



AWS ParallelCluster Guia do usuário (v2)

# AWS ParallelCluster



# AWS ParallelCluster: AWS ParallelCluster Guia do usuário (v2)

Copyright © 2024 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

As marcas comerciais e imagens comerciais da Amazon não podem ser usadas no contexto de nenhum produto ou serviço que não seja da Amazon, nem de qualquer maneira que possa gerar confusão entre os clientes ou que deprecie ou desprestígie a Amazon. Todas as outras marcas comerciais que não pertencem à Amazon pertencem a seus respectivos proprietários, que podem ou não ser afiliados, patrocinados pela Amazon ou ter conexão com ela.

# Table of Contents

O que é AWS ParallelCluster? .....	1
Preços .....	1
Configuração AWS ParallelCluster .....	2
Instalar AWS ParallelCluster .....	2
Instalar AWS ParallelCluster em um ambiente virtual (recomendado) .....	2
Instalar AWS ParallelCluster em um ambiente não virtual usando pip .....	3
Etapas a serem realizadas após a instalação .....	3
Instruções detalhadas para cada ambiente .....	4
Ambiente virtual .....	4
Linux .....	6
macOS .....	10
Windows .....	13
Configuração AWS ParallelCluster .....	16
Práticas recomendadas .....	24
Práticas recomendadas: seleção do tipo de instância do nó principal .....	24
Práticas recomendadas: desempenho da rede .....	24
Práticas recomendadas: alertas de orçamento .....	25
Melhores práticas: migrar um cluster para um novo AWS ParallelCluster versão secundária ou patch .....	26
Passando de CfnCluster para AWS ParallelCluster .....	27
Regiões compatíveis .....	28
Usando AWS ParallelCluster .....	31
Configurações de rede .....	31
AWS ParallelCluster em uma única sub-rede pública .....	32
AWS ParallelCluster usando duas sub-redes .....	33
AWS ParallelCluster em uma única sub-rede privada conectada usando AWS Direct Connect .....	34
AWS ParallelCluster com awsbatch agendador .....	35
Ações de bootstrap personalizadas .....	36
Configuração .....	38
Argumentos .....	38
Exemplo .....	38
Trabalhar com o Amazon S3 .....	40
Exemplos .....	40

Trabalho com Instâncias spot .....	41
Cenário 1: Uma instância spot sem trabalhos em execução é interrompida .....	41
Cenário 2: Uma instância spot que executa trabalhos de nó único é interrompida .....	41
Cenário 3: Uma instância spot que executa trabalhos de vários nós é interrompida .....	43
AWS Identity and Access Management funções em AWS ParallelCluster .....	45
Configurações padrão para criação de clusters .....	45
Usando uma IAM função existente para a Amazon EC2 .....	45
AWS ParallelCluster exemplo de políticas de instância e usuário .....	46
Agendadores suportados por AWS ParallelCluster .....	87
Son of Grid Engine .....	88
Slurm Workload Manager .....	88
Torque Resource Manager .....	100
AWS Batch .....	101
Tags .....	109
CloudWatch Painel da Amazon .....	112
Integração com Amazon CloudWatch Logs .....	114
Elastic Fabric Adapter .....	116
Intel Select Solutions .....	117
Ativar Intel MPI .....	119
Especificação HPC da plataforma Intel .....	120
Bibliotecas de desempenho do Arm .....	121
Conecte-se ao nó principal por meio da Amazon DCV .....	123
DCVHTTPSCertificado Amazon .....	124
Licenciamento da Amazon DCV .....	124
Utilizar o <code>pcluster update</code> .....	124
AMl aplicação de patches e substituição de EC2 instâncias .....	127
Atualização ou substituição da instância do nó principal .....	128
Limitações de armazenamento de instância .....	129
Soluções alternativas para limitações de armazenamento de instâncias .....	129
Interromper e iniciar o nó principal de um cluster .....	130
AWS ParallelCluster CLI comandos .....	133
<code>pcluster</code> .....	133
Argumentos .....	133
Subcomandos: .....	133
<code>pcluster configure</code> .....	134
<code>pcluster create</code> .....	135

pcluster createami .....	137
pcluster dcv .....	141
pcluster delete .....	143
pcluster instances .....	145
pcluster list .....	146
pcluster ssh .....	147
pcluster start .....	148
pcluster status .....	149
pcluster stop .....	150
pcluster update .....	151
pcluster version .....	153
pcluster-config .....	154
Argumentos nomeados .....	154
Configuração .....	156
Layout .....	157
Seção [global] .....	157
cluster_template .....	157
update_check .....	158
sanity_check .....	158
Seção [aws] .....	158
Seção [aliases] .....	159
Seção [cluster] .....	160
additional_cfn_template .....	162
additional_iam_policies .....	162
base_os .....	163
cluster_resource_bucket .....	165
cluster_type .....	166
compute_instance_type .....	167
compute_root_volume_size .....	167
custom_ami .....	168
cw_log_settings .....	169
dashboard_settings .....	169
dcv_settings .....	170
desired_vcpus .....	170
disable_cluster_dns .....	171
disable_hyperthreading .....	171

---

ebs_settings .....	172
ec2_iam_role .....	173
efs_settings .....	173
enable_efa .....	173
enable_efa_gdr .....	174
enable_intel_hpc_platform .....	175
encrypted_ephemeral .....	175
ephemeral_dir .....	176
extra_json .....	176
fsx_settings .....	177
iam_lambda_role .....	177
initial_queue_size .....	178
key_name .....	178
maintain_initial_size .....	179
master_instance_type .....	180
master_root_volume_size .....	180
max_queue_size .....	181
max_vcpus .....	181
min_vcpus .....	181
placement .....	182
placement_group .....	183
post_install .....	183
post_install_args .....	184
pre_install .....	184
pre_install_args .....	185
proxy_server .....	185
queue_settings .....	185
raid_settings .....	186
s3_read_resource .....	187
s3_read_write_resource .....	187
scaling_settings .....	187
scheduler .....	188
shared_dir .....	189
spot_bid_percentage .....	189
spot_price .....	190
tags .....	190

template_url .....	191
vpc_settings .....	192
Seção [compute_resource] .....	192
initial_count .....	193
instance_type .....	193
max_count .....	194
min_count .....	194
spot_price .....	195
Seção [cw_log] .....	195
enable .....	195
retention_days .....	196
Seção [dashboard] .....	196
enable .....	196
Seção [dcv] .....	197
access_from .....	198
enable .....	198
port .....	199
Seção [ebs] .....	199
shared_dir .....	200
ebs_kms_key_id .....	200
ebs_snapshot_id .....	201
ebs_volume_id .....	201
encrypted .....	201
volume_iops .....	201
volume_size .....	202
volume_throughput .....	203
volume_type .....	204
Seção [efs] .....	205
efs_fs_id .....	206
efs_kms_key_id .....	207
encrypted .....	207
performance_mode .....	207
provisioned_throughput .....	208
shared_dir .....	209
throughput_mode .....	209
Seção [fsx] .....	209

auto_import_policy .....	212
automatic_backup_retention_days .....	213
copy_tags_to_backups .....	213
daily_automatic_backup_start_time .....	214
data_compression_type .....	214
deployment_type .....	215
drive_cache_type .....	216
export_path .....	216
fsx_backup_id .....	217
fsx_fs_id .....	217
fsx_kms_key_id .....	218
import_path .....	218
imported_file_chunk_size .....	219
per_unit_storage_throughput .....	219
shared_dir .....	220
storage_capacity .....	220
storage_type .....	222
weekly_maintenance_start_time .....	223
Seção [queue] .....	224
compute_resource_settings .....	224
compute_type .....	225
disable_hyperthreading .....	226
enable_efa .....	226
enable_efa_gdr .....	226
placement_group .....	227
Seção [raid] .....	228
shared_dir .....	229
ebs_kms_key_id .....	229
encrypted .....	229
num_of_raid_volumes .....	230
raid_type .....	230
volume_iops .....	231
volume_size .....	232
volume_throughput .....	232
volume_type .....	233
Seção [scaling] .....	234

scaledown_idletime .....	234
Seção [vpc] .....	235
additional_sg .....	235
compute_subnet_cidr .....	236
compute_subnet_id .....	236
master_subnet_id .....	236
ssh_from .....	236
use_public_ips .....	237
vpc_id .....	237
vpc_security_group_id .....	237
Exemplos .....	40
Exemplo de Slurm .....	239
SGE e exemplo Torque .....	240
Exemplo de AWS Batch .....	241
Como o AWS ParallelCluster funciona .....	242
Processos do AWS ParallelCluster .....	242
SGE and Torque integration processes .....	243
Slurm integration processes .....	249
AWS serviços usados por AWS ParallelCluster .....	249
AWS Auto Scaling .....	250
AWS Batch .....	251
AWS CloudFormation .....	251
Amazon CloudWatch .....	252
CloudWatch Registros da Amazon .....	252
AWS CodeBuild .....	252
Amazon DynamoDB .....	253
Amazon Elastic Block Store .....	253
Amazon Elastic Compute Cloud .....	253
Amazon Elastic Container Registry .....	253
Amazon EFS .....	254
Amazon FSx para Lustre .....	254
AWS Identity and Access Management .....	254
AWS Lambda .....	255
Amazon DCV .....	255
Amazon Route 53 .....	255
Amazon Simple Notification Service .....	255

Amazon Simple Queue Service .....	256
Amazon Simple Storage Service .....	256
Amazon VPC .....	257
Auto Scaling do AWS ParallelCluster .....	257
Aumentar .....	258
Reduzir .....	259
Cluster estático .....	259
Tutoriais .....	260
Como executar o seu primeiro trabalho no AWS ParallelCluster .....	260
Verificar a instalação .....	260
Criação de seu primeiro cluster .....	261
Fazendo login em seu nó principal .....	261
Executar o primeiro trabalho usando o SGE .....	262
Criação de uma AMI personalizada do AWS ParallelCluster .....	263
Como personalizar a AMI do AWS ParallelCluster .....	264
Modificar uma AMI do .....	264
Criar uma AMI personalizada do AWS ParallelCluster .....	267
Usar uma AMI personalizada no tempo de execução .....	268
Execução de uma tarefa de MPI com o AWS ParallelCluster e o programador awsbatch .....	269
Criar o cluster .....	269
Fazendo login em seu nó principal .....	261
Executar o primeiro trabalho usando o AWS Batch .....	271
Executar um trabalho de MPI em um ambiente paralelo de vários nós .....	273
Criptografia de disco com uma chave do KMS personalizada .....	277
Criar a função .....	278
Conceder permissões à chave .....	278
Criar o cluster .....	269
Tutorial do modo de fila múltipla .....	279
Como executar seus trabalhos no AWS ParallelCluster com o modo de várias filas .....	279
Desenvolvimento .....	292
Configuração de um livro de receitas personalizado do AWS ParallelCluster .....	292
Etapas .....	293
Configuração de um pacote de nós personalizado do AWS ParallelCluster .....	294
Etapas .....	293
Solução de problemas .....	296
Recuperando e preservando logs .....	296

Solução de problemas de implantação de pilha .....	297
Solução de problemas em clusters em modo de várias filas .....	297
Logs de chaves .....	298
Solução de problemas de inicialização do nó .....	299
Solução de problemas inesperados de substituições e encerramentos de nós .....	301
Substituindo, encerrando ou desligando instâncias e nós problemáticos .....	302
Solucionando outros problemas conhecidos de nós e trabalhos .....	303
Solução de problemas em clusters em modo de fila única .....	303
Logs de chaves .....	303
Solução de problemas de falha nas operações de inicialização e junção .....	305
Solucionar problemas de escala .....	305
Solução de outros problemas relacionados ao cluster .....	306
Grupos de posicionamento e problemas de execução de instâncias .....	306
Diretórios que não podem ser substituídos .....	307
Solução de problemas na Amazon DCV .....	308
Registros para a Amazon DCV .....	308
Memória do tipo de DCV instância Amazon .....	308
DCVProblemas com o Ubuntu Amazon .....	308
Solução de problemas em clusters com integração AWS Batch .....	309
Problemas no nó principal .....	309
AWS Batch problemas de envio de trabalhos paralelos de vários nós .....	309
Problemas de computação .....	309
Falhas de trabalhos .....	309
Solução de problemas quando um recurso não é criado .....	309
Solução de problemas IAM de tamanho da política .....	311
Suporte adicional .....	311
Política de suporte do AWS ParallelCluster .....	312
Segurança .....	313
Informações de segurança para serviços usados pelo AWS ParallelCluster .....	314
Proteção de dados .....	314
Criptografia de dados .....	315
Consulte também .....	317
Gerenciamento de identidade e acesso .....	317
Validação de conformidade .....	318
Impor o TLS 1.2 .....	319
Determinar os protocolos atualmente compatíveis .....	319

---

Compilar OpenSSL e Python .....	321
Notas de release e histórico de documentos .....	323
.....	ccclxv

# O que é AWS ParallelCluster?

AWS ParallelCluster é uma ferramenta de gerenciamento de cluster de código aberto compatível com a AWS que ajuda a implantar e gerenciar clusters de computação de alta performance (HPC) na Nuvem AWS. Ele configura automaticamente os recursos de computação, agendador e o sistema de arquivos compartilhado necessários. Você pode usar AWS ParallelCluster com os programadores AWS Batch e Slurm.

Com AWS ParallelCluster, você pode criar e implantar rapidamente ambientes computacionais de HPC de prova de conceito e produção. Também é possível criar e implantar fluxos de trabalho de nível superior fundamentados no AWS ParallelCluster, como um portal genômico que automatiza todo o fluxo de trabalho de sequenciamento de DNA.

## Preços

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

# Configuração AWS ParallelCluster

## Tópicos

- [Instalar AWS ParallelCluster](#)
- [Configuração AWS ParallelCluster](#)
- [Práticas recomendadas](#)
- [Passando de CfnCluster para AWS ParallelCluster](#)
- [Regiões compatíveis](#)

## Instalar 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 mais recente CLI 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`.

## Instalar 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`, você pode [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.

## Instalar AWS ParallelCluster em um ambiente não virtual usando pip

O principal método de distribuição para AWS ParallelCluster no 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.

Atual AWS ParallelCluster Version (Versão)

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, você 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 de instalar 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 para seu caminho de linha de comando](#)
- macOS: [Adicione o AWS ParallelCluster executável para seu caminho de linha de comando](#)
- Windows: [Adicione o AWS ParallelCluster executável para seu caminho de linha de comando](#)

Você pode verificar isso AWS ParallelCluster instalado corretamente em execução `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, veja o [AWS ParallelCluster notas 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 \(recomendado\)](#)
- [Instalar AWS ParallelCluster no Linux](#)
- [Instalar 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. Instalar 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 isso 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"
```

## Instalar AWS ParallelCluster no Linux

Você pode instalar AWS ParallelCluster e suas dependências na maioria das distribuições Linux usando `pip`, um gerenciador 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 a seção [Instalar AWS ParallelCluster com pip](#) topic. 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 a seção Instalar [AWS ParallelCluster com pip](#) topic. 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
```

ou

```
$ 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

- [Instalar AWS ParallelCluster com pip](#)
- [Adicione o AWS ParallelCluster executável para seu caminho de linha de comando](#)
- [Instalar o Python no Linux](#)

## Instalar AWS ParallelCluster com **pip**

Use `pip` para instalar AWS ParallelCluster.

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

Quando você usa o `--user` switch, `pip` instala AWS ParallelCluster para `~/.local/bin`.

Verifique isso AWS ParallelCluster 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 para 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 o `--user` switch 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 [Instalar 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
```

ou

```
$ python --version
```

 Note

Se sua distribuição do Linux acompanha Python, poderá ser necessário instalar o pacote de desenvolvedor Python. O pacote do desenvolvedor inclui os cabeçalhos e as bibliotecas necessários para compilar extensões e instalar AWS ParallelCluster. Use seu gerenciador de pacotes para instalar o pacote do 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
```

- Sobre SUSE e derivados, use `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
```

## Instalar AWS ParallelCluster No macOS

Seções

- [Pré-requisitos](#)

- [Instalar AWS ParallelCluster no macOS usando pip](#)
- [Adicione o AWS ParallelCluster executável para 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 [Instalar AWS ParallelCluster no Linux](#).

## Instalar AWS ParallelCluster no macOS usando pip

Você também pode usar pip diretamente para instalar AWS ParallelCluster. Se você não tiver pip, siga as instruções no [tópico principal de instalação](#). 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 seu recém-instalado pip3 para instalar AWS ParallelCluster. Recomendamos que, se você usar o Python versão 3+, use o comando. pip3

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

4. Verifique isso 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 para 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 o aplicativo 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

- [Instalar AWS ParallelCluster usando Python e pip no Windows](#)
- [Adicione o AWS ParallelCluster executável para seu caminho de linha de comando](#)

## Instalar 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 Adicionar Python 3 a. 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. Instalar AWS ParallelCluster usando o `pip`.

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

4. Verifique isso 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 para seu caminho de linha de comando

Depois de instalar AWS ParallelCluster compip, 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 PATH variável (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. Escolha e PATH, em seguida, escolha 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.

# Configuração AWS ParallelCluster

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

Verifique se o seu AWS A conta tem uma função que inclui as permissões necessárias para executar `pcluster` CLI o. Para obter mais informações, consulte [AWS ParallelCluster exemplo de políticas de instância e usuário](#).

Configure seu AWS credenciais. Para obter mais informações, consulte [Configurando o AWS CLI](#) no AWS CLI guia do usuário.

```
$ 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]:
```

A ferramenta Região da AWS onde 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 ao usar AWS Batch como programador em comparação com o uso Slurm. Para obter mais informações sobre a configuração de um cluster, consulte [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 do AWS serviço e AWS Equipes de suporte.

## Slurm

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

**Note**

A lista de Regiões da AWS mostrado é baseado na partição da sua conta e inclui apenas Regiões da AWS que estão habilitados para sua conta. Para obter mais informações sobre como habilitar Regiões da AWS para sua conta, consulte [Gerenciamento Regiões da AWS](#) no Referência geral da AWS. O exemplo mostrado é do AWS Partição global. Se sua conta estiver no AWS GovCloud (US) partição, somente Regiões da AWS nessa partição estão listados (gov-us-east-1 gov-us-west-1). Da mesma forma, se sua conta estiver no AWS Partição da China, somente cn-north-1 e cn-northwest-1 são mostradas. Para obter a lista completa de Regiões da AWS apoiado por AWS ParallelCluster, consulte [Regiões 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 em 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 a partir dos 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 deseja usar uma existente VPC ou deixar AWS ParallelCluster criar um VPC para você. Se você não tiver um configurado corretamente VPC, AWS ParallelCluster pode criar um novo. 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 Região da AWS. Para obter mais informações sobre esse limite e como solicitar um aumento, consulte [VPCe sub-redes](#) no Guia VPC do usuário da Amazon.

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

#### Important

VPCs criados por AWS ParallelCluster não habilitam os registros de VPC fluxo por padrão. VPCs registros de fluxo permitem que você capture informações sobre o tráfego IP que entra e sai das interfaces de rede em seu VPCs. Para obter mais informações, consulte [Logs de VPC fluxo](#) no Guia VPC do usuário da Amazon.

#### Note

Se você escolher `1. Master in a public subnet and compute fleet in a private subnet`, AWS ParallelCluster cria um NAT gateway 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 um novo VPC, deverá selecionar um existente VPC.

Se você optar por ter AWS ParallelCluster crie oVPC, anote o VPC ID para que você possa usar o AWS CLI para excluí-lo posteriormente.

```
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 VPC selecionado, você precisa 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

Da lista de válidos Região da AWS identificadores, escolha o 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, a `linux2` é usado como o sistema operacional.

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

```
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 a partir dos pares de chaves registrados EC2 na Amazon na lista selecionada 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 deseja usar o existente VPCs ou o alugado AWS ParallelCluster criar VPCs para você. Se você não tiver um configurado corretamente VPC, AWS ParallelCluster pode criar um novo. 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 [VPCe sub-redes](#) no Guia VPCdo usuário da Amazon.

**⚠ Important**

VPCs criados por AWS ParallelCluster não habilitam os registros de VPC fluxo por padrão. VPCs com registros de fluxo permitem que você capture informações sobre o tráfego IP que entra e sai das interfaces de rede em seus VPCs. Para obter mais informações, consulte [Logs de VPC fluxo](#) no Guia VPC do usuário da Amazon.

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

**ℹ Note**

Se você escolher `1. Master in a public subnet and compute fleet in a private subnet`, AWS ParallelCluster cria um NAT gateway 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 um novo VPC, deverá selecionar um existente VPC.

Se você optar por ter AWS ParallelCluster criar o VPC, anote o VPC ID para que você possa usar o AWS CLI para excluí-lo posteriormente.

```
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 VPC selecionado, 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 um VPC. O 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:

- O VPC indispensável DNS Resolution = yes DNS Hostnames = yes e.
- O também VPC deve ter DHCP opções com o correto domain-name para o Região da AWS. O conjunto de DHCP opções padrão já especifica o necessário AmazonProvidedDNS. Se especificar mais de um servidor de nomes de domínio, consulte os [conjuntos DHCP de opções](#) no Guia do VPC usuário da Amazon. Ao usar sub-redes privadas, use um NAT gateway 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
```

Depois que o cluster atingir o status COMPLETE "CREATE\_", você poderá se conectar a ele usando as configurações normais SSH do cliente. 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 no VPC, você pode excluir a pilha CloudFormation de rede. O nome da pilha começa com "parallelclusternetworking-" e contém a hora de 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
```

O VPC que [pcluster configure](#) cria para você não é criado na pilha CloudFormation de rede. Você pode excluí-lo VPC manualmente no console ou usando o 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 mestre é o nó que expõe o servidor. NFS Por esse motivo, você deseja escolher um tipo de instância com largura de banda de rede suficiente e largura de banda dedicada suficiente da 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 crie um grupo de colocação com a "compute" estratégia `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.
- Adaptador Elastic Fabric: para oferecer suporte a altos níveis de comunicação escalável entre instâncias, considere escolher interfaces de EFA rede para sua rede. O hardware personalizado EFA de bypass do sistema operacional (SO) aprimora as comunicações entre instâncias com a elasticidade e a flexibilidade sob demanda do AWS nuvem. Para configurar um único Slurm fila de cluster a ser usada EFA, defina `enable_efa = true`. Para obter mais informações sobre como usar EFA com AWS ParallelCluster, veja [Elastic Fabric Adapter enable\\_efa](#). Para obter mais informações sobre isso EFA, consulte [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 [EBS otimizadas da Amazon e tipos de EBS volume da Amazon no Guia](#) do usuário da Amazon EC2.

## Práticas recomendadas: alertas de orçamento

Para gerenciar AWS ParallelCluster custos de recursos, recomendamos que você use AWS Budgets ações para criar um orçamento e alertas de limite de orçamento definido para os selecionados AWS recursos. Para obter mais informações, consulte [Configurando uma ação orçamentária](#) no AWS Budgets Guia do usuário. 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 sua estimativa AWS cobranças](#).

## Melhores práticas: migrar um cluster para um novo AWS ParallelCluster versão secundária ou patch

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

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 para 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 AWS Management Console:
  - [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 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. EFS Para obter mais informações, consulte [Montagem de sistemas de EFS arquivos da Amazon](#) no Guia EFS do usuário da Amazon 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 AMI personalizá-las. Isso otimiza o processo de criação porque AMI não é necessário criar um novo personalizado 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 você use AWS ParallelCluster em vez disso, 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 descritos nas seções a seguir.

AWS ParallelCluster CLI gerencia um conjunto diferente de clusters

Os clusters criados com o não `cfncluster` CLI podem ser gerenciados com `pcluster` CLI o. 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 `cfncluster` CLI o.

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 políticas IAM personalizadas diferentes

IAMPolíticas personalizadas que foram usadas anteriormente para criação de CfnCluster clusters não podem ser usadas com AWS ParallelCluster. Se você precisar de políticas personalizadas para AWS ParallelCluster, você deve criar novos. Veja o AWS ParallelCluster guia.

AWS ParallelCluster e CfnCluster use arquivos de configuração diferentes

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

Se você quiser usar um arquivo de CfnCluster configuração existente com AWS ParallelCluster, então você deve 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, o gânglio está desativado por padrão

Em AWS ParallelCluster, o gânglio está desativado 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 compatíveis

AWS ParallelCluster a versão 2.x está disponível no seguinte Regiões da AWS:

Nome da região	Região
Leste dos EUA (Ohio)	us-east-2
Leste dos EUA (N. da Virgínia)	us-east-1
Oeste dos EUA (N. 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](#)
- [Ações de bootstrap personalizadas](#)
- [Trabalhar com o Amazon S3](#)
- [Trabalho com Instâncias spot](#)
- [AWS Identity and Access Management funções em AWS ParallelCluster](#)
- [Agendadores suportados por AWS ParallelCluster](#)
- [AWS ParallelCluster recursos e marcação](#)
- [CloudWatch Painel da Amazon](#)
- [Integração com Amazon CloudWatch Logs](#)
- [Elastic Fabric Adapter](#)
- [Intel Select Solutions](#)
- [Ativar Intel MPI](#)
- [Especificação HPC da plataforma Intel](#)
- [Bibliotecas de desempenho do Arm](#)
- [Conecte-se ao nó principal por meio da Amazon DCV](#)
- [Utilizar o pcluster update](#)
- [AMl aplicação de patches e substituição de EC2 instâncias](#)

## Configurações de rede

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

O VPC obrigatório DNS Hostnames = yes e DNS Resolution = yes as DHCP opções com o nome de domínio correto para a região. O conjunto de DHCP opções padrão já especifica o necessário AmazonProvidedDNS. Se especificar mais de um servidor de nomes de domínio, consulte os [conjuntos DHCP de opções](#) no Guia do VPC usuário da Amazon.

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 HTTP proxy 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 HTTP proxy para todo o tráfego.

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

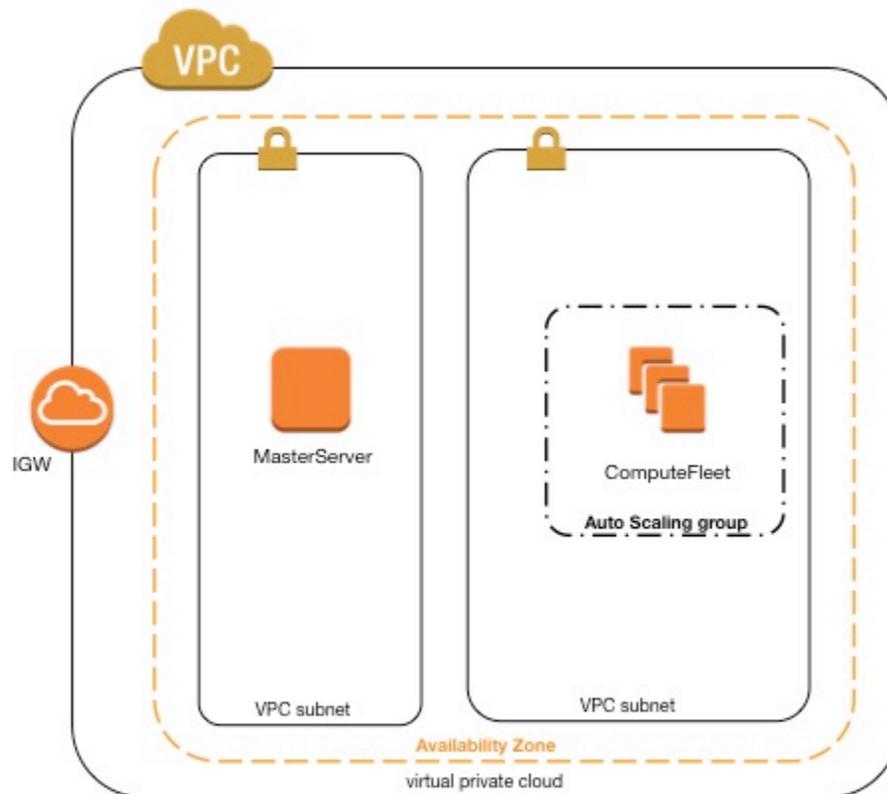
## AWS ParallelCluster em uma única sub-rede pública

A configuração para essa arquitetura requer as seguintes definições:

```
[vpc public]
vpc_id = vpc-xxxxxxx
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 [Habilitando o acesso à Internet](#) no Guia VPC do Usuário da Amazon.

## 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
```

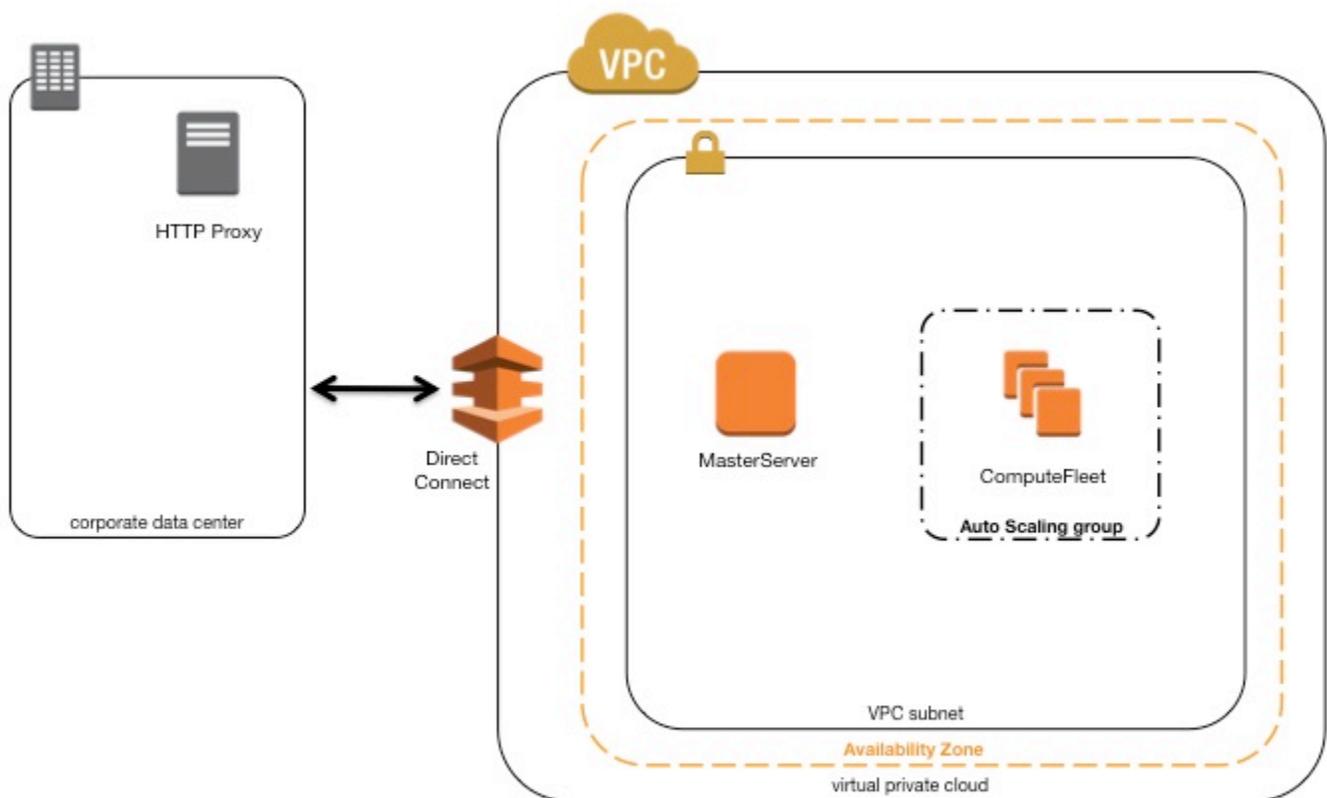
A configuração para usar uma rede privada existente requer as seguintes definições:

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

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

Ambas as configurações exigem um [NATgateway](#) ou um proxy interno para permitir 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` definido como `false`, o VPC deve estar configurado corretamente 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 computacional 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 ECS contêiner da Amazon precisam de acesso externo à rede para se comunicar com o endpoint ECS de serviço da Amazon. Isso se converte nas seguintes situações:

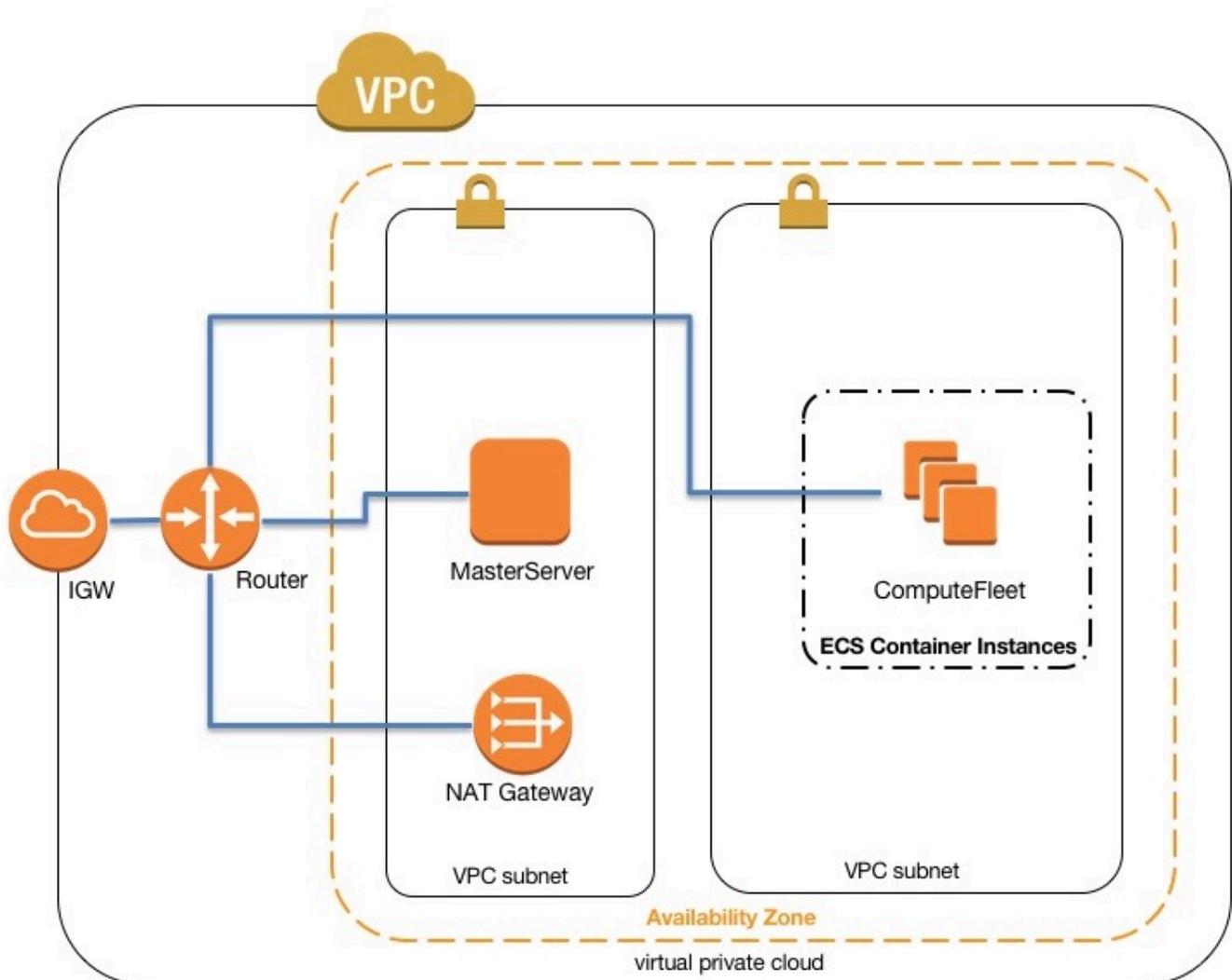
- `compute_subnet` Ele usa um NAT gateway para acessar a internet. (Recomendamos essa abordagem.)
- As instâncias executadas na `compute_subnet` têm endereços IP públicos e podem acessar a Internet por meio de um gateway da Internet.

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

Ao usar o Amazon ECS Task Networking, o modo de `aws_vpc` rede não fornece interfaces de rede elásticas com endereços IP públicos para tarefas que usam o tipo de EC2 lançamento da Amazon. Para acessar a Internet, as tarefas que usam o tipo de EC2 execução da Amazon devem ser iniciadas em uma sub-rede privada configurada para usar um NAT gateway.

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



Para obter mais informações, consulte os tópicos a seguir.

- [AWS Batch ambientes computacionais gerenciados](#)
- [AWS Batch trabalhos paralelos de vários nós](#)
- [Rede de ECS tarefas da Amazon com o modo awsvpc de rede](#)

## 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 do início de qualquer ação de bootstrap de implantação do cluster, como a configuração do NAT Amazon Elastic Block Store (AmazonEBS) ou do agendador. 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 servem como 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 o script seja carregado no 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 protocolo. 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, consulte [AWS 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 um Amazon S3 válido \* ou um Amazon S3ARN. Para obter informações sobre a especificação do Amazon ARNs S3, consulte o formato do [Amazon ARN 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 a Amazon EC2 interrompa ou encerre sua Instância Spot. Para obter mais informações, consulte [Interrupções de instâncias spot](#) no Guia EC2 do usuário da Amazon. 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 obter mais informações, consulte [Função vinculada ao serviço para solicitações de instâncias spot](#) no Guia EC2 do usuário da Amazon.

## 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 `sbatch`, incluindo o `--no-requeue` parâmetro, 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 SGE ou Torque agendadores.

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 dos seguintes parâmetros SGE de configuração:

- `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 SGE ou Torque agendadores.

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 SGE ou Torque agendadores.

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 SGE ou Torque agendadores.

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 obter mais informações sobre instâncias spot, consulte [Instâncias spot](#) no Guia EC2 do usuário da Amazon.

# AWS Identity and Access Management funções em AWS ParallelCluster

AWS ParallelCluster usa funções AWS Identity and Access Management (IAM) para EC2 a Amazon para permitir que instâncias acessem AWS serviços para a implantação e operação de um cluster. Por padrão, a IAM função da Amazon EC2 é criada 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](#)
- [Usando uma IAM função existente para a Amazon EC2](#)
- [AWS ParallelCluster exemplo de políticas de instância e usuário](#)

## Configurações padrão para criação de clusters

Quando você usa as configurações padrão para a criação do cluster, uma IAM função padrão para a Amazon EC2 é criada 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 a criação de uma IAM função para a AmazonEC2. Normalmente, o usuário deve ter as permissões de uma política AdministratorAccessgerenciada ao usar as configurações padrão. Para obter informações sobre políticas gerenciadas, consulte [políticas AWS gerenciadas](#) no Guia IAM do usuário.

## Usando uma IAM função existente para a Amazon EC2

No lugar das configurações padrão, você pode usar uma existente [ec2\\_iam\\_role](#) para criar um cluster, mas deve definir a IAM política e a função antes de tentar iniciar o cluster. Normalmente, você escolhe uma IAM função existente para EC2 a Amazon para minimizar as permissões concedidas aos usuários quando eles iniciam clusters. Eles [AWS ParallelCluster exemplo de políticas de instância e usuário](#) incluem as permissões mínimas exigidas AWS ParallelCluster e

seus recursos. Você deve criar políticas e funções como políticas individuais IAM e, em seguida, anexar as funções e 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 IAM de tamanho da política](#). Nas políticas, substitua `<REGION>`, `<AWS ACCOUNT ID>` e cadeias de caracteres semelhantes com os valores apropriados.

Se sua intenção for adicionar políticas extras às configurações padrão dos nós do cluster, recomendamos que você passe as IAM políticas personalizadas adicionais com a [additional\\_iam\\_policies](#) configuração em vez de usar as [ec2\\_iam\\_role](#) configurações.

## 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. 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 IAM de tamanho da política](#).

### 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](#)
- [ParallelClusterInstancePolicy usando SGE ou Torque](#)

### ParallelClusterInstancePolicy usando Slurm

O exemplo a seguir define o ParallelClusterInstancePolicy uso Slurm como programador.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:AttachVolume",
        "ec2:DescribeInstanceAttribute",
        "ec2:DescribeInstanceStatus",
        "ec2:DescribeInstanceTypes",
        "ec2:DescribeInstances",
        "ec2:DescribeRegions",
        "ec2:TerminateInstances",
        "ec2:DescribeLaunchTemplates",
        "ec2:CreateTags"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow",
      "Sid": "EC2"
    },
    {
```

```

    "Action": "ec2:RunInstances",
    "Resource": [
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:subnet/<COMPUTE SUBNET ID>",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:network-interface/*",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:instance/*",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:volume/*",
      "arn:aws:ec2:<REGION>::image/<IMAGE ID>",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:key-pair/<KEY NAME>",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:security-group/*",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:launch-template/*",
      "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:placement-group*"
    ],
    "Effect": "Allow",
    "Sid": "EC2RunInstances"
  },
  {
    "Action": [
      "dynamodb:ListTables"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow",
    "Sid": "DynamoDBList"
  },
  {
    "Action": [
      "cloudformation:DescribeStacks",
      "cloudformation:DescribeStackResource",
      "cloudformation:SignalResource"
    ],
    "Resource": [
      "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/
parallelcluster-*/*"
    ],
    "Effect": "Allow",
    "Sid": "CloudFormation"
  },
  {
    "Action": [
      "dynamodb:PutItem",
      "dynamodb:Query",
      "dynamodb:GetItem",
      "dynamodb:BatchWriteItem",

```

```

        "dynamodb:DeleteItem",
        "dynamodb:DescribeTable"
    ],
    "Resource": [
        "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*"
    ],
    "Effect": "Allow",
    "Sid": "DynamoDBTable"
},
{
    "Action": [
        "s3:GetObject"
    ],
    "Resource": [
        "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
    ],
    "Effect": "Allow",
    "Sid": "S3GetObject"
},
{
    "Action": [
        "iam:PassRole"
    ],
    "Resource": [
        "*"
    ],
    "Effect": "Allow",
    "Sid": "IAMPassRole",
    "Condition": {
        "StringEquals": {
            "iam:PassedToService": [
                "ec2.amazonaws.com"
            ]
        }
    }
},
{
    "Action": [
        "s3:GetObject"
    ],
    "Resource": [
        "arn:aws:s3:::dcv-license.<REGION>/*"
    ],
    "Effect": "Allow",

```

```
    "Sid": "DcvLicense"
  },
  {
    "Action": [
      "s3:GetObject",
      "s3:GetObjectVersion"
    ],
    "Resource": [
      "arn:aws:s3:::parallelcluster-*/*"
    ],
    "Effect": "Allow",
    "Sid": "GetClusterConfig"
  },
  {
    "Action": [
      "fsx:DescribeFileSystems"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow",
    "Sid": "FSx"
  },
  {
    "Action": [
      "logs:CreateLogStream",
      "logs:PutLogEvents"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow",
    "Sid": "CWLogs"
  },
  {
    "Action": [
      "route53:ChangeResourceRecordSets"
    ],
    "Resource": [
      "arn:aws:route53:::hostedzone/*"
    ],
    "Effect": "Allow",
    "Sid": "Route53"
  }
}
```

```
]
}
```

## ParallelClusterInstancePolicy usando SGE ou Torque

O exemplo a seguir define o ParallelClusterInstancePolicy uso 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 SGE ou Torque agendadores.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "ec2:DescribeVolumes",
        "ec2:AttachVolume",
        "ec2:DescribeInstanceAttribute",
        "ec2:DescribeInstanceStatus",
        "ec2:DescribeInstanceTypes",
        "ec2:DescribeInstances",
        "ec2:DescribeRegions",
        "ec2:TerminateInstances",
        "ec2:DescribeLaunchTemplates",
        "ec2:CreateTags"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow",
      "Sid": "EC2"
    },
    {
      "Action": "ec2:RunInstances",
      "Resource": [
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:subnet/<COMPUTE SUBNET ID>",
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:network-interface/*",

```

```

        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:instance/*",
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:volume/*",
        "arn:aws:ec2:<REGION>::image/<IMAGE ID>",
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:key-pair/<KEY NAME>",
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:security-group/*",
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:launch-template/*",
        "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:placement-group/*"
    ],
    "Effect": "Allow",
    "Sid": "EC2RunInstances"
},
{
    "Action": [
        "dynamodb:ListTables"
    ],
    "Resource": [
        "*"
    ],
    "Effect": "Allow",
    "Sid": "DynamoDBList"
},
{
    "Action": [
        "sqs:SendMessage",
        "sqs:ReceiveMessage",
        "sqs:ChangeMessageVisibility",
        "sqs>DeleteMessage",
        "sqs:GetQueueUrl"
    ],
    "Resource": [
        "arn:aws:sqs:<REGION>:<AWS ACCOUNT ID>:parallelcluster-*"
    ],
    "Effect": "Allow",
    "Sid": "SQSQueue"
},
{
    "Action": [
        "autoscaling:DescribeAutoScalingGroups",
        "autoscaling:TerminateInstanceInAutoScalingGroup",
        "autoscaling:SetDesiredCapacity",
        "autoscaling:UpdateAutoScalingGroup",
        "autoscaling:DescribeTags",
        "autoscaling:SetInstanceHealth"
    ],

```

```

    "Resource": [
        "*"
    ],
    "Effect": "Allow",
    "Sid": "Autoscaling"
},
{
    "Action": [
        "cloudformation:DescribeStacks",
        "cloudformation:DescribeStackResource",
        "cloudformation:SignalResource"
    ],
    "Resource": [
        "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/
parallelcluster-*/*"
    ],
    "Effect": "Allow",
    "Sid": "CloudFormation"
},
{
    "Action": [
        "dynamodb:PutItem",
        "dynamodb:Query",
        "dynamodb:GetItem",
        "dynamodb:BatchWriteItem",
        "dynamodb>DeleteItem",
        "dynamodb:DescribeTable"
    ],
    "Resource": [
        "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*"
    ],
    "Effect": "Allow",
    "Sid": "DynamoDBTable"
},
{
    "Action": [
        "s3:GetObject"
    ],
    "Resource": [
        "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
    ],
    "Effect": "Allow",
    "Sid": "S3GetObject"
},

```

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

```
    ],
    "Effect": "Allow",
    "Sid": "GetClusterConfig"
  },
  {
    "Action": [
      "fsx:DescribeFileSystems"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow",
    "Sid": "FSx"
  },
  {
    "Action": [
      "logs:CreateLogStream",
      "logs:PutLogEvents"
    ],
    "Resource": [
      "*"
    ],
    "Effect": "Allow",
    "Sid": "CWLogs"
  },
  {
    "Action": [
      "route53:ChangeResourceRecordSets"
    ],
    "Resource": [
      "arn:aws:route53:::hostedzone/*"
    ],
    "Effect": "Allow",
    "Sid": "Route53"
  }
]
}
```

## 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 AWS CloudFormation aninhada. `BatchUserRole` O `BatchUserRole` ARN é

fornevido 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, então "*<RESOURCES S3 BUCKET>*" é "parallelcluster-\*". O exemplo a seguir é uma visão geral das permissões necessárias:

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "batch:RegisterJobDefinition",
        "logs:GetLogEvents"
      ],
      "Resource": [
        "*"
      ],
      "Effect": "Allow"
    },
    {
      "Action": [
        "batch:SubmitJob",
        "cloudformation:DescribeStacks",
        "ecs:ListContainerInstances",
        "ecs:DescribeContainerInstances",
        "logs:FilterLogEvents",
        "s3:PutObject",
        "s3:Get*",
        "s3>DeleteObject",
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-
definition/<AWS_BATCH_STACK - JOB_DEFINITION_SERIAL_NAME>:1",
        "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-
definition/<AWS_BATCH_STACK - JOB_DEFINITION_MNP_NAME>*",
        "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-queue/<AWS_BATCH_STACK -
JOB_QUEUE_NAME>",
        "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/<STACK NAME>/
*",
        "arn:aws:s3:::<RESOURCES S3 BUCKET>/batch/*",
        "arn:aws:iam::<AWS ACCOUNT ID>:role/<AWS_BATCH_STACK - JOB_ROLE>",
        "arn:aws:ecs:<REGION>:<AWS ACCOUNT ID>:cluster/<ECS COMPUTE
ENVIRONMENT>",

```

```

        "arn:aws:ecs:<REGION>:<AWS ACCOUNT ID>:container-instance/*",
        "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:log-group:/aws/batch/job:log-
stream:*"
    ],
    "Effect": "Allow"
},
{
    "Action": [
        "s3:List*"
    ],
    "Resource": [
        "arn:aws:s3:::<RESOURCES S3 BUCKET>"
    ],
    "Effect": "Allow"
},
{
    "Action": [
        "batch:DescribeJobQueues",
        "batch:TerminateJob",
        "batch:DescribeJobs",
        "batch:CancelJob",
        "batch:DescribeJobDefinitions",
        "batch:ListJobs",
        "batch:DescribeComputeEnvironments"
    ],
    "Resource": "*",
    "Effect": "Allow"
},
{
    "Action": [
        "ec2:DescribeInstances",
        "ec2:AttachVolume",
        "ec2:DescribeVolumes",
        "ec2:DescribeInstanceAttribute"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EC2"
},
{
    "Action": [
        "cloudformation:DescribeStackResource",
        "cloudformation:SignalResource"
    ],

```

```

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

```

## ParallelClusterUserPolicy usando Slurm

O exemplo a seguir define o `ParallelClusterUserPolicy`, usando Slurm como programador. Neste exemplo, "`<RESOURCES S3 BUCKET>`" é o valor da [cluster\\_resource\\_bucket](#) configuração; se não `cluster_resource_bucket` for especificado, então "`<RESOURCES S3 BUCKET>`" é "parallelcluster-\*".

### Note

Se você usar um papel personalizado `ec2_iam_role = <role_name>`, deverá alterar o IAM recurso para incluir o nome desse papel de:

```
"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"
```

Para:

```
"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/<role_name>"
```

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

```

```
        "ec2:CreateRouteTable",
        "ec2:AssociateRouteTable",
        "ec2:CreateSubnet",
        "ec2:ModifySubnetAttribute"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "NetworkingEasyConfig"
},
{
    "Action": [
        "ec2:CreateVolume",
        "ec2:RunInstances",
        "ec2:AllocateAddress",
        "ec2:AssociateAddress",
        "ec2:AttachNetworkInterface",
        "ec2:AuthorizeSecurityGroupEgress",
        "ec2:AuthorizeSecurityGroupIngress",
        "ec2:CreateNetworkInterface",
        "ec2:CreateSecurityGroup",
        "ec2:ModifyVolumeAttribute",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2>DeleteNetworkInterface",
        "ec2>DeleteVolume",
        "ec2:TerminateInstances",
        "ec2>DeleteSecurityGroup",
        "ec2:DisassociateAddress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:ReleaseAddress",
        "ec2:CreatePlacementGroup",
        "ec2>DeletePlacementGroup"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EC2Modify"
},
{
    "Action": [
        "autoscaling:CreateAutoScalingGroup",
        "ec2:CreateLaunchTemplate",
        "ec2:CreateLaunchTemplateVersion",
        "ec2:ModifyLaunchTemplate",
        "ec2>DeleteLaunchTemplate",
```

```

        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "ScalingModify"
},
{
    "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:ListTagsOfResource"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "DynamoDBDescribe"
},
{
    "Action": [
        "dynamodb:CreateTable",
        "dynamodb>DeleteTable",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Query",
        "dynamodb:TagResource"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "DynamoDBModify"
},
{
    "Action": [
        "route53:ChangeResourceRecordSets",
        "route53:ChangeTagsForResource",
        "route53:CreateHostedZone",
        "route53>DeleteHostedZone",
        "route53:GetChange",
        "route53:GetHostedZone",
        "route53:ListResourceRecordSets",
        "route53:ListQueryLoggingConfigs"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "Route53HostedZones"
},

```

```

{
  "Action": [
    "cloudformation:DescribeStackEvents",
    "cloudformation:DescribeStackResource",
    "cloudformation:DescribeStackResources",
    "cloudformation:DescribeStacks",
    "cloudformation:ListStacks",
    "cloudformation:GetTemplate"
  ],
  "Resource": "*",
  "Effect": "Allow",
  "Sid": "CloudFormationDescribe"
},
{
  "Action": [
    "cloudformation:CreateStack",
    "cloudformation>DeleteStack",
    "cloudformation:UpdateStack"
  ],
  "Effect": "Allow",
  "Resource": "*",
  "Sid": "CloudFormationModify"
},
{
  "Action": [
    "s3:*"
  ],
  "Resource": [
    "arn:aws:s3:::<RESOURCES S3 BUCKET>"
  ],
  "Effect": "Allow",
  "Sid": "S3ResourcesBucket"
},
{
  "Action": [
    "s3:Get*",
    "s3:List*"
  ],
  "Resource": [
    "arn:aws:s3:::<REGION>-aws-parallelcluster*"
  ],
  "Effect": "Allow",
  "Sid": "S3ParallelClusterReadOnly"
},

```

```

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

```

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

```

        "fsx:*"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "FSx"
},
{
    "Action": [
        "elasticfilesystem:*"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EFS"
},
{
    "Action": [
        "logs:DeleteLogGroup",
        "logs:PutRetentionPolicy",
        "logs:DescribeLogGroups",
        "logs:CreateLogGroup",
        "logs:TagResource",
        "logs:UntagResource"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "CloudWatchLogs"
},
{
    "Action": [
        "lambda:CreateFunction",
        "lambda>DeleteFunction",
        "lambda:GetFunctionConfiguration",
        "lambda:GetFunction",
        "lambda:InvokeFunction",
        "lambda:AddPermission",
        "lambda:RemovePermission",
        "lambda:TagResource",
        "lambda:ListTags",
        "lambda:UntagResource"
    ],
    "Resource": [
        "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:parallelcluster-*",
        "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:pcluster-*"
    ]
},

```

```

        "Effect": "Allow",
        "Sid": "Lambda"
    },
    {
        "Sid": "CloudWatch",
        "Effect": "Allow",
        "Action": [
            "cloudwatch:PutDashboard",
            "cloudwatch:ListDashboards",
            "cloudwatch>DeleteDashboards",
            "cloudwatch:GetDashboard"
        ],
        "Resource": "*"
    }
]
}

```

## ParallelClusterUserPolicy 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 SGE ou Torque agendadores.

O exemplo a seguir define o `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, então "`<RESOURCES S3 BUCKET>`" é "parallelcluster-\*".

### Note

Se você usar um papel personalizado `ec2_iam_role = <role_name>`, deverá alterar o IAM recurso para incluir o nome desse papel de:

```
"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"
```

Para:

```
"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/<role_name>"
```

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

```

    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "NetworkingEasyConfig"
  },
  {
    "Action": [
      "ec2:CreateVolume",
      "ec2:RunInstances",
      "ec2:AllocateAddress",
      "ec2:AssociateAddress",
      "ec2:AttachNetworkInterface",
      "ec2:AuthorizeSecurityGroupEgress",
      "ec2:AuthorizeSecurityGroupIngress",
      "ec2:CreateNetworkInterface",
      "ec2:CreateSecurityGroup",
      "ec2:ModifyVolumeAttribute",
      "ec2:ModifyNetworkInterfaceAttribute",
      "ec2>DeleteNetworkInterface",
      "ec2>DeleteVolume",
      "ec2:TerminateInstances",
      "ec2>DeleteSecurityGroup",
      "ec2:DisassociateAddress",
      "ec2:RevokeSecurityGroupIngress",
      "ec2:RevokeSecurityGroupEgress",
      "ec2:ReleaseAddress",
      "ec2:CreatePlacementGroup",
      "ec2>DeletePlacementGroup"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EC2Modify"
  },
  {
    "Action": [
      "autoscaling:DescribeAutoScalingGroups",
      "autoscaling:DescribeAutoScalingInstances"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "AutoScalingDescribe"
  },
  {
    "Action": [

```

```

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

```

```
        "sqs:GetQueueAttributes"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "SQSDescribe"
},
{
    "Action": [
        "sqs:CreateQueue",
        "sqs:SetQueueAttributes",
        "sqs>DeleteQueue",
        "sqs:TagQueue"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "SQSModify"
},
{
    "Action": [
        "sns:ListTopics",
        "sns:GetTopicAttributes"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "SNSDescribe"
},
{
    "Action": [
        "sns:CreateTopic",
        "sns:Subscribe",
        "sns:Unsubscribe",
        "sns>DeleteTopic"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "SNSModify"
},
{
    "Action": [
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStackResource",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStacks",
```

```
        "cloudformation:GetTemplate"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "CloudFormationDescribe"
},
{
    "Action": [
        "cloudformation:CreateStack",
        "cloudformation>DeleteStack",
        "cloudformation:UpdateStack"
    ],
    "Effect": "Allow",
    "Resource": "*",
    "Sid": "CloudFormationModify"
},
{
    "Action": [
        "s3:*"
    ],
    "Resource": [
        "arn:aws:s3:::<RESOURCES S3 BUCKET>"
    ],
    "Effect": "Allow",
    "Sid": "S3ResourcesBucket"
},
{
    "Action": [
        "s3:Get*",
        "s3:List*"
    ],
    "Resource": [
        "arn:aws:s3:::<REGION>-aws-parallelcluster*"
    ],
    "Effect": "Allow",
    "Sid": "S3ParallelClusterReadOnly"
},
{
    "Action": [
        "s3:DeleteBucket",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion"
    ],
    "Resource": [
```

```

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

```

```

    "Sid": "IAMCreateInstanceProfile"
  },
  {
    "Action": [
      "iam:AddRoleToInstanceProfile",
      "iam:RemoveRoleFromInstanceProfile",
      "iam:GetRolePolicy",
      "iam:GetPolicy",
      "iam:AttachRolePolicy",
      "iam:DetachRolePolicy",
      "iam:PutRolePolicy",
      "iam>DeleteRolePolicy"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "IAMInstanceProfile"
  },
  {
    "Action": [
      "elasticfilesystem:DescribeMountTargets",
      "elasticfilesystem:DescribeMountTargetSecurityGroups",
      "ec2:DescribeNetworkInterfaceAttribute"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EFSDescribe"
  },
  {
    "Action": [
      "ssm:GetParametersByPath"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "SSMDescribe"
  },
  {
    "Action": [
      "fsx:*"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "FSx"
  },
  {

```

```

    "Action": [
      "elasticfilesystem:*"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EFS"
  },
  {
    "Action": [
      "logs:DeleteLogGroup",
      "logs:PutRetentionPolicy",
      "logs:DescribeLogGroups",
      "logs:CreateLogGroup",
      "logs:TagResource",
      "logs:UntagResource"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "CloudWatchLogs"
  },
  {
    "Action": [
      "lambda:CreateFunction",
      "lambda:DeleteFunction",
      "lambda:GetFunctionConfiguration",
      "lambda:GetFunction",
      "lambda:InvokeFunction",
      "lambda:AddPermission",
      "lambda:RemovePermission",
      "lambda:TagResource",
      "lambda:ListTags",
      "lambda:UntagResource"
    ],
    "Resource": [
      "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:parallelcluster-*",
      "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:pcluster-*"
    ],
    "Effect": "Allow",
    "Sid": "Lambda"
  },
  {
    "Sid": "CloudWatch",
    "Effect": "Allow",
    "Action": [

```

```

        "cloudwatch:PutDashboard",
        "cloudwatch:ListDashboards",
        "cloudwatch>DeleteDashboards",
        "cloudwatch:GetDashboard"
    ],
    "Resource": "*"
}
]
}

```

## 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, então "**<RESOURCES S3 BUCKET>**" é "parallelcluster-\*".

```

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

```

```
    "Sid": "EC2Describe"
  },
  {
    "Action": [
      "ec2:CreateLaunchTemplate",
      "ec2:CreateLaunchTemplateVersion",
      "ec2:ModifyLaunchTemplate",
      "ec2>DeleteLaunchTemplate",
      "ec2:DescribeLaunchTemplates",
      "ec2:DescribeLaunchTemplateVersions"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EC2LaunchTemplate"
  },
  {
    "Action": [
      "ec2:CreateVpc",
      "ec2:ModifyVpcAttribute",
      "ec2:DescribeNatGateways",
      "ec2:CreateNatGateway",
      "ec2:DescribeInternetGateways",
      "ec2:CreateInternetGateway",
      "ec2:AttachInternetGateway",
      "ec2:DescribeRouteTables",
      "ec2:CreateRoute",
      "ec2:CreateRouteTable",
      "ec2:AssociateRouteTable",
      "ec2:CreateSubnet",
      "ec2:ModifySubnetAttribute"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "NetworkingEasyConfig"
  },
  {
    "Action": [
      "ec2:CreateVolume",
      "ec2:RunInstances",
      "ec2:AllocateAddress",
      "ec2:AssociateAddress",
      "ec2:AttachNetworkInterface",
      "ec2:AuthorizeSecurityGroupEgress",
      "ec2:AuthorizeSecurityGroupIngress",
```

```

        "ec2:CreateNetworkInterface",
        "ec2:CreateSecurityGroup",
        "ec2:ModifyVolumeAttribute",
        "ec2:ModifyNetworkInterfaceAttribute",
        "ec2>DeleteNetworkInterface",
        "ec2>DeleteVolume",
        "ec2:TerminateInstances",
        "ec2>DeleteSecurityGroup",
        "ec2:DisassociateAddress",
        "ec2:RevokeSecurityGroupIngress",
        "ec2:RevokeSecurityGroupEgress",
        "ec2:ReleaseAddress",
        "ec2:CreatePlacementGroup",
        "ec2>DeletePlacementGroup"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "EC2Modify"
},
{
    "Action": [
        "dynamodb:DescribeTable",
        "dynamodb:CreateTable",
        "dynamodb>DeleteTable",
        "dynamodb:GetItem",
        "dynamodb:PutItem",
        "dynamodb:Query",
        "dynamodb:TagResource"
    ],
    "Resource": "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/
parallelcluster-*",
    "Effect": "Allow",
    "Sid": "DynamoDB"
},
{
    "Action": [
        "cloudformation:DescribeStackEvents",
        "cloudformation:DescribeStackResource",
        "cloudformation:DescribeStackResources",
        "cloudformation:DescribeStacks",
        "cloudformation:ListStacks",
        "cloudformation:GetTemplate",
        "cloudformation:CreateStack",
        "cloudformation>DeleteStack",

```

```

        "cloudformation:UpdateStack"
    ],
    "Resource": "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/
parallelcluster-*",
    "Effect": "Allow",
    "Sid": "CloudFormation"
},
{
    "Action": [
        "route53:ChangeResourceRecordSets",
        "route53:ChangeTagsForResource",
        "route53:CreateHostedZone",
        "route53>DeleteHostedZone",
        "route53:GetChange",
        "route53:GetHostedZone",
        "route53:ListResourceRecordSets"
    ],
    "Resource": "arn:aws:route53::hostedzone/*",
    "Effect": "Allow",
    "Sid": "Route53HostedZones"
},
{
    "Action": [
        "sqs:GetQueueAttributes",
        "sqs:CreateQueue",
        "sqs:SetQueueAttributes",
        "sqs>DeleteQueue",
        "sqs:TagQueue"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "SQS"
},
{
    "Action": [
        "sqs:SendMessage",
        "sqs:ReceiveMessage",
        "sqs:ChangeMessageVisibility",
        "sqs>DeleteMessage",
        "sqs:GetQueueUrl"
    ],
    "Resource": "arn:aws:sqs:<REGION>:<AWS ACCOUNT ID>:parallelcluster-*",
    "Effect": "Allow",
    "Sid": "SQSQueue"
}

```

```

    },
    {
      "Action": [
        "sns:ListTopics",
        "sns:GetTopicAttributes",
        "sns:CreateTopic",
        "sns:Subscribe",
        "sns:Unsubscribe",
        "sns>DeleteTopic"
      ],
      "Resource": "*",
      "Effect": "Allow",
      "Sid": "SNS"
    },
    {
      "Action": [
        "iam:PassRole",
        "iam:CreateRole",
        "iam>DeleteRole",
        "iam:GetRole",
        "iam:TagRole",
        "iam:SimulatePrincipalPolicy"
      ],
      "Resource": [
        "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*",
        "arn:aws:iam::<AWS ACCOUNT ID>:role/<PARALLELCLUSTER EC2 ROLE NAME>"
      ],
      "Effect": "Allow",
      "Sid": "IAMRole"
    },
    {
      "Action": [
        "iam>CreateInstanceProfile",
        "iam>DeleteInstanceProfile",
        "iam:GetInstanceProfile",
        "iam:PassRole"
      ],
      "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/*",
      "Effect": "Allow",
      "Sid": "IAMInstanceProfile"
    },
    {
      "Action": [
        "iam:AddRoleToInstanceProfile",

```

```

        "iam:RemoveRoleFromInstanceProfile",
        "iam:GetRolePolicy",
        "iam:PutRolePolicy",
        "iam:DeleteRolePolicy",
        "iam:GetPolicy",
        "iam:AttachRolePolicy",
        "iam:DetachRolePolicy"
    ],
    "Resource": "*",
    "Effect": "Allow",
    "Sid": "IAM"
},
{
    "Action": [
        "s3:*"
    ],
    "Resource": [
        "arn:aws:s3:::<RESOURCES S3 BUCKET>"
    ],
    "Effect": "Allow",
    "Sid": "S3ResourcesBucket"
},
{
    "Action": [
        "s3:Get*",
        "s3:List*"
    ],
    "Resource": [
        "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
    ],
    "Effect": "Allow",
    "Sid": "S3ParallelClusterReadOnly"
},
{
    "Action": [
        "s3:DeleteBucket",
        "s3:DeleteObject",
        "s3:DeleteObjectVersion"
    ],
    "Resource": [
        "arn:aws:s3:::<RESOURCES S3 BUCKET>"
    ],
    "Effect": "Allow",
    "Sid": "S3Delete"
}

```

```

    },
    {
      "Action": [
        "lambda:CreateFunction",
        "lambda>DeleteFunction",
        "lambda:GetFunction",
        "lambda:GetFunctionConfiguration",
        "lambda:InvokeFunction",
        "lambda:AddPermission",
        "lambda:RemovePermission",
        "lambda:TagResource",
        "lambda:ListTags",
        "lambda:UntagResource"
      ],
      "Resource": [
        "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:parallelcluster-*",
        "arn:aws:lambda:<REGION>:<AWS ACCOUNT ID>:function:pcluster-*"
      ],
      "Effect": "Allow",
      "Sid": "Lambda"
    },
    {
      "Action": [
        "logs:*"
      ],
      "Resource": "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:*",
      "Effect": "Allow",
      "Sid": "Logs"
    },
    {
      "Action": [
        "codebuild:*"
      ],
      "Resource": "arn:aws:codebuild:<REGION>:<AWS ACCOUNT ID>:project/parallelcluster-*",
      "Effect": "Allow",
      "Sid": "CodeBuild"
    },
    {
      "Action": [
        "ecr:*"
      ],
      "Resource": "*",
      "Effect": "Allow",

```

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

```

```

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

```

## ParallelClusterLambdaPolicy usando SGE, Slurm, ou Torque

O exemplo a seguir define o `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.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
      ],
      "Resource": "arn:aws:logs:*:*:*",
      "Effect": "Allow",
      "Sid": "CloudWatchLogsPolicy"
    },
    {

```

```
"Action": [
  "s3:DeleteBucket",
  "s3:DeleteObject",
  "s3:DeleteObjectVersion",
  "s3:ListBucket",
  "s3:ListBucketVersions"
],
"Resource": [
  "arn:aws:s3:::*"
],
"Effect": "Allow",
"Sid": "S3BucketPolicy"
},
{
  "Action": [
    "ec2:DescribeInstances"
  ],
  "Resource": "*",
  "Effect": "Allow",
  "Sid": "DescribeInstances"
},
{
  "Action": [
    "ec2:TerminateInstances"
  ],
  "Resource": "*",
  "Effect": "Allow",
  "Sid": "FleetTerminatePolicy"
},
{
  "Action": [
    "dynamodb:GetItem",
    "dynamodb:PutItem"
  ],
  "Resource": "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*",
  "Effect": "Allow",
  "Sid": "DynamoDBTable"
},
{
  "Action": [
    "route53:ListResourceRecordSets",
    "route53:ChangeResourceRecordSets"
  ],
  "Resource": [
```

```
    "arn:aws:route53::hostedzone/*"
  ],
  "Effect": "Allow",
  "Sid": "Route53DeletePolicy"
}
]
}
```

## ParallelClusterLambdaPolicy usando awsbatch

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

```
{
  "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](#)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "MinimumModify",
      "Action": [
        "autoscaling:UpdateAutoScalingGroup",
        "batch:UpdateComputeEnvironment",
        "cloudformation:DescribeStackEvents",
```

```

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

```

## 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 versões 2.11.4 e uso anterior Son of Grid Engine 8.1.9.

## Slurm Workload Manager (**slurm**)

AWS ParallelCluster a versão 2.11.9 usa Slurm 20.11.9. Para obter mais informações sobre 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 Slurm parâmetros de configuração, que são fornecidos por padrão. Quaisquer alterações que você fizer nesses Slurm os parâmetros de configuração são feitos por sua conta e risco. Eles recebem suporte somente com base no melhor esforço.

AWS ParallelCluster versão (ões)	Compatível Slurm versão
2.11.7, 2.11.8, 2.11.9	20.11.9
2.11.4 a 2.11.6	20.11.8

AWS ParallelCluster versão (ões)	Compatível Slurm versão
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
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 [\[queue\]seções](#) é uma partição em Slurm Workload Manager. Para obter mais informações, consulte [Slurm guia para o modo de fila múltipla](#) [Tutorial do modo de fila múltipla](#) e.

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" é "*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, onde \$CLUSTERNAME está o nome da [\[cluster\]seção](#) 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.

## Slurm guia para o modo de fila múltipla

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 Slurm cluster com a arquitetura de escalabilidade recém-introduzida.

### Visão geral

A nova arquitetura de escalabilidade é baseada em Slurm [Guia de agendamento em nuvem](#) e plug-in de economia de energia. Para obter mais informações sobre o plug-in de economia de energia, consulte [Slurm Guia de economia de energia](#). Na nova arquitetura, os recursos que podem ser potencialmente disponibilizados para um cluster são normalmente predefinidos no Slurm configuração 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 nenhuma EC2 instância apoiando o 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 o trabalho para um nó em um POWER\_SAVING estado, o nó é automaticamente transferido para um POWER\_UP estado. 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 e as EC2 instâncias são iniciadas e configuradas para apoiar um POWER\_UP nó.
- 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 em 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 `scaledown-idletime` configuração é salva no Slurm configuração como `SuspendTimeout` configuração.
- Um nó que está off-line aparece com um sufixo \* (por exemplo, down\*) em `sinfo`. Um nó fica off-line se Slurm o controlador não pode entrar em contato com o nó ou se os nós estáticos estiverem desativados e as instâncias de apoio 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 `POWER_SAVING` estado sem nenhuma EC2 instância apoiando-os.

### Trabalhando com um nó disponível

Um nó disponível é apoiado por uma EC2 instância. Por padrão, o nome do nó pode ser usado 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 nós que não estão disponíveis, o `NodeAddr` campo não deve apontar para uma EC2 instância 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 um 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]"
```

Envie um trabalho para um GPU nó da `gpu` fila.

```
$ 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. A Slurm partição é uma fila de entrada. 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 a fila (conhecido 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 POWER\_SAVING estado: execute o `scontrol update nodename=nodename state=power_up` comando ou envie um `sleep 1` trabalho de espaço reservado solicitando um determinado número de nós e confie em Slurm para alimentar o número necessário de nós.
- Desligue os nós dinâmicos antes [scaledown\\_idletime](#): defina os nós dinâmicos 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 EC2 instâncias e atribui os endereços IP privados e os nomes de host de `queue1-dy-c5xlarge-[1-2]`, aguardando `ResumeTimeout` (o período padrão é 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) EC2 instância 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 `POWER_DOWN` estado e o trabalho é enfileirado 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 a hora Slurm espera antes de colocar o nó no estado inativo.
  - Pode ser útil estender isso se o processo de pré/pós-instalação demorar tanto assim.
  - 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 menor `SuspendTimeout` significaria que os nós serão redefinidos mais rapidamente e Slurm é capaz de 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` for mais do que alguns minutos, Slurm tenta percorrer todos os nós do sistema. Um mais longo `SuspendTimeout` pode ser benéfico para sistemas de grande escala (mais de 1.000 nós) para reduzir o estresse em Slurm reenfileirando trabalhos que falharam com frequência.
- Observe que `SuspendTimeout` 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}`, onde `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 `slurmctld` registro para determinar se Slurm já tentou ligar `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

- Nós substituídos/encerrados 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 EC2 relacionado, use ferramentas para verificar o status ou os registros dessa instância. Por exemplo, você pode verificar se a instância tinha eventos programados ou falhou nas verificações do status de EC2 saúde.
- Se `clustermgtd` não encerrou o nó, verifique se o nó `computemgtd` foi encerrado ou se a instância EC2 foi encerrada 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 e `NodeAddr` redefinições de instâncias são concluídos por `clustermgtd`. Slurm coloca nós em IDLE depois `SuspendTimeout`.

### Outros problemas

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

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 versões 2.11.4 e uso anterior Torque Resource Manager 6.1.2. Para obter mais informações sobre Torque Resource Manager 6.1.2, veja <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 versões 2.4.0 e uso anterior 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 CLI comandos para AWS Batch

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

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

#### Important

AWS ParallelCluster não suporta GPU empregos para AWS Batch. Para obter mais informações, consulte [GPUempregos](#).

## 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 GPU empregos para AWS Batch. Para obter mais informações, consulte [GPUempregos](#).

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 vCPUs a serem reservados para o contêiner. Quando usado junto com `nodes`, ele identifica o número de vCPUs 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` timestamp 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>, tipo =<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 o primeiro *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 EC2 execução do cluster, EC2 aplique as tags do 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 EC2 seção do AWS Management Console.

### Visualizar tags

1. Navegue pelo EC2 console 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	<i>clustername</i>
Name	Master
Application	parallelcluster- <i>clustername</i>
aws:ec2launchtemplate:id	<i>lt-1234567890abcdef0</i>
aws:ec2launchtemplate:version	<i>1</i>
aws-parallelcluster-node-type	Master
aws:cloudformation:stack-name	parallelcluster- <i>clustername</i> - MasterServerSubstack- <i>ABCD1234E</i> <i>FGH</i>
aws:cloudformation:logical-id	MasterServer
aws:cloudformation:stack-id	arn:aws:cloudformation: <i>region-id</i> : <i>ACCOUNTID</i> :stack/parallelclu ster- <i>clustername</i> -MasterSe rverSubstack- <i>ABCD1234E</i> <i>FGH</i> / <i>1234abcd-12ab-12ab-12ab-123</i> <i>4567890abcdef0</i>
Version	<i>2.11.9</i>

## AWS ParallelCluster etiquetas de volume raiz do nó principal

Chave de tag	Valor da tag
ClusterName	<i>clustername</i>
Application	parallelcluster- <i>clustername</i>
aws-parallelcluster-node-type	Master

## AWS ParallelCluster tags de instância do nó de computação

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

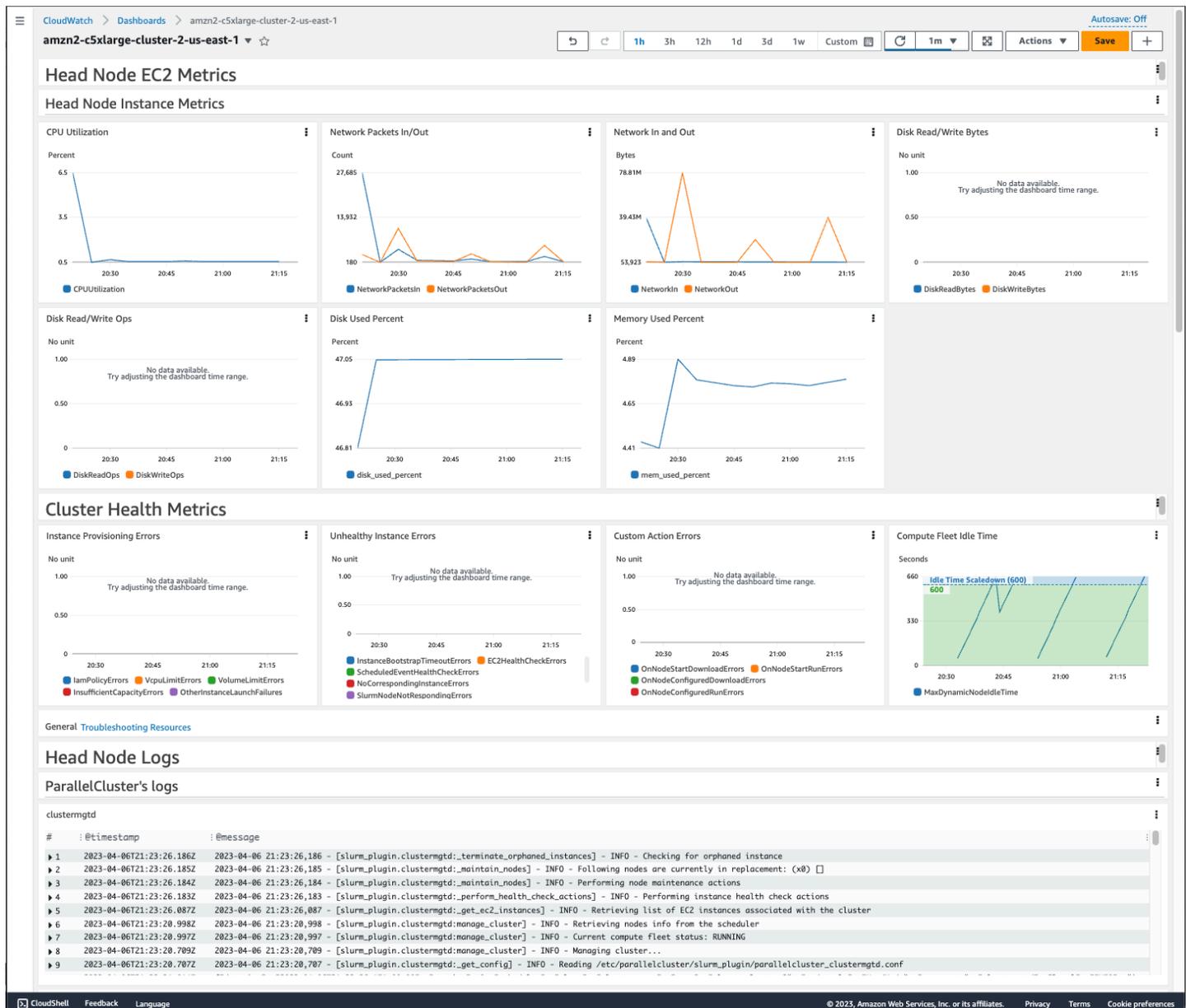
## AWS ParallelCluster tags de volume raiz do nó de computação

Chave de tag	Valor da tag
ClusterName	<i>clustername</i>
Application	parallelcluster- <i>clustername</i>
aws-parallelcluster-node-type	Compute
QueueName	<i>queue-name</i>
Version	<i>2.11.9</i>

## 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 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 EC2 métricas do Head Node. 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 NICE DCV integração 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 `logIdentifier` e caminho 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	awsbatc slurm	HeadNode	dcv-authenticator: /var/log/parallelcluster/parallelcluster_dcv_authenticator.log  dcv-ext-authenticator: /var/log/parallelcluster/parallelcluster_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	awsbatc slurm	ComputeNode HeadNode	cloud-init: /var/log/cloud-init.log  supervisord: /var/log/supervisord.log
amazon centos ubuntu	slurm	ComputeNode	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	Programas	Nodes	Fluxos de log
			slurm_suspend: /var/log/parallelcluster/slurm_suspend.log slurmctld: /var/log/slurmctld.log
amazon centos	awsbatch slurm	Compute HeadNodes	system-messages: /var/log/messages
ubuntu	awsbatch slurm	Compute HeadNodes	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 desvio do sistema operacional para comunicações de rede de baixa latência com outras instâncias na mesma sub-rede.

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

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

Para ver a lista de EC2 instâncias compatíveis EFA, consulte [Tipos de instância compatíveis](#) no Guia do EC2 usuário da Amazon para instâncias 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 ter mais informações, consulte [placement](#) e [placement\\_group](#).

Para obter mais informações, consulte o [adaptador de malha elástica](#) no Guia EC2 do usuário da Amazon e [escale HPC cargas de trabalho com adaptador de malha elástica e AWS ParallelCluster](#) no blog de código AWS aberto.

#### Note

Por padrão, Ubuntu distribuições habilitam ptrace proteção (rastreamento de processo). Começando com AWS ParallelCluster 2,6.0, ptrace a proteção está desativada para que o Libfabric funcione corretamente. Para obter mais informações, consulte [Desativar a proteção ptrace](#) no Guia do EC2 usuário da Amazon.

#### Note

Support EFA em instâncias Graviton2 baseadas em ARM foi adicionado na AWS ParallelCluster versão 2.10.1.

## 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 HPC Plataforma Intel](#), 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 HPC da plataforma Intel, consulte [Especificação HPC da plataforma Intel](#).

## Ativar Intel MPI

A Intel MPI está disponível no AWS ParallelCluster AMIs. Para usar a IntelMPI, você deve reconhecer e aceitar os termos da [licença simplificada do software Intel](#). Por padrão, Abrir MPI é colocado no caminho. Para habilitar o Intel MPI em vez do OpenMPI, você deve primeiro carregar o MPI módulo Intel. 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 a Intel MPI está ativada, 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 MPI módulo Intel é carregado, vários caminhos são alterados para usar as MPI ferramentas Intel. Para executar o código compilado pelas MPI ferramentas da Intel, carregue primeiro o MPI módulo Intel.

#### Note

A Intel MPI não é compatível com instâncias AWS baseadas em Graviton.

#### Note

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

## Especificação HPC da plataforma Intel

AWS ParallelCluster é compatível com a especificação da HPC plataforma Intel. A Especificação da HPC Plataforma Intel fornece um conjunto de requisitos de computação, malha, memória, armazenamento e software para ajudar a alcançar um alto padrão de qualidade e compatibilidade com HPC cargas de trabalho. Para obter mais informações, consulte [Especificação HPC da plataforma Intel](#) e [aplicativos verificados como compatíveis com a especificação da HPC plataforma Intel](#).

Para estar em conformidade com a Especificação da HPC Plataforma Intel, os seguintes requisitos devem ser atendidos:

- O sistema operacional deve ser CentOS (7 [base\\_os](#) = centos7).
- O tipo de instância dos nós de computação deve ter uma Intel CPU 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 (NFScompartilhado a partir do nó principal, Amazon EBS ou FSx 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. `Armp1-21.0.0` precisa GCC de `-9.3` como requisito. Quando você carrega o `armp1/21.0.0` módulo, o `gcc/9.3` módulo 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 armp1`. A saída será conforme segue:

```
$ module avail

----- /usr/share/Modules/modulefiles
-----
armp1/21.0.0      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 da Amazon DCV

DCVA Amazon é uma tecnologia de visualização remota que permite que os usuários se conectem com segurança a aplicativos 3D com uso intensivo de gráficos hospedados em um servidor remoto de alto desempenho. Para obter mais informações, consulte [Amazon DCV](#).

O DCV software da Amazon 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 ARM instância, o DCV software da Amazon será instalado automaticamente nele ao usar `base_os = alinux2`, `base_os = centos7`, ou `base_os = ubuntu1804`.

Para habilitar a Amazon DCV no nó principal, `dcv_settings` deve conter o nome de uma `[dcv]seção` que tem `enable = master` e `base_os` deve ser definida como `alinux2centos7`, `ubuntu1804`, ou `ubuntu2004`. Se o nó principal for uma ARM instância, `base_os` deverá ser definido como `alinux2centos7`, ou `ubuntu1804`. Dessa forma, AWS ParallelCluster define o parâmetro de configuração do cluster `shared_dir` para a [pasta de armazenamento DCV do servidor](#).

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

Para obter mais informações sobre os parâmetros DCV de configuração da Amazon, consulte [dcv\\_settings](#). Para se conectar à DCV sessão da Amazon, use o `pcluster dcv` comando.

### Note

O suporte para Amazon DCV on centos8 foi removido na AWS ParallelCluster versão 2.10.4. O suporte para 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 for Amazon DCV on alinux2 e ubuntu1804 foi adicionado na AWS ParallelCluster versão 2.6.0. O suporte para Amazon DCV on centos7 foi adicionado na AWS ParallelCluster versão 2.5.0.

**Note**

A Amazon não DCV é compatível com instâncias AWS baseadas em Graviton nas AWS ParallelCluster versões 2.8.0 e 2.8.1.

## DCVHTTPSCertificado Amazon

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

Para substituir o DCV certificado padrão autoassinado da Amazon 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 obter mais informações, consulte [Alteração do TLS certificado](#) no Amazon DCV Administrator Guide.

## Licenciamento da Amazon DCV

O DCV servidor da Amazon não exige um servidor de licenças quando executado em EC2 instâncias da Amazon. No entanto, o DCV servidor da Amazon 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 IAM instância personalizada, use as permissões descritas em [Amazon DCV on Amazon EC2](#) no Amazon DCV Administrator Guide.

Para obter dicas de solução de problemas, consulte [Solução de problemas na Amazon DCV](#).

## Utilizar 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` em [\[fsx fs\]](#) 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 e substituição de EC2 instâncias

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

No entanto, devido a problemas de segurança emergentes, os clientes podem querer adicionar patches a eles AMIs e, em seguida, atualizar seus clusters com os patches AMI. Isso se alinha ao [Modelo de Responsabilidade Compartilhada do AWS ParallelCluster](#).

Para ver o conjunto específico de AWS ParallelCluster AMIs suportado pela AWS ParallelCluster CLI versão 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 instâncias no Guia EC2 do usuário da Amazon para instâncias Linux. Você não pode atualizar um AMI para um cluster existente.

A reinicialização e reinicialização do nó principal com AMI atualizações das instâncias de computação do 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 da 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 for EFS, Amazon FSx for Lustre 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 Slurm agendador:

- 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 AWS Management Console 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 CLI comandos

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

Para usar `pcluster`, você deve ter uma IAM função com as [permissões](#) necessárias para executá-la.

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

## Tópicos

- [pcluster](#)
- [pcluster-config](#)

## pcluster

`pcluster` é o AWS ParallelCluster CLI comando principal. Você usa `pcluster` para iniciar e gerenciar HPC clusters no 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 [Configuração 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.

Padronizado como `~/.parallelcluster/config`.

Para obter mais informações, consulte [Configuração 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 noVPC, você pode excluir a pilha CloudFormation de rede. O nome da pilha começa com "parallelclusternetworking-" e contém a hora de 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
```

O VPC que [pcluster configure](#) cria para você não é criado na pilha CloudFormation de rede. Você pode excluí-lo VPC manualmente no console ou usando AWS CLI o.

```
$ 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 é `parallelcluster-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.

Padronizado como `~/.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. `-rou --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.

Padronizado como `False`.

### **-nr, --norollback**

Desabilita a reversão da pilha do quando ocorrer erro.

Padronizado como `False`.

### **-u *TEMPLATE\_URL*, --template-url *TEMPLATE\_URL***

Especifica um 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 um personalizado AMI 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 do 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 `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 base AMI a ser usada para construir o AWS ParallelCluster AMI

**-os *BASE\_AMI\_OS*, --os *BASE\_AMI\_OS***

Especifica o sistema operacional da baseAMI. 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 o AMI

Padronizado como `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 resultante. AWS ParallelCluster AMI

Padronizado como `custom-ami-`.

**-cc** *CUSTOM\_AMI\_COOKBOOK*, **--custom-cookbook** *CUSTOM\_AMI\_COOKBOOK*

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

**--post-install** *POST\_INSTALL\_SCRIPT*

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

Padronizado como `~/.parallelcluster/config`.

**-t *CLUSTER\_TEMPLATE*, --cluster-template *CLUSTER\_TEMPLATE***

Especifica a [seção \[cluster\]](#) do *CONFIG\_FILE* para usar 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 do VPC a ser usado para criar o 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 o AWS ParallelCluster AMI

 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 DCV servidor da Amazon 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 feita antes da URL expiração, execute `pcluster dcv connect` novamente para gerar uma nova 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 chave da SSH chave a ser usada para a conexão.

A chave deve ser aquela especificada no momento da criação do cluster no parâmetro de configuração [key\\_name](#). Esse argumento é opcional, mas se não for especificado, a chave deverá estar disponível por padrão para o SSH cliente. 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 uma única vez URL para se conectar à DCV sessão da Amazon. O navegador padrão não é aberto quando essa opção é especificada.

**i Note**

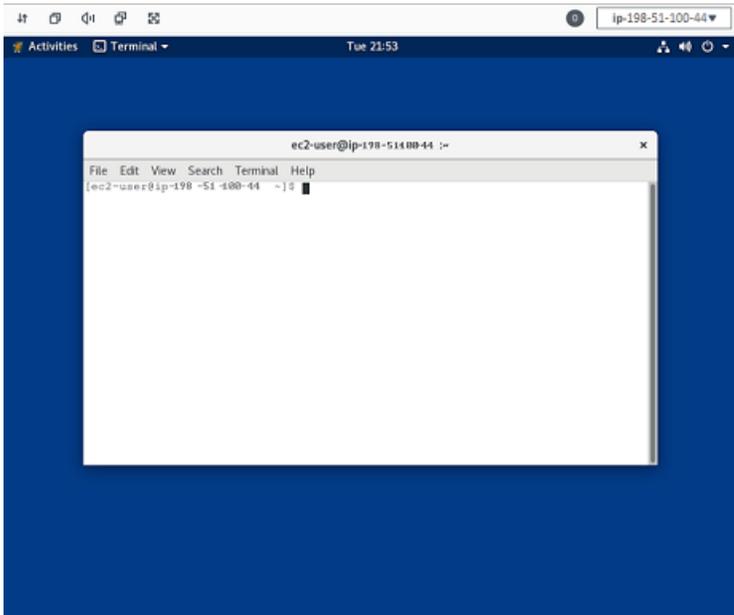
Support para o `--show-url` argumento foi adicionado na AWS ParallelCluster versão 2.5.1.

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 DCV sessão da Amazon em execução no nó principal.

Uma nova DCV sessão da Amazon é 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.

Padronizado como `~/.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 noVPC, você pode excluir a pilha CloudFormation de rede. O nome da pilha começa com "parallelclusternetworking-" e contém a hora de 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
```

O VPC que [pcluster configure](#) cria para você não é criado na pilha CloudFormation de rede. Você pode excluí-lo VPC manualmente no console ou usando AWS CLI o.

```
$ 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

#### **nome\_do\_cluster**

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.

Padronizado como `~/.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.

Padronizado como `False`.

#### **-c CONFIG\_FILE, --config CONFIG\_FILE**

Especifica o arquivo de configuração alternativo a ser usado.

Padronizado como `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.

Padronizado como 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.

Padronizado como `~/.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.

Padronizado como `~/.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.

Padronizado como `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.

Padronizado como `~/.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 do Auto Scaling como mínimo/máximo/desejado = 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 [Utilizar 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.

Padronizado como `~/.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.

Padronizado como `False`.

### **-nw, --nowait**

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

Padronizado como `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.

Padronizado como 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.

Padronizado como `~/.parallelcluster/config`.

Para obter mais informações, consulte [Configuração 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, o AWS ParallelCluster usa o arquivo `~/.parallelcluster/config` 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 arquivo de configuração de exemplo é instalado com o AWS ParallelCluster no diretório Python em `site-packages/aws-parallelcluster/examples/config`. O exemplo de arquivo de configuração também está disponível no GitHub, em <https://github.com/aws/aws-parallelcluster/blob/v2.11.9/cli/src/pcluster/examples/config>.

Atual AWS ParallelCluster versão 2: 2.11.9.

## Tópicos

- [Layout](#)
- [Seção \[global\]](#)
- [Seção \[aws\]](#)
- [Seção \[aliases\]](#)
- [Seção \[cluster\]](#)
- [Seção \[compute\\_resource\]](#)
- [Seção \[cw\\_log\]](#)
- [Seção \[dashboard\]](#)
- [Seção \[dcv\]](#)
- [Seção \[ebs\]](#)
- [Seção \[efs\]](#)
- [Seção \[fsx\]](#)
- [Seção \[queue\]](#)
- [Seção \[raid\]](#)
- [Seção \[scaling\]](#)
- [Seção \[vpc\]](#)
- [Exemplos](#)

# Layout

Uma configuração do AWS ParallelCluster 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](#)
- [update\\_check](#)
- [sanity\\_check](#)

## **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 da seção deve começar com uma letra, conter no máximo 60 caracteres e conter apenas letras, números, hifens (-) e sublinhados (\_).

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 do AWS ParallelCluster versão 2.5.0, o padrão para [sanity\\_check](#) 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. `-rou --region` parâmetro para [pcluster create](#).
2. `AWS_DEFAULT_REGION` variável de ambiente
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, perfis do IAM do Amazon EC2 para o Amazon EC2 ou a [AWS CLI](#), em vez de salvar as credenciais no arquivo de configuração do AWS ParallelCluster.

```
[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, hífen (-) 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.

O valor deve ser HTTP URL de 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 IAM políticas para a AmazonEC2. Essa lista é anexada à função raiz usada no cluster, além das permissões exigidas pelo AWS ParallelCluster separadas por vírgulas. O nome IAM de uma política 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 passe as IAM políticas personalizadas adicionais com a `additional_iam_policies` configuração em vez de usar as [ec2\\_iam\\_role](#) configurações para adicionar suas EC2 políticas 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`. Neste 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* é o nome de uma das AWS CloudFormation pilhas usadas pelo 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 obter mais informações, consulte [Função vinculada ao serviço para solicitações de instâncias spot](#) no Guia EC2 do usuário da Amazon.

```
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 EC2 instância da Amazon que é usado para os nós de computação do 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). Eles AMI devem apoiar `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 o ID de um personalizado AMI a ser usado para os nós principais e de computação, em vez do padrão publicado. AMIs](#) Para obter mais informações, consulte [Modificar uma AMI do](#) ou [Criar uma AMI personalizada do AWS ParallelCluster](#).

Não há valor padrão.

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

Se o personalizado AMI exigir permissões adicionais para seu lançamento, essas permissões deverão ser adicionadas às políticas do usuário e do nó principal.

Por exemplo, se um cliente AMI tiver um snapshot criptografado associado a ele, as seguintes políticas adicionais serão necessárias nas políticas do usuário e do nó principal:

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

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 [dcv] seção com a DCV configuração da Amazon. 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 começa [dcv custom-dcv] é usada para a DCV configuração da Amazon.

```
dcv_settings = custom-dcv
```

### Note

Em instâncias AWS baseadas em Graviton, a Amazon só DCV é suportada em. `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 vCPUs 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 DNS entradas 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 obter uma lista de tipos de instância que suportam a desativação do hyperthreading, consulte [CPU núcleos e threads para cada CPU núcleo para cada tipo de instância](#) no Guia do usuário da 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 `[ebs]` seções com os EBS volumes da Amazon que estão montados no nó principal. Ao usar vários EBS volumes da Amazon, 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, hífens (-) e sublinhados (\_).

Há suporte para até cinco (5) EBS volumes adicionais da Amazon.

Para obter mais informações, consulte a [seção \[ebs\]](#).

Por exemplo, a configuração a seguir especifica as seções que começam `[ebs custom1]` e `[ebs custom2]` são usadas para os EBS volumes da Amazon.

```
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 uma IAM função existente para a Amazon EC2 que está anexada a todas as instâncias no cluster. O nome de uma IAM função 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 for adicionar políticas extras às configurações padrão dos nós do cluster, recomendamos que você passe as IAM políticas personalizadas adicionais com a [additional\\_iam\\_policies](#) configuração em vez de usar as `ec2_iam_role` configurações.

Se essa opção não for especificada, a AWS ParallelCluster IAM função padrão da 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 EFS arquivos da Amazon. 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 configuração a seguir especifica que a seção que começa [`efs customfs`] é usada para a configuração do sistema de EFS arquivos da Amazon.

```
efs_settings = customfs
```

[Política de atualização: se essa configuração for alterada, a atualização não será permitida.](#)

## enable\_efa

(Opcional) Se presente, especifica que o Elastic Fabric Adapter (EFA) está habilitado para os nós de computação. Para ver a lista de EC2 instâncias compatíveis EFA, consulte [Tipos de instância](#)

[compatíveis](#) no Guia do EC2 usuário da Amazon para instâncias 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 ter mais informações, consulte [placement](#) e [placement\\_group](#).

```
enable_efa = compute
```

### Note

Support para EFA instâncias Graviton2 baseadas em ARM foi adicionado na AWS ParallelCluster versão 2.10.1.

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 GPUDirect RDMA (EFA) para (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) para GPUDirect 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`. EFAo suporte para GPUDirect RDMA é suportado por tipos de instância específicos (`p4d.24xlarge`) em sistemas operacionais específicos ([base\\_os](#) é `alinux2centos7`, `ubuntu1804`, ou `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 ter 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 [enable\\_intel\\_hpc\\_platform](#) configuração só é suportada em 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 efêmeras 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 extra JSON que é mesclado no Chef `dnainstaller.json`. Para obter mais informações, consulte [Criação de uma AMI personalizada do AWS ParallelCluster](#).

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 suporta o uso de SGE ou Torque agendadores.

O nome de uma IAM função 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 EC2 instâncias da Amazon 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 suporta o uso de SGE ou Torque agendadores.

Essa configuração é aplicável somente para agendadores tradicionais (SGE, Slurm e Torque). Se a `maintain_initial_size` configuração for `true`, a `initial_queue_size` configuração deverá ser pelo menos uma (1).

Se o programador for `awsbatch`, use `min_vcpus` em vez disso.

Padronizado como 2.

```
initial_queue_size = 2
```

[Política de atualização: essa configuração pode ser alterada durante uma atualização.](#)

## key\_name

(Opcional) Nomeia um par de EC2 chaves da Amazon existente com o qual permitir o SSH acesso 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 suporta o uso de SGE ou Torque agendadores.

(Opcional) Mantém o tamanho inicial do grupo 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 booleano. Se definido como `true`, o grupo de 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 de Auto Scaling pode ter instâncias interrompidas, e o tamanho pode cair para menos de [initial\\_queue\\_size](#).

Se for definido como `false`, o grupo de 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 \[compute\\_resource\]](#).

Padronizado como `false`.

```
maintain_initial_size = false
```

Política de atualização: essa configuração pode ser alterada durante uma atualização.

## master\_instance\_type

(Opcional) Define o tipo de EC2 instância da Amazon 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). Eles AMI devem apoiargrowroot.

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 EC2 instâncias da Amazon 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 suporta o uso de SGE ou Torque agendadores.

Essa configuração é aplicável somente para agendadores tradicionais (SGE, Slurm e Torque).

Se o programador for `awsbatch`, use [max\\_vcpus](#) em vez disso.

Padronizado como 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 vCPUs 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 suporta o uso de SGE ou Torque agendadores.

Se o agendador 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](#).

Padronizado como 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 obter mais informações, consulte [Capacidade de instância insuficiente](#) no Guia EC2 do usuário da Amazon. 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 de grupo de posicionamento de EC2 clusters existente da Amazon

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 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 `t3.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](#) de clusters 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: se essa configuração for alterada, a atualização não será permitida.](#)

## post\_install

(Opcional) Especifica o script URL 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://bucketname/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 o script URL de pré-instalação 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://bucketname/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 HTTPS proxy HTTP ou um servidor proxy, 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 de computação homogênea e quais [seções \[queue\]](#) 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 [raid] seção com a RAID configuração do EBS volume da Amazon. 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` balde e para os objetos no balde.

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 qual os nós do AWS ParallelCluster obtêm acesso leitura/gravação.

Por exemplo, `arn:aws:s3:::my_corporate_bucket/Development/*` fornece acesso de leitura/gravação a todos os objetos na Development pasta do `my_corporate_bucket` balde.

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, hífens (-) 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 suporta o uso de 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 suporta o uso de 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 em que o EBS volume compartilhado da Amazon é montado.

Não use essa opção com vários EBS volumes da Amazon. Em vez disso, forneça valores de [shared\\_dir](#) em cada [seção \[ebs\]](#).

Consulte a [\[ebs\]seção](#) para obter detalhes sobre como trabalhar com vários EBS volumes da Amazon.

O valor padrão é `/shared`.

O exemplo a seguir mostra um EBS volume compartilhado da Amazon 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 suporta o uso de SGE ou Torque agendadores.

(Opcional) Define o preço spot máximo para ComputeFleet os programadores tradicionais (SGE, Slurm e Torque). Usado somente quando a [cluster\\_type](#) configuração 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 AWS 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 JSON formatadas. Não use aspas fora das chaves.

Para obter mais informações, consulte [tipos de tags de recurso do AWS 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.

Padronizado como `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 [vpc] seção com a VPC configuração da Amazon em que o cluster está 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 [vpc public] é usada para a VPC configuração da Amazon.

```
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 apenas letras, números, hifens (-) e sublinhados (\_).

```
[compute_resource cr1]
instance_type = c5.xlarge
min_count = 0
initial_count = 2
max_count = 10
spot_price = 0.5
```

### Note

O suporte para a [\[compute\\_resource\] seção](#) no AWS ParallelCluster foi adicionado na versão 2.9.0.

## Tópicos

- [initial\\_count](#)
- [instance\\_type](#)
- [max\\_count](#)
- [min\\_count](#)
- [spot\\_price](#)

## initial\_count

(Opcional) Define o número inicial de instâncias do Amazon EC2 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](#).

Padronizado como 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 instância do Amazon EC2 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 instâncias do Amazon EC2 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](#).

Padronizado como 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 configuração `max_count` 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 máximo de instâncias do Amazon EC2 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.

Padronizado como 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 configuração `min_count` 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 o 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 apenas letras, números, hifens (-) e sublinhados (\_).

```
[cw_log custom-cw-log]  
enable = true  
retention_days = 14
```

Para obter mais informações, consulte [Integração com Amazon CloudWatch Logs](#), [CloudWatch Painel da Amazon](#) e [Integração com Amazon CloudWatch Logs](#).

### Note

O suporte para cw\_log foi adicionado ao AWS ParallelCluster versão 2.5.0.

## enable

(Opcional) Indica se o CloudWatch Logs está habilitado.

O valor padrão é true. Use false para desativar o CloudWatch Logs.

O exemplo a seguir ativa o CloudWatch Logs.

```
enable = true
```

[Política de atualização: se essa configuração for alterada, a atualização não será permitida.](#)

## retention\_days

(Opcional) Indica quantos dias o CloudWatch Logs retém eventos de log 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 o CloudWatch Logs para reter eventos de log por 30 dias.

```
retention_days = 30
```

[Política de atualização: essa configuração pode ser alterada durante uma atualização.](#)

## Seção [dashboard]

Define as configurações para o painel do CloudWatch.

O formato é [dashboard *dashboard-name*]. O *nome do painel* deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) e sublinhados (\_).

```
[dashboard custom-dashboard]  
enable = true
```

### Note

O suporte para dashboard foi adicionado ao AWS ParallelCluster 2.5.0.

## enable

(Opcional) Indica se o painel do CloudWatch está ativado.

O valor padrão é true. Use false para desativar o painel do CloudWatch.

O exemplo a seguir ativa o painel do CloudWatch.

```
enable = true
```

[Política de atualização: essa configuração pode ser alterada durante uma atualização.](#)

## Seção [dcv]

Define as configurações do DCV servidor Amazon em execução no nó principal.

Para criar e configurar um DCV servidor Amazon, especifique o cluster [dcv\\_settings](#) com o nome que você define na dcv seção e [enable](#) defina como `master`, e [base\\_os](#) como `alinux2centos7`, `ubuntu1804` ou `ubuntu2004`. Se o nó principal for uma ARM instância, [base\\_os](#) defina como `alinux2centos7`, 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, hífen (-) e sublinhados (\_).

```
[dcv custom-dcv]  
enable = master  
port = 8443  
access_from = 0.0.0.0/0
```

Para ter mais informações, consulte [Conecte-se ao nó principal por meio da Amazon DCV](#)

### Important

Por padrão, a DCV porta Amazon configurada por AWS ParallelCluster está aberta a todos os IPv4 endereços. No entanto, você pode se conectar a uma DCV porta da Amazon somente se tiver a URL DCV sessão da Amazon e se conectar à DCV sessão da Amazon dentro de 30 segundos após URL a devolução `opcluster dcv connect`. Use a [access\\_from](#) configuração para restringir ainda mais o acesso à DCV porta da Amazon com um intervalo de CIDR IP formatado e use a [port](#) configuração para definir uma porta não padrão.

### 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 CIDR IP formatado para conexões com a 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 a Amazon DCV está habilitada no nó principal. Para habilitar a Amazon DCV no nó principal e configurar a regra de grupo de segurança necessária, defina a `enable` configuração como `master`.

O exemplo a seguir habilita a Amazon DCV no nó principal.

```
enable = master
```

### Note

A Amazon gera DCV automaticamente um certificado autoassinado que é usado para proteger o tráfego entre o DCV cliente da Amazon e o DCV servidor da Amazon em execução no nó principal. Para configurar seu próprio certificado, consulte [DCVHTTPS Certificado Amazon](#).

[Política de atualização: se essa configuração for alterada, a atualização não será permitida.](#)

## port

(Opcional) Especifica a porta para a AmazonDCV.

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 ter mais informações, consulte [Melhores práticas: migrar um cluster para um novo AWS ParallelCluster versão secundária ou patch](#).

O formato é [ebs *ebs-name*]. *ebs-name* deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) 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](#)
- [ebs\\_kms\\_key\\_id](#)
- [ebs\\_snapshot\\_id](#)
- [ebs\\_volume\\_id](#)
- [encrypted](#)
- [volume\\_iops](#)
- [volume\\_size](#)
- [volume\\_throughput](#)
- [volume\\_type](#)

## 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 ter 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-xxxxxx
```

[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-xxxxxxx
```

[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 do usuário do Amazon EC2.

## **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 [Tipos de volume do Amazon EBS](#) no Guia do usuário do Amazon EC2.

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 [CreateFileSystem](#) na 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 sistema de arquivos do Amazon EFS a partir das configurações [efs] 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: migrar um cluster para um novo AWS ParallelCluster versão secundária ou patch](#).

O formato é [efs *efs-name*]. *efs-name* deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) 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

Ao definir um destino de montagem com o tráfego de entrada e saída do NFS permitido de 0.0.0.0/0, o sistema de arquivos é exposto a solicitações de montagem do NFS de qualquer lugar na zona de disponibilidade do destino de montagem. O AWS não recomenda criar um destino de montagem na zona de disponibilidade da pilha. Em vez

disso, deixe o 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 gerenciada pelo cliente AWS Key Management Service (AWS KMS) a ser usada para proteger o sistema de arquivos criptografado. Se isso estiver definido, a [encrypted](#) configuração deverá ser definida como `true`. Isso corresponde ao parâmetro `KmsKeyId` na Referência de 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 parâmetro [PerformanceMode](#) na Referência de 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 taxa de transferência 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 a taxa de transferência provisionada do sistema de arquivos, medida em MiB/s. Isso corresponde ao parâmetro [ProvisionedThroughputInMibps](#) na Referência de 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 AWS Support.

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 taxa de transferência do sistema de arquivos. Isso corresponde ao parâmetro [ThroughputMode](#) na Referência de 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 definições de configuração para um sistema de arquivos do FSx para Lustre. Para obter mais informações, consulte [CreateFileSystem do Amazon FSx](#) na Referência da API do Amazon FSx.

Se [base\\_os](#) for `alinux2`, `centos7`, `ubuntu1804`, ou `ubuntu2004`, o FSx para Lustre é suportado.

Ao usar o Amazon Linux, o kernel deve ser `4.14.104-78.84.amzn1.x86_64` ou uma versão posterior. Para obter instruções de instalação para outras distribuições Linux, consulte [Instalação do cliente Lustre](#) no Guia de usuário do Amazon FSx para Lustre.

 Note

No momento, não há suporte para o Lustre ao usar `awsbatch` como programador.

 Note

O suporte para FSx para Lustre do `centos8` foi removido do AWS ParallelCluster versão 2.10.4. O suporte para FSx para Lustre do `ubuntu2004` foi adicionado ao AWS ParallelCluster versão 2.11.0. O suporte para FSx para Lustre do `centos8` foi adicionado ao AWS ParallelCluster versão 2.10.0. O suporte para FSx para Lustre do `alinux2`, do `ubuntu1604`, e do `ubuntu1804` foi adicionado ao AWS ParallelCluster versão 2.6.0. O suporte para FSx para Lustre do `centos7` foi adicionado ao 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 IDs de grupos de segurança mais específicos. Isso será feito automaticamente quando não estiver usando [vpc\\_security\\_group\\_id](#).

Para usar um sistema de arquivos existente do Amazon FSx 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](#), o AWS ParallelCluster cria o sistema de arquivos do FSx para Lustre a partir das configurações `[fsx]` 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: migrar um cluster para um novo AWS ParallelCluster versão secundária ou patch](#).

O formato é `[fsx fsx-name]`. *nome-fsx* deve começar com uma letra, conter no máximo 30 caracteres e conter apenas letras, números, hifens (-) 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 para Lustre. Os valores possíveis são os seguintes:

### NEW

O FSx para Lustre importa automaticamente listagens de diretórios de quaisquer novos objetos adicionados ao bucket do S3 vinculado que não existam atualmente no sistema de arquivos FSx para Lustre.

### NEW\_CHANGED

O FSx para Lustre importa automaticamente listagens de arquivos e diretórios de quaisquer novos objetos adicionados ao bucket do S3 e quaisquer objetos existentes que sejam alterados no bucket do S3.

Isso corresponde à propriedade [AutoImportPolicy](#). Para obter mais informações, consulte [Importar atualizações do bucket do S3 automaticamente](#) no Guia do usuário do Amazon FSx para 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 configuração [auto\\_import\\_policy](#) não for especificada, as importações automáticas serão desabilitadas. O FSx para Lustre atualiza somente as listagens de arquivos e diretórios do bucket do S3 vinculado quando o sistema de arquivos é criado.

```
auto_import_policy = NEW_CHANGED
```

### Note

O suporte para [auto\\_import\\_policy](#) foi adicionado ao 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 propriedade [AutomaticBackupRetentionDays](#).

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 propriedade [CopyTagsToBackups](#).

O valor padrão é false.

```
copy_tags_to_backups = true
```

### Note

O suporte para [copy\\_tags\\_to\\_backups](#) foi adicionado ao 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 propriedade [DailyAutomaticBackupStartTime](#).

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 propriedade [dataCompressionType](#). Para obter mais informações, consulte [Compactação de dados do FSx para Lustre](#) no Guia do usuário do Amazon FSx para Lustre.

O único valor válido é LZ4. Para desabilitar a compactação de dados, remova o parâmetro [data\\_compression\\_type](#).

```
data_compression_type = LZ4
```

**Note**

O suporte para [data\\_compression\\_type](#) foi adicionado ao AWS ParallelCluster versão 2.11.0.

Política de atualização: essa configuração pode ser alterada durante uma atualização.

## deployment\_type

(Opcional) Especifica o tipo de implantação do FSx para Lustre. Isso corresponde à propriedade [DeploymentType](#). Para obter mais informações, consulte [Opções de implantação do FSx para Lustre](#) no Guia do usuário do Amazon FSx para 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 uma taxa de transferência de intermitência mais alta 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 para FSx para Lustre. Com esse tipo de implantação, a configuração [storage\\_capacity](#) tem valores possíveis de 1200, 2400 e qualquer múltiplo de 3600. O suporte para SCRATCH\_1 foi adicionado ao 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 [Criptografia de dados em trânsito](#) no Guia do usuário do Amazon FSx para 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. O suporte para PERSISTENT\_1 foi adicionado ao AWS ParallelCluster versão 2.6.0.

O valor padrão é SCRATCH\_1.

```
deployment_type = SCRATCH_2
```

#### Note

O suporte para [deployment\\_type](#) foi adicionado ao 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 propriedade [DriveCachetype](#). Para obter mais informações, consulte [Opções de implantação do FSx para Lustre](#) no Guia do usuário do Amazon FSx para 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

O suporte para [drive\\_cache\\_type](#) foi adicionado ao 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 propriedade [exportPath](#). 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 para repositório de dados](#) no Guia do usuário do Amazon FSx para 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 propriedade [BackupId](#).

```
fsx_backup_id = backup-fedcba98
```

### Note

O suporte para [fsx\\_backup\\_id](#) foi adicionado ao 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 do FSx para Lustre existente.

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 de chave da sua chave gerenciada pelo cliente do AWS Key Management Service (AWS KMS).

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 parâmetro [KmsKeyId](#) na Referência de API do Amazon FSx.

```
fsx_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

### Note

O suporte para [fsx\\_kms\\_key\\_id](#) foi adicionado ao 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 parâmetro [ImportPath](#) na Referência de API do Amazon FSx.

A importação ocorre na criação do cluster. Para obter mais informações, consulte [Importar dados do repositório de dados](#) no Guia do usuário do Amazon FSx para 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 para 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 propriedade [ImportedFileChunkSize](#).

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 taxa de transferência de leitura e gravação para cada 1 tebibyte (TiB) de armazenamento, em MB/s/TiB. A capacidade de throughput do sistema de arquivos é calculada multiplicando-se a capacidade de armazenamento do sistema de arquivos (TiB) pela [per\\_unit\\_storage\\_throughput](#) (MB/s/TiB). Para um sistema de arquivos de 2,4 TiB, o provisionamento de 50 MB/s/TiB [per\\_unit\\_storage\\_throughput](#) gera 120 MB/

s de taxa de transferência do sistema de arquivos. Você paga pela quantidade de throughput que provisiona. Isso corresponde à propriedade [PerUnitStorageThroughput](#).

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 para o sistema de arquivos do FSx para Lustre nos nós principal e 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 propriedade [StorageCapacity](#).

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 versões 2.5.0 e 2.5.1 do AWS ParallelCluster, a [storage\\_capacity](#) tem os valores possíveis de 1200, 2400 e qualquer múltiplo de 3600. Para versões anteriores à 2.5.0 do AWS ParallelCluster, a [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 propriedade [StorageType](#). 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, ou PERSISTENT\_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 propriedade [WeeklyMaintenanceStartTime](#).

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>]. O *nome-fila* deve começar com uma letra minúscula, conter no máximo 30 caracteres e conter apenas letras minúsculas, números e hifens (-).

```
[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](#)
- [compute\\_type](#)
- [disable\\_hyperthreading](#)
- [enable\\_efa](#)
- [enable\\_efa\\_gdr](#)
- [placement\\_group](#)

## 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 do usuário do Amazon EC2](#).

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 do 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 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. 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 sempre habilitado para os nós de computação se for suportado pelo tipo de instância.

### Note

AWS ParallelCluster versão 2.10.0 a 2.11.2: `Set true`, especifica que o Elastic Fabric Adapter (EFA) GPUDirect RDMA (acesso direto remoto à memória) está habilitado para os nós

nessa fila. Definir isso como `true` exige que a configuração [enable\\_efa](#) seja definida como `true`. O EFA GPUDirect RDMA é compatível com os seguintes tipos de instância (p4d.24xlarge) nesses sistemas operacionais (alinux2, centos7, ubuntu1804, 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 do grupo com posicionamento em cluster do Amazon EC2

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 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 obter mais informações, consulte [Capacidade de instância insuficiente](#) 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 `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](#) de clusters 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: 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 apenas letras, números, hifens (-) 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 ter 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](#) do usuário do Amazon EC2.

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 do usuário do Amazon EC2.

## `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 [Tipos de volume do Amazon EBS](#) no Guia do usuário do Amazon EC2.

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 apenas letras, números, hifens (-) 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 VPC configurações da Amazon. Para obter mais informações sobre VPCs, consulte [O que é a Amazon VPC?](#) e [as melhores práticas de segurança para você VPC](#) no Guia VPC do usuário da Amazon.

O formato é [vpc *vpc-name*]. *vpc-name* deve começar com uma letra, conter no máximo 30 caracteres e conter somente letras, números, hífen (-) e sublinhados (\_).

```
[vpc public]
vpc_id = vpc-xxxxxx
master_subnet_id = subnet-xxxxxx
```

### Tópicos

- [additional\\_sg](#)
- [compute\\_subnet\\_cidr](#)
- [compute\\_subnet\\_id](#)
- [master\\_subnet\\_id](#)
- [ssh\\_from](#)
- [use\\_public\\_ips](#)
- [vpc\\_id](#)
- [vpc\\_security\\_group\\_id](#)

## additional\_sg

(Opcional) Fornece um ID adicional do grupo VPC de segurança da Amazon para todas as instâncias.

Não há valor padrão.

```
additional_sg = sg-xxxxxx
```

## compute\_subnet\_cidr

(Opcional) Especifica um bloco de roteamento entre domínios () sem classe. 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, você deverá configurar o acesso NAT 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 um intervalo de CIDR IP formatado a partir do qual permitir SSH o acesso.

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, todas as Contas da AWS estão limitadas a cinco (5) endereços IP elásticos para cada uma 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 da Amazon VPC no qual provisionar o cluster.

```
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 o SSH acesso 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 a Amazon DCV estiver habilitada, o grupo de segurança permitirá o acesso à Amazon DCV usando a porta 8443 (ou o que a [port](#) configuração especificar) 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-idem \[fsx fs\]](#).

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 as configurações do AWS ParallelCluster usando os programadores Slurm, Torque, e AWS Batch.

#### Note

A partir da versão 2.11.5, AWS ParallelCluster não suporta o uso de programadores SGE ou Torque.

## 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 `t3.micro` `spot` 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 (**sg**e) e Torque Resource Manager (**torque**)

### Note

Aplica-se somente ao 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 executa um cluster com o programador torque ou sge Para usar SGE, mude `scheduler = torque` 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 programadores SGE ou Torque. Se você usar essas versões, poderá continuar usando-as ou solucionando problemas de suporte das equipes de serviço AWS 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.

O exemplo de configuração a seguir permite um máximo de 40 vCPUs simultâneas e reduz a zero quando nenhuma tarefa for executada por 10 minutos (ajustável usando a configuração [scaledown\\_idletime](#)).

```
[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 o AWS ParallelCluster funciona

O AWS ParallelCluster foi criado não apenas como uma maneira de gerenciar clusters, mas como uma referência sobre como usar os serviços da AWS para criar seu ambiente de HPC.

## Tópicos

- [Processos do AWS ParallelCluster](#)
- [AWS serviços usados por AWS ParallelCluster](#)
- [Auto Scaling do AWS ParallelCluster](#)

## Processos do AWS ParallelCluster

Esta seção se aplica somente aos clusters de HPC que são implantados com um dos programadores de trabalhos tradicionais compatíveis (SGE, Slurm ou Torque). Quando usado com esses programadores, o AWS ParallelCluster gerencia o provisionamento e a remoção do nó de computação interagindo tanto com o Grupo de Auto Scaling como com o programador de tarefas subjacente.

Para clusters de HPC baseados no AWS Batch, o AWS ParallelCluster conta com os recursos fornecidos pelo AWS Batch para o gerenciamento do nó de computação.

### Note

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 serviço AWS e de suporte AWS.

## Tópicos

- [SGE and Torque integration processes](#)
- [Slurm integration processes](#)

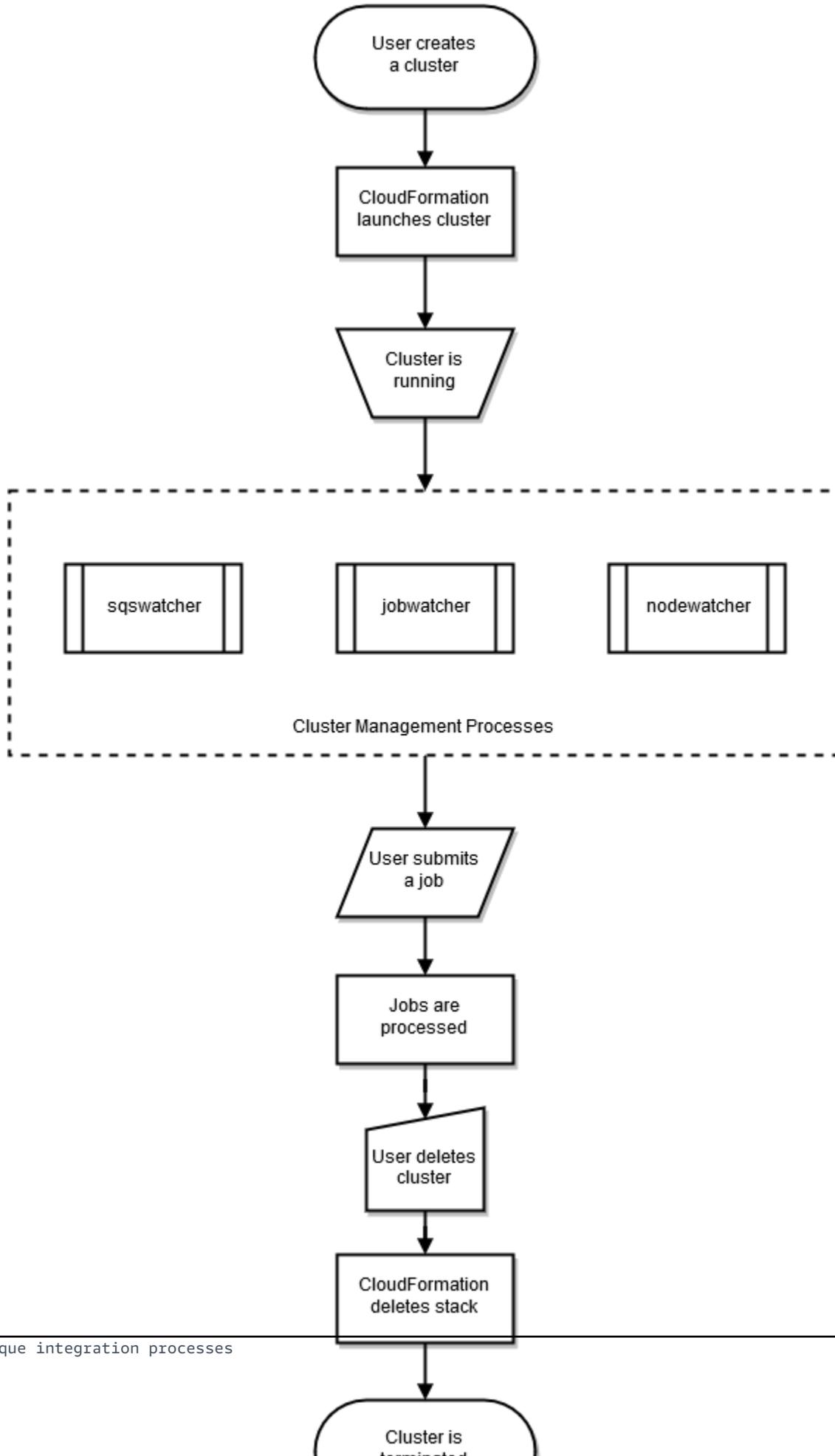
## SGE and Torque integration processes

### Note

Esta seção se aplica somente ao 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 de programadores SGE e Torque, Amazon SNS e Amazon SQS.

### Visão geral

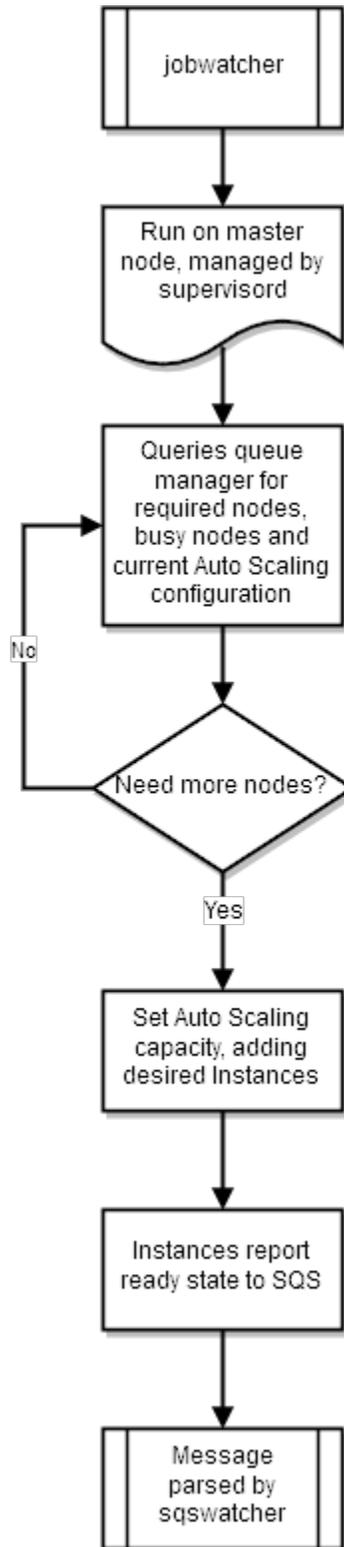
O ciclo de vida de um cluster começa após ele ser criado por um usuário. Normalmente, um cluster é criado a partir da interface de linha de comando (CLI). Após sua criação, um cluster existirá até ser excluído. Os daemons do AWS ParallelCluster são executados nos nós de cluster, principalmente para gerenciar a elasticidade do cluster de 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 daemons do AWS ParallelCluster que são usados para gerenciar o cluster.



Com programadores SGE e Torque, o AWS ParallelCluster usa os processos `nodewatcher`, `jobwatcher`, e `sqswatcher`.

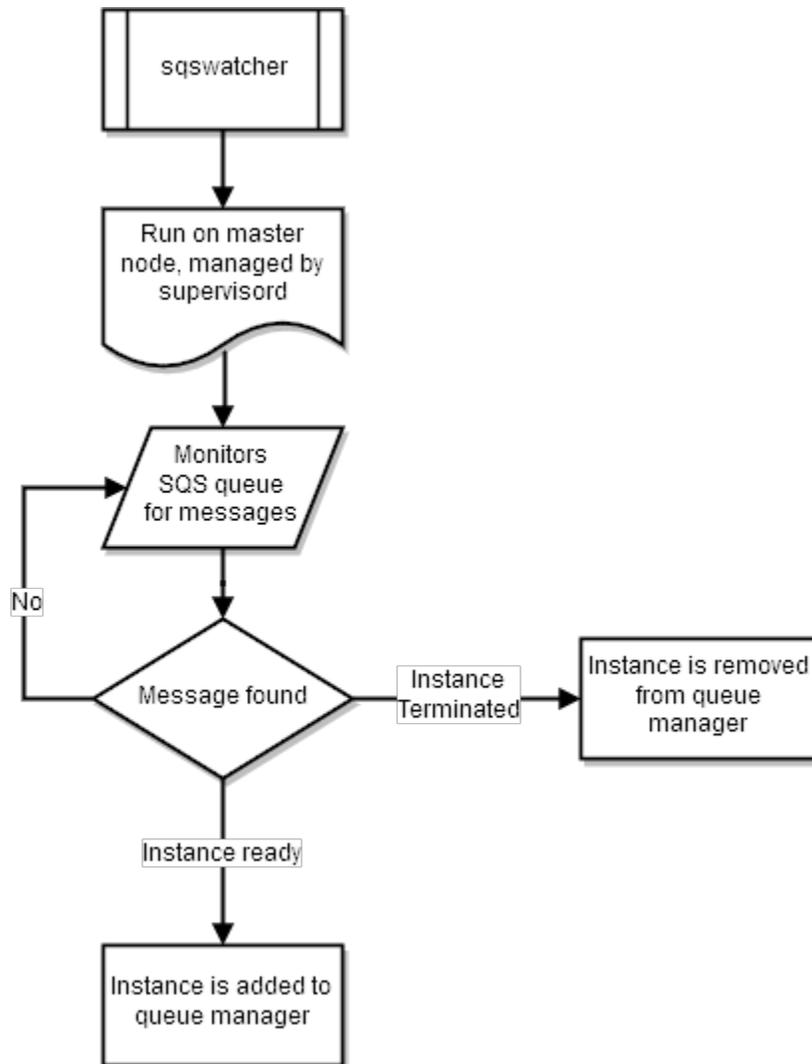
## **jobwatcher**

Quando um cluster está em execução, um processo de propriedade do usuário raiz monitora o programador configurado (SGE ou Torque). A cada minuto, ele avalia a fila para decidir quando aumentar a escala.



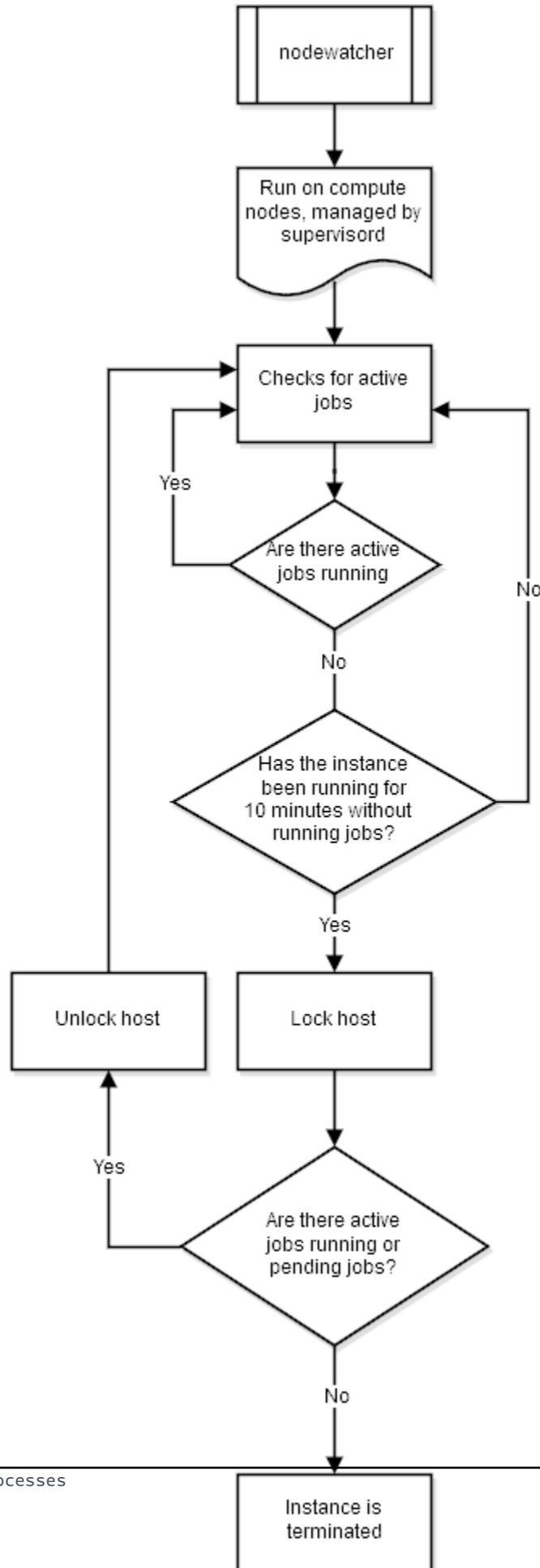
## sqswatcher

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 `sqswatcher`, 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.



## Slurm integration processes

Com programadores Slurm, o AWS ParallelCluster usa os processos `clustermgtd` e `computemgtd`.

### `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
- Programador de sincronização com o Amazon EC2.
- Limpeza de instâncias órfãs
- Restaura o status do nó do programador quando ocorre encerramento do Amazon EC2 fora do fluxo de trabalho de suspensão
- Gerenciamento de instâncias não íntegras do Amazon EC2 (falha nas verificações de integridade do Amazon EC2)
- 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](#)
- [AWS 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 Scaling é 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 Scaling é 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 Batch é um serviço AWS gerenciado de agendamento de tarefas. Ele provisiona dinamicamente a quantidade e o tipo ideais de recursos computacionais (por exemplo, CPU ou instâncias otimizadas para 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 Batch é 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>.

## AWS CloudFormation

AWS CloudFormation é 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 ParallelCluster é representado como uma pilha, e todos os recursos exigidos por cada cluster são definidos no AWS ParallelCluster AWS CloudFormation modelo. Na maioria dos casos, AWS ParallelCluster CLI os comandos correspondem diretamente aos comandos de AWS CloudFormation pilha, como comandos de criação, atualização e exclusão. As instâncias que são executadas em um cluster fazem HTTPS chamadas para o AWS CloudFormation endpoint em Região da AWS que o cluster é executado.

Para obter mais informações sobre AWS 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 rápido e flexível sem banco de dados. SQL 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 (AmazonEBS) é um serviço de armazenamento em blocos de alto desempenho que fornece armazenamento persistente para volumes compartilhados. Todas as EBS configurações da Amazon podem ser passadas pela configuração. EBSOs volumes da Amazon podem ser inicializados vazios ou a partir de um EBS snapshot existente da Amazon.

Para obter mais informações sobre a AmazonEBS, consulte <https://aws.amazon.com/ebs/https://docs.aws.amazon.com/ebs/e>.

## Amazon Elastic Compute Cloud

O Amazon Elastic Compute Cloud (AmazonEC2) fornece a capacidade de computação para AWS ParallelCluster. Os nós principais e de computação são EC2 instâncias da Amazon. Qualquer tipo de instância compatível HVM pode ser selecionado. 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 LVM volumes distribuídos.

Para obter mais informações sobre a AmazonEC2, consulte <https://aws.amazon.com/ec2/https://docs.aws.amazon.com/ec2/e>.

## Amazon Elastic Container Registry

O Amazon Elastic Container Registry (AmazonECR) é um registro de contêineres Docker totalmente gerenciado que facilita o armazenamento, o gerenciamento e a implantação de imagens de contêineres Docker. Em AWS ParallelCluster, a 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.

A Amazon ECR é usada somente com AWS Batch clusters.

Para obter mais informações, consulte <https://aws.amazon.com/ecr/https://docs.aws.amazon.com/ecr/e>.

## Amazon EFS

O Amazon Elastic File System (AmazonEFS) fornece um sistema de NFS arquivos elástico simples, escalável e totalmente gerenciado para uso com Nuvem AWS serviços e recursos locais. A Amazon EFS é usada quando a [efs\\_settings](#) configuração é especificada e se refere a uma [\[efs\]seção](#). O suporte para Amazon EFS foi adicionado na AWS ParallelCluster versão 2.1.0.

Para obter mais informações sobre a AmazonEFS, consulte <https://aws.amazon.com/efs/https://docs.aws.amazon.com/efs/e>.

## Amazon FSx para Lustre

FSxfor Lustre fornece um sistema de arquivos de alto desempenho que usa o sistema de arquivos Lustre de código aberto. FSxfor 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 IAM função menos privilegiada para a Amazon EC2 na instância específica de cada cluster individual. AWS ParallelCluster as instâncias recebem acesso somente às API chamadas específicas necessárias para implantar e gerenciar o cluster.

Com AWS Batch clusters, IAM funções 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 Lambda que têm permissão para adicionar e excluir imagens do Docker de e para o repositório da 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 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 personalizados, como imagens do Docker armazenadas no ECR repositório da Amazon 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

DCVA Amazon é um protocolo de exibição remota de alto desempenho que fornece uma maneira segura de fornecer desktops remotos e streaming de aplicativos para qualquer dispositivo em condições de rede variáveis. A Amazon DCV é usada quando a [dcv\\_settings](#) configuração é especificada e se refere a uma [\[dcv\] seção](#). O suporte para Amazon DCV foi adicionado na AWS ParallelCluster versão 2.5.0.

Para obter mais informações sobre a AmazonDCV, 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 (AmazonSNS) 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 SNS tópico da Amazon para o grupo Auto Scaling é inscrito em uma fila da Amazon. SQS

A Amazon não SNS é usada com AWS Batch clusters.

Para obter mais informações sobre a AmazonSNS, consulte <https://aws.amazon.com/sns/https://docs.aws.amazon.com/sns/e>.

## 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 (AmazonSQS) retém notificações enviadas pelo Auto Scaling, notificações enviadas pela SNS Amazon e notificações enviadas dos nós computacionais. A Amazon SQS dissocia 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.

A Amazon não SQS é usada com AWS Batch clusters.

Para obter mais informações sobre a AmazonSQS, consulte <https://aws.amazon.com/sqs/https://docs.aws.amazon.com/sqs/e>.

## Amazon Simple Storage Service

O Amazon Simple Storage Service (Amazon S3) AWS ParallelCluster armazena modelos localizados em cada um. Região da AWS AWS ParallelCluster pode ser configurado para permitir que CLI SDK / tools use o Amazon S3.

Quando você usa o AWS Batch cluster, um bucket do Amazon S3 na 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/https://docs.aws.amazon.com/s3/e>.

## Amazon VPC

A Amazon VPC define uma rede usada pelos nós em seu cluster. As VPC configurações do cluster são definidas na [\[vpc\] seção](#).

Para obter mais informações sobre a AmazonVPC, consulte <https://aws.amazon.com/vpc/https://docs.aws.amazon.com/vpc/e>.

## Auto Scaling do AWS ParallelCluster

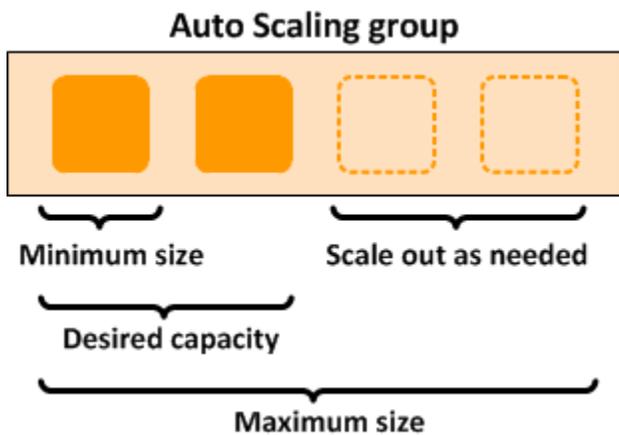
### Note

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 auto scaling descrita neste tópico se aplica aos clusters de HPC implantados com Son of Grid Engine (SGE) ou Torque Resource Manager (). Torque Quando implantado com um desses programadores, o AWS ParallelCluster implementa os recursos de escalabilidade gerenciando o Grupo de Auto Scaling (ASG) dos nós de computação e alterando a configuração do programador conforme necessário. Para clusters de HPC baseados no AWS Batch, o AWS ParallelCluster conta com os recursos de escalabilidade elástica fornecidos pelo AWS gerenciado pela AWS. Para obter mais informações, consulte [O que é o Amazon EC2 Auto Scaling](#) no Guia do usuário do Amazon EC2 Auto Scaling.

Os clusters implantados com o AWS ParallelCluster são elásticos de diversas maneiras. A definição de [initial\\_queue\\_size](#) especifica o valor de tamanho mínimo do grupo de Auto Scaling de ComputeFleet e também o valor da capacidade desejada. A definição de [max\\_queue\\_size](#) especifica o tamanho máximo valor do grupo do Auto Scaling de ComputeFleet.



## Aumentar

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 ASG 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 de computação que corresponde ao número de vCPUs disponíveis depende do tipo de instância do que é especificado na configuração do cluster.

Com os programadores Slurm (antes da versão 2.9.0 do AWS ParallelCluster) e Torque, cada trabalho pode exigir um número de nós e um número de slots por 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 ASG 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

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 API [TerminateInstanceInAutoScalingGroup](#), que remove uma instância se o tamanho do grupo do Auto Scaling for pelo menos do tamanho mínimo exigido para ele. 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 carga de trabalho. 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 ASG de ComputeFleet tenha o mesmo valor para capacidade mínima, máxima e desejada.

# Tutoriais

Os tutoriais a seguir mostram como começar a usar o AWS ParallelCluster e oferecem orientação sobre as melhores práticas para algumas tarefas comuns.

## Tópicos

- [Como executar o seu primeiro trabalho no AWS ParallelCluster](#)
- [Criação de uma AMI personalizada do AWS ParallelCluster](#)
- [Execução de uma tarefa de MPI com o AWS ParallelCluster e o programador awsbatch](#)
- [Criptografia de disco com uma chave do KMS personalizada](#)
- [Tutorial do modo de fila múltipla](#)

## Como executar o seu primeiro trabalho no AWS ParallelCluster

Este tutorial orienta você durante a execução de sua primeira tarefa Hello World no AWS ParallelCluster.

### 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](#).

## Verificar a instalação

Primeiro, verifique se o AWS ParallelCluster está instalado e configurado corretamente.

```
$ pcluster version
```

Isso retorna a versão do AWS ParallelCluster em execução. 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 carga de trabalho desse tutorial não é de desempenho intensivo, podemos usar o tamanho de instância padrão de `t2.micro`. (Para cargas de trabalho 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.

## Fazendo 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	SWAPT0
global	-	-	-	-	-	-	-	-	-
ip-192-168-1-125	lx-amd64	2	1	2	2	0.15	3.7G	130.8M	1024.0M

```
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.

## Executar o primeiro trabalho usando o SGE

### Note

Aplica-se somente ao 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.

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 AMI personalizada do AWS ParallelCluster

### Important

Não recomendamos a criação de uma AMI personalizada como abordagem para personalizar o AWS ParallelCluster.

A razão é porque depois de criar sua própria AMI, você não receberá mais atualizações nem correções de erros com futuras versões do AWS ParallelCluster. Além disso, se você criar uma AMI personalizada, será necessário repetir as etapas usadas para criar sua AMI personalizada com cada versão nova do AWS ParallelCluster.

Antes de continuar a leitura, recomendamos que você consulte a seção [Ações de bootstrap personalizadas](#) para determinar se as modificações que deseja fazer podem ser expressas em scripts e são compatíveis com versões futuras do AWS ParallelCluster.

Embora criar uma AMI personalizada não seja o ideal (devido aos motivos mencionados anteriormente), ainda existem cenários em que é necessário criar uma AMI personalizada para o

AWS ParallelCluster. Este tutorial orienta você pelo processo de criação de uma AMI personalizada para esses cenários.

### Note

A partir do AWS ParallelCluster versão 2.6.1, a maioria das fórmulas de instalação é ignorada por padrão quando os nós são iniciados. 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 startup, 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 AMI do AWS ParallelCluster

Há três maneiras de usar uma AMI personalizada do AWS ParallelCluster, 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 AMI base do AWS ParallelCluster geralmente é atualizada com novas versões, essa AMI tem todos os componentes necessários para o AWS ParallelCluster funcionar quando instalada e configurada. Comece com isso como base.

## New EC2 console

1. Na lista de AMI do AWS ParallelCluster, encontre a AMI que corresponde à sua Região da AWS específica. A lista de AMI que você escolher deve corresponder à versão do AWS ParallelCluster que você usa. Execute `pcluster version` para verificar a versão. Para o 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 AWS Management Console 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 as AMIs da comunidade e insira o ID da AMI do AWS ParallelCluster para sua Região da AWS na caixa de pesquisa.
5. Selecione a AMI, escolha 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 AMI do AWS ParallelCluster, encontre a AMI que corresponde à sua Região da AWS específica. A lista de AMI que você escolher deve corresponder à versão do AWS ParallelCluster que você usa. Execute `pcluster version` para verificar a versão. Para o 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 AWS Management Console 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 AMIs da comunidade, pesquise o ID da AMI do AWS ParallelCluster e selecione-o.
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.

## Criar uma AMI personalizada do AWS ParallelCluster

Se você tiver uma AMI personalizada e software já existentes, aplique as alterações necessárias pelo AWS ParallelCluster.

1. Instale as seguintes ferramentas em seu sistema local, junto com a CLI do AWS ParallelCluster:
  - 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 do AWS ParallelCluster versão 2.8.0, o [Berkshelf](#) (instalado usando o `gem install berkshelf`) era obrigatório para usar o `pcluster createami`.

2. Configure as credenciais de sua Conta da AWS para que o Packer possa fazer chamadas para operações da API da AWS 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 CLI do AWS ParallelCluster para criar uma AMI do AWS ParallelCluster a partir daquela fornecida como base:

```
pcluster createami --ami-id <BASE_AMI> --os <BASE_AMI_OS>
```

### Important

Você não deve usar uma AMI do AWS ParallelCluster de um cluster em execução como `<BASE_AMI>` para o comando `createami`. 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 livro de receitas do AWS ParallelCluster à instância para instalar o software 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 AMI personalizada do AWS ParallelCluster é um `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 com AMIs potencialmente não testadas em runtime, as incompatibilidades com a instalação em runtime do software necessário do AWS ParallelCluster podem fazer com que o AWS ParallelCluster pare de funcionar.

Se não quiser criar nada antecipadamente, use sua AMI e crie um AWS ParallelCluster a partir dessa AMI.

Com esse método, leva mais tempo para que o AWS ParallelCluster seja criado porque todo o software necessário para o AWS ParallelCluster deve estar instalado para a criação do cluster. Além disso, o aumento de escala também leva mais tempo.

- Insira o ID de AMI no campo [custom\\_ami](#) na configuração do cluster.

# Execução de uma tarefa de MPI com o AWS ParallelCluster e o programador **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 do [pcluster](#).

## Criar o cluster

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 o nosso cluster de *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
```

## Fazendo login em seu nó principal

Os comandos da [CLI do Batch do AWS ParallelCluster](#) estão todos disponíveis na máquina cliente em que o AWS ParallelCluster está instalado. No entanto, aplicaremos SSH para o nó principal e enviaremos as tarefas de lá. Isso permite aproveitar o volume de NFS que é compartilhado entre as instâncias principal e do Docker que executam tarefas do AWS Batch.

Use o arquivo pem SSH para fazer login no nó principal.

```
$ pcluster ssh awsbatch-tutorial -i /path/to/keyfile.pem
```

Depois de fazer login, execute os comandos `awsbqueues` e `awsbhosts` para mostrar a fila configurada do AWS Batch 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 quiser exibir detalhes adicionais sobre os hosts e a fila do AWS Batch, adicione o sinalizador `-d` ao comando.

## Executar o primeiro trabalho usando o 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.

```
$ awsbatch 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
jobId                : 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
jobName              : hello
createdAt            : 2018-11-12 09:41:21
startedAt            : 2018-11-12 09:41:29
stoppedAt            : -
status               : RUNNING
statusReason         : -
jobDefinition        : parallelcluster-myBatch:1
jobQueue             : parallelcluster-myBatch
command              : /bin/bash -c 'aws s3 --region us-east-1 cp s3://
parallelcluster-mybatch-lui1ftboklhpsn95/batch/job-hellojob_sh-1542015680924.sh /
tmp/batch/job-hellojob_sh-1542015680924.sh; bash /tmp/batch/job-
hellojob_sh-1542015680924.sh Luca'
exitCode             : -
reason               : -
vcpus                : 1
memory[MB]           : 128
nodes                : 1
logStream            : parallelcluster-myBatch/default/c75dac4a-5aca-4238-
a4dd-078037453554
log                  : https://console.aws.amazon.com/cloudwatch/home?region=us-
east-1#logEventViewer:group=/aws/batch/job;stream=parallelcluster-myBatch/default/
c75dac4a-5aca-4238-a4dd-078037453554
-----
```

Observe que, no momento, a tarefa está em um estado RUNNING. Aguarde 30 segundos para que a tarefa seja concluída e execute `awsbatch` novamente.

```
$ awsbatch
jobId                jobName      status      startedAt
stoppedAt            exitCode
-----
-----
```

Veja que agora a tarefa está com status SUCCEEDED.

```
$ awsbatch -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://parallelcluster-mybatch-lui1ftboklhpn95/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://parallelcluster-awsbatch-tutorial-iwyl4458saiwgwvg/batch/job-
submit_mpi_sh-1543333713772.sh to tmp/batch/job-submit_mpi_sh-1543333713772.sh
2018-11-27 15:51:12: ip container: 10.0.0.180
2018-11-27 15:51:12: ip host: 10.0.0.245
2018-11-27 15:51:12: Compiling...
2018-11-27 15:51:12: Running...
2018-11-27 15:51:12: Hello I'm the main node! I run the mpi job!
2018-11-27 15:51:12: Warning: Permanently added '10.0.0.199' (RSA) to the list of known
hosts.
2018-11-27 15:51:12: Warning: Permanently added '10.0.0.147' (RSA) to the list of known
hosts.
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-180.ec2.internal, rank 1 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-199.ec2.internal, rank 5 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-180.ec2.internal, rank 0 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-199.ec2.internal, rank 4 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-147.ec2.internal, rank 2 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-147.ec2.internal, rank 3 out
of 6 processors

[ec2-user@ip-10-0-0-111 ~]$ awsbatch -s 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#1
2018-11-27 15:50:52: Job id: 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#1
2018-11-27 15:50:52: Initializing the environment...
2018-11-27 15:50:52: Starting ssh agents...
2018-11-27 15:50:52: Agent pid 7
2018-11-27 15:50:52: Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
2018-11-27 15:50:52: Mounting shared file system...
2018-11-27 15:50:52: Generating hostfile...
2018-11-27 15:50:52: Starting the job...
download: s3://parallelcluster-awsbatch-tutorial-iwyl4458saiwgwvg/batch/job-
submit_mpi_sh-1543333713772.sh to tmp/batch/job-submit_mpi_sh-1543333713772.sh
```

```

2018-11-27 15:50:53: ip container: 10.0.0.199
2018-11-27 15:50:53: ip host: 10.0.0.227
2018-11-27 15:50:53: Compiling...
2018-11-27 15:50:53: Running...
2018-11-27 15:50:53: Hello I'm a compute node! I let the main node orchestrate the mpi
  execution!

```

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 ParallelClusterO oferece suporte às opções de configuração `ebs_kms_key_id` e `fsx_kms_key_id`. Essas opções permitem fornecer uma chave personalizada do AWS KMS para criptografia de disco do Amazon EBS ou do FSx para Lustre. Para usá-las, especifique uma `ec2_iam_role`.

Para que o cluster seja criado, a chave do AWS KMS precisa 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 chaves EC2](#).
- Você tem um perfil do IAM com as [permissões](#) necessárias para executar a CLI do [pcluster](#).

## Criar a função

Primeiro, crie uma política:

1. Acesse o console do IAM: <https://console.aws.amazon.com/iam/home>
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.

Clique no botão Adicionar na caixa Usuários de chave, abaixo da guia Política de chaves, e procure o `ParallelClusterInstanceRole` que você acabou de criar. Anexe-a.

## Criar o cluster

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 do FSx para 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 a sistemas de arquivos baseados em Amazon EBS e Amazon FSx.

## Tutorial do modo de fila múltipla

### Como executar seus trabalhos no AWS ParallelCluster com o modo de várias filas

Este tutorial orienta você durante a execução de sua primeira tarefa Hello World no AWS ParallelCluster com [Modo de fila múltipla](#).

#### 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](#).

### Note

O modo de fila múltipla só é compatível com o AWS ParallelCluster versão 2.9.0 ou posterior.

## Configuração do cluster

Primeiro, verifique se o AWS ParallelCluster está instalado corretamente, executando o comando a seguir.

```
$ pcluster version
```

Para ter mais informações sobre o `pcluster version`, consulte [pcluster version](#).

Esse comando retorna a versão do AWS ParallelCluster em execução.

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 ter 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.

## Fazendo login em seu nó de cabeçalho

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 ter 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](#) na documentação Slurm.

```
$ 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 após seu `scaledown_idletime` personalizado de cinco minutos.

Todos os outros nós estão no estado de economia de energia, indicado pelo sufixo `~` no estado do nó, sem nenhuma instância do EC2 suportando-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 [sbatch](#) na 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](#) na documentação do Slurm.

```
$ 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 em AWS ParallelCluster são chamadas de partições no 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 comando `squeue` 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 estado `CF` (CONFIGURING), aguardando que as instâncias aumentem e se juntem ao cluster.

Após cerca de três minutos, os nós deverão estar disponíveis e o trabalho entra no estado `R` (RUNNING).

```
$ 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 e entram no modo `POWER_DOWN`. Observe que o nó estático `spot-st-t2micro-1` não é reduzido.

```

$ 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 [Slurm guia para o modo de fila múltipla](#).

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.



```
|=====|
| 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. Consequentemente, depois de cinco minutos em estado ocioso, seus nós dinâmicos `spot-dy-c5xlarge-1` e `spot-dy-t2micro-1` são automaticamente reduzidos 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 [Slurm guia para o modo de fila múltipla](#).

# Desenvolvimento

Você pode usar as seguintes seções para iniciar 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 do AWS ParallelCluster, com possíveis problemas que podem ser difíceis de depurar. É altamente recomendável pela equipe do AWS ParallelCluster o uso dos scripts em [Ações de bootstrap personalizadas](#) para a personalização, já que os ganchos de pós-instalação geralmente são mais fáceis de depurar e mais portáteis entre as versões do AWS ParallelCluster.

## Tópicos

- [Configuração de um livro de receitas personalizado do AWS ParallelCluster](#)
- [Configuração de um pacote de nós personalizado do AWS ParallelCluster](#)

# Configuração de um livro de receitas personalizado do AWS ParallelCluster

## Important

Veja a seguir as instruções para usar uma versão personalizada das receitas do livro de receitas do AWS ParallelCluster. Esse é um método avançado de personalização do AWS ParallelCluster, com possíveis problemas que podem ser difíceis de depurar. É altamente recomendável pela equipe do AWS ParallelCluster o uso dos scripts em [Ações de bootstrap personalizadas](#) para a personalização, já que os ganchos de pós-instalação geralmente são mais fáceis de depurar e mais portáteis entre as versões do AWS ParallelCluster.

## Etapas

1. Identifique o diretório de trabalho do livro de receitas do AWS ParallelCluster no qual o código do [Livro de receitas do AWS ParallelCluster](#) foi clonado.

```
_cookbookDir=<path to cookbook>
```

2. Detecte a versão atual do livro de receitas do AWS ParallelCluster.

```
_version=$(grep version ${_cookbookDir}/metadata.rb|awk '{print $2}' | tr -d \')
```

3. Crie um arquivo do livro de receitas do AWS ParallelCluster e calcule o 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 a seguinte variável ao arquivo de configuração do AWS ParallelCluster na [seção \[cluster\]](#).

```
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 do AWS ParallelCluster versão 2.6.1, a maioria das receitas de instalação é ignorada por padrão quando os nós são iniciados para melhorar os tempos de inicialização. Para ignorar a maioria das receitas 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](#).

## Configuração de um pacote de nós personalizado do AWS ParallelCluster

**Warning**

Veja a seguir instruções para usar uma versão personalizada do pacote de nós do AWS ParallelCluster. Esse é um método avançado de personalização do AWS ParallelCluster, com possíveis problemas que podem ser difíceis de depurar. É altamente recomendável pela equipe do AWS ParallelCluster o uso dos scripts em [Ações de bootstrap personalizadas](#) para a personalização, já que os ganchos de pós-instalação geralmente são mais fáceis de depurar e mais portáteis entre as versões do AWS ParallelCluster.

### Etapas

1. Identifique o diretório de trabalho do nó do AWS ParallelCluster onde o código do nó do AWS ParallelCluster foi clonado.

```
_nodeDir=<path to node package>
```

2. Detecte a versão atual do nó do AWS ParallelCluster.

```
_version=$(grep "version = \" ${_nodeDir}/setup.py |awk '{print $3}' | tr -d \"")
```

3. Crie um arquivo do nó do AWS ParallelCluster.

```
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 seguinte variável ao arquivo de configuração do AWS ParallelCluster na [seção \[cluster\]](#).

```
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 do AWS ParallelCluster versão 2.6.1, a maioria das receitas de instalação é ignorada por padrão quando os nós são iniciados para melhorar os tempos de inicialização. Para ignorar a maioria das receitas 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](#)
- [Solução de problemas de implantação de pilha](#)
- [Solução de problemas em clusters em modo de várias filas](#)
- [Solução de problemas em clusters em modo de fila única](#)
- [Grupos de posicionamento e problemas de execução de instâncias](#)
- [Diretórios que não podem ser substituídos](#)
- [Solução de problemas na Amazon DCV](#)
- [Solução de problemas em clusters com integração AWS Batch](#)
- [Solução de problemas quando um recurso não é criado](#)
- [Solução de problemas IAM de tamanho da política](#)
- [Suporte adicional](#)

## 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, 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 Slurm agendador de tarefas. Para obter mais informações sobre o modo de várias filas, consulte [Modo de fila múltipla](#).

## Tópicos

- [Logs de chaves](#)
- [Solução de problemas de inicialização do nó](#)
- [Solução de problemas inesperados de substituições e encerramentos de nós](#)
- [Substituindo, encerrando ou desligando instâncias e nós problemáticos](#)
- [Solucionando outros problemas conhecidos de nós e trabalhos](#)

## 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 AWS 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 por meio do 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`

Este é o Slurm registro do daemon de controle. AWS ParallelCluster não toma decisões de escalabilidade. Em vez disso, ele apenas tenta lançar recursos para satisfazer o Slurm requisitos. 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 Slurm log do daemon de computação. 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 ligar `ResumeProgram` com o nó.)
- Observe que permissões incorretas em `ResumeProgram` podem fazer com que `ResumeProgram` falhe silenciosamente. Se você estiver usando um personalizado AMI com modificações na `ResumeProgram` configuração, verifique se ele `ResumeProgram` é de propriedade do `slurm` usuário e tem a permissão `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 elas não foram iniciadas, deve haver uma mensagem de erro clara detalhando a falha na inicialização.

- 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 um personalizado AMI com modificação para Slurm configuração, então pode haver uma Slurm erro relacionado que impede que o nó de computação se junte 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 EC2 relacionado à Amazon, deve haver uma mensagem informativa detalhando o problema EC2 relacionado à Amazon que exigiu a substituição.

- Se o nó `clustermgtd` não foi encerrado, primeiro verifique se essa era uma rescisão esperada pela AmazonEC2, mais especificamente uma rescisão 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 na 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 alugar Slurm crie 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. Todo encerramento da instância e restauração de `NodeAddrs` são feitos por `clustermgtd`. Slurm coloca os nós de volta em um `POWER_SAVING` estado posterior `SuspendTimeout` automaticamente.

## 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 apenas inicia, encerra ou mantém recursos de acordo com Slurm 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 suporta o uso de 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 SGE, Torque, ou Slurm agendadores de trabalho.
- Lançado usando a AWS ParallelCluster versão 2.9.0 ou posterior e SGE ou Torque agendadores de trabalho.

### Tópicos

- [Logs de chaves](#)
- [Solução de problemas de falha nas operações de inicialização e junção](#)
- [Solucionar problemas de escala](#)
- [Solução de outros problemas relacionados ao cluster](#)

## 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 registro do cliente CINC (chef). Ele contém todos os comandos que foram executados 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 `clustermgtd` registro para Slurm agendadores. `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 `jobwatcher` registro para SGE e Torque agendadores. `jobwatcher` monitora a fila do agendador e atualiza o Auto Scaling Group. É útil para solucionar problemas relacionados com aumentar a escala verticalmente dos nós.

```
/var/log/sqswatcher
```

Este é o `sqswatcher` registro para SGE e Torque agendadores. `sqswatcher` processa o evento Instance Ready 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 `nodewatcher` registro. `nodewatcherdaemons` que são executados em cada nó de computação ao usar SGE e Torque agendadores. 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, está usando a AWS ParallelCluster versão v2.9.0 ou posterior, está usando Slurme, para evitar esse problema, você deve atualizar e mudar para um cluster compatível com o modo de várias filas. Para fazer isso, execute [pcluster-config convert](#).

Para clusters de escala ultralarga, talvez seja necessário um ajuste adicional em seu sistema. Para obter mais informações, entre em contato AWS Support.

## 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 obter mais informações, consulte [Solução de problemas de lançamento de instâncias](#) e [funções e limitações de grupos de posicionamento](#) no Guia EC2 do usuário da Amazon

## 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 inicial padrão do usuário (`/home/ec2_user` no Amazon Linux, `/home/centos` em CentOS, e assim `/home/ubuntu` por diante Ubuntu).

`/opt/intel`

Isso inclui IntelMPI, Intel Parallel Studio e arquivos relacionados.

`/opt/sgc`

### Note

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

Isso inclui Son of Grid Engine e arquivos relacionados. (Condicional, somente se [scheduler](#) = `sgc`.)

`/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 suporta o uso de SGE ou Torque agendadores.

Isso inclui Torque Resource Manager e arquivos relacionados. (Condicional, somente se [scheduler](#) = torque.)

## Solução de problemas na Amazon DCV

Tópicos

- [Registros para a Amazon DCV](#)
- [Memória do tipo de DCV instância Amazon](#)
- [DCVProblemas com o Ubuntu Amazon](#)

### Registros para a Amazon DCV

Os registros da Amazon DCV são gravados em arquivos no `/var/log/dcv/` diretório. A revisão desses logs pode ajudar a solucionar problemas.

### Memória do tipo de DCV instância Amazon

O tipo de instância deve ter pelo menos 1,7 gibibyte (GiB) de para executar a Amazon. RAM DCV Nano e micro os tipos de instância não têm memória suficiente para executar a AmazonDCV.

### DCVProblemas com o Ubuntu Amazon

Ao executar o Gnome Terminal em uma DCV sessão no Ubuntu, você pode não ter acesso automático ao ambiente do usuário que o AWS ParallelCluster disponibiliza 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 integração AWS Batch

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 mostrada na resposta do comando anterior, deverá usar tipos de instância que não excedam seu CPU limite atual de v ou solicitar mais CPU capacidade v.

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 AWS Management Console 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 IAM de tamanho da política

Consulte [AWS STS cotas, requisitos de nome e limites de caracteres](#) para verificar as cotas nas políticas gerenciadas associadas às funções. IAM 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 uma IAM função, crie funções adicionais e distribua as políticas entre elas para atender à 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 AWS Support ou abra um [novo GitHub problema](#).

# Política de suporte do AWS ParallelCluster

O AWS ParallelCluster suporta várias versões ao mesmo tempo. Cada versão do AWS ParallelCluster 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 versão `major.minor.patch`. 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.

Versões do AWS ParallelCluster	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 para com a nuvem na AWS é a nossa maior prioridade. Como cliente da AWS, você se contará com um datacenter e uma arquitetura de rede criados para atender aos requisitos das organizações com as maiores exigências de segurança.

A segurança é uma responsabilidade compartilhada entre a AWS e você. O [modelo de responsabilidade compartilhada](#) descreve isso como segurança da nuvem e segurança na nuvem:

- Segurança da nuvem: a AWS é responsável pela proteção da infraestrutura que executa produtos da AWS na Nuvem AWS. A AWS também fornece serviços que podem ser usados com segurança. Auditores de terceiros testam e verificam regularmente a eficácia da nossa segurança como parte dos [Programas de conformidade da AWS](#). Para saber mais sobre os programas de conformidade que se aplicam ao AWS ParallelCluster, consulte [Serviços da AWS em escopo por programa de conformidade](#)
- Segurança na nuvem: sua responsabilidade é determinada pelo serviço ou serviços específicos da AWS 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 aplicar o modelo de responsabilidade compartilhada ao usar o AWS ParallelCluster. Os tópicos a seguir mostram como configurar o AWS ParallelCluster para atender aos seus objetivos de segurança e conformidade. Saiba também como usar o AWS ParallelCluster de uma maneira que ajude você a monitorar e proteger os recursos da AWS.

## Tópicos

- [Informações de segurança para serviços usados pelo AWS ParallelCluster](#)
- [Proteção de dados em AWS ParallelCluster](#)
- [Identity and Access Management para o AWS ParallelCluster](#)
- [Validação de conformidade do AWS ParallelCluster](#)
- [Impor uma versão mínima do TLS 1.2](#)

# Informações de segurança para serviços usados pelo AWS ParallelCluster

- [Segurança no Amazon EC2](#)
- [Segurança no Amazon API Gateway](#)
- [Segurança em AWS Batch](#)
- [Segurança em AWS CloudFormation](#)
- [Segurança no 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 para Lustre](#)
- [Segurança na 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 \(para o AWS ParallelCluster versão 2.x.\)](#)
- [Segurança no Amazon S3](#)
- [Segurança no Amazon VPC](#)

## Proteção de dados em AWS ParallelCluster

A ferramenta AWS modelo de [responsabilidade compartilhada modelo](#) se aplica à proteção de dados em AWS ParallelCluster. Conforme descrito neste modelo, AWS é responsável por proteger a infraestrutura global que executa todas as 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 você usa. Para obter

mais informações sobre privacidade de dados, consulte [Privacidade de dados FAQ](#). Para obter informações sobre proteção de dados na Europa, consulte o [AWS Modelo de responsabilidade compartilhada e postagem no GDPR](#) blog sobre o AWS Blog de segurança.

Para fins de proteção de dados, recomendamos que você proteja Conta da AWS credenciais e configure usuários individuais com AWS IAM Identity Center 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 a autenticação multifator (MFA) com cada conta.
- Use SSL/TLS para se comunicar com AWS recursos. Exigimos TLS 1,2 e recomendamos TLS 1,3.
- Configure API e registre as atividades do usuário com AWS CloudTrail. Para obter informações sobre o uso de CloudTrail trilhas para capturar AWS atividades, consulte [Trabalhando com CloudTrail trilhas](#) no AWS CloudTrail Guia do usuário.
- Use AWS soluções de criptografia, junto com todos os controles de segurança padrão dentro Serviços da AWS.
- Use serviços gerenciados de segurança avançada, como o Amazon Macie, que ajuda a localizar e proteger dados sigilosos armazenados no Amazon S3.
- Se você precisar de FIPS 140-3 módulos criptográficos validados ao acessar AWS por meio de uma interface de linha de comando ou uma API, use um FIPS endpoint. Para obter mais informações sobre os FIPS endpoints disponíveis, consulte [Federal Information Processing Standard \(FIPS\) 140-3](#).

É altamente recomendável que nunca sejam colocadas informações de identificação confidenciais, como endereços de e-mail dos seus clientes, em marcações ou campos de formato livre, como um campo Nome. Isso inclui quando você trabalha com AWS ParallelCluster ou outro Serviços da AWS usando o console API, AWS CLI, ou AWS SDKs. Quaisquer dados inseridos em tags ou 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 altamente recomendável que você não inclua informações de credenciais no URL para validar sua solicitação para esse servidor.

## Criptografia de dados

Um atributo 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 o AWS serviços em nome do usuário.

Para dados nos nós do cluster, os dados podem ser criptografados em repouso.

Para EBS volumes da Amazon, a criptografia é configurada usando as [ebs\\_kms\\_key\\_id](#) configurações na [\[ebs\]seção](#) para AWS ParallelCluster versão 2.x.) Para obter mais informações, consulte a [EBScriptografia da Amazon](#) no Guia EC2 do usuário da Amazon.

Para EFS volumes da Amazon, a criptografia é configurada usando as [efs\\_kms\\_key\\_id](#) configurações [encrypted](#) e na [\[efs\]seção](#) em 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.

FSxPara 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 obter mais informações, consulte [Criptografia em repouso](#) no Guia EC2 do usuário da Amazon.

Se você usa AWS ParallelCluster para invocar um AWS serviço que transmite dados do cliente para seu computador local para armazenamento e, em seguida, 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 AWS os endpoints de serviço são criptografados enviando tudo por meio de uma TLS conexão HTTPS /. O tráfego entre os nós no cluster pode ser criptografado automaticamente, dependendo

dos tipos de instância selecionados. Para obter mais informações, consulte [Criptografia em trânsito](#) no Guia EC2 do usuário da Amazon.

## Consulte também

- [Proteção de dados na Amazon EC2](#)
- [Proteção de dados no EC2 Image Builder](#)
- [Proteção de dados em AWS CloudFormation](#)
- [Proteção de dados na Amazon EFS](#)
- [Proteção de dados no Amazon S3](#)
- [Proteção de dados no FSx Lustre](#)

## Identity and Access Management para o AWS ParallelCluster

O AWS ParallelCluster usa funções para acessar seus recursos da AWS e os serviços dela. As políticas de instância e usuário usadas pelo AWS ParallelCluster para conceder permissões estão documentadas em [AWS 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 da AWS exija uma senha para acessar o console de um serviço da , esse mesmo usuário do exige um par de chaves de acesso para executar as mesmas operações usando o 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 não criptografado e não são criptografadas.

- O arquivo `$HOME/.aws/credentials` armazena credenciais de longo prazo necessárias para acessar os recursos da AWS. Isso inclui o ID de chave de acesso e a chave de acesso secreta.
- As credenciais de curto prazo, como as de funções que você assume, ou que são para serviços do AWS IAM Identity Center, também são armazenadas nas pastas `$HOME/.aws/cli/cache` e `$HOME/.aws/sso/cache`, 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 do AWS ParallelCluster

Audidores externos avaliam a segurança e a conformidade dos serviços da AWS como parte de vários programas de conformidade da AWS. Usar o AWS ParallelCluster para acessar um serviço não altera a conformidade desse serviço.

Para ver a lista de serviços da AWS no escopo de programas de compatibilidade específicos, consulte [Serviços da AWS no escopo por programa de compatibilidade](#). Para obter informações gerais, consulte [Programas de conformidade da AWS](#).

É possível baixar relatórios de auditoria externa usando o AWS Artifact. Para obter mais informações, consulte [Fazer download de relatórios no AWS Artifact](#).

Sua responsabilidade de conformidade ao usar o AWS ParallelCluster é determinada pela confidencialidade dos seus dados, pelos objetivos de conformidade da sua empresa e pelos regulamentos e leis aplicáveis. A AWS fornece os seguintes recursos para ajudar com a conformidade:

- [Guias de início rápido de segurança e compatibilidade](#): esses guias de implantação apresentam considerações de arquitetura e etapas para a implantação de ambientes básicos focados na segurança e na compatibilidade na AWS.
- [Arquitetura para segurança e conformidade com a HIPAA na Amazon Web Services AWS Whitepaper](#) – este whitepaper descreve como as empresas podem utilizar a AWS para criar aplicações em conformidade com a HIPAA.
- [Recursos de compatibilidade da AWS](#): essa coleção de manuais e guias pode ser válida para seu setor e local.
- [Avaliar recursos com regras](#) no Guia do desenvolvedor do AWS Config: o serviço AWS Config avalia como as configurações de recursos estão em compatibilidade com práticas internas, diretrizes do setor e regulamentos.

- [AWS Security Hub](#): esse serviço da AWS fornece uma visão abrangente do estado de sua segurança na AWS que ajuda você a conferir sua conformidade com padrões e práticas recomendadas de segurança do setor.

## Impor uma versão mínima do TLS 1.2

Para aumentar a segurança ao se comunicar com serviços da AWS, é necessário configurar seu AWS ParallelCluster para usar o TLS 1.2 ou posterior. Quando você usa o AWS ParallelCluster, o Python é usado para definir a versão do TLS.

Para garantir que o AWS ParallelCluster não use uma versão do TLS anterior ao TLS 1.2, talvez seja necessário recompilar o OpenSSL para impor esse mínimo e 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 definirá como padrão 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 o AWS ParallelCluster não negocie nada anterior ao TLS 1.2, é necessário 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
<a href="#">Versão somente para documentação</a>	<p>AWS ParallelCluster guia do usuário específico da versão 2 publicado.</p> <p>Versão somente para documentação:</p> <ul style="list-style-type: none"><li>• AWS ParallelCluster a versão 2 tem seu próprio guia de usuário separado.</li></ul>	17 de julho de 2023
<a href="#">AWS ParallelCluster versão 2.11.9 lançada</a>	<p>AWS ParallelCluster versão 2.11.9 lançada.</p> <p>Correções de erros:</p> <ul style="list-style-type: none"><li>• Evite a substituição de sistemas de arquivos gerenciados FSx pelo Lustre e a perda de dados em atualizações de cluster que incluam alterações no <code>vpc_security_group_id</code></li></ul>	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.

14 de novembro de 2022

Alterações:

- Atualize a MPI Biblioteca Intel para a versão 2021, atualização 6 (atualizada da versão 2021, atualização 4). Para obter mais informações, consulte [Intel® MPI Library 2021 Update 6](#).
- Atualize o EFA instalador 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 )
  - AbertoMPI: openmpi40-aws-4.1.4-3 (de openmpi40-aws-4.1.1-2 )

- 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.

13 de maio de 2022

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

[AWS ParallelCluster versão  
2.11.6 lançada](#)

AWS ParallelCluster versão  
2.11.6 lançada.

19 de abril de 2022

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

[AWS ParallelCluster versão 2.11.5 lançada](#)

AWS ParallelCluster versão 2.11.5 lançada.

1º de março de 2022

Melhorias:

- Adicione suporte NEW\_CHANGED\_DELETE como valor FSx para a AutoImportPolicy opção Lustre.
- Remova o suporte SGE e os agendadores de torque.
- Desative o serviço log4j-cve-2021-44228-hotpatch no Amazon Linux para evitar uma possível degradação do desempenho.

Alterações:

- Atualize NVIDIA o driver para a versão 470.103.01 (de470.82.01).
- Atualize o NVIDIA Fabric Manager para a versão 470.103.01 (de470.82.01).
- Atualize a CUDA bibliotec a para a versão 11.4.4 (de11.4.3).
- [Intel MPI](#) atualizada para a versão 2021, atualização 4 (atualizada a partir da

versão 2019, atualização 8).

Para obter mais informações, consulte [Intel® MPI Library 2021 Update 4](#).

- Extensão do tempo limite de criação do nó principal para uma hora.

Correções de erros:

- Corrija DCV a conexão por meio de navegadores.
- Corrija YAML as aspas 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.

20 de dezembro de 2021

As alterações do incluem:

- CentOS 8 suportes removidos. CentOS 8 chega ao fim da vida útil (EOL) em 31 de dezembro de 2021.
- Upgrade Slurm Workload Manager para a versão 20.11.8.
- Atualização do Cinc Client para 17.2.29.
- [A Amazon DCV](#) foi atualizada para a Amazon DCV 2021.2-11190. Para obter mais informações, consulte [DCV2021.2-11190— 11 de outubro de 2021 no Amazon Administrator Guide](#). DCV
- Atualize NVIDIA o driver para a versão 470.82.01 (de 460.73.01).
- Atualize a CUDA bibliotec a para a versão 11.4.3 (de 11.3.0).
- Atualize o NVIDIA Fabric Manager para 470.82.01 o.
- Desabilitação da atualização do pacote no momento do lançamento da instância no Amazon Linux 2.

- Desativar a atualização autônoma de pacotes em Ubuntu e Amazon Linux 2.
- Instale a versão Python 3 dos scripts [AWS CloudFormation auxiliares](#) em CentOS 7 e Ubuntu 18.04. (Eles já foram usados no Amazon Linux 2 e Ubuntu 20.04.)

As correções incluem:

- Desabilitação da atualização do parâmetro [ec2\\_iam\\_role](#).
- Corrija a `CpuOptions` configuração no modelo de lançamento para T2 instâncias.

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 2.11.3 lançada](#)

AWS ParallelCluster versão 2.11.3 lançada.

3 de novembro de 2021

- [pcluster createami](#)  
\_Falha na correção devido a Son of Grid Engine fontes não estão disponíveis em `emarc.liv.ac.uk` .

Atualização do instalador [Elastic Fabric Adapter](#) para 1.14.1 (de 1.13.0)

- EFAconfiguração: `efa-config-1.9-1` (de `efa-config-1.9` )
- EFAperfil: `efa-profile-1.5-1` (sem alteração)
- EFAMódulo do kernel: `efa-1.14.2` (de `efa-1.13.0` )
- RDMAnúcleo: `rdma-core-37.0` (de `rdma-core-35.0amzn` )
- Libfabric: `libfabric-1.13.2` (de `libfabric-1.13.0amzn1.0` )
- AbertoMPI: `openmpi40-aws-4.1.1-2` (sem alteração)

GPUDirectRDMA está sempre habilitado 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.

27 de agosto de 2021

As alterações do incluem:

- Não instale EFA com GPUDirect RDMA (GDR) habilitado no momento da inicialização, se EFA estiver instalado na baseAMI.
- Bloqueie a versão do `nvidia-fabricmanager` pacote para permanecer sincronizada com a versão do NVIDIA driver 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:
  - EFAconfiguração: `efa-config-1.9` (sem alteração)
  - EFAperfil: `efa-profile-1.5-1` (sem alteração)
  - EFAMódulo do kernel: `efa-1.13.0` (sem alteração)
  - RDMAnúcleo: `rdma-core-35.0amzn`

- (derdma-core-32.1amzn )
- Libfabric: libfabric-1.13.0amzn1.0 (de libfabric-1.11.2amzn1.1 )
- AbertoMPI: openmpi40-aws-4.1.1-2 (sem alteração)
- Ao usar um pacote personalizado AMI com um EFA pacote pré-instalado, nenhuma alteração é feita EFA no momento da inicialização do nó. A implantação EFA do pacote original é preservada.

Para obter mais detalhes sobre as mudanças, consulte os CHANGELOG arquivos do [aws-parallelcluster](#) e os pacotes em. [aws-parallelcluster-cookbook](#) GitHub

## [AWS ParallelCluster versão 2.11.1 lançada](#)

AWS ParallelCluster versão 2.11.1 lançada.

23 de julho de 2021

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:
  - EFAconfiguração: `efa-config-1.9` (`deefa-config-1.8-1` )
  - EFAperfil: `efa-profile-1.5-1` (sem alteração)
  - EFAMódulo do kernel: `efa-1.13.0` (`deefa-1.12.3` )
  - RDMAnúcleo: `rdma-core-32.1amzn` (sem alteração)
  - Libfabric: `libfabric-1.11.2amzn1.1` (sem alteração)
  - AbertoMPI: `openmpi40-aws-4.1.1-2` (sem alteração)

- Tente novamente as instalações do `aws-parallelcluster` pacote no nó principal ao usar AWS Batch como agendador.
- Evite falhas ao construir SGE em um tipo de instância com mais de 31vCPUs.
- 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 CHANGELOG arquivos 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.

1.º de julho de 2021

As alterações do incluem:

- Suporte adicional para Ubuntu 20.04 (ubuntu2004) e suporte removido 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 ter mais informações, consulte [compute\\_root\\_volume\\_size](#) e [master\\_root\\_volume\\_size](#).
- Instalador do [Elastic Fabric Adapter](#) atualizado para 1.12.2:
  - EFAconfiguração: efa-config-1.8-1 (deefa-config-1.7)
  - EFAperfil: efa-profile-1.5-1 (deefa-profile-1.4)

- EFA Módulo do kernel: efa-1.12.3 (de efa-1.10.2 )
- RDMA núcleo: rdma-core-32.1amzn (de rdma-core-31.2amzn )
- Libfabric: libfabric-1.11.2amzn1.1 (de libfabric-1.11.1amzn1.0 )
- Aberto MPI: openmpi40-aws-4.1.1-2 (de openmpi40-aws-4.1.0 )
- Atualizado Slurm para a versão 20.11.7 (de 20.02.7).
- Instale o SSM Agente em centos7 centos8 e. (O SSM agente está pré-instalado em alinux2ubuntu1804 , ubuntu2004 e.)
- SGE: sempre use 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.

- Atualize NVIDIA o driver para a versão 460.73.01 (de 450.80.02 ).
- Atualize a CUDA bibliotec a para a versão 11.3.0 (de 11.0).
- Atualize o NVIDIA Fabric Manager para nvidia-fabricmanager-460 o.
- 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 alterações, consulte os CHANGELOG arquivos 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.

15 de maio de 2021

As alterações do incluem:

- Atualizado Slurm para a versão `20.02.7` (de `20.02.4`).

Para obter mais detalhes sobre as mudanças, consulte o CHANGELOG arquivo do pacote [aws-parallelcluster](#) em. GitHub

## [AWS ParallelCluster versão 2.10.3 lançada](#)

AWS ParallelCluster versão 2.10.3 lançada.

18 de março de 2021

As alterações do incluem:

- Suporte adicional 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:
  - EFAconfiguração: efa-config-1.7 (sem alteração)
  - EFAperfil: efa-profile-1.4 (deefa-profile-1.3 )
  - EFAMódulo do kernel: efa-1.10.2 (sem alteração)
  - RDMAnúcleo: rdma-core-31.2amzn (sem alteração)
  - Libfabric: libfabric-1.11.1amzn1.0 (sem alteração)
  - AbertoMPI: openmpi40-aws-4.1.0 (sem alteração)

Para obter mais detalhes sobre as mudanças, consulte o CHANGELOG arquivo do pacote [aws-parallelcluster](#) em GitHub

## [AWS ParallelCluster versão 2.10.2 lançada](#)

AWS ParallelCluster versão 2.10.2 lançada.

2 de março de 2021

As alterações do incluem:

- Melhore a validação da configuração do cluster para usar o destino do cluster AMI ao invocar a EC2 [RunInstances](#)API operação 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.
- Corrija `enable_efa` ao usar centos8 com o Slurm tipos de agendador ou 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 CHANGELOG arquivo do pacote [aws-parallelcluster](#) em. GitHub

## [AWS ParallelCluster versão 2.10.1 lançada](#)

AWS ParallelCluster versão 2.10.1 lançada.

22 de dezembro de 2020

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:
  - FSxpara 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).
  - A 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 os tipos Amazon EBS io2 e de gp3 volume. Para mais informações, consulte a [seção \[ebs\]](#) e a [seção \[raid\]](#).

- 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`, `ubuntu1804` ). Para obter mais informações, consulte [Bibliotecas de desempenho do Arm](#).
- [Intel MPI](#) atualizada para a versão 2019, atualização 8 (atualizada a partir da versão 2019, atualização 7). Para obter mais informações, consulte [Intel® MPI Library 2019 Update 8](#).
- A chamada de AWS CloudFormation `DescribeStacks` API operação foi removida do ponto de entrada do AWS Batch Docker para encerrar as falhas de trabalho causadas pela limitação de. AWS CloudFormation
- Melhorou as chamadas para a chamada de EC2 `DescribeInstanceTy`

pes API operação da Amazon ao validar uma configuração de cluster.

- As imagens Docker do Amazon Linux 2 são extraídas do Amazon ECR Public ao criar a imagem do Docker para o awsbatch agendador.
- 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:
  - EFAconfiguração: efa-config-1.7 (deefa-config-1.5 )
  - EFAperfil: efa-profile-1.3 (deefa-profile-1.1 )
  - EFAMódulo do kernel: efa-1.10.2 (sem alteração)
  - RDMAnúcleo: rdma-core-31.2amzn

- (derdma-core-31.amzn0 )
- Libfabric: libfabric-1.11.1amzn1.0 (de libfabric-1.10.1amzn1.1 )
- AbertoMPI: openmpi40-aws-4.1.0 (deopenmpi40-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.
- [A Amazon DCV](#) foi atualizada para a Amazon DCV 2020.2-9662. Para obter mais informações, consulte [DCV2020.2-9662— 04 de dezembro de 2020 no Amazon Administrator Guide](#). DCV
- A Intel MPI e HPC os pacotes do AWS ParallelCluster são retirados do

Amazon S3. Eles não são mais retirados dos repositórios Intel yum.

- Alterou o padrão `systemd` `runlevel` to `multi-use` `r.target` on all OSs durante a criação do oficial AWS ParallelCluster AMIs. O nível de execução é definido como `graphical.target` no nó principal somente quando 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.
- Aumente o número máximo de tentativas ao se registrar Slurm nós no Amazon Route 53.

Para obter mais detalhes sobre as alterações, consulte os CHANGELOG arquivos 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.

18 de novembro de 2020

As alterações do incluem:

- Suporte adicional para CentOS 8 no total Regiões da AWS (fora das regiões da AWS 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 on EFA usando a nova [enable\\_efa\\_gdr](#) configuração.
- 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 HDD baseados no Amazon FSx for [storage\\_type](#) Lustre usando as [drive\\_cache\\_type](#) configurações e.

- 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](#).
  - `--post-install` Parâmetro adicionado para usar um script de pós-instalação ao criar umAMI.
  - Foi adicionada uma etapa de validação para falhar ao usar uma base AMI criada por uma versão diferente do AWS ParallelCluster.
  - Foi adicionada uma etapa de validação para falhar quando o sistema operacional selecionado

- for diferente do sistema operacional na baseAMI.
- Foi adicionado suporte para o uso de uma AWS ParallelCluster baseAMI.
  - 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 `True`, `slurm_resume` será bem-sucedido somente se todas as instâncias exigidas por todos os trabalhos pendentes em Slurm estará disponível. 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:

- EFAconfiguração: efa-config-1.5 (deefa-config-1.4 )
- EFAperfil: efa-profile-1.1 (deefa-profile-1.0.0 )
- EFAMódulo do kernel: efa-1.10.2 (deefa-1.6.0 )
- RDMAnúcleo: rdma-core-31.amzn0 (derdma-core-28.amzn0 )
- Libfabric: libfabric-1.11.1amzn1.0 (de libfabric-1.10.1amzn1.1 )
- AbertoMPI: openmpi40-aws-4.0.5 (deopenmpi40-aws-4.0.3 )
- Nas AWS GovCloud (US) Regiões, habilite o suporte para Amazon DCV AWS Batch e.
- Nas regiões da AWS China, ative o suporte para o Amazon FSx for Lustre.
- Atualize NVIDIA o driver para a versão 450.80.02 (de 450.51.05).
- Instale o NVIDIA Fabric Manager para habilitar

NVIDIA NVSwitch em plataformas compatíveis.

- Removido o padrão Região da AWS deus-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_regio`  
`n_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 alterações, consulte os CHANGELOG arquivos 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.

11 de setembro de 2020

As alterações do incluem:

- Foi adicionado suporte para várias filas e vários tipos de instância na frota de computação quando usado com 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 DNS resolução de nós de computação quando o Slurm o agendador é usado. Para obter mais informações, consulte [Modo de fila múltipla](#).
- Foi adicionado suporte para a [Amazon DCV em instâncias](#) baseadas em AWS Graviton baseadas em ARM.
- Foi adicionado suporte para desativar o hyperthreading em tipos de instância que não oferecem suporte a CPU opções em modelos de execução (por exemplo, tipos de \*.metal instância ).

- Foi adicionado suporte para NFS 4 para sistemas de arquivos compartilhados a partir do nó principal.
- Foi removida a dependência do [cfn-init](#) ao inicializar os nós de computação para evitar a limitação quando um grande número de nós ingressa no cluster. AWS CloudFormation
- Instalador do [Elastic Fabric Adapter](#) atualizado para 1.9.5:
  - EFAconfiguração: efa-config-1.4 (de efa-config-1.3 )
  - EFAperfil: efa-profile-1.0.0 (novo)
  - Kernel module: efa-1.6.0 (sem alteração)
  - RDMA núcleo: rdma-core-28.amzn0 (sem alteração)
  - Libfabric: libfabric-1.10.1amzn1.1 (sem alteração)
  - AbertoMPI: openmpi40-aws-4.0.3 (sem alteração)
- Atualizado Slurm para a versão 20.02.4 (de 19.05.5).

- [A Amazon DCV](#) foi atualizada para a Amazon DCV 2020.1-9012. Para obter mais informações, consulte as notas de versão de [DCV2020.1-9012 a 24 de agosto de 2020 no Amazon Administrator Guide](#). DCV
- Ao montar NFS unidades compartilhadas, use 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 alterações, consulte os CHANGELOG arquivos 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.

4 de agosto de 2020

As alterações do incluem:

- Desative o bloqueio de tela DCV nas sessões da Amazon 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 alterações, consulte os CHANGELOG arquivos do [aws-parallelcluster](#) e os pacotes em. [aws-parallelcluster-cookbookaws-parallelcluster-node](#) GitHub

## [AWS ParallelCluster versão 2.8.0 lançada](#)

AWS ParallelCluster versão 2.8.0 lançada.

23 de julho de 2020

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 obter mais informações, consulte [automatic\\_backup\\_retention\\_days](#) , [copy\\_tags\\_to\\_backups](#) , [daily\\_automatic\\_backup\\_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 [Utilizar 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 )

- RDMAcore: `rdma-core-28.amzn0`  
(atualizado de `rdma-core-25.0` )
- Libfabric: `libfabric-1.10.1amzn1.1`  
(atualizado de `libfabric-aws-1.9.0amzn1.1` )
- AbertoMPI: `openmpi40-aws-4.0.3` (sem alteração)
- Atualize o NVIDIA driver para a versão 440.95.01 da Tesla em CentOS 6 e versão 450.51.05 em todas as outras distribuições.
- Atualize a CUDA biblioteca para a versão 11.0 em todas as distribuições, exceto CentOS 6.

Para obter mais detalhes sobre as alterações, consulte os CHANGELOG arquivos 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.

19 de maio de 2020

As alterações do incluem:

- [base\\_os](#) agora é um parâmetro obrigatório.
- [scheduler](#) agora é um parâmetro obrigatório.
- [A Amazon DCV](#) foi atualizada para a Amazon DCV 2020.0. Para obter mais informações, consulte [a Amazon DCV lança a versão 2020.0 com som surround 7.1](#) e suporte para caneta.

[Intel MPI](#) atualizada para a versão 2019, atualização 7 (atualizada a partir da versão 2019, atualização 6). Para obter mais informações, 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)
- RDMA núcleo: rdma-core-25.0 (sem alteração)

- Libfabric: libfabric-aws-1.9.0amzn1.1 (sem alteração)
- AbertoMPI: openmpi40-aws-4.0.3 (atualiza do deopenmpi40-aws-4.0.2 )
- Upgrade CentOS 7 AMI para a versão 7.8-2003 (atualizada de 7.7-1908). Para ter mais informações, consulte [CentOS-7 \(2003\) Notas](#) de lançamento.

### [AWS ParallelCluster versão 2.6.1 lançada](#)

AWS ParallelCluster versão 2.6.1 lançada.

17 de abril de 2020

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](#).

[AWS ParallelCluster versão 2.6.0 lançada](#)

AWS ParallelCluster versão 2.6.0 lançada.

27 de fevereiro de 2020

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 informações, consulte [Integração com Amazon CloudWatch Logs](#).
- Foi adicionado suporte para novos tipos de implantação do Amazon FSx for SCRATCH\_2 Lustre PERSISTENT\_1 e Support FSx for Lustre on Ubuntu 18.04 e Ubuntu 16.04. Para obter mais informações, consulte [fsx](#).
- Foi adicionado suporte para a Amazon DCV em Ubuntu 18.04. Para obter mais informações, consulte [Conecte-se ao nó principal por meio da Amazon DCV](#).

[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

---

<a href="#">AWS ParallelCluster introduz suporte para Intel MPI</a>	AWS ParallelCluster a versão 2.4.1 introduz suporte para Intel. MPI	29 de julho de 2019
<a href="#">AWS ParallelCluster introduz suporte para EFA</a>	AWS ParallelCluster a versão 2.4.0 introduz suporte para Elastic Fabric Adapter (EFA).	11 de junho de 2019
<a href="#">AWS ParallelCluster documentação lançada no site de AWS documentação</a>	A AWS ParallelCluster documentação agora está disponível em 10 idiomas e em ambos os HTML PDF formatos.	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.