，则自动扩缩组的成员数可以缩减为零 (0)，以防止在不需要资源时闲置。

如果定义了 queue_settings 设置，则必须在 [compute_resource] 部分中删除此设置并替换为 initial_count 和 min_count 设置。

默认值为 false。

```
maintain_initial_size = false
```

更新策略：可以在更新期间更改此设置。

master_instance_type

（可选）定义用于头节点的 Amazon EC2 实例类型。该实例类型的架构必须与用于 compute_instance_type 设置的架构相同。

如果有免费套餐，则默认为免费套餐实例类型（t2.micro或t3.micro）。 AWS 区域 其中 AWS 区域 没有免费套餐，默认为t3.micro。有关 AWS 免费套餐的更多信息，请参阅AWS 免费套餐 FAQs。

```
master_instance_type = t2.micro
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本之前，全部默认为。`t2.micro` AWS 区域在 AWS ParallelCluster 版本2.10.0 中，头节点不支持 `p4d.24xlarge`。2.8.0 版本中增加了对 AWS 基于 Graviton 的实例（例如`A1`和`C6g`）的 AWS ParallelCluster 支持。

更新策略：如果更改此设置，则不允许更新。

master_root_volume_size

（可选）指定头节点根卷大小，以吉字节 (GiB) 为单位。AMI必须支持`growroot`。

默认值为 35。

> **Note**
>
> 对于 2.5.0 和 2.10.4 之间的 AWS ParallelCluster 版本，默认值为 25。在 2.5.0 AWS ParallelCluster 版本之前，默认值为 20。

```
master_root_volume_size = 35
```

更新策略：如果更改此设置，则不允许更新。

max_queue_size

（可选）设置可在集群中启动的 Amazon EC2 实例的最大数量。如果定义了 queue_settings 设置，则必须在 [compute_resource] 部分中删除此设置并替换为 max_count 设置。

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

此设置仅适用于传统调度器 (SGE, Slurm，以及 Torque).

如果调度器是 awsbatch，请改用 max_vcpus。

默认值为 10。

```
max_queue_size = 10
```

更新策略：可以在更新期间更改此设置，但如果该值降低，则应停止计算实例集。否则，现有节点可能会被终止。

max_vcpus

（可选）指定计算环境 vCPUs 中的最大数量。仅在调度器为 awsbatch 时使用。

默认值为 20。

```
max_vcpus = 20
```

更新策略：更新期间不能减小此设置。

min_vcpus

（可选）为 awsbatch 调度器保持自动扩缩组的初始大小。

> **Note**
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

如果调度器是 SGE, Slurm，或 Torque，maintain_initial_size改用。

计算环境中的成员数绝不会少于 min_vcpus 的值。

默认值为 0。

```
min_vcpus = 0
```

更新策略：可以在更新期间更改此设置。

placement

（可选）定义集群置放群组逻辑，并使整个集群或仅计算实例能够使用集群置放群组。

如果定义了 queue_settings 设置，则对每个 [queue] 部分，应删除此设置并替换为 placement_group 设置。如果将同一个置放群组用于不同的实例类型，则请求更有可能因容量不足错误而失败。有关更多信息，请参阅 Amazon EC2 用户指南中的实例容量不足。只有事先创建了置放群组并在每个队列的 placement_group 设置中进行了配置，多个队列才能共享该置放群组。如果每个 [queue] 部分都定义了 placement_group 设置，则头节点不能位于队列的置放群组中。

有效选项是 cluster 或 compute。

当调度器为 awsbatch 时，不使用此参数。

默认值为 compute。

```
placement = compute
```

更新策略：如果更改此设置，则不允许更新。

placement_group

（可选）定义集群置放群组。如果定义了 queue_settings 设置，则应在 [queue] 部分中删除此设置并替换为 placement_group 设置。

有效选项为以下值：

- DYNAMIC
- 现有的 Amazon EC2 集群置放群组名称

当设置为 DYNAMIC 时，将唯一置放群组作为集群堆栈的一部分进行创建和删除。

当调度器为 awsbatch 时，不使用此参数。

有关置放群组的更多信息，请参阅 Amazon EC2 用户指南中的置放群组。如果将同一个置放群组用于不同的实例类型，则请求更有可能因容量不足错误而失败。有关更多信息，请参阅 Amazon EC2 用户指南中的实例容量不足。

没有默认值。

并非所有实例类型都支持集群置放群组。例如，t3.micro 的默认实例类型不支持集群置放群组。有关支持集群置放群组的实例类型列表的信息，请参阅 Amazon EC2 用户指南中的集群置放群组规则和限制。有关使用置放群组时的提示，请参阅置放群组和实例启动问题。

```
placement_group = DYNAMIC
```

更新策略：如果更改此设置，则不允许更新。

post_install

（可选）指定在URL所有节点引导操作完成后运行的安装后脚本。有关更多信息，请参阅 自定义引导操作。

当使用 awsbatch 作为调度器时，安装后脚本仅在头节点上运行。

参数格式可以是 http://*hostname/path/to/script.sh* 或 s3://*bucketname/path/to/script.sh*。

没有默认值。

```
post_install = s3://<bucket-name>/my-post-install-script.sh
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

post_install_args

（可选）指定要传递到安装后脚本的用双引号引起的参数列表。

没有默认值。

```
post_install_args = "argument-1 argument-2"
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

pre_install

（可选）指定在URL启动任何节点部署引导操作之前运行的预安装脚本。有关更多信息，请参阅 自定义引导操作。

当使用 awsbatch 作为调度器时，预安装脚本仅在头节点上运行。

参数格式可以是 http://*hostname/path/to/script.sh* 或 s3://*bucketname/path/to/script.sh*。

没有默认值。

```
pre_install = s3://<bucket-name>/my-pre-install-script.sh
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

pre_install_args

（可选）指定要传递到预安装脚本的用双引号引起的参数列表。

没有默认值。

```
pre_install_args = "argument-3 argument-4"
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

proxy_server

（可选）通常定义HTTP或HTTPS代理服务器http://*x.x.x.x:8080*。

没有默认值。

```
proxy_server = http://10.11.12.13:8080
```

更新策略：如果更改此设置，则不允许更新。

queue_settings

（可选）指定集群使用队列而不是同构计算实例集，以及使用的 [queue] 部分。列出的第一个 [queue] 部分是默认的调度器队列。queue 部分名称必须以小写字母开头，不能超过 30 个字符，并且只能包含小写字母、数字和连字符 (-)。

> **Important**
>
> 仅在 scheduler 设置为 slurm 时支持 queue_settings。不得指定 cluster_type、compute_instance_type、initial_queue_size、maintain_initial_siz 和 spot_price 设置。disable_hyperthreading 和 enable_efa 设置既可以在 [cluster] 部分中指定，也可以在 [queue] 部分中指定，但不能同时在这些部分中指定。

最多支持五 (5) 个 [queue] 部分。

有关更多信息，请参阅 [queue] 部分。

例如，以下设置指定使用以 [queue q1] 和 [queue q2] 开始的部分。

```
queue_settings = q1, q2
```

> **Note**
>
> 在 2.9.0 queue_settings AWS ParallelCluster 版本中增加了对 Support 的支持。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

raid_settings

（可选）使用 Amazon EBS 卷RAID配置标识该[raid]部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [raid] 部分。

例如，以下设置指定将以 [raid rs] 开始的部分用于自动扩缩配置。

```
raid_settings = rs
```

更新策略：如果更改此设置，则不允许更新。

s3_read_resource

（可选）指定向 AWS ParallelCluster 节点授予只读访问权限的 Amazon S3 资源。

例如，arn:aws:s3:::*my_corporate_bucket**提供对的只读访问权限 *my_corporate_bucket* 存储桶和存储桶中的对象。

有关格式的详细信息，请参阅使用 Amazon S3。

没有默认值。

```
s3_read_resource = arn:aws:s3:::my_corporate_bucket*
```

更新策略：可以在更新期间更改此设置。

s3_read_write_resource

（可选）指定将向 AWS ParallelCluster 节点授予其读/写访问权限的 Amazon S3 资源。

例如，arn:aws:s3:::*my_corporate_bucket*/Development/*提供对Development文件夹中所有对象的读/写访问权限 *my_corporate_bucket* 桶。

有关格式的详细信息，请参阅使用 Amazon S3。

没有默认值。

```
s3_read_write_resource = arn:aws:s3:::my_corporate_bucket/*
```

更新策略：可以在更新期间更改此设置。

scaling_settings

使用自动扩缩配置标识 [scaling] 部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [scaling] 部分。

例如，以下设置指定将以 [scaling custom] 开始的部分用于自动扩缩配置。

```
scaling_settings = custom
```

更新策略：如果更改此设置，则不允许更新。

scheduler

（必需）定义集群调度器。

有效选项为以下值：

awsbatch

AWS Batch

有关 awsbatch 调度器的更多信息，请参阅联网设置和 AWS Batch (awsbatch)。

`sge`

> **Note**
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

Son of Grid Engine (SGE)

`slurm`

Slurm Workload Manager (Slurm)

`torque`

> **Note**
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

Torque Resource Manager (Torque)

> **Note**
> 在 2.7.0 AWS ParallelCluster 版本之前，该`scheduler`参数是可选的，默认值为。`sge`从 2.7.0 AWS ParallelCluster 版开始，该`scheduler`参数为必填项。

```
scheduler = slurm
```

更新策略：如果更改此设置，则不允许更新。

shared_dir

（可选）定义共享 Amazon EBS 卷的挂载路径。

请勿将此选项用于多个 Amazon EBS 卷。相反，在每个 [ebs] 部分下提供 shared_dir 值。

有关使用多个 Amazon EBS 卷的详细信息，请参阅[ebs]部分。

默认值为 /shared。

以下示例显示了安装在的共享 Amazon EBS 卷/myshared。

```
shared_dir = myshared
```

更新策略：如果更改此设置，则不允许更新。

spot_bid_percentage

（可选）设置按需百分比，用于计算计划程序的最高竞价价格。 ComputeFleet awsbatch

如果未指定，则选择当前 Spot 市场价格，最高为按需价格。

```
spot_bid_percentage = 85
```

更新策略：可以在更新期间更改此设置。

spot_price

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

（可选）为传统调度程序设置最高竞价价格（ComputeFleet SGE, Slurm，以及 Torque)。 仅在cluster_type设置设置为时使用spot。如果您不指定值，则按 Spot 价格进行收费，最高为按需价格。如果定义了 queue_settings 设置，则必须在 [compute_resource] 部分中删除此设置并替换为 spot_price 设置。

如果计划程序为 awsbatch，请改用 spot_bid_percentage。

有关查找满足您需求的竞价型实例的帮助，请参阅竞价型实例顾问。

```
spot_price = 1.50
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.5.0 中，如果spot_price未指定cluster_type = spot但未指定，则实例启动 ComputeFleet 失败。此问题已在 2.5.1 AWS ParallelCluster 版本中修复。

更新策略：可以在更新期间更改此设置。

tags

（可选）定义要使用的标签 AWS CloudFormation。

如果通过 --tags 指定了命令行标签，则它们将与配置标签合并。

命令行标签覆盖具有相同键的配置标签。

标签已JSON格式化。请勿在大括号外使用引号。

有关更多信息，请参阅 AWS CloudFormation 用户指南 中的AWS CloudFormation 资源标签类型。

```
tags = {"key" : "value", "key2" : "value2"}
```

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 更新策略不支持更改 AWS ParallelCluster 版本 2.8.0 到版本 2.9.1 的 `tags` 设置。
> 对于版本 2.10.0 到版本 2.11.7，列出的支持更改 `tags` 设置的更新策略不准确。不支持修改此设置时进行集群更新。

template_url

（可选）定义用于创建集群的 AWS CloudFormation 模板的路径。

更新使用最初用于创建堆栈的模板。

默认值为 `https://`*`aws_region_name`*`-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-`*`version`*`.cfn.json`。

> **Warning**
>
> 这是一个高级参数。对此设置进行任何更改需自行承担风险。

```
template_url = https://us-east-1-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-2.11.9.cfn.json
```

更新策略：在更新期间不分析此设置。

vpc_settings

（必需）使用部署集群的 Amazon VPC 配置标识[vpc]部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

有关更多信息，请参阅 [vpc] 部分。

例如，以下设置指定开始部分用[vpc public]于 Amazon VPC 配置。

```
vpc_settings = public
```

更新策略：如果更改此设置，则不允许更新。

[compute_resource] 部分

定义计算资源的配置设置。[compute_resource] 部分由 [queue] 部分中的 compute_resource_settings 设置引用。只有当 scheduler 设置为 slurm 时，才支持 [compute_resource] 部分。

格式为 [compute_resource *<compute-resource-name>*]。*compute-resource-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[compute_resource cr1]
instance_type = c5.xlarge
min_count = 0
initial_count = 2
max_count = 10
spot_price = 0.5
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.9.0 中添加了对 [compute_resource] 部分的支持。

主题

- initial_count
- instance_type

- max_count
- min_count
- spot_price

initial_count

（可选）设置要为此计算资源启动的 Amazon EC2 实例的初始数量。至少在计算资源中启动了这么多节点之后，集群创建才会完成。如果队列的 compute_type 设置为 spot，并且没有足够的竞价型实例可用，则集群创建可能会超时并失败。任何大于 min_count 设置的计数均为受 scaledown_idletime 设置控制的动态容量。该设置替代 initial_queue_size 设置。

默认值为 0。

```
initial_count = 2
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

instance_type

（必需）定义用于此计算资源的 Amazon EC2 实例类型。该实例类型的架构必须与用于 master_instance_type 设置的架构相同。对于 [queue] 部分引用的每个 [compute_resource] 部分，instance_type 设置必须是唯一的。该设置替代 compute_instance_type 设置。

```
instance_type = t2.micro
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

max_count

（可选）设置此计算资源中可以启动的 Amazon EC2 实例的最大数量。任何大于 initial_count 设置的计数都将在关闭模式下启动。该设置替代 max_queue_size 设置。

默认值为 10。

```
max_count = 10
```

更新策略：要将队列大小减至当前节点数以下，需要先停止计算实例集。

> **Note**
>
> 对于 AWS ParallelCluster 版本 2.0.0 至版本 2.9.1，在停止计算实例集之前，更新策略不支持更改 `max_count` 设置。

`min_count`

（可选）设置此计算资源中可以启动的 Amazon EC2 实例的最小数量。这些节点均为静态容量。至少在计算资源中启动了此数量的节点之后，集群创建才会完成。

默认值为 `0`。

```
min_count = 1
```

更新策略：减少队列中静态节点的数量需要先停止计算实例集。

> **Note**
>
> 对于 AWS ParallelCluster 版本 2.0.0 至版本 2.9.1，在停止计算实例集之前，更新策略不支持更改 `min_count` 设置。

`spot_price`

（可选）设置此计算资源的最高竞价价格。仅当包含此计算资源的队列的 `compute_type` 设置设为 `spot` 时才使用。该设置替代 `spot_price` 设置。

如果您不指定值，则按竞价价格进行收费，最高为按需价格。

有关查找满足您需求的竞价型实例的帮助，请参阅竞价型实例顾问。

```
spot_price = 1.50
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

`[cw_log]` 部分

定义 CloudWatch Logs 的配置设置。

格式为 [cw_log *cw-log-name*]。*cw-log-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[cw_log custom-cw-log]
enable = true
retention_days = 14
```

有关更多信息，请参阅与 Amazon CloudWatch 日志集成、亚马逊 CloudWatch 控制面板和与 Amazon CloudWatch 日志集成。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.6.0 中添加了对 cw_log 的支持。

enable

（可选）指示是否启用 CloudWatch Logs。

默认值为 true。使用 false 可禁用 CloudWatch Logs。

以下示例启用 CloudWatch Logs。

```
enable = true
```

更新策略：如果更改此设置，则不允许更新。

retention_days

（可选）指示 CloudWatch Logs 保留单个日志事件的天数。

默认值为 14。支持的值为
1、3、5、7、14、30、60、90、120、150、180、365、400、545、731、1827 和 3653。

以下示例将 CloudWatch Logs 配置为保留日志事件 30 天。

```
retention_days = 30
```

更新策略：可以在更新期间更改此设置。

[dashboard] 部分

定义 CloudWatch 控制面板的配置设置。

格式为 [dashboard *dashboard-name*]。*dashboard-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[dashboard custom-dashboard]
enable = true
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.0 中添加了对 dashboard 的支持。

enable

（可选）指示是否启用 CloudWatch 控制面板。

默认值为 true。使用 false 可禁用 CloudWatch 控制面板。

以下示例启用 CloudWatch 控制面板。

```
enable = true
```

更新策略：可以在更新期间更改此设置。

[dcv] 部分

定义在头节点上运行的 Amazon DCV 服务器的配置设置。

要创建和配置 Amazon DCV 服务器，请dcv_settings使用您在dcv本节中定义的名称指定集群master，enable并将设置base_os为alinux2、和centos7、ubuntu1804或ubuntu2004。如果头节点是ARM实例，则设置base_os为alinux2centos7、或ubuntu1804。

格式为 [dcv *dcv-name*]。*dcv-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[dcv custom-dcv]
```

```
enable = master
port = 8443
access_from = 0.0.0.0/0
```

有关更多信息，请参阅 通过 Amazon 连接到头节点 DCV

> **Important**
>
> 默认情况下，Amazon DCV 端口设置对所有IPv4地址开放。 AWS ParallelCluster 但是，只有当您拥有亚马逊DCV会话的 Amazon DCV 端口，并且在URL从返回的 30 秒内连接到亚马逊DCV会话时，您才能连接到 Amazon 端口`pcluster dcv connect`。URL使用该`access_from`设置进一步限制对具有CIDR格式化的 IP 范围的 Amazon DCV 端口的访问，并使用该`port`设置来设置非标准端口。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.4 中删除了对 `centos8` 上 `[dcv]` 部分的支持。在 2.10.0 AWS ParallelCluster 版本中增加了`centos8`对上`[dcv]`节的支持。2.9.0 版本中增加了对 AWS 基于 Graviton 的实例`[dcv]`部分的 AWS ParallelCluster 支持。在 2.6.0 AWS ParallelCluster 版本中添加`ubuntu1804`了对`alinux2`和`[dcv]`部分的支持。在 2.5.0 AWS ParallelCluster 版本中增加了`centos7`对上`[dcv]`节的支持。

access_from

（可选，推荐）指定连接到 Amazon DCV 的CIDR格式化的 IP 范围。此设置仅在 AWS ParallelCluster 创建安全组时使用。

默认值是 `0.0.0.0/0`，允许从任何 Internet 地址访问。

```
access_from = 0.0.0.0/0
```

更新策略：可以在更新期间更改此设置。

enable

（必填）表示是否在头节点上启用 Amazon DCV。要在头节点DCV上启用 Amazon 并配置所需的安全组规则，请将`enable`设置设置为`master`。

以下示例在头节点DCV上启用 Amazon。

```
enable = master
```

> **Note**
>
> Amazon 会DCV自动生成自签名证书，用于保护在头节点上运行的 Amazon DCV 客户端和亚马逊DCV服务器之间的流量。要配置您自己的证书，请参阅 亚马逊DCVHTTPS证书。

更新策略：如果更改此设置，则不允许更新。

port

（可选）指定 Amazon 的端口DCV。

默认值为 8443。

```
port = 8443
```

更新策略：如果更改此设置，则不允许更新。

[ebs] 部分

定义头节点上挂载且通过 NFS 共享到计算节点的卷的 Amazon EBS 卷配置设置。

要了解如何在集群定义中包含 Amazon EBS 卷，请参阅 [cluster] ##/ebs_settings。

要将现有的 Amazon EBS 卷用于独立于集群生命周期的长期永久性存储，请指定 ebs_volume_id。

如果您未指定ebs_volume_id，则在 AWS ParallelCluster 创建集群时根据[ebs]设置创建 EBS 卷，并在删除集群时删除该卷和数据。

有关更多信息，请参阅 最佳实践：将集群移至新集群 AWS ParallelCluster 次要版本或补丁版本。

格式为 [ebs *ebs-name*]。*ebs-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[ebs custom1]
shared_dir = vol1
ebs_snapshot_id = snap-xxxxx
```

```
volume_type = io1
volume_iops = 200
...

[ebs custom2]
shared_dir = vol2
...

...
```

主题

- shared_dir
- ebs_kms_key_id
- ebs_snapshot_id
- ebs_volume_id
- encrypted
- volume_iops
- volume_size
- volume_throughput
- volume_type

shared_dir

（必需）指定在其中挂载共享 Amazon EBS 卷的路径。

使用多个 Amazon EBS 卷时需要此参数。

当使用一个 Amazon EBS 卷时，此选项将覆盖在 [cluster] 部分下指定的 shared_dir。在以下示例中，卷将挂载到 /vol1。

```
shared_dir = vol1
```

更新策略：如果更改此设置，则不允许更新。

ebs_kms_key_id

（可选）指定用于加密的自定义密 AWS KMS 钥。

此参数必须与 `encrypted = true` 一起使用。它还必须具有自定义 `ec2_iam_role`。

有关更多信息，请参阅 使用自定义 KMS 密钥对磁盘加密。

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

更新策略：如果更改此设置，则不允许更新。

ebs_snapshot_id

（可选）如果将快照用作卷的源，则定义 Amazon EBS 快照 ID。

没有默认值。

```
ebs_snapshot_id = snap-xxxxx
```

更新策略：如果更改此设置，则不允许更新。

ebs_volume_id

（可选）定义要附加到头节点的现有 Amazon EBS 卷的卷 ID。

没有默认值。

```
ebs_volume_id = vol-xxxxxx
```

更新策略：如果更改此设置，则不允许更新。

encrypted

（可选）指定是否加密 Amazon EBS 卷。注意：请勿 使用快照。

默认值为 `false`。

```
encrypted = false
```

更新策略：如果更改此设置，则不允许更新。

volume_iops

（可选）为 `io1`、`io2` 和 `gp3` 类型卷定义 IOPS 数。

默认值、支持的值以及 volume_size/volume_iops 比率因 volume_type 和 volume_size 而异。

volume_type = io1

默认值：volume_iops = 100

支持的值：volume_iops = 100–64000 †

最大 volume_iops/volume_size 比率 = 50 IOPS/GiB。5000 IOPS 需要至少 100 GiB 的 volume_size。

volume_type = io2

默认值：volume_iops = 100

支持的值：volume_iops = 100–64000（io2 Block Express 卷为 256000）†

最大 volume_iops/volume_size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 volume_size。

volume_type = gp3

默认值：volume_iops = 3000

支持的值：volume_iops = 3000–16000

最大 volume_iops/volume_size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 volume_size。

```
volume_iops = 200
```

更新策略：可以在更新期间更改此设置。

† 只有在 Nitro System 上构建的实例配置超过 32000 IOPS 时，才能保证最大 IOPS。其他实例保证最高为 32000 IOPS。除非您修改卷，否则较早的 io1 卷可能无法实现完全性能。io2Block Express 卷支持高达 256000 的 volume_iops 值。有关更多信息，请参阅 io2A mazon EC2 用户指南中的阻止 Express 卷（预览版）。

volume_size

（可选）指定要创建的卷的大小，以 GiB 为单位（如果未使用快照）。

默认值和支持的值因 volume_type 而异。

volume_type = standard

默认值：volume_size = 20 GiB

支持的值：volume_size = 1–1024 GiB

volume_type = gp2、io1、io2 和 gp3

默认值：volume_size = 20 GiB

支持的值：volume_size = 1–16384 GiB

volume_type = sc1 和 st1

默认值：volume_size = 500 GiB

支持的值：volume_size = 500–16384 GiB

```
volume_size = 20
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本之前，所有卷类型的默认值均为 20 GiB。

更新策略：如果更改此设置，则不允许更新。

volume_throughput

（可选）定义 gp3 卷类型的吞吐量，以 MiB/s 为单位。

默认值为 125。

支持的值：volume_throughput = 125–1000 MiB/s

volume_throughput 与 volume_iops 的比率不能超过 0.25。1000 MiB/s 的最大吞吐量要求 volume_iops 设置至少为 4000。

```
volume_throughput = 1000
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本中增加了对 Support 的支持。`volume_throughput`

更新策略：如果更改此设置，则不允许更新。

volume_type

（可选）指定要启动的卷的 Amazon EBS 卷类型。

有效选项为以下卷类型：

`gp2`, `gp3`

通用型 SSD

`io1`, `io2`

预调配 IOPS SSD

`st1`

吞吐量优化型 HDD

`sc1`

Cold HDD

`standard`

上一代磁介质

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 卷类型。

默认值为 `gp2`。

```
volume_type = io2
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本中增加了对`gp3`和`io2`的支持。

更新策略：如果更改此设置，则不允许更新。

[efs] 部分

定义头节点和计算节点上挂载的 Amazon EFS 的配置设置。有关更多信息，请参阅 Amazon EFS API 参考 中的 CreateFileSystem。

要了解如何在集群定义中包含 Amazon EFS 文件系统，请参阅 [cluster] ##/efs_settings。

要使用现有的 Amazon EFS 文件系统进行独立于集群生命周期的长期永久性存储，请指定 efs_fs_id。

如果不指定 efs_fs_id，则 AWS ParallelCluster 会在创建集群时根据 [efs] 设置创建 Amazon EFS 文件系统，并在删除集群时删除该文件系统和数据。

有关更多信息，请参阅 最佳实践：将集群移至新集群 AWS ParallelCluster 次要版本或补丁版本。

格式为 [efs *efs-name*]。*efs-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[efs customfs]
shared_dir = efs
encrypted = false
performance_mode = generalPurpose
```

主题

- efs_fs_id
- efs_kms_key_id
- encrypted
- performance_mode
- provisioned_throughput
- shared_dir
- throughput_mode

efs_fs_id

（可选）为现有文件系统定义 Amazon EFS 文件系统 ID。

指定此选项将使所有其他 Amazon EFS 选项失效，但 shared_dir 除外。

如果设置此选项，则它仅支持以下类型的文件系统：

- 堆栈的可用区中没有挂载目标的文件系统。
- 堆栈的可用区中有一个允许来自 0.0.0.0/0 的入站和出站 NFS 流量的现有挂载目标的文件系统。

用于验证 efs_fs_id 的健全性检查要求 IAM 角色具有以下权限：

- elasticfilesystem:DescribeMountTargets
- elasticfilesystem:DescribeMountTargetSecurityGroups
- ec2:DescribeSubnets
- ec2:DescribeSecurityGroups
- ec2:DescribeNetworkInterfaceAttribute

要避免错误，您必须将这些权限添加到您的 IAM 角色或设置 sanity_check = false。

> **Important**
>
> 在设置允许来自 0.0.0.0/0 的入站和出站 NFS 流量的挂载目标时，它将使文件系统承受来自挂载目标的可用区中任意位置的 NFS 挂载请求。AWS 不建议在堆栈的可用区中创建挂载目标，而是让 AWS 处理这个步骤。如果您想要在堆栈的可用区中拥有挂载目标，请考虑通过在 [vpc] 部分下提供 vpc_security_group_id 选项来使用自定义安全组。然后，将该安全组添加到挂载目标，并关闭 sanity_check 以创建集群。

没有默认值。

```
efs_fs_id = fs-12345
```

更新策略：如果更改此设置，则不允许更新。

efs_kms_key_id

（可选）标识用于保护加密文件系统的 AWS Key Management Service (AWS KMS) 客户托管密钥。如果设置了此参数，则必须将 encrypted 设置为 true。此参数对应于 Amazon EFS API 参考 中的 KmsKeyId 参数。

没有默认值。

```
efs_kms_key_id = 1234abcd-12ab-34cd-56ef-1234567890ab
```

更新策略：如果更改此设置，则不允许更新。

encrypted

（可选）指示是否对文件系统加密。此参数对应于 Amazon EFS API 参考 中的 Encrypted 参数。

默认值为 false。

```
encrypted = true
```

更新策略：如果更改此设置，则不允许更新。

performance_mode

（可选）定义文件系统的性能模式。此参数对应于 Amazon EFS API 参考 中的 PerformanceMode 参数。

有效选项为以下值：

- generalPurpose
- maxIO

这两个值都区分大小写。

对于大多数文件系统，我们推荐使用 generalPurpose 性能模式。

使用 maxIO 性能模式的文件系统可以扩展到更高级别的聚合吞吐量和每秒操作数。但是，对于大多数文件操作来说，代价是稍高的延迟。

创建文件系统后，无法更改此参数。

默认值为 generalPurpose。

```
performance_mode = generalPurpose
```

更新策略：如果更改此设置，则不允许更新。

provisioned_throughput

（可选）定义文件系统的预置吞吐量（以 MiB/s 为单位）。此参数对应于 Amazon EFS API 参考 中的 ProvisionedThroughputInMibps 参数。

如果您使用了此参数，则必须将 throughput_mode 设置为 provisioned。

吞吐量配额为 1024 MiB/s。要请求提高配额，请联系 AWS Support。

最小值为 0.0 MiB/s。

```
provisioned_throughput = 1024
```

更新策略：可以在更新期间更改此设置。

shared_dir

（必需）定义头节点和计算节点上的 Amazon EFS 挂载点。

此参数为必需参数。仅在指定 shared_dir 时使用 Amazon EFS 部分。

请勿使用 NONE 或 /NONE 作为共享目录。

以下示例在 /efs 上挂载 Amazon EFS。

```
shared_dir = efs
```

更新策略：如果更改此设置，则不允许更新。

throughput_mode

（可选）定义文件系统的吞吐量模式。此参数对应于 Amazon EFS API 参考 中的 ThroughputMode 参数。

有效选项为以下值：

- bursting
- provisioned

默认值为 bursting。

```
throughput_mode = provisioned
```

更新策略：可以在更新期间更改此设置。

[fsx] 部分

定义附加的适用于 Lustre 的 FSx 文件系统的配置设置。有关更多信息，请参阅 Amazon FSx API 参考中的 Amazon FSx CreateFileSystem。

如果 base_os 是 alinux2、centos7、ubuntu1804 或 ubuntu2004，则支持适用于 Lustre 的 FSx。

使用 Amazon Linux 时，内核必须为 4.14.104-78.84.amzn1.x86_64 或更高版本。有关说明，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的安装 lustre 客户端。

> **Note**
>
> 目前，在使用 awsbatch 作为调度器时，不支持适用于 Lustre 的 FSx。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.4 中删除了 centos8 上对适用于 Lustre 的 FSx 的支持。在 AWS ParallelCluster 版本 2.11.0 中添加了 ubuntu2004 上对适用于 Lustre 的 FSx 的支持。在 AWS ParallelCluster 版本 2.10.0 中添加了 centos8 上对适用于 Lustre 的 FSx 的支持。在 AWS ParallelCluster 版本 2.6.0 中添加了 alinux2、ubuntu1604 和 ubuntu1804 上对适用于 Lustre 的 FSx 的支持。在 AWS ParallelCluster 版本 2.4.0 中添加了 centos7 上对适用于 Lustre 的 FSx 的支持。

如果使用现有文件系统，则必须将其关联到一个安全组，该安全组允许到端口 988 的入站 TCP 流量。在安全组规则上将源设置为 0.0.0.0/0 时，可以从 VPC 安全组中所有 IP 范围的客户端访问该规则的协议和端口范围。要进一步限制对文件系统的访问，我们建议您对安全组规则使用更具限制性的源。例如，您可以使用更具体的 CIDR 范围、IP 地址或安全组 ID。在未使用 vpc_security_group_id 时，将自动执行此操作。

要使用现有的 Amazon FSx 文件系统进行独立于集群生命周期的长期永久性存储，请指定 `fsx_fs_id`。

如果不指定 `fsx_fs_id`，则 AWS ParallelCluster 会在创建集群时根据 `[fsx]` 设置创建适用于 Lustre 的 FSx 文件系统，并在删除集群时删除该文件系统和数据。

有关更多信息，请参阅 最佳实践：将集群移至新集群 AWS ParallelCluster 次要版本或补丁版本。

格式为 `[fsx fsx-name]`。*fsx-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[fsx fs]
shared_dir = /fsx
fsx_fs_id = fs-073c3803dca3e28a6
```

要创建并配置新的文件系统，请使用以下参数：

```
[fsx fs]
shared_dir = /fsx
storage_capacity = 3600
imported_file_chunk_size = 1024
export_path = s3://bucket/folder
import_path = s3://bucket
weekly_maintenance_start_time = 1:00:00
```

主题

- auto_import_policy
- automatic_backup_retention_days
- copy_tags_to_backups
- daily_automatic_backup_start_time
- data_compression_type
- deployment_type
- drive_cache_type
- export_path
- fsx_backup_id
- fsx_fs_id
- fsx_kms_key_id

- import_path
- imported_file_chunk_size
- per_unit_storage_throughput
- shared_dir
- storage_capacity
- storage_type
- weekly_maintenance_start_time

auto_import_policy

（可选）指定自动导入策略，以便反映用于创建适用于 Lustre 的 FSx 文件系统的 S3 存储桶中的变化。可能的值包括：

NEW

适用于 Lustre 的 FSx 会自动导入添加到链接 S3 存储桶中但当前不存在于适用于 Lustre 的 FSx 文件系统中的任何新对象的目录列表。

NEW_CHANGED

适用于 Lustre 的 FSx 会自动导入添加到 S3 存储桶的任何新对象以及在 S3 存储桶中更改的任何现有对象的文件和目录列表。

此参数对应于 AutoImportPolicy 属性。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的自动从 S3 桶导入更新。指定 auto_import_policy 参数后，不得指定 automatic_backup_retention_days、copy_tags_to_backups、daily_automatic_backup_sta 和 fsx_backup_id 参数。

如果未指定 auto_import_policy 设置，则会禁用自动导入。适用于 Lustre 的 FSx 仅在创建文件系统时更新链接的 S3 存储桶中的文件和目录列表。

```
auto_import_policy = NEW_CHANGED
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.0 中添加了对 auto_import_policy 的支持。

更新策略：如果更改此设置，则不允许更新。

automatic_backup_retention_days

（可选）指定保留自动备份的天数。此参数仅适用于 PERSISTENT_1 部署类型。指定 automatic_backup_retention_days 参数后，不得指定 auto_import_policy、export_path、import_path 和 imported_file_chunk_size 参数。此参数对应于 AutomaticBackupRetentionDays 属性。

默认值为 0。此设置禁用自动备份。可能的值是介于 0 到 35 之间的整数（含 0 和 35）。

```
automatic_backup_retention_days = 35
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.8.0 中添加了对 automatic_backup_retention_days 的支持。

更新策略：可以在更新期间更改此设置。

copy_tags_to_backups

（可选）指定是否将文件系统的标签复制到备份中。此参数仅适用于 PERSISTENT_1 部署类型。指定 copy_tags_to_backups 参数后，必须使用大于 0 的值指定 automatic_backup_retention_days，并且不得指定 auto_import_policy、export_path、import_path 和 imported_file_chunk_size 参数。此参数对应于 CopyTagsToBackups 属性。

默认值为 false。

```
copy_tags_to_backups = true
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.8.0 中添加了对 copy_tags_to_backups 的支持。

更新策略：如果更改此设置，则不允许更新。

daily_automatic_backup_start_time

（可选）指定一天中开始自动备份的时间 (UTC)。此参数仅适用于 PERSISTENT_1 部署类型。指定 daily_automatic_backup_start_time 参数后，必须使用大于 0 的值指定 automatic_backup_retention_days，并且不得指定 auto_import_policy、export_path、import_path 和 imported_file_chunk_size 参数。此参数对应于 DailyAutomaticBackupStartTime 属性。

格式为 HH:MM，其中 HH 是一天中的零填充小时（0-23），MM 是小时中的零填充分钟。例如，1:03 A.M. UTC 如下所示。

```
daily_automatic_backup_start_time = 01:03
```

默认值是介于 00:00 和 23:59 之间的随机时间。

> **Note**
>
> 在 AWS ParallelCluster 版本 2.8.0 中添加了对 daily_automatic_backup_start_time 的支持。

更新策略：可以在更新期间更改此设置。

data_compression_type

（可选）指定适用于 Lustre 的 FSx 数据压缩类型。此参数对应于 DataCompressionType 属性。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的 适用于 Lustre 的 FSx 数据压缩。

唯一有效值为 LZ4。要禁用数据压缩，请删除 data_compression_type 参数。

```
data_compression_type = LZ4
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.11.0 中添加了对 data_compression_type 的支持。

更新策略：可以在更新期间更改此设置。

deployment_type

（可选）指定适用于 Lustre 的 FSx 部署类型。此参数对应于 DeploymentType 属性。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的适用于 Lustre 的 FSx 部署选项。为数据的临时存储和短期处理选择临时部署类型。SCRATCH_2 是最新一代临时文件系统。它提供了超出基准吞吐量的突增吞吐量以及传输中数据加密。

有效值为 SCRATCH_1、SCRATCH_2 和 PERSISTENT_1。

SCRATCH_1

适用于 Lustre 的 FSx 的默认部署类型，storage_capacity 设置的可能值为 1200 和 2400，以及 3600 的任何倍数。在 AWS ParallelCluster 版本 2.4.0 中添加了对 SCRATCH_1 的支持。

SCRATCH_2

最新一代临时文件系统。它支持的工作负载最高可达基准吞吐量的六倍。对于支持的 AWS 区域中支持的实例类型，它还支持传输中数据加密。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的加密传输中数据。对于此部署类型，storage_capacity 设置的可能值为 1200，以及 2400 的任何倍数。在 AWS ParallelCluster 版本 2.6.0 中添加了对 SCRATCH_2 的支持。

PERSISTENT_1

专为长期存储而设计。文件服务器具有高可用性，并且数据在文件系统的 AWS 可用区内复制。对于支持的实例类型，它还支持传输中数据加密。对于此部署类型，storage_capacity 设置的可能值为 1200，以及 2400 的任何倍数。在 AWS ParallelCluster 版本 2.6.0 中添加了对 PERSISTENT_1 的支持。

默认值为 SCRATCH_1。

```
deployment_type = SCRATCH_2
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.6.0 中添加了对 deployment_type 的支持。

更新策略：如果更改此设置，则不允许更新。

drive_cache_type

（可选）指定文件系统具有 SSD 驱动器缓存。只有将 storage_type 设置设为 HDD 后，才能设置此参数。此参数对应于 DriveCacheType 属性。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的适用于 Lustre 的 FSx 部署选项。

唯一有效值为 READ。要禁用 SSD 驱动器缓存，请不要指定 drive_cache_type 设置。

```
drive_cache_type = READ
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.0 中添加了对 drive_cache_type 的支持。

更新策略：如果更改此设置，则不允许更新。

export_path

（可选）指定在其中导出文件系统的根的 Amazon S3 路径。指定 export_path 参数后，不得指定 automatic_backup_retention_days、copy_tags_to_backups、daily_automatic_backup_sta 和 fsx_backup_id 参数。此参数对应于 ExportPath 属性。文件数据和元数据不会自动导出到 export_path。有关导出数据和元数据的信息，请参阅 适用于 Lustre 的 Amazon FSx 用户指南 中的将更改导出到数据存储库。

默认值为 s3://*import-bucket*/FSxLustre*[creation-timestamp]*，其中 *import-bucket* 是 import_path 参数中提供的存储桶。

```
export_path = s3://bucket/folder
```

更新策略：如果更改此设置，则不允许更新。

fsx_backup_id

（可选）指定用于从现有备份还原文件系统的备份 ID。指定 fsx_backup_id 参数后，不得指定 auto_import_policy、deployment_type、export_path、fsx_kms_key_id、import_path、imp 和 per_unit_storage_throughput 参数。这些参数从备份中进行读取。此外，不得指定 auto_import_policy、export_path、import_path 和 imported_file_chunk_size 参数。

此参数对应于 BackupId 属性。

```
fsx_backup_id = backup-fedcba98
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.8.0 中添加了对 fsx_backup_id 的支持。

更新策略：如果更改此设置，则不允许更新。

fsx_fs_id

（可选）附加现有的适用于 Lustre 的 FSx 文件系统。

如果指定了此选项，则仅使用 [fsx] 部分中的 shared_dir 和 fsx_fs_id 设置，并忽略 [fsx] 部分中的任何其他设置。

```
fsx_fs_id = fs-073c3803dca3e28a6
```

更新策略：如果更改此设置，则不允许更新。

fsx_kms_key_id

（可选）指定您的 AWS Key Management Service (AWS KMS) 客户托管密钥的密钥 ID。

此密钥用于加密文件系统中的静态数据。

它必须与自定义 ec2_iam_role 结合使用。有关更多信息，请参阅 使用自定义 KMS 密钥对磁盘加密。此参数对应于 Amazon FSx API 参考 中的 KmsKeyId 参数。

```
fsx_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.6.0 中添加了对 fsx_kms_key_id 的支持。

更新策略：如果更改此设置，则不允许更新。

import_path

（可选）指定将其中的数据加载到文件系统并用作导出存储桶的 S3 存储桶。有关更多信息，请参阅 export_path。如果指定 import_path 参数，则不得指定 automatic_backup_retention_days、copy_tags_to_backups、daily_automatic_backup_sta 和 fsx_backup_id 参数。此参数对应于 Amazon FSx API 参考 中的 ImportPath 参数。

导入在创建集群时发生。有关更多信息，请参阅适用于 Lustre 的 Amazon FSx 用户指南 中的从数据存储库导入数据。导入时，仅导入文件元数据（名称、所有权、时间戳和权限）。在首次访问文件之前，不会从 S3 存储桶导入文件数据。有关预加载文件内容的信息，请参阅 适用于 Lustre 的 Amazon FSx 用户指南 中的将文件预加载到文件系统。

如果未提供值，则该文件系统为空。

```
import_path =  s3://bucket
```

更新策略：如果更改此设置，则不允许更新。

imported_file_chunk_size

（可选）对于从数据存储库导入的文件（使用 import_path），此参数决定单个物理磁盘上存储的每个文件的条带计数和最大数据量（以 MiB 为单位）。可以对单个文件进行条带化的最大磁盘数受构成文件系统的总磁盘数限制。指定 imported_file_chunk_size 参数后，不得指定 automatic_backup_retention_days、copy_tags_to_backups、daily_automatic_backup_sta 和 fsx_backup_id 参数。此参数对应于 ImportedFileChunkSize 属性。

区块大小默认为 1024 (1 GiB)，最大值能够达到 512000 MiB (500 GiB)。Amazon S3 数据元的最大大小为 5 TB。

```
imported_file_chunk_size = 1024
```

更新策略：如果更改此设置，则不允许更新。

per_unit_storage_throughput

（**PERSISTENT_1** 部署类型所必需）对于 deployment_type = PERSISTENT_1 部署类型，描述每 1 TiB 存储的读取和写入吞吐量（以 MB/s/TiB 为单位）。文件系统吞吐能力是将通过文件系统存储容量（TiB）乘以 per_unit_storage_throughput（MB/s/TiB）计算得出的。对于 2.4 TiB 文件系

统，预置 50 MB/s/TiB 的 per_unit_storage_throughput 将得到 120 MB/s 的文件系统吞吐量。您需要为预置的吞吐量付费。此参数对应于 PerUnitStorageThroughput 属性。

可能的值取决于 storage_type 设置的值。

storage_type = SSD

可能的值为 50、100、200。

storage_type = HDD

可能的值为 12、40。

```
per_unit_storage_throughput = 200
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.6.0 中添加了对 per_unit_storage_throughput 的支持。

更新策略：如果更改此设置，则不允许更新。

shared_dir

（必需）定义头节点和计算节点上适用于 Lustre 的 FSx 文件系统的挂载点。

请勿使用 NONE 或 /NONE 作为共享目录。

以下示例在 /fsx 上挂载文件系统。

```
shared_dir = /fsx
```

更新策略：如果更改此设置，则不允许更新。

storage_capacity

（必需）指定文件系统的存储容量（以 GiB 为单位）。此参数对应于 StorageCapacity 属性。

存储容量可以使用的值因 deployment_type 设置而异。

SCRATCH_1

可能的值是 1200 和 2400，以及 3600 的任何倍数。

SCRATCH_2

可能的值是 1200，以及 2400 的任何倍数。

PERSISTENT_1

可能的值根据其他设置的值而有所不同。

storage_type = SSD

可能的值是 1200，以及 2400 的任何倍数。

storage_type = HDD

可能的值根据 per_unit_storage_throughput 设置的设置而有所不同。

per_unit_storage_throughput = 12

可能的值是 6000 的任何倍数。

per_unit_storage_throughput = 40

可能的值是 1800 的任何倍数。

```
storage_capacity = 7200
```

> **Note**
>
> 对于 AWS ParallelCluster 版本 2.5.0 和 2.5.1，storage_capacity 支持的值为 1200 和 2400，以及 3600 的任何倍数。对于早于 AWS ParallelCluster 版本 2.5.0 的版本，storage_capacity 的最小大小为 3600。

更新策略：如果更改此设置，则不允许更新。

storage_type

（可选）指定文件系统的存储类型。此参数对应于 StorageType 属性。可能的值为 SSD 和 HDD。默认为 SSD。

存储类型会更改其他设置的可能值。

`storage_type = SSD`

指定固态驱动器 (SSD) 存储类型。

`storage_type = SSD` 会更改其他几个设置的可能值。

`drive_cache_type`

不能指定此设置。

`deployment_type`

此设置可以设置为 `SCRATCH_1`、`SCRATCH_2` 或 `PERSISTENT_1`。

`per_unit_storage_throughput`

如果 `deployment_type` 被设置为 `PERSISTENT_1`，则必须指定此设置。可能的值为 50、100 或 200。

`storage_capacity`

必须指定此设置。可能的值根据 `deployment_type` 而有所不同。

`deployment_type = SCRATCH_1`

`storage_capacity` 可以是 1200、2400 或 3600 的任何倍数。

`deployment_type = SCRATCH_2` 或 `deployment_type = PERSISTENT_1`

`storage_capacity` 可以是 1200 或 2400 的任何倍数。

`storage_type = HDD`

指定硬盘驱动器 (HDD) 存储类型。

`storage_type = HDD` 会更改其他设置的可能值。

`drive_cache_type`

可以指定此设置。

`deployment_type`

此设置必须设置为 `PERSISTENT_1`。

`per_unit_storage_throughput`

必须指定此设置。可能的值为 12 或 40。

storage_capacity

必须指定此设置。可能的值根据 per_unit_storage_throughput 设置而有所不同。

storage_capacity = 12

storage_capacity 可以是 6000 的任何倍数。

storage_capacity = 40

storage_capacity 可以是 1800 的任何倍数。

```
storage_type = SSD
```

> **Note**
>
> 在 AWS ParallelCluster 版本 2.10.0 中添加了对 storage_type 设置的支持。

更新策略：如果更改此设置，则不允许更新。

weekly_maintenance_start_time

（可选）指定执行每周维护的首选时间，采用 UTC 时区。此参数对应于 WeeklyMaintenanceStartTime 属性。

格式为 [星期几]:[小时]:[分钟]。例如，周一的午夜如下所示。

```
weekly_maintenance_start_time = 1:00:00
```

更新策略：可以在更新期间更改此设置。

[queue] 部分

定义单个队列的配置设置。只有当 scheduler 设置为 slurm 时，才支持 [queue] 部分。

格式为 [queue *<queue-name>*]。*queue-name* 必须以小写字母开头，不能超过 30 个字符，并且只能包含小写字母、数字和连字符 (-)。

```
[queue q1]
```

```
compute_resource_settings = i1,i2
placement_group = DYNAMIC
enable_efa = true
disable_hyperthreading = false
compute_type = spot
```

> **Note**
>
> 2.9.0 AWS ParallelCluster 版本中增加了对该[queue]部分的支持。

主题

- compute_resource_settings
- compute_type
- disable_hyperthreading
- enable_efa
- enable_efa_gdr
- placement_group

compute_resource_settings

（必需）标识包含该队列的计算资源配置的 [compute_resource] 部分。部分名称必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

每个 [compute_resource] 部分最多支持三 (3) 个 [queue] 部分。

例如，以下设置指定使用以 [compute_resource cr1] 和 [compute_resource cr2] 开始的部分。

```
compute_resource_settings = cr1, cr2
```

更新策略：如果更改此设置，则不允许更新。

compute_type

（可选）定义要为此队列启动的实例的类型。该设置替代 cluster_type 设置。

有效的选项为：ondemand 和 spot。

默认值为 ondemand。

有关竞价型实例的更多信息，请参阅使用竞价型实例。

> **Note**
>
> 使用竞价型实例要求您的账户中存在 AWSServiceRoleForEC2Spot 服务相关角色。要使用在您的账户中创建此角色 AWS CLI，请运行以下命令：
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> 有关更多信息，请参阅 Amazon EC2 用户指南中的竞价型实例请求的服务相关角色。

以下示例 SpotInstances 用于此队列中的计算节点。

```
compute_type = spot
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

disable_hyperthreading

（可选）对此队列中的节点禁用超线程。并非所有实例类型都可以禁用超线程。有关支持禁用超线程的实例类型列表，请参阅 Amazon EC2 用户指南中每种实例类型的 CPU 核心和每个 CPU 内核的线程。如果定义了 [cluster] 部分中的 disable_hyperthreading 设置，则不能定义此设置。

默认值为 false。

```
disable_hyperthreading = true
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

enable_efa

（可选）如果设置为 true，则指定为此队列中的节点启用 Elastic Fabric Adapter (EFA)。要查看支持 EFA 的 EC2 实例的列表，请参阅 Amazon EC2 用户指南（适用于 Linux 实例）中的支持的实例类型。如果定义了 [cluster] 部分中的 enable_efa 设置，则不能定义此设置。应使用集群置放群组来最大限度地减少实例之间的延迟。有关更多信息，请参阅 placement 和 placement_group。

```
enable_efa = true
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

enable_efa_gdr

（可选）从 2.11.3 AWS ParallelCluster 版本开始，此设置无效。如果实例类型支持 Elastic Fabric Adapter (EFA)，则始终为计算节点启用对 GPUDirect RDMA（远程直接内存访问）的 Elastic Fabric Adapter (EFA) 支持。

> **Note**
>
> AWS ParallelCluster 版本 2.10.0 到 2.11.2：如果`true`，则指定为该队列中的节点启用弹性结构适配器 (EFA) GpuDirect RDMA（远程直接内存访问）。将此参数设置为 `true` 需要先将 `enable_efa` 设置设为 `true`。这些操作系统（`alinux2`、`centos7`、`ubuntu1804` 或 `ubuntu2004`）上的以下实例类型 (`p4d.24xlarge`) 支持 EFA GPUDirect RDMA。如果定义了 `[cluster]` 部分中的 `enable_efa_gdr` 设置，则不能定义此设置。应使用集群置放群组来最大限度地减少实例之间的延迟。有关更多信息，请参阅 `placement` 和 `placement_group`。

默认值为 `false`。

```
enable_efa_gdr = true
```

> **Note**
>
> 在 2.10.0 AWS ParallelCluster 版本中添加`enable_efa_gdr`了对 Support 的支持。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

placement_group

（可选）如果存在，则定义此队列的置放群组。该设置替代 `placement_group` 设置。

有效选项为以下值：

- DYNAMIC
- 现有的 Amazon EC2 集群置放群组名称

当设置为 DYNAMIC 时，将此队列的唯一置放群组作为集群堆栈的一部分进行创建和删除。

有关置放群组的更多信息，请参阅 Amazon EC2 用户指南中的置放群组。如果将同一个置放群组用于不同的实例类型，则请求更有可能因容量不足错误而失败。有关更多信息，请参阅 Amazon EC2 用户指南中的实例容量不足。

没有默认值。

并非所有实例类型都支持集群置放群组。例如，t2.micro 不支持集群置放群组。有关支持集群置放群组的实例类型列表的信息，请参阅 Amazon EC2 用户指南中的集群置放群组规则和限制。有关使用置放群组时的提示，请参阅置放群组和实例启动问题。

```
placement_group = DYNAMIC
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

[raid] 部分

定义通过多个完全相同的 Amazon EBS 卷构建的 RAID 阵列的配置设置。RAID 驱动器挂载到头节点上并通过 NFS 导出到计算节点。

格式为 [raid *raid-name*]。*raid-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[raid rs]
shared_dir = raid
raid_type = 1
num_of_raid_volumes = 2
encrypted = true
```

主题

- shared_dir
- ebs_kms_key_id
- encrypted

- num_of_raid_volumes
- raid_type
- volume_iops
- volume_size
- volume_throughput
- volume_type

shared_dir

（必需）定义头节点和计算节点上 RAID 阵列的挂载点。

仅在指定此参数时创建 RAID 驱动器。

请勿使用 NONE 或 /NONE 作为共享目录。

以下示例在 /raid 上挂载阵列。

```
shared_dir = raid
```

更新策略：如果更改此设置，则不允许更新。

ebs_kms_key_id

（可选）指定用于加密的自定义密 AWS KMS 钥。

此参数必须与 encrypted = true 一起使用，并且它必须具有自定义 ec2_iam_role。

有关更多信息，请参阅 使用自定义 KMS 密钥对磁盘加密。

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

更新策略：如果更改此设置，则不允许更新。

encrypted

（可选）指定是否对文件系统加密。

默认值为 false。

```
encrypted = false
```

更新策略：如果更改此设置，则不允许更新。

num_of_raid_volumes

（可选）定义用于组装 RAID 阵列的 Amazon EBS 卷数。

最小卷数为 2。

最大卷数为 5。

默认值为 2。

```
num_of_raid_volumes = 2
```

更新策略：如果更改此设置，则不允许更新。

raid_type

（必需）定义 RAID 阵列的 RAID 类型。

仅在指定此参数时创建 RAID 驱动器。

有效选项为以下值：

- `0`
- `1`

有关 RAID 类型的更多信息，请参阅 Amazon EC2 用户指南中的 RAID 信息。

以下示例创建 RAID `0` 阵列：

```
raid_type = 0
```

更新策略：如果更改此设置，则不允许更新。

volume_iops

（可选）为 `io1`、`io2` 和 `gp3` 类型卷定义 IOPS 数。

默认值、支持的值以及 volume_size/volume_iops 比率因 volume_type 和 volume_size 而异。

volume_type = io1

默认值：volume_iops = 100

支持的值：volume_iops = 100–64000 †

最大 volume_iops/volume_size 比率 = 50 IOPS/GiB。5000 IOPS 需要至少 100 GiB 的 volume_size。

volume_type = io2

默认值：volume_iops = 100

支持的值：volume_iops = 100–64000（io2 Block Express 卷为 256000）†

最大 volume_iops/volume_size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 volume_size。

volume_type = gp3

默认值：volume_iops = 3000

支持的值：volume_iops = 3000–16000

最大 volume_iops/volume_size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 volume_size。

```
volume_iops = 3000
```

更新策略：可以在更新期间更改此设置。

† 只有在 Nitro System 上构建的实例配置超过 32000 IOPS 时，才能保证最大 IOPS。其他实例保证最高为 32000 IOPS。除非您修改卷，否则较旧的 io1 卷可能无法实现完全性能。io2Block Express 卷支持高达 256000 的 volume_iops 值。有关更多信息，请参阅 io2A mazon EC2 用户指南中的阻止 Express 卷（预览版）。

volume_size

（可选）定义要创建的卷的大小，以 GiB 为单位。

默认值和支持的值因 volume_type 而异。

volume_type = standard

默认值：volume_size = 20 GiB

支持的值：volume_size = 1–1024 GiB

volume_type = gp2、io1、io2 和 gp3

默认值：volume_size = 20 GiB

支持的值：volume_size = 1–16384 GiB

volume_type = sc1 和 st1

默认值：volume_size = 500 GiB

支持的值：volume_size = 500–16384 GiB

```
volume_size = 20
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本之前，所有卷类型的默认值均为 20 GiB。

更新策略：如果更改此设置，则不允许更新。

volume_throughput

（可选）定义 gp3 卷类型的吞吐量，以 MiB/s 为单位。

默认值为 125。

支持的值：volume_throughput = 125–1000 MiB/s

volume_throughput 与 volume_iops 的比率不能超过 0.25。1000 MiB/s 的最大吞吐量要求 volume_iops 设置至少为 4000。

```
volume_throughput = 1000
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本中增加了对 Support 的支持。`volume_throughput`

更新策略：如果更改此设置，则不允许更新。

volume_type

（可选）定义要构建的卷的类型。

有效选项为以下值：

`gp2`, `gp3`

通用型 SSD

`io1`, `io2`

预调配 IOPS SSD

`st1`

吞吐量优化型 HDD

`sc1`

Cold HDD

`standard`

上一代磁介质

有关更多信息，请参阅《Amazon EC2 用户指南》中的 Amazon EBS 卷类型。

默认值为 `gp2`。

```
volume_type = io2
```

> **Note**
>
> 在 2.10.1 AWS ParallelCluster 版本中增加了对`gp3`和`io2`的支持。

更新策略：如果更改此设置，则不允许更新。

[scaling] 部分

主题

- scaledown_idletime

指定用于定义计算节点如何扩展的设置。

格式为 [scaling *scaling-name*]。*scaling-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[scaling custom]
scaledown_idletime = 10
```

scaledown_idletime

（可选）指定没有作业的时间量（以分钟为单位），经过此时段后，计算节点将终止。

如果 awsbatch 为调度器，则不使用此参数。

默认值为 10。

```
scaledown_idletime = 10
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

[vpc] 部分

指定 Amazon VPC 配置设置。有关的更多信息VPCs，请参阅 Amazon 是什么VPC？ 以及《Amazon VPC 用户指南》VPC中的安全最佳实践。

格式为 [vpc *vpc-name*]。*vpc-name* 必须以字母开头，不能超过 30 个字符，并且只能包含字母、数字、连字符 (-) 和下划线 (_)。

```
[vpc public]
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-xxxxxx
```

主题

additional_sg

（可选）为所有实例提供额外的 Amazon VPC 安全组 ID。

没有默认值。

```
additional_sg = sg-xxxxxx
```

compute_subnet_cidr

（可选）指定无类域间路由 (CIDR) 块。如果要创建计算子网 AWS ParallelCluster ，请使用此参数。

```
compute_subnet_cidr = 10.0.100.0/24
```

更新策略：如果更改此设置，则不允许更新。

compute_subnet_id

（可选）指定要在其中预置计算节点的现有子网的 ID。

如果未指定，则 compute_subnet_id 使用 master_subnet_id 的值。

如果子网为私有子网，则必须设置 NAT Web 访问权限。

```
compute_subnet_id = subnet-xxxxxx
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

master_subnet_id

（必需）指定要在其中预置头节点的现有子网的 ID。

```
master_subnet_id = subnet-xxxxxx
```

更新策略：如果更改此设置，则不允许更新。

ssh_from

（可选）指定允许SSH访问的CIDR格式化的 IP 范围。

此参数仅在 AWS ParallelCluster 创建安全组时使用。

默认值为 `0.0.0.0/0`。

```
ssh_from = 0.0.0.0/0
```

更新策略：可以在更新期间更改此设置。

use_public_ips

（可选）定义是否向计算实例分配公有 IP 地址。

如果设置为 `true`，则弹性 IP 地址与头节点相关联。

如果设置为 `false`，则根据“自动分配公有 IP”子网配置参数的值，头节点将具有或没有公有 IP。

有关示例，请参阅联网配置。

默认值为 `true`。

```
use_public_ips = true
```

> **Important**
>
> 默认情况下，每个地址 AWS 账户 仅限于五 (5) 个弹性 IP 地址 AWS 区域。有关更多信息，请参阅 Amazon EC2 用户指南中的弹性 IP 地址限制。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

vpc_id

（必需）指定要VPC在其中配置集群的 Amazon 的 ID。

```
vpc_id = vpc-xxxxxx
```

更新策略：如果更改此设置，则不允许更新。

vpc_security_group_id

（可选）指定将现有安全组用于所有实例。

没有默认值。

```
vpc_security_group_id = sg-xxxxxx
```

创建的安全组 AWS ParallelCluster 允许使用端口 22 从设置中指定的地址进行SSH访问，如果未指定ssh_from设置，则允许使用所有IPv4地址 (0.0.0.0/0) 进行访问。ssh_from如果启用了AmazonDCV，则安全组允许DCV使用端口 8443（或设置指定的任何内容）从port设置中指定的地址访问亚马逊，如果未指定access_from设置，则允许使用所有IPv4地址 (0.0.0.0/0) 访问亚马逊。access_from

> **Warning**
>
> 您可以更改此参数的值并更新集群（如果[cluster]fsx_settings未指定），或者两者兼而有之，fsx_settings并且fsx-fs-id在[fsx fs]中为 Lustre 文件系统指定了外部现有FSx的 Lustre 文件系统。
> 如果在和中指定了FSx适用于 Lustre 的 AWS ParallelCluster 托管文件系统，则无法更改此参数fsx_settings的[fsx fs]值。

更新政策：如果配置中未指定 AWS ParallelCluster 托管 Amazon FSx for Lustre 文件系统，则可以在更新期间更改此设置。

示例

以下示例配置演示了使用 Slurm、Torque 和 AWS Batch 调度器时的 AWS ParallelCluster 配置。

> **Note**
>
> 从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 或 Torque 调度器。

目录

Slurm Workload Manager (`slurm`)

以下示例使用 `slurm` 计划程序启动集群。该示例配置启动 1 个包含 2 个作业队列的集群。第一个队列 `spot` 最初具有 2 个可用的 `t3.micro` 竞价型实例。它可以纵向扩展到最多 10 个实例，并可在 10 分钟没有作业运行时缩减到最少 1 个实例（可使用 `scaledown_idletime` 设置进行调整）。第二个队列 ondemand 从没有实例开始，最多可以纵向扩展到 5 个 `t3.micro` 按需型实例。

```
[global]
update_check = true
sanity_check = true
cluster_template = slurm

[aws]
aws_region_name = <your AWS ##>

[vpc public]
master_subnet_id = <your subnet>
vpc_id = <your VPC>

[cluster slurm]
key_name = <your EC2 keypair name>
base_os = alinux2                   # optional, defaults to alinux2
scheduler = slurm
master_instance_type = t3.micro     # optional, defaults to t3.micro
vpc_settings = public
queue_settings = spot,ondemand

[queue spot]
compute_resource_settings = spot_i1
compute_type = spot                 # optional, defaults to ondemand
```

```
[compute_resource spot_i1]
instance_type = t3.micro
min_count = 1                       # optional, defaults to 0
initial_count = 2                   # optional, defaults to 0

[queue ondemand]
compute_resource_settings = ondemand_i1

[compute_resource ondemand_i1]
instance_type = t3.micro
max_count = 5                       # optional, defaults to 10
```

Son of Grid Engine (**sge**) 和 Torque Resource Manager (**torque**)

> **Note**
>
> 此示例仅适用于版本 2.11.4 及以前的 AWS ParallelCluster 版本。从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 或 Torque 调度器。

以下示例使用 `torque` 划 `sge` 调度器启动集群。要使用 SGE，请将 `scheduler = torque` 更改为 `scheduler = sge`。该示例配置允许最多 5 个并发节点，且当 10 分钟没有作业运行时缩减到 2 个节点。

```
[global]
update_check = true
sanity_check = true
cluster_template = torque

[aws]
aws_region_name = <your AWS ##>

[vpc public]
master_subnet_id = <your subnet>
vpc_id = <your VPC>

[cluster torque]
key_name = <your EC2 keypair name>but they aren't eligible for future updates
base_os = alinux2                   # optional, defaults to alinux2
scheduler = torque                  # optional, defaults to sge
master_instance_type = t3.micro     # optional, defaults to t3.micro
```

```
vpc_settings = public
initial_queue_size = 2              # optional, defaults to 0
maintain_initial_size = true        # optional, defaults to false
max_queue_size = 5                  # optional, defaults to 10
```

> **Note**
>
> 从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 或 Torque 调度器。如果您使用这些版本，则可以继续使用，或者向 AWS 服务和 AWS Support 团队寻求问题排查支持。

AWS Batch (**awsbatch**)

以下示例使用 awsbatch 计划程序启动集群。它设置为根据您的作业资源需求来选择更好的实例类型。

该示例配置允许最多 40 个并发 vCPU，且当 10 分钟没有作业运行时缩减到零（可使用 scaledown_idletime 设置进行调整）。

```
[global]
update_check = true
sanity_check = true
cluster_template = awsbatch

[aws]
aws_region_name = <your AWS ##>

[vpc public]
master_subnet_id = <your subnet>
vpc_id = <your VPC>

[cluster awsbatch]
scheduler = awsbatch
compute_instance_type = optimal # optional, defaults to optimal
min_vcpus = 0                   # optional, defaults to 0
desired_vcpus = 0               # optional, defaults to 4
max_vcpus = 40                  # optional, defaults to 20
base_os = alinux2               # optional, defaults to alinux2, controls the base_os
 of
                                # the head node and the docker image for the compute
 fleet
key_name = <your EC2 keypair name>
```

```
vpc_settings = public
```

AWS ParallelCluster 的工作原理

AWS ParallelCluster 不仅构建为一种管理集群的方法，还作为有关如何使用 AWS 服务构建 HPC 环境的参考。

主题

AWS ParallelCluster 进程

本节仅适用于使用支持的传统作业调度程序之一（SGE、Slurm 或 Torque）部署的 HPC 集群。在与这些计划程序结合使用时，AWS ParallelCluster 通过同时与自动扩缩组和底层作业调度器交互来管理计算节点的预置和删除。

对于基于 AWS Batch 的 HPC 集群，AWS ParallelCluster 依赖于 AWS Batch 提供的功能来进行计算节点管理。

> **Note**
>
> 从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 或 Torque 调度器。您可以在 2.11.4 及之前的版本中继续使用这些调度器，但它们没有资格获得 AWS 服务和 AWS 支持团队的未来更新或故障排除支持。

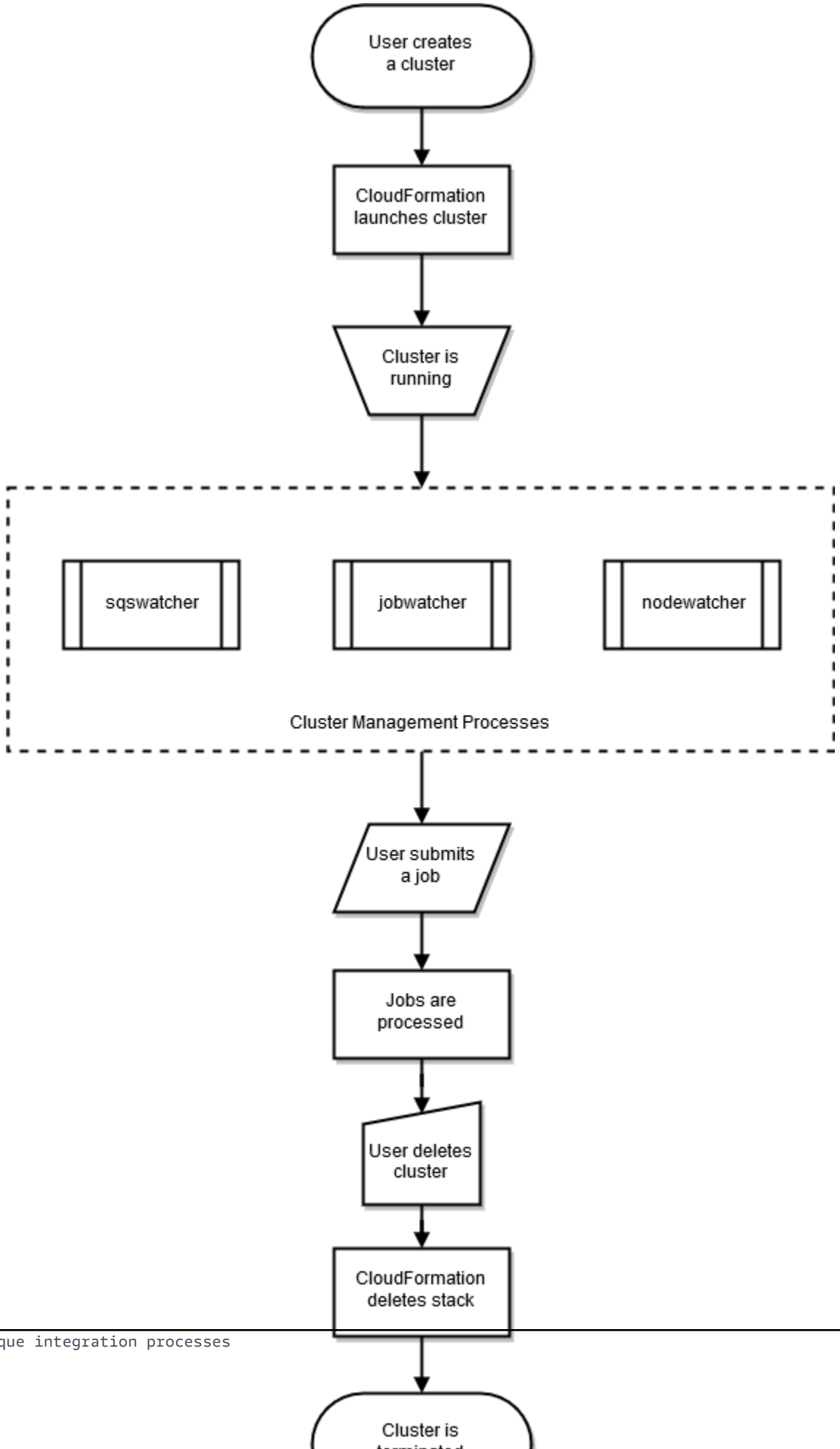
主题

SGE and Torque integration processes

> **Note**
>
> 本节仅适用于 2.11.4 及以下的 AWS ParallelCluster 版本。从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 和 Torque 调度器、Amazon SNS 和 Amazon SQS。

一般概述

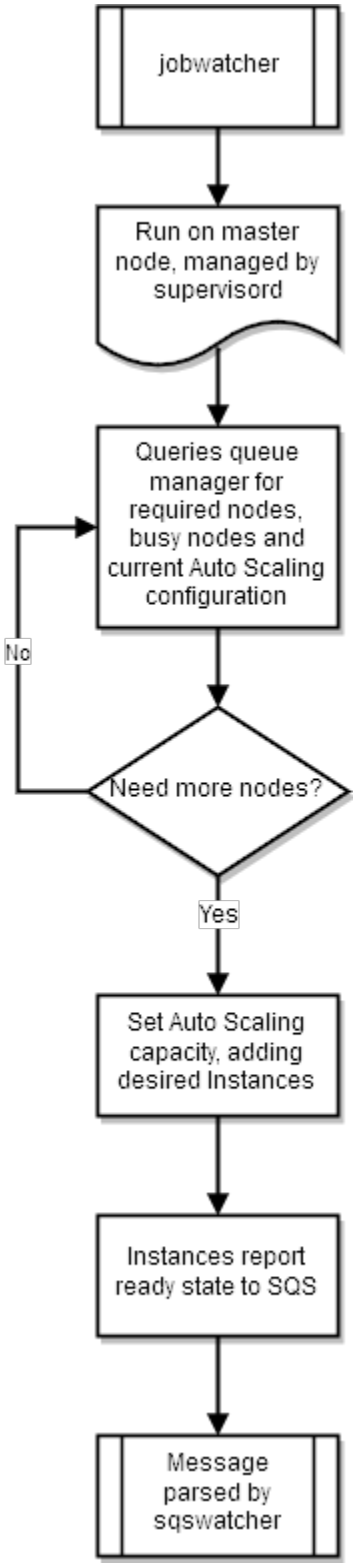
集群的生命周期在用户创建集群后开始。通常，从命令行界面 (CLI) 创建集群。创建集群之后，集群将一直存在，直到被删除为止。AWS ParallelCluster 守护进程在集群节点上运行，主要是为了管理 HPC 集群的弹性。下图显示了用户工作流程和集群生命周期。以下各节描述用于管理集群的 AWS ParallelCluster 守护进程。


User creates a cluster
CloudFormation launches cluster
Cluster is running
sqswatcher
jobwatcher
nodewatcher
Cluster Management Processes
User submits a job
Jobs are processed
User deletes cluster
CloudFormation deletes stack
Cluster is

对于 SGE 和 Torque 调度器，AWS ParallelCluster 使用 `nodewatcher`、`jobwatcher` 和 `sqswatcher` 进程。

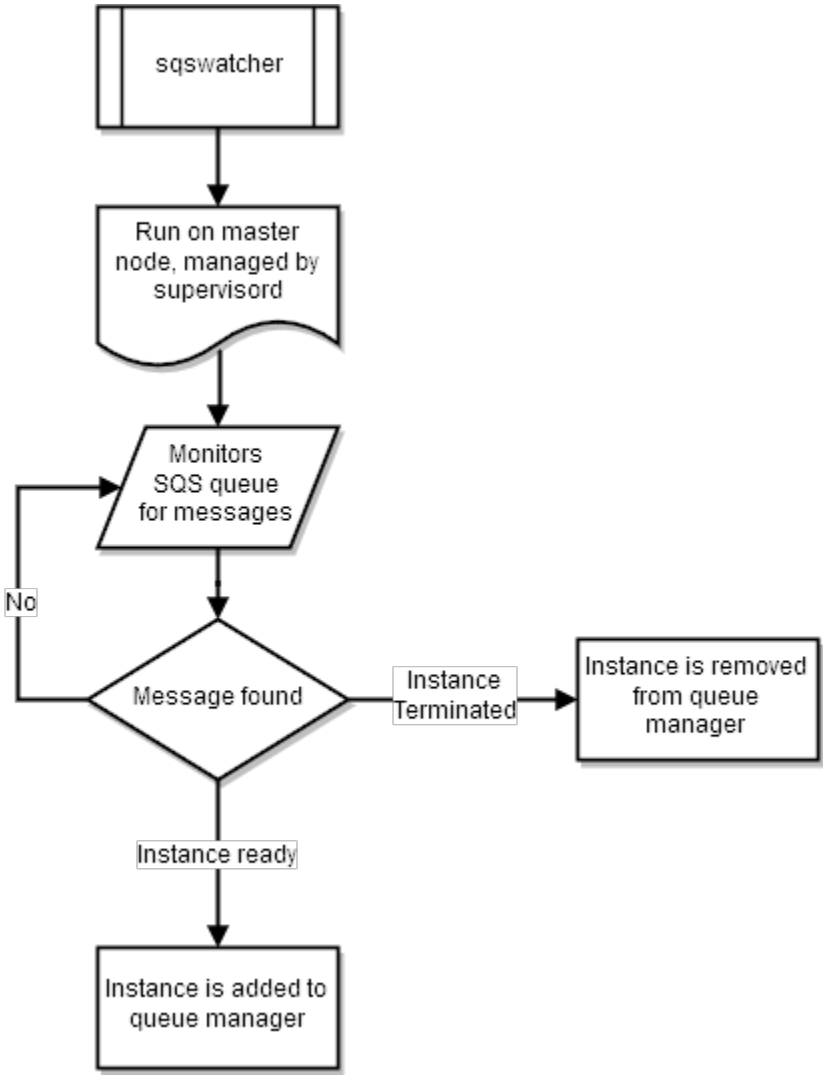
`jobwatcher`

当集群正在运行时，根用户拥有的进程会监控配置的调度器（SGE 或 Torque）。它每分钟评估一次队列，以决定何时纵向扩展。


jobwatcher
Run on master node, managed by supervisord
Queries queue manager for required nodes, busy nodes and current Auto Scaling configuration
No
Need more nodes?
Yes
Set Auto Scaling capacity, adding desired Instances
Instances report ready state to SQS
Message parsed by sqswatcher

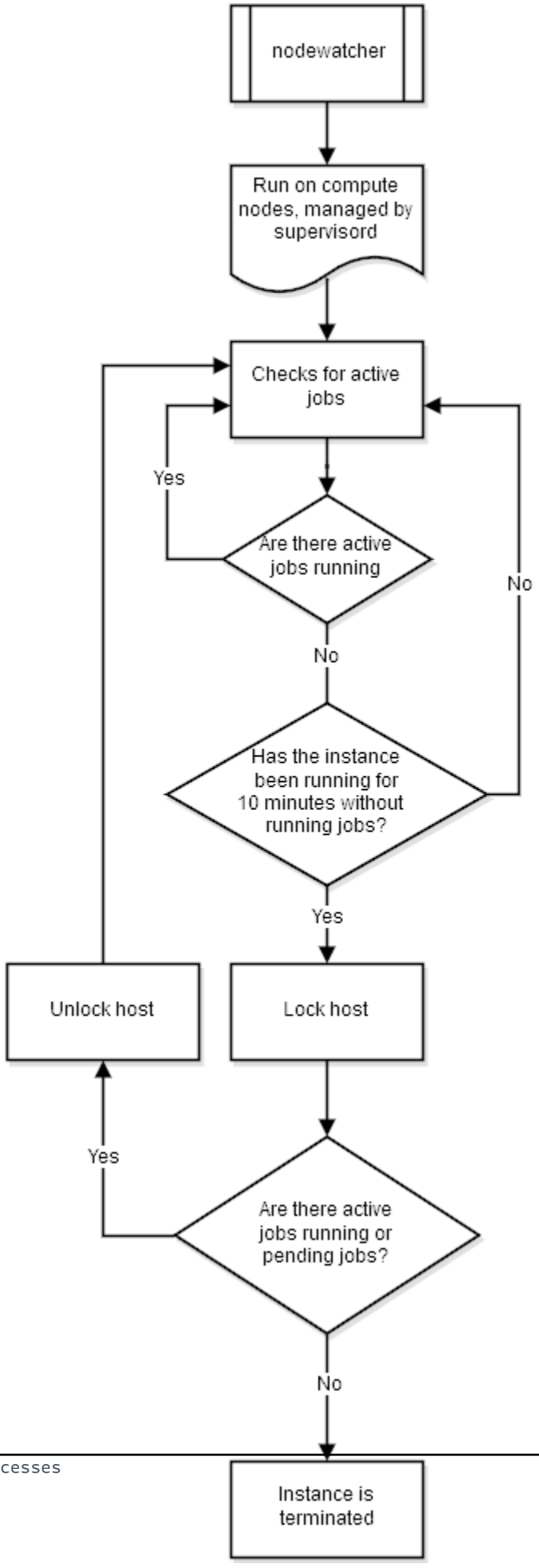
sqswatcher

sqswatcher 进程监控由 Auto Scaling 发送的 Amazon SQS 消息，这会告知您集群内的状态更改。当一个实例联机时，它会向 Amazon SQS 提交“实例就绪”消息。此消息由运行于头节点上的 sqs_watcher 接收。这些消息用于通知队列管理员有新实例联机或遭到终止，以便能够在队列中添加或删除它们。



nodewatcher

nodewatcher 进程在计算队列中的每个节点上运行。在用户定义的 scaledown_idletime 期间之后，实例将终止。


nodewatcher
Run on compute nodes, managed by supervisord
Checks for active jobs
Yes
Are there active jobs running
No
No
Has the instance been running for 10 minutes without running jobs?
Yes
Unlock host
Lock host
Yes
Are there active jobs running or pending jobs?
No
Instance is terminated

Slurm integration processes

对于 Slurm 调度器，AWS ParallelCluster 使用 `clustermgtd` 和 `computemgt` 进程。

clustermgtd

在异构模式（通过指定 `queue_settings` 值来指示）下运行的集群具有在头节点上运行的集群管理进程守护程序 (`clustermgtd`) 进程。以下任务由集群管理进程守护程序执行。

- 非活动分区清理
- 静态容量管理：确保静态容量始终处于正常运行状态
- 将调度器与 Amazon EC2 同步。
- 孤立实例清理
- 在暂停工作流之外发生 Amazon EC2 终止时还原调度器节点状态
- 不正常 Amazon EC2 实例管理（Amazon EC2 运行状况检查失败）
- 定期维护事件管理
- 不正常调度器节点管理（调度器运行状况检查失败）

computemgtd

在异构模式（通过指定 `queue_settings` 值来指示）下运行的集群具有在每个计算节点上运行的计算管理进程守护程序 (`computemgtd`) 进程。每隔五 (5) 分钟，计算管理进程守护程序就会确认头节点可以访问并且运行正常。如果在五 (5) 分钟内无法访问头节点或头节点运行状况不佳，则将关闭计算节点。

AWS 使用的服务 AWS ParallelCluster

使用以下亚马逊 Web Services (AWS) 服务 AWS ParallelCluster。

主题

- AWS Auto Scaling
- AWS Batch
- AWS CloudFormation
- Amazon CloudWatch

- Amazon CloudWatch 日志
- AWS CodeBuild
- Amazon DynamoDB
- Amazon Elastic Block Store
- Amazon Elastic Compute Cloud
- Amazon Elastic Container Registry
- Amazon EFS
- 亚马逊 f FSx or Lustre
- AWS Identity and Access Management
- AWS Lambda
- Amazon DCV
- Amazon Route 53
- Amazon Simple Notification Service
- Amazon Simple Queue Service
- Amazon Simple Storage Service
- Amazon VPC

AWS Auto Scaling

> **Note**
>
> 本节仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从 2.11.5 版开始， AWS ParallelCluster 不支持使用。 AWS Auto Scaling

AWS Auto Scaling 是一项监控您的应用程序并根据您的特定和不断变化的服务要求自动调整容量的服务。该服务以 Auto Scaling 组的形式管理您的 ComputeFleet 实例。该组可以由不断变化的工作负载弹性驱动，也可以由初始实例配置静态固定。

AWS Auto Scaling 用于 ComputeFleet 实例，但不适用于 AWS Batch 集群。

有关的更多信息 AWS Auto Scaling，请参见https://aws.amazon.com/autoscaling/和https://docs.aws.amazon.com/autoscaling/。

AWS Batch

AWS Batch 是一项 AWS 托管作业调度器服务。它可以动态配置集群中计算资源的最佳数量和类型（例如，CPU或内存优化型实例）。 AWS Batch 这些资源是根据批处理作业的特定要求（包括卷要求）预置的。使用 AWS Batch，您无需安装或管理额外的批处理计算软件或服务器集群即可有效地运行作业。

AWS Batch 仅与 AWS Batch 群集一起使用。

有关的更多信息 AWS Batch，请参见https://aws.amazon.com/batch/和https://docs.aws.amazon.com/batch/。

AWS CloudFormation

AWS CloudFormation 是一项为云环境中的第三方应用程序资源建模 AWS 和配置的通用语言的 infrastructure-as-code服务。它是使用的主要服务 AWS ParallelCluster。中的每个集群 AWS ParallelCluster 都表示为一个堆栈，每个群集所需的所有资源都在 AWS ParallelCluster AWS CloudFormation 模板中定义。在大多数情况下， AWS ParallelCluster CLI命令直接对应 AWS CloudFormation 堆栈命令，例如创建、更新和删除命令。在集群内启动的实例会HTTPS调用启动集群 AWS 区域 的 AWS CloudFormation 终端节点。

有关的更多信息 AWS CloudFormation，请参见https://aws.amazon.com/cloudformation/和https://docs.aws.amazon.com/cloudformation/。

Amazon CloudWatch

Amazon CloudWatch (CloudWatch) 是一项监控和可观察性服务，可为您提供数据和可操作的见解。这些见解可用于监控您的应用程序、响应性能变化和服务异常以及优化资源利用率。中 AWS ParallelCluster， CloudWatch 用于仪表板，用于监视和记录 Docker 映像构建步骤和 AWS Batch 作业输出。

在 2.10.0 AWS ParallelCluster 版本之前 CloudWatch ，仅用于集群。 AWS Batch

有关的更多信息 CloudWatch，请参见https://aws.amazon.com/cloudwatch/和https://docs.aws.amazon.com/cloudwatch/。

Amazon CloudWatch 日志

亚马逊 CloudWatch 日CloudWatch 志（日志）是亚马逊的核心功能之一 CloudWatch。您可以使用它来监控、存储、查看和搜索 AWS ParallelCluster中使用的众多组件的日志文件。

在 2.6.0 AWS ParallelCluster 版本之前， CloudWatch 日志仅用于集群。 AWS Batch

有关更多信息，请参阅 与 Amazon CloudWatch 日志集成。

AWS CodeBuild

AWS CodeBuild (CodeBuild) 是一项 AWS 托管的持续集成服务，它符合源代码、运行测试并生成随时可以部署的软件包。中 AWS ParallelCluster， CodeBuild 用于在创建集群时自动透明地构建 Docker 镜像。

CodeBuild 仅与 AWS Batch 群集一起使用。

有关的更多信息 CodeBuild，请参见https://aws.amazon.com/codebuild/和https://docs.aws.amazon.com/codebuild/。

Amazon DynamoDB

Amazon DynamoDB (DynamoDB) 是一项快速灵活的无数据库服务。SQL它用于存储集群的最小状态信息。头节点跟踪 DynamoDB 表中的预置实例。

DynamoDB 不适用于集群。 AWS Batch

有关 DynamoDB 的更多信息，请参阅和。https://aws.amazon.com/dynamodb/https://docs.aws.amazon.com/dynamodb/

Amazon Elastic Block Store

Amazon Elastic Block Store (AmazonEBS) 是一项高性能的块存储服务，可为共享卷提供永久存储。所有 Amazon EBS 设置都可以通过配置传递。可以将亚马逊EBS卷初始化为空，也可以从现有的亚马逊EBS快照中初始化。

有关 Amazon 的更多信息EBS，请参阅https://aws.amazon.com/ebs/和https://docs.aws.amazon.com/ebs/。

Amazon Elastic Compute Cloud

亚马逊弹性计算云 (AmazonEC2) 为提供计算容量 AWS ParallelCluster。头节点和计算节点是 Amazon EC2 实例。HVM可以选择支持的任何实例类型。头节点和计算节点可以是不同的实例类型。此外，如果使用多个队列，则部分或全部计算节点也可以作为竞价型实例启动。在实例上找到的实例存储卷将作为条带LVM卷装载。

有关 Amazon 的更多信息EC2，请参阅https://aws.amazon.com/ec2/和https://docs.aws.amazon.com/ec2/。

Amazon Elastic Container Registry

Amazon Elastic Container Registry（亚马逊ECR）是一个完全托管的 Docker 容器注册表，可轻松存储、管理和部署 Docker 容器镜像。在中 AWS ParallelCluster，Amazon ECR 存储了创建集群时生成的 Docker 镜像。然后，使用 Docker 镜像为提交的作业运行容器。 AWS Batch

Amazon ECR 仅用于 AWS Batch 集群。

有关更多信息，请参阅https://aws.amazon.com/ecr/和https://docs.aws.amazon.com/ecr/。

Amazon EFS

Amazon Elastic File System (AmazonEFS) 提供了一个简单、可扩展且完全托管的弹性NFS文件系统，用于 AWS Cloud 服务和本地资源。EFSAmazon 在指定`efs_settings`设置时使用，并且指的是某个`[efs]`部分。在 2.1.0 AWS ParallelCluster 版本中增加了EFS对亚马逊的支持。

有关 Amazon 的更多信息EFS，请参阅https://aws.amazon.com/efs/和https://docs.aws.amazon.com/efs/。

亚马逊 f FSx or Lustre

FSxfor Lustre 提供了一个使用开源 Lustre 文件系统的高性能文件系统。FSxfor Lustre 在指定`fsx_settings`设置时使用，并且指的是某个`[fsx]`部分。在 2.2.1 AWS ParallelCluster 版本中增加了对 Lustre 的支持。FSx

有关 Lustre FSx 的更多信息，请参阅 lust https://aws.amazon.com/fsx/re/ 和。https://docs.aws.amazon.com/fsx/

AWS Identity and Access Management

AWS Identity and Access Management (IAM) 在中 AWS ParallelCluster 用于为 Amazon EC2 提供特定于每个集群的实例的最低权限IAM角色。 AWS ParallelCluster 实例只能访问部署和管理集群所需的特定API调用。

对于 AWS Batch 集群，还会在创建集群时为与 Docker 镜像构建过程相关的组件创建IAM角色。这些组件包括允许在亚马逊存储库中添加和删除 Docker 镜像的 Lambda 函数。ECR它们还包括允许删除

为集群和 CodeBuild 项目创建的 Amazon S3 存储桶的功能。还有 AWS Batch 资源、实例和作业的角色。

有关的更多信息IAM，请参见https://aws.amazon.com/iam/和https://docs.aws.amazon.com/iam/。

AWS Lambda

AWS Lambda (Lambda) 运行编排 Docker 镜像创建的函数。Lambda 还管理自定义集群资源的清理，例如存储在亚马逊ECR存储库和亚马逊 S3 上的 Docker 镜像。

有关 Lambda 的更多信息，请参阅https://aws.amazon.com/lambda/和。https://docs.aws.amazon.com/lambda/

Amazon DCV

Amazon DCV 是一种高性能远程显示协议，它提供了一种在不同网络条件下将远程桌面和应用程序流传输到任何设备的安全方式。DCVAmazon 在指定`dcv_settings`设置时使用，并且指的是某个`[dcv]`部分。2.5.0 AWS ParallelCluster 版本中增加了DCV对亚马逊的支持。

有关 Amazon 的更多信息DCV，请参阅 https://aws.amazon.com/hpc/dcv/ 和。https://docs.aws.amazon.com/dcv/

Amazon Route 53

Amazon Route 53 (Route 53) 用于使用每个计算节点的主机名和完全限定域名创建托管区。

有关 Route 53 的更多信息，请参阅https://aws.amazon.com/route53/和https://docs.aws.amazon.com/route53/。

Amazon Simple Notification Service

> **Note**
>
> 本节仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从版本 2.11.5 开始， AWS ParallelCluster 不支持使用 Amazon Simple Notification Service。

亚马逊简单通知服务（亚马逊SNS）接收来自 Auto Scaling 的通知。这些事件称为生命周期事件，它们是当实例在自动扩缩组中启动或终止时生成的。在内 AWS ParallelCluster，Auto Scaling 群组的亚马逊SNS主题已订阅到亚马逊SQS队列。

Amazon SNS 不适用于 AWS Batch 集群。

有关 Amazon 的更多信息SNS，请参阅https://aws.amazon.com/sns/和https://docs.aws.amazon.com/sns/。

Amazon Simple Queue Service

> **Note**
>
> 本节仅适用于 AWS ParallelCluster 2.11.4 及以下的版本。从版本 2.11.5 开始， AWS ParallelCluster 不支持使用 Amazon Simple Queue Service。

亚马逊简单队列服务 (AmazonSQS) 保存从 Auto Scaling 发送的通知SNS、通过亚马逊发送的通知以及从计算节点发送的通知。Ama SQS zon 将发送通知与接收通知分开。这使头节点能够通过轮询过程处理通知。在此过程中，头节点运行 Amazon SQSwatcher 并轮询队列。自动扩缩和计算节点向该队列发布消息。

Amazon SQS 不适用于 AWS Batch 集群。

有关 Amazon 的更多信息SQS，请参阅https://aws.amazon.com/sqs/和https://docs.aws.amazon.com/sqs/。

Amazon Simple Storage Service

亚马逊简单存储服务 (Amazon S3) Service 存储的模板位于 AWS ParallelCluster 每个服务中。 AWS 区域 AWS ParallelCluster 可以配置为允许CLI/SDK工具使用 Amazon S3。

当您使用 AWS Batch 集群时，将使用您账户中的 Amazon S3 存储桶来存储相关数据。例如，该存储桶会存储根据提交的作业创建 Docker 映像和脚本时创建的构件。

有关更多信息，请参阅https://aws.amazon.com/s3/和https://docs.aws.amazon.com/s3/。

Amazon VPC

Amazon VPC 定义了您的集群中节点使用的网络。集群的VPC设置在[vpc]部分中定义。

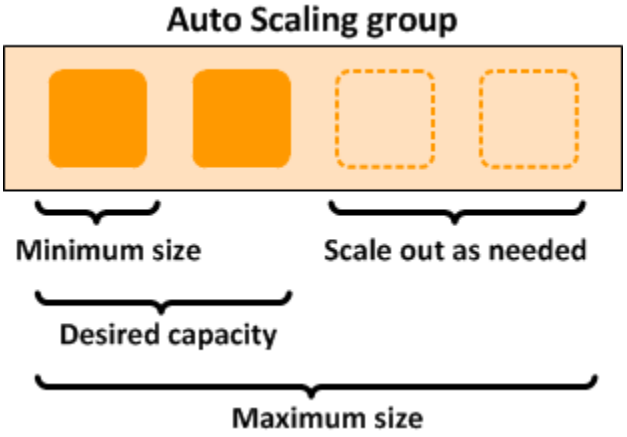
有关 Amazon 的更多信息VPC，请参阅https://aws.amazon.com/vpc/和https://docs.aws.amazon.com/vpc/。

AWS ParallelCluster Auto Scaling

> **Note**
>
> 本节仅适用于 2.11.4 及以下的 AWS ParallelCluster 版本。从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 或 Torque 调度器。您可以在 2.11.4 及之前的版本中继续使用这些调度器，但它们没有资格获得 AWS 服务和 AWS 支持团队的未来更新或故障排除支持。从 AWS ParallelCluster 版本 2.9.0 开始，不支持将自动扩缩与 Slurm Workload Manager (Slurm) 一起使用。要了解有关 Slurm 和多队列扩展的信息，请参阅多队列模式教程。

本主题中介绍的自动扩缩策略适用于使用 Son of Grid Engine (SGE) 或 Torque Resource Manager (Torque) 部署的 HPC 集群。在使用其中一个调度器进行部署时，AWS ParallelCluster 通过管理计算节点的自动扩缩组，然后根据需要更改调度器配置来实现扩展功能。对于基于 AWS Batch 的 HPC 集群，AWS ParallelCluster 依赖于由 AWS 托管的作业调度器提供的弹性扩展功能。有关更多信息，请参阅 Amazon EC2 Auto Scaling User Guide 中的 What is Amazon EC2 Auto Scaling。

使用 AWS ParallelCluster 部署的集群在多个方面是弹性的。设置 `initial_queue_size` 可指定 ComputeFleet 自动扩缩组的最小大小值以及所需的容量值。设置 `max_queue_size` 可指定 ComputeFleet 自动扩缩组的最大大小值。



纵向扩展

名为 `jobwatcher` 的进程时刻在头节点上运行。它评估队列中的待处理作业所需的当前实例数。如果繁忙节点和所请求节点的总数大于自动扩缩组中当前需要的值，则会添加更多实例。如果您提交更多作业，则会重新评估队列并更新自动扩缩组（最多为指定的 `max_queue_size`）。

借助 SGE 计划程序，每个作业需要运行许多槽（一个槽对应于一个处理单元，例如一个 vCPU）。为了评估处理当前的待处理作业所需的实例数，jobwatcher 会将所请求槽的总数除以单个计算节点的容量。与可用 vCPU 的数量对应的计算节点容量依赖于集群配置中指定的 Amazon EC2 实例类型。

借助 Slurm（在 AWS ParallelCluster 版本 2.9.0 之前）和 Torque 调度器，每个作业可能需要多个节点，每个节点可能需要多个槽，取决于具体情况。对于每个请求，jobwatcher 将确定为满足新的计算要求而需要的计算节点数。例如，假设一个集群的计算实例类型为 c5.2xlarge (8 vCPU)，并且有三个已排队的待处理作业，要求如下：

- 作业 1：2 个节点/每个节点 4 个插槽
- 作业 2：3 个节点/每个节点 2 个插槽
- 作业 3：1 个节点/每个节点 4 个插槽

在此示例中，jobwatcher 需要自动扩缩组中的三个新计算实例来处理这三个作业。

当前限制：自动纵向扩展逻辑不考虑部分加载的繁忙节点。例如，正在运行作业的节点即使有空槽，也会被视为忙碌。

缩减

在每个计算节点上，都有一个名为 nodewatcher 的进程运行并评估节点的空闲时间。在满足以下两个条件时，将终止实例：

- 实例不具有作业的时长已超过 scaledown_idletime（默认设置为 10 分钟）
- 集群中没有待处理作业

为了终止实例，nodewatcher 将调用 TerminateInstanceInAutoScalingGroup API 操作，这将在自动扩缩组的大小至少为最小自动扩缩组大小时删除实例。此过程将收缩集群，而不会影响正在运行的作业。它还支持弹性集群，具有固定的实例基数。

静态集群

对于 HPC，Auto Scaling 的值与任何其他工作负载的相同。唯一的区别是，AWS ParallelCluster 的代码使其能够更智能地交互。例如，如果需要静态集群，请将 initial_queue_size 和 max_queue_size 参数设置为所需集群的准确大小，然后将 maintain_initial_size 参数设置为 true。这将导致 ComputeFleet 自动扩缩组对于最小、最大和所需容量具有相同的值。

教程

以下教程介绍如何开始使用 AWS ParallelCluster，并提供一些常见任务的最佳实践指导。

主题

- 在 AWS ParallelCluster 上运行首个作业
- 构建自定义 AWS ParallelCluster AMI
- 使用 AWS ParallelCluster 和 awsbatch 调度器运行 MPI 作业
- 使用自定义 KMS 密钥对磁盘加密
- 多队列模式教程

在 AWS ParallelCluster 上运行首个作业

本教程将引导您完成在 AWS ParallelCluster 上运行第一个 Hello World 作业的过程。

先决条件

- AWS ParallelCluster已安装 ???。
- 已安装并配置 AWS CLI。
- 您拥有 EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

验证安装

首先，我们验证 AWS ParallelCluster 是否已正确安装和配置。

```
$ pcluster version
```

这将返回正在运行的 AWS ParallelCluster 版本。如果输出为您提供有关配置的消息，您将需要运行以下命令来配置 AWS ParallelCluster：

```
$ pcluster configure
```

创建您的第一个集群

现在应该创建您的第一个集群了。由于本教程的工作负载不是性能密集型的，因此，我们可以使用 t2.micro 的默认实例大小。（对于生产工作负载，您需要选择最适合您的需求的实例大小。）

我们将您的集群称作 hello-world。

```
$ pcluster create hello-world
```

创建集群时，您会看到类似以下内容的输出：

```
Starting: hello-world
Status: parallelcluster-hello-world - CREATE_COMPLETE
MasterPublicIP = 54.148.x.x
ClusterUser: ec2-user
MasterPrivateIP = 192.168.x.x
GangliaPrivateURL = http://192.168.x.x/ganglia/
GangliaPublicURL = http://54.148.x.x/ganglia/
```

消息 CREATE_COMPLETE 显示该集群已成功创建。输出还为我们提供头节点的公有 IP 地址和私有 IP 地址。我们需要此 IP 以进行登录。

登录到头节点

使用您的 OpenSSH pem 文件登录到头节点。

```
pcluster ssh hello-world -i /path/to/keyfile.pem
```

登录后，请运行命令 qhost 以验证您的计算节点是否已设置和配置。

```
$ qhost
HOSTNAME                ARCH         NCPU NSOC NCOR NTHR  LOAD  MEMTOT  MEMUSE  SWAPTO
 SWAPUS
----------------------------------------------------------------------------------------------
global                  -               -    -    -    -     -       -       -       -
      -
ip-192-168-1-125        lx-amd64        2    1    2    2  0.15    3.7G  130.8M 1024.0M
    0.0
ip-192-168-1-126        lx-amd64        2    1    2    2  0.15    3.7G  130.8M 1024.0M
    0.0
```

输出显示我们的集群中有两个计算节点，两者都有 2 个线程可用。

使用 SGE 运行首个作业

> **Note**
>
> 此示例仅适用于版本 2.11.4 及以前的 AWS ParallelCluster 版本。从版本 2.11.5 开始，AWS ParallelCluster 不支持使用 SGE 或 Torque 调度器。

接下来，我们将创建一个作业，该作业睡眠一小段时间，然后输出它自己的主机名。

使用以下内容创建名为 `hellojob.sh` 的文件。

```
#!/bin/bash
sleep 30
echo "Hello World from $(hostname)"
```

接下来，使用 `qsub` 提交作业，并验证其是否运行。

```
$ qsub hellojob.sh
Your job 1 ("hellojob.sh") has been submitted
```

现在，您可以查看您的队列并检查该作业的状态。

```
$ qstat
job-ID  prior   name       user         state submit/start at     queue
          slots ja-task-ID
-----------------------------------------------------------------------------------------------
      1 0.55500 hellojob.s ec2-user     r     03/24/2015 22:23:48
 all.q@ip-192-168-1-125.us-west     1
```

输出显示此作业目前处于运行状态。请等候 30 秒，以便作业完成，然后再次运行 `qstat`。

```
$ qstat
$
```

现在，队列中没有作业，我们可以检查当前目录中的输出。

```
$ ls -l
```

```
total 8
-rw-rw-r-- 1 ec2-user ec2-user 48 Mar 24 22:34 hellojob.sh
-rw-r--r-- 1 ec2-user ec2-user  0 Mar 24 22:34 hellojob.sh.e1
-rw-r--r-- 1 ec2-user ec2-user 34 Mar 24 22:34 hellojob.sh.o1
```

在输出中，我们会看到作业脚本中的“e1”和“o1”文件。由于 e1 文件为空，因此没有内容输出到 stderr。如果我们查看 o1 文件，我们可以看到作业的输出。

```
$ cat hellojob.sh.o1
Hello World from ip-192-168-1-125
```

输出还显示我们的作业已在实例 ip-192-168-1-125 上成功运行。

要了解有关创建和使用集群的更多信息，请参阅最佳实践。

构建自定义 AWS ParallelCluster AMI

Important

我们不建议将构建自定义 AMI 作为自定义 AWS ParallelCluster 的方法。
这是因为，在构建您自己的 AMI 后，您就不再随 AWS ParallelCluster 的将来版本收到更新或错误修复。此外，如果构建自定义 AMI，则必须重复执行用于随每个新的 AWS ParallelCluster 版本创建自定义 AMI 的步骤。

在继续阅读之前，我们建议您先查看自定义引导操作部分，以确定您希望进行的更改能否使用将来的 AWS ParallelCluster 版本写入脚本以及是否受这些版本支持。

虽然构建自定义 AMI 并不是理想情况（出于前文中提及的原因），但是在某些场景中，仍需要为 AWS ParallelCluster 构建自定义 AMI。本教程将指导您完成针对这些场景构建自定义 AMI 的过程。

Note

从 AWS ParallelCluster 版本 2.6.1 开始，在启动节点时将默认跳过大多数安装食谱。这样可以缩短启动时间。要以牺牲启动时间为代价运行所有安装食谱以获得更好的向后兼容性，请将 "skip_install_recipes" : "no" 添加到 extra_json 设置中的 cluster 键。例如：

```
extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
```

先决条件

- AWS ParallelCluster已安装 ???。
- 已安装并配置 AWS CLI。
- 您拥有 EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

如何自定义 AWS ParallelCluster AMI

使用后面各节中所述的自定义 AWS ParallelCluster AMI 有三种方式。这三种方式中的两种方式需要您在 AWS 账户下构建新的可用 AMI。第三种方法（在运行时使用自定义 AMI）不需要您提前构建任何对象，但确实会增加部署风险。请选择最适合您需求的方法。

修改 AMI

这是推荐使用的最安全方法。由于基础 AWS ParallelCluster AMI 通常随新版本进行更新，因此该 AMI 具有安装和配置后运行 AWS ParallelCluster 所需的所有组件。您可以此为基础开始操作。

New EC2 console

1. 在 AWS ParallelCluster AMI 列表中，找到与您使用的特定 AWS 区域对应的 AMI。您选择的 AMI 列表必须与您使用的 AWS ParallelCluster 版本相匹配。运行 `pcluster version` 验证版本。对于 AWS ParallelCluster 版本 2.11.9，请转到 https://github.com/aws/aws-parallelcluster/blob/v2.11.9/amis.txt。要选择其他版本，请使用相同的链接，选择标签：2.11.9 按钮，选择标签选项卡，然后选择相应的版本。
2. 登录到 AWS Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
3. 在 Amazon EC2 控制面板上，选择启动实例。
4. 在应用程序和操作系统映像中，选择浏览其他 AMI，导航到社区 AMI，然后将您的 AWS 区域的 AWS ParallelCluster AMI ID 输入到搜索框中。
5. 选择此 AMI，选择您的实例类型和属性，选择您的密钥对，然后选择启动实例。
6. 使用操作系统用户和您的 SSH 密钥登录您的实例。有关更多信息，请导航至实例，选择新实例，然后选择连接。
7. 根据需要自定义您的实例。
8. 运行以下命令以准备实例来创建 AMI：

```
sudo /usr/local/sbin/ami_cleanup.sh
```

9. 导航到实例，选择新实例，然后依次选择实例状态和停止实例。
10. 使用 EC2 控制台或 AWS CLI create-image 根据该实例创建新 AMI。

    在 EC2 控制台中

    a. 在导航窗格中选择实例。

    b. 选择您创建和修改的实例。

    c. 在操作中，选择映像和模板，然后选择创建映像。

    d. 选择创建映像。
11. 在集群配置内的 custom_ami 字段中输入新 AMI ID。

Old EC2 console

1. 在 AWS ParallelCluster AMI 列表中，找到与您使用的特定 AWS 区域对应的 AMI。您选择的 AMI 列表必须与您使用的 AWS ParallelCluster 版本相匹配。运行 `pcluster version` 验证版本。对于 AWS ParallelCluster 版本 2.11.9，请转到 https://github.com/aws/aws-parallelcluster/blob/v2.11.9/amis.txt。要选择其他版本，请使用相同的链接，选择标签：2.11.9 按钮，选择标签选项卡，然后选择相应的版本。
2. 登录到 AWS Management Console 并打开 Amazon EC2 控制台（https://console.aws.amazon.com/ec2/）。
3. 在 Amazon EC2 控制面板上，选择启动实例。
4. 选择社区 AMI，搜索 AWS ParallelCluster AMI ID，然后选择该 AMI。
5. 选择您的实例类型，然后选择下一步：配置实例详细信息或查看并启动以启动您的实例。
6. 选择启动，选择您的密钥对，然后选择启动实例。
7. 使用操作系统用户和您的 SSH 密钥登录您的实例。有关更多信息，请导航至实例，选择新实例，然后选择连接。
8. 根据需要自定义您的实例。
9. 运行以下命令以准备实例来创建 AMI：

```
sudo /usr/local/sbin/ami_cleanup.sh
```

10. 导航到实例，选择新实例，然后依次选择实例状态和停止。

11.使用 EC2 控制台或 AWS CLI create-image 根据该实例创建新 AMI。

在 EC2 控制台中

a. 在导航窗格中选择实例。

b. 选择您创建和修改的实例。

c. 在操作中，依次选择映像和创建映像。

d. 选择创建映像。

12.在集群配置内的 custom_ami 字段中输入新 AMI ID。

构建自定义 AWS ParallelCluster AMI

如果您已具有自定义的 AMI 和软件，则可以在其基础之上应用 AWS ParallelCluster 所需的更改。

1. 连同 AWS ParallelCluster CLI 一起，在您的本地系统中安装以下工具：
   - Packer：从 Packer 网站查找并安装最新的操作系统版本。版本必须至少为 1.4.0，但建议使用最新版本。验证 `packer` 命令是否在您的 PATH 中可用。

   > **Note**
   >
   > 在 AWS ParallelCluster 版本 2.8.0 之前，必须安装 Berkshelf（使用 `gem install berkshelf` 进行安装）才能使用 `pcluster createami`。

2. 配置您的 AWS 账户凭证，以便 Packer 可以代表您调用 AWS API 操作。Packer 工作所需的一组最低权限记录在 Packer 文档 Amazon AMI Builder 主题的 IAM Task or Instance Role 部分。
3. 可以使用 AWS ParallelCluster CLI 中的 `createami` 命令以您作为基础的 AWS ParallelCluster AMI 为起点构建该 AMI：

   ```
   pcluster createami --ami-id <BASE_AMI> --os <BASE_AMI_OS>
   ```

   > **Important**
   >
   > 您不应将正在运行的集群中的 AWS ParallelCluster AMI 用作 `createami` 命令的 *<BASE_AMI>*。否则，该命令将失败。

   对于其他参数，请参阅 `pcluster createami`。

4. 步骤 4 中的命令运行 Packer，后者具体执行以下操作：
   a. 使用提供的基础 AMI 启动实例。
   b. 将 AWS ParallelCluster 说明书应用于实例，以便安装相关软件和执行其他必要的配置任务。
   c. 停止实例。
   d. 从实例创建新的 AMI。
   e. 创建 AMI 之后终止该实例。
   f. 输出用于创建集群的新的 AMI ID 字符串。
5. 要创建集群，请在集群配置内的 custom_ami 字段中输入 AMI ID。

> **Note**
> 用于构建自定义 AWS ParallelCluster AMI 的实例类型是 `t2.xlarge`。此实例类型不符合 AWS 免费套餐的资格，因此您需要为构建此 AMI 时创建的任何实例付费。

在运行时使用自定义 AMI

> **Warning**
> 为避免对 AWS ParallelCluster 使用不兼容 AMI 的风险，我们建议您避免使用此方法。
> 当在运行时使用可能未经测试的 AMI 启动计算节点时，与 AWS ParallelCluster 所需的运行时安装软件不兼容可能会导致 AWS ParallelCluster 停止运行。

如果您不想提前创建任何内容，则可以使用您的 AMI 并基于该 AMI 创建 AWS ParallelCluster。

使用这种方法，创建 AWS ParallelCluster 所需的时间会更长，因为必须要安装创建集群时 AWS ParallelCluster 所需的所有软件。此外，纵向扩展也需要更长的时间。

- 在集群配置内的 custom_ami 字段中输入 AMI ID。

使用 AWS ParallelCluster 和 **`awsbatch`** 调度器运行 MPI 作业

本教程将指导您完成使用 `awsbatch` 作为计划程序运行 MPI 作业的过程。

先决条件

- AWS ParallelCluster已安装 ???。
- 已安装并配置 AWS CLI。
- 您拥有 EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

创建集群

首先，我们为使用 `awsbatch` 作为计划程序的集群创建一个配置。确保将 `vpc` 部分和 `key_name` 字段中的缺失数据与配置时创建的资源一起插入。

```
[global]
sanity_check = true

[aws]
aws_region_name = us-east-1

[cluster awsbatch]
base_os = alinux
# Replace with the name of the key you intend to use.
key_name = key-#######
vpc_settings = my-vpc
scheduler = awsbatch
compute_instance_type = optimal
min_vcpus = 2
desired_vcpus = 2
max_vcpus = 24

[vpc my-vpc]
# Replace with the id of the vpc you intend to use.
vpc_id = vpc-#######
# Replace with id of the subnet for the Head node.
master_subnet_id = subnet-#######
# Replace with id of the subnet for the Compute nodes.
# A NAT Gateway is required for MNP.
compute_subnet_id = subnet-#######
```

现在，您可以开始创建集群了。我们将创建的集群称为 *awsbatch-tutorial*。

```
$ pcluster create -c /path/to/the/created/config/aws_batch.config -t awsbatch awsbatch-tutorial
```

创建集群时，您会看到类似以下内容的输出：

```
Beginning cluster creation for cluster: awsbatch-tutorial
Creating stack named: parallelcluster-awsbatch
Status: parallelcluster-awsbatch - CREATE_COMPLETE
MasterPublicIP: 54.160.xxx.xxx
ClusterUser: ec2-user
MasterPrivateIP: 10.0.0.15
```

登录到头节点

AWS ParallelCluster Batch CLI 命令在安装了 AWS ParallelCluster 的客户端计算机上可用。但是，我们将通过 SSH 进入头节点并从那里提交作业。这使我们能够利用在头实例和所有运行 AWS Batch 作业的 Docker 实例之间共享的 NFS 卷。

使用您的 SSH pem 文件登录到头节点。

```
$ pcluster ssh awsbatch-tutorial -i /path/to/keyfile.pem
```

登录后，请运行命令 `awsbqueues` 和 `awsbhosts`，以显示配置的 AWS Batch 队列和正在运行的 Amazon ECS 实例。

```
[ec2-user@ip-10-0-0-111 ~]$ awsbqueues
jobQueueName                      status
--------------------------------  --------
parallelcluster-awsbatch-tutorial  VALID

[ec2-user@ip-10-0-0-111 ~]$ awsbhosts
ec2InstanceId        instanceType    privateIpAddress    publicIpAddress
 runningJobs
-------------------  --------------  ------------------  -----------------
 -------------
i-0d6a0c8c560cd5bed  m4.large        10.0.0.235          34.239.174.236
 0
```

如您在输出中看到的，我们有一个正在运行的主机。这是由于我们在配置中为 `min_vcpus` 选择的值导致的。如果要显示有关 AWS Batch 队列和主机的其他详细信息，请将 `-d` 标志添加到命令。

使用 AWS Batch 运行首个作业

在迁移到 MPI 之前，我们先创建一个虚拟作业，此作业休眠一小段时间，然后输出其自己的主机名，同时问候作为参数传递的名称。

使用以下内容创建名为“hellojob.sh”的文件。

```
#!/bin/bash

sleep 30
echo "Hello $1 from $HOSTNAME"
echo "Hello $1 from $HOSTNAME" > "/shared/secret_message_for_${1}_by_
${AWS_BATCH_JOB_ID}"
```

接下来，使用 awsbsub 提交作业并验证其是否运行。

```
$ awsbsub -jn hello -cf hellojob.sh Luca
Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2 (hello) has been submitted.
```

查看您的队列并检查该作业的状态。

```
$ awsbstat
jobId                                jobName      status    startedAt
 stoppedAt    exitCode
-------------------------------------  -----------  --------  -------------------
 -----------  ----------
6efe6c7c-4943-4c1a-baf5-edbfeccab5d2  hello        RUNNING   2018-11-12 09:41:29  -
        -
```

输出提供了改作业的详细信息。

```
$ awsbstat 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
jobId                    : 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
jobName                  : hello
createdAt                : 2018-11-12 09:41:21
startedAt                : 2018-11-12 09:41:29
stoppedAt                : -
status                   : RUNNING
statusReason             : -
jobDefinition            : parallelcluster-myBatch:1
jobQueue                 : parallelcluster-myBatch
```

```
command                   : /bin/bash -c 'aws s3 --region us-east-1 cp s3://
parallelcluster-mybatch-lui1ftboklhpns95/batch/job-hellojob_sh-1542015680924.sh /
tmp/batch/job-hellojob_sh-1542015680924.sh; bash /tmp/batch/job-
hellojob_sh-1542015680924.sh Luca'
exitCode                  : -
reason                    : -
vcpus                     : 1
memory[MB]                : 128
nodes                     : 1
logStream                 : parallelcluster-myBatch/default/c75dac4a-5aca-4238-
a4dd-078037453554
log                       : https://console.aws.amazon.com/cloudwatch/home?region=us-
east-1#logEventViewer:group=/aws/batch/job;stream=parallelcluster-myBatch/default/
c75dac4a-5aca-4238-a4dd-078037453554
-------------------------
```

请注意，作业当前处于 RUNNING 状态。请等候 30 秒，以便作业完成，然后再次运行 awsbstat。

```
$ awsbstat
jobId                                 jobName      status    startedAt
 stoppedAt    exitCode
------------------------------------  -----------  --------  -------------------
 -----------  ----------
```

现在，您可以看到作业处于 SUCCEEDED 状态。

```
$ awsbstat -s SUCCEEDED
jobId                                 jobName      status     startedAt
 stoppedAt              exitCode
------------------------------------  -----------  ---------  -------------------
 -------------------  ----------
6efe6c7c-4943-4c1a-baf5-edbfeccab5d2  hello        SUCCEEDED  2018-11-12 09:41:29
 2018-11-12 09:42:00           0
```

由于队列中现在没有作业，因此，我们可以通过 awsbout 命令检查输出。

```
$ awsbout 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
2018-11-12 09:41:29: Starting Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
download: s3://parallelcluster-mybatch-lui1ftboklhpns95/batch/job-
hellojob_sh-1542015680924.sh to tmp/batch/job-hellojob_sh-1542015680924.sh
2018-11-12 09:42:00: Hello Luca from ip-172-31-4-234
```

我们可以看到作业在实例“ip-172-31-4-234”上成功运行。

如果您查看 /shared 目录，您将找到您的私有消息。

要浏览本教程中未涵盖的所有可用功能，请参阅 AWS ParallelCluster Batch CLI 文档。在您准备好继续本教程后，我们继续并查看如何提交 MPI 作业。

在多节点并行环境中运行 MPI 作业

当仍然登录到头节点时，在 /shared 目录中创建一个名为 mpi_hello_world.c 的文件。将以下 MPI 程序添加到该文件中：

```
// Copyright 2011 www.mpitutorial.com
//
// An intro MPI hello world program that uses MPI_Init, MPI_Comm_size,
// MPI_Comm_rank, MPI_Finalize, and MPI_Get_processor_name.
//
#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();
}
```

现在，将以下代码保存为 submit_mpi.sh：

```
#!/bin/bash
echo "ip container: $(/sbin/ip -o -4 addr list eth0 | awk '{print $4}' | cut -d/ -f1)"
echo "ip host: $(curl -s "http://169.254.169.254/latest/meta-data/local-ipv4")"

# get shared dir
IFS=',' _shared_dirs=(${PCLUSTER_SHARED_DIRS})
_shared_dir=${_shared_dirs[0]}
_job_dir="${_shared_dir}/${AWS_BATCH_JOB_ID%#*}-${AWS_BATCH_JOB_ATTEMPT}"
_exit_code_file="${_job_dir}/batch-exit-code"

if [[ "${AWS_BATCH_JOB_NODE_INDEX}" -eq  "${AWS_BATCH_JOB_MAIN_NODE_INDEX}" ]]; then
    echo "Hello I'm the main node $HOSTNAME! I run the mpi job!"

    mkdir -p "${_job_dir}"

    echo "Compiling..."
    /usr/lib64/openmpi/bin/mpicc -o "${_job_dir}/mpi_hello_world" "${_shared_dir}/
mpi_hello_world.c"

    echo "Running..."
    /usr/lib64/openmpi/bin/mpirun --mca btl_tcp_if_include eth0 --allow-run-as-root --
machinefile "${HOME}/hostfile" "${_job_dir}/mpi_hello_world"

    # Write exit status code
    echo "0" > "${_exit_code_file}"
    # Waiting for compute nodes to terminate
    sleep 30
else
    echo "Hello I'm the compute node $HOSTNAME! I let the main node orchestrate the mpi
 processing!"
    # Since mpi orchestration happens on the main node, we need to make sure the
 containers representing the compute
    # nodes are not terminated. A simple trick is to wait for a file containing the
 status code to be created.
    # All compute nodes are terminated by AWS Batch if the main node exits abruptly.
    while [ ! -f "${_exit_code_file}" ]; do
```

```
        sleep 2
    done
    exit $(cat "${_exit_code_file}")
fi
```

我们现在已准备就绪，可以提交第一个 MPI 作业并使其在 3 个节点上并发运行：

```
$ awsbsub -n 3 -cf submit_mpi.sh
```

现在，我们监控作业状态并等待其进入 RUNNING 状态：

```
$ watch awsbstat -d
```

在作业进入 RUNNING 状态后，我们可以查看其输出。要显示主节点的输出，请将 #0 附加到作业 ID。要显示计算节点的输出，请使用 #1 和 #2：

```
[ec2-user@ip-10-0-0-111 ~]$ awsbout -s 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#0
2018-11-27 15:50:10: Job id: 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#0
2018-11-27 15:50:10: Initializing the environment...
2018-11-27 15:50:10: Starting ssh agents...
2018-11-27 15:50:11: Agent pid 7
2018-11-27 15:50:11: Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
2018-11-27 15:50:11: Mounting shared file system...
2018-11-27 15:50:11: Generating hostfile...
2018-11-27 15:50:11: Detected 1/3 compute nodes. Waiting for all compute nodes to
 start.
2018-11-27 15:50:26: Detected 1/3 compute nodes. Waiting for all compute nodes to
 start.
2018-11-27 15:50:41: Detected 1/3 compute nodes. Waiting for all compute nodes to
 start.
2018-11-27 15:50:56: Detected 3/3 compute nodes. Waiting for all compute nodes to
 start.
2018-11-27 15:51:11: Starting the job...
download: s3://parallelcluster-awsbatch-tutorial-iwyl4458saiwgwvg/batch/job-
submit_mpi_sh-1543333713772.sh to tmp/batch/job-submit_mpi_sh-1543333713772.sh
2018-11-27 15:51:12: ip container: 10.0.0.180
2018-11-27 15:51:12: ip host: 10.0.0.245
2018-11-27 15:51:12: Compiling...
2018-11-27 15:51:12: Running...
2018-11-27 15:51:12: Hello I'm the main node! I run the mpi job!
2018-11-27 15:51:12: Warning: Permanently added '10.0.0.199' (RSA) to the list of known
 hosts.
```

```
2018-11-27 15:51:12: Warning: Permanently added '10.0.0.147' (RSA) to the list of known
 hosts.
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-180.ec2.internal, rank 1 out
 of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-199.ec2.internal, rank 5 out
 of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-180.ec2.internal, rank 0 out
 of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-199.ec2.internal, rank 4 out
 of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-147.ec2.internal, rank 2 out
 of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-147.ec2.internal, rank 3 out
 of 6 processors

[ec2-user@ip-10-0-0-111 ~]$ awsbout -s 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#1
2018-11-27 15:50:52: Job id: 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#1
2018-11-27 15:50:52: Initializing the environment...
2018-11-27 15:50:52: Starting ssh agents...
2018-11-27 15:50:52: Agent pid 7
2018-11-27 15:50:52: Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
2018-11-27 15:50:52: Mounting shared file system...
2018-11-27 15:50:52: Generating hostfile...
2018-11-27 15:50:52: Starting the job...
download: s3://parallelcluster-awsbatch-tutorial-iwyl4458saiwgwvg/batch/job-
submit_mpi_sh-1543333713772.sh to tmp/batch/job-submit_mpi_sh-1543333713772.sh
2018-11-27 15:50:53: ip container: 10.0.0.199
2018-11-27 15:50:53: ip host: 10.0.0.227
2018-11-27 15:50:53: Compiling...
2018-11-27 15:50:53: Running...
2018-11-27 15:50:53: Hello I'm a compute node! I let the main node orchestrate the mpi
 execution!
```

我们现在可以确认作业已成功完成：

```
[ec2-user@ip-10-0-0-111 ~]$ awsbstat -s ALL
jobId                                    jobName        status     startedAt
 stoppedAt            exitCode
-------------------------------------  -------------  ---------  -------------------
 -------------------  ----------
5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d  submit_mpi_sh  SUCCEEDED  2018-11-27 15:50:10
 2018-11-27 15:51:26  -
```

注意：如果要在作业结束之前终止作业，您可以使用 `awsbkill` 命令。

使用自定义 KMS 密钥对磁盘加密

AWS ParallelCluster 支持配置选项 `ebs_kms_key_id` 和 `fsx_kms_key_id`。这些选项允许您为 Amazon EBS 磁盘加密或适用于 Lustre 的 FSx 提供自定义 AWS KMS。要使用这些选项，请指定一个 `ec2_iam_role`。

为了创建集群，AWS KMS 密钥需要知道集群的角色的名称。这可防止您使用在创建集群时创建的角色，而需要自定义的 `ec2_iam_role`。

先决条件

- AWS ParallelCluster已安装 ???。
- 已安装并配置 AWS CLI。
- 您拥有 EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

创建角色

首先，创建一个策略：

1. 转到 IAM 控制台：https://console.aws.amazon.com/iam/home。
2. 在策略下的创建策略中，单击 JSON 选项卡。
3. 作为策略的正文，粘贴实例策略。请务必替换 *<AWS ACCOUNT ID>* 和 *<REGION>* 的所有匹配项。
4. 将策略命名为 `ParallelClusterInstancePolicy`，然后单击创建策略。

接下来，创建角色：

1. 在角色下，创建一个角色。
2. 单击 EC2 作为可信实体。
3. 在权限下，搜索您刚创建的 `ParallelClusterInstancePolicy` 角色并附加此角色。
4. 将角色命名为 `ParallelClusterInstanceRole`，然后单击创建角色。

授予您的密钥权限

在 AWS KMS 控制台 > 客户托管密钥中，单击密钥的别名或密钥 ID。

单击密钥策略选项卡下方密钥用户框中的添加按钮，然后搜索您刚创建的 ParallelClusterInstanceRole。附加此角色。

创建集群

现在创建集群。以下是具有加密的 `Raid 0` 驱动器的集群示例：

```
[cluster default]
...
raid_settings = rs
ec2_iam_role = ParallelClusterInstanceRole

[raid rs]
shared_dir = raid
raid_type = 0
num_of_raid_volumes = 2
volume_size = 100
encrypted = true
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

以下是适用于 Lustre 的 FSx 文件系统的示例：

```
[cluster default]
...
fsx_settings = fs
ec2_iam_role = ParallelClusterInstanceRole

[fsx fs]
shared_dir = /fsx
storage_capacity = 3600
imported_file_chunk_size = 1024
export_path = s3://bucket/folder
import_path = s3://bucket
weekly_maintenance_start_time = 1:00:00
fsx_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

类似的配置适用于基于 Amazon EBS 和 Amazon FSx 的文件系统。

多队列模式教程

在具有多队列模式的 AWS ParallelCluster 上运行作业

本教程将引导您完成在 AWS ParallelCluster 上运行第一个 Hello World 作业的过程。

先决条件

- AWS ParallelCluster已安装 ???。
- 已安装并配置 AWS CLI。
- 您拥有 EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

> **Note**
>
> 仅对 AWS ParallelCluster 版本 2.9.0 或更高版本支持多队列模式。

配置集群

首先，通过运行以下命令，验证是否已正确安装 AWS ParallelCluster。

```
$ pcluster version
```

有关 `pcluster version` 的更多信息，请参阅 `pcluster version`。

此命令将返回正在运行的 AWS ParallelCluster 版本。

接下来，运行 `pcluster configure` 以生成基本配置文件。按照运行此命令后的所有提示进行操作。

```
$ pcluster configure
```

有关 `pcluster configure` 命令的更多信息，请参阅`pcluster configure`。

完成此步骤后，`~/.parallelcluster/config` 下应该有一个基本配置文件。此文件应包含基本集群配置和 VPC 部分。

本教程的下一部分将概述如何修改新创建的配置以及如何启动具有多个队列的集群。

> **Note**
>
> 本教程中使用的某些实例不符合免费套餐资格。

在本教程中，使用以下配置。

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue

[aws]
aws_region_name = <Your AWS ##>

[scaling demo]
scaledown_idletime = 5              # optional, defaults to 10 minutes

[cluster multi-queue-special]
key_name = < Your key name >
base_os = alinux2                   # optional, defaults to alinux2
scheduler = slurm
master_instance_type = c5.xlarge    # optional, defaults to t2.micro
vpc_settings = <Your VPC section>
scaling_settings = demo             # optional, defaults to no custom scaling settings
queue_settings = efa,gpu

[cluster multi-queue]
key_name = <Your SSH key name>
base_os = alinux2                   # optional, defaults to alinux2
scheduler = slurm
master_instance_type = c5.xlarge    # optional, defaults to t2.micro
vpc_settings = <Your VPC section>
scaling_settings = demo
queue_settings = spot,ondemand

[queue spot]
compute_resource_settings = spot_i1,spot_i2
compute_type = spot                 # optional, defaults to ondemand

[compute_resource spot_i1]
```

```
instance_type = c5.xlarge
min_count = 0                       # optional, defaults to 0
max_count = 10                      # optional, defaults to 10

[compute_resource spot_i2]
instance_type = t2.micro
min_count = 1
initial_count = 2

[queue ondemand]
compute_resource_settings = ondemand_i1
disable_hyperthreading = true       # optional, defaults to false

[compute_resource ondemand_i1]
instance_type = c5.2xlarge
```

创建集群

本节详细介绍如何创建多队列模式集群。

首先，将您的集群命名为 multi-queue-hello-world，然后根据上一节中定义的 multi-queue 集群部分创建集群。

```
$ pcluster create multi-queue-hello-world -t multi-queue
```

有关 pcluster create 的更多信息，请参阅 pcluster create。

创建集群后，将显示以下输出：

```
Beginning cluster creation for cluster: multi-queue-hello-world
Creating stack named: parallelcluster-multi-queue-hello-world
Status: parallelcluster-multi-queue-hello-world - CREATE_COMPLETE
MasterPublicIP: 3.130.xxx.xx
ClusterUser: ec2-user
MasterPrivateIP: 172.31.xx.xx
```

消息 CREATE_COMPLETE 指示已成功创建该集群。输出还提供头节点的公有 IP 地址和私有 IP 地址。

登录到头节点

使用您的私有 SSH 密钥文件登录到头节点。

```
$ pcluster ssh multi-queue-hello-world -i ~/path/to/keyfile.pem
```

有关 pcluster ssh 的更多信息，请参阅 pcluster ssh。

登录后，运行命令 sinfo 以验证是否已设置和配置调度器队列。

有关 sinfo 的更多信息，请参阅 Slurm 文档 中的 sinfo。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
spot*        up   infinite     18  idle~ spot-dy-c5xlarge-[1-10],spot-dy-t2micro-[2-9]
spot*        up   infinite      2   idle spot-dy-t2micro-1,spot-st-t2micro-1
```

输出显示您的集群中有两个处于 idle 状态的 t2.micro 计算节点。

> **Note**
>
> - spot-st-t2micro-1 的名称中包含 st，是静态节点。此节点始终可用，并且对应于集群配置中的 min_count = 1。
> - spot-dy-t2micro-1 的名称中包含 dy，是动态节点。此节点当前可用，因为根据集群配置，它对应于 initial_count - min_count = 1。经过五分钟的自定义 scaledown_idletime 后，此节点将会缩减。

其他节点都处于节能状态，通过节点状态中的 ~ 后缀指示，没有支持它们的 EC2 实例。默认队列由队列名称后面的 * 后缀指定，所以您的默认作业队列是 spot。

在多队列模式下运行作业

接下来，尝试将作业运行到睡眠模式一段时间。该作业稍后将输出自己的主机名。确保当前用户可以运行此脚本。

```
$ cat hellojob.sh
#!/bin/bash
sleep 30
echo "Hello World from $(hostname)"
```

```
$ chmod +x hellojob.sh
$ ls -l hellojob.sh
-rwxrwxr-x 1 ec2-user ec2-user 57 Sep 23 21:57 hellojob.sh
```

使用 sbatch 命令提交作业。使用 -N 2 选项为该作业请求两个节点，然后验证作业是否成功提交。有关 sbatch 的更多信息，请参阅 Slurm 文档 中的 sbatch。

```
$ sbatch -N 2 --wrap "srun hellojob.sh"
Submitted batch job 2
```

您可以使用 squeue 命令查看您的队列并检查该作业的状态。请注意，由于您未指定特定队列，因此使用默认队列 (spot)。有关 squeue 的更多信息，请参阅 Slurm 文档 中的 squeue。

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 2      spot     wrap ec2-user  R       0:10      2 spot-dy-
t2micro-1,spot-st-t2micro-1
```

输出显示此作业目前处于运行状态。请等候 30 秒，以便作业完成，然后再次运行 squeue。

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

现在，队列中的作业已全部完成，请在当前目录中查找输出文件 slurm-2.out。

```
$ cat slurm-2.out
Hello World from spot-dy-t2micro-1
Hello World from spot-st-t2micro-1
```

输出还显示我们的作业已在 spot-st-t2micro-1 和 spot-st-t2micro-2 节点上成功运行。

现在，通过使用以下命令为特定实例指定约束条件来提交相同的作业。

```
$ sbatch -N 3 -p spot -C "[c5.xlarge*1&t2.micro*2]" --wrap "srun hellojob.sh"
Submitted batch job 3
```

您对 sbatch 使用了以下参数。

- -N 3：请求三个节点

- `-p spot`：将作业提交到 spot 队列。您也可以通过指定 `-p ondemand`，将作业提交到 ondemand 队列。
- `-C "[c5.xlarge*1&t2.micro*2]"`：指定该作业的特定节点约束条件。这将请求对该作业使用一 (1) 个 c5.xlarge 节点和两 (2) 个 t2.micro 节点。

运行 `sinfo` 命令查看节点和队列。（AWS ParallelCluster 中的队列在 Slurm 中称为分区。）

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
spot*        up   infinite      1   mix# spot-dy-c5xlarge-1
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[2-10],spot-dy-t2micro-[2-9]
spot*        up   infinite      2  alloc spot-dy-t2micro-1,spot-st-t2micro-1
```

节点正在启动。这由节点状态上的 # 后缀表示。运行 squeue 命令查看集群中作业的信息。

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 3      spot     wrap ec2-user CF       0:04      3 spot-dy-
c5xlarge-1,spot-dy-t2micro-1,spot-st-t2micro-1
```

您的作业处于 CF (CONFIGURING) 状态，正在等待实例纵向扩展并加入集群。

大约三分钟后，节点应可用，并且作业进入 R (RUNNING) 状态。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[2-10],spot-dy-t2micro-[2-9]
spot*        up   infinite      1    mix spot-dy-c5xlarge-1
spot*        up   infinite      2  alloc spot-dy-t2micro-1,spot-st-t2micro-1
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 3      spot     wrap ec2-user  R       0:04      3 spot-dy-
c5xlarge-1,spot-dy-t2micro-1,spot-st-t2micro-1
```

作业完成，所有三个节点都处于 `idle` 状态。

```
$ squeue
```

```
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[2-10],spot-dy-t2micro-[2-9]
spot*        up   infinite      3   idle spot-dy-c5xlarge-1,spot-dy-t2micro-1,spot-st-
t2micro-1
```

然后，当队列中没有剩余作业后，您可以在本地目录中查看 slurm-3.out。

```
$ cat slurm-3.out
Hello World from spot-dy-c5xlarge-1
Hello World from spot-st-t2micro-1
Hello World from spot-dy-t2micro-1
```

输出还显示作业在相应的节点上成功运行。

您可以观察缩减过程。在您的集群配置中，您指定了 5 分钟的自定义 scaledown_idletime。处于空闲状态五分钟后，您的动态节点 spot-dy-c5xlarge-1 和 spot-dy-t2micro-1 会自动缩减并进入 POWER_DOWN 模式。请注意，静态节点 spot-st-t2micro-1 不会缩减。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
spot*        up   infinite      2  idle% spot-dy-c5xlarge-1,spot-dy-t2micro-1
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[2-10],spot-dy-t2micro-[2-9]
spot*        up   infinite      1   idle spot-st-t2micro-1
```

从上面的代码中，您可以看到 spot-dy-c5xlarge-1 和 spot-dy-t2micro-1 处于 POWER_DOWN 模式。这由 % 后缀表示。相应的实例会立即终止，但节点仍处于 POWER_DOWN 状态，并且在 120 秒（两分钟）内不可用。在此时间之后，节点将恢复节能状态，可以再次使用。有关更多信息，请参阅 Slurm 多队列模式指南。

以下应该是集群的最终状态：

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
spot*        up   infinite     19  idle~ spot-dy-c5xlarge-[1-10],spot-dy-t2micro-[1-9]
spot*        up   infinite      1   idle spot-st-t2micro-1
```

注销集群后，您可以通过运行 pcluster delete 来进行清理。有关 pcluster list 和 pcluster delete 的更多信息，请参阅 pcluster list 和 pcluster delete。

```
$ pcluster list
multi-queue CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue
Deleting: multi-queue
...
```

在具有 EFA 和 GPU 实例的集群上运行作业

教程的这一部分详细介绍如何修改配置和启动具有多个队列并包含使用 EFA 网络和 GPU 资源的实例的集群。请注意，本教程中使用的实例是价格较高的实例。

在继续执行本教程中概述的步骤之前，请检查您的账户限制以确保您有权使用这些实例。

通过使用以下配置修改配置文件。

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue-special

[aws]
aws_region_name = <Your AWS ##>

[scaling demo]
scaledown_idletime = 5

[cluster multi-queue-special]
key_name = <Your SSH key name>
base_os = alinux2                    # optional, defaults to alinux2
scheduler = slurm
master_instance_type = c5.xlarge     # optional, defaults to t2.micro
vpc_settings = <Your VPC section>
scaling_settings = demo
queue_settings = efa,gpu

[queue gpu]
compute_resource_settings = gpu_i1
disable_hyperthreading = true        # optional, defaults to false

[compute_resource gpu_i1]
```

```
instance_type = g3.8xlarge

[queue efa]
compute_resource_settings = efa_i1
enable_efa = true
placement_group = DYNAMIC           # optional, defaults to no placement group settings

[compute_resource efa_i1]
instance_type = c5n.18xlarge
max_count = 5
```

创建集群

```
$ pcluster create multi-queue-special -t multi-queue-special
```

创建集群后，使用您的私有 SSH 密钥文件登录到头节点。

```
$ pcluster ssh multi-queue-special -i ~/path/to/keyfile.pem
```

以下应该是集群的初始状态：

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa*         up   infinite      5  idle~ efa-dy-c5n18xlarge-[1-5]
gpu          up   infinite     10  idle~ gpu-dy-g38xlarge-[1-10]
```

本节介绍如何通过提交一些作业来检查节点是否具有 EFA 或 GPU 资源。

首先，编写作业脚本。`efa_job.sh` 将睡眠 30 秒。之后，在 `lspci` 命令的输出中查找 EFA。`gpu_job.sh` 将睡眠 30 秒。之后，运行 `nvidia-smi` 以显示有关该节点的 GPU 信息。

```
$ cat efa_job.sh
#!/bin/bash

sleep 30
lspci | grep "EFA"

$ cat gpu_job.sh
#!/bin/bash

sleep 30
```

```
nvidia-smi

$ chmod +x efa_job.sh
$ chmod +x gpu_job.sh
```

使用 sbatch 提交作业，

```
$ sbatch -p efa --wrap "srun efa_job.sh"
Submitted batch job 2
$ sbatch -p gpu --wrap "srun gpu_job.sh" -G 1
Submitted batch job 3
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 2       efa     wrap ec2-user CF       0:32      1 efa-dy-
c5n18xlarge-1
                 3       gpu     wrap ec2-user CF       0:20      1 gpu-dy-g38xlarge-1
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa*         up   infinite      1   mix# efa-dy-c5n18xlarge-1
efa*         up   infinite      4  idle~ efa-dy-c5n18xlarge-[2-5]
gpu          up   infinite      1   mix# gpu-dy-g38xlarge-1
gpu          up   infinite      9  idle~ gpu-dy-g38xlarge-[2-10]
```

几分钟后，您应该能够看到在线节点和正在运行的作业。

```
[ec2-user@ip-172-31-15-251 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa*         up   infinite      4  idle~ efa-dy-c5n18xlarge-[2-5]
efa*         up   infinite      1    mix efa-dy-c5n18xlarge-1
gpu          up   infinite      9  idle~ gpu-dy-g38xlarge-[2-10]
gpu          up   infinite      1    mix gpu-dy-g38xlarge-1
[ec2-user@ip-172-31-15-251 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 4       gpu     wrap ec2-user  R       0:06      1 gpu-dy-g38xlarge-1
                 5       efa     wrap ec2-user  R       0:01      1 efa-dy-
c5n18xlarge-1
```

作业完成后，检查输出。从 slurm-2.out 文件的输出中，您可以看到 efa-dy-c5n18xlarge-1 节点上存在 EFA。从 slurm-3.out 文件的输出中，您可以看到 nvidia-smi 输出中包含 gpu-dy-g38xlarge-1 节点的 GPU 信息。

```
$ cat slurm-2.out
```

```
00:06.0 Ethernet controller: Amazon.com, Inc. Elastic Fabric Adapter (EFA)

$ cat slurm-3.out
Thu Oct  1 22:19:18 2020
+-----------------------------------------------------------------------------+
| NVIDIA-SMI 450.51.05    Driver Version: 450.51.05    CUDA Version: 11.0     |
|-------------------------------+----------------------+----------------------+
| GPU  Name        Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|         Memory-Usage | GPU-Util  Compute M. |
|                               |                      |               MIG M. |
|===============================+======================+======================|
|   0  Tesla M60           Off  | 00000000:00:1D.0 Off |                    0 |
| N/A   28C    P0    38W / 150W |      0MiB /  7618MiB |      0%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+
|   1  Tesla M60           Off  | 00000000:00:1E.0 Off |                    0 |
| N/A   36C    P0    37W / 150W |      0MiB /  7618MiB |     98%      Default |
|                               |                      |                  N/A |
+-------------------------------+----------------------+----------------------+

+-----------------------------------------------------------------------------+
| Processes:                                                                  |
|  GPU   GI   CI        PID   Type   Process name                  GPU Memory |
|        ID   ID                                                   Usage      |
|=============================================================================|
|  No running processes found                                                 |
+-----------------------------------------------------------------------------+
```

您可以观察缩减过程。在集群配置中，您之前指定了五分钟的自定义 scaledown_idletime。因此，在处于空闲状态五分钟后，您的动态节点 spot-dy-c5xlarge-1 和 spot-dy-t2micro-1 会自动缩减并进入 POWER_DOWN 模式。最终，这些节点进入节能模式，可以再次使用。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa*         up   infinite      1  idle% efa-dy-c5n18xlarge-1
efa*         up   infinite      4  idle~ efa-dy-c5n18xlarge-[2-5]
gpu          up   infinite      1  idle% gpu-dy-g38xlarge-1
gpu          up   infinite      9  idle~ gpu-dy-g38xlarge-[2-10]

# After 120 seconds
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa*         up   infinite      5  idle~ efa-dy-c5n18xlarge-[1-5]
```

```
gpu          up   infinite     10  idle~ gpu-dy-g38xlarge-[1-10]
```

注销集群后，您可以通过运行 pcluster delete *<cluster name>* 来进行清理。

```
$ pcluster list
multi-queue-special CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue-special
Deleting: multi-queue-special
...
```

有关更多信息，请参阅 Slurm 多队列模式指南。

开发

您可以使用以下部分来开始开发 AWS ParallelCluster。

> **Important**
>
> 以下部分包含有关使用自定义版本的说明书配方和自定义 AWS ParallelCluster 节点程序包的说明。此信息涵盖了自定义 AWS ParallelCluster 的高级方法，以及难以调试的潜在问题。AWS ParallelCluster 团队强烈建议使用自定义引导操作中的脚本进行自定义，因为安装后挂钩通常更易于调试和更易于跨 AWS ParallelCluster 版本进行移植。

主题

设置自定义 AWS ParallelCluster 说明书

> **Important**
>
> 以下是有关使用自定义版本的 AWS ParallelCluster 说明书食谱的说明。这是自定义 AWS ParallelCluster 的高级方法，具有难以调试的潜在问题。AWS ParallelCluster 团队强烈建议使用自定义引导操作中的脚本进行自定义，因为安装后挂钩通常更易于调试和更易于跨 AWS ParallelCluster 版本进行移植。

步骤

1. 确定您在其中克隆了 AWS ParallelCluster 说明书代码的 AWS ParallelCluster 说明书工作目录。

```
_cookbookDir=<path to cookbook>
```

2. 检测 AWS ParallelCluster 说明书的当前版本。

```
_version=$(grep version ${_cookbookDir}/metadata.rb|awk '{print $2}'| tr -d \')
```

3. 创建 AWS ParallelCluster 说明书的存档，并计算其 md5。

```
cd "${_cookbookDir}"
_stashName=$(git stash create)
git archive --format tar --prefix="aws-parallelcluster-cookbook-${_version}/"
 "${_stashName:-HEAD}" | gzip > "aws-parallelcluster-cookbook-${_version}.tgz"
md5sum "aws-parallelcluster-cookbook-${_version}.tgz" > "aws-parallelcluster-
cookbook-${_version}.md5"
```

4. 创建一个 Amazon S3 存储桶，并将存档、其 md5 及其上次修改日期上传到此存储桶。通过公共读取 ACL 授予公共可读权限。

```
_bucket=<the bucket name>
aws s3 cp --acl public-read aws-parallelcluster-cookbook-${_version}.tgz s3://
${_bucket}/cookbooks/aws-parallelcluster-cookbook-${_version}.tgz
aws s3 cp --acl public-read aws-parallelcluster-cookbook-${_version}.md5 s3://
${_bucket}/cookbooks/aws-parallelcluster-cookbook-${_version}.md5
aws s3api head-object --bucket ${_bucket} --key cookbooks/aws-parallelcluster-
cookbook-${_version}.tgz --output text --query LastModified > aws-parallelcluster-
cookbook-${_version}.tgz.date
aws s3 cp --acl public-read aws-parallelcluster-cookbook-${_version}.tgz.date s3://
${_bucket}/cookbooks/aws-parallelcluster-cookbook-${_version}.tgz.date
```

5. 将以下变量添加到 AWS ParallelCluster 配置文件中的 [cluster] 部分下。

```
custom_chef_cookbook = https://${_bucket}.s3.<the bucket region>.amazonaws.com/
cookbooks/aws-parallelcluster-cookbook-${_version}.tgz
extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
```

> **Note**
>
> 从 AWS ParallelCluster 版本 2.6.1 开始，在启动节点时将默认跳过大多数安装食谱以缩短启动时间。要以牺牲向后兼容性为代价跳过大多数安装食谱以缩短启动时间，请从 extra_json 设置中的 cluster 键中删除 "skip_install_recipes" : "no"。

设置自定义 AWS ParallelCluster 节点程序包

> **Warning**
>
> 以下是有关使用自定义版本的 AWS ParallelCluster 节点程序包的说明。这是自定义 AWS ParallelCluster 的高级方法，具有难以调试的潜在问题。AWS ParallelCluster 团队强烈建议使用自定义引导操作中的脚本进行自定义，因为安装后挂钩通常更易于调试和更易于跨 AWS ParallelCluster 版本进行移植。

步骤

1. 确定您在其中克隆了 AWS ParallelCluster 节点代码的 AWS ParallelCluster 节点工作目录。

```
_nodeDir=<path to node package>
```

2. 检测 AWS ParallelCluster 节点的当前版本。

```
_version=$(grep "version = \"" ${_nodeDir}/setup.py |awk '{print $3}' | tr -d \")
```

3. 创建 AWS ParallelCluster 节点的存档。

```
cd "${_nodeDir}"
_stashName=$(git stash create)
git archive --format tar --prefix="aws-parallelcluster-node-${_version}/"
 "${_stashName:-HEAD}" | gzip > "aws-parallelcluster-node-${_version}.tgz"
```

4. 创建一个 Amazon S3 存储桶并将存档上传到该存储桶。通过公共读取 ACL 授予公共可读权限。

```
_bucket=<the bucket name>
aws s3 cp --acl public-read aws-parallelcluster-node-${_version}.tgz s3://${_bucket}/
node/aws-parallelcluster-node-${_version}.tgz
```

5. 将以下变量添加到 AWS ParallelCluster 配置文件中的 [cluster] 部分下。

```
extra_json = { "cluster" : { "custom_node_package" : "https://${_bucket}.s3.<the
 bucket region>.amazonaws.com/node/aws-parallelcluster-node-${_version}.tgz",
 "skip_install_recipes" : "no" } }
```

> **Note**
>
> 从 AWS ParallelCluster 版本 2.6.1 开始，在启动节点时将默认跳过大多数安装食谱以缩短启动时间。要以牺牲向后兼容性为代价跳过大多数安装食谱以缩短启动时间，请从 `extra_json` 设置中的 `cluster` 键中删除 `"skip_install_recipes" : "no"`。

AWS ParallelCluster 故障排除

AWS ParallelCluster 社区维护着一个 Wiki 页面，在 AWS ParallelCluster GitHub Wi ki 上提供了许多疑难解答技巧。有关已知问题的列表，请参阅已知问题。

主题

检索和保留日志

日志是用于排查问题的有用资源。在使用日志对 AWS ParallelCluster 资源进行问题排查之前，应先创建集群日志存档。按照 AWS ParallelCluster GitHub Wiki 上创建集群日志存档主题中描述的步骤开始此过程。

如果您的一个正在运行的集群遇到问题，则应在开始排查问题之前，通过运行 `pcluster stop <cluster_name>` 命令将该集群置于 STOPPED 状态。这样可以防止产生任何意外成本。

如果 `pcluster` 停止运行，或者您想删除集群但仍保留其日志，请运行 `pcluster delete --keep-logs <cluster_name>` 命令。运行此命令会删除集群，但保留存储在 Amazon 中的日志组 CloudWatch。有关此命令的更多信息，请参阅 `pcluster delete` 文档。

排查堆栈部署问题

如果您的集群创建失败并回滚堆栈创建，则可以通过查看以下日志文件来诊断问题。您需要在这些日志中找到 `ROLLBACK_IN_PROGRESS` 的输出。失败消息的内容应与以下内容类似：

```
$ pcluster create mycluster
Creating stack named: parallelcluster-mycluster
Status: parallelcluster-mycluster - ROLLBACK_IN_PROGRESS
Cluster creation failed.  Failed events:
  - AWS::EC2::Instance MasterServer Received FAILURE signal with UniqueId
 i-07af1cb218dd6a081
```

要诊断问题，请使用 pcluster create（包括 --norollback 标志）重新创建该集群。然后，SSH进入集群：

```
$ pcluster create mycluster --norollback
...
$ pcluster ssh mycluster
```

登录到头节点后，您应该可以找到三个主要的日志文件，可以用它们来精准确定错误。

- /var/log/cfn-init.log 是 cfn-init 脚本的日志。首先查看此日志。在此日志中，您可能会看到类似“Command chef failed”的错误。查看此行前面的几行，了解与该错误消息相关的更多细节。有关更多信息，请参阅 cfn-init。
- /var/log/cloud-init.log 是 cloud-init 的日志。如果您在 cfn-init.log 中没有看到任何内容，请接下来尝试查看此日志。
- /var/log/cloud-init-output.log 是 cloud-init 运行的命令的输出。这包括 cfn-init 的输出。在大多数情况下，排查此类问题无需查看此日志。

排查多队列模式集群中的问题

本节与使用 2.9.0 及更高 AWS ParallelCluster 版本安装的集群相关 Slurm 作业调度器。有关多队列模式的更多信息，请参阅多队列模式。

主题

关键日志

下表概述了头节点的关键日志：

`/var/log/cfn-init.log`

AWS CloudFormation 这是初始化日志。其中包含设置实例时运行的所有命令。可以用它来排查初始化问题。

`/var/log/chef-client.log`

这是 Chef 客户端日志。它包含通过 ChefCINC/运行的所有命令。可以用它来排查初始化问题。

`/var/log/parallelcluster/slurm_resume.log`

这是 `ResumeProgram` 日志。它启动动态节点的实例，可用于排查动态节点启动问题。

`/var/log/parallelcluster/slurm_suspend.log`

这是 `SuspendProgram` 日志。在终止动态节点的实例时会调用该日志，可用于排查动态节点终止问题。查看此日志时，还应检查 `clustermgtd` 日志。

`/var/log/parallelcluster/clustermgtd`

这是 `clustermgtd` 日志。它作为集中式进程守护程序运行，用于管理大多数集群操作。可以用它来排查任何启动、终止或集群操作问题。

`/var/log/slurmctld.log`

这是 Slurm 控制守护程序日志。 AWS ParallelCluster 不会做出扩展决策。相反，它只是尝试启动资源来满足 Slurm 要求。它可用于排查扩展和分配问题、与作业相关的问题以及与调度器相关的任何启动和终止问题。

以下是计算节点的关键说明：

`/var/log/cloud-init-output.log`

这是 cloud-init 日志。其中包含设置实例时运行的所有命令。可以用它来排查初始化问题。

`/var/log/parallelcluster/computemgtd`

这是 `computemgtd` 日志。它在每个计算节点上运行，用于在头节点上的 `clustermgtd` 进程守护程序离线的罕见事件中监控节点。可以用它来排查意外终止问题。

`/var/log/slurmd.log`

这是 Slurm 计算守护程序日志。可以用它来排查初始化和计算失败相关问题。

排查节点初始化问题

本节介绍如何排查节点初始化问题。这包括节点无法启动、开机或加入集群的问题。

头节点：

适用日志：

- /var/log/cfn-init.log
- /var/log/chef-client.log
- /var/log/parallelcluster/clustermgtd
- /var/log/parallelcluster/slurm_resume.log
- /var/log/slurmctld.log

检查 /var/log/cfn-init.log 和 /var/log/chef-client.log 日志。这些日志应包含设置头节点时运行的所有操作。设置过程中发生的大多数错误的错误消息应该都包含在 /var/log/chef-client.log 日志中。如果在集群的配置中指定了预安装或安装后脚本，请通过日志消息仔细检查脚本是否成功运行。

创建集群时，头节点必须等待计算节点加入集群，然后才能加入集群。因此，如果计算节点加入集群失败，则头节点也会失败。根据您使用的计算节点的类型，您可以按照其中一组过程来排查此类问题：

动态计算节点：

- 搜索计算节点名称的 ResumeProgram 日志 (/var/log/parallelcluster/slurm_resume.log) 以查看是否对该节点调用过 ResumeProgram。（如果ResumeProgram从未被调用，则可以查看slurmctld日志 (/var/log/slurmctld.log) 以确定是否 Slurm 曾经尝试ResumeProgram与该节点通话。）
- 请注意，ResumeProgram 的权限不正确可能会导致 ResumeProgram 静默失败。如果您使用的是经过修改ResumeProgram设置AMI的自定义，请检查该用户是否ResumeProgram为slurm用户所有并具有 744 (rwxr--r--) 权限。
- 如果调用了 ResumeProgram，请查看是否为该节点启动了实例。如果未启动任何实例，则应该能够看到一条描述启动失败的错误消息。
- 如果启动了实例，则在设置过程中可能出现了问题。您应该会从 ResumeProgram 日志中看到相应的私有 IP 地址和实例 ID。此外，您可以查看特定实例的相应设置日志。有关排查计算节点设置错误的更多信息，请参阅下一节。

静态计算节点：

- 检查 clustermgtd (/var/log/parallelcluster/clustermgtd) 日志，查看是否为该节点启动了实例。如果未启动，则应该有详细说明启动失败的明确错误消息。
- 如果启动了实例，则表示设置过程中出现了问题。您应该会从 ResumeProgram 日志中看到相应的私有 IP 地址和实例 ID。此外，您可以查看特定实例的相应设置日志。

- 计算节点：
  - 适用日志：
    - /var/log/cloud-init-output.log
    - /var/log/slurmd.log
  - 如果启动了计算节点，请先检查 /var/log/cloud-init-output.log，其中应包含类似于头节点 /var/log/chef-client.log 日志的设置日志。设置过程中发生的大多数错误的错误消息应该都包含在 /var/log/cloud-init-output.log 日志中。如果在集群配置中指定了预安装或安装后脚本，请检查它们是否成功运行。
  - 如果你使用的是经过修改AMI的自定义 Slurm 配置，那么可能有一个 Slurm 导致计算节点无法加入集群的相关错误。对于与调度器相关的错误，请检查 /var/log/slurmd.log 日志。

排查意外节点替换和终止问题

本节继续探讨如何排查节点相关问题，特别是在节点意外替换或终止时。

- 适用日志：
  - /var/log/parallelcluster/clustermgtd（头节点）
  - /var/log/slurmctld.log（头节点）
  - /var/log/parallelcluster/computemgtd（计算节点）
- 节点意外替换或终止
  - 检查 clustermgtd 日志 (/var/log/parallelcluster/clustermgtd) 以查看 clustermgtd 是否执行了替换或终止节点的操作。请注意，clustermgtd 处理所有正常的节点维护操作。
  - 如果 clustermgtd 替换或终止了该节点，则应该会有详细说明为何对该节点执行此操作的消息。如果原因与调度器有关（例如，因为节点处于 DOWN 状态），请查看 slurmctld 日志以获取更多信息。如果原因EC2与亚马逊有关，则应附上信息性消息，详细说明需要更换的亚马逊EC2相关问题。

  - 如果clustermgtd没有终止节点，请先检查这是否是 Amazon 的预期终止EC2，更具体地说是即时终止。 computemgtd，在计算节点上运行，如果clustermgtd节点被确定为运行状况不佳，也可以采取措施终止节点。检查 computemgtd 日志 (/var/log/parallelcluster/computemgtd) 以查看 computemgtd 是否终止了该节点。
- 节点失败
  - 检查 slurmctld 日志 (/var/log/slurmctld.log) 以查看作业或节点失败的原因。请注意，如果节点失败，作业会自动重新排队。
  - 如果slurm_resume报告该节点已启动，并在几分钟后clustermgtd报告说 Amazon EC2 中没有该节点的相应实例，则该节点可能会在设置过程中失败。要从计算 (/var/log/cloud-init-output.log) 中检索日志，请执行以下步骤：
    - 提交一份待租的职位 Slurm 启动一个新节点。
    - 节点启动后，使用以下命令启用终止保护。

```
aws ec2 modify-instance-attribute --instance-id i-xyz --disable-api-termination
```

    - 使用以下命令从该节点检索控制台输出。

```
aws ec2 get-console-output --instance-id i-xyz --output text
```

替换、终止或关闭有问题的实例和节点

- 适用日志：
  - /var/log/parallelcluster/clustermgtd（头节点）
  - /var/log/parallelcluster/slurm_suspend.log（头节点）
- 在大多数情况下，clustermgtd 会处理所有预期的实例终止操作。检查 clustermgtd 日志以查看其无法替换或终止节点的原因。
- 对于 scaledown_idletime 失败的动态节点，请检查 SuspendProgram 日志以查看 slurmctld 是否以特定节点作为参数调用了 SuspendProgram。请注意，SuspendProgram 实际上并不执行任何操作，它只是记录被调用时的时间。所有实例终止和 NodeAddr 重置均由 clustermgtd 完成。Slurm 之后会SuspendTimeout自动将节点恢复到POWER_SAVING状态。

排查其他已知的节点和作业问题

另一种已知问题是 AWS ParallelCluster 可能无法分配工作岗位或做出扩展决策。对于此类问题，AWS ParallelCluster 只能根据以下条件启动、终止或维护资源 Slurm 指令。对于这些问题，请查看 `slurmctld` 日志以排查问题。

排查单队列模式集群中的问题

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

本节适用于采用以下两种配置之一并且没有多队列模式的集群：

- 使用 2.9.0 之前的 AWS ParallelCluster 版本启动，并且 SGE, Torque，或 Slurm 作业调度器。
- 使用 2.9.0 或 AWS ParallelCluster 更高版本启动，并且 SGE 或者 Torque 作业调度器。

主题

关键日志

以下日志文件是头节点的关键日志。

对于 2.9.0 或更高 AWS ParallelCluster 版本：

`/var/log/chef-client.log`

这是CINC（厨师）客户端日志。它包含所有运行过的命令CINC。可以用它来排查初始化问题。

对于所有 AWS ParallelCluster 版本：

`/var/log/cfn-init.log`

这是 `cfn-init` 日志。其中包含设置实例时运行的所有命令，因此可用于排查初始化问题。有关更多信息，请参阅 cfn-init。

`/var/log/clustermgtd.log`

这是的`clustermgtd`日志 Slurm 调度器。 `clustermgtd`作为管理大多数集群操作操作的集中式守护程序运行。可以用它来排查任何启动、终止或集群操作问题。

`/var/log/jobwatcher`

这是的`jobwatcher`日志 SGE 以及 Torque 调度器。 `jobwatcher`监控调度器队列并更新 Auto Scaling 组。可以用它来排查与纵向扩展节点相关的问题。

`/var/log/sqswatcher`

这是的`sqswatcher`日志 SGE 以及 Torque 调度器。 `sqswatcher`处理计算实例在成功初始化后发送的实例就绪事件。它还会向调度器配置中添加计算节点。可以使用此日志来排查一个或多个节点无法加入集群的问题。

以下是计算节点的关键日志。

AWS ParallelCluster 版本 2.9.0 或更高版本

`/var/log/cloud-init-output.log`

这是云初始化日志。其中包含设置实例时运行的所有命令。可以用它来排查初始化问题。

AWS ParallelCluster 2.9.0 之前的版本

`/var/log/cfn-init.log`

CloudFormation 这是初始化日志。其中包含设置实例时运行的所有命令。可以用它来排查初始化问题

所有版本

`/var/log/nodewatcher`

这是日`nodewatcher`志。 `nodewatcher`使用时在每个计算节点上运行的守护程序 SGE 以及 Torque 调度器。如果某个节点处于空闲状态，他们会缩减该节点。可以使用此日志来排查与缩减资源相关的任何问题。

排查启动和加入操作失败问题

- 适用日志：
  - /var/log/cfn-init-cmd.log（头节点和计算节点）
  - /var/log/sqswatcher（头节点）
- 如果节点启动失败，请检查 /var/log/cfn-init-cmd.log 日志以查看具体的错误消息。在大多数情况下，节点启动失败是由设置失败引起的。
- 如果计算节点虽然设置成功，但仍无法加入调度器配置，请检查 /var/log/sqswatcher 日志以查看 sqswatcher 是否处理了该事件。在大多数情况下，这些问题是因为 sqswatcher 未处理该事件。

排查扩展问题

- 适用日志：
  - /var/log/jobwatcher（头节点）
  - /var/log/nodewatcher（计算节点）
- 纵向扩展问题：对于头节点，检查 /var/log/jobwatcher 日志以查看 jobwatcher 进程守护程序是否计算出正确的所需节点数并更新了自动扩缩组。请注意，jobwatcher 会监控调度器队列并更新自动扩缩组。
- 缩减问题：对于计算节点，检查出问题节点上的 /var/log/nodewatcher 日志以查看缩减该节点的原因。请注意，nodewatcher 进程守护程序会缩减处于空闲状态的计算节点。

排查其他集群相关问题

一个已知问题是随机计算节点在大规模集群（特别是具有 500 或更多计算节点的集群）上失败。此问题与单队列集群的扩展架构限制有关。如果你想使用大型集群，正在使用 v2.9.0 或更高 AWS ParallelCluster 版本，正在使用 Slurm，为了避免此问题，您应该升级并切换到支持多队列模式的集群。您可以通过运行 pcluster-config convert 来实现这一目的。

对于超大规模集群，可能需要对系统进行额外调整。欲了解更多信息，请联系 AWS Support。

置放群组和实例启动问题

为了获得最低的节点间延迟，请使用置放群组。置放群组可确保您的实例位于同一网络主干中。如果发出请求时没有足够的可用实例，则会返回 InsufficientInstanceCapacity 错误。要在使

用集群置放群组时降低收到此错误的可能性，请将 placement_group 参数设置为 DYNAMIC 并将 placement 参数设置为 compute。

如果您需要高性能的共享文件系统，请考虑使用 for Lustre FSx。

如果头节点必须位于置放群组中，则对头节点和所有计算节点使用相同的实例类型和子网。这样可确保 compute_instance_type 参数与 master_instance_type 参数具有相同的值，placement 参数设置为 cluster 且未指定 compute_subnet_id 参数。使用此配置时，master_subnet_id 参数的值用于计算节点。

有关更多信息，请参阅 Amazon EC2 用户指南中的排查实例启动问题和置放群组角色和限制

无法替换的目录

以下目录在节点之间共享，无法替换。

/home

这包括默认的用户主文件夹（/home/ec2_user在 Amazon Linux /home/centos 上，CentOS，/home/ubuntu等等 Ubuntu).

/opt/intel

这包括英特尔MPI、英特尔 Parallel Studio 和相关文件。

/opt/sge

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

这包括 Son of Grid Engine 和相关文件。（有条件，仅当 scheduler = sge 时。）

/opt/slurm

这包括 Slurm Workload Manager 和相关文件。（有条件，仅当 scheduler = slurm 时。）

/opt/torque

> **Note**
>
> 从 2.11.5 版开始， AWS ParallelCluster 不支持使用 SGE 或者 Torque 调度器。

这包括 Torque Resource Manager 和相关文件。（有条件，仅当 `scheduler = torque` 时。）

对 Amazon 中的问题进行故障排除 DCV

主题

亚马逊日志 DCV

Amazon DCV 的日志将写入`/var/log/dcv/`目录中的文件中。查看这些日志有助于排查问题。

Amazon DCV 实例类型内存

实例类型应至少有 1.7 千兆字节 (GiB) 才能RAM运行 Amazon。DCVNano 以及 micro 实例类型没有足够的内存来运行 Amazon DCV。

Ubuntu Amazon 问题 DCV

在 Ubuntu 上通过DCV会话运行 Gnome 终端时，你可能无法自动访问通过登录外壳 AWS ParallelCluster 提供的用户环境。该用户环境提供 openmpi 或 intelmpi 等环境模块以及其他用户设置。

Gnome 终端的默认设置会阻止 Shell 作为登录 Shell 启动。这意味着 shell 配置文件不会自动获取，也不会加载 AWS ParallelCluster 用户环境。

要正确获取外壳配置文件并访问 AWS ParallelCluster 用户环境，请执行以下操作之一：

- 更改默认终端设置：
  1. 在 Gnome 终端中选择编辑菜单。
  2. 选择首选项，然后选择配置文件。
  3. 选择命令，然后选择作为登录 Shell 运行命令。
  4. 打开新终端。
- 使用命令行获取可用的配置文件：

```
$ source /etc/profile && source $HOME/.bashrc
```

在采用 AWS Batch 集成的集群中排查问题

本节与具有 AWS Batch 调度程序集成的集群相关。

头节点问题

与头节点相关的设置问题可以按照与单队列集群相同的方式进行排查。有关这些问题的更多信息，请参阅排查单队列模式集群中的问题。

AWS Batch 多节点 parallel 作业提交问题

如果在 AWS Batch 用作作业调度器时提交多节点 parallel 作业时遇到问题，则应升级到 2.5.0 AWS ParallelCluster 版。如果这不可行，则可以使用以下主题中详细介绍的解决方法：Self patch a cluster used for submitting multi-node parallel jobs through AWS Batch。

计算问题

AWS Batch 管理服务的扩展和计算方面。如果您遇到与计算相关的问题，请参阅 AWS Batch 故障排除文档以获取帮助。

作业失败

如果作业失败，您可以运行 `awsbout` 命令来检索作业输出。您也可以运行`awsbstat -d`命令以获取指向 Amazon 存储的任务日志的链接 CloudWatch。

资源创建失败时排查问题

本节内容与集群资源创建失败有关。

当资源创建失败时，会 ParallelCluster 返回如下错误消息。

```
pcluster create -c config my-cluster
Beginning cluster creation for cluster: my-cluster
WARNING: The instance type 'p4d.24xlarge' cannot take public IPs. Please make sure that
 the subnet with
id 'subnet-1234567890abcdef0' has the proper routing configuration to allow private IPs
 reaching the
```

```
Internet (e.g. a NAT Gateway and a valid route table).
WARNING: The instance type 'p4d.24xlarge' cannot take public IPs. Please make sure that
 the subnet with
id 'subnet-1234567890abcdef0' has the proper routing configuration to allow private IPs
 reaching the Internet
(e.g. a NAT Gateway and a valid route table).
Info: There is a newer version 3.0.3 of AWS ParallelCluster available.
Creating stack named: parallelcluster-my-cluster
Status: parallelcluster-my-cluster - ROLLBACK_IN_PROGRESS
Cluster creation failed.  Failed events:
- AWS::CloudFormation::Stack MasterServerSubstack Embedded stack
arn:aws:cloudformation:region-id:123456789012:stack/parallelcluster-my-cluster-
MasterServerSubstack-ABCDEFGHIJKL/a1234567-b321-c765-d432-dcba98766789
was not successfully created:
The following resource(s) failed to create: [MasterServer].
- AWS::CloudFormation::Stack parallelcluster-my-cluster-MasterServerSubstack-
ABCDEFGHIJKL The following resource(s) failed to create: [MasterServer].
- AWS::EC2::Instance MasterServer You have requested more vCPU capacity than your
 current vCPU limit of 0 allows for the instance bucket that the
specified instance type belongs to. Please visit http://aws.amazon.com/contact-us/ec2-
request to request an adjustment to this limit.
(Service: AmazonEC2; Status Code: 400; Error Code: VcpuLimitExceeded; Request ID:
 a9876543-b321-c765-d432-dcba98766789; Proxy: null)
}
```

例如，如果您看到之前的命令响应中显示的状态消息，则必须使用不会超过当前 v CPU 限制的实例类型或请求更多 v CPU 容量。

您还可以使用 CloudFormation 控制台查看有关"`Cluster creation failed`"状态的信息。

从控制台查看 CloudFormation 错误消息。

1. 登录 AWS Management Console 并导航到 https://console.aws.amazon.com/cloudformation。
2. 选择名为 parallelcluster-的堆栈*`cluster_name`*.
3. 选择事件选项卡。
4. 通过按逻辑 ID 滚动浏览资源事件列表，查看创建失败的资源的状态。如果子任务创建失败，请向后移动，找到失败的资源事件。
5. AWS CloudFormation 错误消息示例：

```
2022-02-07 11:59:14 UTC-0800 MasterServerSubstack CREATE_FAILED Embedded stack
```

```
arn:aws:cloudformation:region-id:123456789012:stack/parallelcluster-my-cluster-
MasterServerSubstack-ABCDEFGHIJKL/a1234567-b321-c765-d432-dcba98766789
was not successfully created: The following resource(s) failed to create:
 [MasterServer].
```

解决IAM策略大小问题

要验证附加到IAM角色的托管策略的配 AWS STS 额，请参阅和配额、名称要求和字符限制。如果托管策略的大小超过配额，请将该策略拆分为两个或多个策略。如果某个角色所关联的策略数量超过了配额，请创建其他角色并在这些角色之间分配策略以满足配额。IAM

其他支持

有关已知问题的列表，请参阅 GitHubWiki 主页面或问题页面。如需更紧急的问题，请联系 AWS Support 或打开新 GitHub问题。

AWS ParallelCluster 支持政策

AWS ParallelCluster 同时支持多个版本。每个 AWS ParallelCluster 版本都有计划的支持生命周期结束 (EOSL) 日期。在 EOSL 日期之后，不再为该版本提供进一步的支持或维护。

AWS ParallelCluster 使用 `major.minor.patch` 版本方案。新功能、性能改进、安全更新和错误修复包含在最新主要版本的新次要版本中。次要版本在主要版本中向后兼容。对于关键问题，AWS 通过发布补丁来提供修复，但仅适用于尚未到达 EOSL 的最新次要版本。如果要使用新版本的更新，则需要升级到新的次要版本或补丁版本。

AWS ParallelCluster 版本	支持生命周期结束 (EOSL) 日期
2.10.4 及更高版本	2021 年 12 月 31 日
2.11.*x*	2022 年 12 月 31 日

AWS ParallelCluster 中的安全性

AWS 十分重视云安全性。作为 AWS 客户，您将从专为满足大多数安全敏感型企业的要求而打造的数据中心和网络架构中受益。

安全性是AWS和您的共同责任。责任共担模式 将其描述为云的安全性和云中的安全性：

- 云的安全性 – AWS 负责保护在 AWS 云中运行 AWS 服务的基础设施。AWS 还向您提供可安全使用的服务。作为AWS 合规性计划的一部分，第三方审计人员将定期测试和验证安全性的有效性。要了解适用于 AWS ParallelCluster 的合规性计划，请参阅合规性计划范围内的 AWS 服务。
- 云中的安全性 - 您的责任由您使用的特定 AWS 服务决定。您还需要对多种其它相关因素负责，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。

此文档介绍您应该如何在使用 AWS ParallelCluster 时应用责任共担模式。以下主题说明如何配置 AWS ParallelCluster 以实现您的安全性和合规性目标。您还将了解如何使用 AWS ParallelCluster 以帮助您监控和保护 AWS 资源。

主题

AWS ParallelCluster 所用服务的安全信息

- Amazon ECR 中的安全性
- Amazon ECS 中的安全性
- Amazon EFS 中的安全性
- 适用于 Lustre 的 FSx 中的安全性
- AWS Identity and Access Management (IAM) 中的安全性
- EC2 Image Builder 中的安全性
- AWS Lambda 中的安全性
- Amazon Route 53 中的安全性
- Amazon SNS 中的安全性
- Amazon SQS 中的安全性（适用于 AWS ParallelCluster 版本 2.x。）
- Amazon S3 中的安全性
- Amazon VPC 中的安全性

中的数据保护 AWS ParallelCluster

这些区域有： AWS 分担责任模型适用于以下领域的数据保护 AWS ParallelCluster。 如本模型所述，AWS 负责保护运行所有内容的全球基础设施 AWS Cloud。 您有责任保持对托管在此基础架构上的内容的控制。您还负责以下各项的安全配置和管理任务 AWS 服务 你用的。有关数据隐私的更多信息，请参阅数据隐私FAQ。 有关欧洲数据保护的信息，请参阅 AWS 责任共担模型和GDPR博客文章 AWS 安全博客。

出于数据保护的目的，我们建议您进行保护 AWS 账户 凭据并使用设置个人用户 AWS IAM Identity Center 或者 AWS Identity and Access Management (IAM)。这样，每个用户只获得履行其工作职责所需的权限。我们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)。
- 使用SSL/TLS与之通信 AWS 资源的费用。我们需要 TLS 1.2，建议使用 TLS 1.3。
- 使用API进行设置和用户活动记录 AWS CloudTrail。 有关使用 CloudTrail 轨迹捕获的信息 AWS 活动，请参阅使用中的 CloudTrail 轨迹 AWS CloudTrail 用户指南。
- 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。
- 如果您在访问时需要 FIPS 140-3 经过验证的加密模块 AWS 通过命令行界面或API，使用FIPS端点。有关可用FIPS端点的更多信息，请参阅联邦信息处理标准 (FIPS) 140-3。

我们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）。这包括你何时使用 AWS ParallelCluster 或其他 AWS 服务 使用控制台，API， AWS CLI，或 AWS SDKs。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URL向外部服务器提供，我们强烈建议您不要在中包含凭据信息，URL以验证您对该服务器的请求。

数据加密

所有安全服务均具有一项重要功能，即信息在未处于活动使用状态时都会加密。

静态加密

AWS ParallelCluster 除了与之交互所需的凭据外，本身不存储任何客户数据 AWS 代表用户提供服务。

对于集群中节点上的数据，可以对数据进行静态加密。

对于 Amazon EBS 卷，使用以下[ebs]部分中的ebs_kms_key_id设置配置加密 AWS ParallelCluster 版本 2.x。）有关更多信息，请参阅《亚马逊EC2用户指南》中的亚马逊EBS加密。

对于 Amazon EFS 卷，使用中[efs]部分中的encrypted和efs_kms_key_id设置来配置加密 AWS ParallelCluster 版本 2.x）。有关更多信息，请参阅 Amazon Elastic File System User Guide 中的 How encryption at rest works。

对FSx于 Lustre 文件系统，创建 Amazon FSx 文件系统时会自动启用静态数据加密。有关更多信息，请参阅 Amazon for Lustre 用户指南FSx中的加密静态数据。

对于带有NVMe卷的实例类型，NVMe实例存储卷上的数据使用在实例的硬件模块上实现的 XTS-AES-256 密码进行加密。加密密钥使用硬件模块生成，并且对于每个NVMe实例存储设备都是唯一的。当实例停止或终止并且无法恢复时，将销毁所有加密密钥。无法禁用此加密，并且无法提供自己的加密密钥。有关更多信息，请参阅 Amazon EC2 用户指南中的静态加密。

如果你使用 AWS ParallelCluster 调用 AWS 将客户数据传输到您的本地计算机进行存储的服务，有关如何存储、保护和加密这些数据的信息，请参阅该服务的《用户指南》中的 “安全与合规性” 一章。

传输中加密

默认情况下，从运行的客户端计算机传输的所有数据 AWS ParallelCluster 以及 AWS 通过HTTPS/TLS连接发送所有内容来加密服务端点。可以自动加密集群中节点之间的流量，具体取决于所选的实例类型。有关更多信息，请参阅 Amazon EC2 用户指南中的传输中加密。

另请参阅

- 亚马逊的数据保护 EC2
- EC2Image Builder 中的数据保护
- 中的数据保护 AWS CloudFormation
- 亚马逊的数据保护 EFS
- Amazon S3 中的数据保护
- FSx为 Lustre 提供数据保护

适用于 AWS ParallelCluster 的 Identity and Access Management

AWS ParallelCluster 使用角色访问您的 AWS 资源及其服务。AWS ParallelCluster 用于授予权限的实例和用户策略记录在 AWS Identity and Access Management 中的角色 AWS ParallelCluster上。

唯一的主要区别在于使用标准用户和长期凭证时如何进行身份验证。尽管用户需要密码才能访问 AWS 服务的控制台，但同一用户需要访问密钥对才能使用 AWS ParallelCluster 执行相同的操作。所有其他短期凭证的使用方式与在控制台中使用时相同。

AWS ParallelCluster 使用的凭证存储在纯文本文件中，并且不 加密。

- `$HOME/.aws/credentials` 文件存储访问 AWS 资源所需的长期凭证。这包括访问密钥 ID 和秘密访问密钥。
- 短期凭证（例如您承担的角色或用于 AWS IAM Identity Center 服务的角色的凭证）也分别存储在 `$HOME/.aws/cli/cache` 和 $HOME/.aws/sso/cache 文件夹中。

风险防范

- 我们强烈建议您在 `$HOME/.aws` 文件夹及其子文件夹和文件上配置文件系统权限，仅限授权用户访问。
- 尽可能使用具有临时凭证的角色，以减少凭证泄露时造成损坏的机会。仅使用长期凭证来请求和刷新短期角色凭证。

AWS ParallelCluster 的合规性验证

作为多个 AWS 合规性计划的一部分，第三方审计员将评估 AWS 服务的安全性和合规性。使用 AWS ParallelCluster 访问服务不会改变该服务的合规性。

有关特定合规性计划范围内的 AWS 服务列表，请参阅合规性计划范围内的 AWS 服务。有关常规信息，请参阅 AWS 合规性计划。

您可以使用下载第三方审计报告AWS Artifact 有关更多信息，请参阅在 AWS Artifact 中下载报告。

您使用 AWS ParallelCluster 的合规性责任取决于您数据的敏感度、贵公司的合规性目标以及适用的法律法规。AWS 提供以下资源来帮助满足合规性：

- 安全性与合规性 Quick Start 指南 – 这些部署指南讨论了架构注意事项，并提供了在 AWS 上部署关注安全性和合规性的基准环境的步骤。
- Amazon Web Services AWS 上的 HIPAA 安全性和合规性架构设计白皮书 – 此白皮书介绍了公司如何使用 AWS 创建符合 HIPAA 标准的应用程序。
- AWS 合规性资源 – 此业务手册和指南集合可能适用于您的行业和位置。
- 《AWS Config 开发人员指南》中的使用规则评估资源 – 此 AWS Config 服务评估您的资源配置对内部实践、行业指南和法规的遵循情况。
- AWS Security Hub – 此 AWS 服务提供了 AWS 中安全状态的全面视图，可帮助您检查是否符合安全行业标准和最佳实践。

强制实施最低版本 TLS 1.2

要提高与 AWS 服务通信时的安全性，您应将 AWS ParallelCluster 配置为使用 TLS 1.2 或更高版本。使用 AWS ParallelCluster 时，Python 用于设置 TLS 版本。

要确保 AWS ParallelCluster 不使用 TLS 1.2 之前的 TLS 版本，您可能需要重新编译 OpenSSL 以强制实施此最低版本，然后重新编译 Python 以使用新构建的 OpenSSL。

确定当前支持的协议

首先，使用 OpenSSL 创建一个自签名证书，以用于测试服务器和 Python 开发工具包。

```
$ openssl req -subj '/CN=localhost' -x509 -newkey rsa:4096 -nodes -keyout key.pem -out
 cert.pem -days 365
```

然后，使用 OpenSSL 启动测试服务器。

```
$ openssl s_server -key key.pem -cert cert.pem -www
```

在新的终端窗口中，创建虚拟环境并安装 Python 开发工具包。

```
$ python3 -m venv test-env
source test-env/bin/activate
pip install botocore
```

创建一个名为 check.py 的新 Python 脚本，该脚本使用此开发工具包的底层 HTTP 库。

```
$ import urllib3
URL = 'https://localhost:4433/'

http = urllib3.PoolManager(
ca_certs='cert.pem',
cert_reqs='CERT_REQUIRED',
)
r = http.request('GET', URL)
print(r.data.decode('utf-8'))
```

运行您的新脚本。

```
$ python check.py
```

这将显示有关所建立的连接的详细信息。在输出中搜索“协议：”。如果输出为“TLSv1.2”或更高版本，则开发工具包默认为 TLS v1.2 或更高版本。如果它是较早的版本，则需要重新编译 OpenSSL 并重新编译 Python。

但是，即使 Python 的安装默认为 TLS v1.2 或更高版本，如果服务器不支持 TLS v1.2 或更高版本，则 Python 仍可能重新协商到 TLS v1.2 之前的版本。要检查 Python 是否不会自动重新协商到较早版本，请使用以下命令重新启动测试服务器。

```
$ openssl s_server -key key.pem -cert cert.pem -no_tls1_3 -no_tls1_2 -www
```

如果您使用的是较早版本的 OpenSSL，则可能没有可用的 -no_tls_3 标志。如果是这种情况，请删除该标志，因为您使用的 OpenSSL 版本不支持 TLS v1.3。然后，运行 Python 脚本。

```
$ python check.py
```

如果您正确安装了 Python 但未针对 TLS 1.2 之前的版本进行重新协商，则应收到 SSL 错误。

```
$ urllib3.exceptions.MaxRetryError: HTTPSConnectionPool(host='localhost',
 port=4433): Max retries exceeded with url: / (Caused by SSLError(SSLError(1, '[SSL:
 UNSUPPORTED_PROTOCOL] unsupported protocol (_ssl.c:1108)')))
```

如果您能够建立连接，则需要重新编译 OpenSSL 和 Python 以禁用对早于 TLS v1.2 的协议进行协商。

编译 OpenSSL 和 Python

为了确保 AWS ParallelCluster 不对 TLS 1.2 之前的任何版本进行协商，您需要重新编译 OpenSSL 和 Python。要执行此操作，请复制以下内容以创建脚本并运行脚本。

```
#!/usr/bin/env bash
set -e

OPENSSL_VERSION="1.1.1d"
OPENSSL_PREFIX="/opt/openssl-with-min-tls1_2"
PYTHON_VERSION="3.8.1"
PYTHON_PREFIX="/opt/python-with-min-tls1_2"


curl -O "https://www.openssl.org/source/openssl-$OPENSSL_VERSION.tar.gz"
tar -xzf "openssl-$OPENSSL_VERSION.tar.gz"
cd openssl-$OPENSSL_VERSION
./config --prefix=$OPENSSL_PREFIX no-ssl3 no-tls1 no-tls1_1 no-shared
make > /dev/null
sudo make install_sw > /dev/null


cd /tmp
curl -O "https://www.python.org/ftp/python/$PYTHON_VERSION/Python-$PYTHON_VERSION.tgz"
tar -xzf "Python-$PYTHON_VERSION.tgz"
cd Python-$PYTHON_VERSION
./configure --prefix=$PYTHON_PREFIX --with-openssl=$OPENSSL_PREFIX --disable-shared > /
dev/null
make > /dev/null
sudo make install > /dev/null
```

这会编译具有静态链接的 OpenSSL 的 Python 版本，该版本不会自动协商早于 TLS 1.2 的任何版本。这也会在 `/opt/openssl-with-min-tls1_2` 目录中安装 OpenSSL，并在 `/opt/python-with-min-tls1_2` 目录中安装 Python。运行此脚本后，请确认安装新版本的 Python。

```
$ /opt/python-with-min-tls1_2/bin/python3 --version
```

这应该打印出以下内容。

```
Python 3.8.1
```

要确认此新版本的 Python 不协商早于 TLS 1.2 的版本，请使用新安装的 Python 版本（即 `/opt/python-with-min-tls1_2/bin/python3`）重新运行 确定当前支持的协议 中的步骤。

发行说明和文档历史记录

下表描述了 AWS ParallelCluster 用户指南 的主要更新和新功能。我们还经常更新文档来处理发送给我们的反馈意见。

变更	说明	日期
仅文档发布	AWS ParallelCluster 第 2 版特定用户指南已发布。 仅文档发布： • AWS ParallelCluster 版本 2 有自己的单独用户指南。	2023 年 7 月 17 日
AWS ParallelCluster 版本 2.11.9 已发布	AWS ParallelCluster 版本 2.11.9 已发布。 错误修复： • 防止替换托管FSx的 Lustre 文件系统，防止群集更新中包含更改`vpc_security_group_id` 的数据丢失。 有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包`CHANGELOG` 文件。 GitHub	2022 年 12 月 2 日
AWS ParallelCluster 版本 2.11.8 已发布	AWS ParallelCluster 2.11.8 版本已发布。	2022 年 11 月 14 日

更改：

- 将英特尔MPI库升级到 2021 版更新 6（从 2021 版更新 4 更新）。有关更多信息，请参阅英特尔® MPI 库 2021 年更新 6。
- 将EFA安装程序升级到 1.19.0
  - Efa-driver：`efa-1.16.0-1`
  - Efa-config：`efa-config-1.11-1`（从 `efa-config-1.9-1`）
  - Efa-profile：`efa-profile-1.5-1`（无变化）
  - Libfabric-aws：`libfabric-aws-1.16.0-1`（从 `libfabric-1.13.2`）
  - Rdma-core：`rdma-core-41.0-2`（从 `rdma-core-37.0`）
  - 打开MPI：`openmpi40-aws-4.1.4-3`（从`openmpi40-aws-4.1.1-2`）
- 将 Lambda 函数在 AWS Batch 集成中使用的 Python 运行时升级到 python3.9。

错误修复：

- 防止在更新期间更改集群标签，因为不支持更改。

	有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包的CHANGELOG 文件。GitHub	
AWS ParallelCluster 版本 2.11.7 已发布	AWS ParallelCluster 版本 2.11.7 已发布。 更改： • 将 Slurm 升级到版本 20.11.9。 有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包的CHANGELOG 文件。GitHub	2022 年 5 月 13 日

AWS ParallelCluster 版本 2.11.6 已发布	AWS ParallelCluster 版本 2.11.6 已发布。 增强功能： • 改进了网络中断时的异常管理。 更改： • 操作系统程序包更新和安全修复。 有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包的CHANGELOG 文件。GitHub	2022 年 4 月 19 日

AWS ParallelCluster 版本 2.11.5 已发布	AWS ParallelCluster 版本 2.11.5 已发布。	2022 年 3 月 1 日

增强功能：

- 为 Lustre `AutoImportPolicy` 选项添加FSx对`NEW_CHANGED_DELETED` 作为值的支持。
- 移除对SGE和 Torque 调度器的支持。
- 在 Amazon Linux 上禁用 `log4j-cve-2021-44228-hotpatch` 服务以避免引发潜在的性能降低。

更改：

- 将NVIDIA驱动程序升级到版本`470.103.01`（从`470.82.01`）。
- 将NVIDIA结构管理器升级到版本`470.103.01`（从`470.82.01`）。
- 将CUDA库升级到版本`11.4.4`（从`11.4.3`）。
- 英特尔MPI已更新至 2021 版更新 4（从 2019 年版本更新 8 更新）。有关更多信息，请参阅英特尔® MPI 库 2021 年更新 4。
- 将头节点创建超时时间延长至一小时。

错误修复：

- 修复通过浏览器的DCV连接。
- 修复YAML引号以防止自定义标签被解析为数字。

有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包的CHANGELOG 文件。GitHub

AWS ParallelCluster 版本 2.11.4 已发布	AWS ParallelCluster 2.11.4 版本已发布。 更改包括： • CentOS 已移除 8 支撑。CentOS 8 将于 2021 年 12 月 31 日达到生命尽头 (EOL)。 • Upgrade Slurm Workload Manager 到版本 20.11.8。 • 将 Cinc 客户端升级到 17.2.29。 • 亚马逊DCV已更新至亚马逊 DCV 2021.2-11190。有关更多信息，请参阅《亚马逊管理员指南》中的 DCV 2021.2-11190 — 2021 年 10 月 11 日。DCV • 将NVIDIA驱动程序升级到版本470.82.01 （从460.73.01 ）。 • 将CUDA库升级到版本11.4.3（从11.3.0）。 • 将NVIDIA结构管理器升级到470.82.01 。 • 在 Amazon Linux 2 上禁用实例启动时的程序包更新。 • 禁用无人值守的包裹更新 Ubuntu 还有亚马逊 Linux 2。 • 在上安装 Python 3 版本的AWS CloudFormation 帮助脚本 CentOS 7 和 Ubuntu 18.04。（它们已经在亚马	2021 年 12 月 20 日

逊 Linux 2 上使用过 Ubuntu 20.04。)

修复包括：

- 禁用 `ec2_iam_role` 参数更新。
- 修复启动模板中的`CpuOptions` 配置 T2 实例。

有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件包。aws-parallelcluster-cookbookaws-parallelcluster-node GitHub

AWS ParallelCluster 版本 2.11.3 已发布	AWS ParallelCluster 版本 2.11.3 已发布。	2021 年 11 月 3 日
	• 修复由于以下原因导致的pcluster createami失败 Son of Grid Engine 消息来源不在网址上 arc.liv.ac.uk 。	
	将 Elastic Fabric Adapter 安装程序升级到 1.14.1（从 1.13.0）	
	• EFA配置:efa-config-1.9-1 （来自efa-config-1.9 ） • EFA个人资料:efa-profile-1.5-1 （无变化） • EFA内核模块：efa-1.14.2 （来自efa-1.13.0 ） • RDMA核心：rdma-core-37.0 （来自rdma-core-35.0amzn ） • libfabric：libfabric-1.13.2 （从 libfabric-1.13.0amzn1.0 ） • 打开MPI：openmpi40-aws-4.1.1-2 （无变化）	
	GPUDirectRDMA如果实例类型支持，则始终处于启用状态。	

- `enable_efa_gdr` 和 `enable_efa_gdr` 配置选项没有效果。

有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件包。aws-parallelcluster-cookbookaws-parallelcluster-node GitHub

AWS ParallelCluster 版本 2.11.2 已发布	AWS ParallelCluster 2.11.2 版本已发布。
	
	更改包括：
	
	• 如果EFA安装EFA在基础AMI版中，则不要在引导时启用 GPUDirect RDMA (GDR) 的情况下进行安装。
	• 锁定`nvidia-fabricmanager` 软件包版本以与安装的NVIDIA驱动程序版本保持同步 AWS ParallelCluster。
	• Slurm：修复了节点处于开机状态下停止并重启集群时导致的问题。
	• Elastic Fabric Adapter 安装程序更新为 1.13.0：
	  • EFA配置:`efa-config-1.9` （没有变化）
	  • EFA个人资料:`efa-profile-1.5-1` （无变化）
	  • EFA内核模块:`efa-1.13.0` （无变化）
	  • RDMA核心：`rdma-core-35.0amzn` （来自`rdma-core-32.1amzn` ）
	  • libfabric：`libfabric-1.13.0amzn1.0` （从`libfabric-1.11.2amzn1.1` ）
		2021 年 8 月 27 日

- 打开MPI：`openmpi40-aws-4.1.1-2` （无变化）
- 使用AMI带有预装EFA软件包的自定义软件包时，EFA在节点引导时不会进行任何更改。保留原始EFA软件包部署。

有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和软件包。aws-parallelcluster-cookbook GitHub

AWS ParallelCluster 版本 2.11.1 已发布	AWS ParallelCluster 2.11.1 版本已发布。
	
	更改包括：
	
	• 使用 noatime 挂载选项挂载文件系统以在读取文件时停止记录上次访问时间。这提高了远程文件系统的性能。
	• Elastic Fabric Adapter 安装程序更新为 1.12.3：
	  • EFA配置:efa-config-1.9 （来自efa-config-1.8-1 ）
	  • EFA个人资料:efa-profile-1.5-1 （无变化）
	  • EFA内核模块：efa-1.13.0 （来自efa-1.12.3 ）
	  • RDMA核心:rdma-core-32.1amzn （无变化）
	  • Libfabric：libfabric-1.11.2amzn1.1 （无更改）
	  • 打开MPI：openmpi40-aws-4.1.1-2 （无变化）
	• AWS Batch 用作调度程序时，请重试在头节点上安装 aws-parallelcluster 软件包。
	• 建造时避免故障 SGE 在超过 31 的实例类型上vCPUs。
		2021 年 7 月 23 日

- 已固定到 A CloudWatch mazon Agent 的 1.247347.6 版本，以避免 1.247348.0 版本中出现的问题。

有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和软件包。aws-parallelcluster-cookbook GitHub

AWS ParallelCluster 2.11.0 版本已发布	AWS ParallelCluster 2.11.0 版本已发布。
	
	更改包括：
	
	• 添加了对以下内容的支持 Ubuntu 20.04 (ubuntu2004) 并删除了对的支持 Ubuntu 16.04 (ubuntu1604) 和亚马逊 Linux (alinux)。仍然完全支持 Amazon Linux 2 (alinux2)。有关更多信息，请参阅 base_os。
	• 删除了对 3.6 以下 Python 版本的支持。
	• 默认根卷大小增加到 35 吉字节 (GiB)。有关更多信息，请参阅compute_root_volume_size 和master_root_volume_size 。
	• Elastic Fabric Adapter 安装程序更新为 1.12.2：
	  • EFA配置:efa-config-1.8-1 （来自efa-config-1.7 ）
	  • EFA个人资料:efa-profile-1.5-1 （来自efa-profile-1.4 ）
	  • EFA内核模块：efa-1.12.3 （来自efa-1.10.2 ）
	  • RDMA核心：rdma-core-32.1amzn
		2021 年 7 月 1 日

（来自rdma-core -31.2amzn ）

- libfabric：libfabric -1.11.2amzn1.1 （从libfabric-1.11.1amzn1.0 ）
- 打开MPI：openmpi40 -aws-4.1. 1-2 （从openmpi40-aws-4.1.0 ）

- 已升级 Slurm 到版本20.11.7（从20.02.7）。
- 在centos7和上安装SSM代理centos8。（SSM代理已预安装在alinux2ubuntu1804 、和ubuntu2004 。）
- SGE: 请务必使用短名称作为主机名过滤器。qstat
- 使用实例元数据服务版本 2 (IMDSv2) 而不是实例元数据服务版本 1 (IMDSv1) 来检索实例元数据。有关更多信息，请参阅 Amazon 用户指南中的实例元数据和EC2用户数据。
- 将NVIDIA驱动程序升级到版本460.73.01 （从450.80.02 ）。
- 将CUDA库升级到版本11.3.0（从11.0）。
- 将NVIDIA结构管理器升级到nvidia-fabricmanager-460 。

- 将 v AWS ParallelC luster irtualenvs 中使用的 Python 升级到3.7.10（从3.6.13）。
- 将 Cinc 客户端升级到 16.13.16。
- 升级以下第三方依赖项 aws-parallelcluster-cookbook：
  - apt-7.4.0 （从 apt-7.3.0 ）。
  - iptables-8.0.0 （从 iptables-7.1.0 ）。
  - line-4.0.1 （从 line-2.9.0 ）。
  - openssh-2.9.1 （从 openssh-2.8.1 ）。
  - pyenv-3.4.2 （从 pyenv-3.1.1 ）。
  - selinux-3.1.1 （从 selinux-2.1.1 ）。
  - ulimit-1.1.1 （从 ulimit-1.0.0 ）。
  - yum-6.1.1 （从 yum-5.1.0 ）。
  - yum-epel-4.1.2 （从 yum-epel-3.3.0 ）。

有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub

AWS ParallelCluster 2.10.4 版本已发布	AWS ParallelCluster 2.10.4 版本已发布。 更改包括： • 已升级 Slurm 到版本20.02.7（从20.02.4）。 有关更改的更多详细信息，请参阅上的 aws-parallel cluster 软件包CHANGELOG文件。GitHub	2021 年 5 月 15 日

AWS ParallelCluster 2.10.3 版本已发布	AWS ParallelCluster 2.10.3 版本已发布。
	
	更改包括：
	
	• 添加了对以下内容的支持 Ubuntu 18.04 和 Amazon Linux 2 在中国基于 ARM 的 AWS Graviton 实例上以及。AWS AWS GovCloud (US) AWS 区域
	• Elastic Fabric Adapter 安装程序更新为 1.11.2：
	  • EFA配置:`efa-config-1.7` （没有变化）
	  • EFA个人资料:`efa-profile-1.4` （来自`efa-profile-1.3` ）
	  • EFA内核模块:`efa-1.10.2` （无变化）
	  • RDMA核心:`rdma-core-31.2amzn` （无变化）
	  • Libfabric：`libfabric-1.11.1amzn1.0` （无更改）
	  • 打开MPI：`openmpi40-aws-4.1.0` （无变化）
	
	有关更改的更多详细信息，请参阅上的 aws-parallel cluster 软件包CHANGELOG文件。GitHub
		2021 年 3 月 18 日

AWS ParallelCluster 2.10.2 版本已发布	AWS ParallelCluster 2.10.2 版本已发布。 更改包括： • 改进集群配置验证，以便在`--dry-run` 模式下调用 Amazon EC2 RunInstancesAPI操作AMI时使用集群目标。 • 将 AWS ParallelCluster 虚拟环境中使用的 Python 版本更新到 3.6.13。 • 修复了 Arm 实例类型的`sanity_check` 。 • `centos8`与一起使用`enable_efa` 时已修复 Slurm 调度器或 Arm 实例类型。 • 在非交互模式 (`-y`) 下运行`apt update`。 • 对 `alinux2` 和 `centos8` 修复了`encrypted_ephemeral` = true。 有关更改的更多详细信息，请参阅上的 aws-parallel cluster 软件包CHANGELOG文件。GitHub	2021 年 3 月 2 日

AWS ParallelCluster 2.10.1 版本已发布	AWS ParallelCluster 2.10.1 版本已发布。
	
	更改包括：
	
	• 增加了对非洲（开普敦）(af-south-1)、欧洲（米兰）（me-south-1 ）和中东（巴林）（me-south-1 ）的支持 AWS 区域。在发布时，支持受到以下限制：
	  • FSx其中任何一个都不支持基于 Lustre 和基于 ARM 的 Graviton 实例。AWS 区域
	  • AWS Batch 在非洲（开普敦）不支持。
	  • 非洲（开普敦）EBSio2和欧洲（米兰）不支持 Amazon 和gp3卷类型 AWS 区域。
	• 增加了对 Amazon EBS io2 和gp3卷类型的支持。有关更多信息，请参阅 [ebs] 部分和 [raid] 部分。
	• 在运行 alinux2、ubuntu1804 或 ubuntu2004 的基于 Arm 的 Graviton2 实例上添加了对 Elastic Fabric Adapter 的支持。有关更多信息，请参阅 Elastic Fabric Adapter。
		2020 年 12 月 22 日

- 在 ArmAMIs（alinux2、centos8和ubuntu1804 ）上安装 Arm 性能库 20.2.1。有关更多信息，请参阅 Arm Performance Libraries。
- 英特尔MPI已更新至 2019 版更新 8（从 2019 年版本更新 7 更新）。有关更多信息，请参阅英特尔® MPI 库 2019 年更新 8。
- 从 AWS Batch Docker 入口点移除了 AWS CloudFormation DescribeStacks API操作调用，以结束因限制而导致的作业失败。 AWS CloudFormation
- 改进了验证集群配置时EC2DescribeInstanceTypes API对 Amazon 操作调用的调用。
- Amazon Linux 2 Docker 镜像是在为调度程序构建 Docker 镜像时从亚马逊 ECR公共镜像中提取的 。awsbatch
- 的默认实例类型从硬编码t2.micro实例类型更改为免费套餐实例类型 AWS 区域（t2.micro或t3.micro，视情况而定 AWS 区域）。AWS 区域 没有免费套餐的t3.micro实例类型默认。

- Elastic Fabric Adapter 安装程序更新为 1.11.1：
  - EFA配置:efa-config-1.7 （来自efa-config-1.5 ）
  - EFA个人资料:efa-profile-1.3 （来自efa-profile-1.1 ）
  - EFA内核模块:efa-1.10.2 （无变化）
  - RDMA核心：rdma-core-31.2amzn （来自rdma-core-31.amzn0 ）
  - libfabric：libfabric-1.11.1amzn1.0 （从libfabric-1.10.1amzn1.1 ）
  - 打开MPI：openmpi40-aws-4.1.0 （从openmpi40-aws-4.0.5 ）
- vpc_settings 、vpc_id 和 master_subnet_id 参数现在是必需参数。
- 头节点中的 nfsd 进程守护程序现在设置为使用至少 8 个线程。如果内核超过 8 个，它将使用与内核数量一样多的线程。使用 ubuntu1604 时，该设置仅在节点重启后才会更改。
- 亚马逊DCV已更新为亚马逊 DCV 2020.2-9662。有

关更多信息，请参阅《亚马逊管理员指南》中的 DCV 2020.2-9662 — 2020 年 12 月 4 日。DCV

- 英特尔MPI和的HPC软件包 AWS ParallelCluster 是从 Amazon S3 中提取的。不再从 Intel yum 存储库中拉取。
- 更改了默认值 systemd `multi-user.target` 在创建官方 AWS ParallelCluster AMIs版本OSs期间，全部运行等级。仅当启用时，头节点`graphical.target` 上的运行级别才会设置DCV为。这样可以防止图形服务（例如 `x/gdm`）在不需要时运行。
- 在头节点上启用了对 `p4d.24xlarge` 实例的支持。
- 增加注册时的最大重试次数 Slurm 亚马逊 Route 53 中的节点。

有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub

AWS ParallelCluster 2.10.0 版本已发布	AWS ParallelCluster 2.10.0 版本已发布。
	
	更改包括：
	
	• 添加了对以下内容的支持 CentOS 总共有 8 个 AWS 区域（ AWS 中国和 AWS GovCloud （美国）地区以外）。已移除对的支持 CentOS 6.
	• 为计算节点添加了对 `p4d.24xlarge` 实例的支持。
	• 使用新`enable_efa_gdr` 设置添加NVIDIAGPUDirectRDMAEFA了对开启的支持。
	• 增加了对 Amazon for Lustre 功能FSx的支持。
	  • 使用该设置将您FSx的 Amazon for Lustre 文件系统配置为导入首选项。`auto_import_policy`
	  • 使用和`drive_cache_type` 设置增加了对HDD基FSx于 Amazon for Lustre 文件系统的支持。`storage_type`
	• 添加了 Amazon CloudWatch 控制面板，包括头节点指标和轻松访问集群日志。有关更多信息，请参阅 亚马逊 CloudWatch 控制面板。
		2020 年 11 月 18 日

- 使用 cluster_resource_bucket 设置，添加了对使用现有 Amazon S3 存储桶存储集群配置信息的支持。
- 增强了 pcluster createami 命令。
  - 添加了在构建时使用安装后脚本的--post-install AMI参数。
  - 添加了一个验证步骤，当使用由不同版本的AMI创建的基础时，验证步骤会失败 AWS ParallelCluster。
  - 添加了一个验证步骤，当所选操作系统与基础操作系统不同时，验证步骤将失败AMI。
  - 增加了对使用 AWS ParallelCluster 底座的支持AMI。
- 增强了 pcluster update 命令。
  - 现在可以在更新期间更改 tags 设置。
  - 现在可以在更新期间调整队列的大小，而无需停止计算实例集
- 为 slurm_resume 脚本添加了 all_or_nothing_batch 配置参数。何时True，仅当中所有待处理任务所需的所有实例都满足时，才slurm_res

ume 会成功 Slurm 将可用。如需了解更多信息，请参阅上的 AWS ParallelCluster Wiki 中的all_or_nothing_batch 产品发布简介 GitHub。

- Elastic Fabric Adapter 安装程序更新为 1.10.1：
  - EFA配置:efa-config-1.5 （来自efa-config-1.4 ）
  - EFA个人资料:efa-profile-1.1 （来自efa-profile-1.0.0 ）
  - EFA内核模块：efa-1.10.2 （来自efa-1.6.0 ）
  - RDMA核心：rdma-core-31.amzn0 （来自rdma-core-28.amzn0 ）
  - libfabric：libfabric-1.11.1amzn1.0 （从libfabric-1.10.1amzn1.1 ）
  - 打开MPI：openmpi40-aws-4.0.5 （从openmpi40-aws-4.0.3 ）
- 在 AWS GovCloud (US) 区域中，启用对 Amazon DCV 和 AWS Batch。

- AWS 在中国区域，启用对 Amazon for Lustre FSx 的支持。
- 将NVIDIA驱动程序升级到版本 450.80.02（从 450.51.05 开始）。
- 安装 NVIDIA Fabric Manager 以NVIDIANV Switch在支持的平台上启用。
- 移除了默认值 AWS 区域 `us-east-1` 。默认值使用以下查找顺序。
  - AWS 区域 在`-r`或`--region`参数中指定。
  - `AWS_DEFAULT_REGION` 环境变量。
  - `aws_region_name` 在 AWS ParallelCluster 配置文件`[aws]`部分中设置（默认为`~/.parallelcluster/config`）。
  - `region`在 AWS CLI 配置文件`[default]` 部分中设置（默认为`~/aws/config` ）。

有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub

AWS ParallelCluster 2.9.0 版本已发布	AWS ParallelCluster 2.9.0 版本已发布。 更改包括： • 与一起使用时，增加了对计算队列中的多个队列和多种实例类型的支持 Slurm Workload Manager使用队列时， 上将不再使用自动扩缩组。Slurm。 现在，已在集群中创建了 Amazon Route 53 托管区域，用于DNS解析计算节点 Slurm 使用了调度器。有关更多信息，请参阅多队列模式。 • 在基于 ARM 的 AWS Gravit on DCV 实例上增加了对 Amazon 的支持。 • 增加了对不支持启动模板中CPU选项的实例类型（例如实例类型）禁用超线`*.metal`程的支持。 • 为从头节点共享的文件系统添加了对 NFS 4 的支持。 • 移除了在引导计算节点时对 cfn-in it 的依赖，以避免在大量节点加入集群 AWS CloudFormation 时受到限制。 • Elastic Fabric Adapter 安装程序更新为 1.9.5： • EFA配置:`efa-config-1.4` （来自`efa-config-1.3` ）	2020 年 9 月 11 日

  - EFA个人资料:`efa-profile-1.0.0` （新）
  - 内核模块：`efa-1.6.0` （无更改）
  - RDMA核心:`rdma-core-28.amzn0` （无变化）
  - Libfabric：`libfabric-1.10.1amzn1.1` （无更改）
  - 打开MPI：`openmpi40-aws-4.0.3` （无变化）
- 已升级 Slurm 到版本`20.02.4`（从`19.05.5`）。
- 亚马逊DCV已更新至亚马逊 DCV 2020.1-9012。有关更多信息，请参阅《亚马逊管理员指南》中的 DCV 2020.1-9012 — 2020 年 8 月 24 日发行说明。DCV
- 挂载共享云NFS端硬盘时，请使用头节点私有 IP 地址而不是主机名。
- 在 Logs 中添加了新的 CloudWatch 日志流：`chef-client clustermgtd computemgtd`、`slurm_resume`、、和`slurm_suspend` 。
- 在预安装和安装后脚本中添加了对队列名称的支持。
- 在中 AWS GovCloud (US) AWS 区域，使用亚马逊 DynamoDB 按需计费选项。有关更多信息，请参阅

	Amazon DynamoDB 开发人员指南 中的按需模式。 有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	
AWS ParallelCluster 版本 2.8.1 已发布	AWS ParallelCluster 版本 2.8.1 已发布。 更改包括： • 禁用 Amazon DCV 会话的屏幕锁定，以防止用户被锁定。 • 修复了包含基于 ARM 的 AWS Graviton 实例类型时的 `pcluster configure` 。 有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	2020 年 8 月 4 日

AWS ParallelCluster 2.8.0 版本已发布	AWS ParallelCluster 2.8.0 版本已发布。
	
	更改包括：
	
	• 增加了对基于 ARM 的基于 AWS Graviton 的实例（如 A1 和 C6g）的支持。
	• 增加了对 Amazon for Lustre 的每日自动备份功能FSx的支持。有关更多信息，请参阅 automatic_backup_retention_days、copy_tags_to_backups、daily_automatic_backup_start_time 和 fsx_backup_id。
	• 从 pcluster createami 中删除了对 Berkshelf 的依赖。
	• 改进了 pcluster update 的可靠性和用户体验。有关更多信息，请参阅 使用 pcluster update。
	• Elastic Fabric Adapter 安装程序更新为 1.9.4：
	  • 内核模块：efa-1.6.0（从 efa-1.5.1 进行更新）
	  • RDMA核心:rdma-core-28.amzn0（更新自rdma-core-25.0）
		2020 年 7 月 23 日

    - Libfabric：`libfabric-1.10.1amzn1.1` （从 `libfabric-aws-1.9.0amzn1.1` 进行更新）
    - 打开MPI：`openmpi40-aws-4.0.3` （无变化）
- 将NVIDIA驱动程序升级到特斯拉版本 440.95.01 CentOS 6 和 450.51.05 版本适用于所有其他发行版。
- 在除此之外的所有发行版上将CUDA库升级到 11.0 版 CentOS 6.

有关更改的更多详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub

AWS ParallelCluster 2.7.0 版本已发布	AWS ParallelCluster 2.7.0 版本已发布。 更改包括： • base_os 现在是必需的参数。 • scheduler 现在是必需的参数。 • 亚马逊DCV已更新至亚马逊 DCV 2020.0。有关更多信息，请参阅 Amazon DCV 发布的支持环绕声 7.1 和触控笔的 2020.0 版。 英特尔MPI已更新至 2019 版更新 7（从 2019 年版本更新 6 更新）。有关更多信息，请参阅英特尔® MPI 库 2019 年更新 7。 Elastic Fabric Adapter 安装程序更新为 1.8.4： • 内核模块：efa-1.5.1（无更改） • RDMA核心:rdma-core-25.0 （无变化） • Libfabric：libfabric-aws-1.9.0amzn1.1 （无更改） • 打开MPI:openmpi40-aws-4.0.3 （更新自openmpi40-aws-4.0.2 ） • Upgrade CentOS 7 AMI 到 7.8-2003 版本（从 7.7-1908	2020 年 5 月 19 日

	年更新）。有关更多信息，请参阅 CentOS-7 (2003) 发行说明。	
AWS ParallelCluster 版本 2.6.1 已发布	AWS ParallelCluster 版本 2.6.1 已发布。 更改包括： • 已`cfn-wire`从存储在 Amazon 日志中的 CloudWatch 日志中移除`cfn-init-cmd` 和。有关更多信息，请参阅 与 Amazon CloudWatch 日志集成。	2020 年 4 月 17 日

AWS ParallelCluster 2.6.0 版本已发布	AWS ParallelCluster 2.6.0 版本已发布。 更改包括： • 添加了对 Amazon Linux 2 的支持 • 现在，Amazon CloudWatch 日志用于收集集群和计划程序日志。有关更多信息，请参阅 与 Amazon CloudWatch 日志集成。 • 增加了对全新 Amazon f FSx or Lustre 部署类型的支持SCRATCH_2 ，以及。PERSISTENT_1 开启 Support FSx for Lustre Ubuntu 18.04 和 Ubuntu 16.04。有关更多信息，请参阅 fsx。 • 增加了对 Amaz DCV on 的支持 Ubuntu 18.04。有关更多信息，请参阅 通过 Amazon 连接到头节点 DCV。	2020 年 2 月 27 日
AWS ParallelCluster 版本 2.5.1 已发布	AWS ParallelCluster 版本 2.5.1 已发布。	2019 年 12 月 13 日
AWS ParallelCluster 版本 2.5.0 已发布	AWS ParallelCluster 版本 2.5.0 已发布。	2019 年 11 月 18 日
AWS ParallelCluster 引入了对英特尔的支持 MPI	AWS ParallelCluster 版本 2.4.1 引入了对英特尔的MPI支持。	2019 年 7 月 29 日

AWS ParallelCluster 引入了对的支持 EFA	AWS ParallelCluster 版本 2.4.0 引入了对弹性结构适配器 (EFA) 的支持。	2019 年 6 月 11 日
AWS ParallelCluster 文档已在 AWS 文档网站上发布	该 AWS ParallelCluster 文档现在有 10 种语言版本，有两种语言HTML和两种PDF格式。	2018 年 5 月 24 日

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。