AWS ParallelCluster Guia do usuário (v2)

AWS ParallelCluster

AWS ParallelCluster: AWS ParallelCluster Guia do usuário (v2)

Table of Contents

O que é AWS ParallelCluster

AWS ParallelCluster é uma ferramenta de gerenciamento de cluster de código aberto AWS compatível que ajuda você a implantar e gerenciar clusters de computação de alto desempenho (HPC) no Nuvem AWS. Ele configura automaticamente os recursos de computação, agendador e o sistema de arquivos compartilhado necessários. Você pode usar AWS ParallelCluster com AWS Batch e Slurm agendadores.

Com AWS ParallelCluster, você pode criar e implantar rapidamente ambientes computacionais de HPC de prova de conceito e produção. Além disso, você pode criar e implantar um fluxo de trabalho de alto nível AWS ParallelCluster, como um portal de genômica que automatiza todo o fluxo de trabalho de sequenciamento de DNA.

Preços

Ao usar a interface de linha de AWS ParallelCluster comando (CLI) ou a API, você paga apenas pelos AWS recursos criados ao criar ou atualizar AWS ParallelCluster imagens e clusters. Para obter mais informações, consulte AWS serviços usados por AWS ParallelCluster.

Conf AWS ParallelCluster iguração

Tópicos

Instalando AWS ParallelCluster

AWS ParallelCluster é distribuído como um pacote Python e é instalado usando o gerenciador `pip` de pacotes Python. Para obter mais informações sobre como instalar pacotes Python, consulte Instalar pacotes no Guia do usuário de pacotes do Python.

Formas de instalar AWS ParallelCluster:

- Usando um ambiente virtual (recomendado)
- Como usar o `pip`

Você pode encontrar o número da versão da CLI mais recente na página de lançamentos em. GitHub

Neste guia, os exemplos de comando pressupõem que você tenha Python v3 instalado. Os exemplos do comando `pip` usam a versão `pip3`.

Instalação AWS ParallelCluster em um ambiente virtual (recomendado)

Recomendamos que você instale AWS ParallelCluster em um ambiente virtual. Se você encontrar problemas ao tentar instalar AWS ParallelCluster com`pip3`, poderá instalar AWS ParallelCluster em um ambiente virtual para isolar a ferramenta e suas dependências. Ou você pode usar uma versão do Python diferente da que você normalmente usa.

Instalando AWS ParallelCluster em um ambiente não virtual usando pip

O principal método de distribuição para AWS ParallelCluster Linux, Windows e macOS é`pip`, que é um gerenciador de pacotes para Python. Ele fornece uma maneira de instalar, atualizar e remover pacotes Python e suas dependências.

AWS ParallelCluster Versão atual

AWS ParallelCluster é atualizado regularmente. Para determinar se você tem a versão mais recente, consulte a página de lançamentos em GitHub.

Se você já tem `pip` uma versão compatível do Python, pode instalar AWS ParallelCluster usando o comando a seguir. Se você tiver o Python versão 3+ instalado, recomendamos usar o comando **`pip3`**.

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

Etapas a serem realizadas após a instalação

Depois da instalação AWS ParallelCluster, talvez seja necessário adicionar o caminho do arquivo executável à sua `PATH` variável. Para instruções específicas da plataforma, consulte os seguintes tópicos:

- Linux: Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando
- macOS: Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando
- Windows: Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando

Você pode verificar se AWS ParallelCluster está instalado corretamente executando`pcluster version`.

```
$ pcluster version
2.11.9
```

AWS ParallelCluster é atualizado regularmente. Para atualizar para a versão mais recente do AWS ParallelCluster, execute o comando de instalação novamente. Para obter detalhes sobre a versão mais recente do AWS ParallelCluster, consulte as notas AWS ParallelCluster de lançamento.

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

Para desinstalar AWS ParallelCluster, use`pip uninstall`.

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

Se não tem o Python nem o `pip`, use o procedimento para o seu ambiente.

Instruções detalhadas para cada ambiente

- Instalar AWS ParallelCluster em um ambiente virtual (recomendado)
- Instale AWS ParallelCluster no Linux
- Instale AWS ParallelCluster no macOS
- Instalar AWS ParallelCluster no Windows

Instalar AWS ParallelCluster em um ambiente virtual (recomendado)

Recomendamos que você instale AWS ParallelCluster em um ambiente virtual para evitar conflitos de versão de requisitos com outros `pip` pacotes.

Pré-requisitos

- Verifique se `pip` e Python estão instalados. Recomendamos `pip3` e Python 3 versão 3.8. Se você estiver usando o Python 2, use `pip` em vez de `pip3` e `virtualenv` em vez de `venv`.

Para instalar AWS ParallelCluster em um ambiente virtual

1. Se o `virtualenv` não estiver instalado, instale o `virtualenv` usando `pip3`. Se `python3 -m virtualenv help` exibir informações de ajuda, vá para a etapa 2.

   Linux, macOS, or Unix

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

   Execute `exit` para sair da janela do terminal atual e abrir uma nova janela de terminal para selecionar as alterações no ambiente.

Windows

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

Execute `exit` para sair do prompt de comando atual e abra um novo prompt de comando para selecionar alterações no ambiente.

2. Crie um ambiente virtual e nomeie-o.

   Linux, macOS, or Unix

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

   Como alternativa, você pode usar a opção `-p` para especificar uma versão do Python.

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

   Windows

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

3. Ative seu novo ambiente virtual.

   Linux, macOS, or Unix

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

   Windows

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

4. Instale AWS ParallelCluster em seu ambiente virtual.

   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. Verifique se AWS ParallelCluster está instalado corretamente.

   Linux, macOS, or Unix

   ```
   $ pcluster version
   2.11.9
   ```

   Windows

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

Use o comando `deactivate` para sair do ambiente virtual. Toda vez que iniciar uma sessão, é necessário reativar o ambiente.

Para atualizar para a versão mais recente do AWS ParallelCluster, execute o comando de instalação novamente.

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"
```

Instale AWS ParallelCluster no Linux

Você pode instalar AWS ParallelCluster e suas dependências na maioria das distribuições Linux usando um gerenciador `pip` de pacotes para Python. Primeiro, determine se o Python e o `pip` estão instalados:

1. Para determinar se a sua versão do Linux inclui Python e `pip`, execute `pip --version`.

```
$ pip --version
```

Se você `pip` instalou, vá para o tópico Instalar AWS ParallelCluster com pip. Caso contrário, continue na Etapa 2.

2. Para determinar se o Python está instalado, execute `python --version`.

```
$ python --version
```

Se você tiver o Python 3 versão 3.6+ ou o Python 2 versão 2.7 instalado, vá para o tópico Instalar com pip. AWS ParallelCluster Caso contrário, instale o Python e retorne a este procedimento para instalar o `pip`.

3. Instale o `pip` usando o script fornecido pela Python Packaging Authority.
4. Use o comando `curl` a seguir para baixar o script de instalação.

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

5. Execute o script com Python para fazer download e instalar a versão mais recente do `pip` e de outros pacotes de suporte necessários.

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

or

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

Quando você inclui a chave `--user`, o script instala o `pip` no caminho `~/.local/bin`.

6. Para garantir que a pasta que contém `pip` faça parte da variável `PATH`, faça o seguinte:

   a. Encontre o script de perfil do shell em sua pasta de usuário. Se não tiver certeza de qual shell você tem, execute `basename $SHELL`.

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

   - Bash: `.bash_profile`, `.profile` ou `.bash_login`
   - Zsh: `.zshrc`

- Tcsh: `.tcshrc`, `.cshrc` ou `.login`.

b. Adicione um comando de exportação ao final do script de perfil que é semelhante ao exemplo a seguir.

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

O comando de exportação insere o caminho, que é `~/.local/bin` neste exemplo, na frente da variável PATH existente.

c. Para colocar essas alterações em vigor, recarregue o perfil em sua sessão atual.

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

7. Verifique se o `pip` está instalado corretamente.

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

Seções

- Instale AWS ParallelCluster com pip
- Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando
- Instalar o Python no Linux

Instale AWS ParallelCluster com **`pip`**

Use `pip` para instalar AWS ParallelCluster.

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

Ao usar a chave `--user`, o `pip` instala o AWS ParallelCluster em `~/.local/bin`.

Verifique se AWS ParallelCluster está instalado corretamente.

```
$ pcluster version
2.11.9
```

Para atualizar para a versão mais recente, execute o comando de instalação novamente.

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

Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando

Após a instalação com o `pip`, talvez seja necessário adicionar o executável `pcluster` à variável de ambiente `PATH` do seu sistema operacional.

Para verificar a pasta na qual `pip` está instalado AWS ParallelCluster, execute o comando a seguir.

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

Se você omitiu a `--user` opção ao instalar AWS ParallelCluster, o executável pode estar na `bin` pasta da instalação do Python. Se você não souber onde o Python está instalado, execute este comando.

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

Observe que a saída pode ser o caminho para um symlink, não para o executável em si. Para ver onde o symlink aponta, execute `ls -al`.

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

Se essa for a mesma pasta que você adicionou ao caminho na etapa 3 em Instalando AWS ParallelCluster, a instalação estará concluída. Caso contrário, será necessário realizar as etapas 3a a 3c novamente, incluindo essa pasta adicional ao caminho.

Instalar o Python no Linux

Se sua distribuição não veio com o Python ou veio com uma versão anterior, instale o Python antes de instalar e. `pip` AWS ParallelCluster

Para instalar Python 3 no Linux

1. Verifique se o Python já está instalado.

   ```
   $ python3 --version
   ```

or

```
$ python --version
```

> **Note**
>
> Se sua distribuição do Linux acompanha Python, poderá ser necessário instalar o pacote de desenvolvedor Python. O pacote de desenvolvedor inclui os cabeçalhos e as bibliotecas que são necessárias para compilar extensões e instalar o AWS ParallelCluster. Use o gerenciador de pacotes para instalar o pacote de desenvolvedor. Geralmente é chamado de `python-dev` ou `python-devel`.

2. Se Python 2.7 ou posterior não estiver instalado, instale Python com o gerenciador de pacote de distribuição. O comando e o nome do pacote varia de:

   - No derivados do Debian, como Ubuntu, use `apt`.

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

   - No Red Hat e derivados, use `yum`.

     ```
     $ sudo yum install python3
     ```

   - No SUSE e derivados, use o `zypper`.

     ```
     $ sudo zypper install python3
     ```

3. Para verificar se o Python foi instalado corretamente, abra um prompt de comando ou shell e execute o comando a seguir.

   ```
   $ python3 --version
   Python 3.8.11
   ```

Instale AWS ParallelCluster no macOS

Seções

- Pré-requisitos

- Instale AWS ParallelCluster no macOS usando pip
- Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando

Pré-requisitos

- Python 3 versão 3.7+ ou Python 2 versão 2.7

Verifique a instalação do Python.

```
$ python --version
```

Se o Python ainda não foi instalado no computador ou se você deseja instalar uma versão diferente do Python, siga o procedimento em Instale AWS ParallelCluster no Linux.

Instale AWS ParallelCluster no macOS usando pip

Você também pode usar `pip` diretamente para instalar AWS ParallelCluster. Se você não tem o `pip`, siga as instruções no tópico de instalação principal. Execute o `pip3 --version` para ver se a sua versão do macOS já inclui o Python e o `pip3`.

```
$ pip3 --version
```

Para instalar AWS ParallelCluster no macOS

1. Faça download e instale a versão mais recente do Python da página de download em Python.org.
2. Faça download e execute o script de instalação `pip3` fornecido pela Python Packaging Authority.

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

3. Use o recém-instalado `pip3` para instalar AWS ParallelCluster. Se você usar o Python versão 3+, recomendamos usar o comando `pip3`.

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

4. Verifique se AWS ParallelCluster está instalado corretamente.

```
$ pcluster version
2.11.9
```

Se o programa não for encontrado, adicione-o ao caminho da linha de comando.

Para atualizar para a versão mais recente, execute o comando de instalação novamente.

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

Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando

Após a instalação com o `pip`, talvez seja necessário adicionar o programa `pcluster` à variável de ambiente PATH do seu sistema operacional. A localização do programa depende de onde o Python está instalado.

Example AWS ParallelCluster local de instalação - macOS com Python 3.6 e (modo de usuário) **`pip`**

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

Substitua a versão do Python que você tem pela versão no exemplo anterior.

Se você não souber onde o Python está instalado, execute `which python`.

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

A saída pode ser o caminho para um symlink, e não o caminho para programa real. Execute `ls -al` para saber para onde ele aponta.

```
$ 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
```

O `pip` instala programas na mesma pasta que contém a aplicação Python. Adicione esta pasta à variável PATH.

Para modificar a variável **PATH** (Linux, Unix ou macOS)

1. Encontre o script de perfil do shell em sua pasta de usuário. Se não tiver certeza de qual shell você tem, execute `echo $SHELL`.

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

   - Bash: `.bash_profile`, `.profile` ou `.bash_login`
   - Zsh: `.zshrc`
   - Tcsh – `.tcshrc`, `.cshrc`, ou `.login`
2. Adicione um comando de exportação ao script de perfil.

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

   Este comando adiciona um caminho, `~/.local/bin` neste exemplo, para a variável `PATH` atual.
3. Carregue o perfil em sua sessão atual.

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

Instalar AWS ParallelCluster no Windows

Você pode instalar AWS ParallelCluster no Windows usando`pip`, que é um gerenciador de pacotes para Python. Caso já tenha `pip`, siga as instruções no tópico de instalação principal.

Seções

- Instale AWS ParallelCluster usando Python e pip no Windows
- Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando

Instale AWS ParallelCluster usando Python e **`pip`** no Windows

O Python Software Foundation fornece instaladores para Windows que incluem `pip`.

Para instalar o Python e o **`pip`** (Windows)

1. Faça download do instalador do Python Windows x86-64 na página de downloads do Python.org.
2. Execute o instalador.
3. Escolha Add Python 3 to PATH (Adicionar Python 3 ao PATH).
4. Escolha Instalar agora.

O instalador instala o Python em sua pasta de usuário e adiciona suas pastas do programa ao caminho do usuário.

Para instalar AWS ParallelCluster com **`pip3`** (Windows)

Se você usar o Python versão 3+, recomendamos usar o comando `pip3`.

1. Abra o Command Prompt (Prompt de comando) no menu Start (Iniciar).
2. Use os seguintes comandos para verificar se o Python e o `pip` estão instalados corretamente.

```
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. Instale AWS ParallelCluster usando`pip`.

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

4. Verifique se AWS ParallelCluster está instalado corretamente.

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

Para atualizar para a versão mais recente, execute o comando de instalação novamente.

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

Adicione o AWS ParallelCluster executável ao seu caminho de linha de comando

Depois de instalar AWS ParallelCluster com`pip`, adicione o `pcluster` programa à variável de PATH ambiente do seu sistema operacional.

É possível encontrar onde o programa `pcluster` está instalado executando o comando a seguir.

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

Se esse comando não retornar nenhum resultado, você deverá adicionar o caminho manualmente. Use a linha de comando ou o Windows Explorer para descobrir onde ele está instalado no computador. Os caminhos típicos incluem:

- Python 3 e **`pip3`** – `C:\Python38\Scripts\`
- Python 3 e **`pip3`** --opção do usuário – `%APPDATA%\Python\Python38\Scripts`

> **Note**
>
> Os nomes de pasta que incluem os números de versão podem variar. Os exemplos anteriores mostram Python38. Substitua conforme necessário com o número da versão que você está usando.

Para modificar sua variável PATH (Windows)

1. Pressione a tecla Windows e digite **`environment variables`**.
2. Escolha Edit environment variables for your account (Editar variáveis de ambiente para sua conta).
3. Selecione PATH e, em seguida, Edit (Editar).
4. Adicione o caminho ao campo Variable value (Valor da variável). Por exemplo: ***`C:\new\path`***
5. Escolha OK duas vezes para aplicar as novas configurações.
6. Feche todos os prompts de comando em execução e abra novamente a janela do prompt de comando.

Configurando AWS ParallelCluster

Depois de instalar AWS ParallelCluster, conclua as etapas de configuração a seguir.

Verifique se sua AWS conta tem uma função que inclui as permissões necessárias para executar a `pcluster` CLI. Para obter mais informações, consulte AWS ParallelCluster exemplo de políticas de instância e usuário.

Configure suas AWS credenciais. Para obter mais informações, consulte Configurar a AWS CLI no Guia do usuário da AWS CLI .

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

O Região da AWS local em que o cluster é lançado deve ter pelo menos um par de EC2 chaves da Amazon. Para obter mais informações, consulte os pares de EC2 chaves da Amazon no Guia EC2 do usuário da Amazon.

```
$ pcluster configure
```

O assistente de configuração solicita todas as informações necessárias para criar o cluster. Os detalhes da sequência diferem quando usada AWS Batch como agendador em comparação com o uso Slurm. Para obter mais informações sobre a configuração de um cluster, consulteConfiguração.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE or Torque agendadores. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.

Slurm

Na lista de Região da AWS identificadores válidos, escolha Região da AWS onde você deseja que seu cluster seja executado.

> **Note**
>
> A lista Regiões da AWS exibida é baseada na partição da sua conta e inclui apenas as Regiões da AWS que estão habilitadas para sua conta. Para obter mais informações sobre como habilitar Regiões da AWS sua conta, consulte Gerenciando Regiões da AWS no Referência geral da AWS. O exemplo mostrado é da partição AWS global. Se sua conta estiver na AWS GovCloud (US) partição, somente Regiões da AWS nessa partição serão listadas (`gov-us-east-1egov-us-west-1`). Da mesma forma, se sua conta estiver na partição AWS da China, somente `cn-north-1` e `cn-northwest-1` serão mostradas. Para obter a lista completa dos Regiões da AWS produtos suportados por AWS ParallelCluster, consulteRegiões do compatíveis.

```
Allowed values for the Região da 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
Região da AWS ID [ap-northeast-1]:
```

Escolha o programador a ser usado com seu cluster.

```
Allowed values for Scheduler:
1. slurm
```

```
2. awsbatch
Scheduler [slurm]:
```

Escolha o sistema operacional.

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

> **Note**
>
> Support for `alinux2` adicionado na AWS ParallelCluster versão 2.6.0.

Os tamanhos mínimo e máximo do cluster de nós de computação são inseridos. Isso é medido em número de instâncias.

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

Os tipos de instância de nós principais e de computação são inseridos. Para tipos de instância, seus limites de instância de conta são grandes o suficiente para atender às suas necessidades. Para obter mais informações, consulte Limites de instâncias sob demanda no Guia do EC2 usuário da Amazon.

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

O par de chaves é selecionado entre os pares de chaves registrados EC2 na Amazon no selecionado Região da AWS.

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

Depois que as etapas anteriores forem concluídas, decida se quer usar uma VPC existente ou AWS ParallelCluster criar uma VPC para você. Se você não tiver uma VPC configurada corretamente, AWS ParallelCluster poderá criar uma nova. Ela usa os nós principais e de computação na mesma sub-rede pública ou somente o nó principal em uma sub-rede pública com todos os nós em uma sub-rede privada. É possível atingir seu limite de número de VPCs em um Região da AWS. O limite padrão é cinco VPCs para cada um Região da AWS. Para obter mais informações sobre esse limite e como solicitar um aumento, consulte VPC e sub-redes no Guia de usuário do Amazon VPC.

Se você deixar AWS ParallelCluster criar uma VPC, deverá decidir se todos os nós devem estar em uma sub-rede pública.

> **Important**
>
> VPCs criado por AWS ParallelCluster não habilitar registros de fluxo de VPC por padrão. Os registros de fluxo de VPC permitem que você capture informações sobre o tráfego IP que entra e sai das interfaces de rede em sua. VPCs Para obter mais informações, consulte Logs de fluxo da VPC no Guia do usuário do Amazon Virtual Private Cloud.

> **Note**
>
> Se você escolher `1. Master in a public subnet and compute fleet in a private subnet`, o AWS ParallelCluster cria um gateway NAT que resulta em custo adicional, mesmo se você especificar recursos de nível gratuito.

```
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
```

Se você não criar uma nova VPC, você deverá selecionar uma VPC existente.

Se você optar por AWS ParallelCluster criar a VPC, anote a ID da VPC para poder usá-la para excluí-la posteriormente AWS CLI .

```
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
```

Depois de selecionar a VPC, você deve decidir se deseja usar sub-redes existentes ou criar novas.

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

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

AWS Batch

Na lista de Região da AWS identificadores válidos, escolha Região da AWS onde você deseja que seu cluster seja executado.

```
Allowed values for Região da 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
```

```
Região da AWS ID [ap-northeast-1]:
```

Escolha o programador a ser usado com seu cluster.

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

Quando `awsbatch` é selecionado como o programador, `alinux2` é usado como o sistema operacional.

Os tamanhos mínimo e máximo do cluster de nós de computação são inseridos. Isso é medido em CPUs v.

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

O tipo de instância do nó principal é inserido. Ao usar o programador `awsbatch`, os nós de computação usam um tipo de instância de `optimal`.

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

O par de EC2 chaves da Amazon é selecionado entre os pares de chaves registrados EC2 na Amazon no selecionado Região da AWS.

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

Decida se quer usar o existente VPCs ou deixar AWS ParallelCluster criar VPCs para você. Se você não tiver uma VPC configurada corretamente, AWS ParallelCluster poderá criar uma nova. Ela usa os nós principais e de computação na mesma sub-rede pública ou somente o nó principal em uma sub-rede pública com todos os nós em uma sub-rede privada. É possível atingir seu limite de número de VPCs em um Região da AWS. O número padrão de VPCs é cinco. Para obter mais informações sobre esse limite e como solicitar um aumento, consulte VPC e sub-redes no Guia de usuário do Amazon VPC.

> **Important**
>
> VPCs criado por AWS ParallelCluster não habilitar registros de fluxo de VPC por padrão. Os registros de fluxo de VPC permitem que você capture informações sobre o tráfego IP que entra e sai das interfaces de rede em sua. VPCs Para obter mais informações, consulte Logs de fluxo da VPC no Guia do usuário do Amazon Virtual Private Cloud.

Se você deixar AWS ParallelCluster criar uma VPC, decida se todos os nós devem estar em uma sub-rede pública.

> **Note**
>
> Se você escolher `1. Master in a public subnet and compute fleet in a private subnet`, o AWS ParallelCluster cria um gateway NAT que resulta em custo adicional, mesmo se você especificar recursos de nível gratuito.

```
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
```

Se você não criar uma nova VPC, você deverá selecionar uma VPC existente.

Se você optar por AWS ParallelCluster criar a VPC, anote a ID da VPC para poder usá-la para excluí-la posteriormente AWS CLI .

```
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
```

Depois de selecionar a VPC, decida se deseja usar sub-redes existentes ou criar novas.

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

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

Depois de concluir as etapas anteriores, um cluster simples é iniciado em uma VPC. A VPC usa uma sub-rede existente que oferece suporte a endereços IP públicos. A tabela de rotas para a sub-rede é `0.0.0.0/0 => igw-xxxxxx`. Atenção às condições a seguir:

- A VPC deve ter `DNS Resolution = yes` e `DNS Hostnames = yes`.
- A VPC também deve ter opções DHCP com o `domain-name` correto para a Região da AWS. O conjunto de opções DHCP padrão já especifica o necessário AmazonProvidedDNS. Se especificar mais de um servidor de nomes de domínio, consulte os conjuntos de opções de DHCP no Guia do usuário da Amazon VPC. Ao usar sub-redes privadas, use um gateway NAT ou um proxy interno para permitir o acesso à web para nós de computação. Para obter mais informações, consulte Configurações de rede.

Quando todas as configurações tiverem valores válidos, você poderá inicializar o cluster executando o comando de criação.

```
$ pcluster create mycluster
```

Quando o cluster atingir o status "CREATE_COMPLETE", você poderá se conectar a ele usando as configurações normais de cliente SSH. Para obter mais informações sobre a conexão com EC2 instâncias da Amazon, consulte o Guia EC2 do usuário no Guia EC2 do usuário da Amazon.

Para excluir o cluster, execute o comando a seguir.

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

Para excluir os recursos de rede na VPC, você pode excluir a pilha CloudFormation de rede. O nome da pilha começa com”parallelclusternetworking-“e contém a hora da criação no formato “YYYYMMDDHHMMSS”. Você pode listar as pilhas usando o comando list-stacks.

```
$ aws --region us-east-1 cloudformation list-stacks \
```

```
   --stack-status-filter "CREATE_COMPLETE" \
   --query "StackSummaries[].StackName" | \
   grep -e "parallelclusternetworking-"
    "parallelclusternetworking-pubpriv-20191029205804"
```

A pilha pode ser excluída usando o comando delete-stack.

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

A VPC que `pcluster configure` cria para você não é criada na pilha de CloudFormation rede. Você pode excluir essa VPC manualmente no console ou usando a AWS CLI.

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

Práticas recomendadas

Práticas recomendadas: seleção do tipo de instância do nó principal

Mesmo que o nó principal não execute uma tarefa, suas funções e seu tamanho são cruciais para o desempenho geral do cluster.

Ao escolher o tipo de instância a ser usado para seu nó principal, você deve avaliar os seguintes itens:

- Tamanho do cluster: o nó principal orquestra a lógica de escalabilidade do cluster e é responsável por anexar novos nós ao programador. Se você precisar aumentar e reduzir verticalmente a escala do cluster de uma quantidade considerável de nós, deverá dar ao nó principal alguma capacidade computacional extra.
- Sistemas de arquivos compartilhados: ao usar sistemas de arquivos compartilhados para compartilhar artefatos entre os nós de computação e o nó principal, leve em consideração que o nó principal é o nó que expõe o servidor NFS. Por esse motivo, é preferível escolher um tipo de instância com largura de banda da rede suficiente e largura de banda dedicada suficiente do Amazon EBS para lidar com seus fluxos de trabalho.

Práticas recomendadas: desempenho da rede

Há três dicas que abrangem toda a gama de possibilidades para melhorar a comunicação de rede.

- Grupo de posicionamento: um grupo de posicionamento de cluster é um agrupamento lógico de instâncias dentro de uma única zona de disponibilidade. Para obter mais informações sobre grupos de posicionamento, consulte grupos de posicionamento no Guia EC2 do usuário da Amazon. Você pode configurar o cluster para usar seu próprio grupo de posicionamento com `placement_group = your-placement-group-name` ou deixar AWS ParallelCluster criar um grupo de posicionamento com a estratégia `"compute"` com `placement_group = DYNAMIC`. Para obter mais informações, consulte `placement_group` para o modo de fila múltipla e `placement_group` para o modo de fila única.
- Rede aprimorada: considere escolher um tipo de instância que ofereça suporte à rede avançada. Para obter mais informações, consulte redes aprimoradas no Linux no Guia EC2 do usuário da Amazon.
- Elastic Fabric Adapter: para oferecer suporte a altos níveis de comunicação escalável entre instâncias, considere escolher interfaces de rede EFA para sua rede. O hardware de desvio do sistema operacional (SO) personalizado da EFA aprimora as comunicações entre instâncias com a elasticidade e a flexibilidade sob demanda da nuvem. AWS Para configurar um único Slurm fila de cluster para usar o EFA, definida. `enable_efa = true` Para obter mais informações sobre como usar o EFA com AWS ParallelCluster, consulte Elastic Fabric Adapter e. `enable_efa` Para obter mais informações sobre o EFA, consulte o Elastic Fabric Adapter no Guia EC2 do usuário da Amazon para instâncias Linux.
- Largura de banda da instância: a largura de banda aumenta de acordo com o tamanho da instância. Considere escolher o tipo de instância que melhor atenda às suas necessidades. Consulte Instâncias otimizadas para Amazon EBS e tipos de volume do Amazon EBS no Guia do Usuário da Amazon. EC2

Práticas recomendadas: alertas de orçamento

Para gerenciar os custos dos AWS ParallelCluster recursos, recomendamos que você use AWS Budgets ações para criar um orçamento e definir alertas de limite de orçamento para AWS os recursos selecionados. Para obter mais informações, consulte Como configurar uma ação de orçamento no Manual do usuário do AWS Budgets . Você também pode usar CloudWatch a Amazon para criar um alarme de cobrança. Para obter mais informações, consulte Criação de um alarme de cobrança para monitorar suas AWS cobranças estimadas.

Melhores práticas: mover um cluster para uma nova versão AWS ParallelCluster secundária ou de patch

Atualmente, cada versão AWS ParallelCluster secundária é independente junto com sua `pcluster` CLI. Para mover um cluster para uma nova versão secundária ou de patch, você deve recriar o cluster usando a CLI da nova versão.

Para otimizar o processo de mover um cluster para uma nova versão secundária ou para salvar seus dados de armazenamento compartilhado por outros motivos, recomendamos que você use as práticas recomendadas a seguir.

- Salve dados pessoais em volumes externos, como Amazon EFS e FSx for Lustre. Ao fazer isso, você pode mover facilmente os dados de um cluster para outro.
- Crie sistemas de armazenamento compartilhado dos tipos listados abaixo usando o AWS CLI ou Console de gerenciamento da AWS:
  - Seção `[ebs]`
  - Seção `[efs]`
  - Seção `[fsx]`

  Adicione-os à nova configuração do cluster como sistemas de arquivos existentes. Dessa forma, eles serão preservados quando você excluir o cluster e podem ser anexados a um novo cluster. Os sistemas de armazenamento compartilhado geralmente incorrem em cobranças, estejam eles conectados ou desconectados de um cluster.

  Recomendamos que você use os sistemas de arquivos Amazon EFS ou Amazon FSx for Lustre porque eles podem ser anexados a vários clusters ao mesmo tempo e você pode anexá-los ao novo cluster antes de excluir o cluster antigo. Para obter mais informações, consulte Montagem de sistemas de arquivos do Amazon EFS no Guia do usuário do Amazon EFS e Acesso aos sistemas FSx de arquivos Lustre no Guia do usuário do Amazon FSx for Lustre Lustre.
- Use ações de bootstrap personalizadas para personalizar suas instâncias em vez de usar uma AMI personalizada. Isso otimiza o processo de criação porque uma nova AMI personalizada não precisa ser criada para cada nova versão.
- Sequência recomendada.
  1. Atualize a configuração do cluster para usar as definições existentes do sistema de arquivos.
  2. Verifique a versão do `pcluster` e atualize-a, se necessário.
  3. Crie e teste o novo cluster.

- Verifique se os dados estão disponíveis no cluster novo.
- Verifique se a aplicação funciona no cluster novo.

4. Se seu novo cluster estiver totalmente testado e operacional e você tiver certeza de que não usará o cluster antigo, exclua-o.

Passando de CfnCluster para AWS ParallelCluster

AWS ParallelCluster é uma versão aprimorada do CfnCluster.

Se você usa atualmente CfnCluster, recomendamos que use AWS ParallelCluster em vez disso e crie novos clusters com ele. Mesmo que você possa continuar usando CfnCluster, ele não está mais sendo desenvolvido e nenhum novo recurso ou funcionalidade será adicionado.

As principais diferenças entre CfnCluster e AWS ParallelCluster estão descritas nas seções a seguir.

AWS ParallelCluster A CLI gerencia um conjunto diferente de clusters

Os clusters criados pela CLI do `cfncluster` não podem ser gerenciados com a CLI do `pcluster`. Os comandos a seguir não funcionam em clusters criados por CfnCluster:

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

Para gerenciar clusters que você criou com CfnCluster, você deve usar a `cfncluster` CLI.

Se você precisar de um CfnCluster pacote para gerenciar seus clusters antigos, recomendamos que você o instale e use em um ambiente virtual Python.

AWS ParallelCluster e CfnCluster use diferentes políticas personalizadas do IAM

As políticas personalizadas do IAM que foram usadas anteriormente para a criação de CfnCluster clusters não podem ser usadas com AWS ParallelCluster. Se você precisar de políticas personalizadas para AWS ParallelCluster, deverá criar novas. Consulte o guia do AWS ParallelCluster .

AWS ParallelCluster e CfnCluster use arquivos de configuração diferentes

O arquivo de AWS ParallelCluster configuração reside na `~/.parallelcluster` pasta. O arquivo de configuração do CfnCluster reside na pasta `~/.cfncluster`.

Se você quiser usar um arquivo CfnCluster de configuração existente com AWS ParallelCluster, deverá concluir as seguintes ações:

1. Mover o arquivo de configuração de `~/.cfncluster/config` para `~/.parallelcluster/config`.
2. Se você usar o parâmetro de configuração `extra_json`, altere-o conforme mostrado.

CfnCluster configuração:

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

AWS ParallelCluster configuração:

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

Em AWS ParallelCluster, os gânglios estão desativados por padrão

Em AWS ParallelCluster, os gânglios são desativados por padrão. Para ativar os gânglios, conclua as seguintes etapas:

1. Defina o parâmetro `extra_json` conforme mostrado:

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

2. Altere o grupo de segurança principal para permitir conexões com a porta 80.

   O grupo de segurança `parallelcluster-<CLUSTER_NAME>-MasterSecurityGroup-<xxx>` deve ser modificado por meio da adição de uma nova regra do grupo de segurança para permitir a conexão de entrada para a porta 80 de seu IP público. Para obter mais informações, consulte Adicionar regras a um grupo de segurança no Guia EC2 do usuário da Amazon.

Regiões do compatíveis

AWS ParallelCluster a versão 2.x está disponível da seguinte Regiões da AWS forma:

Nome da região	Região
Leste dos EUA (Ohio)	us-east-2
Leste dos EUA (Norte da Virgínia)	us-east-1
Oeste dos EUA (Norte da Califórnia)	us-west-1
Oeste dos EUA (Oregon)	us-west-2
África (Cidade do Cabo)	af-south-1
Ásia-Pacífico (Hong Kong)	ap-east-1
Ásia-Pacífico (Mumbai)	ap-south-1
Ásia-Pacífico (Seul)	ap-northeast-2
Ásia-Pacífico (Singapura)	ap-southeast-1
Ásia-Pacífico (Sydney)	ap-southeast-2
Ásia-Pacífico (Tóquio)	ap-northeast-1
Canadá (Central)	ca-central-1
China (Pequim)	cn-north-1
China (Ningxia)	cn-northwest-1
Europa (Frankfurt)	eu-central-1
Europa (Irlanda)	eu-west-1
Europa (Londres)	eu-west-2
Europa (Milão)	eu-south-1
Europa (Paris)	eu-west-3
Europa (Estocolmo)	eu-north-1

Nome da região	Região
Oriente Médio (Barém)	me-south-1
América do Sul (São Paulo)	sa-east-1
AWS GovCloud (Leste dos EUA)	us-gov-east-1
AWS GovCloud (Oeste dos EUA)	us-gov-west-1

Usando AWS ParallelCluster

Tópicos

Configurações de rede

AWS ParallelCluster usa a Amazon Virtual Private Cloud (VPC) para redes. A VPC fornece uma plataforma de rede flexível e configurável onde você pode implantar clusters.

A VPC deve ter `DNS Resolution = yes` opções de DHCP com o nome de domínio correto para a região. `DNS Hostnames = yes` O conjunto de opções DHCP padrão já especifica o DNS necessárioAmazonProvided. Se estiver especificando mais de um servidor de nomes de domínio, consulte Conjuntos de opções DHCP no Amazon VPC User Guide.

AWS ParallelCluster suporta as seguintes configurações de alto nível:

- Uma sub-rede para nós principais e de computação.
- Duas sub-redes, com o nó principal em uma sub-rede pública, e nós de computação em uma sub-rede privada. As sub-redes podem ser novas ou existentes.

Todas essas configurações podem operar com ou sem endereçamento IP público. AWS ParallelCluster também pode ser implantado para usar um proxy HTTP para todas as AWS solicitações. As combinações dessas configurações resultam em muitos cenários de implantação. Por exemplo, você pode configurar uma única sub-rede pública com todo o acesso pela Internet. Ou você pode configurar uma rede totalmente privada usando AWS Direct Connect um proxy HTTP para todo o tráfego.

Consulte os seguintes diagramas de arquitetura para obter ilustrações de alguns desses cenários:

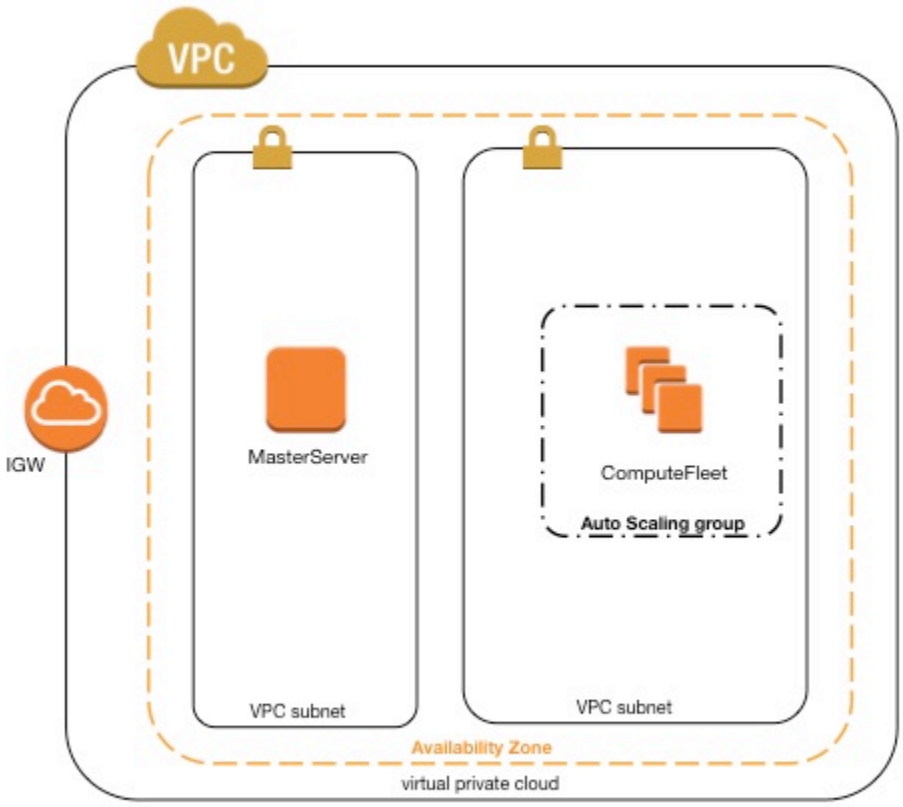
AWS ParallelCluster em uma única sub-rede pública

A configuração para essa arquitetura requer as seguintes definições:

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

A configuração `use_public_ips` não pode ser definida como `false`, porque o gateway de Internet requer que todas as instâncias tenham um endereço IP exclusivo globalmente. Para obter mais informações, consulte Habilitar o acesso à Internet no Manual do usuário da Amazon VPC.

AWS ParallelCluster usando duas sub-redes



A configuração para criar uma nova sub-rede privada para instâncias de computação requer as seguintes definições:

Observe que todos os valores são somente exemplos.

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

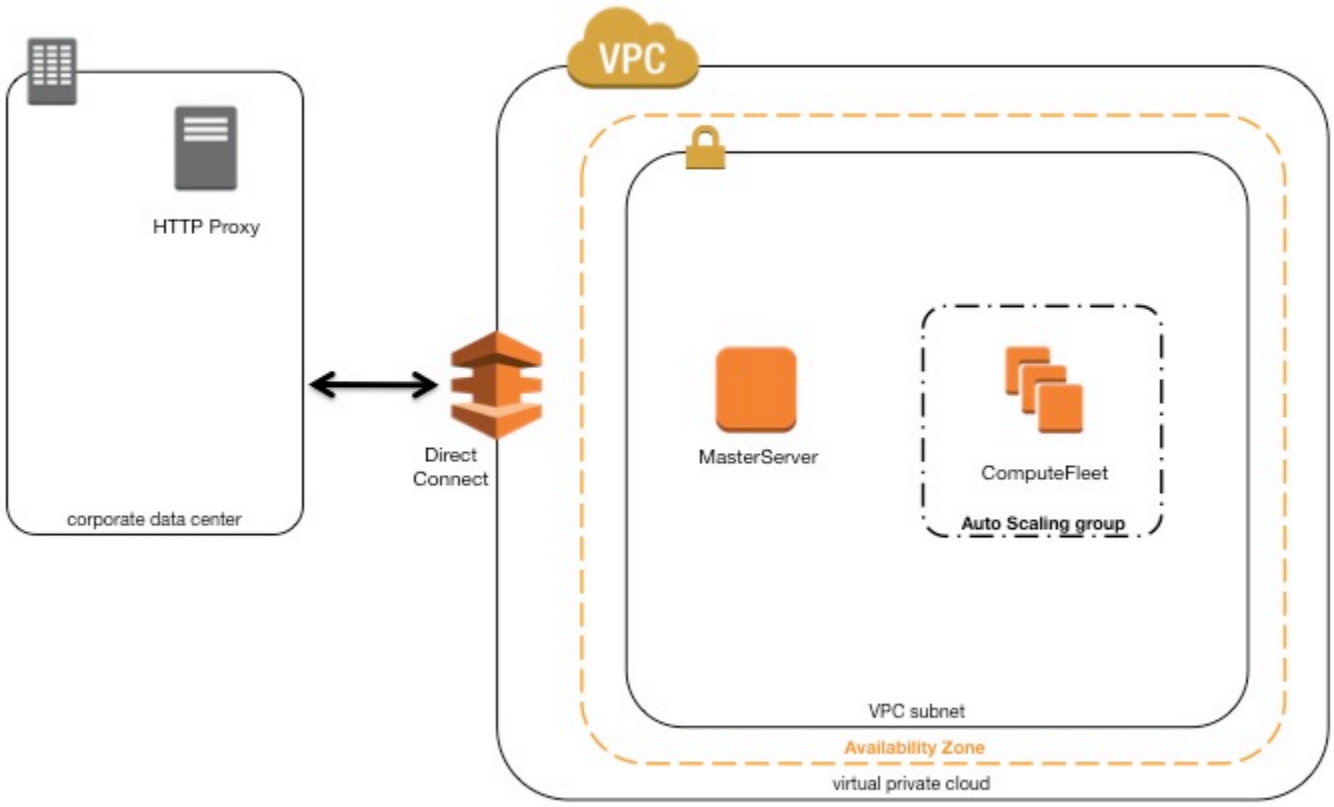
A configuração para usar uma rede privada existente requer as seguintes definições:

```
[vpc public-private-existing]
```

```
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-<public>
compute_subnet_id = subnet-<private>
```

Ambas as configurações exigem um gateway NAT ou um PROXY interno para habilitar o acesso à web para instâncias de computação.

AWS ParallelCluster em uma única sub-rede privada conectada usando AWS Direct Connect



A configuração para essa arquitetura requer as seguintes definições:

```
[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
```

Quando `use_public_ips` for definido como `false` a VPC deverá ser corretamente configurada para usar o Proxy para todo o tráfego. O acesso à Web é necessário tanto para os nós principais quanto para os nós de computação.

AWS ParallelCluster com **`awsbatch`** agendador

Quando você usa `awsbatch` como tipo de agendador, AWS ParallelCluster cria um ambiente de computação AWS Batch gerenciado. O AWS Batch ambiente cuida do gerenciamento das instâncias de contêiner do Amazon Elastic Container Service (Amazon ECS), que são lançadas no. `compute_subnet` AWS Batch Para funcionar corretamente, as instâncias de contêiner do Amazon ECS precisam de acesso externo à rede para se comunicarem com o endpoint do serviço Amazon ECS. Isso se converte nas seguintes situações:

- O `compute_subnet` usa um gateway NAT para acessar a Internet. (Recomendamos essa abordagem.)
- As instâncias executadas na `compute_subnet` têm endereços IP públicos e podem acessar a Internet por meio de um gateway da Internet.

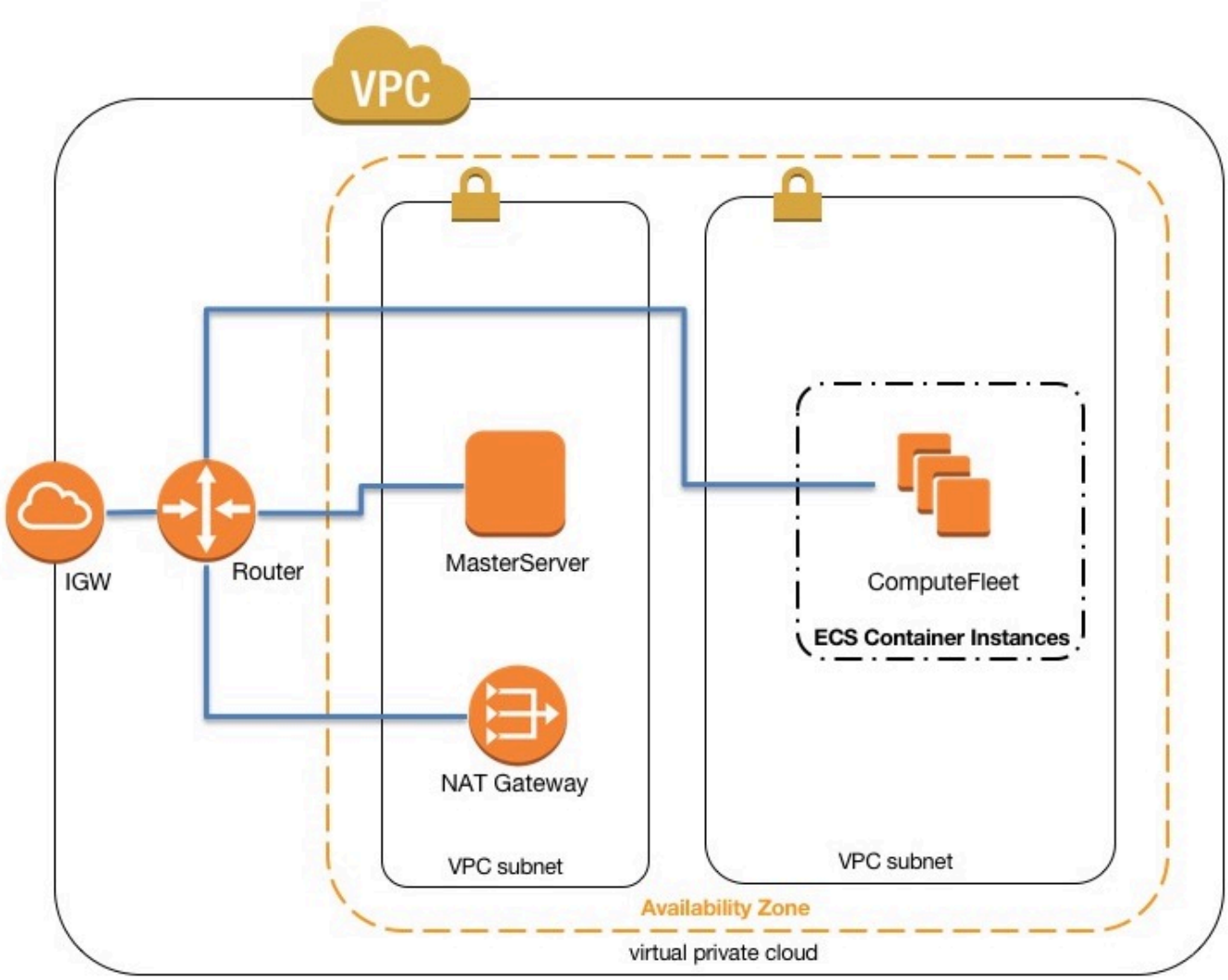
Além disso, se você estiver interessado em trabalhos em paralelo de vários nós (dos documentos do AWS Batch):

AWS Batch trabalhos paralelos de vários nós usam o modo de `awsvpc` rede do Amazon ECS, que fornece aos contêineres de trabalhos paralelos de vários nós as mesmas propriedades de rede das instâncias do Amazon EC2. Cada contêiner de trabalho paralelo de vários nós obtém sua própria interface de rede elástica, um endereço IP privado primário e um nome de host DNS interno. A interface de rede é criada na mesma sub-rede Amazon VPC que seu recurso de computação do host. Todos os grupos de segurança aplicados aos seus recursos de computação também são aplicados a ele.

Ao usar redes de trabalho da Rede de tarefas do Amazon ECS, o modo de rede `awsvpc` não fornece interfaces de rede elásticas com endereços IP públicos para as tarefas que usam o tipo de inicialização do Amazon EC2. Para acessar a Internet, as tarefas que usam o tipo de execução do

Amazon EC2 devem ser executadas em uma sub-rede privada que esteja configurada para usar um gateway NAT.

Você deve configurar um gateway NAT para permitir que o cluster execute trabalhos paralelos de vários nós.



Para saber mais, consulte os seguintes tópicos:

- AWS Batch ambientes computacionais gerenciados
- AWS Batch trabalhos paralelos de vários nós
- Redes de tarefas do Amazon ECS com o modo de rede `awsvpc`

Ações de bootstrap personalizadas

AWS ParallelCluster pode executar código arbitrário antes (pré-instalação) ou depois (pós-instalação) da ação principal de bootstrap quando o cluster é criado. Na maioria dos casos, esse código é armazenado no Amazon Simple Storage Service (Amazon S3) e acessado por meio de uma conexão HTTPS. O código é executado como root e pode estar em qualquer linguagem de script compatível com o sistema operacional do cluster. Muitas vezes, o código está em Bash ou Python.

As ações de pré-instalação são chamadas antes de qualquer bootstrap de implantação do cluster, como a configuração de NAT, do Amazon Elastic Block Store (Amazon EBS) ou o programador. Entre as ações de pré-instalação típicas estão a modificação do armazenamento, a adição de usuários e a inclusão de pacotes adicionais.

As ações pós-instalação são chamadas após a conclusão dos processos de inicialização do cluster. As ações pós-instalação são as últimas ações que ocorrem antes que uma instância seja considerada totalmente configurada e concluída. Algumas ações de pós-instalação podem incluir a alteração de configurações do programador, a modificação do armazenamento ou a modificação de pacotes.

Os argumentos podem ser transmitidos para scripts, especificando-os na configuração. Para tanto, você deve enviá-los entre aspas duplas para as ações de pré-instalação e pós-instalação.

Se uma ação de pré-instalação ou pós-instalação falhar, o bootstrap da instância também falha. O êxito é sinalizado com um código de saída de zero (0). Qualquer outro código de saída indica que o bootstrap da instância falhou.

Você pode diferenciar entre cabeçote de execução e nós de computação. Gere o arquivo `/etc/parallelcluster/cfnconfig` e avalie a variável de ambiente `cfn_node_type`, que possui valores "`MasterServer`" e "`ComputeFleet`" para os nós principais e de computação, respectivamente.

```
#!/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
```

Configuração

As seguintes definições de configuração são usadas para definir ações e argumentos de pré-instalação e pós-instalação.

```
# 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
```

Argumentos

Os dois primeiros argumentos — `$0` e `$1` — são reservados para o url e o nome do script.

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

Exemplo

As etapas a seguir criam um script simples de pós-instalação que instala os pacotes R em um cluster.

1. Crie um script.

```
#!/bin/bash

echo "post-install script has $# arguments"
```

```
for arg in "$@"
do
    echo "arg: ${arg}"
done

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

2. Faça upload do script com as permissões corretas para o Amazon S3. Se as permissões de leitura pública não forem apropriadas para você, use um `s3_read_resource` ou `s3_read_write_resource` os parâmetros para conceder acesso. Para obter mais informações, consulte Trabalhar com o Amazon S3.

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

> **Important**
>
> Se o script foi editado no Windows, as terminações de linha devem ser alteradas de CRLF para LF antes que seja feito upload do script para o Amazon S3.

3. Atualize a AWS ParallelCluster configuração para incluir a nova ação pós-instalação.

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

Se o bucket não tiver permissão de leitura pública, use `s3` como o protocolo de URL.

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

4. Execute os clusters.

```
$ pcluster create mycluster
```

5. Verifique a saída.

```
$ 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
```

Trabalhar com o Amazon S3

Para fornecer permissão aos recursos do cluster para acessar os buckets do Amazon S3, especifique o bucket ARNs na `s3_read_resource` e os `s3_read_write_resource` parâmetros na configuração. AWS ParallelCluster Para obter mais informações sobre como controlar o acesso com AWS ParallelCluster, consulteAWS Identity and Access Management funções em 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/*
```

Ambos os parâmetros aceitam * ou um ARN válido do Amazon S3. Para obter informações sobre a especificação do Amazon ARNs S3, consulte o formato ARN do Amazon S3 no. Referência geral da AWS

Exemplos

O exemplo a seguir concede acesso de leitura a qualquer objeto no bucket do Amazon S3 my_corporate_bucket.

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

O exemplo a seguir concede acesso de leitura ao bucket, mas não permite que você leia itens do bucket.

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

Este último exemplo fornece acesso de leitura ao bucket e aos itens armazenados no bucket.

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

Trabalho com Instâncias spot

AWS ParallelCluster usa instâncias spot se a configuração do cluster tiver definido `cluster_type` = spot. As instâncias spot são mais econômicas do que as instâncias sob demanda, mas podem ser interrompidas. O efeito da interrupção varia dependendo do programador específico usado. Pode ser útil aproveitar os avisos de interrupção da Instância Spot, que fornecem um aviso de dois minutos antes que o Amazon EC2 deva parar ou encerrar sua Instância Spot. Para ter mais informações, consulte Interrupções de instâncias spot no Guia do usuário do Amazon EC2. As seções a seguir descrevem três cenários nos quais as instâncias spot podem ser interrompidas.

> **Note**
>
> O uso de Instâncias spot exige que a função `AWSServiceRoleForEC2Spot` vinculada ao serviço exista na sua conta. Para criar essa função na sua conta usando o AWS CLI, execute o seguinte comando:
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> Para ter mais informações, consulte Função vinculada ao serviço para solicitações de instâncias spot no Guia do usuário do Amazon EC2.

Cenário 1: Uma instância spot sem trabalhos em execução é interrompida

Quando essa interrupção ocorre, AWS ParallelCluster tenta substituir a instância se a fila do agendador tiver trabalhos pendentes que exijam instâncias adicionais ou se o número de instâncias ativas for menor que a configuração. `initial_queue_size` Se o AWS ParallelCluster não

conseguir provisionar novas instâncias, uma solicitação de novas instâncias será repetida periodicamente.

Cenário 2: Uma instância spot que executa trabalhos de nó único é interrompida

O comportamento dessa interrupção depende do programador que está sendo usado.

Slurm

O trabalho falha com um código de estado de `NODE_FAIL`, e o trabalho é colocado novamente na fila (a menos que `--no-requeue` seja especificado quando o trabalho é enviado). Se o nó for estático, ele será substituído. Se o nó for um nó dinâmico, o nó será encerrado e redefinido. Para obter mais informações sobre o `sbatch`, incluindo o parâmetro `--no-requeue`, consulte sbatch na documentação do Slurm.

> **Note**
>
> Esse comportamento mudou na AWS ParallelCluster versão 2.9.0. As versões anteriores encerraram o trabalho com um código de estado de `NODE_FAIL` e o nó foi removido da fila do programador.

SGE

> **Note**
>
> Isso se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

O trabalho está encerrado. Se o trabalho tiver habilitado o sinalizador de nova execução (usando um `qsub -r yes` ou `qalter -r yes`) ou se a fila tiver a configuração `rerun` definida como `TRUE`, o trabalho será reprogramado. A instância de computação é removida da fila do programador. Esse comportamento vem destes parâmetros de configuração SGE:

- `reschedule_unknown 00:00:30`

- `ENABLE_FORCED_QDEL_IF_UNKNOWN`
- `ENABLE_RESCHEDULE_KILL=1`

Torque

> **Note**
>
> Isso se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

O trabalho é removido do sistema e o nó é removido do programador. O trabalho não é executado novamente. Se vários trabalhos estiverem em execução na instância quando ela for interrompida, o tempo limite do Torque poderá se esgotar durante a remoção do nó. Um erro pode ser exibido no arquivo de log `sqswatcher`. Isso não afeta a lógica de escalabilidade e uma limpeza adequada é realizada por novas tentativas subsequentes.

Cenário 3: Uma instância spot que executa trabalhos de vários nós é interrompida

O comportamento dessa interrupção depende do programador que está sendo usado.

Slurm

O trabalho falha com um código de estado de `NODE_FAIL`, e o trabalho é colocado novamente na fila (a menos que `--no-requeue` seja especificado quando o trabalho é enviado). Se o nó for estático, ele será substituído. Se o nó for um nó dinâmico, o nó será encerrado e redefinido. Outros nós que estavam executando os trabalhos encerrados podem ser reduzidos após o término do tempo `scaledown_idletime` configurado.

> **Note**
>
> Esse comportamento mudou na AWS ParallelCluster versão 2.9.0. As versões anteriores encerraram o trabalho com um código de estado de `NODE_FAIL` e o nó foi removido da fila do programador. Outros nós que estavam executando os trabalhos encerrados podem ser reduzidos após o término do tempo `scaledown_idletime` configurado.

SGE

> **Note**
>
> Isso se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

O trabalho não é encerrado e continua a ser executado nos nós restantes. O nó de computação é removido da fila do programador, mas aparecerá na lista de hosts como um nó órfão e indisponível.

O usuário deve excluir o trabalho quando isso ocorre (`qdel <jobid>`). O nó ainda é exibido na lista de hosts (`qhost`), embora isso não afete o AWS ParallelCluster. Para remover o host da lista, execute o seguinte comando depois de substituir a instância.

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

Torque

> **Note**
>
> Isso se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

O trabalho é removido do sistema e o nó é removido do programador. O trabalho não é executado novamente. Se vários trabalhos estiverem em execução na instância quando ela for interrompida, o tempo limite do Torque poderá se esgotar durante a remoção do nó. Um erro pode ser exibido no arquivo de log `sqswatcher`. Isso não afeta a lógica de escalabilidade e uma limpeza adequada é realizada por novas tentativas subsequentes.

Para ter mais informações sobre instâncias spot, consulte Instâncias spot no Guia do usuário do Amazon EC2.

AWS Identity and Access Management funções em AWS ParallelCluster

AWS ParallelCluster usa funções AWS Identity and Access Management (IAM) para o Amazon EC2 para permitir que instâncias acessem AWS serviços para a implantação e operação de um cluster. Por padrão, o perfil do IAM para o Amazon EC2 é criado quando o cluster é criado. Isso significa que o usuário que cria o cluster deve ter o nível adequado de permissões, conforme descrito nas seções a seguir.

AWS ParallelCluster usa vários AWS serviços para implantar e operar um cluster. Consulte a lista completa na seção Serviços da AWS usados no AWS ParallelCluster.

Você pode acompanhar as alterações nas políticas de exemplo na AWS ParallelCluster documentação em GitHub.

Tópicos

Configurações padrão para criação de clusters

Ao usar as configurações padrão para a criação do cluster, um perfil do IAM padrão para o Amazon EC2 é criado pelo cluster. O usuário que cria o cluster deve ter o nível adequado de permissões para criar todos os recursos necessários para executar o cluster. Isso inclui criar um perfil do IAM para o Amazon EC2. Normalmente, o usuário deve ter as permissões de uma política AdministratorAccessgerenciada ao usar as configurações padrão. Para obter mais informações sobre políticas gerenciadas, consulte Políticas gerenciadas pela AWS no Guia do usuário do IAM.

Usar um perfil do IAM existente para o Amazon EC2

No lugar das configurações padrão, você pode usar uma `ec2_iam_role` existente ao criar um cluster, mas primeiro é necessário definir a política do IAM e o perfil antes de tentar executar o cluster. Normalmente, você escolhe um perfil do IAM existente para o Amazon EC2 a fim de minimizar as permissões que são concedidas aos usuários à medida que executam clusters. Eles AWS ParallelCluster exemplo de políticas de instância e usuário incluem as permissões

mínimas exigidas AWS ParallelCluster e seus recursos. É necessário criar tanto as políticas como os perfis como políticas individuais no IAM e depois anexar os perfis e as políticas aos recursos apropriados. Algumas das políticas de função podem ficar grandes e causar erros de cota. Para obter mais informações, consulte Solução de problemas de tamanho da política do IAM. Nas políticas, substitua`<REGION>`,`<AWS ACCOUNT ID>`, e cadeias similares pelos valores apropriados.

Se sua intenção é adicionar políticas extras às configurações padrão dos nós do cluster, recomendamos que você transmita as políticas adicionais personalizadas do IAM com a configuração `additional_iam_policies` em vez de usar as configurações `ec2_iam_role`.

AWS ParallelCluster exemplo de políticas de instância e usuário

Os exemplos de políticas a seguir incluem Amazon Resource Names (ARNs) para os recursos. Se você estiver trabalhando nas partições AWS GovCloud (US) ou AWS na China, elas ARNs devem ser alteradas. Especificamente, eles devem ser alterados de “arn:aws” para “arn:aws-us-gov" para a AWS GovCloud (US) partição ou “arn:aws-cn” para a partição da China. AWS Para obter mais informações, consulte Amazon Resource Names (ARNs) em AWS GovCloud (US) Regiões no Guia AWS GovCloud (US) do usuário e ARNs para AWS serviços na China em Introdução aos AWS serviços na China.

Essas políticas incluem as permissões mínimas atualmente exigidas por AWS ParallelCluster, seus recursos e recursos. Algumas das políticas de função podem ficar grandes e causar erros de cota. Para obter mais informações, consulte Solução de problemas de tamanho da política do IAM.

Tópicos

- ParallelClusterInstancePolicy usando SGE, Slurm ou Torque
- ParallelClusterInstancePolicy usando awsbatch
- ParallelClusterUserPolicy usando Slurm
- ParallelClusterUserPolicy usando SGE ou Torque
- ParallelClusterUserPolicy usando awsbatch
- ParallelClusterLambdaPolicy usando SGE, Slurm ou Torque
- ParallelClusterLambdaPolicy usando awsbatch
- ParallelClusterUserPolicy para usuários

ParallelClusterInstancePolicy usando SGE, Slurm ou Torque

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE ou Torque agendadores. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.

Tópicos

ParallelClusterInstancePolicy usando Slurm

O exemplo a seguir define a `ParallelClusterInstancePolicy`, usando Slurm como programador.

JSON

```
{
    "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:us-east-1:111122223333:subnet/<COMPUTE SUBNET ID>",
                "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-east-1:111122223333:volume/*",
                "arn:aws:ec2:us-east-1::image/<IMAGE ID>",
                "arn:aws:ec2:us-east-1:111122223333:key-pair/<KEY NAME>",
                "arn:aws:ec2:us-east-1:111122223333:security-group/*",
                "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
                "arn:aws:ec2:us-east-1:111122223333: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:us-east-1:111122223333:stack/
parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
```

```
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:GetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:DeleteItem",
                "dynamodb:DescribeTable"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-east-1:111122223333:table/parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "DynamoDBTable"
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::us-east-1-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.us-east-1/*"
            ],
            "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` usando SGE ou Torque

O exemplo a seguir define `ParallelClusterInstancePolicy` usando SGE ou Torque como programador.

> **Note**
>
> Esta política se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

JSON

```
{
    "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:us-east-1:111122223333:subnet/<COMPUTE SUBNET ID>",
                "arn:aws:ec2:us-east-1:111122223333:network-interface/*",
                "arn:aws:ec2:us-east-1:111122223333:instance/*",
                "arn:aws:ec2:us-east-1:111122223333:volume/*",
                "arn:aws:ec2:us-east-1::image/<IMAGE ID>",
                "arn:aws:ec2:us-east-1:111122223333:key-pair/<KEY NAME>",
                "arn:aws:ec2:us-east-1:111122223333:security-group/*",
                "arn:aws:ec2:us-east-1:111122223333:launch-template/*",
                "arn:aws:ec2:us-east-1:111122223333: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:us-east-1:111122223333: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:us-east-1:111122223333:stack/
parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:GetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:DeleteItem",
                "dynamodb:DescribeTable"
            ],
            "Resource": [
                "arn:aws:dynamodb:us-east-1:111122223333:table/parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "DynamoDBTable"
        },
        {
            "Action": [
```

```
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::us-east-1-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.us-east-1/*"
            ],
            "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` usando `awsbatch`

O exemplo a seguir define a `ParallelClusterInstancePolicy`, usando `awsbatch` como programador. Você deve incluir as mesmas políticas atribuídas às definidas na pilha AWS Batch CloudFormation aninhada. `BatchUserRole` O ARN do `BatchUserRole` é fornecido como uma saída de pilha. Neste exemplo, "*<RESOURCES S3 BUCKET>*" é o valor da `cluster_resource_bucket` configuração; se não `cluster_resource_bucket` for especificado, "" será "parallelcluster-**<RESOURCES S3 BUCKET>*". O exemplo a seguir é uma visão geral das permissões necessárias:

JSON

```
{
    "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:us-east-1:111122223333:job-
definition/<AWS_BATCH_STACK - JOB_DEFINITION_SERIAL_NAME>:1",
                "arn:aws:batch:us-east-1:111122223333:job-
definition/<AWS_BATCH_STACK - JOB_DEFINITION_MNP_NAME>*",
                "arn:aws:batch:us-east-1:111122223333:job-queue/<AWS_BATCH_STACK
 - JOB_QUEUE_NAME>",
                "arn:aws:cloudformation:us-east-1:111122223333:stack/<STACK
 NAME>/*",
                "arn:aws:s3:::amzn-s3-demo-bucket/batch/*",
                "arn:aws:iam::111122223333:role/<AWS_BATCH_STACK - JOB_ROLE>",
                "arn:aws:ecs:us-east-1:111122223333:cluster/<ECS COMPUTE
 ENVIRONMENT>",
                "arn:aws:ecs:us-east-1:111122223333:container-instance/*",
                "arn:aws:logs:us-east-1:111122223333:log-group:/aws/batch/
job:log-stream:*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-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` usando Slurm

O exemplo a seguir define o `ParallelClusterUserPolicy`, usando Slurm como o programador. Neste exemplo, "*<RESOURCES S3 BUCKET>*" é o valor da `cluster_resource_bucket` configuração; se não `cluster_resource_bucket` for especificado, "" será "parallelcluster-*<RESOURCES S3 BUCKET>*".

> **Note**
>
> Se você usar uma função personalizada, `ec2_iam_role = <role_name>`, altere o recurso do IAM para incluir o nome dessa função a partir de:
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"`
> Para:
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/<role_name>"`

JSON

```
{
    "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:::amzn-s3-demo-bucket"
            ],
            "Effect": "Allow",
            "Sid": "S3ResourcesBucket"
        },
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::us-east-1-aws-parallelcluster*"
            ],
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ],
            "Effect": "Allow",
            "Sid": "S3Delete"
        },
        {
            "Action": [
                "iam:PassRole",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/<PARALLELCLUSTER EC2 ROLE NAME>",
                "arn:aws:iam::111122223333: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::111122223333:role/aws-service-role/*",
            "Effect": "Allow",
            "Sid": "IAMServiceLinkedRole"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile"
            ],
            "Resource": "arn:aws:iam::111122223333: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:us-east-1:111122223333:function:parallelcluster-
*",
                "arn:aws:lambda:us-east-1:111122223333:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Sid": "CloudWatch",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutDashboard",
                "cloudwatch:ListDashboards",
                "cloudwatch:DeleteDashboards",
                "cloudwatch:GetDashboard"
            ],
            "Resource": "*"
        }
    ]
}
```

ParallelClusterUserPolicy usando SGE ou Torque

> **Note**
>
> Esta seção se aplica somente às AWS ParallelCluster versões até e incluindo a versão 2.11.4. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

O exemplo a seguir define `ParallelClusterUserPolicy`, usando SGE ou Torque como programador. Neste exemplo, "*<RESOURCES S3 BUCKET>*" é o valor da `cluster_resource_bucket` configuração; se não `cluster_resource_bucket` for especificado, "" será "parallelcluster-**<RESOURCES S3 BUCKET>*".

> **Note**
>
> Se você usar uma função personalizada, `ec2_iam_role = <role_name>`, altere o recurso do IAM para incluir o nome dessa função a partir de:
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"`
> Para:
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/<role_name>"`

JSON

```
{
    "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:::amzn-s3-demo-bucket"
            ],
            "Effect": "Allow",
            "Sid": "S3ResourcesBucket"
        },
```

```
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::us-east-1-aws-parallelcluster*"
            ],
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket"
            ],
            "Effect": "Allow",
            "Sid": "S3Delete"
        },
        {
            "Action": [
                "iam:PassRole",
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:TagRole",
                "iam:SimulatePrincipalPolicy"
            ],
            "Resource": [
                "arn:aws:iam::111122223333:role/<PARALLELCLUSTER EC2 ROLE NAME>",
                "arn:aws:iam::111122223333: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::111122223333:role/aws-service-role/*",
        "Effect": "Allow",
        "Sid": "IAMServiceLinkedRole"
    },
    {
        "Action": [
            "iam:CreateInstanceProfile",
            "iam:DeleteInstanceProfile"
        ],
        "Resource": "arn:aws:iam::111122223333: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:us-east-1:111122223333:function:parallelcluster-
*",
                "arn:aws:lambda:us-east-1:111122223333:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Sid": "CloudWatch",
            "Effect": "Allow",
            "Action": [
                "cloudwatch:PutDashboard",
                "cloudwatch:ListDashboards",
                "cloudwatch:DeleteDashboards",
                "cloudwatch:GetDashboard"
            ],
            "Resource": "*"
        }
    ]
}
```

`ParallelClusterUserPolicy` usando `awsbatch`

O exemplo a seguir define a `ParallelClusterUserPolicy`, usando `awsbatch` como programador. Neste exemplo, "*<RESOURCES S3 BUCKET>*" é o valor da `cluster_resource_bucket` configuração; se não `cluster_resource_bucket` for especificado, "" será "parallelcluster-**<RESOURCES S3 BUCKET>*".

JSON

```
{
    "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:us-east-1:111122223333: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:us-east-1:111122223333: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:us-east-1:111122223333: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::111122223333:role/parallelcluster-*",
                "arn:aws:iam::111122223333:role/<PARALLELCLUSTER EC2 ROLE NAME>"
            ],
            "Effect": "Allow",
            "Sid": "IAMRole"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:GetInstanceProfile",
                "iam:PassRole"
            ],
            "Resource": "arn:aws:iam::111122223333: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:::amzn-s3-demo-bucket"
            ],
            "Effect": "Allow",
            "Sid": "S3ResourcesBucket"
```

```
        },
        {
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::us-east-1-aws-parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
                "s3:DeleteBucket",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-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:us-east-1:111122223333:function:parallelcluster-
*",
                "arn:aws:lambda:us-east-1:111122223333:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
```

```
        },
        {
            "Action": [
                "logs:*"
            ],
            "Resource": "arn:aws:logs:us-east-1:111122223333:*",
            "Effect": "Allow",
            "Sid": "Logs"
        },
        {
            "Action": [
                "codebuild:*"
            ],
            "Resource": "arn:aws:codebuild:us-east-1:111122223333: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` usando SGE, Slurm ou Torque

O exemplo a seguir define `ParallelClusterLambdaPolicy`, usando SGE, Slurm ou Torque como programador.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE ou Torque agendadores.

JSON

```
{
  "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:us-east-1:111122223333:table/parallelcluster-
*",
      "Effect": "Allow",
      "Sid": "DynamoDBTable"
    },
    {
      "Action": [
        "route53:ListResourceRecordSets",
        "route53:ChangeResourceRecordSets"
      ],
      "Resource": [
        "arn:aws:route53:::hostedzone/*"
      ],
      "Effect": "Allow",
      "Sid": "Route53DeletePolicy"
    }
  ]
}
```

`ParallelClusterLambdaPolicy` usando **`awsbatch`**

O exemplo a seguir define a `ParallelClusterLambdaPolicy`, usando `awsbatch` como programador.

JSON

```
{
  "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"
```

```
    }
  ]
}
```

`ParallelClusterUserPolicy` para usuários

O exemplo a seguir define o `ParallelClusterUserPolicy` para usuários que não precisam criar ou atualizar clusters. Os seguintes comandos são compatíveis.

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

JSON

```
{
    "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:us-
east-1:111122223333:autoScalingGroup:*:autoScalingGroupName/parallelcluster-*",
                "arn:aws:batch:us-east-1:111122223333:compute-environment/*",
                "arn:aws:cloudformation:us-
east-1:111122223333:stack/<CLUSTERNAME>/*",
                "arn:aws:dynamodb:us-east-1:111122223333:table/<CLUSTERNAME>"
            ]
        },
        {
            "Sid": "Describe",
            "Action": [
                "cloudformation:DescribeStacks",
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus"
            ],
            "Effect": "Allow",
            "Resource": "*"
        }
    ]
}
```

Agendadores suportados por AWS ParallelCluster

AWS ParallelCluster suporta vários agendadores, definidos usando a `scheduler` configuração.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE ou Torque agendadores. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.

Tópicos

- Son of Grid Engine (sge)
- Slurm Workload Manager (slurm)
- Torque Resource Manager (torque)
- AWS Batch (awsbatch)

Son of Grid Engine (`sge`)

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE ou Torque agendadores. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.

AWS ParallelCluster as versões 2.11.4 e anteriores usam Son of Grid Engine 8.1.9.

Slurm Workload Manager (`slurm`)

AWS ParallelCluster a versão 2.11.9 usa Slurm 20.11.9. Para saber mais sobre o Slurm, consulte https://slurm.schedmd.com/. Para downloads, consulte https://github.com/SchedMD/slurm/tags. Para o código-fonte, consulte https://github.com/SchedMD/slurm.

> **Important**
>
> AWS ParallelCluster é testado com parâmetros de Slurm configuração, que são fornecidos por padrão. Qualquer alteração feita nesses parâmetros de configuração Slurm é feita por sua conta e risco. Eles recebem suporte somente com base no melhor esforço.

AWS ParallelCluster versão (ões)	Versão do Slurm compatível
2.11.7, 2.11.8, 2.11.9	20.11.9
2.11.4 a 2.11.6	20.11.8
2.11.0 a 2.11.3	20.11.7
2.10.4	20.02.7
2.9.0 a 2.10.3	20.02.4
2.6 a 2.8.1	19.05.5

AWS ParallelCluster versão (ões)	Versão do Slurm compatível
2.5.0, 2.5.1	19.05.3-2
2.3.1 a 2.4.1	18.08.6-2
antes de 2.3.1	16.05.3-1

Modo de fila múltipla

AWS ParallelCluster a versão 2.9.0 introduziu o modo de fila múltipla. O modo de fila múltipla é suportado quando `scheduler` é definido como `slurm` e a configuração `queue_settings` é definida. Esse modo permite que diferentes tipos de instância coexistam nos nós de computação. Os recursos de computação que contêm os diferentes tipos de instância podem ser ampliados ou reduzidos verticalmente conforme necessário. No modo de fila, até cinco (5) filas são suportadas, e cada seção `[queue]` pode se referir a até três (3) seções `[compute_resource]`. Cada uma dessas seções `[queue]` é uma partição em Slurm Workload Manager. Para obter mais informações, consulte Guia do Slurm para o modo de várias filas e Tutorial do modo de fila múltipla.

Cada seção `[compute_resource]` em uma fila deve ter um tipo de instância diferente, e cada uma dessas `[compute_resource]` é dividida em nós estáticos e dinâmicos. Os nós estáticos de cada `[compute_resource]` são numerados de 1 até o valor de`min_count`. Os nós dinâmicos de cada `[compute_resource]` são numerados de um (1) a (`max_count` - `min_count`). Por exemplo, se `min_count` for 2 e `max_count` for 10, os nós dinâmicos para essa `[compute_resource]` serão numerados de um (1) a oito (8). A qualquer momento, pode haver entre zero (0) e o número máximo de nós dinâmicos em uma `[compute_resource]`.

As instâncias que são lançadas na frota de computação são atribuídas dinamicamente. Para ajudar a gerenciar isso, nomes de host são gerados para cada nó. O formato do nome de host é o seguinte:

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

- `$QUEUE` é o nome da fila. Por exemplo, se a seção começar`[queue` *`queue-name`*`]`, "`$QUEUE`" será "*`queue-name`*".
- `$STATDYN` é `st` para nós estáticos ou `dy` para nós dinâmicos.
- `$INSTANCE_TYPE` é o tipo de instância para o `[compute_resource]`, da configuração `instance_type`.

- `$NODENUM` é o número do nó. `$NODENUM` fica entre um (1) e o valor de `min_count`para nós estáticos e entre um (1) e (`max_count` - `min_count`) para nós dinâmicos.

Tanto os nomes de host quanto os nomes de domínio totalmente qualificados (FQDN) são criados usando zonas hospedadas do Amazon Route 53. O FQDN é `$HOSTNAME.$CLUSTERNAME.pcluster`, sendo `$CLUSTERNAME` o nome da seção `[cluster]` usada para o cluster.

Para converter sua configuração em um modo de fila, use o comando `pcluster-config convert`. Ele grava uma configuração atualizada com uma única seção `[queue]` chamada `[queue compute]`. Essa fila contém uma única seção `[compute_resource]` chamada `[compute_resource default]`. As configurações `[queue compute]` e `[compute_resource default]` foram migradas da seção `[cluster]` especificada.

Guia do Slurm para o modo de várias filas

AWS ParallelCluster a versão 2.9.0 introduziu o modo de fila múltipla e uma nova arquitetura de escalonamento para (). Slurm Workload Manager Slurm

As seções a seguir fornecem uma visão geral sobre o uso de um cluster Slurm com a arquitetura de escalabilidade recém-introduzida.

Visão geral do

A nova arquitetura de escalabilidade é baseada no Cloud Scheduling Guide do Slurm e no plug-in de economia de energia. Para obter mais informações sobre o plug-in de economia de energia, consulte o Guia de economia de energia do Slurm. Na nova arquitetura, os recursos com potencial para serem disponibilizados em um cluster geralmente são predefinidos na configuração do Slurm como nós de nuvem.

Ciclo de vida do nó da nuvem

Durante todo o ciclo de vida, os nós da nuvem entram em vários, se não em todos, dos seguintes estados: `POWER_SAVING`, `POWER_UP` (`pow_up`), `ALLOCATED` (`alloc`) e `POWER_DOWN` (`pow_dn`). Em alguns casos, um nó da nuvem pode entrar no estado `OFFLINE`. A lista a seguir detalha vários aspectos desses estados no ciclo de vida do nó na nuvem.

- Um nó em um estado `POWER_SAVING` aparece com um sufixo ~ (por exemplo, `idle~`) em `sinfo`. Nesse estado, não há instância do EC2 que serve de apoio ao nó. No entanto, Slurm ainda pode alocar trabalhos para o nó.

- Um nó em transição para um estado `POWER_UP` aparece com um sufixo # (por exemplo, `idle#`) em `sinfo`.
- Quando Slurm aloca um trabalho para um nó em um estado `POWER_SAVING`, o nó faz transferência automática para um estado `POWER_UP`. Caso contrário, os nós podem ser colocados no estado `POWER_UP` manualmente usando o comando `scontrol update nodename=`*`nodename`* `state=power_up`. Nesse estágio, o `ResumeProgram` é invocado, as instâncias do EC2 são iniciadas e configuradas para servir de backup para o nó `POWER_UP`.
- Um nó atualmente disponível para uso aparece sem um sufixo (por exemplo `idle`) em `sinfo`. Depois que o nó é configurado e se une ao cluster, ele fica disponível para executar trabalhos. Nesse estágio, o nó está configurado corretamente e pronto para uso. Como regra geral, recomendamos que o número de instâncias no EC2 seja igual ao número de nós disponíveis. Na maioria dos casos, os nós estáticos ficam sempre disponíveis após a criação do cluster.
- Um nó em transição para um estado `POWER_DOWN` aparece com um sufixo `%` (por exemplo, `idle%`) em `sinfo`. Os nós dinâmicos entram automaticamente no estado `POWER_DOWN` depois do `scaledown_idletime`. Por outro lado, os nós estáticos na maioria dos casos não são desligados. Contudo, os nós podem ser colocados no estado `POWER_DOWN` manualmente usando o comando `scontrol update nodename=`*`nodename`* `state=powering_down`. Nesse estado, a instância associada a um nó é encerrada e o nó retorna ao estado `POWER_SAVING` e para uso após o `scaledown_idletime`. A opção `scaledown-idletime` é salva na configuração Slurm como opção `SuspendTimeout`.
- Um nó que está off-line aparece com um sufixo * (por exemplo, `down*`) em `sinfo`. Um nó fica off-line se o controlador Slurm não conseguir entrar em contato com o nó ou se os nós estáticos forem desativados e as instâncias de backup forem encerradas.

Agora, considere os estados dos nós mostrados no exemplo `sinfo` a seguir.

```
$ 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]
```

Os nós `spot-st-t2large-[1-2]` e `efa-st-c5n18xlarge-1` já têm instâncias de backup configuradas e estão disponíveis para uso. Os nós `ondemand-dy-c52xlarge-[1-2]` estão no estado `POWER_UP` e devem estar disponíveis em instantes. O nó `gpu-dy-g38xlarge-1` está no estado `POWER_DOWN` e passará para o estado `POWER_SAVING` depois de `scaledown_idletime` (o padrão é 120 segundos).

Todos os outros nós estão no estado `POWER_SAVING` sem nenhuma instância do EC2 como backup.

Trabalhando com um nó disponível

Um nó disponível é apoiado por uma instância do EC2. Por padrão, o nome do nó pode ser usado para fazer SSH diretamente na instância (por exemplo, `ssh efa-st-c5n18xlarge-1`). O endereço IP privado da instância pode ser recuperado usando o comando `scontrol show nodes` *`nodename`* e verificando o campo `NodeAddr`. Para os nós que não estão disponíveis, o campo `NodeAddr` não deve apontar para uma instância do EC2 em execução. Em vez disso, deve ser igual ao nome do nó.

Estados do trabalho e envio

Na maioria dos casos, os trabalhos enviados são imediatamente alocados aos nós do sistema ou colocados como pendentes se todos os nós estiverem alocados.

Se os nós alocados para um trabalho incluírem qualquer nó em um estado `POWER_SAVING`, o trabalho começará com um estado `CF` ou `CONFIGURING`. Nesse momento, o trabalho aguarda para que os nós no estado `POWER_SAVING` façam a transição para o estado `POWER_UP` e fiquem disponíveis.

Depois que todos os nós alocados para uma trabalho estiverem disponíveis, o trabalho entrará no estado `RUNNING` (`R`).

Por padrão, todos os trabalhos são enviados para a fila padrão (conhecida como partição em Slurm). Isso é representado por um sufixo `*` após o nome da fila. Você pode selecionar uma fila usando a opção de envio de trabalho `-p`.

Todos os nós são configurados com os seguintes recursos, que podem ser usados nos comandos de envio de tarefas:

- Um tipo de instância (por exemplo, `c5.xlarge`)
- Um tipo de nó (que pode ser `dynamic` ou `static`.)

Você pode ver todos os recursos disponíveis para um determinado nó usando o comando `scontrol show nodes` *nodename* e verificando a lista `AvailableFeatures`.

Outra consideração são os trabalhos. Considere o estado inicial do cluster, que você pode ver executando o comando `sinfo`.

```
$ 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]
```

Observe que a lista padrão é `spot`. É indicada pelo sufixo `*`.

Envia um trabalho para um nó estático para a fila padrão (`spot`).

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

Envia um trabalho para um nó dinâmico para a fila EFA.

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

Envia um trabalho para oito (8) nós `c5.2xlarge` e dois (2) nós `t2.xlarge` para a fila ondemand.

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

Envia um trabalho para um nó da GPU para a fila gpu.

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

Agora, considere o estado dos trabalhos usando o comando `squeue`.

```
$ 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
```

Os trabalhos 7 e 8 (nas filas `spot` e `efa`) já estão em execução (`R`). Os trabalhos 12 e 13 ainda estão com configuração em andamento (`CF`), provavelmente aguardando a disponibilização das instâncias.

```
# 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
```

Recursos e estado do nó

Na maioria dos casos, os estados dos nós são totalmente gerenciados de AWS ParallelCluster acordo com os processos específicos no ciclo de vida do nó na nuvem descritos anteriormente neste tópico.

No entanto, AWS ParallelCluster também substitui ou encerra nós não íntegros em `DRAINED` estados `DOWN` e nós que têm instâncias de backup não íntegras. Para obter mais informações, consulte `clustermgtd`.

Estados de partição

AWS ParallelCluster suporta os seguintes estados de partição. Uma partição Slurm é uma fila no AWS ParallelCluster.

- UP: indica que a partição está em um estado ativo. Esse é o valor padrão de uma partição. Nesse estado, todos os nós da partição ficam ativos e disponíveis para uso.

- `INACTIVE`: indica que a partição está em um estado inativo. Nesse estado, todas as instâncias de backup dos nós de uma partição inativa são encerradas. Novas instâncias não são iniciadas para nós em uma partição inativa.

início e término do pcluster

Quando `pcluster stop` é executado, todas as partições são colocadas no `INACTIVE` estado e os AWS ParallelCluster processos mantêm as partições no `INACTIVE` estado.

Quando `pcluster start` é executado, todas as partições são inicialmente colocadas no estado `UP`. No entanto, AWS ParallelCluster os processos não mantêm a partição em um `UP` estado. Você precisa alterar os estados das partições manualmente. Todos os nós estáticos ficam disponíveis após alguns minutos. Observe que definir uma partição como `UP` não ativa nenhuma capacidade dinâmica. Se `initial_count` for maior que `max_count`, então `initial_count` talvez não seja satisfeito quando o estado da partição for alterado para o estado `UP`.

Quando `pcluster start` e `pcluster stop` estão em execução, você pode verificar o estado do cluster executando o comando `pcluster status` e verificando o `ComputeFleetStatus`. A seguir, listamos estados possíveis:

- `STOP_REQUESTED`: a solicitação `pcluster stop` é enviada ao cluster.
- `STOPPING`: o processo `pcluster` está interrompendo o cluster no momento.
- `STOPPED`: o processo `pcluster` finalizou o processo de interrupção, todas as partições estão em estado `INACTIVE` e todas as instâncias de computação foram encerradas.
- `START_REQUESTED`: a solicitação `pcluster start` é enviada ao cluster.
- `STARTING`: o processo `pcluster` está iniciando o cluster no momento
- `RUNNING`: o processo `pcluster` finalizou o processo inicial, todas as partições estão no estado `UP` e os nós estáticos ficam disponíveis após alguns minutos.

Controle manual das filas

Em alguns casos, talvez você queira ter algum controle manual sobre os nós ou sobre a fila (conhecida como partição em Slurm) em um cluster. Você pode gerenciar nós em um cluster por meio dos seguintes procedimentos comuns.

- Ative os nós dinâmicos no estado `POWER_SAVING`: execute o comando `scontrol update nodename=`*`nodename`* `state=power_up` ou envie um trabalho de espaço reservado `sleep 1`

solicitando um determinado número de nós e dependa do Slurm para ativar o número necessário de nós.

- Desligue os nós dinâmicos antes[scaledown_idletime](): defina os nós dinâmicos para DOWN com o `scontrol update nodename=nodename state=down` comando. AWS ParallelCluster encerra e redefine automaticamente os nós dinâmicos abatidos. Em geral, não recomendamos definir os nós para POWER_DOWN diretamente com o comando `scontrol update nodename=nodename state=power_down`. Isso porque AWS ParallelCluster gerencia automaticamente o processo de desligamento. Nenhuma intervenção manual é necessária. Portanto, convém tentar definir nós para DOWN sempre que possível.
- Desativar uma fila (partição) ou interromper todos os nós estáticos em uma partição específica: defina uma fila específica INACTIVE com o comando `scontrol update partition=queue name state=inactive`. Isso encerra todas as instâncias de backup de nós na partição.
- Ativar uma fila (partição): defina uma fila específica para INACTIVE com o comando `scontrol update partition=queue name state=up`.

Comportamento e ajustes de escalabilidade

Veja a seguir o exemplo do fluxo de trabalho de escalabilidade normal:

- O programador recebe um trabalho que requer dois nós.
- O programador faz a transição de dois nós para um estado POWER_UP e chama `ResumeProgram` com os nomes dos nós (por exemplo, `queue1-dy-c5xlarge-[1-2]`).
- `ResumeProgram` inicia duas instâncias do EC2 e atribui os endereços IP e nomes de host privados de `queue1-dy-c5xlarge-[1-2]`, aguardando `ResumeTimeout` (o período padrão é de 60 minutos (1 hora)) antes de redefinir os nós.
- As instâncias são configuradas e se unem ao cluster. O trabalho começa a ser executado nas instâncias.
- O trabalho é concluído.
- Depois de decorrido o `SuspendTime` configurado (que está definido como [scaledown_idletime]()), as instâncias são colocadas no estado POWER_SAVING pelo programador. O programador coloca `queue1-dy-c5xlarge-[1-2]` no estado POWER_DOWN e chama o `SuspendProgram` com os nomes dos nós.
- `SuspendProgram` é chamado para dois nós. Os nós permanecem no estado POWER_DOWN, por exemplo, permanecendo `idle%` por um `SuspendTimeout` (o período padrão é 120 segundos (2 minutos)). Depois de `clustermgtd` detectar que os nós estão sendo desligados, ele encerra

as instâncias de backup. Em seguida, ele se configura `queue1-dy-c5xlarge-[1-2]` no estado ocioso e redefine o endereço IP privado e o nome do host para que possam ser ativados novamente para futuros trabalhos.

Agora, se algo der errado e uma instância de um determinado nó não puder ser iniciada por algum motivo, acontece o seguinte.

- O programador recebe um trabalho que requer dois nós.
- O programador coloca dois nós de expansão na nuvem para o estado `POWER_UP` e chama `ResumeProgram` com os nomes dos nós (por exemplo `queue1-dy-c5xlarge-[1-2]`).
- `ResumeProgram` inicia somente uma (1) instância do EC2 e configura `queue1-dy-c5xlarge-1`, mas falhou ao iniciar uma instância para `queue1-dy-c5xlarge-2`.
- `queue1-dy-c5xlarge-1` não será afetado e ficará online após atingir o estado `POWER_UP`.
- `queue1-dy-c5xlarge-2` é colocado no estado `POWER_DOWN` e o trabalho é reenfileirado automaticamente porque Slurm detecta uma falha no nó.
- `queue1-dy-c5xlarge-2` fica disponível após `SuspendTimeout` (o padrão é 120 segundos (2 minutos)). Enquanto isso, o trabalho é recolocado na fila e pode começar a ser executado em outro nó.
- O processo acima se repete até que o trabalho possa ser executado em um nó disponível sem que ocorra uma falha.

Há dois parâmetros de temporização que podem ser ajustados, se necessário.

- `ResumeTimeout` (o padrão é 60 minutos (1 hora)): `ResumeTimeout` controla o tempo que Slurm espera antes de colocar o nó para o estado inativo.
  - Pode ser útil estender isso se o processo de pre/post instalação demorar quase esse tempo.
  - Esse também é o tempo máximo de AWS ParallelCluster espera antes de substituir ou redefinir um nó, caso haja algum problema. Os nós de computação terminam automaticamente se ocorrer algum erro durante a inicialização ou a configuração. Em seguida, os processos do AWS ParallelCluster também substituem o nó quando percebem que a instância foi encerrada.
- `SuspendTimeout` (o padrão é 120 segundos (2 minutos)): `SuspendTimeout` controla a rapidez com que os nós são colocados de volta no sistema e que ficam prontos para uso novamente.
  - Um `SuspendTimeout` menor significaria que os nós serão redefinidos mais rapidamente, e que Slurm pode tentar executar instâncias com mais frequência.

- Um `SuspendTimeout` mais longo faz com que os nós com falha reiniciem mais lentamente. Enquanto isso, Slurm tenta usar outros nós. Se `SuspendTimeout` demorar mais do que alguns minutos, Slurm tenta percorrer todos os nós do sistema. Um `SuspendTimeout` mais longo pode ser benéfico para sistemas de grande escala (mais de 1.000 nós) para reduzir o estresse em Slurm ao tentar enfileirar novamente os trabalhos que falham com frequência.
- Observe que `SuspendTimeout` isso não se refere ao tempo AWS ParallelCluster esperado para encerrar uma instância de apoio para um nó. As instâncias de backup para nós de `power` down são encerradas imediatamente. O processo de encerramento geralmente é concluído em alguns minutos. No entanto, durante esse período, o nó permanece no estado desligado e indisponível para uso no programador.

Logs para a nova arquitetura

A lista a seguir contém os principais logs da arquitetura de várias filas.
O nome do stream de log usado com o Amazon CloudWatch Logs tem o formato*`{hostname}.{instance_id}.{logIdentifier}`*, que *`logIdentifier`* segue os nomes dos registros. Para obter mais informações, consulte Integração com Amazon CloudWatch Logs.

- `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`)

Problemas comuns e como depurar:

Nós que falharam ao iniciar, ativar ou ingressar no cluster:

- Nós dinâmicos:
  - Verifique o log `ResumeProgram` para ver se `ResumeProgram` foi chamado alguma vez com o nó. Caso contrário, verifique o log `slurmctld` para determinar se Slurm alguma vez tentou chamar `ResumeProgram` com o nó. Observe que permissões incorretas em `ResumeProgram` podem fazer com que ele falhe silenciosamente.
  - Se `ResumeProgram` for chamado, verifique se uma instância foi executada para o nó. Se a instância não puder ser iniciada, deve haver uma mensagem de erro clara sobre o motivo da falha na inicialização da instância.
  - Se uma instância foi iniciada, pode ter havido algum problema durante o processo de bootstrap. Encontre o endereço IP privado e o ID da instância correspondentes no `ResumeProgram` registro e veja os registros de bootstrap correspondentes para a instância específica em CloudWatch Logs.
- Nós estáticos:
  - Verifique o log `clustermgtd` para ver se foram iniciadas instâncias para o nó. Caso negativo, deve haver mensagens de erro claras sobre o motivo da falha na inicialização da instância.
  - Se uma instância foi iniciada, houve algum problema durante o processo de bootstrap. Encontre o IP privado e o ID da instância correspondentes no `clustermgtd` registro e veja os registros de bootstrap correspondentes para a instância específica em CloudWatch Logs.

Nós substituídos ou encerrados inesperadamente e falhas nos nós

- Nodos replaced/terminated inesperadamente
  - Na maioria dos casos, `clustermgtd` lida com todas as ações de manutenção do nó. Para verificar se um nó foi `clustermgtd` substituído ou encerrado, verifique o log `clustermgtd`.
  - Se `clustermgtd` tiver substituído ou encerrado o nó, deverá haver uma mensagem indicando o motivo da ação. Se o motivo estiver relacionado com o programador (por exemplo, o nó for `DOWN`), verifique o log `slurmctld` para obter mais detalhes. Se o motivo estiver relacionado ao EC2, use ferramentas para verificar o status ou os logs dessa instância. Por exemplo, você pode verificar se a instância tinha eventos programados ou falhou nas verificações do status de integridade do EC2.

  - Se `clustermgtd` não encerrou o nó, verifique se o nó foi encerrado por `computemgtd` ou se a instância foi encerrada pelo EC2 para recuperar uma instância Spot.
- Falhas do nó
  - Na maioria dos casos, os trabalhos são automaticamente enfileirados se um nó falhar. Examine o log `slurmctld` para ver por que um trabalho ou um nó falhou e analise a situação a partir daí.

Falha ao substituir ou encerrar instâncias, falha ao desligar os nós

- Em geral, `clustermgtd` lida com todas as ações esperadas de encerramento da instância. Examine o log `clustermgtd` para ver por que ele não conseguiu substituir ou encerrar um nó.
- Se os nós dinâmicos falharem por `scaledown_idletime`, consulte o log `SuspendProgram` para ver se há um programa pelo `slurmctld` com o nó específico como argumento. Na verdade, `SuspendProgram` não executa nenhuma ação específica. Em vez disso, ele só cria logs quando é chamado. Todos os encerramentos de instâncias e redefinições de `NodeAddr` são concluídos por `clustermgtd`. Slurm coloca os nós para `IDLE` após `SuspendTimeout`.

Outros problemas

- AWS ParallelCluster não toma decisões de alocação de tarefas ou escalabilidade. Ele só tenta iniciar, encerrar e manter os recursos de acordo com as instruções do Slurm.

  Para problemas relacionados à alocação de trabalhos, alocação de nós e decisão de escalabilidade, consulte o log `slurmctld` em busca de erros.

Torque Resource Manager (**`torque`**)

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE ou Torque agendadores. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.

AWS ParallelCluster as versões 2.11.4 e anteriores usam Torque Resource Manager 6.1.2.
Para obter mais informações sobre o Torque Resource Manager 6.1.2, consulte http://

docs.adaptivecomputing.com/torque/6-1-2/releaseNotes/torquerelnote.htm. Para obter a documentação, consulte http://docs.adaptivecomputing.com/torque/6-1-2/adminGuide/torque.htm. Para o código-fonte, consulte https://github.com/adaptivecomputing/torque/tree/6.1.2.

AWS ParallelCluster as versões 2.4.0 e anteriores usam Torque Resource Manager 6.0.2. Para obter as notas de release, consulte http://docs.adaptivecomputing.com/torque/6-0-2/releaseNotes/torqueReleaseNotes6.0.2.pdf. Para obter a documentação, consulte http://docs.adaptivecomputing.com/torque/6-0-2/adminGuide/help.htm. Para o código-fonte, consulte https://github.com/adaptivecomputing/torque/tree/6.0.2.

AWS Batch (`awsbatch`)

Para obter informações sobre AWS Batch, consulte AWS Batch. Para obter a documentação, consulte o Guia do usuário do AWS Batch.

AWS ParallelCluster Comandos CLI para AWS Batch

Quando você usa o `awsbatch` agendador, os comandos da AWS ParallelCluster CLI AWS Batch para são instalados automaticamente no nó AWS ParallelCluster principal. A CLI usa operações de AWS Batch API e permite as seguintes operações:

- Enviar e gerenciar trabalhos.
- Monitorar trabalhos, filas e hosts.
- Espelhar comandos do programador tradicional.

> **Important**
>
> AWS ParallelCluster não suporta trabalhos de GPU para AWS Batch. Para obter mais informações, consulte Trabalhos de GPU.

Tópicos

- awsbsub
- awsbstat
- awsbout
- awsbkill
- awsbqueues

- awsbhosts

awsbsub

Envia trabalhos para a fila de trabalhos do cluster.

```
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 não suporta trabalhos de GPU para AWS Batch. Para obter mais informações, consulte Trabalhos de GPU.

Argumentos posicionais

command

Envia o trabalho (o comando especificado deve estar disponível nas instâncias de computação) ou o nome do arquivo a ser transferido. Consulte também `--command-file`.

`arguments`

(Opcional) Especifica argumentos para o comando ou arquivo de comando.

Argumentos nomeados

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

Nomeia a tarefa. O primeiro caractere deve ser uma letra ou um número. O nome do trabalho pode conter letras (minúsculas e maiúsculas), números, hifens e sublinhados, e ter até 128 caracteres de comprimento.

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

Especifica o cluster a ser usado.

`-cf, --command-file`

Indica que o comando é um arquivo a ser transferido para as instâncias de computação.

Padrão: False

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

Especifica a pasta a ser usada como diretório de trabalho da tarefa. Se um diretório de trabalho não for especificado, o trabalho será executado na subpasta `job-`*`<AWS_BATCH_JOB_ID>`* do diretório inicial do usuário. Você pode usar esse parâmetro ou o parâmetro `--parent-working-dir`.

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

Especifica a pasta pai do diretório de trabalho da tarefa. Se um diretório de trabalho pai não for especificado, o padrão será o diretório inicial do usuário. Uma subpasta chamada `job-`*`<AWS_BATCH_JOB_ID>`* será criada no diretório de trabalho pai. Você pode usar esse parâmetro ou o parâmetro `--working-dir`.

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

Especifica o arquivo a ser transferido para as instâncias de computação, no diretório de trabalho do trabalho. Você pode especificar vários parâmetros de arquivo de entrada.

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

Especifica o número de v CPUs a ser reservado para o contêiner. Quando usado junto com `–nodes`, ele identifica o número de v CPUs para cada nó.

Padrão: 1

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

Especifica o limite rígido de memória (em MiB) a ser fornecido para a tarefa. Se o trabalho tentar exceder o limite de memória especificado aqui, ele será encerrado.

Padrão: 128

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

Especifica uma lista separada por vírgulas dos nomes das variáveis de ambiente a serem exportadas para o ambiente da tarefa. Para exportar todas as variáveis de ambiente, especifique "all". Observe que uma lista de variáveis de ambiente "all" não inclui as listadas no parâmetro `–env-blacklist`, nem variáveis que começam com os prefixos `PCLUSTER_*` e `AWS_*`.

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

Especifica uma lista separada por vírgulas dos nomes das variáveis de ambiente a não serem exportadas para o ambiente da tarefa. Por padrão, `HOME`, `PWD`, `USER`, `PATH`, `LD_LIBRARY_PATH`, `TERM` e `TERMCAP` não são exportadas.

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

Especifica o número de vezes que um trabalho será movido para o status `RUNNABLE`. Você pode especificar entre 1 e 10 tentativas. Se o valor de tentativas for maior que 1, o trabalho será repetido se falhar, até ser movido para um status `RUNNABLE` esse número especificado de vezes.

Padrão: 1

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

Especifica a duração em segundos (medida a partir do `startedAt` carimbo de data/hora da tentativa de trabalho) após a qual AWS Batch encerra seu trabalho se ele não tiver sido concluído. O valor de tempo limite deve ser pelo menos 60 segundos.

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

Especifica o número de nós a serem reservados para a tarefa. Especifique um valor para esse parâmetro a fim de habilitar o envio paralelo com vários nós.

> **Note**
>
> Trabalhos paralelos de vários nós não são compatíveis quando o parâmetro `cluster_type` é definido como `spot`.

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

Indica o tamanho da matriz. Você pode especificar um valor entre 2 e 10.000. Se você especificar propriedades de matriz para uma tarefa, ela se torna uma tarefa de matriz.

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

Especifica uma lista de dependências separadas por ponto-e-vírgula para uma tarefa. Uma tarefa pode depender de, no máximo, outras 20 tarefas. Você pode especificar um tipo de dependência `SEQUENTIAL` sem especificar um ID de trabalho para trabalhos de matriz. Uma dependência sequencial permite que cada tarefa filho da matriz seja concluída sequencialmente, a partir do índice 0. Você também pode especificar uma dependência do tipo N_TO_N com um ID de tarefa

para tarefas de matriz. Uma dependência N_TO_N significa que cada índice filho dessa tarefa precisa aguardar que o índice filho correspondente de cada dependência seja concluído antes de poder começar. A sintaxe desse parâmetro é “jobId=*<string>*, type=*<string>*;...”.

awsbstat

Mostra as tarefas que são enviadas na fila de tarefas do cluster.

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

Argumentos posicionais

job_ids

Especifica a lista de trabalhos separados por espaços IDs a serem mostrados na saída. Se o trabalho for uma matriz de trabalhos, todos os trabalhos filho são exibidos. Se uma única tarefa for solicitada, seus detalhes são exibidos.

Argumentos nomeados

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

Indica o cluster a ser usado.

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

Especifica uma lista separada por vírgulas de status de tarefa a ser incluída. O status de tarefa padrão é "ativo". Os valores aceitos são: `SUBMITTED`, `PENDING`, `RUNNABLE`, `STARTING`, `RUNNING`, `SUCCEEDED`, `FAILED` e `ALL`.

Padrão: “`SUBMITTED,PENDING,RUNNABLE,STARTING,RUNNING`”

-e, --expand-children

Expande tarefas com filhos (matriz e paralelo com vários nós).

Padrão: False

-d, --details

Mostra os detalhes da tarefa.

Padrão: False

awsbout

Mostra a saída de uma tarefa específica.

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

Argumentos posicionais

job_id

Especifica o ID da tarefa.

Argumentos nomeados

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

Indica o cluster a ser usado.

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

Obtém as primeiras *HEAD* linhas da saída do trabalho.

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

Obtém as últimas linhas <finais> da saída da tarefa.

-s, --stream

Obtém a saída da tarefa e aguarda a saída adicional que será produzida. Esse argumento pode ser usado em conjunto com –tail para iniciar a partir das últimas linhas <finais> da saída da tarefa.

Padrão: False

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

Define o período de streaming.

Padrão: 5

awsbkill

Cancela ou encerra tarefas enviadas no cluster.

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

Argumentos posicionais

job_ids

Especifica a lista de trabalhos separados por espaços a IDs serem cancelados ou encerrados.

Argumentos nomeados

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

Indica o nome do cluster a ser usado.

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

Indica a mensagem que será anexada a uma tarefa, explicando o motivo do cancelamento.

Padrão: “Terminated by the user”

awsbqueues

Mostra a fila de tarefas associada ao cluster.

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

Argumentos posicionais

job_queues

Especifica a lista separada por espaços de nomes de fila a ser exibida. Se uma única fila for solicitada, seus detalhes são exibidos.

Argumentos nomeados

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

Especifica o nome do cluster a ser usado.

-d, --details

Indica se deve mostrar os detalhes das filas.

Padrão: False

awsbhosts

Mostra os hosts que pertencem ao ambiente de computação do cluster.

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

Argumentos posicionais

instance_ids

Especifica uma lista de instâncias separadas por espaços. IDs Se uma única instância for solicitada, seus detalhes são exibidos.

Argumentos nomeados

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

Especifica o nome do cluster a ser usado.

-d, --details

Indica se deve mostrar os detalhes dos hosts.

Padrão: False

AWS ParallelCluster recursos e marcação

Com AWS ParallelCluster você pode criar tags para rastrear e gerenciar seus AWS ParallelCluster recursos. Você define as tags que deseja AWS CloudFormation criar e propagar para todos os recursos do cluster na `tags` seção do arquivo de configuração do cluster. Você também pode usar tags que AWS ParallelCluster são geradas automaticamente para rastrear e gerenciar seus recursos.

Quando você cria um cluster, o cluster e seus recursos são marcados com as tags AWS ParallelCluster e AWS systems definidas nesta seção.

AWS ParallelCluster aplica tags às instâncias, volumes e recursos do cluster. Para identificar a pilha do cluster, AWS CloudFormation aplique as tags AWS do sistema às instâncias do cluster. Para

identificar os modelos de execução do cluster EC2, o EC2 aplica tags de sistema às instâncias. Você pode usar essas tags para visualizar e gerenciar seus AWS ParallelCluster recursos.

Você não pode modificar as tags AWS do sistema. Para evitar impactos na AWS ParallelCluster funcionalidade, não modifique as AWS ParallelCluster tags.

Veja a seguir um exemplo de uma tag AWS do sistema para um AWS ParallelCluster recurso. Não é possível modificá-la.

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

Veja a seguir exemplos de AWS ParallelCluster tags aplicadas a um recurso. Não a modifique.

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

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

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

Você pode ver essas tags na seção EC2 do Console de gerenciamento da AWS.

Visualizar tags

1. Navegue pelo console do EC2 em. https://console.aws.amazon.com/ec2/
2. Para visualizar todas as tags do cluster, escolha Tags no painel de navegação.
3. Para visualizar as tags de cluster por instância, escolha Instâncias no painel de navegação.
4. Selecione uma instância de cluster.
5. Escolha a guia Gerenciar tags nos detalhes da instância e visualize as tags.
6. Escolha a guia Armazenamento nos detalhes da instância.
7. Selecione o ID do Volume.
8. Em Volumes, escolha o volume.
9. Escolha a guia Tags nos detalhes do volume e visualize as tags.

AWS ParallelCluster tags de instância do head node

Chave	Valor da tag
ClusterName	*clustername*
Name	Master
Application	parallelcluster- *clustername*
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 etiquetas de volume raiz do nó principal

Chave de tag	Valor da tag
ClusterName	*clustername*
Application	parallelcluster- *clustername*
aws-parallelcluster-node-type	Master

AWS ParallelCluster tags de instância do nó de computação

Chave	Valor da tag
`ClusterName`	*`clustername`*
`aws-parallelcluster-node-type`	`Compute`
`aws:ec2launchtemplate:id`	*`lt-1234567890abcdef0`*
`aws:ec2launchtemplate:version`	*`1`*
`QueueName`	*`queue-name`*
`Version`	*`2.11.9`*

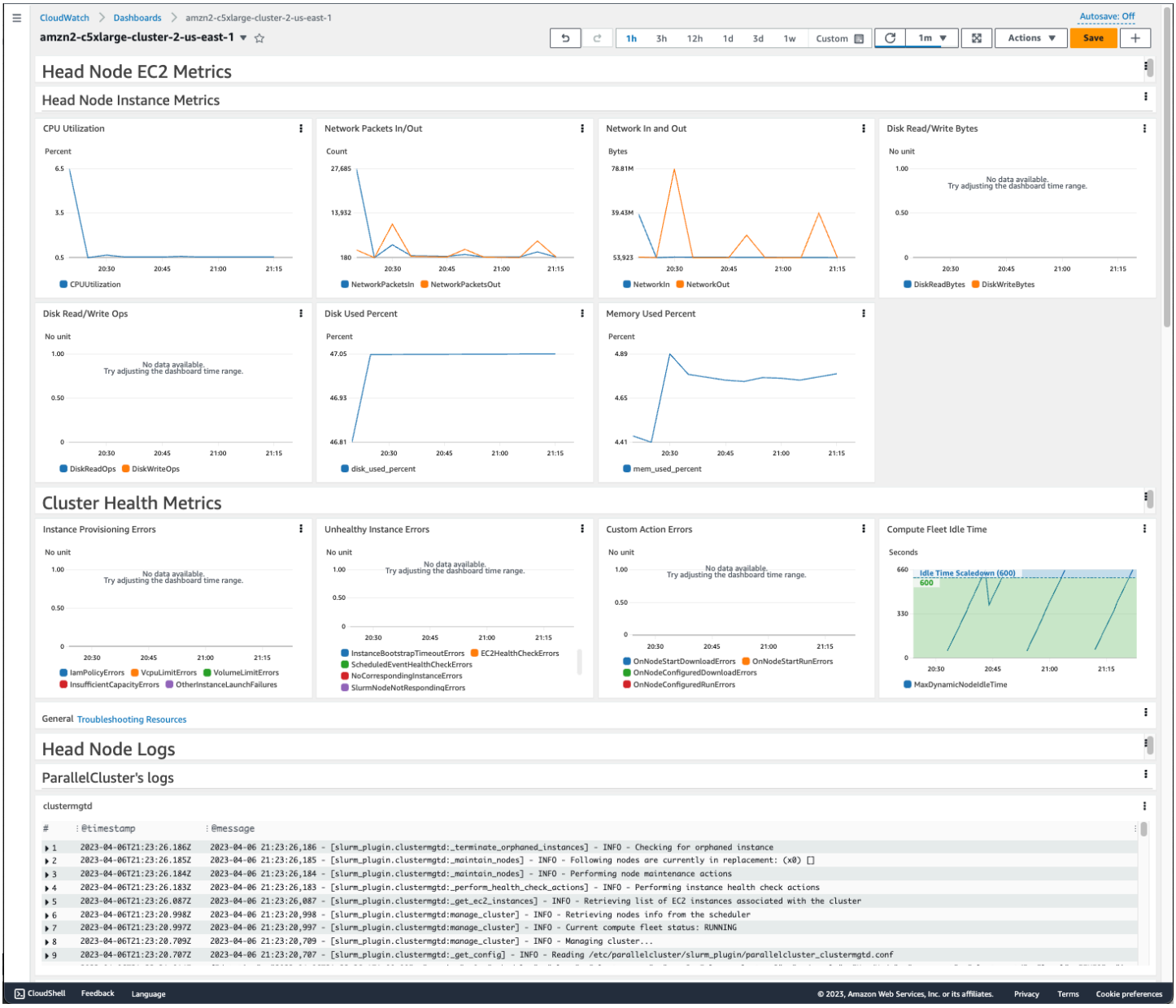
AWS ParallelCluster tags de volume raiz do nó de computação

Chave de tag	Valor da tag
`ClusterName`	*`clustername`*
`Application`	`parallelcluster-` *`clustername`*
`aws-parallelcluster-node-type`	`Compute`
`QueueName`	*`queue-name`*
`Version`	*`2.11.9`*

CloudWatch Painel da Amazon

A partir da AWS ParallelCluster versão 2.10.0, um CloudWatch painel da Amazon é criado quando o cluster é criado. Isso facilita o monitoramento dos nós em seu cluster e a visualização dos registros armazenados no Amazon CloudWatch Logs. O nome do painel é `parallelcluster-`*`ClusterName`*`-`*`Region`*. *`ClusterName`* é o nome do seu cluster e *`Region`* é o nome Região da AWS do cluster. Você pode acessar o painel no console ou abrindo `https://console.aws.amazon.com/cloudwatch/home?region=`*`Region`*`#dashboards:name=parallelcluster-`*`ClusterName`*.

A imagem a seguir mostra um exemplo de CloudWatch painel para um cluster.



A primeira seção do painel exibe gráficos das métricas do EC2 do nó principal. Se seu cluster tiver armazenamento compartilhado, a próxima seção mostrará métricas de armazenamento compartilhado. A seção final lista os registros do Head Node agrupados por registros ParallelCluster do, registros do Scheduler, registros de integração do NICE DCV e registros do sistema.

Para obter mais informações sobre os CloudWatch painéis da Amazon, consulte Como usar CloudWatch painéis da Amazon no Guia CloudWatch do usuário da Amazon.

Se você não quiser criar o CloudWatch painel da Amazon, você deve concluir estas etapas: Primeiro, adicione uma `[dashboard]`seção ao seu arquivo de configuração e, em seguida, adicione o nome dessa seção como o valor da `dashboard_settings` configuração em sua `[cluster]`seção. Na sua seção `[dashboard]`, defina `enable = false`.

Por exemplo, se sua seção `[dashboard]` for nomeada `myDashboard` e sua seção `[cluster]` tiver um nome `myCluster`, suas alterações serão semelhantes a isso.

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

[dashboard MyDashboard]
enable = false
```

Integração com Amazon CloudWatch Logs

A partir da AWS ParallelCluster versão 2.6.0, os registros comuns são armazenados em CloudWatch Registros por padrão. Para obter mais informações sobre CloudWatch registros, consulte o Guia do usuário do Amazon CloudWatch Logs. Para configurar a integração do CloudWatch Logs, consulte a `[cw_log]`seção e a `cw_log_settings` configuração.

Um grupo de logs é criado para cada cluster com um nome `/aws/parallelcluster/`*`cluster-name`* (por exemplo, `/aws/parallelcluster/testCluster`). Cada log (ou conjunto de logs, se o caminho contiver um `*`) em cada nó possui um fluxo de logs denominado *`{hostname}`*`.`*`{instance_id}`*`.`*`{logIdentifier}`*. (Por exemplo, `ip-172-31-10-46.i-02587cf29cc3048f3.nodewatcher`). Os dados de log são enviados CloudWatch pelo CloudWatch agente, que é executado como `root` em todas as instâncias do cluster.

A partir da AWS ParallelCluster versão 2.10.0, um CloudWatch painel da Amazon é criado quando o cluster é criado. Esse painel facilita a revisão dos registros armazenados em CloudWatch Registros. Para obter mais informações, consulte CloudWatch Painel da Amazon.

Essa lista contém o caminho *`logIdentifier`* e para os fluxos de log disponíveis para plataformas, agendadores e nós.

Fluxos de log disponíveis para plataformas, programadores e nós

Plataformas	Programadores	Nodes	Fluxos de log
amazon centos ubuntu	awsbatch slurm	HeadNode	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` 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	awsbatch slurm	ComputeFleet HeadNode	cloud-init: `/var/log/cloud-init.log` supervisord: `/var/log/supervisord.log`
amazon centos ubuntu	slurm	ComputeFleet	cloud-init-output: `/var/log/cloud-init-output.log` computemgtd: `/var/log/parallelcluster/computemgtd` slurmd: `/var/log/slurmd.log`
amazon centos ubuntu	slurm	HeadNode	clustermgtd: `/var/log/parallelcluster/clustermgtd` slurm_resume: `/var/log/parallelcluster/slurm_resume.log`

Plataformas	Programadores	Nodes	Fluxos de 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`
ubuntu	awsbatc slurm	Comput eet HeadNc	syslog: `/var/log/syslog`

Os trabalhos em clusters que usam AWS Batch armazenam a saída de trabalhos que atingiram um `FAILED` estado `RUNNINGSUCCEEDED`, ou em CloudWatch Logs. O grupo de logs é `/aws/batch/job`, e o formato do nome do fluxo de logs é *`jobDefinitionName`*`/default/`*`ecs_task_id`*. Por padrão, esses logs são configurados para nunca expirar, mas você pode modificar o período de retenção. Para obter mais informações, consulte Alterar a retenção de dados de log em CloudWatch Logs no Guia do usuário do Amazon CloudWatch Logs.

> **Note**
>
> `chef-client`,`cloud-init-output`,`clustermgtd`,, `computemgtd`slurm_resume, e `slurm_suspend` foram adicionados na AWS ParallelCluster versão 2.9.0. Para a AWS ParallelCluster versão 2.6.0, `/var/log/cfn-init-cmd.log` (`cfn-init-cmd`) e `/var/log/cfn-wire.log` (`cfn-wire`) também foram armazenados em CloudWatch Logs.

Elastic Fabric Adapter

O Elastic Fabric Adapter (EFA) é um dispositivo de rede que tem recursos de OS-bypass para comunicações de rede de baixa latência com outras instâncias na mesma sub-rede. O EFA é

exposto usando Libfabric e pode ser usado por aplicativos que usam o padrão MPI (Messaging Passing Interface).

Para usar o EFA com AWS ParallelCluster, adicione a linha `enable_efa = true` à `[queue]`seção.

Para visualizar uma lista de tipos de instâncias do EC2 compatíveis com EFAs, consulte Tipos de instância compatíveis no Guia do usuário do Amazon EC2 para instâncias do Linux.

Para obter mais informações sobre a configuração `enable_efa`, consulte `enable_efa` na seção `[queue]`.

Um placement group de cluster deve ser usado para minimizar latências entre instâncias. Para obter mais informações, consulte `placement` e `placement_group`.

Para ter mais informações, consulte Elastic Fabric Adapter no Guia do usuário do Amazon EC2 e Scale HPC workloads with elastic fabric adapter and AWS ParallelCluster no AWS Open Source Blog.

> **Note**
>
> Por padrão, as distribuições do Ubuntu habilitam a proteção ptrace (rastreamento do processo). A partir do AWS ParallelCluster 2.6.0, a proteção ptrace foi desabilitada para que o Libfabric funcione corretamente. Para ter mais informações, consulte Desabilitar a proteção ptrace no Guia do usuário do Amazon EC2.

> **Note**
>
> O suporte EFA em instâncias Graviton2 baseadas em ARM foi adicionado na versão 2.10.1. AWS ParallelCluster

Intel Select Solutions

AWS ParallelCluster está disponível como uma solução Intel Select para simulação e modelagem. As configurações são verificadas para atender aos padrões estabelecidos pela Especificação da Plataforma Intel HPC, usam tipos específicos de instância Intel e são configuradas para usar a

interface de rede Elastic Fabric Adapter (EFA). AWS ParallelCluster é a primeira solução em nuvem a atender aos requisitos do programa Intel Select Solutions. Os tipos de instância compatíveis incluem `c5n.18xlarge`, `m5n.24xlarge` e `r5n.24xlarge`. Um exemplo de configuração compatível com o padrão Intel Select Solutions é fornecido abaixo.

Example Configuração do Intel Select Solutions

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

[aws]
aws_region_name = <Your Região da 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
```

Para obter mais informações sobre AWS ParallelCluster e a especificação da plataforma Intel HPC, consulteEspecificação da plataforma Intel HPC.

Habilitar o Intel MPI

O Intel MPI está disponível no AWS ParallelCluster AMIs. Para usar o Intel MPI, você deve concordar e aceitar os termos da Licença de Software Simplificada da Intel. Por padrão, o Open MPI é colocado no caminho. Para habilitar o Intel MPI em vez do Open MPI, o módulo Intel MPI deve ser carregado primeiro. Em seguida, você deve instalar a versão mais recente usando `module load intelmpi`. O nome exato do módulo muda com cada atualização. Para ver quais módulos estão disponíveis, execute `module avail`. A saída é a seguinte:

```
$ 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
```

Para ver quais módulos estão carregados, execute `module list`.

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

Para verificar se o Intel MPI está habilitado, execute `mpirun --version`.

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

Depois que o módulo Intel MPI é carregado, vários caminhos são alterados para usar as ferramentas Intel MPI. Para executar o código compilado pelas ferramentas Intel MPI, carregue o módulo Intel MPI primeiro.

> **Note**
>
> O Intel MPI não é compatível com instâncias baseadas em AWS Graviton.

> **Note**
>
> Antes da AWS ParallelCluster versão 2.5.0, o Intel MPI não estava disponível AWS ParallelCluster AMIs nas regiões da China (Pequim) e China (Ningxia).

Especificação da plataforma Intel HPC

AWS ParallelCluster é compatível com a especificação da plataforma Intel HPC. A especificação da plataforma Intel HPC fornece um conjunto de requisitos de computação, malha, memória, armazenamento e software de tal modo a ajudar a atingir um alto padrão de qualidade e compatibilidade com workloads de HPC. Para obter mais informações, consulte Especificação da plataforma Intel HPC e aplicativos compatíveis verificados com a especificação da plataforma Intel HPC.

Para ser compatível com a especificação da plataforma Intel HPC, os seguintes requisitos devem ser atendidos:

- O sistema operacional deve ser CentOS 7 (`base_os = centos7`).
- O tipo de instância para os nós de computação deve ter uma CPU Intel e pelo menos 64 GB de memória. Para a família `c5` de tipos de instância, isso significa que o tipo de instância deve ser pelo menos um `c5.9xlarge` (`compute_instance_type = c5.9xlarge`).
- O nó principal deve ter pelo menos 200 GB de armazenamento.
- O Contrato de licença de usuário final do Intel Parallel Studio deve ser aceito (`enable_intel_hpc_platform = true`).
- Cada nó de computação deve ter pelo menos 80 GB de armazenamento (`compute_root_volume_size = 80`).

O armazenamento pode ser local ou em uma rede (NFS compartilhado do nó principal, Amazon EBS ou FSx para Lustre) e pode ser compartilhado.

Bibliotecas de desempenho do Arm

A partir da AWS ParallelCluster versão 2.10.1, as Bibliotecas de Desempenho Arm estão disponíveis nos `ubuntu2004` valores AWS ParallelCluster AMIs for `alinux2centos8`,`ubuntu1804`, e para a `base_os` configuração. As bibliotecas de desempenho do Arm fornecem um padrão de bibliotecas matemáticas básicas otimizadas para aplicativos de computação de alto desempenho em processadores Arm. Para usar as Bibliotecas de Desempenho do Arm, você deve concordar e aceitar os termos das Bibliotecas de Desempenho do Arm (versão gratuita) - Contrato de Licença de Usuário Final. Para obter mais informações sobre bibliotecas de desempenho do Arm, consulte Bibliotecas de desempenho do Free Arm.

Para habilitar as Bibliotecas de Desempenho do Arm, você deve primeiro carregar o módulo Bibliotecas de Desempenho do Arm. `Armpl-21.0.0` precisa do GCC-9.3 como requisito, quando você carrega o módulo `armpl/21.0.0`, o módulo `gcc/9.3` também será carregado. O nome exato do módulo muda com cada atualização. Para ver quais módulos estão disponíveis, execute `module avail`. Em seguida, você deve instalar a versão mais recente usando `module load armpl`. A saída será conforme segue:

```
$ 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
```

Para carregar um módulo, execute `module load` *`modulename`*. Você pode adicionar isso ao script usado para executar `mpirun`.

```
$ 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
```

Para ver quais módulos estão carregados, execute `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
```

Para verificar se as bibliotecas de desempenho do Arm estão habilitadas, execute testes de exemplo.

```
$ 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
```

Depois que o módulo Arm Performance Libraries é carregado, vários caminhos são alterados para usar as ferramentas da Biblioteca de Desempenho do Arm. Para executar o código compilado pelas ferramentas Arm Performance Libraries, carregue primeiro o módulo Arm Performance Libraries.

> **Note**
>
> AWS ParallelCluster versões entre 2.10.1 e 2.10.4 usam. `armpl/20.2.1`

Conecte-se ao nó principal por meio do Amazon DCV

O Amazon DCV é uma tecnologia de visualização remota que permite que os usuários se conectem com segurança a aplicações 3D que usam muito processamento gráfico e que são hospedadas em um servidor remoto de alto desempenho. Para ter mais informações, consulte Amazon DCV.

O software Amazon DCV é instalado automaticamente no nó principal ao usar `base_os = alinux2`, `base_os = centos7`, `base_os = ubuntu1804` ou `base_os = ubuntu2004`.

Se o nó principal for uma instância ARM, o software Amazon DCV será instalado automaticamente nele ao usar `base_os = alinux2`, `base_os = centos7`, ou `base_os = ubuntu1804`.

Para habilitar o Amazon DCV no nó principal, `dcv_settings` deve conter o nome de uma seção `[dcv]` que tem `enable = master`, e `base_os` deve ser definido como `alinux2`, `centos7`, `ubuntu1804` ou `ubuntu2004`. Se o nó principal for uma instância ARM, `base_os` deve ser definida como `alinux2`, `centos7` ou `ubuntu1804`. Dessa forma, AWS ParallelCluster define o parâmetro de configuração do cluster `shared_dir` para a pasta de armazenamento do servidor DCV.

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

Para ter mais informações sobre parâmetros de configuração do Amazon DCV, consulte `dcv_settings`. Para se conectar à sessão do Amazon DCV, use o comando `pcluster dcv`.

> **Note**
>
> O suporte para Amazon DCV on `centos8` foi removido na AWS ParallelCluster versão 2.10.4. Support for Amazon DCV on `centos8` foi adicionado na AWS ParallelCluster versão 2.10.0. O suporte para Amazon DCV em instâncias AWS baseadas em Graviton foi adicionado na AWS ParallelCluster versão 2.9.0. Support para Amazon DCV `alinux2` e

`ubuntu1804` foi adicionado na AWS ParallelCluster versão 2.6.0. Support para Amazon DCV on `centos7` foi adicionado na AWS ParallelCluster versão 2.5.0.

> **Note**
>
> O Amazon DCV não é compatível com instâncias AWS baseadas em Graviton nas AWS ParallelCluster versões 2.8.0 e 2.8.1.

Certificado HTTPS do Amazon DCV

O Amazon DCV gera automaticamente um certificado autoassinado para proteger o tráfego entre o cliente do Amazon DCV e o servidor do Amazon DCV.

Para substituir o certificado autoassinado padrão do Amazon DCV por outro certificado, primeiro conecte-se ao nó principal. Depois, copie o certificado e a chave para a pasta `/etc/dcv` antes de executar o comando `pcluster dcv`.

Para ter mais informações, consulte Changing the TLS certificate no Guia do administrador do Amazon DCV.

Licenciamento do Amazon DCV

O servidor do Amazon DCV não requer um servidor de licença quando executado em instâncias do Amazon EC2. No entanto, o servidor do Amazon DCV deve se conectar periodicamente a um bucket do Amazon S3 para determinar se uma licença válida está disponível.

AWS ParallelCluster adiciona automaticamente as permissões necessárias ao`ParallelClusterInstancePolicy`. Ao usar uma política de instância do IAM personalizada, use as permissões descritas em Amazon DCV on Amazon EC2 no Guia do administrador do Amazon DCV.

Para obter dicas de solução de problemas, consulte Solução de problemas no Amazon DCV.

Usar o **`pcluster update`**

A partir da AWS ParallelCluster versão 2.8.0, `pcluster update` analisa as configurações usadas para criar o cluster atual e as configurações no arquivo de configuração em busca de problemas.

Se algum problema for descoberto, ele será relatado e as etapas a serem seguidas para corrigi-los serão exibidas. Por exemplo, se a configuração `compute_instance_type` for alterada para um tipo de instância diferente, a frota de computação deverá ser interrompida antes que uma atualização possa continuar. Esse problema é relatado quando é descoberto. Se nenhum problema de bloqueio for relatado, você será perguntado se deseja aplicar as alterações.

A documentação de cada configuração define a política de atualização para essa configuração.

Política de atualização: essas configurações podem ser alteradas durante uma atualização., Política de atualização: essa configuração pode ser alterada durante uma atualização.

Essas configurações podem ser alteradas, e o cluster pode ser atualizado usando o `pcluster update`.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

Essas configurações não podem ser alteradas se o cluster existente não tiver sido excluído. A alteração deve ser revertida ou o cluster deve ser excluído (usando `pcluster delete`) e, em seguida, um novo cluster é criado (usando`pcluster create`) no lugar do antigo cluster.

Política de atualização: essa configuração não é analisada durante uma atualização.

Essas configurações podem ser alteradas, e o cluster pode ser atualizado usando o `pcluster update`.

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

Essas configurações não podem ser alteradas enquanto a frota de computação existir. A alteração deve ser revertida ou a frota de computação deve ser interrompida (usando `pcluster stop`), atualizada (usando `pcluster update`) e, em seguida, uma nova frota de computação é criada (usando `pcluster start`).

Política de atualização: essa configuração não pode ser diminuída durante uma atualização.

Essas configurações podem ser alteradas, mas não podem ser diminuídas. Se essas configurações precisarem ser reduzidas, será necessário excluir o cluster (usando `pcluster delete`) e criar um novo cluster (usando `pcluster create`).

Política de atualização: reduzir o tamanho de uma fila abaixo do número atual de nós exige que a frota de computação seja interrompida primeiro.

Essas configurações podem ser alteradas, mas se a alteração reduzir o tamanho da fila abaixo do tamanho atual, a frota de computação deverá ser interrompida (usando `pcluster stop`),

atualizada (usando `pcluster update`), e depois uma nova frota de computação é criada (usando `pcluster start`).

Política de atualização: reduzir o número de nós estáticos na fila exige que a frota de computação seja interrompida primeiro.

Essas configurações podem ser alteradas, mas se a alteração reduzir o número de nós estáticos na fila abaixo do tamanho atual, a frota de computação deverá ser interrompida (usando `pcluster stop`), atualizada (usando `pcluster update`), e depois uma nova frota de computação é criada (usando `pcluster start`).

Política de atualização: se essa configuração for alterada, a atualização não será permitida. A atualização dessa configuração não pode ser forçada.

Essas configurações não podem ser alteradas se o cluster existente não tiver sido excluído. A alteração deve ser revertida ou o cluster deve ser excluído (usando `pcluster delete`) e, em seguida, um novo cluster é criado (usando`pcluster create`) no lugar do antigo cluster.

Política de atualização: se os sistemas de arquivos AWS ParallelCluster gerenciados do Amazon FSx for Lustre não estiverem especificados na configuração, essa configuração poderá ser alterada durante uma atualização.

Essa configuração pode ser alterada se `[cluster]fsx_settings`não for especificada ou se ambos `fsx_settings` e `fsx-fs-id`in `[fsx fs]`forem especificados para montar um sistema de arquivos externo existente FSx para o Lustre.

Este exemplo demonstra uma `pcluster update` com algumas alterações que bloqueiam a atualização.

```
$ 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.
```

Aplicação de patches de AMI e substituição de instâncias do EC2

Para garantir que todos os nós de computação do cluster lançados dinamicamente se comportem de maneira consistente, AWS ParallelCluster desative as atualizações automáticas do sistema operacional da instância do cluster. Além disso, um conjunto específico de AWS ParallelCluster AMIs é criado para cada versão AWS ParallelCluster e sua CLI associada. Esse conjunto específico AMIs permanece inalterado e é suportado apenas pela AWS ParallelCluster versão para a qual foram criados. AWS ParallelCluster AMIsas versões lançadas não são atualizadas.

No entanto, devido a problemas de segurança emergentes, talvez os clientes queiram adicionar patches a eles AMIs e depois atualizar seus clusters com a AMI corrigida. Isso se alinha ao Modelo de Responsabilidade Compartilhada do AWS ParallelCluster.

Para ver o conjunto específico de AWS ParallelCluster AMIs suportado pela versão da AWS ParallelCluster CLI que você está usando atualmente, execute:

```
$ pcluster version
```

Em seguida, visualize o amis.txt no AWS ParallelCluster GitHub repositório.

O nó AWS ParallelCluster principal é uma instância estática e você pode atualizá-lo manualmente. A reinicialização e a reinicialização do nó principal são totalmente suportadas a partir da AWS ParallelCluster versão 2.11, se o tipo de instância não tiver um armazenamento de instâncias. Para obter mais informações, consulte Tipos de instância com volumes de armazenamento de instância no Guia do usuário do Amazon EC2 para instâncias do Linux. Você não pode atualizar uma AMI para um cluster existente.

A reinicialização e reinicialização do nó principal com atualizações da AMI de instâncias de computação em cluster são totalmente suportadas a partir da AWS ParallelCluster versão 3.0.0. Considere fazer o upgrade para a versão mais recente para usar esses recursos.

Atualização ou substituição da instância do nó principal

Em algumas circunstâncias, talvez seja necessário reiniciar ou fazer reboot do nó principal. Por exemplo, isso é necessário quando você atualiza manualmente o sistema operacional ou quando há uma desativação programada da instância da AWS que impõe a reinicialização da instância do nó principal.

Se sua instância não tiver unidades efêmeras, você poderá interrompê-la e reiniciá-la a qualquer momento. No caso de uma desativação programada, ao iniciar a instância parada, ela será migrada para usar o novo hardware.

Da mesma forma, você pode parar e iniciar manualmente uma instância que não tenha armazenamentos de instâncias. Para esse caso e para outros casos de instâncias sem volumes efêmeros, continue para Interromper e iniciar o nó principal de um cluster.

Se sua instância tiver unidades efêmeras e estiver interrompida, os dados no armazenamento de instâncias serão perdidos. Você pode determinar se o tipo de instância usado para o nó principal tem armazenamento de instâncias na tabela encontrada em Volumes de armazenamento de instâncias.

As seções a seguir descrevem as limitações do uso de instâncias com volumes de armazenamento de instâncias.

Limitações de armazenamento de instância

As limitações no uso da AWS ParallelCluster versão 2.11 e dos tipos de instância com um armazenamento de instâncias são as seguintes:

- Quando unidades efêmeras não são criptografadas (o `encrypted_ephemeral`parâmetro está definido como `false` ou não definido), uma AWS ParallelCluster instância não consegue inicializar após a interrupção de uma instância. Isso ocorre porque as informações sobre dados efêmeros antigos e inexistentes são gravadas no `fstab` e o sistema operacional tenta montar um armazenamento inexistente.
- Quando unidades efêmeras são criptografadas (o `encrypted_ephemeral`parâmetro é definido como`true`), uma AWS ParallelCluster instância pode ser iniciada após uma parada, mas as novas unidades efêmeras não estão configuradas, montadas ou disponíveis.
- Quando unidades efêmeras são criptografadas, uma AWS ParallelCluster instância pode ser reinicializada, mas unidades efêmeras antigas (que sobrevivem à reinicialização da instância) não podem ser acessadas porque a chave de criptografia é criada na memória perdida com a reinicialização.

O único caso compatível é a reinicialização da instância, quando unidades efêmeras não são criptografadas. Isso ocorre porque a unidade sobrevive à reinicialização e é montada novamente devido à entrada escrita em `fstab`.

Soluções alternativas para limitações de armazenamento de instâncias

Primeiro, salve seus dados. Para verificar se você tem dados que precisam ser preservados, visualize o conteúdo na pasta `ephemeral_dir` (`/scratch` por padrão). Você pode transferir os dados para o volume raiz ou para os sistemas de armazenamento compartilhado conectados ao cluster, como Amazon FSx, Amazon EFS ou Amazon EBS. Observe que a transferência de dados para o armazenamento remoto pode incorrer em custos adicionais.

A causa raiz das limitações está na lógica AWS ParallelCluster usada para formatar e montar volumes de armazenamento de instâncias. A lógica adiciona uma entrada `/etc/fstab` ao formulário:

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

`${ephemeral_dir}` é o valor do parâmetro `ephemeral_dir` do arquivo de configuração do pcluster (o padrão é `/scratch`).

Essa linha é adicionada para casos em que, se ou quando um nó for reinicializado, os volumes do armazenamento de instâncias sejam remontados automaticamente. Isso é desejável porque os dados em unidades efêmeras persistem durante a reinicialização. No entanto, os dados nas unidades efêmeras não persistem durante um ciclo de partida ou parada. Isso significa que eles são formatados e montados sem dados.

O único caso compatível é a reinicialização da instância, quando unidades efêmeras não são criptografadas. Isso ocorre porque a unidade sobrevive à reinicialização e é montada novamente pois está escrita em `fstab`.

Para preservar os dados em todos os outros casos, você deve remover a entrada do volume lógico antes de interromper a instância. Por exemplo, remova `/dev/vg.01/lv_ephemeral` de `/etc/fstab` antes de interromper a instância. Depois de fazer isso, você inicia a instância sem montar os volumes temporários. No entanto, a montagem do armazenamento de instâncias novamente não estará disponível após a interrupção ou o início da instância.

Depois de salvar seus dados e remover a entrada `fstab`, siga para a próxima seção.

Interromper e iniciar o nó principal de um cluster

> **Note**
>
> A partir da AWS ParallelCluster versão 2.11, o head node stop and start só é suportado se o tipo de instância não tiver um armazenamento de instâncias.

1. Verifique se não há nenhum trabalho em execução no cluster.

   Ao usar um programador Slurm:

   - Se a opção `sbatch --no-requeue` não for especificada, os trabalhos em execução serão enfileirados novamente.
   - Se a opção `--no-requeue` for especificada, os trabalhos em execução falharão.

2. Solicite a interrupção da frota de computação em cluster:

```
$ 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. Espere até que o status da frota de computação seja STOPPED:

```
$ pcluster status cluster-name
...
ComputeFleetStatus: STOP_REQUESTED
$ pcluster status cluster-name
...
ComputeFleetStatus: STOPPED
```

4. Para atualizações manuais com a reinicialização do sistema operacional ou a reinicialização da instância, você pode usar o Console de gerenciamento da AWS ou AWS CLI. Veja a seguir um exemplo de uso da AWS CLI.

```
$ 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. Iniciar a frota de computação do cluster:

```
$ 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 Comandos CLI

`pclustere` `pcluster-config` são os comandos da AWS ParallelCluster CLI. Você usa `pcluster` para iniciar e gerenciar clusters de HPC no Nuvem AWS e `pcluster-config` atualizar sua configuração.

Para usar o `pcluster`, você tem um perfil do IAM com as permissões necessárias para executar.

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

Tópicos

- pcluster
- pcluster-config

`pcluster`

`pclusteré` o comando principal da AWS ParallelCluster CLI. Use a `pcluster` para executar e gerenciar clusters HPC na Nuvem AWS.

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

Argumentos

`pcluster` ***`command`***

Possíveis opções: `configure`, `create`, `createami`, `dcv`, `delete`, `instances`, `list`, `ssh`, `start`, `status`, `stop`, `update`, `version`

Subcomandos:

Tópicos

- 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

Inicia uma AWS ParallelCluster configuração. Para obter mais informações, consulte Configurando AWS ParallelCluster.

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

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster configure`.

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

Especifica o caminho completo do arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

Para obter mais informações, consulte Configurando AWS ParallelCluster.

`-r REGION, --region REGION`

Especifica o a Região da AWS ser usado. Se isso for especificado, a configuração ignora a Região da AWS detecção.

Para excluir os recursos de rede na VPC, você pode excluir a pilha CloudFormation de rede. O nome da pilha começa com"parallelclusternetworking-"e contém a hora da criação no formato "YYYYMMDDHHMMSS". Você pode listar as pilhas usando o comando list-stacks.

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

A pilha pode ser excluída usando o comando delete-stack.

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

A VPC que `pcluster configure` cria para você não é criada na pilha de CloudFormation rede. Você pode excluir essa VPC manualmente no console ou usando a AWS CLI.

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

pcluster create

Cria um cluster.

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

Argumentos posicionais

`cluster_name`

Define o nome do cluster. O nome da AWS CloudFormation pilha éparallelcluster-*`cluster_name`*.

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster create`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

`-r REGION, --region REGION`

Especifica o a Região da AWS ser usado. A ordem de prioridade usada para selecionar o Região da AWS para um novo cluster é a seguinte:

1. `-r`ou `--region` parâmetro para`pcluster create`.
2. Variável de ambiente `AWS_DEFAULT_REGION`.
3. `aws_region_name`configuração na `[aws]` seção do arquivo de AWS ParallelCluster configuração (o local padrão é`~/.parallelcluster/config`.) Esse é o local atualizado pelo comando `pcluster configure`.
4. `region`configuração na `[default]` seção do arquivo de AWS CLI configuração (`~/.aws/config`.)

`-nw, --nowait`

Indica para não aguardar eventos de pilha depois de executar um comando de pilha.

O padrão é `False`.

`-nr, --norollback`

Desabilita a reversão da pilha do quando ocorrer erro.

O padrão é `False`.

`-u TEMPLATE_URL, --template-url TEMPLATE_URL`

Especifica uma URL para o AWS CloudFormation modelo personalizado se ele foi usado quando criado.

`-t CLUSTER_TEMPLATE, --cluster-template CLUSTER_TEMPLATE`

Indica o modelo de cluster a ser usado.

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

Adiciona parâmetros adicionais à criação da pilha.

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

Especifica outras tags a serem adicionadas à pilha.

Quando o comando é chamado e começa a sondar o status dessa chamada, é seguro usar "Ctrl+C" para sair. Você pode voltar a visualizar o status atual chamando `pcluster status mycluster`.

Exemplos usando a AWS ParallelCluster versão 2.11.7:

```
$ 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) Cria uma AMI personalizada para usar com. 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 ]
```

Dependências necessárias

Além da AWS ParallelCluster CLI, a seguinte dependência é necessária para ser executada: `pcluster createami`

- Packer: faça download da versão mais recente em https://developer.hashicorp.com/packer/downloads.

> **Note**
>
> Antes da AWS ParallelCluster versão 2.8.0, era necessário usar o Berkshelf (instalado usando`gem install berkshelf`). `pcluster createami`

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster createami`.

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

Especifica a AMI básica a ser usada para criar a AWS ParallelCluster AMI.

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

Especifica o sistema operacional da AMI de base. As opções válidas são: `alinux2`, `ubuntu1804`, `ubuntu2004` e `centos7`.

> **Note**
>
> O suporte do sistema operacional muda em diferentes AWS ParallelCluster versões:
>
> - Support for `centos8` removido na AWS ParallelCluster versão 2.10.4.
> - O suporte para `centos8` foi adicionado e o suporte para `centos6` foi removido no AWS ParallelCluster 2.10.0.
> - Support para alinux2 foi adicionado na AWS ParallelCluster versão 2.6.0.
> - Support para ubuntu1804 foi adicionado na versão 2.5.0. AWS ParallelCluster

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

Especifica o tipo de instância a ser usado para criar a AMI.

O padrão é `t2.xlarge`.

> **Note**
>
> Support para o `--instance-type` argumento foi adicionado na AWS ParallelCluster versão 2.4.1.

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

Especifica o nome do prefixo da AMI AWS ParallelCluster resultante.

O padrão é `custom-ami-`.

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

Especifica o livro de receitas a ser usado para criar a AMI AWS ParallelCluster .

`--post-install` *`POST_INSTALL_SCRIPT`*

Especifica o caminho para o script de pós-instalação. Os caminhos devem usar um esquema de URL `s3://`, `https://`, ou `file://`. Os exemplos incluem:

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

> **Note**
>
> Support para o `--post-install` argumento foi adicionado na AWS ParallelCluster versão 2.10.0.

`--no-public-ip`

Não associe um endereço IP público à instância usada para criar a AMI. Por padrão, um endereço IP público está associado à instância.

> **Note**
>
> Support para o `--no-public-ip` argumento foi adicionado na AWS ParallelCluster versão 2.5.0.

`-c CONFIG_FILE, --config CONFIG_FILE`

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

`-t CLUSTER_TEMPLATE, --cluster-template CLUSTER_TEMPLATE`

Especifica a seção [cluster] do a *CONFIG_FILE* ser usada para recuperar as configurações de VPC e sub-rede.

> **Note**
>
> Support para o `--cluster-template` argumento foi adicionado na AWS ParallelCluster versão 2.4.0.

`--vpc-id VPC_ID`

Especifica o ID da VPC a ser usada para criar a AWS ParallelCluster AMI.

> **Note**
>
> Support para o `--vpc-id` argumento foi adicionado na AWS ParallelCluster versão 2.5.0.

`--subnet-id SUBNET_ID`

Especifica o ID da sub-rede a ser usada para criar a AMI AWS ParallelCluster .

> **Note**
>
> Support para o `--vpc-id` argumento foi adicionado na AWS ParallelCluster versão 2.5.0.

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

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

pcluster dcv

Interage com o servidor Amazon DCV em execução no nó principal.

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

pcluster dcv *command*

Possíveis opções: connect

> **Note**
>
> Alterações no suporte do sistema operacional para o `pcluster dcv` comando em diferentes AWS ParallelCluster versões:
>
> - O suporte para o comando `pcluster dcv` no `centos8` foi adicionado ao AWS ParallelCluster versão 2.10.0.
> - Support para o `pcluster dcv` comando em instâncias AWS baseadas em Graviton foi adicionado na AWS ParallelCluster versão 2.9.0.
> - O suporte para o comando `pcluster dcv` no `ubuntu1804` foi adicionado ao AWS ParallelCluster versão 2.6.0.
> - O suporte para o comando `pcluster dcv` no `centos7` foi adicionado ao AWS ParallelCluster versão 2.5.0.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster dcv`.

Subcomandos

pcluster dcv connect

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

> **Important**
>
> O URL expira 30 segundos após ser emitido. Se a conexão não for estabelecida antes de o URL expirar, execute o `pcluster dcv connect` novamente para gerar um novo URL.

Argumentos posicionais

cluster_name

Especifica o nome do cluster ao qual conectar-se.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster dcv connect`.

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

Caminho da chave SSH a ser usada para a conexão.

A chave deve ser aquela especificada no momento da criação do cluster no parâmetro de configuração `key_name`. Este argumento é opcional, mas, se não for especificado, a chave deverá estar disponível por padrão para o cliente SSH. Por exemplo, adicione-o ao `ssh-agent` com `ssh-add`.

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

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

-s, --show-url

Exibe um URL único para conectar-se à sessão do Amazon DCV. O navegador padrão não é aberto quando essa opção é especificada.

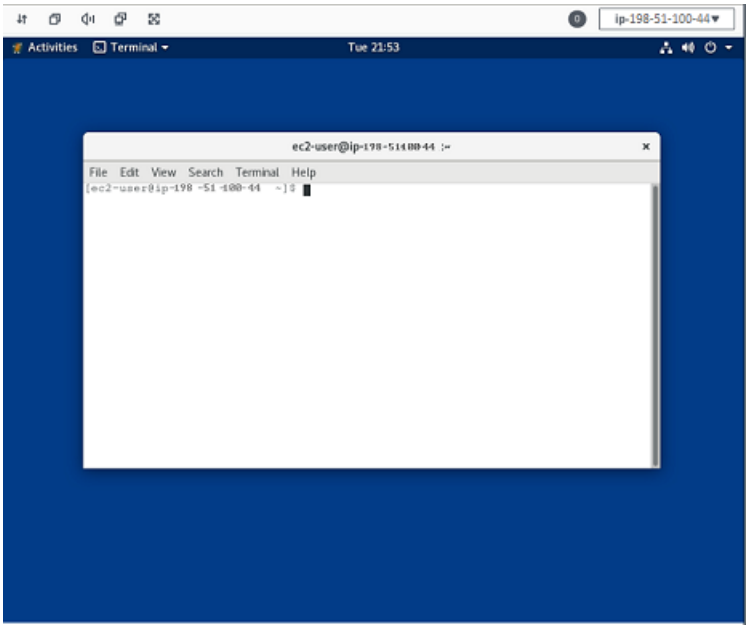
> **Note**
>
> Support para o `--show-url` argumento foi adicionado na AWS ParallelCluster versão 2.5.1.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

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

Abre o navegador padrão para se conectar à sessão do Amazon DCV em execução no nó principal.

Uma nova sessão do Amazon DCV será criada se uma ainda não tiver sido iniciada.



pcluster delete

Exclui um cluster.

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

Argumentos posicionais

cluster_name

Especifica o nome do cluster a ser excluído.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster delete`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

`--keep-logs`

Mantenha os dados do CloudWatch Logs depois de excluir o cluster. O grupo de logs permanece até que você o exclua manualmente, mas os eventos de log expirarão com base na configuração `retention_days`. A configuração padrão será de 14 dias.

> **Note**
>
> O suporte para o argumento **`--keep-logs`** foi adicionado ao AWS ParallelCluster versão 2.6.0.

`-r REGION, --region REGION`

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

Quando o comando é chamado e começa a sondar o status dessa chamada, é seguro usar "Ctrl+C" para sair. Você pode voltar a visualizar o status atual chamando `pcluster status mycluster`.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

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

Para excluir os recursos de rede na VPC, você pode excluir a pilha CloudFormation de rede. O nome da pilha começa com”parallelclusternetworking-“e contém a hora da criação no formato “YYYYMMDDHHMMSS”. Você pode listar as pilhas usando o comando list-stacks.

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

```
   grep -e "parallelclusternetworking-"
    "parallelclusternetworking-pubpriv-20191029205804"
```

A pilha pode ser excluída usando o comando delete-stack.

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

A VPC que `pcluster configure` cria para você não é criada na pilha de CloudFormation rede. Você pode excluir essa VPC manualmente no console ou usando a AWS CLI.

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

pcluster instances

Exibe uma lista de todas as instâncias de um cluster.

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

Argumentos posicionais

cluster_name

Exibe as instâncias do cluster com o nome fornecido.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster instances`.

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

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

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

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

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

pcluster list

Exibe uma lista das pilhas associadas a. AWS ParallelCluster

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

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster list`.

--color

Exibe o status do cluster em cores.

O padrão é `False`.

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

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `c`.

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

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

Lista o nome de todas as AWS CloudFormation pilhas nomeadas`parallelcluster-*`.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

```
$ 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

Executa um comando `ssh` com o nome de usuário e o endereço IP do cluster pré-preenchidos. Os argumentos arbitrários são anexados ao final do comando `ssh`. Esse comando pode ser personalizado na seção de aliases do arquivo de configuração.

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

Argumentos posicionais

cluster_name

Especifica o nome do cluster ao qual conectar-se.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster ssh`.

-d, --dryrun

Imprime o comando que seria executado e sai.

O padrão é `False`.

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

Especifica o a Região da AWS ser usado. O padrão é a região especificada usando o comando `pcluster configure`.

Exemplos usando a AWS ParallelCluster versão 2.11.7:

```
$ 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
```

Executa um comando `ssh` com o nome de usuário e o endereço IP do cluster pré-preenchidos:

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

O comando `ssh` é definido no arquivo de configuração global em Seção `[aliases]`. Ele pode ser personalizado da maneira indicada a seguir.

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

Variáveis substituídas:

`CFN_USER`

O nome de usuário para o `base_os` que está selecionado.

`MASTER_IP`

O endereço IP do nó principal.

`ARGS`

Os argumentos opcionais a serem passados para o comando `ssh`.

pcluster start

Inicia a frota de computação para um cluster que foi interrompido.

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

Argumentos posicionais

cluster_name

Inicia a frota de computação do nome de cluster fornecido.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster start`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

`-r REGION, --region REGION`

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

```
$ 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
```

Esse comando define os parâmetros do Grupo do Auto Scaling para um dos seguintes:

- Os valores da configuração inicial (`max_queue_size` e `initial_queue_size`) a partir do modelo que foi usado para criar o cluster.
- Os valores de configuração que foram usados para atualizar o cluster desde que foi criado pela primeira vez.

pcluster status

Recebe o status atual do cluster.

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

Argumentos posicionais

`cluster_name`

Mostra o status do cluster com o nome fornecido.

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster status`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

`-r REGION, --region REGION`

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

`-nw, --nowait`

Indica para não aguardar eventos de pilha depois de processar um comando de pilha.

O padrão é `False`.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

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

pcluster stop

Interrompe a frota de computação, deixando o nó principal em execução.

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

Argumentos posicionais

`cluster_name`

Interrompe a frota de computação do nome de cluster fornecido.

Exemplo usando a AWS ParallelCluster versão 2.11.7:

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster stop`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

`-r REGION, --region REGION`

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

```
$ 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
```

Define os parâmetros do grupo Auto Scaling como min/max/desired = 0/0/0 e encerra a frota de computação. O cabeçalho permanece em execução. Para encerrar todos os EC2 recursos e evitar EC2 cobranças, considere excluir o cluster.

`pcluster update`

Analisa o arquivo de configuração para determinar se o cluster pode ser atualizado com segurança. Se a análise determinar que o cluster pode ser atualizado, precisará confirmar a alteração. Se a análise mostrar que o cluster não pode ser atualizado, as configurações que são a origem dos conflitos são enumeradas com detalhes. Para obter mais informações, consulte Usar o `pcluster update`.

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

Argumentos posicionais

cluster_name

Especifica o nome do cluster a ser atualizado.

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster update`.

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

Especifica o arquivo de configuração alternativo a ser usado.

O padrão é `~/.parallelcluster/config`.

--force

Habilita uma atualização mesmo se uma ou mais configurações tiverem uma alteração de bloqueio ou se for necessária uma ação pendente (como interromper a frota de computação) antes que a atualização possa continuar. Isso não deve ser combinado com o argumento `--yes`.

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

Especifica o a Região da AWS ser usado. O padrão é o Região da AWS especificado usando o `pcluster configure` comando.

-nr, --norollback

Desativa a reversão AWS CloudFormation da pilha em caso de erro.

O padrão é `False`.

-nw, --nowait

Indica para não aguardar eventos de pilha depois de processar um comando de pilha.

O padrão é `False`.

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

Especifica a seção do modelo de cluster a ser usada.

`-p EXTRA_PARAMETERS, --extra-parameters EXTRA_PARAMETERS`

Adiciona parâmetros adicionais a uma atualização de pilha.

`-rd, --reset-desired`

Redefine a capacidade atual de um Grupo do Auto Scaling para os valores da configuração inicial.

O padrão é `False`.

`--yes`

Pressupõe automaticamente que a resposta a todos os avisos é sim. Isso não deve ser combinado com o argumento `--force`.

```
$ 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
```

Quando o comando é chamado e começa a sondar o status dessa chamada, é seguro usar "Ctrl+C" para sair. Você pode voltar a visualizar o status atual chamando `pcluster status mycluster`.

`pcluster version`

Exibe a AWS ParallelCluster versão.

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

Para sinalizadores específicos de comando, execute: `pcluster [command] --help`.

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster version`.

Quando o comando é chamado e começa a sondar o status dessa chamada, é seguro usar "Ctrl+C" para sair. Você pode voltar a visualizar o status atual chamando `pcluster status mycluster`.

```
$ pcluster version
2.11.7
```

pcluster-config

Atualiza o arquivo AWS ParallelCluster de configuração.

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

Para sinalizadores específicos de comando, execute: `pcluster-config [command] -h`.

Argumentos nomeados

`-h, --help`

Mostra o texto de ajuda para `pcluster-config`.

> **Note**
>
> O `pcluster-config` comando foi adicionado na AWS ParallelCluster versão 2.9.0.

Subcomandos

`pcluster-config convert`

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

```
                        [ -o OUTPUT_FILE ]
```

Argumentos nomeados

-h, --help

Mostra o texto de ajuda para `pcluster-config convert`.

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

Especifica o caminho do arquivo de configuração a ser lido.

O padrão é `~/.parallelcluster/config`.

Para obter mais informações, consulte Configurando AWS ParallelCluster.

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

Indica o Seção [cluster] a ser usado. Se esse argumento não for especificado, `pcluster-config convert` usará a configuração `cluster_template` no Seção [global]. Se isso não for especificado, a seção `[cluster default]` será usada.

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

Especifica o caminho do arquivo de configuração convertido a ser gravado. Por padrão, a saída é gravada em `STDOUT`.

Exemplo:

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

Converte a configuração do cluster especificada na seção `[cluster alpha]` de `~/.parallelcluster/config`, gravando o arquivo de configuração convertido em `~/.parallelcluster/multiinstance`.

Configuração

Por padrão, AWS ParallelCluster usa o `~/.parallelcluster/config` arquivo para todos os parâmetros de configuração. Você pode especificar um arquivo de configuração personalizado usando a opção da linha de comando `-c` ou `--config`, ou a variável de ambiente `AWS_PCLUSTER_CONFIG_FILE`.

Um exemplo de arquivo de configuração é instalado AWS ParallelCluster no diretório Python em. `site-packages/aws-parallelcluster/examples/config` O arquivo de configuração de exemplo também está disponível em GitHub, emhttps://github.com/aws/aws-parallelcluster/blob/v2.11.9/cli/src/pcluster/examples/config.

Versão AWS ParallelCluster 2 atual: 2.11.9.

Tópicos

Layout

Uma AWS ParallelCluster configuração é definida em várias seções.

As seções a seguir são obrigatórias: `[global]` seção e `[aws]` seção.

Também é preciso incluir no mínimo uma `[cluster]` seção e uma sessão `[vpc]`.

Uma seção começa com o nome da seção entre colchetes, seguido por parâmetros e pela configuração.

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

Seção **[global]**

Especifica opções de configuração globais relacionadas a `pcluster`.

```
[global]
```

Tópicos

`cluster_template`

Define o nome da seção do `cluster` que será usada por padrão para o cluster. Para obter mais informações sobre seções de `cluster`, consulte seção do `[cluster]`. O nome do cluster deve começar com uma letra, conter, no máximo, 60 caracteres e conter apenas letras, números e hifens (-).

Por exemplo, a configuração a seguir especifica que a seção que começa com `[cluster default]` é usada por padrão.

```
cluster_template = default
```

Política de atualização: essa configuração não é analisada durante uma atualização.

`update_check`

(Opcional) Verifica se há atualizações para o `pcluster`.

O valor padrão é `true`.

```
update_check = true
```

Política de atualização: essa configuração não é analisada durante uma atualização.

`sanity_check`

(Opcional) Tenta validar a configuração dos recursos que estão definidos nos parâmetros do cluster.

O valor padrão é `true`.

> **Warning**
>
> Se `sanity_check` estiver definido como `false`, verificações importantes serão ignoradas. Isso pode fazer com que sua configuração não funcione conforme o esperado.

```
sanity_check = true
```

> **Note**
>
> Antes da AWS ParallelCluster versão 2.5.0, o `sanity_check` padrão era. `false`

Política de atualização: essa configuração não é analisada durante uma atualização.

Seção `[aws]`

(Opcional) Usado para selecionar Região da AWS o.

A criação do cluster usa essa ordem de prioridade para selecionar o Região da AWS para um novo cluster:

1. `-r`ou `--region` parâmetro para`pcluster create`.
2. Variável de ambiente `AWS_DEFAULT_REGION`.
3. `aws_region_name`configuração na `[aws]` seção do arquivo de AWS ParallelCluster configuração (o local padrão é`~/.parallelcluster/config`.) Esse é o local atualizado pelo comando `pcluster configure`.
4. `region`configuração na `[default]` seção do arquivo de AWS CLI configuração (`~/.aws/config`.)

> **Note**
>
> Antes da AWS ParallelCluster versão 2.10.0, essas configurações eram necessárias e aplicadas a todos os clusters.

Para armazenar credenciais, você pode usar o ambiente, as funções do IAM para a Amazon ou o EC2 AWS CLI, em vez de salvar as credenciais no AWS ParallelCluster arquivo de configuração.

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

Política de atualização: essa configuração não é analisada durante uma atualização.

Seção `[aliases]`

Especifica aliases e permite personalizar o comando `ssh`.

Observe as seguintes configurações padrão:

- `CFN_USER` está definido como o nome de usuário padrão do SO
- `MASTER_IP` é definido como o endereço IP do nó principal
- `ARGS` é definido para quaisquer argumentos que o usuário forneça após *`pcluster ssh cluster_name`*

```
[aliases]
# This is the aliases section, you can configure
# ssh alias here
```

```
ssh = ssh {CFN_USER}@{MASTER_IP} {ARGS}
```

Política de atualização: essa configuração não é analisada durante uma atualização.

Seção `[cluster]`

Define um modelo de cluster que pode ser usado para criar um cluster. Um arquivo de configuração pode conter várias seções `[cluster]`.

O mesmo modelo de cluster pode ser usado para criar vários clusters.

O formato é `[cluster` *`cluster-template-name`*`]`. A seção `[cluster]` nomeada pela configuração `cluster_template` na seção `[global]` é usada por padrão, mas pode ser substituída na linha de comando `pcluster`.

*`cluster-template-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[cluster default]
```

Tópicos

- 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

(Opcional) Define um AWS CloudFormation modelo adicional a ser executado junto com o cluster. Esse modelo adicional é usado para a criação de recursos que existem fora do cluster, mas que fazem parte do ciclo de vida do cluster.

Esse valor deve ser um URL HTTP para um modelo público, com todos os parâmetros fornecidos.

Não há valor padrão.

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

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

additional_iam_policies

(Opcional) Especifica uma lista de nomes de recursos da Amazon (ARNs) de políticas do IAM para o Amazon EC2. Essa lista é anexada à função raiz usada no cluster, além das permissões exigidas pelo AWS ParallelCluster separadas por vírgulas. O nome da política do IAM e seu ARN são diferentes. Os nomes não podem ser usados como um argumento para `additional_iam_policies`.

Se sua intenção é adicionar políticas extras às configurações padrão dos nós do cluster, recomendamos que você transmita as políticas adicionais personalizadas do IAM com a configuração `additional_iam_policies` em vez de usar as configurações `ec2_iam_role` para incluir suas políticas de EC2 específicas. Isso ocorre porque `additional_iam_policies` são adicionados às permissões padrão AWS ParallelCluster exigidas. Um `ec2_iam_role` existente

deve incluir todas as permissões necessárias. No entanto, como as permissões necessárias geralmente mudam de uma versão para outra à medida que os recursos são adicionados, um `ec2_iam_role` pode se tornar obsoleto.

Não há valor padrão.

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

> **Note**
>
> O suporte para `additional_iam_policies` foi adicionado ao AWS ParallelCluster versão 2.5.0.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

base_os

(Obrigatório) Especifica qual tipo de SO é usado no cluster.

As opções disponíveis são:

- `alinux2`
- `centos7`
- `ubuntu1804`
- `ubuntu2004`

> **Note**
>
> Somente para instâncias AWS baseadas em Graviton `alinux2ubuntu1804`, ou `ubuntu2004` são compatíveis.

> **Note**
>
> Support for `centos8` removido na AWS ParallelCluster versão 2.11.4. O suporte para `ubuntu2004` foi adicionado e o suporte para `alinux` e `ubuntu1604` foi removido na versão

2.11.0 do AWS ParallelCluster . O suporte para `centos8` foi adicionado e o suporte para `centos6` foi removido na AWS ParallelCluster versão 2.10.0. O suporte para `alinux2` foi adicionado ao AWS ParallelCluster versão 2.6.0. O suporte para `ubuntu1804` foi adicionado e o suporte para `ubuntu1404` foi removido no AWS ParallelCluster versão 2.5.0.

Além dos específicos Regiões da AWS mencionados na tabela a seguir, que não são compatíveis`centos7`. Todas as outras regiões AWS comerciais oferecem suporte a todos os sistemas operacionais a seguir.

Partição (Regiões da AWS)	**`alinux2`**	**`centos7`**	**`ubuntu1804`** e **`ubuntu2004`**
Comercial (tudo Regiões da AWS não mencionado especificamente)	Verdadeiro	Verdadeiro	Verdadeiro
AWS GovCloud (Leste dos EUA) (`us-gov-east-1`)	Verdadeiro	Falso	Verdadeiro
AWS GovCloud (Oeste dos EUA) (`us-gov-west-1`)	Verdadeiro	Falso	Verdadeiro
China (Pequim) (`cn-north-1`)	Verdadeiro	Falso	Verdadeiro
China (Ningxia) (`cn-northwest-1`)	Verdadeiro	Falso	Verdadeiro

> **Note**
>
> O parâmetro `base_os` também determina o nome de usuário que é usado para fazer login no cluster.

- `centos7`: `centos`
- `ubuntu1804` e `ubuntu2004`: `ubuntu`
- `alinux2`: `ec2-user`

> **Note**
>
> Antes da AWS ParallelCluster versão 2.7.0, o `base_os` parâmetro era opcional e o padrão era. `alinux` A partir do AWS ParallelCluster versão 2.7.0, o parâmetro `base_os` é obrigatório.

> **Note**
>
> Se o parâmetro `scheduler` for `awsbatch`, somente `alinux2` será aceito.

```
base_os = alinux2
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

cluster_resource_bucket

(Opcional) Especifica o nome do bucket do Amazon S3 usado para hospedar recursos que são gerados quando o cluster é criado. O bucket deve ter o versionamento ativado. Para obter mais informações, consulte Usando versionamento no Guia do usuário do Amazon Simple Storage Service. Esse bucket pode ser usado para vários clusters. O bucket do deve estar na mesma região que o cluster.

Se esse parâmetro não for especificado, um novo bucket será criado quando o cluster for criado. O novo bucket tem o nome de `parallelcluster-`*`random_string`*. Nesse nome, *`random_string`* é uma sequência aleatória de caracteres alfanuméricos. Todos os recursos do cluster são armazenados nesse bucket em um caminho com o formulário*`bucket_name`*`/`*`resource_directory`*. `resource_directory`tem o formulário*`stack_name`*`-`*`random_string`*, onde *`stack_name`* está o nome de uma das CloudFormation pilhas usadas por AWS ParallelCluster. O valor de *`bucket_name`* pode ser encontrado no `ResourcesS3Bucket` valor na saída da `parallelcluster-`*`clustername`* pilha. O valor de *`resource_directory`* pode ser encontrado no valor da `ArtifactS3RootDirectory` saída da mesma pilha.

O valor padrão é `parallelcluster-`*`random_string`*.

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

> **Note**
>
> Support for `cluster_resource_bucket` adicionado na AWS ParallelCluster versão 2.10.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida. A atualização dessa configuração não pode ser forçada.

`cluster_type`

(Opcional) Define o tipo de cluster a ser executado. Se a configuração `queue_settings` estiver definida, essa configuração deverá ser substituída pelas configurações `compute_type` nas seções`[queue]`.

As opções válidas são `ondemand` e `spot`.

O valor padrão é `ondemand`.

Para obter mais informações sobre instâncias spot, consulte Trabalho com Instâncias spot.

> **Note**
>
> O uso de Instâncias spot exige que a função `AWSServiceRoleForEC2Spot` vinculada ao serviço exista na sua conta. Para criar essa função na sua conta usando o AWS CLI, execute o seguinte comando:
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> Para ter mais informações, consulte Função vinculada ao serviço para solicitações de instâncias spot no Guia do usuário do Amazon EC2.

```
cluster_type = ondemand
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

compute_instance_type

(Opcional) Define o tipo de instância do Amazon EC2 que é usado para os nós de computação em cluster. A arquitetura do tipo de instância deve ser a mesma usada para a configuração master_instance_type. Se a configuração queue_settings estiver definida, essa configuração deverá ser substituída pelas configurações instance_type nas seções[compute_resource].

Se você estiver usando o awsbatch agendador, consulte a criação de ambientes de computação na AWS Batch interface do usuário para ver uma lista dos tipos de instância compatíveis.

O padrão é t2.micro, optimal quando o programador for awsbatch.

```
compute_instance_type = t2.micro
```

> **Note**
>
> Support para instâncias AWS baseadas em Graviton (incluindo C6g instâncias A1 e) foi adicionado na AWS ParallelCluster versão 2.8.0.

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

compute_root_volume_size

(Opcional) Especifica o tamanho do volume ComputeFleet raiz em gibibytes (GiB). A AMI deve oferecer suporte para growroot.

O valor padrão é 35.

> **Note**
>
> Para AWS ParallelCluster versões entre 2.5.0 e 2.10.4, o padrão era 25. Antes da AWS ParallelCluster versão 2.5.0, o padrão era 20.

```
compute_root_volume_size = 35
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

custom_ami

(Opcional) Especifica a ID de uma AMI personalizada a ser usada nos nós principais e de computação, em vez da publicação padrão. AMIs Para acessar mais informações, consulte Modificar uma AMI do ou Crie uma AWS ParallelCluster AMI personalizada.

Não há valor padrão.

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

Se a AMI personalizada exigir permissões adicionais para seu lançamento, essas permissões deverão ser adicionadas às políticas do usuário e do nó principal.

Por exemplo, se uma AMI personalizada tiver um snapshot criptografado associado a ela, as seguintes políticas adicionais serão necessárias nas políticas do usuário e do nó principal:

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey",
                "kms:ReEncrypt*",
                "kms:CreateGrant",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:us-east-1:111122223333:key/<AWS_KMS_KEY_ID>"
            ]
        }
    ]
}
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

cw_log_settings

(Opcional) Identifica a `[cw_log]` seção com a configuração de CloudWatch registros. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para mais informações, consulte a seção `[cw_log]`, CloudWatch Painel da Amazon, e Integração com Amazon CloudWatch Logs.

Por exemplo, a configuração a seguir especifica que a seção que começa `[cw_log custom-cw]` é usada para a configuração de CloudWatch registros.

```
cw_log_settings = custom-cw
```

> **Note**
>
> Support for `cw_log_settings` adicionado na AWS ParallelCluster versão 2.6.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

dashboard_settings

(Opcional) Identifica a `[dashboard]` seção com a configuração do CloudWatch painel. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a seção `[dashboard]`.

Por exemplo, a configuração a seguir especifica que a seção que começa `[dashboard custom-dashboard` é usada para a configuração do CloudWatch painel.

```
dashboard_settings = custom-dashboard
```

> **Note**
>
> Support for `dashboard_settings` adicionado na AWS ParallelCluster versão 2.10.0.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

dcv_settings

(Opcional) Identifica a seção `[dcv]` com a configuração do Amazon DCV. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a seção `[dcv]`.

Por exemplo, a configuração a seguir especifica que a seção que inicia `[dcv custom-dcv]` é usada para a configuração do Amazon DCV.

```
dcv_settings = custom-dcv
```

> **Note**
>
> Em instâncias AWS baseadas em Graviton, o Amazon DCV só é compatível com. `alinux2`

> **Note**
>
> Support for `dcv_settings` adicionado na AWS ParallelCluster versão 2.5.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

desired_vcpus

(Opcional) Especifica o número desejado de v CPUs no ambiente computacional. Usador apenas se o programador for `awsbatch`.

O valor padrão é `4`.

```
desired_vcpus = 4
```

Política de atualização: essa configuração não é analisada durante uma atualização.

disable_cluster_dns

(Opcional) Especifica se as entradas de DNS do cluster não devem ser criadas. Por padrão, AWS ParallelCluster cria uma zona hospedada do Route 53. Se `disable_cluster_dns` estiver definido como `true`, a zona hospedada não será criada.

O valor padrão é `false`.

```
disable_cluster_dns = true
```

> **Warning**
>
> É necessário um sistema de resolução de nomes para que o cluster funcione adequadamente. Se `disable_cluster_dns` estiver definido como `true`, um sistema adicional de resolução de nomes também deverá ser fornecido.

> **Important**
>
> `disable_cluster_dns`= só `true` é suportado se a configuração `queue_settings` for especificada.

> **Note**
>
> Support for `disable_cluster_dns` adicionado na AWS ParallelCluster versão 2.9.1.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

disable_hyperthreading

(Opcional) Desabilita o hyperthreading nos nós principais e de computação. Nem todos os tipos de instância podem desabilitar o hyperthreading. Para ver quais tipos de instância permitem desabilitar o hyperthreading, consulte a seção sobre núcleos e threads de CPU para cada núcleo de CPU por tipo de instância no Guia do usuário do Amazon EC2. Se a configuração `queue_settings` for definida, essa configuração poderá ser definida ou as configurações `disable_hyperthreading` nas seções `[queue]` poderão ser definidas.

O valor padrão é `false`.

```
disable_hyperthreading = true
```

> **Note**
>
> `disable_hyperthreading` afeta apenas o nó principal quando `scheduler = awsbatch`.

> **Note**
>
> O suporte para `disable_hyperthreading` foi adicionado ao AWS ParallelCluster versão 2.5.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ebs_settings

(Opcional) Identifica as seções `[ebs]` com os volumes do Amazon EBS que estão montados no nó principal. Ao usar vários volumes do Amazon EBS, insira esses parâmetros em uma lista com cada um separado por uma vírgula. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Há suporte para até cinco (5) volumes adicionais do Amazon EBS.

Para obter mais informações, consulte a seção `[ebs]`.

Por exemplo, a seguinte configuração especifica que as seções que começam com `[ebs custom1]` e `[ebs custom2]` são usados para os volumes do Amazon EBS.

```
ebs_settings = custom1, custom2
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ec2_iam_role

(Opcional) Define o nome de um perfil do IAM existente para o Amazon EC2 que é anexada a todas as instâncias do cluster. Um nome de função do IAM e seu Amazon Resource Name (ARN) são diferentes. ARNs não pode ser usado como argumento para`ec2_iam_role`.

Se essa opção for especificada, a configuração `additional_iam_policies` será ignorada. Se sua intenção é adicionar políticas extras às configurações padrão dos nós do cluster, recomendamos que você transmita as políticas adicionais personalizadas do IAM com a configuração `additional_iam_policies` em vez de usar as configurações `ec2_iam_role`.

Se essa opção não for especificada, a função padrão AWS ParallelCluster do IAM para o Amazon EC2 será usada. Para obter mais informações, consulte AWS Identity and Access Management funções em AWS ParallelCluster.

Não há valor padrão.

```
ec2_iam_role = ParallelClusterInstanceRole
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

efs_settings

(Opcional) Especifica as configurações relacionadas ao sistema de arquivos do Amazon EFS. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a seção `[efs]`.

Por exemplo, a seguinte configuração especifica que a seção que começa com `[efs customfs]` é usada para a configuração do sistema de arquivos do Amazon EFS.

```
efs_settings = customfs
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

enable_efa

(Opcional) Se estiver presente, especificará que o Elastic Fabric Adapter (EFA) está habilitado para os nós de computação. Para visualizar uma lista de tipos de instâncias do EC2 compatíveis com EFAs, consulte Tipos de instância compatíveis no Guia do usuário do Amazon EC2 para

instâncias do Linux. Para obter mais informações, consulte Elastic Fabric Adapter. Se a configuração queue_settings for definida, essa configuração poderá ser definida ou as configurações enable_efa na seção [queue] poderão ser definidas. Um placement group de cluster deve ser usado para minimizar latências entre instâncias. Para obter mais informações, consulte placement e placement_group.

```
enable_efa = compute
```

> **Note**
>
> O suporte para EFA em instâncias Graviton2 baseadas em ARM foi adicionado na versão 2.10.1. AWS ParallelCluster

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

enable_efa_gdr

(Opcional) A partir da AWS ParallelCluster versão 2.11.3, essa configuração não tem efeito. O suporte do Elastic Fabric Adapter (EFA) para GPUDirect RDMA (acesso direto remoto à memória) está sempre ativado se for suportado pelo tipo de instância e pelo sistema operacional.

> **Note**
>
> AWS ParallelCluster versão 2.10.0 a 2.11.2: Se`compute`, especifica que o suporte do Elastic Fabric Adapter (EFA) GPUDirect para RDMA (acesso direto remoto à memória) está habilitado para os nós de computação. Definir essa configuração como `compute` requer que a configuração enable_efa seja definida como `compute`. O suporte do EFA para GPUDirect RDMA é suportado por tipos de instância específicos (`p4d.24xlarge`) em sistemas operacionais específicos (base_os`alinux2`, `centos7ubuntu1804`, o`ubuntu2004`). Se a configuração queue_settings for definida, essa configuração poderá ser definida ou as configurações enable_efa_gdr nas seções [queue] poderão ser definidas. Um placement group de cluster deve ser usado para minimizar latências entre instâncias. Para obter mais informações, consulte placement e placement_group.

```
enable_efa_gdr = compute
```

> **Note**
>
> Support for `enable_efa_gdr` adicionado na AWS ParallelCluster versão 2.10.0.

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

`enable_intel_hpc_platform`

(Opcional) Se estiver presente, indica que o Contrato de licença de usuário final do Intel Parallel Studio é aceito. Isso faz com que o Intel Parallel Studio seja instalado no nó principal e compartilhado com os nós de computação. Isso adiciona vários minutos ao tempo que o nó principal leva para ser inicializado. A configuração `enable_intel_hpc_platform` só é compatível com CentOS 7 (`base_os = centos7`).

O valor padrão é `false`.

```
enable_intel_hpc_platform = true
```

> **Note**
>
> O `enable_intel_hpc_platform` parâmetro não é compatível com instâncias AWS baseadas em Graviton.

> **Note**
>
> O suporte para `enable_intel_hpc_platform` foi adicionado ao AWS ParallelCluster versão 2.5.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`encrypted_ephemeral`

(Opcional) Criptografa os volumes de armazenamento de instâncias efêmeros com chaves na memória não recuperáveis usando LUKS (Linux Unified Key Setup).

Para obter mais informações, consulte https://gitlab.com/cryptsetup/cryptsetup/blob/master/README.md.

O valor padrão é `false`.

```
encrypted_ephemeral = true
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ephemeral_dir

(Opcional) Define o caminho onde volumes de armazenamento de instâncias são montados, caso sejam usados.

O valor padrão é `/scratch`.

```
ephemeral_dir = /scratch
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

extra_json

(Opcional) Define o JSON extra que é mesclado no. Chef `dna.json`. Para obter mais informações, consulte Criação de uma AWS ParallelCluster AMI personalizada.

O valor padrão é `{}`.

```
extra_json = {}
```

> **Note**
>
> A partir da AWS ParallelCluster versão 2.6.1, a maioria das receitas de instalação são ignoradas por padrão ao iniciar os nós para melhorar os tempos de inicialização. Para executar todas as fórmulas de instalação para uma melhor compatibilidade com versões anteriores em detrimento dos tempos de inicialização, adicione `"skip_install_recipes" : "no"` à chave `cluster` na configuração `extra_json`. Por exemplo:
>
> ```
> extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
> ```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

fsx_settings

(Opcional) Especifica a seção que define a configuração do FSx Lustre. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a seção [fsx].

Por exemplo, a configuração a seguir especifica que a seção que começa [fsx fs] é usada FSx para a configuração for Lustre.

```
fsx_settings = fs
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

iam_lambda_role

(Opcional) Define o nome de uma função de AWS Lambda execução existente. Esta função está vinculada a todas as funções do Lambda no cluster. Para obter mais informações, consulte perfil do IAM para execução do AWS Lambda no Guia do desenvolvedor do AWS Lambda .

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Um nome de função do IAM e seu Amazon Resource Name (ARN) são diferentes. ARNs não pode ser usado como argumento para`iam_lambda_role`. Se ambos `ec2_iam_role` e `iam_lambda_role` forem definidos e se `scheduler` for `sge`, `slurm` ou `torque`, não haverá funções criadas. Se `scheduler` for `awsbatch`, então haverá funções criadas durante `pcluster start`. Para obter exemplos de políticas, consulte `ParallelClusterLambdaPolicy` usando SGE, Slurm ou Torque e `ParallelClusterLambdaPolicy` usando `awsbatch`.

Não há valor padrão.

```
iam_lambda_role = ParallelClusterLambdaRole
```

> **Note**
>
> Support for `iam_lambda_role` adicionado na AWS ParallelCluster versão 2.10.1.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

`initial_queue_size`

(Opcional) Define o número inicial de instâncias do Amazon EC2 a serem executadas como nós de computação no cluster. Se a configuração `queue_settings` estiver definida, essa configuração deverá ser removida e substituída pelas configurações `initial_count` nas seções `[compute_resource]`.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Essa configuração é aplicável apenas para programadores tradicionais (SGE, Slurm e Torque). Se a configuração `maintain_initial_size` for `true`, a configuração `initial_queue_size` deverá ser pelo menos uma (1).

Se o programador for `awsbatch`, use `min_vcpus` em vez disso.

O padrão é 2.

```
initial_queue_size = 2
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

`key_name`

(Opcional) Nomeia um par de chaves do Amazon EC2 existente para habilitar o acesso SSH às instâncias.

```
key_name = mykey
```

> **Note**
>
> Antes da AWS ParallelCluster versão 2.11.0, `key_name` era uma configuração obrigatória.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`maintain_initial_size`

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

(Opcional) Mantém o tamanho inicial do grupo do Auto Scaling para programadores tradicionais (SGE, Slurm, e Torque).

Se o programador for `awsbatch`, use `desired_vcpus` em vez disso.

Essa configuração é um sinalizador booliano. Se definido como `true`, o grupo di Auto Scaling nunca terá menos membros do que o valor de `initial_queue_size`, e o valor de `initial_queue_size` deve ser um (1) ou maior. O cluster ainda pode aumentar até o valor de `max_queue_size`. Se `cluster_type = spot`, o grupo do Auto Scaling pode ter instâncias interrompidas, e o tamanho pode cair para menos de `initial_queue_size`.

Se for definido como `false`, o grupo do Auto Scaling poderá ser reduzido para zero (0) membros a fim de impedir que os recursos fiquem ociosos quando não forem necessários.

Se a configuração `queue_settings` estiver definida, essa configuração deverá ser removida e substituída pelas configurações `initial_count` e `min_count` nas seções `[compute_resource]`.

O padrão é `false`.

```
maintain_initial_size = false
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

master_instance_type

(Opcional) Define o tipo de instância do Amazon EC2 que é usado para o nó principal. A arquitetura do tipo de instância deve ser a mesma usada para a configuração `compute_instance_type`.

Se Regiões da AWS tiver um nível gratuito, o padrão é o tipo de instância de nível gratuito (`t2.microout3.micro`). Se Regiões da AWS não tiver um nível gratuito, o padrão é. `t3.micro` Para obter mais informações sobre o nível AWS gratuito, consulte Nível AWS gratuito FAQs.

```
master_instance_type = t2.micro
```

> **Note**
>
> Antes da AWS ParallelCluster versão 2.10.1, o padrão era in all. `t2.micro` Regiões da AWS Na AWS ParallelCluster versão 2.10.0, o `p4d.24xlarge` não tinha suporte para o nó principal. Support para instâncias AWS baseadas em Graviton (como `A1` `eC6g`) foi adicionado na AWS ParallelCluster versão 2.8.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

master_root_volume_size

(Opcional) Especifica o tamanho do volume raiz do nó principal em gibibytes (GiB). A AMI deve oferecer suporte para `growroot`.

O valor padrão é `35`.

> **Note**
>
> Para AWS ParallelCluster versões entre 2.5.0 e 2.10.4, o padrão era 25. Antes da AWS ParallelCluster versão 2.5.0, o padrão era 20.

```
master_root_volume_size = 35
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

max_queue_size

(Opcional) Define o número máximo de instâncias do Amazon EC2 que podem ser executadas no cluster. Se a configuração `queue_settings` estiver definida, essa configuração deverá ser removida e substituída pelas configurações `max_count` nas seções `[compute_resource]`.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Essa configuração é aplicável apenas para programadores tradicionais (SGE, Slurm e Torque).

Se o programador for `awsbatch`, use `max_vcpus` em vez disso.

O padrão é `10`.

```
max_queue_size = 10
```

Política de atualização: essa configuração pode ser alterada durante uma atualização, mas a frota computacional deve ser interrompida se o valor for reduzido. Caso contrário, os nós existentes poderão ser encerrados.

max_vcpus

(Opcional) Especifica o número máximo de v CPUs no ambiente computacional. Usador apenas se o programador for `awsbatch`.

O valor padrão é `20`.

```
max_vcpus = 20
```

Política de atualização: essa configuração não pode ser diminuída durante uma atualização.

min_vcpus

(Opcional) Mantém o tamanho inicial do grupo do Auto Scaling para o programador `awsbatch`.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Se o programador for SGE, Slurm ou Torque, use `maintain_initial_size` em vez disso.

O ambiente de computação nunca terá menos membros do que o valor de `min_vcpus`.

O padrão é `0`.

```
min_vcpus = 0
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

`placement`

(Opcional) Define a lógica de grupo de posicionamento de cluster, permitindo que todo o cluster ou somente as instâncias de computação usem o grupo de posicionamento de cluster.

Se a configuração `queue_settings` estiver definida, essa configuração deverá ser removida e substituída pelas configurações `placement_group` para cada uma das seções de `[queue]`. Se o mesmo grupo de posicionamento for usado para diferentes tipos de instância, é mais provável que a solicitação falhe devido a um erro de capacidade insuficiente. Para ter mais informações, consulte Capacidade insuficiente da instância no Guia do usuário do Amazon EC2. Várias filas só podem compartilhar um grupo de posicionamento se ele for criado com antecedência e configurado na configuração `placement_group` de cada fila. Se cada seção `[queue]` definir uma configuração `placement_group`, o nó principal não poderá estar no grupo de posicionamento de uma fila.

As opções válidas são `cluster` ou `compute`.

Esse parâmetro não é usado quando o programador for `awsbatch`.

O valor padrão é `compute`.

```
placement = compute
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

placement_group

(Opcional) Define o grupo de posicionamento de cluster. Se a configuração queue_settings estiver definida, essa configuração deverá ser removida e substituída pelas configurações placement_group nas seções [queue].

As opções válidas são as seguintes:

- DYNAMIC
- Um nome do grupo com posicionamento em cluster do Amazon EC2

Quando definido como DYNAMIC, um placement group exclusivo será criado e excluído como parte da pilha do cluster.

Esse parâmetro não é usado quando o programador for awsbatch.

Para ter mais informações sobre grupos de posicionamento, consulte Grupos de posicionamento no Guia do usuário do Amazon EC2. Se o mesmo grupo de posicionamento for usado para diferentes tipos de instância, é mais provável que a solicitação falhe devido a um erro de capacidade insuficiente. Para ter mais informações, consulte Capacidade insuficiente da instância no Guia do usuário do Amazon EC2.

Não há valor padrão.

Nem todos os tipos de instância oferecem suporte para placement groups de cluster. Por exemplo, o tipo de instância padrão de t3.micro não oferece suporte para grupos de posicionamento de cluster. Para ter informações sobre a lista de tipos de instância que oferecem suporte a grupos de posicionamento de cluster, consulte as regras e limitações para grupos de posicionamento de cluster no Guia do usuário do Amazon EC2. Consulte Grupos de posicionamento e problemas de execução de instâncias para obter dicas ao trabalhar com placement groups.

```
placement_group = DYNAMIC
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

post_install

(Opcional) Especifica a URL de um script de pós-instalação que é executado após a conclusão de todas as ações de bootstrap do nó. Para obter mais informações, consulte Ações de bootstrap personalizadas.

Ao usar o `awsbatch` como o programador, o script de pós-instalação será executado somente no nó principal.

O formato do parâmetro pode ser `http://`*`hostname/path/to/script.sh`* ou `s3://`*`bucket-name/path/to/script.sh`*.

Não há valor padrão.

```
post_install = s3://<bucket-name>/my-post-install-script.sh
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

post_install_args

(Opcional) Especifica uma lista de argumentos entre aspas a serem enviados ao script de pós-instalação.

Não há valor padrão.

```
post_install_args = "argument-1 argument-2"
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

pre_install

(Opcional) Especifica a URL de um script de pré-instalação que é executado antes do início de qualquer ação de bootstrap de implantação do nó. Para obter mais informações, consulte Ações de bootstrap personalizadas.

Ao usar o `awsbatch` como o programador, o script de pré-instalação será executado somente no nó principal.

O formato do parâmetro pode ser `http://`*`hostname/path/to/script.sh`* ou `s3://`*`bucket-name/path/to/script.sh`*.

Não há valor padrão.

```
pre_install = s3://bucket-name/my-pre-install-script.sh
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

pre_install_args

(Opcional) Especifica uma lista de argumentos entre aspas a serem enviados ao script de pré-instalação.

Não há valor padrão.

```
pre_install_args = "argument-3 argument-4"
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

proxy_server

(Opcional) Define um servidor de proxy HTTP ou HTTPS, normalmente `http://x.x.x.x:8080`.

Não há valor padrão.

```
proxy_server = http://10.11.12.13:8080
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

queue_settings

(Opcional) Especifica que o cluster usa filas em vez de uma frota computacional homogênea e quais `[queue]` seções são usadas. A primeira seção `[queue]` listada é a fila padrão do programador. Os nomes da seção `queue` deve começar com uma letra minúscula, conter no máximo 30 caracteres e conter apenas letras minúsculas, números e hifens (-).

> **Important**
>
> `queue_settings` só é compatível quando `scheduler` está definido para `slurm`. As configurações `cluster_type`, `compute_instance_type`, `initial_queue_size`,

`maintain_initial_size`, `max_queue_size`, `placement`, `placement_group`, e `spot_price` não devem ser especificadas. As configurações `disable_hyperthreading` e `enable_efa` podem ser especificadas na `seção [cluster]` ou nas `seções[queue]`, mas não em ambas.

São suportadas até cinco (5) `seções [queue]`.

Para obter mais informações, consulte a `seção [queue]`.

Por exemplo, a seguinte configuração especifica que as seções que começam com `[queue q1]` e `[queue q2]` são usados.

```
queue_settings = q1, q2
```

Note

Support for `queue_settings` adicionado na AWS ParallelCluster versão 2.9.0.

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

raid_settings

(Opcional) Identifica a seção `[raid]` com a configuração de RAID de volume do Amazon EBS. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a `seção [raid]`.

Por exemplo, a configuração a seguir especifica que a seção que começa com `[raid rs]` seja usada para a configuração do Auto Scaling.

```
raid_settings = rs
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

s3_read_resource

(Opcional) Especifica um recurso do Amazon S3 ao AWS ParallelCluster qual os nós recebem acesso somente de leitura.

Por exemplo, `arn:aws:s3:::my_corporate_bucket*` fornece acesso somente de leitura ao *my_corporate_bucket* bucket e aos objetos no bucket.

Consulte trabalhar com o Amazon S3 para obter detalhes sobre o formato.

Não há valor padrão.

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

Política de atualização: essa configuração pode ser alterada durante uma atualização.

s3_read_write_resource

(Opcional) Especifica um recurso do Amazon S3 ao AWS ParallelCluster qual os nós têm acesso read/write concedido.

Por exemplo, `arn:aws:s3:::my_corporate_bucket/Development/*` fornece read/write acesso a todos os objetos na `Development` pasta do *my_corporate_bucket* bucket.

Consulte trabalhar com o Amazon S3 para obter detalhes sobre o formato.

Não há valor padrão.

```
s3_read_write_resource = arn:aws:s3:::my_corporate_bucket/*
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

scaling_settings

Identifica a seção `[scaling]` com a configuração do Auto Scaling. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a seção `[scaling]`.

Por exemplo, a configuração a seguir especifica que a seção que começa com `[scaling custom]` seja usada para a configuração do Auto Scaling.

```
scaling_settings = custom
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`scheduler`

(Obrigatório) Define o programador do cluster.

As opções válidas são as seguintes:

`awsbatch`

AWS Batch

Para obter mais informações sobre o programador `awsbatch`, consulte configuração de redes e AWS Batch (`awsbatch`).

`sge`

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Son of Grid Engine (SGE)

`slurm`

Slurm Workload Manager (Slurm)

`torque`

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Torque Resource Manager (Torque)

> **Note**
>
> Antes da AWS ParallelCluster versão 2.7.0, o `scheduler` parâmetro era opcional e o padrão era. `sge` A partir da AWS ParallelCluster versão 2.7.0, o `scheduler` parâmetro é obrigatório.

```
scheduler = slurm
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

shared_dir

(Opcional) Define o caminho onde o volume compartilhado do Amazon EBS é montado.

Não use esta opção com vários volumes do Amazon EBS. Em vez disso, forneça valores de `shared_dir` em cada seção `[ebs]`.

Consulte a seção `[ebs]` para obter detalhes sobre como trabalhar com vários volumes do Amazon EBS.

O valor padrão é `/shared`.

O exemplo a seguir mostra um volume compartilhado do Amazon EBS montado em `/myshared`.

```
shared_dir = myshared
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

spot_bid_percentage

(Opcional) Define a porcentagem sob demanda usada para calcular o preço spot máximo para o ComputeFleet, quando `awsbatch` é o programador.

Se não for especificado, o preço de mercado spot atual será selecionado, limitado ao preço sob demanda.

```
spot_bid_percentage = 85
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

spot_price

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

(Opcional) Define o preço spot máximo para ComputeFleet os programadores tradicionais (SGE,Slurm, eTorque). Usado somente quando a configuração `cluster_type` está definida como `spot`. Se você não especificar um valor, será cobrado o preço spot, limitado ao preço sob demanda. Se a configuração `queue_settings` estiver definida, essa configuração deverá ser removida e substituída pelas configurações `spot_price` nas seções `[compute_resource]`.

Se o programador for `awsbatch`, use spot_bid_percentage em vez disso.

Para obter ajuda para encontrar uma instância spot que atenda às suas necessidades, consulte o Spot Instance Advisor.

```
spot_price = 1.50
```

> **Note**
>
> Na AWS ParallelCluster versão 2.5.0, se for especificada, `cluster_type = spot` mas `spot_price` não for especificada, a instância será iniciada em caso de ComputeFleet falha. Isso foi corrigido na AWS ParallelCluster versão 2.5.1.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

tags

(Opcional) Define as tags a serem usadas por CloudFormation.

Se as tags da linha de comando estiverem especificadas por --tags, elas são mescladas com as tags de configuração.

As tags da linha de comando substituem as tags de configuração que têm a mesma chave.

As tags são formatadas por JSON. Não use aspas fora das chaves.

Para obter mais informações, consulte tipos de tags de recurso do CloudFormation no Guia de usuário do AWS CloudFormation .

```
tags = {"key" : "value", "key2" : "value2"}
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

> **Note**
>
> A política de atualização não suportou a alteração da configuração `tags` para AWS ParallelCluster versão 2.8.0 até a versão 2.9.1.
> Para as versões 2.10.0 até a versão 2.11.7, a política de atualização listada que suportava a alteração da configuração `tags` não é precisa. Não há suporte para uma atualização de cluster ao modificar essa configuração.

`template_url`

(Opcional) Define o caminho para o AWS CloudFormation modelo usado para criar o cluster.

Atualiza o uso do modelo que foi usado originalmente para criar a pilha.

O padrão é `https://`*`aws_region_name`*`-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-`*`version`*`.cfn.json`.

> **Warning**
>
> Este é um parâmetro avançado. Qualquer alteração nessa configuração é feita por sua conta e risco.

```
template_url = https://us-east-1-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-2.11.9.cfn.json
```

Política de atualização: essa configuração não é analisada durante uma atualização.

`vpc_settings`

(Obrigatório) Identifica a seção `[vpc]` com a configuração da Amazon VPC na qual o cluster é implantado. O nome da seção deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Para obter mais informações, consulte a seção `[vpc]`.

Por exemplo, a configuração a seguir especifica que a seção que começa com `[vpc public]` é usada para a configuração do Amazon VPC.

```
vpc_settings = public
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

Seção `[compute_resource]`

Define as configurações de um recurso computacional. `[compute_resource]`As seções são referenciadas pela configuração `compute_resource_settings` na `[queue]`seção. `[compute_resource]`As seções só são suportadas quando `scheduler` está definido como `slurm`.

O formato é`[compute_resource` *<compute-resource-name>*`]`. *compute-resource-name*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[compute_resource cr1]
instance_type = c5.xlarge
min_count = 0
initial_count = 2
max_count = 10
spot_price = 0.5
```

> **Note**
>
> Support para a `[compute_resource]`seção foi adicionado na AWS ParallelCluster versão 2.9.0.

Tópicos

initial_count

(Opcional) Define o número inicial de EC2 instâncias da Amazon a serem executadas para esse recurso computacional. A criação do cluster não é concluída até que pelo menos tantos nós tenham sido lançados no recurso computacional. Se a configuração `compute_type` da fila for `spot` e não houver instâncias spot suficientes disponíveis, a criação do cluster poderá expirar e falhar. Qualquer contagem maior do que a configuração `min_count` é a capacidade dinâmica sujeita à configuração `scaledown_idletime`. Essa configuração substitui a configuração `initial_queue_size`.

O padrão é `0`.

```
initial_count = 2
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

instance_type

(Obrigatório) Define o tipo de EC2 instância da Amazon que é usado para esse recurso computacional. A arquitetura do tipo de instância deve ser a mesma usada para a configuração `master_instance_type`. A configuração `instance_type` deve ser exclusiva para cada `[compute_resource]`seção referenciada por uma `[queue]`seção. Essa configuração substitui a configuração `compute_instance_type`.

```
instance_type = t2.micro
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

max_count

(Opcional) Define o número máximo de EC2 instâncias da Amazon que podem ser executadas nesse recurso computacional. Qualquer contagem maior do que a configuração `initial_count` é iniciada no modo de desligamento. Essa configuração substitui a configuração `max_queue_size`.

O padrão é `10`.

```
max_count = 10
```

Política de atualização: reduzir o tamanho de uma fila abaixo do número atual de nós exige que a frota de computação seja interrompida primeiro.

> **Note**
>
> A política de atualização não suportou a alteração da `max_count` configuração até que a frota computacional fosse interrompida da AWS ParallelCluster versão 2.0.0 até a versão 2.9.1.

min_count

(Opcional) Define o número mínimo de EC2 instâncias da Amazon que podem ser executadas nesse recurso computacional. Esses nós são todos de capacidade estática. A criação do cluster não é concluída até que pelo menos esse número de nós tenham sido lançados no recurso computacional.

O padrão é `0`.

```
min_count = 1
```

Política de atualização: reduzir o número de nós estáticos na fila exige que a frota de computação seja interrompida primeiro.

> **Note**
>
> A política de atualização não suportou a alteração da `min_count` configuração até que a frota computacional fosse interrompida da AWS ParallelCluster versão 2.0.0 até a versão 2.9.1.

spot_price

(Opcional) Define o preço máximo de spot para esse recurso computacional. Usado somente quando a configuração compute_type da fila que contém esses recursos computacionais está definida como spot. Essa configuração substitui a configuração spot_price.

Se você não especificar um valor, será cobrado o preço spot, limitado ao preço sob demanda.

Para obter ajuda para encontrar uma instância spot que atenda às suas necessidades, consulte o Spot Instance Advisor.

```
spot_price = 1.50
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

Seção [cw_log]

Define as configurações para CloudWatch Logs.

O formato é[cw_log *cw-log-name*]. *cw-log-name*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[cw_log custom-cw-log]
enable = true
retention_days = 14
```

Para ter mais informações, consulte Integração com Amazon CloudWatch Logs, CloudWatch Painel da Amazon e Integração com Amazon CloudWatch Logs.

> **Note**
>
> Support for cw_log adicionado na AWS ParallelCluster versão 2.6.0.

enable

(Opcional) Indica se CloudWatch os registros estão habilitados.

O valor padrão é true. Use false para desativar CloudWatch os registros.

O exemplo a seguir ativa CloudWatch os registros.

```
enable = true
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`retention_days`

(Opcional) Indica por quantos dias o CloudWatch Logs retém eventos de registro individuais.

O valor padrão é 14. Os valores compatíveis são 1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827 e 3653.

O exemplo a seguir configura os CloudWatch registros para reter eventos de registro por 30 dias.

```
retention_days = 30
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

Seção `[dashboard]`

Define as configurações do CloudWatch painel.

O formato é`[dashboard` *`dashboard-name`*`]`. *`dashboard-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[dashboard custom-dashboard]
enable = true
```

> **Note**
>
> Support for `dashboard` adicionado na AWS ParallelCluster versão 2.10.0.

`enable`

(Opcional) Indica se o CloudWatch painel está ativado.

O valor padrão é `true`. Use `false` para desativar o CloudWatch painel.

O exemplo a seguir ativa o CloudWatch painel.

```
enable = true
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

Seção `[dcv]`

Define as configurações para o servidor Amazon DCV em execução no nó principal.

Para criar e configurar um servidor Amazon DCV, especifique o cluster `dcv_settings` com o nome definido na seção `dcv` e defina `enable` como `master`, e `base_os` como `alinux2`, `centos7`, `ubuntu1804` ou `ubuntu2004`. Se o nó principal for uma instância ARM, defina `base_os` como `alinux2`, `centos7` ou `ubuntu1804`.

O formato é`[dcv dcv-name]`. *`dcv-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[dcv custom-dcv]
enable = master
port = 8443
access_from = 0.0.0.0/0
```

Para obter mais informações, consulte Conecte-se ao nó principal por meio do Amazon DCV.

Important

Por padrão, a porta Amazon DCV configurada por AWS ParallelCluster está aberta para todos os IPv4 endereços. Contudo, a conexão a uma porta do Amazon DCV só será possível se você tiver o URL da sessão Amazon DCV e se conectar à sessão Amazon DCV dentro de 30 segundos após o retorno do URL pelo `pcluster dcv connect`. Use a configuração `access_from` para restringir ainda mais o acesso à porta do Amazon DCV com um intervalo IP no formato CIDR e use a configuração `port` para definir uma porta alternativa.

Note

Suporte para a seção `[dcv]` no `centos8` foi removido no AWS ParallelCluster versão 2.10.4. Support para a `[dcv]`seção on `centos8` foi adicionado na AWS ParallelCluster

versão 2.10.0. Support para a `[dcv]`seção sobre instâncias AWS baseadas em Graviton foi adicionado na AWS ParallelCluster versão 2.9.0. Support para a `[dcv]`seção on `alinux2` e `ubuntu1804` foi adicionado na AWS ParallelCluster versão 2.6.0. Support para a `[dcv]`seção on `centos7` foi adicionado na AWS ParallelCluster versão 2.5.0.

`access_from`

(Opcional, recomendado) Especifica o intervalo de IP no formato CIDR para conexões com o Amazon DCV. Essa configuração é usada somente ao AWS ParallelCluster criar o grupo de segurança.

O valor padrão é `0.0.0.0/0` e permite o acesso a partir de qualquer endereço de Internet.

```
access_from = 0.0.0.0/0
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

`enable`

(Obrigatório) Indica se o Amazon DCV está habilitado no nó principal. Para habilitar o Amazon DCV no nó principal e configurar a regra de grupo de segurança necessária, defina a configuração `enable` como `master`.

O exemplo a seguir habilita o Amazon DCV no nó principal.

```
enable = master
```

> **Note**
>
> O Amazon DCV gera automaticamente um certificado autoassinado, que é utilizado para proteger o tráfego entre o cliente Amazon DCV e o servidor Amazon DCV em execução no nó principal. Para configurar seu próprio certificado, consulte Certificado HTTPS do Amazon DCV.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`port`

(Opcional) Especifica a porta para o Amazon DCV.

O valor padrão é `8443`.

```
port = 8443
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

Seção `[ebs]`

Define as definições de configuração de volume do Amazon EBS para os volumes que são montados no nó de cabeçalho e compartilhados pelo NFS com os nós de computação.

Para saber como incluir volumes do Amazon EBS em sua definição de cluster, consulte Seção `[cluster]` / `ebs_settings`.

Para usar um volume existente do Amazon EBS para armazenamento permanente de longo prazo que seja independente do ciclo de vida do cluster, especifique `ebs_volume_id`.

Se você não especificar`ebs_volume_id`, AWS ParallelCluster cria o volume do EBS a partir das `[ebs]` configurações ao criar o cluster e exclui o volume e os dados quando o cluster é excluído.

Para obter mais informações, consulte Melhores práticas: mover um cluster para uma nova versão AWS ParallelCluster secundária ou de patch.

O formato é`[ebs` *`ebs-name`*`]`. *`ebs-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[ebs custom1]
shared_dir = vol1
ebs_snapshot_id = snap-xxxxx
volume_type = io1
volume_iops = 200
...

[ebs custom2]
shared_dir = vol2
...
```

```
...
```

Tópicos

shared_dir

(Obrigatório) Especifica o caminho em que o volume compartilhado do Amazon EBS é montado.

Esse parâmetro é obrigatório ao usar vários volumes do Amazon EBS.

Ao usar um volume do Amazon EBS, essa opção substitui o `shared_dir` que está especificado na seção `[cluster]`. No exemplo a seguir, o volume é montado em `/vol1`.

```
shared_dir = vol1
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ebs_kms_key_id

(Opcional) Especifica uma AWS KMS chave personalizada a ser usada para criptografia.

Esse parâmetro deve ser usado em conjunto com `encrypted = true`. Também deve ter uma `ec2_iam_role` personalizada.

Para obter mais informações, consulte Criptografia de disco com uma chave do KMS personalizada.

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ebs_snapshot_id

(Opcional) Define o ID do snapshot do Amazon EBS, caso esteja usando um snapshot como a origem do volume.

Não há valor padrão.

```
ebs_snapshot_id = snap-xxxxx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ebs_volume_id

(Opcional) Define o ID do volume de um volume do Amazon EBS existente a ser anexado ao nó principal.

Não há valor padrão.

```
ebs_volume_id = vol-xxxxxx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

encrypted

(Opcional) Especifica se um volume do Amazon EBS é criptografado. Observação: Não use com snapshots.

O valor padrão é `false`.

```
encrypted = false
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

volume_iops

(Opcional) Define o número de IOPS para os volumes do tipo `io1`, `io2`, e `gp3`.

O valor padrão, os valores suportados e a proporção do `volume_iops` para `volume_size` variam de acordo com `volume_type` e `volume_size`.

`volume_type` = `io1`

Padrão `volume_iops` = 100

Valores suportados `volume_iops` = 100–64000 †

Proporção máxima de `volume_iops` para `volume_size` = 50 IOPS para cada GiB. 5000 IOPS exigem um `volume_size` de pelo menos 100 GiB.

`volume_type` = `io2`

Padrão `volume_iops` = 100

Valores suportados `volume_iops` = 100–64000 (256000 para volumes do `io2` Block Express) †

Proporção máxima de `volume_iops` para `volume_size` = 500 IOPS para cada GiB. 5000 IOPS exigem um `volume_size` de pelo menos 10 GiB.

`volume_type` = `gp3`

Padrão `volume_iops` = 3000

Valores suportados `volume_iops` = 3000–16000

Proporção máxima de `volume_iops` para `volume_size` = 500 IOPS para cada GiB. 5000 IOPS exigem um `volume_size` de pelo menos 10 GiB.

```
volume_iops = 200
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

† O número máximo de IOPS é garantido somente em instâncias criadas no Nitro System provisionadas com mais de 32.000 IOPS. Outras instâncias garantem até 32.000 IOPS. A menos que você modifique o volume, os volumes de `io1` mais antigos podem não atingir o desempenho total. `io2` Os volumes do Block Express oferecem suporte a valores `volume_iops` de até 256.000. Para obter mais informações, consulte Volumes do `io2` Block Express (em versão prévia) no Guia EC2 do usuário da Amazon.

volume_size

(Opcional) Especifica o tamanho do volume a ser criado em GiB (caso não esteja usando um snapshot).

O valor padrão e os valores suportados variam de acordo com o volume_type.

volume_type = standard

Padrão volume_size = 20 GiB

Valores suportados volume_size = 1–1024 GiB

volume_type = gp2, io1, io2, e gp3

Padrão volume_size = 20 GiB

Valores suportados volume_size = 1–16384 GiB

volume_type = sc1 e st1

Padrão volume_size = 500 GiB

Valores suportados volume_size = 500–16384 GiB

```
volume_size = 20
```

> **Note**
>
> Antes da AWS ParallelCluster versão 2.10.1, o valor padrão para todos os tipos de volume era de 20 GiB.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

volume_throughput

(Opcional) Define o throughput para tipos de volume gp3, em MiB/s.

O valor padrão é 125.

Valores suportados volume_throughput = 125–1000 MiB/s

A proporção de `volume_throughput` para `volume_iops` não pode ser superior a 0,25. O throughput máximo de 1000 MiB/s exige que a configuração `volume_iops` seja de pelo menos 4000.

```
volume_throughput = 1000
```

> **Note**
>
> Support for `volume_throughput` adicionado na AWS ParallelCluster versão 2.10.1.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`volume_type`

(Opcional) Especifica o tipo de volume do Amazon EBS do volume que você deseja executar.

As opções válidas são os seguintes tipos de volume:

`gp2`, `gp3`

SSD de uso geral

`io1`, `io2`

Provisioned IOPS SSD

`st1`

HDD otimizado para throughput

`sc1`

Disco rígido frio

`standard`

Geração magnética anterior

Para obter mais informações, consulte os tipos de volume do Amazon EBS no Guia do EC2 usuário da Amazon.

O valor padrão é `gp2`.

```
volume_type = io2
```

> **Note**
>
> Support `gp3` e `io2` foi adicionado na AWS ParallelCluster versão 2.10.1.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

Seção `[efs]`

Define a configuração para o Amazon EFS que está montado nos nós principal e de computação. Para obter mais informações, consulte CreateFileSystema Referência da API do Amazon EFS.

Para saber como incluir sistemas de arquivos do Amazon EFS em sua definição de cluster, consulte `Seção [cluster]` / `efs_settings`.

Para usar um sistema de arquivos existente do Amazon EFS para armazenamento permanente de longo prazo que seja independente do ciclo de vida do cluster, especifique `efs_fs_id`.

Se você não especificar`efs_fs_id`, AWS ParallelCluster cria o sistema de arquivos Amazon EFS a partir das `[efs]` configurações ao criar o cluster e exclui o sistema de arquivos e os dados quando o cluster é excluído.

Para obter mais informações, consulte Melhores práticas: mover um cluster para uma nova versão AWS ParallelCluster secundária ou de patch.

O formato é`[efs` *`efs-name`*`]`. *`efs-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[efs customfs]
shared_dir = efs
encrypted = false
performance_mode = generalPurpose
```

Tópicos

- efs_fs_id
- efs_kms_key_id

- encrypted
- performance_mode
- provisioned_throughput
- shared_dir
- throughput_mode

efs_fs_id

(Opcional) Define o ID do sistema de arquivos do Amazon EFS para um sistema de arquivos existente.

Especificar essa opção invalida todas as outras opções do Amazon EFS, exceto `shared_dir`.

Se você definir essa opção, serão compatíveis apenas os seguintes sistemas de arquivos:

- Sistemas de arquivos que não tenham um destino de montagem na zona de disponibilidade da pilha.
- Sistemas de arquivos que têm um destino de montagem existente na zona de disponibilidade da pilha, com o tráfego de entrada e saída do NFS permitidos a partir de `0.0.0.0/0`.

A verificação de sanidade para validar o `efs_fs_id` requer a função do IAM para ter as seguintes permissões:

- `elasticfilesystem:DescribeMountTargets`
- `elasticfilesystem:DescribeMountTargetSecurityGroups`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaceAttribute`

Para evitar erros, é necessário adicionar essas permissões à função do IAM ou definir `sanity_check = false`.

> **Important**
>
> Quando você define um destino de montagem com tráfego NFS de entrada e saída permitido`0.0.0.0/0`, ele expõe o sistema de arquivos às solicitações de montagem do

NFS de qualquer lugar na Zona de Disponibilidade do destino de montagem. AWS não recomenda criar um alvo de montagem na zona de disponibilidade da pilha. Em vez disso, vamos AWS lidar com essa etapa. Caso deseje ter um destino de montagem na zona de disponibilidade da pilha, considere o uso de um grupo de segurança personalizado, fornecendo uma opção `vpc_security_group_id` na seção `[vpc]`. Depois, adicione o grupo de segurança ao destino de montagem e desative `sanity_check` para criar o cluster.

Não há valor padrão.

```
efs_fs_id = fs-12345
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`efs_kms_key_id`

(Opcional) Identifica a chave AWS Key Management Service (AWS KMS) gerenciada pelo cliente a ser usada para proteger o sistema de arquivos criptografados. Se isso estiver definido, a `encrypted` configuração deverá ser definida como `true`. Isso corresponde ao KmsKeyIdparâmetro na Referência da API do Amazon EFS.

Não há valor padrão.

```
efs_kms_key_id = 1234abcd-12ab-34cd-56ef-1234567890ab
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`encrypted`

(Opcional) Indica se o sistema de arquivos está criptografado. Isso corresponde ao parâmetro Encrypted na Referência de API do Amazon EFS.

O valor padrão é `false`.

```
encrypted = true
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

performance_mode

(Opcional) Define o modo de desempenho do sistema de arquivos. Isso corresponde ao PerformanceModeparâmetro na Referência da API do Amazon EFS.

As opções válidas são as seguintes:

- `generalPurpose`
- `maxIO`

Ambos os valores diferenciam maiúsculas de minúsculas.

Recomendamos o modo de desempenho `generalPurpose` para a maioria dos sistemas de arquivos.

Os sistemas de arquivos que usam o modo de desempenho `maxIO` podem ser dimensionados para níveis superiores de throughput e operações por segundo agregadas. No entanto, há um compromisso com latências um pouco mais altas para a maioria das operações de arquivo.

Esse parâmetro não poderá ser alterado depois que o sistema de arquivos for criado.

O valor padrão é `generalPurpose`.

```
performance_mode = generalPurpose
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

provisioned_throughput

(Opcional) Define o throughput provisionado do sistema de arquivos, medido em MiB/s. Isso corresponde ao ProvisionedThroughputInMibpsparâmetro na Referência da API do Amazon EFS.

Se você usar esse parâmetro, deverá definir `throughput_mode` como `provisioned`.

A quota do throughput é de `1024` MiB/s. Para solicitar um aumento na cota, entre em contato com o Suporte.

O valor mínimo é `0.0` MiB/s

```
provisioned_throughput = 1024
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

`shared_dir`

(Obrigatório) Define o ponto de montagem do Amazon EFS nos nós principal e de computação.

Esse parâmetro é obrigatório. A seção do Amazon EFS só é usada se `shared_dir` for especificado.

Não use `NONE` ou `/NONE` como o diretório compartilhado.

O exemplo a seguir monta o Amazon EFS no `/efs`.

```
shared_dir = efs
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`throughput_mode`

(Opcional) Define o modo de throughput do sistema de arquivos. Isso corresponde ao ThroughputModeparâmetro na Referência da API do Amazon EFS.

As opções válidas são as seguintes:

- `bursting`
- `provisioned`

O valor padrão é `bursting`.

```
throughput_mode = provisioned
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

Seção `[fsx]`

Define as configurações de um sistema de arquivos FSx anexado ao Lustre. Para obter mais informações, consulte Amazon FSx CreateFileSystem na Amazon FSx API Reference.

Se for `base_os` suportado o é `alinux2` `centos7`ubuntu1804,, ouubuntu2004, FSx for Lustre.

Ao usar o Amazon Linux, o kernel deve ser `4.14.104-78.84.amzn1.x86_64` ou uma versão posterior. Para obter instruções, consulte Instalação do cliente lustre no Guia do usuário do Amazon FSx for Lustre.

> **Note**
>
> FSx O for Lustre atualmente não é suportado quando usado `awsbatch` como agendador.

> **Note**
>
> O suporte FSx para Lustre on `centos8` foi removido na AWS ParallelCluster versão 2.10.4. Support FSx for Lustre on `ubuntu2004` foi adicionado na AWS ParallelCluster versão 2.11.0. Support FSx for Lustre on `centos8` foi adicionado na AWS ParallelCluster versão 2.10.0. Support FSx for Lustre on`alinux2`,`ubuntu1604`, e `ubuntu1804` foi adicionado na AWS ParallelCluster versão 2.6.0. Support FSx for Lustre on `centos7` foi adicionado na AWS ParallelCluster versão 2.4.0.

Se estiver usando um sistema de arquivos existente, ele deve ser associado a um grupo de segurança que permita o tráfego de entrada do TCP à porta 988. Definir a origem como `0.0.0.0/0` em uma regra de grupo de segurança fornece acesso ao cliente de todos os intervalos de IP no grupo de segurança da VPC para o protocolo e o intervalo de portas dessa regra. Para limitar ainda mais o acesso aos seus sistemas de arquivos, recomendamos o uso de fontes mais restritivas para suas regras de grupo de segurança. Por exemplo, você pode usar intervalos CIDR, endereços IP ou grupos IDs de segurança mais específicos. Isso será feito automaticamente quando não estiver usando `vpc_security_group_id`.

Para usar um sistema de FSx arquivos existente da Amazon para armazenamento permanente de longo prazo que seja independente do ciclo de vida do cluster, especifique`fsx_fs_id`.

Se você não especificar`fsx_fs_id`, AWS ParallelCluster cria o sistema de arquivos FSx for Lustre a partir das `[fsx]` configurações ao criar o cluster e exclui o sistema de arquivos e os dados quando o cluster for excluído.

Para obter mais informações, consulte Melhores práticas: mover um cluster para uma nova versão AWS ParallelCluster secundária ou de patch.

O formato é`[fsx fsx-name]`. *fsx-name*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[fsx fs]
shared_dir = /fsx
fsx_fs_id = fs-073c3803dca3e28a6
```

Para criar e configurar um novo sistema de arquivos, use os seguintes parâmetros:

```
[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
```

Tópicos

- 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

(Opcional) Especifica a política de importação automática para refletir as alterações no bucket do S3 usado para criar o sistema de arquivos FSx for Lustre. Os valores possíveis são os seguintes:

`NEW`

FSx for Lustre importa automaticamente listagens de diretórios de quaisquer novos objetos adicionados ao bucket S3 vinculado que não existam atualmente no sistema de arquivos FSx for Lustre.

`NEW_CHANGED`

FSx for Lustre importa automaticamente as listagens de arquivos e diretórios de todos os novos objetos adicionados ao bucket do S3 e de quaisquer objetos existentes que sejam alterados no bucket do S3.

Isso corresponde à AutoImportPolicypropriedade. Para obter mais informações, consulte Importar atualizações automaticamente do seu bucket do S3 no Guia do usuário do Amazon FSx for Lustre. Quando o parâmetro `auto_import_policy` é especificado, os parâmetros `automatic_backup_retention_days`, `copy_tags_to_backups`, `daily_automatic_backup_start_time`, e `fsx_backup_id` não devem ser especificados.

Se a `auto_import_policy` configuração não for especificada, as importações automáticas serão desativadas. FSx for Lustre somente atualiza as listagens de arquivos e diretórios do bucket S3 vinculado quando o sistema de arquivos é criado.

```
auto_import_policy = NEW_CHANGED
```

> **Note**
>
> Support for `auto_import_policy` adicionado na AWS ParallelCluster versão 2.10.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

automatic_backup_retention_days

(Opcional) Especifica o número de dias em que os backups automáticos serão retidos. Válido somente para uso com tipos de implantação PERSISTENT_1. Quando o parâmetro automatic_backup_retention_days é especificado, os parâmetros auto_import_policy, export_path, import_path, e imported_file_chunk_size não devem ser especificados. Isso corresponde à AutomaticBackupRetentionDayspropriedade.

O valor padrão é 0. Esta opção desabilita os backups automáticos. Os valores possíveis são números inteiros entre 0 e 35, inclusive.

```
automatic_backup_retention_days = 35
```

> **Note**
>
> O suporte para automatic_backup_retention_days foi adicionado ao AWS ParallelCluster versão 2.8.0.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

copy_tags_to_backups

(Opcional) Especifica se as tags do sistema de arquivos são copiadas para os backups. Válido somente para uso com tipos de implantação PERSISTENT_1. Quando o parâmetro copy_tags_to_backups é especificado, o automatic_backup_retention_days deve ser especificado com um valor maior que 0 e os parâmetros auto_import_policy, export_path, import_path e imported_file_chunk_size não devem ser especificados. Isso corresponde à CopyTagsToBackupspropriedade.

O valor padrão é false.

```
copy_tags_to_backups = true
```

> **Note**
>
> Support for copy_tags_to_backups adicionado na AWS ParallelCluster versão 2.8.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

daily_automatic_backup_start_time

(Opcional) Especifica a hora do dia (UTC) para iniciar os backups automáticos. Válido somente para uso com tipos de implantação PERSISTENT_1. Quando o parâmetro daily_automatic_backup_start_time é especificado, o automatic_backup_retention_days deve ser especificado com um valor maior que 0 e os parâmetros auto_import_policy, export_path, import_path e imported_file_chunk_size não devem ser especificados. Isso corresponde à DailyAutomaticBackupStartTimepropriedade.

O formato é HH:MM, onde HH é a hora do dia preenchida com zeros (0-23) e MM é o minuto da hora preenchido com zeros. Por exemplo, 1:03 da manhã UTC é o seguinte.

```
daily_automatic_backup_start_time = 01:03
```

O valor padrão é um tempo randomizado entre 00:00 e 23:59.

> **Note**
>
> O suporte para daily_automatic_backup_start_time foi adicionado ao AWS ParallelCluster versão 2.8.0.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

data_compression_type

(Opcional) Especifica o tipo de compactação de dados FSx para Lustre. Isso corresponde à DataCompressionTypepropriedade. Para obter mais informações, consulte a compactação FSx de dados do Lustre no Guia do usuário do Amazon FSx for Lustre.

O único valor válido é LZ4. Para desabilitar a compactação de dados, remova o parâmetro data_compression_type.

```
data_compression_type = LZ4
```

> **Note**
>
> Support for `data_compression_type` adicionado na AWS ParallelCluster versão 2.11.0.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

deployment_type

(Opcional) Especifica o tipo FSx de implantação do Lustre. Isso corresponde à DeploymentTypepropriedade. Para obter mais informações, consulte as opções FSx de implantação do Lustre no Guia do usuário do Amazon FSx for Lustre. Escolha um tipo de implantação temporária para armazenamento temporário e processamento de dados em curto prazo. O `SCRATCH_2` é a última geração de sistemas de arquivos transitórios. Ele oferece um throughput de intermitência mais alto que o throughput basal e criptografia de dados em trânsito.

Os valores válidos são `SCRATCH_1`, `SCRATCH_2` e `PERSISTENT_1`.

`SCRATCH_1`

O tipo de implantação padrão FSx para o Lustre. Com esse tipo de implantação, a configuração `storage_capacity` tem valores possíveis de 1200, 2400 e qualquer múltiplo de 3600. Support for `SCRATCH_1` adicionado na AWS ParallelCluster versão 2.4.0.

`SCRATCH_2`

A última geração de sistemas de arquivos transitórios. Suporta até seis vezes o throughput basal para workloads com picos. Ele também oferece suporte à criptografia de dados em trânsito para tipos de instância compatíveis em Regiões da AWS suportadas. Para obter mais informações, consulte Criptografar dados em trânsito no Guia do usuário do Amazon FSx for Lustre. Com esse tipo de implantação, a configuração `storage_capacity` tem valores possíveis de 1200 e qualquer múltiplo de 2400. O suporte para `SCRATCH_2` foi adicionado ao AWS ParallelCluster versão 2.6.0.

`PERSISTENT_1`

Projetada para armazenamento de longo prazo. Os servidores de arquivos são altamente disponíveis e os dados são replicados automaticamente dentro da zona de disponibilidade da AWS do sistema de arquivos. Oferece suporte à criptografia de dados em trânsito para tipos de instância suportadas. Com esse tipo de implantação, a configuração `storage_capacity` tem

valores possíveis de 1200 e qualquer múltiplo de 2400. Support for `PERSISTENT_1` adicionado na AWS ParallelCluster versão 2.6.0.

O valor padrão é `SCRATCH_1`.

```
deployment_type = SCRATCH_2
```

> **Note**
>
> Support for `deployment_type` adicionado na AWS ParallelCluster versão 2.6.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`drive_cache_type`

(Opcional) Especifica que o sistema de arquivos tem um cache de unidade SSD. Se isso estiver definido, a configuração `storage_type` deverá ser definida como `HDD`. Isso corresponde à DriveCacheTypepropriedade. Para obter mais informações, consulte as opções FSx de implantação do Lustre no Guia do usuário do Amazon FSx for Lustre.

O único valor válido é `READ`. Para desabilitar o cache da unidade SSD, não especifique a configuração `drive_cache_type`.

```
drive_cache_type = READ
```

> **Note**
>
> Support for `drive_cache_type` adicionado na AWS ParallelCluster versão 2.10.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`export_path`

(Opcional) Especifica o caminho do Amazon S3 para onde a raiz do sistema de arquivos é exportada. Quando o parâmetro `export_path` é especificado, os

parâmetros automatic_backup_retention_days, copy_tags_to_backups, daily_automatic_backup_start_time, e fsx_backup_id não devem ser especificados. Isso corresponde à ExportPathpropriedade. Os dados e metadados dos arquivos não são exportados automaticamente para o export_path. Para obter informações sobre a exportação de dados e metadados, consulte Exportação de alterações no repositório de dados no Guia do usuário do Amazon FSx for Lustre.

O valor padrão é s3://*import-bucket*/FSxLustre*[creation-timestamp]*, em que *import-bucket* é o bucket fornecido no parâmetro import_path.

```
export_path = s3://bucket/folder
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

fsx_backup_id

(Opcional) Especifica o ID do backup a ser usado para restaurar o sistema de arquivos a partir de um backup existente. Quando o parâmetro fsx_backup_id é especificado, os parâmetros auto_import_policy, deployment_type, export_path, fsx_kms_key_id, import_path, imported_file_chunk_size, storage_capacity e per_unit_storage_throughput não devem ser especificados. Esses parâmetros são lidos do backup. Além disso os parâmetros auto_import_policy, export_path, import_path e imported_file_chunk_size não devem ser especificados.

Isso corresponde à BackupIdpropriedade.

```
fsx_backup_id = backup-fedcba98
```

> **Note**
>
> Support for fsx_backup_id adicionado na AWS ParallelCluster versão 2.8.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

fsx_fs_id

(Opcional) Anexa um sistema de arquivos existente FSx para Lustre.

Se essa opção for especificada, somente as configurações `shared_dir` e `fsx_fs_id` da seção `[fsx]` serão usadas e todas as outras configurações da seção `[fsx]` serão ignoradas.

```
fsx_fs_id = fs-073c3803dca3e28a6
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

fsx_kms_key_id

(Opcional) Especifica o ID da chave da sua AWS Key Management Service (AWS KMS) chave gerenciada pelo cliente.

Essa chave é usada para criptografar os dados no sistema de arquivos em repouso.

Isso deve ser usado com uma `ec2_iam_role` personalizada. Para obter mais informações, consulte Criptografia de disco com uma chave do KMS personalizada. Isso corresponde ao KmsKeyIdparâmetro na Amazon FSx API Reference.

```
fsx_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

> **Note**
>
> Support for `fsx_kms_key_id` adicionado na AWS ParallelCluster versão 2.6.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

import_path

(Opcional) Especifica o bucket do S3 do qual serão carregados os dados no sistema de arquivos e que servirá como bucket de exportação. Para obter mais informações, consulte `export_path`. Se você especificar o parâmetro `import_path`, os parâmetros `automatic_backup_retention_days`, `copy_tags_to_backups`, `daily_automatic_backup_start_time`, e `fsx_backup_id` não devem ser especificados. Isso corresponde ao ImportPathparâmetro na Amazon FSx API Reference.

A importação ocorre na criação do cluster. Para obter mais informações, consulte Importação de dados do seu repositório de dados no Guia do usuário do Amazon FSx for Lustre. Na importação,

somente os metadados do arquivo (nome, propriedade, registro de data e hora e permissões) são importados. Os dados do arquivo não são importados do bucket do S3 até que o arquivo seja acessado pela primeira vez. Para obter informações sobre o pré-carregamento do conteúdo do arquivo, consulte Pré-carregamento de arquivos em seu sistema de arquivos no Guia do usuário do Amazon FSx for Lustre.

Se um valor não for fornecido, o sistema de arquivos ficará vazio.

```
import_path =  s3://bucket
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

imported_file_chunk_size

(Opcional) Determina a contagem de stripe e a quantidade máxima de dados para cada arquivo (em MiB) armazenados em um único disco físico, para arquivos que são importados de um repositório de dados (usando `import_path`). O número máximo de discos nos quais um único arquivo pode ser distribuído é limitado pelo número total de discos que compõem o sistema de arquivos. Quando o parâmetro `imported_file_chunk_size` é especificado, os parâmetros `automatic_backup_retention_days`, `copy_tags_to_backups`, `daily_automatic_backup_start_time`, e `fsx_backup_id` não devem ser especificados. Isso corresponde à ImportedFileChunkSizepropriedade.

O tamanho de bloco padrão é `1024` (1 GiB) e pode chegar a 512.000 MiB (500 GiB). Os objetos do Amazon S3 têm um tamanho máximo de 5 TB.

```
imported_file_chunk_size = 1024
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

per_unit_storage_throughput

(Necessário para tipos de implantação **`PERSISTENT_1`**) Para o tipo de implantação `deployment_type = PERSISTENT_1`, descreve a quantidade de throughput de leitura e gravação para cada 1 tebibyte (TiB) de armazenamento, em MB/s/TiB. A capacidade de taxa de transferência do sistema de arquivos é calculada multiplicando a capacidade de armazenamento do sistema de arquivos (TiB) pela `per_unit_storage_throughput` (de) `per_unit_storage_throughput` produz 120 MB/s de taxa de transferência do sistema MB/s/TiB). For a 2.4 TiB file system,

provisioning 50 MB/s/TiB de arquivos. Você paga pela quantidade de throughput que provisiona. Isso corresponde à PerUnitStorageThroughputpropriedade.

Os valores possíveis dependem do valor da configuração `storage_type`.

`storage_type = SSD`

Os valores possíveis são 50, 100, 200.

`storage_type = HDD`

Os valores possíveis são 12, 40.

```
per_unit_storage_throughput = 200
```

> **Note**
>
> O suporte para `per_unit_storage_throughput` foi adicionado ao AWS ParallelCluster versão 2.6.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

shared_dir

(Obrigatório) Define o ponto de montagem do sistema de arquivos FSx for Lustre na cabeça e nos nós de computação.

Não use `NONE` ou `/NONE` como o diretório compartilhado.

O exemplo a seguir monta o sistema de arquivos em `/fsx`.

```
shared_dir = /fsx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

storage_capacity

(Obrigatório) Especifica a capacidade de armazenamento do sistema de arquivos, em GiB. Isso corresponde à StorageCapacitypropriedade.

Os valores possíveis da capacidade de armazenamento variam de acordo com a configuração deployment_type.

SCRATCH_1

Os valores possíveis são 1200, 2400 e qualquer múltiplo de 3600.

SCRATCH_2

Os valores possíveis são 1200 e qualquer múltiplo de 2400.

PERSISTENT_1

Os valores possíveis variam de acordo com os valores de outras configurações.

storage_type = SSD

Os valores possíveis são 1200 e qualquer múltiplo de 2400.

storage_type = HDD

Os valores possíveis variam de acordo com definição da configuração per_unit_storage_throughput.

per_unit_storage_throughput = 12

Os valores possíveis são qualquer múltiplo de 6000.

per_unit_storage_throughput = 40

Os valores possíveis são qualquer múltiplo de 1800.

```
storage_capacity = 7200
```

> **Note**
>
> Para as AWS ParallelCluster versões 2.5.0 e 2.5.1, storage_capacity suportou valores possíveis de 1200, 2400 e qualquer múltiplo de 3600. Para versões anteriores à AWS ParallelCluster versão 2.5.0, storage_capacity tinha um tamanho mínimo de 3600.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

storage_type

(Opcional) Especifica o tipo de armazenamento do sistema de arquivos. Isso corresponde à StorageTypepropriedade. Os valores possíveis são SSD e HDD. O padrão é SSD.

O tipo de armazenamento altera os valores possíveis de outras configurações.

storage_type = SSD

Especifica o tipo de armazenamento em unidade de estado sólido (SSD).

storage_type = SSD altera os valores possíveis de várias outras configurações.

drive_cache_type

Essa configuração pode ser especificada.

deployment_type

Essa configuração pode ser definida como SCRATCH_1, SCRATCH_2, ouPERSISTENT_1.

per_unit_storage_throughput

Essa configuração deve ser especificada se deployment_type for definida como PERSISTENT_1. Os valores possíveis são 50, 100, ou 200.

storage_capacity

Essa configuração deve ser especificada. Os valores possíveis variam de acordo com deployment_type.

deployment_type = SCRATCH_1

storage_capacity pode ser 1200, 2400 ou qualquer múltiplo de 3600.

deployment_type = SCRATCH_2 ou deployment_type = PERSISTENT_1

storage_capacity pode ser 1200 ou qualquer múltiplo de 2400.

storage_type = HDD

Especifica o tipo de armazenamento em unidade de disco rígido (HDD).

storage_type = HDD altera os valores possíveis de várias outras configurações.

drive_cache_type

Essa configuração pode ser especificada.

`deployment_type`

Essa configuração deve ser definida como `PERSISTENT_1`.

`per_unit_storage_throughput`

Essa configuração deve ser especificada. Os valores possíveis são 12, ou 40.

`storage_capacity`

Essa configuração deve ser especificada. Os valores possíveis variam de acordo com a configuração `per_unit_storage_throughput`.

`storage_capacity = 12`

`storage_capacity` pode ser qualquer múltiplo de 6000.

`storage_capacity = 40`

`storage_capacity` pode ser qualquer múltiplo de 1800.

```
storage_type = SSD
```

> **Note**
>
> O suporte para a configuração `storage_type` foi adicionado no AWS ParallelCluster versão 2.10.0.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

`weekly_maintenance_start_time`

(Opcional) Especifica um tempo preferencial para executar a manutenção semanal, no fuso horário UTC. Isso corresponde à WeeklyMaintenanceStartTimepropriedade.

O formato é [dia da semana]:[hora do dia]:[minuto da hora]. Por exemplo, segunda-feira à meia-noite é o seguinte.

```
weekly_maintenance_start_time = 1:00:00
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

Seção `[queue]`

Define as configurações para uma única fila. As seções `[queue]` só são suportadas quando `scheduler` está definido como `slurm`.

O formato é`[queue` *`<queue-name>`*`]`. *`queue-name`*deve começar com uma letra minúscula, conter no máximo 30 caracteres e conter somente letras minúsculas, números e hífens (-).

```
[queue q1]
compute_resource_settings = i1,i2
placement_group = DYNAMIC
enable_efa = true
disable_hyperthreading = false
compute_type = spot
```

> **Note**
>
> Support para a `[queue]`seção foi adicionado na AWS ParallelCluster versão 2.9.0.

Tópicos

`compute_resource_settings`

(Obrigatório) Identifica as seções de `[compute_resource]` que contêm as configurações dos recursos computacionais dessa fila. Os nomes das seções devem começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (_).

Até três (3) seções de `[compute_resource]` são suportadas para cada seção de `[queue]`.

Por exemplo, a seguinte configuração especifica que as seções que começam com `[compute_resource cr1]` e `[compute_resource cr2]` são usados.

```
compute_resource_settings = cr1, cr2
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

compute_type

(Opcional) Define o tipo de instâncias a serem executadas para essa fila. Essa configuração substitui a configuração `cluster_type`.

As opções válidas são `ondemand` e `spot`.

O valor padrão é `ondemand`.

Para obter mais informações sobre instâncias spot, consulte Trabalho com Instâncias spot.

> **Note**
>
> O uso de Instâncias spot exige que a função `AWSServiceRoleForEC2Spot` vinculada ao serviço exista na sua conta. Para criar essa função na sua conta usando o AWS CLI, execute o seguinte comando:
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> Para obter mais informações, consulte Função vinculada ao serviço para solicitações de instâncias spot no Guia EC2 do usuário da Amazon.

O exemplo a seguir usa SpotInstances os nós de computação nessa fila.

```
compute_type = spot
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

disable_hyperthreading

(Opcional) Desabilita o hyperthreading nos nós dessa fila. Nem todos os tipos de instância podem desabilitar o hyperthreading. Para obter uma lista de tipos de instância que suportam a desativação

do hyperthreading, consulte Núcleos e threads de CPU para cada núcleo de CPU por tipo de instância no Guia do usuário da Amazon EC2 . Se a configuração `disable_hyperthreading` na seção do `[cluster]` estiver definida, essa configuração não poderá ser definida.

O valor padrão é `false`.

```
disable_hyperthreading = true
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

enable_efa

(Opcional) Se for definida como `true`, especifica que o Elastic Fabric Adapter (EFA) está habilitado para os nós desta fila. Para ver a lista de EC2 instâncias que oferecem suporte ao EFA, consulte Tipos de instância compatíveis no Guia do EC2 usuário da Amazon para instâncias Linux. Se a configuração `enable_efa` na seção do `[cluster]` estiver definida, essa configuração não poderá ser definida. Um placement group de cluster deve ser usado para minimizar latências entre instâncias. Para obter mais informações, consulte `placement` e `placement_group`.

```
enable_efa = true
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

enable_efa_gdr

(Opcional) A partir da AWS ParallelCluster versão 2.11.3, essa configuração não tem efeito. O suporte do Elastic Fabric Adapter (EFA) para GPUDirect RDMA (acesso direto remoto à memória) está habilitado para os nós de computação e está sempre ativado se for suportado pelo tipo de instância.

> **Note**
>
> AWS ParallelCluster versão 2.10.0 a 2.11.2: Se`true`, especifica que o Elastic Fabric Adapter GPUDirect (EFA) RDMA (acesso direto remoto à memória) está habilitado para os nós nessa fila. Definir isso `true` exige que a `enable_efa` configuração seja definida como `true` .EFA GPUDirect RDMA é compatível com os seguintes tipos de instância (`p4d.24xlarge`)

nesses sistemas operacionais (`alinux2`, `centos7ubuntu1804`, ou). `ubuntu2004`
Se a configuração `enable_efa_gdr` na seção do `[cluster]` estiver definida, essa configuração não poderá ser definida. Um placement group de cluster deve ser usado para minimizar latências entre instâncias. Para obter mais informações, consulte `placement` e `placement_group`.

O valor padrão é `false`.

```
enable_efa_gdr = true
```

> **Note**
>
> Support for `enable_efa_gdr` adicionado na AWS ParallelCluster versão 2.10.0.

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

`placement_group`

(Opcional) Se presente, define o grupo de posicionamento dessa fila. Essa configuração substitui a configuração `placement_group`.

As opções válidas são as seguintes:

- `DYNAMIC`
- Um nome de grupo de posicionamento de EC2 clusters existente da Amazon

Quando definido como `DYNAMIC`, um grupo de posicionamento exclusivo será criado e excluído como parte da pilha do cluster.

Para obter mais informações sobre grupos de posicionamento, consulte Grupos de posicionamento no Guia EC2 do usuário da Amazon. Se o mesmo grupo de posicionamento for usado para diferentes tipos de instância, é mais provável que a solicitação falhe devido a um erro de capacidade insuficiente. Para obter mais informações, consulte Capacidade de instância insuficiente no Guia EC2 do usuário da Amazon.

Não há valor padrão.

Nem todos os tipos de instância oferecem suporte para placement groups de cluster. Por exemplo, o tipo de instância padrão de `t2.micro` não oferece suporte para grupos de posicionamento de cluster. Para obter informações sobre a lista de tipos de instância que oferecem suporte a grupos de posicionamento de clusters, consulte Regras e limitações de grupos de posicionamento no Guia EC2 do usuário da Amazon. Consulte Grupos de posicionamento e problemas de execução de instâncias para obter dicas ao trabalhar com placement groups.

```
placement_group = DYNAMIC
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

Seção **[raid]**

Define as definições de configuração para uma matriz RAID que é criada a partir de vários volumes idênticos do Amazon EBS. O disco RAID é montado no nó principal e é exportado para os nós de computação com o NFS.

O formato é`[raid` *`raid-name`*`]`. *`raid-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[raid rs]
shared_dir = raid
raid_type = 1
num_of_raid_volumes = 2
encrypted = true
```

Tópicos

- shared_dir
- ebs_kms_key_id
- encrypted
- num_of_raid_volumes
- raid_type
- volume_iops

- volume_size
- volume_throughput
- volume_type

shared_dir

(Obrigatório) Define o ponto de montagem para a matriz RAID nos nós principal e de computação.

A unidade RAID só é criada se esse parâmetro for especificado.

Não use `NONE` ou `/NONE` como o diretório compartilhado.

O exemplo a seguir monta a matriz em `/raid`.

```
shared_dir = raid
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ebs_kms_key_id

(Opcional) Especifica uma AWS KMS chave personalizada a ser usada para criptografia.

Esse parâmetro deve ser usado em conjunto com `encrypted = true` e deve ter um `ec2_iam_role` personalizado.

Para obter mais informações, consulte Criptografia de disco com uma chave do KMS personalizada.

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

encrypted

(Opcional) Especifica se o sistema de arquivos está criptografado.

O valor padrão é `false`.

```
encrypted = false
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

num_of_raid_volumes

(Opcional) Define o número de volumes do Amazon EBS a partir dos quais montar a matriz RAID.

Número mínimo de volumes = 2.

Número máximo de volumes = 5.

O valor padrão é 2.

```
num_of_raid_volumes = 2
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

raid_type

(Obrigatório) Define o tipo de RAID para a matriz RAID.

A unidade RAID só é criada se esse parâmetro for especificado.

As opções válidas são as seguintes:

- `0`
- `1`

Para obter mais informações sobre os tipos de RAID, consulte as informações de RAID no Guia EC2 do usuário da Amazon.

O exemplo a seguir cria uma matriz RAID `0`:

```
raid_type = 0
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

volume_iops

(Opcional) Define o número de IOPS para os volumes do tipo `io1`, `io2`, e `gp3`.

O valor padrão, os valores suportados e a proporção do `volume_iops` para `volume_size` variam de acordo com `volume_type` e `volume_size`.

`volume_type` = `io1`

Padrão `volume_iops` = 100

Valores suportados `volume_iops` = 100–64000 †

Proporção máxima de `volume_iops` para `volume_size` = 50 IOPS por GiB. 5000 IOPS exigem um `volume_size` de pelo menos 100 GiB.

`volume_type` = `io2`

Padrão `volume_iops` = 100

Valores suportados `volume_iops` = 100–64000 (256000 para volumes do `io2` Block Express) †

Proporção máxima de `volume_iops` para `volume_size` = 500 IOPS por GiB. 5000 IOPS exigem um `volume_size` de pelo menos 10 GiB.

`volume_type` = `gp3`

Padrão `volume_iops` = 3000

Valores suportados `volume_iops` = 3000–16000

Proporção máxima de `volume_iops` para `volume_size` = 500 IOPS por GiB. 5000 IOPS exigem um `volume_size` de pelo menos 10 GiB.

```
volume_iops = 3000
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

† O número máximo de IOPS é garantido somente em instâncias criadas no Nitro System provisionadas com mais de 32.000 IOPS. Outras instâncias garantem até 32.000 IOPS. Os volumes de `io1` mais antigos podem não atingir o desempenho total, a menos que você modifique o volume. `io2` Os volumes do Block Express oferecem suporte a valores `volume_iops` de até 256000. Para obter mais informações, consulte Volumes do `io2` Block Express (em versão prévia) no Guia EC2 do usuário da Amazon.

volume_size

(Opcional) Define o tamanho do volume a ser criado, em GiB.

O valor padrão e os valores suportados variam de acordo com o volume_type.

volume_type = standard

Padrão volume_size = 20 GiB

Valores suportados volume_size = 1–1024 GiB

volume_type = gp2, io1, io2, e gp3

Padrão volume_size = 20 GiB

Valores suportados volume_size = 1–16384 GiB

volume_type = sc1 e st1

Padrão volume_size = 500 GiB

Valores suportados volume_size = 500–16384 GiB

```
volume_size = 20
```

> **Note**
>
> Antes da AWS ParallelCluster versão 2.10.1, o valor padrão para todos os tipos de volume era de 20 GiB.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

volume_throughput

(Opcional) Define o throughput para tipos de volume gp3, em MiB/s.

O valor padrão é 125.

Valores suportados volume_throughput = 125–1000 MiB/s

A proporção de `volume_throughput` para `volume_iops` não pode ser superior a 0,25. O throughput máximo de 1000 MiB/s exige que a configuração `volume_iops` seja de pelo menos 4000.

```
volume_throughput = 1000
```

> **Note**
>
> Support for `volume_throughput` adicionado na AWS ParallelCluster versão 2.10.1.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

volume_type

(Opcional) Define o tipo de volume a ser criado.

As opções válidas são as seguintes:

`gp2`, `gp3`

General purpose SSD

`io1`, `io2`

Provisioned IOPS SSD

`st1`

HDD otimizado para throughput

`sc1`

Disco rígido frio

`standard`

Geração magnética anterior

Para obter mais informações, consulte os tipos de volume do Amazon EBS no Guia do EC2 usuário da Amazon.

O valor padrão é gp2.

```
volume_type = io2
```

> **Note**
>
> Support gp3 e io2 foi adicionado na AWS ParallelCluster versão 2.10.1.

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

Seção **[scaling]**

Tópicos

- scaledown_idletime

Especifica configurações que definem como dimensionar os nós de computação.

O formato é`[scaling` *`scaling-name`*`]`. *`scaling-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[scaling custom]
scaledown_idletime = 10
```

scaledown_idletime

(Opcional) Especifica a quantidade de tempo em minutos sem uma tarefa, após a qual o nó de computação será encerrado.

Esse parâmetro não é usado se `awsbatch` for o programador.

O valor padrão é `10`.

```
scaledown_idletime = 10
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

Seção `[vpc]`

Especifica as configurações do Amazon VPC. Para obter mais informações sobre VPCs, consulte O que é Amazon VPC? e as melhores práticas de segurança para sua VPC no Guia do usuário da Amazon VPC.

O formato é`[vpc` *`vpc-name`*`]`. *`vpc-name`*deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífens (-) e sublinhados (_).

```
[vpc public]
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-xxxxxx
```

Tópicos

`additional_sg`

(Opcional) Fornece um grupo de segurança da Amazon VPC adicional para todas as instâncias.

Não há valor padrão.

```
additional_sg = sg-xxxxxx
```

`compute_subnet_cidr`

(Opcional) Especifica um bloco de Encaminhamento Entre Domínios Sem Classificação (CIDR). Use esse parâmetro se quiser AWS ParallelCluster criar uma sub-rede de computação.

```
compute_subnet_cidr = 10.0.100.0/24
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

compute_subnet_id

(Opcional) Especifica o ID de uma sub-rede existente na qual os nós de computação serão provisionados.

Se não for especificado, `compute_subnet_id` usará o valor de `master_subnet_id`.

Se a sub-rede for privada, é necessário configurar o NAT para acesso à web.

```
compute_subnet_id = subnet-xxxxxx
```

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

master_subnet_id

(Obrigatório) Especifica o ID de uma sub-rede existente na qual o servidor provisiona o nó principal.

```
master_subnet_id = subnet-xxxxxx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

ssh_from

(Opcional) Especifica o intervalo de IP formatado em CIDR a partir do qual será permitido o acesso SSH.

Esse parâmetro é usado somente ao AWS ParallelCluster criar o grupo de segurança.

O valor padrão é `0.0.0.0/0`.

```
ssh_from = 0.0.0.0/0
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

use_public_ips

(Opcional) Define se deseja atribuir endereços IP públicos a instâncias de computação.

Se definido como `true`, um endereço IP elástico é associado ao nó principal.

Se definido como `false`, o nó principal terá um IP público (ou não) de acordo com o valor do parâmetro de configuração da sub-rede "Atribuir IP público automaticamente".

Para obter exemplos, consulte a configuração de redes.

O valor padrão é `true`.

```
use_public_ips = true
```

> **Important**
>
> Por padrão, todos Contas da AWS estão limitados a cinco (5) endereços IP elásticos para cada um Região da AWS. Para obter mais informações, consulte Limite de endereço IP elástico no Guia EC2 do usuário da Amazon.

Política de atualização: a frota de computação deve ser interrompida para que essa configuração seja alterada para uma atualização.

vpc_id

(Obrigatório) Especifica o ID do Amazon VPC onde o cluster deve ser provisionado.

```
vpc_id = vpc-xxxxxx
```

Política de atualização: se essa configuração for alterada, a atualização não será permitida.

vpc_security_group_id

(Opcional) Especifica o uso de um grupo de segurança para todas as instâncias.

Não há valor padrão.

```
vpc_security_group_id = sg-xxxxxx
```

O grupo de segurança criado por AWS ParallelCluster permite acesso SSH usando a porta 22 dos endereços especificados na `ssh_from` configuração ou todos os IPv4 endereços (`0.0.0.0/0`) se a `ssh_from` configuração não for especificada. Se o Amazon DCV estiver ativado, o grupo de segurança permitirá o acesso ao Amazon DCV usando a porta 8443 (ou qualquer outra especificação da `port` configuração) a partir dos endereços especificados na `access_from` configuração ou de todos os IPv4 endereços (`0.0.0.0/0`) se a `access_from` configuração não for especificada.

Warning

Você pode alterar o valor desse parâmetro e atualizar o cluster se `[cluster]fsx_settings`não for especificado ou ambos `fsx_settings` e se um sistema de arquivos externo existente FSx para Lustre estiver especificado `fsx-fs-id`em `[fsx fs]`.

Você não pode alterar o valor desse parâmetro se um sistema de arquivos AWS ParallelCluster gerenciado FSx para Lustre estiver especificado em `fsx_settings` e. `[fsx fs]`

Política de atualização: se os sistemas de arquivos AWS ParallelCluster gerenciados do Amazon FSx for Lustre não estiverem especificados na configuração, essa configuração poderá ser alterada durante uma atualização.

Exemplos

Os exemplos de configurações a seguir demonstram AWS ParallelCluster configurações usando Slurm, Torquee AWS Batch agendadores.

Note

A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE or Torque agendadores.

Sumário

- Slurm Workload Manager (slurm)
- Son of Grid Engine (sge) e Torque Resource Manager (torque)
- AWS Batch (awsbatch)

Slurm Workload Manager (`slurm`)

O exemplo a seguir executa um cluster com o programador `slurm`. O exemplo de configuração inicia 1 cluster com 2 filas de trabalhos. A primeira fila, `spot`, tem inicialmente duas instâncias spot `t3.micro` disponíveis. Ele pode ser expandido até um máximo de 10 instâncias e reduzido para um mínimo de 1 instância quando nenhum trabalho for executado por 10 minutos (ajustável usando a configuração `scaledown_idletime`). A segunda fila, `ondemand`, começa sem instâncias e pode ser escalada até um máximo de 5 instâncias `t3.micro` sob demanda.

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

[aws]
aws_region_name = <your Região da 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`) e Torque Resource Manager (`torque`)

> **Note**
>
> Este exemplo se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE or Torque agendadores.

O exemplo a seguir executa um cluster com o programador `torque` ou `sge`. Para usar SGE, `scheduler = torque` mude para`scheduler = sge`. O exemplo de configuração a seguir permite um máximo de 5 nós simultâneos e reduz a dois quando nenhuma tarefa for executada por 10 minutos.

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

[aws]
aws_region_name = <your Região da 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**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE or Torque agendadores. Se você usar essas versões, poderá continuar usando-as ou solucionando problemas de suporte das equipes de AWS serviço e AWS Support.

AWS Batch (**`awsbatch`**)

O exemplo a seguir executa um cluster com o programador `awsbatch`. Ele é definido para escolher o tipo de instância melhor, com base em suas necessidades de recursos de tarefa.

A configuração de exemplo permite um máximo de 40 v CPUs simultâneos e é reduzida para zero quando nenhum trabalho é executado por 10 minutos (ajustável usando a [`scaledown_idletime`] configuração).

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

[aws]
aws_region_name = <your Região da 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
```

Como AWS ParallelCluster funciona

AWS ParallelCluster foi criado não apenas como uma forma de gerenciar clusters, mas como uma referência sobre como usar AWS serviços para criar seu ambiente de HPC.

Tópicos

AWS ParallelCluster processos

Esta seção se aplica somente aos clusters de HPC que são implantados com um dos agendadores de tarefas tradicionais compatíveis (SGE, Slurm ou Torque). Quando usado com esses agendadores, AWS ParallelCluster gerencia o provisionamento e a remoção de nós de computação interagindo com o grupo Auto Scaling e com o agendador de tarefas subjacente.

Para clusters de HPC baseados em AWS Batch, AWS ParallelCluster depende dos recursos fornecidos pelo AWS Batch para o gerenciamento de nós de computação.

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE or Torque agendadores. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.

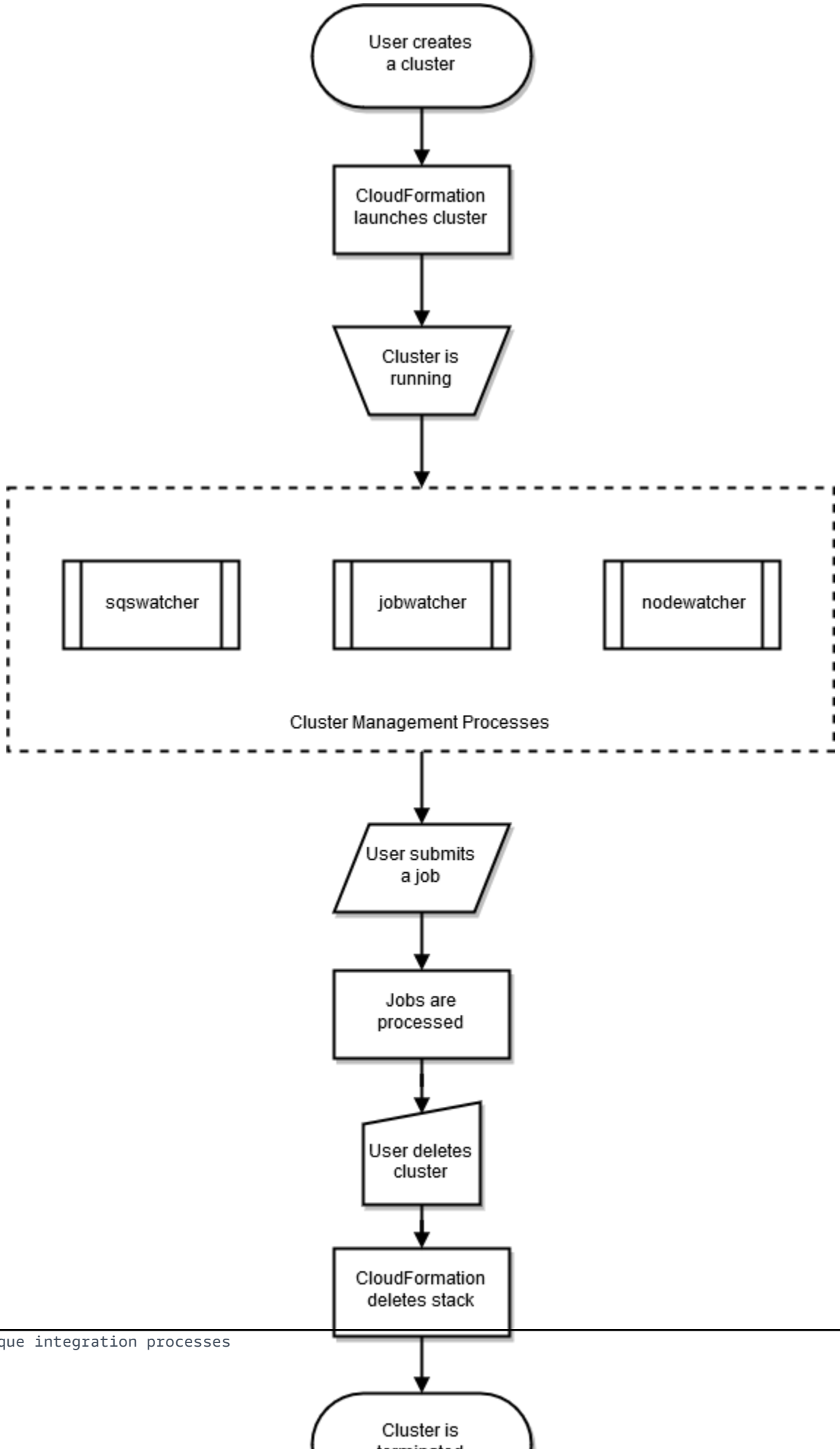
Tópicos

SGE and Torque integration processes

> **Note**
>
> Esta seção se aplica somente às AWS ParallelCluster versões até e incluindo a versão 2.11.4. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE and Torque agendadores, Amazon SNS e Amazon SQS.

Visão geral

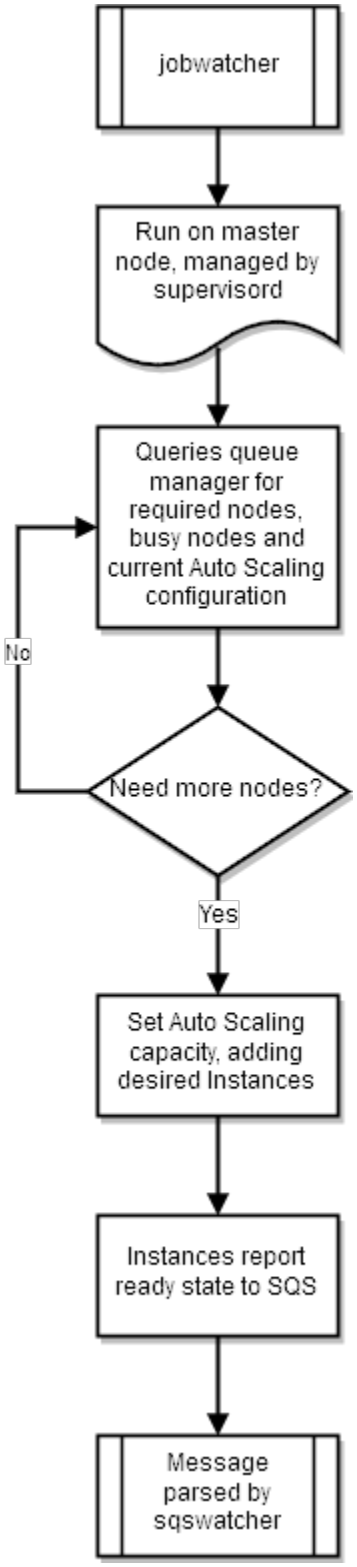
O ciclo de vida de um cluster começa após ele ser criado por um usuário. Normalmente, um cluster é criado a partir da interface de linha de comando (CLI). Depois de criado, um cluster existe até ser excluído. AWS ParallelCluster os daemons são executados nos nós do cluster, principalmente para gerenciar a elasticidade do cluster HPC. O diagrama a seguir mostra um fluxo de trabalho do usuário e o ciclo de vida do cluster. As seções a seguir descrevem os AWS ParallelCluster daemons usados para gerenciar o cluster.


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

With SGE and Torque agendadores `nodewatcher`, `jobwatcher` AWS ParallelCluster usos e `sqswatcher` processos.

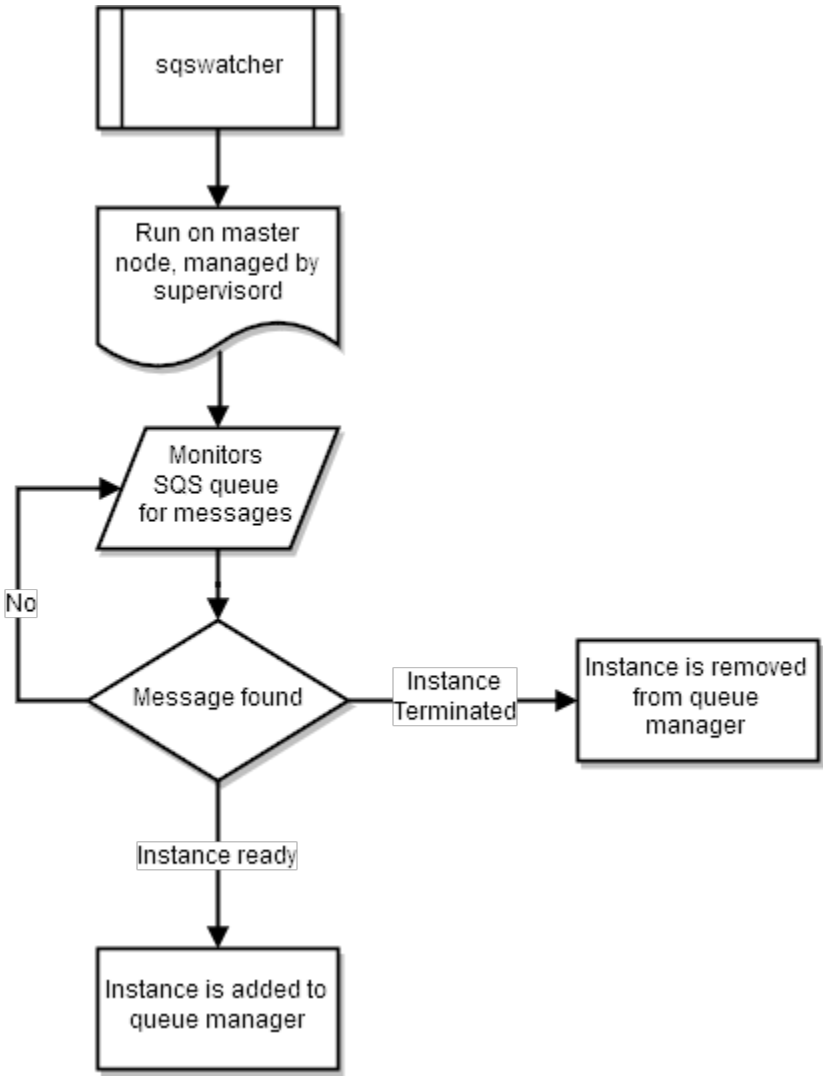
`jobwatcher`

Quando um cluster está em execução, um processo de propriedade do usuário root monitora o agendador configurado (SGE or Torque). A cada minuto, ele avalia a fila para decidir quando aumentar a escala.


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

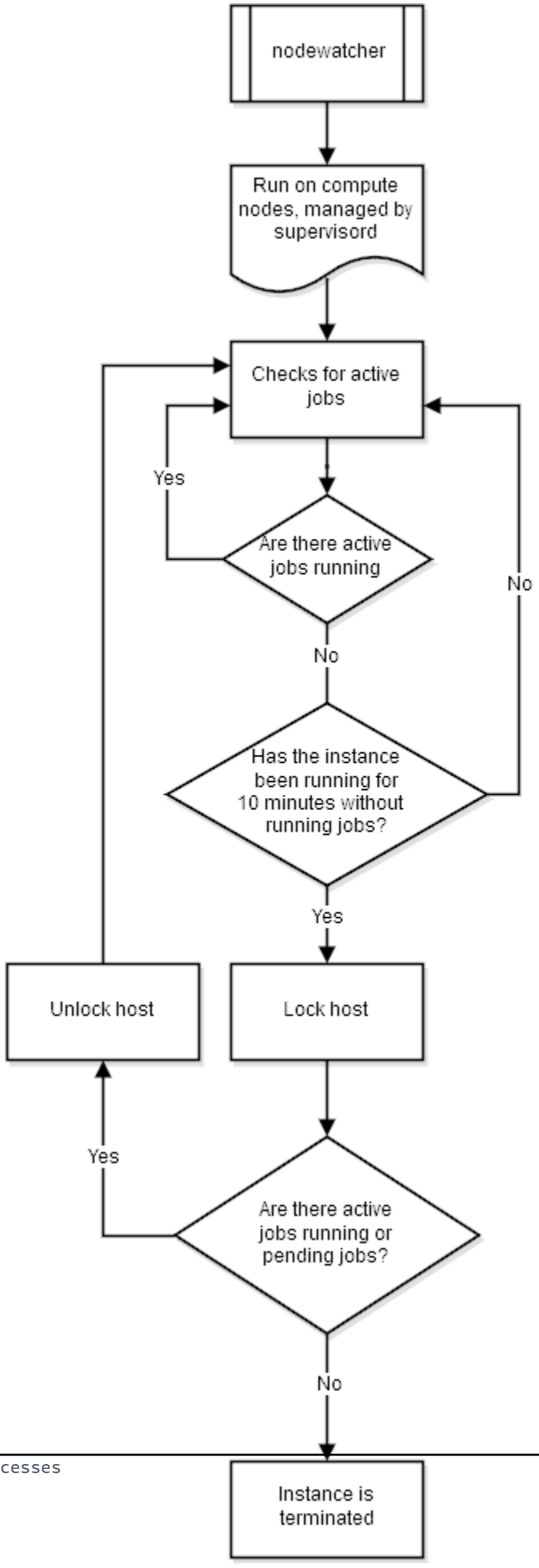
sqswatcher

O processo `sqswatcher` monitora mensagens do Amazon SQS que são enviadas por ajuste de escala automático, a fim de notificar você sobre as alterações de estado no cluster. Quando uma instância fica online, ela envia uma mensagem "instância pronta" ao Amazon SQS. Essa mensagem é capturada por `sqs_watcher`, em execução no nó principal. Essas mensagens são usadas para notificar o gerenciador da fila quando novas instâncias ficam online ou são encerradas, para que elas possam ser adicionadas ou removidas da fila.



nodewatcher

O processo `nodewatcher` é executado em cada nó da frota de computação. Após o período de `scaledown_idletime`, conforme definido pelo usuário, a instância é encerrada.


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

With Slurm agendadores, AWS ParallelCluster usos `clustermgtd` e `computemgt` processos.

clustermgtd

Clusters executados em modo heterogêneo (indicado pela especificação de um valor `queue_settings`) têm um processo daemon (`clustermgtd`) de gerenciamento de cluster executado no nó principal. Estas tarefas são executadas pelo daemon de gerenciamento de cluster.

- Limpeza de partições inativas
- Gerenciamento de capacidade estática: certifique-se de que a capacidade estática esteja sempre ativa e saudável
- Sincronize o agendador com a Amazon EC2.
- Limpeza de instâncias órfãs
- Restaure o status do nó do agendador na EC2 rescisão da Amazon que ocorre fora do fluxo de trabalho de suspensão
- Gerenciamento de EC2 instâncias insalubres da Amazon (falha nas verificações de EC2 saúde da Amazon)
- Gerenciamento de eventos de manutenção programados
- Gerenciamento de nós não íntegros do Scheduler (falha nas verificações de integridade do Scheduler)

computemgtd

Clusters executados em modo heterogêneo (indicado pela especificação de um valor `queue_settings`) têm um processo daemon (`computemgtd`) de gerenciamento de computação executado em cada nó de computação. A cada cinco (5) minutos, o daemon de gerenciamento de computação confirma que o nó principal pode ser alcançado e está íntegro. Se passarem cinco (5) minutos durante os quais o nó principal não puder ser alcançado ou não estiver íntegro, o nó de computação será encerrado.

AWS serviços usados por AWS ParallelCluster

Os seguintes serviços da Amazon Web Services (AWS) são usados por AWS ParallelCluster.

Tópicos

- AWS Auto Scaling
- AWS Batch
- CloudFormation
- Amazon CloudWatch
- CloudWatch Registros da Amazon
- AWS CodeBuild
- Amazon DynamoDB
- Amazon Elastic Block Store
- Amazon Elastic Compute Cloud
- Amazon Elastic Container Registry
- Amazon EFS
- Amazon FSx para 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**
>
> Esta seção se aplica somente às AWS ParallelCluster versões até e incluindo a versão 2.11.4. A partir da versão 2.11.5,AWS ParallelCluster não suporta o uso de.AWS Auto Scaling

AWS Auto Scalingé um serviço que monitora seus aplicativos e ajusta automaticamente a capacidade com base em seus requisitos de serviço específicos e variáveis. Esse serviço gerencia suas ComputeFleet instâncias como um grupo de Auto Scaling. O grupo pode ser conduzido de

forma elástica pela mudança da workload ou fixado estaticamente pelas configurações iniciais da instância.

AWS Auto Scalingé usado com ComputeFleet instâncias, mas não é usado com AWS Batch clusters.

Para obter mais informações sobre AWS Auto Scaling, consulte https://aws.amazon.com/autoscaling/https://docs.aws.amazon.com/autoscaling/e.

AWS Batch

AWS Batché um serviço AWS gerenciado de agendamento de tarefas. Ele provisiona dinamicamente a quantidade e o tipo ideais de recursos computacionais (por exemplo, instâncias otimizadas para CPU ou memória) em clusters.AWS Batch Esses recursos são provisionados com base nos requisitos específicos de seus trabalhos em lotes, incluindo os requisitos de volume. Com AWS Batch, você não precisa instalar ou gerenciar software de computação em lote adicional ou clusters de servidores para executar suas tarefas com eficiência.

AWS Batché usado somente com AWS Batch clusters.

Para obter mais informações sobre AWS Batch, consulte https://aws.amazon.com/batch/https://docs.aws.amazon.com/batch/e.

CloudFormation

CloudFormationé um infrastructure-as-code serviço que fornece uma linguagem comum para modelar AWS e provisionar recursos de aplicativos de terceiros em seu ambiente de nuvem. É o principal serviço usado pelo AWS ParallelCluster. Cada cluster em AWS ParallelClusteré representado como uma pilha, e todos os recursos exigidos por cada cluster são definidos no AWS ParallelClusterCloudFormation modelo. Na maioria dos casos, os comandos da AWS ParallelCluster CLI correspondem diretamente aos comandos de CloudFormation pilha, como os comandos create, update e delete. As instâncias que são executadas em um cluster fazem chamadas HTTPS para o CloudFormation endpoint em Região da AWS que o cluster é executado.

Para obter mais informações sobre CloudFormation, consulte https://aws.amazon.com/cloudformation/https://docs.aws.amazon.com/cloudformation/e.

Amazon CloudWatch

O Amazon CloudWatch (CloudWatch) é um serviço de monitoramento e observabilidade que fornece dados e insights acionáveis. Esses insights podem ser usados para monitorar seus aplicativos, responder a mudanças de desempenho e exceções de serviço e otimizar a utilização de recursos. In

AWS ParallelCluster, CloudWatch é usado para um painel, para monitorar e registrar as etapas de criação da imagem do Docker e a saída dos AWS Batch trabalhos.

Antes da AWS ParallelCluster versão 2.10.0, CloudWatch era usado somente com AWS Batch clusters.

Para obter mais informações sobre CloudWatch, consulte https://aws.amazon.com/cloudwatch/https://docs.aws.amazon.com/cloudwatch/e.

CloudWatch Registros da Amazon

O Amazon CloudWatch CloudWatch Logs (Logs) é um dos principais recursos da Amazon CloudWatch. Você pode usá-lo para monitorar, armazenar, exibir e pesquisar os arquivos de log de diversos componentes usados pelo AWS ParallelCluster.

Antes da AWS ParallelCluster versão 2.6.0, o CloudWatch Logs era usado apenas com AWS Batch clusters.

Para obter mais informações, consulte Integração com Amazon CloudWatch Logs.

AWS CodeBuild

AWS CodeBuild(CodeBuild) é um serviço AWS gerenciado de integração contínua que cumpre o código-fonte, executa testes e produz pacotes de software prontos para implantação. In AWS ParallelCluster, CodeBuild é usado para criar imagens do Docker de forma automática e transparente quando os clusters são criados.

CodeBuild é usado somente com AWS Batch clusters.

Para obter mais informações sobre CodeBuild, consulte https://aws.amazon.com/codebuild/https://docs.aws.amazon.com/codebuild/e.

Amazon DynamoDB

O Amazon DynamoDB (DynamoDB) é um serviço de banco de dados NoSQL rápido e flexível. É usado para armazenar o estado mínimo de informações do cluster. O nó principal rastreia as instâncias provisionadas em uma tabela do DynamoDB.

O DynamoDB não é usado com clusters.AWS Batch

Para obter mais informações sobre o DynamoDB, consulte e. https://aws.amazon.com/dynamodb/https://docs.aws.amazon.com/dynamodb/

Amazon Elastic Block Store

O Amazon Elastic Block Store (Amazon EBS) é um serviço de armazenamento em bloco de alto desempenho que oferece armazenamento persistente para volumes compartilhados. Todas as configurações do Amazon EBS podem ser passadas pela configuração. Os volumes do Amazon EBS podem ser inicializados vazios ou a partir de um snapshot existente do Amazon EBS.

Para obter mais informações sobre o Amazon EBS, consulte https://aws.amazon.com/ebs/e. https://docs.aws.amazon.com/ebs/

Amazon Elastic Compute Cloud

O Amazon Elastic Compute Cloud (Amazon EC2) fornece a capacidade de computação para AWS ParallelCluster. Os nós principais e de computação são EC2 instâncias da Amazon. Qualquer tipo de instância que ofereça suporte ao HVM pode ser selecionada. Os nós principais e de computação podem ser tipos de instância diferentes. Além disso, se várias filas forem usadas, alguns ou todos os nós de computação também poderão ser executados como uma instância spot. Os volumes de armazenamento de instâncias encontrados nas instâncias são montados como volumes distribuídos do LVM.

Para obter mais informações sobre a Amazon EC2, consulte https://aws.amazon.com/ec2/https://docs.aws.amazon.com/ec2/e.

Amazon Elastic Container Registry

O Amazon Elastic Container Registry (Amazon ECR) é um registro de contêiner do Docker totalmente gerenciado que facilita o armazenamento, o gerenciamento e a implantação de imagens de contêiner do Docker. Em AWS ParallelCluster, o Amazon ECR armazena as imagens do Docker que são criadas quando os clusters são criados. As imagens do Docker são então usadas AWS Batch para executar os contêineres dos trabalhos enviados.

O Amazon ECR é usado somente com AWS Batch clusters.

Para obter mais informações, consulte https://aws.amazon.com/ecr/ e https://docs.aws.amazon.com/ecr/.

Amazon EFS

O Amazon Elastic File System (Amazon EFS) é um sistema de arquivos NFS simples, escalável e elástico totalmente gerenciado para uso com serviços e recursos on-premises da Nuvem AWS. O

Amazon EFS é usado quando a configuração do `efs_settings` é especificada e se refere a uma `[efs]`seção. O suporte para o Amazon EFS foi adicionado na AWS ParallelCluster versão 2.1.0.

Para obter mais informações sobre o Amazon EFS, consulte https://aws.amazon.com/efs/https://docs.aws.amazon.com/efs/e.

Amazon FSx para Lustre

FSx for Lustre fornece um sistema de arquivos de alto desempenho que usa o sistema de arquivos Lustre de código aberto. FSx for Lustre é usado quando a `fsx_settings` configuração é especificada e se refere a uma `[fsx]`seção. Support FSx for Lustre foi adicionado na AWS ParallelCluster versão 2.2.1.

Para obter mais informações sobre o FSx Lustre, consulte https://aws.amazon.com/fsx/lustre/ e. https://docs.aws.amazon.com/fsx/

AWS Identity and Access Management

AWS Identity and Access Management(IAM) é usado AWS ParallelCluster para fornecer uma função IAM menos privilegiada para a Amazon EC2 para a instância específica de cada cluster individual.AWS ParallelCluster as instâncias têm acesso somente às chamadas de API específicas necessárias para implantar e gerenciar o cluster.

Com AWS Batch os clusters, as funções do IAM também são criadas para os componentes envolvidos no processo de criação de imagens do Docker quando os clusters são criados. Esses componentes incluem as funções do Lambda que têm permissão para adicionar e excluir imagens do Docker de e para o repositório Amazon ECR. Eles também incluem as funções permitidas para excluir o bucket do Amazon S3 criado para o cluster e CodeBuild o projeto. Também há funções para AWS Batch recursos, instâncias e trabalhos.

Para obter mais informações sobre o IAM, consulte https://aws.amazon.com/iam/https://docs.aws.amazon.com/iam/e.

AWS Lambda

AWS Lambda(Lambda) executa as funções que orquestram a criação de imagens do Docker. O Lambda também gerencia a limpeza de recursos de cluster personalizado, como imagens do Docker armazenadas no repositório do Amazon ECR e no Amazon S3.

Para obter mais informações sobre o Lambda, consulte e. https://aws.amazon.com/lambda/https://docs.aws.amazon.com/lambda/

Amazon DCV

O Amazon DCV é um protocolo de exibição remota de alto desempenho que permite fornecer áreas de trabalho remotas e streaming de aplicações de maneira segura para qualquer dispositivo, mesmo em condições variadas de rede. O Amazon DCV é usado quando a configuração `dcv_settings` é especificada e se refere a uma seção `[dcv]`. O suporte para Amazon DCV foi adicionado na AWS ParallelCluster versão 2.5.0.

Para obter mais informações sobre o Amazon DCV, consulte https://aws.amazon.com/hpc/dcv/ e. https://docs.aws.amazon.com/dcv/

Amazon Route 53

O Amazon Route 53 (Route 53) é usado para criar zonas hospedadas com nomes de host e nomes de domínio totalmente qualificados para cada um dos nós de computação.

Para obter mais informações sobre o Route 53, consulte https://aws.amazon.com/route53/https://docs.aws.amazon.com/route53/e.

Amazon Simple Notification Service

> **Note**
>
> Esta seção se aplica somente às AWS ParallelCluster versões até e incluindo a versão 2.11.4. A partir da versão 2.11.5,AWS ParallelCluster não oferece suporte ao uso do Amazon Simple Notification Service.

O Amazon Simple Notification Service (Amazon SNS) recebe notificações do Auto Scaling. Esses eventos são chamados de eventos de ciclo de vida e são gerados quando uma instância é executada ou encerrada em um grupo do Auto Scaling. Dentro AWS ParallelCluster, o tópico do Amazon SNS para o grupo Auto Scaling é inscrito em uma fila do Amazon SQS.

O Amazon SNS não é usado com AWS Batch clusters.

Para obter mais informações sobre o Amazon SNS, consulte e. https://aws.amazon.com/sns/https://docs.aws.amazon.com/sns/

Amazon Simple Queue Service

> **Note**
>
> Esta seção se aplica somente às AWS ParallelCluster versões até e incluindo a versão 2.11.4. A partir da versão 2.11.5,AWS ParallelCluster não oferece suporte ao uso do Amazon Simple Queue Service.

O Amazon Simple Queue Service (Amazon SQS) retém as notificações enviadas pelo Auto Scaling, as notificações enviadas pelo Amazon SNS e as notificações enviadas dos nós de computação. O Amazon SQS separa o envio de notificações do recebimento de notificações. Isso permite que o nó principal manipule as notificações por meio de um processo de pesquisa. Nesse processo, o nó principal executa a Amazon SQSwatcher e pesquisa a fila. O Auto Scaling e os nós de computação publicam mensagens na fila.

O Amazon SQS não é usado com AWS Batch clusters.

Para obter mais informações sobre o Amazon SQS, consulte e. https://aws.amazon.com/sqs/https://docs.aws.amazon.com/sqs/

Amazon Simple Storage Service

O Amazon Simple Storage Service (Amazon S3)AWS ParallelCluster armazena modelos localizados em cada um.Região da AWSAWS ParallelCluster pode ser configurado para permitir que CLI/SDK as ferramentas usem o Amazon S3.

Quando você usa o AWS Batch cluster, um bucket do Amazon S3 em sua conta é usado para armazenar dados relacionados. Por exemplo, o bucket armazena os artefatos usados pela criação de imagens do Docker e scripts de tarefas enviadas.

Para obter mais informações, consulte https://aws.amazon.com/s3/ e https://docs.aws.amazon.com/s3/.

Amazon VPC

O Amazon VPC define uma rede usada pelos nós em seu cluster. As configurações de VPC para o cluster são definidas na `[vpc]` seção.

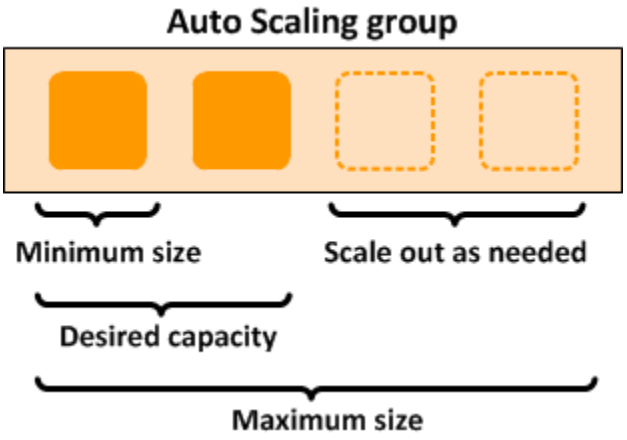
Para obter mais informações sobre a Amazon VPC, consulte e. https://aws.amazon.com/vpc/https://docs.aws.amazon.com/vpc/

AWS ParallelCluster Auto Scaling

> **Note**
>
> Esta seção se aplica somente às AWS ParallelCluster versões até e incluindo a versão 2.11.4. A partir da versão 2.11.5,AWS ParallelCluster não suporta o uso de programadores SGE ou Torque. Você pode continuar usando-os nas versões até a 2.11.4, inclusive, mas eles não estão qualificados para futuras atualizações ou suporte para solução de problemas das equipes de AWS serviço e de AWS suporte.
> A partir da AWS ParallelCluster versão 2.9.0, o Auto Scaling não é suportado para uso Slurm Workload Manager com (). Slurm Para saber mais sobre Slurm o dimensionamento de várias filas, consulte o. Tutorial do modo de fila múltipla

A estratégia de ajuste de escala automático descrita neste tópico se aplica aos clusters de HPC implantados com o Son of Grid Engine (SGE) ou o Torque Resource Manager (Torque). Quando implantado com um desses programadores, o AWS ParallelCluster implementa os recursos de escalabilidade gerenciando o Grupo do Auto Scaling (ASG) dos nós de computação e alterando a configuração do programador conforme necessário. Para clusters de HPC baseados em AWS Batch,AWS ParallelCluster depende dos recursos de escalabilidade elástica fornecidos pelo agendador de tarefas AWS gerenciado. Para obter mais informações, consulte O que é o Amazon EC2 Auto Scaling no Guia do usuário do Amazon Auto EC2 Scaling.

Os clusters implantados com AWS ParallelCluster são elásticos de várias maneiras. A configuração `initial_queue_size` especifica o valor mínimo do tamanho do grupo ComputeFleet Auto Scaling e também o valor de capacidade desejado. Definir o `max_queue_size` especifica o valor máximo do tamanho do grupo ComputeFleet Auto Scaling.



Aumentar a escala verticalmente

A cada minuto, um processo chamado `jobwatcher` é executado no nó principal. Ele avalia o número atual de instâncias exigidas pelas tarefas pendentes na fila. Se o número total de nós ocupados e nós solicitados for maior que o valor desejado atual no grupo do Auto Scaling, ele adicionará mais instâncias. Se você enviar mais tarefas, a fila será reavaliada e o grupo do Auto Scaling será atualizado, até o `max_queue_size` especificado.

Com um programador SGE, cada trabalho exige um número de slots para ser executado (um slot corresponde a uma unidade de processamento, por exemplo, uma vCPU). Para avaliar o número de instâncias necessárias para atender às tarefas pendentes atualmente, o `jobwatcher` divide o número total de slots solicitados pela capacidade de um único nó de computação. A capacidade de um nó computacional que corresponde ao número de v disponíveis CPUs depende do tipo de EC2 instância da Amazon especificado na configuração do cluster.

Com Slurm (antes da AWS ParallelCluster versão 2.9.0) e Torque programadores, cada trabalho pode exigir vários nós e vários slots para cada nó, dependendo das circunstâncias. Para cada solicitação, o `jobwatcher` determina o número de nós de computação necessários para cumprir os novos requisitos de computação. Por exemplo, vamos assumir um cluster com `c5.2xlarge` (8 vCPUs) como o tipo de instância de computação e três tarefas pendentes na fila com os seguintes requisitos:

- job1: 2 nós/4 slots cada
- job2: 3 nós/2 slots cada
- job3: 1 nó/4 slots cada

Neste exemplo, o `jobwatcher` requer três novas instâncias de computação no grupo do Auto Scaling para atender aos três trabalhos.

Limitação atual: a lógica de aumento automático de escala não considera nós ocupados parcialmente carregados. Por exemplo, um nó que está executando um trabalho é considerado ocupado mesmo se houver slots vazios.

Reduzir a escala verticalmente

Em cada nó de computação, um processo chamado `nodewatcher` executa e avalia o tempo de ociosidade do nó. Uma instância é encerrada quando ambas as condições a seguir são atendidas:

- Uma instância não tem tarefas para um período maior do que o `scaledown_idletime` (a configuração padrão é 10 minutos)
- Não há tarefas pendentes no cluster

Para encerrar uma instância, `nodewatcher` chama a operação de TerminateInstanceInAutoScalingGroupAPI, que remove uma instância se o tamanho do grupo de Auto Scaling for pelo menos o tamanho mínimo do grupo de Auto Scaling. Esse processo diminui um cluster sem afetar as tarefas em execução. Também habilita um cluster elástico, com um número fixo de instâncias de base.

Cluster estático

O valor de escalabilidade automática é o mesmo para HPC como com qualquer outra workload. A única diferença é que o AWS ParallelCluster tem código que permite interagir com mais inteligência. Por exemplo, se um cluster estático for necessário, defina os parâmetros `initial_queue_size` e `max_queue_size` como o tamanho exato de cluster que é necessário e defina o parâmetro `maintain_initial_size` como verdadeiro. Isso faz com que o grupo ComputeFleet Auto Scaling tenha o mesmo valor para a capacidade mínima, máxima e desejada.

Tutoriais

Os tutoriais a seguir mostram como começar e fornecem orientações sobre as melhores práticas para algumas tarefas comuns. AWS ParallelCluster

Tópicos

Executando seu primeiro trabalho em AWS ParallelCluster

Este tutorial explica como executar seu primeiro trabalho no Hello World em AWS ParallelCluster.

Pré-requisitos

- AWS ParallelCluster está instalado.
- O AWS CLI está instalado e configurado.
- Você tem um par de EC2 chaves.
- Você tem um perfil do IAM com as permissões necessárias para executar a CLI `pcluster`.

Verificar a instalação

Primeiro, verificamos se AWS ParallelCluster está instalado e configurado corretamente.

```
$ pcluster version
```

Isso retorna a versão em execução do AWS ParallelCluster. Se a saída fornecer uma mensagem sobre configuração, será necessário executar o seguinte para configurar o AWS ParallelCluster:

```
$ pcluster configure
```

Criação de seu primeiro cluster

Agora é hora de criar seu primeiro cluster. Como a workload desse tutorial não é de desempenho intensivo, podemos usar o tamanho de instância padrão de `t2.micro`. (Para workloads de produção, escolha um tamanho de instância que melhor atenda às suas necessidades.)

Vamos chamar o cluster de hello-world.

```
$ pcluster create hello-world
```

Quando o cluster for criado, você verá uma saída semelhante à seguinte:

```
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/
```

A mensagem `CREATE_COMPLETE` mostra que o cluster foi criado com êxito. A saída também fornece os endereços IP público e privado do nosso nó principal. Esse IP é necessário para fazer login.

Fazer login em seu nó principal

Use o arquivo pem OpenSSH para fazer login no nó principal.

```
pcluster ssh hello-world -i /path/to/keyfile.pem
```

Depois de fazer login, execute o comando `qhost` para verificar se os nós de computação estão definidos e configurados.

```
$ 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
```

A saída mostra que há dois nós de computação no cluster, ambos com duas threads disponíveis para eles.

Executando seu primeiro trabalho usando SGE

> **Note**
>
> Este exemplo se aplica somente às AWS ParallelCluster versões até a versão 2.11.4, inclusive. A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de SGE or Torque agendadores.

Depois, criamos uma tarefa que permanece em espera por um tempo e então emite seu próprio nome de host como saída.

Crie um arquivo chamado `hellojob.sh` com o seguinte conteúdo:

```
#!/bin/bash
sleep 30
echo "Hello World from $(hostname)"
```

Depois, envie a tarefa usando `qsub` e verifique se ela é executada.

```
$ qsub hellojob.sh
Your job 1 ("hellojob.sh") has been submitted
```

Agora, você pode visualizar a fila e verificar o status do trabalho.

```
$ 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
```

A saída mostra que, no momento, a tarefa está em um estado de execução. Aguarde 30 segundos para que a tarefa seja concluída e execute `qstat` novamente.

```
$ qstat
$
```

Agora que não há trabalhos na fila, podemos verificar a saída em nosso diretório atual.

```
$ 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
```

Na saída, vemos um arquivo “e1” e um “o1” no script da tarefa. Como o arquivo `e1` está vazio, não houve saída para stderr. Se visualizarmos o arquivo `o1`, é possível ver a saída da tarefa.

```
$ cat hellojob.sh.o1
Hello World from ip-192-168-1-125
```

A saída também mostra que a tarefa foi executada com êxito na instância `ip-192-168-1-125`.

Para saber mais sobre como criar e usar clusters, consulte Práticas recomendadas.

Criação de uma AWS ParallelCluster AMI personalizada

> **Important**
>
> Não recomendamos a criação de uma AMI personalizada como abordagem para personalizar o AWS ParallelCluster.
> Isso ocorre porque, depois de criar sua própria AMI, você não recebe mais atualizações ou correções de bugs com versões futuras do AWS ParallelCluster. Além disso, se você criar uma AMI personalizada, deverá repetir as etapas usadas para criar sua AMI personalizada a cada nova AWS ParallelCluster versão.

Antes de continuar lendo, recomendamos que você primeiro confira a seção Ações personalizadas do Bootstrap para determinar se as modificações que você deseja fazer podem ser programadas e suportadas em versões futuras AWS ParallelCluster .

Embora criar uma AMI personalizada não seja o ideal (devido aos motivos mencionados anteriormente), ainda existem cenários em que AWS ParallelCluster é necessário criar uma AMI

personalizada. Este tutorial orienta você pelo processo de criação de uma AMI personalizada para esses cenários.

> **Note**
>
> A partir da AWS ParallelCluster versão 2.6.1, a maioria das receitas de instalação são ignoradas por padrão ao iniciar os nós. Isso é para melhorar os tempos de startup. Para executar todas as fórmulas de instalação para uma melhor compatibilidade com versões anteriores em detrimento dos tempos de inicialização, adicione `"skip_install_recipes" : "no"` à chave `cluster` na configuração `extra_json`. Por exemplo:
>
> ```
> extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
> ```

Pré-requisitos

- AWS ParallelCluster está instalado.
- O AWS CLI está instalado e configurado.
- Você tem um par de chaves EC2.
- Você tem um perfil do IAM com as permissões necessárias para executar a CLI do `pcluster`.

Como personalizar a AWS ParallelCluster AMI

Há três maneiras de usar uma AWS ParallelCluster AMI personalizada descritas nas próximas seções. Dois desses três métodos exigem a criação de uma nova AMI que esteja disponível na sua Conta da AWS. O terceiro método (usar uma AMI personalizada em runtime) não exige que você crie nada com antecedência, mas adiciona riscos à implantação. Escolha o método que melhor atenda às suas necessidades.

Modificar uma AMI do

Esse é o método mais simples e recomendado. Como a AWS ParallelCluster AMI básica geralmente é atualizada com novas versões, essa AMI tem todos os componentes necessários AWS ParallelCluster para funcionar quando instalada e configurada. Comece com isso como base.

New EC2 console

1. Na lista de AWS ParallelCluster AMI, encontre a AMI que corresponde à específica Região da AWS que você usa. A lista de AMI que você escolher deve corresponder à versão AWS ParallelCluster que você usa. Execute `pcluster version` para verificar a versão. Para a AWS ParallelCluster versão 2.11.9, acesse https://github.com/aws/aws- parallelcluster/blob/v2.11.9/amis .txt. Para selecionar outra versão, use o mesmo link, escolha o botão Tag: 2.11.9, selecione a guia Tags e escolha a versão apropriada.
2. Faça login no Console de gerenciamento da AWS e abra o console do Amazon EC2 em. https://console.aws.amazon.com/ec2/
3. No painel do Amazon EC2, escolha Executar instância.
4. Em Imagens do aplicativo e do sistema operacional, escolha Procurar mais AMIs, navegue até Comunidade AMIs e insira o ID da AWS ParallelCluster AMI para você Região da AWS na caixa de pesquisa.
5. Selecione a AMI, escolhe o tipo de instância e as propriedades, selecione seu Par de chaves, e Executar instância.
6. Faça login na instância usando o usuário do SO e chave SSH. Para obter mais informações, navegue até Instâncias, selecione a nova instância e Conectar.
7. Personalize a instância conforme necessário.
8. Execute o seguinte comando para preparar a instância para a criação da AMI:

   ```
   sudo /usr/local/sbin/ami_cleanup.sh
   ```

9. Vá para Instâncias, escolha a nova instância, selecione Estado da instância, e Interromper instância.
10. Crie uma nova AMI a partir da instância usando o console do EC2 ou AWS CLI create-image.

    Do console do EC2

    a. No painel de navegação, escolha Instances (Instâncias).

    b. Escolha a instância que você criou e modificou.

    c. Em Ações, escolha Imagens e modelos e depois Criar imagem.

    d. Escolha Create Image.

11. Insira o novo ID de AMI no campo custom_ami na configuração do cluster.

Old EC2 console

1. Na lista de AWS ParallelCluster AMI, encontre a AMI que corresponde à específica Região da AWS que você usa. A lista de AMI que você escolher deve corresponder à versão AWS ParallelCluster que você usa. Execute `pcluster version` para verificar a versão. Para a AWS ParallelCluster versão 2.11.9, acesse https://github.com/aws/aws- parallelcluster/blob/v2.11.9/amis .txt. Para selecionar outra versão, use o mesmo link, escolha o botão Tag: 2.11.9, selecione a guia Tags e escolha a versão apropriada.
2. Faça login no Console de gerenciamento da AWS e abra o console do Amazon EC2 em. https://console.aws.amazon.com/ec2/
3. No painel do Amazon EC2, escolha Executar instância.
4. Escolha Comunidade AMIs, pesquise a ID da AWS ParallelCluster AMI e selecione-a.
5. Escolha seu tipo de instância e selecione Avançar: Configurar detalhes da instância, ou Revisar e iniciar para iniciar sua instância.
6. Escolha Executar, selecione seu par de chaves, e Executar instâncias.
7. Faça login na instância usando o usuário do SO e chave SSH. Para obter mais informações, navegue até Instâncias, selecione a nova instância e Conectar.
8. Personalize a instância conforme necessário.
9. Execute o seguinte comando para preparar a instância para a criação da AMI:

   ```
   sudo /usr/local/sbin/ami_cleanup.sh
   ```

10. Vá para Instâncias, escolha a nova instância, selecione Estado da instância, e Interromper
11. Crie uma nova AMI a partir da instância usando o console do EC2 ou AWS CLI create-image.

    Do console do EC2

    a. No painel de navegação, escolha Instances (Instâncias).
    b. Escolha a instância que você criou e modificou.
    c. Em Ações, escolha Imagem, e, em seguida, Criar imagem.
    d. Escolha Create Image.

12. Insira o novo ID de AMI no campo custom_ami na configuração do cluster.

Crie uma AWS ParallelCluster AMI personalizada

Se você tiver uma AMI personalizada e software já existentes, aplique as alterações necessárias pelo AWS ParallelCluster .

1. Instale o seguinte em seu sistema local, junto com a AWS ParallelCluster CLI:
   - Packer: encontre a versão mais recente para o seu SO no site do Packer e instale-a. A versão deve ser pelo menos 1.4.0, mas a versão mais recente é recomendada. Verifique se o comando `packer` está disponível em seu PATH.

   > **Note**
   >
   > Antes da AWS ParallelCluster versão 2.8.0, era necessário usar o Berkshelf (que é instalado usando`gem install berkshelf`). `pcluster createami`

2. Configure suas Conta da AWS credenciais para que o Packer possa fazer chamadas para operações de AWS API em seu nome. O conjunto mínimo de permissões necessárias para que o Packer funcione está documentado na seção Tarefa do IAM ou função da instância do tópico do Amazon AMI Builder na documentação do Packer.
3. Use o comando `createami` na AWS ParallelCluster CLI para criar uma AWS ParallelCluster AMI a partir da que você fornece como base:

   ```
   pcluster createami --ami-id <BASE_AMI> --os <BASE_AMI_OS>
   ```

   > **Important**
   >
   > Você não deve usar uma AWS ParallelCluster AMI de um cluster em execução como *<BASE_AMI>* para o `createami` comando. Caso contrário, o comando falhará.

   Para Outros parâmetros, consulte `pcluster createami`.
4. O comando da Etapa 4 executa o Packer, que faz especificamente o seguinte:
   a. Executa uma instância usando a AMI de base fornecida.
   b. Aplica o AWS ParallelCluster livro de receitas à instância para instalar o software relevante e realizar outras tarefas de configuração necessárias.
   c. Interrompe a instância.

d. Cria uma AMI a partir da instância.

e. Encerra a instância após a criação da AMI.

f. Produz a string do ID da nova AMI a ser usada para a criação do cluster.

5. Para criar o cluster, insira o ID de AMI no campo custom_ami na configuração do cluster.

> **Note**
>
> O tipo de instância usado para criar uma AWS ParallelCluster AMI personalizada é `t2.xlarge`. Esse tipo de instância não se qualifica para o nível AWS gratuito, então você é cobrado por todas as instâncias criadas ao criar essa AMI.

Usar uma AMI personalizada no tempo de execução

> **Warning**
>
> Para evitar o risco de usar uma AMI que não seja compatível com AWS ParallelCluster, recomendamos que você evite usar esse método.
> Quando os nós de computação são iniciados sem testes potenciais AMIs em tempo de execução, as incompatibilidades com a instalação em tempo de AWS ParallelCluster execução do software necessário podem fazer AWS ParallelCluster com que ele pare de funcionar.

Se você não quiser criar nada com antecedência, você pode usar sua AMI e criar uma a AWS ParallelCluster partir dessa AMI.

Com esse método, a criação do AWS ParallelCluster leva mais tempo porque todo o software necessário para AWS ParallelCluster a criação do cluster deve estar instalado. Além disso, aumentar a escala verticalmente também leva mais tempo.

- Insira o ID de AMI no campo custom_ami na configuração do cluster.

Executando uma tarefa MPI com um AWS ParallelCluster agendador **awsbatch**

Este tutorial orienta você durante a execução de uma tarefa de MPI com o `awsbatch` como um programador.

Pré-requisitos

- AWS ParallelCluster está instalado.
- O AWS CLI está instalado e configurado.
- Você tem um par de chaves EC2.
- Você tem um perfil do IAM com as permissões necessárias para executar a CLI `pcluster`.

Criar o cluster do

Primeiro, vamos criar uma configuração para um cluster que usa `awsbatch` como o programador. Certifique-se de inserir os dados ausentes na seção `vpc` e no campo `key_name` pelos recursos que foram criados no momento da configuração.

```
[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-#######
```

Agora você pode iniciar a criação do cluster. Vamos chamar nosso cluster*awsbatch-tutorial*.

```
$ pcluster create -c /path/to/the/created/config/aws_batch.config -t awsbatch awsbatch-tutorial
```

Quando o cluster for criado, você verá uma saída semelhante à seguinte:

```
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
```

Fazer login em seu nó principal

Os comandos do AWS ParallelCluster Batch CLI estão todos disponíveis na máquina cliente em que AWS ParallelCluster está instalado. No entanto, aplicaremos SSH para o nó principal e enviaremos as tarefas de lá. Isso nos permite aproveitar o volume NFS que é compartilhado entre o chefe e todas as instâncias do Docker que AWS Batch executam trabalhos.

Use o arquivo pem SSH para fazer login no nó principal.

```
$ pcluster ssh awsbatch-tutorial -i /path/to/keyfile.pem
```

Quando você estiver logado, execute os comandos `awsbqueues` e mostre `awsbhosts` a AWS Batch fila configurada e as instâncias do Amazon ECS em execução.

```
[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
```

Como pode ser visto na saída, temos um único host em execução. Isso deve-se ao valor escolhido para `min_vcpus` na configuração. Se você quiser exibir detalhes adicionais sobre a AWS Batch fila e os hosts, adicione o `-d` sinalizador ao comando.

Executando seu primeiro trabalho usando AWS Batch

Antes de mudar para MPI vamos criar uma tarefa de simulação que permanece em espera por um tempo e gera seu próprio nome de host, saudando o nome transmitido como um parâmetro.

Crie um arquivo chamado "hellojob.sh" com o conteúdo a seguir.

```
#!/bin/bash

sleep 30
echo "Hello $1 from $HOSTNAME"
echo "Hello $1 from $HOSTNAME" > "/shared/secret_message_for_${1}_by_
${AWS_BATCH_JOB_ID}"
```

Depois, envie a tarefa usando `awsbsub` e verifique se ela é executada.

```
$ awsbsub -jn hello -cf hellojob.sh Luca
Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2 (hello) has been submitted.
```

Visualize a fila e verifique o status da tarefa.

```
$ awsbstat
jobId                                jobName      status    startedAt
 stoppedAt    exitCode
-----------------------------------  -----------  --------  -------------------
 -----------  ----------
6efe6c7c-4943-4c1a-baf5-edbfeccab5d2  hello        RUNNING   2018-11-12 09:41:29  -
        -
```

A saída fornece informações detalhadas sobre a tarefa.

```
$ 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-exampleBatch:1
jobQueue                 : parallelcluster-exampleBatch
command                  : /bin/bash -c 'aws s3 --region us-east-1 cp
 s3://amzn-s3-demo-bucket/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-exampleBatch/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-exampleBatch/default/
c75dac4a-5aca-4238-a4dd-078037453554
-------------------------
```

Observe que, no momento, a tarefa está em um estado `RUNNING`. Aguarde 30 segundos para que a tarefa seja concluída e execute `awsbstat` novamente.

```
$ awsbstat
jobId                                jobName      status    startedAt
 stoppedAt    exitCode
------------------------------------  -----------  --------  -------------------
 -----------  ----------
```

Veja que agora a tarefa está com status `SUCCEEDED`.

```
$ 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
```

Como agora não há tarefas na fila, podemos verificar a saída por meio do comando `awsbout`.

```
$ awsbout 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
2018-11-12 09:41:29: Starting Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
download: s3://amzn-s3-demo-bucket/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
```

Podemos ver que nosso trabalho foi executado com êxito na instância "ip-172-31-4-234".

Se você analisar o diretório `/shared`, encontrará uma mensagem secreta para você.

Para explorar todos os recursos disponíveis que não fazem parte desse tutorial, consulte a documentação da CLI do Batch do AWS ParallelCluster. Quando estiver pronto para continuar o tutorial, prosseguiremos e veremos como enviar uma tarefa de MPI.

Executar um trabalho de MPI em um ambiente paralelo de vários nós

Ainda conectado ao nó principal, crie um arquivo no diretório `/shared` chamado `mpi_hello_world.c`. Adicione o seguinte programa de MPI ao arquivo:

```
// 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();
}
```

Agora, salve o código a seguir como `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
```

Agora, estamos prontos para enviar nossa primeira tarefa de MPI e executá-la simultaneamente em três nós:

```
$ awsbsub -n 3 -cf submit_mpi.sh
```

Vamos monitorar o status da tarefa e aguardar até que ela entre no status `RUNNING`:

```
$ watch awsbstat -d
```

Assim que entrar no status `RUNNING`, podemos verificar sua saída. Para mostrar a saída do nó principal, acrescente `#0` ao ID da tarefa. Para mostrar a saída dos nós de computação, use #1 e #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://amzn-s3-demo-bucket/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://amzn-s3-demo-bucket/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!
```

Agora, podemos confirmar se o trabalho foi concluído com êxito:

```
[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  -
```

Observação: Se quiser encerrar uma tarefa antes que seja finalizada, use o comando `awsbkill`.

Criptografia de disco com uma chave do KMS personalizada

AWS ParallelCluster O oferece suporte às opções de configuração `ebs_kms_key_id` e `fsx_kms_key_id`. Essas opções permitem que você forneça uma AWS KMS chave personalizada para a criptografia de disco do Amazon EBS ou FSx para o Lustre. Para usá-las, especifique uma `ec2_iam_role`.

Para que o cluster seja criado, a AWS KMS chave deve saber o nome da função do cluster. Isso impede que você use a função criada na criação do cluster, exigindo uma `ec2_iam_role` personalizada.

Pré-requisitos

- AWS ParallelCluster está instalado.
- O AWS CLI está instalado e configurado.
- Você tem um par de EC2 chaves.
- Você tem um perfil do IAM com as permissões necessárias para executar a CLI `pcluster`.

Criar a função

Primeiro, crie uma política:

1. Acesse o console do IAM: https://console.aws.amazon.com/iam/página inicial.
2. Em Policies (Políticas), Create policy (Criar política), clique na guia JSON.
3. No corpo da política, cole a Política de instância. Lembre-se de substituir todas as ocorrências de *<AWS ACCOUNT ID>* e *<REGION>*.
4. Nomeie a política como `ParallelClusterInstancePolicy` e clique em Create Policy (Criar política).

Depois, crie uma função:

1. Em Roles (Funções), crie uma função.
2. Clique em `EC2` como a entidade confiável.
3. Em Permissions (Permissões), procure a função `ParallelClusterInstancePolicy` que acabou de ser criada e anexe-a.
4. Nomeie a função como `ParallelClusterInstanceRole` e clique em Create Role (Criar função).

Conceder permissões à chave

No AWS KMS Console > Chaves gerenciadas pelo cliente > clique no alias ou ID da chave da sua chave.

Clique no botão Adicionar na caixa Usuários principais, abaixo da guia Política de chaves, e pesquise o ParallelClusterInstanceRoleque você acabou de criar. Anexe-a.

Criar o cluster do

Agora, crie um cluster. Veja a seguir um exemplo de um cluster com unidades `Raid 0` criptografadas:

```
[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
```

Veja a seguir um exemplo com o sistema de arquivos FSx for Lustre:

```
[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
```

Configurações semelhantes se aplicam ao Amazon EBS e aos sistemas de arquivos FSx baseados na Amazon.

Tutorial do modo de fila múltipla

Executando seus trabalhos AWS ParallelCluster com o modo de várias filas

Este tutorial explica como executar seu primeiro trabalho do Hello World AWS ParallelCluster comModo de fila múltipla.

Pré-requisitos

- AWS ParallelCluster está instalado.
- O AWS CLI está instalado e configurado.

- Você tem um par de EC2 chaves.
- Você tem um perfil do IAM com as permissões necessárias para executar a CLI `pcluster`.

> **Note**
>
> O modo de fila múltipla só é compatível com a AWS ParallelCluster versão 2.9.0 ou posterior.

Configuração do cluster

Primeiro, verifique se AWS ParallelCluster está instalado corretamente executando o comando a seguir.

```
$ pcluster version
```

Para obter mais informações sobre o `pcluster version`, consulte `pcluster version`.

Esse comando retorna a versão em execução do AWS ParallelCluster.

Em seguida, execute o `pcluster configure` para gerar um arquivo de configuração básico. Siga todas as instruções que seguem esse comando.

```
$ pcluster configure
```

Para obter mais informações sobre o comando `pcluster configure`, consulte `pcluster configure`.

Depois de concluir esta etapa, você deverá ter um arquivo de configuração básica em `~/.parallelcluster/config`. Esse arquivo deve conter configurações básicas de cluster e uma seção de VPC.

A próxima parte do tutorial descreve como modificar sua configuração recém-criada e iniciar um cluster com várias filas.

> **Note**
>
> Algumas instâncias que este tutorial usa não são elegíveis para o nível gratuito.

Para este tutorial, use a seguinte configuração.

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue

[aws]
aws_region_name = <Your Região da 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
```

Como criar seu cluster

Esta seção detalha como criar o cluster no modo de fila múltipla.

Primeiro, nomeie seu cluster `multi-queue-hello-world` e crie o cluster de acordo com a seção de cluster de `multi-queue` definida na seção anterior.

```
$ pcluster create multi-queue-hello-world -t multi-queue
```

Para obter mais informações sobre o `pcluster create`, consulte pcluster create.

Quando o cluster é criado, a seguinte saída é exibida:

```
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
```

A mensagem `CREATE_COMPLETE` mostra que o cluster foi criado com êxito. A saída também fornece os endereços IP público e privado do nó principal.

Fazer login em seu nó principal

Use seu arquivo de chave SSH privada para fazer login no nó principal.

```
$ pcluster ssh multi-queue-hello-world -i ~/path/to/keyfile.pem
```

Para obter mais informações sobre o `pcluster ssh`, consulte pcluster ssh.

Depois de fazer login, execute o comando `sinfo` para verificar se suas filas do programador estão definidas e configuradas.

Para obter mais informações sobre`sinfo`, consulte sinfo no Slurm documentação.

```
$ 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
```

A saída mostra que você tem dois nós de computação `t2.micro` no estado `idle` que estão disponíveis no seu cluster.

> **Note**
>
> - `spot-st-t2micro-1` é um nó estático com `st` em seu nome. Esse nó está sempre disponível e corresponde ao `min_count = 1` da configuração do seu cluster.
> - `spot-dy-t2micro-1` é um nó dinâmico com `dy` em seu nome. Esse nó está disponível no momento porque corresponde a `initial_count - min_count = 1` de acordo com a configuração do seu cluster. Esse nó é reduzido verticalmente após seu `scaledown_idletime` personalizado de cinco minutos.

Outros nós estão todos no estado de economia de energia, mostrado pelo ~ sufixo no estado do nó, sem nenhuma EC2 instância apoiando-os. A fila padrão é indicada por um sufixo * após o nome da fila. A `spot` é sua fila de trabalhos padrão.

Como executar o trabalho no modo de várias filas

Em seguida, tente fazer com que o trabalho fique em latência por um tempo. Posteriormente, o trabalho gera seu próprio nome de host. Certifique-se de que esse script possa ser executado pelo usuário atual.

```
$ 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
```

Envie o trabalho usando o comando `sbatch`. Solicite dois nós para esse trabalho com a opção `-N 2` e verifique se o trabalho foi enviado com êxito. Para obter mais informações sobre `sbatch`, consulte sbatchna documentação do Slurm.

```
$ sbatch -N 2 --wrap "srun hellojob.sh"
Submitted batch job 2
```

Agora, você pode visualizar a fila e verificar o status do trabalho com o comando `squeue`. Observe que, como você não especificou uma fila específica, a fila padrão (`spot`) é usada. Para obter mais informações sobre `squeue`, consulte squeue no Slurmdocumentação.

```
$ 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
```

A saída mostra que, no momento, a tarefa está em um estado de execução. Aguarde 30 segundos para que a tarefa seja concluída e execute `squeue` novamente.

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

Agora que todos os trabalhos na fila foram concluídos, procure o arquivo de saída `slurm-2.out` em seu diretório atual.

```
$ cat slurm-2.out
Hello World from spot-dy-t2micro-1
Hello World from spot-st-t2micro-1
```

A saída também mostra que o trabalho foi executado com êxito nos nós `spot-st-t2micro-1` e `spot-st-t2micro-2`.

Agora, envie o mesmo trabalho especificando restrições para instâncias específicas com os comandos a seguir.

```
$ sbatch -N 3 -p spot -C "[c5.xlarge*1&t2.micro*2]" --wrap "srun hellojob.sh"
Submitted batch job 3
```

Você usou esses parâmetros para `sbatch`.

- `-N 3`— solicita três nós
- `-p spot`— envia o trabalho para a fila `spot`. Você também pode enviar um trabalho para a fila `ondemand`, especificando `-p ondemand`.
- `-C "[c5.xlarge*1&t2.micro*2]"`— especifica as restrições específicas do nó para esse trabalho. Isso solicita que um (1) nó `c5.xlarge` e dois (2) nós `t2.micro` sejam usados para esse trabalho.

Execute o comando `sinfo` para visualizar os nós e as filas. (As filas de entrada AWS ParallelCluster são chamadas de partições em Slurm.)

```
$ 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
```

Os nós estão sendo ativados. Isso é indicado pelo sufixo # no estado do nó. Execute o squeue comando para visualizar informações sobre os trabalhos no cluster.

```
$ 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
```

Seu trabalho está no `CF` (CONFIGURING) state, aguardando que as instâncias aumentem e se juntem ao cluster.

Após cerca de três minutos, os nós devem estar disponíveis e o trabalho entra no `R` (RUNNING) estado.

```
$ 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
```

O trabalho termina e todos os três nós estão no estado `idle`.

```
$ 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
```

Então, depois que nenhum trabalho permanecer na fila, procure `slurm-3.out` no seu diretório local.

```
$ 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
```

A saída também mostra que o trabalho foi executado com êxito nos nós correspondentes.

Você pode observar o processo de redução de escala. Em sua configuração de cluster, você especificou um [scaledown_idletime] personalizado de 5 minutos. Depois de cinco minutos em estado ocioso, seus nós dinâmicos `spot-dy-c5xlarge-1` e `spot-dy-t2micro-1` são automaticamente reduzidos verticalmente e entram no modo `POWER_DOWN`. Observe que o nó estático `spot-st-t2micro-1` não é reduzido verticalmente.

```
$ 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
```

No código acima, você pode ver que `spot-dy-c5xlarge-1` e `spot-dy-t2micro-1` estão no modo `POWER_DOWN`. Isso é indicado pelo sufixo `%`. As instâncias correspondentes são encerradas imediatamente, mas os nós permanecem no estado `POWER_DOWN` e não ficam disponíveis para uso por 120 segundos (dois minutos). Após esse período, os nós retornam economizando energia e

estão disponíveis para uso novamente. Para obter mais informações, consulte Guia do Slurm para o modo de várias filas.

Esse deve ser o estado final do cluster:

```
$ 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
```

Depois de se desconectar do cluster, você pode limpá-lo, executando o `pcluster delete`. Para obter mais informações sobre `pcluster list` e `pcluster delete`, consulte `pcluster list` e `pcluster delete`.

```
$ pcluster list
multi-queue CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue
Deleting: multi-queue
...
```

Execução de trabalhos em cluster com instâncias EFA e GPU

Esta parte do tutorial detalha como modificar a configuração e iniciar um cluster com várias filas que contêm instâncias com rede EFA e atributos GPU. Observe que as instâncias usadas neste tutorial têm preços mais altos.

Verifique os limites da sua conta para ter certeza de que você está autorizado a usar essas instâncias antes de prosseguir com as etapas descritas neste tutorial.

Modifique o arquivo de configuração da seguinte forma.

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue-special

[aws]
aws_region_name = <Your Região da 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
```

Criar um cluster

```
$ pcluster create multi-queue-special -t multi-queue-special
```

Depois da criação do cluster, use seu arquivo de chave SSH privada para fazer login no nó principal.

```
$ pcluster ssh multi-queue-special -i ~/path/to/keyfile.pem
```

Esse deve ser o estado inicial do cluster:

```
$ 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]
```

Esta seção descreve como enviar alguns trabalhos para verificar se os nós têm atributos de EFA ou GPU.

Primeiro, escreva os scripts de trabalho. O `efa_job.sh` aguarda por 30 segundos. Depois disso, procure o EFA na saída do comando `lspci`. O `gpu_job.sh` aguarda por 30 segundos. Depois disso, execute `nvidia-smi` para mostrar as informações do GPU sobre o nó.

```
$ 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
```

Envie o trabalho com `sbatch`.

```
$ 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]
```

Após alguns minutos, você deverá ver os nós on-line e os trabalhos em execução.

```
[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
```

Depois que o trabalho for concluído, verifique a saída. Na saída do arquivo `slurm-2.out`, você pode ver que o EFA está presente no nó `efa-dy-c5n18xlarge-1`. Na saída do arquivo `slurm-3.out`, você pode ver que a saída `nvidia-smi` contém informações de GPU para o nó `gpu-dy-g38xlarge-1`.

```
$ 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                                                 |
+-----------------------------------------------------------------------------+
```

Você pode observar o processo de redução de escala. Em sua configuração de cluster, você especificou um `scaledown_idletime` personalizado de cinco minutos. Por isso, depois de cinco minutos em estado ocioso, seus nós dinâmicos `spot-dy-c5xlarge-1` e `spot-dy-t2micro-1` são automaticamente reduzidos verticalmente e entram no modo `POWER_DOWN`. Por fim, os nós entram no modo de economia de energia e ficam disponíveis para uso novamente.

```
$ 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]
```

Depois de se desconectar do cluster, você pode limpá-lo, executando o `pcluster delete <cluster name>`.

```
$ pcluster list
multi-queue-special CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue-special
Deleting: multi-queue-special
...
```

Para obter mais informações, consulte Guia do Slurm para o modo de várias filas.

Desenvolvimento

Você pode usar as seções a seguir para começar o desenvolvimento do AWS ParallelCluster.

> **Important**
>
> As seções a seguir incluem instruções para usar uma versão personalizada das receitas do livro de receitas e um pacote de nós personalizado do AWS ParallelCluster . Essas informações abrangem um método avançado de personalização AWS ParallelCluster, com possíveis problemas que podem ser difíceis de depurar. A AWS ParallelCluster equipe recomenda fortemente o uso dos scripts em ações personalizadas do Bootstrap para personalização, porque os ganchos pós-instalação geralmente são mais fáceis de depurar e mais portáteis em todas as versões do. AWS ParallelCluster

Tópicos

Configurando um livro de AWS ParallelCluster receitas personalizado

> **Important**
>
> A seguir estão as instruções para usar uma versão personalizada das receitas do AWS ParallelCluster livro de receitas. Esse é um método avançado de personalização AWS ParallelCluster, com possíveis problemas que podem ser difíceis de depurar. A AWS ParallelCluster equipe recomenda fortemente o uso dos scripts em ações personalizadas do Bootstrap para personalização, porque os ganchos pós-instalação geralmente são mais fáceis de depurar e mais portáteis em todas as versões do. AWS ParallelCluster

Etapas

1. Identifique o diretório de trabalho do AWS ParallelCluster Livro de Receitas em que você clonou o código do livro de AWS ParallelCluster receitas.

```
_cookbookDir=<path to cookbook>
```

2. Detecte a versão atual do AWS ParallelCluster Livro de Receitas.

```
_version=$(grep version ${_cookbookDir}/metadata.rb|awk '{print $2}'| tr -d \')
```

3. Crie um arquivo do AWS ParallelCluster Livro de Receitas e calcule seu 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. Crie um bucket do Amazon S3 e carregue o arquivo, seu md5 e a data da última modificação no bucket. Conceda permissão de leitura pública por meio de uma ACL public-read.

```
_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. Adicione as seguintes variáveis ao arquivo AWS ParallelCluster de configuração, na [cluster]seção.

```
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**
>
> A partir da AWS ParallelCluster versão 2.6.1, a maioria das receitas de instalação são ignoradas por padrão ao iniciar os nós para melhorar os tempos de inicialização. Para ignorar a maioria das fórmulas de instalação a fim de obter melhores tempos de inicialização em detrimento da compatibilidade com versões anteriores, remova `"skip_install_recipes" : "no"` da chave `cluster` na configuração `extra_json`.

Configurando um pacote de AWS ParallelCluster nós personalizado

> **Warning**
>
> A seguir estão as instruções para usar uma versão personalizada do pacote AWS ParallelCluster node. Esse é um método avançado de personalização AWS ParallelCluster, com possíveis problemas que podem ser difíceis de depurar. A AWS ParallelCluster equipe recomenda fortemente o uso dos scripts em ações personalizadas do Bootstrap para personalização, porque os ganchos pós-instalação geralmente são mais fáceis de depurar e mais portáteis em todas as versões do. AWS ParallelCluster

Etapas

1. Identifique o diretório de trabalho do AWS ParallelCluster nó em que você clonou o código do AWS ParallelCluster nó.

```
_nodeDir=<path to node package>
```

2. Detecte a versão atual do AWS ParallelCluster nó.

```
_version=$(grep "version = \"" ${_nodeDir}/setup.py |awk '{print $3}' | tr -d \")
```

3. Crie um arquivo do AWS ParallelCluster Node.

```
cd "${_nodeDir}"
_stashName=$(git stash create)
```

```
git archive --format tar --prefix="aws-parallelcluster-node-${_version}/"
 "${_stashName:-HEAD}" | gzip > "aws-parallelcluster-node-${_version}.tgz"
```

4. Crie um bucket do Amazon S3 e faça upload do arquivo no bucket. Conceda permissão de leitura pública por meio de uma ACL public-read.

```
_bucket=<the bucket name>
aws s3 cp --acl public-read aws-parallelcluster-node-${_version}.tgz s3://${_bucket}/
node/aws-parallelcluster-node-${_version}.tgz
```

5. Adicione a variável a seguir ao arquivo de AWS ParallelCluster configuração, na [cluster]seção.

```
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**
>
> A partir da AWS ParallelCluster versão 2.6.1, a maioria das receitas de instalação são ignoradas por padrão ao iniciar os nós para melhorar os tempos de inicialização. Para ignorar a maioria das fórmulas de instalação a fim de obter melhores tempos de inicialização em detrimento da compatibilidade com versões anteriores, remova `"skip_install_recipes" : "no"` da chave `cluster` na configuração `extra_json`.

AWS ParallelCluster solução de problemas

A AWS ParallelCluster comunidade mantém uma página Wiki que fornece muitas dicas de solução de problemas na AWS ParallelCluster GitHub Wiki. Para obter uma lista de problemas conhecidos, consulte Problemas conhecidos.

Tópicos

Recuperando e preservando logs

Os logs são um recurso útil para solucionar problemas. Antes de usar logs para solucionar problemas com seus recursos do AWS ParallelCluster , você deve primeiro criar um arquivo de logs de cluster. Siga as etapas descritas no tópico Criando um arquivo dos registros de um cluster no AWS ParallelCluster GitHub Wiki para iniciar esse processo.

Se um de seus clusters em execução estiver enfrentando problemas, coloque o cluster em um estado `STOPPED` executando o comando `pcluster stop` <*cluster_name*> antes de começar a solucionar o problema. Isso evita incorrer em custos inesperados.

Se `pcluster` parar de funcionar ou se você quiser excluir um cluster enquanto ainda preserva seus logs, execute o comando `pcluster delete` –`keep-logs` <*cluster_name*>. A execução desse comando exclui o cluster, mas retém o grupo de registros armazenado na Amazon. CloudWatch Para obter mais informações sobre esse comando, consulte a documentação do `pcluster delete`.

Solução de problemas de implantação de pilha

Se o cluster não for criado e reverter a criação da pilha, você poderá examinar os arquivos de log a seguir para diagnosticar o problema. Você deseja procurar a saída de `ROLLBACK_IN_PROGRESS` nesses logs. A mensagem de falha se parecerá com a seguinte:

```
$ 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
```

Para diagnosticar o problema, crie o cluster novamente usando `pcluster create`, incluindo o sinalizador `--norollback`. Em seguida, execute o SSH no cluster:

```
$ pcluster create mycluster --norollback
...
$ pcluster ssh mycluster
```

Depois de fazer login no nó principal, você deve encontrar três arquivos de log principais que podem ser usados para encontrar o erro.

- `/var/log/cfn-init.log` é o log do script `cfn-init`. Primeiro, verifique esse log. É provável que você veja um erro como o `Command chef failed` nesse log. Veja as linhas imediatamente antes dessa linha para obter mais detalhes relacionados à mensagem de erro. Para mais informações, consulte cfn-init.
- `/var/log/cloud-init.log` é o log do cloud-init. Se não vir nada no `cfn-init.log`, tente verificar esse log a seguir.
- `/var/log/cloud-init-output.log` é a saída dos comandos que foram executados pelo cloud-init. Isso inclui a saída de `cfn-init`. Na maioria dos casos, não é preciso consultar esse log para solucionar esse tipo de problema.

Solução de problemas em clusters em modo de várias filas

Esta seção é relevante para clusters que foram instalados usando a AWS ParallelCluster versão 2.9.0 e posterior com o agendador de Slurm tarefas. Para obter mais informações sobre o modo de várias filas, consulte Modo de fila múltipla.

Tópicos

Logs de chaves

A tabela a seguir fornece uma visão geral dos principais logs do nó principal:

`/var/log/cfn-init.log`

Esse é o log CloudFormation de inicialização. Ele contém todos os comandos que foram executados quando uma instância foi configurada. É útil para solucionar problemas de inicialização.

`/var/log/chef-client.log`

Este é o log do cliente Chef. Ele contém todos os comandos que foram executados pelo Chef/CINC. É útil para solucionar problemas de inicialização.

`/var/log/parallelcluster/slurm_resume.log`

Esse é um log do `ResumeProgram`. Ele lança instâncias para nós dinâmicos e é útil para solucionar problemas de inicialização de nós dinâmicos.

`/var/log/parallelcluster/slurm_suspend.log`

Esse é o log do `SuspendProgram`. É chamado quando as instâncias são encerradas para nós dinâmicos e é útil para solucionar problemas de encerramento de nós dinâmicos. Ao verificar esse log, você também deve verificar o log do `clustermgtd`.

`/var/log/parallelcluster/clustermgtd`

Esse é o log do `clustermgtd`. Ele é executado como o daemon centralizado que gerencia a maioria das ações de operação do cluster. É útil para solucionar qualquer problema de inicialização, encerramento ou operação do cluster.

`/var/log/slurmctld.log`

Esse é o log do daemon de Slurm controle. AWS ParallelCluster não toma decisões de escalabilidade. Em vez disso, ele apenas tenta lançar recursos para satisfazer os requisitos do Slurm. É útil para problemas de escala e alocação, problemas relacionados ao trabalho e quaisquer problemas de inicialização e encerramento relacionados ao programador.

Estes são os principais logs dos nós de computação:

`/var/log/cloud-init-output.log`

Esse é o log cloud-init. Ele contém todos os comandos que foram executados quando uma instância foi configurada. É útil para solucionar problemas de inicialização.

`/var/log/parallelcluster/computemgtd`

Esse é o log do `computemgtd`. Ele é executado em cada nó de computação para monitorar o nó no raro caso de o daemon `clustermgtd` no nó principal estar off-line. É útil para solucionar problemas inesperados de encerramento.

`/var/log/slurmd.log`

Este é o log do daemon de computação do Slurm. É útil para solucionar problemas de inicialização e de falhas de computação.

Solução de problemas de inicialização do nó

Esta seção aborda como você pode solucionar problemas de inicialização do nó. Isso inclui problemas em que o nó falha ao iniciar, ligar ou ingressar em um cluster.

Nó principal:

Logs aplicáveis:

- `/var/log/cfn-init.log`
- `/var/log/chef-client.log`
- `/var/log/parallelcluster/clustermgtd`
- `/var/log/parallelcluster/slurm_resume.log`
- `/var/log/slurmctld.log`

Verifique os logs `/var/log/cfn-init.log` e `/var/log/chef-client.log`. Esses logs devem conter todas as ações que foram executadas quando o nó principal foi configurado. A maioria dos erros que ocorrem durante a configuração deve ter mensagens de erro localizadas no log `/var/log/chef-client.log`. Se os scripts pré-instalação ou pós-instalação forem especificados na configuração do cluster, verifique novamente se o script é executado com êxito por meio de mensagens de log.

Quando um cluster é criado, o nó principal precisa esperar que os nós de computação se juntem ao cluster antes que ele possa se juntar ao cluster. Sendo assim, se os nós de computação não conseguirem se juntar ao cluster, o nó principal também falhará. É possível seguir um desses conjuntos de procedimentos, dependendo do tipo de nós computacionais usados, para solucionar esse tipo de problema:

Nós de computação dinâmicos:

- Pesquise no log `ResumeProgram` (`/var/log/parallelcluster/slurm_resume.log`) o nome do seu nó de computação para ver se alguma vez o `ResumeProgram` foi chamado com o nó. (Se nunca `ResumeProgram` foi chamado, você pode verificar o `slurmctld` log (`/var/log/slurmctld.log`) para determinar se Slurm já tentou chamar `ResumeProgram` com o nó.)
- Observe que permissões incorretas em `ResumeProgram` podem fazer com que `ResumeProgram` falhe silenciosamente. Se você estiver usando uma AMI personalizada com modificações na configuração `ResumeProgram`, verifique se a `ResumeProgram` é de propriedade do usuário `slurm` e tem a permissão de `744` (`rwxr--r--`).
- Se `ResumeProgram` for chamado, verifique se uma instância foi executada para o nó. Se nenhuma instância foi iniciada, você pode ver uma mensagem de erro que descreve a falha na inicialização.
- Se a instância for executada, é possível que um problema tenha ocorrido durante o processo de configuração. Você deve ver o endereço IP privado e o ID da instância correspondentes no log `ResumeProgram`. Além disso, você pode ver os logs de configuração correspondentes para a instância específica. Para ter mais informações sobre como solucionar um erro de configuração com um nó de computação, consulte a próxima seção.

Nós de computação estáticos:

- Verifique o log `clustermgtd` (`/var/log/parallelcluster/clustermgtd`) para ver se foram iniciadas instâncias para o nó. Se eles não foram lançados, deve haver uma mensagem de erro clara detalhando a falha no lançamento.

- Se a instância for iniciada, haverá algum problema durante o processo de configuração. Você deve ver o endereço IP privado e o ID da instância correspondentes no log `ResumeProgram`. Além disso, você pode ver os logs de configuração correspondentes para a instância específica.

- Nós de computação:
  - Logs aplicáveis:
    - `/var/log/cloud-init-output.log`
    - `/var/log/slurmd.log`
  - Se um nó de computação for iniciado, primeiro verifique o `/var/log/cloud-init-output.log`, que deve conter os logs de configuração semelhantes ao log `/var/log/chef-client.log` do nó principal. A maioria dos erros que ocorrem durante a configuração deve ter mensagens de erro localizadas no log `/var/log/cloud-init-output.log`. Se scripts de pré-instalação ou pós-instalação forem especificados na configuração do cluster, verifique se eles foram executados com êxito.
  - Se você estiver usando uma AMI personalizada com modificação na configuração Slurm, talvez haja um erro relacionado ao Slurm que impeça o nó de computação de se juntar ao cluster. Para erros relacionados ao programador, verifique o log `/var/log/slurmd.log`.

Solução de problemas inesperados de substituições e encerramentos de nós

Esta seção continua explorando como você pode solucionar problemas relacionados ao nó, especificamente quando um nó é substituído ou encerrado inesperadamente.

- Logs aplicáveis:
  - `/var/log/parallelcluster/clustermgtd` (nó principal)
  - `/var/log/slurmctld.log` (nó principal)
  - `/var/log/parallelcluster/computemgtd` (nó de computação)
- Nós substituídos ou encerrados inesperadamente
  - Verifique no log `clustermgtd` (`/var/log/parallelcluster/clustermgtd`) se `clustermgtd` realizou a ação de substituir ou encerrar um nó. Observe que `clustermgtd` trata de todas as ações normais de manutenção do nó.
  - Se o `clustermgtd` substituiu ou encerrou um nó, deverá haver uma mensagem detalhando por que essa ação foi tomada no nó. Se o motivo estiver relacionado com o programador (por exemplo, o nó estiver em `DOWN`), verifique o log `slurmctld` para mais informações. Se o motivo

for relacionado ao Amazon EC2, deve haver uma mensagem informativa detalhando o problema relacionado ao Amazon EC2 que exigiu a substituição.

- Se o nó `clustermgtd` não foi encerrado, primeiro verifique se esse era um encerramento esperado pelo Amazon EC2, mais especificamente um encerramento pontual. `computemgtd`, executado em um nó de computação, também pode realizar uma ação para encerrar um nó se `clustermgtd` for determinado como não íntegro. Verifique no log `computemgtd` (`/var/log/parallelcluster/computemgtd`) se `computemgtd` encerrou o nó.

- Falha nos nós
  - Verifique no log `slurmctld` (`/var/log/slurmctld.log`) para ver por que um trabalho ou um nó falhou. Observe que os trabalhos são automaticamente enfileirados novamente se um nó falhar.
  - Se `slurm_resume` relatar que o nó foi iniciado e `clustermgtd` relatar após alguns minutos que não há nenhuma instância correspondente no Amazon EC2 para esse nó, o nó pode falhar durante a configuração. Para recuperar o log de um computador (`/var/log/cloud-init-output.log`), execute as seguintes etapas:
    - Envie um trabalho para permitir que o Slurm gere um novo nó.
    - Depois que o nó for iniciado, ative a proteção contra encerramento usando esse comando.

      ```
      aws ec2 modify-instance-attribute --instance-id i-xyz --disable-api-termination
      ```

    - Recupere a saída do console do nó com esse comando.

      ```
      aws ec2 get-console-output --instance-id i-xyz --output text
      ```

Substituindo, encerrando ou desligando instâncias e nós problemáticos

- Logs aplicáveis:
  - `/var/log/parallelcluster/clustermgtd` (nó principal)
  - `/var/log/parallelcluster/slurm_suspend.log` (nó principal)
- Na maioria dos casos, `clustermgtd` processa todas as ações esperadas de encerramento da instância. Examine o log `clustermgtd` para ver por que ele não conseguiu substituir ou encerrar um nó.
- Se os nós dinâmicos falharem no `scaledown_idletime`, consulte o log `SuspendProgram` para ver se o `SuspendProgram` foi chamado pelo `slurmctld`com o nó específico como argumento. Observe que o `SuspendProgram` não executa nenhuma ação. Em vez disso, ele só cria logs

quando é chamado. Todos encerramentos de instância e redefinições de `NodeAddr` são feitos por `clustermgtd`. O Slurm coloca automaticamente os nós de volta em um estado `POWER_SAVING` depois de `SuspendTimeout`.

Solucionando outros problemas conhecidos de nós e trabalhos

Outro tipo de problema conhecido é que AWS ParallelCluster pode falhar na alocação de trabalhos ou na tomada de decisões de escalabilidade. Com esse tipo de problema, AWS ParallelCluster somente inicia, encerra ou mantém recursos de acordo com Slurm as instruções. Para esses problemas, verifique o log `slurmctld` para solucioná-los.

Solução de problemas em clusters em modo de fila única

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Esta seção se aplica a clusters que não têm o modo de várias filas com uma das duas configurações a seguir:

- Lançado usando uma AWS ParallelCluster versão anterior à 2.9.0 e SGETorque, ou agendadores de Slurm tarefas.
- Lançado usando a AWS ParallelCluster versão 2.9.0 ou posterior e/ou agendadores SGE de Torque tarefas.

Tópicos

Logs de chaves

Os arquivos de log a seguir são os principais registros do nó principal.

Para a AWS ParallelCluster versão 2.9.0 ou posterior:

`/var/log/chef-client.log`

Este é o log do cliente CINC (chef). Ele contém todos os comandos que foram executados pelo CINC. É útil para solucionar problemas de inicialização.

Para todas as AWS ParallelCluster versões:

`/var/log/cfn-init.log`

Esse é o log do `cfn-init`. Ele contém todos os comandos que foram executados quando uma instância foi configurada e, portanto, é útil para solucionar problemas de inicialização. Para mais informações, consulte cfn-init.

`/var/log/clustermgtd.log`

Este é o log `clustermgtd` para programadores Slurm. O `clustermgtd` é executado como o daemon centralizado que gerencia a maioria das ações de operação do cluster. É útil para solucionar qualquer problema de inicialização, encerramento ou operação do cluster.

`/var/log/jobwatcher`

Este é o log `jobwatcher` para programadores SGE e Torque. O `jobwatcher` monitora a fila do programador e atualiza o grupo do Auto Scaling. É útil para solucionar problemas relacionados com aumentar a escala verticalmente dos nós.

`/var/log/sqswatcher`

Este é o log `sqswatcher` para os programadores SGE e Torque. `sqswatcher` processa o evento de instância pronta enviado por uma instância de computação após a inicialização bem-sucedida. Ele também adiciona nós de computação à configuração do programador. Esse registro é útil para solucionar o motivo pelo qual um nó ou nós falharam em ingressar em um cluster.

Estes são os principais logs dos nós de computação.

AWS ParallelCluster versão 2.9.0 ou posterior

`/var/log/cloud-init-output.log`

Esse é o log de inicialização do Cloud. Ele contém todos os comandos que foram executados quando uma instância foi configurada. É útil para solucionar problemas de inicialização.

AWS ParallelCluster versões anteriores à 2.9.0

`/var/log/cfn-init.log`

Esse é o log CloudFormation de inicialização. Ele contém todos os comandos que foram executados quando uma instância foi configurada. É útil para solucionar problemas de inicialização

Todas as versões

`/var/log/nodewatcher`

Esse é o log `nodewatcher`. Daemons `nodewatcher` que são executados em cada nó de computação ao usar programadores SGE e Torque. Eles reduzem verticalmente a escala de um nó se ele estiver ocioso. Esse log é útil para qualquer problema relacionado à redução vertical da escala dos recursos.

Solução de problemas de falha nas operações de inicialização e junção

- Logs aplicáveis:
  - `/var/log/cfn-init-cmd.log` (nó principal e nó de computação)
  - `/var/log/sqswatcher` (nó principal)
- Se os nós falharem na inicialização, verifique o log `/var/log/cfn-init-cmd.log` para ver a mensagem de erro específica. Na maioria dos casos, as falhas de inicialização do nó são causadas por uma falha na configuração.
- Se os nós de computação não conseguirem ingressar na configuração do programador, apesar da configuração bem-sucedida, verifique o log `/var/log/sqswatcher` para ver se o evento `sqswatcher` foi processado. Na maioria dos casos, esses problemas ocorrem porque `sqswatcher` não processou o evento.

Solucionar problemas de escala

- Logs aplicáveis:
  - `/var/log/jobwatcher` (nó principal)
  - `/var/log/nodewatcher` (nó de computação)

- Problemas de aumento de escala: para o nó principal, verifique o log `/var/log/jobwatcher` para ver se o daemon `jobwatcher` calculou o número adequado de nós necessários e atualizou o Auto Scaling Group. Observe que `jobwatcher` monitora a fila do programador e atualiza o Auto Scaling Group.
- Problemas de redução de escala: para nós de computação, verifique o log `/var/log/nodewatcher` no nó problemático para ver por que a escala do nó foi reduzida verticalmente. Observe que os daemons `nodewatcher` reduzem verticalmente a escala de um nó de computação se ele estiver ocioso.

Solução de outros problemas relacionados ao cluster

Um problema conhecido é que as notas de computação aleatórias falham em clusters de grande escala, especificamente aqueles com 500 ou mais nós de computação. Esse problema está relacionado a uma limitação da arquitetura de escalabilidade do cluster de fila única. Se você quiser usar um cluster de grande escala, estiver usando a AWS ParallelCluster versão v2.9.0 ou posterior, estiver usando e quiser evitar esse problemaSlurm, você deve atualizar e mudar para um cluster compatível com o modo de várias filas. Para fazer isso, execute `pcluster-config convert`.

Para ultra-large-scale clusters, talvez seja necessário um ajuste adicional em seu sistema. Para obter mais informações, entre em contato Suporte.

Grupos de posicionamento e problemas de execução de instâncias

Para obter a menor latência entre os nós, use um grupo de posicionamento. Um placement group garante que suas instâncias estejam no mesmo suporte principal de rede. Se não houver instâncias suficientes disponíveis quando a solicitação é feita, um erro `InsufficientInstanceCapacity` será gerado. Para reduzir a possibilidade de receber esse erro ao usar grupos de posicionamento de cluster, defina o parâmetro `placement_group` para `DYNAMIC` e defina o parâmetro `placement` como `compute`.

Se você precisar de um sistema de arquivos compartilhado de alto desempenho, considere usar FSx o Lustre.

Se o nó principal precisar estar no grupo de posicionamento, use o mesmo tipo de instância e sub-rede para os nós principais e também para os nós de computação. Ao fazer isso, o parâmetro `compute_instance_type` terá o mesmo valor que o parâmetro `master_instance_type`, o parâmetro `placement` é definido como `cluster` e o parâmetro `compute_subnet_id` não é

especificado. Com essa configuração, o valor do parâmetro `master_subnet_id` é usado para os nós de computação.

Para ter mais informações, consulte Solucionar problemas de inicialização de instâncias e Perfis e limitações de grupos de posicionamento no Guia do usuário do Amazon EC2.

Diretórios que não podem ser substituídos

Os diretórios a seguir são compartilhados entre os nós e não podem ser substituídos.

`/home`

Isso inclui a pasta base do usuário padrão (`/home/ec2_user` no Amazon Linux, `/home/centos` no CentOS, e `/home/ubuntu` no Ubuntu).

`/opt/intel`

Isso inclui Intel MPI, Intel Parallel Studio e arquivos relacionados.

`/opt/sge`

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Isso inclui Son of Grid Engine e arquivos relacionados. (Condicional, somente se `scheduler = sge`.)

`/opt/slurm`

Isso inclui Slurm Workload Manager e arquivos relacionados. (Condicional, somente se `scheduler = slurm`.)

`/opt/torque`

> **Note**
>
> A partir da versão 2.11.5, AWS ParallelCluster não oferece suporte ao uso de agendadores SGE ou Torque agendadores.

Isso inclui Torque Resource Manager e arquivos relacionados. (Condicional, somente se `scheduler = torque`.)

Solução de problemas no Amazon DCV

Tópicos

Logs do Amazon DCV

Os logs do Amazon DCV são gravados em arquivos no diretório `/var/log/dcv/`. A revisão desses logs pode ajudar a solucionar problemas.

Memória do tipo de instância do Amazon DCV

O tipo de instância deve ter pelo menos 1,7 gibibytes (GiB) de RAM para executar o Amazon DCV. Os tipos de instância Nano e micro não têm memória suficiente para executar o Amazon DCV.

Problemas no Ubuntu Amazon DCV

Ao executar o Gnome Terminal em uma sessão DCV no Ubuntu, você pode não ter acesso automático ao ambiente do usuário AWS ParallelCluster disponibilizado por meio do shell de login. O ambiente do usuário fornece módulos de ambiente, como openmpi ou intelmpi, e outras configurações do usuário.

As configurações padrão do Gnome Terminal evitam que o shell inicie como um shell de login. Isso significa que os perfis de shell não são fornecidos automaticamente e o ambiente AWS ParallelCluster do usuário não é carregado.

Para obter corretamente o perfil do shell e acessar o ambiente do AWS ParallelCluster usuário, faça o seguinte:

- Altere as configurações do terminal padrão:
  1. Escolha o menu Editar no terminal Gnome.
  2. Selecione Preferências, e em seguida, Perfis.

3. Escolha Comando e selecione Executar comando como shell de login.
4. Abrir um novo terminal.

- Use a linha de comando para obter os perfis disponíveis:

```
$ source /etc/profile && source $HOME/.bashrc
```

Solução de problemas em clusters com AWS Batch integração

Esta seção é relevante para clusters com integração com AWS Batch agendadores.

Problemas no nó principal

Os problemas de configuração relacionados ao nó principal podem ser solucionados da mesma forma que um cluster de fila única. Para obter mais informações sobre esses problemas, consulte Solução de problemas em clusters em modo de fila única.

AWS Batch problemas de envio de trabalhos paralelos de vários nós

Se você tiver problemas ao enviar trabalhos paralelos de vários nós ao usar AWS Batch como agendador de trabalhos, atualize para a AWS ParallelCluster versão 2.5.0. Se isso não for viável, você pode usar a solução alternativa detalhada no tópico: Autocorreção de um cluster usado para enviar trabalhos paralelos de vários nós com o AWS Batch.

Problemas de computação

AWS Batch gerencia os aspectos de escalabilidade e computação de seus serviços. Se você encontrar problemas relacionados à computação, consulte a documentação de AWS Batch solução de problemas para obter ajuda.

Falhas de trabalhos

Se um trabalho falhar, você poderá executar o comando `awsbout` para recuperar a saída do trabalho. Você também pode executar o `awsbstat -d` comando para obter um link para os registros de trabalhos armazenados pela Amazon CloudWatch.

Solução de problemas quando um recurso não é criado

Esta seção é relevante para recursos de cluster quando eles não são criados.

Quando um recurso falha na criação, ParallelCluster retorna uma mensagem de erro como a seguinte.

```
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)
}
```

Por exemplo, se você ver a mensagem de status exibida na resposta de comando anterior, deverá usar tipos de instância que não excedam seu limite atual de vCPU ou solicitar mais capacidade de vCPU.

Você também pode usar o CloudFormation console para ver informações sobre o "`Cluster creation failed`" status.

Veja as mensagens de CloudFormation erro do console.

1. Faça login no Console de gerenciamento da AWS e navegue até https://console.aws.amazon.com/cloudformation.
2. Selecione a pilha chamada parallelcluster-. *cluster_name*
3. Escolha a guia Eventos.
4. Verifique o status do recurso que não foi criado percorrendo a lista de eventos do recurso por ID lógico. Se houver falha na criação de uma subtarefa, retroceda para encontrar o evento de recurso com falha.
5. Um exemplo de mensagem AWS CloudFormation de erro:

```
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].
```

Solução de problemas de tamanho da política do IAM

Consulte IAM e AWS STS cotas, requisitos de nome e limites de caracteres para verificar as cotas nas políticas gerenciadas vinculadas às funções. Se o tamanho de uma política gerenciada exceder a cota, divida a política em duas ou mais políticas. Se você exceder a cota do número de políticas anexadas a um perfil do IAM, crie funções adicionais e distribua as políticas entre elas para atingir a cota.

Suporte adicional

Para obter uma lista de problemas conhecidos, consulte a página principal GitHubdo Wiki ou a página de problemas. Para problemas mais urgentes, entre em contato Suporte ou abra um novo GitHub problema.

AWS ParallelCluster política de suporte

AWS ParallelCluster suporta vários lançamentos ao mesmo tempo. Cada AWS ParallelCluster lançamento tem uma data programada de End of Support Life (EOSL). Após a data do EOSL, nenhum suporte ou manutenção adicional é fornecido para essa versão.

AWS ParallelCluster usa um esquema de `major.minor.patch` versão. Novos recursos, melhorias de desempenho, atualizações de segurança e correções de erros estão incluídos nas novas versões secundárias da versão principal mais recente. As versões secundárias são compatíveis com versões anteriores em uma versão principal. Para problemas críticos, AWS fornece correções por meio de lançamentos de patches, mas somente para as versões secundárias mais recentes de versões que não chegaram ao EOSL. Se você quiser usar as atualizações de uma nova versão, precisará atualizar para a nova versão secundária ou patch.

AWS ParallelCluster versões	Data de fim da vida útil suportada (EOSL)
2.10.4 e anterior	31/12/2021
2.11. *x*	31/12/2022

Segurança em AWS ParallelCluster

A segurança na nuvem AWS é a maior prioridade. Como AWS cliente, você se beneficia de uma arquitetura de data center e rede criada para atender aos requisitos das organizações mais sensíveis à segurança.

A segurança é uma responsabilidade compartilhada entre você AWS e você. O modelo de responsabilidade compartilhada descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem — AWS é responsável por proteger a infraestrutura que executa AWS os serviços na AWS nuvem. AWS também fornece serviços que você pode usar com segurança. Auditores terceirizados testam e verificam regularmente a eficácia de nossa segurança como parte dos Programas de Conformidade Programas de AWS de . Para saber mais sobre os programas de conformidade aplicáveis AWS ParallelCluster, consulte AWS Serviços no escopo do programa de conformidade AWS .
- Segurança na nuvem — Sua responsabilidade é determinada pelo AWS serviço ou serviços específicos que você usa. Você também é responsável por diversos outros fatores relacionados, incluindo a confidencialidade dos dados, os requisitos da empresa e as leis e regulamentos aplicáveis.

Esta documentação descreve como você deve aplicar o modelo de responsabilidade compartilhada ao usar AWS ParallelCluster. Os tópicos a seguir mostram como configurar para atender AWS ParallelCluster aos seus objetivos de segurança e conformidade. Você também aprende a usar de uma AWS ParallelCluster forma que o ajude a monitorar e proteger seus AWS recursos.

Tópicos

Informações de segurança para serviços usados por AWS ParallelCluster

- Segurança no Amazon EC2
- Segurança no Amazon API Gateway
- Segurança em AWS Batch
- Segurança em CloudFormation
- Segurança na Amazon CloudWatch
- Segurança em AWS CodeBuild
- Segurança no Amazon DynamoDB
- Segurança no Amazon ECR
- Segurança no Amazon ECS
- Segurança no Amazon EFS
- Segurança no FSx Lustre
- Segurança em AWS Identity and Access Management (IAM)
- Segurança no EC2 Image Builder
- Segurança em AWS Lambda
- Segurança no Amazon Route 53
- Segurança no Amazon SNS
- Segurança no Amazon SQS (para a AWS ParallelCluster versão 2.x.)
- Segurança no Amazon S3
- Segurança no Amazon VPC

Proteção de dados em AWS ParallelCluster

O modelo de responsabilidade AWS compartilhada modelo se aplica à proteção de dados em. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todos os Nuvem AWS. Você é responsável por manter o controle sobre o conteúdo hospedado nessa infraestrutura. Você também é responsável pelas tarefas de configuração e gerenciamento de segurança dos Serviços da AWS que usa. Para saber mais sobre a privacidade de dados, consulte as Data Privacy FAQ. Para saber mais sobre a proteção de dados na Europa,

consulte a postagem do blog AWS Shared Responsibility Model and RGPD no Blog de segurança da AWS .

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS as credenciais e configure usuários individuais com Centro de Identidade do AWS IAM ou AWS Identity and Access Management (IAM). Dessa maneira, cada usuário receberá apenas as permissões necessárias para cumprir suas obrigações de trabalho. Recomendamos também que você proteja seus dados das seguintes formas:

- Use uma autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com AWS os recursos. Exigimos TLS 1.2 e recomendamos TLS 1.3.
- Configure a API e o registro de atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte Como trabalhar com CloudTrail trilhas no Guia AWS CloudTrail do usuário.
- Use soluções de AWS criptografia, juntamente com todos os controles de segurança padrão Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sensíveis armazenados no Amazon S3.
- Se você precisar de módulos criptográficos validados pelo FIPS 140-3 ao acessar AWS por meio de uma interface de linha de comando ou de uma API, use um endpoint FIPS. Para saber mais sobre os endpoints FIPS disponíveis, consulte Federal Information Processing Standard (FIPS) 140-3.

É altamente recomendável que nunca sejam colocadas informações confidenciais ou sensíveis, como endereços de e-mail de clientes, em tags ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com ou Serviços da AWS usa o console, a API ou AWS SDKs. AWS CLI Quaisquer dados inseridos em tags ou em campos de texto de formato livre usados para nomes podem ser usados para logs de faturamento ou de diagnóstico. Se você fornecer um URL para um servidor externo, é fortemente recomendável que não sejam incluídas informações de credenciais no URL para validar a solicitação nesse servidor.

Criptografia de dados

Um recurso fundamental de qualquer serviço seguro é que as informações sejam criptografadas quando não estão sendo usadas ativamente.

Criptografia em repouso

AWS ParallelCluster por si só, não armazena nenhum dado do cliente além das credenciais necessárias para interagir com os AWS serviços em nome do usuário.

Para dados nos nós do cluster, os dados podem ser criptografados em repouso.

Para volumes do Amazon EBS, a criptografia é configurada usando as `ebs_kms_key_id` configurações na `[ebs]`seção para a AWS ParallelCluster versão 2.x.) Para ter mais informações, consulte Amazon EBS encryption no Guia do usuário do Amazon EC2.

Para volumes do Amazon EFS, a criptografia é configurada usando as `efs_kms_key_id` configurações `encrypted` e na `[efs]`seção (na AWS ParallelCluster versão 2.x). Para obter mais informações, consulte Como funciona a Criptografia de dados em repouso no Guia do usuário do Amazon Elastic File System.

FSx Para sistemas de arquivos Lustre, a criptografia de dados em repouso é ativada automaticamente ao criar um sistema de FSx arquivos da Amazon. Para obter mais informações, consulte Criptografar dados em repouso no Guia do usuário do Amazon FSx for Lustre.

Para tipos de instância com NVMe volumes, os dados nos volumes de armazenamento de NVMe instâncias são criptografados usando uma cifra XTS-AES-256 implementada em um módulo de hardware na instância. As chaves de criptografia são geradas usando o módulo de hardware e são exclusivas para cada dispositivo de armazenamento de NVMe instância. Todas as chaves de criptografia são destruídas quando a instância é interrompida ou encerrada e não podem ser recuperadas. Você não pode desativar essa criptografia e não pode fornecer sua própria chave de criptografia. Para ter mais informações, consulte Criptografia em repouso no Guia do usuário do Amazon EC2.

Se você costuma AWS ParallelCluster invocar um AWS serviço que transmite dados do cliente para seu computador local para armazenamento, consulte o capítulo Segurança e Conformidade no Guia do Usuário desse serviço para obter informações sobre como esses dados são armazenados, protegidos e criptografados.

Criptografia em trânsito

Por padrão, todos os dados transmitidos do computador cliente em execução AWS ParallelCluster e dos pontos finais do AWS serviço são criptografados enviando tudo por meio de uma HTTPS/TLS conexão. O tráfego entre os nós no cluster pode ser criptografado automaticamente, dependendo

dos tipos de instância selecionados. Para ter mais informações, consulte Criptografia em trânsito no Guia do usuário do Amazon EC2.

Consulte também

- Proteção de dados no Amazon EC2
- Proteção de dados no EC2 Image Builder
- Proteção de dados em CloudFormation
- Proteção de dados no Amazon EFS
- Proteção de dados no Amazon S3
- Proteção de dados no FSx Lustre

Identity and Access Management para AWS ParallelCluster

AWS ParallelCluster usa funções para acessar seus AWS recursos e seus serviços. As políticas de instância e usuário AWS ParallelCluster usadas para conceder permissões estão documentadas emAWS Identity and Access Management funções em AWS ParallelCluster.

A única diferença importante é como você faz a autenticação ao usar um usuário do padrão e credenciais de longo prazo. Embora um usuário exija uma senha para acessar o console de um AWS serviço, esse mesmo usuário precisa de um par de chaves de acesso para realizar as mesmas operações usando AWS ParallelCluster. Todas as outras credenciais de curto prazo são usadas da mesma maneira que são usadas com o console.

As credenciais usadas pelo AWS ParallelCluster são armazenadas em arquivos de texto simples e não são criptografadas.

- O arquivo `$HOME/.aws/credentials` armazena credenciais de longo prazo necessárias para acessar os recursos da AWS . Isso inclui o ID de chave de acesso e a chave de acesso secreta.
- Credenciais de curto prazo, como aquelas para funções que você assume ou que são para Centro de Identidade do AWS IAM serviços, também são armazenadas no `$HOME/.aws/cli/cache` e $HOME/. aws/sso/cachepastas, respectivamente.

Mitigação de riscos

- É altamente recomendável que você configure as permissões do sistema de arquivos na pasta `$HOME/.aws` e suas respectivas pastas e arquivos filho para restringir o acesso somente a utilizadores autorizados.
- Use funções com credenciais temporárias sempre que possível para reduzir a oportunidade de danos se as credenciais estiverem comprometidas. Use credenciais de longo prazo apenas para solicitar e atualizar credenciais de função de curto prazo.

Validação de conformidade para AWS ParallelCluster

Auditores terceirizados avaliam a segurança e a conformidade dos AWS serviços como parte de vários programas de AWS conformidade. Usar AWS ParallelCluster para acessar um serviço não altera a conformidade desse serviço.

Para obter uma lista de AWS serviços no escopo de programas de conformidade específicos, consulte AWS serviços no escopo por programa de conformidade AWS . Para obter informações gerais, consulte programas de AWS conformidade programas AWS de .

Você pode baixar relatórios de auditoria de terceiros usando AWS Artifact o. Para obter mais informações, consulte Fazer download de relatórios no AWS Artifact.

Sua responsabilidade de conformidade ao usar AWS ParallelCluster é determinada pela confidencialidade de seus dados, pelos objetivos de conformidade de sua empresa e pelas leis e regulamentações aplicáveis. AWS fornece os seguintes recursos para ajudar na conformidade:

- Guias de início rápido sobre sobre segurança e conformidade — Esses guias de implantação discutem considerações arquitetônicas e fornecem etapas para a implantação de ambientes básicos com foco em segurança e conformidade em. AWS
- Whitepaper sobre arquitetura para segurança e conformidade com a HIPAA na Amazon Web Services — Este AWS whitepaper descreve como as empresas podem usar para criar aplicativos compatíveis com a HIPAA. AWS
- AWS recursos de https://aws.amazon.com/compliance/resources/ de conformidade — Essa coleção de pastas de trabalho e guias pode ser aplicada ao seu setor e local.
- Avaliação de recursos com as regras do Guia do AWS Config Desenvolvedor — O AWS Config serviço avalia se suas configurações de recursos estão em conformidade com as práticas internas, as diretrizes e os regulamentos do setor.

- AWS Security Hub CSPM— Esse AWS serviço fornece uma visão abrangente do seu estado de segurança interno, AWS que ajuda você a verificar sua conformidade com os padrões e as melhores práticas do setor de segurança.

Impor uma versão mínima do TLS 1.2

Para aumentar a segurança ao se comunicar com AWS os serviços, você deve configurá-lo AWS ParallelCluster para usar o TLS 1.2 ou posterior. Quando você usa AWS ParallelCluster, o Python é usado para definir a versão do TLS.

Para garantir que não AWS ParallelCluster use uma versão TLS anterior ao TLS 1.2, talvez seja necessário recompilar o OpenSSL para impor esse mínimo e, em seguida, recompilar o Python para usar o OpenSSL recém-criado.

Determinar os protocolos atualmente compatíveis

Primeiro, crie um certificado autoassinado a ser usado para o servidor de teste e o Python SDK usando OpenSSL.

```
$ openssl req -subj '/CN=localhost' -x509 -newkey rsa:4096 -nodes -keyout key.pem -out
 cert.pem -days 365
```

Depois, crie um servidor de teste usando OpenSSL.

```
$ openssl s_server -key key.pem -cert cert.pem -www
```

Em uma nova janela do terminal, crie um ambiente virtual e instale o Python SDK.

```
$ python3 -m venv test-env
source test-env/bin/activate
pip install botocore
```

Crie um script Python chamado `check.py` que usa a biblioteca HTTP subjacente do SDK.

```
$ 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'))
```

Execute o novo script.

```
$ python check.py
```

Ele exibe detalhes sobre a conexão feita. Procure “Protocol : ” na saída. Se a saída for "TLSv1.2" ou posterior, o SDK usará como padrão o TLS v1.2 ou posterior. Se for uma versão anterior, é necessário recompilar o OpenSSL e recompilar o Python.

No entanto, mesmo que a instalação do Python defina como padrão TLS v1.2 ou posterior, ainda será possível que o Python renegocie para uma versão anterior ao TLS v1.2 se o servidor não for compatível com TLS v1.2 ou posterior. Para verificar se o Python não renegocia automaticamente para versões anteriores, reinicie o servidor de teste com o seguinte.

```
$ openssl s_server -key key.pem -cert cert.pem -no_tls1_3 -no_tls1_2 -www
```

Se estiver usando uma versão anterior do OpenSSL, talvez você não tenha o sinalizador `-no_tls_3` disponível. Se esse for o caso, remova o sinalizador porque a versão do OpenSSL que você está usando não é compatível com TLS v1.3. Execute novamente o script Python.

```
$ python check.py
```

Se a instalação do Python não renegocia para versões anteriores ao TLS 1.2 corretamente, você deve receber um erro SSL.

```
$ 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)')))
```

Se você conseguir fazer uma conexão, será necessário recompilar o OpenSSL e o Python para desabilitar a negociação de protocolos anteriores ao TLS v1.2.

Compilar OpenSSL e Python

Para garantir que isso AWS ParallelCluster não negocie nada anterior ao TLS 1.2, você precisa recompilar o OpenSSL e o Python. Para fazer isso, copie o seguinte conteúdo para criar um script e executá-lo.

```
#!/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
```

Isso compila uma versão do Python que tem um OpenSSL vinculado estaticamente que não negocia automaticamente nada anterior ao TLS 1.2. Isso também instala o OpenSSL no diretório `/opt/openssl-with-min-tls1_2` e instala o Python no diretório `/opt/python-with-min-tls1_2`. Depois de executar esse script, verifique a instalação da nova versão do Python.

```
$ /opt/python-with-min-tls1_2/bin/python3 --version
```

Isso deve imprimir o seguinte.

```
Python 3.8.1
```

Para verificar se essa nova versão do Python não negocia uma versão anterior ao TLS 1.2, execute novamente as etapas em Determinar os protocolos atualmente compatíveis usando a versão do Python recém-instalada (ou seja, `/opt/python-with-min-tls1_2/bin/python3`).

Notas de release e histórico de documentos

A tabela a seguir descreve as principais atualizações e novos atributos para o Guia do usuário do AWS ParallelCluster . Também atualizamos a documentação com frequência para abordar os comentários enviados por você.

Alteração	Descrição	Data
Versão somente para documentação	AWS ParallelCluster guia do usuário específico da versão 2 publicado. Versão somente para documentação: • AWS ParallelCluster a versão 2 tem seu próprio guia de usuário separado.	17 de julho de 2023
AWS ParallelCluster versão 2.11.9 lançada	AWS ParallelCluster versão 2.11.9 lançada. Correções de erros: • Evite a substituição de sistemas de arquivos gerenciados FSx pelo Lustre e a perda de dados em atualizações de cluster que incluam alterações no. `vpc_security_group_id`	2 de dezembro de 2022

Para obter detalhes sobre as mudanças, consulte o `CHANGELOG` arquivo do pacote aws-parallelcluster em. GitHub

AWS ParallelCluster versão 2.11.8 lançada	AWS ParallelCluster versão 2.11.8 lançada.
	
	Alterações:
	
	• Biblioteca Intel MPI atualizada para a versão 2021 atualização 6 (atualizada a partir da versão 2021 atualização 4). Para obter mais informações, consulte Intel® MPI Library 2021 Update 6.
	• Atualização do instalador EFA para 1.19.0
	  • Driver EFA: `efa-1.16.0-1`
	  • Efa-config: `efa-config-1.11-1` (era `efa-config-1.9-1`)
	  • Perfil EFA: `efa-profile-1.5-1` (sem alteração)
	  • Libfabric-aws: `libfabric-aws-1.16.0-1` (de `libfabric-1.13.2`)
	  • Núcleo RDMA: `rdma-core-41.0-2` (era `rdma-core-37.0`)
	  • Open MPI: `openmpi40-aws-4.1.4-3` (era `openmpi40-aws-4.1.1-2`)
		14 de novembro de 2022

	• Atualize o tempo de execução do Python, usado pelas funções do Lambda na integração AWS Batch , para python3.9. Correções de erros: • Evite que as tags de cluster sejam alteradas durante uma atualização por não serem compatíveis. Para obter detalhes sobre as mudanças, consulte os `CHANGELOG` arquivos do pacote aws-parallelcluster em. GitHub	
AWS ParallelCluster versão 2.11.7 lançada	AWS ParallelCluster versão 2.11.7 lançada. Alterações: • Atualização do Slurm para a versão 20.11.9. Para obter detalhes sobre as mudanças, consulte os `CHANGELOG` arquivos do pacote aws-parallelcluster em. GitHub	13 de maio de 2022

AWS ParallelCluster versão 2.11.6 lançada	AWS ParallelCluster versão 2.11.6 lançada. Melhorias: • Melhoria do gerenciamento de exceções em caso de falta de rede. Alterações: • Atualizações no pacote do sistema operacional e correções de segurança. Para obter detalhes sobre as mudanças, consulte os `CHANGELOG` arquivos do pacote aws-parallelcluster em. GitHub	19 de abril de 2022

AWS ParallelCluster versão 2.11.5 lançada	AWS ParallelCluster versão 2.11.5 lançada. Melhorias: • Adicione suporte `NEW_CHANGED_DELETED` como valor FSx para a `AutoImportPolicy` opção Lustre. • Remova o suporte para programadores Torque e SGE. • Desative o serviço `log4j-cve-2021-44228-hotpatch` no Amazon Linux para evitar uma possível degradação do desempenho. Alterações: • Atualização do driver NVIDIA para a versão `470.103.01` (era `470.82.01`). • Atualização do NVIDIA Fabric Manager para a versão `470.103.01` (era `470.82.01`). • Atualização da bibliotec a CUDA para a versão `11.4.4` (de `11.4.3`).	1º de março de 2022

- Intel MPI atualizado para a versão 2021 atualizaç ão 4 (atualizado a partir da versão 2019 atualização 8). Para obter mais informaçõ es, consulte Intel® MPI Library 2021 Update 4.
- Extensão do tempo limite de criação do nó principal para uma hora.

Correções de erros:

- Correção da conexão DCV por meio de navegadores.
- Correção de menções YAML para evitar que tags personalizadas sejam analisadas como números.

Para obter detalhes sobre as mudanças, consulte os `CHANGELOG` arquivos do pacote aws-parallelcluster em. GitHub

AWS ParallelCluster versão 2.11.4 lançada	AWS ParallelCluster versão 2.11.4 lançada. As alterações do incluem: • Suporte para CentOS 8 removido. O CentOS 8 chega ao fim da vida útil (EOL) em 31 de dezembro de 2021. • Atualização do Slurm Workload Manager para a versão 20.11.8. • Atualização do Cinc Client para `17.2.29`. • Amazon DCV atualizado para Amazon DCV 2021.2-11190. Para ter mais informações, consulte DCV 2021.2-11190 — October 11, 2021 no Guia do administrador do Amazon DCV. • Atualização do driver NVIDIA para a versão `470.82.01` (era `460.73.01`). • Atualização da biblioteca CUDA para a versão `11.4.3` (de `11.3.0`). • Atualização do NVIDIA Fabric Manager para `470.82.01`. • Desabilitação da atualização do pacote no momento do	20 de dezembro de 2021

lançamento da instância no Amazon Linux 2.

- Desabilitação da atualizaç ão autônoma de pacotes em Ubuntu e Amazon Linux 2.
- Instalação da versão Python 3 dos scripts auxiliares do CloudFormation em CentOS 7 e Ubuntu 18.04. (Já foram usados no Amazon Linux 2 e no Ubuntu 20.04.)

As correções incluem:

- Desabilitação da atualização do parâmetro `ec2_iam_r ole` .
- Correção da configuração `CpuOptions` no modelo de execução para instância s T2.

Para obter detalhes sobre as mudanças, consulte os `CHANGELOG` arquivos do aws-parallelcluster e os pacotes em. aws-parallelcluste r-cookbookaws-parallelcluster-node GitHub

AWS ParallelCluster versão 2.11.3 lançada	AWS ParallelCluster versão 2.11.3 lançada. • Correção da falha `pcluster createami` devido ao fato de as fontes de Son of Grid Engine não estarem disponíveis em `arc.liv.ac.uk` . Atualização do instalador Elastic Fabric Adapter para 1.14.1 (de 1.13.0) • Config. EFA: `efa-config-1.9-1` (era `efa-config-1.9`) • Perfil EFA: `efa-profile-1.5-1` (sem alteração) • Módulo Kernel EFA: `efa-1.14.2` (era `efa-1.13.0`) • Núcleo RDMA: `rdma-core-37.0` (de `rdma-core-35.0amzn`) • Libfabric: `libfabric-1.13.2` (de `libfabric-1.13.0amzn1.0`) • Open MPI: `openmpi40-aws-4.1.1-2` (sem alteração)	3 de novembro de 2021

GPUDirect O RDMA está sempre ativado se for compatível com o tipo de instância.

- As opções de configuração `enable_efa_gdr` e `enable_efa_gdr` não entram em vigor.

Para obter detalhes sobre as mudanças, consulte os `CHANGELOG` arquivos do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbookaws-parallelcluster-node GitHub

AWS ParallelCluster versão 2.11.2 lançada	AWS ParallelCluster versão 2.11.2 lançada.
	
	As alterações do incluem:
	
	• Não instale o EFA com o GPUDirect RDMA (GDR) habilitado no momento da inicialização se o EFA estiver instalado na AMI base.
	• Bloqueie a versão do `nvidia-fabricmanager` pacote para permanecer sincronizada com a versão do driver NVIDIA instalada pelo AWS ParallelCluster.
	• Slurm: correção do problema causado quando o cluster era interrompido e reiniciado enquanto um nó estava sendo ativado.
	• Instalador do Elastic Fabric Adapter atualizado para 1.13.0:
	  • Config. EFA: `efa-config-1.9` (sem alteração)
	  • Perfil EFA: `efa-profile-1.5-1` (sem alteração)
	  • Módulo Kernel EFA: `efa-1.13.0` (sem alteração)
	  • Núcleo RDMA: `rdma-core-35.0amzn` (de
		27 de agosto de 2021

`rdma-core-32.1amzn`)

- Libfabric: `libfabric -1.13.0amzn1.0` (de `libfabric-1.11.2am zn1.1`)
- Open MPI: `openmpi40 -aws-4.1.1-2` (sem alteração)

- Ao usar uma AMI personali zada com um pacote EFA pré-instalado, nenhuma alteração é feita no EFA no momento da inicialização do nó. A implantação original do pacote EFA é preservad a.

Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluste r-cookbook GitHub

AWS ParallelCluster versão 2.11.1 lançada	AWS ParallelCluster versão 2.11.1 lançada.
	
	As alterações do incluem:
	
	• Montagem de sistemas de arquivos usando a opção de montagem `noatime` para interromper a gravação do último horário de acesso quando um arquivo é lido. Isso melhora o desempenho do sistema de arquivos remoto.
	• Instalador do Elastic Fabric Adapter atualizado para 1.12.3:
	  • Config. EFA: `efa-config-1.9` (era `efa-config-1.8-1`)
	  • Perfil EFA: `efa-profile-1.5-1` (sem alteração)
	  • Módulo Kernel EFA: `efa-1.13.0` (era `efa-1.12.3`)
	  • Núcleo RDMA: `rdma-core-32.1amzn` (sem alteração)
	  • Libfabric: `libfabric-1.11.2amzn1.1` (sem alteração)
	  • Open MPI: `openmpi40-aws-4.1.1-2` (sem alteração)
		23 de julho de 2021

- Tente novamente as instalações do `aws-parallelcluster` pacote no nó principal ao usar AWS Batch como agendador.
- Evite falhas ao criar SGE um tipo de instância com mais de 31 CPUs v.
- Fixado na versão 1.247347.6 do CloudWatch Amazon Agent para evitar problemas vistos na versão 1.247348.0.

Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbook GitHub

AWS ParallelCluster versão 2.11.0 lançada	AWS ParallelCluster versão 2.11.0 lançada.
	
	As alterações do incluem:
	
	• Foi adicionado suporte para Ubuntu 20.04 (`ubuntu2004`) e removido o suporte para Ubuntu 16.04 (`ubuntu1604`) e Amazon Linux (`alinux`). O Amazon Linux 2 (`alinux2`) continua com suporte integral. Para obter mais informações, consulte `base_os`.
	• Foi removido o suporte para versões de Python abaixo de 3.6.
	• O tamanho do volume raiz padrão aumentou para 35 gibibytes (GiB). Para obter mais informações, consulte `compute_root_volume_size` e `master_root_volume_size` .
	• Instalador do Elastic Fabric Adapter atualizado para 1.12.2:
	  • Config. EFA: `efa-config-1.8-1` (era `efa-config-1.7`)
	  • Perfil EFA: `efa-profile-1.5-1` (era `efa-profile-1.4`)
		1.º de julho de 2021

  - Módulo Kernel EFA: `efa-1.12.3` (era `efa-1.10.2`)
  - Núcleo RDMA: `rdma-core-32.1amzn` (de `rdma-core-31.2amzn`)
  - Libfabric: `libfabric-1.11.2amzn1.1` (de `libfabric-1.11.1amzn1.0`)
  - Open MPI: `openmpi40-aws-4.1.1-2` (era `openmpi40-aws-4.1.0`)
- Slurm atualizado para a versão `20.11.7` (de `20.02.7`).
- Instalação do SSM Agent em `centos7` e `centos8`. (O SSM Agent está pré-instalado em `alinux2` `ubuntu1804`, e `ubuntu2004`)
- SGE: sempre usar o nome curto como filtro do nome do host com `qstat`.
- Use o serviço de metadados da instância Versão 2 (IMDSv2) em vez do serviço de metadados da instância Versão 1 (IMDSv1) para recuperar os metadados da instância. Para obter mais informações, consulte

Metadados da instância e dados do usuário no Guia do EC2 usuário da Amazon.

- Atualização do driver NVIDIA para a versão `460.73.01` (era `450.80.02`).
- Atualização da bibliotec a CUDA para a versão `11.3.0` (de `11.0`).
- Atualização do NVIDIA Fabric Manager para `nvidia-fabricmanager-460`.
- Atualize o Python usado em AWS ParallelCluster virtualenvs para (de). `3.7.10` `3.6.13`
- Atualização do Cinc Client para `16.13.16`.
- Atualize dependências de terceiros de aws-parallelcluster-cookbook:
  - `apt-7.4.0` (de `apt-7.3.0`).
  - `iptables-8.0.0` (de `iptables-7.1.0`).
  - `line-4.0.1` (de `line-2.9.0`).
  - `openssh-2.9.1` (de `openssh-2.8.1`).
  - `pyenv-3.4.2` (de `pyenv-3.1.1`).

	• `selinux-3.1.1` (de `selinux-2.1.1`). • `ulimit-1.1.1` (de `ulimit-1.0.0`). • `yum-6.1.1` (de `yum-5.1.0`). • `yum-epel-4.1.2` (de `yum-epel-3.3.0`). Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbookaws-parallelcluster-node GitHub	
AWS ParallelCluster versão 2.10.4 lançada	AWS ParallelCluster versão 2.10.4 lançada. As alterações do incluem: • Slurm atualizado para a versão `20.02.7` (de `20.02.4`). Para obter mais detalhes sobre as mudanças, consulte o arquivo CHANGELOG do pacote aws-parallelcluster em. GitHub	15 de maio de 2021

AWS ParallelCluster versão 2.10.3 lançada	AWS ParallelCluster versão 2.10.3 lançada.
	
	As alterações do incluem:
	
	• Foi adicionado suporte para Ubuntu 18.04 e Amazon Linux 2 em instâncias AWS Graviton baseadas em ARM na China e. AWS AWS GovCloud (US) Regiões da AWS
	• Instalador do Elastic Fabric Adapter atualizado para 1.11.2:
	  • Config. EFA: `efa-config-1.7` (sem alteração)
	  • Perfil EFA: `efa-profile-1.4` (era `efa-profile-1.3`)
	  • Módulo Kernel EFA: `efa-1.10.2` (sem alteração)
	  • Núcleo RDMA: `rdma-core-31.2amzn` (sem alteração)
	  • Libfabric: `libfabric-1.11.1amzn1.0` (sem alteração)
	  • Open MPI: `openmpi40-aws-4.1.0` (sem alteração)
	
	Para obter mais detalhes sobre as mudanças, consulte
		18 de março de 2021

	o arquivo CHANGELOG do pacote aws-parallelcluster em. GitHub

AWS ParallelCluster versão 2.10.2 lançada	AWS ParallelCluster versão 2.10.2 lançada. As alterações do incluem: • Melhore a validação da configuração do cluster para usar a AMI de destino do cluster ao invocar a operação da EC2 RunInstancesAPI da Amazon no `--dry-run` modo. • Atualize a versão do Python usada nos ambientes AWS ParallelCluster virtuais para 3.6.13. • Correção de `sanity_check` para tipos de instância Arm. • Correção de `enable_efa` ao usar `centos8` com o programador Slurm ou os tipos de instância Arm. • Execução da `apt update` no modo não interativo (`-y`). • Correção de `encrypted_ephemeral` = verdadeiro com `alinux2` e `centos8`. Para obter mais detalhes sobre as mudanças, consulte o arquivo CHANGELOG do pacote aws-parallelcluster em. GitHub	2 de março de 2021

AWS ParallelCluster versão 2.10.1 lançada	AWS ParallelCluster versão 2.10.1 lançada.
	
	As alterações do incluem:
	
	• Foi adicionado suporte para a África (Cidade do Cabo) (`af-south-1`), Europa (Milão) (`me-south-1`) e Oriente Médio (Bahrein) (). `me-south-1` Regiões da AWS No lançamento, o suporte é limitado das seguintes maneiras:
	  • FSx para Lustre e instâncias de Graviton baseadas em ARM não são suportadas em nenhuma delas. Regiões da AWS
	  • AWS Batch não é suportado na África (Cidade do Cabo).
	  • O Amazon EBS `io2` e os tipos de `gp3` volume não são compatíveis na África (Cidade do Cabo) e na Europa (Milão) Regiões da AWS.
	• Foi adicionado suporte para o Amazon EBS `io2` e tipos de volume `gp3`. Para mais informações, consulte a seção `[ebs]` e a seção `[raid]`.
		22 de dezembro de 2020

- Foi adicionado suporte para Elastic Fabric Adapter em instâncias Graviton2 baseadas em ARM que executam `alinux2`, `ubuntu1804` , ou `ubuntu2004` . Para obter mais informações, consulte Elastic Fabric Adapter.
- Instale as bibliotecas de desempenho Arm 20.2.1 no Arm AMIs (`alinux2centos8`, `eubuntu1804`). Para obter mais informações, consulte Bibliotecas de desempenho do Arm.
- Intel MPI atualizado para a versão 2019 atualizaç ão 8 (atualizado a partir da versão 2019 atualização 7). Para obter mais informaçõ es, consulte Intel® MPI Library 2019 Update 8.
- A chamada de operação da CloudFormation `DescribeStacks` API foi removida do ponto de entrada do AWS Batch Docker para encerrar as falhas de trabalho causadas pela limitação de. CloudFormation
- Melhorou as chamadas para a chamada de operação de EC2 `DescribeI`

`nstanceTypes` API da Amazon ao validar uma configuração de cluster.

- As imagens do Docker do Amazon Linux 2 são extraídas do Amazon ECR Public ao criar a imagem do Docker para o programador do `awsbatch`.
- O tipo de instância padrão mudou do tipo de `t2.micro` instância codificada para o tipo de instância de nível gratuito para o Região da AWS (`t2.microout3.micro`, dependendo do Região da AWS). Regiões da AWS que não têm um nível gratuito padrão para o tipo de `t3.micro` instância.
- Instalador do Elastic Fabric Adapter atualizado para 1.11.1:
  - Config. EFA: `efa-config-1.7` (era `efa-config-1.5`)
  - Perfil EFA: `efa-profile-1.3` (era `efa-profile-1.1`)
  - Módulo Kernel EFA: `efa-1.10.2` (sem alteração)
  - Núcleo RDMA: `rdma-core-31.2amzn` (de

`rdma-core-31.amzn0`)

- Libfabric: `libfabric-1.11.1amzn1.0` (de `libfabric-1.10.1amzn1.1`)
- Open MPI: `openmpi40-aws-4.1.0` (era `openmpi40-aws-4.0.5`)

- Os parâmetros `vpc_settings` , `vpc_id` e `master_subnet_id` agora são obrigatórios.
- O daemon do `nfsd` no nó principal agora está configurado para usar pelo menos 8 threads. Se houver mais de 8 núcleos, ele usará tantos threads quanto houver núcleos. Ao usar `ubuntu1604` , a configuração só muda depois que o nó é reinicializado.
- Amazon DCV atualizado para Amazon DCV 2020.2-9662. Para ter mais informações, consulte DCV 2020.2-9662 — December 04, 2020 no Guia do administrador do Amazon DCV.
- Os pacotes Intel MPI e HPC para AWS ParallelCluster são retirados do Amazon

S3. Eles não são mais retirados dos repositórios Intel yum.

- O systemd nível de execução padrão foi alterado para `multi-user.target` ativado OSs durante a criação do oficial AWS ParallelCluster AMIs. O nível de execução é definido como `graphical.target` no nó principal somente quando o DCV está ativado. Isso impede que serviços gráficos (como `x/gdm`) sejam executados quando não são necessários.
- Suporte habilitado para instâncias `p4d.24xlarge` no nó principal.
- Aumento do número máximo de tentativas ao registrar nós Slurm no Amazon Route 53.

Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbookaws-parallelcluster-node GitHub

AWS ParallelCluster versão 2.10.0 lançada	AWS ParallelCluster versão 2.10.0 lançada.
	
	As alterações do incluem:
	
	• Foi adicionado suporte para CentOS 8 no total Regiões da AWS (fora das regiões AWS da China e AWS GovCloud (EUA)). Suporte removido para CentOS 6.
	• Foi adicionado suporte para instâncias `p4d.24xlarge` de nós de computação.
	• Foi adicionado suporte para NVIDIA GPUDirect RDMA no EFA usando a nova configuração. `enable_efa_gdr`
	• Foi adicionado suporte para os recursos do Amazon FSx for Lustre.
	  • Configure seu sistema de arquivos Amazon FSx for Lustre para importar preferências usando a `auto_import_policy` configuração.
	  • Foi adicionado suporte para sistemas de arquivos Amazon FSx for Lustre baseados em HDD usando as `storage_type` configurações e. `drive_cache_type`
		18 de novembro de 2020

- Foi adicionado um CloudWatch painel da Amazon, incluindo métricas do nó principal e fácil acesso aos registros do cluster. Para obter mais informações, consulte CloudWatch Painel da Amazon.
- Foi adicionado suporte para o uso de um bucket Amazon S3 existente para armazenar informações de configuração do cluster, usando a configuração `cluster_resource_bucket`.
- Melhoria do comando `pcluster createami`.
  - Foi adicionado um parâmetro `--post-install` para usar um script de pós-instalação ao criar uma AMI.
  - Foi adicionada uma etapa de validação para falhar ao usar uma AMI básica criada por uma versão diferente do AWS ParallelCluster.
  - Foi adicionada uma etapa de validação que indica falha quando o sistema operacional selecionado

for diferente do sistema operacional na AMI base.

- Foi adicionado suporte para o uso de uma AMI AWS ParallelCluster básica.

- Melhoria do comando `pcluster update`.
  - Agora, a configuração `tags` pode ser alterada durante uma atualização.
  - As filas agora podem ser redimensionadas durante uma atualização sem interromper a frota de computação
- Parâmetro de configuração `all_or_nothing_batch` adicionado para o script `slurm_resume`. Quando for `True`, `slurm_resume` será bem-sucedida somente se todas as instâncias exigidas por todos os trabalhos pendentes em Slurm estiverem disponíveis. Para obter mais informações, consulte Apresentando `all_or_nothing_batch` lançamentos no AWS ParallelCluster Wiki em GitHub.

- Instalador do Elastic Fabric Adapter atualizado para 1.10.1:
  - Config. EFA: `efa-config-1.5` (era `efa-config-1.4`)
  - Perfil EFA: `efa-profile-1.1` (era `efa-profile-1.0.0`)
  - Módulo Kernel EFA: `efa-1.10.2` (era `efa-1.6.0`)
  - Núcleo RDMA: `rdma-core-31.amzn0` (de `rdma-core-28.amzn0`)
  - Libfabric: `libfabric-1.11.1amzn1.0` (de `libfabric-1.10.1amzn1.1`)
  - Open MPI: `openmpi40-aws-4.0.5` (era `openmpi40-aws-4.0.3`)
- Nas AWS GovCloud (US) regiões, habilite o suporte para Amazon DCV e. AWS Batch
- Nas regiões da AWS China, ative o suporte para o Amazon FSx for Lustre.
- Atualização do driver NVIDIA para a versão 450.80.02 (era 450.51.05).

- Instale o NVIDIA Fabric Manager para habilitar a NVIDIA NVSwitch em plataformas compatíveis.
- Removido o padrão Região da AWS de `us-east-1` . O padrão usa essa ordem de pesquisa.
  - Região da AWS especificado em `-r` nosso `--region` argumento.
  - Variável de ambiente `AWS_DEFAULT_REGION` .
  - `aws_region_name` configuração na `[aws]` seção do arquivo de AWS ParallelCluster configuração (o padrão é). `~/.parallelcluster/config`
  - `region` configuração na `[default]` seção do arquivo de AWS CLI configuração (o padrão é). `~/aws/config`

Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub

AWS ParallelCluster versão 2.9.0 lançada	AWS ParallelCluster versão 2.9.0 lançada. As alterações do incluem: • Foi adicionado suporte para várias filas e vários tipos de instância na frota de computação usando o Slurm Workload Manager. Ao usar filas, os grupos do Auto Scaling não são mais usados no Slurm. Uma zona hospedada do Amazon Route 53 agora é criada com o cluster e é usada para resolução de DNS de nós computacionais ao usar o programador do Slurm. Para obter mais informaçõ es, consulte Modo de fila múltipla. • Foi adicionado suporte para o Amazon DCV em instâncias baseadas em AWS Graviton baseadas em ARM. • Foi adicionado suporte para desativar o hyperthre ading em tipos de instância que não oferecem suporte às opções de CPU nos modelos de execução (por exemplo, tipos de instância `*.metal`).	11 de setembro de 2020

- Foi adicionado suporte para NFS 4 para sistemas de arquivos compartilhados a partir do nó principal.
- Foi removida a dependênc ia do cfn-init ao inicializ ar os nós de computaçã o para evitar a limitação quando um grande número de nós ingressa no cluster. CloudFormation
- Instalador do Elastic Fabric Adapter atualizado para 1.9.5:
  - Config. EFA: `efa-confi g-1.4` (era `efa-confi g-1.3`)
  - Perfil EFA: `efa-profi le-1.0.0` (novo)
  - Kernel module: `efa-1.6.0` (sem alteração)
  - Núcleo RDMA: `rdma-core-28.amzn0` (sem alteração)
  - Libfabric: `libfabric -1.10.1amzn1.1` (sem alteração)
  - Open MPI: `openmpi40 -aws-4.0.3` (sem alteração)
- Slurm atualizado para a versão `20.02.4` (de `19.05.5`).

- Amazon DCV atualizado para Amazon DCV 2020.1-9012. Para ter mais informações, consulte as DCV 2020.1-9012 — August 24, 2020 no Guia do administrador do Amazon DCV.
- Ao montar unidades NFS compartilhadas, passou a ser usado o endereço IP privado do nó principal em vez do nome do host.
- Foram adicionados novos fluxos de log aos CloudWatch Logs: `chef-client clustermgtd` , `computemgtd` , `slurm_resume` , e. `slurm_suspend`
- Foi adicionado suporte para nomes de filas em scripts de pré-instalação e pós-instalação.
- No AWS GovCloud (US) Regiões da AWS, use a opção de cobrança sob demanda do Amazon DynamoDB. Para obter mais informações, consulte Modo sob demanda no Guia do desenvolvedor do Amazon DynamoDB.

	Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbookaws-parallelcluster-node GitHub	
AWS ParallelCluster versão 2.8.1 lançada	AWS ParallelCluster versão 2.8.1 lançada. As alterações do incluem: • Desabilitação do bloqueio de tela para sessões do Amazon DCV para evitar que os usuários sejam bloqueados. • Correção do `pcluster configure` ao incluir um tipo de instância AWS baseada em Graviton e baseada em ARM. Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbookaws-parallelcluster-node GitHub	4 de agosto de 2020

AWS ParallelCluster versão 2.8.0 lançada	AWS ParallelCluster versão 2.8.0 lançada. As alterações do incluem: • Foi adicionado suporte para instâncias baseadas em AWS Graviton baseadas em ARM (como e). `A1` `C6g` • Foi adicionado suporte para os recursos de backup diário automático do Amazon FSx for Lustre. Para ter mais informaçõ es, consulte `automatic _backup_retention_ days` , `copy_tags _to_backups` , `daily_automatic_ba ckup_start_time` e `fsx_backup_id` . • Removida a dependência de Berkshelf de `pcluster createami` . • Melhoria da robustez e da experiência do usuário de `pcluster update`. Para obter mais informações, consulte Usar o `pcluster update`. • Instalador do Elastic Fabric Adapter atualizado para 1.9.4: • Módulo do kernel: `efa-1.6.0` (atualizado de `efa-1.5.1`)	23 de julho de 2020

    - Núcleo RDMA: `rdma-core-28.amzn0` (atualizado de `rdma-core-25.0`)
    - Libfabric: `libfabric-1.10.1amzn1.1` (atualizado de `libfabric-aws-1.9.0amzn1.1`)
    - Open MPI: `openmpi40-aws-4.0.3` (sem alteração)
- Atualização do driver NVIDIA para a versão 440.95.01 da Tesla em CentOS 6 e para a versão 450.51.05 em todas as outras distribuições.
- Atualização da biblioteca CUDA para a versão 11.0 em todas as distribuições, exceto CentOS 6.

Para obter mais detalhes sobre as mudanças, consulte os arquivos CHANGELOG do aws-parallelcluster e os pacotes em. aws-parallelcluster-cookbookaws-parallelcluster-node GitHub

AWS ParallelCluster versão 2.7.0 lançada	AWS ParallelCluster versão 2.7.0 lançada. As alterações do incluem: • `base_os` agora é um parâmetro obrigatório. • `scheduler` agora é um parâmetro obrigatório. • Amazon DCV atualizado para Amazon DCV 2020.0. Para ter mais informações, consulte NICE DCV lança versão 2020.0 com sistema surround 7.1 e suporte para caneta. Intel MPI atualizado para a versão 2019 atualizaç ão 7 (atualizado a partir da versão 2019 atualização 6). Para obter mais informaçõ es, consulte Intel® MPI Library 2019 Update 7. Instalador do Elastic Fabric Adapter atualizado para 1.8.4: • Kernel module: `efa-1.5.1` (sem alteração) • Núcleo RDMA: `rdma-core-25.0` (sem alteração)	19 de maio de 2020

	• Libfabric: `libfabric-aws-1.9.0amzn1.1` (sem alteração) • Open MPI: `openmpi40-aws-4.0.3` (atualizado de `openmpi40-aws-4.0.2`) • Atualize o CentOS 7 AMI para a versão 7.8-2003 (atualizada de 7.7-1908). Para obter mais informações, consulte Notas de release do CentOS-7 (2003).	
AWS ParallelCluster versão 2.6.1 lançada	AWS ParallelCluster versão 2.6.1 lançada. As alterações do incluem: • Removido `cfn-init-cmd` e `cfn-wire` dos registros armazenados no Amazon CloudWatch Logs. Para obter mais informações, consulte Integração com Amazon CloudWatch Logs.	17 de abril de 2020

AWS ParallelCluster versão 2.6.0 lançada	AWS ParallelCluster versão 2.6.0 lançada. As alterações do incluem: • Adicionado suporte para o Amazon Linux 2. • Agora, o Amazon CloudWatch Logs é usado para coletar registros do cluster e do agendador. Para obter mais informaçõ es, consulte Integração com Amazon CloudWatch Logs. • Foi adicionado suporte para novos tipos de implantaç ão do Amazon FSx for `SCRATCH_2` Lustre `PERSISTENT_1` e. Support FSx for Lustre em Ubuntu 18.04 e Ubuntu 16.04. Para obter mais informações, consulte fsx. • Suporte adicionado para o Amazon DCV no Ubuntu 18.04. Para obter mais informações, consulte Conecte-se ao nó principal por meio do Amazon DCV.	27 de fevereiro de 2020
AWS ParallelCluster versão 2.5.1 lançada	AWS ParallelCluster versão 2.5.1 lançada.	13 de dezembro de 2019
AWS ParallelCluster versão 2.5.0 lançada	AWS ParallelCluster versão 2.5.0 lançada.	18 de novembro de 2019

AWS ParallelCluster introduz suporte para Intel MPI	AWS ParallelCluster a versão 2.4.1 introduz suporte para Intel MPI.	29 de julho de 2019
AWS ParallelCluster introduz suporte para EFA	AWS ParallelCluster a versão 2.4.0 introduz suporte para o Elastic Fabric Adapter (EFA).	11 de junho de 2019
AWS ParallelCluster documentação lançada no site de AWS documentação	A AWS ParallelCluster documentação agora está disponível em 10 idiomas e nos formatos HTML e PDF.	24 de maio de 2018

As traduções são geradas por tradução automática. Em caso de conflito entre o conteúdo da tradução e da versão original em inglês, a versão em inglês prevalecerá.