



AWS ParallelCluster Panduan Pengguna (v2)

AWS ParallelCluster



AWS ParallelCluster: AWS ParallelCluster Panduan Pengguna (v2)

Copyright © 2025 Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Merek dagang dan tampilan dagang Amazon tidak boleh digunakan sehubungan dengan produk atau layanan apa pun yang bukan milik Amazon, dengan cara apa pun yang dapat menyebabkan kebingungan di antara pelanggan, atau dengan cara apa pun yang merendahkan atau mendiskreditkan Amazon. Semua merek dagang lain yang tidak dimiliki oleh Amazon merupakan hak milik masing-masing pemiliknya, yang mungkin atau tidak terafiliasi, terkait dengan, atau disponsori oleh Amazon.

Table of Contents

Apa itu AWS ParallelCluster	1
Harga	1
Menyiapkan AWS ParallelCluster	2
Instalasi AWS ParallelCluster	2
Instalasi AWS ParallelCluster di lingkungan virtual (disarankan)	2
Menginstal AWS ParallelCluster di lingkungan non-virtual menggunakan pip	3
Langkah-langkah yang harus diambil setelah instalasi	3
Instruksi terperinci untuk setiap lingkungan	4
Lingkungan virtual	4
Linux	6
macOS	10
Windows	13
Mengkonfigurasi AWS ParallelCluster	15
Praktik terbaik	24
Praktik terbaik: pemilihan tipe instans master	24
Praktik terbaik: kinerja jaringan	24
Praktik terbaik: peringatan anggaran	25
Praktik terbaik: memindahkan cluster ke versi AWS ParallelCluster minor atau patch baru	25
Pindah dari CfnCluster ke AWS ParallelCluster	26
Wilayah yang Didukung	28
Menggunakan AWS ParallelCluster	30
Konfigurasi jaringan	30
AWS ParallelCluster dalam satu subnet publik	31
AWS ParallelCluster menggunakan dua subnet	32
AWS ParallelCluster dalam satu subnet pribadi yang terhubung menggunakan AWS Direct Connect	33
AWS ParallelCluster dengan awsbatch scheduler	34
Tindakan Bootstrap Kustom	35
Konfigurasi	37
Pendapat	37
Contoh	37
Bekerja dengan Amazon S3	39
Contoh	39
Berkeja dengan Instans Spot	40

Skenario 1: Instans Spot tanpa pekerjaan yang berjalan terganggu	40
Skenario 2: Instance Spot yang menjalankan pekerjaan node tunggal terputus	40
Skenario 3: Instans Spot yang menjalankan pekerjaan multi-node terganggu	42
AWS Identity and Access Management peran dalam AWS ParallelCluster	44
Pengaturan default untuk pembuatan cluster	44
Menggunakan peran IAM yang ada untuk Amazon EC2	44
AWS ParallelCluster contoh contoh dan kebijakan pengguna	45
Penjadwal didukung oleh AWS ParallelCluster	86
Son of Grid Engine	87
Slurm Workload Manager	87
Torque Resource Manager	100
AWS Batch	100
Penandaan	108
CloudWatch Dasbor Amazon	111
Integrasi dengan Amazon CloudWatch Logs	113
Elastic Fabric Adapter	115
Solusi Pilihan Intel	116
Aktifkan Intel MPI	118
Spesifikasi Platform Intel HPC	119
Perpustakaan Kinerja Arm	120
Connect ke head node melalui Amazon DCV	121
Sertifikat Amazon DCV HTTPS	123
Perizinan Amazon DCV	123
Menggunakan <code>pcluster update</code>	123
Penambahan AMI dan penggantian EC2 instance	126
Pembaruan atau penggantian instance node kepala	127
Keterbatasan penyimpanan instans	127
Solusi keterbatasan penyimpanan instans	128
Berhenti dan mulai node kepala cluster	129
AWS ParallelCluster Perintah CLI	131
<code>pcluster</code>	131
Pendapat	131
Sub-perintah:	131
<code>pcluster configure</code>	132
<code>pcluster create</code>	133
<code>pcluster createami</code>	135

pcluster dcv	139
pcluster delete	141
pcluster instances	143
pcluster list	144
pcluster ssh	145
pcluster start	146
pcluster status	147
pcluster stop	148
pcluster update	149
pcluster version	151
pcluster-config	152
Argumen bernama	152
Konfigurasi	154
Tata Letak	155
Bagian [global]	155
cluster_template	155
update_check	156
sanity_check	156
Bagian [aws]	156
Bagian [aliases]	157
Bagian [cluster]	158
additional_cfn_template	160
additional_iam_policies	160
base_os	161
cluster_resource_bucket	163
cluster_type	164
compute_instance_type	164
compute_root_volume_size	165
custom_ami	165
cw_log_settings	166
dashboard_settings	167
dcv_settings	167
desired_vcpus	168
disable_cluster_dns	168
disable_hyperthreading	169
ebs_settings	170

ec2_iam_role	170
efs_settings	171
enable_efa	171
enable_efa_gdr	172
enable_intel_hpc_platform	172
encrypted_ephemeral	173
ephemeral_dir	173
extra_json	174
fsx_settings	174
iam_lambda_role	175
initial_queue_size	175
key_name	176
maintain_initial_size	176
master_instance_type	177
master_root_volume_size	178
max_queue_size	178
max_vcpus	179
min_vcpus	179
placement	180
placement_group	180
post_install	181
post_install_args	181
pre_install	182
pre_install_args	182
proxy_server	182
queue_settings	183
raid_settings	184
s3_read_resource	184
s3_read_write_resource	184
scaling_settings	185
scheduler	185
shared_dir	186
spot_bid_percentage	187
spot_price	187
tags	188
template_url	188

vpc_settings	189
Bagian [compute_resource]	189
initial_count	190
instance_type	190
max_count	191
min_count	191
spot_price	192
Bagian [cw_log]	192
enable	193
retention_days	193
Bagian [dashboard]	193
enable	194
Bagian [dcv]	194
access_from	195
enable	195
port	196
Bagian [ebs]	196
shared_dir	197
ebs_kms_key_id	198
ebs_snapshot_id	198
ebs_volume_id	198
encrypted	198
volume_iops	199
volume_size	200
volume_throughput	201
volume_type	201
Bagian [efs]	202
efs_fs_id	203
efs_kms_key_id	204
encrypted	204
performance_mode	205
provisioned_throughput	205
shared_dir	206
throughput_mode	206
Bagian [fsx]	206
auto_import_policy	208

automatic_backup_retention_days	209
copy_tags_to_backups	210
daily_automatic_backup_start_time	210
data_compression_type	211
deployment_type	211
drive_cache_type	213
export_path	213
fsx_backup_id	213
fsx_fs_id	214
fsx_kms_key_id	214
import_path	215
imported_file_chunk_size	215
per_unit_storage_throughput	216
shared_dir	216
storage_capacity	217
storage_type	218
weekly_maintenance_start_time	220
Bagian [queue]	220
compute_resource_settings	221
compute_type	221
disable_hyperthreading	222
enable_efa	222
enable_efa_gdr	223
placement_group	223
Bagian [raid]	224
shared_dir	225
ebs_kms_key_id	225
encrypted	226
num_of_raid_volumes	226
raid_type	226
volume_iops	227
volume_size	228
volume_throughput	229
volume_type	229
Bagian [scaling]	230
scaledown_idletime	230

Bagian [vpc]	231
additional_sg	231
compute_subnet_cidr	232
compute_subnet_id	232
master_subnet_id	232
ssh_from	232
use_public_ips	233
vpc_id	233
vpc_security_group_id	233
Contoh	39
Slurm contoh	235
SGE dan Torque contoh	236
AWS Batch contoh	237
Bagaimana cara AWS ParallelCluster kerja	239
AWS ParallelCluster proses	239
SGE and Torque integration processes	240
Slurm integration processes	246
AWS layanan yang digunakan oleh AWS ParallelCluster	246
AWS Auto Scaling	247
AWS Batch	248
AWS CloudFormation	248
Amazon CloudWatch	248
CloudWatch Log Amazon	249
AWS CodeBuild	249
Amazon DynamoDB	249
Amazon Elastic Block Store	250
Amazon Elastic Compute Cloud	250
Amazon Elastic Container Registry	250
Amazon EFS	251
Amazon FSx untuk Lustre	251
AWS Identity and Access Management	251
AWS Lambda	252
Amazon DCV	252
Amazon Route 53	252
Amazon Simple Notification Service	252
Amazon Simple Queue Service	253

Amazon Simple Storage Service	253
Amazon VPC	254
AWS ParallelCluster Auto Scaling	254
Menskalakan	255
Menskalakan	256
Cluster statis	256
Tutorial	257
Menjalankan pekerjaan pertama Anda di AWS ParallelCluster	257
Memverifikasi instalasi Anda	257
Membuat cluster pertama Anda	258
Masuk ke node kepala Anda	258
Menjalankan pekerjaan pertama Anda menggunakan SGE	259
Membangun AWS ParallelCluster AMI Kustom	260
Cara Menyesuaikan AWS ParallelCluster AMI	261
Memodifikasi AMI	261
Bangun AWS ParallelCluster AMI Kustom	264
Gunakan AMI Kustom saat Runtime	265
Menjalankan pekerjaan MPI dengan AWS ParallelCluster dan awsbatch penjadwal	266
Membuat cluster	266
Masuk ke node kepala Anda	258
Menjalankan pekerjaan pertama Anda menggunakan AWS Batch	268
Menjalankan pekerjaan MPI di lingkungan paralel multi-node	270
Enkripsi disk dengan Kunci KMS khusus	274
Menciptakan peran	275
Berikan izin kunci Anda	275
Membuat cluster	266
Beberapa mode antrian tutorial	276
Menjalankan pekerjaan Anda AWS ParallelCluster dengan beberapa mode antrian	276
Pengembangan	289
Menyiapkan AWS ParallelCluster buku masak khusus	289
Langkah-langkah	289
Menyiapkan paket AWS ParallelCluster node kustom	291
Langkah-langkah	291
Pemecahan Masalah	293
Mengambil dan melestarikan log	293
Memecahkan masalah penyebaran tumpukan	294

Memecahkan masalah di beberapa cluster mode antrian	295
Log kunci	295
Memecahkan masalah inialisasi node	296
Memecahkan masalah penggantian dan penghentian node yang tidak terduga	298
Mengganti, mengakhiri, atau mematikan instance dan node masalah	299
Memecahkan masalah node dan pekerjaan lain yang diketahui	300
Memecahkan masalah dalam kluster mode antrian tunggal	300
Log kunci	301
Pemecahan masalah gagal meluncurkan dan bergabung dengan operasi	302
Memecahkan masalah penskalaan	303
Memecahkan masalah terkait kluster lainnya	303
Grup penempatan dan masalah peluncuran instance	303
Direktori yang tidak dapat diganti	304
Memecahkan masalah di Amazon DCV	305
Log untuk Amazon DCV	305
Memori jenis instans Amazon DCV	305
Masalah Ubuntu Amazon DCV	305
Memecahkan masalah dalam cluster dengan integrasi AWS Batch	306
Masalah simpul kepala	306
AWS Batch masalah pengiriman pekerjaan paralel multi-node	306
Masalah komputasi	307
Kegagalan Job	307
Pemecahan masalah saat sumber daya gagal dibuat	307
Memecahkan masalah ukuran kebijakan IAM	308
Dukungan tambahan	309
AWS ParallelCluster kebijakan dukungan	310
Keamanan	311
Informasi keamanan untuk layanan yang digunakan oleh AWS ParallelCluster	312
Perlindungan data	312
Enkripsi data	313
Lihat juga	315
Identity and Access Management	315
Validasi kepatuhan	316
Menegakkan TLS 1.2	317
Tentukan Protokol yang Didukung Saat Ini	317
Kompilasi OpenSSL dan Python	318

Catatan rilis dan riwayat dokumen 320

..... ccclxii

Apa itu AWS ParallelCluster

AWS ParallelCluster adalah alat manajemen cluster open source yang AWS didukung yang membantu Anda menyebarkan dan mengelola cluster komputasi kinerja tinggi (HPC) di AWS Cloud. Ini secara otomatis mengatur sumber daya komputasi, penjadwal, dan sistem file bersama yang diperlukan. Anda dapat menggunakan AWS ParallelCluster dengan AWS Batch dan Slurm penjadwal.

Dengan AWS ParallelCluster, Anda dapat dengan cepat membangun dan menerapkan bukti konsep dan lingkungan komputasi HPC produksi. Anda juga dapat membangun dan menerapkan alur kerja tingkat tinggi di atasnya AWS ParallelCluster, seperti portal genomik yang mengotomatiskan seluruh alur kerja pengurutan DNA.

Harga

Saat menggunakan antarmuka baris AWS ParallelCluster perintah (CLI) atau API, Anda hanya membayar AWS sumber daya yang dibuat saat Anda membuat atau memperbarui AWS ParallelCluster gambar dan cluster. Untuk informasi selengkapnya, lihat [AWS layanan yang digunakan oleh AWS ParallelCluster](#).

Menyiapkan AWS ParallelCluster

Topik

- [Instalasi AWS ParallelCluster](#)
- [Mengkonfigurasi AWS ParallelCluster](#)
- [Praktik terbaik](#)
- [Pindah dari CfnCluster ke AWS ParallelCluster](#)
- [Wilayah yang Didukung](#)

Instalasi AWS ParallelCluster

AWS ParallelCluster didistribusikan sebagai paket Python dan diinstal menggunakan `pip`, manajer paket Python. Untuk informasi selengkapnya tentang menginstal paket Python, lihat [Menginstal paket di Panduan](#) Pengguna Kemasan Python.

Cara menginstal AWS ParallelCluster:

- [Menggunakan lingkungan virtual \(disarankan\)](#)
- [Menggunakan `pip`](#)

Anda dapat menemukan nomor versi CLI terbaru di [halaman rilis](#) di GitHub

Dalam panduan ini, contoh perintah mengasumsikan bahwa Anda telah menginstal Python v3. Contoh `pip` perintah menggunakan `pip3` versi.

Instalasi AWS ParallelCluster di lingkungan virtual (disarankan)

Kami menyarankan Anda menginstal AWS ParallelCluster di lingkungan virtual. Jika Anda mengalami masalah saat mencoba menginstal AWS ParallelCluster `pip3`, Anda dapat [menginstal AWS ParallelCluster di lingkungan virtual](#) untuk mengisolasi alat dan dependensinya. Atau Anda dapat menggunakan versi Python yang berbeda dari biasanya.

Menginstal AWS ParallelCluster di lingkungan non-virtual menggunakan pip

Metode distribusi utama untuk AWS ParallelCluster Linux, Windows, dan macOS adalah `pip`, yang merupakan manajer paket untuk Python. Ini menyediakan cara untuk menginstal, meningkatkan, dan menghapus paket Python dan dependensi mereka.

AWS ParallelCluster Versi Saat Ini

AWS ParallelCluster diperbarui secara berkala. Untuk menentukan apakah Anda memiliki versi terbaru, lihat [halaman rilis di GitHub](#).

Jika Anda sudah memiliki `pip` dan versi Python yang didukung, Anda dapat menginstal AWS ParallelCluster dengan menggunakan perintah berikut. Jika Anda telah menginstal Python versi 3+, kami sarankan Anda menggunakan perintah tersebut. **pip3**

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

Langkah-langkah yang harus diambil setelah instalasi

Setelah Anda menginstal AWS ParallelCluster, Anda mungkin perlu menambahkan path file executable ke variabel `PATH`. Untuk instruksi khusus platform, lihat topik berikut:

- Linux – [Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda](#)
- macOS – [Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda](#)
- Windows – [Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda](#)

Anda dapat memverifikasi bahwa AWS ParallelCluster diinstal dengan benar dengan menjalankan `pcluster version`.

```
$ pcluster version
2.11.9
```

AWS ParallelCluster diperbarui secara berkala. Untuk memperbarui ke versi terbaru AWS ParallelCluster, jalankan perintah instalasi lagi. Untuk detail tentang versi terbaru AWS ParallelCluster, lihat [catatan AWS ParallelCluster rilis](#).

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

Untuk menghapus instalasi AWS ParallelCluster, gunakan `pip uninstall`.

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

Jika Anda tidak memiliki Python dan pip, gunakan prosedur untuk lingkungan Anda.

Instruksi terperinci untuk setiap lingkungan

- [Instal AWS ParallelCluster di lingkungan virtual \(disarankan\)](#)
- [Instal AWS ParallelCluster di Linux](#)
- [Instal AWS ParallelCluster di macOS](#)
- [Instal AWS ParallelCluster di Windows](#)

Instal AWS ParallelCluster di lingkungan virtual (disarankan)

Kami menyarankan Anda menginstal AWS ParallelCluster di lingkungan virtual untuk menghindari konflik versi persyaratan dengan pip paket lain.

Prasyarat

- Verifikasi bahwa pip dan Python diinstal. Kami merekomendasikan pip3, dan Python 3 versi 3.8. Jika Anda menggunakan Python 2, gunakan pip sebagai pengganti pip3 dan virtualenv bukan. venv

Untuk menginstal AWS ParallelCluster di lingkungan virtual

1. Jika virtualenv tidak diinstal, instal virtualenv menggunakan pip3. Jika `python3 -m virtualenv help` menampilkan informasi bantuan, lanjutkan ke langkah 2.

Linux, macOS, or Unix

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

Jalankan `exit` untuk meninggalkan jendela terminal saat ini dan buka jendela terminal baru untuk mengambil perubahan pada lingkungan.

Windows

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

Jalankan `exit` untuk meninggalkan command prompt saat ini dan buka command prompt baru untuk mengambil perubahan pada lingkungan.

2. Buat lingkungan virtual dan beri nama.

Linux, macOS, or Unix

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

Atau, Anda dapat menggunakan `-p` opsi untuk menentukan versi Python tertentu.

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

Windows

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

3. Aktifkan lingkungan virtual baru Anda.

Linux, macOS, or Unix

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

Windows

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

4. Instal AWS ParallelCluster ke lingkungan virtual Anda.

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. Verifikasi bahwa AWS ParallelCluster sudah terpasang dengan benar.

Linux, macOS, or Unix

```
$ pcluster version  
2.11.9
```

Windows

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

Anda dapat menggunakan perintah `deactivate` untuk keluar dari lingkungan virtual. Setiap kali Anda memulai sesi, Anda harus [mengaktifkan kembali lingkungan](#).

Untuk meng-upgrade ke versi terbaru AWS ParallelCluster, jalankan perintah instalasi lagi.

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"
```

Instal AWS ParallelCluster di Linux

Anda dapat menginstal AWS ParallelCluster dan dependensinya pada sebagian besar distribusi Linux dengan menggunakan `pip`, manajer paket untuk Python. Pertama, tentukan apakah Python dan `pip` terinstal:

1. Untuk menentukan apakah versi Linux Anda menyertakan Python dan `pip`, jalankan. `pip --version`

```
$ pip --version
```

Jika Anda telah `pip` menginstal, lanjutkan ke topik [Instal AWS ParallelCluster dengan pip](#). Jika tidak, lanjutkan dengan Langkah 2.

2. Untuk menentukan apakah Python terinstal, jalankan. `python --version`

```
$ python --version
```

[Jika Anda menginstal Python 3 versi 3.6+ atau Python 2 versi 2.7, lanjutkan ke topik Instal dengan pip. AWS ParallelCluster](#) Jika tidak, [instal Python](#), dan kemudian kembali ke prosedur ini untuk menginstal. pip

3. Instal pip dengan menggunakan skrip yang disediakan oleh Python Packaging Authority.
4. Gunakan curl perintah untuk mengunduh skrip instalasi.

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

5. Jalankan skrip dengan Python untuk mengunduh dan menginstal versi terbaru pip dan paket dukungan lain yang diperlukan.

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

atau

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

Saat Anda menyertakan --user sakelar, skrip akan dipasang pip ke jalur ~/.local/bin.

6. Untuk memverifikasi bahwa folder yang berisi pip adalah bagian dari PATH variabel Anda, lakukan hal berikut:
 - a. Temukan penulisan profil shell Anda di folder pengguna Anda. Jika Anda tidak yakin shell mana yang Anda miliki, jalankan `basename $SHELL`.

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

- Bash — .bash_profile, .profile, atau .bash_login
- Zsh — .zshrc
- Tcsh — .tcshrc, atau .cshrc .login

- b. Tambahkan perintah ekspor di akhir skrip profil Anda yang mirip dengan contoh berikut.

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

Perintah ekspor menyisipkan jalur, yang ada ~/.local/bin dalam contoh ini, di depan PATH variabel yang ada.

- c. Untuk menerapkan perubahan ini, muat ulang profil ke sesi Anda saat ini.

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

7. Verifikasi bahwa pip terpasang dengan benar.

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

Bagian-bagian

- [Instal AWS ParallelCluster dengan pip](#)
- [Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda](#)
- [Menginstal Python di Linux](#)

Instal AWS ParallelCluster dengan **pip**

Gunakan pip untuk menginstal AWS ParallelCluster.

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

Saat Anda menggunakan --user sakelar, pip instal AWS ParallelCluster ke ~/.local/bin.

Verifikasi yang AWS ParallelCluster diinstal dengan benar.

```
$ pcluster version  
2.11.9
```

Untuk memperbarui ke versi terbaru, jalankan perintah pemasangan lagi.

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

Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda

Setelah menginstal dengan pip, Anda mungkin perlu menambahkan pcluster executable ke variabel PATH lingkungan sistem operasi Anda.

Untuk memverifikasi folder yang pip diinstal AWS ParallelCluster, jalankan perintah berikut.

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

Jika Anda menghilangkan `--user` sakelar saat Anda menginstal AWS ParallelCluster, executable mungkin ada di folder instalasi Python bin Anda. Jika Anda tidak tahu di mana Python diinstal, jalankan perintah ini.

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

Perhatikan bahwa output mungkin jalur ke symlink, bukan ke executable yang sebenarnya. Untuk melihat di mana titik symlink, jalankan `ls -al`.

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

Jika ini adalah folder yang sama yang Anda tambahkan ke jalur di langkah 3 di [Instalasi AWS ParallelCluster](#), Anda selesai dengan instalasi. Jika tidak, Anda harus melakukan langkah 3a - 3c lagi, menambahkan folder tambahan ini ke jalur.

Menginstal Python di Linux

Jika distribusi Anda tidak datang dengan Python, atau datang dengan versi sebelumnya, instal Python sebelum menginstal dan. pip AWS ParallelCluster

Untuk menginstal Python 3 di Linux

1. Periksa untuk melihat apakah Python sudah diinstal.

```
$ python3 --version
```

atau

```
$ python --version
```

 Note

Jika distribusi Linux Anda datang dengan Python, Anda mungkin perlu menginstal paket pengembang Python. Paket pengembang mencakup header dan pustaka yang diperlukan untuk mengkompilasi ekstensi dan menginstal. AWS ParallelCluster Gunakan manajer paket Anda untuk menginstal paket pengembang. Biasanya dinamai `python-dev` atau `python-devel`.

2. Jika Python 2.7 atau yang lebih baru tidak diinstal, instal Python dengan manajer paket distribusi Anda. Nama perintah dan paket bervariasi:

- Pada turunan Debian seperti Ubuntu, gunakan `apt`

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

- Di Red Hat dan turunannya, gunakan `yum`.

```
$ sudo yum install python3
```

- Di SUSE dan turunannya, gunakan `zypper`.

```
$ sudo zypper install python3
```

3. Untuk memverifikasi bahwa Python diinstal dengan benar, buka command prompt atau shell dan jalankan perintah berikut.

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

Instal AWS ParallelCluster di macOS

Bagian-bagian

- [Prasyarat](#)
- [Instal AWS ParallelCluster di macOS menggunakan pip](#)
- [Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda](#)

Prasyarat

- Python 3 versi 3.7+ atau Python 2 versi 2.7

Periksa instalasi Python Anda.

```
$ python --version
```

Jika komputer Anda belum menginstal Python, atau jika Anda ingin menginstal versi Python yang berbeda, ikuti prosedurnya. [Instal AWS ParallelCluster di Linux](#)

Instal AWS ParallelCluster di macOS menggunakan pip

Anda juga dapat menggunakan pip langsung untuk menginstal AWS ParallelCluster. Jika belum, ikuti petunjuk di [topik penginstalan](#) utama. Jalankan `pip3 --version` untuk melihat apakah versi macOS Anda sudah menyertakan Python dan `pip3`

```
$ pip3 --version
```

Untuk menginstal AWS ParallelCluster di macOS

1. [Unduh dan instal versi terbaru Python dari halaman unduhan Python.org.](#)
2. Download dan jalankan script `pip3` instalasi yang disediakan oleh Python Packaging Authority.

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

3. Gunakan yang baru diinstal `pip3` untuk menginstal AWS ParallelCluster. Kami menyarankan jika Anda menggunakan Python versi 3+, Anda menggunakan perintah. `pip3`

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

4. Verifikasi bahwa AWS ParallelCluster sudah terpasang dengan benar.

```
$ pcluster version  
2.11.9
```

Jika program tidak ditemukan, [tambahkan ke jalur baris perintah Anda.](#)

Untuk memperbarui ke versi terbaru, jalankan perintah pemasangan lagi.

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

Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda

Setelah menginstal dengan `pip`, Anda mungkin perlu menambahkan `pcluster` program ke variabel `PATH` lingkungan sistem operasi Anda. Lokasi program tergantung di mana Python diinstal.

Example AWS ParallelCluster instal lokasi - macOS dengan Python 3.6 dan (mode pengguna) **pip**

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

Gantikan versi Python yang Anda miliki untuk versi pada contoh sebelumnya.

Jika Anda tidak tahu di mana Python diinstal, jalankan `which python`

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

Outputnya mungkin jalur ke symlink, bukan jalur ke program yang sebenarnya. Jalankan `ls -al` untuk melihat ke mana ia menunjuk.

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

`pip` menginstal program di folder yang sama yang berisi aplikasi Python. Tambahkan folder ini ke `PATH` variabel Anda.

Untuk memodifikasi **PATH** variabel Anda (Linux, macOS, atau Unix)

1. Temukan penulisan profil shell Anda di folder pengguna Anda. Jika Anda tidak yakin shell mana yang Anda miliki, jalankan `echo $SHELL`.

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

- Bash — `.bash_profile`, `.profile`, atau `.bash_login`

- Zsh – `.zshrc`
 - Tcsh — `.tcshrc`, `.cshrc` atau `.login`
2. Tambahkan perintah ekspor ke penulisan profil Anda.

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

Perintah ini menambahkan jalur, `~/.local/bin` dalam contoh ini, ke PATH variabel saat ini.

3. Muat profil ke sesi Anda saat ini.

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

Instal AWS ParallelCluster di Windows

Anda dapat menginstal AWS ParallelCluster pada Windows dengan menggunakan `pip`, yang merupakan manajer paket untuk Python. Jika Anda sudah memiliki `pip`, ikuti instruksi dalam [topik instalasi](#) utama.

Bagian-bagian

- [Instal AWS ParallelCluster menggunakan Python dan pip pada Windows](#)
- [Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda](#)

Instal AWS ParallelCluster menggunakan Python dan **pip** pada Windows

Python Software Foundation menyediakan pemasangan untuk Windows yang mencakup `pip`.

Untuk menginstal Python dan **pip** (Windows)

1. [Unduh penginstal Python Windows x86-64 dari halaman unduhan Python.org.](#)
2. Jalankan pemasang.
3. Pilih Tambahkan Python 3 ke PATH.
4. Pilih Pasang Sekarang.

Penginstal menginstal Python di folder pengguna Anda dan menambahkan folder programnya ke jalur pengguna Anda.

Untuk menginstal AWS ParallelCluster dengan **pip3** (Windows)

Jika Anda menggunakan Python versi 3+, kami sarankan Anda menggunakan perintah. **pip3**

1. Buka Command Prompt dari menu Start.
2. Gunakan perintah berikut untuk memverifikasi bahwa Python dan pip keduanya diinstal dengan benar.

```
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. Instal AWS ParallelCluster menggunakan **pip**.

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

4. Verifikasi bahwa AWS ParallelCluster sudah terpasang dengan benar.

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

Untuk memperbarui ke versi terbaru, jalankan perintah pemasangan lagi.

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

Tambahkan AWS ParallelCluster executable ke jalur baris perintah Anda

Setelah menginstal AWS ParallelCluster dengan **pip**, tambahkan **pcluster** program ke variabel **PATH** lingkungan sistem operasi Anda.

Anda dapat menemukan di mana **pcluster** program diinstal dengan menjalankan perintah berikut.

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

Jika perintah itu tidak mengembalikan hasil apa pun, maka Anda harus menambahkan jalur secara manual. Gunakan baris perintah atau Windows Explorer untuk menemukan di mana ia diinstal pada komputer Anda. Jalur khas meliputi:

- Python 3 dan - pip3 C:\Python38\Scripts\
- Opsi Python 3 dan **pip3** --user - %APPDATA%\Python\Python38\Scripts

Note

Nama folder yang menyertakan nomor versi dapat bervariasi. Contoh sebelumnya menunjukkan Python38. Ganti sesuai kebutuhan dengan nomor versi yang Anda gunakan.

Untuk memodifikasi variabel PATH Anda (Windows)

1. Tekan tombol Windows dan masukkan **environment variables**.
2. Pilih Mengedit variabel lingkungan untuk akun Anda.
3. Pilih PATH, lalu pilih Edit.
4. Tambahkan jalur ke bidang Nilai variabel. Sebagai contoh: **C:\new\path**
5. Pilih OK dua kali untuk menerapkan pengaturan baru.
6. Tutup semua command prompt yang sedang berjalan dan buka kembali jendela command prompt.

Mengkonfigurasi AWS ParallelCluster

Setelah Anda menginstal AWS ParallelCluster, selesaikan langkah-langkah konfigurasi berikut.

Verifikasi bahwa AWS Akun Anda memiliki peran yang mencakup izin yang diperlukan untuk menjalankan [pcluster](#) CLI. Untuk informasi selengkapnya, lihat [AWS ParallelCluster contoh contoh dan kebijakan pengguna](#).

Siapkan AWS kredensial Anda. Untuk informasi selengkapnya, lihat [Mengonfigurasi AWS CLI](#) dalam panduan AWS CLI pengguna.

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

Wilayah AWS Tempat cluster diluncurkan harus memiliki setidaknya satu EC2 key pair Amazon. Untuk informasi selengkapnya, lihat [pasangan EC2 kunci Amazon](#) di Panduan EC2 Pengguna Amazon.

```
$ pcluster configure
```

Wizard konfigurasi meminta Anda untuk semua informasi yang diperlukan untuk membuat cluster Anda. Rincian urutan berbeda saat menggunakan AWS Batch sebagai penjadwal dibandingkan dengan menggunakan Slurm. Untuk informasi selengkapnya tentang konfigurasi klaster, lihat [Konfigurasi](#).

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk 2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS

Slurm

Dari daftar Wilayah AWS pengidentifikasi yang valid, pilih Wilayah AWS tempat yang Anda inginkan untuk menjalankan klaster Anda.

Note

Daftar yang Wilayah AWS ditampilkan didasarkan pada partisi akun Anda, dan hanya mencakup Wilayah AWS yang diaktifkan untuk akun Anda. Untuk informasi selengkapnya tentang mengaktifkan Wilayah AWS akun Anda, lihat [Mengelola Wilayah AWS](#) di Referensi Umum AWS Contoh yang ditampilkan adalah dari partisi AWS Global. Jika akun Anda ada di AWS GovCloud (US) partisi, hanya Wilayah AWS di partisi yang terdaftar (gov-us-east-1 dan gov-us-west-1). Demikian pula, jika akun Anda berada di partisi AWS Tiongkok, cn-north-1 hanya cn-northwest-1 dan ditampilkan. Untuk daftar lengkap yang Wilayah AWS didukung oleh AWS ParallelCluster, lihat [Wilayah yang Didukung](#).

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

Pilih penjadwal yang akan digunakan dengan cluster Anda.

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

Pilih sistem operasi.

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

Note

Support untuk `alinux2` ditambahkan di AWS ParallelCluster versi 2.6.0.

Ukuran minimum dan maksimum cluster node komputasi dimasukkan. Ini diukur dalam jumlah kasus.

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

Jenis instance head dan compute node dimasukkan. Misalnya jenis, batas instans akun Anda cukup besar untuk memenuhi kebutuhan Anda. Untuk informasi selengkapnya, lihat [Batas Instans Sesuai Permintaan](#) di Panduan EC2 Pengguna Amazon.

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

Key pair dipilih dari pasangan kunci yang terdaftar dengan Amazon EC2 di yang dipilih Wilayah AWS.

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

Setelah langkah-langkah sebelumnya selesai, putuskan apakah akan menggunakan VPC yang ada atau biarkan AWS ParallelCluster membuat VPC untuk Anda. Jika Anda tidak memiliki VPC yang dikonfigurasi dengan benar, AWS ParallelCluster dapat membuat yang baru. Ini baik menggunakan node head dan compute di subnet publik yang sama, atau hanya node kepala di subnet publik dengan semua node di subnet pribadi. Dimungkinkan untuk mencapai batas Anda pada jumlah VPCs dalam a Wilayah AWS. Batas default adalah lima VPCs untuk masing-masing Wilayah AWS. Untuk informasi selengkapnya tentang batas ini dan cara meminta peningkatan, lihat [VPC dan subnet](#) di Panduan Pengguna Amazon VPC.

Jika Anda membiarkan AWS ParallelCluster membuat VPC, Anda harus memutuskan apakah semua node harus berada dalam subnet publik.

 Important

VPCs dibuat oleh AWS ParallelCluster jangan aktifkan VPC Flow Logs secara default. VPC Flow Logs memungkinkan Anda untuk menangkap informasi tentang lalu lintas IP

yang pergi ke dan dari antarmuka jaringan di Anda. VPCs Untuk informasi selengkapnya, lihat [Log Alur VPC](#) di Panduan Pengguna Amazon VPC.

Note

Jika Anda memilih 1. Master in a public subnet and compute fleet in a private subnet, AWS ParallelCluster buat gateway NAT yang menghasilkan biaya tambahan, bahkan jika Anda menentukan sumber daya tingkat gratis.

```
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
```

Jika Anda tidak membuat VPC baru, Anda harus memilih VPC yang ada.

Jika Anda memilih untuk AWS ParallelCluster membuat VPC, catat ID VPC sehingga Anda dapat menggunakan AWS CLI untuk menghapusnya nanti.

```
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
```

Setelah VPC dipilih, Anda perlu memutuskan apakah akan menggunakan subnet yang ada atau membuat yang baru.

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

```
Creating CloudFormation stack...
```

```
Do not leave the terminal until the process has finished
```

AWS Batch

Dari daftar Wilayah AWS pengidentifikasi yang valid, pilih Wilayah AWS tempat yang Anda inginkan untuk menjalankan klaster Anda.

```
Allowed values for Wilayah 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

```
Wilayah AWS ID [ap-northeast-1]:
```

Pilih penjadwal yang akan digunakan dengan cluster Anda.

```
Allowed values for Scheduler:
```

1. slurm
2. awsbatch

```
Scheduler [awsbatch]:
```

Kapan awsbatch dipilih sebagai penjadwal, `alinux2` digunakan sebagai sistem operasi.

Ukuran minimum dan maksimum cluster node komputasi dimasukkan. Ini diukur dalam vCPUs.

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

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

Jenis instance head node dimasukkan. Saat menggunakan awsbatch penjadwal, node komputasi menggunakan jenis instance. `optimal`

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

Amazon EC2 key pair dipilih dari pasangan kunci yang terdaftar dengan Amazon EC2 di yang dipilih Wilayah AWS.

Allowed values for EC2 Key Pair Name:

1. prod-uswest1-key
2. test-uswest1-key

```
EC2 Key Pair Name [prod-uswest1-key]:
```

Putuskan apakah akan menggunakan yang ada VPCs atau biarkan AWS ParallelCluster buat VPCs untuk Anda. Jika Anda tidak memiliki VPC yang dikonfigurasi dengan benar, AWS ParallelCluster dapat membuat yang baru. Ini baik menggunakan node head dan compute di subnet publik yang sama, atau hanya node kepala di subnet publik dengan semua node di subnet pribadi. Dimungkinkan untuk mencapai batas Anda pada jumlah VPCs dalam a Wilayah AWS. Jumlah default VPCs adalah lima. Untuk informasi selengkapnya tentang batas ini dan cara meminta peningkatan, lihat [VPC dan subnet](#) di Panduan Pengguna Amazon VPC.

Important

VPCs dibuat oleh AWS ParallelCluster jangan aktifkan VPC Flow Logs secara default. VPC Flow Logs memungkinkan Anda untuk menangkap informasi tentang lalu lintas IP yang pergi ke dan dari antarmuka jaringan di Anda. VPCs Untuk informasi selengkapnya, lihat [Log Alur VPC](#) di Panduan Pengguna Amazon VPC.

Jika Anda membiarkan AWS ParallelCluster membuat VPC, putuskan apakah semua node harus berada dalam subnet publik.

Note

Jika Anda memilih 1. Master in a public subnet and compute fleet in a private subnet, AWS ParallelCluster buat gateway NAT yang menghasilkan biaya tambahan, bahkan jika Anda menentukan sumber daya tingkat gratis.

```
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

Jika Anda tidak membuat VPC baru, Anda harus memilih VPC yang ada.

Jika Anda memilih untuk AWS ParallelCluster membuat VPC, catat ID VPC sehingga Anda dapat menggunakan AWS CLI untuk menghapusnya nanti.

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**

Setelah VPC dipilih, putuskan apakah akan menggunakan subnet yang ada atau membuat yang baru.

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

Creating CloudFormation stack...

Do not leave the terminal until the process has finished

Ketika Anda telah menyelesaikan langkah-langkah sebelumnya, sebuah cluster sederhana diluncurkan ke VPC. VPC menggunakan subnet yang ada yang mendukung alamat IP publik. Tabel rute untuk subnet adalah $0.0.0.0/0 \Rightarrow igw-xxxxxx$. Perhatikan kondisi berikut:

- VPC harus memiliki DNS Resolution = yes dan DNS Hostnames = yes
- VPC juga harus memiliki opsi DHCP dengan benar domain-name untuk. Wilayah AWS DHCP Option Set default sudah menentukan yang diperlukan AmazonProvidedDNS. Jika menentukan lebih dari satu server nama domain, lihat [set opsi DHCP di Panduan Pengguna Amazon VPC](#). Saat menggunakan subnet pribadi, gunakan gateway NAT atau proxy internal untuk mengaktifkan akses web untuk node komputasi. Untuk informasi selengkapnya, lihat [Konfigurasi jaringan](#).

Ketika semua pengaturan berisi nilai yang valid, Anda dapat meluncurkan cluster dengan menjalankan perintah `create`.

```
$ pcluster create mycluster
```

Setelah cluster mencapai status “CREATE_COMPLETE”, Anda dapat menghubungkannya dengan menggunakan pengaturan klien SSH normal Anda. Untuk informasi selengkapnya tentang menghubungkan ke EC2 instans Amazon, lihat [Panduan EC2 Pengguna di Panduan EC2](#) Pengguna Amazon.

Untuk menghapus cluster, jalankan perintah berikut.

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

Untuk menghapus sumber daya jaringan di VPC, Anda dapat menghapus tumpukan CloudFormation jaringan. Nama tumpukan dimulai dengan “parallelclusternetworking-” dan berisi waktu pembuatan dalam format “YYYYMMDDHHMMSS”. Anda dapat membuat daftar tumpukan menggunakan perintah [daftar-tumpukan](#).

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

Tumpukan dapat dihapus menggunakan perintah [delete-stack](#).

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

VPC yang dibuat untuk [pcluster configure](#) Anda tidak dibuat di tumpukan CloudFormation jaringan. Anda dapat menghapus VPC itu secara manual di konsol atau dengan menggunakan file. AWS CLI

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

Praktik terbaik

Praktik terbaik: pemilihan tipe instans master

Meskipun master node tidak menjalankan pekerjaan apa pun, fungsi dan ukurannya sangat penting untuk kinerja cluster secara keseluruhan.

Saat memilih jenis instance yang akan digunakan untuk node master Anda, Anda ingin mengevaluasi item berikut:

- Ukuran cluster: node master mengatur logika penskalaan cluster dan bertanggung jawab untuk melampirkan node baru ke scheduler. Jika Anda perlu meningkatkan dan menurunkan cluster dari sejumlah besar node maka Anda ingin memberikan node master beberapa kapasitas komputasi tambahan.
- Sistem file bersama: saat menggunakan sistem file bersama untuk berbagi artefak antara node komputasi dan node master memperhitungkan bahwa master adalah node yang mengekspos server NFS. Untuk alasan ini, Anda ingin memilih jenis instans dengan bandwidth jaringan yang cukup dan bandwidth Amazon EBS khusus yang cukup untuk menangani alur kerja Anda.

Praktik terbaik: kinerja jaringan

Ada tiga petunjuk yang mencakup seluruh jajaran kemungkinan untuk meningkatkan komunikasi jaringan.

- Grup penempatan: grup penempatan cluster adalah pengelompokan logis instance dalam satu Availability Zone. Untuk informasi selengkapnya tentang grup [penempatan](#), lihat [grup penempatan](#) di Panduan EC2 Pengguna Amazon. Anda dapat mengonfigurasi cluster untuk menggunakan grup penempatan Anda sendiri dengan `placement_group = your-placement-group-name` atau membiarkan AWS ParallelCluster membuat grup penempatan dengan "compute" strategi dengan `placement_group = DYNAMIC`. Untuk informasi selengkapnya, lihat [placement_group](#) untuk beberapa mode antrian dan [placement_group](#) untuk mode antrian tunggal.
- Jaringan yang disempurnakan: pertimbangkan untuk memilih jenis instance yang mendukung Jaringan yang Ditingkatkan. Untuk informasi selengkapnya, lihat [jaringan yang disempurnakan di Linux](#) di Panduan EC2 Pengguna Amazon.
- Adaptor Kain Elastis: Untuk mendukung komunikasi antar instans tingkat tinggi yang dapat diskalakan, pertimbangkan untuk memilih antarmuka jaringan EFA untuk jaringan Anda. Perangkat

keras bypass sistem operasi (OS) yang dibuat khusus EFA meningkatkan komunikasi antar instans dengan elastisitas dan fleksibilitas cloud sesuai permintaan. AWS Untuk mengkonfigurasi satu Slurm antrian cluster untuk menggunakan EFA, atur. `enable_efa = true` Untuk informasi lebih lanjut tentang menggunakan EFA dengan AWS ParallelCluster, lihat [Elastic Fabric Adapter](#) dan [enable_efa](#). Untuk informasi selengkapnya tentang EFA, lihat [Adaptor Kain Elastis](#) di Panduan EC2 Pengguna Amazon untuk Instans Linux.

- Bandwidth instans: skala bandwidth dengan ukuran instans, pertimbangkan untuk memilih jenis instans yang lebih sesuai dengan kebutuhan Anda, lihat [instans Amazon EBS yang dioptimalkan](#) dan jenis [volume Amazon EBS di Panduan Pengguna](#) Amazon. EC2

Praktik terbaik: peringatan anggaran

Untuk mengelola biaya AWS ParallelCluster sumber daya, sebaiknya gunakan AWS Budgets tindakan untuk membuat anggaran dan peringatan ambang anggaran yang ditentukan untuk AWS sumber daya yang dipilih. Untuk informasi selengkapnya, lihat [Mengonfigurasi tindakan anggaran](#) di Panduan AWS Budgets Pengguna. Anda juga dapat menggunakan Amazon CloudWatch untuk membuat alarm penagihan. Untuk informasi selengkapnya, lihat [Membuat alarm penagihan untuk memantau perkiraan AWS biaya](#).

Praktik terbaik: memindahkan cluster ke versi AWS ParallelCluster minor atau patch baru

Saat ini setiap versi AWS ParallelCluster minor mandiri bersama dengan CLI-nya `apcluster`. Untuk memindahkan cluster ke versi minor atau patch baru, Anda harus membuat ulang cluster menggunakan CLI versi baru.

Untuk mengoptimalkan proses pemindahan klaster ke versi minor baru atau menyimpan data penyimpanan bersama Anda karena alasan lain, sebaiknya gunakan praktik terbaik berikut.

- Simpan data pribadi dalam volume eksternal, seperti Amazon EFS dan FSx untuk Lustre. Dengan melakukan ini, Anda dapat dengan mudah memindahkan data dari satu cluster ke cluster lainnya.
- Buat sistem penyimpanan bersama dari jenis yang tercantum di bawah ini menggunakan AWS CLI atau AWS Management Console:
 - [Bagian \[ebs\]](#)
 - [Bagian \[efs\]](#)
 - [Bagian \[fsx\]](#)

Tambahkan mereka ke konfigurasi cluster baru sebagai sistem file yang ada. Dengan cara ini, mereka dipertahankan ketika Anda menghapus cluster dan dapat dilampirkan ke cluster baru. Sistem penyimpanan bersama umumnya dikenakan biaya apakah terpasang atau terlepas dari cluster.

Kami menyarankan Anda menggunakan Amazon EFS, atau Amazon FSx untuk sistem file Lustre karena mereka dapat dilampirkan ke beberapa cluster pada saat yang sama dan Anda dapat melampirkannya ke cluster baru sebelum menghapus cluster lama. Untuk informasi selengkapnya, lihat [Memasang sistem file Amazon EFS](#) di Panduan Pengguna Amazon EFS dan [Mengakses FSx sistem file Lustre](#) di Panduan Pengguna Amazon FSx for Lustre Lustre.

- Gunakan [tindakan bootstrap kustom](#) untuk menyesuaikan instance Anda daripada AMI kustom. Ini mengoptimalkan proses pembuatan karena AMI kustom baru tidak perlu dibuat untuk setiap versi baru.
- Urutan yang direkomendasikan.
 1. Perbarui konfigurasi cluster untuk menggunakan definisi sistem file yang ada.
 2. Verifikasi `pccluster` versi dan perbarui jika diperlukan.
 3. Buat dan uji cluster baru.
 - Pastikan data Anda tersedia di cluster baru.
 - Pastikan aplikasi Anda bekerja di cluster baru.
 4. Jika klaster baru Anda sepenuhnya diuji dan operasional dan Anda yakin tidak akan menggunakan cluster lama, hapus.

Pindah dari CfnCluster ke AWS ParallelCluster

AWS ParallelCluster adalah versi yang disempurnakan dari CfnCluster.

Jika saat ini Anda menggunakan CfnCluster, kami sarankan Anda menggunakan AWS ParallelCluster sebagai gantinya dan membuat cluster baru dengannya. Meskipun Anda dapat terus menggunakannya CfnCluster, itu tidak lagi dikembangkan, dan tidak ada fitur atau fungsionalitas baru yang akan ditambahkan.

Perbedaan utama antara CfnCluster dan AWS ParallelCluster dijelaskan di bagian berikut.

AWS ParallelCluster CLI mengelola kumpulan cluster yang berbeda

Cluster yang dibuat dengan `cfnccluster` CLI tidak dapat dikelola dengan CLI. `pcluster` Perintah berikut tidak berfungsi pada cluster yang dibuat oleh CfnCluster:

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

Untuk mengelola cluster yang Anda buat dengan CfnCluster, Anda harus menggunakan `cfnccluster` CLI.

Jika Anda memerlukan CfnCluster paket untuk mengelola cluster lama Anda, kami sarankan Anda menginstal dan menggunakannya dari lingkungan virtual [Python](#).

AWS ParallelCluster dan CfnCluster menggunakan kebijakan khusus IAM yang berbeda

Kebijakan IAM khusus yang sebelumnya digunakan untuk pembuatan CfnCluster klaster tidak dapat digunakan. AWS ParallelCluster Jika Anda memerlukan kebijakan khusus untuk AWS ParallelCluster, Anda harus membuat yang baru. Lihat AWS ParallelCluster panduannya.

AWS ParallelCluster dan CfnCluster menggunakan file konfigurasi yang berbeda

File AWS ParallelCluster konfigurasi berada di `~/.parallelcluster` folder. File CfnCluster konfigurasi berada di `~/.cfnccluster` folder.

Jika Anda ingin menggunakan file CfnCluster konfigurasi yang ada dengan AWS ParallelCluster, maka Anda harus menyelesaikan tindakan berikut:

1. Pindahkan file konfigurasi dari `~/.cfnccluster/config` ke `~/.parallelcluster/config`.
2. Jika Anda menggunakan parameter [extra_json](#) konfigurasi, ubah seperti yang ditunjukkan.

CfnCluster pengaturan:

```
extra_json = { "cfnccluster" : { } }
```

AWS ParallelCluster pengaturan:

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

Di AWS ParallelCluster, ganglia dinonaktifkan secara default

Di AWS ParallelCluster, ganglia dinonaktifkan secara default. Untuk mengaktifkan ganglia, selesaikan langkah-langkah ini:

1. Atur [extra_json](#) parameter seperti yang ditunjukkan:

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

2. Ubah grup keamanan kepala untuk memungkinkan koneksi ke port 80.

Grup `parallelcluster-<CLUSTER_NAME>-MasterSecurityGroup-<xxx>` keamanan harus dimodifikasi dengan menambahkan aturan grup keamanan baru untuk mengizinkan koneksi masuk ke port 80 dari IP Publik Anda. Untuk informasi selengkapnya, lihat [Menambahkan aturan ke grup keamanan](#) di Panduan EC2 Pengguna Amazon.

Wilayah yang Didukung

AWS ParallelCluster versi 2.x tersedia sebagai berikut: Wilayah AWS

Nama Wilayah	Wilayah
US East (Ohio)	us-east-2
US East (N. Virginia)	us-east-1
US West (N. California)	us-west-1
US West (Oregon)	as-barat-2
Afrika (Cape Town)	af-selatan-1
Asia Pasifik (Hong Kong)	ap-timur-1
Asia Pasifik (Mumbai)	ap-south-1
Asia Pacific (Seoul)	ap-northeast-2
Asia Pacific (Singapore)	ap-southeast-1

Nama Wilayah	Wilayah
Asia Pacific (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-sentral-1
China (Beijing)	cn-north-1
China (Ningxia)	cn-northwest-1
Europe (Frankfurt)	eu-central-1
Europe (Ireland)	eu-west-1
Europe (London)	eu-barat-2
Eropa (Milan)	eu-selatan-1
Eropa (Paris)	eu-west-3
Europe (Stockholm)	eu-utara-1
Timur Tengah (Bahrain)	me-selatan-1
Amerika Selatan (São Paulo)	sa-east-1
AWS GovCloud (AS-Timur)	us-gov-east-1
AWS GovCloud (AS-Barat)	us-gov-west-1

Menggunakan AWS ParallelCluster

Topik

- [Konfigurasi jaringan](#)
- [Tindakan Bootstrap Kustom](#)
- [Bekerja dengan Amazon S3](#)
- [Berkeja dengan Instans Spot](#)
- [AWS Identity and Access Management peran dalam AWS ParallelCluster](#)
- [Penjadwal didukung oleh AWS ParallelCluster](#)
- [AWS ParallelCluster sumber daya dan penandaan](#)
- [CloudWatch Dasbor Amazon](#)
- [Integrasi dengan Amazon CloudWatch Logs](#)
- [Elastic Fabric Adapter](#)
- [Solusi Pilihan Intel](#)
- [Aktifkan Intel MPI](#)
- [Spesifikasi Platform Intel HPC](#)
- [Perpustakaan Kinerja Arm](#)
- [Connect ke head node melalui Amazon DCV](#)
- [Menggunakan pcluster update](#)
- [Penambahan AMI dan penggantian EC2 instance](#)

Konfigurasi jaringan

AWS ParallelCluster menggunakan Amazon Virtual Private Cloud (VPC) untuk jaringan. VPC menyediakan platform jaringan yang fleksibel dan dapat dikonfigurasi di mana Anda dapat menyebarkan cluster.

VPC harus memiliki `DNS Resolution = yes`, `DNS Hostnames = yes` dan opsi DHCP dengan nama domain yang benar untuk Wilayah. DHCP Option Set default sudah menentukan DNS yang diperlukan Amazon Provided. Jika menentukan lebih dari satu server nama domain, lihat [Pilihan DHCP disetel](#) di Panduan Pengguna Amazon VPC.

AWS ParallelCluster mendukung konfigurasi tingkat tinggi berikut:

- Satu subnet untuk node head dan compute.
- Dua subnet, dengan node kepala dalam satu subnet publik, dan menghitung node dalam subnet pribadi. Subnet dapat berupa yang baru atau yang sudah ada.

Semua konfigurasi ini dapat beroperasi dengan atau tanpa alamat IP publik. AWS ParallelCluster juga dapat digunakan untuk menggunakan proxy HTTP untuk semua AWS permintaan. Kombinasi konfigurasi ini menghasilkan banyak skenario penerapan. Misalnya, Anda dapat mengkonfigurasi subnet publik tunggal dengan semua akses melalui internet. Atau, Anda dapat mengonfigurasi jaringan pribadi sepenuhnya menggunakan AWS Direct Connect dan proxy HTTP untuk semua lalu lintas.

Lihat diagram arsitektur berikut untuk ilustrasi dari beberapa skenario ini:

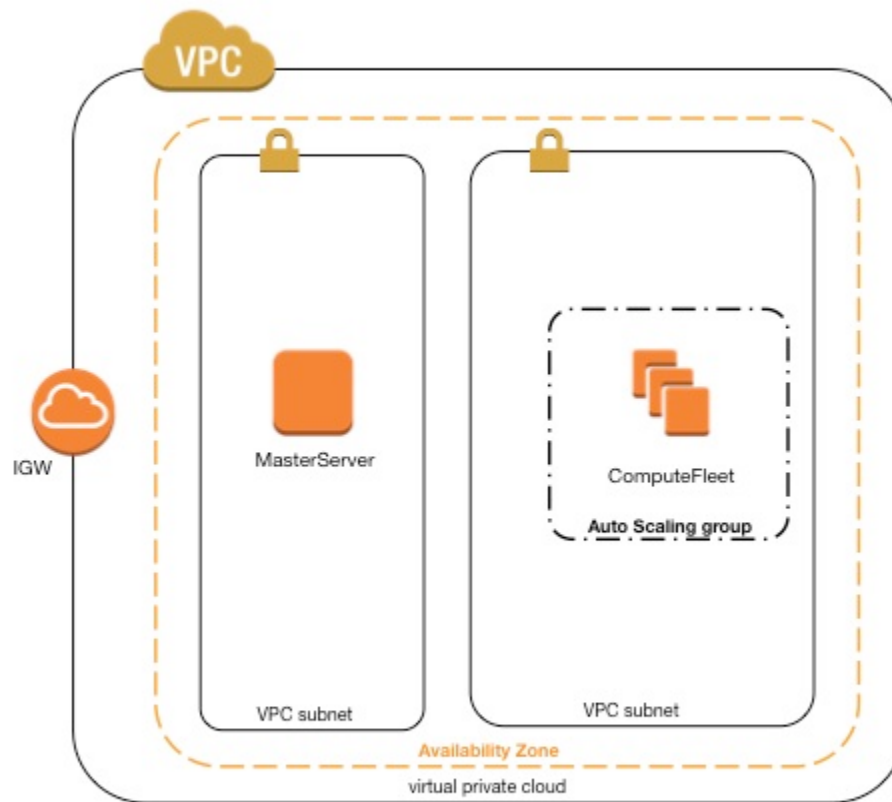
AWS ParallelCluster dalam satu subnet publik

Konfigurasi untuk arsitektur ini memerlukan pengaturan berikut:

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

[use_public_ips](#) Pengaturan tidak dapat diatur ke `false`, karena gateway internet mengharuskan semua instance memiliki alamat IP yang unik secara global. Untuk informasi selengkapnya, lihat [Mengaktifkan akses internet](#) di Panduan Pengguna Amazon VPC.

AWS ParallelCluster menggunakan dua subnet



Konfigurasi untuk membuat subnet pribadi baru untuk instance komputasi memerlukan pengaturan berikut:

Perhatikan bahwa semua nilai hanya disediakan sebagai contoh.

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

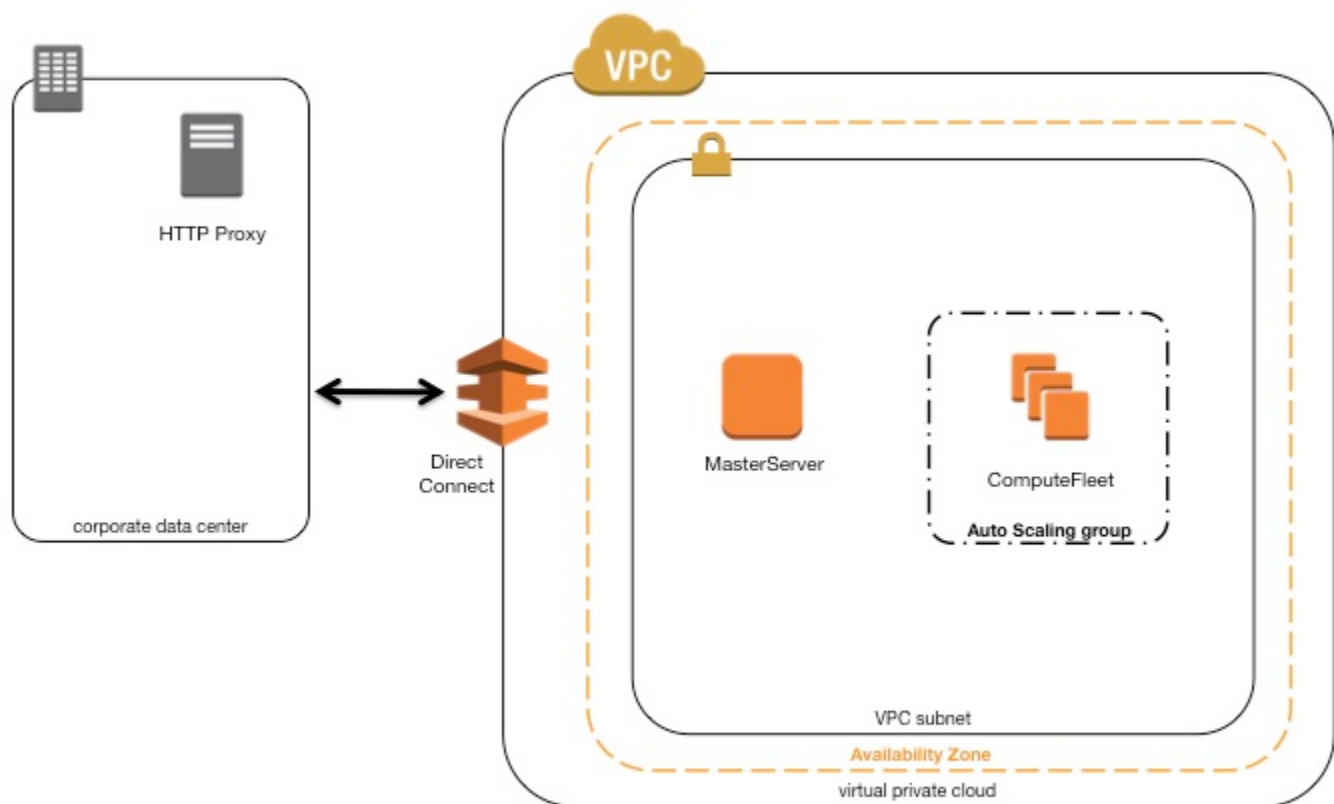
Konfigurasi untuk menggunakan jaringan pribadi yang ada memerlukan pengaturan berikut:

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

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

Kedua konfigurasi ini memerlukan [gateway NAT](#) atau proxy internal untuk mengaktifkan akses web untuk instance komputasi.

AWS ParallelCluster dalam satu subnet pribadi yang terhubung menggunakan AWS Direct Connect



Konfigurasi untuk arsitektur ini memerlukan pengaturan berikut:

```
[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
```

Ketika `use_public_ips` diatur ke `false`, VPC harus diatur dengan benar untuk menggunakan Proxy untuk semua lalu lintas. Akses web diperlukan untuk node head dan compute.

AWS ParallelCluster dengan **awsbatch** scheduler

Saat Anda menggunakan `awsbatch` sebagai tipe penjadwal, AWS ParallelCluster buat lingkungan komputasi AWS Batch terkelola. AWS Batch Lingkungan menangani pengelolaan instans penampung Amazon Elastic Container Service (Amazon ECS) Container Service (Amazon ECS), yang diluncurkan di `compute_subnet` AWS Batch. Agar berfungsi dengan benar, instans penampung Amazon ECS memerlukan akses jaringan eksternal untuk berkomunikasi dengan titik akhir layanan Amazon ECS. Ini diterjemahkan ke dalam skenario berikut:

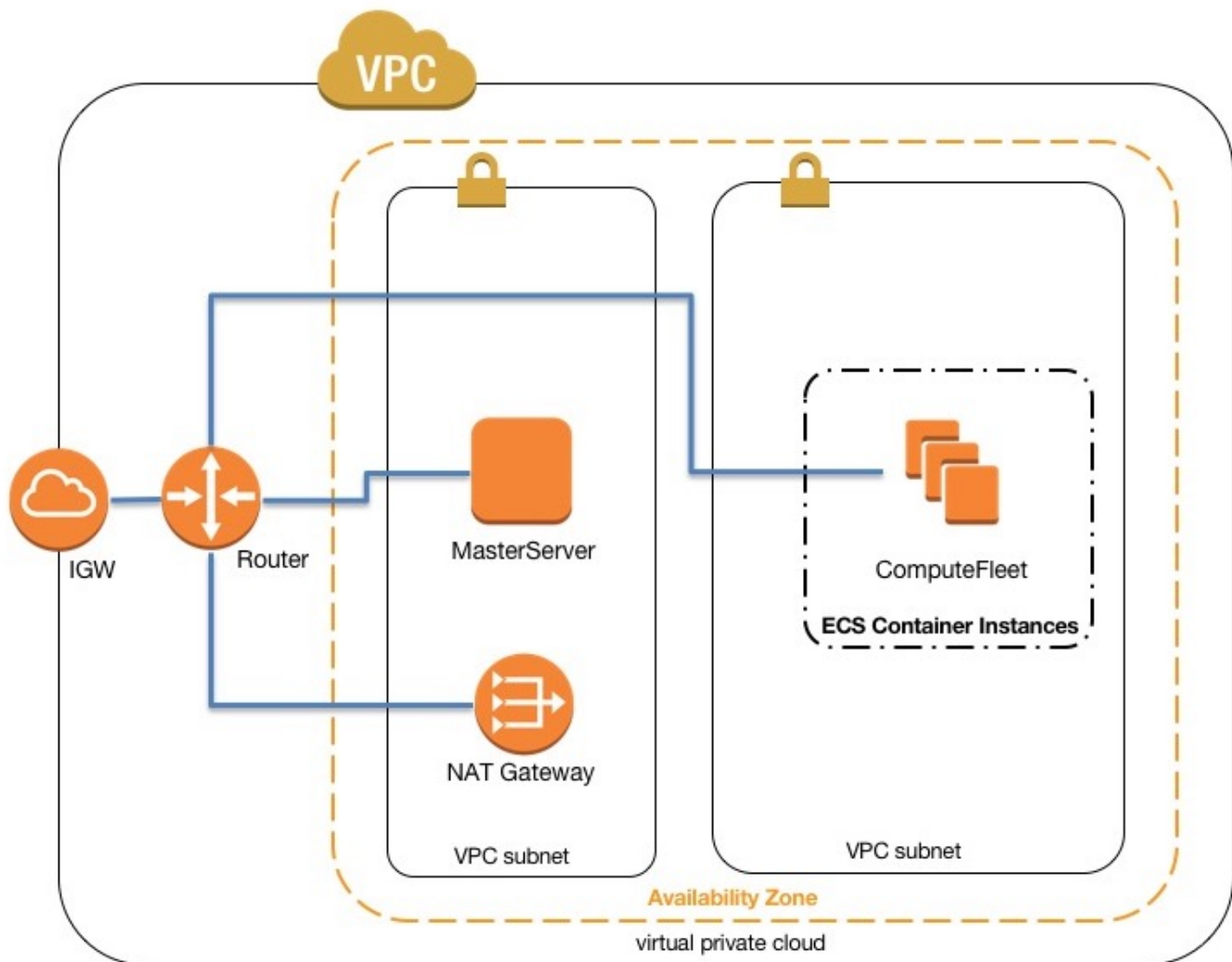
- `compute_subnet` Menggunakan gateway NAT untuk mengakses internet. (Kami merekomendasikan pendekatan ini.)
- Instans yang diluncurkan di `compute_subnet` memiliki alamat IP publik dan dapat menjangkau internet melalui Internet Gateway.

Selain itu, jika Anda tertarik dengan pekerjaan paralel multi-node (dari [AWS Batch dokumen](#)):

AWS Batch multi-node parallel jobs menggunakan mode `awsvpc` jaringan Amazon ECS, yang memberikan container kerja paralel multi-node Anda properti jaringan yang sama dengan instans Amazon EC2. Setiap kontainer tugas paralel multisimpul mendapat antarmuka jaringan elastis sendiri, alamat IP privat utama, dan nama host DNS internal. Antarmuka jaringan dibuat di subnet VPC Amazon yang sama dengan sumber daya komputasi host-nya. Grup keamanan yang diterapkan pada sumber daya komputasi Anda juga diterapkan pada antarmuka jaringan.

Saat menggunakan Jaringan Tugas Amazon ECS, mode `awsvpc` jaringan tidak menyediakan antarmuka jaringan elastis dengan alamat IP publik untuk tugas yang menggunakan jenis EC2 peluncuran Amazon. Untuk mengakses internet, tugas yang menggunakan jenis EC2 peluncuran Amazon harus diluncurkan di subnet pribadi yang dikonfigurasi untuk menggunakan gateway NAT.

Anda harus mengonfigurasi gateway NAT untuk mengaktifkan cluster menjalankan tugas paralel multi-node.



Untuk informasi selengkapnya, lihat topik berikut:

- [AWS Batch lingkungan komputasi terkelola](#)
- [AWS Batch pekerjaan paralel multi-node](#)
- [Jaringan tugas Amazon ECS dengan mode awsvpc jaringan](#)

Tindakan Bootstrap Kustom

AWS ParallelCluster dapat menjalankan kode arbitrer baik sebelum (pra-instal) atau setelah (pasca-instal) tindakan bootstrap utama saat cluster dibuat. Dalam kebanyakan kasus, kode ini disimpan di Amazon Simple Storage Service (Amazon S3) dan diakses melalui koneksi HTTPS. Kode dijalankan

sebagai root dan dapat dalam bahasa skrip apa pun yang didukung oleh OS cluster. Seringkali kode dalam Bash atau Python.

Tindakan pra-instal dipanggil sebelum tindakan bootstrap penerapan cluster dimulai, seperti mengonfigurasi NAT, Amazon Elastic Block Store (Amazon EBS) atau penjadwal. Beberapa tindakan pra-instal termasuk memodifikasi penyimpanan, menambahkan pengguna tambahan, dan menambahkan paket.

Tindakan pasca-instal dipanggil setelah proses bootstrap cluster selesai. Tindakan pasca-instal adalah tindakan terakhir yang terjadi sebelum sebuah instance dianggap sepenuhnya dikonfigurasi dan selesai. Beberapa tindakan pasca-instal termasuk mengubah pengaturan penjadwal, memodifikasi penyimpanan, dan memodifikasi paket.

Anda dapat meneruskan argumen ke skrip dengan menentukannya selama konfigurasi. Untuk ini, Anda meneruskannya dikutip ganda ke tindakan pra-instal atau pasca-instal.

Jika tindakan pra-instal atau pasca-instal gagal, bootstrap instance juga gagal. Keberhasilan ditandai dengan kode keluar nol (0). Kode keluar lainnya menunjukkan instance bootstrap gagal.

Anda dapat membedakan antara running head dan compute node. Sumber `/etc/parallelcluster/cfnconfig` file dan evaluasi variabel `cfn_node_type` lingkungan yang memiliki nilai `""` dan `MasterServer` `"ComputeFleet"` untuk node head dan compute, masing-masing.

```
#!/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
```

Konfigurasi

Pengaturan konfigurasi berikut digunakan untuk menentukan tindakan dan argumen pra-instal dan pasca-instal.

```
# 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
```

Pendapat

Dua argumen pertama — \$0 dan \$1 — dicadangkan untuk nama skrip dan url.

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

Contoh

Langkah-langkah berikut membuat skrip pasca-instal sederhana yang menginstal paket R dalam sebuah cluster.

1. Membuat skrip.

```
#!/bin/bash

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

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

2. Unggah skrip dengan izin yang benar ke Amazon S3. Jika izin baca publik tidak sesuai untuk Anda, gunakan salah satu [s3_read_resource](#) atau [s3_read_write_resource](#) parameter untuk memberikan akses. Untuk informasi selengkapnya, lihat [Bekerja dengan Amazon S3](#).

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

Important

Jika skrip diedit di Windows, akhiran baris harus diubah dari CRLF ke LF sebelum skrip diunggah ke Amazon S3.

3. Perbarui AWS ParallelCluster konfigurasi untuk menyertakan tindakan pasca-instal baru.

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

Jika bucket tidak memiliki izin baca publik, gunakan s3 sebagai protokol URL.

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

4. Luncurkan cluster.

```
$ pcluster create mycluster
```

5. Verifikasi output.

```
$ 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
```

Bekerja dengan Amazon S3

Untuk memberikan izin sumber daya kluster untuk mengakses bucket Amazon S3, tentukan bucket ARNs di [s3_read_resource](#) dan [s3_read_write_resource](#) parameter dalam konfigurasi. AWS ParallelCluster Untuk informasi selengkapnya tentang mengontrol akses dengan AWS ParallelCluster, lihat [AWS Identity and Access Management peran dalam 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/*
```

Kedua parameter menerima salah satu * atau Amazon S3 ARN yang valid. Untuk informasi tentang menentukan Amazon ARNs S3, lihat format [Amazon S3 ARN](#) di. Referensi Umum AWS

Contoh

Contoh berikut memberi Anda akses baca ke objek apa pun di bucket Amazon S3 my_corporate_bucket.

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

Contoh berikut ini memberi Anda akses baca ke bucket, tetapi tidak memungkinkan Anda membaca item dari ember.

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

Contoh terakhir ini memberi Anda akses baca ke bucket dan item yang disimpan di bucket.

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

Berkeja dengan Instans Spot

AWS ParallelCluster menggunakan Instans Spot jika konfigurasi cluster telah menetapkan `cluster_type` = spot. Instans Spot lebih hemat biaya daripada Instans Sesuai Permintaan, tetapi mungkin akan terganggu. Efek interupsi bervariasi tergantung pada penjadwal spesifik yang digunakan. Mungkin bermanfaat untuk memanfaatkan pemberitahuan interupsi Instans Spot, yang memberikan peringatan dua menit sebelum Amazon EC2 harus menghentikan atau menghentikan Instans Spot Anda. Untuk informasi selengkapnya, lihat [Interupsi Instans Spot](#) di EC2 Panduan Pengguna Amazon. Bagian berikut menjelaskan tiga skenario di mana Instans Spot dapat diinterupsi.

Note

Menggunakan Instans Spot mengharuskan peran `AWSServiceRoleForEC2Spot` terkait layanan ada di akun Anda. Untuk membuat peran ini di akun Anda menggunakan AWS CLI, jalankan perintah berikut:

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

Untuk informasi selengkapnya, lihat [Peran terkait layanan untuk permintaan Instans Spot](#) di EC2 Panduan Pengguna Amazon.

Skenario 1: Instans Spot tanpa pekerjaan yang berjalan terganggu

Ketika interupsi ini terjadi, AWS ParallelCluster coba ganti instance jika antrian penjadwal memiliki pekerjaan yang tertunda yang memerlukan instance tambahan, atau jika jumlah instance aktif lebih rendah dari pengaturan. `initial_queue_size` Jika tidak AWS ParallelCluster dapat menyediakan instance baru, maka permintaan untuk instance baru diulang secara berkala.

Skenario 2: Instance Spot yang menjalankan pekerjaan node tunggal terputus

Perilaku interupsi ini tergantung pada penjadwal yang digunakan.

Slurm

Pekerjaan gagal dengan kode negara bagian `NODE_FAIL`, dan pekerjaan tersebut diminta kembali (kecuali `--no-requeue` ditentukan saat pekerjaan diserahkan). Jika node adalah node statis, itu diganti. Jika node adalah node dinamis, node dihentikan dan diatur ulang. Untuk informasi selengkapnya tentang `sbatch`, termasuk `--no-requeue` parameter, lihat [sbatch](#) di dokumentasi Slurm.

Note

Perilaku ini berubah di AWS ParallelCluster versi 2.9.0. Versi sebelumnya menghentikan pekerjaan dengan kode status `NODE_FAIL` dan node telah dihapus dari antrian penjadwal.

SGE

Note

Ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Pekerjaan dihentikan. Jika pekerjaan telah mengaktifkan flag `rerun` ulang (menggunakan salah satu atau `qalter -r yes`) `qsub -r yes` atau antrean memiliki `rerun` konfigurasi yang disetel ke `TRUE`, maka pekerjaan dijadwalkan ulang. Instance komputasi dihapus dari antrian scheduler. Perilaku ini berasal dari parameter konfigurasi SGE ini:

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

Torque

Note

Ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Pekerjaan dihapus dari sistem dan node dihapus dari penjadwal. Pekerjaannya tidak dijalankan kembali. Jika beberapa pekerjaan berjalan pada instance saat terputus, Torque mungkin habis selama penghapusan node. Kesalahan mungkin ditampilkan dalam file [sqswatcher](#) log. Ini tidak memengaruhi logika penskalaan, dan pembersihan yang tepat dilakukan dengan percobaan ulang berikutnya.

Skenario 3: Instans Spot yang menjalankan pekerjaan multi-node terganggu

Perilaku interupsi ini tergantung pada penjadwal yang digunakan.

Slurm

Pekerjaan gagal dengan kode negara bagian `NODE_FAIL`, dan pekerjaan tersebut diminta kembali (kecuali `--no-requeue` ditentukan saat pekerjaan diajukan). Jika node adalah node statis, itu diganti. Jika node adalah node dinamis, node dihentikan dan diatur ulang. Node lain yang menjalankan pekerjaan yang dihentikan mungkin dialokasikan ke pekerjaan tertunda lainnya, atau diperkecil setelah [scaledown_idletime](#) waktu yang dikonfigurasi berlalu.

Note

Perilaku ini berubah di AWS ParallelCluster versi 2.9.0. Versi sebelumnya menghentikan pekerjaan dengan kode status `NODE_FAIL` dan node telah dihapus dari antrian penjadwal. Node lain yang menjalankan pekerjaan yang dihentikan mungkin diperkecil setelah [scaledown_idletime](#) waktu yang dikonfigurasi berlalu.

SGE

Note

Ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Pekerjaan tidak dihentikan dan terus berjalan di node yang tersisa. Node komputasi dihapus dari antrian penjadwal, tetapi akan muncul di daftar host sebagai simpul yatim piatu dan tidak tersedia.

Pengguna harus menghapus pekerjaan ketika ini terjadi (`qdel <jobid>`). Node masih ditampilkan di daftar host (`qhost`), meskipun ini tidak mempengaruhi AWS ParallelCluster. Untuk menghapus host dari daftar, jalankan perintah berikut setelah mengganti instance.

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

Torque

Note

Ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Pekerjaan dihapus dari sistem dan node dihapus dari penjadwal. Pekerjaannya tidak dijalankan kembali. Jika beberapa pekerjaan berjalan pada instance saat terputus, Torque mungkin habis selama penghapusan node. Kesalahan mungkin ditampilkan dalam file [sqswatcher](#) log. Ini tidak memengaruhi logika penskalaan, dan pembersihan yang tepat dilakukan dengan percobaan ulang berikutnya.

Untuk informasi selengkapnya tentang Instans Spot, lihat [Instans Spot](#) di EC2 Panduan Pengguna Amazon.

AWS Identity and Access Management peran dalam AWS ParallelCluster

AWS ParallelCluster menggunakan peran AWS Identity and Access Management (IAM) untuk Amazon EC2 untuk mengaktifkan instans mengakses AWS layanan untuk penerapan dan pengoperasian kluster. Secara default, peran IAM untuk Amazon EC2 dibuat saat cluster dibuat. Ini berarti bahwa pengguna yang membuat cluster harus memiliki tingkat izin yang sesuai, seperti yang dijelaskan di bagian berikut.

AWS ParallelCluster menggunakan beberapa AWS layanan untuk menyebarkan dan mengoperasikan cluster. Lihat daftar lengkap di bagian [AWS Layanan yang digunakan di AWS ParallelCluster](#) bagian.

Anda dapat melacak perubahan pada contoh kebijakan dalam [AWS ParallelCluster dokumentasi pada GitHub](#).

Topik

- [Pengaturan default untuk pembuatan cluster](#)
- [Menggunakan peran IAM yang ada untuk Amazon EC2](#)
- [AWS ParallelCluster contoh contoh dan kebijakan pengguna](#)

Pengaturan default untuk pembuatan cluster

Saat Anda menggunakan pengaturan default untuk pembuatan kluster, peran IAM default untuk Amazon EC2 dibuat oleh cluster. Pengguna yang membuat cluster harus memiliki tingkat izin yang tepat untuk membuat semua sumber daya yang diperlukan untuk meluncurkan cluster. Ini termasuk membuat peran IAM untuk Amazon EC2. Biasanya, pengguna harus memiliki izin kebijakan AdministratorAccessterkelola saat menggunakan pengaturan default. Untuk informasi tentang kebijakan terkelola, lihat [kebijakan AWS terkelola](#) di Panduan Pengguna IAM.

Menggunakan peran IAM yang ada untuk Amazon EC2

Sebagai pengganti pengaturan default, Anda dapat menggunakan yang sudah ada [ec2_iam_role](#) saat membuat kluster, tetapi Anda harus menentukan kebijakan dan peran IAM sebelum mencoba meluncurkan kluster. Biasanya, Anda memilih peran IAM yang ada untuk Amazon EC2 untuk meminimalkan izin yang diberikan kepada pengguna saat mereka meluncurkan cluster.

[AWS ParallelCluster contoh contoh dan kebijakan pengguna](#) Termasuk izin minimum yang diperlukan oleh AWS ParallelCluster dan fitur-fiturnya. Anda harus membuat kebijakan dan peran sebagai kebijakan individu di IAM dan kemudian melampirkan peran dan kebijakan ke sumber daya yang sesuai. Beberapa kebijakan peran mungkin menjadi besar dan menyebabkan kesalahan kuota. Untuk informasi selengkapnya, lihat [Memecahkan masalah ukuran kebijakan IAM](#). Dalam kebijakan, ganti `<REGION><AWS_ACCOUNT_ID>`, dan string serupa dengan nilai yang sesuai.

Jika maksud Anda adalah menambahkan kebijakan tambahan ke setelan default untuk node kluster, sebaiknya Anda meneruskan kebijakan IAM kustom tambahan dengan [additional_iam_policies](#) setelan alih-alih menggunakan pengaturan. [ec2_iam_role](#)

AWS ParallelCluster contoh contoh dan kebijakan pengguna

Contoh kebijakan berikut mencakup Amazon Resource Names (ARNs) untuk sumber daya. Jika Anda bekerja di partisi AWS GovCloud (US) atau AWS China, ARNs harus diubah. Secara khusus, mereka harus diubah dari “arn:aws” menjadi “arn:aws-us-gov” untuk AWS GovCloud (US) partisi atau “arn:aws-cn” untuk partisi Tiongkok. AWS Untuk informasi selengkapnya, lihat [Nama Sumber Daya Amazon \(ARNs\) di AWS GovCloud \(US\) Wilayah](#) di Panduan AWS GovCloud (US) Pengguna dan [ARNs untuk AWS layanan di Tiongkok](#) dalam AWS Memulai layanan di Tiongkok.

Kebijakan ini mencakup izin minimum yang saat ini diperlukan oleh AWS ParallelCluster, fitur-fiturnya, dan sumber dayanya. Beberapa kebijakan peran mungkin menjadi besar dan menyebabkan kesalahan kuota. Untuk informasi selengkapnya, lihat [Memecahkan masalah ukuran kebijakan IAM](#).

Topik

- [ParallelClusterInstancePolicy menggunakan SGE, Slurm, atau Torque](#)
- [ParallelClusterInstancePolicy menggunakan awsbatch](#)
- [ParallelClusterUserPolicy menggunakan Slurm](#)
- [ParallelClusterUserPolicy menggunakan SGE atau Torque](#)
- [ParallelClusterUserPolicy menggunakan awsbatch](#)
- [ParallelClusterLambdaPolicy menggunakan SGE, Slurm, atau Torque](#)
- [ParallelClusterLambdaPolicy menggunakan awsbatch](#)
- [ParallelClusterUserPolicy untuk pengguna](#)

ParallelClusterInstancePolicy menggunakan SGE, Slurm, atau Torque

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk 2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS

Topik

- [ParallelClusterInstancePolicy menggunakan Slurm](#)
- [ParallelClusterInstancePolicy menggunakan SGE atau Torque](#)

ParallelClusterInstancePolicy menggunakan Slurm

Contoh berikut menetapkan ParallelClusterInstancePolicy penggunaan Slurm sebagai penjadwal.

```
{
  "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 menggunakan SGE atau Torque

Contoh berikut menetapkan ParallelClusterInstancePolicy penggunaan SGE atau Torque sebagai penjadwal.

Note

Kebijakan ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

```

{
  "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 menggunakan awsbatch

Contoh berikut menetapkan ParallelClusterInstancePolicy penggunaan awsbatch sebagai penjadwal. Anda harus menyertakan kebijakan yang sama yang ditetapkan ke BatchUserRole yang ditentukan dalam tumpukan AWS Batch AWS CloudFormation bersarang. BatchUserRoleARN disediakan sebagai output tumpukan. Dalam contoh ini, “**<RESOURCES S3 BUCKET>**” adalah nilai [cluster_resource_bucket](#) pengaturan; jika tidak ditentukan maka “**<RESOURCES S3 BUCKET>**” [cluster_resource_bucket](#) adalah “parallelcluster-*”. Contoh berikut adalah ikhtisar izin yang diperlukan:

```
{
  "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 menggunakan Slurm

Contoh berikut menetapkan `ParallelClusterUserPolicy`, menggunakan Slurm sebagai penjadwal. Dalam contoh ini, “**<RESOURCES S3 BUCKET>**” adalah nilai [cluster_resource_bucket](#) pengaturan; jika tidak ditentukan maka “**<RESOURCES S3 BUCKET>**” [cluster_resource_bucket](#) adalah “parallelcluster-*”.

Note

Jika Anda menggunakan peran kustom `ec2_iam_role = <role_name>`, Anda harus mengubah sumber daya IAM untuk menyertakan nama peran tersebut dari:

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

Ke:

"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 menggunakan SGE atau Torque

Note

Bagian ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Contoh berikut menetapkan `ParallelClusterUserPolicy`, menggunakan SGE atau Torque sebagai penjadwal. Dalam contoh ini, “`<RESOURCES S3 BUCKET>`” adalah nilai [cluster_resource_bucket](#) pengaturan; jika tidak ditentukan maka “`<RESOURCES S3 BUCKET>`” [cluster_resource_bucket](#) adalah “`parallelcluster-*`”.

Note

Jika Anda menggunakan peran kustom [ec2_iam_role](#) = `<role_name>`, Anda harus mengubah sumber daya IAM untuk menyertakan nama peran tersebut dari:

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

Contoh berikut menetapkan ParallelClusterUserPolicy penggunaan awsbatch sebagai penjadwal. Dalam contoh ini, “**<RESOURCES S3 BUCKET>**” adalah nilai [cluster_resource_bucket](#) pengaturan; jika tidak ditentukan maka “**<RESOURCES S3 BUCKET>**” [cluster_resource_bucket](#) adalah “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 menggunakan SGE, Slurm, atau Torque

Contoh berikut menetapkan ParallelClusterLambdaPolicy, menggunakan, SGE, Slurm, atau Torque sebagai penjadwal.

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

```

{
  "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 menggunakan **awsbatch**

Contoh berikut menetapkan ParallelClusterLambdaPolicy penggunaan awsbatch sebagai penjadwal.

```
{
  "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 untuk pengguna

Contoh berikut menetapkan ParallelClusterUserPolicy untuk pengguna yang tidak perlu membuat atau memperbarui cluster. Perintah berikut didukung.

- [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": "*"
}
]
}

```

Penjadwal didukung oleh AWS ParallelCluster

AWS ParallelCluster mendukung beberapa penjadwal, diatur menggunakan pengaturan. [scheduler](#)

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk

2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS

Topik

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

Son of Grid Engine (**sge**)

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk 2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS

AWS ParallelCluster versi 2.11.4 dan sebelumnya menggunakan Son of Grid Engine 8.1.9.

Slurm Workload Manager (**slurm**)

AWS ParallelCluster versi 2.11.9 menggunakan Slurm 20.11.9. Untuk informasi tentang Slurm, lihat <https://slurm.schedmd.com/>. Untuk unduhan, lihat <https://github.com/SchedMD/slurm/tags>. Untuk kode sumber, lihat <https://github.com/SchedMD/slurm>.

Important

AWS ParallelCluster diuji dengan parameter Slurm konfigurasi, yang disediakan secara default. Setiap perubahan yang Anda buat pada parameter Slurm konfigurasi ini dilakukan dengan risiko Anda sendiri. Mereka didukung hanya atas dasar upaya terbaik.

AWS ParallelCluster versi (s)	SlurmVersi yang didukung
2.11.7, 2.11.8, 2.11.9	20.11.9
2.11.4 hingga 2.11.6	20.11.8
2.11.0 hingga 2.11.3	20.11.7
2.10.4	20.02.7
2.9.0 hingga 2.10.3	20.02.4
2.6 hingga 2.8.1	19.05.5
2.5.0, 2.5.1	19.05.3-2
2.3.1 hingga 2.4.1	18.08.6-2
sebelum 2.3.1	16.05.3-1

Mode antrian ganda

AWS ParallelCluster versi 2.9.0 memperkenalkan beberapa mode antrian. Beberapa mode antrian didukung ketika [scheduler](#) diatur ke `slurm` dan [queue_settings](#) pengaturan ditentukan. Mode ini memungkinkan jenis instance yang berbeda untuk hidup berdampingan di node komputasi. Sumber daya komputasi yang berisi berbagai jenis instance dapat meningkatkan atau menurunkan skala sesuai kebutuhan. [Dalam mode antrian, hingga lima \(5\) antrian didukung, dan setiap \[queue\]bagian dapat merujuk hingga tiga \(3\) bagian. \[compute_resource\]](#) Masing-masing [\[queue\]bagian](#) ini adalah partisi diSlurm Workload Manager. Untuk informasi selengkapnya, lihat [Slurmpanduan untuk beberapa mode antrian](#) dan [Beberapa mode antrian tutorial](#).

Setiap [\[compute_resource\]bagian](#) dalam antrian harus memiliki jenis instance yang berbeda, dan masing-masing `[compute_resource]` dibagi lagi menjadi node statis dan dinamis. Node statis untuk masing-masing `[compute_resource]` diberi nomor dari 1 ke nilai [min_count](#). Node dinamis untuk masing-masing `[compute_resource]` diberi nomor dari satu (1) ke [\(max_count-min_count\)](#). Misalnya, jika `min_count` adalah 2 dan `max_count` 10, node dinamis untuk itu `[compute_resource]` diberi nomor dari satu (1) hingga delapan (8). Kapan saja, bisa ada antara nol (0) dan jumlah maksimal node dinamis dalam `a[compute_resource]`.

Instance yang diluncurkan ke armada komputasi ditetapkan secara dinamis. Untuk membantu mengelola ini, nama host dibuat untuk setiap node. Format nama host adalah sebagai berikut:

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

- `$QUEUE` adalah nama antrian. Misalnya, jika bagian dimulai [queue *queue-name*] maka "`$QUEUE`" adalah "*queue-name*".
- `$STATDYN` adalah `st` untuk node statis atau `dy` untuk node dinamis.
- `$INSTANCE_TYPE` adalah tipe instance untuk [compute_resource], dari [instance_type](#) pengaturan.
- `$NODENUM` adalah jumlah node. `$NODENUM` adalah antara satu (1) dan nilai [min_count](#) untuk node statis dan antara satu (1) dan ([max_count](#)-min_count) untuk node dinamis.

Nama host dan nama domain yang sepenuhnya memenuhi syarat (FQDN) dibuat menggunakan zona yang dihosting Amazon Route 53. FQDN adalah `$HOSTNAME.$CLUSTERNAME.pcluster`, di mana `$CLUSTERNAME` adalah nama [\[cluster\]bagian](#) yang digunakan untuk cluster.

Untuk mengonversi konfigurasi Anda ke mode antrian, gunakan [pcluster-config convert](#) perintah. Ini menulis konfigurasi yang diperbarui dengan satu [\[queue\]bagian](#) bernama [queue compute]. Antrian itu berisi satu [\[compute_resource\]bagian](#) yang diberi nama [compute_resource default]. Pengaturan [queue compute] dan [compute_resource default] memiliki migrasi dari [\[cluster\]bagian](#) yang ditentukan.

Slurmpanduan untuk beberapa mode antrian

AWS ParallelCluster versi 2.9.0 memperkenalkan beberapa mode antrian dan arsitektur penskalaan baru untuk (). Slurm Workload Manager Slurm

Bagian berikut memberikan gambaran umum tentang penggunaan Slurm cluster dengan arsitektur penskalaan yang baru diperkenalkan.

Gambaran Umum

Arsitektur penskalaan baru didasarkan pada Slurm [Panduan Penjadwalan Cloud](#) dan plugin hemat daya. Untuk informasi selengkapnya tentang plugin hemat daya, lihat [Panduan Penghematan Slurm Daya](#). Dalam arsitektur baru, sumber daya yang berpotensi tersedia untuk cluster biasanya telah ditentukan sebelumnya dalam Slurm konfigurasi sebagai node cloud.

Siklus hidup simpul awan

Sepanjang siklus hidupnya, node cloud memasukkan beberapa jika tidak semua status berikut: `POWER_SAVING`, `POWER_UP` (`pow_up`), `()`, dan `ALLOCATED POWER_DOWN` (`allocpow_dn`). Dalam beberapa kasus, node cloud mungkin memasuki `OFFLINE` status. Daftar berikut merinci beberapa aspek status ini dalam siklus hidup node cloud.

- Sebuah simpul dalam `POWER_SAVING` keadaan muncul dengan `~` akhiran (misalnya `idle~`) di `sinfo`. Dalam keadaan ini, tidak ada EC2 instance yang mendukung node. Namun, masih Slurm dapat mengalokasikan pekerjaan ke node.
- Sebuah simpul yang beralih ke `POWER_UP` status muncul dengan `#` akhiran (misalnya `idle#`) di `sinfo`.
- Ketika Slurm mengalokasikan pekerjaan ke node dalam `POWER_SAVING` keadaan, node secara otomatis mentransfer ke keadaan `POWER_UP`. Jika tidak, node dapat ditempatkan dalam `POWER_UP` keadaan secara manual menggunakan `scontrol update nodename=nodename state=power_up` perintah. Pada tahap ini, `ResumeProgram` dipanggil, dan EC2 instance diluncurkan dan dikonfigurasi untuk mendukung node. `POWER_UP`.
- Node yang saat ini tersedia untuk digunakan muncul tanpa akhiran apa pun (misalnya `idle`) di `sinfo`. Setelah node diatur dan telah bergabung dengan cluster, itu menjadi tersedia untuk menjalankan pekerjaan. Pada tahap ini, node dikonfigurasi dengan benar dan siap digunakan. Sebagai aturan umum, kami merekomendasikan bahwa jumlah instance sama dengan jumlah node yang tersedia. EC2 Dalam kebanyakan kasus, node statis selalu tersedia setelah cluster dibuat.
- Node yang bertransisi ke `POWER_DOWN` status muncul dengan `%` akhiran (misalnya `idle%`) di `sinfo`. Node dinamis secara otomatis memasuki `POWER_DOWN` status setelahnya `scaledown_idletime`. Sebaliknya, node statis dalam banyak kasus tidak dimatikan. Namun, node dapat ditempatkan dalam `POWER_DOWN` keadaan secara manual menggunakan `scontrol update nodename=nodename state=powering_down` perintah. Dalam keadaan ini, instance yang terkait dengan node dihentikan, dan node diatur ulang kembali ke `POWER_SAVING` status ke future use after `scaledown_idletime`. `scaledown-idletime` Pengaturan disimpan ke Slurm konfigurasi sebagai `SuspendTimeout` pengaturan.
- Node yang offline muncul dengan `*` akhiran (misalnya `down*`) di `sinfo`. Sebuah node akan offline jika Slurm controller tidak dapat menghubungi node atau jika node statis dinonaktifkan dan instance backing dihentikan.

Sekarang perhatikan status node yang ditunjukkan dalam `sinfo` contoh berikut.

```
$ 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]
```

efa-st-c5n18xlarge-1Node spot-st-t2large-[1-2] dan sudah memiliki instance pendukung yang disiapkan dan tersedia untuk digunakan. ondemand-dy-c52xlarge-[1-2]Node berada dalam POWER_UP keadaan, dan mereka akan tersedia dalam beberapa menit. gpu-dy-g38xlarge-1Node dalam POWER_DOWN keadaan, dan akan bertransisi ke POWER_SAVING keadaan setelah [scaledown_idletime](#) (default ke 120 detik).

Semua node lain dalam POWER_SAVING keadaan tanpa EC2 instance yang mendukungnya.

Bekerja dengan node yang tersedia

Sebuah node yang tersedia didukung oleh sebuah EC2 instance. Secara default, nama node dapat digunakan untuk langsung SSH ke dalam instance (misalnyassh efa-st-c5n18xlarge-1). Alamat IP pribadi dari instance dapat diambil menggunakan `scontrol show nodes nodename` perintah dan memeriksa NodeAddr bidang. Untuk node yang tidak tersedia, NodeAddr bidang tidak boleh menunjuk ke EC2 instance yang sedang berjalan. Sebaliknya, itu harus sama dengan nama node.

Status pekerjaan dan pengajuan

Pekerjaan yang dikirimkan dalam banyak kasus segera dialokasikan ke node dalam sistem, atau ditempatkan di pending jika semua node dialokasikan.

Jika node yang dialokasikan untuk pekerjaan menyertakan node apa pun dalam suatu POWER_SAVING keadaan, pekerjaan dimulai denganCF, atau CONFIGURING status. Pada saat ini, pekerjaan menunggu node di POWER_SAVING negara bagian untuk beralih ke POWER_UP status dan menjadi tersedia.

Setelah semua node yang dialokasikan untuk pekerjaan tersedia, pekerjaan memasuki status RUNNING (R).

Secara default, semua pekerjaan dikirimkan ke antrian default (dikenal sebagai partisi diSlurm). Ini ditandai dengan * akhiran setelah nama antrian. Anda dapat memilih antrian menggunakan opsi pengiriman `-p` pekerjaan.

Semua node dikonfigurasi dengan fitur berikut, yang dapat digunakan dalam perintah pengiriman pekerjaan:

- Jenis instance (misalnya `c5.xlarge`)
- Tipe simpul (Ini adalah salah satu `dynamic` atau `static`.)

Anda dapat melihat semua fitur yang tersedia untuk node tertentu dengan menggunakan `scontrol show nodes nodename` perintah dan memeriksa AvailableFeatures daftar.

Pertimbangan lain adalah pekerjaan. Pertama pertimbangkan keadaan awal cluster, yang dapat Anda lihat dengan menjalankan `sinfo` perintah.

```
$ 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]
```

Perhatikan bahwa `spot` adalah antrian default. Ini ditunjukkan oleh * sufiks.

Kirim pekerjaan ke satu node statis ke antrian default (`spot`).

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

Kirim pekerjaan ke satu node dinamis ke EFA antrian.

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

Kirim pekerjaan ke delapan (8) `c5.2xlarge` node dan dua (2) `t2.xlarge` node ke `ondemand` antrian.

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

Kirim pekerjaan ke satu node GPU ke gpu antrian.

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

Sekarang pertimbangkan keadaan pekerjaan menggunakan squeue perintah.

```
$ 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

Pekerjaan 7 dan 8 (dalam spot dan efa antrian) sudah berjalan (R). Pekerjaan 12 dan 13 masih mengkonfigurasi (CF), mungkin menunggu instance tersedia.

```
# 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

Status dan fitur node

Dalam kebanyakan kasus, status node sepenuhnya dikelola oleh AWS ParallelCluster sesuai dengan proses spesifik dalam siklus hidup node cloud yang dijelaskan sebelumnya dalam topik ini.

Namun, AWS ParallelCluster juga menggantikan atau mengakhiri node yang tidak sehat di DOWN dan DRAINED status dan node yang memiliki instance dukungan yang tidak sehat. Untuk informasi selengkapnya, lihat [clustermgtd](#).

Status partisi

AWS ParallelCluster mendukung status partisi berikut. SlurmPartisi adalah antrian di AWS ParallelCluster.

- UP: Menunjukkan bahwa partisi dalam keadaan aktif. Ini adalah keadaan default partisi. Dalam keadaan ini, semua node di partisi aktif dan tersedia untuk digunakan.
- INACTIVE: Menunjukkan bahwa partisi dalam keadaan tidak aktif. Dalam keadaan ini, semua instance backing node dari partisi tidak aktif dihentikan. Instance baru tidak diluncurkan untuk node di partisi yang tidak aktif.

pcluster mulai dan berhenti

Ketika [pcluster stop](#) dijalankan, semua partisi ditempatkan di INACTIVE negara bagian, dan AWS ParallelCluster proses menjaga partisi dalam keadaan. INACTIVE

Ketika [pcluster start](#) dijalankan, semua partisi awalnya ditempatkan di UP negara bagian. Namun, AWS ParallelCluster proses tidak menjaga partisi dalam UP keadaan. Anda perlu mengubah status partisi secara manual. Semua node statis menjadi tersedia setelah beberapa menit. Perhatikan bahwa menyetel partisi ke UP tidak menyalakan kapasitas dinamis apa pun. Jika [initial_count](#) lebih besar dari [max_count](#), maka [initial_count](#) mungkin tidak puas ketika status partisi diubah ke UP status.

Kapan [pcluster start](#) dan [pcluster stop](#) sedang berjalan, Anda dapat memeriksa status cluster dengan menjalankan [pcluster status](#) perintah dan memeriksa `ComputeFleetStatus`. Berikut daftar kemungkinan status:

- STOP_REQUESTED: [pcluster stop](#) Permintaan dikirim ke cluster.
- STOPPING: pcluster Proses saat ini menghentikan cluster.
- STOPPED: pcluster Proses menyelesaikan proses penghentian, semua partisi dalam INACTIVE keadaan, dan semua instance komputasi dihentikan.
- START_REQUESTED: [pcluster start](#) Permintaan dikirim ke cluster.
- STARTING: pcluster Proses saat ini memulai cluster

- **RUNNING:** `pcluster` Proses menyelesaikan proses awal, semua partisi dalam UP keadaan, dan node statis akan tersedia setelah beberapa menit.

Kontrol manual pada antrian

Dalam beberapa kasus, Anda mungkin ingin memiliki beberapa kontrol manual atas node atau antrian (dikenal sebagai partisi diSlurm) dalam sebuah cluster. Anda dapat mengelola node dalam cluster melalui prosedur umum berikut.

- Nyalakan node dinamis dalam **POWER_SAVING** status: Jalankan `scontrol update nodename=nodename state=power_up` perintah atau kirimkan `sleep 1` pekerjaan placeholder yang meminta sejumlah node tertentu dan mengandalkan Slurm untuk meningkatkan jumlah node yang diperlukan.
- Matikan node dinamis sebelumnya [scaledown_idletime](#): Atur node dinamis DOWN dengan `scontrol update nodename=nodename state=down` perintah. AWS ParallelCluster secara otomatis mengakhiri dan me-reset node dinamis yang jatuh. Secara umum, kami tidak menyarankan pengaturan node untuk **POWER_DOWN** langsung menggunakan `scontrol update nodename=nodename state=power_down` perintah. Ini karena AWS ParallelCluster secara otomatis menangani proses power down. Tidak diperlukan intervensi manual. Oleh karena itu, kami menyarankan Anda mencoba mengatur node DOWN kapan pun memungkinkan.
- Nonaktifkan antrian (partisi) atau hentikan semua node statis di partisi tertentu: Tetapkan antrian tertentu **INACTIVE** dengan perintah. `scontrol update partition=queue name state=inactive` Melakukan hal ini mengakhiri semua instance backing node di partisi.
- Aktifkan antrian (partisi): Tetapkan antrian tertentu **INACTIVE** dengan perintah. `scontrol update partition=queue name state=up`

Perilaku dan penyesuaian penskalaan

Berikut adalah contoh alur kerja penskalaan normal:

- Penjadwal menerima pekerjaan yang membutuhkan dua node.
- Penjadwal mentransisikan dua node ke **POWER_UP** status, dan memanggil `ResumeProgram` dengan nama node (misalnya `queue1-dy-c5xlarge-[1-2]`).
- `ResumeProgram` meluncurkan dua EC2 instance dan menetapkan alamat IP pribadi dan nama host dari `queue1-dy-c5xlarge-[1-2]`, menunggu `ResumeTimeout` (periode default adalah 60 menit (1 jam)) sebelum mengatur ulang node.

- Instans dikonfigurasi dan bergabung dengan cluster. Job mulai berjalan pada instance.
- Job sudah selesai.
- Setelah konfigurasi SuspendTime telah berlalu (yang diatur ke [scaledown_idletime](#)), instance dimasukkan ke POWER_SAVING status oleh penjadwal. Scheduler menempatkan queue1-dy-c5xlarge-[1-2] ke POWER_DOWN status dan panggilan SuspendProgram dengan nama node.
- SuspendProgram disebut untuk dua node. Node tetap dalam POWER_DOWN keadaan, misalnya, dengan tetap idle% selama a SuspendTimeout (periode default adalah 120 detik (2 menit)). Setelah clustermgtd mendeteksi bahwa node dimatikan, itu mengakhiri instance dukungan. Kemudian, mengkonfigurasi queue1-dy-c5xlarge-[1-2] ke dalam keadaan idle dan mengatur ulang alamat IP pribadi dan nama host sehingga mereka dapat diaktifkan untuk pekerjaan masa depan lagi.

Sekarang, jika ada yang salah dan instance untuk node tertentu tidak dapat diluncurkan karena alasan tertentu, maka hal berikut terjadi.

- Scheduler menerima pekerjaan yang membutuhkan dua node.
- Scheduler menempatkan dua node cloud bursting ke POWER_UP status dan memanggil ResumeProgram dengan nama node, (misalnya). queue1-dy-c5xlarge-[1-2]
- ResumeProgram meluncurkan hanya satu (1) EC2 instance dan mengkonfigurasi queue1-dy-c5xlarge-1, tetapi gagal meluncurkan instance untuk. queue1-dy-c5xlarge-2
- queue1-dy-c5xlarge-1 tidak akan terpengaruh dan akan online setelah mencapai POWER_UP negara bagian.
- queue1-dy-c5xlarge-2 ditempatkan dalam POWER_DOWN keadaan, dan pekerjaan diminta ulang secara otomatis karena Slurm mendeteksi kegagalan node.
- queue1-dy-c5xlarge-2 menjadi tersedia setelah SuspendTimeout (defaultnya adalah 120 detik (2 menit)). Sementara itu, pekerjaan tersebut diminta ulang dan dapat mulai berjalan di node lain.
- Proses di atas diulang sampai pekerjaan dapat berjalan pada node yang tersedia tanpa terjadi kegagalan.

Ada dua parameter waktu yang dapat disesuaikan jika diperlukan.

- ResumeTimeout (defaultnya adalah 60 menit (1 jam)): ResumeTimeout mengontrol waktu Slurm menunggu sebelum meletakkan node status down.

- Mungkin berguna untuk memperpanjang ini jika proses pre/post instalasi Anda memakan waktu hampir selama itu.
- Ini juga merupakan waktu maksimum yang AWS ParallelCluster menunggu sebelum mengganti atau mengatur ulang node jika ada masalah. Menghitung node berhenti sendiri jika ada kesalahan yang terjadi selama peluncuran atau penyiapan. Selanjutnya, AWS ParallelCluster proses juga menggantikan node ketika melihat bahwa instance dihentikan.
- SuspendTimeout(defaultnya adalah 120 detik (2 menit)): SuspendTimeout mengontrol seberapa cepat node ditempatkan kembali ke sistem dan siap digunakan lagi.
- Yang lebih pendek SuspendTimeout berarti bahwa node akan diatur ulang lebih cepat, dan Slurm dapat mencoba meluncurkan instance lebih sering.
- SuspendTimeoutYang lebih lama membuat node yang gagal diatur ulang lebih lambat. Sementara itu, Slurm ban menggunakan node lain. Jika SuspendTimeout lebih dari beberapa menit, cobalah Slurm untuk menelusuri semua node dalam sistem. Yang lebih lama SuspendTimeout mungkin bermanfaat untuk sistem skala besar (lebih dari 1.000 node) untuk mengurangi stres Slurm dengan sering mengantri ulang pekerjaan yang gagal.
- Perhatikan bahwa SuspendTimeout tidak mengacu pada waktu AWS ParallelCluster menunggu untuk mengakhiri instance dukungan untuk sebuah node. Instance pendukung untuk power down node segera dihentikan. Proses penghentian biasanya selesai beberapa menit. Namun, selama waktu ini, node tetap dalam keadaan mati daya dan tidak tersedia untuk digunakan dalam penjadwal.

Log untuk arsitektur baru

Lit berikut berisi log kunci untuk arsitektur antrian ganda. Nama aliran log yang digunakan dengan Amazon CloudWatch Logs memiliki format `{hostname}.{instance_id}.{logIdentifier}`, di mana `logIdentifier` mengikuti nama log. Untuk informasi selengkapnya, lihat [Integrasi dengan 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)
```

Masalah umum dan cara men-debug:

Node yang gagal diluncurkan, dinyalakan, atau bergabung dengan cluster:

- Node dinamis:
 - Periksa ResumeProgram log untuk melihat apakah ResumeProgram pernah dipanggil dengan node. Jika tidak, periksa `slurmctld` log untuk menentukan apakah Slurm pernah mencoba memanggil ResumeProgram dengan node. Perhatikan bahwa izin yang salah ResumeProgram dapat menyebabkannya gagal secara diam-diam.
 - Jika ResumeProgram dipanggil, periksa untuk melihat apakah sebuah instance diluncurkan untuk node. Jika instance tidak dapat diluncurkan, harus ada pesan kesalahan yang jelas mengapa instance gagal diluncurkan.
 - Jika sebuah instance diluncurkan, mungkin ada beberapa masalah selama proses bootstrap. Temukan alamat IP pribadi dan ID instance yang sesuai dari ResumeProgram log dan lihat log bootstrap yang sesuai untuk instance tertentu di CloudWatch Log.
- Node statis:
 - Periksa `clustermgtd` log untuk melihat apakah instance diluncurkan untuk node. Jika tidak, harus ada kesalahan yang jelas tentang mengapa instance gagal diluncurkan.
 - Jika sebuah instance diluncurkan, ada beberapa masalah selama proses bootstrap. Temukan IP pribadi dan ID instance yang sesuai dari `clustermgtd` log dan lihat log bootstrap yang sesuai untuk instance tertentu di CloudWatch Log.

Node diganti atau dihentikan secara tak terduga, kegagalan node

- Node secara replaced/terminated tak terduga
 - Dalam kebanyakan kasus, `clustermgtd` menangani semua tindakan pemeliharaan node. Untuk memeriksa apakah `clustermgtd` diganti atau dihentikan node, periksa `clustermgtd` log.
 - Jika `clustermgtd` diganti atau dihentikan node, harus ada pesan yang menunjukkan alasan tindakan tersebut. Jika alasannya terkait penjadwal (misalnya, `simpulnyaDOWN`), periksa `slurmctld` log untuk lebih jelasnya. Jika alasannya EC2 terkait, gunakan alat untuk memeriksa status atau log untuk contoh itu. Misalnya, Anda dapat memeriksa apakah instans memiliki acara terjadwal atau pemeriksaan status EC2 kesehatan yang gagal.
 - Jika `clustermgtd` tidak mengakhiri node, periksa apakah node `computemgtd` dihentikan atau jika EC2 menghentikan instance untuk merebut kembali Instance Spot.
- Kegagalan simpul
 - Dalam kebanyakan kasus, pekerjaan secara otomatis diminta ulang jika node gagal. Lihat di `slurmctld` log untuk melihat mengapa pekerjaan atau node gagal dan analisis situasi dari sana.

Kegagalan saat mengganti atau menghentikan instance, kegagalan saat mematikan node

- Secara umum, `clustermgtd` menangani semua tindakan penghentian instance yang diharapkan. Lihat di `clustermgtd` log untuk melihat mengapa gagal mengganti atau mengakhiri node.
- Untuk node dinamis gagal [scaledown_idletime](#), lihat di `SuspendProgram` log untuk melihat apakah sebuah program `slurmctld` dengan node tertentu sebagai argumen. Catatan `SuspendProgram` tidak benar-benar melakukan tindakan tertentu. Sebaliknya, itu hanya log ketika dipanggil. Semua penghentian dan `NodeAddr` reset instans diselesaikan oleh `clustermgtd`. `Slurmmenempatkan` node ke dalam IDLE setelah `SuspendTimeout`.

Masalah lainnya

- AWS ParallelCluster tidak membuat alokasi pekerjaan atau keputusan penskalaan. Sederhana mencoba meluncurkan, menghentikan, dan memelihara sumber daya sesuai dengan Slurm instruksi.

Untuk masalah terkait alokasi pekerjaan, alokasi node, dan keputusan penskalaan, lihat `slurmctld` log untuk kesalahan.

Torque Resource Manager (**torque**)

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk 2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS

AWS ParallelCluster versi 2.11.4 dan sebelumnya menggunakan Torque Resource Manager 6.1.2. Untuk informasi lebih lanjut tentang Torque Resource Manager 6.1.2, lihat <http://docs.adaptivecomputing.com/torque/6-1-2/releaseNotes/torquerelnote.htm>. Untuk dokumentasi, lihat <http://docs.adaptivecomputing.com/torque/6-1-2/adminGuide/torque.htm>. Untuk kode sumber, lihat <https://github.com/adaptivecomputing/torque/tree/6.1.2>.

AWS ParallelCluster versi 2.4.0 dan sebelumnya menggunakan Torque Resource Manager 6.0.2. Untuk catatan rilis, lihat <http://docs.adaptivecomputing.com/torque/6-0-2/releaseNotes/torqueReleaseNotes6.0.2.pdf>. Untuk dokumentasi, lihat <http://docs.adaptivecomputing.com/torque/6-0-2/adminGuide/help.htm>. Untuk kode sumber, lihat <https://github.com/adaptivecomputing/torque/tree/6.0.2>.

AWS Batch (**awsbatch**)

Untuk informasi tentang AWS Batch, lihat [AWS Batch](#). Untuk dokumentasi, lihat [Panduan AWS Batch Pengguna](#).

AWS ParallelCluster Perintah CLI untuk AWS Batch

Ketika Anda menggunakan `awsbatch` scheduler, perintah AWS ParallelCluster CLI AWS Batch untuk secara otomatis diinstal di node AWS ParallelCluster kepala. CLI menggunakan operasi AWS Batch API dan mengizinkan operasi berikut:

- Kirim dan kelola pekerjaan.
- Pantau pekerjaan, antrian, dan host.
- Cerminkan perintah penjadwal tradisional.

⚠ Important

AWS ParallelCluster tidak mendukung pekerjaan GPU untuk AWS Batch. Untuk informasi lebih lanjut, lihat [pekerjaan GPU](#).

Topik

- [awsbsub](#)
- [awsbstat](#)
- [awsbout](#)
- [awsbkill](#)
- [awsbqueues](#)
- [awsbhosts](#)

awsbsub

Mengirimkan pekerjaan ke antrian pekerjaan 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 tidak mendukung pekerjaan GPU untuk AWS Batch. Untuk informasi lebih lanjut, lihat [pekerjaan GPU](#).

Argumen Posisi***command***

Mengirimkan pekerjaan (perintah yang ditentukan harus tersedia pada contoh komputasi) atau nama file yang akan ditransfer. Lihat juga `--command-file`.

arguments

(Opsional) Menentukan argumen untuk perintah atau perintah-file.

Argumen Bernama

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

Nama pekerjaan. Karakter pertama harus berupa huruf atau angka. Nama pekerjaan dapat berisi huruf (huruf besar dan kecil), angka, tanda hubung, dan garis bawah, dan panjangnya hingga 128 karakter.

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

Menentukan cluster untuk digunakan.

-cf, --command-file

Menunjukkan bahwa perintah adalah file yang akan ditransfer ke instance komputasi.

Default: Salah

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

Menentukan folder untuk digunakan sebagai direktori kerja pekerjaan ini. Jika direktori kerja tidak ditentukan, pekerjaan dijalankan di `job-<AWS_BATCH_JOB_ID>` subfolder direktori home pengguna. Anda dapat menggunakan parameter ini atau `--parent-working-dir` parameter.

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

Menentukan folder induk dari direktori kerja pekerjaan ini. Jika direktori kerja induk tidak ditentukan, itu default ke direktori home pengguna. Sebuah subfolder bernama `job-<AWS_BATCH_JOB_ID>` dibuat di direktori kerja induk. Anda dapat menggunakan parameter ini atau `--working-dir` parameter.

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

Menentukan file yang akan ditransfer ke contoh komputasi, di direktori kerja pekerjaan. Anda dapat menentukan beberapa parameter file input.

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

Menentukan jumlah v CPUs untuk cadangan untuk wadah. Ketika digunakan bersama-sama dengan `--nodes`, itu mengidentifikasi jumlah v CPUs untuk setiap node.

Default: 1

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

Menentukan batas keras memori (dalam MiB) untuk menyediakan pekerjaan. Jika pekerjaan Anda mencoba untuk melebihi batas memori yang ditentukan di sini, pekerjaan berakhir.

Default: 128

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

Menentukan daftar dipisahkan koma nama variabel lingkungan untuk mengekspor ke lingkungan kerja. Untuk mengekspor semua variabel lingkungan, tentukan 'semua'. Perhatikan bahwa daftar variabel lingkungan 'semua' tidak termasuk yang tercantum dalam `--env-blacklist` parameter, atau variabel yang dimulai dengan `AWS_*` awalan `PCLUSTER_*` atau.

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

Menentukan daftar dipisahkan koma nama variabel lingkungan untuk tidak mengekspor ke lingkungan kerja. Secara default, `HOME`, `PWD`, `USER`, `PATH`, `LD_LIBRARY_PATH`, `TERM`, dan tidak `TERMCAP` diekspor.

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

Menentukan jumlah kali untuk memindahkan pekerjaan ke `RUNNABLE` status. Anda dapat menentukan usaha antara 1 dan 10. Jika nilai percobaan lebih besar dari 1, pekerjaan akan dicoba lagi jika gagal, sampai telah pindah ke `RUNNABLE` status untuk jumlah yang ditentukan kali.

Default: 1

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

Menentukan durasi waktu dalam hitungan detik (diukur dari `startedAt` stempel waktu usaha pekerjaan) setelah itu AWS Batch mengakhiri pekerjaan Anda jika belum selesai. Nilai batas waktu harus minimal 60 detik.

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

Menentukan jumlah node untuk cadangan untuk pekerjaan itu. Tentukan nilai untuk parameter ini untuk mengaktifkan pengiriman paralel multi-node.

Note

Pekerjaan paralel multi-node tidak didukung saat [cluster_type](#) parameter disetel kespot.

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

Menunjukkan ukuran array. Anda dapat menentukan nilai antara 2 dan 10.000. Jika Anda menentukan properti array untuk suatu tugas, itu menjadi tugas array.

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

Menentukan daftar dependensi yang dipisahkan titik koma untuk pekerjaan. Sebuah pekerjaan dapat bergantung pada maksimal 20 pekerjaan. Anda dapat menentukan ketergantungan SEQUENTIAL tipe tanpa menentukan ID pekerjaan untuk pekerjaan array. Ketergantungan sekuensial memungkinkan setiap pekerjaan array anak untuk menyelesaikan secara berurutan, dimulai dari indeks 0. Anda juga dapat menentukan dependensi tipe N_TO_N dengan ID pekerjaan untuk pekerjaan array. Ketergantungan N_TO_N berarti bahwa setiap turunan indeks dari pekerjaan ini harus menunggu turunan indeks yang sesuai dari setiap dependensi selesai sebelum dapat dimulai. Sintaks untuk parameter ini adalah "joBid=<*stringstring*

awsbstat

Menampilkan pekerjaan yang dikirimkan dalam antrean pekerjaan klaster.

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

Argumen Posisi

job_ids

Menentukan daftar spasi-dipisahkan dari pekerjaan IDs untuk ditampilkan dalam output. Jika pekerjaan adalah array pekerjaan, semua pekerjaan anak ditampilkan. Jika satu pekerjaan diminta, itu ditampilkan dalam versi terperinci.

Argumen Bernama

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

Menunjukkan cluster yang akan digunakan.

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

Menentukan daftar status pekerjaan yang dipisahkan koma untuk disertakan.

Status pekerjaan default adalah “aktif.”. Nilai yang diterima adalah:

SUBMITTEDPENDING,,RUNNABLE,STARTING,RUNNING,SUCCEEDED,FAILED, danALL.

Default: “SUBMITTEDPENDING,RUNNABLE,,STARTING,RUNNING”

-e, --expand-children

Memperluas pekerjaan dengan anak-anak (baik array dan multi-node parallel).

Default: Salah

-d, --details

Menampilkan detail pekerjaan.

Default: Salah

awsbout

Menunjukkan output dari pekerjaan yang diberikan.

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

Argumen Posisi

job_id

Menentukan ID pekerjaan.

Argumen Bernama

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

Menunjukkan cluster yang akan digunakan.

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

Mendapat *HEAD* baris pertama dari output pekerjaan.

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

Mendapat <tail>baris terakhir dari output pekerjaan.

-s, --stream

Mendapat output pekerjaan, dan kemudian menunggu output tambahan yang akan diproduksi. Argumen ini dapat digunakan bersama dengan `—tail` untuk memulai dari <tail>baris terbaru dari output pekerjaan.

Default: Salah

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

Mengatur periode streaming.

Default: 5

awsbkill

Membatalkan atau menghentikan pekerjaan yang dikirimkan dalam klaster.

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

Argumen Posisi

job_ids

Menentukan daftar spasi-dipisahkan dari pekerjaan IDs untuk membatalkan atau mengakhiri.

Argumen Bernama

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

Menunjukkan nama cluster yang akan digunakan.

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

Menunjukkan pesan untuk dilampirkan ke pekerjaan, menjelaskan alasan untuk membatalkannya.

Default: "Terminated by the user"

awsbqueues

Menampilkan antrian pekerjaan yang terkait dengan cluster.

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

Argumen posisi

job_queues

Menentukan daftar spasi-dipisahkan dari nama antrian untuk ditampilkan. Jika antrian tunggal diminta, itu ditampilkan dalam versi rinci.

Argumen bernama

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

Menentukan nama cluster untuk digunakan.

-d, --details

Menunjukkan apakah akan menampilkan detail antrian.

Default: Salah

awsbhosts

Menampilkan host yang termasuk dalam lingkungan komputasi cluster.

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

Argumen Posisi

instance_ids

Menentukan daftar spasi-dipisahkan dari contoh. IDs Jika satu contoh diminta, itu ditampilkan dalam versi rinci.

Argumen Bernama

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

Menentukan nama cluster untuk digunakan.

-d, --details

Menunjukkan apakah akan menampilkan detail host.

Default: Salah

AWS ParallelCluster sumber daya dan penandaan

Dengan AWS ParallelCluster Anda dapat membuat tag untuk melacak dan mengelola AWS ParallelCluster sumber daya Anda. Anda menentukan tag yang AWS CloudFormation ingin Anda buat dan propagasikan ke semua sumber daya cluster di [tags](#) bagian file konfigurasi cluster. Anda juga dapat menggunakan tag yang AWS ParallelCluster secara otomatis menghasilkan untuk melacak dan mengelola sumber daya Anda.

Saat Anda membuat klaster, cluster dan sumber dayanya ditandai dengan tag AWS ParallelCluster dan AWS sistem yang ditentukan di bagian ini.

AWS ParallelCluster menerapkan tag ke instance, volume, dan sumber daya cluster. Untuk mengidentifikasi tumpukan cluster, AWS CloudFormation terapkan tag AWS sistem ke instance cluster. Untuk mengidentifikasi template EC2 peluncuran cluster, EC2 terapkan tag sistem ke instance. Anda dapat menggunakan tag ini untuk melihat dan mengelola AWS ParallelCluster sumber daya Anda.

Anda tidak dapat memodifikasi tag AWS sistem. Untuk menghindari dampak pada AWS ParallelCluster fungsionalitas, jangan memodifikasi AWS ParallelCluster tag.

Berikut ini adalah contoh tag AWS sistem untuk sumber AWS ParallelCluster daya. Anda tidak dapat memodifikasinya.

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

Berikut ini adalah contoh AWS ParallelCluster tag yang diterapkan ke sumber daya. Jangan memodifikasinya.

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

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

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

Anda dapat melihat tag ini di EC2 bagian AWS Management Console.

Lihat tag

1. Navigasikan EC2 konsol di <https://console.aws.amazon.com/ec2/>.
2. Untuk melihat semua tag cluster, pilih Tag di panel navigasi.
3. Untuk melihat tag cluster berdasarkan contoh, pilih Instans di panel navigasi.
4. Pilih contoh cluster.
5. Pilih tab Kelola tag di detail instance dan lihat tag.
6. Pilih tab Penyimpanan di detail instance.
7. Pilih ID Volume.
8. Di Volume, pilih volume.
9. Pilih tab Tag di detail volume dan lihat tag.

AWS ParallelCluster tag instance node kepala

Kunci	Nilai tanda
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

Kunci	Nilai tanda
<code>aws:cloudformation:stack-name</code>	<code>parallelcluster- <i>clustername</i> - MasterServerSubstack- <i>ABCD1234EFGH</i></code>
<code>aws:cloudformation:logical-id</code>	MasterServer
<code>aws:cloudformation:stack-id</code>	<code>arn:aws:cloudformation: <i>region-id</i> :<i>ACCOUNTID</i> :stack/parallelcluster- <i>clustername</i> -MasterServerSubstack- <i>ABCD1234EFGH</i> /1234abcd-12ab-12ab-12ab-1234567890abcdef0</code>
Version	<i>2.11.9</i>

AWS ParallelCluster tag volume akar simpul kepala

Kunci tanda	Nilai tanda
ClusterName	<i>clustername</i>
Application	<code>parallelcluster- <i>clustername</i></code>
<code>aws-parallelcluster-node-type</code>	Master

AWS ParallelCluster menghitung tag instance node

Kunci	Nilai tanda
ClusterName	<i>clustername</i>
<code>aws-parallelcluster-node-type</code>	Compute
<code>aws:ec2launchtemplate:id</code>	<i>lt-1234567890abcdef0</i>
<code>aws:ec2launchtemplate:version</code>	<i>1</i>

Kunci	Nilai tanda
QueueName	<i>queue-name</i>
Version	<i>2.11.9</i>

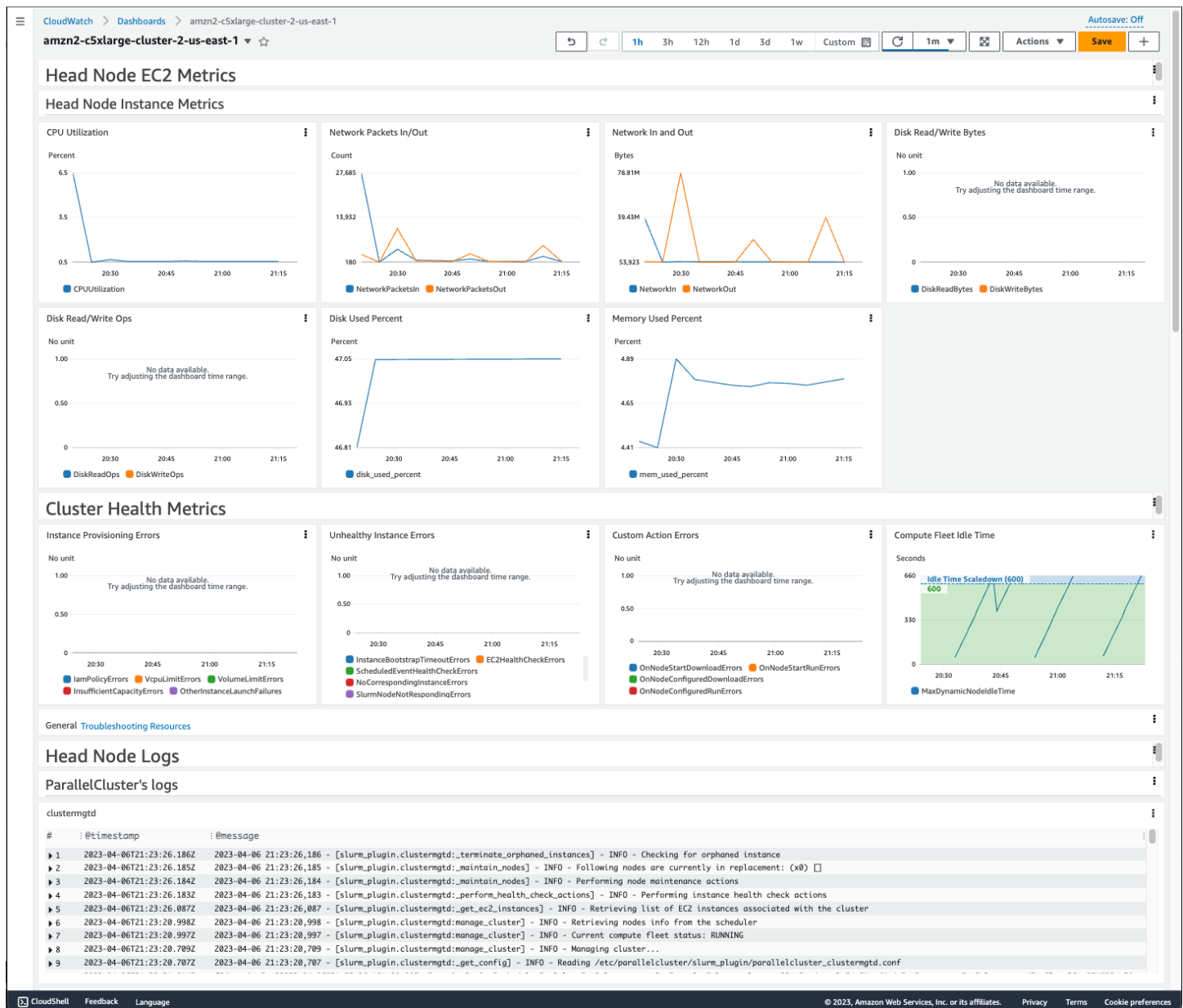
AWS ParallelCluster menghitung tag volume akar simpul

Kunci tanda	Nilai tanda
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 Dasbor Amazon

Dimulai dengan AWS ParallelCluster versi 2.10.0, CloudWatch dasbor Amazon dibuat saat cluster dibuat. Hal ini memudahkan untuk memantau node di cluster Anda, dan untuk melihat log yang disimpan di Amazon CloudWatch Logs. Nama dasbornya adalah `parallelcluster-ClusterName-Region`. *ClusterName* adalah nama cluster Anda dan *Region* merupakan cluster. Wilayah AWS Anda dapat mengakses dasbor di konsol, atau dengan membuka `https://console.aws.amazon.com/cloudwatch/home?region=Region#dashboards:name=parallelcluster-ClusterName`.

Gambar berikut menunjukkan contoh CloudWatch dasbor untuk sebuah cluster.



Bagian pertama dasbor menampilkan grafik EC2 metrik Head Node. Jika klaster Anda memiliki penyimpanan bersama, bagian selanjutnya akan menampilkan metrik penyimpanan bersama. Bagian terakhir mencantumkan Head Node Logs yang dikelompokkan berdasarkan ParallelCluster log, log Scheduler, log integrasi NICE DCV, dan log Sistem.

Untuk informasi selengkapnya tentang CloudWatch dasbor Amazon, lihat [Menggunakan CloudWatch dasbor Amazon](#) di CloudWatch Panduan Pengguna Amazon.

Jika Anda tidak ingin membuat CloudWatch dasbor Amazon, Anda harus menyelesaikan langkah-langkah ini: Pertama, tambahkan [\[dashboard\]](#) bagian ke file konfigurasi Anda, lalu tambahkan

nama bagian itu sebagai nilai [dashboard_settings](#) pengaturan di [\[cluster\]bagian](#) Anda. Di [\[dashboard\]bagian](#) Anda, atur [enable](#) = false.

Misalnya, jika [\[dashboard\]bagian](#) Anda diberi nama myDashboard dan [\[cluster\]bagian](#) Anda diberi nama myCluster, perubahan Anda menyerupai ini.

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

[dashboard MyDashboard]
enable = false
```

Integrasi dengan Amazon CloudWatch Logs

Dimulai dengan AWS ParallelCluster versi 2.6.0, log umum disimpan di CloudWatch Log secara default. Untuk informasi selengkapnya tentang CloudWatch Log, lihat [Panduan Pengguna Amazon CloudWatch Logs](#). Untuk mengkonfigurasi integrasi CloudWatch Log, lihat [\[cw_log\]bagian](#) dan [cw_log_settings](#) pengaturan.

Grup log dibuat untuk setiap cluster dengan nama `/aws/parallelcluster/cluster-name` (misalnya, `/aws/parallelcluster/testCluster`). Setiap log (atau kumpulan log jika jalur berisi `*`) pada setiap node memiliki aliran log bernama `{hostname}.{instance_id}.{logIdentifier}`. (Misalnya `ip-172-31-10-46.i-02587cf29cc3048f3.nodewatcher`.) Data log dikirim CloudWatch oleh [CloudWatch agen](#), yang berjalan seperti root pada semua instance cluster.

Dimulai dengan AWS ParallelCluster versi 2.10.0, CloudWatch dasbor Amazon dibuat saat cluster dibuat. Dasbor ini memudahkan untuk meninjau log yang disimpan di CloudWatch Log. Untuk informasi selengkapnya, lihat [CloudWatch Dasbor Amazon](#).

Daftar ini berisi *logIdentifier* dan jalur untuk aliran log yang tersedia untuk platform, penjadwal, dan node.

Aliran log tersedia untuk platform, penjadwal, dan node

Platform	Penjadw	Simpul	Pengaliran Log
amazon	awsbatc	HeadNc	dcv-autentikator: /var/log/parallelcluster/pcluster_dc v_authenticator.log

Platform	Penjadw	Simpul	Pengaliran Log
centos ubuntu	slurm		dcv-ext-authenticator: /var/log/parallelcluster/pc luster_dcv_connect.log dcv-agent: /var/log/dcv/agent.*.log dcv-xsesi: /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 koki-klien: /var/log/chef-client.log
amazon centos ubuntu	awsbatc slurm	Comput eet HeadNc	cloud-init: /var/log/cloud-init.log pengawas: /var/log/supervisord.log
amazon centos ubuntu	slurm	Comput eet	cloud-init-output: /var/log/cloud-init-output.log komputemgtd: /var/log/parallelcluster/computemgtd slurmd: /var/log/slurmd.log
amazon centos ubuntu	slurm	HeadNc	clustermgtd: /var/log/parallelcluster/clustermgtd slurm_resume: /var/log/parallelcluster/slurm_resum e.log slurm_menangguhkan: /var/log/parallelcluster/sl urm_suspend.log slurmctld: /var/log/slurmctld.log

Platform	Penjadw	Simpul	Pengaliran Log
amazon	awsbatch	Compute	pesan-sistem: /var/log/messages
centos	slurm	HeadNode	
ubuntu	awsbatch	Compute	syslog: /var/log/syslog
	slurm	HeadNode	

Pekerjaan dalam kelompok yang menggunakan AWS Batch menyimpan output pekerjaan yang mencapai `RUNNING`, `SUCCEEDED`, atau `FAILED` status di CloudWatch Log. Grup log adalah `/aws/batch/job`, dan format nama aliran log adalah `jobDefinitionName/default/ecs_task_id`. Secara default, log ini diatur agar tidak pernah kedaluwarsa, tetapi Anda dapat mengubah periode penyimpanannya. Untuk informasi selengkapnya, lihat [Mengubah penyimpanan data CloudWatch log di Log](#) di Panduan Pengguna CloudWatch Log Amazon.

Note

`chef-client`, `cloud-init-output`, `clustermgtd`, `computemgtd`, `slurm_resume`, dan `slurm_suspend` ditambahkan dalam AWS ParallelCluster versi 2.9.0. Untuk AWS ParallelCluster versi 2.6.0, `/var/log/cfn-init-cmd.log` (`cfn-init-cmd`) dan `/var/log/cfn-wire.log` (`cfn-wire`) juga disimpan di CloudWatch Log.

Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) adalah perangkat jaringan yang memiliki kemampuan OS-bypass untuk komunikasi jaringan latensi rendah dengan instans lain pada subnet yang sama. EFA diekspos dengan menggunakan Libfabric, dan dapat digunakan oleh aplikasi menggunakan Messaging Passing Interface (MPI).

Untuk menggunakan EFA dengan AWS ParallelCluster, tambahkan baris `enable_efa = true` ke [\[queue\]bagian](#).

Untuk melihat daftar EC2 instance yang mendukung EFA, lihat [Jenis instans yang didukung](#) di Panduan EC2 Pengguna Amazon untuk Instans Linux.

Untuk informasi selengkapnya tentang `enable_efa` pengaturan, lihat [enable_efa](#) di [\[queue\]bagian](#).

Grup penempatan cluster harus digunakan untuk meminimalkan latensi antar instance. Untuk informasi selengkapnya, lihat [placement](#) dan [placement_group](#).

Untuk informasi selengkapnya, lihat [Adaptor Kain Elastis](#) di Panduan EC2 Pengguna Amazon dan [Skalikan beban kerja HPC dengan adaptor kain elastis dan AWS ParallelCluster](#) di Blog Sumber AWS Terbuka.

Note

Secara default, Ubuntu distribusi mengaktifkan perlindungan ptrace (jejak proses). Dimulai dengan AWS ParallelCluster 2,6.0, ptrace perlindungan dinonaktifkan sehingga Libfabric berfungsi dengan baik. Untuk informasi selengkapnya, lihat [Menonaktifkan perlindungan ptrace](#) di Panduan EC2 Pengguna Amazon.

Note

Support EFA pada instance Graviton2 berbasis ARM telah ditambahkan di versi 2.10.1. AWS ParallelCluster

Solusi Pilihan Intel

AWS ParallelCluster tersedia sebagai Intel Select Solution untuk simulasi dan pemodelan. Konfigurasi diverifikasi untuk memenuhi standar yang ditetapkan oleh [Spesifikasi Platform Intel HPC](#), menggunakan jenis instans Intel tertentu, dan dikonfigurasi untuk menggunakan antarmuka jaringan [Elastic Fabric Adapter](#) (EFA). AWS ParallelCluster adalah solusi cloud pertama yang memenuhi persyaratan untuk program Intel Select Solutions. Jenis instans yang didukung meliputi `c5n.18xlarge`, `m5n.24xlarge`, dan `r5n.24xlarge`. Contoh konfigurasi yang kompatibel dengan standar Intel Select Solutions disediakan di bawah ini.

Example Konfigurasi Intel Select Solutions

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

[aws]
aws_region_name = <Your Wilayah 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
```

Untuk informasi selengkapnya tentang AWS ParallelCluster dan Spesifikasi Platform Intel HPC, lihat [Spesifikasi Platform Intel HPC](#).

Aktifkan Intel MPI

Intel MPI tersedia di AWS ParallelCluster AMIs Untuk menggunakan Intel MPI, Anda harus mengakui dan menerima persyaratan lisensi [perangkat lunak Intel yang disederhanakan](#). Secara default, Buka MPI ditempatkan di jalur. Untuk mengaktifkan Intel MPI alih-alih Open MPI, Anda harus terlebih dahulu memuat modul Intel MPI. Kemudian, Anda perlu menginstal versi terbaru dengan menggunakan `module load intelmpi`. Nama pasti modul berubah dengan setiap pembaruan. Untuk melihat modul mana yang tersedia, jalankan `module avail`. Outputnya adalah sebagai berikut.

```
$ 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
```

Untuk melihat modul mana yang dimuat, jalankan `module list`.

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

Untuk memverifikasi bahwa Intel MPI diaktifkan, jalankan `mpirun --version`.

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

Setelah modul Intel MPI dimuat, beberapa jalur diubah untuk menggunakan alat Intel MPI. Untuk menjalankan kode yang dikompilasi oleh alat Intel MPI, muat modul Intel MPI terlebih dahulu.

Note

Intel MPI tidak kompatibel dengan instans berbasis AWS Graviton.

Note

Sebelum AWS ParallelCluster versi 2.5.0, Intel MPI tidak tersedia di Wilayah Tiongkok (Beijing) dan China (Ningxia). AWS ParallelCluster AMIs

Spesifikasi Platform Intel HPC

AWS ParallelCluster sesuai dengan Spesifikasi Platform Intel HPC. Spesifikasi Platform Intel HPC menyediakan serangkaian persyaratan komputasi, kain, memori, penyimpanan, dan perangkat lunak untuk membantu mencapai standar kualitas dan kompatibilitas yang tinggi dengan beban kerja HPC. Untuk informasi selengkapnya, lihat [Spesifikasi Platform Intel HPC](#) dan [Aplikasi Terverifikasi Kompatibel dengan Spesifikasi Platform Intel HPC](#).

Agar sesuai dengan Spesifikasi Platform Intel HPC, persyaratan berikut harus dipenuhi:

- Sistem operasi harus CentOS 7 (`base_os` = centos7).
- Jenis instans untuk node komputasi harus memiliki CPU Intel dan setidaknya 64 GB memori. Untuk c5 keluarga tipe instance, ini berarti bahwa tipe instance minimal harus a c5.9xlarge (`compute_instance_type` = c5.9xlarge).

- Node kepala harus memiliki setidaknya 200 GB penyimpanan.
- Perjanjian Lisensi Pengguna Akhir untuk Intel Parallel Studio harus diterima ([enable_intel_hpc_platform](#) = true).
- Setiap node komputasi harus memiliki setidaknya 80 GB penyimpanan ([compute_root_volume_size](#) = 80).

Penyimpanan dapat bersifat lokal atau di jaringan (NFS dibagikan dari node kepala, Amazon EBS atau FSx untuk Lustre), dan dapat dibagikan.

Perpustakaan Kinerja Arm

Dimulai dengan AWS ParallelCluster versi 2.10.1, Arm Performance Libraries tersedia pada `ubuntu2004` nilai AWS ParallelCluster AMIs for `alinux2centos8,ubuntu1804`, dan untuk pengaturan. [base_os](#) Perpustakaan Kinerja Arm menyediakan pustaka matematika inti standar yang dioptimalkan untuk aplikasi komputasi berkinerja tinggi pada prosesor Arm. Untuk menggunakan Arm Performance Libraries, Anda harus mengakui dan menerima ketentuan [Arm Performance Libraries \(versi gratis\) - Perjanjian Lisensi Pengguna Akhir](#). Untuk informasi selengkapnya tentang Arm Performance Libraries, lihat [Free Arm Performance Libraries](#).

Untuk mengaktifkan Arm Performance Libraries, Anda harus terlebih dahulu memuat modul Arm Performance Libraries. `armpl-21.0.0` membutuhkan GCC-9.3 sebagai persyaratan, saat Anda memuat `armpl/21.0.0` modul, modul juga akan dimuat. `gcc/9.3` Nama pasti modul berubah dengan setiap pembaruan. Untuk melihat modul mana yang tersedia, jalankan `module avail`. Kemudian, Anda perlu menginstal versi terbaru dengan `module load armpl` menggunakan. Outputnya adalah sebagai berikut.

```
$ module avail

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

Untuk memuat modul, jalankan `module load modulename`. Anda dapat menambahkan ini ke skrip yang digunakan untuk menjalankan `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

Untuk melihat modul mana yang dimuat, jalankan `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

Untuk memverifikasi bahwa Arm Performance Libraries diaktifkan, jalankan pengujian contoh.

```
$ 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
```

Setelah modul Arm Performance Libraries dimuat, beberapa jalur diubah untuk menggunakan alat Arm Performance Libraries. Untuk menjalankan kode yang dikompilasi oleh alat Arm Performance Libraries, muat modul Arm Performance Libraries terlebih dahulu.

Note

AWS ParallelCluster versi antara penggunaan 2.10.1 dan 2.10.4. `armpl/20.2.1`

Connect ke head node melalui Amazon DCV

Amazon DCV adalah teknologi visualisasi jarak jauh yang memungkinkan pengguna terhubung dengan aman ke aplikasi 3D intensif grafis yang di-host di server berkinerja tinggi jarak jauh. Untuk informasi selengkapnya, lihat [Amazon DCV](#).

Perangkat lunak Amazon DCV secara otomatis diinstal pada node kepala saat menggunakan `base_os = alinux2`, `base_os = centos7`, `base_os = ubuntu1804` atau `base_os = ubuntu2004`.

Jika node kepala adalah instance ARM, perangkat lunak Amazon DCV diinstal secara otomatis saat menggunakan `base_os = alinux2`, `base_os = centos7`, atau `base_os = ubuntu1804`.

Untuk mengaktifkan Amazon DCV pada node kepala, `dcv_settings` harus berisi nama `[dcv]bagian` yang memiliki `enable = master` dan `base_os` harus disetel ke `alinux2`, `centos7`, `ubuntu1804`, atau `ubuntu2004`. Jika node kepala adalah instance ARM, `base_os` harus diatur ke `alinux2`, `centos7`, atau `ubuntu1804`. Dengan cara ini, AWS ParallelCluster atur parameter konfigurasi cluster `shared_dir` ke `folder penyimpanan server DCV`.

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

Untuk informasi selengkapnya tentang parameter konfigurasi Amazon DCV, lihat `dcv_settings`. Untuk terhubung ke sesi Amazon DCV, gunakan `pcluster dcv` perintah.

Note

Support untuk Amazon DCV centos8 telah dihapus di AWS ParallelCluster versi 2.10.4. Support untuk Amazon DCV centos8 telah ditambahkan dalam AWS ParallelCluster versi 2.10.0. Support untuk Amazon DCV pada instans AWS berbasis Graviton telah ditambahkan di versi 2.9.0. AWS ParallelCluster Support untuk Amazon DCV aktif `alinux2` dan `ubuntu1804` ditambahkan dalam AWS ParallelCluster versi 2.6.0. Support untuk Amazon DCV centos7 telah ditambahkan di AWS ParallelCluster versi 2.5.0.

Note

Amazon DCV tidak didukung pada instans AWS berbasis Graviton di versi 2.8.0 dan 2.8.1. AWS ParallelCluster

Sertifikat Amazon DCV HTTPS

Amazon DCV secara otomatis menghasilkan sertifikat yang ditandatangani sendiri untuk mengamankan lalu lintas antara klien Amazon DCV dan server Amazon DCV.

Untuk mengganti sertifikat DCV Amazon yang ditandatangani sendiri default dengan sertifikat lain, sambungkan terlebih dahulu ke node kepala. Kemudian, salin sertifikat dan kunci ke `/etc/dcv` folder sebelum menjalankan [pcluster dcv](#) perintah.

Untuk informasi selengkapnya, lihat [Mengubah sertifikat TLS](#) di Panduan Administrator Amazon DCV.

Perizinan Amazon DCV

Server Amazon DCV tidak memerlukan server lisensi saat berjalan di EC2 instans Amazon. Namun, server Amazon DCV harus terhubung secara berkala ke bucket Amazon S3 untuk menentukan apakah lisensi yang valid tersedia.

AWS ParallelCluster secara otomatis menambahkan izin yang diperlukan ke file.

`ParallelClusterInstancePolicy` Saat menggunakan Kebijakan Instans IAM khusus, gunakan izin yang dijelaskan di Amazon [DCV di Amazon di Panduan Administrator EC2](#) DCV Amazon.

Untuk tips pemecahan masalah, lihat. [Memecahkan masalah di Amazon DCV](#)

Menggunakan **pcluster update**

Dimulai dengan AWS ParallelCluster versi 2.8.0, [pcluster update](#) menganalisis pengaturan yang digunakan untuk membuat cluster saat ini dan pengaturan dalam file konfigurasi untuk masalah. Jika ada masalah yang ditemukan, mereka dilaporkan, dan langkah-langkah yang harus diambil untuk memperbaiki masalah ditampilkan. Misalnya, jika [compute_instance_type](#) pengaturan diubah ke jenis instans yang berbeda, armada komputasi harus dihentikan sebelum pembaruan dapat dilanjutkan. Masalah ini dilaporkan ketika ditemukan. Jika tidak ada masalah pemblokiran yang dilaporkan, Anda akan diminta apakah Anda ingin menerapkan perubahan.

Dokumentasi untuk setiap setelan menentukan kebijakan pembaruan untuk setelan tersebut.

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan., Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

Pengaturan ini dapat diubah, dan cluster dapat diperbarui menggunakan [pcluster update](#).

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

Pengaturan ini tidak dapat diubah jika klaster yang ada belum dihapus. Entah perubahan harus dikembalikan atau cluster harus dihapus (menggunakan [pcluster delete](#)), dan kemudian cluster baru dibuat (menggunakan [pcluster create](#)) di tempat cluster lama.

Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.

Pengaturan ini dapat diubah, dan cluster diperbarui menggunakan [pcluster update](#).

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

Pengaturan ini tidak dapat diubah saat armada komputasi ada. Perubahan harus dikembalikan atau armada komputasi harus dihentikan (menggunakan [pcluster stop](#)), diperbarui (menggunakan [pcluster update](#)), dan kemudian armada komputasi baru yang dibuat (menggunakan). [pcluster start](#)

Kebijakan pembaruan: Pengaturan ini tidak dapat dikurangi selama pembaruan.

Pengaturan ini dapat diubah, tetapi tidak dapat dikurangi. Jika pengaturan ini harus dikurangi, perlu untuk menghapus cluster (menggunakan [pcluster delete](#)), dan membuat cluster baru (menggunakan [pcluster create](#)).

Kebijakan pembaruan: Mengurangi ukuran antrian di bawah jumlah node saat ini mengharuskan armada komputasi dihentikan terlebih dahulu.

Pengaturan ini dapat diubah, tetapi jika perubahan akan mengurangi ukuran antrian di bawah ukuran saat ini, armada komputasi harus dihentikan (menggunakan [pcluster stop](#)), diperbarui (menggunakan [pcluster update](#)), dan kemudian armada komputasi baru dibuat (menggunakan). [pcluster start](#)

Kebijakan pembaruan: Mengurangi jumlah node statis dalam antrian mengharuskan armada komputasi dihentikan terlebih dahulu.

Pengaturan ini dapat diubah, tetapi jika perubahan akan mengurangi jumlah node statis dalam antrian di bawah ukuran saat ini, armada komputasi harus dihentikan (menggunakan [pcluster stop](#)), diperbarui (menggunakan [pcluster update](#)), dan kemudian armada komputasi baru dibuat (menggunakan). [pcluster start](#)

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan. Memperbarui pengaturan ini tidak dapat dipaksakan.

Pengaturan ini tidak dapat diubah jika klaster yang ada belum dihapus. Entah perubahan harus dikembalikan atau cluster harus dihapus (menggunakan [pcluster delete](#)), dan kemudian cluster baru dibuat (menggunakan [pcluster create](#)) di tempat cluster lama.

Kebijakan pembaruan: Jika sistem file Amazon FSx untuk Lustre yang AWS ParallelCluster dikelola tidak ditentukan dalam konfigurasi, setelah ini dapat diubah selama pembaruan.

Pengaturan ini dapat diubah jika [\[cluster\]fsx_settings](#) tidak ditentukan atau jika keduanya [fsx_settings](#) dan [fsx-fs-id](#) di ditentukan untuk [\[fsx fs\]](#) me-mount eksternal yang ada FSx untuk sistem file Lustre.

Contoh ini menunjukkan [pcluster update](#) dengan beberapa perubahan yang memblokir pembaruan.

```
$ 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.

Penambahan AMI dan penggantian EC2 instance

Untuk memastikan bahwa semua node komputasi cluster yang diluncurkan secara dinamis berperilaku konsisten, AWS ParallelCluster menonaktifkan pembaruan OS otomatis instance cluster. Selain itu, satu set khusus AWS ParallelCluster AMIs dibangun untuk setiap versi AWS ParallelCluster dan CLI terkait. Kumpulan spesifik ini AMIs tetap tidak berubah dan hanya didukung oleh AWS ParallelCluster versi yang mereka buat. AWS ParallelCluster AMIs untuk versi yang dirilis tidak diperbarui.

Namun, karena masalah keamanan yang muncul, pelanggan mungkin ingin menambahkan tambalan ke ini AMIs dan kemudian memperbarui cluster mereka dengan AMI yang ditambah. Ini sejalan dengan [Model Tanggung Jawab AWS ParallelCluster Bersama](#).

Untuk melihat kumpulan spesifik yang AWS ParallelCluster AMIs didukung oleh versi AWS ParallelCluster CLI yang saat ini Anda gunakan, jalankan:

```
$ pcluster version
```

Kemudian lihat [amis.txt](#) di AWS ParallelCluster GitHub repositori.

AWS ParallelCluster Head node adalah instance statis dan Anda dapat memperbaruinya secara manual. Mulai ulang dan reboot node kepala didukung sepenuhnya dimulai dengan AWS ParallelCluster versi 2.11, jika jenis instance tidak memiliki penyimpanan instance. Untuk informasi

selengkapnya, lihat [Jenis instans dengan volume penyimpanan](#) instans di Panduan EC2 Pengguna Amazon untuk Instans Linux. Anda tidak dapat memperbarui AMI untuk klaster yang ada.

Head node restart dan reboot dengan pembaruan AMI dari instance komputasi cluster didukung sepenuhnya dimulai dengan AWS ParallelCluster versi 3.0.0. Pertimbangkan untuk meningkatkan ke versi terbaru untuk menggunakan fitur-fitur ini.

Pembaruan atau penggantian instance node kepala

Dalam beberapa keadaan, Anda mungkin diminta untuk memulai ulang atau me-reboot node kepala. Misalnya, ini diperlukan saat Anda memperbarui OS secara manual, atau ketika ada [AWS instance pensiun terjadwal](#) yang memaksakan restart instance head node.

Jika instans Anda tidak memiliki drive fana, Anda dapat berhenti dan memulainya lagi kapan saja. Dalam kasus pensiun terjadwal, memulai instance yang dihentikan memigrasikannya untuk menggunakan perangkat keras baru.

Demikian pula, Anda dapat secara manual menghentikan dan memulai sebuah instance yang tidak memiliki penyimpanan instance. Untuk kasus ini dan untuk kasus kasus lain tanpa volume fana, lanjutkan ke. [Berhenti dan mulai node kepala cluster](#)

Jika instans Anda memiliki drive sementara dan dihentikan, data di penyimpanan instance akan hilang. Anda dapat menentukan apakah jenis instance yang digunakan untuk node kepala memiliki penyimpanan instance dari tabel yang ditemukan dalam [volume penyimpanan Instance](#).

Bagian berikut menjelaskan batasan dalam menggunakan instance dengan volume penyimpanan instance.

Keterbatasan penyimpanan instans

Keterbatasan dalam menggunakan AWS ParallelCluster versi 2.11 dan tipe instance dengan penyimpanan instance adalah sebagai berikut:

- Ketika drive sementara tidak dienkripsi ([encrypted_ephemeral](#) parameter disetel ke `false` atau tidak disetel), sebuah AWS ParallelCluster instance tidak dapat boot setelah instance berhenti. Ini karena informasi tentang fana lama yang tidak ada ditulis ke dalam `fstab` dan OS mencoba memasang penyimpanan yang tidak ada.
- Ketika drive sementara dienkripsi ([encrypted_ephemeral](#) parameter disetel ke `true`), sebuah AWS ParallelCluster instance dapat dimulai setelah berhenti tetapi drive fana baru tidak diatur, dipasang, atau tersedia.

- Ketika drive sementara dienkrpsi, sebuah AWS ParallelCluster instance dapat di-boot ulang tetapi drive fana lama (yang bertahan dari reboot instance) tidak dapat diakses karena kunci enkripsi dibuat dalam memori yang hilang dengan reboot.

Satu-satunya kasus yang didukung adalah instance reboot, ketika drive sementara tidak dienkrpsi.

Ini karena drive bertahan dari reboot dan dipasang kembali karena entri yang ditulis. `fstab`

Solusi keterbatasan penyimpanan instans

Pertama, simpan data Anda. Untuk memeriksa apakah Anda memiliki data yang perlu dipertahankan, lihat konten di [ephemeral_dir](#) folder (secara `/scratch` default). Anda dapat mentransfer data ke volume root atau sistem penyimpanan bersama yang terpasang ke cluster, seperti Amazon, Amazon FSx EFS, atau Amazon EBS. Perhatikan bahwa transfer data ke penyimpanan jarak jauh dapat menimbulkan biaya tambahan.

Akar penyebab keterbatasan ada dalam logika yang AWS ParallelCluster digunakan untuk memformat dan memasang volume penyimpanan instance. Logika menambahkan `/etc/fstab` entri ke formulir:

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

`${ephemeral_dir}` adalah nilai [ephemeral_dir](#) parameter dari file konfigurasi `pcluster` (default ke). `/scratch`

Baris ini ditambahkan sehingga jika atau ketika sebuah node di-boot ulang, volume penyimpanan instance dipasang kembali secara otomatis. Ini diinginkan karena data dalam drive fana tetap ada melalui reboot. Namun, data pada drive fana tidak bertahan melalui siklus start atau stop. Ini berarti mereka diformat dan dipasang tanpa data.

Satu-satunya kasus yang didukung adalah instance reboot ketika drive sementara tidak dienkrpsi. Ini karena drive bertahan dari reboot dan dipasang kembali karena sudah tertulis `fstab`.

Untuk menyimpan data dalam semua kasus lain, Anda harus menghapus entri volume logis sebelum menghentikan instance. Misalnya, hapus `/dev/vg.01/lv_ephemeral` dari `/etc/fstab` sebelum menghentikan instance. Setelah melakukan ini, Anda memulai instance tanpa memasang volume fana. Namun, pemasangan penyimpanan instance lagi tidak akan tersedia setelah penghentian atau dimulainya instance.

Setelah menyimpan data Anda dan kemudian menghapus `fstab` entri, lanjutkan ke bagian berikutnya.

Berhenti dan mulai node kepala cluster

Note

Dimulai dengan AWS ParallelCluster versi 2.11, head node stop dan start hanya didukung jika jenis instance tidak memiliki penyimpanan instance.

1. Pastikan tidak ada pekerjaan yang berjalan di cluster.

Saat menggunakan Slurm penjadwal:

- Jika `sbatch --no-requeue` opsi tidak ditentukan, pekerjaan yang sedang berjalan akan diminta ulang.
- Jika `--no-requeue` opsi ditentukan, menjalankan pekerjaan gagal.

2. Minta penghentian armada komputasi 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. Tunggu hingga status armada komputasi adalah STOPPED:

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

4. Untuk pembaruan manual dengan reboot OS atau restart instance, Anda dapat menggunakan AWS Management Console atau AWS CLI. Berikut ini adalah contoh penggunaan 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. Mulai armada komputasi 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 Perintah CLI

`pcluster` dan `pcluster-config` merupakan perintah AWS ParallelCluster CLI. Anda gunakan `pcluster` untuk meluncurkan dan mengelola klaster HPC di AWS Cloud dan `pcluster-config` untuk memperbarui konfigurasi Anda.

Untuk menggunakannya `pcluster`, Anda harus memiliki peran IAM dengan [izin](#) yang diperlukan untuk menjalankannya.

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

Topik

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

pcluster

`pcluster` adalah perintah AWS ParallelCluster CLI utama. Anda gunakan `pcluster` untuk meluncurkan dan mengelola cluster HPC di AWS Cloud

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

Pendapat

`pcluster` *command*

Pilihan yang memungkinkan: [configurecreate](#), [createami](#), [dcv](#), [delete](#), [instances](#), [list](#), [ssh](#), [start](#), [status](#), [stop](#), [update](#), [version](#)

Sub-perintah:

Topik

- [pcluster configure](#)
- [pcluster create](#)
- [pcluster createami](#)
- [pcluster dcw](#)
- [pcluster delete](#)
- [pcluster instances](#)
- [pcluster list](#)
- [pcluster ssh](#)
- [pcluster start](#)
- [pcluster status](#)
- [pcluster stop](#)
- [pcluster update](#)
- [pcluster version](#)

pcluster configure

Memulai AWS ParallelCluster konfigurasi. Untuk informasi selengkapnya, lihat [Mengkonfigurasi AWS ParallelCluster](#).

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

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster configure`.

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

Menentukan path lengkap dari file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

Untuk informasi selengkapnya, lihat [Mengkonfigurasi AWS ParallelCluster](#).

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

Menentukan Wilayah AWS untuk menggunakan. Jika ini ditentukan, konfigurasi melewati Wilayah AWS deteksi.

Untuk menghapus sumber daya jaringan di VPC, Anda dapat menghapus tumpukan CloudFormation jaringan. Nama tumpukan dimulai dengan "parallelclusternetworking-" dan berisi waktu pembuatan dalam format "YYYYMMDDHHMMSS". Anda dapat membuat daftar tumpukan menggunakan perintah [daftar-tumpukan](#).

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

Tumpukan dapat dihapus menggunakan perintah [delete-stack](#).

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

VPC yang dibuat untuk [pcluster configure](#) Anda tidak dibuat di tumpukan CloudFormation jaringan. Anda dapat menghapus VPC itu secara manual di konsol atau dengan menggunakan file. AWS CLI

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

pcluster create

Membuat cluster baru.

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

Argumen posisi

cluster_name

Mendefinisikan nama cluster. Nama AWS CloudFormation tumpukan adalah `parallelcluster-cluster_name`.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster create`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

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

Menentukan Wilayah AWS untuk menggunakan. Urutan prioritas yang digunakan Wilayah AWS untuk memilih cluster baru adalah sebagai berikut:

1. `-r` atau `--region` parameter ke [pcluster create](#).
2. `AWS_DEFAULT_REGION` variabel lingkungan.
3. `aws_region_name` pengaturan di `[aws]` bagian file AWS ParallelCluster konfigurasi (lokasi default adalah `~/.parallelcluster/config`.) Ini adalah lokasi yang diperbarui oleh [pcluster configure](#) perintah.
4. `region` pengaturan di `[default]` bagian file AWS CLI konfigurasi (`~/.aws/config`.)

-nw, --nowait

Menunjukkan untuk tidak menunggu peristiwa tumpukan setelah menjalankan perintah tumpukan.

Default ke `False`.

-nr, --norollback

Menonaktifkan tumpukan rollback pada kesalahan.

Default ke `False`.

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

Menentukan URL untuk AWS CloudFormation template kustom jika digunakan ketika dibuat.

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

Menunjukkan template cluster untuk digunakan.

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

Menambahkan parameter tambahan untuk membuat tumpukan.

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

Menentukan tag tambahan untuk menambah tumpukan.

Ketika perintah dipanggil dan mulai polling untuk status panggilan itu, aman untuk menggunakan “Ctrl-C” untuk keluar. Anda dapat kembali melihat status saat ini dengan menelepon `pcluster status mycluster`.

Contoh menggunakan AWS ParallelCluster versi 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) Membuat AMI khusus untuk digunakan. 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 ]
```

Dependensi yang dibutuhkan

Selain AWS ParallelCluster CLI, ketergantungan berikut diperlukan untuk menjalankan: `pcluster createami`

- Packer: Unduh versi terbaru dari <https://developer.hashicorp.com/packer/downloads>.

Note

Sebelum AWS ParallelCluster versi 2.8.0, [Berkshelf](#) (diinstal menggunakan `install berkshelf`) diperlukan untuk menggunakan `pcluster createami`

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster createami`.

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

Menentukan AMI dasar yang akan digunakan untuk membangun AWS ParallelCluster AMI.

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

Menentukan OS dari basis AMI. Opsi yang valid adalah: `alinux2`, `ubuntu1804`, `ubuntu2004`, dan `centos7`.

Note

Perubahan dukungan OS dalam AWS ParallelCluster versi yang berbeda:

- Support untuk `centos8` telah dihapus di AWS ParallelCluster versi 2.10.4.
- Support untuk `centos8` ditambahkan, dan dukungan untuk `centos6` telah dihapus di AWS ParallelCluster versi 2.10.0.
- Support untuk `alinux2` ditambahkan di AWS ParallelCluster versi 2.6.0.
- Support untuk `ubuntu1804` ditambahkan di AWS ParallelCluster versi 2.5.0.

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

Menentukan jenis instance yang akan digunakan untuk membuat AMI.

Default ke t2.xlarge.

 Note

Support untuk --instance-type argumen ditambahkan di AWS ParallelCluster versi 2.4.1.

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

Menentukan nama awalan dari AMI AWS ParallelCluster yang dihasilkan.

Default ke custom-ami-.

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

Menentukan buku masak yang akan digunakan untuk membangun AMI AWS ParallelCluster .

--post-install *POST_INSTALL_SCRIPT*

Menentukan jalur ke skrip pasca-instal. Jalur harus menggunakan skema s3://https://, atau file:// URL. Contohnya meliputi hal berikut:

- 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 untuk --post-install argumen ditambahkan di AWS ParallelCluster versi 2.10.0.

--no-public-ip

Jangan mengaitkan alamat IP publik ke instance yang digunakan untuk membuat AMI. Secara default, alamat IP publik dikaitkan dengan instance.

 Note

Support untuk `--no-public-ip` argumen ditambahkan di AWS ParallelCluster versi 2.5.0.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

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

Menentukan [bagian \[cluster\]](#) dari yang akan digunakan *CONFIG_FILE* untuk mengambil pengaturan VPC dan subnet.

 Note

Support untuk `--cluster-template` argumen ditambahkan di AWS ParallelCluster versi 2.4.0.

--vpc-id *VPC_ID*

Menentukan ID VPC yang akan digunakan untuk membangun AMI. AWS ParallelCluster

 Note

Support untuk `--vpc-id` argumen ditambahkan di AWS ParallelCluster versi 2.5.0.

--subnet-id *SUBNET_ID*

Menentukan ID subnet yang akan digunakan untuk membangun AMI AWS ParallelCluster .

 Note

Support untuk `--vpc-id` argumen ditambahkan di AWS ParallelCluster versi 2.5.0.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

pcluster dcv

Berinteraksi dengan server Amazon DCV yang berjalan di node kepala.

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

pcluster dcv *command*

Pilihan yang memungkinkan: [connect](#)

Note

Dukungan OS berubah untuk `pcluster dcv` perintah dalam AWS ParallelCluster versi yang berbeda:

- Support untuk `pcluster dcv` perintah centos8 telah ditambahkan dalam AWS ParallelCluster versi 2.10.0.
- Support untuk `pcluster dcv` perintah pada instance AWS berbasis Graviton ditambahkan di versi 2.9.0. AWS ParallelCluster
- Support untuk `pcluster dcv` perintah ubuntu1804 telah ditambahkan di AWS ParallelCluster versi 2.6.0.
- Support untuk `pcluster dcv` perintah centos7 telah ditambahkan di AWS ParallelCluster versi 2.5.0.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster dcv`.

Sub-perintah

pcluster dcv connect

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

Important

URL kedaluwarsa 30 detik setelah dikeluarkan. Jika koneksi tidak dibuat sebelum URL kedaluwarsa, jalankan `pcluster dcv connect` lagi untuk menghasilkan URL baru.

Argumen posisi

cluster_name

Menentukan nama cluster untuk terhubung ke.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster dcv connect`.

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

Jalur kunci kunci SSH yang akan digunakan untuk koneksi.

Kuncinya harus yang ditentukan pada waktu pembuatan cluster dalam parameter [key_name](#) konfigurasi. Argumen ini opsional, tetapi jika tidak ditentukan, maka kunci harus tersedia secara default untuk klien SSH. Misalnya, tambahkan ke `ssh-agent` with `ssh-add`.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

-s, --show-url

Menampilkan URL satu kali untuk menghubungkan ke sesi Amazon DCV. Browser default tidak dibuka saat opsi ini ditentukan.

Note

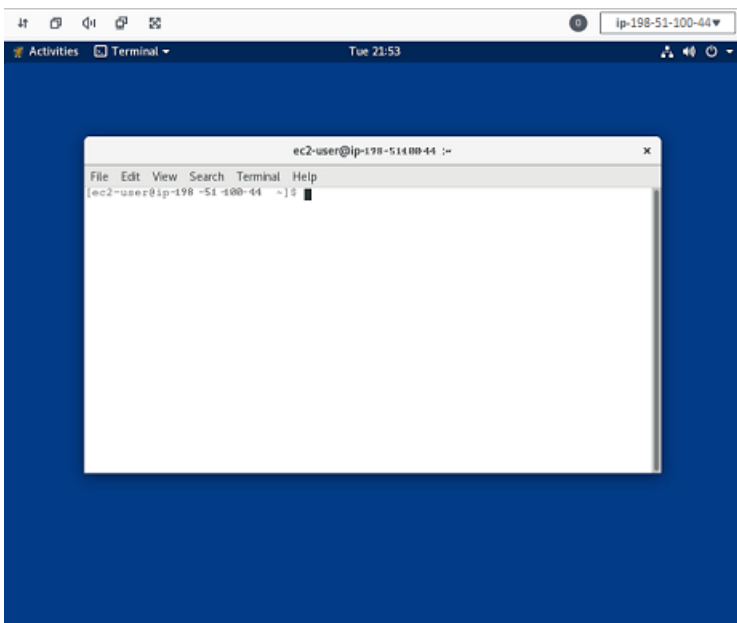
Support untuk `--show-url` argumen ditambahkan di AWS ParallelCluster versi 2.5.1.

Contoh menggunakan AWS ParallelCluster versi 2.11.7:

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

Membuka browser default untuk terhubung ke sesi Amazon DCV yang berjalan di node kepala.

Sesi Amazon DCV baru dibuat jika belum dimulai.



pcluster delete

Menghapus cluster.

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

Argumen posisi

cluster_name

Menentukan nama cluster untuk menghapus.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster delete`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

--keep-logs

Simpan data CloudWatch Log setelah menghapus cluster. Grup log tetap ada sampai Anda menghapusnya secara manual, tetapi peristiwa log kedaluwarsa berdasarkan [retention_days](#) pengaturan. Pengaturan default ke 14 hari.

Note

Support untuk **--keep-logs** argumen ditambahkan di AWS ParallelCluster versi 2.6.0.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

Ketika perintah dipanggil dan mulai polling untuk status panggilan itu, aman untuk menggunakan “Ctrl-C” untuk keluar. Anda dapat kembali melihat status saat ini dengan menelepon `pcluster status mycluster`.

Contoh menggunakan AWS ParallelCluster versi 2.11.7:

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

Untuk menghapus sumber daya jaringan di VPC, Anda dapat menghapus tumpukan CloudFormation jaringan. Nama tumpukan dimulai dengan “parallelclusternetworking-” dan berisi waktu pembuatan

dalam format “YYYYMMDDHHMMSS”. Anda dapat membuat daftar tumpukan menggunakan perintah [daftar-tumpukan](#).

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

Tumpukan dapat dihapus menggunakan perintah [delete-stack](#).

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

VPC yang dibuat untuk [pcluster configure](#) Anda tidak dibuat di tumpukan CloudFormation jaringan. Anda dapat menghapus VPC itu secara manual di konsol atau dengan menggunakan file. AWS CLI

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

pcluster instances

Menampilkan daftar semua instance dalam sebuah cluster.

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

Argumen posisi

cluster_name

Menampilkan instance untuk cluster dengan nama yang disediakan.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster instances`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke ~/.parallelcluster/config.

-r REGION, --region REGION

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

Contoh menggunakan AWS ParallelCluster versi 2.11.7:

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

pcluster list

Menampilkan daftar tumpukan yang terkait dengan AWS ParallelCluster.

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

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster list`.

--color

Menampilkan status cluster dalam warna.

Default ke False.

-c CONFIG_FILE, --config CONFIG_FILE

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke c.

-r REGION, --region REGION

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

Daftar nama setiap AWS CloudFormation tumpukan bernama `parallelcluster-*`.

Contoh menggunakan AWS ParallelCluster versi 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

Menjalankan ssh perintah dengan nama pengguna dan alamat IP cluster yang sudah terisi sebelumnya. Argumen arbitrer ditambahkan ke akhir perintah. ssh Perintah ini dapat disesuaikan di bagian alias dari file konfigurasi.

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

Argumen posisi

cluster_name

Menentukan nama cluster untuk terhubung ke.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster ssh`.

-d, --dryrun

Mencetak perintah yang akan dijalankan dan keluar.

Default ke False.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke Region ditentukan dengan menggunakan perintah. [pcluster configure](#)

Contoh menggunakan AWS ParallelCluster versi 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
```

Menjalankan ssh perintah dengan nama pengguna dan alamat IP cluster yang sudah terisi sebelumnya:

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

sshPerintah didefinisikan dalam file konfigurasi global di bawah file [Bagian \[aliases\]](#). Hal ini dapat disesuaikan sebagai berikut.

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

Variabel yang diganti:

CFN_USER

Nama pengguna untuk [base_os](#) yang dipilih.

MASTER_IP

Alamat IP dari node kepala.

ARGS

Argumen opsional untuk diteruskan ke ssh perintah.

pcluster start

Memulai armada komputasi untuk cluster yang telah dihentikan.

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

Argumen posisi

cluster_name

Memulai armada komputasi dari nama cluster yang disediakan.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster start`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

Contoh menggunakan AWS ParallelCluster versi 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
```

Perintah ini menetapkan parameter Auto Scaling Group ke salah satu dari berikut ini:

- Nilai konfigurasi awal (`max_queue_size` dan `initial_queue_size`) dari template yang digunakan untuk membuat cluster.
- Nilai konfigurasi yang digunakan untuk memperbarui cluster sejak pertama kali dibuat.

pcluster status

Menarik status cluster saat ini.

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

Argumen posisi

cluster_name

Menunjukkan status cluster dengan nama yang disediakan.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster status`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

-nw, --nowait

Menunjukkan untuk tidak menunggu peristiwa tumpukan setelah memproses perintah tumpukan.

Default ke `False`.

Contoh menggunakan AWS ParallelCluster versi 2.11.7:

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

pcluster stop

Menghentikan armada komputasi, membiarkan node kepala berjalan.

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

Argumen posisi

cluster_name

Menghentikan armada komputasi dari nama cluster yang disediakan.

Contoh menggunakan AWS ParallelCluster versi 2.11.7:

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster stop`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

```
$ 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
```

Menetapkan parameter grup Auto Scaling ke min/max/desired = 0/0/0, dan mengakhiri armada komputasi. Kepala tetap berjalan. Untuk menghentikan semua EC2 sumber daya dan menghindari EC2 biaya, pertimbangkan untuk menghapus cluster.

pcluster update

Menganalisis file konfigurasi untuk menentukan apakah cluster dapat diperbarui dengan aman. Jika analisis menentukan cluster dapat diperbarui, Anda diminta untuk mengkonfirmasi perubahan. Jika analisis menunjukkan klaster tidak dapat diperbarui, pengaturan konfigurasi yang merupakan sumber konflik akan disebutkan dengan detail. Untuk informasi selengkapnya, lihat [Menggunakan pcluster update](#).

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

Argumen posisi

cluster_name

Menentukan nama cluster untuk memperbarui.

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster update`.

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

Menentukan file konfigurasi alternatif untuk digunakan.

Default ke `~/.parallelcluster/config`.

--force

Mengaktifkan pembaruan meskipun satu atau beberapa pengaturan memiliki perubahan pemblokiran atau jika tindakan yang belum selesai diperlukan (seperti menghentikan armada komputasi) sebelum pembaruan dapat dilanjutkan. Ini tidak harus digabungkan dengan `--yes` argumen.

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

Menentukan Wilayah AWS untuk menggunakan. Default ke yang Wilayah AWS ditentukan dengan menggunakan perintah. [pcluster configure](#)

-nr, --norollback

Menonaktifkan AWS CloudFormation tumpukan rollback pada kesalahan.

Default ke `False`.

-nw, --nowait

Menunjukkan untuk tidak menunggu peristiwa tumpukan setelah memproses perintah tumpukan.

Default ke `False`.

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

Menentukan bagian dari template cluster untuk digunakan.

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

Menambahkan parameter tambahan ke pembaruan tumpukan.

-rd, --reset-desired

Mengatur ulang kapasitas Grup Auto Scaling saat ini ke nilai konfigurasi awal.

Default ke False.

--yes

Secara otomatis mengasumsikan bahwa jawaban untuk semua prompt adalah ya. Ini tidak boleh digabungkan dengan --force argumen.

```
$ 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
```

Ketika perintah dipanggil dan mulai polling untuk status panggilan itu, aman untuk menggunakan “Ctrl-C” untuk keluar. Anda dapat kembali melihat status saat ini dengan menelepon `pcluster status mycluster`.

pcluster version

Menampilkan AWS ParallelCluster versi.

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

Untuk flag khusus perintah, jalankan: `pcluster [command] --help`

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster version`.

Ketika perintah dipanggil dan mulai polling untuk status panggilan itu, aman untuk menggunakan “Ctrl-C” untuk keluar. Anda dapat kembali melihat status saat ini dengan menelepon `pcluster status mycluster`.

```
$ pcluster version
2.11.7
```

pcluster-config

Memperbarui file AWS ParallelCluster konfigurasi.

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

Untuk flag khusus perintah, jalankan: `pcluster-config [command] -h`

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster-config`.

Note

`pcluster-config` Perintah ditambahkan dalam AWS ParallelCluster versi 2.9.0.

Sub-perintah

pcluster-config convert

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

Argumen bernama

-h, --help

Menampilkan teks bantuan untuk `pcluster-config convert`.

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

Menentukan path dari file konfigurasi untuk membaca.

Default ke `~/.parallelcluster/config`.

Untuk informasi selengkapnya, lihat [Mengkonfigurasi AWS ParallelCluster](#).

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

Menunjukkan [Bagian \[cluster\]](#) untuk digunakan. Jika argumen ini tidak ditentukan, `pcluster-config convert` akan menggunakan `cluster_template` pengaturan di [Bagian \[global\]](#). Jika itu tidak ditentukan, maka `[cluster default]` bagian tersebut digunakan.

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

Menentukan path dari file konfigurasi dikonversi untuk ditulis. Secara default, output ditulis ke `STDOUT`.

Contoh:

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

Mengonversi konfigurasi cluster yang ditentukan di `[cluster alpha]` bagian `~/.parallelcluster/config`, menulis file konfigurasi yang dikonversi ke `~/.parallelcluster/multiinstance`.

Konfigurasi

Secara default, AWS ParallelCluster menggunakan `~/.parallelcluster/config` file untuk semua parameter konfigurasi. Anda dapat menentukan file konfigurasi kustom dengan menggunakan opsi baris `--config` perintah `-c` atau variabel `AWS_PCLUSTER_CONFIG_FILE` lingkungan.

Contoh file konfigurasi diinstal dengan AWS ParallelCluster di direktori Python di `site-packages/aws-parallelcluster/examples/config`. Contoh file konfigurasi juga tersedia di GitHub, at <https://github.com/aws/aws-parallelcluster/blob/v2.11.9/cli/src/pcluster/examples/config>.

Versi AWS ParallelCluster 2 saat ini: 2.11.9.

Topik

- [Tata Letak](#)
- [Bagian \[global\]](#)
- [Bagian \[aws\]](#)
- [Bagian \[aliases\]](#)
- [Bagian \[cluster\]](#)
- [Bagian \[compute_resource\]](#)
- [Bagian \[cw_log\]](#)
- [Bagian \[dashboard\]](#)
- [Bagian \[dcv\]](#)
- [Bagian \[ebs\]](#)
- [Bagian \[efs\]](#)
- [Bagian \[fsx\]](#)
- [Bagian \[queue\]](#)
- [Bagian \[raid\]](#)
- [Bagian \[scaling\]](#)
- [Bagian \[vpc\]](#)
- [Contoh](#)

Tata Letak

AWS ParallelCluster Konfigurasi didefinisikan dalam beberapa bagian.

Bagian berikut diperlukan: [\[global\]bagian](#) dan [\[aws\]bagian](#).

Anda juga harus menyertakan setidaknya satu [\[cluster\]bagian](#) dan satu [\[vpc\]bagian](#).

Bagian dimulai dengan nama bagian dalam tanda kurung, diikuti oleh parameter dan konfigurasi.

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

Bagian **[global]**

Menentukan pilihan konfigurasi global yang terkait pcluster dengan.

```
[global]
```

Topik

- [cluster_template](#)
- [update_check](#)
- [sanity_check](#)

cluster_template

Mendefinisikan nama `cluster` bagian yang digunakan untuk cluster secara default. Untuk informasi tambahan tentang `cluster` bagian, lihat [\[cluster\]bagian](#). Nama cluster harus dimulai dengan huruf, berisi tidak lebih dari 60 karakter, dan hanya berisi huruf, angka, dan tanda hubung (-).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai digunakan `[cluster default]` secara default.

```
cluster_template = default
```

[Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.](#)

update_check

(Opsional) Memeriksa pembaruan kepccluster.

Nilai default-nya adalah true.

```
update_check = true
```

[Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.](#)

sanity_check

(Opsional) Upaya untuk memvalidasi konfigurasi sumber daya yang didefinisikan dalam parameter cluster.

Nilai default-nya adalah true.

Warning

Jika `sanity_check` disetel ke `false`, pemeriksaan penting dilewati. Ini dapat menyebabkan konfigurasi Anda tidak berfungsi sebagaimana dimaksud.

```
sanity_check = true
```

Note

Sebelum AWS ParallelCluster versi 2.5.0, [sanity_check](#) default ke. `false`

[Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.](#)

Bagian [aws]

(Opsional) Digunakan untuk memilih Wilayah AWS.

Pembuatan cluster menggunakan urutan prioritas ini Wilayah AWS untuk memilih klaster baru:

1. `-ratau --region` parameter ke [pcluster create](#).
2. `AWS_DEFAULT_REGION` variabel lingkungan.
3. `aws_region_name` pengaturan di `[aws]` bagian file AWS ParallelCluster konfigurasi (lokasi default adalah `~/.parallelcluster/config`.) Ini adalah lokasi yang diperbarui oleh [pcluster configure](#) perintah.
4. `region` pengaturan di `[default]` bagian file AWS CLI konfigurasi (`~/.aws/config`.)

Note

Sebelum AWS ParallelCluster versi 2.10.0, pengaturan ini diperlukan dan diterapkan ke semua cluster.

Untuk menyimpan kredensial, Anda dapat menggunakan lingkungan, peran IAM untuk Amazon EC2, atau [AWS CLI](#), daripada menyimpan kredensi ke dalam file konfigurasi. AWS ParallelCluster

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

Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.

Bagian **[aliases]**

Menentukan alias, dan memungkinkan Anda untuk menyesuaikan perintah. `ssh`

Perhatikan pengaturan default berikut:

- `CFN_USER` diatur ke nama pengguna default untuk OS
- `MASTER_IP` diatur ke alamat IP node kepala
- `ARGS` disetel ke argumen apa pun yang diberikan pengguna setelahnya `pcluster ssh cluster_name`

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

[Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.](#)

Bagian **[cluster]**

Mendefinisikan template cluster yang dapat digunakan untuk membuat cluster. File konfigurasi dapat berisi beberapa `[cluster]` bagian.

Template cluster yang sama dapat digunakan untuk membuat beberapa cluster.

Formatnya adalah `[cluster cluster-template-name]`. [\[cluster\]Bagian](#) yang dinamai oleh [cluster_template](#) pengaturan di [\[global\]bagian](#) ini digunakan secara default, tetapi dapat diganti pada baris perintah. [pcluster](#)

cluster-template-name harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[cluster default]
```

Topik

- [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

(Opsional) Mendefinisikan AWS CloudFormation template tambahan untuk diluncurkan bersama dengan cluster. Template tambahan ini digunakan untuk membuat sumber daya yang berada di luar cluster tetapi merupakan bagian dari siklus hidup cluster.

Nilai harus berupa URL HTTP ke template publik, dengan semua parameter yang disediakan.

Tidak ada nilai default.

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

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

additional_iam_policies

(Opsional) Menentukan daftar Nama Sumber Daya Amazon (ARNs) kebijakan IAM untuk Amazon. EC2 Daftar ini dilampirkan ke peran root yang digunakan dalam cluster selain izin yang diperlukan oleh AWS ParallelCluster dipisahkan oleh koma. Nama kebijakan IAM dan ARN-nya berbeda. Nama tidak dapat digunakan sebagai argumen untuk `additional_iam_policies`.

Jika maksud Anda adalah menambahkan kebijakan tambahan ke setelan default untuk node kluster, sebaiknya Anda meneruskan kebijakan IAM kustom tambahan dengan `additional_iam_policies` setelan alih-alih menggunakan `ec2_iam_role` pengaturan untuk menambahkan kebijakan spesifik EC2 Anda. Hal ini karena `additional_iam_policies` ditambahkan ke izin default yang AWS ParallelCluster membutuhkan. Yang sudah ada `ec2_iam_role` harus menyertakan semua izin yang diperlukan. Namun, karena izin yang diperlukan sering berubah dari rilis ke rilis saat fitur ditambahkan, yang sudah ada `ec2_iam_role` dapat menjadi usang.

Tidak ada nilai default.

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

Note

Support untuk [additional_iam_policies](#) ditambahkan di AWS ParallelCluster versi 2.5.0.

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

base_os

(Wajib) Menentukan jenis OS yang digunakan dalam cluster.

Pilihan yang tersedia adalah:

- `alinux2`
- `centos7`
- `ubuntu1804`
- `ubuntu2004`

Note

Untuk instance AWS berbasis Graviton, hanya `alinux2` dan `ubuntu1804` yang didukung.

Note

Support untuk `centos8` telah dihapus di AWS ParallelCluster versi 2.11.4. Support untuk `ubuntu2004` ditambahkan dan dukungan untuk `alinux` dan `ubuntu1604` telah dihapus di AWS ParallelCluster versi 2.11.0. Support untuk `centos8` ditambahkan dan dukungan untuk `centos6` telah dihapus di AWS ParallelCluster versi 2.10.0. Support untuk `alinux2` ditambahkan di AWS ParallelCluster versi 2.6.0. Support untuk `ubuntu1804` ditambahkan, dan dukungan untuk `ubuntu1404` telah dihapus di AWS ParallelCluster versi 2.5.0.

Selain spesifik yang Wilayah AWS disebutkan dalam tabel berikut yang tidak mendukung `centos7`. Semua Wilayah AWS komersial lainnya mendukung semua sistem operasi berikut.

Partisi (Wilayah AWS)	alinux2	centos7	ubuntu1804 dan ubuntu2004
Komersil (Semua Wilayah AWS tidak disebutkan secara khusus)	True	Benar	True
AWS GovCloud (AS-Timur) (us-gov-east-1)	True	Salah	True
AWS GovCloud (AS-Barat) (us-gov-west-1)	True	Salah	True
China (Beijing) (cn-north-1)	True	Salah	True
China (Ningxia) (cn-northwest-1)	True	Salah	True

Note

[base_os](#) Parameter juga menentukan nama pengguna yang digunakan untuk masuk ke cluster.

- `centos7`: `centos`
- `ubuntu1804` dan `ubuntu2004`: `ubuntu`
- `alinux2`: `ec2-user`

Note

Sebelum AWS ParallelCluster versi 2.7.0, [base_os](#) parameternya opsional, dan defaultnya adalah. `alinux` Dimulai dengan AWS ParallelCluster versi 2.7.0, [base_os](#) parameter diperlukan.

Note

Jika `scheduler` parameternya `awsbatch`, hanya `alinux2` didukung.

```
base_os = alinux2
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

cluster_resource_bucket

(Opsional) Menentukan nama bucket Amazon S3 yang digunakan untuk meng-host sumber daya yang dihasilkan saat cluster dibuat. Bucket harus mengaktifkan versi. Untuk informasi selengkapnya, lihat [Menggunakan versi](#) di Panduan Pengguna Layanan Penyimpanan Sederhana Amazon. Bucket ini dapat digunakan untuk beberapa cluster. Bucket harus berada di Wilayah yang sama dengan cluster.

Jika parameter ini tidak ditentukan, bucket baru akan dibuat saat cluster dibuat. Ember baru memiliki nama `parallelcluster-random_string`. Dalam nama ini, *random_string* adalah string acak karakter alfanumerik. Semua sumber daya cluster disimpan dalam bucket ini di jalur dengan formulir `bucket_name/resource_directory`. `resource_directory` memiliki bentuk `stack_name-random_string`, di *stack_name* mana nama salah satu AWS CloudFormation tumpukan yang digunakan oleh AWS ParallelCluster. Nilai *bucket_name* dapat ditemukan dalam `ResourcesS3Bucket` nilai dalam output `parallelcluster-clustername` tumpukan. Nilai *resource_directory* dapat ditemukan dalam nilai `ArtifactS3RootDirectory` output dari tumpukan yang sama.

Nilai default-nya adalah `parallelcluster-random_string`.

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

Note

Support untuk `cluster_resource_bucket` ditambahkan dalam AWS ParallelCluster versi 2.10.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan. Memperbarui pengaturan ini tidak dapat dipaksakan.

cluster_type

(Opsional) Mendefinisikan jenis cluster yang akan diluncurkan. Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus diganti dengan [compute_type](#) pengaturan di [\[queue\]bagian](#).

Opsi yang valid adalah:ondemand, danspot.

Nilai default-nya adalah ondemand.

Untuk informasi selengkapnya tentang Instans Spot, lihat[Berkeja dengan Instans Spot](#).

Note

Menggunakan Instans Spot mengharuskan peran `AWSServiceRoleForEC2Spot` terkait layanan ada di akun Anda. Untuk membuat peran ini di akun Anda menggunakan AWS CLI, jalankan perintah berikut:

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

Untuk informasi selengkapnya, lihat [Peran terkait layanan untuk permintaan Instans Spot](#) di EC2 Panduan Pengguna Amazon.

```
cluster_type = ondemand
```

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

compute_instance_type

(Opsional) Mendefinisikan jenis EC2 instans Amazon yang digunakan untuk node komputasi cluster. Arsitektur tipe instance harus sama dengan arsitektur yang digunakan untuk [master_instance_type](#) pengaturan. Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus diganti dengan [instance_type](#) pengaturan di [\[compute_resource\]bagian](#).

Jika Anda menggunakan `awsbatch` penjadwal, lihat pembuatan Lingkungan Komputasi di AWS Batch UI untuk daftar jenis instans yang didukung.

Defaultnya `t2.micro`, optimal saat penjadwal berada. `awsbatch`

```
compute_instance_type = t2.micro
```

Note

Support untuk instance AWS berbasis Graviton (termasuk A1 dan C6g instance) ditambahkan di versi 2.8.0. AWS ParallelCluster

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

compute_root_volume_size

(Opsional) Menentukan ukuran volume ComputeFleet root di gibibytes (GiB). AMI harus mendukung `growroot`.

Nilai default-nya adalah 35.

Note

Untuk AWS ParallelCluster versi antara 2.5.0 dan 2.10.4, defaultnya adalah 25. Sebelum AWS ParallelCluster versi 2.5.0, defaultnya adalah 20.

```
compute_root_volume_size = 35
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

custom_ami

(Opsional) [Menentukan ID AMI kustom yang akan digunakan untuk node head dan compute, bukan default yang diterbitkan. AMIs](#) Untuk informasi selengkapnya, lihat [Memodifikasi AMI](#) atau [Bangun AWS ParallelCluster AMI Kustom.](#)

Tidak ada nilai default.

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

Jika AMI kustom memerlukan izin tambahan untuk peluncurannya, izin ini harus ditambahkan ke kebijakan pengguna dan node kepala.

Misalnya, jika AMI kustom memiliki snapshot terenkripsi yang terkait dengannya, kebijakan tambahan berikut diperlukan dalam kebijakan pengguna dan node kepala:

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

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

cw_log_settings

(Opsional) Mengidentifikasi [cw_log] bagian dengan konfigurasi CloudWatch Log. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[cw_log\] bagian](#), [CloudWatch Dasbor Amazon](#), dan [Integrasi dengan Amazon CloudWatch Logs](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai [cw_log custom-cw] digunakan untuk konfigurasi CloudWatch Log.

```
cw_log_settings = custom-cw
```

 Note

Support untuk [cw_log_settings](#) ditambahkan di AWS ParallelCluster versi 2.6.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

dashboard_settings

(Opsional) Mengidentifikasi [dashboard] bagian dengan konfigurasi CloudWatch dasbor. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[dashboard\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai [dashboard custom-dashboard digunakan untuk konfigurasi CloudWatch dasbor.

```
dashboard_settings = custom-dashboard
```

 Note

Support untuk [dashboard_settings](#) ditambahkan dalam AWS ParallelCluster versi 2.10.0.

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

dcv_settings

(Opsional) Mengidentifikasi [dcv] bagian dengan konfigurasi Amazon DCV. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[dcv\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai [dcv custom-dcv] digunakan untuk konfigurasi Amazon DCV.

```
dcv_settings = custom-dcv
```

Note

Pada instans AWS berbasis Graviton, Amazon DCV hanya didukung pada. `alinux2`

Note

Support untuk [dcv_settings](#) ditambahkan di AWS ParallelCluster versi 2.5.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

desired_vcpus

(Opsional) Menentukan jumlah v yang diinginkan CPUs di lingkungan komputasi. Digunakan hanya jika penjadwal. `awsbatch`

Nilai default-nya adalah 4.

```
desired_vcpus = 4
```

Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.

disable_cluster_dns

(Opsional) Menentukan apakah entri DNS untuk cluster tidak boleh dibuat. Secara default, AWS ParallelCluster membuat zona yang dihosting Route 53. Jika `disable_cluster_dns` disetel `true`, zona yang dihosting tidak dibuat.

Nilai default-nya adalah `false`.

```
disable_cluster_dns = true
```

⚠ Warning

Sistem resolusi nama diperlukan agar cluster dapat beroperasi dengan benar. Jika `disable_cluster_dns` diatur ke `true`, sistem resolusi nama tambahan juga harus disediakan.

⚠ Important

[disable_cluster_dns](#) = `true` hanya didukung jika [queue_settings](#) pengaturan ditentukan.

i Note

Support untuk [disable_cluster_dns](#) ditambahkan di AWS ParallelCluster versi 2.9.1.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

disable_hyperthreading

(Opsional) Menonaktifkan hyperthreading di kepala dan menghitung node. Tidak semua tipe instance dapat menonaktifkan hyperthreading. Untuk daftar jenis instans yang mendukung penonaktifan hyperthreading, lihat [inti CPU dan thread untuk setiap inti CPU untuk setiap jenis instans di Panduan Pengguna Amazon EC2](#). Jika [queue_settings](#) pengaturan ditentukan, pengaturan ini dapat ditentukan, atau [disable_hyperthreading](#) pengaturan di [\[queue\]bagian](#) dapat ditentukan.

Nilai default-nya adalah `false`.

```
disable_hyperthreading = true
```

i Note

[disable_hyperthreading](#) hanya mempengaruhi simpul kepala saat [scheduler](#) = `awsbatch`.

Note

Support untuk [disable_hyperthreading](#) ditambahkan di AWS ParallelCluster versi 2.5.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

ebs_settings

(Opsional) Mengidentifikasi [ebs] bagian dengan volume Amazon EBS yang dipasang pada node kepala. Saat menggunakan beberapa volume Amazon EBS, masukkan parameter ini dalam daftar dengan masing-masing dipisahkan dengan koma. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Hingga lima (5) volume Amazon EBS tambahan didukung.

Untuk informasi lebih lanjut, lihat [\[ebs\]bagian](#).

Misalnya, setelan berikut menentukan bahwa bagian yang dimulai [ebs custom1] dan [ebs custom2] digunakan untuk volume Amazon EBS.

```
ebs_settings = custom1, custom2
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

ec2_iam_role

(Opsional) Mendefinisikan nama peran IAM yang ada untuk Amazon EC2 yang dilampirkan ke semua instance di cluster. Nama peran IAM dan Nama Sumber Daya Amazon (ARN) berbeda. ARNs tidak dapat digunakan sebagai argumen untuk `ec2_iam_role`.

Jika opsi ini ditentukan, [additional_iam_policies](#) pengaturan diabaikan. Jika maksud Anda adalah menambahkan kebijakan tambahan ke setelan default untuk node kluster, sebaiknya Anda meneruskan kebijakan IAM kustom tambahan dengan [additional_iam_policies](#) setelan alih-alih menggunakan pengaturan. `ec2_iam_role`

Jika opsi ini tidak ditentukan, peran AWS ParallelCluster IAM default untuk Amazon akan EC2 digunakan. Untuk informasi selengkapnya, lihat [AWS Identity and Access Management peran dalam AWS ParallelCluster](#).

Tidak ada nilai default.

```
ec2_iam_role = ParallelClusterInstanceRole
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

efs_settings

(Opsional) Menentukan pengaturan yang terkait dengan sistem file Amazon EFS. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[efs\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai [efs customfs] digunakan untuk konfigurasi sistem file Amazon EFS.

```
efs_settings = customfs
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

enable_efa

(Opsional) Jika ada, tentukan bahwa Elastic Fabric Adapter (EFA) diaktifkan untuk node komputasi. Untuk melihat daftar EC2 instance yang mendukung EFA, lihat [Jenis instans yang didukung](#) di Panduan EC2 Pengguna Amazon untuk Instans Linux. Untuk informasi selengkapnya, lihat [Elastic Fabric Adapter](#). Jika [queue_settings](#) pengaturan ditentukan, pengaturan ini dapat ditentukan, atau [enable_efa](#) pengaturan di [\[queue\]bagian](#) dapat ditentukan. Grup penempatan cluster harus digunakan untuk meminimalkan latensi antar instance. Untuk informasi selengkapnya, lihat [placement](#) dan [placement_group](#).

```
enable_efa = compute
```

Note

Support untuk EFA pada instance Graviton2 berbasis ARM telah ditambahkan di versi 2.10.1. AWS ParallelCluster

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

enable_efa_gdr

(Opsional) Dimulai dengan AWS ParallelCluster versi 2.11.3, pengaturan ini tidak berpengaruh. Dukungan Elastic Fabric Adapter (EFA) untuk GPUDirect RDMA (akses memori langsung jarak jauh) selalu diaktifkan jika didukung oleh jenis instans dan sistem operasi.

Note

AWS ParallelCluster versi 2.10.0 hingga 2.11.2: Jika `compute`, menentukan bahwa dukungan Elastic Fabric Adapter (EFA) GPUDirect untuk RDMA (akses memori langsung jarak jauh) diaktifkan untuk node komputasi. Menyetel pengaturan ini untuk `compute` mengharuskan [enable_efa](#) pengaturan diatur ke `compute`. Dukungan EFA untuk GPUDirect RDMA didukung oleh jenis instance tertentu (p4d.24xlarge) pada sistem operasi tertentu (`base_osisalinux2`, `centos7ubuntu1804`, atau `ubuntu2004`). Jika [queue_settings](#) pengaturan ditentukan, pengaturan ini dapat ditentukan, atau [enable_efa_gdr](#) pengaturan di [\[queue\]bagian](#) dapat ditentukan. Grup penempatan cluster harus digunakan untuk meminimalkan latensi antar instance. Untuk informasi selengkapnya, lihat [placement](#) dan [placement_group](#).

```
enable_efa_gdr = compute
```

Note

Support untuk `enable_efa_gdr` ditambahkan dalam AWS ParallelCluster versi 2.10.0.

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

enable_intel_hpc_platform

(Opsional) Jika ada, menunjukkan bahwa [perjanjian lisensi pengguna akhir](#) untuk Intel Parallel Studio diterima. Hal ini menyebabkan Intel Parallel Studio diinstal pada node kepala dan dibagikan dengan node komputasi. Ini menambahkan beberapa menit ke waktu yang dibutuhkan node kepala untuk

bootstrap. [enable_intel_hpc_platform](#) Pengaturan hanya didukung pada CentOS 7 ([base_os](#) = centos7).

Nilai default-nya adalah false.

```
enable_intel_hpc_platform = true
```

 Note

[enable_intel_hpc_platform](#) Parameter tidak kompatibel dengan instance berbasis AWS Graviton.

 Note

Support untuk [enable_intel_hpc_platform](#) ditambahkan di AWS ParallelCluster versi 2.5.0.

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

encrypted_ephemeral

(Opsional) Mengenkripsi volume penyimpanan instans singkat dengan kunci dalam memori yang tidak dapat dipulihkan, menggunakan LUKS (Linux Unified Key Setup).

Untuk informasi selengkapnya, lihat <https://gitlab.com/cryptsetup/cryptsetup/blob/master/README.md>.

Nilai default-nya adalah false.

```
encrypted_ephemeral = true
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

ephemeral_dir

(Opsional) Mendefinisikan jalur di mana volume penyimpanan instance dipasang jika digunakan.

Nilai default-nya adalah `/scratch`.

```
ephemeral_dir = /scratch
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

extra_json

(Opsional) Mendefinisikan JSON tambahan yang digabungkan ke dalam `chef-dna.json`. Untuk informasi selengkapnya, lihat [Membangun AWS ParallelCluster AMI Kustom](#).

Nilai default-nya adalah `{}`.

```
extra_json = {}
```

Note

Dimulai dengan AWS ParallelCluster versi 2.6.1, sebagian besar resep penginstalan dilewati secara default saat meluncurkan node untuk meningkatkan waktu mulai. Untuk menjalankan semua resep penginstalan untuk kompatibilitas mundur yang lebih baik dengan mengorbankan waktu startup, tambahkan `"skip_install_recipes" : "no"` ke `cluster` kunci dalam [extra_json](#) pengaturan. Misalnya:

```
extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
```

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

fsx_settings

(Opsional) Menentukan bagian yang mendefinisikan FSx untuk konfigurasi Lustre. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[fsx\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai `[fsx fs]` digunakan untuk konfigurasi FSx for Lustre.

```
fsx_settings = fs
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

iam_lambda_role

(Opsional) Mendefinisikan nama peran AWS Lambda eksekusi yang ada. Peran ini melekat pada semua fungsi Lambda di cluster. Untuk informasi selengkapnya, lihat [peran AWS Lambda eksekusi](#) di Panduan AWS Lambda Pengembang.

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Nama peran IAM dan Nama Sumber Daya Amazon (ARN) berbeda. ARNs tidak dapat digunakan sebagai argumen untuk `iam_lambda_role`. Jika `iam_lambda_role` keduanya [ec2_iam_role](#) dan didefinisikan, dan [scheduler](#) adalah `sge`, `slurm`, atau `torque`, maka tidak akan ada peran yang dibuat. Jika [scheduler](#) adalah `awsbatch`, maka akan ada peran yang dibuat selama [pcluster start](#). Misalnya kebijakan, lihat [ParallelClusterLambdaPolicy menggunakan SGE, Slurm, atau Torque](#) dan [ParallelClusterLambdaPolicy menggunakan awsbatch](#).

Tidak ada nilai default.

```
iam_lambda_role = ParallelClusterLambdaRole
```

Note

Support untuk `iam_lambda_role` ditambahkan di AWS ParallelCluster versi 2.10.1.

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

initial_queue_size

(Opsional) Menetapkan jumlah awal EC2 instans Amazon yang akan diluncurkan sebagai node komputasi di cluster. Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus dihapus dan diganti dengan [initial_count](#) pengaturan di [\[compute_resource\]bagian](#).

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Pengaturan ini hanya berlaku untuk penjadwal tradisional (SGE, Slurm, dan Torque). Jika [maintain_initial_size](#) pengaturannya `true`, maka [initial_queue_size](#) pengaturannya harus minimal satu (1).

Jika penjadwal adalah `awsbatch`, gunakan [min_vcpus](#) sebagai gantinya.

Default ke 2.

```
initial_queue_size = 2
```

[Kebijakan pembaruan](#): Pengaturan ini dapat diubah selama pembaruan.

key_name

(Opsional) Menamai EC2 key pair Amazon yang sudah ada untuk mengaktifkan akses SSH ke instance.

```
key_name = mykey
```

Note

Sebelum AWS ParallelCluster versi 2.11.0, `key_name` adalah pengaturan yang diperlukan.

[Kebijakan pembaruan](#): Jika pengaturan ini diubah, pembaruan tidak diizinkan.

maintain_initial_size

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

(Opsional) Mempertahankan ukuran awal grup Auto Scaling untuk penjadwal tradisional (SGE, Slurm, dan Torque).

Jika penjadwal adalah `awsbatch`, gunakan [desired_vcpus](#) sebagai gantinya.

Pengaturan ini adalah bendera Boolean. Jika disetel ke `true`, grup Auto Scaling tidak pernah memiliki anggota yang lebih sedikit daripada nilai [initial_queue_size](#), dan nilai [initial_queue_size](#) harus satu (1) atau lebih besar. Cluster masih dapat meningkatkan nilai [max_queue_size](#). Jika `cluster_type = spot` kemudian grup Auto Scaling dapat memiliki instance terputus dan ukurannya bisa turun di bawah. [initial_queue_size](#)

Jika disetel ke `false`, grup Auto Scaling dapat menurunkan skala ke nol (0) anggota untuk mencegah sumber daya diam saat tidak diperlukan.

Jika [queue_settings](#) pengaturan ditentukan maka pengaturan ini harus dihapus dan diganti dengan [initial_count](#) dan [min_count](#) pengaturan di [\[compute_resource\]](#) bagian.

Default ke `false`.

```
maintain_initial_size = false
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

master_instance_type

(Opsional) Mendefinisikan jenis EC2 instans Amazon yang digunakan untuk node kepala. Arsitektur tipe instance harus sama dengan arsitektur yang digunakan untuk [compute_instance_type](#) pengaturan.

Dalam Wilayah AWS yang memiliki Tingkat Gratis, default ke jenis instans Tingkat Gratis (t2.micro atau t3.micro). Dalam hal Wilayah AWS itu tidak memiliki Tingkat Gratis, defaultnya t3.micro. Untuk informasi selengkapnya tentang Tingkat AWS Gratis, lihat [Tingkat AWS Gratis FAQs](#).

```
master_instance_type = t2.micro
```

Note

Sebelum AWS ParallelCluster versi 2.10.1, default ke semua t2.micro Wilayah AWS. Di AWS ParallelCluster versi 2.10.0, p4d.24xlarge tidak didukung untuk node kepala. Support

untuk instance AWS berbasis Graviton (seperti A1 dan C6g) ditambahkan di versi 2.8.0. AWS ParallelCluster

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

master_root_volume_size

(Opsional) Menentukan ukuran volume akar simpul kepala di gibibytes (GiB). AMI harus mendukung `growroot`.

Nilai default-nya adalah 35.

Note

Untuk AWS ParallelCluster versi antara 2.5.0 dan 2.10.4, defaultnya adalah 25. Sebelum AWS ParallelCluster versi 2.5.0, defaultnya adalah 20.

```
master_root_volume_size = 35
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

max_queue_size

(Opsional) Menetapkan jumlah maksimum EC2 instans Amazon yang dapat diluncurkan di cluster. Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus dihapus dan diganti dengan [max_count](#) pengaturan di [\[compute_resource\]bagian](#).

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Pengaturan ini hanya berlaku untuk penjadwal tradisional (SGE, Slurm, dan Torque).

Jika penjadwal adalah `awsbatch`, gunakan [max_vcpus](#) sebagai gantinya.

Default ke 10.

```
max_queue_size = 10
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan, tetapi armada komputasi harus dihentikan jika nilainya berkurang. Jika tidak, node yang ada dapat dihentikan.

max_vcpus

(Opsional) Menentukan jumlah maksimum v CPUs di lingkungan komputasi. Digunakan hanya jika penjadwal. `awsbatch`

Nilai default-nya adalah 20.

```
max_vcpus = 20
```

[Kebijakan pembaruan: Pengaturan ini tidak dapat dikurangi selama pembaruan.](#)

min_vcpus

(Opsional) Mempertahankan ukuran awal grup Auto Scaling untuk penjadwal. `awsbatch`

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Jika penjadwal adalah SGE, Slurm, atau Torque, gunakan [maintain_initial_size](#) sebagai gantinya.

Lingkungan komputasi tidak pernah memiliki anggota yang lebih sedikit daripada nilai. [min_vcpus](#)

Default ke 0.

```
min_vcpus = 0
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

placement

(Opsional) Mendefinisikan logika grup penempatan cluster, memungkinkan seluruh cluster atau hanya instance komputasi untuk menggunakan grup penempatan cluster.

Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus dihapus dan diganti dengan [placement_group](#) pengaturan untuk masing-masing [\[queue\]bagian](#). Jika grup penempatan yang sama digunakan untuk jenis instans yang berbeda, kemungkinan besar permintaan tersebut mungkin gagal karena kesalahan kapasitas yang tidak mencukupi. Untuk informasi selengkapnya, lihat [Kapasitas instans tidak mencukupi](#) di Panduan EC2 Pengguna Amazon. Beberapa antrian hanya dapat berbagi grup penempatan jika dibuat sebelumnya dan dikonfigurasi dalam [placement_group](#) pengaturan untuk setiap antrian. Jika setiap [\[queue\]bagian](#) mendefinisikan [placement_group](#) pengaturan, maka node kepala tidak dapat berada di grup penempatan untuk antrian.

Pilihan yang valid adalah `cluster` atau `compute`.

Parameter ini tidak digunakan saat penjadwal. `awsbatch`

Nilai default-nya adalah `compute`.

```
placement = compute
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

placement_group

(Opsional) Mendefinisikan kelompok penempatan cluster. Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus dihapus dan diganti dengan [placement_group](#) pengaturan di [\[queue\]bagian](#).

Opsi yang valid adalah nilai-nilai berikut:

- DYNAMIC
- Nama grup penempatan EC2 klaster Amazon yang ada

Ketika diatur ke DYNAMIC, grup penempatan unik dibuat dan dihapus sebagai bagian dari tumpukan cluster.

Parameter ini tidak digunakan saat penjadwal. `awsbatch`

Untuk informasi selengkapnya tentang grup [penempatan](#), lihat [Grup penempatan](#) di Panduan EC2 Pengguna Amazon. Jika grup penempatan yang sama digunakan untuk jenis instans yang berbeda, kemungkinan besar permintaan tersebut mungkin gagal karena kesalahan kapasitas yang tidak mencukupi. Untuk informasi selengkapnya, lihat [Kapasitas instans tidak mencukupi](#) di Panduan EC2 Pengguna Amazon.

Tidak ada nilai default.

Tidak semua tipe instance mendukung grup penempatan klaster. Misalnya, tipe instance default `t3.micro` tidak mendukung grup penempatan klaster. Untuk informasi tentang daftar jenis instans yang mendukung grup penempatan klaster, lihat [Aturan dan batasan grup penempatan klaster](#) di Panduan EC2 Pengguna Amazon. Lihat [Grup penempatan dan masalah peluncuran instance](#) tip saat bekerja dengan grup penempatan.

```
placement_group = DYNAMIC
```

[Kebijakan pembaruan](#): Jika pengaturan ini diubah, pembaruan tidak diizinkan.

post_install

(Opsional) Menentukan URL skrip pasca-instal yang dijalankan setelah semua tindakan bootstrap node selesai. Untuk informasi selengkapnya, lihat [Tindakan Bootstrap Kustom](#).

Saat menggunakan `awsbatch` sebagai penjadwal, skrip pasca-instal dijalankan hanya pada node kepala.

Format parameter dapat berupa `http://hostname/path/to/script.sh` atau `s3://bucket-name/path/to/script.sh`.

Tidak ada nilai default.

```
post_install = s3://<bucket-name>/my-post-install-script.sh
```

[Kebijakan pembaruan](#): Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

post_install_args

(Opsional) Menentukan daftar argumen yang dikutip untuk diteruskan ke skrip pasca-instal.

Tidak ada nilai default.

```
post_install_args = "argument-1 argument-2"
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

pre_install

(Opsional) Menentukan URL skrip pra-instal yang dijalankan sebelum tindakan bootstrap penyebaran node dimulai. Untuk informasi selengkapnya, lihat [Tindakan Bootstrap Kustom](#).

Saat menggunakan awsbatch sebagai penjadwal, skrip pra-instal dijalankan hanya pada node kepala.

Format parameter dapat berupa `http://hostname/path/to/script.sh` atau `s3://bucket-name/path/to/script.sh`.

Tidak ada nilai default.

```
pre_install = s3://bucket-name/my-pre-install-script.sh
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

pre_install_args

(Opsional) Menentukan daftar kutipan argumen untuk diteruskan ke skrip pra-instal.

Tidak ada nilai default.

```
pre_install_args = "argument-3 argument-4"
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

proxy_server

(Opsional) Mendefinisikan server proxy HTTP atau HTTPS, biasanya `http://x.x.x.x:8080`.

Tidak ada nilai default.

```
proxy_server = http://10.11.12.13:8080
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

queue_settings

(Opsional) [Menentukan bahwa cluster menggunakan antrian bukan armada komputasi homogen, dan bagian mana yang digunakan](#)[\[queue\]](#). [\[queue\]](#)Bagian pertama yang tercantum adalah antrian penjadwal default. Nama queue bagian harus dimulai dengan huruf kecil, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf kecil, angka, dan tanda hubung (-).

⚠ Important

[queue_settings](#) hanya didukung ketika [scheduler](#) disetel ke `slurm`.

[spot_price](#) Pengaturan

[cluster_type](#) `compute_instance_type`, `initial_queue_size`, `maintain_initial_size`, `max`

[placement](#) `placement_group`, dan tidak boleh ditentukan. [enable_efa](#) Pengaturan

[disable_hyperthreading](#) dan dapat ditentukan di bagian atau

[\[cluster\]](#) [\[queue\]](#) bagian, tetapi tidak keduanya.

Hingga lima (5) [\[queue\]](#) bagian didukung.

Untuk informasi lebih lanjut, lihat [\[queue\]](#) bagian.

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai `[queue q1]` dan `[queue q2]` digunakan.

```
queue_settings = q1, q2
```

ℹ Note

Support untuk [queue_settings](#) ditambahkan di AWS ParallelCluster versi 2.9.0.

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

raid_settings

(Opsional) Mengidentifikasi [raid] bagian dengan konfigurasi RAID volume Amazon EBS. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[raid\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang mulai [raid rs] digunakan untuk konfigurasi Auto Scaling.

```
raid_settings = rs
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

s3_read_resource

(Opsional) Menentukan sumber daya Amazon S3 AWS ParallelCluster yang node diberikan akses hanya-baca.

Misalnya, `arn:aws:s3:::my_corporate_bucket*` menyediakan akses read-only ke `my_corporate_bucket` bucket dan objek di bucket.

Lihat [bekerja dengan Amazon S3](#) untuk detail tentang format.

Tidak ada nilai default.

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

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

s3_read_write_resource

(Opsional) Menentukan sumber daya Amazon S3 AWS ParallelCluster yang node diberikan akses baca/tulis.

Misalnya, `arn:aws:s3:::my_corporate_bucket/Development/*` menyediakan akses baca/tulis ke semua objek di Development folder ember `my_corporate_bucket`.

Lihat [bekerja dengan Amazon S3](#) untuk detail tentang format.

Tidak ada nilai default.

```
s3_read_write_resource = arn:aws:s3:::my_corporate_bucket/*
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

scaling_settings

Mengidentifikasi [scaling] bagian dengan konfigurasi Auto Scaling. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[scaling\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai [scaling custom] digunakan untuk konfigurasi Auto Scaling.

```
scaling_settings = custom
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

scheduler

(Wajib) Mendefinisikan penjadwal cluster.

Opsi yang valid adalah nilai-nilai berikut:

awsbatch

AWS Batch

Untuk informasi selengkapnya tentang awsbatch penjadwal, lihat [penyiapan jaringan](#) dan [AWS Batch \(awsbatch\)](#).

sge

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Son of Grid Engine (SGE)

slurm

Slurm Workload Manager (Slurm)

torque

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Torque Resource Manager (Torque)

Note

Sebelum AWS ParallelCluster versi 2.7.0, `scheduler` parameternya opsional, dan defaultnya adalah `sgs`. Dimulai dengan AWS ParallelCluster versi 2.7.0, `scheduler` parameter diperlukan.

```
scheduler = slurm
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

shared_dir

(Opsional) Mendefinisikan jalur tempat volume Amazon EBS bersama dipasang.

Jangan gunakan opsi ini dengan beberapa volume Amazon EBS. Sebagai gantinya, berikan `shared_dir` nilai di bawah setiap [\[ebs\]bagian](#).

Lihat [\[ebs\]bagian](#) untuk detail tentang bekerja dengan beberapa volume Amazon EBS.

Nilai default-nya adalah `/shared`.

Contoh berikut menunjukkan volume Amazon EBS bersama yang dipasang di `/myshared`.

```
shared_dir = myshared
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

spot_bid_percentage

(Opsional) Menetapkan persentase sesuai permintaan yang digunakan untuk menghitung harga Spot maksimum untuk ComputeFleet, kapan awsbatch adalah penjadwal.

Jika tidak ditentukan, harga pasar spot saat ini dipilih, dibatasi pada harga On-Demand.

```
spot_bid_percentage = 85
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

spot_price

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

(Opsional) Menetapkan harga Spot maksimum untuk ComputeFleet penjadwal tradisional (SGE, Slurm, dan Torque). Digunakan hanya ketika [cluster_type](#) pengaturan diatur kespot. Jika Anda tidak menentukan nilai, Anda akan dikenakan harga Spot, dibatasi pada harga On-Demand. Jika [queue_settings](#) pengaturan ditentukan, maka pengaturan ini harus dihapus dan diganti dengan [spot_price](#) pengaturan di [\[compute_resource\]bagian](#).

Jika schedulernyaawsbatch, gunakan [spot_bid_percentage](#) sebagai gantinya.

Untuk bantuan menemukan Instans Spot yang memenuhi kebutuhan Anda, lihat [penasihat Instans Spot](#).

```
spot_price = 1.50
```

Note

Di AWS ParallelCluster versi 2.5.0, jika `cluster_type = spot` tetapi [spot_price](#) tidak ditentukan, instance akan meluncurkan kegagalan. ComputeFleet Ini diperbaiki di AWS ParallelCluster versi 2.5.1.

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

tags

(Opsional) Mendefinisikan tag yang akan digunakan oleh AWS CloudFormation.

Jika tag baris perintah ditentukan melalui `--tags`, mereka digabungkan dengan tag konfigurasi.

Tag baris perintah menimpa tag konfigurasi yang memiliki kunci yang sama.

Tag diformat JSON. Jangan gunakan tanda kutip di luar kurung kurawal.

Untuk informasi selengkapnya, lihat [jenis tag AWS CloudFormation sumber daya](#) di Panduan AWS CloudFormation Pengguna.

```
tags = {"key" : "value", "key2" : "value2"}
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

Note

Kebijakan pembaruan tidak mendukung perubahan tags pengaturan untuk AWS ParallelCluster versi 2.8.0 hingga versi 2.9.1.

Untuk versi 2.10.0 hingga versi 2.11.7, kebijakan pembaruan yang tercantum yang mendukung perubahan tags setelah tidak akurat. Pembaruan kluster saat memodifikasi setelah ini tidak didukung.

template_url

(Opsional) Mendefinisikan jalur ke AWS CloudFormation template yang digunakan untuk membuat cluster.

Pembaruan menggunakan template yang awalnya digunakan untuk membuat tumpukan.

Default ke `https://aws_region_name-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-version.cfn.json`.

⚠ Warning

Ini adalah parameter lanjutan. Setiap perubahan pada pengaturan ini dilakukan dengan risiko Anda sendiri.

```
template_url = https://us-east-1-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-2.11.9.cfn.json
```

Kebijakan pembaruan: Pengaturan ini tidak dianalisis selama pembaruan.

vpc_settings

(Wajib) Mengidentifikasi [vpc] bagian dengan konfigurasi VPC Amazon tempat cluster digunakan. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Untuk informasi lebih lanjut, lihat [\[vpc\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai [vpc public] digunakan untuk konfigurasi VPC Amazon.

```
vpc_settings = public
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

Bagian [compute_resource]

Mendefinisikan pengaturan konfigurasi untuk sumber daya komputasi.

[\[compute_resource\]bagian](#) direferensikan oleh [compute_resource_settings](#) pengaturan di [\[queue\]bagian](#). [\[compute_resource\]bagian](#) hanya didukung ketika [scheduler](#) diatur keslurm.

Formatnya adalah [compute_resource *<compute-resource-name>*]. *compute-resource-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[compute_resource cr1]
instance_type = c5.xlarge
min_count = 0
initial_count = 2
```

```
max_count = 10
spot_price = 0.5
```

Note

Support untuk [\[compute_resource\]bagian](#) ditambahkan dalam AWS ParallelCluster versi 2.9.0.

Topik

- [initial_count](#)
- [instance_type](#)
- [max_count](#)
- [min_count](#)
- [spot_price](#)

initial_count

(Opsional) Menetapkan jumlah awal EC2 instans Amazon yang akan diluncurkan untuk sumber daya komputasi ini. Pembuatan cluster tidak selesai sampai setidaknya banyak node ini diluncurkan ke sumber daya komputasi. Jika [compute_type](#) pengaturan untuk antrean spot dan tidak ada cukup Instans Spot yang tersedia, pembuatan klaster mungkin habis dan gagal. Hitungan apa pun yang lebih besar dari [min_count](#) pengaturan adalah kapasitas dinamis yang tunduk pada [scaledown_idletime](#) pengaturan. Pengaturan ini menggantikan pengaturan [initial_queue_size](#).

Default ke 0.

```
initial_count = 2
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

instance_type

(Wajib) Mendefinisikan jenis EC2 instans Amazon yang digunakan untuk sumber daya komputasi ini. Arsitektur tipe instance harus sama dengan arsitektur yang digunakan untuk

[master_instance_type](#) pengaturan. `instance_type` Pengaturan harus unik untuk setiap [\[compute_resource\]bagian](#) yang direferensikan oleh [\[queue\]bagian](#). Pengaturan ini menggantikan pengaturan [compute_instance_type](#).

```
instance_type = t2.micro
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

max_count

(Opsional) Menetapkan jumlah maksimum EC2 instans Amazon yang dapat diluncurkan di sumber daya komputasi ini. Hitungan apa pun yang lebih besar dari [initial_count](#) pengaturan dimulai dalam mode power down. Pengaturan ini menggantikan pengaturan [max_queue_size](#).

Default ke 10.

```
max_count = 10
```

[Kebijakan pembaruan: Mengurangi ukuran antrian di bawah jumlah node saat ini mengharuskan armada komputasi dihentikan terlebih dahulu.](#)

Note

Kebijakan pembaruan tidak mendukung perubahan `max_count` pengaturan hingga armada komputasi dihentikan untuk AWS ParallelCluster versi 2.0.0 hingga versi 2.9.1.

min_count

(Opsional) Menetapkan jumlah minimum EC2 instans Amazon yang dapat diluncurkan di sumber daya komputasi ini. Node ini semua kapasitas statis. Pembuatan cluster tidak selesai sampai setidaknya jumlah node ini diluncurkan ke sumber daya komputasi.

Default ke 0.

```
min_count = 1
```

Kebijakan pembaruan: Mengurangi jumlah node statis dalam antrian mengharuskan armada komputasi dihentikan terlebih dahulu.

Note

Kebijakan pembaruan tidak mendukung perubahan `min_count` pengaturan hingga armada komputasi dihentikan untuk AWS ParallelCluster versi 2.0.0 hingga versi 2.9.1.

spot_price

(Opsional) Menetapkan harga Spot maksimum untuk sumber daya komputasi ini. Digunakan hanya ketika `compute_type` pengaturan untuk antrian yang berisi sumber daya komputasi ini diatur ke `spot`. Pengaturan ini menggantikan pengaturan `spot_price`.

Jika Anda tidak menentukan nilai, Anda akan dikenakan harga Spot, dibatasi pada harga Sesuai Permintaan.

Untuk bantuan menemukan Instans Spot yang memenuhi kebutuhan Anda, lihat [penasihat Instans Spot](#).

```
spot_price = 1.50
```

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

Bagian [cw_log]

Mendefinisikan pengaturan konfigurasi untuk CloudWatch Log.

Formatnya adalah `[cw_log cw-log-name]`. *cw-log-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

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

Lihat informasi selengkapnya di [Integrasi dengan Amazon CloudWatch Logs](#), [CloudWatch Dasbor Amazon](#), dan [Integrasi dengan Amazon CloudWatch Logs](#).

Note

Support untuk `cw_log` ditambahkan di AWS ParallelCluster versi 2.6.0.

enable

(Opsional) Menunjukkan apakah CloudWatch Log diaktifkan.

Nilai default-nya adalah `true`. Gunakan `false` untuk menonaktifkan CloudWatch Log.

Contoh berikut memungkinkan CloudWatch Log.

```
enable = true
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

retention_days

(Opsional) Menunjukkan berapa hari CloudWatch Log mempertahankan peristiwa log individu.

Nilai default-nya adalah 14. Nilai yang didukung adalah 1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827, dan 3653.

Contoh berikut mengonfigurasi CloudWatch Log untuk mempertahankan peristiwa log selama 30 hari.

```
retention_days = 30
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

Bagian [dashboard]

Mendefinisikan pengaturan konfigurasi untuk CloudWatch dasbor.

Formatnya adalah `[dashboard dashboard-name]`. *dashboard-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

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

Note

Support untuk dashboard ditambahkan dalam AWS ParallelCluster versi 2.10.0.

enable

(Opsional) Menunjukkan apakah CloudWatch dasbor diaktifkan.

Nilai default-nya adalah `true`. Gunakan `false` untuk menonaktifkan CloudWatch dasbor.

Contoh berikut memungkinkan CloudWatch dasbor.

```
enable = true
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

Bagian [dcv]

Mendefinisikan pengaturan konfigurasi untuk server Amazon DCV yang berjalan di node kepala.

Untuk membuat dan mengonfigurasi server Amazon DCV, tentukan cluster [dcv_settings](#) dengan nama yang Anda tentukan di dcv bagian, dan atur [enable](#) kemaster, dan [base_os](#) ke `linux2,centos7, ubuntu1804` atau `ubuntu2004`. Jika node kepala adalah instance ARM, atur [base_os](#) ke `linux2,centos7, atau ubuntu1804`.

Formatnya adalah `[dcv dcv-name]`. *dcv-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

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

Untuk informasi selengkapnya, lihat [Connect ke head node melalui Amazon DCV](#)

⚠ Important

Secara default, pengaturan port Amazon DCV terbuka untuk semua IPv4 alamat. AWS ParallelCluster Namun, Anda dapat terhubung ke port Amazon DCV hanya jika Anda memiliki URL untuk sesi Amazon DCV dan terhubung ke sesi Amazon DCV dalam waktu 30 detik sejak URL dikembalikan. `pcluster dcv connect` Gunakan [access_from](#) pengaturan untuk lebih membatasi akses ke port Amazon DCV dengan rentang IP berformat CIDR, dan gunakan [port](#) pengaturan untuk menyetel port yang tidak standar.

ℹ Note

Support untuk [\[dcv\]bagian](#) pada centos8 telah dihapus di AWS ParallelCluster versi 2.10.4. Support untuk [\[dcv\]bagian](#) pada centos8 ditambahkan dalam AWS ParallelCluster versi 2.10.0. Support untuk [\[dcv\]bagian](#) tentang instance AWS berbasis Graviton telah ditambahkan di versi 2.9.0. AWS ParallelCluster Support untuk [\[dcv\]bagian](#) aktif a1linux2 dan ubuntu1804 ditambahkan dalam AWS ParallelCluster versi 2.6.0. Support untuk [\[dcv\]bagian](#) pada centos7 ditambahkan dalam AWS ParallelCluster versi 2.5.0.

access_from

(Opsional, Disarankan) Menentukan rentang IP berformat CIDR untuk koneksi ke Amazon DCV. Pengaturan ini hanya digunakan saat AWS ParallelCluster membuat grup keamanan.

Nilai defaultnya adalah `0.0.0.0/0`, yang memungkinkan akses dari alamat internet apa pun.

```
access_from = 0.0.0.0/0
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

enable

(Wajib) Menunjukkan apakah Amazon DCV diaktifkan pada node kepala. Untuk mengaktifkan Amazon DCV di node kepala dan mengonfigurasi aturan grup keamanan yang diperlukan, setel `enable` pengaturan ke `master`.

Contoh berikut memungkinkan Amazon DCV pada node kepala.

```
enable = master
```

Note

Amazon DCV secara otomatis menghasilkan sertifikat yang ditandatangani sendiri yang digunakan untuk mengamankan lalu lintas antara klien Amazon DCV dan server Amazon DCV yang berjalan di node kepala. Untuk mengonfigurasi sertifikat Anda sendiri, lihat [Sertifikat Amazon DCV HTTPS](#).

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

port

(Opsional) Menentukan port untuk Amazon DCV.

Nilai default-nya adalah 8443.

```
port = 8443
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

Bagian [ebs]

Mendefinisikan pengaturan konfigurasi volume Amazon EBS untuk volume yang dipasang pada node kepala dan dibagikan ke node komputasi melalui NFS.

Untuk mempelajari cara menyertakan volume Amazon EBS dalam definisi kluster Anda, lihat [Bagian \[cluster\]/ebs_settings](#).

Untuk menggunakan volume Amazon EBS yang ada untuk penyimpanan permanen jangka panjang yang tidak tergantung pada siklus hidup kluster, tentukan [ebs_volume_id](#).

Jika Anda tidak menentukan [ebs_volume_id](#), AWS ParallelCluster buat volume EBS dari [ebs] pengaturan saat membuat cluster dan menghapus volume dan data saat cluster dihapus.

Untuk informasi selengkapnya, lihat [Praktik terbaik: memindahkan cluster ke versi AWS ParallelCluster minor atau patch baru](#).

Formatnya adalah `[ebs ebs-name]`. *ebs-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[ebs custom1]
shared_dir = vol1
ebs_snapshot_id = snap-xxxxx
volume_type = io1
volume_iops = 200
...

[ebs custom2]
shared_dir = vol2
...

...
```

Topik

- [shared_dir](#)
- [ebs_kms_key_id](#)
- [ebs_snapshot_id](#)
- [ebs_volume_id](#)
- [encrypted](#)
- [volume_iops](#)
- [volume_size](#)
- [volume_throughput](#)
- [volume_type](#)

shared_dir

(Wajib) Menentukan jalur tempat volume Amazon EBS bersama dipasang.

Parameter ini diperlukan saat menggunakan beberapa volume Amazon EBS.

[Saat Anda menggunakan satu volume Amazon EBS, opsi ini akan menempa volume `shared\_dir` yang ditentukan di bawah bagian `\[cluster\]`.](#) Dalam contoh berikut, volume dipasang ke/vol1.

```
shared_dir = vol1
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

ebs_kms_key_id

(Opsional) Menentukan AWS KMS kunci kustom untuk digunakan untuk enkripsi.

Parameter ini harus digunakan bersama dengan `encrypted = true`. Itu juga harus memiliki kebiasaan [ec2_iam_role](#).

Untuk informasi selengkapnya, lihat [Enkripsi disk dengan Kunci KMS khusus](#).

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

ebs_snapshot_id

(Opsional) Mendefinisikan ID snapshot Amazon EBS jika Anda menggunakan snapshot sebagai sumber volume.

Tidak ada nilai default.

```
ebs_snapshot_id = snap-xxxxx
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

ebs_volume_id

(Opsional) Mendefinisikan ID volume volume Amazon EBS yang ada untuk dilampirkan ke node kepala.

Tidak ada nilai default.

```
ebs_volume_id = vol-xxxxxx
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

encrypted

(Opsional) Menentukan apakah volume Amazon EBS dienkripsi. Catatan: Jangan gunakan dengan snapshot.

Nilai default-nya adalah `false`.

```
encrypted = false
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

volume_iops

(Opsional) Mendefinisikan jumlah IOPS untuk `io1`, `io2`, dan `gp3` jenis volume.

Nilai default, nilai yang didukung, dan `volume_iops` `volume_size` rasio bervariasi menurut [volume_type](#) dan [volume_size](#).

`volume_type = io1`

Default `volume_iops` = 100

Nilai yang didukung `volume_iops` = 100—64000 †

Maksimum `volume_iops` terhadap `volume_size` rasio = 50 IOPS untuk setiap GiB. 5000 IOPS membutuhkan `volume_size` setidaknya 100 GiB.

`volume_type = io2`

Default `volume_iops` = 100

Nilai yang didukung `volume_iops` = 100—64000 (256000 untuk `io2` volume Block Express) †

`volume_size` Rasio maksimum `volume_iops` = 500 IOPS untuk setiap GiB. 5000 IOPS membutuhkan `volume_size` minimal 10 GiB.

`volume_type = gp3`

Default `volume_iops` = 3000

Nilai yang didukung `volume_iops` = 3000—16000

`volume_size` Rasio maksimum `volume_iops` = 500 IOPS untuk setiap GiB. 5000 IOPS membutuhkan `volume_size` minimal 10 GiB.

```
volume_iops = 200
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

† IOPS maksimum dijamin hanya pada [Instans yang dibangun di atas Sistem Nitro yang](#) disediakan dengan lebih dari 32.000 IOPS. Instans lain menjamin hingga 32.000 IOPS. Kecuali Anda [memodifikasi volume](#), io1 volume sebelumnya mungkin tidak mencapai kinerja penuh. io2 Volume Block Express mendukung volume_iops nilai hingga 256.000. Untuk informasi selengkapnya, lihat [io2Blokir volume Express \(Dalam pratinjau\)](#) di Panduan EC2 Pengguna Amazon.

volume_size

(Opsional) Menentukan ukuran volume yang akan dibuat, di GiB (jika Anda tidak menggunakan snapshot).

Nilai default dan nilai yang didukung bervariasi menurut nilai [volume_type](#).

volume_type = standard

Default volume_size = 20 GiB

Nilai yang didukung volume_size = 1-1024 GiB

volume_type=gp2,io1,io2, dan gp3

Default volume_size = 20 GiB

Nilai yang didukung volume_size = 1-16384 GiB

volume_type= sc1 dan st1

Default volume_size = 500 GiB

Nilai yang didukung volume_size = 500—16384 GiB

```
volume_size = 20
```

Note

Sebelum AWS ParallelCluster versi 2.10.1, nilai default untuk semua jenis volume adalah 20 GiB.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

volume_throughput

(Opsional) Mendefinisikan throughput untuk tipe gp3 volume, dalam MiB/s.

Nilai default-nya adalah 125.

Nilai yang didukung volume_throughput = 125—1000 MiB/s

Rasio volume_throughput to volume_iops bisa tidak lebih dari 0,25. Throughput maksimum 1000 MiB/s mengharuskan volume_iops pengaturan setidaknya 4000.

```
volume_throughput = 1000
```

Note

Support untuk volume_throughput ditambahkan di AWS ParallelCluster versi 2.10.1.

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

volume_type

(Opsional) Menentukan [jenis volume Amazon EBS](#) dari volume yang ingin Anda luncurkan.

Opsi yang valid adalah jenis volume berikut:

gp2, gp3

SSD tujuan umum

io1, io2

SSD IOPS Terprovisi

st1

Throughput HDD yang dioptimalkan

sc1

Cold HDD

standard

Magnetik generasi sebelumnya

Untuk informasi selengkapnya, lihat [Jenis volume Amazon EBS](#) di Panduan EC2 Pengguna Amazon.

Nilai default-nya adalah gp2.

```
volume_type = io2
```

Note

Support untuk gp3 dan io2 ditambahkan dalam AWS ParallelCluster versi 2.10.1.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

Bagian [efs]

Mendefinisikan pengaturan konfigurasi untuk Amazon EFS yang dipasang di head dan node komputasi. Untuk informasi selengkapnya, lihat [CreateFileSystem](#) di Referensi API Amazon EFS.

Untuk mempelajari cara menyertakan sistem file Amazon EFS dalam definisi cluster Anda, lihat [Bagian \[cluster\]/efs_settings](#).

Untuk menggunakan sistem file Amazon EFS yang ada untuk penyimpanan permanen jangka panjang yang tidak tergantung pada siklus hidup cluster, tentukan [efs_fs_id](#).

Jika Anda tidak menentukan [efs_fs_id](#), AWS ParallelCluster membuat sistem file Amazon EFS dari [efs] pengaturan saat membuat cluster dan menghapus sistem file dan data saat cluster dihapus.

Untuk informasi selengkapnya, lihat [Praktik terbaik: memindahkan cluster ke versi AWS ParallelCluster minor atau patch baru](#).

Formatnya adalah [efs *efs-name*]. *efs-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[efs customfs]
shared_dir = efs
encrypted = false
performance_mode = generalPurpose
```

Topik

- [efs_fs_id](#)

- [efs_kms_key_id](#)
- [encrypted](#)
- [performance_mode](#)
- [provisioned_throughput](#)
- [shared_dir](#)
- [throughput_mode](#)

efs_fs_id

(Opsional) Mendefinisikan ID sistem file Amazon EFS untuk sistem file yang ada.

Menentukan opsi ini membatalkan semua opsi Amazon EFS lainnya kecuali untuk [shared_dir](#)

Jika Anda mengatur opsi ini, itu hanya mendukung jenis sistem file berikut:

- Sistem file yang tidak memiliki target pemasangan di Availability Zone stack.
- Sistem file yang memiliki target pemasangan yang ada di Availability Zone stack dengan lalu lintas NFS masuk dan keluar diizinkan. `0.0.0.0/0`

Pemeriksaan kewarasan untuk memvalidasi [efs_fs_id](#) memerlukan peran IAM untuk memiliki izin berikut:

- `elasticfilesystem:DescribeMountTargets`
- `elasticfilesystem:DescribeMountTargetSecurityGroups`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaceAttribute`

Untuk menghindari kesalahan, Anda harus menambahkan izin ini ke peran IAM Anda, atau mengatur `sanity_check = false`

Important

Saat Anda menetapkan target pemasangan dengan lalu lintas NFS masuk dan keluar yang diizinkan `0.0.0.0/0`, itu akan mengekspos sistem file ke permintaan pemasangan

NFS dari mana saja di Availability Zone target mount. AWS tidak merekomendasikan membuat target pemasangan di Availability Zone tumpukan. Sebaliknya, biarkan AWS menangani langkah ini. Jika Anda ingin memiliki target pemasangan di Availability Zone tumpukan, pertimbangkan untuk menggunakan grup keamanan khusus dengan memberikan [vpc_security_group_id](#) opsi di bawah [\[vpc\]bagian](#) tersebut. Kemudian, tambahkan grup keamanan itu ke target mount dan matikan `sanity_check` untuk membuat cluster.

Tidak ada nilai default.

```
efs_fs_id = fs-12345
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

efs_kms_key_id

(Opsional) Mengidentifikasi AWS Key Management Service (AWS KMS) kunci yang dikelola pelanggan yang akan digunakan untuk melindungi sistem file terenkripsi. Jika ini diatur, [encrypted](#) pengaturan harus diatur ke `true`. Ini sesuai dengan [KmsKeyId](#) parameter di Referensi API Amazon EFS.

Tidak ada nilai default.

```
efs_kms_key_id = 1234abcd-12ab-34cd-56ef-1234567890ab
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

encrypted

(Opsional) Menunjukkan apakah sistem file dienkripsi. Ini sesuai dengan parameter [Terenkripsi](#) di Referensi Amazon EFS API.

Nilai default-nya adalah `false`.

```
encrypted = true
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

performance_mode

(Opsional) Mendefinisikan mode kinerja sistem file. Ini sesuai dengan [PerformanceMode](#) parameter di Referensi API Amazon EFS.

Opsi yang valid adalah nilai-nilai berikut:

- `generalPurpose`
- `maxIO`

Kedua nilai tersebut peka huruf besar/kecil.

Kami merekomendasikan mode `generalPurpose` kinerja untuk sebagian besar sistem file.

Sistem file yang menggunakan mode `maxIO` kinerja dapat menskalakan ke tingkat throughput agregat dan operasi per detik yang lebih tinggi. Namun, ada trade-off latensi yang sedikit lebih tinggi untuk sebagian besar operasi file.

Setelah sistem file dibuat, parameter ini tidak dapat diubah.

Nilai default-nya adalah `generalPurpose`.

```
performance_mode = generalPurpose
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

provisioned_throughput

(Opsional) Mendefinisikan throughput yang disediakan dari sistem file, diukur dalam MIB/s. Ini sesuai dengan [ProvisionedThroughputInMibps](#) parameter di Referensi API Amazon EFS.

Jika Anda menggunakan parameter ini, Anda harus mengatur [throughput_mode](#) ke `provisioned`.

Kuota pada throughput adalah 1024 MIB/s. Untuk meminta kenaikan kuota, hubungi Dukungan.

Nilai minimum adalah 0.0 MIB/s.

```
provisioned_throughput = 1024
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

shared_dir

(Wajib) Mendefinisikan titik pemasangan Amazon EFS di head dan node komputasi.

Parameter ini diperlukan. Bagian Amazon EFS hanya digunakan jika [shared_dir](#) ditentukan.

Jangan gunakan NONE atau /NONE sebagai direktori bersama.

Contoh berikut memasang Amazon EFS di/efs.

```
shared_dir = efs
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

throughput_mode

(Opsional) Mendefinisikan mode throughput dari sistem file. Ini sesuai dengan [ThroughputMode](#) parameter di Referensi API Amazon EFS.

Opsi yang valid adalah nilai-nilai berikut:

- bursting
- provisioned

Nilai default-nya adalah bursting.

```
throughput_mode = provisioned
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

Bagian [fsx]

Mendefinisikan pengaturan konfigurasi untuk dilampirkan FSx untuk sistem file Lustre. Untuk informasi selengkapnya, lihat [Amazon FSx CreateFileSystem](#) di Referensi Amazon FSx API.

Jika [base_os](#) ada linux2, centos7, atau ubuntu1804ubuntu2004, FSx untuk Lustre didukung.

Saat menggunakan Amazon Linux, kernel harus 4.14.104-78.84.amzn1.x86_64 atau versi yang lebih baru. Untuk petunjuk, lihat [Menginstal klien kilau di Amazon FSx untuk Panduan Pengguna Lustre](#).

Note

FSx untuk Lustre saat ini tidak didukung saat menggunakan `awsbatch` sebagai penjadwal.

Note

Support for FSx for Lustre on centos8 telah dihapus di AWS ParallelCluster versi 2.10.4. Support for FSx for Lustre on ubuntu2004 ditambahkan di AWS ParallelCluster versi 2.11.0. Support for FSx for Lustre on centos8 ditambahkan di AWS ParallelCluster versi 2.10.0. Support for FSx for Lustre on alinux2, ubuntu1604, dan ubuntu1804 ditambahkan di AWS ParallelCluster versi 2.6.0. Support for FSx for Lustre on centos7 ditambahkan di AWS ParallelCluster versi 2.4.0.

Jika menggunakan sistem file yang ada, itu harus dikaitkan dengan grup keamanan yang memungkinkan lalu lintas TCP masuk ke port. 988 Menyetel sumber ke `0.0.0.0/0` aturan grup keamanan menyediakan akses klien dari semua rentang IP dalam grup keamanan VPC Anda untuk protokol dan rentang port untuk aturan itu. Untuk lebih membatasi akses ke sistem file Anda, sebaiknya gunakan sumber yang lebih ketat untuk aturan grup keamanan Anda. Misalnya, Anda dapat menggunakan rentang CIDR, alamat IP, atau grup IDs keamanan yang lebih spesifik. Ini dilakukan secara otomatis saat tidak menggunakan [vpc_security_group_id](#).

Untuk menggunakan sistem FSx file Amazon yang ada untuk penyimpanan permanen jangka panjang yang tidak tergantung pada siklus hidup kluster, tentukan [fsx_fs_id](#).

Jika Anda tidak menentukan [fsx_fs_id](#), AWS ParallelCluster membuat sistem file FSx untuk Lustre dari `[fsx]` pengaturan saat membuat cluster dan menghapus sistem file dan data saat cluster dihapus.

Untuk informasi selengkapnya, lihat [Praktik terbaik: memindahkan cluster ke versi AWS ParallelCluster minor atau patch baru](#).

Formatnya adalah `[fsx fsx-name]`. *fsx-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[fsx fs]
shared_dir = /fsx
fsx_fs_id = fs-073c3803dca3e28a6
```

Untuk membuat dan mengkonfigurasi sistem file baru, gunakan parameter berikut:

```
[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
```

Topik

- [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

(Opsional) Menentukan kebijakan impor otomatis untuk mencerminkan perubahan dalam bucket S3 yang digunakan untuk membuat sistem file FSx Lustre. Nilai yang mungkin adalah sebagai berikut:

NEW

FSx untuk Lustre secara otomatis mengimpor daftar direktori objek baru apa pun yang ditambahkan ke bucket S3 tertaut yang saat ini tidak ada di sistem file FSx for Lustre.

NEW_CHANGED

FSx untuk Lustre secara otomatis mengimpor daftar file dan direktori objek baru apa pun yang ditambahkan ke bucket S3 dan objek apa pun yang ada yang diubah di bucket S3.

Ini sesuai dengan [AutoImportPolicy](#) properti. Untuk informasi selengkapnya, lihat [Mengimpor pembaruan secara otomatis dari bucket S3 Anda](#) di Panduan Pengguna Amazon FSx for Lustre. Ketika [auto_import_policy](#) parameter ditentukan, parameter [automatic_backup_retention_days](#), [copy_tags_to_backups](#), [daily_automatic_backup](#) dan [fsx_backup_id](#) tidak boleh ditentukan.

Jika [auto_import_policy](#) pengaturan tidak ditentukan, impor otomatis akan dinonaktifkan. FSx untuk Lustre hanya memperbarui daftar file dan direktori dari bucket S3 yang ditautkan saat sistem file dibuat.

```
auto_import_policy = NEW_CHANGED
```

Note

Support untuk [auto_import_policy](#) ditambahkan dalam AWS ParallelCluster versi 2.10.0.

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

automatic_backup_retention_days

(Opsional) Menentukan jumlah hari untuk mempertahankan backup otomatis. Ini hanya berlaku untuk digunakan dengan tipe `PERSISTENT_1` penerapan. Ketika [automatic_backup_retention_days](#) parameter ditentukan, parameter [auto_import_policy](#), [export_path](#), [import_path](#), dan [imported_file_chunk_size](#) tidak boleh ditentukan. Ini sesuai dengan [AutomaticBackupRetentionDays](#) properti.

Nilai default-nya adalah 0. Pengaturan ini menonaktifkan pencadangan otomatis. Nilai yang mungkin adalah bilangan bulat antara 0 dan 35, inklusif.

```
automatic_backup_retention_days = 35
```

Note

Support untuk [automatic_backup_retention_days](#) ditambahkan di AWS ParallelCluster versi 2.8.0.

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

copy_tags_to_backups

(Opsional) Menentukan apakah tag untuk filesystem disalin ke backup. Ini hanya berlaku untuk digunakan dengan tipe PERSISTENT_1 penerapan. Ketika [copy_tags_to_backups](#) parameter ditentukan, [automatic_backup_retention_days](#) harus ditentukan dengan nilai lebih besar dari 0, dan [auto_import_policy](#), [export_pathimport_path](#), dan [imported_file_chunk_size](#) parameter tidak boleh ditentukan. Ini sesuai dengan [CopyTagsToBackups](#) properti.

Nilai default-nya adalah `false`.

```
copy_tags_to_backups = true
```

Note

Support untuk [copy_tags_to_backups](#) ditambahkan di AWS ParallelCluster versi 2.8.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

daily_automatic_backup_start_time

(Opsional) Menentukan waktu hari (UTC) untuk memulai pencadangan otomatis.

Ini hanya berlaku untuk digunakan dengan tipe PERSISTENT_1 penerapan.

Ketika [daily_automatic_backup_start_time](#) parameter ditentukan, [automatic_backup_retention_days](#) harus ditentukan dengan nilai lebih besar dari 0, dan [auto_import_policy](#), [export_pathimport_path](#), dan [imported_file_chunk_size](#) parameter tidak boleh ditentukan. Ini sesuai dengan [DailyAutomaticBackupStartTime](#) properti.

Formatnya adalah HH:MM, di mana HH adalah jam empuk nol dalam sehari (0-23), dan MM merupakan menit empuk nol dari jam itu. Sebagai contoh, 1:03 AM UTC adalah sebagai berikut.

```
daily_automatic_backup_start_time = 01:03
```

Nilai default adalah waktu acak antara 00:00 dan 23:59.

Note

Support untuk [daily_automatic_backup_start_time](#) ditambahkan di AWS ParallelCluster versi 2.8.0.

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

data_compression_type

(Opsional) Menentukan FSx untuk Lustre jenis kompresi data. Ini sesuai dengan [DataCompressionType](#) properti. Untuk informasi selengkapnya, lihat [FSx kompresi data Lustre](#) di Amazon FSx for Lustre User Guide.

Satu-satunya nilai yang valid adalah LZ4. Untuk menonaktifkan kompresi data, hapus [data_compression_type](#) parameter.

```
data_compression_type = LZ4
```

Note

Support untuk [data_compression_type](#) ditambahkan dalam AWS ParallelCluster versi 2.11.0.

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

deployment_type

(Opsional) Menentukan FSx untuk jenis penyebaran Lustre. Ini sesuai dengan [DeploymentType](#) properti. Untuk informasi selengkapnya, lihat [FSx opsi penerapan Lustre](#) di

Amazon FSx for Lustre User Guide. Pilih jenis penyebaran awal untuk penyimpanan sementara dan pemrosesan data jangka pendek. SCRATCH_2 adalah generasi terbaru dari sistem file scratch. Ini menawarkan throughput burst yang lebih tinggi dibandingkan throughput dasar dan enkripsi data dalam transit.

Nilai yang valid adalah SCRATCH_1, SCRATCH_2, dan PERSISTENT_1.

SCRATCH_1

Jenis penerapan default FSx untuk Lustre. Dengan jenis penerapan ini, [storage_capacity](#) pengaturan memiliki kemungkinan nilai 1200, 2400, dan kelipatan 3600. Support untuk SCRATCH_1 ditambahkan di AWS ParallelCluster versi 2.4.0.

SCRATCH_2

Generasi terbaru dari sistem file scratch. Ini mendukung hingga enam kali throughput dasar untuk beban kerja runcing. Ini juga mendukung enkripsi data dalam transit untuk jenis instans yang didukung dalam dukungan Wilayah AWS. Untuk informasi selengkapnya, lihat [Mengenakripsi data saat transit di](#) Amazon FSx for Lustre User Guide. Dengan jenis penerapan ini, [storage_capacity](#) pengaturan memiliki kemungkinan nilai 1200 dan kelipatan 2400. Support untuk SCRATCH_2 ditambahkan di AWS ParallelCluster versi 2.6.0.

PERSISTENT_1

Dirancang untuk penyimpanan jangka panjang. Server file sangat tersedia dan data direplikasi dalam AWS Availability Zone sistem file. Ini mendukung enkripsi data dalam transit untuk jenis instans yang didukung. Dengan jenis penerapan ini, [storage_capacity](#) pengaturan memiliki kemungkinan nilai 1200 dan kelipatan 2400. Support untuk PERSISTENT_1 ditambahkan di AWS ParallelCluster versi 2.6.0.

Nilai default-nya adalah SCRATCH_1.

```
deployment_type = SCRATCH_2
```

Note

Support untuk [deployment_type](#) ditambahkan di AWS ParallelCluster versi 2.6.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

drive_cache_type

(Opsional) Menentukan bahwa sistem file memiliki cache drive SSD. Ini hanya dapat diatur jika [storage_type](#) pengaturan diatur keHDD. Ini sesuai dengan [DriveCacheType](#) properti. Untuk informasi selengkapnya, lihat [FSx opsi penerapan Lustre](#) di Amazon FSx for Lustre User Guide.

Satu-satunya nilai yang valid adalah READ. Untuk menonaktifkan cache drive SSD, jangan tentukan `drive_cache_type` pengaturannya.

```
drive_cache_type = READ
```

Note

Support untuk [drive_cache_type](#) ditambahkan dalam AWS ParallelCluster versi 2.10.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

export_path

(Opsional) Menentukan jalur Amazon S3 tempat root sistem file Anda diekspor. Ketika [export_path](#) parameter ditentukan, parameter [automatic_backup_retention_days](#), [copy_tags_to_backups](#), [daily_automatic_backup](#) dan [fsx_backup_id](#) tidak boleh ditentukan. Ini sesuai dengan [ExportPath](#) properti. Data file dan metadata tidak secara otomatis diekspor ke file. `export_path` Untuk informasi tentang mengekspor data dan metadata, lihat [Mengekspor perubahan pada repositori data di Panduan Pengguna](#) Amazon FSx for Lustre.

Nilai *import-bucket* defaultnya adalah `s3://import-bucket/FSxLustre[creation-timestamp]`, di mana bucket disediakan dalam [import_path](#) parameter.

```
export_path = s3://bucket/folder
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

fsx_backup_id

(Opsional) Menentukan ID cadangan yang akan digunakan untuk memulihkan sistem file dari cadangan yang ada. Ketika [fsx_backup_id](#) parameter

ditentukan,,[auto_import_policy](#),[deployment_type](#),[export_path](#),[fsx_kms_key_id](#),[import_path](#),[imported_file_chunk_size](#)[storage_capacity](#), dan [per_unit_storage_throughput](#)

parameter tidak boleh ditentukan. Parameter ini dibaca dari cadangan.

Selain itu[auto_import_policy](#), [imported_file_chunk_size](#) parameter[export_path](#),[import_path](#),, dan tidak boleh ditentukan.

Ini sesuai dengan [BackupId](#) properti.

```
fsx_backup_id = backup-fedcba98
```

Note

Support untuk [fsx_backup_id](#) ditambahkan di AWS ParallelCluster versi 2.8.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

fsx_fs_id

(Opsional) Melampirkan yang ada FSx untuk sistem file Lustre.

Jika opsi ini ditentukan, hanya [fsx_fs_id](#) pengaturan [shared_dir](#) dan di [\[fsx\]bagian](#) yang digunakan dan pengaturan lain di [\[fsx\]bagian](#) ini diabaikan.

```
fsx_fs_id = fs-073c3803dca3e28a6
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

fsx_kms_key_id

(Opsional) Menentukan ID kunci kunci AWS Key Management Service (AWS KMS) kunci terkelola pelanggan Anda.

Kunci ini digunakan untuk mengenkripsi data dalam sistem file Anda saat istirahat.

Ini harus digunakan dengan kebiasaan[ec2_iam_role](#). Untuk informasi selengkapnya, lihat [Enkripsi disk dengan Kunci KMS khusus](#). Ini sesuai dengan [KmsKeyId](#) parameter di Referensi Amazon FSx API.

```
fsx_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxxxx
```

Note

Support untuk [fsx_kms_key_id](#) ditambahkan di AWS ParallelCluster versi 2.6.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

import_path

(Opsional) Menentukan bucket S3 untuk memuat data dari ke dalam sistem file dan berfungsi sebagai bucket ekspor. Untuk informasi selengkapnya, lihat [export_path](#). Jika Anda menentukan [import_path](#) parameter, [automatic_backup_retention_days](#), [copy_tags_to_backupsdaily_automatic_backup_start_time](#), dan [fsx_backup_id](#) parameter tidak boleh ditentukan. Ini sesuai dengan [ImportPath](#) parameter di Referensi Amazon FSx API.

Impor terjadi pada pembuatan cluster. Untuk informasi selengkapnya, lihat [Mengimpor data dari repositori data Anda](#) di Amazon FSx for Lustre User Guide. Saat impor, hanya metadata file (nama, kepemilikan, stempel waktu, dan izin) yang diimpor. Data file tidak diimpor dari bucket S3 sampai file pertama kali diakses. Untuk informasi tentang pra-pemuatan konten file, lihat [Pra-pemuatan file ke dalam sistem file Anda di Panduan](#) Pengguna Amazon FSx for Lustre.

Jika nilai tidak diberikan, sistem file kosong.

```
import_path = s3://bucket
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

imported_file_chunk_size

(Opsional) Menentukan jumlah garis dan jumlah maksimum data untuk setiap file (dalam MiB) yang disimpan pada disk fisik tunggal untuk file yang diimpor dari repositori data (menggunakan). [import_path](#) Jumlah maksimum disk yang suatu file dapat diberi stripe dibatasi oleh jumlah total disk yang membentuk sistem file. Ketika [imported_file_chunk_size](#) parameter ditentukan, parameter [automatic_backup_retention_days](#), [copy_tags_to_backupsdaily_automatic_backup_start_time](#), dan [fsx_backup_id](#) tidak boleh ditentukan. Ini sesuai dengan [ImportedFileChunkSize](#) properti.

Ukuran potongan default adalah 1024 (1 GiB), dan bisa mencapai 512.000 MiB (500 GiB). Objek Amazon S3 memiliki ukuran maksimal 5 TB.

```
imported_file_chunk_size = 1024
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

per_unit_storage_throughput

(Diperlukan untuk jenis **PERSISTENT_1** penerapan) Untuk jenis [deployment_type](#) = PERSISTENT_1 penerapan, jelaskan jumlah throughput baca dan tulis untuk setiap 1 tebibyte (TiB) penyimpanan, dalam MB/s/Tib. Kapasitas throughput sistem file dihitung dengan mengalikan kapasitas penyimpanan sistem file (TiB) dengan [per_unit_storage_throughput](#) (MB/s/TiB). For a 2.4 TiB file system, provisioning 50 MB/s/TiB dari [per_unit_storage_throughput](#) hasil 120 MB/s throughput sistem file. Anda membayar untuk jumlah throughput yang Anda sediakan. Ini sesuai dengan [PerUnitStorageThroughput](#) properti.

Nilai yang mungkin tergantung pada nilai [storage_type](#) pengaturan.

[storage_type](#) = SSD

Nilai yang mungkin adalah 50, 100, 200.

[storage_type](#) = HDD

Nilai yang mungkin adalah 12, 40.

```
per_unit_storage_throughput = 200
```

Note

Support untuk [per_unit_storage_throughput](#) ditambahkan di AWS ParallelCluster versi 2.6.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

shared_dir

(Wajib) Mendefinisikan titik pemasangan untuk sistem file FSx for Lustre di kepala dan node komputasi.

Jangan gunakan NONE atau /NONE sebagai direktori bersama.

Contoh berikut memasang sistem file di `/fsx`.

```
shared_dir = /fsx
```

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

storage_capacity

(Wajib) Menentukan kapasitas penyimpanan sistem file, di GiB. Ini sesuai dengan [StorageCapacity](#) properti.

Nilai kapasitas penyimpanan yang mungkin bervariasi berdasarkan [deployment_type](#) pengaturan.

SCRATCH_1

Nilai yang mungkin adalah 1200, 2400, dan kelipatan 3600.

SCRATCH_2

Nilai yang mungkin adalah 1200 dan kelipatan 2400.

PERSISTENT_1

Nilai yang mungkin bervariasi berdasarkan nilai pengaturan lainnya.

[storage_type](#) = SSD

Nilai yang mungkin adalah 1200 dan kelipatan 2400.

[storage_type](#) = HDD

Nilai yang mungkin bervariasi berdasarkan pengaturan [per_unit_storage_throughput](#) pengaturan.

[per_unit_storage_throughput](#) = 12

Nilai yang mungkin adalah kelipatan 6000.

[per_unit_storage_throughput](#) = 40

Nilai yang mungkin adalah kelipatan 1800.

```
storage_capacity = 7200
```

Note

Untuk AWS ParallelCluster versi 2.5.0 dan 2.5.1, [storage_capacity](#) didukung kemungkinan nilai 1200, 2400, dan kelipatan 3600. Untuk versi lebih awal dari AWS ParallelCluster versi 2.5.0, [storage_capacity](#) memiliki ukuran minimum 3600.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

storage_type

(Opsional) Menentukan jenis penyimpanan sistem file. Ini sesuai dengan [StorageType](#) properti. Nilai yang mungkin adalah SSD dan HDD. Nilai default-nya SSD.

Jenis penyimpanan mengubah nilai yang mungkin dari pengaturan lain.

```
storage_type = SSD
```

Menentukan jenis penyimpanan sold-state drive (SSD).

`storage_type = SSD` mengubah nilai yang mungkin dari beberapa pengaturan lainnya.

[drive_cache_type](#)

Pengaturan ini tidak dapat ditentukan.

[deployment_type](#)

Pengaturan ini dapat diatur ke `SCRATCH_1`, `SCRATCH_2`, atau `PERSISTENT_1`.

[per_unit_storage_throughput](#)

Pengaturan ini harus ditentukan jika [deployment_type](#) diatur pada `PERSISTENT_1`. Nilai yang mungkin adalah 50, 100, atau 200.

[storage_capacity](#)

Pengaturan ini harus ditentukan. Nilai yang mungkin bervariasi berdasarkan [deployment_type](#).

```
deployment_type = SCRATCH_1
```

[storage_capacity](#) bisa 1200, 2400, atau kelipatan 3600.

```
deployment_type = SCRATCH_2 atau deployment_type = PERSISTENT_1
```

[storage_capacity](#) bisa 1200 atau kelipatan 2400.

```
storage_type = HDD
```

Menentukan jenis penyimpanan hard disk drive (HDD).

`storage_type = HDD` mengubah nilai yang mungkin dari pengaturan lain.

[drive_cache_type](#)

Pengaturan ini dapat ditentukan.

[deployment_type](#)

Pengaturan ini harus diatur ke `PERSISTENT_1`.

[per_unit_storage_throughput](#)

Pengaturan ini harus ditentukan. Nilai yang mungkin adalah 12, atau 40.

[storage_capacity](#)

Pengaturan ini harus ditentukan. Nilai yang mungkin bervariasi berdasarkan

[per_unit_storage_throughput](#) pengaturan.

```
storage_capacity = 12
```

[storage_capacity](#) bisa kelipatan 6000.

```
storage_capacity = 40
```

[storage_capacity](#) bisa kelipatan 1800.

```
storage_type = SSD
```

Note

Support untuk [storage_type](#) pengaturan ditambahkan di AWS ParallelCluster versi 2.10.0.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

weekly_maintenance_start_time

(Opsional) Menentukan waktu yang diinginkan untuk melakukan pemeliharaan mingguan, di zona waktu UTC. Ini sesuai dengan [WeeklyMaintenanceStartTime](#) properti.

Formatnya adalah [hari dalam seminggu]: [jam hari]: [menit jam]. Misalnya, Senin di Tengah Malam adalah sebagai berikut.

```
weekly_maintenance_start_time = 1:00:00
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

Bagian [queue]

Mendefinisikan pengaturan konfigurasi untuk antrian tunggal. [\[queue\]bagian](#) hanya didukung ketika [scheduler](#) diatur ke `splurm`.

Formatnya adalah `[queue <queue-name>]`. *queue-name* harus dimulai dengan huruf kecil, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf kecil, angka, dan tanda hubung (-).

```
[queue q1]
compute_resource_settings = i1,i2
placement_group = DYNAMIC
enable_efa = true
disable_hyperthreading = false
compute_type = spot
```

Note

Support untuk [\[queue\]bagian](#) ditambahkan dalam AWS ParallelCluster versi 2.9.0.

Topik

- [compute_resource_settings](#)
- [compute_type](#)

- [disable_hyperthreading](#)
- [enable_efa](#)
- [enable_efa_gdr](#)
- [placement_group](#)

compute_resource_settings

(Wajib) Mengidentifikasi [\[compute_resource\]bagian](#) yang berisi konfigurasi sumber daya komputasi untuk antrian ini. Nama bagian harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

Hingga tiga (3) [\[compute_resource\]bagian](#) didukung untuk setiap [\[queue\]bagian](#).

Misalnya, pengaturan berikut menentukan bahwa bagian yang dimulai `[compute_resource cr1]` dan `[compute_resource cr2]` digunakan.

```
compute_resource_settings = cr1, cr2
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

compute_type

(Opsional) Mendefinisikan jenis instance yang akan diluncurkan untuk antrian ini. Pengaturan ini menggantikan pengaturan [cluster_type](#).

Opsi yang valid adalah:ondemand, danspot.

Nilai default-nya adalah ondemand.

Untuk informasi selengkapnya tentang Instans Spot, lihat[Berkeja dengan Instans Spot](#).

Note

Menggunakan Instans Spot mengharuskan peran `AWSServiceRoleForEC2Spot` terkait layanan ada di akun Anda. Untuk membuat peran ini di akun Anda menggunakan AWS CLI, jalankan perintah berikut:

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

Untuk informasi selengkapnya, lihat [Peran terkait layanan untuk permintaan Instans Spot](#) di EC2 Panduan Pengguna Amazon.

Contoh berikut menggunakan SpotInstances untuk node komputasi dalam antrian ini.

```
compute_type = spot
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

disable_hyperthreading

(Opsional) Menonaktifkan hyperthreading pada node dalam antrian ini. Tidak semua tipe instance dapat menonaktifkan hyperthreading. Untuk daftar jenis instans yang mendukung penonaktifan hyperthreading, lihat [inti CPU dan thread untuk setiap inti CPU per jenis instans di](#) Panduan Pengguna Amazon. EC2 Jika [disable_hyperthreading](#) pengaturan di [\[cluster\]bagian](#) ditentukan, maka pengaturan ini tidak dapat ditentukan.

Nilai default-nya adalah `false`.

```
disable_hyperthreading = true
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

enable_efa

(Opsional) Jika diatur ke `true`, menentukan bahwa Elastic Fabric Adapter (EFA) diaktifkan untuk node dalam antrian ini. Untuk melihat daftar EC2 instance yang mendukung EFA, lihat [Jenis instans yang didukung](#) di Panduan EC2 Pengguna Amazon untuk Instans Linux. Jika [enable_efa](#) pengaturan di [\[cluster\]bagian](#) ditentukan, maka pengaturan ini tidak dapat ditentukan. Grup penempatan cluster harus digunakan untuk meminimalkan latensi antar instance. Untuk informasi selengkapnya, lihat [placement](#) dan [placement_group](#).

```
enable_efa = true
```

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

enable_efa_gdr

(Opsional) Dimulai dengan AWS ParallelCluster versi 2.11.3, pengaturan ini tidak berpengaruh. Dukungan Elastic Fabric Adapter (EFA) untuk GPUDirect RDMA (akses memori langsung jarak jauh) diaktifkan untuk node komputasi selalu diaktifkan jika didukung oleh jenis instance.

Note

AWS ParallelCluster versi 2.10.0 hingga 2.11.2: Jika `true`, menentukan bahwa Elastic Fabric Adapter GPUDirect (EFA) RDMA (akses memori langsung jarak jauh) diaktifkan untuk node dalam antrian ini. Menyetel ini ke `true` mengharuskan [enable_efa](#) pengaturan disetel `true` ke EFA GPUDirect RDMA didukung oleh jenis instance berikut (p4d.24xlarge) pada sistem operasi ini (alinux2, centos7, ubuntu1804 atau ubuntu2004). Jika [enable_efa_gdr](#) pengaturan di [\[cluster\]bagian](#) ditentukan, maka pengaturan ini tidak dapat ditentukan. Grup penempatan cluster harus digunakan untuk meminimalkan latensi antar instance. Untuk informasi selengkapnya, lihat [placement](#) dan [placement_group](#).

Nilai default-nya adalah `false`.

```
enable_efa_gdr = true
```

Note

Support untuk `enable_efa_gdr` ditambahkan dalam AWS ParallelCluster versi 2.10.0.

Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.

placement_group

(Opsional) Jika ada, mendefinisikan grup penempatan untuk antrian ini. Pengaturan ini menggantikan pengaturan [placement_group](#).

Opsi yang valid adalah nilai-nilai berikut:

- DYNAMIC
- Nama grup penempatan EC2 klaster Amazon yang ada

Ketika diatur keDYNAMIC, grup penempatan unik untuk antrian ini dibuat dan dihapus sebagai bagian dari tumpukan cluster.

Untuk informasi selengkapnya tentang grup [penempatan](#), lihat [Grup penempatan](#) di Panduan EC2 Pengguna Amazon. Jika grup penempatan yang sama digunakan untuk jenis instans yang berbeda, kemungkinan besar permintaan tersebut mungkin gagal karena kesalahan kapasitas yang tidak mencukupi. Untuk informasi selengkapnya, lihat [Kapasitas instans tidak mencukupi](#) di Panduan EC2 Pengguna Amazon.

Tidak ada nilai default.

Tidak semua tipe instance mendukung grup penempatan klaster. Misalnya, `t2.micro` tidak mendukung grup penempatan klaster. Untuk informasi tentang daftar jenis instans yang mendukung grup penempatan klaster, lihat [Aturan dan batasan grup penempatan klaster](#) di Panduan EC2 Pengguna Amazon. Lihat [Grup penempatan dan masalah peluncuran instance](#) tip saat bekerja dengan grup penempatan.

```
placement_group = DYNAMIC
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

Bagian **[raid]**

Mendefinisikan pengaturan konfigurasi untuk array RAID yang dibangun dari sejumlah volume Amazon EBS yang identik. Drive RAID dipasang pada node kepala dan diekspor untuk menghitung node dengan NFS.

Formatnya adalah `[raid raid-name]`. *raid-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[raid rs]
shared_dir = raid
raid_type = 1
num_of_raid_volumes = 2
```

```
encrypted = true
```

Topik

- [shared_dir](#)
- [ebs_kms_key_id](#)
- [encrypted](#)
- [num_of_raid_volumes](#)
- [raid_type](#)
- [volume_iops](#)
- [volume_size](#)
- [volume_throughput](#)
- [volume_type](#)

shared_dir

(Wajib) Mendefinisikan titik pemasangan untuk array RAID di kepala dan node komputasi.

Drive RAID dibuat hanya jika parameter ini ditentukan.

Jangan gunakan NONE atau /NONE sebagai direktori bersama.

Contoh berikut memasang array di/raid.

```
shared_dir = raid
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

ebs_kms_key_id

(Opsional) Menentukan AWS KMS kunci kustom untuk digunakan untuk enkripsi.

Parameter ini harus digunakan bersama dengan `encrypted = true`, dan harus memiliki kustom [ec2_iam_role](#).

Untuk informasi selengkapnya, lihat [Enkripsi disk dengan Kunci KMS khusus](#).

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

encrypted

(Opsional) Menentukan apakah sistem file dienkrupsi.

Nilai default-nya adalah `false`.

```
encrypted = false
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

num_of_raid_volumes

(Opsional) Mendefinisikan jumlah volume Amazon EBS untuk merakit array RAID dari.

Jumlah volume minimum adalah 2.

Jumlah volume maksimum adalah 5.

Nilai default-nya adalah 2.

```
num_of_raid_volumes = 2
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

raid_type

(Wajib) Mendefinisikan tipe RAID untuk array RAID.

Drive RAID dibuat hanya jika parameter ini ditentukan.

Opsi yang valid adalah nilai-nilai berikut:

- 0
- 1

Untuk informasi selengkapnya tentang jenis [RAID, lihat info RAID](#) di Panduan EC2 Pengguna Amazon.

Contoh berikut membuat 0 array RAID:

```
raid_type = 0
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

volume_iops

(Opsional) Mendefinisikan jumlah IOPS untuk `io1`, `io2`, dan `gp3` jenis volume.

Nilai default, nilai yang didukung, dan `volume_iops` `volume_size` rasio bervariasi menurut [volume_type](#) dan [volume_size](#).

`volume_type = io1`

Default `volume_iops` = 100

Nilai yang didukung `volume_iops` = 100—64000 †

`volume_size` Rasio maksimum `volume_iops` = 50 IOPS per GiB. 5000 IOPS membutuhkan `volume_size` setidaknya 100 GiB.

`volume_type = io2`

Default `volume_iops` = 100

Nilai yang didukung `volume_iops` = 100—64000 (256000 untuk `io2` volume Block Express) †

`volume_size` Rasio maksimum `volume_iops` = 500 IOPS per GiB. 5000 IOPS membutuhkan `volume_size` minimal 10 GiB.

`volume_type = gp3`

Default `volume_iops` = 3000

Nilai yang didukung `volume_iops` = 3000—16000

`volume_size` Rasio maksimum `volume_iops` = 500 IOPS per GiB. 5000 IOPS membutuhkan `volume_size` minimal 10 GiB.

```
volume_iops = 3000
```

Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.

† IOPS maksimum dijamin hanya pada [Instans yang dibangun di atas Sistem Nitro yang](#) disediakan dengan lebih dari 32.000 IOPS. Instans lain menjamin hingga 32.000 IOPS. `io1` Volume yang lebih lama mungkin tidak mencapai performa penuh kecuali Anda [memodifikasi volume](#). `io2` Volume Block Express mendukung `volume_iops` nilai hingga 256000. Untuk informasi selengkapnya, lihat [io2Blokir volume Express \(Dalam pratinjau\)](#) di Panduan EC2 Pengguna Amazon.

volume_size

(Opsional) Mendefinisikan ukuran volume yang akan dibuat, di GiB.

Nilai default dan nilai yang didukung bervariasi menurut [volume_type](#).

`volume_type = standard`

Default `volume_size` = 20 GiB

Nilai yang didukung `volume_size` = 1-1024 GiB

`volume_type=gp2,io1,io2, dan gp3`

Default `volume_size` = 20 GiB

Nilai yang didukung `volume_size` = 1-16384 GiB

`volume_type= sc1 dan st1`

Default `volume_size` = 500 GiB

Nilai yang didukung `volume_size` = 500—16384 GiB

```
volume_size = 20
```

Note

Sebelum AWS ParallelCluster versi 2.10.1, nilai default untuk semua jenis volume adalah 20 GiB.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

volume_throughput

(Opsional) Mendefinisikan throughput untuk tipe gp3 volume, dalam MiB/s.

Nilai default-nya adalah 125.

Nilai yang didukung volume_throughput = 125—1000 MiB/s

Rasio volume_throughput to volume_iops bisa tidak lebih dari 0,25. Throughput maksimum 1000 MiB/s mengharuskan volume_iops pengaturan setidaknya 4000.

```
volume_throughput = 1000
```

Note

Support untuk volume_throughput ditambahkan di AWS ParallelCluster versi 2.10.1.

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

volume_type

(Opsional) Mendefinisikan jenis volume yang akan dibangun.

Opsi yang valid adalah nilai-nilai berikut:

gp2, gp3

SSD tujuan umum

io1, io2

SSD IOPS Terprovisi

st1

Throughput HDD yang dioptimalkan

sc1

Cold HDD

standard

Magnetik generasi sebelumnya

Untuk informasi selengkapnya, lihat [Jenis volume Amazon EBS](#) di Panduan EC2 Pengguna Amazon.

Nilai default-nya adalah gp2.

```
volume_type = io2
```

Note

Support untuk gp3 dan io2 ditambahkan dalam AWS ParallelCluster versi 2.10.1.

Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.

Bagian [scaling]

Topik

- [scaledown_idletime](#)

Menentukan pengaturan yang menentukan bagaimana skala node komputasi.

Formatnya adalah [scaling *scaling-name*]. *scaling-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

```
[scaling custom]
scaledown_idletime = 10
```

scaledown_idletime

(Opsional) Menentukan jumlah waktu dalam menit tanpa pekerjaan, setelah itu node komputasi berakhir.

Parameter ini tidak digunakan jika awsbatch adalah penjadwal.

Nilai default-nya adalah 10.

```
scaledown_idletime = 10
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

Bagian [vpc]

Menentukan pengaturan konfigurasi Amazon VPC. Untuk informasi selengkapnya VPCs, lihat [Apa itu Amazon VPC?](#) dan [Praktik terbaik keamanan untuk VPC Anda](#) di Panduan Pengguna Amazon VPC.

Formatnya adalah [vpc *vpc-name*]. *vpc-name* harus dimulai dengan huruf, berisi tidak lebih dari 30 karakter, dan hanya berisi huruf, angka, tanda hubung (-), dan garis bawah (_).

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

Topik

- [additional_sg](#)
- [compute_subnet_cidr](#)
- [compute_subnet_id](#)
- [master_subnet_id](#)
- [ssh_from](#)
- [use_public_ips](#)
- [vpc_id](#)
- [vpc_security_group_id](#)

additional_sg

(Opsional) Menyediakan Id grup keamanan Amazon VPC tambahan untuk semua instans.

Tidak ada nilai default.

```
additional_sg = sg-xxxxxx
```

compute_subnet_cidr

(Opsional) Menentukan blok Classless Inter-Domain Routing (CIDR). Gunakan parameter ini jika Anda AWS ParallelCluster ingin membuat subnet komputasi.

```
compute_subnet_cidr = 10.0.100.0/24
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

compute_subnet_id

(Opsional) Menentukan ID dari subnet yang ada di mana untuk menyediakan node komputasi.

Jika tidak ditentukan, [compute_subnet_id](#) gunakan nilai [master_subnet_id](#).

Jika subnet bersifat pribadi, Anda harus mengatur NAT untuk akses web.

```
compute_subnet_id = subnet-xxxxxx
```

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

master_subnet_id

(Wajib) Menentukan ID dari subnet yang ada untuk menyediakan node kepala.

```
master_subnet_id = subnet-xxxxxx
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

ssh_from

(Opsional) Menentukan rentang IP berformat CIDR untuk memungkinkan akses SSH dari.

Parameter ini hanya digunakan ketika AWS ParallelCluster membuat grup keamanan.

Nilai default-nya adalah 0.0.0.0/0.

```
ssh_from = 0.0.0.0/0
```

[Kebijakan pembaruan: Pengaturan ini dapat diubah selama pembaruan.](#)

use_public_ips

(Opsional) Mendefinisikan apakah akan menetapkan alamat IP publik untuk menghitung instance.

Jika disetel ke `true`, alamat IP Elastis dikaitkan dengan node kepala.

Jika diatur ke `false`, node kepala memiliki IP publik (atau tidak) sesuai dengan nilai parameter konfigurasi subnet “Auto-assign Public IP”.

Sebagai contoh, lihat [konfigurasi jaringan](#).

Nilai default-nya adalah `true`.

```
use_public_ips = true
```

Important

Secara default, semuanya Akun AWS dibatasi hingga lima (5) alamat IP Elastis untuk masing-masing alamat Wilayah AWS. Untuk informasi selengkapnya, lihat [Batas alamat IP elastis](#) di Panduan EC2 Pengguna Amazon.

[Kebijakan pembaruan: Armada komputasi harus dihentikan agar pengaturan ini diubah untuk pembaruan.](#)

vpc_id

(Wajib) Menentukan ID VPC Amazon untuk menyediakan klaster.

```
vpc_id = vpc-xxxxxx
```

[Kebijakan pembaruan: Jika pengaturan ini diubah, pembaruan tidak diizinkan.](#)

vpc_security_group_id

(Opsional) Menentukan penggunaan grup keamanan yang ada untuk semua instance.

Tidak ada nilai default.

```
vpc_security_group_id = sg-xxxxxx
```

Grup keamanan yang dibuat oleh AWS ParallelCluster memungkinkan akses SSH menggunakan port 22 dari alamat yang ditentukan dalam [ssh_from](#) pengaturan, atau semua IPv4 alamat (0.0.0.0/0) jika [ssh_from](#) pengaturan tidak ditentukan. Jika Amazon DCV diaktifkan, grup keamanan mengizinkan akses ke Amazon DCV menggunakan port 8443 (atau apa pun yang ditentukan oleh [port](#) pengaturan) dari alamat yang ditentukan dalam [access_from](#) setelan, atau semua IPv4 alamat (0.0.0.0/0) jika setelan tidak ditentukan. [access_from](#)

Warning

Anda dapat mengubah nilai parameter ini dan memperbarui cluster jika [\[cluster\]fsx_settings](#) tidak ditentukan atau keduanya `fsx_settings` dan eksternal yang ada FSx untuk sistem file Lustre ditentukan untuk [fsx-fs-iddi](#). [\[fsx fs\]](#) Anda tidak dapat mengubah nilai parameter ini jika sistem file Lustre yang AWS ParallelCluster dikelola FSx ditentukan dalam `fsx_settings` dan. [\[fsx fs\]](#)

[Kebijakan pembaruan: Jika sistem file Amazon FSx untuk Lustre yang AWS ParallelCluster dikelola tidak ditentukan dalam konfigurasi, setelan ini dapat diubah selama pembaruan.](#)

Contoh

Contoh konfigurasi berikut menunjukkan AWS ParallelCluster konfigurasi menggunakan Slurm, Torque, dan AWS Batch penjadwal.

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Daftar Isi

- [Slurm Workload Manager \(slurm\)](#)

- [Son of Grid Engine \(sge\) dan Torque Resource Manager \(torque\)](#)
- [AWS Batch \(awsbatch\)](#)

Slurm Workload Manager (**slurm**)

Contoh berikut meluncurkan cluster dengan `slurm` scheduler. Contoh konfigurasi meluncurkan 1 cluster dengan 2 antrian pekerjaan. Antrian pertama, `spot`, awalnya memiliki 2 instans `t3.micro` Spot yang tersedia. Ini dapat menskalakan hingga maksimum 10 instance, dan menurunkan skala ke minimum 1 instans ketika tidak ada pekerjaan yang dijalankan selama 10 menit (dapat disesuaikan menggunakan [scaledown_idletime](#) pengaturan). Antrian kedua `ondemand`, dimulai tanpa instans dan dapat meningkatkan hingga maksimal 5 instans `t3.micro` On-Demand.

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

[aws]
aws_region_name = <your Wilayah AWS>

[vpc public]
master_subnet_id = <your subnet>
vpc_id = <your VPC>

[cluster slurm]
key_name = <your EC2 keypair name>
base_os = alinux2                # optional, defaults to alinux2
scheduler = slurm
master_instance_type = t3.micro    # optional, defaults to t3.micro
vpc_settings = public
queue_settings = spot,ondemand

[queue spot]
compute_resource_settings = spot_i1
compute_type = spot              # optional, defaults to ondemand

[compute_resource spot_i1]
instance_type = t3.micro
min_count = 1                   # optional, defaults to 0
initial_count = 2               # optional, defaults to 0
```

```
[queue ondemand]
compute_resource_settings = ondemand_i1

[compute_resource ondemand_i1]
instance_type = t3.micro
max_count = 5                                # optional, defaults to 10
```

Son of Grid Engine (**sge**) dan Torque Resource Manager (**torque**)

Note

Contoh ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Contoh berikut meluncurkan cluster dengan torque atau sge scheduler. Untuk menggunakan SGE, ubah `scheduler = torque` ke `scheduler = sge`. Konfigurasi contoh memungkinkan maksimum 5 node bersamaan, dan skala turun menjadi dua ketika tidak ada pekerjaan yang berjalan selama 10 menit.

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

[aws]
aws_region_name = <your Wilayah 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

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Jika Anda menggunakan versi ini, Anda dapat terus menggunakannya, atau mengatasi masalah dukungan dari tim AWS layanan dan AWS Support.

AWS Batch (**awsbatch**)

Contoh berikut meluncurkan cluster dengan awsbatchescheduler. Ini diatur untuk memilih jenis instance yang lebih baik berdasarkan kebutuhan sumber daya pekerjaan Anda.

Konfigurasi contoh memungkinkan maksimum 40 v bersamaan CPUs, dan skala turun ke nol ketika tidak ada pekerjaan yang berjalan selama 10 menit (dapat disesuaikan menggunakan [scaledown_idletime](#) pengaturan).

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

[aws]
aws_region_name = <your Wilayah AWS>

[vpc public]
master_subnet_id = <your subnet>
vpc_id = <your VPC>

[cluster awsbatchescheduler]
scheduler = awsbatchescheduler
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
```

Bagaimana cara AWS ParallelCluster kerja

AWS ParallelCluster dibangun tidak hanya sebagai cara untuk mengelola cluster, tetapi sebagai referensi tentang cara menggunakan AWS layanan untuk membangun lingkungan HPC Anda.

Topik

- [AWS ParallelCluster proses](#)
- [AWS layanan yang digunakan oleh AWS ParallelCluster](#)
- [AWS ParallelCluster Auto Scaling](#)

AWS ParallelCluster proses

Bagian ini hanya berlaku untuk kluster HPC yang digunakan dengan salah satu penjadwal pekerjaan tradisional yang didukung (SGE, Slurm, atau Torque). Saat digunakan dengan penjadwal ini, AWS ParallelCluster mengelola penyediaan dan penghapusan node komputasi dengan berinteraksi dengan grup Auto Scaling dan penjadwal pekerjaan yang mendasarinya.

Untuk cluster HPC yang didasarkan pada AWS Batch, AWS ParallelCluster bergantung pada kemampuan yang disediakan oleh AWS Batch untuk manajemen node komputasi.

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk 2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS

Topik

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

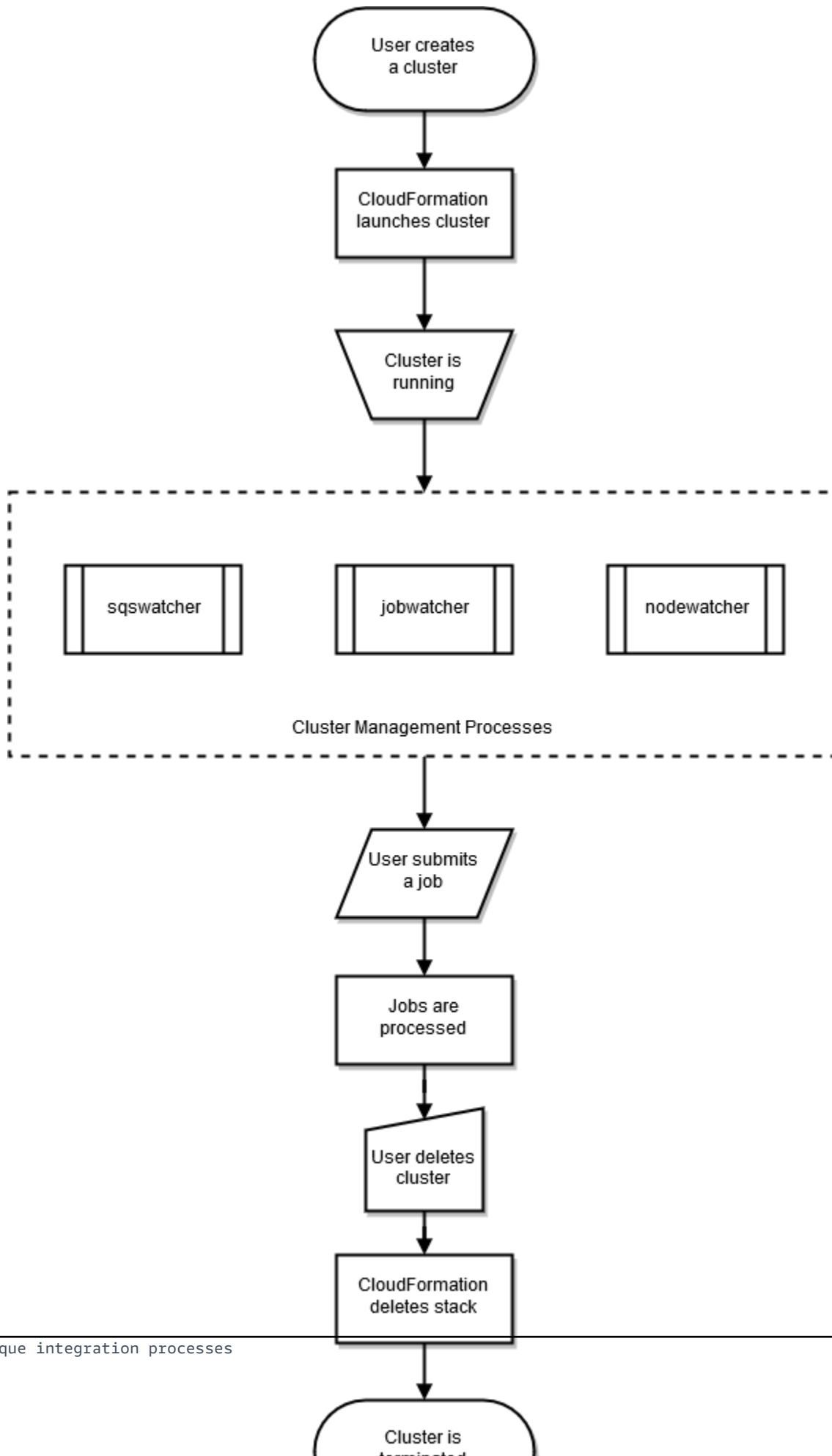
SGE and Torque integration processes

Note

Bagian ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE and Torque penjadwal, Amazon SNS, dan Amazon SQS.

Gambaran umum

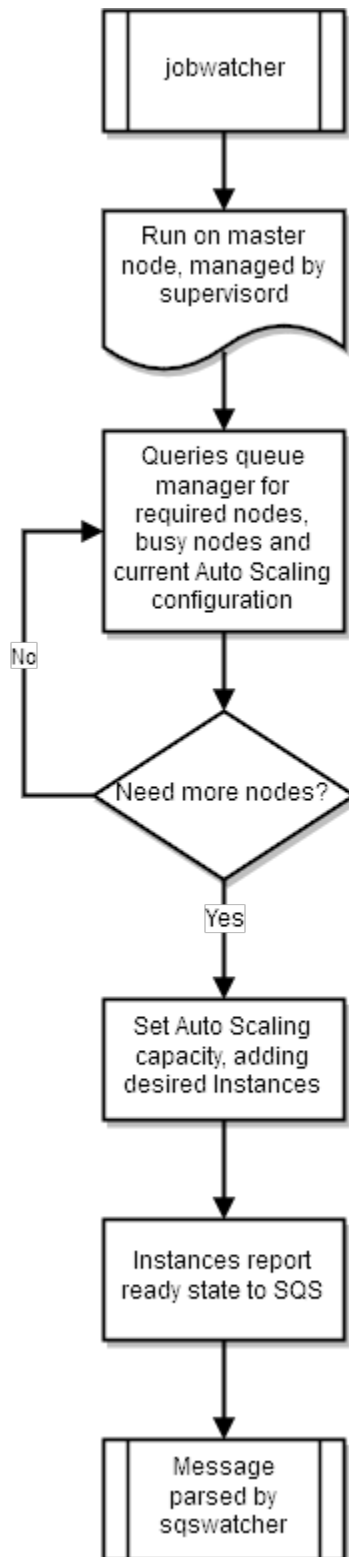
Siklus hidup cluster dimulai setelah dibuat oleh pengguna. Biasanya, sebuah cluster dibuat dari Command Line Interface (CLI). Setelah dibuat, sebuah cluster ada sampai dihapus. AWS ParallelCluster daemon berjalan pada node cluster, terutama untuk mengelola elastisitas cluster HPC. Diagram berikut menunjukkan alur kerja pengguna dan siklus hidup cluster. Bagian berikut menjelaskan AWS ParallelCluster daemon yang digunakan untuk mengelola cluster.



Dengan SGE and Torque penjadwal, AWS ParallelCluster penggunaan `nodewatcher`, `jobwatcher`, dan `sqswatcher` proses.

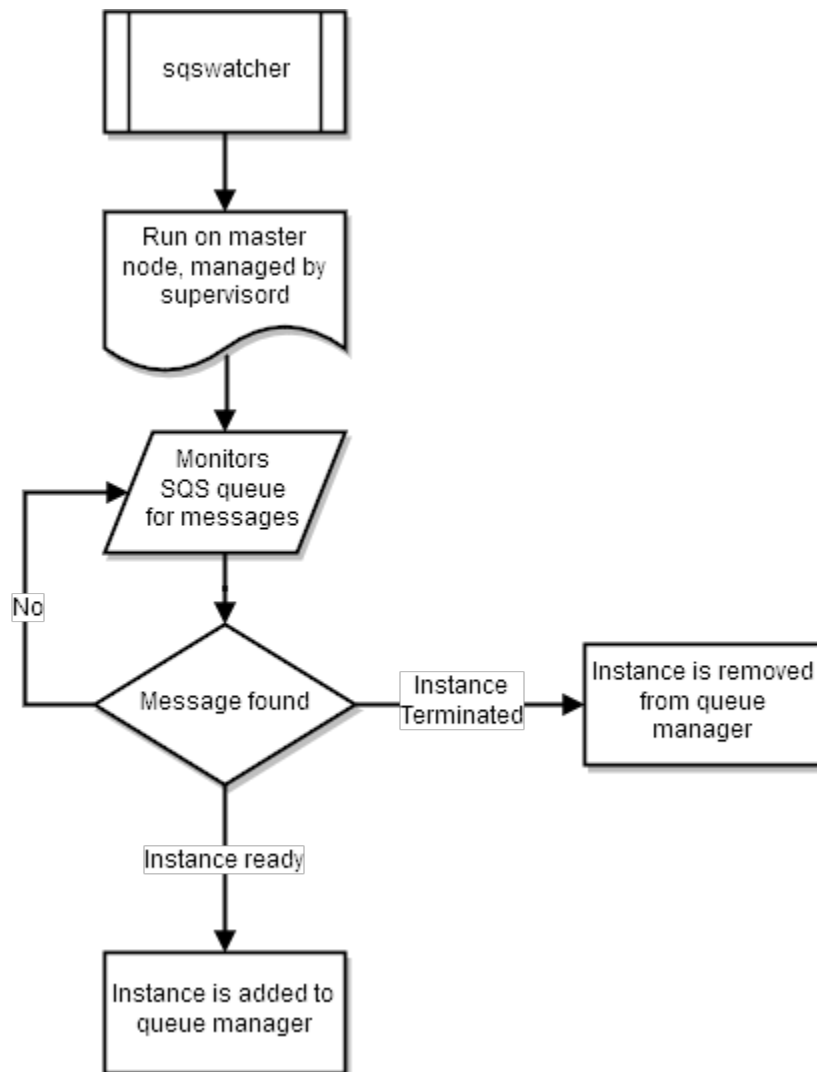
jobwatcher

Saat cluster berjalan, proses yang dimiliki oleh pengguna root memantau penjadwal yang dikonfigurasi (SGE atau Torque). Setiap menit mengevaluasi antrian untuk memutuskan kapan harus meningkatkan.



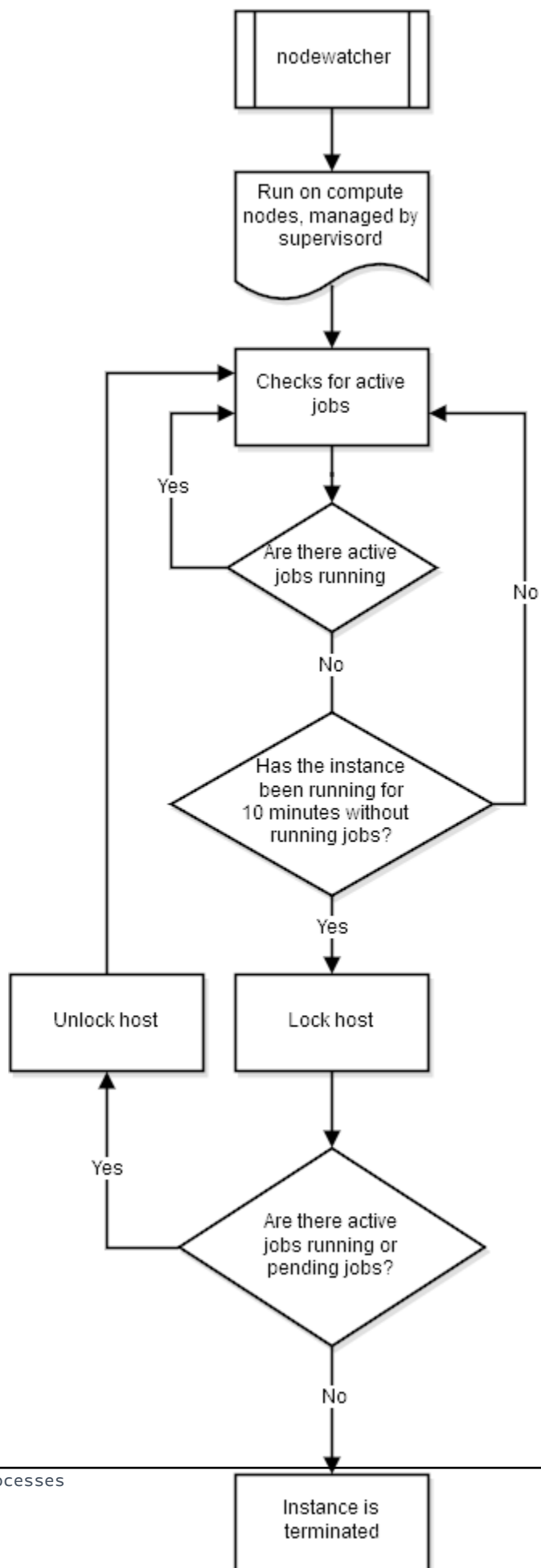
sqswatcher

`sqswatcher` Proses ini memantau pesan Amazon SQS yang dikirim oleh Auto Scaling untuk memberi tahu Anda tentang perubahan status dalam kluster. Ketika sebuah instance online, ia mengirimkan pesan “siap instance” ke Amazon SQS. Pesan ini diambil oleh `sqswatcher`, berjalan pada node kepala. Pesan-pesan ini digunakan untuk memberi tahu pengelola antrian saat instance baru online atau dihentikan, sehingga pesan tersebut dapat ditambahkan atau dihapus dari antrian.



nodewatcher

`nodewatcher` Proses berjalan pada setiap node dalam armada komputasi. Setelah `scaledown_idletime` periode, seperti yang didefinisikan oleh pengguna, instance dihentikan.



Slurm integration processes

Dengan Slurm penjadwal, AWS ParallelCluster penggunaan `clustermgtd` dan `computemgt` proses.

clustermgtd

Cluster yang berjalan dalam mode heterogen (ditunjukkan dengan menentukan [queue_settings](#) nilai) memiliki proses daemon manajemen klaster (`clustermgtd`) yang berjalan pada node kepala. Tugas-tugas ini dilakukan oleh daemon manajemen cluster.

- Pembersihan partisi tidak aktif
- Manajemen kapasitas statis: pastikan kapasitas statis selalu naik dan sehat
- Sinkronkan penjadwal dengan Amazon EC2.
- Pembersihan contoh yatim piatu
- Mengembalikan status node scheduler pada EC2 penghentian Amazon yang terjadi di luar alur kerja penangguhan
- Manajemen EC2 instans Amazon yang tidak sehat (gagal pemeriksaan EC2 kesehatan Amazon)
- Manajemen acara pemeliharaan terjadwal
- Manajemen node Scheduler yang tidak sehat (gagal pemeriksaan kesehatan Scheduler)

computemgtd

Cluster yang berjalan dalam mode heterogen (ditunjukkan dengan menentukan [queue_settings](#) nilai) memiliki proses daemon manajemen komputasi (`computemgtd`) yang berjalan pada setiap node komputasi. Setiap lima (5) menit, daemon manajemen komputasi menegaskan bahwa node kepala dapat dijangkau dan sehat. Jika lima (5) menit berlalu di mana simpul kepala tidak dapat dicapai atau tidak sehat, simpul komputasi dimatikan.

AWS layanan yang digunakan oleh AWS ParallelCluster

Layanan Amazon Web Services (AWS) berikut digunakan oleh AWS ParallelCluster.

Topik

- [AWS Auto Scaling](#)
- [AWS Batch](#)

- [AWS CloudFormation](#)
- [Amazon CloudWatch](#)
- [CloudWatch Log Amazon](#)
- [AWS CodeBuild](#)
- [Amazon DynamoDB](#)
- [Amazon Elastic Block Store](#)
- [Amazon Elastic Compute Cloud](#)
- [Amazon Elastic Container Registry](#)
- [Amazon EFS](#)
- [Amazon FSx untuk 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

Bagian ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan. AWS Auto Scaling

AWS Auto Scaling adalah layanan yang memantau aplikasi Anda dan secara otomatis menyesuaikan kapasitas berdasarkan persyaratan layanan spesifik dan berubah Anda. Layanan ini mengelola ComputeFleet instans Anda sebagai grup Auto Scaling. Grup dapat didorong secara elastis oleh perubahan beban kerja Anda atau diperbaiki secara statis oleh konfigurasi instans awal Anda.

AWS Auto Scaling digunakan dengan ComputeFleet instance tetapi tidak digunakan dengan AWS Batch cluster.

Untuk informasi lebih lanjut tentang AWS Auto Scaling, lihat <https://aws.amazon.com/autoscaling/> dan <https://docs.aws.amazon.com/autoscaling/>.

AWS Batch

AWS Batch adalah layanan penjadwal pekerjaan AWS terkelola. Ini secara dinamis menyediakan kuantitas dan jenis sumber daya komputasi yang optimal (misalnya, CPU atau instance yang dioptimalkan memori) dalam cluster. AWS Batch Sumber daya ini disediakan berdasarkan persyaratan spesifik pekerjaan batch Anda, termasuk persyaratan volume. Dengan AWS Batch, Anda tidak perlu menginstal atau mengelola perangkat lunak komputasi batch tambahan atau cluster server untuk menjalankan pekerjaan Anda secara efektif.

AWS Batch hanya digunakan dengan AWS Batch cluster.

Untuk informasi lebih lanjut tentang AWS Batch, lihat <https://aws.amazon.com/batch/> dan <https://docs.aws.amazon.com/batch/>.

AWS CloudFormation

AWS CloudFormation adalah infrastructure-as-code layanan yang menyediakan bahasa umum untuk model dan penyediaan AWS dan sumber daya aplikasi pihak ketiga di lingkungan cloud Anda. Ini adalah layanan utama yang digunakan oleh AWS ParallelCluster. Setiap cluster di AWS ParallelCluster direpresentasikan sebagai tumpukan, dan semua sumber daya yang dibutuhkan oleh setiap cluster didefinisikan dalam AWS ParallelCluster AWS CloudFormation template. Dalam kebanyakan kasus, perintah AWS ParallelCluster CLI secara langsung sesuai dengan perintah AWS CloudFormation stack, seperti membuat, memperbarui, dan menghapus perintah. Instans yang diluncurkan dalam kluster membuat panggilan HTTPS ke AWS CloudFormation titik akhir di Wilayah AWS mana cluster diluncurkan.

Untuk informasi lebih lanjut tentang AWS CloudFormation, lihat <https://aws.amazon.com/cloudformation/> dan <https://docs.aws.amazon.com/cloudformation/>.

Amazon CloudWatch

Amazon CloudWatch (CloudWatch) adalah layanan pemantauan dan observabilitas yang memberi Anda data dan wawasan yang dapat ditindaklanjuti. Wawasan ini dapat digunakan untuk memantau

aplikasi Anda, menanggapi perubahan kinerja dan pengecualian layanan, dan mengoptimalkan pemanfaatan sumber daya. In AWS ParallelCluster, CloudWatch digunakan untuk dasbor, untuk memantau dan mencatat langkah-langkah pembuatan gambar Docker dan output AWS Batch pekerjaan.

Sebelum AWS ParallelCluster versi 2.10.0, hanya CloudWatch digunakan dengan AWS Batch cluster.

Untuk informasi lebih lanjut tentang CloudWatch, lihat <https://aws.amazon.com/cloudwatch/> dan <https://docs.aws.amazon.com/cloudwatch/>.

CloudWatch Log Amazon

Amazon CloudWatch CloudWatch Logs (Log) adalah salah satu fitur inti Amazon CloudWatch. Anda dapat menggunakannya untuk memantau, menyimpan, melihat, dan mencari file log untuk banyak komponen yang digunakan oleh AWS ParallelCluster.

Sebelum AWS ParallelCluster versi 2.6.0, CloudWatch Log hanya digunakan dengan AWS Batch cluster.

Untuk informasi selengkapnya, lihat [Integrasi dengan Amazon CloudWatch Logs](#).

AWS CodeBuild

AWS CodeBuild (CodeBuild) adalah layanan integrasi berkelanjutan AWS terkelola yang mematuhi kode sumber, menjalankan pengujian, dan menghasilkan paket perangkat lunak yang siap digunakan. In AWS ParallelCluster, CodeBuild digunakan untuk secara otomatis dan transparan membangun gambar Docker saat cluster dibuat.

CodeBuild hanya digunakan dengan AWS Batch cluster.

Untuk informasi lebih lanjut tentang CodeBuild, lihat <https://aws.amazon.com/codebuild/> dan <https://docs.aws.amazon.com/codebuild/>.

Amazon DynamoDB

Amazon DynamoDB (DynamoDB) adalah layanan database NoSQL yang cepat dan fleksibel. Ini digunakan untuk menyimpan informasi status minimal cluster. Node kepala melacak instance yang disediakan dalam tabel DynamoDB.

DynamoDB tidak digunakan dengan cluster. AWS Batch

Untuk informasi selengkapnya tentang DynamoDB, lihat dan <https://aws.amazon.com/dynamodb/https://docs.aws.amazon.com/dynamodb/>

Amazon Elastic Block Store

Amazon Elastic Block Store (Amazon EBS) adalah layanan penyimpanan blok berkinerja tinggi yang menyediakan penyimpanan persisten untuk volume bersama. Semua pengaturan Amazon EBS dapat dilewatkan melalui konfigurasi. Volume Amazon EBS dapat diinisialisasi kosong atau dari snapshot Amazon EBS yang ada.

Untuk informasi selengkapnya tentang Amazon EBS, lihat <https://aws.amazon.com/ebs/> dan <https://docs.aws.amazon.com/ebs/>.

Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud (Amazon EC2) menyediakan kapasitas komputasi untuk AWS ParallelCluster. Node head dan compute adalah EC2 instance Amazon. Tipe instans apa pun yang mendukung HVM dapat dipilih. Node head dan compute dapat berupa tipe instance yang berbeda. Selain itu, jika beberapa antrian digunakan, beberapa atau semua node komputasi juga dapat diluncurkan sebagai Instans Spot. Volume penyimpanan instans yang ditemukan pada instance dipasang sebagai volume LVM bergaris.

Untuk informasi selengkapnya tentang Amazon EC2, lihat <https://aws.amazon.com/ec2/> dan <https://docs.aws.amazon.com/ec2/>.

Amazon Elastic Container Registry

Amazon Elastic Container Registry (Amazon ECR) adalah registri kontainer Docker yang dikelola sepenuhnya yang memudahkan untuk menyimpan, mengelola, dan menyebarkan gambar kontainer Docker. Di AWS ParallelCluster, Amazon ECR menyimpan gambar Docker yang dibuat saat cluster dibuat. Gambar Docker kemudian digunakan oleh AWS Batch untuk menjalankan kontainer untuk pekerjaan yang dikirimkan.

Amazon ECR hanya digunakan dengan AWS Batch cluster.

Untuk informasi selengkapnya, silakan lihat <https://aws.amazon.com/ecr/> dan <https://docs.aws.amazon.com/ecr/>.

Amazon EFS

Amazon Elastic File System (Amazon EFS) menyediakan sistem file NFS elastis yang sederhana, dapat diskalakan, dan dikelola sepenuhnya untuk digunakan dengan AWS Cloud layanan dan sumber daya lokal. Amazon EFS digunakan saat [efs_settings](#) pengaturan ditentukan dan mengacu pada [\[efs\]bagian](#). Support untuk Amazon EFS telah ditambahkan dalam AWS ParallelCluster versi 2.1.0.

Untuk informasi selengkapnya tentang Amazon EFS, lihat <https://aws.amazon.com/efs/> dan <https://docs.aws.amazon.com/efs/>.

Amazon FSx untuk Lustre

FSx untuk Lustre menyediakan sistem file berkinerja tinggi yang menggunakan sistem file Lustre open-source. FSx [untuk Lustre digunakan ketika fsx_settings pengaturan ditentukan dan mengacu pada bagian. \[fsx\]](#) Support for FSx for Lustre ditambahkan di AWS ParallelCluster versi 2.2.1.

Untuk informasi lebih lanjut tentang FSx Lustre, lihat <https://aws.amazon.com/fsx/lustre/> dan <https://docs.aws.amazon.com/fsx/>.

AWS Identity and Access Management

AWS Identity and Access Management (IAM) digunakan di dalam AWS ParallelCluster untuk menyediakan peran IAM yang paling tidak memiliki hak istimewa untuk Amazon EC2 untuk instance yang spesifik untuk setiap cluster individu. AWS ParallelCluster instance diberikan akses hanya ke panggilan API tertentu yang diperlukan untuk menerapkan dan mengelola klaster.

Dengan AWS Batch cluster, peran IAM juga dibuat untuk komponen yang terlibat dengan proses pembuatan image Docker saat cluster dibuat. Komponen-komponen ini termasuk fungsi Lambda yang diizinkan untuk menambah dan menghapus gambar Docker ke dan dari repositori Amazon ECR. Mereka juga menyertakan fungsi yang diizinkan untuk menghapus bucket Amazon S3 yang dibuat untuk cluster dan CodeBuild proyek. Ada juga peran untuk AWS Batch sumber daya, contoh, dan pekerjaan.

Untuk informasi lebih lanjut tentang IAM, lihat <https://aws.amazon.com/iam/> dan <https://docs.aws.amazon.com/iam/>.

AWS Lambda

AWS Lambda (Lambda) menjalankan fungsi yang mengatur pembuatan gambar Docker. Lambda juga mengelola pembersihan sumber daya cluster khusus, seperti gambar Docker yang disimpan di repositori Amazon ECR dan di Amazon S3.

Untuk informasi lebih lanjut tentang Lambda, lihat <https://aws.amazon.com/lambda/> dan <https://docs.aws.amazon.com/lambda/>

Amazon DCV

Amazon DCV adalah protokol tampilan jarak jauh berkinerja tinggi yang menyediakan cara aman untuk mengirimkan desktop jarak jauh dan streaming aplikasi ke perangkat apa pun dalam berbagai kondisi jaringan. Amazon DCV digunakan saat [dcv_settings](#) pengaturan ditentukan dan mengacu pada [\[dcv\]bagian](#). Support untuk Amazon DCV telah ditambahkan dalam AWS ParallelCluster versi 2.5.0.

Untuk informasi selengkapnya tentang Amazon DCV, lihat <https://aws.amazon.com/hpc/dcv/> dan <https://docs.aws.amazon.com/dcv/>

Amazon Route 53

Amazon Route 53 (Route 53) digunakan untuk membuat zona yang dihosting dengan nama host dan nama domain yang memenuhi syarat untuk setiap node komputasi.

Untuk informasi lebih lanjut tentang Rute 53, lihat <https://aws.amazon.com/route53/> dan <https://docs.aws.amazon.com/route53/>.

Amazon Simple Notification Service

Note

Bagian ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan Amazon Simple Notification Service.

Amazon Simple Notification Service (Amazon SNS) menerima notifikasi dari Auto Scaling. Peristiwa ini disebut peristiwa siklus hidup dan dihasilkan saat instance diluncurkan atau dihentikan dalam

grup Auto Scaling. Di dalamnya AWS ParallelCluster, topik Amazon SNS untuk grup Auto Scaling berlangganan antrian Amazon SQS.

Amazon SNS tidak digunakan dengan AWS Batch cluster.

Untuk informasi selengkapnya tentang Amazon SNS, lihat <https://aws.amazon.com/sns/> dan <https://docs.aws.amazon.com/sns/>

Amazon Simple Queue Service

Note

Bagian ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan Amazon Simple Queue Service.

Amazon Simple Queue Service (Amazon SQS) menyimpan notifikasi yang dikirim dari Auto Scaling, notifikasi yang dikirim melalui Amazon SNS, dan notifikasi yang dikirim dari node komputasi. Amazon SQS memisahkan pengiriman notifikasi dari penerimaan notifikasi. Hal ini memungkinkan node kepala untuk menangani notifikasi melalui proses polling. Dalam proses ini, node kepala menjalankan Amazon SQSwatcher dan melakukan polling antrian. Auto Scaling dan node komputasi memposting pesan ke antrian.

Amazon SQS tidak digunakan dengan AWS Batch cluster.

Untuk informasi selengkapnya tentang Amazon SQS, lihat <https://aws.amazon.com/sqs/> dan <https://docs.aws.amazon.com/sqs/>

Amazon Simple Storage Service

Amazon Simple Storage Service (Amazon S3) AWS ParallelCluster menyimpan template yang terletak di masing-masing. Wilayah AWS AWS ParallelCluster dapat dikonfigurasi untuk memungkinkan alat CLI/SDK menggunakan Amazon S3.

Saat Anda menggunakan AWS Batch cluster, bucket Amazon S3 di akun Anda digunakan untuk menyimpan data terkait. Misalnya, bucket menyimpan artefak yang dibuat saat image dan skrip Docker dibuat dari pekerjaan yang dikirimkan.

Untuk informasi selengkapnya, silakan lihat <https://aws.amazon.com/s3/> dan <https://docs.aws.amazon.com/s3/>.

Amazon VPC

Amazon VPC mendefinisikan jaringan yang digunakan oleh node di cluster Anda. [Pengaturan VPC untuk cluster didefinisikan di bagian ini. \[vpc\]](#)

Untuk informasi selengkapnya tentang Amazon VPC, lihat <https://aws.amazon.com/vpc/> dan <https://docs.aws.amazon.com/vpc/>

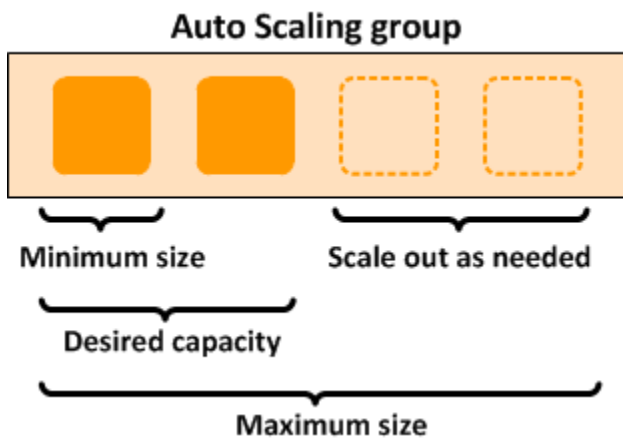
AWS ParallelCluster Auto Scaling

Note

Bagian ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal. Anda dapat terus menggunakannya dalam versi hingga dan termasuk 2.11.4, tetapi mereka tidak memenuhi syarat untuk pembaruan masa depan atau dukungan pemecahan masalah dari tim layanan AWS dan Support. AWS Dimulai dengan AWS ParallelCluster versi 2.9.0, Auto Scaling tidak didukung untuk digunakan Slurm Workload Manager (Slurm). Untuk belajar tentang Slurm dan beberapa penskalaan antrian, lihat [Beberapa mode antrian tutorial](#)

Strategi penskalaan otomatis yang dijelaskan dalam topik ini berlaku untuk kluster HPC yang digunakan dengan salah satu Son of Grid Engine (SGE) atau Torque Resource Manager (Torque). Saat digunakan dengan salah satu penjadwal ini, AWS ParallelCluster implementasikan kemampuan penskalaan dengan mengelola grup Auto Scaling dari node komputasi, dan kemudian mengubah konfigurasi penjadwal sesuai kebutuhan. Untuk kluster HPC yang didasarkan pada AWS Batch, AWS ParallelCluster bergantung pada kemampuan penskalaan elastis yang disediakan oleh penjadwal pekerjaan AWS terkelola. Untuk informasi selengkapnya, lihat [Apa itu EC2 Auto Scaling Amazon](#) di Panduan Pengguna Amazon Auto EC2 Scaling.

Cluster yang digunakan AWS ParallelCluster elastis dalam beberapa cara. Pengaturan [initial_queue_size](#) menentukan nilai ukuran minimum grup ComputeFleet Auto Scaling, dan juga nilai kapasitas yang diinginkan. Menyetel [max_queue_size](#) menentukan nilai ukuran maksimum grup ComputeFleet Auto Scaling.



Menskalakan

Setiap menit, sebuah proses yang disebut [jobwatcher](#) berjalan pada node kepala. Ini mengevaluasi jumlah instans saat ini yang diperlukan oleh pekerjaan yang tertunda dalam antrian. Jika jumlah total node sibuk dan node yang diminta lebih besar dari nilai yang diinginkan saat ini dalam grup Auto Scaling, itu menambahkan lebih banyak instance. Jika Anda mengirimkan lebih banyak pekerjaan, antrian akan dievaluasi ulang dan grup Auto Scaling diperbarui, hingga yang ditentukan. [max_queue_size](#)

Dengan SGE scheduler, setiap pekerjaan membutuhkan sejumlah slot untuk dijalankan (satu slot sesuai dengan satu unit pemrosesan, misalnya, vCPU). Untuk mengevaluasi jumlah instance yang diperlukan untuk melayani pekerjaan yang saat ini tertunda, `jobwatcher` membagi jumlah total slot yang diminta dengan kapasitas node komputasi tunggal. Kapasitas node komputasi yang sesuai dengan jumlah v yang tersedia CPUs bergantung pada jenis EC2 instans Amazon yang ditentukan dalam konfigurasi cluster.

Dengan Slurm (sebelum AWS ParallelCluster versi 2.9.0) dan Torque penjadwal, setiap pekerjaan mungkin memerlukan sejumlah node dan sejumlah slot untuk setiap node, tergantung pada keadaan. Untuk setiap permintaan, `jobwatcher` menentukan jumlah node komputasi yang diperlukan untuk memenuhi persyaratan komputasi baru. Sebagai contoh, mari kita asumsikan sebuah cluster dengan `c5.2xlarge` (8 vCPU) sebagai jenis instance komputasi, dan tiga pekerjaan tertunda antrian dengan persyaratan berikut:

- `job1:2 node /4 slot` masing-masing
- `job2:3 node/2 slot` masing-masing
- `job3:1 simpul/4 slot` masing-masing

Dalam contoh ini, `jobwatcher` memerlukan tiga instans komputasi baru dalam grup Auto Scaling untuk melayani tiga pekerjaan.

Batasan saat ini: logika penskalaan otomatis tidak mempertimbangkan node sibuk yang dimuat sebagian. Misalnya, node yang menjalankan pekerjaan dianggap sibuk bahkan jika ada slot kosong.

Menskalakan

Pada setiap node komputasi, proses yang disebut `nodewatcher` berjalan dan mengevaluasi waktu idle node. Sebuah instance dihentikan ketika kedua kondisi berikut terpenuhi:

- Sebuah instance tidak memiliki pekerjaan untuk jangka waktu yang lebih lama dari `scaledown_idletime` (pengaturan default adalah 10 menit)
- Tidak ada pekerjaan yang tertunda di cluster

Untuk menghentikan instance, `nodewatcher` panggil operasi `TerminateInstanceInAutoScalingGroup` API, yang menghapus instance jika ukuran grup Auto Scaling setidaknya adalah ukuran grup Auto Scaling minimum. Proses ini menurunkan skala cluster tanpa memengaruhi pekerjaan yang sedang berjalan. Ini juga memungkinkan cluster elastis dengan jumlah dasar instans tetap.

Cluster statis

Nilai penskalaan otomatis sama untuk HPC seperti halnya beban kerja lainnya. Satu-satunya perbedaan adalah bahwa AWS ParallelCluster memiliki kode yang membuatnya berinteraksi lebih cerdas. Misalnya, jika klaster statis diperlukan, Anda mengatur `max_queue_size` parameter `initial_queue_size` dan ke ukuran cluster yang tepat yang diperlukan,. Dan kemudian Anda mengatur `maintain_initial_size` parameter ke `true`. Hal ini menyebabkan grup `ComputeFleet` Auto Scaling memiliki nilai yang sama untuk kapasitas minimum, maksimum, dan yang diinginkan.

Tutorial

Tutorial berikut menunjukkan cara memulai AWS ParallelCluster, dan memberikan panduan praktik terbaik untuk beberapa tugas umum.

Topik

- [Menjalankan pekerjaan pertama Anda di AWS ParallelCluster](#)
- [Membangun AWS ParallelCluster AMI Kustom](#)
- [Menjalankan pekerjaan MPI dengan AWS ParallelCluster dan awsbatch penjadwal](#)
- [Enkripsi disk dengan Kunci KMS khusus](#)
- [Beberapa mode antrian tutorial](#)

Menjalankan pekerjaan pertama Anda di AWS ParallelCluster

Tutorial ini memandu Anda menjalankan pekerjaan Hello World pertama Anda AWS ParallelCluster.

Prasyarat

- AWS ParallelCluster [diinstal](#).
- AWS CLI [itu diinstal dan dikonfigurasi](#).
- Anda memiliki [EC2 key pair](#).
- Anda memiliki peran IAM dengan [izin](#) yang diperlukan untuk menjalankan CLI [pcluster](#).

Memverifikasi instalasi Anda

Pertama, kami memverifikasi bahwa AWS ParallelCluster diinstal dan dikonfigurasi dengan benar.

```
$ pcluster version
```

Ini mengembalikan versi yang sedang berjalan dari AWS ParallelCluster. Jika output memberi Anda pesan tentang konfigurasi, Anda perlu menjalankan yang berikut ini untuk mengkonfigurasi AWS ParallelCluster:

```
$ pcluster configure
```

Membuat cluster pertama Anda

Sekarang saatnya membuat cluster pertama Anda. Karena beban kerja untuk tutorial ini tidak intensif kinerja, kita dapat menggunakan ukuran instance default. `t2.micro` (Untuk beban kerja produksi, Anda memilih ukuran instans yang paling sesuai dengan kebutuhan Anda.)

Mari kita sebut cluster hello-world Anda.

```
$ pcluster create hello-world
```

Ketika cluster dibuat, Anda melihat output yang mirip dengan berikut ini:

```
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/
```

Pesan `CREATE_COMPLETE` menunjukkan bahwa cluster berhasil dibuat. Outputnya juga memberi kita alamat IP publik dan pribadi dari node kepala kita. Kami membutuhkan IP ini untuk masuk.

Masuk ke node kepala Anda

Gunakan file pem OpenSSH Anda untuk masuk ke node kepala Anda.

```
pcluster ssh hello-world -i /path/to/keyfile.pem
```

Setelah Anda masuk, jalankan perintah `qhost` untuk memverifikasi bahwa node komputasi Anda telah diatur dan dikonfigurasi.

```
$ 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 0.0	1x-amd64	2	1	2	2	0.15	3.7G	130.8M	1024.0M
-------------------------	----------	---	---	---	---	------	------	--------	---------

Outputnya menunjukkan bahwa kita memiliki dua node komputasi di cluster kita, keduanya dengan dua utas yang tersedia untuk mereka.

Menjalankan pekerjaan pertama Anda menggunakan SGE

Note

Contoh ini hanya berlaku untuk AWS ParallelCluster versi hingga dan termasuk versi 2.11.4. Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Selanjutnya, kami membuat pekerjaan yang tidur sebentar dan kemudian mengeluarkan nama hostnya sendiri.

Buat file bernama `hellojob.sh`, dengan konten berikut.

```
#!/bin/bash
sleep 30
echo "Hello World from $(hostname)"
```

Selanjutnya, kirimkan pekerjaan menggunakan `qsub`, dan verifikasi bahwa itu berjalan.

```
$ qsub hellojob.sh
Your job 1 ("hellojob.sh") has been submitted
```

Sekarang, Anda dapat melihat antrian Anda dan memeriksa status pekerjaan.

```
$ 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
```

Output menunjukkan bahwa pekerjaan saat ini dalam keadaan berjalan. Tunggu 30 detik hingga pekerjaan selesai, lalu jalankan `qstat` lagi.

```
$ qstat
$
```

Sekarang tidak ada pekerjaan dalam antrian, kita dapat memeriksa output di direktori kita saat ini.

```
$ 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
```

Dalam output, kita melihat file "e1" dan "o1" dalam skrip pekerjaan kami. Karena e1 file kosong, tidak ada output ke stderr. Jika kita melihat o1 file, kita dapat melihat output dari pekerjaan kita.

```
$ cat hellojob.sh.o1
Hello World from ip-192-168-1-125
```

Outputnya juga menunjukkan bahwa pekerjaan kami berhasil berjalan pada instanceip-192-168-1-125.

Untuk mempelajari lebih lanjut tentang membuat dan menggunakan cluster, lihat [Praktik terbaik](#).

Membangun AWS ParallelCluster AMI Kustom

Important

Kami tidak menyarankan membangun AMI kustom sebagai pendekatan untuk menyesuaikan AWS ParallelCluster.

Ini karena, setelah Anda membuat AMI sendiri, Anda tidak lagi menerima pembaruan atau perbaikan bug dengan rilis future. AWS ParallelCluster Selain itu, jika Anda membuat AMI kustom, Anda harus mengulangi langkah-langkah yang Anda gunakan untuk membuat AMI kustom Anda dengan setiap AWS ParallelCluster rilis baru.

Sebelum membaca lebih lanjut, kami sarankan Anda terlebih dahulu memeriksa bagian [Tindakan Bootstrap Kustom](#) untuk menentukan apakah modifikasi yang ingin Anda buat dapat ditulis dan didukung dengan AWS ParallelCluster rilis future.

Meskipun membangun AMI khusus tidak ideal (karena alasan yang disebutkan sebelumnya), masih ada skenario di mana membangun AMI khusus AWS ParallelCluster diperlukan. Tutorial ini memandu Anda melalui proses membangun AMI khusus untuk skenario ini.

Note

Dimulai dengan AWS ParallelCluster versi 2.6.1, sebagian besar resep penginstalan dilewati secara default saat meluncurkan node. Ini untuk meningkatkan waktu startup. Untuk menjalankan semua resep instalasi untuk kompatibilitas mundur yang lebih baik dengan mengorbankan waktu startup, tambahkan "skip_install_recipes" : "no" ke cluster kunci dalam [extra_json](#) pengaturan. Misalnya:

```
extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
```

Prasyarat

- AWS ParallelCluster [diinstal](#).
- AWS CLI [itu diinstal dan dikonfigurasi](#).
- Anda memiliki [EC2 key pair](#).
- Anda memiliki peran IAM dengan [izin](#) yang diperlukan untuk menjalankan CLI [pcluster](#).

Cara Menyesuaikan AWS ParallelCluster AMI

Ada tiga cara untuk menggunakan AWS ParallelCluster AMI kustom yang dijelaskan di bagian berikutnya. Dua dari tiga metode ini mengharuskan Anda untuk membangun AMI baru yang tersedia di bawah Anda Akun AWS. Metode ketiga (Gunakan AMI Kustom saat Runtime) tidak mengharuskan Anda membuat apa pun terlebih dahulu, tetapi menambah risiko pada penerapan. Pilih metode yang paling sesuai dengan kebutuhan Anda.

Memodifikasi AMI

Ini adalah metode yang paling aman dan paling direkomendasikan. Karena AWS ParallelCluster AMI dasar sering diperbarui dengan rilis baru, AMI ini memiliki semua komponen yang diperlukan AWS ParallelCluster untuk berfungsi saat diinstal dan dikonfigurasi. Anda bisa mulai dengan ini sebagai basis.

New EC2 console

1. Dalam daftar AWS ParallelCluster AMI, temukan AMI yang sesuai dengan spesifik Wilayah AWS yang Anda gunakan. Daftar AMI yang Anda pilih harus sesuai dengan versi AWS ParallelCluster yang Anda gunakan. Jalankan `pcluster version` untuk memverifikasi versi. Untuk AWS ParallelCluster versi 2.11.9, buka <https://github.com/aws/aws-parallelcluster/blob/v2.11.9/amis> .txt. Untuk memilih versi lain, gunakan tautan yang sama, pilih tombol Tag: 2.11.9, pilih tab Tag, lalu pilih versi yang sesuai.
2. Masuk ke AWS Management Console dan buka EC2 konsol Amazon di <https://console.aws.amazon.com/ec2/>.
3. Di EC2 Dasbor Amazon, pilih Launch instance.
4. Pada gambar Aplikasi dan OS, pilih Jelajahi lebih banyak AMIs, navigasikan ke Komunitas AMIs, dan masukkan ID AWS ParallelCluster AMI untuk Anda Wilayah AWS ke dalam kotak pencarian.
5. Pilih AMI, pilih jenis dan properti Instance Anda, pilih pasangan Kunci Anda, dan Launch instance.
6. Masuk ke instans Anda menggunakan pengguna OS dan kunci SSH Anda. Untuk informasi selengkapnya, navigasikan ke Instans, pilih instans baru, dan Connect.
7. Sesuaikan instance Anda sesuai kebutuhan.
8. Jalankan perintah berikut untuk menyiapkan instance Anda untuk pembuatan AMI:

```
sudo /usr/local/sbin/ami_cleanup.sh
```

9. Arahkan ke Instans, pilih instans baru, pilih Status instans, dan Stop instance.
10. Buat AMI baru dari instance menggunakan EC2 konsol atau AWS CLI [create-image](#).

Dari EC2 konsol

- a. Pilih Instans di panel navigasi.
 - b. Pilih instance yang Anda buat dan modifikasi.
 - c. Di Tindakan, pilih Gambar dan templat, lalu Buat gambar.
 - d. Pilih Buat Gambar.
11. Masukkan id AMI baru di bidang [custom_ami](#) di konfigurasi cluster Anda.

Old EC2 console

1. Dalam daftar AWS ParallelCluster AMI, temukan AMI yang sesuai dengan spesifik Wilayah AWS yang Anda gunakan. Daftar AMI yang Anda pilih harus sesuai dengan versi AWS ParallelCluster yang Anda gunakan. Jalankan `pcluster version` untuk memverifikasi versi. Untuk AWS ParallelCluster versi 2.11.9, buka <https://github.com/aws/aws-parallelcluster/blob/v2.11.9/amis> .txt. Untuk memilih versi lain, gunakan tautan yang sama, pilih tombol Tag: 2.11.9, pilih tab Tag, lalu pilih versi yang sesuai.
2. Masuk ke AWS Management Console dan buka EC2 konsol Amazon di <https://console.aws.amazon.com/ec2/>.
3. Di EC2 Dasbor Amazon, pilih Launch instance.
4. Pilih Komunitas AMIs, cari ID AWS ParallelCluster AMI, dan Pilih.
5. Pilih jenis instans Anda dan pilih Next: Configure Instance Details, atau Review and Launch untuk meluncurkan instans Anda.
6. Pilih Luncurkan, pilih pasangan Kunci Anda, dan Luncurkan Instans.
7. Masuk ke instans Anda menggunakan pengguna OS dan kunci SSH Anda. Untuk informasi selengkapnya, navigasikan ke Instans, pilih instans baru, dan Connect.
8. Sesuaikan instance Anda sesuai kebutuhan.
9. Jalankan perintah berikut untuk menyiapkan instance Anda untuk pembuatan AMI:

```
sudo /usr/local/sbin/ami_cleanup.sh
```

- 10Arahkan ke Instans, pilih instans baru, pilih Status Instance, dan Stop
- 11Buat AMI baru dari instance menggunakan EC2 konsol atau AWS CLI [create-image](#).

Dari EC2 konsol

- a. Pilih Instans di panel navigasi.
 - b. Pilih instance yang Anda buat dan modifikasi.
 - c. Di Actions, pilih Image, dan kemudian Create Image.
 - d. Pilih Buat Gambar.
- 12Masukkan id AMI baru di bidang [custom_ami](#) di konfigurasi cluster Anda.

Bangun AWS ParallelCluster AMI Kustom

Jika Anda sudah memiliki AMI dan perangkat lunak yang disesuaikan, Anda dapat menerapkan perubahan yang diperlukan AWS ParallelCluster di atasnya.

1. Instal yang berikut ini di sistem lokal Anda, bersama dengan AWS ParallelCluster CLI:

- Packer: temukan versi terbaru untuk OS Anda dari [situs web Packer](#), dan instal. Versi harus setidaknya 1.4.0, tetapi versi terbaru direkomendasikan. Verifikasi bahwa packer perintah tersedia di PATH Anda.

Note

Sebelum AWS ParallelCluster versi 2.8.0, [Berkshelf](#) (yang diinstal dengan menggunakan `install berkshelf`) diperlukan untuk digunakan. `pcluster createami`

2. Konfigurasi Akun AWS kredensial Anda sehingga Packer dapat melakukan panggilan ke operasi AWS API atas nama Anda. Kumpulan minimal izin yang diperlukan yang diperlukan agar Packer berfungsi didokumentasikan di bagian [IAM Task atau Instance Role](#) dari topik Amazon AMI Builder dalam dokumentasi Packer.
3. Gunakan perintah `createami` di AWS ParallelCluster CLI untuk membangun AWS ParallelCluster AMI mulai dari yang Anda berikan sebagai basis:

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

Important

Anda tidak boleh menggunakan AWS ParallelCluster AMI dari cluster yang sedang berjalan seperti `<BASE_AMI>` untuk `createami` perintah. Jika tidak, perintah gagal.

Untuk parameter lainnya, lihat [pcluster createami](#).

4. Perintah di Langkah 4 menjalankan Packer, yang secara khusus melakukan hal berikut:
- a. Meluncurkan instance menggunakan AMI dasar yang disediakan.
 - b. Menerapkan AWS ParallelCluster buku masak ke instance untuk menginstal perangkat lunak yang relevan dan melakukan tugas konfigurasi lain yang diperlukan.

- c. Menghentikan instance.
 - d. Membuat AMI baru dari instance.
 - e. Mengakhiri instance setelah AMI dibuat.
 - f. Mengeluarkan string ID AMI baru yang akan digunakan untuk membuat klaster Anda.
5. Untuk membuat klaster Anda, masukkan ID AMI di bidang [custom_ami](#) dalam konfigurasi cluster Anda.

Note

Jenis instance yang digunakan untuk membangun AWS ParallelCluster AMI kustom adalah `t2.xlarge`. Jenis instans ini tidak memenuhi syarat untuk tingkat AWS gratis, jadi Anda dikenakan biaya untuk setiap instans yang dibuat saat Anda membuat AMI ini.

Gunakan AMI Kustom saat Runtime

Warning

Untuk menghindari risiko menggunakan AMI yang tidak kompatibel dengannya AWS ParallelCluster, sebaiknya Anda menghindari penggunaan metode ini.

Ketika node komputasi diluncurkan dengan potensi belum teruji AMIs saat runtime, ketidakcocokan dengan instalasi runtime dari perangkat lunak yang diperlukan dapat menyebabkan berhenti AWS ParallelCluster bekerja. AWS ParallelCluster

Jika Anda tidak ingin membuat apa pun sebelumnya, Anda dapat menggunakan AMI Anda dan membuat AWS ParallelCluster dari AMI itu.

Dengan metode ini, dibutuhkan waktu lebih lama AWS ParallelCluster untuk dibuat karena semua perangkat lunak yang dibutuhkan AWS ParallelCluster ketika cluster dibuat harus diinstal. Selain itu, peningkatan skala juga membutuhkan waktu lebih lama.

- Masukkan id AMI di bidang [custom_ami](#) dalam konfigurasi cluster Anda.

Menjalankan pekerjaan MPI dengan AWS ParallelCluster dan **awsbatch** penjadwal

Tutorial ini memandu Anda melalui menjalankan pekerjaan MPI dengan awsbatch sebagai penjadwal.

Prasyarat

- AWS ParallelCluster [diinstal](#).
- AWS CLI [itu diinstal dan dikonfigurasi](#).
- Anda memiliki [EC2 key pair](#).
- Anda memiliki peran IAM dengan [izin](#) yang diperlukan untuk menjalankan CLI [pcluster](#).

Membuat cluster

Pertama, mari kita buat konfigurasi untuk cluster yang digunakan awsbatch sebagai scheduler. Pastikan untuk memasukkan data yang hilang di vpc bagian dan key_name bidang dengan sumber daya yang Anda buat pada waktu konfigurasi.

```
[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-#####
```

Anda sekarang dapat memulai pembuatan cluster. Mari kita sebut cluster kita *awsbatch-tutorial*.

```
$ pcluster create -c /path/to/the/created/config/aws_batch.config -t awsbatch awsbatch-
tutorial
```

Ketika cluster dibuat, Anda melihat output yang mirip dengan berikut ini:

```
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
```

Masuk ke node kepala Anda

Perintah [AWS ParallelCluster Batch CLI](#) semuanya tersedia di mesin klien tempat diinstal AWS ParallelCluster. Namun, kita akan SSH ke node kepala dan mengirimkan pekerjaan dari sana. Ini memungkinkan kita untuk memanfaatkan volume NFS yang dibagi antara head dan semua instance Docker yang menjalankan pekerjaan. AWS Batch

Gunakan file pem SSH Anda untuk masuk ke node kepala Anda.

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

Saat Anda masuk, jalankan perintah `awsbqueues` dan `awsbhosts` untuk menampilkan AWS Batch antrian yang dikonfigurasi dan instans Amazon ECS yang sedang berjalan.

```
[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
```

Seperti yang Anda lihat dari output, kami memiliki satu host berjalan tunggal. Ini karena nilai yang kami pilih [min_vcpus](#) dalam konfigurasi. Jika Anda ingin menampilkan detail tambahan tentang AWS Batch antrian dan host, tambahkan `-d` bendera ke perintah.

Menjalankan pekerjaan pertama Anda menggunakan AWS Batch

Sebelum pindah ke MPI, mari kita buat pekerjaan dummy yang tidur sebentar dan kemudian mengeluarkan nama hostnya sendiri, menyapa nama yang diteruskan sebagai parameter.

Buat file bernama "hellojob.sh" dengan konten berikut.

```
#!/bin/bash

sleep 30
echo "Hello $1 from $HOSTNAME"
echo "Hello $1 from $HOSTNAME" > "/shared/secret_message_for_${1}_by_
${AWS_BATCH_JOB_ID}"
```

Selanjutnya, kirimkan pekerjaan menggunakan `awsbsub` dan verifikasi bahwa itu berjalan.

```
$ awsbsub -jn hello -cf hellojob.sh Luca
Job 6efe6c7c-4943-4c1a-baf5-edbfecab5d2 (hello) has been submitted.
```

Lihat antrian Anda, dan periksa status pekerjaan.

```
$ awsbstat
jobId              jobName      status      startedAt
stoppedAt      exitCode
-----
-----
6efe6c7c-4943-4c1a-baf5-edbfecab5d2  hello        RUNNING    2018-11-12 09:41:29  -
```

Output memberikan informasi rinci untuk pekerjaan itu.

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

Perhatikan bahwa pekerjaan saat ini dalam RUNNING keadaan. Tunggu 30 detik hingga pekerjaan selesai, lalu jalankan `awsbatchstat` lagi.

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

Sekarang Anda dapat melihat bahwa pekerjaan itu dalam SUCCEEDED status.

```
$ awsbatchstat -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
```

Karena tidak ada pekerjaan dalam antrian sekarang, kita dapat memeriksa output melalui `awsbout` perintah.

```
$ awsbout 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
2018-11-12 09:41:29: Starting Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
download: s3://amzn-s3-demo-bucket/batch/job-hellojob_sh-1542015680924.sh to tmp/batch/
job-hellojob_sh-1542015680924.sh
2018-11-12 09:42:00: Hello Luca from ip-172-31-4-234
```

Kita dapat melihat bahwa pekerjaan kita berhasil berjalan pada contoh “ip-172-31-4-234”.

Jika Anda melihat ke `/shared` direktori, Anda menemukan pesan rahasia untuk Anda.

Untuk menjelajahi semua fitur yang tersedia yang bukan bagian dari tutorial ini, lihat dokumentasi [AWS ParallelCluster Batch CLI](#). Ketika Anda siap untuk melanjutkan tutorial, mari kita lanjutkan dan lihat cara mengirimkan pekerjaan MPI.

Menjalankan pekerjaan MPI di lingkungan paralel multi-node

Saat masih masuk ke node kepala, buat file di `/shared` direktori bernama `mpi_hello_world.c`. Tambahkan program MPI berikut ke file:

```
// 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();
}
```

Sekarang simpan kode berikut sebagai `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

```

Kami sekarang siap untuk mengirimkan pekerjaan MPI pertama kami dan membuatnya berjalan secara bersamaan di tiga node:

```
$ awsbsub -n 3 -cf submit_mpi.sh
```

Sekarang mari kita pantau status pekerjaan, dan tunggu sampai masuk RUNNING status:

```
$ watch awsbstat -d
```

Ketika pekerjaan memasuki RUNNING status, kita dapat melihat outputnya. Untuk menampilkan output dari node utama, tambahkan #0 ke id pekerjaan. Untuk menampilkan output dari node komputasi, gunakan #1 dan #2:

```

[ec2-user@ip-10-0-0-111 ~]$ awsbout -s 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#0
2018-11-27 15:50:10: Job id: 5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d#0
2018-11-27 15:50:10: Initializing the environment...
2018-11-27 15:50:10: Starting ssh agents...
2018-11-27 15:50:11: Agent pid 7
2018-11-27 15:50:11: Identity added: /root/.ssh/id_rsa (/root/.ssh/id_rsa)
2018-11-27 15:50:11: Mounting shared file system...
2018-11-27 15:50:11: Generating hostfile...

```

```
2018-11-27 15:50:11: Detected 1/3 compute nodes. Waiting for all compute nodes to
start.
2018-11-27 15:50:26: Detected 1/3 compute nodes. Waiting for all compute nodes to
start.
2018-11-27 15:50:41: Detected 1/3 compute nodes. Waiting for all compute nodes to
start.
2018-11-27 15:50:56: Detected 3/3 compute nodes. Waiting for all compute nodes to
start.
2018-11-27 15:51:11: Starting the job...
download: s3://amzn-s3-demo-bucket/batch/job-submit_mpi_sh-1543333713772.sh to tmp/
batch/job-submit_mpi_sh-1543333713772.sh
2018-11-27 15:51:12: ip container: 10.0.0.180
2018-11-27 15:51:12: ip host: 10.0.0.245
2018-11-27 15:51:12: Compiling...
2018-11-27 15:51:12: Running...
2018-11-27 15:51:12: Hello I'm the main node! I run the mpi job!
2018-11-27 15:51:12: Warning: Permanently added '10.0.0.199' (RSA) to the list of known
hosts.
2018-11-27 15:51:12: Warning: Permanently added '10.0.0.147' (RSA) to the list of known
hosts.
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-180.ec2.internal, rank 1 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-199.ec2.internal, rank 5 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-180.ec2.internal, rank 0 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-199.ec2.internal, rank 4 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-147.ec2.internal, rank 2 out
of 6 processors
2018-11-27 15:51:13: Hello world from processor ip-10-0-0-147.ec2.internal, rank 3 out
of 6 processors

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

```

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

```

Kami sekarang dapat mengonfirmasi bahwa pekerjaan berhasil diselesaikan:

```

[ec2-user@ip-10-0-0-111 ~]$ awsbststat -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  -

```

Catatan: jika Anda ingin menghentikan pekerjaan sebelum berakhir, Anda dapat menggunakan `awsbkill` perintah.

Enkripsi disk dengan Kunci KMS khusus

AWS ParallelCluster mendukung opsi konfigurasi `ebs_kms_key_id` dan `fsx_kms_key_id`. Opsi ini memungkinkan Anda untuk memberikan AWS KMS kunci khusus untuk enkripsi Amazon EBS Disk atau FSx untuk Lustre. Untuk menggunakannya, Anda menentukan `fileec2_iam_role`.

Agar cluster dapat membuat, AWS KMS kunci harus mengetahui nama peran cluster. Ini mencegah Anda menggunakan peran yang dibuat pada pembuatan klaster, yang membutuhkan `kustomec2_iam_role`.

Prasyarat

- AWS ParallelCluster [diinstal](#).
- AWS CLI [itu diinstal dan dikonfigurasi](#).
- Anda memiliki [EC2 key pair](#).
- Anda memiliki peran IAM dengan [izin](#) yang diperlukan untuk menjalankan CLI [pcluster](#).

Menciptakan peran

Pertama, Anda membuat kebijakan:

1. Pergi ke Konsol IAM: <https://console.aws.amazon.com/iam/rumah>.
2. Di bawah Kebijakan, Buat kebijakan, klik tab JSON.
3. Sebagai badan kebijakan, tempel di [Kebijakan Instance](#). Pastikan untuk mengganti semua kejadian **<AWS ACCOUNT ID>** dan **<REGION>**.
4. Beri nama kebijakan `ParallelClusterInstancePolicy`, lalu klik Buat Kebijakan.

Selanjutnya buat peran:

1. Di bawah Peran, buat peran.
2. Klik EC2 sebagai entitas terpercaya.
3. Di bawah Izin, cari `ParallelClusterInstancePolicy` peran yang baru saja Anda buat, dan lampirkan.
4. Beri nama peran `ParallelClusterInstanceRole`, lalu klik Buat Peran.

Berikan izin kunci Anda

Di AWS KMS Konsol > Kunci terkelola pelanggan > klik Alias atau ID Kunci kunci Anda.

Klik tombol Tambah di kotak Pengguna kunci, di bawah tab Kebijakan kunci, dan cari yang baru saja `ParallelClusterInstanceRole` Anda buat. Pasang itu.

Membuat cluster

Sekarang buat cluster. Berikut ini adalah contoh cluster dengan drive terenkripsi `Raid 0`:

```
[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
```

Berikut ini adalah contoh dengan sistem file FSx untuk 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
```

Konfigurasi serupa berlaku untuk Amazon EBS dan sistem file FSx berbasis Amazon.

Beberapa mode antrian tutorial

Menjalankan pekerjaan Anda AWS ParallelCluster dengan beberapa mode antrian

Tutorial ini memandu Anda melalui menjalankan pekerjaan Hello World pertama Anda AWS ParallelCluster dengan [Mode antrian ganda](#).

Prasyarat

- AWS ParallelCluster [diinstal](#).
- AWS CLI [itu diinstal dan dikonfigurasi](#).
- Anda memiliki [EC2 key pair](#).
- Anda memiliki peran IAM dengan [izin](#) yang diperlukan untuk menjalankan CLI [pcluster](#).

Note

Mode antrian ganda hanya didukung untuk AWS ParallelCluster versi 2.9.0 atau yang lebih baru.

Mengkonfigurasi klaster Anda

Pertama, verifikasi AWS ParallelCluster yang diinstal dengan benar dengan menjalankan perintah berikut.

```
$ pcluster version
```

Untuk informasi selengkapnya tentang `pcluster version`, lihat [pcluster version](#).

Perintah ini mengembalikan versi yang sedang berjalan dari AWS ParallelCluster.

Selanjutnya, jalankan `pcluster configure` untuk menghasilkan file konfigurasi dasar. Ikuti semua petunjuk yang mengikuti perintah ini.

```
$ pcluster configure
```

Untuk informasi selengkapnya tentang `pcluster configure` perintah, lihat [pcluster configure](#).

Setelah Anda menyelesaikan langkah ini, Anda harus memiliki file konfigurasi dasar di bawah `~/.parallelcluster/config`. File ini harus berisi konfigurasi cluster dasar dan bagian VPC.

Bagian selanjutnya dari tutorial ini menguraikan cara memodifikasi konfigurasi yang baru Anda buat dan meluncurkan cluster dengan beberapa antrian.

Note

Beberapa contoh yang digunakan dalam tutorial ini tidak memenuhi syarat tingkat bebas.

Untuk tutorial ini, gunakan konfigurasi berikut.

```
[global]
```

```
update_check = true
sanity_check = true
cluster_template = multi-queue

[aws]
aws_region_name = <Your Wilayah 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
```

Membuat cluster Anda

Bagian ini merinci cara membuat cluster mode antrian ganda.

Pertama, beri nama cluster `Andamulti-queue-hello-world`, dan buat cluster sesuai dengan bagian `multi-queue` cluster yang ditentukan di bagian sebelumnya.

```
$ pcluster create multi-queue-hello-world -t multi-queue
```

Untuk informasi selengkapnya tentang `pcluster create`, lihat [pcluster create](#).

Ketika cluster dibuat, output berikut ditampilkan:

```
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
```

Pesan tersebut `CREATE_COMPLETE` menunjukkan bahwa cluster berhasil dibuat. Output juga menyediakan alamat IP publik dan pribadi dari node kepala.

Masuk ke node kepala Anda

Gunakan file kunci SSH pribadi Anda untuk masuk ke node kepala Anda.

```
$ pcluster ssh multi-queue-hello-world -i ~/path/to/keyfile.pem
```

Untuk informasi selengkapnya tentang `pcluster ssh`, lihat [pcluster ssh](#).

Setelah masuk, jalankan `sinfo` perintah untuk memverifikasi bahwa antrian penjadwal Anda sudah diatur dan dikonfigurasi.

Untuk informasi lebih lanjut tentang `sinfo`, lihat [sinfo](#) di Slurm dokumentasi.

```
$ 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

Output menunjukkan bahwa Anda memiliki dua node `t2.micro` komputasi dalam `idle` status yang tersedia di cluster Anda.

Note

- `spot-st-t2micro-1` adalah simpul `st` statis dengan namanya. Node ini selalu tersedia dan sesuai dengan konfigurasi cluster Anda. `min_count` = 1
- `spot-dy-t2micro-1` adalah simpul dinamis `dy` dengan namanya. Node ini saat ini tersedia karena `initial_count` - `min_count` = 1 sesuai dengan konfigurasi cluster Anda. Node ini turun setelah kebiasaan `scaledown_idletime` Anda selama lima menit.

Node lain semuanya dalam keadaan hemat daya, ditunjukkan oleh `~` sufiks dalam keadaan simpul, tanpa EC2 instance yang mendukungnya. Antrian default ditetapkan oleh `*` akhiran setelah nama antreannya, begitu `spot` juga antrian pekerjaan default Anda.

Menjalankan pekerjaan dalam beberapa mode antrian

Selanjutnya, cobalah menjalankan pekerjaan untuk tidur sebentar. Pekerjaan itu nantinya akan menampilkan nama hostnya sendiri. Pastikan skrip ini dapat dijalankan oleh pengguna saat ini.

```
$ 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
```

Kirim pekerjaan menggunakan `sbatch` perintah. Minta dua node untuk pekerjaan ini dengan `-N 2` opsi, dan verifikasi bahwa pekerjaan berhasil dikirim. Untuk informasi selengkapnya tentang `sbatch`, lihat [sbatch](#) dalam dokumentasi Slurm.

```
$ sbatch -N 2 --wrap "srun hellojob.sh"
Submitted batch job 2
```

Anda dapat melihat antrian Anda dan memeriksa status pekerjaan dengan `squeue` perintah. Perhatikan bahwa, karena Anda tidak menentukan antrian tertentu, antrian default (spot) digunakan. Untuk informasi selengkapnya tentang `squeue`, lihat [squeue](#) di Slurmdokumentasi.

```
$ 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
```

Output menunjukkan bahwa pekerjaan saat ini dalam keadaan berjalan. Tunggu 30 detik hingga pekerjaan selesai, lalu jalankan `squeue` lagi.

```
$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	NODELIST(REASON)
-------	-----------	------	------	----	------	-------	------------------

Sekarang pekerjaan dalam antrian telah selesai, cari file output `slurm-2.out` di direktori Anda saat ini.

```
$ cat slurm-2.out
Hello World from spot-dy-t2micro-1
Hello World from spot-st-t2micro-1
```

Outputnya juga menunjukkan bahwa pekerjaan kami berhasil berjalan di `spot-st-t2micro-2` node `spot-st-t2micro-1` dan.

Sekarang kirimkan pekerjaan yang sama dengan menentukan batasan untuk instance tertentu dengan perintah berikut.

```
$ sbatch -N 3 -p spot -C "[c5.xlarge*1&t2.micro*2]" --wrap "srun hellojob.sh"
Submitted batch job 3
```

Anda menggunakan parameter ini untuk `sbatch`.

- `-N 3`— meminta tiga node
- `-p spot`— mengirimkan pekerjaan ke spot antrian. Anda juga dapat mengirimkan pekerjaan ke ondemand antrian dengan menentukan `-p ondemand`.

- `-C "[c5.xlarge*1&t2.micro*2]"`— menentukan kendala node spesifik untuk pekerjaan ini. Ini meminta satu (1) `c5.xlarge` node dan dua (2) `t2.micro` node untuk digunakan untuk pekerjaan ini.

Jalankan `sinfo` perintah untuk melihat node dan antrian. (Antrian dalam disebut AWS ParallelCluster partisi di 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
```

Node menyala. Ini ditandai dengan `#` akhiran pada keadaan simpul. Jalankan `squeue` perintah untuk melihat informasi tentang pekerjaan di 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
```

Pekerjaan Anda ada di `CF` (`CONFIGURING`) state, menunggu instance untuk ditingkatkan dan bergabung dengan cluster.

Setelah sekitar tiga menit, node harus tersedia dan pekerjaan memasuki `R` (`RUNNING`) negara.

```
$ 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
```

Pekerjaan selesai, dan ketiga node berada dalam `idle` keadaan.

```
$ 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			

Kemudian, setelah tidak ada pekerjaan yang tersisa dalam antrian, Anda dapat memeriksa `slurm-3.out` di direktori lokal Anda.

```
$ 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
```

Output juga menunjukkan bahwa pekerjaan berjalan dengan sukses pada node yang sesuai.

Anda dapat mengamati proses penurunan skala. Dalam konfigurasi cluster Anda yang Anda tentukan [scaledown_idletime](#) kustom 5 menit. Setelah lima menit dalam keadaan idle, node dinamis Anda `spot-dy-c5xlarge-1` dan `spot-dy-t2micro-1` secara otomatis menurunkan skala dan masuk ke `POWER_DOWN` mode. Perhatikan bahwa node statis `spot-st-t2micro-1` tidak menurunkan skala.

```
$ 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

Dari kode di atas, Anda dapat melihatnya `spot-dy-c5xlarge-1` dan `spot-dy-t2micro-1` berada dalam `POWER_DOWN` mode. Ini ditunjukkan oleh `%` sufiks. Instance yang sesuai segera dihentikan, tetapi node tetap dalam `POWER_DOWN` status dan tidak tersedia untuk digunakan selama 120 detik (dua menit). Setelah waktu ini, node kembali hemat daya dan tersedia untuk digunakan lagi. Untuk informasi selengkapnya, lihat [Slurmpanduan untuk beberapa mode antrian](#).

Ini harus menjadi keadaan akhir 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
```

Setelah log off dari cluster, Anda dapat membersihkan dengan menjalankan `pcluster delete`. Untuk informasi lebih lanjut, tentang `pcluster list` dan `pcluster delete`, lihat [pcluster list](#) dan [pcluster delete](#).

```
$ pcluster list
multi-queue CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue
Deleting: multi-queue
...
```

Menjalankan pekerjaan di cluster dengan instans EFA dan GPU

Bagian tutorial ini merinci cara memodifikasi konfigurasi dan meluncurkan cluster dengan beberapa antrian yang berisi instance dengan jaringan EFA dan sumber daya GPU. Perhatikan bahwa instance yang digunakan dalam tutorial ini adalah instance dengan harga lebih tinggi.

Periksa batas akun Anda untuk memastikan bahwa Anda berwenang untuk menggunakan instance ini sebelum melanjutkan dengan langkah-langkah yang diuraikan dalam tutorial ini.

Ubah file konfigurasi dengan menggunakan yang berikut ini.

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue-special

[aws]
aws_region_name = <Your Wilayah 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

```

Buat cluster

```
$ pcluster create multi-queue-special -t multi-queue-special
```

Setelah cluster dibuat, gunakan file kunci SSH pribadi Anda untuk masuk ke node kepala Anda.

```
$ pcluster ssh multi-queue-special -i ~/path/to/keyfile.pem
```

Ini harus menjadi keadaan awal 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]

```

Bagian ini menjelaskan cara mengirimkan beberapa pekerjaan untuk memeriksa apakah node memiliki sumber daya EFA atau GPU.

Pertama, tulis skrip pekerjaan. `efa_job.sh` akan tidur selama 30 detik. Setelah itu, cari EFA di output `lspci` perintah. `gpu_job.sh` akan tidur selama 30 detik. Setelah itu, jalankan `nvidia-smi` untuk menampilkan informasi GPU tentang node.

```
$ 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
```

Kirim pekerjaan dengan `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]

Setelah beberapa menit, Anda akan melihat node online dan pekerjaan berjalan.

```
[ec2-user@ip-172-31-15-251 ~]$ sinfo
```

PARTITION	AVAIL	TIMELIMIT	NODES	STATE	NODELIST
efa*	up	infinite	4	idle~	efa-dy-c5n18xlarge-[2-5]
efa*	up	infinite	1	mix	efa-dy-c5n18xlarge-1
gpu	up	infinite	9	idle~	gpu-dy-g38xlarge-[2-10]
gpu	up	infinite	1	mix	gpu-dy-g38xlarge-1

```
[ec2-user@ip-172-31-15-251 ~]$ squeue
```

JOBID	PARTITION	NAME	USER	ST	TIME	NODES	ODELIST(Reason)
4	gpu	wrap	ec2-user	R	0:06	1	gpu-dy-g38xlarge-1
5	efa	wrap	ec2-user	R	0:01	1	efa-dy-

c5n18xlarge-1

Setelah pekerjaan selesai, periksa outputnya. Dari output dalam `slurm-2.out` file, Anda dapat melihat bahwa EFA hadir pada `efa-dy-c5n18xlarge-1` node. Dari output dalam `slurm-3.out` file, Anda dapat melihat `nvidia-smi` output berisi informasi GPU untuk `gpu-dy-g38xlarge-1` node.

```
$ cat slurm-2.out
00:06.0 Ethernet controller: Amazon.com, Inc. Elastic Fabric Adapter (EFA)

$ cat slurm-3.out
Thu Oct 1 22:19:18 2020

+-----+
| NVIDIA-SMI 450.51.05      Driver Version: 450.51.05      CUDA Version: 11.0      |
|-----+-----+-----+
| GPU   Name                Persistence-M| Bus-Id        Disp.A | Volatile Uncorr. ECC |
| Fan  Temp  Perf  Pwr:Usage/Cap|      Memory-Usage | GPU-Util  Compute M. |
|                                           | MIG M.         |
|=====+=====+=====+
|    0   Tesla M60                Off | 00000000:00:1D.0 Off |                    0 |
| N/A   28C    P0     38W / 150W |      0MiB /  7618MiB |      0%      Default |
|                                           | N/A            |
+-----+-----+-----+
|    1   Tesla M60                Off | 00000000:00:1E.0 Off |                    0 |
| N/A   36C    P0     37W / 150W |      0MiB /  7618MiB |     98%      Default |
|                                           | N/A            |
+-----+-----+-----+

+-----+
| Processes:                                     |
|  GPU   GI    CI          PID    Type    Process name                        GPU Memory |
|          ID    ID                                   Usage          |
|=====+=====+=====+
| No running processes found                                     |
+-----+
```

Anda dapat mengamati proses penurunan skala. Dalam konfigurasi cluster, Anda sebelumnya menetapkan kustom [scaledown_idletime](#) lima menit. Akibatnya, setelah lima menit dalam keadaan idle, node dinamis Anda, `spot-dy-c5xlarge-1` dan `spot-dy-t2micro-1`, secara

otomatis menurunkan skala dan masuk ke POWER_DOWN mode. Akhirnya, node memasuki mode hemat daya dan tersedia untuk digunakan lagi.

```
$ 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]
```

Setelah log off dari cluster, Anda dapat membersihkan dengan menjalankan [pcluster delete <cluster name>](#).

```
$ pcluster list
multi-queue-special CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue-special
Deleting: multi-queue-special
...
```

Untuk informasi selengkapnya, lihat [Slurmpanduan untuk beberapa mode antrian](#).

Pengembangan

Anda dapat menggunakan bagian berikut untuk memulai pengembangan AWS ParallelCluster.

Important

Bagian berikut mencakup instruksi untuk menggunakan versi kustom resep buku masak dan paket AWS ParallelCluster node kustom. Informasi ini mencakup metode penyesuaian lanjutan AWS ParallelCluster, dengan potensi masalah yang sulit untuk di-debug. AWS ParallelCluster Tim sangat merekomendasikan penggunaan skrip dalam [Tindakan Bootstrap Kustom](#) untuk penyesuaian, karena kait pasca-instal umumnya lebih mudah untuk di-debug dan lebih portabel di seluruh rilis. AWS ParallelCluster

Topik

- [Menyiapkan AWS ParallelCluster buku masak khusus](#)
- [Menyiapkan paket AWS ParallelCluster node kustom](#)

Menyiapkan AWS ParallelCluster buku masak khusus

Important

Berikut ini adalah petunjuk untuk menggunakan versi kustom resep AWS ParallelCluster buku masak. Ini adalah metode penyesuaian lanjutan AWS ParallelCluster, dengan potensi masalah yang sulit untuk di-debug. AWS ParallelCluster Tim sangat merekomendasikan penggunaan skrip dalam [Tindakan Bootstrap Kustom](#) untuk penyesuaian, karena kait pasca-instal umumnya lebih mudah untuk di-debug dan lebih portabel di seluruh rilis. AWS ParallelCluster

Langkah-langkah

1. Identifikasi direktori kerja AWS ParallelCluster Cookbook tempat Anda telah mengkloning kode [AWS ParallelCluster buku masak](#).

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

2. Deteksi versi AWS ParallelCluster Cookbook saat ini.

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

3. Buat arsip AWS ParallelCluster Cookbook dan hitung md5-nya.

```
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. Buat bucket Amazon S3 dan unggah arsip, md5-nya, dan tanggal modifikasi terakhir ke dalam bucket. Berikan izin yang dapat dibaca publik melalui ACL yang dibaca publik.

```
_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. Tambahkan variabel berikut ke file AWS ParallelCluster konfigurasi, di bawah [\[cluster\]bagian](#).

```
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

Dimulai dengan AWS ParallelCluster versi 2.6.1, sebagian besar resep penginstalan dilewati secara default saat meluncurkan node untuk meningkatkan waktu startup. Untuk melewati sebagian besar resep penginstalan untuk waktu startup yang lebih baik dengan

mengorbankan kompatibilitas mundur, hapus "skip_install_recipes" : "no" dari cluster kunci dalam [extra_json](#) pengaturan.

Menyiapkan paket AWS ParallelCluster node kustom

Warning

Berikut ini adalah petunjuk untuk menggunakan versi kustom dari paket AWS ParallelCluster node. Ini adalah metode penyesuaian lanjutan AWS ParallelCluster, dengan potensi masalah yang sulit untuk di-debug. AWS ParallelCluster Tim sangat merekomendasikan penggunaan skrip dalam [Tindakan Bootstrap Kustom](#) untuk penyesuaian, karena kait pasca-instal umumnya lebih mudah untuk di-debug dan lebih portabel di seluruh rilis. AWS ParallelCluster

Langkah-langkah

1. Identifikasi direktori kerja AWS ParallelCluster node tempat Anda telah mengkloning kode AWS ParallelCluster node.

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

2. Mendeteksi versi AWS ParallelCluster node saat ini.

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

3. Buat arsip AWS ParallelCluster Node.

```
cd "${_nodeDir}"
_stashName=$(git stash create)
git archive --format tar --prefix="aws-parallelcluster-node-${_version}/"
"${_stashName}:-HEAD" | gzip > "aws-parallelcluster-node-${_version}.tgz"
```

4. Buat bucket Amazon S3 dan unggah arsip ke dalam bucket. Berikan izin yang dapat dibaca publik melalui ACL yang dibaca publik.

```
_bucket=<the bucket name>
aws s3 cp --acl public-read aws-parallelcluster-node-${_version}.tgz s3://${_bucket}/
node/aws-parallelcluster-node-${_version}.tgz
```

5. Tambahkan variabel berikut ke file AWS ParallelCluster konfigurasi, di bawah [\[cluster\]bagian](#).

```
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

Dimulai dengan AWS ParallelCluster versi 2.6.1, sebagian besar resep penginstalan dilewati secara default saat meluncurkan node untuk meningkatkan waktu startup. Untuk melewati sebagian besar resep penginstalan untuk waktu startup yang lebih baik dengan mengorbankan kompatibilitas mundur, hapus "skip_install_recipes" : "no" dari cluster kunci dalam [extra_json](#) pengaturan.

AWS ParallelCluster pemecahan masalah

AWS ParallelCluster [Komunitas memelihara halaman Wiki yang menyediakan banyak tips pemecahan masalah di Wiki.AWS ParallelCluster GitHub](#) Untuk daftar masalah yang diketahui, lihat [Masalah yang diketahui](#).

Topik

- [Mengambil dan melestarikan log](#)
- [Memecahkan masalah penyebaran tumpukan](#)
- [Memecahkan masalah di beberapa cluster mode antrian](#)
- [Memecahkan masalah dalam kluster mode antrian tunggal](#)
- [Grup penempatan dan masalah peluncuran instance](#)
- [Direktori yang tidak dapat diganti](#)
- [Memecahkan masalah di Amazon DCV](#)
- [Memecahkan masalah dalam cluster dengan integrasi AWS Batch](#)
- [Pemecahan masalah saat sumber daya gagal dibuat](#)
- [Memecahkan masalah ukuran kebijakan IAM](#)
- [Dukungan tambahan](#)

Mengambil dan melestarikan log

Log adalah sumber daya yang berguna untuk memecahkan masalah. Sebelum Anda dapat menggunakan log untuk memecahkan masalah AWS ParallelCluster sumber daya Anda, Anda harus terlebih dahulu membuat arsip log kluster. Ikuti langkah-langkah yang dijelaskan dalam topik [Membuat Arsip Log Cluster](#) di [AWS ParallelCluster GitHub Wiki](#) untuk memulai proses ini.

Jika salah satu cluster yang sedang berjalan mengalami masalah, Anda harus menempatkan cluster dalam STOPPED status dengan menjalankan `pcluster stop <cluster_name>` perintah sebelum Anda mulai memecahkan masalah. Ini mencegah timbulnya biaya tak terduga.

Jika `pcluster` berhenti berfungsi atau jika Anda ingin menghapus cluster sambil tetap mempertahankan lognya, jalankan `pcluster delete --keep-logs <cluster_name>` perintah. Menjalankan perintah ini menghapus cluster namun mempertahankan grup log yang disimpan di

Amazon. CloudWatch Untuk informasi selengkapnya tentang perintah ini, lihat [pcluster delete](#) dokumentasi.

Memecahkan masalah penyebaran tumpukan

Jika klaster Anda gagal dibuat dan memutar kembali pembuatan tumpukan, Anda dapat melihat file log berikut untuk mendiagnosis masalah. Anda ingin mencari output dari `ROLLBACK_IN_PROGRESS` dalam log ini. Pesan kegagalan akan terlihat seperti berikut:

```
$ 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
```

Untuk mendiagnosis masalah, buat kembali cluster menggunakan [pcluster create](#), termasuk `--norollback` flag. Kemudian, SSH ke dalam cluster:

```
$ pcluster create mycluster --norollback
...
$ pcluster ssh mycluster
```

Setelah Anda masuk ke node kepala, Anda harus menemukan tiga file log utama yang dapat Anda gunakan untuk menentukan kesalahan.

- `/var/log/cfn-init.log` adalah log untuk `cfn-init` skrip. Pertama periksa log ini. Anda mungkin melihat kesalahan seperti `Command chef failed` di log ini. Lihatlah baris segera sebelum baris ini untuk lebih spesifik yang terkait dengan pesan kesalahan. Untuk informasi lebih lanjut, lihat [cfn-init](#).
- `/var/log/cloud-init.log` adalah log untuk [cloud-init](#). Jika Anda tidak melihat apa pun `cfn-init.log`, coba periksa log ini selanjutnya.
- `/var/log/cloud-init-output.log` adalah output dari perintah yang dijalankan oleh [cloud-init](#). Ini termasuk output dari `cfn-init`. Dalam kebanyakan kasus, Anda tidak perlu melihat log ini untuk memecahkan masalah jenis ini.

Memecahkan masalah di beberapa cluster mode antrian

Bagian ini relevan dengan cluster yang diinstal menggunakan AWS ParallelCluster versi 2.9.0 dan yang lebih baru dengan Slurm penjadwal pekerjaan. Untuk informasi selengkapnya tentang beberapa mode antrian, lihat [Mode antrian ganda](#).

Topik

- [Log kunci](#)
- [Memecahkan masalah inisialisasi node](#)
- [Memecahkan masalah penggantian dan penghentian node yang tidak terduga](#)
- [Mengganti, mengakhiri, atau mematikan instance dan node masalah](#)
- [Memecahkan masalah node dan pekerjaan lain yang diketahui](#)

Log kunci

Tabel berikut memberikan ikhtisar log kunci untuk node kepala:

`/var/log/cfn-init.log`

Ini adalah log AWS CloudFormation init. Ini berisi semua perintah yang dijalankan ketika sebuah instance diatur. Ini berguna untuk memecahkan masalah inisialisasi.

`/var/log/chef-client.log`

Ini adalah log klien Chef. Ini berisi semua perintah yang dijalankan melalui chef/CINC. Ini berguna untuk memecahkan masalah inisialisasi.

`/var/log/parallelcluster/slurm_resume.log`

Ini adalah ResumeProgram log. Ini meluncurkan instance untuk node dinamis dan berguna untuk memecahkan masalah peluncuran node dinamis.

`/var/log/parallelcluster/slurm_suspend.log`

Ini SuspendProgram log. Ini disebut ketika instance dihentikan untuk node dinamis, dan berguna untuk memecahkan masalah penghentian node dinamis. Saat Anda memeriksa log ini, Anda juga harus memeriksa `clustermgtd` log.

`/var/log/parallelcluster/clustermgtd`

Ini `clustermgtd` log. Ini berjalan sebagai daemon terpusat yang mengelola sebagian besar tindakan operasi cluster. Ini berguna untuk memecahkan masalah peluncuran, penghentian, atau masalah operasi klaster apa pun.

`/var/log/slurmctld.log`

Ini adalah Slurm kontrol log daemon. AWS ParallelCluster tidak membuat keputusan penskalaan. Sebaliknya, ia hanya mencoba meluncurkan sumber daya untuk memuaskan Slurm persyaratan. Ini berguna untuk masalah penskalaan dan alokasi, masalah terkait pekerjaan, dan masalah peluncuran dan penghentian terkait penjadwal.

Ini adalah catatan kunci untuk node Compute:

`/var/log/cloud-init-output.log`

Ini adalah log [cloud-init](#). Ini berisi semua perintah yang dijalankan ketika sebuah instance diatur. Ini berguna untuk memecahkan masalah inisialisasi.

`/var/log/parallelcluster/computemgtd`

Ini `computemgtd` log. Ini berjalan pada setiap node komputasi untuk memantau node dalam peristiwa langka bahwa `clustermgtd` daemon pada node kepala sedang offline. Ini berguna untuk memecahkan masalah penghentian yang tidak terduga.

`/var/log/slurmd.log`

Ini adalah Slurm menghitung log daemon. Ini berguna untuk memecahkan masalah inisialisasi dan masalah terkait kegagalan komputasi.

Memecahkan masalah inisialisasi node

Bagian ini mencakup bagaimana Anda dapat memecahkan masalah inisialisasi node. Ini termasuk masalah di mana node gagal diluncurkan, dinyalakan, atau bergabung dengan cluster.

Simpul kepala:

Log yang berlaku:

- `/var/log/cfn-init.log`

- `/var/log/chef-client.log`
- `/var/log/parallelcluster/clustermgtd`
- `/var/log/parallelcluster/slurm_resume.log`
- `/var/log/slurmctld.log`

Periksa `/var/log/cfn-init.log` dan `/var/log/chef-client.log` log. Log ini harus berisi semua tindakan yang dijalankan saat node kepala diatur. Sebagian besar kesalahan yang terjadi selama pengaturan harus memiliki pesan kesalahan yang terletak di `/var/log/chef-client.log` log. Jika skrip pra-instal atau pasca-instal ditentukan dalam konfigurasi cluster, periksa kembali apakah skrip berjalan dengan sukses melalui pesan log.

Ketika sebuah cluster dibuat, node kepala harus menunggu node komputasi untuk bergabung dengan cluster sebelum dapat bergabung dengan cluster. Dengan demikian, jika node komputasi gagal bergabung dengan cluster, maka node kepala juga gagal. Anda dapat mengikuti salah satu dari rangkaian prosedur ini, tergantung pada jenis catatan komputasi yang Anda gunakan, untuk memecahkan masalah jenis ini:

Node komputasi dinamis:

- Cari ResumeProgram log (`/var/log/parallelcluster/slurm_resume.log`) untuk nama node komputasi Anda untuk melihat apakah pernah ResumeProgram dipanggil dengan node. (Jika ResumeProgram tidak pernah dipanggil, Anda dapat memeriksa `slurmctld` log (`/var/log/slurmctld.log`) untuk menentukan apakah Slurm pernah mencoba menelepon ResumeProgram dengan node.)
- Perhatikan bahwa izin yang salah untuk ResumeProgram dapat menyebabkan kegagalan ResumeProgram secara diam-diam. Jika Anda menggunakan AMI kustom dengan modifikasi untuk ResumeProgram penyiapan, periksa apakah AMI dimiliki oleh `slurm` pengguna dan memiliki izin `744 (rwxr--r--)`. ResumeProgram
- Jika ResumeProgram dipanggil, periksa untuk melihat apakah sebuah instance diluncurkan untuk node. Jika tidak ada instance yang diluncurkan, Anda seharusnya dapat melihat pesan kesalahan yang menjelaskan kegagalan peluncuran.
- Jika instance diluncurkan, maka mungkin ada masalah selama proses penyiapan. Anda akan melihat alamat IP pribadi dan ID instance yang sesuai dari ResumeProgram log. Selain itu, Anda dapat melihat log pengaturan yang sesuai untuk contoh tertentu. Untuk informasi selengkapnya tentang pemecahan masalah kesalahan penyiapan dengan node komputasi, lihat bagian selanjutnya.

Node komputasi statis:

- Periksa log `clustermgtd (/var/log/parallelcluster/clustermgtd)` untuk melihat apakah instance diluncurkan untuk node. Jika tidak diluncurkan, harus ada pesan kesalahan yang jelas yang merinci kegagalan peluncuran.
- Jika instance diluncurkan, ada beberapa masalah selama proses penyiapan. Anda akan melihat alamat IP pribadi dan ID instance yang sesuai dari `ResumeProgram` log. Selain itu, Anda dapat melihat log pengaturan yang sesuai untuk instance tertentu.
- Hitung node:
 - Log yang berlaku:
 - `/var/log/cloud-init-output.log`
 - `/var/log/slurmd.log`
 - Jika node komputasi diluncurkan, periksa terlebih dahulu `/var/log/cloud-init-output.log`, yang harus berisi log pengaturan yang mirip dengan `/var/log/chef-client.log` log pada node kepala. Sebagian besar kesalahan yang terjadi selama pengaturan harus memiliki pesan kesalahan yang terletak di `/var/log/cloud-init-output.log` log. Jika skrip pra-instal atau pasca-instal ditentukan dalam konfigurasi cluster, periksa apakah skrip tersebut berhasil dijalankan.
 - Jika Anda menggunakan AML kustom dengan modifikasi Slurm konfigurasi, maka mungkin ada Slurm kesalahan terkait yang mencegah node komputasi bergabung dengan cluster. Untuk kesalahan terkait penjadwal, periksa `/var/log/slurmd.log` log.

Memecahkan masalah penggantian dan penghentian node yang tidak terduga

Bagian ini terus mengeksplorasi bagaimana Anda dapat memecahkan masalah terkait node, khususnya ketika node diganti atau dihentikan secara tidak terduga.

- Log yang berlaku:
 - `/var/log/parallelcluster/clustermgtd`(simpul kepala)
 - `/var/log/slurmctld.log`(simpul kepala)
 - `/var/log/parallelcluster/computemgtd`(simpul komputasi)
- Node diganti atau dihentikan secara tak terduga

- Periksa `clustermgtd` log (`/var/log/parallelcluster/clustermgtd`) untuk melihat apakah `clustermgtd` mengambil tindakan untuk mengganti atau mengakhiri node. Perhatikan bahwa `clustermgtd` menangani semua tindakan pemeliharaan node normal.
- Jika `clustermgtd` diganti atau dihentikan node, harus ada pesan yang merinci mengapa tindakan ini diambil pada node. Jika alasannya terkait penjadwal (misalnya, karena node masukDOWN), periksa `slurmctld` log untuk informasi lebih lanjut. Jika alasannya EC2 terkait Amazon, harus ada pesan informatif yang merinci masalah EC2 terkait Amazon yang memerlukan penggantian.
- Jika `clustermgtd` tidak menghentikan node, periksa terlebih dahulu apakah ini adalah penghentian yang diharapkan oleh Amazon EC2, lebih khusus lagi penghentian spot. `computemgtd`, berjalan pada node Compute, juga dapat mengambil tindakan untuk mengakhiri node jika `clustermgtd` ditentukan sebagai tidak sehat. Periksa `computemgtd` log (`/var/log/parallelcluster/computemgtd`) untuk melihat apakah node `computemgtd` dihentikan.
- Node gagal
 - Periksa `slurmctld` log (`/var/log/slurmctld.log`) untuk melihat mengapa pekerjaan atau node gagal. Perhatikan bahwa pekerjaan secara otomatis diantrian ulang jika node gagal.
 - Jika `slurm_resume` melaporkan bahwa node diluncurkan dan `clustermgtd` melaporkan setelah beberapa menit bahwa tidak ada instance yang sesuai di Amazon EC2 untuk node itu, node mungkin gagal selama penyiapan. Untuk mengambil log dari compute (`/var/log/cloud-init-output.log`), lakukan langkah-langkah berikut:
 - Kirim pekerjaan untuk membiarkan Slurm memutar node baru.
 - Setelah node dimulai, aktifkan perlindungan terminasi menggunakan perintah ini.

```
aws ec2 modify-instance-attribute --instance-id i-xyz --disable-api-termination
```

- Ambil output konsol dari node dengan perintah ini.

```
aws ec2 get-console-output --instance-id i-xyz --output text
```

Mengganti, mengakhiri, atau mematikan instance dan node masalah

- Log yang berlaku:
 - `/var/log/parallelcluster/clustermgtd`(simpul kepala)
 - `/var/log/parallelcluster/slurm_suspend.log`(simpul kepala)

- Dalam kebanyakan kasus, `clustermgtd` menangani semua tindakan penghentian instance yang diharapkan. Periksa di `clustermgtd` log untuk melihat mengapa gagal mengganti atau mengakhiri node.
- Untuk node dinamis gagal [scaledown_idletime](#), periksa `SuspendProgram` log untuk melihat apakah `SuspendProgram` dipanggil `slurmctld` dengan node tertentu sebagai argumen. Perhatikan bahwa `SuspendProgram` tidak benar-benar melakukan tindakan apa pun. Sebaliknya, itu hanya log ketika dipanggil. Semua penghentian dan `NodeAddr` reset instance dilakukan oleh `clustermgtd`. Slurm menempatkan node kembali ke `POWER_SAVING` keadaan setelah `SuspendTimeout` secara otomatis.

Memecahkan masalah node dan pekerjaan lain yang diketahui

Jenis lain dari masalah yang diketahui adalah yang AWS ParallelCluster mungkin gagal mengalokasikan pekerjaan atau membuat keputusan penskalaan. Dengan jenis masalah ini, AWS ParallelCluster hanya meluncurkan, mengakhiri, atau memelihara sumber daya menurut Slurm instruksi. Untuk masalah ini, periksa `slurmctld` log untuk memecahkan masalah ini.

Memecahkan masalah dalam kluster mode antrian tunggal

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Bagian ini berlaku untuk cluster yang tidak memiliki beberapa mode antrian dengan salah satu dari dua konfigurasi berikut:

- Diluncurkan menggunakan AWS ParallelCluster versi lebih awal dari 2.9.0 dan SGE, Torque, atau Slurm penjadwal pekerjaan.
- Diluncurkan menggunakan AWS ParallelCluster versi 2.9.0 atau yang lebih baru dan SGE atau Torque penjadwal pekerjaan.

Topik

- [Log kunci](#)
- [Pemecahan masalah gagal meluncurkan dan bergabung dengan operasi](#)

- [Memecahkan masalah penskalaan](#)
- [Memecahkan masalah terkait kluster lainnya](#)

Log kunci

File log berikut adalah log kunci untuk node kepala.

Untuk AWS ParallelCluster versi 2.9.0 atau yang lebih baru:

`/var/log/chef-client.log`

Ini adalah log klien CINC (koki). Ini berisi semua perintah yang dijalankan melalui CINC. Ini berguna untuk memecahkan masalah inisialisasi.

Untuk semua AWS ParallelCluster versi:

`/var/log/cfn-init.log`

Ini `cfn-init` log. Ini berisi semua perintah yang dijalankan ketika sebuah instance disiapkan, dan oleh karena itu berguna untuk memecahkan masalah inisialisasi. Untuk informasi lebih lanjut, lihat [cfn-init](#).

`/var/log/clustermgtd.log`

Ini adalah `clustermgtd` log untuk Slurm penjadwal. `clustermgtd` berjalan sebagai daemon terpusat yang mengelola sebagian besar tindakan operasi cluster. Ini berguna untuk memecahkan masalah peluncuran, penghentian, atau masalah operasi kluster apa pun.

`/var/log/jobwatcher`

Ini adalah `jobwatcher` log untuk SGE and Torque penjadwal. `jobwatcher` memantau antrian penjadwal dan memperbarui Grup Auto Scaling. Ini berguna untuk memecahkan masalah yang terkait dengan penskalaan node.

`/var/log/sqswatcher`

Ini adalah `sqswatcher` log untuk SGE and Torque penjadwal. `sqswatcher` memproses peristiwa siap instance yang dikirim oleh instance komputasi setelah inisialisasi berhasil. Ini juga menambahkan node komputasi ke konfigurasi penjadwal. Log ini berguna untuk memecahkan masalah mengapa node atau node gagal bergabung dengan cluster.

Berikut ini adalah log kunci untuk node komputasi.

AWS ParallelCluster versi 2.9.0 atau yang lebih baru

`/var/log/cloud-init-output.log`

Ini adalah log init Cloud. Ini berisi semua perintah yang dijalankan ketika sebuah instance diatur. Ini berguna untuk memecahkan masalah inisialisasi.

AWS ParallelCluster versi sebelum 2.9.0

`/var/log/cfn-init.log`

Ini adalah log CloudFormation init. Ini berisi semua perintah yang dijalankan ketika sebuah instance diatur. Ini berguna untuk memecahkan masalah inisialisasi

Semua versi

`/var/log/nodewatcher`

Ini nodewatcher log. nodewatcherdaemon yang berjalan di setiap node Compute saat menggunakan SGE and Torque penjadwal. Mereka menurunkan node jika itu menganggur. Log ini berguna untuk masalah apa pun yang terkait dengan mengurangi sumber daya.

Pemecahan masalah gagal meluncurkan dan bergabung dengan operasi

- Log yang berlaku:
 - `/var/log/cfn-init-cmd.log`(simpul kepala dan simpul komputasi)
 - `/var/log/sqswatcher`(simpul kepala)
- Jika node gagal diluncurkan, periksa `/var/log/cfn-init-cmd.log` log untuk melihat pesan kesalahan tertentu. Dalam kebanyakan kasus, kegagalan peluncuran node disebabkan oleh kegagalan pengaturan.
- Jika node komputasi gagal bergabung dengan konfigurasi penjadwal meskipun penyiapan berhasil, periksa `/var/log/sqswatcher` log untuk melihat apakah peristiwa sqswatcher diproses. Masalah-masalah ini dalam banyak kasus adalah karena sqswatcher tidak memproses acara.

Memecahkan masalah penskalaan

- Log yang berlaku:
 - `/var/log/jobwatcher(simpul kepala)`
 - `/var/log/nodewatcher(simpul komputasi)`
- Masalah skala: Untuk node kepala, periksa `/var/log/jobwatcher` log untuk melihat apakah `jobwatcher` daemon menghitung jumlah node yang diperlukan dan memperbarui Grup Auto Scaling. Perhatikan bahwa `jobwatcher` memantau antrian penjadwal dan memperbarui Grup Auto Scaling.
- Perkecil masalah: Untuk node komputasi, periksa `/var/log/nodewatcher` log pada node masalah untuk melihat mengapa node diperkecil. Perhatikan bahwa `nodewatcher` daemon menurunkan skala node komputasi jika idle.

Memecahkan masalah terkait klaster lainnya

Satu masalah yang diketahui adalah catatan komputasi acak gagal pada cluster skala besar, khususnya yang memiliki 500 atau lebih node komputasi. Masalah ini terkait dengan batasan arsitektur penskalaan cluster antrian tunggal. Jika Anda ingin menggunakan cluster skala besar, menggunakan AWS ParallelCluster versi v2.9.0 atau yang lebih baru, sedang menggunakan Slurm, dan ingin menghindari masalah ini, Anda harus memutakhirkan dan beralih ke cluster yang didukung mode antrian ganda. Anda dapat melakukannya dengan berlari [pcluster-config convert](#).

Untuk ultra-large-scale cluster, penyetelan tambahan ke sistem Anda mungkin diperlukan. Untuk informasi lebih lanjut, hubungi Dukungan.

Grup penempatan dan masalah peluncuran instance

Untuk mendapatkan latensi antar simpul terendah, gunakan grup penempatan. Grup penempatan menjamin bahwa instans Anda berada di tulang punggung jaringan yang sama. Jika tidak ada cukup instance yang tersedia saat permintaan dibuat, `InsufficientInstanceCapacity` kesalahan akan dikembalikan. Untuk mengurangi kemungkinan menerima kesalahan ini saat menggunakan grup penempatan cluster, atur [placement_group](#) parameter ke DYNAMIC dan atur [placement](#) parameter ke `compute`.

[Jika Anda memerlukan sistem file bersama berkinerja tinggi, pertimbangkan untuk menggunakan FSx Lustre.](#)

Jika node kepala harus berada dalam grup penempatan, gunakan jenis instance dan subnet yang sama untuk head serta semua node komputasi. Dengan melakukan ini, [compute_instance_type](#) parameter memiliki nilai yang sama dengan [master_instance_type](#) parameter, [placement](#) parameter diatur ke `cluster`, dan [compute_subnet_id](#) parameter tidak ditentukan. Dengan konfigurasi ini, nilai [master_subnet_id](#) parameter digunakan untuk node komputasi.

Untuk informasi selengkapnya, lihat [Memecahkan masalah peluncuran instans](#) serta [Peran dan batasan grup penempatan](#) di Panduan Pengguna Amazon EC2

Direktori yang tidak dapat diganti

Direktori berikut dibagi antara node dan tidak dapat diganti.

`/home`

Ini termasuk folder home pengguna default (`/home/ec2_user` di Amazon Linux, `/home/centos` di CentOS, dan `/home/ubuntu` pada Ubuntu).

`/opt/intel`

Ini termasuk Intel MPI, Intel Parallel Studio, dan file terkait.

`/opt/sge`

Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Ini termasuk Son of Grid Engine dan file terkait. (Bersyarat, hanya jika [scheduler](#) = `sge`.)

`/opt/slurm`

Ini termasuk Slurm Workload Manager dan file terkait. (Bersyarat, hanya jika [scheduler](#) = `slurm`.)

/opt/torque

 Note

Dimulai dengan versi 2.11.5, AWS ParallelCluster tidak mendukung penggunaan SGE atau Torque penjadwal.

Ini termasuk Torque Resource Manager dan file terkait. (Bersyarat, hanya jika [scheduler](#) = torque.)

Memecahkan masalah di Amazon DCV

Topik

- [Log untuk Amazon DCV](#)
- [Memori jenis instans Amazon DCV](#)
- [Masalah Ubuntu Amazon DCV](#)

Log untuk Amazon DCV

Log untuk Amazon DCV ditulis ke file di `/var/log/dcv/` direktori. Meninjau log ini dapat membantu memecahkan masalah.

Memori jenis instans Amazon DCV

Jenis instans harus memiliki setidaknya 1,7 gibibyte (GiB) RAM untuk menjalankan Amazon DCV. Nano and micro tipe instance tidak memiliki cukup memori untuk menjalankan Amazon DCV.

Masalah Ubuntu Amazon DCV

Saat menjalankan Terminal Gnome melalui sesi DCV di Ubuntu, Anda mungkin tidak secara otomatis memiliki akses ke lingkungan pengguna yang AWS ParallelCluster tersedia melalui shell login. Lingkungan pengguna menyediakan modul lingkungan seperti `openmpi` atau `intelmpi`, dan pengaturan pengguna lainnya.

Pengaturan default Terminal Gnome mencegah shell dimulai sebagai shell login. Ini berarti bahwa profil shell tidak bersumber secara otomatis dan lingkungan AWS ParallelCluster pengguna tidak dimuat.

Untuk mendapatkan sumber profil shell dengan benar dan mengakses lingkungan AWS ParallelCluster pengguna, lakukan salah satu hal berikut:

- Ubah pengaturan terminal default:
 1. Pilih menu Edit di terminal Gnome.
 2. Pilih Preferensi, lalu Profil.
 3. Pilih Command dan pilih Run Command sebagai shell login.
 4. Buka terminal baru.
- Gunakan baris perintah untuk sumber profil yang tersedia:

```
$ source /etc/profile && source $HOME/.bashrc
```

Memecahkan masalah dalam cluster dengan integrasi AWS Batch

Bagian ini relevan dengan cluster dengan integrasi AWS Batch scheduler.

Masalah simpul kepala

Masalah penyiapan terkait node kepala dapat dipecahkan masalah dengan cara yang sama seperti cluster antrian tunggal. Untuk informasi lebih lanjut tentang masalah ini, lihat [Memecahkan masalah dalam kluster mode antrian tunggal](#).

AWS Batch masalah pengiriman pekerjaan paralel multi-node

Jika Anda memiliki masalah dalam mengirimkan pekerjaan paralel multi-node saat AWS Batch menggunakan sebagai penjadwal pekerjaan, Anda harus meningkatkan AWS ParallelCluster ke versi 2.5.0. Jika itu tidak layak, Anda dapat menggunakan solusi yang dirinci dalam topik: [Menambal sendiri cluster yang digunakan untuk mengirimkan pekerjaan](#) paralel multi-node. AWS Batch

Masalah komputasi

AWS Batch mengelola aspek penskalaan dan komputasi layanan Anda. Jika Anda mengalami masalah terkait komputasi, lihat dokumentasi AWS Batch [pemecahan masalah](#) untuk mendapatkan bantuan.

Kegagalan Job

Jika pekerjaan gagal, Anda dapat menjalankan [awsboud](#) perintah untuk mengambil output pekerjaan. Anda juga dapat menjalankan [awsbstat](#) -d perintah untuk mendapatkan tautan ke log pekerjaan yang disimpan oleh Amazon CloudWatch.

Pemecahan masalah saat sumber daya gagal dibuat

Bagian ini relevan dengan sumber daya cluster ketika mereka gagal untuk membuat.

Ketika sumber daya gagal untuk membuat, ParallelCluster mengembalikan pesan kesalahan seperti berikut ini.

```
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)
}
```

Sebagai contoh, jika Anda melihat pesan status yang ditampilkan dalam respons perintah sebelumnya, Anda harus menggunakan jenis instance yang tidak akan melebihi batas vCPU Anda saat ini atau meminta lebih banyak kapasitas vCPU.

Anda juga dapat menggunakan CloudFormation konsol untuk melihat informasi tentang "Cluster creation failed" status.

Lihat pesan CloudFormation kesalahan dari konsol.

1. Masuk ke AWS Management Console dan navigasikan ke <https://console.aws.amazon.com/cloudformation>.
2. Pilih tumpukan bernama parallelcluster-. *cluster_name*
3. Pilih tab Acara.
4. Periksa Status sumber daya yang gagal dibuat dengan menggulir daftar peristiwa sumber daya berdasarkan ID Logis. Jika subtask gagal dibuat, kerjakan mundur untuk menemukan peristiwa sumber daya yang gagal.
5. Contoh pesan AWS CloudFormation kesalahan:

```
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].
```

Memecahkan masalah ukuran kebijakan IAM

Lihat [IAM dan AWS STS kuota, persyaratan nama, dan batas karakter](#) untuk memverifikasi kuota pada kebijakan terkelola yang dilampirkan pada peran. Jika ukuran kebijakan terkelola melebihi

kuota, bagi kebijakan menjadi dua atau lebih kebijakan. Jika Anda melebihi kuota pada jumlah kebijakan yang dilampirkan pada peran IAM, buat peran tambahan dan distribusikan kebijakan di antara mereka untuk memenuhi kuota.

Dukungan tambahan

Untuk daftar masalah yang diketahui, lihat halaman [GitHubWiki](#) utama atau halaman [masalah](#). Untuk masalah yang lebih mendesak, hubungi Dukungan atau buka [GitHubmasalah baru](#).

AWS ParallelCluster kebijakan dukungan

AWS ParallelCluster mendukung beberapa rilis pada saat yang sama. Setiap AWS ParallelCluster rilis memiliki tanggal End of Support Life (EOSL) yang dijadwalkan. Setelah tanggal EOSL, tidak ada dukungan atau pemeliharaan lebih lanjut yang diberikan untuk rilis tersebut.

AWS ParallelCluster menggunakan skema `major.minor.patch` versi. Fitur baru, peningkatan kinerja, pembaruan keamanan, dan perbaikan bug disertakan dalam rilis versi minor baru untuk rilis versi utama terbaru. Versi minor kompatibel ke belakang dalam versi utama. Untuk masalah kritis, AWS menyediakan perbaikan melalui rilis patch, tetapi hanya untuk versi minor terbaru dari rilis yang belum mencapai EOSL. Jika Anda ingin menggunakan pembaruan dari rilis versi baru, Anda perlu meningkatkan ke versi minor atau patch yang baru.

AWS ParallelCluster versi	Tanggal akhir masa pakai yang didukung (EOSL)
2.10.4 dan sebelumnya	12/31/2021
2.11. x	12/31/2022

Keamanan di AWS ParallelCluster

Keamanan cloud di AWS adalah prioritas tertinggi. Sebagai AWS pelanggan, Anda mendapat manfaat dari pusat data dan arsitektur jaringan yang dibangun untuk memenuhi persyaratan organisasi yang paling sensitif terhadap keamanan.

Keamanan adalah tanggung jawab bersama antara Anda AWS dan Anda. [Model tanggung jawab bersama](#) model menjelaskan hal ini sebagai keamanan dari cloud dan keamanan dalam cloud:

- Keamanan cloud — AWS bertanggung jawab untuk melindungi infrastruktur yang menjalankan AWS layanan di AWS Cloud. AWS juga memberi Anda layanan yang dapat Anda gunakan dengan aman. Auditor pihak ketiga secara teratur menguji dan memverifikasi efektivitas keamanan kami sebagai bagian dari [Program AWS Kepatuhan](#) . Untuk mempelajari tentang program kepatuhan yang berlaku AWS ParallelCluster, lihat [AWS Layanan dalam Lingkup oleh AWS Layanan Program Kepatuhan](#) .
- Keamanan di cloud — Tanggung jawab Anda ditentukan oleh AWS layanan atau layanan tertentu yang Anda gunakan. Anda juga bertanggung jawab atas beberapa faktor terkait lainnya termasuk sensitivitas data Anda, persyaratan perusahaan Anda, dan hukum dan peraturan yang berlaku.

Dokumentasi ini menjelaskan bagaimana Anda harus menerapkan model tanggung jawab bersama saat menggunakan AWS ParallelCluster. Topik berikut menunjukkan cara mengonfigurasi AWS ParallelCluster untuk memenuhi tujuan keamanan dan kepatuhan Anda. Anda juga belajar cara menggunakan dengan AWS ParallelCluster cara yang membantu Anda memantau dan mengamankan AWS sumber daya Anda.

Topik

- [Informasi keamanan untuk layanan yang digunakan oleh AWS ParallelCluster](#)
- [Perlindungan data di AWS ParallelCluster](#)
- [Identity and Access Management untuk AWS ParallelCluster](#)
- [Validasi kepatuhan untuk AWS ParallelCluster](#)
- [Menegakkan Versi Minimum TLS 1.2](#)

Informasi keamanan untuk layanan yang digunakan oleh AWS ParallelCluster

- [Keamanan di Amazon EC2](#)
- [Keamanan di Amazon API Gateway](#)
- [Keamanan di AWS Batch](#)
- [Keamanan di AWS CloudFormation](#)
- [Keamanan di Amazon CloudWatch](#)
- [Keamanan di AWS CodeBuild](#)
- [Keamanan di Amazon DynamoDB](#)
- [Keamanan di Amazon ECR](#)
- [Keamanan di Amazon ECS](#)
- [Keamanan di Amazon EFS](#)
- [Keamanan FSx untuk Lustre](#)
- [Keamanan di AWS Identity and Access Management \(IAM\)](#)
- [Keamanan di EC2 Image Builder](#)
- [Keamanan di AWS Lambda](#)
- [Keamanan di Amazon Route 53](#)
- [Keamanan di Amazon SNS](#)
- [Keamanan di Amazon SQS \(Untuk AWS ParallelCluster versi 2.x.\)](#)
- [Keamanan di Amazon S3](#)
- [Keamanan di Amazon VPC](#)

Perlindungan data di AWS ParallelCluster

[Model tanggung jawab AWS bersama model](#) berlaku untuk perlindungan data di AWS ParallelCluster. Seperti yang dijelaskan dalam model AWS ini, bertanggung jawab untuk melindungi infrastruktur global yang menjalankan semua AWS Cloud. Anda bertanggung jawab untuk mempertahankan kendali atas konten yang di-host pada infrastruktur ini. Anda juga bertanggung jawab atas tugas-tugas konfigurasi dan manajemen keamanan untuk Layanan AWS yang Anda gunakan. Lihat

informasi yang lebih lengkap tentang privasi data dalam [Pertanyaan Umum Privasi Data](#). Lihat informasi tentang perlindungan data di Eropa di pos blog [Model Tanggung Jawab Bersama dan GDPR AWS](#) di Blog Keamanan AWS .

Untuk tujuan perlindungan data, kami menyarankan Anda melindungi Akun AWS kredensial dan mengatur pengguna individu dengan AWS IAM Identity Center atau AWS Identity and Access Management (IAM). Dengan cara itu, setiap pengguna hanya diberi izin yang diperlukan untuk memenuhi tanggung jawab tugasnya. Kami juga menyarankan supaya Anda mengamankan data dengan cara-cara berikut:

- Gunakan autentikasi multi-faktor (MFA) pada setiap akun.
- Gunakan SSL/TLS untuk berkomunikasi dengan sumber daya. AWS Kami mensyaratkan TLS 1.2 dan menganjurkan TLS 1.3.
- Siapkan API dan pencatatan aktivitas pengguna dengan AWS CloudTrail. Untuk informasi tentang penggunaan CloudTrail jejak untuk menangkap AWS aktivitas, lihat [Bekerja dengan CloudTrail jejak](#) di AWS CloudTrail Panduan Pengguna.
- Gunakan solusi AWS enkripsi, bersama dengan semua kontrol keamanan default di dalamnya Layanan AWS.
- Gunakan layanan keamanan terkelola tingkat lanjut seperti Amazon Macie, yang membantu menemukan dan mengamankan data sensitif yang disimpan di Amazon S3.
- Jika Anda memerlukan modul kriptografi tervalidasi FIPS 140-3 saat mengakses AWS melalui antarmuka baris perintah atau API, gunakan titik akhir FIPS. Lihat informasi selengkapnya tentang titik akhir FIPS yang tersedia di [Standar Pemrosesan Informasi Federal \(FIPS\) 140-3](#).

Kami sangat merekomendasikan agar Anda tidak pernah memasukkan informasi identifikasi yang sensitif, seperti nomor rekening pelanggan Anda, ke dalam tanda atau bidang isian bebas seperti bidang Nama. Ini termasuk saat Anda bekerja dengan AWS ParallelCluster atau lainnya Layanan AWS menggunakan konsol, API AWS CLI, atau AWS SDKs. Data apa pun yang Anda masukkan ke dalam tanda atau bidang isian bebas yang digunakan untuk nama dapat digunakan untuk log penagihan atau log diagnostik. Saat Anda memberikan URL ke server eksternal, kami sangat menganjurkan supaya Anda tidak menyertakan informasi kredensial di dalam URL untuk memvalidasi permintaan Anda ke server itu.

Enkripsi data

Fitur utama dari setiap layanan aman adalah bahwa informasi dienkripsi ketika tidak aktif digunakan.

Enkripsi diam

AWS ParallelCluster tidak menyimpan data pelanggan selain kredensial yang dibutuhkan untuk berinteraksi dengan AWS layanan atas nama pengguna.

Untuk data pada node di cluster, data dapat dienkripsi saat istirahat.

Untuk volume Amazon EBS, enkripsi dikonfigurasi menggunakan [ebs_kms_key_id](#) pengaturan di [\[ebs\]bagian](#) untuk AWS ParallelCluster versi 2.x.) Untuk informasi selengkapnya, lihat [enkripsi Amazon EBS](#) di Panduan EC2 Pengguna Amazon.

Untuk volume Amazon EFS, enkripsi dikonfigurasi menggunakan [encrypted](#) dan [efs_kms_key_id](#) pengaturan di [\[efs\]bagian](#) dalam AWS ParallelCluster versi 2.x). Untuk informasi selengkapnya, lihat [Cara kerja enkripsi saat istirahat](#) di Panduan Pengguna Amazon Elastic File System.

FSx Untuk sistem file Lustre, enkripsi data saat istirahat diaktifkan secara otomatis saat membuat sistem FSx file Amazon. Untuk informasi selengkapnya, lihat [Mengekripsi data saat istirahat di Amazon FSx for Lustre User Guide](#).

Misalnya tipe dengan NVMe volume, data pada volume penyimpanan NVMe instance dienkripsi menggunakan cipher XTS-AES-256 yang diimplementasikan pada modul perangkat keras pada instance. Kunci enkripsi dihasilkan menggunakan modul perangkat keras dan unik untuk setiap perangkat penyimpanan NVMe instance. Semua kunci enkripsi tersebut akan dihancurkan saat instans dihentikan atau diakhiri dan tidak dapat dipulihkan. Anda tidak dapat menonaktifkan enkripsi ini dan Anda tidak dapat menyediakan kunci enkripsi Anda sendiri. Untuk informasi selengkapnya, [lihat Enkripsi saat istirahat](#) di Panduan EC2 Pengguna Amazon.

Jika Anda menggunakan AWS ParallelCluster AWS layanan yang mentransmisikan data pelanggan ke komputer lokal Anda untuk penyimpanan, lihat bagian Keamanan dan Kepatuhan dalam Panduan Pengguna layanan tersebut untuk informasi tentang cara data tersebut disimpan, dilindungi, dan dienkripsi.

Enkripsi bergerak

Secara default, semua data yang dikirimkan dari komputer klien yang berjalan AWS ParallelCluster dan titik akhir AWS layanan dienkripsi dengan mengirimkan semuanya melalui koneksi HTTPS/TLS. Lalu lintas antar node di cluster dapat dienkripsi secara otomatis, tergantung pada jenis instance yang dipilih. Untuk informasi selengkapnya, lihat [Enkripsi dalam perjalanan](#) di Panduan EC2 Pengguna Amazon.

Lihat juga

- [Perlindungan data di Amazon EC2](#)
- [Perlindungan data di EC2 Image Builder](#)
- [Perlindungan data di AWS CloudFormation](#)
- [Perlindungan data di Amazon EFS](#)
- [Perlindungan data di Amazon S3](#)
- [Perlindungan data FSx untuk Lustre](#)

Identity and Access Management untuk AWS ParallelCluster

AWS ParallelCluster menggunakan peran untuk mengakses AWS sumber daya Anda dan layanan mereka. Kebijakan instance dan pengguna yang AWS ParallelCluster digunakan untuk memberikan izin didokumentasikan di [AWS Identity and Access Management peran dalam AWS ParallelCluster](#).

Satu-satunya perbedaan utama adalah bagaimana Anda mengautentikasi saat menggunakan pengguna standar dan kredensi jangka panjang. Meskipun pengguna memerlukan kata sandi untuk mengakses konsol AWS layanan, pengguna yang sama memerlukan access key pair untuk melakukan operasi yang sama menggunakan AWS ParallelCluster. Semua kredensial jangka pendek lainnya digunakan dengan cara yang sama dengan yang digunakan bersama konsol.

Kredensi yang digunakan oleh AWS ParallelCluster disimpan dalam file teks biasa dan tidak dienkripsi.

- File `$HOME/.aws/credentials` menyimpan kredensial jangka panjang yang diperlukan untuk mengakses sumber daya AWS Anda. Ini termasuk access key ID dan secret access key Anda.
- Kredensi jangka pendek, seperti untuk peran yang Anda asumsikan, atau yang untuk AWS IAM Identity Center layanan, juga disimpan dalam `$HOME/.aws/cli/cache` dan `$HOME/.aws/sso/cachefolder`, masing-masing.

Mitigasi Risiko

- Kami sangat merekomendasikan agar Anda mengkonfigurasi izin sistem file pada direktori `$HOME/.aws` dan direktori dan file anaknya untuk membatasi akses hanya untuk pengguna yang berwenang.

- Gunakan peran dengan kredensial sementara sedapat mungkin untuk mengurangi peluang terjadinya kerusakan jika kredensialnya dikurangi. Gunakan kredensial jangka panjang hanya untuk meminta dan menyegarkan kredensial peran jangka pendek.

Validasi kepatuhan untuk AWS ParallelCluster

Auditor pihak ketiga menilai keamanan dan kepatuhan AWS layanan sebagai bagian dari beberapa program AWS kepatuhan. Menggunakan AWS ParallelCluster untuk mengakses layanan tidak mengubah kepatuhan layanan tersebut.

Untuk daftar AWS layanan dalam lingkup program kepatuhan tertentu, lihat [AWS layanan dalam lingkup oleh AWS layanan program kepatuhan](#) . Untuk informasi umum, lihat [program AWS kepatuhan program AWS](#) .

Anda dapat mengunduh laporan audit pihak ketiga menggunakan AWS Artifact. Untuk informasi selengkapnya, lihat [Mengunduh laporan di AWS Artifact](#).

Tanggung jawab kepatuhan Anda saat menggunakan AWS ParallelCluster ditentukan oleh sensitivitas data Anda, tujuan kepatuhan perusahaan Anda, dan hukum dan peraturan yang berlaku. AWS menyediakan sumber daya berikut untuk membantu kepatuhan:

- [Panduan memulai cepat keamanan dan kepatuhan Panduan memulai](#) penerapan ini membahas pertimbangan arsitektur dan memberikan langkah-langkah untuk menerapkan lingkungan dasar yang berfokus pada keamanan dan kepatuhan. AWS
- [Arsitektur untuk keamanan dan Kepatuhan HIPAA di Whitepaper Amazon Web Services — AWS Whitepaper](#) ini menjelaskan bagaimana perusahaan dapat menggunakannya untuk membuat aplikasi yang sesuai dengan HIPAA. AWS
- [AWS sumber daya AWS kepatuhan](#) — Kumpulan buku kerja dan panduan ini mungkin berlaku untuk industri dan lokasi Anda.
- [Mengevaluasi sumber daya dengan aturan](#) dalam Panduan AWS Config Pengembang — AWS Config Layanan menilai seberapa baik konfigurasi sumber daya Anda mematuhi praktik internal, pedoman industri, dan peraturan.
- [AWS Security Hub](#)— AWS Layanan ini memberikan pandangan komprehensif tentang keadaan keamanan Anda di dalamnya AWS yang membantu Anda memeriksa kepatuhan Anda terhadap standar industri keamanan dan praktik terbaik.

Menegakkan Versi Minimum TLS 1.2

Untuk menambahkan peningkatan keamanan saat berkomunikasi dengan AWS layanan, Anda harus mengonfigurasi penggunaan TLS 1.2 atau yang lebih baru. AWS ParallelCluster Saat Anda menggunakan AWS ParallelCluster, Python digunakan untuk mengatur versi TLS.

Untuk memastikan tidak AWS ParallelCluster menggunakan versi TLS lebih awal dari TLS 1.2, Anda mungkin perlu mengkompilasi ulang OpenSSL untuk menerapkan minimum ini dan kemudian mengkompilasi ulang Python untuk menggunakan OpenSSL yang baru dibangun.

Tentukan Protokol yang Didukung Saat Ini

Pertama, buat sertifikat yang ditandatangani sendiri untuk digunakan untuk server pengujian dan SDK Python menggunakan OpenSSL.

```
$ openssl req -subj '/CN=localhost' -x509 -newkey rsa:4096 -nodes -keyout key.pem -out cert.pem -days 365
```

Kemudian putar server uji menggunakan OpenSSL.

```
$ openssl s_server -key key.pem -cert cert.pem -www
```

Di jendela terminal baru, buat lingkungan virtual dan instal Python SDK.

```
$ python3 -m venv test-env
source test-env/bin/activate
pip install botocore
```

Buat skrip Python baru bernama `check.py` yang menggunakan pustaka HTTP dasar 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'))
```

Jalankan skrip baru Anda.

```
$ python check.py
```

Ini menampilkan detail tentang koneksi yang dibuat. Cari "Protocol:" di output. Jika outputnya adalah "TLSv1.2" atau yang lebih baru, SDK default ke TLS v1.2 atau yang lebih baru. Jika ini adalah versi sebelumnya, Anda perlu mengkompilasi ulang OpenSSL dan mengkompilasi ulang Python.

Namun, bahkan jika instalasi Python Anda default ke TLS v1.2 atau yang lebih baru, Python masih mungkin melakukan negosiasi ulang ke versi lebih awal dari TLS v1.2 jika server tidak mendukung TLS v1.2 atau yang lebih baru. Untuk memeriksa bahwa Python tidak secara otomatis melakukan negosiasi ulang ke versi sebelumnya, restart server pengujian dengan yang berikut ini.

```
$ openssl s_server -key key.pem -cert cert.pem -no_tls1_3 -no_tls1_2 -www
```

Jika Anda menggunakan versi OpenSSL yang lebih lama, Anda mungkin tidak memiliki `-no_tls_3` flag yang tersedia. Jika ini masalahnya, hapus bendera karena versi OpenSSL yang Anda gunakan tidak mendukung TLS v1.3. Kemudian jalankan kembali skrip Python.

```
$ python check.py
```

Jika instalasi Python Anda dengan benar tidak menegosiasikan ulang untuk versi yang lebih awal dari TLS 1.2, Anda akan menerima kesalahan SSL.

```
$ urllib3.exceptions.MaxRetryError: HTTPConnectionPool(host='localhost',  
port=4433): Max retries exceeded with url: / (Caused by SSLError(SSLError(1, '[SSL:  
UNSUPPORTED_PROTOCOL] unsupported protocol (_ssl.c:1108)')))
```

Jika Anda dapat membuat koneksi, Anda perlu mengkompilasi ulang OpenSSL dan Python untuk menonaktifkan negosiasi protokol lebih awal dari TLS v1.2.

Kompilasi OpenSSL dan Python

Untuk memastikan bahwa AWS ParallelCluster tidak bernegosiasi untuk apa pun yang lebih awal dari TLS 1.2, Anda perlu mengkompilasi ulang OpenSSL dan Python. Untuk melakukan ini, salin konten berikut untuk membuat skrip dan menjalankannya.

```
#!/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
```

Ini mengkompilasi versi Python yang memiliki OpenSSL yang ditautkan secara statis yang tidak secara otomatis menegosiasikan apa pun lebih awal dari TLS 1.2. Ini juga menginstal OpenSSL di `/opt/openssl-with-min-tls1_2` direktori dan menginstal Python di direktori. `/opt/python-with-min-tls1_2` Setelah Anda menjalankan skrip ini, konfirmasi instalasi versi baru Python.

```
$ /opt/python-with-min-tls1_2/bin/python3 --version
```

Ini harus mencetak yang berikut ini.

```
Python 3.8.1
```

Untuk mengonfirmasi versi baru Python ini tidak menegosiasikan versi lebih awal dari TLS 1.2, jalankan kembali langkah-langkah dari menggunakan versi Python [Tentukan Protokol yang Didukung Saat Ini](#) yang baru diinstal (yaitu,). `/opt/python-with-min-tls1_2/bin/python3`

Catatan rilis dan riwayat dokumen

Tabel berikut menjelaskan pembaruan utama dan fitur baru untuk Panduan AWS ParallelCluster Pengguna. Kami juga rutin memperbarui dokumentasi untuk menjawab umpan balik yang Anda kirimkan kepada kami.

Perubahan	Deskripsi	Tanggal
Dokumentasi hanya rilis	AWS ParallelCluster versi 2 panduan pengguna khusus diterbitkan. Rilis hanya dokumentasi: • AWS ParallelCluster versi 2 memiliki panduan pengguna tersendiri.	Juli 17, 2023
AWS ParallelCluster versi 2.11.9 dirilis	AWS ParallelCluster versi 2.11.9 dirilis. Perbaikan bug: • Mencegah penggantian sistem berkas Lustre yang dikelola FSx dan hilangnya data pada pembaruan klaster yang menyertakan perubahan pada file. <code>vpc_security_group_id</code> Untuk detail perubahannya, lihat <code>CHANGELOG</code> file untuk	Desember 2, 2022

paket [aws-parallelcluster](#) di.
GitHub

[AWS ParallelCluster versi
2.11.8 dirilis](#)

AWS ParallelCluster versi
2.11.8 dirilis.

November 14, 2022

Perubahan:

- Tingkatkan Perpustakaan Intel MPI ke Pembaruan Versi 2021.6 (diperbarui dari Pembaruan Versi 2021.4). Untuk informasi selengkapnya, lihat [Intel® MPI Library 2021 Update 6](#).
- Tingkatkan penginstal EFA ke 1.19.0
 - Pengemudi EFA: `efa-1.16.0-1`
 - EFA-config: `efa-config-1.11-1` (dari `efa-config-1.9-1`)
 - Profil EFA: `efa-profile-1.5-1` (tidak ada perubahan)
 - libFabric-aws: `libfabric-aws-1.16.0-1` (dari `libfabric-1.13.2`)
 - RDMA-core: `rdma-core-41.0-2` (dari `rdma-core-37.0`)
 - Buka MPI: `openmpi40-aws-4.1.4-3` (dari `openmpi40-aws-4.1.1-2`)

- Tingkatkan runtime Python, yang digunakan oleh fungsi Lambda dalam integrasi AWS Batch , ke python3.9.

Perbaiki bug:

- Mencegah tag kluster diubah selama pembaruan karena tidak didukung.

Untuk detail perubahannya, lihat CHANGELOG file untuk paket [aws-parallelcluster](#) aktif. GitHub

[AWS ParallelCluster versi 2.11.7 dirilis](#)

AWS ParallelCluster versi 2.11.7 dirilis.

Mei 13, 2022

Perubahan:

- Tingkatkan Slurm ke versi 20.11.9.

Untuk detail perubahannya, lihat CHANGELOG file untuk paket [aws-parallelcluster](#) aktif. GitHub

[AWS ParallelCluster versi
2.11.6 dirilis](#)

AWS ParallelCluster versi
2.11.6 dirilis.

19 April 2022

Perangkat tambahan:

- Tingkatkan manajemen pengecualian jika jaringan hilang.

Perubahan:

- Pembaruan paket OS dan perbaikan keamanan.

Untuk detail perubahannya, lihat CHANGELOG file untuk paket [aws-parallelcluster](#) aktif. [GitHub](#)

[AWS ParallelCluster versi
2.11.5 dirilis](#)

AWS ParallelCluster versi
2.11.5 dirilis.

1 Maret 2022

Perangkat tambahan:

- Tambahkan dukungan untuk NEW_CHANGED_DELETE sebagai nilai FSx untuk opsi LustreAutoImportPolicy .
- Hapus dukungan untuk penjadwal SGE dan Torque.
- Nonaktifkan log4j-cve-2021-44228-hotpatch layanan di Amazon Linux untuk menghindari timbulnya potensi penurunan kinerja.

Perubahan:

- Tingkatkan driver NVIDIA ke versi 470.103.01 (dari 470.82.01).
- Tingkatkan manajer NVIDIA Fabric ke versi 470.103.01 (dari 470.82.01).
- Tingkatkan pustaka CUDA ke versi 11.4.4 (dari 11.4.3).
- [Intel MPI](#) diperbarui ke Pembaruan Versi 2021 4 (diperbarui dari Pembaruan Versi 2019 8). Untuk

informasi selengkapnya,
lihat [Intel® MPI Library 2021
Update 4](#).

- Perpanjang batas waktu pembuatan node kepala menjadi satu jam.

Perbaiki bug:

- Perbaiki koneksi DCV melalui browser.
- Perbaiki kutipan YAMM untuk mencegah Tag kustom diurai sebagai angka.

Untuk detail perubahannya,
lihat CHANGELOG file untuk paket [aws-parallelcluster](#) aktif.
GitHub

AWS ParallelCluster versi 2.11.4 dirilis

AWS ParallelCluster versi
2.11.4 dirilis.

Desember 20, 2021

Perubahan meliputi:

- CentOS8 dukungan dihapus. CentOS8 mencapai akhir kehidupan (EOL) pada 31 Desember 2021.
- Tingkatkan Slurm Workload Manager ke versi 20.11.8.
- Tingkatkan Klien Cnc ke 17.2.29.
- [Amazon DCV](#) diperbarui ke Amazon DCV 2021.2-11190. Untuk informasi lebih lanjut, lihat [DCV 2021.2-11190— 11 Oktober 2021](#) di Panduan Administrator Amazon DCV.
- Tingkatkan driver NVIDIA ke versi 470.82.01 (dari 460.73.01).
- Tingkatkan pustaka CUDA ke versi 11.4.3 (dari 11.3.0).
- Tingkatkan NVIDIA Fabric Manager ke 470.82.01.
- Nonaktifkan pembaruan paket pada waktu peluncuran instans di Amazon Linux 2.
- Nonaktifkan pembaruan paket tanpa pengawasan

Ubuntu dan Amazon Linux

2.

- Instal [skrip AWS CloudFormation pembantu versi Python 3 pada CentOS 7 dan 18.04](#). Ubuntu (Ini sudah digunakan di Amazon Linux 2 dan Ubuntu 20.04.)

Perbaikan meliputi:

- Nonaktifkan pembaruan [ec2_iam_role](#) parameter .
- Perbaiki CpuOptions konfigurasi di template peluncuran untuk T2 instance.

Untuk detail perubahan, lihat CHANGELOG file untuk [aws-parallelcluster](#), [aws-parallelcluster-cookbook](#) dan paket aktif. [aws-parallelcluster-node](#) GitHub

[AWS ParallelCluster versi 2.11.3 dirilis](#)

AWS ParallelCluster versi 2.11.3 dirilis.

3 November 2021

- Perbaiki [pcluster createami](#) kegagalan karena Son of Grid Engine sumber tidak tersedia `diarc.liv.ac.uk` .

Tingkatkan [Elastic Fabric Adapter](#) penginstal ke 1.14.1 (dari 1.13.0)

- Konfigurasi EFA: `efa-config-1.9-1` (dari `efa-config-1.9`)
- Profil EFA: `efa-profile-1.5-1` (tidak ada perubahan)
- Modul Kernel EFA: `efa-1.14.2` (dari `efa-1.13.0`)
- Inti RDMA: `rdma-core-37.0` (dari `rdma-core-35.0amzn`)
- Libfabric: `libfabric-1.13.2` (dari `libfabric-1.13.0amzn1.0`)
- Buka MPI: `openmpi40-aws-4.1.1-2` (tidak ada perubahan)

GPUDirect RDMA selalu diaktifkan jika didukung oleh jenis instance.

- Opsi [enable_efa_gdr](#) dan [enable_efa_gdr](#) konfigurasi tidak berpengaruh.

Untuk detail perubahan, lihat CHANGELOG file untuk [aws-parallelcluster](#), [aws-parallelcluster-cookbook](#) dan paket aktif. [aws-parallelcluster-node](#) GitHub

[AWS ParallelCluster versi 2.11.2 dirilis](#)

AWS ParallelCluster versi 2.11.2 dirilis.

Agustus 27, 2021

Perubahan meliputi:

- Jangan menginstal EFA dengan GPUDirect RDMA (GDR) diaktifkan pada waktu bootstrap jika EFA diinstal di dasar AMI.
- Kunci versi `nvidia-fabricmanager` paket agar tetap sinkron dengan versi driver NVIDIA yang diinstal oleh AWS ParallelCluster.
- Slurm: Perbaiki masalah yang disebabkan saat cluster dihentikan dan dimulai ulang saat node dinyalakan.
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.13.0:
 - Konfigurasi EFA: `efa-config-1.9` (tidak ada perubahan)
 - Profil EFA: `efa-profile-1.5-1` (tidak ada perubahan)
 - Modul Kernel EFA: `efa-1.13.0` (tidak ada perubahan)
 - Inti RDMA: `rdma-core-35.0amzn` (dari)

`rdma-core-32.1amzn`

- Libfabric: `libfabric-1.13.0amzn1.0`
(dari `libfabric-1.11.2amzn1.1`)
- Buka MPI: `openmpi40-aws-4.1.1-2` (tidak ada perubahan)
- Saat menggunakan AMI khusus dengan paket EFA yang sudah diinstal sebelumnya, tidak ada perubahan yang dilakukan pada EFA pada waktu bootstrap node. Penyebaran paket EFA asli dipertahankan.

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#) dan paketnya. [aws-parallelcluster-cookbook](#) GitHub

[AWS ParallelCluster versi 2.11.1 dirilis](#)

AWS ParallelCluster versi 2.11.1 dirilis.

23 Juli 2021

Perubahan meliputi:

- Pasang sistem file menggunakan opsi `noatime mount` untuk berhenti merekam waktu akses terakhir saat file dibaca. Ini meningkatkan kinerja sistem file jarak jauh.
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.12.3:
 - Konfigurasi EFA: `efa-config-1.9` (dari `efa-config-1.8-1`)
 - Profil EFA: `efa-profile-1.5-1` (tidak ada perubahan)
 - Modul Kernel EFA: `efa-1.13.0` (dari `efa-1.12.3`)
 - Inti RDMA: `rdma-core-32.1amzn` (tidak ada perubahan)
 - Libfabric: `libfabric-1.11.2amzn1.1` (tidak ada perubahan)
 - Buka MPI: `openmpi40-aws-4.1.1-2` (tidak ada perubahan)
- Coba lagi instalasi `aws-parallelcluster`

paket pada node kepala saat menggunakan AWS Batch sebagai penjadwal.

- Hindari kegagalan saat membangun SGE tipe instans dengan lebih dari 31 vCPUs.
- Disematkan ke versi 1.247347.6 dari Agen CloudWatch Amazon untuk menghindari masalah yang terlihat di versi 1.247348.0.

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#) dan paketnya. [aws-parallelcluster-cookbook](#) GitHub

[AWS ParallelCluster versi 2.11.0 dirilis](#)

AWS ParallelCluster versi 2.11.0 dirilis.

1 Juli 2021

Perubahan meliputi:

- Menambahkan dukungan untuk Ubuntu 20.04 (`ubuntu2004`) dan menghapus dukungan untuk Ubuntu 16.04 (`ubuntu1604`) dan Amazon Linux (`alinux`). Amazon Linux 2 (`alinux2`) tetap didukung penuh. Untuk informasi selengkapnya, lihat [base_os](#).
- Dihapus dukungan untuk versi Python di bawah 3.6.
- Ukuran volume root default meningkat menjadi 35 gibibytes (GiB). Untuk informasi selengkapnya, lihat [compute_root_volume_size](#) dan [master_root_volume_size](#).
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.12.2:
 - Konfigurasi EFA: `efa-config-1.8-1` (dari `efa-config-1.7`)
 - Profil EFA: `efa-profile-1.5-1` (dari `efa-profile-1.4`)

- Modul Kernel EFA:
efa-1.12.3
(dari efa-1.10.2)
- Inti RDMA: rdma-core
-32.1amzn (dari)
rdma-core-31.2amzn
- Libfabric: libfabric
-1.11.2amzn1.1
(dari libfabric
-1.11.1amzn1.0)
- Buka MPI: openmpi40
-aws-4.1.1-2
(dari openmpi40-
aws-4.1.0)
- Ditingkatkan Slurm ke versi
20.11.7 (dari 20.02.7).
- Instal Agen SSM di
centos7 dan centos8.
(Agen SSM sudah
diinstal sebelumnya
di linux2, ubuntu1804 ,
dan ubuntu2004 .)
- SGE: Selalu gunakan
shortname sebagai filter
nama host dengan. qstat
- Gunakan layanan metadata
instance Versi 2 (IMDSv2)
alih-alih layanan metadata
instance Versi 1 (IMDSv1)
untuk mengambil metadata
instance. Untuk informasi
selengkapnya, lihat
[Metadata instans dan data](#)

[pengguna](#) di EC2 Panduan Pengguna Amazon.

- Tingkatkan driver NVIDIA ke versi 460.73.01 (dari 450.80.02).
- Tingkatkan pustaka CUDA ke versi 11.3.0 (dari 11.0).
- Tingkatkan NVIDIA Fabric Manager `kenvidia-fabricmanager-460` .
- Upgrade Python yang digunakan dalam AWS ParallelCluster `virtualenvs` ke (from). 3.7.10 3.6.13
- Tingkatkan Klien Cnc ke 16.13.16.
- Tingkatkan dependensi pihak ketiga dari: [aws-parallelcluster-cookbook](#)
 - `apt-7.4.0` (dari `apt-7.3.0`).
 - `iptables-8.0.0` (dari `iptables-7.1.0`).
 - `line-4.0.1` (dari `line-2.9.0`).
 - `openssh-2.9.1` (dari `openssh-2.8.1`).
 - `pyenv-3.4.2` (dari `pyenv-3.1.1`).
 - `selinux-3.1.1` (dari `selinux-2.1.1`).

- `ulimit-1.1.1` (dari `ulimit-1.0.0`).
- `yum-6.1.1` (dari `yum-5.1.0`).
- `yum-epel-4.1.2` (dari `yum-epel-3.3.0`).

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#), dan paket aktif. [aws-parallelcluster-cookbook](#) [aws-parallelcluster-node](#) GitHub

[AWS ParallelCluster versi 2.10.4 dirilis](#)

AWS ParallelCluster versi 2.10.4 dirilis.

15 Mei 2021

Perubahan meliputi:

- Ditingkatkan Slurm ke versi `20.02.7` (dari `20.02.4`).

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk paket [aws-parallelcluster](#) aktif. GitHub

[AWS ParallelCluster versi 2.10.3 dirilis](#)

AWS ParallelCluster versi 2.10.3 dirilis.

18 Maret 2021

Perubahan meliputi:

- Menambahkan dukungan untuk Ubuntu 18.04 dan Amazon Linux 2 pada instance AWS Graviton berbasis ARM di Tiongkok dan. AWS AWS GovCloud (US) Wilayah AWS
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.11.2:
 - Konfigurasi EFA: `efa-config-1.7` (tidak ada perubahan)
 - Profil EFA: `efa-profile-1.4` (dari `efa-profile-1.3`)
 - Modul Kernel EFA: `efa-1.10.2` (tidak ada perubahan)
 - Inti RDMA: `rdma-core-31.2amzn` (tidak ada perubahan)
 - Libfabric: `libfabric-1.11.1amzn1.0` (tidak ada perubahan)
 - Buka MPI: `openmpi40-aws-4.1.0` (tidak ada perubahan)

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk paket [aws-parallelcluster](#) aktif. [GitHub](#)

[AWS ParallelCluster versi 2.10.2 dirilis](#)

AWS ParallelCluster versi 2.10.2 dirilis.

2 Maret 2021

Perubahan meliputi:

- Tingkatkan validasi konfigurasi cluster untuk menggunakan AMI target cluster saat menjalankan operasi Amazon EC2 [RunInstances](#) API dalam mode. `--dry-run`
- Perbarui versi Python yang digunakan di lingkungan AWS ParallelCluster virtual ke 3.6.13.
- Perbaiki [sanity_check](#) untuk jenis instance Arm.
- Perbaiki `enable_efa` saat menggunakan centos8 dengan Slurm penjadwal atau tipe instance Arm.
- Jalankan `apt update` dalam mode non-interaktif (`-y`).
- Fix [encrypted_ephemeral](#) = `true` dengan `alinux2` dan `centos8`.

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk paket [aws-parallelcluster](#) aktif. [GitHub](#)

[AWS ParallelCluster versi 2.10.1 dirilis](#)

AWS ParallelCluster versi 2.10.1 dirilis.

22 Desember 2020

Perubahan meliputi:

- Menambahkan dukungan untuk Afrika (Cape Town) (af-south-1), Eropa (Milan) (me-south-1), dan Timur Tengah (Bahrain) (me-south-1). Wilayah AWS Saat peluncuran, dukungan dibatasi dengan cara-cara berikut:
 - FSx untuk instance Graviton berbasis Lustre dan ARM tidak didukung dalam semua ini. Wilayah AWS
 - AWS Batch tidak didukung di Afrika (Cape Town).
 - Amazon EBS io2 dan jenis gp3 volume tidak didukung di Afrika (Cape Town) dan Eropa (Milan) Wilayah AWS.
- Menambahkan dukungan untuk Amazon EBS io2 dan jenis gp3 volume. Untuk informasi lebih lanjut, lihat [\[ebs\]bagian](#) dan [\[raid\]bagian](#).
- Menambahkan dukungan untuk [Elastic Fabric Adapter](#) instance Graviton2 berbasis

ARM yang berjalan,, atau.
a) linux2 ubuntu1804
4 ubuntu2004 Untuk informasi selengkapnya, lihat [Elastic Fabric Adapter](#).

- Instal Arm Performance Libraries 20.2.1 di Arm AMIs (a) linux2, centos8, dan ubuntu1804). Untuk informasi selengkapnya, lihat [Perpustakaan Kinerja Arm](#).
- [Intel MPI](#) diperbarui ke Pembaruan Versi 2019 8 (diperbarui dari Pembaruan Versi 2019 7). Untuk informasi selengkapnya, lihat [Intel® MPI Library 2019 Update 8](#).
- Menghapus panggilan operasi AWS CloudFormation DescribeStacks API dari titik masuk AWS Batch Docker untuk mengakhiri kegagalan pekerjaan yang disebabkan oleh pembatasan oleh. AWS CloudFormation
- Meningkatkan panggilan ke panggilan operasi Amazon EC2 DescribeInstanceTypes API saat memvalidasi konfigurasi cluster.

- Gambar Amazon Linux 2 Docker diambil dari Amazon ECR Public saat membuat gambar Docker untuk penjadwal. `awsbatch`
- Jenis instans default diubah dari tipe `t2.micro` instance hardcode ke tipe instans Tingkat Gratis untuk Wilayah AWS (`t2.micro` atau `t3.micro`, tergantung pada Wilayah AWS). Wilayah AWS yang tidak memiliki default Tingkat Gratis untuk jenis `t3.micro` instance.
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.11.1:
 - Konfigurasi EFA: `efa-config-1.7` (dari `efa-config-1.5`)
 - Profil EFA: `efa-profile-1.3` (dari `efa-profile-1.1`)
 - Modul Kernel EFA: `efa-1.10.2` (tidak ada perubahan)
 - Inti RDMA: `rdma-core-31.2amzn` (dari `rdma-core-31.amzn0`)
 - Libfabric: `libfabric-1.11.1amzn1.0`

- (dari libfabric-1.10.1-1.amzn1.1)
- Buka MPI: openmpi40-aws-4.1.0 (dari openmpi40-aws-4.0.5)
- [master_subnet_id](#) Parameter [vpc_settings vpc_id](#), dan sekarang diperlukan.
- nfsd Daemon di node kepala sekarang diatur untuk menggunakan setidaknya 8 utas. Jika ada lebih dari 8 core, itu akan menggunakan thread sebanyak yang ada core. Kapan ubuntu1604 digunakan, pengaturan hanya berubah setelah node di-boot ulang.
- [Amazon DCV diperbarui ke Amazon DCV 2020.2-9662](#). Untuk informasi lebih lanjut, lihat [DCV 2020.2-9662— 04 Desember 2020 di Panduan Administrator](#) Amazon DCV.
- Paket Intel MPI dan HPC untuk ditarik dari AWS ParallelCluster Amazon S3. Mereka tidak lagi ditarik dari repo Intel yum.
- Mengubah systemd runlevel default menjadi multi-user.target on all

OSs selama pembuatan resmi AWS ParallelCluster AMIs. Runlevel diatur ke `graphical.target` node kepala hanya ketika DCV diaktifkan. Ini mencegah layanan grafis (seperti `x/gdm`) berjalan saat tidak diperlukan.

- Diaktifkan dukungan untuk `p4d.24xlarge` instance pada node kepala.
- Tingkatkan jumlah maksimum percobaan ulang saat mendaftarkan Slurm node di Amazon Route 53.

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#), dan paket aktif. [aws-parallelcluster-cookbook](#), [aws-parallelcluster-node](#) dan [aws-parallelcluster-node](#) di GitHub

[AWS ParallelCluster versi 2.10.0 dirilis](#)

AWS ParallelCluster versi 2.10.0 dirilis.

18 November 2020

Perubahan meliputi:

- Menambahkan dukungan untuk CentOS 8 di semua Wilayah AWS (di luar Wilayah AWS Tiongkok AWS GovCloud dan (AS)). Dukungan dihapus untuk CentOS 6.
- Menambahkan dukungan untuk p4d.24xlarge instance untuk node komputasi.
- Ditambahkan dukungan untuk NVIDIA GPUDirect RDMA pada EFA dengan menggunakan pengaturan baru. [enable_efa_gdr](#)
- Ditambahkan dukungan untuk Amazon FSx untuk fitur Lustre.
 - Konfigurasi sistem file Amazon FSx for Lustre Anda untuk mengimpor preferensi menggunakan pengaturan. [auto_import_policy](#)
 - Ditambahkan dukungan untuk Amazon berbasis HDD FSx untuk sistem file Lustre menggunakan dan pengaturan. [storage_t](#)

[ype_drive_cac he_type](#)

- Menambahkan CloudWatch dashboard Amazon, termasuk metrik simpul kepala dan akses mudah ke log kluster. Untuk informasi selengkapnya, lihat [CloudWatch Dashboard Amazon](#).
- Menambahkan dukungan untuk menggunakan bucket Amazon S3 yang ada untuk menyimpan informasi konfigurasi cluster, menggunakan pengaturan. [cluster_resource_bucket](#)
- Meningkatkan [pcluster createami](#) perintah.
 - Menambahkan `--post-install` parameter untuk menggunakan skrip pasca-instalasi saat membangun AMI.
 - Menambahkan langkah validasi untuk gagal saat menggunakan AMI dasar yang dibuat oleh versi yang berbeda dari AWS ParallelCluster
 - Ditambahkan langkah validasi untuk gagal ketika jika pilih OS berbeda dari OS di dasar AMI.

- Menambahkan dukungan untuk menggunakan AMI AWS ParallelCluster dasar.
- Meningkatkan [pcluster update](#) perintah.
 - [tags](#) Pengaturan sekarang dapat diubah selama pembaruan.
 - Antrian sekarang dapat diubah ukurannya selama pembaruan tanpa menghentikan armada komputasi
- Ditambahkan parameter `all_or_nothing_batch` konfigurasi untuk `slurm_resume` script. `KapanTrue`, `slurm_resume` akan berhasil hanya jika semua contoh yang diperlukan oleh semua pekerjaan yang tertunda Slurm akan tersedia. Untuk informasi selengkapnya, lihat [Memperkenalkan all_or_nothing_batch peluncuran](#) di AWS ParallelCluster Wiki pada GitHub.
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.10.1:

- Konfigurasi EFA: `efa-config-1.5` (dari `efa-config-1.4`)
- Profil EFA: `efa-profile-1.1` (dari `efa-profile-1.0.0`)
- Modul Kernel EFA: `efa-1.10.2` (dari `efa-1.6.0`)
- Inti RDMA: `rdma-core-31.amzn0` (dari `rdma-core-28.amzn0`)
- Libfabric: `libfabric-1.11.1amzn1.0` (dari `libfabric-1.10.1amzn1.1`)
- Buka MPI: `openmpi40-aws-4.0.5` (dari `openmpi40-aws-4.0.3`)
- Di AWS GovCloud (US) Wilayah, aktifkan dukungan untuk Amazon DCV dan AWS Batch.
- Di Wilayah AWS Tiongkok, aktifkan dukungan untuk FSx Amazon untuk Lustre.
- Tingkatkan driver NVIDIA ke versi 450.80.02 (dari 450.51.05).
- Instal NVIDIA Fabric Manager untuk mengaktif

kan NVIDIA NVSwitch pada platform yang didukung.

- Dihapus default Wilayah AWS dari `aws-east-1`. Default menggunakan urutan pencarian ini.
 - Wilayah AWS ditentukan dalam `-r` atau `--region` argumen.
 - `AWS_DEFAULT_REGION` variabel lingkungan.
 - `aws_region_name` pengaturan di [\[aws\]bagian](#) file AWS ParallelCluster konfigurasi (default ke). `~/.parallelcluster/config`
 - `region` pengaturan di `[default]` bagian file AWS CLI konfigurasi (default ke). `~/.aws/config`

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#), dan paket aktif. [aws-parallelcluster-cookbook](#) [aws-parallelcluster-node](#) GitHub

[AWS ParallelCluster versi 2.9.0 dirilis](#)

AWS ParallelCluster versi 2.9.0 dirilis.

11 September 2020

Perubahan meliputi:

- Menambahkan dukungan untuk beberapa antrian dan beberapa jenis instans dalam armada komputasi saat digunakan. Slurm Workload Manager Saat menggunakan antrian, grup Auto Scaling tidak lagi digunakan. Slurm Zona yang dihosting Amazon Route 53 sekarang dibuat dengan cluster dan digunakan untuk resolusi DNS node komputasi saat Slurm penjadwal digunakan . Untuk informasi selengkapnya, lihat [Mode antrian ganda](#).
- Menambahkan dukungan untuk [Amazon DCV](#) pada instance berbasis AWS Graviton berbasis ARM.
- Menambahkan dukungan untuk menonaktifkan hyperthreading pada tipe instans yang tidak mendukung opsi CPU di template peluncuran (misalnya *.metal jenis instance).

- Menambahkan dukungan untuk NFS 4 untuk sistem file yang dibagikan dari node kepala.
- Ketergantungan yang dihapus pada [cfn-init](#) saat mem-boot node komputasi untuk menghindari pelambatan AWS CloudFormation ketika sejumlah besar node bergabung dengan cluster.
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.9.5:
 - Konfigurasi EFA: efa-config-1.4 (dari efa-config-1.3)
 - Profil EFA: efa-profile-1.0.0 (baru)
 - Modul kernel: efa-1.6.0 (tidak ada perubahan)
 - Inti RDMA: rdma-core-28.amzn0 (tidak ada perubahan)
 - Libfabric: libfabric-1.10.1amzn1.1 (tidak ada perubahan)
 - Buka MPI: openmpi40-aws-4.0.3 (tidak ada perubahan)
- Ditingkatkan Slurm ke versi 20.02.4 (dari 19.05.5).

- [Amazon DCV](#) diperbarui ke Amazon DCV 2020.1-9012. Untuk informasi lebih lanjut, lihat [DCV 2020.1-9012— 24 Agustus 2020 Catatan Rilis di Panduan Administrator Amazon DCV](#).
- Saat memasang drive NFS bersama, gunakan alamat IP pribadi node kepala alih-alih nama host.
- Menambahkan aliran log baru ke CloudWatch Log:chef-client ,clustermgmt , compute mgtd slurm_resource , dan slurm_suspend .
- Menambahkan dukungan untuk nama antrian dalam skrip pra-instal dan pasca-instal.
- Di AWS GovCloud (US) Wilayah AWS, gunakan opsi penagihan berdasarkan permintaan Amazon DynamoDB. Untuk informasi selengkapnya, lihat [Mode Sesuai Permintaan](#) di Panduan Pengembang Amazon DynamoDB.

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#)

[lelcluster](#),, dan paket aktif.
[aws-parallelcluster-cookboo](#)
[kaws-parallelcluster-node](#)
GitHub

AWS ParallelCluster versi 2.8.1 dirilis

AWS ParallelCluster versi
2.8.1 dirilis.

4 Agustus 2020

Perubahan meliputi:

- Nonaktifkan kunci layar untuk sesi Amazon DCV untuk mencegah pengguna terkunci.
- Perbaiki [pcluster configure](#) saat menyertakan tipe instance berbasis AWS Graviton berbasis ARM.

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-paral](#)
[lelcluster](#),, dan paket aktif.
[aws-parallelcluster-cookboo](#)
[kaws-parallelcluster-node](#)
GitHub

[AWS ParallelCluster versi 2.8.0 dirilis](#)

AWS ParallelCluster versi 2.8.0 dirilis.

23 Juli 2020

Perubahan meliputi:

- Menambahkan dukungan untuk instance berbasis AWS Graviton berbasis ARM (seperti dan). A1 C6g
- Ditambahkan dukungan untuk fitur backup harian otomatis Amazon FSx untuk Lustre. Untuk informasi lebih lanjut, lihat [automatic_backup_retention_days](#), [copy_tags_to_backups](#), [daily_automatic_backup_start_time](#), dan [fsx_backup_id](#).
- Dihapus ketergantungan pada Berkshelf dari. [pcluster createami](#)
- Meningkatkan ketahanan dan pengalaman pengguna. [pcluster update](#)
Untuk informasi selengkapnya, lihat [Menggunakan pcluster update](#).
- [Elastic Fabric Adapter](#) installer diperbarui ke 1.9.4:
 - Modul kernel:
efa-1.6.0 (diperbarui dari efa-1.5.1)

- Inti RDMA: `rdma-core-28.amzn0` (diperbarui dari `rdma-core-25.0`)
- Libfabric: `libfabric-1.10.1amzn1.1` (diperbarui dari `libfabric-aws-1.9.0amzn1.1`)
- Buka MPI: `openmpi40-aws-4.0.3` (tidak ada perubahan)
- Tingkatkan driver NVIDIA ke Tesla versi 440.95.01 pada CentOS 6 dan versi 450.51.05 pada semua distribusi lainnya.
- Tingkatkan pustaka CUDA ke versi 11.0 pada semua distribusi selain 6. CentOS

Untuk detail selengkapnya tentang perubahan, lihat file CHANGELOG untuk [aws-parallelcluster](#), dan paket aktif. [aws-parallelcluster-cookbook](#) [aws-parallelcluster-node](#) [GitHub](#)

[AWS ParallelCluster versi 2.7.0 dirilis](#)

AWS ParallelCluster versi 2.7.0 dirilis.

19 Mei 2020

Perubahan meliputi:

- [base_os](#) sekarang menjadi parameter yang diperlukan.
- [scheduler](#) sekarang menjadi parameter yang diperlukan.
- [Amazon DCV](#) diperbarui ke Amazon DCV 2020.0. Untuk informasi selengkapnya, lihat [Amazon DCV merilis versi 2020.0 dengan surround sound 7.1](#) dan dukungan stylus.

[Intel MPI](#) diperbarui ke Pembaruan Versi 2019 7 (diperbarui dari Pembaruan Versi 2019 6). Untuk informasi selengkapnya, lihat [Intel® MPI Library 2019 Update 7](#).

[Elastic Fabric](#)

[Adapter](#) installer diperbarui ke 1.8.4:

- Modul kernel: `efa-1.5.1` (tidak ada perubahan)
- Inti RDMA: `rdma-core-25.0` (tidak ada perubahan)

- Libfabric: libfabric-aws-1.9.0amzn1.1 (tidak ada perubahan)
- Buka MPI: openmpi40-aws-4.0.3 (diperbarui dari openmpi40-aws-4.0.2)
- Tingkatkan CentOS 7 AMI ke versi 7.8-2003 (diperbarui dari 7.7-1908). Untuk informasi lebih lanjut, lihat [CentOS-7 \(2003\) Catatan Rilis](#).

[AWS ParallelCluster versi 2.6.1 dirilis](#)

AWS ParallelCluster versi 2.6.1 dirilis.

17 April 2020

Perubahan meliputi:

- Dihapus `cfn-init-cmd` dan `cfn-wire` dari log yang disimpan di Amazon CloudWatch Logs. Untuk informasi selengkapnya, lihat [Integrasi dengan Amazon CloudWatch Logs](#).

[AWS ParallelCluster versi
2.6.0 dirilis](#)

AWS ParallelCluster versi
2.6.0 dirilis.

27 Februari 2020

Perubahan meliputi:

- Menambahkan dukungan untuk Amazon Linux 2.
- Sekarang Amazon CloudWatch Logs digunakan untuk mengumpulkan log cluster dan scheduler. Untuk informasi selengkapnya, lihat [Integrasi dengan Amazon CloudWatch Logs](#).
- Menambahkan dukungan untuk Amazon baru FSx untuk jenis penerapan Lustre dan. SCRATCH_2 PERSISTENT_1 Support FSx untuk Lustre pada Ubuntu 18.04 dan 16.04. Untuk informasi lebih lanjut, lihat [fsx](#).
- Menambahkan dukungan untuk Amazon DCV pada Ubuntu 18.04. Untuk informasi selengkapnya, lihat [Connect ke head node melalui Amazon DCV](#).

[AWS ParallelCluster versi
2.5.1 dirilis](#)

AWS ParallelCluster versi
2.5.1 dirilis.

13 Desember 2019

[AWS ParallelCluster versi
2.5.0 dirilis](#)

AWS ParallelCluster versi
2.5.0 dirilis.

18 November 2019

[AWS ParallelCluster memperkenalkan dukungan untuk Intel MPI](#)

AWS ParallelCluster versi 2.4.1 memperkenalkan dukungan untuk Intel MPI.

29 Juli 2019

[AWS ParallelCluster memperkenalkan dukungan untuk EFA](#)

AWS ParallelCluster versi 2.4.0 memperkenalkan dukungan untuk Elastic Fabric Adapter (EFA).

11 Juni 2019

[AWS ParallelCluster dokumentasi dirilis di situs AWS dokumentasi](#)

AWS ParallelCluster Dokumentasi sekarang tersedia dalam 10 bahasa dan dalam format HTML dan PDF.

24 Mei 2018

Terjemahan disediakan oleh mesin penerjemah. Jika konten terjemahan yang diberikan bertentangan dengan versi bahasa Inggris aslinya, utamakan versi bahasa Inggris.