AWS ParallelCluster Guide de l'utilisateur (v2)

AWS ParallelCluster

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

AWS ParallelCluster: AWS ParallelCluster Guide de l'utilisateur (v2)

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

Les marques commerciales et la présentation commerciale d’Amazon ne peuvent pas être utilisées en relation avec un produit ou un service extérieur à Amazon, d’une manière susceptible d’entraîner une confusion chez les clients, ou d’une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce que AWS ParallelCluster 1
Tarification 1
Con AWS ParallelCluster figuration 2
Installation AWS ParallelCluster 2
Installation AWS ParallelCluster dans un environnement virtuel (recommandé) 2
Installation AWS ParallelCluster dans un environnement non virtuel à l'aide de pip 3
Étapes à suivre après l'installation 3
Instructions détaillées pour chaque environnement 4
Environnement virtuel 4
Linux 6
macOS 11
Windows 13
Configuration AWS ParallelCluster 16
Bonnes pratiques 24
Bonnes pratiques : sélection du type d'instance principale 24
Bonnes pratiques : performances du réseau 25
Bonnes pratiques : alertes budgétaires 25
Bonnes pratiques : déplacement d'un cluster vers une nouvelle version AWS ParallelCluster mineure ou un correctif 26
Passage de CfnCluster à AWS ParallelCluster 27
Régions prises en charge 29
En utilisant AWS ParallelCluster 31
Configurations réseau 31
AWS ParallelCluster dans un seul sous-réseau public 32
AWS ParallelCluster en utilisant deux sous-réseaux 33
AWS ParallelCluster dans un seul sous-réseau privé connecté à l'aide de AWS Direct Connect 34
AWS ParallelCluster avec `awsbatch` planificateur 35
Actions d'amorçage personnalisées 36
Configuration 38
Arguments 38
exemple 38
Travailler avec Amazon S3 40
Exemples 40

Utilisation de instances Spot ... 41
Scénario 1 : Une instance Spot sans tâches en cours d'exécution est interrompue ... 41
Scénario 2 : Une instance Spot exécutant des tâches à nœud unique est interrompue ... 42
Scénario 3 : Une instance Spot exécutant des tâches à plusieurs nœuds est interrompue ... 43
AWS Identity and Access Management rôles dans AWS ParallelCluster ... 45
Paramètres par défaut pour la création de clusters ... 45
Utilisation d'un rôle IAM existant pour Amazon EC2 ... 45
AWS ParallelCluster exemple de politiques relatives aux instances et aux utilisateurs ... 46
Planificateurs pris en charge par AWS ParallelCluster ... 87
Son of Grid Engine ... 88
Slurm Workload Manager ... 88
Torque Resource Manager ... 101
AWS Batch ... 102
Identification ... 109
Tableau de CloudWatch bord Amazon ... 112
Intégration à Amazon CloudWatch Logs ... 114
Elastic Fabric Adapter ... 117
Solutions Intel Select ... 118
Activation d'Intel MPI ... 119
Spécifications de plateforme HPC Intel ... 120
Bibliothèques de performances Arm ... 121
Connectez-vous au nœud principal via Amazon DCV ... 123
Certificat HTTPS Amazon DCV ... 124
Licence Amazon DCV ... 124
Utiliser `pcluster update` ... 125
Corriger les AMI et remplacer les EC2 instances ... 128
Mise à jour ou remplacement de l'instance du nœud principal ... 128
Limites du stockage d'instances ... 129
Solutions de contournement des limites du stockage d'instances ... 130
Arrêter et démarrer le nœud principal d'un cluster ... 131
AWS ParallelCluster Commandes CLI ... 133
`pcluster` ... 133
Arguments ... 133
Sous-commandes : ... 133
`pcluster configure` ... 134
`pcluster create` ... 135

pcluster createami 137
pcluster dcv 141
pcluster delete 143
pcluster instances 145
pcluster list 146
pcluster ssh 147
pcluster start 148
pcluster status 150
pcluster stop 150
pcluster update 151
pcluster version 154
pcluster-config 154
Arguments nommés 154
Configuration 157
Disposition 158
[global] Section 158
cluster_template 158
update_check 159
sanity_check 159
[aws] Section 159
[aliases] Section 160
[cluster] Section 161
additional_cfn_template 163
additional_iam_policies 163
base_os 164
cluster_resource_bucket 166
cluster_type 167
compute_instance_type 168
compute_root_volume_size 168
custom_ami 169
cw_log_settings 170
dashboard_settings 170
dcv_settings 171
desired_vcpus 171
disable_cluster_dns 172
disable_hyperthreading 172

ebs_settings 173
ec2_iam_role 174
efs_settings 174
enable_efa 174
enable_efa_gdr 175
enable_intel_hpc_platform 176
encrypted_ephemeral 177
ephemeral_dir 177
extra_json 177
fsx_settings 178
iam_lambda_role 178
initial_queue_size 179
key_name 180
maintain_initial_size 180
master_instance_type 181
master_root_volume_size 181
max_queue_size 182
max_vcpus 182
min_vcpus 183
placement 183
placement_group 184
post_install 185
post_install_args 185
pre_install 185
pre_install_args 186
proxy_server 186
queue_settings 186
raid_settings 187
s3_read_resource 188
s3_read_write_resource 188
scaling_settings 188
scheduler 189
shared_dir 190
spot_bid_percentage 190
spot_price 191
tags 191

template_url 192
vpc_settings 193
[compute_resource] Section 193
initial_count 194
instance_type 194
max_count 195
min_count 195
spot_price 196
[cw_log] Section 196
enable 196
retention_days 197
[dashboard] Section 197
enable 198
[dcv] Section 198
access_from 199
enable 199
port 200
[ebs] Section 200
shared_dir 201
ebs_kms_key_id 202
ebs_snapshot_id 202
ebs_volume_id 202
encrypted 202
volume_iops 203
volume_size 204
volume_throughput 205
volume_type 205
[efs] Section 206
efs_fs_id 207
efs_kms_key_id 208
encrypted 208
performance_mode 209
provisioned_throughput 209
shared_dir 210
throughput_mode 210
[fsx] Section 211

- `auto_import_policy` ... 213
- `automatic_backup_retention_days` ... 214
- `copy_tags_to_backups` ... 214
- `daily_automatic_backup_start_time` ... 215
- `data_compression_type` ... 216
- `deployment_type` ... 216
- `drive_cache_type` ... 217
- `export_path` ... 218
- `fsx_backup_id` ... 218
- `fsx_fs_id` ... 219
- `fsx_kms_key_id` ... 219
- `import_path` ... 220
- `imported_file_chunk_size` ... 220
- `per_unit_storage_throughput` ... 221
- `shared_dir` ... 221
- `storage_capacity` ... 222
- `storage_type` ... 223
- `weekly_maintenance_start_time` ... 225

`[queue]` Section ... 225
- `compute_resource_settings` ... 226
- `compute_type` ... 226
- `disable_hyperthreading` ... 227
- `enable_efa` ... 227
- `enable_efa_gdr` ... 228
- `placement_group` ... 228

`[raid]` Section ... 229
- `shared_dir` ... 230
- `ebs_kms_key_id` ... 230
- `encrypted` ... 231
- `num_of_raid_volumes` ... 231
- `raid_type` ... 231
- `volume_iops` ... 232
- `volume_size` ... 233
- `volume_throughput` ... 234
- `volume_type` ... 234

`[scaling]` Section ... 235

- `scaledown_idletime` ... 236
- `[vpc]` Section ... 236
  - `additional_sg` ... 237
  - `compute_subnet_cidr` ... 237
  - `compute_subnet_id` ... 237
  - `master_subnet_id` ... 237
  - `ssh_from` ... 238
  - `use_public_ips` ... 238
  - `vpc_id` ... 239
  - `vpc_security_group_id` ... 239
- Exemples ... 40
  - Slurm exemple ... 240
  - SGE et Torque exemple ... 241
  - AWS Batch exemple ... 242

Comment AWS ParallelCluster fonctionne ... 244
- AWS ParallelCluster processus ... 244
  - `SGE and Torque integration processes` ... 245
  - `Slurm integration processes` ... 251
- AWS services utilisés par AWS ParallelCluster ... 251
  - AWS Auto Scaling ... 252
  - AWS Batch ... 253
  - AWS CloudFormation ... 253
  - Amazon CloudWatch ... 254
  - Amazon CloudWatch Logs ... 254
  - AWS CodeBuild ... 254
  - Amazon DynamoDB ... 255
  - Amazon Elastic Block Store ... 255
  - Amazon Elastic Compute Cloud ... 255
  - Amazon Elastic Container Registry ... 255
  - Amazon EFS ... 256
  - Amazon FSx pour Lustre ... 256
  - AWS Identity and Access Management ... 256
  - AWS Lambda ... 257
  - Amazon DCV ... 257
  - Amazon Route 53 ... 257
  - Amazon Simple Notification Service ... 257

Amazon Simple Queue Service ... 258
Amazon Simple Storage Service ... 258
Amazon VPC ... 259
AWS ParallelCluster Auto Scaling ... 259
Mise à l'échelle ... 260
Diminution ... 261
Cluster statique ... 261
Didacticiels ... 262
Exécution de votre premier job sur AWS ParallelCluster ... 262
Vérification de votre installation ... 262
Création de votre premier cluster ... 263
Connexion à votre nœud principal ... 263
Exécution de votre première tâche en utilisant SGE ... 264
Création d'une AWS ParallelCluster AMI personnalisée ... 265
Comment personnaliser l' AWS ParallelCluster AMI ... 266
Modification d’une AMI ... 267
Création d'une AWS ParallelCluster AMI personnalisée ... 269
Utilisation d'une AMI personnalisée lors de l'exécution ... 270
Exécution d'une tâche MPI avec un AWS ParallelCluster planificateur `awsbatch` ... 271
Création du cluster ... 271
Connexion à votre nœud principal ... 263
Exécution de votre première tâche en utilisant AWS Batch ... 273
Exécution d'une tâche MPI dans un environnement parallèle à plusieurs nœuds ... 275
Chiffrement du disque avec une clé KMS personnalisée ... 279
Création du rôle ... 280
Donnez vos autorisations clés ... 280
Création du cluster ... 271
Tutoriel en mode file d'attente multiple ... 281
Exécution de vos tâches AWS ParallelCluster en mode file d'attente multiple ... 282
Développement ... 295
Configuration d'un livre de AWS ParallelCluster recettes personnalisé ... 295
Étapes ... 296
Configuration d'un package de AWS ParallelCluster nœuds personnalisé ... 297
Étapes ... 297
Résolution des problèmes ... 299
Récupération et conservation des journaux ... 299

Résolution des problèmes de déploiement des piles ... 300
Résolution des problèmes dans plusieurs clusters en mode file d'attente ... 301
Journaux clés ... 301
Résolution des problèmes d'initialisation des nœuds ... 302
Résolution des problèmes de remplacement et de terminaison inattendus de nœuds ... 304
Remplacement, arrêt ou mise hors tension des instances et des nœuds problématiques ... 306
Résolution d'autres problèmes connus liés aux nœuds et aux tâches ... 306
Résolution des problèmes dans les clusters en mode file d'attente unique ... 306
Journaux clés ... 307
Résolution des problèmes d'échec des opérations de lancement et de jointure ... 309
Résolution des problèmes de dimensionnement ... 309
Résolution d'autres problèmes liés au cluster ... 309
Problèmes liés aux groupes de placement et au lancement d'instances ... 310
Répertoires qui ne peuvent pas être remplacés ... 310
Résolution des problèmes dans Amazon DCV ... 311
Journaux pour Amazon DCV ... 311
Mémoire de type d'instance Amazon DCV ... 312
Problèmes liés à Ubuntu Amazon DCV ... 312
Résolution des problèmes dans les clusters avec AWS Batch intégration ... 312
Problèmes liés au nœud principal ... 313
AWS Batch problèmes de soumission de tâches parallèles sur plusieurs nœuds ... 313
Problèmes de calcul ... 313
Échecs au travail ... 313
Résolution des problèmes en cas d'échec de création d'une ressource ... 313
Résolution des problèmes liés à la taille des politiques IAM ... 315
Support supplémentaire ... 315
AWS ParallelCluster politique de support ... 316
Sécurité ... 317
Informations de sécurité pour les services utilisés par AWS ParallelCluster ... 318
Protection des données ... 318
Chiffrement des données ... 319
Consultez aussi ... 321
Gestion de l’identité et des accès ... 321
Validation de conformité ... 322
Application de TLS 1.2 ... 323
Déterminez vos protocoles actuellement pris en charge ... 323

Compiler OpenSSL et Python 325
Notes de publication et historique du document 327
..... ccclxxiv

Qu'est-ce que AWS ParallelCluster

AWS ParallelCluster est un outil de gestion de clusters open source pris en AWS charge qui vous aide à déployer et à gérer des clusters de calcul haute performance (HPC) dans le AWS Cloud. Il configure automatiquement les ressources de calcul, le planificateur et le système de fichiers partagé requis. Vous pouvez utiliser AWS ParallelCluster avec AWS Batch et Slurm planificateurs.

Vous pouvez ainsi créer et déployer rapidement des environnements de calcul HPC de production et de validation de concept. AWS ParallelCluster Vous pouvez également créer et déployer un flux de travail de haut niveau par-dessus AWS ParallelCluster, tel qu'un portail génomique qui automatise l'intégralité d'un flux de travail de séquençage de l'ADN.

Tarification

Lorsque vous utilisez l'interface de ligne de AWS ParallelCluster commande (CLI) ou l'API, vous ne payez que pour les AWS ressources créées lorsque vous créez ou mettez à jour AWS ParallelCluster des images et des clusters. Pour de plus amples informations, veuillez consulter AWS services utilisés par AWS ParallelCluster.

Con AWS ParallelCluster figuration

Rubriques

- Installation AWS ParallelCluster
- Configuration AWS ParallelCluster
- Bonnes pratiques
- Passage de CfnCluster à AWS ParallelCluster
- Régions prises en charge

Installation AWS ParallelCluster

AWS ParallelCluster est distribué sous forme de package Python et est installé à l'aide `pip` du gestionnaire de packages Python. Pour plus d'informations sur l'installation de packages Python, consultez la section Installation de packages dans le Guide de l'utilisateur des packages Python.

Modes d'installation AWS ParallelCluster :

- Utilisation d'un environnement virtuel (recommandé)
- Utilisation de `pip`

Vous trouverez le numéro de version de la CLI la plus récente sur la page des versions sur GitHub.

Dans ce guide, les exemples de commande supposent que vous avez installé Python v3. Les exemples de commande `pip` utilisent la version `pip3`.

Installation AWS ParallelCluster dans un environnement virtuel (recommandé)

Nous vous recommandons d'effectuer l'installation AWS ParallelCluster dans un environnement virtuel. Si vous rencontrez des problèmes lorsque vous tentez d'effectuer une installation AWS ParallelCluster avec`pip3`, vous pouvez l'installer AWS ParallelCluster dans un environnement virtuel afin d'isoler l'outil et ses dépendances. Vous pouvez également utiliser une version différente de Python que celle que vous utilisez normalement.

Installation AWS ParallelCluster dans un environnement non virtuel à l'aide de pip

La principale méthode de distribution pour AWS ParallelCluster Linux, Windows et macOS est `pip` un gestionnaire de packages pour Python. Il fournit un moyen d'installer, de mettre à niveau et de supprimer des packages Python et leurs dépendances.

AWS ParallelCluster Version actuelle

AWS ParallelCluster est régulièrement mis à jour. Pour savoir si vous disposez de la dernière version, consultez la page des versions sur GitHub.

Si vous possédez `pip` déjà une version compatible de Python, vous pouvez l'installer à AWS ParallelCluster l'aide de la commande suivante. Si vous avez Python version 3+ installé, nous vous recommandons d'utiliser la commande **`pip3`**.

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

Étapes à suivre après l'installation

Après l'installation AWS ParallelCluster, vous devrez peut-être ajouter le chemin du fichier exécutable à votre `PATH` variable. Pour obtenir des instructions spécifiques à la plate-forme, consultez les rubriques suivantes :

- Linux : Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande
- macOS : Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande
- Windows – Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande

Vous pouvez vérifier qu'il AWS ParallelCluster est correctement installé en exécutant`pcluster version`.

```
$ pcluster version
2.11.9
```

AWS ParallelCluster est régulièrement mis à jour. Pour effectuer la mise à jour vers la dernière version de AWS ParallelCluster, réexécutez la commande d'installation. Pour plus de détails sur la dernière version de AWS ParallelCluster, consultez les notes AWS ParallelCluster de publication.

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

Pour désinstaller AWS ParallelCluster, utilisez`pip uninstall`.

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

Si vous n'avez pas Python ni `pip`, utilisez la procédure indiquée pour votre environnement.

Instructions détaillées pour chaque environnement

- Installation AWS ParallelCluster dans un environnement virtuel (recommandé)
- Installation AWS ParallelCluster sous Linux
- Installation AWS ParallelCluster sur macOS
- Installation AWS ParallelCluster sous Windows

Installation AWS ParallelCluster dans un environnement virtuel (recommandé)

Nous vous recommandons d'effectuer l'installation AWS ParallelCluster dans un environnement virtuel afin d'éviter tout conflit de version entre les exigences et les autres `pip` packages.

Prérequis

- Vérifiez que `pip` et Python sont installés. Nous recommandons `pip3` la version 3.8 de Python 3. Si vous utilisez Python 2, utilisez `pip` au lieu de `pip3` et `virtualenv` au lieu de `venv`.

Pour effectuer une installation AWS ParallelCluster dans un environnement virtuel

1. Si `virtualenv` n'est pas installé, installez `virtualenv` à l'aide de `pip3`. Si `python3 -m virtualenv help` affiche des informations d'aide, passez à l'étape 2.

   Linux, macOS, or Unix

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

Exécutez `exit` pour quitter la fenêtre de terminal actuelle et ouvrir une nouvelle fenêtre de terminal afin de récupérer les modifications apportées à l'environnement.

Windows

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

Exécutez `exit` pour quitter l'invite de commandes actuelle et ouvrir une nouvelle invite de commandes afin de récupérer les modifications apportées à l'environnement.

2. Créez et nommez un environnement virtuel.

   Linux, macOS, or Unix

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

   Vous pouvez également utiliser l'option `-p` pour spécifier une version spécifique de Python.

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

   Windows

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

3. Activez votre nouvel environnement virtuel.

   Linux, macOS, or Unix

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

   Windows

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

4. Effectuez AWS ParallelCluster l'installation dans votre environnement virtuel.

   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. Vérifiez qu'il AWS ParallelCluster est correctement installé.

   Linux, macOS, or Unix

   ```
   $ pcluster version
   2.11.9
   ```

   Windows

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

La commande `deactivate` vous permet de quitter l'environnement virtuel. Chaque fois que vous démarrez une session, vous devez réactiver l'environnement.

Pour effectuer une mise à niveau vers la dernière version de AWS ParallelCluster, réexécutez la commande d'installation.

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"
```

Installation AWS ParallelCluster sous Linux

Vous pouvez installer AWS ParallelCluster et ses dépendances sur la plupart des distributions Linux en utilisant `pip` un gestionnaire de paquets pour Python. Tout d'abord, déterminez si Python et `pip` sont installés :

1. Pour déterminer si votre version de Linux comprend Python et `pip`, exécutez `pip --version`.

```
$ pip --version
```

Si vous l'avez `pip` installé, passez à la rubrique Installer AWS ParallelCluster avec pip. Sinon, poursuivez avec l'étape 2

2. Pour déterminer si Python est installé, exécutez `python --version`.

```
$ python --version
```

Si Python 3 version 3.6+ ou Python 2 version 2.7 est installé, passez à la rubrique Installer AWS ParallelCluster avec pip. Sinon, installez Python, puis revenez à cette procédure pour installer `pip`.

3. Installez `pip` en utilisant le script fourni par Python Packaging Authority.
4. Utilisez la commande `curl` pour télécharger le script d'installation.

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

5. Exécutez le script avec Python pour télécharger et installer manuellement la version la plus récente de `pip` et des autres packages de prise en charge requis.

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

or

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

Lorsque vous incluez le commutateur `--user`, le script `pip` s'installe dans le chemin d'accès `~/.local/bin`.

6. Pour vérifier que le dossier qui contient `pip` fait partie de votre `PATH` variable, procédez comme suit :

   a. Recherchez le script de profil de votre shell dans votre dossier utilisateur. Si vous n'êtes pas certain du shell utilisé, exécutez `basename $SHELL`.

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

- Bash — `.bash_profile.profile`, ou `.bash_login`
- Zsh : `.zshrc`
- Tcsh —`.tcshrc`, ou `.cshrc .login`

b. Ajoutez une commande d'exportation à la fin de votre script de profil similaire à l'exemple suivant.

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

Cette commande ajoute le chemin d'accès, `~/.local/bin` dans cet exemple, devant la variable PATH actuelle.

c. Rechargez le profil dans la session en cours pour appliquer ces modifications.

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

7. Vérifiez que `pip` est installé correctement.

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

Sections

- Installation AWS ParallelCluster avec pip
- Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande
- Installation de Python sous Linux

Installation AWS ParallelCluster avec **`pip`**

`pip`À utiliser pour installer AWS ParallelCluster.

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

Lorsque vous utilisez le commutateur `--user`, `pip` installe AWS ParallelCluster sur `~/.local/bin`.

Vérifiez qu'il AWS ParallelCluster est correctement installé.

```
$ pcluster version
```

```
2.11.9
```

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

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

Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande

Après avoir effectué l'installation avec `pip`, vous allez peut-être devoir ajouter le fichier exécutable `pcluster` à la variable d'environnement `PATH` de votre système d'exploitation.

Pour vérifier le dossier dans lequel il `pip` est installé AWS ParallelCluster, exécutez la commande suivante.

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

Si vous avez omis le `--user` commutateur lors de l'installation AWS ParallelCluster, l'exécutable se trouve peut-être dans le `bin` dossier de votre installation Python. Si vous ne savez pas où est installé Python, exécutez cette commande.

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

Notez que la sortie peut être le chemin d'accès vers un lien symbolique, et non le fichier exécutable. Pour savoir où se situent les points du lien symbolique, exécutez `ls -al`.

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

S'il s'agit du même dossier que celui que vous avez ajouté au chemin d'accès à l'étape 3 dans Installation AWS ParallelCluster, vous avez terminé. Dans le cas contraire, vous devez effectuer à nouveau les étapes 3a à 3c, en ajoutant ce dossier supplémentaire au chemin.

Installation de Python sous Linux

Si votre distribution n'est pas fournie avec Python ou est fournie avec une version antérieure, installez Python avant d'installer `pip` et AWS ParallelCluster.

Pour installer Python 3 sous Linux

1. Vérifiez si Python est déjà installé.

```
$ python3 --version
```

or

```
$ python --version
```

> **Note**
>
> Si votre distribution Linux a été fournie avec Python, il se peut que vous ayez besoin d'installer le package développeur Python. Le package développeur inclut les en-têtes et les bibliothèques requises pour compiler les extensions et pour installer AWS ParallelCluster. Utilisez votre gestionnaire de package pour installer le package développeur. Il est généralement nommé `python-dev` ou `python-devel`.

2. Si Python 2.7 ou version ultérieure n'est pas installé, installez Python à l'aide du gestionnaire de package de votre distribution. Le nom de la commande et du package varie :

   - Sur les dérivés Debian, comme Ubuntu, utilisez `apt`.

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

   - Sur Red Hat et dérivés, utilisez `yum`.

     ```
     $ sudo yum install python3
     ```

   - Sur SUSE et dérivés, utilisez `zypper`.

     ```
     $ sudo zypper install python3
     ```

3. Ouvrez une invite de commande ou un shell et exécutez la commande suivante pour vérifier que Python est installé correctement.

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

Installation AWS ParallelCluster sur macOS

Sections

- Prérequis
- Installation AWS ParallelCluster sur macOS à l'aide de pip
- Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande

Prérequis

- Python 3 version 3.7+ ou Python 2 version 2.7

Vérifiez votre installation de Python.

```
$ python --version
```

Si Python n'est pas déjà installé sur votre ordinateur, ou si vous souhaitez installer une version différente de Python, suivez la procédure de la section Installation AWS ParallelCluster sous Linux.

Installation AWS ParallelCluster sur macOS à l'aide de pip

Vous pouvez également l'utiliser `pip` directement pour installer AWS ParallelCluster. Si vous disposez de `pip`, suivez les instructions décrites dans la rubrique d'installation principale. Exécutez `pip3 --version` pour déterminer si votre version de macOS comprend déjà Python et `pip3`.

```
$ pip3 --version
```

Pour installer AWS ParallelCluster sur macOS

1. Téléchargez et installez la dernière version de Python à partir de la page de téléchargement de Python.org.
2. Téléchargez et exécutez le script d'installation `pip3` fourni par Python Packaging Authority.

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

3. Utilisez votre appareil nouvellement installé `pip3` pour l'installer AWS ParallelCluster. Si vous utilisez Python version 3+, nous vous recommandons d'utiliser la commande `pip3`.

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

4. Vérifiez qu'il AWS ParallelCluster est correctement installé.

```
$ pcluster version
2.11.9
```

Si le programme est introuvable, ajoutez-le à votre chemin de ligne de commande.

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

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

Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande

Après avoir effectué l'installation à l'aide de `pip`, vous allez peut-être devoir ajouter le programme `pcluster` à la variable d'environnement `PATH` de votre système d'exploitation. L'emplacement du programme dépend de l'endroit où Python est installé.

Example AWS ParallelCluster emplacement d'installation - macOS avec Python 3.6 et **`pip`** (mode utilisateur)

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

Remplacez votre version de Python par celle de l'exemple précédent.

Si vous ne savez pas où est installé Python, exécutez `which python`.

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

La sortie pourrait être le chemin d'accès à un lien symbolique, et non le chemin d'accès au programme réel. Exécutez `ls -al` pour voir vers où il pointe.

```
$ 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` installe les programmes dans le dossier qui contient l'application Python. Ajoutez ce dossier à votre variable `PATH`.

Pour modifier votre **`PATH`** variable (Linux, macOS ou Unix)

1. Recherchez le script de profil de votre shell dans votre dossier utilisateur. Si vous n'êtes pas certain du shell utilisé, exécutez `echo $SHELL`.

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

   - Bash — `.bash_profile.profile`, ou `.bash_login`
   - Zsh : `.zshrc`
   - Tcsh — `.tcshrc.cshrc`, ou `.login`
2. Ajoutez une commande d'exportation à votre script de profil.

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

   Cette commande ajoute un chemin d'accès, `~/.local/bin` dans cet exemple, à la variable `PATH` actuelle.
3. Chargez le profil dans la session en cours.

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

Installation AWS ParallelCluster sous Windows

Vous pouvez l'installer AWS ParallelCluster sous Windows en utilisant`pip`, qui est un gestionnaire de paquets pour Python. Si vous disposez déjà de `pip`, suivez les instructions décrites dans la rubrique d'installation principale.

Sections

- Installation AWS ParallelCluster à l'aide de Python et pip sous Windows
- Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande

Installation AWS ParallelCluster à l'aide de Python et **pip** sous Windows

La Python Software Foundation fournit des programmes d'installation pour Windows qui incluent `pip`.

Pour installer Python et **pip** (Windows)

1. Téléchargez le programme d'installation Windows x86-64 de Python à partir de la page de téléchargements de Python.org.
2. Exécutez le programme d'installation.
3. Choisissez Add Python 3 to PATH.
4. Choisissez Install Now (Installer maintenant).

Le programme d'installation installe Python dans votre dossier d'utilisateur et ajoute ses répertoires de programme à votre chemin d'utilisateur.

Pour installer AWS ParallelCluster avec **pip3** (Windows)

Si vous utilisez Python version 3+, nous vous recommandons d'utiliser la commande `pip3`.

1. Ouvrez l'Invite de commandes Windows dans le menu Démarrer.
2. Utilisez les commandes suivantes pour vérifier que Python et `pip` sont tous deux installés correctement.

```
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. Installez à AWS ParallelCluster l'aide de`pip`.

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

4. Vérifiez qu'il AWS ParallelCluster est correctement installé.

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

Pour effectuer une mise à niveau vers la dernière version, exécutez à nouveau la commande d'installation.

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

Ajoutez l' AWS ParallelCluster exécutable au chemin de votre ligne de commande

Après avoir installé AWS ParallelCluster avec`pip`, ajoutez le `pcluster` programme à la variable d'`PATH`environnement de votre système d'exploitation.

Vous pouvez trouver où le programme `pcluster` est installé en exécutant la commande suivante.

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

Si cette commande ne renvoie aucun résultat, vous devez ajouter le chemin d'accès manuellement. Utilisez la ligne de commande ou l'Explorateur Windows pour découvrir où il est installé sur votre ordinateur. Les chemins incluent généralement :

- Python 3 et **pip3** — `C:\Python38\Scripts\`
- Python 3 et option **pip3** --user — `%APPDATA%\Python\Python38\Scripts`

> **Note**
>
> Les noms de dossiers qui incluent les numéros de version peuvent varier. Les exemples précédents montrent Python38. Remplacez au besoin par le numéro de version que vous utilisez.

Pour modifier votre variable PATH (Windows)

1. Appuyez sur la touche Windows et entrez **`environment variables`**.
2. Sélectionnez Modifier les variables d'environnement pour votre compte.
3. Choisissez PATH, puis Modifier.
4. Ajoutez le chemin au champ de Valeur de variable. Par exemple : ***`C:\new\path`***
5. Choisissez OK deux fois pour appliquer les nouveaux paramètres.

6. Fermez toute invite de commande en cours d'exécution et rouvrez une invite de commande.

Configuration AWS ParallelCluster

Après l'installation AWS ParallelCluster, effectuez les étapes de configuration suivantes.

Vérifiez que votre AWS compte possède un rôle qui inclut les autorisations nécessaires pour exécuter la `pcluster` CLI. Pour de plus amples informations, veuillez consulter AWS ParallelCluster exemple de politiques relatives aux instances et aux utilisateurs.

Configurez vos AWS informations d'identification. Pour plus d'informations, consultez Configuration de l' AWS CLI dans le Guide de l'utilisateur de l'AWS CLI .

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

L' Région AWS endroit où le cluster est lancé doit comporter au moins une paire de EC2 clés Amazon. Pour plus d'informations, consultez les paires de EC2 clés Amazon dans le guide de EC2 l'utilisateur Amazon.

```
$ pcluster configure
```

L'assistant de configuration vous invite à entrer toutes les informations nécessaires pour créer votre cluster. Les détails de la séquence diffèrent lors de l'utilisation en AWS Batch tant que planificateur par rapport à l'utilisation Slurm. Pour plus d'informations sur la configuration d'un cluster, consultezConfiguration.

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs. Vous pouvez continuer à les utiliser dans les versions antérieures à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.

Slurm

Dans la liste des Région AWS identifiants valides, choisissez l' Région AWS endroit où vous souhaitez que votre cluster s'exécute.

> **Note**
>
> La liste des Régions AWS éléments affichés est basée sur la partition de votre compte et inclut uniquement Régions AWS les partitions activées pour votre compte. Pour plus d'informations sur l'activation Régions AWS de votre compte, consultez la section Gestion Régions AWS dans le Références générales AWS. L'exemple présenté provient de la partition AWS globale. Si votre compte se trouve dans la AWS GovCloud (US) partition, seule Régions AWS cette partition est répertoriée (`gov-us-east-1`et`gov-us-west-1`). De même, si votre compte se trouve dans la partition AWS Chine, uniquement `cn-north-1` et `cn-northwest-1` s'affiche. Pour la liste complète des produits Régions AWS pris en charge par AWS ParallelCluster, voirRégions prises en charge.

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

Choisissez le planificateur à utiliser avec votre cluster.

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

Choisissez le système d'exploitation.

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

> **Note**
>
> Support pour `alinux2` a été ajouté dans la AWS ParallelCluster version 2.6.0.

Les tailles minimale et maximale du cluster de nœuds de calcul sont entrées. Elles sont mesurées en nombre d'instances.

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

Les types d'instance des nœuds de tête et de calcul sont saisis. Pour les types d'instance, les limites d'instance de votre compte sont suffisamment grandes pour répondre à vos besoins. Pour plus d'informations, consultez la section Limites des instances à la demande dans le guide de EC2 l'utilisateur Amazon.

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

La paire de clés est sélectionnée parmi les paires de clés enregistrées auprès d'Amazon EC2 dans le champ sélectionné Région AWS.

```
Allowed values for EC2 Key Pair Name:
```

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

Une fois les étapes précédentes terminées, décidez d'utiliser un VPC existant ou de laisser AWS ParallelCluster créer un VPC pour vous. Si vous ne disposez pas d'un VPC correctement configuré, AWS ParallelCluster vous pouvez en créer un nouveau. Il utilise soit les nœuds de tête et de calcul du même sous-réseau public, soit uniquement le nœud principal d'un sous-réseau public avec tous les nœuds d'un sous-réseau privé. Il est possible d'atteindre votre limite de nombre de personnes VPCs dans un Région AWS. La limite par défaut est de cinq VPCs pour chacune Région AWS. Pour plus d'informations sur cette limite et sur la manière de demander une augmentation, consultez la section VPC et sous-réseaux dans le guide de l'utilisateur Amazon VPC.

Si vous autorisez la AWS ParallelCluster création d'un VPC, vous devez décider si tous les nœuds doivent se trouver dans un sous-réseau public.

> **Important**
>
> VPCs créé par AWS ParallelCluster ne pas activer les journaux de flux VPC par défaut. Les journaux de flux VPC vous permettent de capturer des informations sur le trafic IP à destination et en provenance des interfaces réseau de votre. VPCs Pour plus d'informations, consultez la rubrique Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

> **Note**
>
> Si vous le souhaitez`1. Master in a public subnet and compute fleet in a private subnet`, AWS ParallelCluster crée une passerelle NAT qui entraîne des coûts supplémentaires, même si vous spécifiez des ressources de niveau gratuit.

```
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
```

Si vous ne créez pas de nouveau VPC, vous devez sélectionner un VPC existant.

Si vous choisissez de AWS ParallelCluster créer le VPC, notez l'ID du VPC afin de pouvoir l'utiliser AWS CLI pour le supprimer ultérieurement.

```
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
```

Une fois le VPC sélectionné, vous devez décider d'utiliser les sous-réseaux existants ou d'en créer de nouveaux.

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

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

AWS Batch

Dans la liste des Région AWS identifiants valides, choisissez l' Région AWS endroit où vous souhaitez que votre cluster s'exécute.

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

Choisissez le planificateur à utiliser avec votre cluster.

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

Lorsque `awsbatch` est sélectionné comme planificateur, `alinux2` est utilisé comme système d'exploitation.

Les tailles minimale et maximale du cluster de nœuds de calcul sont entrées. Ceci est mesuré en CPUs v.

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

Le type d'instance du nœud principal est saisi. Lors de l'utilisation du planificateur `awsbatch`, les nœuds de calcul utilisent un type d'instance `optimal`.

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

La paire de EC2 clés Amazon est sélectionnée parmi les paires de clés enregistrées auprès d'Amazon EC2 dans le champ sélectionné Région AWS.

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

Décidez si vous souhaitez utiliser l'existant VPCs ou laisser AWS ParallelCluster créer VPCs pour vous. Si vous ne disposez pas d'un VPC correctement configuré, AWS ParallelCluster vous pouvez en créer un nouveau. Il utilise soit les nœuds de tête et de calcul du même sous-réseau public, soit uniquement le nœud principal d'un sous-réseau public avec tous les nœuds d'un sous-réseau privé. Il est possible d'atteindre votre limite de nombre de personnes VPCs dans un

Région AWS. Le nombre par défaut VPCs est cinq. Pour plus d'informations sur cette limite et sur la manière de demander une augmentation, consultez la section VPC et sous-réseaux dans le guide de l'utilisateur Amazon VPC.

> **Important**
>
> VPCs créé par AWS ParallelCluster ne pas activer les journaux de flux VPC par défaut. Les journaux de flux VPC vous permettent de capturer des informations sur le trafic IP à destination et en provenance des interfaces réseau de votre. VPCs Pour plus d'informations, consultez la rubrique Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

Si vous autorisez la AWS ParallelCluster création d'un VPC, décidez si tous les nœuds doivent se trouver dans un sous-réseau public.

> **Note**
>
> Si vous le souhaitez`1. Master in a public subnet and compute fleet in a private subnet`, AWS ParallelCluster crée une passerelle NAT qui entraîne des coûts supplémentaires, même si vous spécifiez des ressources de niveau gratuit.

```
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
```

Si vous ne créez pas de nouveau VPC, vous devez sélectionner un VPC existant.

Si vous choisissez de AWS ParallelCluster créer le VPC, notez l'ID du VPC afin de pouvoir l'utiliser AWS CLI pour le supprimer ultérieurement.

```
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
```

Une fois le VPC sélectionné, décidez si vous souhaitez utiliser des sous-réseaux existants ou en créer de nouveaux.

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

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

Lorsque vous avez terminé les étapes précédentes, un simple cluster est lancé dans un VPC. Le VPC utilise un sous-réseau existant qui prend en charge les adresses IP publiques. La table de routage pour le sous-réseau est`0.0.0.0/0 => igw-xxxxxx`. Notez les conditions suivantes :

- Le VPC doit comporter les paramètres `DNS Resolution = yes` et `DNS Hostnames = yes`.
- Le VPC doit également avoir des options DHCP avec la bonne `domain-name` valeur pour. Région AWS Le jeu d'options DHCP par défaut spécifie déjà les éléments requis AmazonProvidedDNS. Si vous spécifiez plusieurs serveurs de noms de domaine, consultez les ensembles d'options DHCP dans le guide de l'utilisateur Amazon VPC. Lorsque vous utilisez des sous-réseaux privés, utilisez une passerelle NAT ou un proxy interne pour permettre l'accès Web aux nœuds de calcul. Pour de plus amples informations, veuillez consulter Configurations réseau.

Une fois que tous les paramètres contiennent les valeurs valides, vous pouvez lancer le cluster en exécutant la commande create.

```
$ pcluster create mycluster
```

Une fois que le cluster a atteint le statut « CREATE_COMPLETE », vous pouvez vous y connecter à l'aide de vos paramètres client SSH habituels. Pour plus d'informations sur la connexion aux EC2 instances Amazon, consultez le guide de EC2 l'utilisateur dans le guide de EC2 l'utilisateur Amazon.

Pour supprimer le cluster, exécutez la commande suivante.

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

Pour supprimer les ressources réseau du VPC, vous pouvez supprimer la pile CloudFormation réseau. Le nom de la pile commence par »parallelclusternetworking-» et contient l'heure de création au format « YYYYMMDDHHMMSS ». Vous pouvez répertorier les piles à l'aide de la commande list-stacks.

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

La pile peut être supprimée à l'aide de la commande delete-stack.

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

Le VPC `pcluster configure` créé pour vous n'est pas créé dans la pile CloudFormation réseau. Vous pouvez supprimer ce VPC manuellement dans la console ou à l'aide du. AWS CLI

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

Bonnes pratiques

Bonnes pratiques : sélection du type d'instance principale

Bien que le nœud principal n'exécute aucune tâche, ses fonctions et son dimensionnement sont essentiels aux performances globales du cluster.

Lorsque vous choisissez le type d'instance à utiliser pour votre nœud maître, vous souhaitez évaluer les éléments suivants :

- Taille du cluster : le nœud principal orchestre la logique de dimensionnement du cluster et est chargé d'attacher les nouveaux nœuds au planificateur. Si vous devez augmenter ou diminuer le cluster d'un nombre considérable de nœuds, vous devez donner au nœud maître une capacité de calcul supplémentaire.
- Systèmes de fichiers partagés : lorsque vous utilisez des systèmes de fichiers partagés pour partager des artefacts entre des nœuds de calcul et le nœud principal, tenez compte du fait que

le maître est le nœud exposant le serveur NFS. C'est pourquoi vous souhaitez choisir un type d'instance doté d'une bande passante réseau suffisante et d'une bande passante Amazon EBS dédiée suffisante pour gérer vos flux de travail.

Bonnes pratiques : performances du réseau

Trois conseils couvrent l'ensemble des possibilités d'amélioration de la communication réseau.

- Groupe de placement : un groupe de placement de clusters est un regroupement logique d'instances au sein d'une même zone de disponibilité. Pour plus d'informations sur les groupes de placement, consultez la section Groupes de placement dans le guide de EC2 l'utilisateur Amazon. Vous pouvez configurer le cluster pour utiliser votre propre groupe de placement `placement_group = your-placement-group-name` ou laisser AWS ParallelCluster créer un groupe de placement avec la `"compute"` stratégie avec`placement_group = DYNAMIC`. Pour plus d'informations, voir `placement_group` pour le mode de file d'attente multiple et `placement_group` pour le mode de file d'attente unique.
- Mise en réseau améliorée : pensez à choisir un type d'instance compatible avec la mise en réseau améliorée. Pour plus d'informations, consultez la section Mise en réseau améliorée sous Linux dans le guide de EC2 l'utilisateur Amazon.
- Adaptateur Elastic Fabric : pour prendre en charge des niveaux élevés de communication entre instances évolutives, pensez à choisir des interfaces réseau EFA pour votre réseau. Le matériel de contournement du système d'exploitation (OS) personnalisé de l'EFA améliore les communications entre instances grâce à l'élasticité et à la flexibilité à la demande du AWS cloud. Pour configurer un seul Slurm file d'attente de cluster pour utiliser EFA, définie`enable_efa = true`. Pour plus d'informations sur l'utilisation d'EFA avec AWS ParallelCluster, consultez Elastic Fabric Adapter et `enable_efa`. Pour plus d'informations sur EFA, consultez Elastic Fabric Adapter dans le guide de EC2 l'utilisateur Amazon pour les instances Linux.
- Bande passante de l'instance : la bande passante varie en fonction de la taille de l'instance. Pensez à choisir le type d'instance qui répond le mieux à vos besoins. Consultez les instances optimisées pour Amazon EBS et les types de volumes Amazon EBS dans le guide de l'utilisateur Amazon EC2 .

Bonnes pratiques : alertes budgétaires

Pour gérer les coûts AWS ParallelCluster des ressources, nous vous recommandons d'utiliser AWS Budgets des actions pour créer un budget et de définir des alertes de seuil budgétaire pour

les AWS ressources sélectionnées. Pour plus d'informations, consultez la section Configuration d'une action budgétaire dans le Guide de AWS Budgets l'utilisateur. Vous pouvez également utiliser Amazon CloudWatch pour créer une alarme de facturation. Pour plus d'informations, consultez la section Création d'une alarme de facturation pour surveiller vos AWS frais estimés.

Bonnes pratiques : déplacement d'un cluster vers une nouvelle version AWS ParallelCluster mineure ou un correctif

Actuellement, chaque version AWS ParallelCluster mineure est autonome avec sa `pcluster` CLI. Pour déplacer un cluster vers une nouvelle version mineure ou corrective, vous devez recréer le cluster à l'aide de la CLI de la nouvelle version.

Pour optimiser le processus de déplacement d'un cluster vers une nouvelle version mineure ou pour enregistrer vos données de stockage partagées pour d'autres raisons, nous vous recommandons d'appliquer les meilleures pratiques suivantes.

- Enregistrez les données personnelles dans des volumes externes, tels que Amazon EFS et FSx pour Lustre. De cette manière, vous pouvez facilement déplacer les données d'un cluster à un autre.
- Créez des systèmes de stockage partagés des types répertoriés ci-dessous à l'aide du AWS CLI ou AWS Management Console :
  - `[ebs]` Section
  - `[efs]` Section
  - `[fsx]` Section

  Ajoutez-les à la nouvelle configuration du cluster en tant que systèmes de fichiers existants. Ils sont ainsi préservés lorsque vous supprimez le cluster et peuvent être attachés à un nouveau cluster. Les systèmes de stockage partagés sont généralement payants, qu'ils soient connectés ou détachés d'un cluster.

  Nous vous recommandons d'utiliser les systèmes de fichiers Amazon EFS ou Amazon FSx for Lustre, car ils peuvent être attachés à plusieurs clusters en même temps et vous pouvez les associer au nouveau cluster avant de supprimer l'ancien cluster. Pour plus d'informations, consultez les sections Montage des systèmes de fichiers Amazon EFS dans le guide de l'utilisateur Amazon EFS et Accès aux systèmes de fichiers Amazon FSx for Lustre dans le guide de l'utilisateur Amazon FSx for Lustre Lustre.

- Utilisez des actions d'amorçage personnalisées pour personnaliser vos instances plutôt qu'une AMI personnalisée. Cela optimise le processus de création car il n'est pas nécessaire de créer une nouvelle AMI personnalisée pour chaque nouvelle version.
- Séquence recommandée.
  1. Mettez à jour la configuration du cluster pour utiliser les définitions de systèmes de fichiers existantes.
  2. Vérifiez la `pcluster` version et mettez-la à jour si nécessaire.
  3. Créez et testez le nouveau cluster.
     - Assurez-vous que vos données sont disponibles dans le nouveau cluster.
     - Assurez-vous que votre application fonctionne dans le nouveau cluster.
  4. Si votre nouveau cluster est entièrement testé et opérationnel et que vous êtes certain de ne pas utiliser l'ancien cluster, supprimez-le.

Passage de CfnCluster à AWS ParallelCluster

AWS ParallelCluster est une version améliorée de CfnCluster.

Si vous l'utilisez actuellement CfnCluster, nous vous recommandons de l'utiliser à la AWS ParallelCluster place et d'en créer de nouveaux. Même si vous pouvez continuer à l'utiliser CfnCluster, il n'est plus en cours de développement et aucune nouvelle fonctionnalité ne sera ajoutée.

Les principales différences entre CfnCluster et AWS ParallelCluster sont décrites dans les sections suivantes.

AWS ParallelCluster La CLI gère un ensemble différent de clusters

Les clusters créés avec la `cfncluster` CLI ne peuvent pas être gérés avec la `pcluster` CLI. Les commandes suivantes ne fonctionnent pas sur les clusters créés par CfnCluster :

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

Pour gérer les clusters que vous avez créés avec CfnCluster, vous devez utiliser la `cfncluster` CLI.

Si vous avez besoin d'un CfnCluster package pour gérer vos anciens clusters, nous vous recommandons de l'installer et de l'utiliser à partir d'un environnement virtuel Python.

AWS ParallelCluster et CfnCluster utilisez différentes politiques personnalisées IAM

Les politiques IAM personnalisées qui étaient précédemment utilisées pour la création de CfnCluster clusters ne peuvent pas être utilisées avec AWS ParallelCluster. Si vous avez besoin de politiques personnalisées pour AWS ParallelCluster, vous devez en créer de nouvelles. Consultez le AWS ParallelCluster guide.

AWS ParallelCluster et CfnCluster utilisez différents fichiers de configuration

Le fichier AWS ParallelCluster de configuration se trouve dans le `~/.parallelcluster` dossier. Le fichier de configuration CfnCluster se trouve dans le dossier `~/.cfncluster`.

Si vous souhaitez utiliser un fichier CfnCluster de configuration existant avec AWS ParallelCluster, vous devez effectuer les actions suivantes :

1. Déplacer le fichier de configuration de `~/.cfncluster/config` vers `~/.parallelcluster/config`.
2. Si vous utilisez le paramètre de configuration `extra_json`, modifiez-le comme indiqué.

CfnCluster réglage :

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

AWS ParallelCluster réglage :

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

Dans AWS ParallelCluster, les ganglions sont désactivés par défaut

Dans AWS ParallelCluster, ganglia est désactivé par défaut. Pour activer Ganglia, procédez comme suit :

1. Définissez le paramètre `extra_json` comme illustré :

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

2. Modifiez le groupe de sécurité principal pour autoriser les connexions au port 80.

Le groupe de sécurité `parallelcluster-<CLUSTER_NAME>-MasterSecurityGroup-<xxx>` doit être modifié en ajoutant une nouvelle règle de groupe de sécurité permettant d'autoriser la connexion entrante au port 80 à partir de votre adresse IP publique. Pour plus d'informations, consultez la section Ajouter des règles à un groupe de sécurité dans le guide de EC2 l'utilisateur Amazon.

Régions prises en charge

AWS ParallelCluster la version 2.x est disponible dans les versions suivantes : Régions AWS

Nom de la région	Région
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1
US West (N. California)	us-west-1
US West (Oregon)	us-west-2
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1
Asia Pacific (Seoul)	ap-northeast-2
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asia Pacific (Tokyo)	ap-northeast-1
Canada (Central)	ca-central-1
China (Beijing)	cn-north-1
China (Ningxia)	cn-northwest-1

Nom de la région	Région
Europe (Francfort)	eu-central-1
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Milan)	eu-south-1
Europe (Paris)	eu-west-3
Europe (Stockholm)	eu-north-1
Moyen-Orient (Bahreïn)	me-south-1
Amérique du Sud (São Paulo)	sa-east-1
AWS GovCloud (USA Est)	us-gov-east-1
AWS GovCloud (US-Ouest)	us-gov-west-1

En utilisant AWS ParallelCluster

Rubriques

- Configurations réseau
- Actions d'amorçage personnalisées
- Travailler avec Amazon S3
- Utilisation de instances Spot
- AWS Identity and Access Management rôles dans AWS ParallelCluster
- Planificateurs pris en charge par AWS ParallelCluster
- AWS ParallelCluster ressources et balisage
- Tableau de CloudWatch bord Amazon
- Intégration à Amazon CloudWatch Logs
- Elastic Fabric Adapter
- Solutions Intel Select
- Activation d'Intel MPI
- Spécifications de plateforme HPC Intel
- Bibliothèques de performances Arm
- Connectez-vous au nœud principal via Amazon DCV
- Utiliser pcluster update
- Corriger les AMI et remplacer les EC2 instances

Configurations réseau

AWS ParallelCluster utilise Amazon Virtual Private Cloud (VPC) pour la mise en réseau. Le VPC fournit une plate-forme réseau flexible et configurable sur laquelle vous pouvez déployer des clusters.

Le VPC doit avoir `DNS Resolution = yes` `DNS Hostnames = yes` et des options DHCP avec le nom de domaine correct pour la région. Le jeu d'options DHCP par défaut spécifie déjà le AmazonProvidedDNS requis. Si vous spécifiez plusieurs serveurs de noms de domaine, consultez les ensembles d'options DHCP dans le guide de l'utilisateur Amazon VPC.

AWS ParallelCluster prend en charge les configurations de haut niveau suivantes :

- Un sous-réseau pour les nœuds de tête et de calcul.
- Deux sous-réseaux, avec le nœud principal dans un sous-réseau public et des nœuds de calcul dans un sous-réseau privé. Les sous-réseaux peuvent être nouveaux ou existants.

Toutes ces configurations peuvent fonctionner avec ou sans adressage IP public. AWS ParallelCluster peut également être déployé pour utiliser un proxy HTTP pour toutes les AWS requêtes. Les combinaisons de ces configurations se traduisent par de nombreux scénarios de déploiement. Par exemple, vous pouvez configurer un sous-réseau public unique avec tous les accès via Internet. Vous pouvez également configurer un réseau entièrement privé à l'aide AWS Direct Connect d'un proxy HTTP pour l'ensemble du trafic.

Consultez les schémas d'architecture suivants pour les illustrations de certains de ces scénarios :

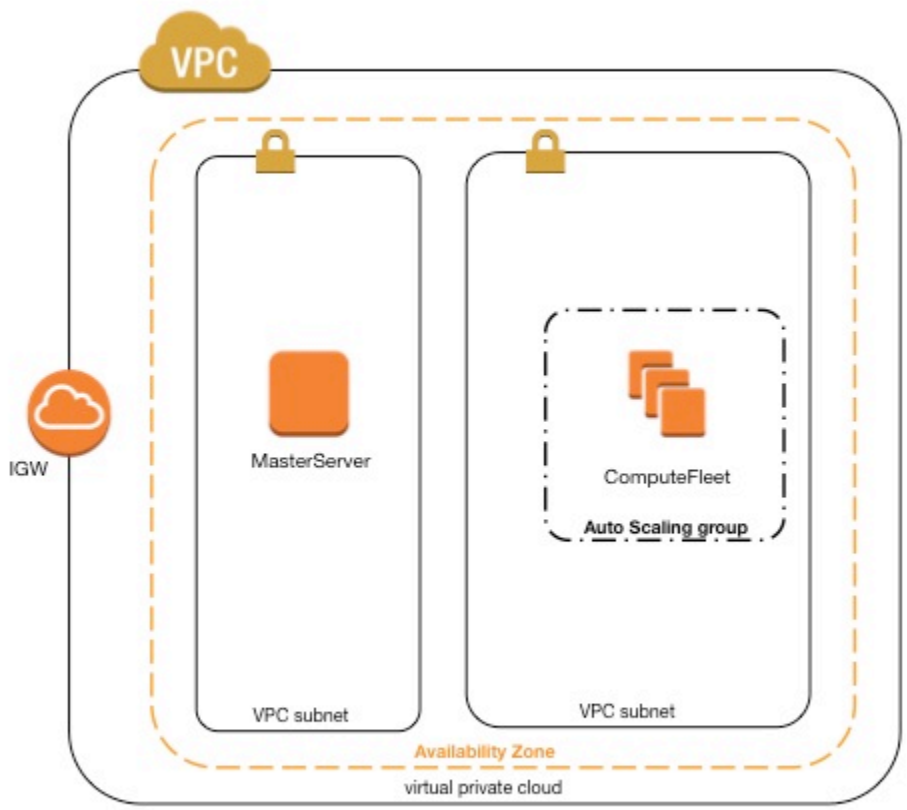
AWS ParallelCluster dans un seul sous-réseau public

La configuration de cette architecture nécessite les paramètres suivants :

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

Le paramètre `use_public_ips` ne peut pas être défini sur `false`, car la passerelle Internet requiert que toutes les instances disposent d'une adresse IP unique au niveau mondial. Pour plus d'informations, consultez la section Activation de l'accès à Internet dans le guide de l'utilisateur Amazon VPC.

AWS ParallelCluster en utilisant deux sous-réseaux



La configuration pour créer un nouveau sous-réseau privé pour les instances de calcul nécessite les paramètres suivants :

Notez que toutes les valeurs ne sont fournies qu'à titre d'exemple.

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

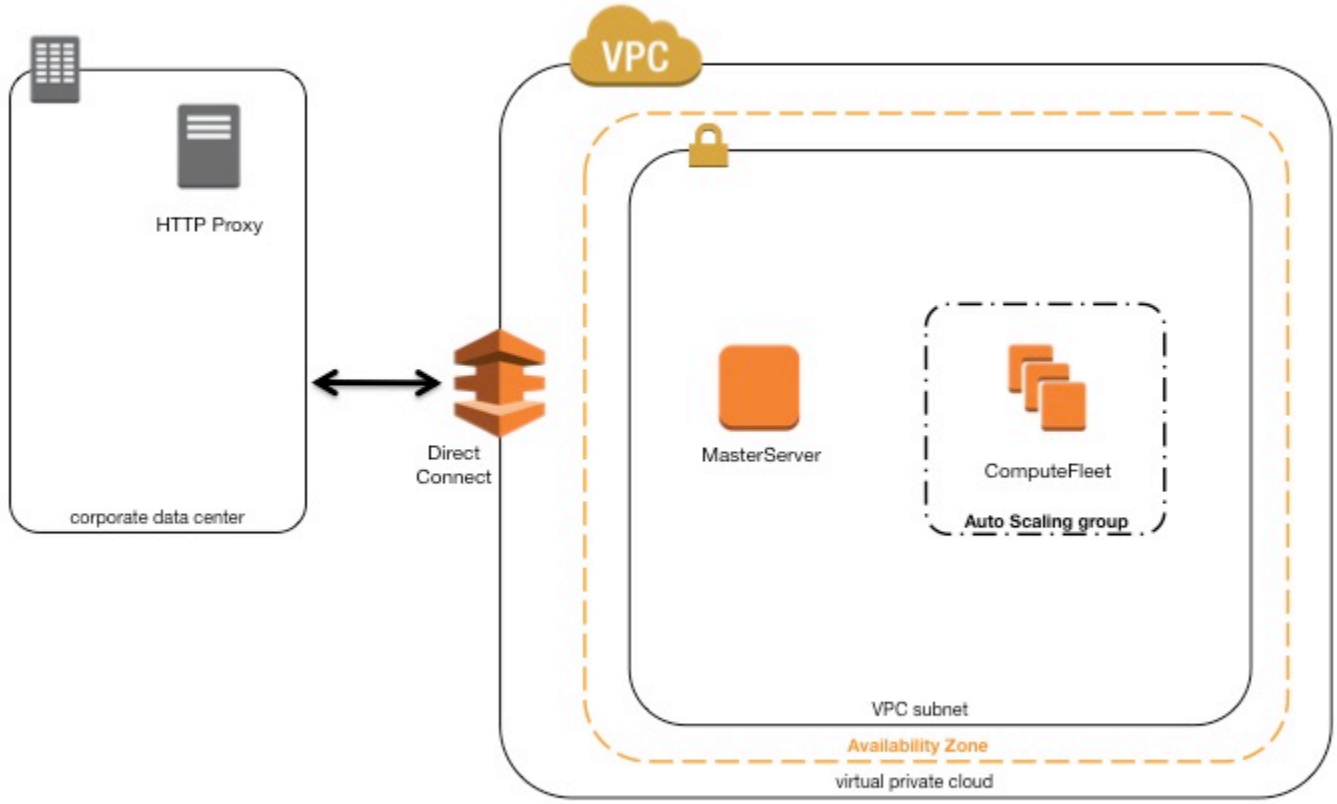
La configuration pour utiliser un réseau privé existant nécessite les paramètres suivants :

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

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

Ces deux configurations nécessitent une passerelle NAT ou un proxy interne pour permettre l'accès Web aux instances de calcul.

AWS ParallelCluster dans un seul sous-réseau privé connecté à l'aide de AWS Direct Connect



La configuration de cette architecture nécessite les paramètres suivants :

```
[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
```

Lorsque `use_public_ips` est défini sur `false`, le VPC doit être correctement configuré pour utiliser le proxy pour l'ensemble du trafic. L'accès au Web est requis pour les nœuds de tête et de calcul.

AWS ParallelCluster avec **`awsbatch`** planificateur

Lorsque vous l'utilisez `awsbatch` comme type de planificateur, AWS ParallelCluster crée un environnement informatique AWS Batch géré. L' AWS Batch environnement prend en charge la gestion des instances de conteneur Amazon Elastic Container Service (Amazon ECS), qui sont lancées dans le. `compute_subnet` AWS Batch Pour fonctionner correctement, les instances de conteneur Amazon ECS ont besoin d'un accès réseau externe pour communiquer avec le point de terminaison du service Amazon ECS. Cela se traduit par les scénarios suivants :

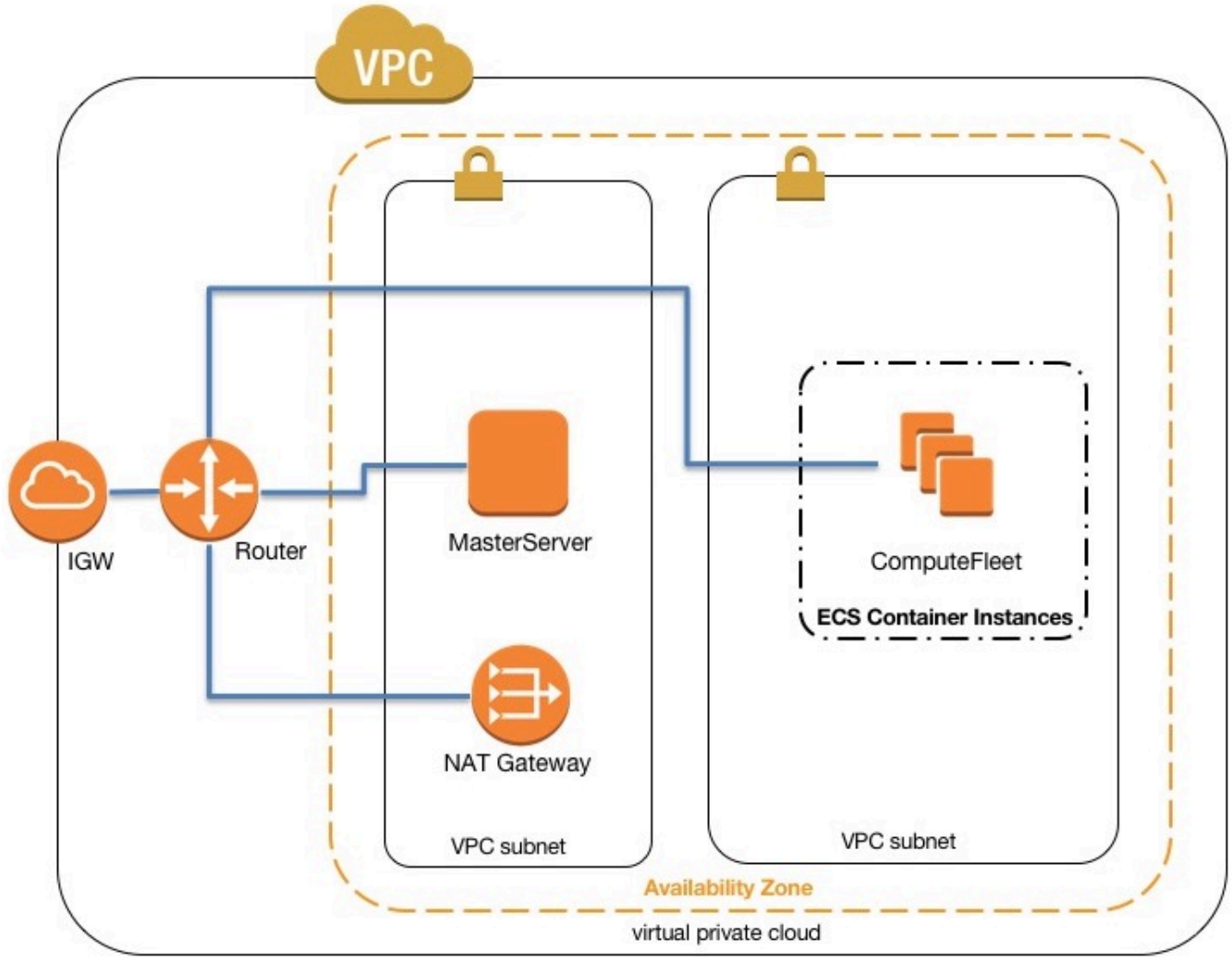
- Il `compute_subnet` utilise une passerelle NAT pour accéder à Internet. (Nous avons recommandé cette approche.)
- Les instances lancées dans `compute_subnet` ont des adresses IP publiques et peuvent accéder à Internet via une passerelle Internet.

De plus, si vous êtes intéressé par les jobs parallèles sur plusieurs nœuds (voir la AWS Batch documentation) :

AWS Batch les tâches parallèles à nœuds multiples utilisent le mode `awsvpc` réseau Amazon ECS, qui confère à vos conteneurs de tâches parallèles à nœuds multiples les mêmes propriétés réseau que les instances Amazon EC2 . Chaque conteneur de tâche parallèle à plusieurs nœuds obtient sa propre interface réseau Elastic, une adresse IP privée principale et un nom d'hôte DNS interne. L'interface réseau est créée dans le même sous-réseau Amazon VPC que sa ressource de calcul hôte. Tous les groupes de sécurité appliqués à vos ressources de calcul lui sont également appliqués.

Lorsque vous utilisez Amazon ECS Task Networking, le mode `awsvpc` réseau ne fournit pas d'interfaces réseau élastiques avec des adresses IP publiques pour les tâches utilisant le type de EC2 lancement Amazon. Pour accéder à Internet, les tâches utilisant le type de EC2 lancement Amazon doivent être lancées dans un sous-réseau privé configuré pour utiliser une passerelle NAT.

Vous devez configurer une passerelle NAT afin de permettre au cluster d'exécuter des tâches parallèles sur plusieurs nœuds.



Pour plus d’informations, consultez les rubriques suivantes :

- AWS Batch environnements informatiques gérés
- AWS Batch tâches parallèles sur plusieurs nœuds
- Mise en réseau des tâches Amazon ECS avec le mode `awsvpc` réseau

Actions d'amorçage personnalisées

AWS ParallelCluster peut exécuter du code arbitraire avant (pré-installation) ou après (post-installation) l'action d'amorçage principale lors de la création du cluster. Dans la plupart des cas, ce

code est stocké dans Amazon Simple Storage Service (Amazon S3) et accessible via une connexion HTTPS. Le code est exécuté en tant que root et peut être écrit dans n'importe quel langage de script pris en charge par le système d'exploitation du cluster. Le code est souvent en Bash ou en Python.

Les actions de préinstallation sont appelées avant le lancement de toute action de démarrage du déploiement du cluster, telle que la configuration de NAT, d'Amazon Elastic Block Store (Amazon EBS) ou du planificateur. Certaines actions de préinstallation incluent la modification du stockage, l'ajout d'utilisateurs supplémentaires et l'ajout de packages.

Les actions de post-installation sont appelées une fois les processus d'amorçage du cluster terminés. Les actions postérieures à l'installation sont les dernières actions effectuées avant qu'une instance ne soit considérée comme entièrement configurée et terminée. Certaines actions post-installation incluent la modification des paramètres du planificateur, la modification du stockage et la modification des packages.

Vous pouvez transmettre des arguments aux scripts en les spécifiant lors de la configuration. Pour cela, vous les transmettez entre guillemets aux actions de pré-installation ou de post-installation.

Si une action de pré-installation ou de post-installation échoue, le bootstrap de l'instance échoue également. Le succès est signalé par un code de sortie égal à zéro (0). Tout autre code de sortie indique que le bootstrap de l'instance a échoué.

Vous pouvez faire la différence entre les nœuds Running Head et les nœuds de calcul. Recherchez le `/etc/parallelcluster/cfnconfig` fichier et évaluez les variables d'`cfn_node_type`environnement dont les valeurs sont respectivement `MasterServer` « » et `ComputeFleet` « » pour les nœuds de tête et de calcul.

```
#!/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
```

Configuration

Les paramètres de configuration suivants sont utilisés pour définir les actions de pré-installation et de post-installation, ainsi que les arguments.

```
# 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
```

Arguments

Les deux premiers arguments, `$0` et `$1`, sont réservés pour le nom du script et l'URL.

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

exemple

Voici quelques étapes permettant de créer un simple script de post-installation qui installe les packages R dans un cluster.

1. Créez un script.

```
#!/bin/bash

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

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

2. Téléchargez le script avec les autorisations appropriées sur Amazon S3. Si les autorisations de lecture publiques ne vous conviennent pas, utilisez l'un `s3_read_resource` ou l'autre `s3_read_write_resource` des paramètres pour accorder l'accès. Pour de plus amples informations, veuillez consulter Travailler avec Amazon S3.

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

> **Important**
>
> Si le script a été modifié sous Windows, les fins de ligne doivent passer de CRLF à LF avant que le script ne soit chargé sur Amazon S3.

3. Mettez à jour la AWS ParallelCluster configuration pour inclure la nouvelle action de post-installation.

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

Si le bucket ne dispose pas d'une autorisation de lecture publique, `s3` utilisez-le comme protocole URL.

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

4. Lancement du cluster

```
$ pcluster create mycluster
```

5. Vérifiez la sortie.

```
$ 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
```

Travailler avec Amazon S3

Pour autoriser les ressources du cluster à accéder aux compartiments Amazon S3, spécifiez le compartiment ARNs dans les `s3_read_write_resource` paramètres `s3_read_resource` et de la AWS ParallelCluster configuration. Pour plus d'informations sur le contrôle de l'accès avec AWS ParallelCluster, consultezAWS Identity and Access Management rôles dans 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/*
```

Les deux paramètres acceptent un ARN Amazon S3 valide `*` ou un ARN Amazon S3 valide. Pour plus d'informations sur la spécification d'Amazon S3 ARNs, consultez le format ARN Amazon S3 dans le Références générales AWS.

Exemples

L'exemple suivant vous donne un accès en lecture à n'importe quel objet du compartiment Amazon S3 my_corporate_bucket.

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

Cet exemple suivant vous donne un accès en lecture au compartiment, mais ne vous permet pas de lire les objets du compartiment.

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

Ce dernier exemple vous donne un accès en lecture au bucket et aux éléments qui y sont stockés.

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

Utilisation de instances Spot

AWS ParallelCluster utilise des instances Spot si la configuration du cluster a défini `cluster_type` = spot. Les instances ponctuelles sont plus économiques que les instances à la demande, mais elles peuvent être interrompues. L'effet de l'interruption varie en fonction du planificateur spécifique utilisé. Il peut être utile de tirer parti des avis d'interruption des instances Spot, qui fournissent un avertissement de deux minutes avant qu'Amazon ne EC2 doive arrêter ou résilier votre instance Spot. Pour plus d'informations, consultez la section Interruptions des instances Spot dans le guide de EC2 l'utilisateur Amazon. Les sections suivantes décrivent trois scénarios dans lesquels des instances Spot peuvent être interrompues.

> **Note**
>
> L'utilisation d'instances Spot nécessite que le rôle `AWSServiceRoleForEC2Spot` lié au service existe dans votre compte. Pour créer ce rôle dans votre compte à l'aide de AWS CLI, exécutez la commande suivante :
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> Pour plus d'informations, consultez la section Rôle lié au service pour les demandes d'instance Spot dans le guide de EC2 l'utilisateur Amazon.

Scénario 1 : Une instance Spot sans tâches en cours d'exécution est interrompue

Lorsque cette interruption se produit, AWS ParallelCluster tente de remplacer l'instance si la file d'attente du planificateur contient des tâches en attente qui nécessitent des instances supplémentaires, ou si le nombre d'instances actives est inférieur au `initial_queue_size` paramètre. Si AWS ParallelCluster vous ne parvenez pas à approvisionner de nouvelles instances, une demande de nouvelles instances est répétée périodiquement.

Scénario 2 : Une instance Spot exécutant des tâches à nœud unique est interrompue

Le comportement de cette interruption dépend du planificateur utilisé.

Slurm

La tâche échoue avec un code d'état de`NODE_FAIL`, et la tâche est mise en attente (sauf indication contraire `--no-requeue` lors de la soumission de la tâche). Si le nœud est un nœud statique, il est remplacé. Si le nœud est un nœud dynamique, le nœud est arrêté et réinitialisé. Pour plus d'informations`sbatch`, notamment sur le `--no-requeue` paramètre, consultez la sbatchdocumentation de Slurm.

> **Note**
>
> Ce comportement a changé dans la AWS ParallelCluster version 2.9.0. Les versions antérieures ont mis fin à la tâche avec un code d'état de `NODE_FAIL` et le nœud a été supprimé de la file d'attente du planificateur.

SGE

> **Note**
>
> Cela ne s'applique qu'aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

La tâche est arrêtée. Si la tâche a activé l'indicateur rerun (à l'aide de `qsub -r yes` ou de `qalter -r yes`) ou si la configuration `rerun` pour la file d'attente est définie sur `TRUE`, la tâche est replanifiée. L'instance de calcul est supprimée de la file d'attente du planificateur. Ce comportement est provoqué par ces paramètres de configuration SGE :

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

Torque

> **Note**
>
> Cela ne s'applique qu'aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

La tâche est supprimée du système et le nœud est supprimé du planificateur. Le job n'est pas refait. Si plusieurs tâches sont en cours d'exécution sur l'instance lorsqu'elle est interrompue, Torque peut expirer pendant la suppression du nœud. Une erreur peut s'afficher dans le fichier `sqswatcher` journal. Cela n'affecte pas la logique de mise à l'échelle, et un nettoyage approprié est effectué lors de nouvelles tentatives ultérieures.

Scénario 3 : Une instance Spot exécutant des tâches à plusieurs nœuds est interrompue

Le comportement de cette interruption dépend du planificateur utilisé.

Slurm

La tâche échoue avec un code d'état de`NODE_FAIL`, et la tâche est mise en attente (sauf si cela `--no-requeue` a été spécifié lors de la soumission de la tâche). Si le nœud est un nœud statique, il est remplacé. Si le nœud est un nœud dynamique, le nœud est arrêté et réinitialisé. Les autres nœuds qui exécutaient les tâches terminées peuvent être affectés à d'autres tâches en attente ou réduits une fois le `scaledown_idletime` délai configuré dépassé.

> **Note**
>
> Ce comportement a changé dans la AWS ParallelCluster version 2.9.0. Les versions antérieures ont mis fin à la tâche avec un code d'état de `NODE_FAIL` et le nœud a été supprimé de la file d'attente du planificateur. Les autres nœuds qui exécutaient les tâches terminées peuvent être réduits une fois le `scaledown_idletime` délai configuré dépassé.

SGE

> **Note**
>
> Cela ne s'applique qu'aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

La tâche n'est pas terminée et continue de s'exécuter sur les nœuds restants. Le nœud de calcul est supprimé de la file d'attente du planificateur, mais apparaîtra dans la liste des hôtes en tant que nœud orphelin et indisponible.

L'utilisateur doit supprimer la tâche lorsque cela se produit (`qdel <jobid>`). Le nœud s'affiche toujours dans la liste des hôtes (`qhost`), mais cela n'a aucune incidence AWS ParallelCluster. Pour supprimer l'hôte de la liste, exécutez la commande suivante après avoir remplacé l'instance.

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

Torque

> **Note**
>
> Cela ne s'applique qu'aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

La tâche est supprimée du système et le nœud est supprimé du planificateur. Le job n'est pas refait. Si plusieurs tâches sont en cours d'exécution sur l'instance lorsqu'elle est interrompue, Torque peut expirer pendant la suppression du nœud. Une erreur peut s'afficher dans le fichier `sqswatcher` journal. Cela n'affecte pas la logique de mise à l'échelle, et un nettoyage approprié est effectué lors de nouvelles tentatives ultérieures.

Pour plus d'informations sur les instances Spot, consultez la section Instances Spot dans le guide de EC2 l'utilisateur Amazon.

AWS Identity and Access Management rôles dans AWS ParallelCluster

AWS ParallelCluster utilise des rôles AWS Identity and Access Management (IAM) pour Amazon afin EC2 de permettre aux instances d'accéder aux AWS services pour le déploiement et le fonctionnement d'un cluster. Par défaut, le rôle IAM pour Amazon EC2 est créé lors de la création du cluster. Cela signifie que l'utilisateur qui crée le cluster doit avoir le niveau approprié d'autorisations, comme décrit dans les sections suivantes.

AWS ParallelCluster utilise plusieurs AWS services pour déployer et exploiter un cluster. Consultez la liste complète dans la section AWS Services utilisés dans AWS ParallelCluster la section.

Vous pouvez suivre les modifications apportées aux exemples de politiques dans AWS ParallelCluster la documentation sur GitHub.

Rubriques

- Paramètres par défaut pour la création de clusters
- Utilisation d'un rôle IAM existant pour Amazon EC2
- AWS ParallelCluster exemple de politiques relatives aux instances et aux utilisateurs

Paramètres par défaut pour la création de clusters

Lorsque vous utilisez les paramètres par défaut pour créer un cluster, un rôle IAM par défaut pour Amazon EC2 est créé par le cluster. L'utilisateur qui crée le cluster doit disposer du niveau d'autorisations approprié pour créer toutes les ressources nécessaires au lancement du cluster. Cela inclut la création d'un rôle IAM pour Amazon EC2. Généralement, l'utilisateur doit disposer des autorisations d'une politique AdministratorAccessgérée lorsqu'il utilise les paramètres par défaut. Pour plus d'informations sur les politiques gérées, voir les politiques AWS gérées dans le guide de l'utilisateur IAM.

Utilisation d'un rôle IAM existant pour Amazon EC2

À la place des paramètres par défaut, vous pouvez utiliser un paramètre existant `ec2_iam_role`lors de la création d'un cluster, mais vous devez définir la stratégie et le rôle IAM avant de tenter de lancer le cluster. Généralement, vous choisissez un rôle IAM existant pour Amazon EC2 afin de minimiser les autorisations accordées aux utilisateurs lorsqu'ils lancent des clusters. Ils AWS ParallelCluster exemple de politiques relatives aux instances et aux utilisateurs incluent les

autorisations minimales requises par AWS ParallelCluster et ses fonctionnalités. Vous devez créer des politiques et des rôles en tant que politiques individuelles dans IAM, puis associer les rôles et les politiques aux ressources appropriées. Certaines politiques de rôle peuvent devenir volumineuses et entraîner des erreurs de quota. Pour de plus amples informations, veuillez consulter Résolution des problèmes liés à la taille des politiques IAM. Dans les politiques, remplacez *<REGION><AWS ACCOUNT ID>*, et les chaînes similaires par les valeurs appropriées.

Si votre intention est d'ajouter des politiques supplémentaires aux paramètres par défaut des nœuds de cluster, nous vous recommandons de transmettre les politiques IAM personnalisées supplémentaires avec le `additional_iam_policies`paramètre au lieu de `ec2_iam_role`les utiliser.

AWS ParallelCluster exemple de politiques relatives aux instances et aux utilisateurs

Les exemples de politiques suivants incluent Amazon Resource Names (ARNs) pour les ressources. Si vous travaillez dans les partitions AWS GovCloud (US) ou sur la partition AWS Chine, elles ARNs doivent être modifiées. Plus précisément, ils doivent être remplacés par « arn:aws » par « arn : aws-us-gov » pour la AWS GovCloud (US) partition ou « arn:aws-cn » pour la partition Chine. AWS Pour plus d'informations, consultez Amazon Resource Names (ARNs) dans la section AWS GovCloud (US) Régions du guide de AWS GovCloud (US) l'utilisateur et ARNs pour les AWS services en Chine dans Getting Started with AWS services in China.

Ces politiques incluent les autorisations minimales actuellement requises par AWS ParallelCluster ses fonctionnalités et ses ressources. Certaines politiques de rôle peuvent devenir volumineuses et entraîner des erreurs de quota. Pour de plus amples informations, veuillez consulter Résolution des problèmes liés à la taille des politiques IAM.

Rubriques

- ParallelClusterInstancePolicyen utilisant SGESlurm, ou Torque
- ParallelClusterInstancePolicy utilisant awsbatch
- ParallelClusterUserPolicy utilisant Slurm
- ParallelClusterUserPolicyen utilisant SGE ou Torque
- ParallelClusterUserPolicy utilisant awsbatch
- ParallelClusterLambdaPolicyen utilisant SGESlurm, ou Torque
- ParallelClusterLambdaPolicy utilisant awsbatch

- ParallelClusterUserPolicypour les utilisateurs

`ParallelClusterInstancePolicy`en utilisant SGESlurm, ou Torque

> **Note**
>
> À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque Vous pouvez continuer à les utiliser dans les versions antérieures à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.

Rubriques

- ParallelClusterInstancePolicy utilisant Slurm
- ParallelClusterInstancePolicyen utilisant SGE ou Torque

`ParallelClusterInstancePolicy` utilisant Slurm

L'exemple suivant définit l'`ParallelClusterInstancePolicy`utilisation Slurm comme planificateur.

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

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

```
        {
            "Action": [
                "dynamodb:PutItem",
                "dynamodb:Query",
                "dynamodb:GetItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:DeleteItem",
                "dynamodb:DescribeTable"
            ],
            "Resource": [
                "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "DynamoDBTable"
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "S3GetObj"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "*"
            ],
            "Effect": "Allow",
            "Sid": "IAMPassRole",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        },
        {
            "Action": [
```

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

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

`ParallelClusterInstancePolicy`en utilisant SGE ou Torque

L'exemple suivant définit l'`ParallelClusterInstancePolicy`utilisation SGE ou en Torque tant que planificateur.

> **Note**
>
> Cette politique s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

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

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

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

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

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

`ParallelClusterInstancePolicy` utilisant `awsbatch`

L'exemple suivant définit l'`ParallelClusterInstancePolicy`utilisation `awsbatch` comme planificateur. Vous devez inclure les mêmes politiques que celles attribuées à celles définies dans la pile AWS Batch AWS CloudFormation imbriquée. `BatchUserRole` L'ARN `BatchUserRole` est fourni sous la forme d'une sortie de pile. Dans cet exemple, « *<RESOURCES S3 BUCKET>* » est la valeur du `cluster_resource_bucket` paramètre ; si elle n'`cluster_resource_bucket`est pas spécifiée, « *<RESOURCES S3 BUCKET>* » est « parallelcluster-* ». L'exemple suivant donne un aperçu des autorisations requises :

```
{
    "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` utilisant Slurm

L'exemple suivant montre comment définir la stratégie `ParallelClusterUserPolicy`, en utilisant Slurm comme planificateur. Dans cet exemple, « *<RESOURCES S3 BUCKET>* » est la valeur du `cluster_resource_bucket` paramètre ; si elle n'`cluster_resource_bucket`est pas spécifiée, « *<RESOURCES S3 BUCKET>* » est « parallelcluster-* ».

> **Note**
>
> Si vous utilisez un rôle personnalisé `ec2_iam_role = <role_name>`, vous devez modifier la ressource IAM pour inclure le nom de ce rôle à partir de :
>
> `"Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster-*"`
>
> Pour :
>
> `"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`en utilisant SGE ou Torque

> **Note**
>
> Cette section s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

L'exemple suivant définit le`ParallelClusterUserPolicy`, en utilisant SGE ou en Torque tant que planificateur. Dans cet exemple, « *<RESOURCES S3 BUCKET>* » est la valeur du `cluster_resource_bucket` paramètre ; si elle n'`cluster_resource_bucket`est pas spécifiée, « *<RESOURCES S3 BUCKET>* » est « parallelcluster-* ».

> **Note**
>
> Si vous utilisez un rôle personnalisé `ec2_iam_role = <role_name>`, vous devez modifier la ressource IAM pour inclure le nom de ce rôle à partir de :

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

Pour :

```
"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` utilisant `awsbatch`

L'exemple suivant définit l'`ParallelClusterUserPolicy`utilisation `awsbatch` comme planificateur. Dans cet exemple, « *<RESOURCES S3 BUCKET>* » est la valeur du `cluster_resource_bucket` paramètre ; si elle n'`cluster_resource_bucket`est pas spécifiée, « *<RESOURCES S3 BUCKET>* » est « 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`en utilisant SGESlurm, ou Torque

L'exemple suivant montre comment définir la stratégie `ParallelClusterLambdaPolicy`, en utilisant SGE, Slurm ou Torque comme planificateur.

> **Note**
>
> À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque

```
{
  "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` utilisant `awsbatch`

L'exemple suivant définit l'`ParallelClusterLambdaPolicy`utilisation `awsbatch` comme planificateur.

```
{
  "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`pour les utilisateurs

L'exemple suivant définit le `ParallelClusterUserPolicy` pour les utilisateurs qui n'ont pas besoin de créer ou de mettre à jour des clusters. Les commandes suivantes sont prises en charge.

- `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": "*"
        }
    ]
}
```

Planificateurs pris en charge par AWS ParallelCluster

AWS ParallelCluster prend en charge plusieurs planificateurs, définis à l'aide du `scheduler` paramètre.

> **Note**
>
> À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque Vous pouvez continuer à les utiliser dans les versions antérieures

à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.

Rubriques

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

Son of Grid Engine (`sge`)

> **Note**
>
> À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque Vous pouvez continuer à les utiliser dans les versions antérieures à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.

AWS ParallelCluster les versions 2.11.4 et antérieures utilisent Son of Grid Engine 8.1.9.

Slurm Workload Manager (`slurm`)

AWS ParallelCluster la version 2.11.9 utilise Slurm 20.11.9. Pour de plus amples informations sur Slurm, consultez https://slurm.schedmd.com/. Pour les téléchargements, veuillez consulter https://github.com/SchedMD/slurm/tags. Pour le code source, consultez https://github.com/SchedMD/slurm.

> **Important**
>
> AWS ParallelCluster est testé avec les paramètres de Slurm configuration fournis par défaut. Toute modification apportée à ces paramètres Slurm de configuration se fait à vos propres risques. Ils ne sont pris en charge que dans la mesure du possible.

AWS ParallelCluster version (s)	Version Slurm prise en charge
2,11.7, 2,11.8, 2,11.9	20,11.9
2.11.4 à 2.11.6	20,11,8
2.11.0 à 2.11.3	20,11.7
2.10.4	20,02,7
2.9.0 à 2.10.3	20,02,4
2.6 à 2.8.1	19,05,5
2.5.0, 2.5.1	19,05.3-2
2.3.1 à 2.4.1	18,088-6-2
antérieur à la version 2.3.1	16,05.3-1

Mode de file d'attente multiple

AWS ParallelCluster la version 2.9.0 a introduit le mode de file d'attente multiple. Le mode de file d'attente multiple est pris en charge lorsqu'il `scheduler` est défini sur `slurm` et que le `queue_settings` paramètre est défini. Ce mode permet à différents types d'instances de coexister dans les nœuds de calcul. Les ressources de calcul qui contiennent les différents types d'instances peuvent être augmentées ou diminuées selon les besoins. En mode file d'attente, jusqu'à cinq (5) files d'attente sont prises en charge, et chaque `[queue]`section peut faire référence à un maximum de trois (3) `[compute_resource]`sections. Chacune de ces `[queue]`sections constitue une partitionSlurm Workload Manager. Pour plus d'informations, consultez Slurmguide pour le mode de file d'attente multiple et Tutoriel en mode file d'attente multiple.

Chaque `[compute_resource]`section d'une file d'attente doit avoir un type d'instance différent, et chacune d'entre elles `[compute_resource]` est ensuite divisée en nœuds statiques et dynamiques. Les nœuds statiques de chacun `[compute_resource]` sont numérotés de 1 à la valeur de`min_count`. Les nœuds dynamiques de chacun `[compute_resource]` sont numérotés de un (1) à (`max_count`-`min_count`). Par exemple, si la valeur `min_count` `max_count` est 2 et 10, les nœuds dynamiques correspondants `[compute_resource]` sont numérotés de un (1) à huit

(8). À tout moment, il peut y avoir entre zéro (0) et le nombre maximum de nœuds dynamiques dans `[compute_resource]` a.

Les instances lancées dans le parc informatique sont attribuées dynamiquement. Pour faciliter cette gestion, des noms d'hôtes sont générés pour chaque nœud. Le format du nom d'hôte est le suivant :

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

- `$QUEUE`est le nom de la file d'attente. Par exemple, si la section commence`[queue queue-name]`, « `$QUEUE` » est « *queue-name* ».
- `$STATDYN`est `st` destiné aux nœuds statiques ou `dy` aux nœuds dynamiques.
- `$INSTANCE_TYPE`est le type d'instance pour le`[compute_resource]`, à partir du `instance_type` paramètre.
- `$NODENUM`est le numéro du nœud. `$NODENUM`est compris entre un (1) et la valeur de `min_count` pour les nœuds statiques et entre un (1) et (`max_count`-`min_count`) pour les nœuds dynamiques.

Les noms d'hôte et les noms de domaine complets (FQDN) sont créés à l'aide des zones hébergées Amazon Route 53. Le FQDN est`$HOSTNAME.$CLUSTERNAME.pcluster`, où `$CLUSTERNAME` est le nom de la `[cluster]`section utilisée pour le cluster.

Pour convertir votre configuration en mode file d'attente, utilisez la `pcluster-config convert` commande. Il écrit une configuration mise à jour avec une seule `[queue]`section nommée`[queue compute]`. Cette file d'attente contient une seule `[compute_resource]`section nommée`[compute_resource default]`. Les paramètres `[queue compute]` et ont `[compute_resource default]` été migrés depuis la `[cluster]`section spécifiée.

Slurmguide pour le mode de file d'attente multiple

AWS ParallelCluster la version 2.9.0 a introduit le mode de file d'attente multiple et une nouvelle architecture de dimensionnement pour Slurm Workload Manager (Slurm).

Les sections suivantes fournissent un aperçu général de l'utilisation d'un Slurm cluster avec la nouvelle architecture de mise à l'échelle.

Présentation

La nouvelle architecture de mise à l'échelle est basée sur le guide Slurm de planification dans le cloud et sur le plugin d'économie d'énergie. Pour plus d'informations sur le plug-in d'économie

d'énergie, consultez le Guide d'économie d'Slurménergie. Dans la nouvelle architecture, les ressources susceptibles d'être mises à disposition pour un cluster sont généralement prédéfinies dans la Slurm configuration sous forme de nœuds cloud.

Cycle de vie des nœuds cloud

Tout au long de leur cycle de vie, les nœuds cloud entrent dans plusieurs des états suivantsPOWER_SAVING, POWER_UP voire dans tous les cas :, ALLOCATED (alloc), () et POWER_DOWN (pow_dn). pow_up Dans certains cas, un nœud de cloud peut entrer dans OFFLINE cet état. La liste suivante détaille plusieurs aspects de ces états dans le cycle de vie des nœuds de cloud.

- Un nœud dans un POWER_SAVING état apparaît avec un ~ suffixe (par exempleidle~) danssinfo. Dans cet état, aucune EC2 instance ne soutient le nœud. Cependant, Slurm vous pouvez toujours allouer des tâches au nœud.
- Un nœud en transition vers un POWER_UP état apparaît avec un # suffixe (par exempleidle#) dans. sinfo
- Lorsqu'il Slurm alloue une tâche à un nœud dans un POWER_SAVING état, le nœud est automatiquement transféré vers un POWER_UP état. Sinon, les nœuds peuvent être placés dans l'POWER_UPétat manuellement à l'aide de la scontrol update nodename=*nodename* state=power_up commande. À ce stade, le ResumeProgram est invoqué, et les EC2 instances sont lancées et configurées pour sauvegarder un POWER_UP nœud.
- Un nœud actuellement disponible apparaît sans suffixe (par exempleidle) danssinfo. Une fois le nœud configuré et rejoint le cluster, il devient disponible pour exécuter des tâches. À ce stade, le nœud est correctement configuré et prêt à être utilisé. En règle générale, nous recommandons que le nombre d'instances EC2 soit identique au nombre de nœuds disponibles. Dans la plupart des cas, les nœuds statiques sont toujours disponibles après la création du cluster.
- Un nœud en transition vers un POWER_DOWN état apparaît avec un % suffixe (par exempleidle %) dans. sinfo Les nœuds dynamiques entrent automatiquement dans leur POWER_DOWN état par la suitescaledown_idletime. En revanche, dans la plupart des cas, les nœuds statiques ne sont pas hors tension. Cependant, les nœuds peuvent être placés dans l'POWER_DOWNétat manuellement à l'aide de la scontrol update nodename=*nodename* state=powering_down commande. Dans cet état, l'instance associée à un nœud est interrompue et le nœud est remis à son POWER_SAVING état pour une utilisation future par la suitescaledown_idletime. Le scaledown-idletime paramètre est enregistré dans la Slurm configuration en tant que SuspendTimeout paramètre.

- Un nœud hors ligne apparaît avec un `*` suffixe (par exemple`down*`) dans`sinfo`. Un nœud se déconnecte si le Slurm contrôleur ne parvient pas à le contacter ou si les nœuds statiques sont désactivés et que les instances de support sont résiliées.

Examinons maintenant les états des nœuds illustrés dans l'`sinfo`exemple suivant.

```
$ 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]
```

Les `efa-st-c5n18xlarge-1` nœuds `spot-st-t2large-[1-2]` et ont déjà des instances de sauvegarde configurées et peuvent être utilisées. Les `ondemand-dy-c52xlarge-[1-2]` nœuds sont en `POWER_UP` bon état et devraient être disponibles dans quelques minutes. Le `gpu-dy-g38xlarge-1` nœud est dans l'`POWER_DOWN`état, et il passera à `POWER_SAVING` l'état par la suite `scaledown_idletime` (120 secondes par défaut).

Tous les autres nœuds sont en `POWER_SAVING` état et aucune EC2 instance ne les soutient.

Utilisation d'un nœud disponible

Un nœud disponible est soutenu par une EC2 instance. Par défaut, le nom du nœud peut être utilisé pour accéder directement à l'instance en SSH (par exemple`ssh efa-st-c5n18xlarge-1`). L'adresse IP privée de l'instance peut être récupérée à l'aide de la `scontrol show nodes` *`nodename`* commande et en vérifiant le `NodeAddr` champ. Pour les nœuds qui ne sont pas disponibles, le `NodeAddr` champ ne doit pas pointer vers une EC2 instance en cours d'exécution. Il doit plutôt être identique au nom du nœud.

État des offres d'emploi et soumission

Dans la plupart des cas, les tâches soumises sont immédiatement allouées aux nœuds du système ou mises en attente si tous les nœuds sont alloués.

Si les nœuds alloués à une tâche incluent des nœuds dans un `POWER_SAVING` état, la tâche commence par un `CF CONFIGURING` état ou. À ce stade, la tâche attend que les nœuds de l'`POWER_SAVING`état passent à l'`POWER_UP`état et soient disponibles.

Une fois que tous les nœuds alloués à une tâche sont disponibles, la tâche passe à l'état `RUNNING` (`R`).

Par défaut, toutes les tâches sont soumises à la file d'attente par défaut (connue sous le nom de partition inSlurm). Cela est indiqué par un `*` suffixe après le nom de la file d'attente. Vous pouvez sélectionner une file d'attente à l'aide de l'option de soumission des `-p` tâches.

Tous les nœuds sont configurés avec les fonctionnalités suivantes, qui peuvent être utilisées dans les commandes de soumission de tâches :

- Un type d'instance (par exemple`c5.xlarge`)
- Type de nœud (il s'agit de l'un `dynamic` ou `static` de l'autre)

Vous pouvez voir toutes les fonctionnalités disponibles pour un nœud en particulier en utilisant la `scontrol show nodes` *`nodename`* commande et en consultant la `AvailableFeatures` liste.

Les emplois constituent un autre facteur à prendre en compte. Examinez d'abord l'état initial du cluster, que vous pouvez visualiser en exécutant la `sinfo` commande.

```
$ 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]
```

Notez qu'il s'`spot`agit de la file d'attente par défaut. Il est indiqué par le `*` suffixe.

Soumettez une tâche à un nœud statique de la file d'attente par défaut (`spot`).

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

Soumettez une tâche à un nœud dynamique de la `EFA` file d'attente.

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

Soumettez une tâche à huit (8) `c5.2xlarge` nœuds et à deux (2) `t2.xlarge` nœuds de la `ondemand` file d'attente.

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

Soumettez une tâche à un nœud GPU de la `gpu` file d'attente.

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

Examinez maintenant l'état des tâches à l'aide de la `squeue` commande.

```
$ 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
```

Les tâches 7 et 8 (dans les `efa` files d'attente `spot` et) sont déjà en cours d'exécution (`R`). Les jobs 12 et 13 sont toujours en cours de configuration (`CF`), attendant probablement que les instances soient disponibles.

```
# 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
```

État et fonctionnalités du nœud

Dans la plupart des cas, les états des nœuds sont entièrement gérés AWS ParallelCluster conformément aux processus spécifiques du cycle de vie des nœuds de cloud décrits plus haut dans cette rubrique.

Toutefois, remplace ou arrête AWS ParallelCluster également les nœuds défectueux dans les `DRAINED` états `DOWN` et les nœuds dont les instances de sauvegarde sont défectueuses. Pour de plus amples informations, veuillez consulter `clustermgtd`.

États de partition

AWS ParallelCluster prend en charge les états de partition suivants. Une Slurm partition est une file d'attente dans AWS ParallelCluster.

- `UP`: indique que la partition est active. Il s'agit de l'état par défaut d'une partition. Dans cet état, tous les nœuds de la partition sont actifs et peuvent être utilisés.
- `INACTIVE`: indique que la partition est inactive. Dans cet état, toutes les instances qui soutiennent les nœuds d'une partition inactive sont arrêtées. Les nouvelles instances ne sont pas lancées pour les nœuds d'une partition inactive.

démarrage et arrêt de pcluster

Lorsqu'il `pcluster stop` est exécuté, toutes les partitions sont placées dans l'`INACTIVE`état, et les AWS ParallelCluster processus les maintiennent dans `INACTIVE` cet état.

Lorsqu'il `pcluster start` est exécuté, toutes les partitions sont initialement placées dans `UP` cet état. Cependant, AWS ParallelCluster les processus ne maintiennent pas la partition dans un `UP` état. Vous devez modifier l'état des partitions manuellement. Tous les nœuds statiques sont disponibles au bout de quelques minutes. Notez que le fait de définir une partition sur `UP` n'augmente aucune capacité dynamique. Si `initial_count` cette valeur est supérieure à`max_count`, elle `initial_count` peut ne pas être satisfaite lorsque l'état de la partition passe à l'`UP`état actuel.

Lorsqu'il est `pcluster start` `pcluster stop` en cours d'exécution, vous pouvez vérifier l'état du cluster en exécutant la `pcluster status` commande et en vérifiant le`ComputeFleetStatus`. La liste suivante répertorie les états possibles :

- `STOP_REQUESTED`: La `pcluster stop` demande est envoyée au cluster.
- `STOPPING`: le `pcluster` processus arrête actuellement le cluster.
- `STOPPED`: le `pcluster` processus a terminé le processus d'arrêt, toutes les partitions sont en `INACTIVE` état et toutes les instances de calcul sont terminées.
- `START_REQUESTED`: La `pcluster start` demande est envoyée au cluster.
- `STARTING`: le `pcluster` processus démarre actuellement le cluster
- `RUNNING`: le `pcluster` processus de démarrage est terminé, toutes les partitions sont en `UP` état et les nœuds statiques seront disponibles au bout de quelques minutes.

Contrôle manuel des files d'attente

Dans certains cas, vous souhaiterez peut-être contrôler manuellement les nœuds ou la file d'attente (appelée partition dansSlurm) d'un cluster. Vous pouvez gérer les nœuds d'un cluster à l'aide des procédures courantes suivantes.

- Activez les nœuds dynamiques en `POWER_SAVING` état : exécutez la `scontrol update nodename=`*`nodename`* `state=power_up` commande ou soumettez une `sleep 1` tâche fictive demandant un certain nombre de nœuds et comptez sur cette tâche Slurm pour alimenter le nombre de nœuds requis.
- Éteignez les nœuds dynamiques avant `scaledown_idletime` : définissez les nœuds dynamiques sur à l'`DOWN`aide de la `scontrol update nodename=`*`nodename`* `state=down` commande. AWS ParallelCluster arrête et réinitialise automatiquement les nœuds dynamiques tombés en panne. En général, nous ne recommandons pas de configurer les nœuds pour qu'`POWER_DOWN`ils utilisent directement la `scontrol update nodename=`*`nodename`* `state=power_down` commande. Cela est dû au fait que le processus de mise hors tension est AWS ParallelCluster automatiquement géré. Aucune intervention manuelle n'est nécessaire. Par conséquent, nous vous recommandons d'essayer de définir les nœuds sur `DOWN` chaque fois que cela est possible.
- Désactiver une file d'attente (partition) ou arrêter tous les nœuds statiques d'une partition spécifique : définissez la file d'attente sur `INACTIVE` avec la `scontrol update partition=`*`queue name`* `state=inactive` commande. Cela met fin à toutes les instances qui soutiennent les nœuds de la partition.
- Activer une file d'attente (partition) : définissez spécifiquement la file d'attente à l'`INACTIVE`aide de la `scontrol update partition=`*`queue name`* `state=up` commande.

Comportement de dimensionnement et ajustements

Voici un exemple du flux de travail de dimensionnement normal :

- Le planificateur reçoit une tâche qui nécessite deux nœuds.
- Le planificateur fait passer deux nœuds à un `POWER_UP` état et appelle `ResumeProgram` avec les noms des nœuds (par exemple`queue1-dy-c5xlarge-[1-2]`).
- `ResumeProgram`lance deux EC2 instances et attribue les adresses IP privées et les noms d'hôte de`queue1-dy-c5xlarge-[1-2]`, en attendant `ResumeTimeout` (la période par défaut est de 60 minutes (1 heure)) avant de réinitialiser les nœuds.
- Les instances sont configurées et rejoignent le cluster. Job commence à s'exécuter sur des instances.
- Job est terminé.
- Une fois la configuration `SuspendTime` expirée (qui est définie sur`scaledown_idletime`), les instances sont mises dans l'`POWER_SAVING`état par le planificateur. Le planificateur place `queue1-dy-c5xlarge-[1-2]` dans `POWER_DOWN` l'état et appelle `SuspendProgram` avec les noms des nœuds.
- `SuspendProgram`est appelé pour deux nœuds. Les nœuds restent dans `POWER_DOWN` cet état, par exemple en restant `idle%` pendant un `SuspendTimeout` (la période par défaut est de 120 secondes (2 minutes)). Après avoir `clustermgtd` détecté que les nœuds sont hors tension, il met fin aux instances de sauvegarde. Ensuite, il passe à `queue1-dy-c5xlarge-[1-2]` l'état inactif et réinitialise l'adresse IP privée et le nom d'hôte afin qu'ils puissent être réalimentés pour de futures tâches.

Maintenant, si les choses tournent mal et qu'une instance pour un nœud particulier ne peut pas être lancée pour une raison quelconque, voici ce qui se passe.

- Le planificateur reçoit une tâche qui nécessite deux nœuds.
- Le planificateur place deux nœuds éclatants dans le cloud en `POWER_UP` état et appelle `ResumeProgram` avec les noms de nœuds (par exemple). `queue1-dy-c5xlarge-[1-2]`
- `ResumeProgram`lance une (1) seule EC2 instance et la configure`queue1-dy-c5xlarge-1`, mais aucune instance n'a pu être lancée pour`queue1-dy-c5xlarge-2`.
- `queue1-dy-c5xlarge-1`ne sera pas affecté et sera mis en ligne après avoir atteint `POWER_UP` l'état.

- `queue1-dy-c5xlarge-2`est mis en `POWER_DOWN` état et la tâche est automatiquement mise en attente car elle Slurm détecte une défaillance du nœud.
- `queue1-dy-c5xlarge-2`devient disponible après `SuspendTimeout` (la valeur par défaut est de 120 secondes (2 minutes)). Entre-temps, la tâche est mise en attente et peut commencer à s'exécuter sur un autre nœud.
- Le processus ci-dessus est répété jusqu'à ce que la tâche puisse s'exécuter sur un nœud disponible sans qu'une défaillance se produise.

Deux paramètres de chronométrage peuvent être ajustés si nécessaire.

- `ResumeTimeout`(la valeur par défaut est de 60 minutes (1 heure)) : `ResumeTimeout` contrôle le temps d'Slurmattente avant de mettre le nœud en état inactif.
  - Il peut être utile de l'étendre si votre processus pre/post d'installation prend presque autant de temps.
  - Il s'agit également du temps d' AWS ParallelCluster attente maximal avant de remplacer ou de réinitialiser un nœud en cas de problème. Les nœuds de calcul s'arrêtent automatiquement en cas d'erreur lors du lancement ou de la configuration. Ensuite, AWS ParallelCluster les processus remplacent également le nœud lorsqu'il constate que l'instance est terminée.
- `SuspendTimeout`(la valeur par défaut est de 120 secondes (2 minutes)) : `SuspendTimeout` contrôle la rapidité avec laquelle les nœuds sont réinsérés dans le système et prêts à être réutilisés.
  - Une valeur plus courte `SuspendTimeout` signifierait que les nœuds seront réinitialisés plus rapidement et Slurm qu'ils pourront essayer de lancer des instances plus fréquemment.
  - Une durée plus `SuspendTimeout` longue ralentit la réinitialisation des nœuds défaillants. En attendant, essaie Slurm d'utiliser d'autres nœuds. Si cela `SuspendTimeout` dure plus de quelques minutes, Slurm essaie de parcourir tous les nœuds du système. Un délai plus long `SuspendTimeout` peut être bénéfique pour les systèmes à grande échelle (plus de 1 000 nœuds) car il permet de réduire le stress en remettant fréquemment Slurm en file d'attente les tâches défaillantes.
  - Notez que `SuspendTimeout` cela ne fait pas référence au temps d' AWS ParallelCluster attente pour mettre fin à une instance de sauvegarde pour un nœud. Les instances de sauvegarde pour `power down` les nœuds sont immédiatement résiliées. Le processus de terminaison est généralement terminé en quelques minutes. Cependant, pendant cette période, le nœud reste hors tension et n'est pas disponible pour une utilisation dans le planificateur.

Journaux pour une nouvelle architecture

La liste suivante contient les journaux clés de l'architecture à files d'attente multiples. Le nom du flux de journal utilisé avec Amazon CloudWatch Logs est au format `{hostname}.{instance_id}.{logIdentifier}` `logIdentifier` suivant lequel suivent les noms des journaux. Pour de plus amples informations, veuillez consulter Intégration à 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`)

Problèmes courants et procédure de débogage :

Nœuds qui n'ont pas réussi à démarrer, à démarrer ou à rejoindre le cluster :

- Nœuds dynamiques :
  - Consultez le `ResumeProgram` journal pour voir s'il `ResumeProgram` a déjà été appelé avec le nœud. Si ce n'est pas le cas, consultez le `slurmctld` journal pour déterminer si vous avez Slurm déjà essayé `ResumeProgram` d'appeler le nœud. Notez que des autorisations incorrectes `ResumeProgram` peuvent entraîner son échec silencieux.

  - S'`ResumeProgram`il est appelé, vérifiez si une instance a été lancée pour le nœud. Si l'instance ne peut pas être lancée, un message d'erreur clair devrait s'afficher expliquant pourquoi le lancement de l'instance a échoué.
  - Si une instance a été lancée, il se peut qu'il y ait eu un problème pendant le processus de démarrage. Trouvez l'adresse IP privée et l'ID d'instance correspondants dans le `ResumeProgram` journal et consultez les journaux de démarrage correspondants pour l'instance spécifique dans CloudWatch Logs.
- Nœuds statiques :
  - Consultez le `clustermgtd` journal pour voir si des instances ont été lancées pour le nœud. Dans le cas contraire, il devrait y avoir des erreurs claires expliquant pourquoi les instances n'ont pas pu être lancées.
  - Si une instance a été lancée, il y a un problème pendant le processus de démarrage. Trouvez l'adresse IP privée et l'ID d'instance correspondants dans le `clustermgtd` journal et consultez les journaux de démarrage correspondants pour l'instance spécifique dans CloudWatch Logs.

Nœuds remplacés ou interrompus de façon inattendue, défaillances de nœuds

- Des nœuds de replaced/terminated façon inattendue
  - Dans la plupart des cas, `clustermgtd` gère toutes les actions de maintenance des nœuds. Pour vérifier si un nœud a été `clustermgtd` remplacé ou résilié, consultez le `clustermgtd` journal.
  - En cas de `clustermgtd` remplacement ou de résiliation du nœud, un message doit apparaître indiquant le motif de l'action. Si la raison est liée au planificateur (par exemple, le nœud l'était`DOWN`), consultez le `slurmctld` journal pour plus de détails. Si la raison est EC2 liée, utilisez des outils pour vérifier le statut ou les journaux de cette instance. Par exemple, vous pouvez vérifier si l'instance a connu des événements planifiés ou si les vérifications de son état EC2 de santé ont échoué.
  - Si le nœud `clustermgtd` n'a pas été résilié, vérifiez s'il `computemgtd` a été résilié ou si l'instance EC2 a été résiliée pour récupérer une instance Spot.
- Défaillances de nœuds
  - Dans la plupart des cas, les tâches sont automatiquement mises en attente en cas de défaillance d'un nœud. Consultez le `slurmctld` journal pour savoir pourquoi une tâche ou un nœud a échoué et analysez la situation à partir de là.

Défaillance lors du remplacement ou de l'arrêt d'instances, défaillance lors de la mise hors tension des nœuds

- En général, `clustermgtd` gère toutes les actions de fermeture d'instance attendues. Consultez le `clustermgtd` journal pour savoir pourquoi il n'a pas réussi à remplacer ou à mettre fin à un nœud.
- En cas d'échec d'un nœud dynamique`scaledown_idletime`, consultez le `SuspendProgram` journal pour voir si un programme `slurmctld` utilise le nœud spécifique comme argument. Remarque `SuspendProgram` n'effectue en fait aucune action spécifique. Au contraire, il ne se connecte que lorsqu'il est appelé. La résiliation et la `NodeAddr` réinitialisation de toutes les instances sont terminées par`clustermgtd`. Slurmplace les nœuds dans `IDLE` After`SuspendTimeout`.

Autres problèmes

- AWS ParallelCluster ne prend pas de décisions relatives à l'attribution des tâches ou à la mise à l'échelle. Il essaie simplement de lancer, de résilier et de maintenir les ressources conformément Slurm aux instructions.

  Pour les problèmes liés à l'allocation des tâches, à l'allocation des nœuds et à la décision de dimensionnement, consultez le `slurmctld` journal pour détecter les erreurs.

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

> **Note**
>
> À partir de la version 2.11.5, l'utilisation des planificateurs AWS ParallelCluster n'est pas prise en SGE charge. Torque Vous pouvez continuer à les utiliser dans les versions antérieures à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.

AWS ParallelCluster les versions 2.11.4 et antérieures utilisent Torque Resource Manager 6.1.2. Pour de plus amples informations sur Torque Resource Manager 6.1.2, veuillez consulter http://docs.adaptivecomputing.com/torque/6-1-2/releaseNotes/torquerelnote.htm. Pour obtenir la

documentation, veuillez consulter http://docs.adaptivecomputing.com/torque/6-1-2/adminGuide/torque.htm. Pour le code source, consultez https://github.com/adaptivecomputing/torque/tree/6.1.2.

AWS ParallelCluster les versions 2.4.0 et antérieures utilisent Torque Resource Manager 6.0.2. Pour obtenir les notes de mise à jour, veuillez consulter http://docs.adaptivecomputing.com/torque/6-0-2/releaseNotes/torqueReleaseNotes6.0.2.pdf. Pour obtenir la documentation, veuillez consulter http://docs.adaptivecomputing.com/torque/6-0-2/adminGuide/help.htm. Pour le code source, consultez https://github.com/adaptivecomputing/torque/tree/6.0.2.

AWS Batch (`awsbatch`)

Pour plus d'informations sur AWS Batch, voir AWS Batch. Pour obtenir de la documentation, consultez le guide de AWS Batch l'utilisateur.

AWS ParallelCluster Commandes CLI pour AWS Batch

Lorsque vous utilisez le `awsbatch` planificateur, les commandes AWS ParallelCluster CLI pour AWS Batch sont automatiquement installées dans le nœud AWS ParallelCluster principal. La CLI utilise des opérations d' AWS Batch API et autorise les opérations suivantes :

- Envoyer et gérer les tâches.
- Surveiller les tâches, les files d'attente et les hôtes.
- Mettre en miroir les commandes traditionnelles du planificateur.

> **Important**
>
> AWS ParallelCluster ne prend pas en charge les tâches GPU pour AWS Batch. Pour plus d'informations, consultez la section Tâches GPU.

Rubriques

- awsbsub
- awsbstat
- awsbout
- awsbkill
- awsbqueues

- awsbhosts

awsbsub

Soumet les tâches à la file d'attente des tâches du 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 ne prend pas en charge les tâches GPU pour AWS Batch. Pour plus d'informations, consultez la section Tâches GPU.

Arguments positionnels

command

Soumet la tâche (la commande spécifiée doit être disponible sur les instances de calcul) ou le nom du fichier à transférer. Voir aussi `--command-file`.

arguments

(Facultatif) Spécifie les arguments de la commande ou du fichier de commandes.

Arguments nommés

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

Nom de la tâche. Le premier caractère doit être une lettre ou un chiffre. Le nom du travail peut contenir des lettres (majuscules et minuscules), des chiffres, des traits d'union et des traits de soulignement, et comporter jusqu'à 128 caractères.

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

Spécifie le cluster à utiliser.

-cf, --command-file

Indique que la commande est un fichier à transférer aux instances de calcul.

Par défaut : false

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

Spécifie le dossier à utiliser en tant que répertoire de travail des tâches. Si aucun répertoire de travail n'est spécifié, la tâche est exécutée dans le `job-<AWS_BATCH_JOB_ID>` sous-dossier du répertoire personnel de l'utilisateur. Vous pouvez utiliser ce paramètre ou le paramètre `--parent-working-dir`.

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

Spécifie le dossier parent du répertoire de travail de la tâche. Si aucun répertoire de travail parent n'est spécifié, il s'agit par défaut du répertoire personnel de l'utilisateur. Un sous-dossier nommé `job-<AWS_BATCH_JOB_ID>` est créé dans le répertoire de travail parent. Vous pouvez utiliser ce paramètre ou le paramètre `--working-dir`.

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

Spécifie le fichier à transférer vers les instances de calcul, dans le répertoire de travail de la tâche. Vous pouvez spécifier plusieurs paramètres de fichiers d'entrée.

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

Spécifie le nombre de v CPUs à réserver pour le conteneur. Lorsqu'il est utilisé avec`–nodes`, il identifie le nombre de v CPUs pour chaque nœud.

Valeur par défaut : 1

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

Spécifie la limite stricte de la mémoire (en Mio) à fournir pour la tâche. Si votre tâche tente de dépasser la limite de mémoire spécifiée ici, elle est terminée.

Valeur par défaut : 128

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

Spécifie une liste de noms de variables d'environnement séparés par des virgules à exporter vers l'environnement de la tâche. Pour exporter toutes les variables d'environnement, spécifiez « all ». Notez qu'une liste de « toutes » les variables d'environnement n'inclut pas celles répertoriées dans le `–env-blacklist` paramètre, ni les variables commençant par le `AWS_*` préfixe `PCLUSTER_*` ou.

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

Spécifie une liste de noms de variable d'environnement séparés par des virgules à ne pas exporter vers l'environnement de la tâche. Par défaut, `HOME`, `PWD`, `USER`, `PATH`, `LD_LIBRARY_PATH`, `TERM` et `TERMCAP` ne sont pas exportées.

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

Spécifie le nombre de fois qu'une tâche doit passer au `RUNNABLE` statut. Vous pouvez indiquer entre 1 et 10 tentatives. Si la valeur des tentatives est supérieure à 1, la tâche est réessayée en cas d'échec, jusqu'à ce qu'elle passe à un `RUNNABLE` statut pour le nombre de fois spécifié.

Valeur par défaut : 1

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

Spécifie la durée en secondes (mesurée à partir de l'`startedAt`horodatage de la tentative de travail) après laquelle votre tâche s' AWS Batch arrête si elle n'est pas terminée. La valeur du délai d'expiration doit être au moins égal à 60 secondes.

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

Spécifie le nombre de nœuds à réserver pour la tâche. Spécifiez une valeur pour ce paramètre afin d'activer la soumission parallèle sur plusieurs nœuds.

> **Note**
>
> Les tâches parallèles à nœuds multiples ne sont pas prises en charge lorsque le `cluster_type` paramètre est défini sur. `spot`

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

Indique la taille du tableau. Vous pouvez spécifier une valeur comprise entre 2 et 10 000. Si vous spécifiez les propriétés de tableau pour une tâche, elle devient une tâche de tableau.

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

Spécifie une liste de dépendances séparées par un point-virgule pour une tâche. Une tâche peut compter jusqu'à 20 tâches au plus. Vous pouvez spécifier une dépendance `SEQUENTIAL` de type sans spécifier d'ID de tâche pour les tâches matricielles. Une dépendance séquentielle permet à chaque tâche de tableau enfant de se terminer de manière séquentielle, à partir de l'index 0. Vous pouvez également spécifier une dépendance de type N_TO_N avec un ID de tâche pour les

tâches de tableau. Une dépendance N_TO_N signifie que chaque enfant de l'index de cette tâche doit attendre que l'enfant de l'index correspondant de chaque dépendance soit terminé avant de pouvoir commencer. La syntaxe de ce paramètre est « jobID=*<string>*, type= *<string>* ;... ».

awsbstat

Affiche les tâches soumises dans la file d'attente des tâches du cluster.

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

Arguments positionnels

job_ids

Spécifie la liste des tâches séparées par des espaces IDs à afficher dans la sortie. Si la tâche est un tableau de tâches, toutes les tâches enfants sont affichées. Si une tâche unique est demandée, elle apparaît dans une version détaillée.

Arguments nommés

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

Indique le cluster à utiliser.

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

Spécifie la liste des statuts de tâche séparés par une virgule à inclure. Par défaut, le statut de la tâche est « active ». Les valeurs acceptées sont : `SUBMITTED`, `PENDING`, `RUNNABLE`, `STARTING`, `RUNNING`, `SUCCEEDED`, `FAILED` et `ALL`.

Par défaut : « `SUBMITTED,PENDING,RUNNABLE,STARTING,RUNNING` »

-e, --expand-children

Étend les tâches avec enfants (tableau et parallèle à plusieurs nœuds).

Par défaut : false

-d, --details

Affiche les détails des tâches.

Par défaut : false

awsbout

Affiche la sortie d'une tâche donnée.

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

Arguments positionnels

job_id

Spécifie l'ID de tâche.

Arguments nommés

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

Indique le cluster à utiliser.

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

Obtient les premières *HEAD* lignes du résultat de la tâche.

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

Permet d'obtenir les dernières lignes <tail> de la sortie de tâche.

-s, --stream

Permet d'obtenir la sortie de tâche, puis attend qu'une sortie supplémentaire soit produite. Cet argument peut être utilisé avec -tail pour démarrer à partir des dernières lignes <tail> de la sortie de tâche.

Par défaut : false

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

Définit la période de streaming.

Par défaut: 5

awsbkill

Annule ou met fin à des tâches soumises dans le cluster.

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

Arguments positionnels

job_ids

Spécifie la liste séparée par des espaces des tâches IDs à annuler ou à terminer.

Arguments nommés

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

Indique le nom du cluster à utiliser.

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

Indique le message que vous souhaitez attacher à une tâche, en expliquant la raison de l'annulation.

Par défaut : « Terminated by the user »

awsbqueues

Affiche la file d'attente de tâches associée au cluster.

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

Arguments positionnels

job_queues

Spécifie la liste des noms de file d'attente séparés par des espaces à afficher. Si une seule file d'attente est demandée, elle est affichée dans une version détaillée.

Arguments nommés

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

Spécifie le nom du cluster à utiliser.

`-d, --details`

Indique s'il convient d'afficher les détails des files d'attente.

Par défaut : false

`awsbhosts`

Affiche les hôtes qui appartiennent à l'environnement de calcul du cluster.

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

Arguments positionnels

`instance_ids`

Spécifie une liste d'instances séparées par des espaces. IDs Si une seule instance est demandée, elle apparaît dans une version détaillée.

Arguments nommés

`-c CLUSTER, --cluster CLUSTER`

Spécifie le nom du cluster à utiliser.

`-d, --details`

Indique s'il convient d'afficher les détails des hôtes.

Par défaut : false

AWS ParallelCluster ressources et balisage

Avec AWS ParallelCluster vous pouvez créer des balises pour suivre et gérer vos AWS ParallelCluster ressources. Vous définissez les balises que vous souhaitez créer et propager AWS CloudFormation à toutes les ressources du cluster dans la `tags` section du fichier de configuration du cluster. Vous pouvez également utiliser des balises AWS ParallelCluster générées automatiquement pour suivre et gérer vos ressources.

Lorsque vous créez un cluster, le cluster et ses ressources sont étiquetés avec les balises AWS ParallelCluster et AWS systems définies dans cette section.

AWS ParallelCluster applique des balises aux instances, aux volumes et aux ressources du cluster. Pour identifier la pile de clusters, AWS CloudFormation applique des balises AWS système aux instances de cluster. Pour identifier les modèles de EC2 lancement de clusters EC2 , appliquez des balises système aux instances. Vous pouvez utiliser ces balises pour afficher et gérer vos AWS ParallelCluster ressources.

Vous ne pouvez pas modifier les balises AWS système. Afin d'éviter tout impact sur les AWS ParallelCluster fonctionnalités, ne modifiez pas les AWS ParallelCluster balises.

Voici un exemple de balise AWS système pour une AWS ParallelCluster ressource. Vous ne pouvez pas les modifier.

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

Voici des exemples de AWS ParallelCluster balises appliquées à une ressource. Ne les modifiez pas.

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

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

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

Vous pouvez consulter ces balises dans la EC2 section du AWS Management Console.

Affichage des balises

1. Naviguez dans la EC2 console à l'adresse https://console.aws.amazon.com/ec2/.
2. Pour afficher toutes les balises de cluster, choisissez Tags dans le volet de navigation.
3. Pour afficher les balises de cluster par instance, choisissez Instances dans le volet de navigation.
4. Sélectionnez une instance de cluster.
5. Choisissez l'onglet Gérer les balises dans les détails de l'instance et consultez les balises.
6. Choisissez l'onglet Stockage dans les détails de l'instance.
7. Sélectionnez l'ID du volume.
8. Dans Volumes, choisissez le volume.
9. Choisissez l'onglet Tags dans les détails du volume et visualisez les tags.

AWS ParallelCluster balises d'instance du nœud principal

Clé	Valeur de balise
ClusterName	*clustername*
Name	Master
Application	parallelcluster- *clustername*
aws:ec2launchtemplate:id	*lt-1234567890abcdef0*
aws:ec2launchtemplate:version	*1*
aws-parallelcluster-node-type	Master
aws:cloudformation:stack-name	parallelcluster- *clustername* -MasterServerSubstack- *ABCD1234EFGH*
aws:cloudformation:logical-id	MasterServer
aws:cloudformation:stack-id	arn:aws:cloudformation: *region-id* :*ACCOUNTID* :stack/parallelcluster- *clustername* -MasterServerSubstack- *ABCD1234EFGH* /*1234abcd-12ab-12ab-12ab-1234567890abcdef0*
Version	*2.11.9*

AWS ParallelCluster balises de volume racine du nœud principal

Clé de balise	Valeur de balise
ClusterName	*clustername*
Application	parallelcluster- *clustername*
aws-parallelcluster-node-type	Master

AWS ParallelCluster balises d'instance de nœud de calcul

Clé	Valeur de balise
`ClusterName`	*`clustername`*
`aws-parallelcluster-node-type`	`Compute`
`aws:ec2launchtemplate:id`	*`lt-1234567890abcdef0`*
`aws:ec2launchtemplate:version`	*`1`*
`QueueName`	*`queue-name`*
`Version`	*`2.11.9`*

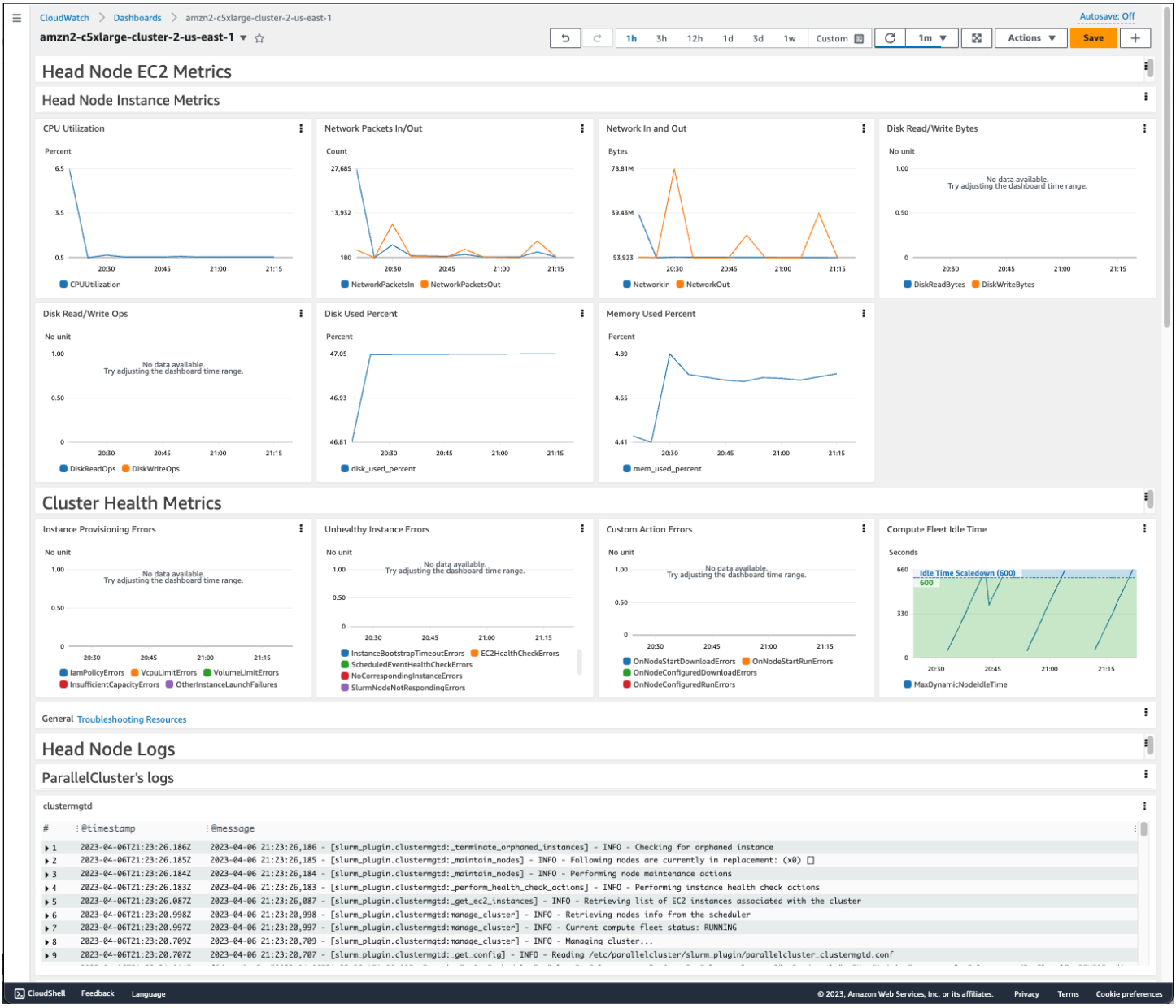
AWS ParallelCluster balises de volume racine du nœud de calcul

Clé de balise	Valeur de balise
`ClusterName`	*`clustername`*
`Application`	`parallelcluster-` *`clustername`*
`aws-parallelcluster-node-type`	`Compute`
`QueueName`	*`queue-name`*
`Version`	*`2.11.9`*

Tableau de CloudWatch bord Amazon

À partir de AWS ParallelCluster la version 2.10.0, un tableau de CloudWatch bord Amazon est créé lors de la création du cluster. Cela facilite la surveillance des nœuds de votre cluster et l'affichage des journaux stockés dans Amazon CloudWatch Logs. Le nom du tableau de bord est`parallelcluster-`*`ClusterName`*`-`*`Region`*. *`ClusterName`*est le nom de votre cluster et *`Region`* le nom Région AWS du cluster. Vous pouvez accéder au tableau de bord dans la console ou en l'ouvrant`https://console.aws.amazon.com/cloudwatch/home?region=`*`Region`*`#dashboards:name=parallelcluster-`*`ClusterName`*.

L'image suivante montre un exemple de CloudWatch tableau de bord pour un cluster.



La première section du tableau de bord affiche des graphiques des EC2 métriques du nœud principal. Si votre cluster dispose d'un stockage partagé, la section suivante présente les métriques de stockage partagé. La dernière section répertorie les journaux Head Node regroupés par journaux, journaux ParallelCluster du planificateur, journaux d'intégration NICE DCV et journaux du système.

Pour plus d'informations sur les CloudWatch tableaux de bord Amazon, consultez la section Utilisation des CloudWatch tableaux de bord Amazon dans le guide de CloudWatch l'utilisateur Amazon.

Si vous ne souhaitez pas créer le tableau de CloudWatch bord Amazon, vous devez suivre les étapes suivantes : ajoutez d'abord une `[dashboard]`section à votre fichier de configuration, puis ajoutez le nom de cette section comme valeur du `dashboard_settings` paramètre dans votre `[cluster]`section. Dans votre `[dashboard]`section, définissez`enable = false`.

Par exemple, si votre `[dashboard]`section est nommée `myDashboard` et que votre `[cluster]`section est nommée`myCluster`, vos modifications ressemblent à ceci.

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

[dashboard MyDashboard]
enable = false
```

Intégration à Amazon CloudWatch Logs

À partir de AWS ParallelCluster la version 2.6.0, les journaux courants sont stockés dans CloudWatch Logs par défaut. Pour plus d'informations sur les CloudWatch journaux, consultez le guide de l'utilisateur d'Amazon CloudWatch Logs. Pour configurer l'intégration CloudWatch des journaux, consultez la `[cw_log]`section et le `cw_log_settings` paramètre.

Un groupe de journaux est créé pour chaque cluster avec un nom `/aws/parallelcluster/`*`cluster-name`* (par exemple,`/aws/parallelcluster/testCluster`). Chaque journal (ou ensemble de journaux si le chemin contient un`*`) sur chaque nœud possède un flux de journal nommé*`{hostname}`*`.`*`{instance_id}`*`.`*`{logIdentifier}`*. (Par exemple`ip-172-31-10-46.i-02587cf29cc3048f3.nodewatcher`.) Les données du journal sont envoyées CloudWatch par l'CloudWatch agent, qui s'exécute comme `root` sur toutes les instances de cluster.

À partir de AWS ParallelCluster la version 2.10.0, un tableau de CloudWatch bord Amazon est créé lors de la création du cluster. Ce tableau de bord permet de consulter facilement les journaux stockés dans CloudWatch Logs. Pour de plus amples informations, veuillez consulter Tableau de CloudWatch bord Amazon.

Cette liste contient le chemin *`logIdentifier`* et le chemin des flux de journaux disponibles pour les plateformes, les planificateurs et les nœuds.

Flux de journaux disponibles pour les plateformes, les planificateurs et les nœuds

Platefores	Schedus	Nœuds	Flux de journaux
amazon centos ubuntu	awsbatc boue	HeadNc	authentificateur DCV : `/var/log/parallelcluster/pcluster_dcv_authenticator.log` dcv-ext-authenticator: `/var/log/parallelcluster/pcluster_dcv_connect.log` agent DCV : `/var/log/dcv/agent.*.log` Séance DCV : `/var/log/dcv/dcv-xsession.*.log` serveur DCV : `/var/log/dcv/server.log` dcv-session-launcher: `/var/log/dcv/sessionlauncher.log` XDCV : `/var/log/dcv/Xdcv.*.log` cfn-init : `/var/log/cfn-init.log` chef-client : `/var/log/chef-client.log`
amazon centos ubuntu	awsbatc boue	Computeet HeadNc	init dans le cloud : `/var/log/cloud-init.log` supervisé : `/var/log/supervisord.log`
amazon centos ubuntu	boue	Computeet	cloud-init-output: `/var/log/cloud-init-output.log` computemgtd : `/var/log/parallelcluster/computemgtd` bouffé : `/var/log/slurmd.log`
amazon centos ubuntu	boue	HeadNc	clustermgtd : `/var/log/parallelcluster/clustermgtd` slurm_resume : `/var/log/parallelcluster/slurm_resume.log`

Platefores	Schedus	Nœuds	Flux de journaux
			slurm_suspend : `/var/log/parallelcluster/slurm_suspend.log` slurmctld : `/var/log/slurmctld.log`
amazon centos	awsbatc boue	Computeet HeadNc	messages du système : `/var/log/messages`
ubuntu	awsbatc boue	Computeet HeadNc	journal système : `/var/log/syslog`

Les tâches des clusters qui l'utilisent AWS Batch stockent le résultat des tâches ayant atteint un `FAILED` état `RUNNINGSUCCEEDED`, ou dans CloudWatch des journaux. Le groupe de journaux est`/aws/batch/job`, et le format du nom du flux de journaux est*`jobDefinitionName`*`/default/`*`ecs_task_id`*. Par défaut, ces journaux sont définis pour ne jamais expirer, mais vous pouvez modifier la période de conservation. Pour plus d'informations, consultez la section Conservation des données du journal des modifications dans CloudWatch Logs du guide de l'utilisateur Amazon CloudWatch Logs.

> **Note**
>
> `chef-client`,`cloud-init-output`,`clustermgtd`, `computemgtd``slurm_resume`, et `slurm_suspend` ont été ajoutés dans la AWS ParallelCluster version 2.9.0. Pour AWS ParallelCluster la version 2.6.0, `/var/log/cfn-init-cmd.log` (`cfn-init-cmd`) et `/var/log/cfn-wire.log` (`cfn-wire`) étaient également stockés dans CloudWatch des journaux.

Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) est un périphérique réseau doté de fonctionnalités de contournement du système d'exploitation pour les communications réseau à faible latence avec d'autres instances sur le même sous-réseau. EFA est exposé à l'aide de Libfabric et peut être utilisé par des applications utilisant l'interface MPI (Messaging Passing Interface).

Pour utiliser EFA avec AWS ParallelCluster, ajoutez la ligne `enable_efa = true` à la `[queue]`section.

Pour consulter la liste des EC2 instances compatibles EFA, consultez la section Types d'instances pris en charge dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux.

Pour plus d'informations sur le `enable_efa` paramètre, consultez `enable_efa`la `[queue]`section.

Un groupe de placement de cluster doit être utilisé pour minimiser les latences entre les instances. Pour plus d'informations, consultez `placement` et `placement_group`.

Pour plus d'informations, consultez Elastic Fabric Adapter dans le guide de EC2 l'utilisateur Amazon et Scale HPC workloads with Elastic Fabric Adapter et AWS ParallelCluster sur le blog AWS Open Source.

> **Note**
>
> Par défaut, les distributions Ubuntu activent la protection ptrace (trace de processus). À partir de AWS ParallelCluster version 2,6.0, la protection ptrace est désactivée afin que Libfabric fonctionne correctement. Pour plus d'informations, consultez la section Désactiver la protection ptrace dans le guide de EC2 l'utilisateur Amazon.

> **Note**
>
> Support EFA sur les instances Graviton2 basées sur ARM a été ajouté dans AWS ParallelCluster la version 2.10.1.

Solutions Intel Select

AWS ParallelCluster est disponible sous forme de solution Intel Select pour la simulation et la modélisation. Les configurations sont vérifiées pour répondre aux normes définies par la spécification de la plate-forme Intel HPC, utiliser des types d'instances Intel spécifiques et sont configurées pour utiliser l'interface réseau Elastic Fabric Adapter (EFA). AWS ParallelCluster est la première solution cloud répondant aux exigences du programme Intel Select Solutions. Les types d'instances pris en charge incluent `c5n.18xlargem5n.24xlarge`, et`r5n.24xlarge`. Un exemple de configuration compatible avec la norme Intel Select Solutions est fourni ci-dessous.

Example Configuration des solutions Intel Select

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

[aws]
aws_region_name = <Your Région 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
```

Pour plus d'informations sur AWS ParallelCluster les spécifications de la plate-forme Intel HPC, consultezSpécifications de plateforme HPC Intel.

Activation d'Intel MPI

Intel MPI est disponible sur le AWS ParallelCluster AMIs. Pour utiliser Intel MPI, vous devez connaître et accepter les termes de la licence logicielle simplifiée Intel. Par défaut, Open MPI est placé sur le chemin. Pour activer Intel MPI au lieu d'Open MPI, vous devez d'abord charger le module Intel MPI. Ensuite, vous devez installer la dernière version en utilisant`module load intelmpi`. Le nom exact du module change avec chaque mise à jour. Pour voir quels modules sont disponibles, exécutez `module avail`. La sortie est la suivante.

```
$ 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
```

Pour savoir quels modules sont chargés, exécutez `module list`.

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

Pour vérifier qu'Intel MPI est activé, exécutez `mpirun --version`.

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

Une fois que le module Intel MPI a été chargé, plusieurs chemins sont modifiés pour utiliser les outils Intel MPI. Pour exécuter le code compilé par les outils Intel MPI, chargez le module Intel MPI en premier.

> **Note**
>
> Intel MPI n'est pas compatible avec les instances basées AWS sur Graviton.

> **Note**
>
> Avant AWS ParallelCluster la version 2.5.0, Intel MPI n'était pas disponible AWS ParallelCluster AMIs dans les régions de Chine (Pékin) et de Chine (Ningxia).

Spécifications de plateforme HPC Intel

AWS ParallelCluster est conforme aux spécifications de la plate-forme Intel HPC. La spécification de la plate-forme Intel HPC fournit un ensemble d'exigences en matière de calcul, de structure,

de mémoire, de stockage et de logiciels pour aider à atteindre un niveau élevé de qualité et de compatibilité avec les charges de travail HPC. Pour plus d'informations, consultez les spécifications de la plate-forme Intel HPC et les applications dont la compatibilité a été vérifiée avec la spécification de la plate-forme Intel HPC.

Pour assurer la conformité aux spécifications de la plateforme HPC Intel, les conditions suivantes doivent être remplies :

- Le système d'exploitation doit être CentOS 7 (`base_os = centos7`).
- Le type d'instance pour les nœuds de calcul doit avoir un processeur Intel et au moins 64 Go de mémoire. Pour la `c5` famille de types d'instances, cela signifie que le type d'instance doit être au moins un `c5.9xlarge` (`compute_instance_type = c5.9xlarge`).
- Le nœud principal doit disposer d'au moins 200 Go de stockage.
- Le contrat de licence de l'utilisateur final pour Intel Parallel Studio doit être accepté (`enable_intel_hpc_platform = true`).
- Chaque nœud de calcul doit avoir au moins 80 Go de stockage (`compute_root_volume_size = 80`).

Le stockage peut être local ou sur un réseau (NFS partagé depuis le nœud principal, Amazon EBS ou FSx pour Lustre), et il peut être partagé.

Bibliothèques de performances Arm

À partir de AWS ParallelCluster la version 2.10.1, les bibliothèques de performances Arm sont disponibles sur les `ubuntu2004` valeurs AWS ParallelCluster AMIs for `alinux2` `centos8ubuntu1804`, et pour le `base_os` paramètre. Les bibliothèques de performances Arm fournissent des bibliothèques mathématiques de base standard optimisées pour les applications informatiques hautes performances sur les processeurs Arm. Pour utiliser les bibliothèques Arm Performance, vous devez connaître et accepter les termes du contrat de licence utilisateur final relatif aux bibliothèques Arm Performance (version gratuite). Pour plus d'informations sur les bibliothèques de performances Arm, consultez Free Arm Performance Libraries.

Pour activer les bibliothèques de performances Arm, vous devez d'abord charger le module Arm Performance Libraries. `Armpl-21.0.0`nécessite GCC-9.3 comme exigence, lorsque vous chargez le `armpl/21.0.0` module, le `gcc/9.3` module sera également chargé. Le nom exact du module change avec chaque mise à jour. Pour voir quels modules sont disponibles, exécutez `module`

`avail`. Ensuite, vous devez installer la dernière version en utilisant`module load armpl`. Le résultat est le suivant.

```
$ 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
```

Pour charger un module, exécutez `module load` *`modulename`*. Vous pouvez ajouter ceci au script utilisé pour exécuter `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
```

Pour savoir quels modules sont chargés, exécutez `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
```

Pour vérifier que les bibliothèques de performances Arm sont activées, exécutez des exemples de tests.

```
$ 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
```

Une fois le module Arm Performance Libraries chargé, plusieurs chemins sont modifiés pour utiliser les outils Arm Performance Libraries. Pour exécuter du code compilé par les outils Arm Performance Libraries, chargez d'abord le module Arm Performance Libraries.

> **Note**
>
> AWS ParallelCluster les versions entre 2.10.1 et 2.10.4 sont utilisées. `armpl/20.2.1`

Connectez-vous au nœud principal via Amazon DCV

Amazon DCV est une technologie de visualisation à distance qui permet aux utilisateurs de se connecter en toute sécurité à des applications 3D gourmandes en graphismes hébergées sur un serveur distant à hautes performances. Pour plus d'informations, consultez Amazon DCV.

Le logiciel Amazon DCV est automatiquement installé sur le nœud principal lors de l'utilisation de `base_os = alinux2` `base_os = centos7`, `base_os = ubuntu1804` ou `base_os = ubuntu2004`.

Si le nœud principal est une instance ARM, le logiciel Amazon DCV y est automatiquement installé lors de l'utilisation de `base_os = alinux2` `base_os = centos7`, ou `base_os = ubuntu1804`.

Pour activer Amazon DCV sur le nœud principal, vous `dcv_settings` devez contenir le nom d'une `[dcv]`section qui possède `enable = master` et `base_os` doit être définie sur `alinux2` `centos7`ubuntu1804, ouubuntu2004. Si le nœud principal est une instance ARM, `base_os` il doit être défini sur `alinux2`centos7, ouubuntu1804. De cette façon, AWS ParallelCluster définit le paramètre de configuration du cluster `shared_dir` sur le dossier de stockage du serveur DCV.

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

Pour plus d'informations sur les paramètres de configuration d'Amazon DCV, consultez`dcv_settings`. Pour vous connecter à la session Amazon DCV, utilisez la `pcluster dcv` commande.

> **Note**
>
> Support pour Amazon DCV activé `centos8` a été supprimé dans la AWS ParallelCluster version 2.10.4. Support pour Amazon DCV on `centos8` a été ajouté dans la AWS ParallelCluster version 2.10.0. Support pour Amazon DCV sur les instances AWS basées sur Graviton a été ajouté dans AWS ParallelCluster la version 2.9.0. Support pour Amazon DCV activé `alinux2` et `ubuntu1804` ajouté dans la AWS ParallelCluster version 2.6.0. Support pour Amazon DCV on `centos7` a été ajouté dans la AWS ParallelCluster version 2.5.0.

> **Note**
>
> Amazon DCV n'est pas pris en charge sur les instances AWS basées sur Graviton dans les AWS ParallelCluster versions 2.8.0 et 2.8.1.

Certificat HTTPS Amazon DCV

Amazon DCV génère automatiquement un certificat auto-signé pour sécuriser le trafic entre le client Amazon DCV et le serveur Amazon DCV.

Pour remplacer le certificat Amazon DCV autosigné par défaut par un autre certificat, connectez-vous d'abord au nœud principal. Ensuite, copiez le certificat et la clé dans le dossier `/etc/dcv` avant d'exécuter la commande `pcluster dcv`.

Pour plus d'informations, consultez la section Modification du certificat TLS dans le manuel Amazon DCV Administrator Guide.

Licence Amazon DCV

Le serveur Amazon DCV n'a pas besoin de serveur de licences lorsqu'il est exécuté sur des EC2 instances Amazon. Cependant, le serveur Amazon DCV doit se connecter régulièrement à un compartiment Amazon S3 pour déterminer si une licence valide est disponible.

AWS ParallelCluster ajoute automatiquement les autorisations requises au`ParallelClusterInstancePolicy`. Lorsque vous utilisez une politique d'instance IAM personnalisée, utilisez les autorisations décrites dans Amazon DCV sur Amazon EC2 dans le guide de l'administrateur Amazon DCV.

Pour obtenir des conseils de dépannage, veuillez consulter Résolution des problèmes dans Amazon DCV.

Utiliser `pcluster update`

À partir de AWS ParallelCluster la version 2.8.0, `pcluster update` analyse les paramètres utilisés pour créer le cluster actuel et les paramètres du fichier de configuration pour détecter les problèmes. Si des problèmes sont découverts, ils sont signalés et les étapes à suivre pour les résoudre sont affichées. Par exemple, si le `compute_instance_type` paramètre est remplacé par un autre type d'instance, le parc de calcul doit être arrêté avant qu'une mise à jour puisse être effectuée. Ce problème est signalé lorsqu'il est découvert. Si aucun problème de blocage n'est signalé, il vous est demandé si vous souhaitez appliquer les modifications.

La documentation de chaque paramètre définit la politique de mise à jour pour ce paramètre.

Politique de mise à jour : Ces paramètres peuvent être modifiés lors d'une mise à jour., Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

> Ces paramètres peuvent être modifiés et le cluster peut être mis à jour à l'aide de`pcluster update`.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

> Ces paramètres ne peuvent pas être modifiés si le cluster existant n'a pas été supprimé. La modification doit être annulée ou le cluster doit être supprimé (en utilisant`pcluster delete`), puis un nouveau cluster doit être créé (en utilisant`pcluster create`) à la place de l'ancien cluster.

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

> Ces paramètres peuvent être modifiés et le cluster mis à jour à l'aide de`pcluster update`.

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

> Ces paramètres ne peuvent pas être modifiés tant que le parc informatique existe. La modification doit être annulée ou le parc informatique doit être arrêté (utilisation`pcluster stop`), mis à jour (utilisation`pcluster update`), puis un nouveau parc informatique créé (utilisation`pcluster start`).

Politique de mise à jour : ce paramètre ne peut pas être réduit lors d'une mise à jour.

> Ces paramètres peuvent être modifiés, mais ils ne peuvent pas être diminués. Si ces paramètres doivent être réduits, il est nécessaire de supprimer le cluster (en utilisant`pcluster delete`) et en créer un nouveau (en utilisant`pcluster create`).

Politique de mise à jour : pour réduire la taille d'une file d'attente en dessous du nombre actuel de nœuds, il faut d'abord arrêter le parc informatique.

> Ces paramètres peuvent être modifiés, mais si la modification réduit la taille de la file d'attente en dessous de la taille actuelle, le parc informatique doit être arrêté (utilisation`pcluster stop`), mis à jour (utilisation`pcluster update`), puis un nouveau parc informatique créé (utilisation`pcluster start`).

Politique de mise à jour : pour réduire le nombre de nœuds statiques dans une file d'attente, il faut d'abord arrêter le parc informatique.

> Ces paramètres peuvent être modifiés, mais si cela réduit le nombre de nœuds statiques dans la file d'attente en dessous de la taille actuelle, le parc informatique doit être arrêté (utilisation`pcluster stop`), mis à jour (utilisation`pcluster update`), puis un nouveau parc informatique créé (utilisation`pcluster start`).

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée. La mise à jour de ce paramètre ne peut pas être forcée.

> Ces paramètres ne peuvent pas être modifiés si le cluster existant n'a pas été supprimé. La modification doit être annulée ou le cluster doit être supprimé (en utilisant`pcluster delete`), puis un nouveau cluster doit être créé (en utilisant`pcluster create`) à la place de l'ancien cluster.

Politique de mise à jour : si les systèmes de fichiers Amazon FSx for Lustre AWS ParallelCluster gérés ne sont pas spécifiés dans la configuration, ce paramètre peut être modifié lors d'une mise à jour.

> Ce paramètre peut être modifié s'il `[cluster]fsx_settings`n'est pas spécifié ou si `fsx_settings` les deux `fsx-fs-id`options `[fsx fs]`sont spécifiées pour monter un système de fichiers externe existant FSx pour Lustre.

Cet exemple illustre certaines modifications qui bloquent la mise à jour. `pcluster update`

```
$ 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.
```

Corriger les AMI et remplacer les EC2 instances

Pour garantir que tous les nœuds de calcul de cluster lancés dynamiquement se comportent de manière cohérente, AWS ParallelCluster désactivez les mises à jour automatiques du système d'exploitation des instances de cluster. En outre, un ensemble spécifique de AWS ParallelCluster AMIs est créé pour chaque version de AWS ParallelCluster et la CLI associée. Cet ensemble spécifique AMIs reste inchangé et ils ne sont pris en charge que par la AWS ParallelCluster version pour laquelle ils ont été conçus. AWS ParallelCluster AMIscar les versions publiées ne sont pas mises à jour.

Toutefois, en raison de problèmes de sécurité émergents, les clients souhaiteront peut-être y ajouter des correctifs, AMIs puis mettre à jour leurs clusters avec l'AMI corrigée. Cela correspond au modèle de responsabilitéAWS ParallelCluster partagée.

Pour afficher l'ensemble spécifique AWS ParallelCluster AMIs pris en charge par la version de la AWS ParallelCluster CLI que vous utilisez actuellement, exécutez :

```
$ pcluster version
```

Affichez ensuite le fichier amis.txt dans AWS ParallelCluster le GitHub référentiel.

Le nœud AWS ParallelCluster principal est une instance statique et vous pouvez le mettre à jour manuellement. Le redémarrage et le redémarrage du nœud principal sont entièrement pris en charge à partir de AWS ParallelCluster la version 2.11, si le type d'instance ne possède pas de magasin d'instance. Pour plus d'informations, consultez la section Types d'instances avec volumes de stockage d'instance dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux. Vous ne pouvez pas mettre à jour une AMI pour un cluster existant.

Le redémarrage du nœud principal et le redémarrage avec les mises à jour de l'AMI des instances de calcul du cluster sont entièrement pris en charge à partir de AWS ParallelCluster la version 3.0.0. Envisagez de passer à la version la plus récente pour utiliser ces fonctionnalités.

Mise à jour ou remplacement de l'instance du nœud principal

Dans certains cas, il peut vous être demandé de redémarrer ou de redémarrer le nœud principal. Par exemple, cela est nécessaire lorsque vous mettez à jour manuellement le système d'exploitation ou lorsqu'une mise hors service planifiée d'une AWS instance impose le redémarrage de l'instance du nœud principal.

Si votre instance ne possède pas de lecteurs éphémères, vous pouvez l'arrêter et la redémarrer à tout moment. En cas de mise hors service planifiée, le démarrage de l'instance arrêtée permet de la faire migrer pour utiliser le nouveau matériel.

De même, vous pouvez arrêter et démarrer manuellement une instance qui ne possède pas de magasins d'instances. Pour ce cas et pour les autres cas d'instances sans volumes éphémères, passez à. Arrêter et démarrer le nœud principal d'un cluster

Si votre instance possède des disques éphémères et qu'elle a été arrêtée, les données du magasin d'instances sont perdues. Vous pouvez déterminer si le type d'instance utilisé pour le nœud principal comporte des magasins d'instance à partir du tableau figurant dans la section Volumes de stockage d'instance.

Les sections suivantes décrivent les limites liées à l'utilisation d'instances avec des volumes de stockage d'instance.

Limites du stockage d'instances

Les limites liées à l'utilisation de AWS ParallelCluster la version 2.11 et des types d'instance avec un magasin d'instances sont les suivantes :

- Lorsque les lecteurs éphémères ne sont pas chiffrés (le `encrypted_ephemeral`paramètre est défini sur `false` ou non), une AWS ParallelCluster instance ne peut pas démarrer après son arrêt. Cela est dû au fait que des informations sur d'anciennes données éphémères inexistantes sont enregistrées `fstab` et que le système d'exploitation essaie de monter un stockage inexistant.
- Lorsque les lecteurs éphémères sont chiffrés (le `encrypted_ephemeral`paramètre est défini sur`true`), une AWS ParallelCluster instance peut être démarrée après un arrêt, mais les nouveaux lecteurs éphémères ne sont pas configurés, montés ou disponibles.
- Lorsque les disques éphémères sont chiffrés, une AWS ParallelCluster instance peut être redémarrée, mais les anciens disques éphémères (qui survivent au redémarrage de l'instance) ne sont pas accessibles car la clé de chiffrement est créée dans la mémoire perdue lors du redémarrage.

Le seul cas pris en charge est le redémarrage de l'instance, lorsque les disques éphémères ne sont pas chiffrés. Cela est dû au fait que le lecteur survit au redémarrage et qu'il est remonté grâce à `fstab` l'entrée écrite.

Solutions de contournement des limites du stockage d'instances

Enregistrez d'abord vos données. Pour vérifier si certaines données doivent être conservées, consultez le contenu du `ephemeral_dir` dossier (`/scratch`par défaut). Vous pouvez transférer les données vers le volume racine ou vers les systèmes de stockage partagés connectés au cluster, tels qu'Amazon FSx, Amazon EFS ou Amazon EBS. Notez que le transfert de données vers un stockage à distance peut entraîner des coûts supplémentaires.

La cause première de ces limitations réside dans la logique AWS ParallelCluster utilisée pour formater et monter les volumes de stockage d'instance. La logique ajoute une entrée `/etc/fstab` au formulaire :

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

`${ephemeral_dir}`est la valeur du `ephemeral_dir` paramètre du fichier de configuration de pcluster (par défaut). `/scratch`

Cette ligne est ajoutée afin que si ou lorsqu'un nœud est redémarré, les volumes de stockage d'instance soient automatiquement remontés. Cela est souhaitable car les données contenues dans les disques éphémères persistent après le redémarrage. Cependant, les données des disques éphémères ne sont pas conservées pendant un cycle de démarrage ou d'arrêt. Cela signifie qu'ils sont formatés et montés sans aucune donnée.

Le seul cas pris en charge est le redémarrage de l'instance lorsque les disques éphémères ne sont pas chiffrés. Cela est dû au fait que le lecteur survit au redémarrage et est remonté parce qu'il est écrit`fstab`.

Pour préserver les données dans tous les autres cas, vous devez supprimer l'entrée de volume logique avant d'arrêter l'instance. Par exemple, supprimez `/dev/vg.01/lv_ephemeral` de `/etc/fstab` avant d'arrêter l'instance. Ensuite, vous démarrez l'instance sans monter les volumes éphémères. Cependant, le montage du magasin d'instance ne sera à nouveau plus disponible après l'arrêt ou le démarrage de l'instance.

Après avoir enregistré vos données, puis supprimé l'`fstab`entrée, passez à la section suivante.

Arrêter et démarrer le nœud principal d'un cluster

> **Note**
>
> À partir de AWS ParallelCluster la version 2.11, l'arrêt et le démarrage du nœud principal ne sont pris en charge que si le type d'instance ne possède pas de magasin d'instance.

1. Vérifiez qu'aucune tâche n'est en cours d'exécution dans le cluster.

   Lorsque vous utilisez un Slurm planificateur :

   - Si l'`sbatch--no-requeue`option n'est pas spécifiée, les tâches en cours d'exécution sont requises.
   - Si l'`--no-requeue`option est spécifiée, les tâches en cours d'exécution échouent.

2. Demandez l'arrêt d'un parc de calcul en 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. Attendez que l'état du parc informatique soit le suivant `STOPPED` :

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

4. Pour les mises à jour manuelles avec redémarrage du système d'exploitation ou redémarrage d'une instance, vous pouvez utiliser le AWS Management Console ou AWS CLI. Voici un exemple d'utilisation du 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. Démarrez le parc de calcul du 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 Commandes CLI

`pcluster`et `pcluster-config` sont les commandes de la AWS ParallelCluster CLI. Vous pouvez `pcluster` lancer et gérer des clusters HPC dans AWS Cloud et `pcluster-config` pour mettre à jour votre configuration.

Pour l'utiliser`pcluster`, vous devez disposer d'un rôle IAM doté des autorisations requises pour l'exécuter.

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

Rubriques

- pcluster
- pcluster-config

`pcluster`

`pcluster`est la principale commande de la AWS ParallelCluster CLI. Vous pouvez `pcluster` lancer et gérer des clusters HPC dans le AWS Cloud.

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

Arguments

`pcluster` *`command`*

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

Sous-commandes :

Rubriques

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

pcluster configure

Commence une AWS ParallelCluster configuration. Pour de plus amples informations, veuillez consulter Configuration AWS ParallelCluster.

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

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster configure`.

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

Spécifie le chemin complet de l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

Pour de plus amples informations, veuillez consulter Configuration AWS ParallelCluster.

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

Spécifie le Région AWS à utiliser. Si cela est spécifié, la configuration ignore la Région AWS détection.

Pour supprimer les ressources réseau du VPC, vous pouvez supprimer la pile CloudFormation réseau. Le nom de la pile commence par »parallelclusternetworking-» et contient l'heure de création au format « YYYYMMDDHHMMSS ». Vous pouvez répertorier les piles à l'aide de la commande list-stacks.

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

La pile peut être supprimée à l'aide de la commande delete-stack.

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

Le VPC `pcluster configure` créé pour vous n'est pas créé dans la pile CloudFormation réseau. Vous pouvez supprimer ce VPC manuellement dans la console ou à l'aide du. AWS CLI

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

pcluster create

Crée un nouveau cluster.

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

Arguments positionnels

cluster_name

Définit le nom du cluster. Le nom de la AWS CloudFormation pile est`parallelcluster-`*`cluster_name`*.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster create`.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

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

Spécifie le Région AWS à utiliser. L'ordre de priorité utilisé pour sélectionner le Région AWS pour un nouveau cluster est le suivant :

1. `-r`ou `--region` paramètre to`pcluster create`.
2. `AWS_DEFAULT_REGION`variable d'environnement.
3. `aws_region_name`paramètre dans `[aws]` la section du fichier de AWS ParallelCluster configuration (l'emplacement par défaut est`~/.parallelcluster/config`.) Il s'agit de l'emplacement mis à jour par la `pcluster configure` commande.
4. `region`réglage dans `[default]` la section du fichier de AWS CLI configuration (`~/.aws/config`.)

-nw, --nowait

Indique de ne pas attendre les événements de pile après avoir exécuté une commande de pile.

La valeur par défaut est `False`.

-nr, --norollback

Désactive la restauration de la pile en cas d'erreur.

La valeur par défaut est `False`.

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

Spécifie une URL pour le AWS CloudFormation modèle personnalisé s'il a été utilisé lors de sa création.

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

Indique le cluster de modèle à utiliser.

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

Ajoute des paramètres supplémentaires pour créer la pile.

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

Spécifie des balises supplémentaires à ajouter à la pile.

Lorsque la commande est appelée et commence à demander l'état de cet appel, vous pouvez utiliser « Ctrl-C » pour quitter l'appel en toute sécurité. Vous pouvez revenir à l'affichage du statut actuel en appelant `pcluster status mycluster`.

Exemples utilisant AWS ParallelCluster la version 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) Crée une AMI personnalisée à utiliser avec. 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 ]
```

Dépendances requises

Outre la AWS ParallelCluster CLI, la dépendance suivante est requise pour fonctionner `pcluster createami` :

- Packer : Téléchargez la dernière version à partir de https://developer.hashicorp.com/packer/downloads.

> **Note**
>
> Avant AWS ParallelCluster la version 2.8.0, Berkshelf (installé à l'aide de`gem install berkshelf`) devait être utilisé. `pcluster createami`

Arguments nommés

`-h, --help`

Affiche le texte d'aide pour`pcluster createami`.

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

Spécifie l'AMI de base à utiliser pour créer l' AWS ParallelCluster AMI.

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

Spécifie le système d'exploitation de l'AMI de base. Les options valides sont : `alinux2`, `ubuntu1804`, `ubuntu2004` et `centos7`.

> **Note**
>
> Le système d'exploitation prend en charge les modifications dans AWS ParallelCluster les différentes versions :
>
> - Support pour `centos8` a été supprimé dans la AWS ParallelCluster version 2.10.4.
> - Support pour `centos8` a été ajouté, et le support pour `centos6` a été supprimé dans la AWS ParallelCluster version 2.10.0.
> - Support pour alinux2 a été ajouté dans AWS ParallelCluster la version 2.6.0.
> - Support pour ubuntu1804 a été ajouté dans la version 2.5.0. AWS ParallelCluster

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

Spécifie le type d'instance à utiliser pour créer l'AMI.

La valeur par défaut est `t2.xlarge`.

> **Note**
>
> Support pour `--instance-type` cet argument a été ajouté dans la AWS ParallelCluster version 2.4.1.

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

Spécifie le nom du préfixe de l' AWS ParallelCluster AMI résultante.

La valeur par défaut est `custom-ami-`.

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

Spécifie le livre de recettes à utiliser pour créer l' AWS ParallelCluster AMI.

`--post-install POST_INSTALL_SCRIPT`

Spécifie le chemin d'accès au script de post-installation. Les chemins doivent utiliser un schéma d'`file://`URL `s3://`` https://`, ou. Voici quelques exemples :

- `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 pour `--post-install` cet argument a été ajouté dans la AWS ParallelCluster version 2.10.0.

`--no-public-ip`

N'associez pas d'adresse IP publique à l'instance utilisée pour créer l'AMI. Par défaut, une adresse IP publique est associée à l'instance.

> **Note**
>
> Support pour `--no-public-ip` cet argument a été ajouté dans la AWS ParallelCluster version 2.5.0.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

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

Spécifie la section `[cluster]` *`CONFIG_FILE`* à utiliser pour récupérer les paramètres du VPC et du sous-réseau.

> **Note**
>
> Support pour `--cluster-template` cet argument a été ajouté dans la AWS ParallelCluster version 2.4.0.

`--vpc-id` *`VPC_ID`*

Spécifie l'ID du VPC à utiliser pour créer l'AMI AWS ParallelCluster .

> **Note**
>
> Support pour `--vpc-id` cet argument a été ajouté dans la AWS ParallelCluster version 2.5.0.

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

Spécifie l'ID du sous-réseau à utiliser pour créer l' AWS ParallelCluster AMI.

> **Note**
>
> Support pour `--vpc-id` cet argument a été ajouté dans la AWS ParallelCluster version 2.5.0.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

pcluster dcv

Interagit avec le serveur Amazon DCV exécuté sur le nœud principal.

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

pcluster dcv *command*

Choix possibles : `connect`

> **Note**
>
> Le système d'exploitation prend en charge les modifications apportées à la `pcluster dcv` commande dans différentes AWS ParallelCluster versions :
>
> - Support pour la `pcluster dcv` commande on `centos8` a été ajouté dans la AWS ParallelCluster version 2.10.0.
> - Support de la `pcluster dcv` commande sur les instances AWS basées sur Graviton a été ajouté dans la AWS ParallelCluster version 2.9.0.
> - Support pour la `pcluster dcv` commande on `ubuntu1804` a été ajouté dans la AWS ParallelCluster version 2.6.0.
> - Support pour la `pcluster dcv` commande on `centos7` a été ajouté dans la AWS ParallelCluster version 2.5.0.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster dcv`.

Sous-commandes

pcluster dcv connect

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

> **Important**
>
> L'URL expire 30 secondes après sa publication. Si la connexion n'est pas établie avant l'expiration de l'URL, `pcluster dcv connect` réexécutez pour générer une nouvelle URL.

Arguments positionnels

cluster_name

Spécifie le nom du cluster auquel se connecter.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster dcv connect`.

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

Chemin de la clé SSH à utiliser pour la connexion.

La clé doit être celle spécifiée au moment de la création du cluster dans le paramètre de configuration `key_name`. Cet argument est facultatif, mais s'il n'est pas spécifié, la clé doit être disponible par défaut pour le client SSH. Par exemple, ajoutez-le à `ssh-agent` à l'aide de `ssh-add`.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

-s, --show-url

Affiche une URL à usage unique permettant de se connecter à la session Amazon DCV. Le navigateur par défaut n'est pas ouvert lorsque cette option est spécifiée.

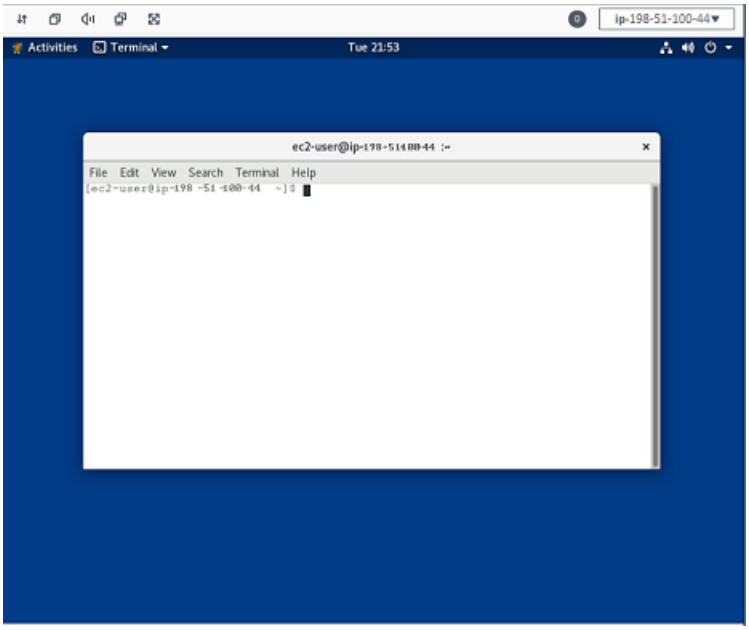
> **Note**
>
> Support pour `--show-url` cet argument a été ajouté dans la AWS ParallelCluster version 2.5.1.

Exemple utilisant AWS ParallelCluster la version 2.11.7 :

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

Ouvre le navigateur par défaut pour se connecter à la session Amazon DCV exécutée sur le nœud principal.

Une nouvelle session Amazon DCV est créée si ce n'est pas déjà fait.



pcluster delete

Supprime un cluster.

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

Arguments positionnels

cluster_name

Spécifie le nom du cluster à supprimer.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster delete`.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

--keep-logs

Conservez les données des CloudWatch journaux après avoir supprimé le cluster. Le groupe de journaux est conservé jusqu'à ce que vous le supprimiez manuellement, mais les événements du journal expirent en fonction du `retention_days` paramètre. Le paramètre par défaut est de 14 jours.

> **Note**
>
> Support pour **`--keep-logs`** cet argument a été ajouté dans la AWS ParallelCluster version 2.6.0.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

Lorsque la commande est appelée et commence à demander l'état de cet appel, vous pouvez utiliser « Ctrl-C » pour quitter l'appel en toute sécurité. Vous pouvez revenir à l'affichage du statut actuel en appelant `pcluster status mycluster`.

Exemple utilisant AWS ParallelCluster la version 2.11.7 :

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

Pour supprimer les ressources réseau du VPC, vous pouvez supprimer la pile CloudFormation réseau. Le nom de la pile commence par »parallelclusternetworking-» et contient l'heure de création au format « YYYYMMDDHHMMSS ». Vous pouvez répertorier les piles à l'aide de la commande list-stacks.

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

La pile peut être supprimée à l'aide de la commande delete-stack.

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

Le VPC `pcluster configure` créé pour vous n'est pas créé dans la pile CloudFormation réseau. Vous pouvez supprimer ce VPC manuellement dans la console ou à l'aide du. AWS CLI

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

pcluster instances

Affiche la liste de toutes les instances d'un cluster.

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

Arguments positionnels

cluster_name

Affiche les instances du cluster dont le nom est fourni.

Arguments nommés

`-h, --help`

Affiche le texte d'aide pour`pcluster instances`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

`-r REGION, --region REGION`

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

Exemple utilisant AWS ParallelCluster la version 2.11.7 :

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

`pcluster list`

Affiche la liste des piles associées AWS ParallelClusterà.

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

Arguments nommés

`-h, --help`

Affiche le texte d'aide pour`pcluster list`.

`--color`

Affiche le statut du cluster en couleur.

La valeur par défaut est `False`.

`-c CONFIG_FILE, --config CONFIG_FILE`

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `c`.

`-r REGION, --region REGION`

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

Répertorie le nom de toutes les AWS CloudFormation piles nommées`parallelcluster-*`.

Exemple utilisant AWS ParallelCluster la version 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

Exécute une commande `ssh` avec le nom d'utilisateur et l'adresse IP du cluster préremplis. Des arguments arbitraires sont ajoutés à la fin de la commande `ssh`. Cette commande peut être personnalisée dans la section des alias du fichier de configuration.

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

Arguments positionnels

`cluster_name`

Spécifie le nom du cluster auquel se connecter.

Arguments nommés

`-h, --help`

Affiche le texte d'aide pour`pcluster ssh`.

`-d, --dryrun`

Imprime la commande qui sera exécutée et quitte.

La valeur par défaut est `False`.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est la région spécifiée à l'aide de la commande `pcluster configure`.

Exemples utilisant AWS ParallelCluster la version 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
```

Exécute la `ssh` commande avec le nom d'utilisateur et l'adresse IP du cluster préremplis :

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

La commande `ssh` est définie dans le fichier de configuration globale sous `[aliases]` Section. Elle peut être personnalisée comme suit.

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

Variables substituées :

`CFN_USER`

Nom d'utilisateur pour le `base_os` sélectionné.

`MASTER_IP`

Adresse IP du nœud principal.

`ARGS`

Arguments facultatifs à transmettre à la commande `ssh`.

pcluster start

Démarre le parc d'instances de calcul pour un cluster qui a été arrêté.

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

Arguments positionnels

cluster_name

Démarre le parc d'instances de calcul du nom de cluster fourni.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster start`.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

Exemple utilisant AWS ParallelCluster la version 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
```

Cette commande définit les paramètres du groupe Auto Scaling comme suit :

- Les valeurs de configuration initiale (`max_queue_size` et `initial_queue_size`) à partir du modèle qui a été utilisé pour créer le cluster.
- Les valeurs de configuration qui ont été utilisées pour mettre à jour le cluster depuis sa première création.

pcluster status

Extrait le statut actuel du cluster.

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

Arguments positionnels

cluster_name

Indique le statut du cluster dont le nom est fourni.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster status`.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

-nw, --nowait

Indique de ne pas attendre les événements de pile après le traitement d'une commande de pile.

La valeur par défaut est `False`.

Exemple utilisant AWS ParallelCluster la version 2.11.7 :

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

pcluster stop

Arrête le parc informatique, laissant le nœud principal fonctionner.

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

Arguments positionnels

cluster_name

Arrête le parc d'instances de calcul du nom de cluster fourni.

Exemple utilisant AWS ParallelCluster la version 2.11.7 :

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster stop`.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

```
$ 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
```

Définit les paramètres du groupe Auto Scaling sur min/max/desired = 0/0/0 et met fin au parc informatique. La tête continue de courir. Pour mettre fin à toutes les EC2 ressources et éviter EC2 des frais, pensez à supprimer le cluster.

pcluster update

Analyse le fichier de configuration pour déterminer si le cluster peut être mis à jour en toute sécurité. Si l'analyse détermine que le cluster peut être mis à jour, vous êtes invité à confirmer la modification. Si l'analyse montre que le cluster ne peut pas être mis à jour, les paramètres de configuration à

l'origine des conflits sont énumérés avec des détails. Pour de plus amples informations, veuillez consulter Utiliser `pcluster update`.

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

Arguments positionnels

cluster_name

Spécifie le nom du cluster à mettre à jour.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster update`.

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

Spécifie l'autre fichier de configuration à utiliser.

La valeur par défaut est `~/.parallelcluster/config`.

--force

Active une mise à jour même si un ou plusieurs paramètres présentent une modification bloquante ou si une action en suspens est requise (telle que l'arrêt du parc informatique) avant que la mise à jour ne puisse se poursuivre. Cela ne doit pas être combiné avec l'`--yes`argument.

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

Spécifie le Région AWS à utiliser. La valeur par défaut est celle Région AWS spécifiée à l'aide de la `pcluster configure` commande.

-nr, --norollback

Désactive la restauration de la AWS CloudFormation pile en cas d'erreur.

La valeur par défaut est `False`.

-nw, --nowait

Indique de ne pas attendre les événements de pile après le traitement d'une commande de pile.

La valeur par défaut est `False`.

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

Spécifie la section du modèle de cluster à utiliser.

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

Ajoute des paramètres supplémentaires à une mise à jour de pile.

`-rd, --reset-desired`

Réinitialise la capacité actuelle d'un groupe Auto Scaling sur les valeurs de configuration initiale.

La valeur par défaut est `False`.

`--yes`

Suppose automatiquement que la réponse à toutes les questions est « oui ». Cela ne doit pas être combiné avec l'`--force`argument.

```
$ 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
```

Lorsque la commande est appelée et commence à demander l'état de cet appel, vous pouvez utiliser « Ctrl-C » pour quitter l'appel en toute sécurité. Vous pouvez revenir à l'affichage du statut actuel en appelant `pcluster status mycluster`.

pcluster version

Affiche la AWS ParallelCluster version.

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

Pour les indicateurs spécifiques à la commande, exécutez : `pcluster [command] --help`.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster version`.

Lorsque la commande est appelée et commence à demander l'état de cet appel, vous pouvez utiliser « Ctrl-C » pour quitter l'appel en toute sécurité. Vous pouvez revenir à l'affichage du statut actuel en appelant `pcluster status mycluster`.

```
$ pcluster version
2.11.7
```

pcluster-config

Met à jour le fichier AWS ParallelCluster de configuration.

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

Pour les indicateurs spécifiques à la commande, exécutez : `pcluster-config [command] -h`.

Arguments nommés

-h, --help

Affiche le texte d'aide pour`pcluster-config`.

> **Note**
>
> La `pcluster-config` commande a été ajoutée dans la AWS ParallelCluster version 2.9.0.

Sous-commandes

`pcluster-config convert`

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

Arguments nommés

`-h, --help`

Affiche le texte d'aide pour`pcluster-config convert`.

`-c CONFIG_FILE, --config-file CONFIG_FILE`

Spécifie le chemin du fichier de configuration à lire.

La valeur par défaut est `~/.parallelcluster/config`.

Pour de plus amples informations, veuillez consulter Configuration AWS ParallelCluster.

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

Indique le `[cluster]` Section à utiliser. Si cet argument n'est pas spécifié, `pcluster-config convert` utilisera le `cluster_template` paramètre défini dans le`[global]` Section. Si cela n'est pas spécifié, la `[cluster default]` section est utilisée.

`-o OUTPUT_FILE, --output OUTPUT_FILE`

Spécifie le chemin du fichier de configuration converti à écrire. Par défaut, la sortie est écrite dans`STDOUT`.

Exemple :

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

Convertit la configuration de cluster spécifiée dans la `[cluster alpha]` section de`~/.parallelcluster/config`, en écrivant le fichier de configuration converti dans`~/.parallelcluster/multiinstance`.

Configuration

Par défaut, AWS ParallelCluster utilise le `~/.parallelcluster/config` fichier pour tous les paramètres de configuration. Vous pouvez spécifier un fichier de configuration personnalisé à l'aide de l'option de ligne de `--config` commande `-c` ou de la variable d'`AWS_PCLUSTER_CONFIG_FILE`environnement.

Un exemple de fichier de configuration est installé AWS ParallelCluster dans le répertoire Python à l'adresse`site-packages/aws-parallelcluster/examples/config`. L'exemple de fichier de configuration est également disponible sur GitHub, à l'adressehttps://github.com/aws/aws-parallelcluster/blob/v2.11.9/cli/src/pcluster/examples/config.

Version AWS ParallelCluster 2 actuelle : 2.11.9.

Rubriques

- Disposition
- [global] Section
- [aws] Section
- [aliases] Section
- [cluster] Section
- [compute_resource] Section
- [cw_log] Section
- [dashboard] Section
- [dcv] Section
- [ebs] Section
- [efs] Section
- [fsx] Section
- [queue] Section
- [raid] Section
- [scaling] Section
- [vpc] Section
- Exemples

Disposition

Une AWS ParallelCluster configuration est définie dans plusieurs sections.

Les sections suivantes sont obligatoires : `[global]`section et `[aws]`section.

Vous devez également inclure au moins une `[cluster]`section et une `[vpc]`section.

Une section commence avec le nom de section entre parenthèses, suivi des paramètres et de la configuration.

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

`[global]` Section

Spécifie les options de configuration globales associées à `pcluster`.

```
[global]
```

Rubriques

- cluster_template
- update_check
- sanity_check

`cluster_template`

Définit le nom de la `cluster` section utilisée par défaut pour le cluster. Pour plus d'informations sur `cluster` les sections, consultez `[cluster]`la section. Le nom du cluster doit commencer par une lettre, ne pas contenir plus de 60 caractères et uniquement contenir des lettres, des chiffres et des tirets (-).

Par exemple, le paramètre suivant spécifie que la section qui commence par `[cluster default]` est utilisée par défaut.

```
cluster_template = default
```

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

`update_check`

(Facultatif) Vérifie les mises à jour dep`cluster`.

La valeur par défaut est `true`.

```
update_check = true
```

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

`sanity_check`

(Facultatif) Tente de valider la configuration des ressources définies dans les paramètres du cluster.

La valeur par défaut est `true`.

> **Warning**
>
> S'il `sanity_check` est défini sur`false`, les vérifications importantes sont ignorées. Cela peut empêcher votre configuration de fonctionner comme prévu.

```
sanity_check = true
```

> **Note**
>
> Avant AWS ParallelCluster la version 2.5.0, la `sanity_check` valeur par défaut était. `false`

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

`[aws]` Section

(Facultatif) Utilisé pour sélectionner le Région AWS.

La création d'un cluster utilise cet ordre de priorité pour sélectionner le Région AWS pour un nouveau cluster :

1. `-r` ou `--region` paramètre to `pcluster create`.
2. `AWS_DEFAULT_REGION` variable d'environnement.
3. `aws_region_name` paramètre dans `[aws]` la section du fichier de AWS ParallelCluster configuration (l'emplacement par défaut est `~/.parallelcluster/config`.) Il s'agit de l'emplacement mis à jour par la `pcluster configure` commande.
4. `region` réglage dans `[default]` la section du fichier de AWS CLI configuration (`~/.aws/config`.)

> **Note**
>
> Avant AWS ParallelCluster la version 2.10.0, ces paramètres étaient obligatoires et appliqués à tous les clusters.

Pour stocker les informations d'identification, vous pouvez utiliser l'environnement, les rôles IAM pour Amazon ou le EC2 AWS CLI, plutôt que d'enregistrer les informations d'identification dans le fichier de AWS ParallelCluster configuration.

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

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

`[aliases]` Section

Spécifie les alias, et vous permet de personnaliser la commande `ssh`.

Notez les paramètres par défaut suivants :

- `CFN_USER` est défini sur le nom d'utilisateur par défaut du système d'exploitation
- `MASTER_IP` est défini sur l'adresse IP du nœud principal
- `ARGS` est défini sur tous les arguments fournis par l'utilisateur après *`pcluster ssh cluster_name`*

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

```
# ssh alias here
ssh = ssh {CFN_USER}@{MASTER_IP} {ARGS}
```

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

`[cluster]` Section

Définit un modèle de cluster qui peut être utilisé pour créer un cluster. Un fichier de configuration peut contenir plusieurs `[cluster]` sections.

Le même modèle de cluster peut être utilisé pour créer plusieurs clusters.

Le format est `[cluster cluster-template-name]`. La `[cluster]`section nommée par le `cluster_template` paramètre de la `[global]`section est utilisée par défaut, mais elle peut être remplacée sur la `pcluster` ligne de commande.

*`cluster-template-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

```
[cluster default]
```

Rubriques

- 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

(Facultatif) Définit un AWS CloudFormation modèle supplémentaire à lancer avec le cluster. Ce modèle supplémentaire est utilisé pour créer des ressources extérieures au cluster mais qui font partie du cycle de vie du cluster.

La valeur doit être une URL HTTP vers un modèle public, avec tous les paramètres fournis.

Il n'existe aucune valeur par défaut.

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

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

additional_iam_policies

(Facultatif) Spécifie une liste de noms de ressources Amazon (ARNs) des politiques IAM pour Amazon EC2. Cette liste est attachée au rôle root utilisé dans le cluster, en plus des autorisations requises, AWS ParallelCluster séparées par des virgules. Le nom d'une politique IAM et son ARN sont différents. Les noms ne peuvent pas être utilisés comme argument pour`additional_iam_policies`.

Si votre intention est d'ajouter des politiques supplémentaires aux paramètres par défaut des nœuds de cluster, nous vous recommandons de transmettre les politiques IAM personnalisées supplémentaires avec le `additional_iam_policies` paramètre au lieu de `ec2_iam_role`les utiliser pour ajouter vos EC2 politiques spécifiques. Cela est dû au fait que `additional_iam_policies` sont ajoutés aux autorisations par défaut AWS ParallelCluster

requises. Un document existant `ec2_iam_role`doit inclure toutes les autorisations requises. Cependant, étant donné que les autorisations requises changent souvent d'une version à l'autre au fur et à mesure que des fonctionnalités sont ajoutées, une version existante `ec2_iam_role`peut devenir obsolète.

Il n'existe aucune valeur par défaut.

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

> **Note**
>
> Support pour `additional_iam_policies`a été ajouté dans la AWS ParallelCluster version 2.5.0.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`base_os`

(Obligatoire)Spécifie le type de système d'exploitation utilisé dans le cluster.

Les options disponibles sont :

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

> **Note**
>
> Pour les instances AWS basées sur Graviton, uniquement `alinux2``ubuntu1804`, ou `ubuntu2004` sont prises en charge.

> **Note**
>
> Support pour `centos8` a été supprimé dans la AWS ParallelCluster version 2.11.4. Le support pour `ubuntu2004` a été ajouté et le support pour `alinux` et `ubuntu1604` a été

supprimé dans la AWS ParallelCluster version 2.11.0. Support pour `centos8` a été ajouté et le support pour `centos6` a été supprimé dans la AWS ParallelCluster version 2.10.0. Support pour `alinux2` a été ajouté dans la AWS ParallelCluster version 2.6.0. Support pour `ubuntu1804` a été ajouté, et le support pour `ubuntu1404` a été supprimé dans la AWS ParallelCluster version 2.5.0.

À l'exception des informations spécifiques Régions AWS mentionnées dans le tableau suivant qui ne sont pas prises en charge`centos7`. Toutes les autres régions AWS commerciales prennent en charge tous les systèmes d'exploitation suivants.

Cloison (Régions AWS)	**`alinux2`**	**`centos7`**	**`ubuntu1804`** et **`ubuntu2004`**
Commercial (tout cela Régions AWS n'est pas spécifiquement mentionné)	True	True	True
AWS GovCloud (USA Est) (`us-gov-east-1`)	True	False	True
AWS GovCloud (US-Ouest) (`us-gov-west-1`)	True	False	True
Chine (Beijing) (`cn-north-1`)	True	False	True
Chine (Ningxia) (`cn-northwest-1`)	True	False	True

> **Note**
>
> Le `base_os` paramètre détermine également le nom d'utilisateur utilisé pour se connecter au cluster.

- `centos7`: `centos`
- `ubuntu1804` et `ubuntu2004` : `ubuntu`

- `alinux2`: `ec2-user`

> **Note**
>
> Avant AWS ParallelCluster la version 2.7.0, le `base_os` paramètre était facultatif et le paramètre par défaut était`alinux`. À partir de AWS ParallelCluster la version 2.7.0, le `base_os` paramètre est obligatoire.

> **Note**
>
> Si le paramètre `scheduler` est `awsbatch`, seul `alinux2` est pris en charge.

```
base_os = alinux2
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`cluster_resource_bucket`

(Facultatif) Spécifie le nom du compartiment Amazon S3 utilisé pour héberger les ressources générées lors de la création du cluster. Le contrôle de version du bucket doit être activé. Pour plus d'informations, consultez la section Utilisation du versionnement dans le guide de l'utilisateur d'Amazon Simple Storage Service. Ce bucket peut être utilisé pour plusieurs clusters. Le compartiment doit se trouver dans la même région que le cluster.

Si ce paramètre n'est pas spécifié, un nouveau compartiment est créé lors de la création du cluster. Le nouveau compartiment porte le nom de`parallelcluster-`*`random_string`*. Dans ce nom, *`random_string`* il y a une chaîne aléatoire de caractères alphanumériques. Toutes les ressources du cluster sont stockées dans ce compartiment dans un chemin associé au formulaire*`bucket_name`*`/`*`resource_directory`*. `resource_directory`a la forme*`stack_name`*`-`*`random_string`*, où *`stack_name`* est le nom de l'une des AWS CloudFormation piles utilisées par AWS ParallelCluster. La valeur de *`bucket_name`* se trouve dans la `ResourcesS3Bucket` valeur de la sortie de la `parallelcluster-`*`clustername`* pile. La valeur de *`resource_directory`* se trouve dans la valeur de la `ArtifactS3RootDirectory` sortie de la même pile.

La valeur par défaut est `parallelcluster-random_string`.

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

> **Note**
>
> Support pour `cluster_resource_bucket` a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée. La mise à jour de ce paramètre ne peut pas être forcée.

`cluster_type`

(Facultatif) Définit le type de cluster à lancer. Si le `queue_settings` paramètre est défini, il doit être remplacé par les `compute_type` paramètres des `[queue]`sections.

Les options valides sont `ondemand` et `spot`.

La valeur par défaut est `ondemand`.

Pour plus d'informations sur les instances Spot, consultez Utilisation de instances Spot.

> **Note**
>
> L'utilisation d'instances Spot nécessite que le rôle `AWSServiceRoleForEC2Spot` lié au service existe dans votre compte. Pour créer ce rôle dans votre compte à l'aide de AWS CLI, exécutez la commande suivante :
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> Pour plus d'informations, consultez la section Rôle lié au service pour les demandes d'instance Spot dans le guide de EC2 l'utilisateur Amazon.

```
cluster_type = ondemand
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

compute_instance_type

(Facultatif) Définit le type d' EC2 instance Amazon utilisé pour les nœuds de calcul du cluster. L'architecture du type d'instance doit être identique à celle utilisée pour le `master_instance_type` paramètre. Si le `queue_settings` paramètre est défini, il doit être remplacé par les `instance_type` paramètres des `[compute_resource]`sections.

Si vous utilisez le `awsbatch` planificateur, consultez la section Création d'environnements de calcul dans l' AWS Batch interface utilisateur pour obtenir la liste des types d'instances pris en charge.

Valeur par défaut `t2.micro`, `optimal` lorsque le planificateur est `awsbatch`.

```
compute_instance_type = t2.micro
```

> **Note**
>
> Support pour les instances AWS basées sur Graviton (y compris les `C6g` instances `A1` et) a été ajouté dans la AWS ParallelCluster version 2.8.0.

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

compute_root_volume_size

(Facultatif) Spécifie la taille du volume ComputeFleet racine en gibioctets (GiB). L'AMI doit prendre en charge `growroot`.

La valeur par défaut est `35`.

> **Note**
>
> Pour AWS ParallelCluster les versions entre 2.5.0 et 2.10.4, la valeur par défaut était 25. Avant AWS ParallelCluster la version 2.5.0, la valeur par défaut était 20.

```
compute_root_volume_size = 35
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

custom_ami

(Facultatif) Spécifie l'ID d'une AMI personnalisée à utiliser pour les nœuds de tête et de calcul au lieu de l'ID publié par défaut AMIs. Pour plus d'informations, consultez Modification d'une AMI ou Création d'une AWS ParallelCluster AMI personnalisée.

Il n'existe aucune valeur par défaut.

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

Si l'AMI personnalisée nécessite des autorisations supplémentaires pour son lancement, ces autorisations doivent être ajoutées aux politiques de l'utilisateur et du nœud principal.

Par exemple, si un instantané chiffré est associé à une AMI personnalisée, les politiques supplémentaires suivantes sont requises à la fois dans les politiques de l'utilisateur et du nœud principal :

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

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

cw_log_settings

(Facultatif) Identifie la `[cw_log]` section avec la configuration CloudWatch des journaux. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez les `[cw_log]`sectionsTableau de CloudWatch bord Amazon, etIntégration à Amazon CloudWatch Logs.

Par exemple, le paramètre suivant indique que la section qui démarre `[cw_log custom-cw]` est utilisée pour la configuration CloudWatch des journaux.

```
cw_log_settings = custom-cw
```

> **Note**
>
> Support pour `cw_log_settings` a été ajouté dans la AWS ParallelCluster version 2.6.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

dashboard_settings

(Facultatif) Identifie la `[dashboard]` section avec la configuration du CloudWatch tableau de bord. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[dashboard]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[dashboard custom-dashboard` est utilisée pour la configuration du CloudWatch tableau de bord.

```
dashboard_settings = custom-dashboard
```

> **Note**
>
> Support pour `dashboard_settings` a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

dcv_settings

(Facultatif) Identifie la `[dcv]` section avec la configuration Amazon DCV. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[dcv]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[dcv custom-dcv]` est utilisée pour la configuration Amazon DCV.

```
dcv_settings = custom-dcv
```

> **Note**
>
> Sur les instances AWS basées sur Graviton, Amazon DCV n'est pris en charge que sur. `alinux2`

> **Note**
>
> Support pour `dcv_settings` a été ajouté dans la AWS ParallelCluster version 2.5.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

desired_vcpus

(Facultatif) Spécifie le nombre de v souhaité CPUs dans l'environnement de calcul. Utilisé uniquement si le planificateur est `awsbatch`.

La valeur par défaut est `4`.

```
desired_vcpus = 4
```

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

disable_cluster_dns

(Facultatif) Spécifie si les entrées DNS du cluster ne doivent pas être créées. Par défaut, AWS ParallelCluster crée une zone hébergée Route 53. Si `disable_cluster_dns` ce paramètre est défini sur`true`, la zone hébergée n'est pas créée.

La valeur par défaut est `false`.

```
disable_cluster_dns = true
```

> **Warning**
>
> Un système de résolution de noms est nécessaire au bon fonctionnement du cluster. Si `disable_cluster_dns` ce paramètre est défini sur`true`, un système de résolution de noms supplémentaire doit également être fourni.

> **Important**
>
> `disable_cluster_dns`= n'`true`est pris en charge que si le `queue_settings` paramètre est spécifié.

> **Note**
>
> Support pour `disable_cluster_dns` a été ajouté dans la AWS ParallelCluster version 2.9.1.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

disable_hyperthreading

(Facultatif) Désactive l'hyperthreading sur les nœuds de tête et de calcul. Tous les types d'instance ne peuvent pas désactiver l'hyper-threading. Pour obtenir la liste des types d'instances qui prennent en charge la désactivation de l'hyperthreading, consultez la section relative aux cœurs de processeur et aux threads de chaque cœur de processeur pour chaque type d'instance dans le guide de

l'utilisateur Amazon EC2 . Si le `queue_settings` paramètre est défini, soit ce paramètre peut être défini, soit les `disable_hyperthreading` paramètres des `[queue]`sections peuvent être définis.

La valeur par défaut est `false`.

```
disable_hyperthreading = true
```

> **Note**
>
> `disable_hyperthreading`n'affecte le nœud principal que lorsque`scheduler` = `awsbatch`.

> **Note**
>
> Support pour `disable_hyperthreading` a été ajouté dans la AWS ParallelCluster version 2.5.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`ebs_settings`

(Facultatif) Identifie les `[ebs]` sections contenant les volumes Amazon EBS montés sur le nœud principal. Lorsque vous utilisez plusieurs volumes Amazon EBS, entrez ces paramètres dans une liste, chacun étant séparé par une virgule. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Jusqu'à cinq (5) volumes Amazon EBS supplémentaires sont pris en charge.

Pour plus d'informations, consultez la `[ebs]`section.

Par exemple, le paramètre suivant indique que les sections démarrent `[ebs custom1]` et `[ebs custom2]` sont utilisées pour les volumes Amazon EBS.

```
ebs_settings = custom1, custom2
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ec2_iam_role

(Facultatif) Définit le nom d'un rôle IAM existant pour Amazon EC2 qui est attaché à toutes les instances du cluster. Le nom d'un rôle IAM et son Amazon Resource Name (ARN) sont différents. ARNs ne peut pas être utilisé comme argument pour`ec2_iam_role`.

Si cette option est spécifiée, le paramètre `additional_iam_policies` est ignoré. Si votre intention est d'ajouter des politiques supplémentaires aux paramètres par défaut des nœuds de cluster, nous vous recommandons de transmettre les politiques IAM personnalisées supplémentaires avec le `additional_iam_policies`paramètre au lieu de `ec2_iam_role` les utiliser.

Si cette option n'est pas spécifiée, le rôle AWS ParallelCluster IAM par défaut pour Amazon EC2 est utilisé. Pour de plus amples informations, veuillez consulter AWS Identity and Access Management rôles dans AWS ParallelCluster.

Il n'existe aucune valeur par défaut.

```
ec2_iam_role = ParallelClusterInstanceRole
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

efs_settings

(Facultatif) Spécifie les paramètres relatifs au système de fichiers Amazon EFS. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[efs]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[efs customfs]` est utilisée pour la configuration du système de fichiers Amazon EFS.

```
efs_settings = customfs
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

enable_efa

(Facultatif) Le cas échéant, indique qu'Elastic Fabric Adapter (EFA) est activé pour les nœuds de calcul. Pour consulter la liste des EC2 instances compatibles EFA, consultez la section Types

d'instances pris en charge dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux. Pour de plus amples informations, veuillez consulter Elastic Fabric Adapter. Si le `queue_settings` paramètre est défini, soit ce paramètre peut être défini, soit les `enable_efa` paramètres de la `[queue]`section peuvent être définis. Un groupe de placement de cluster doit être utilisé pour minimiser les latences entre les instances. Pour plus d'informations, consultez `placement` et `placement_group`.

```
enable_efa = compute
```

> **Note**
>
> Support de l'EFA sur les instances Graviton2 basées sur ARM a été ajouté dans AWS ParallelCluster la version 2.10.1.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`enable_efa_gdr`

(Facultatif) À partir de AWS ParallelCluster la version 2.11.3, ce paramètre n'a aucun effet. La prise en charge GPUDirect du RDMA (Remote Direct Memory Access) par Elastic Fabric Adapter (EFA) est toujours activée si elle est prise en charge à la fois par le type d'instance et par le système d'exploitation.

> **Note**
>
> AWS ParallelCluster versions 2.10.0 à 2.11.2 : si`compute`, indique que la prise en charge du GPUDirect RDMA (accès direct à distance à la mémoire) par Elastic Fabric Adapter (EFA) est activée pour les nœuds de calcul. Pour définir ce paramètre sur, il `compute` doit être défini sur`compute`. `enable_efa` Le support EFA pour GPUDirect RDMA est pris en charge par des types d'instances spécifiques (`p4d.24xlarge`) sur des systèmes d'exploitation spécifiques (`base_os`is `alinux2` `centos7``ubuntu1804`, ou`ubuntu2004`). Si le `queue_settings` paramètre est défini, soit ce paramètre peut être défini, soit les `enable_efa_gdr` paramètres des `[queue]`sections peuvent être définis. Un groupe de placement de cluster doit être utilisé pour minimiser les latences entre les instances. Pour plus d'informations, consultez `placement` et `placement_group`.

```
enable_efa_gdr = compute
```

> **Note**
>
> Support pour `enable_efa_gdr` a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

`enable_intel_hpc_platform`

(Facultatif) Le cas échéant, indique que le contrat de licence utilisateur final pour Intel Parallel Studio est accepté. Cela entraîne l'installation d'Intel Parallel Studio sur le nœud principal et le partage avec les nœuds de calcul. Cela ajoute plusieurs minutes au temps nécessaire au nœud principal pour démarrer. Le `enable_intel_hpc_platform` paramètre n'est pris en charge que sur CentOS (7 `base_os = centos7`).

La valeur par défaut est `false`.

```
enable_intel_hpc_platform = true
```

> **Note**
>
> Le `enable_intel_hpc_platform` paramètre n'est pas compatible avec les instances AWS basées sur Graviton.

> **Note**
>
> Support pour `enable_intel_hpc_platform` a été ajouté dans la AWS ParallelCluster version 2.5.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

encrypted_ephemeral

(Facultatif) Chiffre les volumes de stockage de l'instance éphémère avec des clés en mémoire non récupérables, à l'aide de LUKS (Linux Unified Key Setup).

Pour de plus amples informations, veuillez consulter https://gitlab.com/cryptsetup/cryptsetup/blob/master/README.md.

La valeur par défaut est `false`.

```
encrypted_ephemeral = true
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ephemeral_dir

(Facultatif) Définit le chemin où les volumes de stockage d'instance sont montés s'ils sont utilisés.

La valeur par défaut est `/scratch`.

```
ephemeral_dir = /scratch
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

extra_json

(Facultatif) Définit le JSON supplémentaire qui est fusionné dans Chef `dna.json`. Pour plus d'informations, consultez Création d'une AWS ParallelCluster AMI personnalisée.

La valeur par défaut est `{}`.

```
extra_json = {}
```

> **Note**
> À partir de AWS ParallelCluster la version 2.6.1, la plupart des recettes d'installation sont ignorées par défaut lors du lancement des nœuds afin d'améliorer les temps de démarrage. Pour exécuter toutes les recettes d'installation afin d'améliorer la rétrocompatibilité au

détriment des temps de démarrage, ajoutez `"skip_install_recipes" : "no"` la `cluster` clé dans le `extra_json` paramètre. Par exemple :

```
extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

fsx_settings

(Facultatif) Spécifie la section qui définit la configuration FSx pour Lustre. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[fsx]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[fsx fs]` est utilisée pour la configuration FSx for Lustre.

```
fsx_settings = fs
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

iam_lambda_role

(Facultatif) Définit le nom d'un rôle AWS Lambda d'exécution existant. Ce rôle est associé à toutes les fonctions Lambda du cluster. Pour plus d'informations, veuillez consulter Rôle d'exécution AWS Lambda dans le Guide du développeur AWS Lambda .

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Le nom d'un rôle IAM et son Amazon Resource Name (ARN) sont différents. ARNs ne peut pas être utilisé comme argument pour`iam_lambda_role`. Si les deux `ec2_iam_role` `iam_lambda_role`

sont définis et que le « `scheduler` est `sge` » ou « ou »`torque`, aucun rôle ne sera créé. `slurm` Si tel `scheduler` est le cas`awsbatch`, des rôles seront créés pendant`pcluster start`. Par exemple, les politiques, voir `ParallelClusterLambdaPolicy`en utilisant SGESlurm, ou Torque et`ParallelClusterLambdaPolicy` utilisant `awsbatch`.

Il n'existe aucune valeur par défaut.

```
iam_lambda_role = ParallelClusterLambdaRole
```

> **Note**
>
> Support pour `iam_lambda_role` a été ajouté dans la AWS ParallelCluster version 2.10.1.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`initial_queue_size`

(Facultatif) Définit le nombre initial d' EC2 instances Amazon à lancer en tant que nœuds de calcul dans le cluster. Si le `queue_settings` paramètre est défini, il doit être supprimé et remplacé par les `initial_count` paramètres des `[compute_resource]`sections.

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Ce paramètre s'applique uniquement aux planificateurs traditionnels (SGE, Slurm, et Torque). Si le `maintain_initial_size` réglage est le `initial_queue_size` cas`true`, il doit être d'au moins un (1).

Si le planificateur est `awsbatch`, utilisez `min_vcpus` à la place.

La valeur par défaut est 2.

```
initial_queue_size = 2
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

key_name

(Facultatif) Nomme une paire de EC2 clés Amazon existante avec laquelle activer l'accès SSH aux instances.

```
key_name = mykey
```

> **Note**
>
> Avant AWS ParallelCluster la version 2.11.0, `key_name` c'était un paramètre obligatoire.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

maintain_initial_size

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

(Facultatif) Conserve la taille initiale du groupe Auto Scaling pour les planificateurs traditionnels (SGE, Slurm, et Torque).

Si le planificateur est `awsbatch`, utilisez `desired_vcpus` à la place.

Ce paramètre est un indicateur booléen. S'il est défini sur`true`, le groupe Auto Scaling ne compte jamais moins de membres que la valeur de`initial_queue_size`, et la valeur de `initial_queue_size` doit être supérieure ou égale à un (1). Le cluster peut encore être mis à l'échelle jusqu'à la valeur de `max_queue_size`. Dans `cluster_type = spot` ce cas, les instances du groupe Auto Scaling peuvent être interrompues et leur taille peut diminuer`initial_queue_size`.

S'il est défini sur`false`, le groupe Auto Scaling peut être réduit à zéro (0) membre afin d'éviter que les ressources ne restent inutilisées lorsqu'elles ne sont pas nécessaires.

Si le `queue_settings` paramètre est défini, il doit être supprimé et remplacé par les `min_count` paramètres `initial_count` et dans les `[compute_resource]`sections.

La valeur par défaut est `false`.

```
maintain_initial_size = false
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

master_instance_type

(Facultatif) Définit le type d' EC2 instance Amazon utilisé pour le nœud principal. L'architecture du type d'instance doit être identique à celle utilisée pour le `compute_instance_type` paramètre.

Dans Régions AWS le cas d'un niveau gratuit, le type d'instance par défaut est le niveau gratuit (`t2.microout3.micro`). Si vous Régions AWS n'avez pas de niveau gratuit, la valeur par défaut est. `t3.micro` Pour plus d'informations sur le niveau AWS gratuit, consultez le niveau AWS gratuit FAQs.

```
master_instance_type = t2.micro
```

> **Note**
>
> Avant AWS ParallelCluster la version 2.10.1, la valeur par défaut était « `t2.micro` in all ». Régions AWS Dans AWS ParallelCluster la version 2.10.0, le nœud `p4d.24xlarge` principal n'était pas pris en charge. Support pour les instances AWS basées sur Graviton (telles que `A1` et`C6g`) a été ajouté dans la AWS ParallelCluster version 2.8.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

master_root_volume_size

(Facultatif) Spécifie la taille du volume racine du nœud principal en gibioctets (GiB). L'AMI doit prendre en charge `growroot`.

La valeur par défaut est `35`.

> **Note**
>
> Pour AWS ParallelCluster les versions entre 2.5.0 et 2.10.4, la valeur par défaut était 25. Avant AWS ParallelCluster la version 2.5.0, la valeur par défaut était 20.

```
master_root_volume_size = 35
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

max_queue_size

(Facultatif) Définit le nombre maximum d' EC2 instances Amazon pouvant être lancées dans le cluster. Si le `queue_settings` paramètre est défini, il doit être supprimé et remplacé par les `max_count` paramètres des `[compute_resource]`sections.

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Ce paramètre s'applique uniquement aux planificateurs traditionnels (SGE, Slurm, et Torque).

Si le planificateur est `awsbatch`, utilisez `max_vcpus` à la place.

La valeur par défaut est `10`.

```
max_queue_size = 10
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour, mais le parc informatique doit être arrêté si la valeur est réduite. Dans le cas contraire, les nœuds existants risquent d'être résiliés.

max_vcpus

(Facultatif) Spécifie le nombre maximal de v CPUs dans l'environnement de calcul. Utilisé uniquement si le planificateur est `awsbatch`.

La valeur par défaut est `20`.

```
max_vcpus = 20
```

Politique de mise à jour : ce paramètre ne peut pas être réduit lors d'une mise à jour.

min_vcpus

(Facultatif) Conserve la taille initiale du groupe Auto Scaling pour le `awsbatch` planificateur.

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Si le planificateur est SGE, Slurm, ou Torque, utilisez `maintain_initial_size` plutôt.

L'environnement de calcul ne compte jamais moins de membres que la valeur de `min_vcpus`.

La valeur par défaut est `0`.

```
min_vcpus = 0
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

placement

(Facultatif) Définit la logique du groupe de placement du cluster, permettant à l'ensemble du cluster ou uniquement aux instances de calcul d'utiliser le groupe de placement du cluster.

Si le `queue_settings` paramètre est défini, il doit être supprimé et remplacé par des `placement_group` paramètres pour chacune des `[queue]`sections. Si le même groupe de placement est utilisé pour différents types d'instances, il est plus probable que la demande échoue en raison d'une erreur de capacité insuffisante. Pour plus d'informations, consultez la section Capacité d'instance insuffisante dans le guide de EC2 l'utilisateur Amazon. Plusieurs files d'attente ne peuvent partager un groupe de placement que s'il a été créé à l'avance et configuré dans les `placement_group` paramètres de chaque file d'attente. Si chaque `[queue]`section définit un `placement_group` paramètre, le nœud principal ne peut pas figurer dans le groupe de placement d'une file d'attente.

Les options valides sont `cluster` ou `compute`.

Ce paramètre n'est pas utilisé lorsque le planificateur l'est. `awsbatch`

La valeur par défaut est `compute`.

```
placement = compute
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

placement_group

(Facultatif) Définit le groupe de placement du cluster. Si le `queue_settings` paramètre est défini, il doit être supprimé et remplacé par les `placement_group` paramètres des `[queue]`sections.

Les options valides sont les valeurs suivantes :

- `DYNAMIC`
- Nom d'un groupe de placement de EC2 clusters Amazon existant

Lorsque `DYNAMIC` est défini, un groupe de placement unique est créé et supprimé dans la pile du cluster.

Ce paramètre n'est pas utilisé lorsque le planificateur l'est. `awsbatch`

Pour plus d'informations sur les groupes de placement, consultez la section Groupes de placement dans le guide de EC2 l'utilisateur Amazon. Si le même groupe de placement est utilisé pour différents types d'instances, il est plus probable que la demande échoue en raison d'une erreur de capacité insuffisante. Pour plus d'informations, consultez la section Capacité d'instance insuffisante dans le guide de EC2 l'utilisateur Amazon.

Il n'existe aucune valeur par défaut.

Tous les types d'instance ne prennent pas en charge les groupes de placement de cluster. Par exemple, le type d'instance par défaut `t3.micro` ne prend pas en charge les groupes de placement de clusters. Pour plus d'informations sur la liste des types d'instances qui prennent en charge les groupes de placement de clusters, consultez la section Règles et limites des groupes de placement de clusters dans le guide de EC2 l'utilisateur Amazon. Consultez Problèmes liés aux groupes de placement et au lancement d'instances pour obtenir des conseils pour l'utilisation des groupes de placement.

```
placement_group = DYNAMIC
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

post_install

(Facultatif) Spécifie l'URL d'un script de post-installation exécuté une fois que toutes les actions d'amorçage du nœud sont terminées. Pour de plus amples informations, veuillez consulter Actions d'amorçage personnalisées.

Lorsqu'il est utilisé `awsbatch` comme planificateur, le script de post-installation est exécuté uniquement sur le nœud principal.

Le format du paramètre peut être `http://`*`hostname/path/to/script.sh`* ou `s3://`*`bucket-name/path/to/script.sh`*.

Il n'existe aucune valeur par défaut.

```
post_install = s3://<bucket-name>/my-post-install-script.sh
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

post_install_args

(Facultatif) Spécifie une liste d'arguments entre guillemets à transmettre au script de post-installation.

Il n'existe aucune valeur par défaut.

```
post_install_args = "argument-1 argument-2"
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

pre_install

(Facultatif) Spécifie l'URL d'un script de préinstallation exécuté avant le lancement de toute action de démarrage du déploiement d'un nœud. Pour de plus amples informations, veuillez consulter Actions d'amorçage personnalisées.

Lorsqu'il est utilisé `awsbatch` comme planificateur, le script de préinstallation est exécuté uniquement sur le nœud principal.

Le format du paramètre peut être `http://hostname/path/to/script.sh` ou `s3://bucket-name/path/to/script.sh`.

Il n'existe aucune valeur par défaut.

```
pre_install = s3://bucket-name/my-pre-install-script.sh
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

pre_install_args

(Facultatif) Spécifie une liste d'arguments entre guillemets à transmettre au script de préinstallation.

Il n'existe aucune valeur par défaut.

```
pre_install_args = "argument-3 argument-4"
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

proxy_server

(Facultatif) Définit généralement un serveur proxy HTTP ou HTTPS`http://x.x.x.x:8080`.

Il n'existe aucune valeur par défaut.

```
proxy_server = http://10.11.12.13:8080
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

queue_settings

(Facultatif) Spécifie que le cluster utilise des files d'attente au lieu d'un parc informatique homogène, et indique quelles `[queue]`sections sont utilisées. La première `[queue]`section répertoriée est la file d'attente du planificateur par défaut. Les noms de `queue` section doivent commencer par une lettre minuscule, ne pas contenir plus de 30 caractères et ne contenir que des lettres minuscules, des chiffres et des tirets (-).

> **Important**
>
> `queue_settings`n'est pris en charge que lorsqu'`scheduler`il est défini sur`slurm`. Les `spot_price` paramètres `cluster_typecompute_instance_type`,`initial_queue_size`,`maintain_initial_size`,`max` `placementplacement_group`, et ne doivent pas être spécifiés. Les `enable_efa` paramètres `disable_hyperthreading` et peuvent être spécifiés dans la `[cluster]`section ou dans les `[queue]`sections, mais pas dans les deux.

Jusqu'à cinq (5) `[queue]`sections sont prises en charge.

Pour plus d'informations, consultez la `[queue]`section.

Par exemple, le paramètre suivant indique que les sections qui démarrent `[queue q1]` et `[queue q2]` sont utilisées.

```
queue_settings = q1, q2
```

> **Note**
>
> Support pour `queue_settings` a été ajouté dans la AWS ParallelCluster version 2.9.0.

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

`raid_settings`

(Facultatif) Identifie la `[raid]` section avec la configuration RAID du volume Amazon EBS. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[raid]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[raid rs]` doit être utilisée pour la configuration Auto Scaling.

```
raid_settings = rs
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

s3_read_resource

(Facultatif) Spécifie une ressource Amazon S3 à laquelle les AWS ParallelCluster nœuds bénéficient d'un accès en lecture seule.

`arn:aws:s3:::`*`my_corporate_bucket`*`*`Fournit, par exemple, un accès en lecture seule au *`my_corporate_bucket`* compartiment et aux objets qu'il contient.

Consultez la section Travailler avec Amazon S3 pour plus de détails sur le format.

Il n'existe aucune valeur par défaut.

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

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

s3_read_write_resource

(Facultatif) Spécifie une ressource Amazon S3 à laquelle les AWS ParallelCluster nœuds ont un accès en lecture/écriture.

`arn:aws:s3:::`*`my_corporate_bucket`*`/Development/*`Fournit, par exemple, un accès en lecture/écriture à tous les objets du `Development` dossier du *`my_corporate_bucket`* compartiment.

Consultez la section Travailler avec Amazon S3 pour plus de détails sur le format.

Il n'existe aucune valeur par défaut.

```
s3_read_write_resource = arn:aws:s3:::my_corporate_bucket/*
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

scaling_settings

Identifie la `[scaling]` section avec la configuration Auto Scaling. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[scaling]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[scaling custom]` est utilisée pour la configuration Auto Scaling.

```
scaling_settings = custom
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`scheduler`

(Obligatoire)Définit le planificateur du cluster.

Les options valides sont les valeurs suivantes :

`awsbatch`

AWS Batch

Pour plus d'informations sur le `awsbatch` planificateur, consultez les sections Configuration réseau et. AWS Batch (`awsbatch`)

`sge`

> **Note**
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Son of Grid Engine (SGE)

`slurm`

Slurm Workload Manager (Slurm)

`torque`

> **Note**
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Torque Resource Manager (Torque)

> **Note**
>
> Avant AWS ParallelCluster la version 2.7.0, le `scheduler` paramètre était facultatif et le paramètre par défaut étaitsge. À partir de AWS ParallelCluster la version 2.7.0, le `scheduler` paramètre est obligatoire.

```
scheduler = slurm
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

shared_dir

(Facultatif) Définit le chemin où le volume Amazon EBS partagé est monté.

N'utilisez pas cette option avec plusieurs volumes Amazon EBS. Indiquez plutôt des `shared_dir` valeurs sous chaque `[ebs]`section.

Consultez la `[ebs]`section pour en savoir plus sur l'utilisation de plusieurs volumes Amazon EBS.

La valeur par défaut est `/shared`.

L'exemple suivant montre un volume Amazon EBS partagé monté sur`/myshared`.

```
shared_dir = myshared
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

spot_bid_percentage

(Facultatif) Définit le pourcentage à la demande utilisé pour calculer le ComputeFleet prix spot maximum pour `awsbatch` le planificateur.

S'il n'est pas spécifié, le prix du marché spot actuel est sélectionné, plafonné au prix à la demande.

```
spot_bid_percentage = 85
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

spot_price

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

(Facultatif) Définit le prix spot maximum pour ComputeFleet les planificateurs traditionnels (SGE, Slurm, et Torque). Utilisé uniquement lorsque le `cluster_type` paramètre est défini sur`spot`. Si vous ne spécifiez aucune valeur, le prix au comptant vous est facturé, plafonné au prix à la demande. Si le `queue_settings` paramètre est défini, il doit être supprimé et remplacé par les `spot_price` paramètres des `[compute_resource]`sections.

Si le planificateur est `awsbatch`, utilisez spot_bid_percentage à la place.

Pour obtenir de l'aide pour trouver une instance Spot qui répond à vos besoins, consultez le conseiller en matière d'instances Spot.

```
spot_price = 1.50
```

> **Note**
>
> Dans AWS ParallelCluster la version 2.5.0, si l'`cluster_type = spot`erreur `spot_price` n'est pas spécifiée, l'instance démarre en cas d' ComputeFleet échec. Cela a été corrigé dans la AWS ParallelCluster version 2.5.1.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

tags

(Facultatif) Définit les balises à utiliser par AWS CloudFormation.

Si les balises de la ligne de commande sont spécifiées via --tags, elles sont fusionnées aux balises de configuration.

Les balises de ligne de commande remplacent les balises de configuration qui ont la même clé.

Les balises sont au format JSON. N'utilisez pas de guillemets en dehors des bretelles.

Pour plus d'informations, consultez la section Type de balises de AWS CloudFormation ressource dans le Guide de AWS CloudFormation l'utilisateur.

```
tags = {"key" : "value", "key2" : "value2"}
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

> **Note**
>
> La politique de mise à jour ne permettait pas de modifier les `tags` paramètres de la AWS ParallelCluster version 2.8.0 à la version 2.9.1.
> Pour les versions 2.10.0 à 2.11.7, la politique de mise à jour répertoriée qui prenait en charge la modification du `tags` paramètre n'est pas exacte. La mise à jour du cluster lors de la modification de ce paramètre n'est pas prise en charge.

template_url

(Facultatif) Définit le chemin d'accès au AWS CloudFormation modèle utilisé pour créer le cluster.

Met à jour l'utilisation du modèle initialement utilisé pour créer la pile.

La valeur par défaut est `https://`*`aws_region_name`*`-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-`*`version`*`.cfn.json`.

> **Warning**
>
> Il s'agit d'un paramètre avancé. Toute modification de ce paramètre est effectuée à vos risques et périls.

```
template_url = https://us-east-1-aws-parallelcluster.s3.amazonaws.com/templates/aws-parallelcluster-2.11.9.cfn.json
```

Politique de mise à jour : ce paramètre n'est pas analysé lors d'une mise à jour.

vpc_settings

(Obligatoire) Identifie la `[vpc]` section contenant la configuration Amazon VPC dans laquelle le cluster est déployé. Le nom de section doit commencer par une lettre, ne pas contenir plus de 30 caractères et uniquement contenir des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Pour plus d'informations, consultez la `[vpc]`section.

Par exemple, le paramètre suivant indique que la section qui commence `[vpc public]` est utilisée pour la configuration Amazon VPC.

```
vpc_settings = public
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`[compute_resource]` Section

Définit les paramètres de configuration d'une ressource de calcul. `[compute_resource]`les sections sont référencées par le `compute_resource_settings` paramètre de la `[queue]`section. `[compute_resource]`les sections ne sont prises en charge que lorsqu'elles `scheduler` sont définies sur`slurm`.

Le format est`[compute_resource <compute-resource-name>]`. *`compute-resource-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

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

> **Note**
>
> Support pour `[compute_resource]`cette section a été ajouté dans la AWS ParallelCluster version 2.9.0.

Rubriques

- initial_count
- instance_type
- max_count
- min_count
- spot_price

initial_count

(Facultatif) Définit le nombre initial d' EC2 instances Amazon à lancer pour cette ressource de calcul. La création du cluster n'est pas terminée tant qu'au moins autant de nœuds n'ont pas été lancés dans la ressource de calcul. Si le `compute_type` paramètre de la file d'attente est défini `spot` et qu'il n'y a pas suffisamment d'instances Spot disponibles, la création du cluster peut expirer et échouer. Tout nombre supérieur au `min_count` paramètre correspond à la capacité dynamique soumise au `scaledown_idletime` réglage. Ce paramètre remplace le paramètre `initial_queue_size`.

La valeur par défaut est `0`.

```
initial_count = 2
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

instance_type

(Obligatoire) Définit le type d' EC2 instance Amazon utilisé pour cette ressource de calcul. L'architecture du type d'instance doit être identique à celle utilisée pour le `master_instance_type` paramètre. Le `instance_type` paramètre doit être unique pour chaque `[compute_resource]`section référencée par une `[queue]`section. Ce paramètre remplace le paramètre `compute_instance_type`.

```
instance_type = t2.micro
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

max_count

(Facultatif) Définit le nombre maximum d' EC2 instances Amazon pouvant être lancées dans cette ressource de calcul. Tout compte supérieur au `initial_count` réglage est lancé en mode hors tension. Ce paramètre remplace le paramètre `max_queue_size`.

La valeur par défaut est `10`.

```
max_count = 10
```

Politique de mise à jour : pour réduire la taille d'une file d'attente en dessous du nombre actuel de nœuds, il faut d'abord arrêter le parc informatique.

> **Note**
>
> La politique de mise à jour ne permettait pas de modifier le `max_count` paramètre tant que le parc informatique n'était pas arrêté pour les AWS ParallelCluster versions 2.0.0 à 2.9.1.

min_count

(Facultatif) Définit le nombre minimum d' EC2 instances Amazon pouvant être lancées dans cette ressource de calcul. Ces nœuds ont tous une capacité statique. La création du cluster n'est pas terminée tant qu'au moins ce nombre de nœuds n'a été lancé dans la ressource de calcul.

La valeur par défaut est `0`.

```
min_count = 1
```

Politique de mise à jour : pour réduire le nombre de nœuds statiques dans une file d'attente, il faut d'abord arrêter le parc informatique.

> **Note**
>
> La politique de mise à jour ne permettait pas de modifier le `min_count` paramètre tant que le parc informatique n'était pas arrêté pour les AWS ParallelCluster versions 2.0.0 à 2.9.1.

spot_price

(Facultatif) Définit le prix spot maximal pour cette ressource de calcul. Utilisé uniquement lorsque le `compute_type` paramètre de la file d'attente contenant ces ressources de calcul est défini sur`spot`. Ce paramètre remplace le paramètre `spot_price`.

Si vous ne spécifiez aucune valeur, le prix au comptant vous est facturé, plafonné au prix à la demande.

Pour obtenir de l'aide pour trouver une instance Spot qui répond à vos besoins, consultez le conseiller en matière d'instances Spot.

```
spot_price = 1.50
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

[cw_log] Section

Définit les paramètres de configuration pour CloudWatch les journaux.

Le format est`[cw_log cw-log-name]`. *`cw-log-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

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

Pour plus d'informations, consultez Intégration à Amazon CloudWatch Logs, Tableau de CloudWatch bord Amazon et Intégration à Amazon CloudWatch Logs.

> **Note**
>
> Support pour `cw_log` a été ajouté dans la AWS ParallelCluster version 2.6.0.

enable

(Facultatif) Indique si CloudWatch les journaux sont activés.

La valeur par défaut est `true`. `false`À utiliser pour désactiver CloudWatch les journaux.

L'exemple suivant active CloudWatch les journaux.

```
enable = true
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`retention_days`

(Facultatif) Indique le nombre de jours pendant CloudWatch lesquels Logs conserve les événements individuels du journal.

La valeur par défaut est `14`. Les valeurs prises en charge sont 1, 3, 5, 7, 14, 30, 60, 90, 120, 150, 180, 365, 400, 545, 731, 1827 et 3653.

L'exemple suivant configure les CloudWatch journaux pour conserver les événements des journaux pendant 30 jours.

```
retention_days = 30
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`[dashboard]` Section

Définit les paramètres de configuration du CloudWatch tableau de bord.

Le format est`[dashboard dashboard-name]`. *`dashboard-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

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

> **Note**
>
> Support pour `dashboard` a été ajouté dans la AWS ParallelCluster version 2.10.0.

enable

(Facultatif) Indique si le CloudWatch tableau de bord est activé.

La valeur par défaut est `true`. Utilisez-le `false` pour désactiver le CloudWatch tableau de bord.

L'exemple suivant active le CloudWatch tableau de bord.

```
enable = true
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

[dcv] Section

Définit les paramètres de configuration pour le serveur Amazon DCV exécuté sur le nœud principal.

Pour créer et configurer un serveur Amazon DCV, spécifiez le cluster `dcv_settings` avec le nom que vous avez défini dans la `dcv` section, et définissez `enable` sur`master`, et `base_os` sur `alinux2centos7`, `ubuntu1804` ou`ubuntu2004`. Si le nœud principal est une instance ARM, définissez `base_os` sur `alinux2centos7`, ou`ubuntu1804`.

Le format est`[dcv dcv-name]`. *`dcv-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

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

Pour de plus amples informations, consultez Connectez-vous au nœud principal via Amazon DCV.

> **Important**
>
> Par défaut, le port Amazon DCV configuré par AWS ParallelCluster est ouvert à toutes les IPv4 adresses. Toutefois, vous ne pouvez vous connecter à un port Amazon DCV que si vous disposez de l'URL de la session Amazon DCV et si vous vous connectez à la session Amazon DCV dans les 30 secondes suivant le renvoi de l'URL. `pcluster dcv connect` Utilisez le `access_from` paramètre pour restreindre davantage l'accès au port Amazon DCV

avec une plage d'adresses IP au format CIDR, et utilisez-le pour `port` définir un port non standard.

> **Note**
>
> Support pour la `[dcv]section` sur `centos8` a été supprimé dans la AWS ParallelCluster version 2.10.4. Support pour la `[dcv]section` sur `centos8` a été ajouté dans la AWS ParallelCluster version 2.10.0. Support pour la `[dcv]section` sur les instances AWS basées sur Graviton a été ajouté dans la AWS ParallelCluster version 2.9.0. Support pour la `[dcv]section` sur `alinux2` et `ubuntu1804` a été ajouté dans la AWS ParallelCluster version 2.6.0. Support pour la `[dcv]section` sur `centos7` a été ajouté dans la AWS ParallelCluster version 2.5.0.

`access_from`

(Facultatif, recommandé) Spécifie la plage d'adresses IP au format CIDR pour les connexions à Amazon DCV. Ce paramètre est utilisé uniquement lors de la AWS ParallelCluster création du groupe de sécurité.

La valeur par défaut est `0.0.0.0/0`, ce qui permet l'accès à partir de n'importe quelle adresse Internet.

```
access_from = 0.0.0.0/0
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`enable`

(Obligatoire) Indique si Amazon DCV est activé sur le nœud principal. Pour activer Amazon DCV sur le nœud principal et configurer la règle de groupe de sécurité requise, définissez le `enable` paramètre sur. `master`

L'exemple suivant active Amazon DCV sur le nœud principal.

```
enable = master
```

> **Note**
>
> Amazon DCV génère automatiquement un certificat auto-signé qui est utilisé pour sécuriser le trafic entre le client Amazon DCV et le serveur Amazon DCV exécuté sur le nœud principal. Pour configurer votre propre certificat, veuillez consulter Certificat HTTPS Amazon DCV.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`port`

(Facultatif) Spécifie le port pour Amazon DCV.

La valeur par défaut est `8443`.

```
port = 8443
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`[ebs]` Section

Définit les paramètres de configuration des volumes Amazon EBS pour les volumes montés sur le nœud principal et partagés avec les nœuds de calcul via NFS.

Pour savoir comment inclure les volumes Amazon EBS dans la définition de votre cluster, consultez`[cluster] Section`/`ebs_settings`.

Pour utiliser un volume Amazon EBS existant pour un stockage permanent à long terme indépendant du cycle de vie du cluster, spécifiez`ebs_volume_id`.

Si vous ne le spécifiez pas`ebs_volume_id`, AWS ParallelCluster crée le volume EBS à partir des `[ebs]` paramètres lors de la création du cluster et supprime le volume et les données lorsque le cluster est supprimé.

Pour de plus amples informations, veuillez consulter Bonnes pratiques : déplacement d'un cluster vers une nouvelle version AWS ParallelCluster mineure ou un correctif.

Le format est`[ebs ebs-name]`. *`ebs-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

```
[ebs custom1]
shared_dir = vol1
ebs_snapshot_id = snap-xxxxx
volume_type = io1
volume_iops = 200
...

[ebs custom2]
shared_dir = vol2
...

...
```

Rubriques

- shared_dir
- ebs_kms_key_id
- ebs_snapshot_id
- ebs_volume_id
- encrypted
- volume_iops
- volume_size
- volume_throughput
- volume_type

shared_dir

(Obligatoire) Spécifie le chemin où le volume Amazon EBS partagé est monté.

Ce paramètre est obligatoire lors de l'utilisation de plusieurs volumes Amazon EBS.

Lorsque vous utilisez un volume Amazon EBS, cette option remplace le `shared_dir` volume spécifié dans la `[cluster]` section. Dans l'exemple suivant, le volume est monté sur `/vol1`.

```
shared_dir = vol1
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ebs_kms_key_id

(Facultatif) Spécifie une AWS KMS clé personnalisée à utiliser pour le chiffrement.

Ce paramètre doit être utilisé en même temps que `encrypted = true`. Il doit aussi avoir un `ec2_iam_role` personnalisé.

Pour de plus amples informations, veuillez consulter Chiffrement du disque avec une clé KMS personnalisée.

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ebs_snapshot_id

(Facultatif) Définit l'ID d'instantané Amazon EBS si vous utilisez un instantané comme source pour le volume.

Il n'existe aucune valeur par défaut.

```
ebs_snapshot_id = snap-xxxxx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ebs_volume_id

(Facultatif) Définit l'ID de volume d'un volume Amazon EBS existant à associer au nœud principal.

Il n'existe aucune valeur par défaut.

```
ebs_volume_id = vol-xxxxxx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

encrypted

(Facultatif) Spécifie si le volume Amazon EBS est chiffré. Remarque : ne pas utiliser avec les instantanés.

La valeur par défaut est `false`.

```
encrypted = false
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

volume_iops

(Facultatif) Définit le nombre d'IOPS pour io1io2, et le gp3 type de volumes.

La valeur par défaut, les valeurs prises en charge et volume_iops le volume_size ratio de production varient selon volume_type etvolume_size.

volume_type = io1

Valeur par défaut volume_iops = 100

Valeurs prises en charge volume_iops = 100—64 000 †

volume_sizeRapport volume_iops maximum = 50 IOPS pour chaque GiB. 5000 IOPS nécessitent au volume_size moins 100 GiB.

volume_type = io2

Valeur par défaut volume_iops = 100

Valeurs prises en charge volume_iops = 100 à 64 000 (256 000 pour les volumes io2 Block Express) †

volume_sizeRapport volume_iops maximum = 500 IOPS pour chaque GiB. 5000 IOPS nécessitent au volume_size moins 10 GiB.

volume_type = gp3

Valeur par défaut volume_iops = 3000

Valeurs prises en charge volume_iops = 3000 à 16 000

volume_sizeRapport volume_iops maximum = 500 IOPS pour chaque GiB. 5000 IOPS nécessitent au volume_size moins 10 GiB.

```
volume_iops = 200
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

† Le maximum d'IOPS est garanti uniquement sur les instances basées sur le système Nitro et approvisionnées avec plus de 32 000 IOPS. Les autres instances garantissent jusqu'à 32 000 IOPS. À moins que vous ne modifiiez le volume, `io1` les volumes antérieurs risquent de ne pas atteindre leurs performances optimales. `io2` Les volumes Block Express prennent en charge `volume_iops` des valeurs allant jusqu'à 256 000. Pour plus d'informations, consultez les volumes `io2` Block Express (en version préliminaire) dans le guide de EC2 l'utilisateur Amazon.

`volume_size`

(Facultatif) Spécifie la taille du volume à créer, en GiB (si vous n'utilisez pas de capture instantanée).

La valeur par défaut et les valeurs prises en charge varient de`volume_type`.

`volume_type` = `standard`

Par défaut `volume_size` = 20 GiB

Valeurs prises en charge `volume_size` = 1 à 1024 GiB

`volume_type`= `gp2io1`,`io2`, et `gp3`

Par défaut `volume_size` = 20 GiB

Valeurs prises en charge `volume_size` = 1 à 16384 GiB

`volume_type`= `sc1` et `st1`

Par défaut `volume_size` = 500 GiB

Valeurs prises en charge `volume_size` = 500—16384 GiB

```
volume_size = 20
```

> **Note**
>
> Avant AWS ParallelCluster la version 2.10.1, la valeur par défaut pour tous les types de volumes était de 20 GiB.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

volume_throughput

(Facultatif) Définit le débit pour les types de gp3 volumes, en Mbits/s.

La valeur par défaut est 125.

Valeurs prises en charge volume_throughput = 125-1000 MiB/s

Le rapport de volume_throughput to ne volume_iops peut pas être supérieur à 0,25. Le débit maximal de 1 000 Mbits/s nécessite que le volume_iops paramètre soit d'au moins 4 000.

```
volume_throughput = 1000
```

> **Note**
>
> Support pour volume_throughput a été ajouté dans la AWS ParallelCluster version 2.10.1.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

volume_type

(Facultatif) Spécifie le type de volume Amazon EBS que vous souhaitez lancer.

Les options valides sont les types de volumes suivants :

gp2, gp3

SSD à usage général

io1, io2

Provisioned IOPS SSD

st1

Disque dur à débit optimisé

sc1

HDD à froid

standard

Magnétique de génération précédente

Pour plus d'informations, consultez les types de volumes Amazon EBS dans le guide de l' EC2 utilisateur Amazon.

La valeur par défaut est `gp2`.

```
volume_type = io2
```

> **Note**
>
> Support pour `gp3` et `io2` a été ajouté dans la AWS ParallelCluster version 2.10.1.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`[efs]` Section

Définit les paramètres de configuration pour l'Amazon EFS monté sur les nœuds de tête et de calcul. Pour plus d'informations, consultez CreateFileSystemle manuel Amazon EFS API Reference.

Pour savoir comment inclure les systèmes de fichiers Amazon EFS dans la définition de votre cluster, consultez`[cluster] Section`/`efs_settings`.

Pour utiliser un système de fichiers Amazon EFS existant pour un stockage permanent à long terme indépendant du cycle de vie du cluster, spécifiez`efs_fs_id`.

Si vous ne le spécifiez pas`efs_fs_id`, AWS ParallelCluster crée le système de fichiers Amazon EFS à partir des `[efs]` paramètres lors de la création du cluster et supprime le système de fichiers et les données lorsque le cluster est supprimé.

Pour de plus amples informations, veuillez consulter Bonnes pratiques : déplacement d'un cluster vers une nouvelle version AWS ParallelCluster mineure ou un correctif.

Le format est`[efs efs-name]`. *`efs-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

```
[efs customfs]
shared_dir = efs
```

```
encrypted = false
performance_mode = generalPurpose
```

Rubriques

- efs_fs_id
- efs_kms_key_id
- encrypted
- performance_mode
- provisioned_throughput
- shared_dir
- throughput_mode

efs_fs_id

(Facultatif) Définit l'ID du système de fichiers Amazon EFS pour un système de fichiers existant.

La spécification de cette option annule toutes les autres options Amazon EFS, à l'exception de`shared_dir`.

Si vous définissez cette option, elle ne prend en charge que les types de systèmes de fichiers suivants :

- Systèmes de fichiers qui n'ont pas de cible de montage dans la zone de disponibilité de la pile.
- Systèmes de fichiers dotés d'une cible de montage existante dans la zone de disponibilité de la pile et dont le trafic NFS entrant et sortant est autorisé à partir de. `0.0.0.0/0`

La vérification d'intégrité pour valider `efs_fs_id` exige que le rôle IAM dispose des autorisations suivantes :

- `elasticfilesystem:DescribeMountTargets`
- `elasticfilesystem:DescribeMountTargetSecurityGroups`
- `ec2:DescribeSubnets`
- `ec2:DescribeSecurityGroups`
- `ec2:DescribeNetworkInterfaceAttribute`

Veuillez ajouter ces autorisations à votre rôle IAM ou définissez `sanity_check = false` pour éviter les erreurs.

> **Important**
>
> Lorsque vous définissez une cible de montage dont le trafic NFS entrant et sortant est autorisé`0.0.0.0/0`, le système de fichiers est exposé aux demandes de montage NFS provenant de n'importe quel endroit de la zone de disponibilité de la cible de montage. AWS ne recommande pas de créer une cible de montage dans la zone de disponibilité de la pile. Laissons plutôt cette étape s' AWS occuper de cette étape. Si vous souhaitez avoir une cible de montage dans la zone de disponibilité de la pile, pensez à utiliser un groupe de sécurité personnalisé en fournissant une `vpc_security_group_id` option dans la `[vpc]`section. Ajoutez ensuite ce groupe de sécurité à la cible de montage et désactivez-le `sanity_check` pour créer le cluster.

Il n'existe aucune valeur par défaut.

```
efs_fs_id = fs-12345
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`efs_kms_key_id`

(Facultatif) Identifie la clé gérée par le client AWS Key Management Service (AWS KMS) à utiliser pour protéger le système de fichiers chiffré. Si ce paramètre est défini, le paramètre `encrypted` doit être défini sur `true`. Cela correspond au KmsKeyIdparamètre indiqué dans le manuel Amazon EFS API Reference.

Il n'existe aucune valeur par défaut.

```
efs_kms_key_id = 1234abcd-12ab-34cd-56ef-1234567890ab
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`encrypted`

(Facultatif) Indique si le système de fichiers est crypté. Cela correspond au paramètre Encrypted dans la référence d'API Amazon EFS.

La valeur par défaut est `false`.

```
encrypted = true
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`performance_mode`

(Facultatif) Définit le mode de performance du système de fichiers. Cela correspond au PerformanceModeparamètre indiqué dans le manuel Amazon EFS API Reference.

Les options valides sont les valeurs suivantes :

- `generalPurpose`
- `maxIO`

Les deux valeurs distinguent les majuscules et minuscules.

Nous recommandons le mode de performance `generalPurpose` pour la plupart des systèmes de fichiers.

Les systèmes de fichiers qui utilisent le mode de performance `maxIO` peuvent évoluer vers des niveaux plus élevés de débit cumulé et d'opérations par seconde. Cependant, il existe un compromis entre des latences légèrement plus élevées pour la plupart des opérations sur les fichiers.

Une fois le système de fichiers créé, ce paramètre ne peut pas être modifié.

La valeur par défaut est `generalPurpose`.

```
performance_mode = generalPurpose
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`provisioned_throughput`

(Facultatif) Définit le débit provisionné du système de fichiers, mesuré en Mbits/s. Cela correspond au ProvisionedThroughputInMibpsparamètre indiqué dans le manuel Amazon EFS API Reference.

Si vous utilisez ce paramètre, vous devez définir `throughput_mode` sur `provisioned`.

Le quota de débit est de `1024` Mbits/s. Pour demander une augmentation de quota, contactez Support.

La valeur minimale est `0.0` MiO/s.

```
provisioned_throughput = 1024
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

shared_dir

(Obligatoire) Définit le point de montage Amazon EFS sur la tête et les nœuds de calcul.

Ce paramètre est obligatoire. La section Amazon EFS n'est utilisée que si elle `shared_dir` est spécifiée.

N'utilisez pas `NONE` ou `/NONE` comme répertoire partagé.

L'exemple suivant montre comment monter Amazon EFS à`/efs`.

```
shared_dir = efs
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

throughput_mode

(Facultatif) Définit le mode de débit du système de fichiers. Cela correspond au ThroughputModeparamètre indiqué dans le manuel Amazon EFS API Reference.

Les options valides sont les valeurs suivantes :

- `bursting`
- `provisioned`

La valeur par défaut est `bursting`.

```
throughput_mode = provisioned
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`[fsx]` Section

Définit les paramètres de configuration d'un système FSx de fichiers attaché à Lustre. Pour plus d'informations, consultez Amazon FSx CreateFileSystem dans le Amazon FSx API Reference.

Si le `base_os` est `alinux2`, ou `centos7` `ubuntu1804` `ubuntu2004`, FSx pour Lustre est pris en charge.

Lorsque vous utilisez Amazon Linux, le noyau doit être une version `4.14.104-78.84.amzn1.x86_64` ou une version ultérieure. Pour obtenir des instructions, consultez la section Installation du client Lustre dans le guide de l'utilisateur d'Amazon FSx pour Lustre.

> **Note**
>
> FSx for Lustre n'est actuellement pas pris en charge lors de son utilisation en `awsbatch` tant que planificateur.

> **Note**
>
> Support FSx pour Lustre activé `centos8` a été supprimé dans la AWS ParallelCluster version 2.10.4. Support FSx pour Lustre activé `ubuntu2004` a été ajouté dans la AWS ParallelCluster version 2.11.0. Support FSx pour Lustre activé `centos8` a été ajouté dans la AWS ParallelCluster version 2.10.0. Support FSx pour Lustre activé `suralinux2`,`ubuntu1604`, et `ubuntu1804` a été ajouté dans la AWS ParallelCluster version 2.6.0. Support FSx pour Lustre activé `centos7` a été ajouté dans la AWS ParallelCluster version 2.4.0.

Si vous utilisez un système de fichiers existant, celui-ci doit être associé à un groupe de sécurité qui autorise le trafic TCP entrant vers le port 988. La définition de la source `0.0.0.0/0` sur une règle de groupe de sécurité fournit un accès client à partir de toutes les plages d'adresses IP de votre groupe de sécurité VPC pour le protocole et la plage de ports pour cette règle. Pour limiter davantage l'accès à vos systèmes de fichiers, nous vous recommandons d'utiliser des sources plus restrictives pour les règles de votre groupe de sécurité. Par exemple, vous pouvez utiliser des plages d'adresses CIDR, des adresses IP ou des groupes IDs de sécurité plus spécifiques. Cette opération est effectuée automatiquement lorsque vous n'utilisez pas `vpc_security_group_id`.

Pour utiliser un système de FSx fichiers Amazon existant pour un stockage permanent à long terme indépendant du cycle de vie du cluster, spécifiez`fsx_fs_id`.

Si vous ne le spécifiez pas`fsx_fs_id`, AWS ParallelCluster crée le système de fichiers FSx for Lustre à partir des `[fsx]` paramètres lors de la création du cluster et supprime le système de fichiers et les données lorsque le cluster est supprimé.

Pour de plus amples informations, veuillez consulter Bonnes pratiques : déplacement d'un cluster vers une nouvelle version AWS ParallelCluster mineure ou un correctif.

Le format est`[fsx fsx-name]`. *`fsx-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

```
[fsx fs]
shared_dir = /fsx
fsx_fs_id = fs-073c3803dca3e28a6
```

Pour créer et configurer un nouveau système de fichiers, utilisez les paramètres suivants :

```
[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
```

Rubriques

- 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

(Facultatif) Spécifie la politique d'importation automatique pour refléter les modifications apportées au compartiment S3 utilisé pour créer le système de fichiers FSx for Lustre. Les valeurs possibles sont les suivantes :

`NEW`

FSx for Lustre importe automatiquement les listes de répertoires de tous les nouveaux objets ajoutés au compartiment S3 lié qui n'existent pas actuellement dans le système de fichiers FSx for Lustre.

`NEW_CHANGED`

FSx for Lustre importe automatiquement les listes de fichiers et de répertoires de tous les nouveaux objets ajoutés au compartiment S3 et de tous les objets existants modifiés dans le compartiment S3.

Cela correspond à la AutoImportPolicypropriété. Pour plus d'informations, consultez Importer automatiquement les mises à jour depuis votre compartiment S3 dans le guide de l'utilisateur d'Amazon FSx for Lustre. Lorsque le `auto_import_policy` paramètre est spécifié, les `fsx_backup_id` paramètres `automatic_backup_retention_days`,`copy_tags_to_backups`,`daily_automatic_backup_start_` et ne doivent pas être spécifiés.

Si le `auto_import_policy` paramètre n'est pas spécifié, les importations automatiques sont désactivées. FSx for Lustre met uniquement à jour les listes de fichiers et de répertoires à partir du compartiment S3 lié lorsque le système de fichiers est créé.

```
auto_import_policy = NEW_CHANGED
```

> **Note**
>
> Support pour `auto_import_policy` a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`automatic_backup_retention_days`

(Facultatif) Spécifie le nombre de jours pendant lesquels les sauvegardes automatiques sont conservées. Ceci n'est valable que pour les types de `PERSISTENT_1` déploiement. Lorsque le `automatic_backup_retention_days` paramètre est spécifié, les `imported_file_chunk_size` paramètres `auto_import_policyexport_path`,`import_path`, et ne doivent pas être spécifiés. Cela correspond à la AutomaticBackupRetentionDayspropriété.

La valeur par défaut est 0. Ce paramètre désactive les sauvegardes automatiques. Les valeurs possibles sont des entiers compris entre 0 et 35 inclus.

```
automatic_backup_retention_days = 35
```

> **Note**
>
> Support pour `automatic_backup_retention_days` a été ajouté dans la AWS ParallelCluster version 2.8.0.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`copy_tags_to_backups`

(Facultatif) Spécifie si les balises du système de fichiers sont copiées dans les sauvegardes. Ceci n'est valable que pour les types de `PERSISTENT_1` déploiement. Lorsque le

copy_tags_to_backups paramètre est spécifié, automatic_backup_retention_days il doit être spécifié avec une valeur supérieure à 0, et les imported_file_chunk_size paramètres auto_import_policyexport_path,import_path, et ne doivent pas être spécifiés. Cela correspond à la CopyTagsToBackupspropriété.

La valeur par défaut est `false`.

```
copy_tags_to_backups = true
```

> **Note**
>
> Support pour copy_tags_to_backups a été ajouté dans la AWS ParallelCluster version 2.8.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

daily_automatic_backup_start_time

(Facultatif) Spécifie l'heure (UTC) à laquelle démarrer les sauvegardes automatiques. Ceci n'est valable que pour les types de `PERSISTENT_1` déploiement. Lorsque le daily_automatic_backup_start_time paramètre est spécifié, automatic_backup_retention_days il doit être spécifié avec une valeur supérieure à 0, et les imported_file_chunk_size paramètres auto_import_policyexport_path,import_path, et ne doivent pas être spécifiés. Cela correspond à la DailyAutomaticBackupStartTimepropriété.

Le format est le `HH:MM` suivant : où `HH` est l'heure du jour remplie de zéro (0-23) et où `MM` est la minute de l'heure remplie de zéro. Par exemple, 1 h 03 UTC est le suivant.

```
daily_automatic_backup_start_time = 01:03
```

La valeur par défaut est une durée aléatoire comprise entre `00:00` et`23:59`.

> **Note**
>
> Support pour daily_automatic_backup_start_time a été ajouté dans la AWS ParallelCluster version 2.8.0.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

data_compression_type

(Facultatif) Spécifie le type de compression de données FSx pour Lustre. Cela correspond à la DataCompressionTypepropriété. Pour plus d'informations, consultez la section relative FSx à la compression des données Lustre dans le guide de l'utilisateur d'Amazon FSx for Lustre.

La seule valeur valide est `LZ4`. Pour désactiver la compression des données, supprimez le `data_compression_type`paramètre.

```
data_compression_type = LZ4
```

> **Note**
>
> Support pour `data_compression_type` a été ajouté dans la AWS ParallelCluster version 2.11.0.

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

deployment_type

(Facultatif) Spécifie le type de déploiement FSx pour Lustre. Cela correspond à la DeploymentTypepropriété. Pour plus d'informations, consultez FSx les options de déploiement de Lustre dans le guide de l'utilisateur d'Amazon FSx pour Lustre. Choisissez un type de déploiement temporaire pour le stockage temporaire et le traitement des données à court terme. `SCRATCH_2`est la dernière génération de systèmes de fichiers scratch. Il offre un débit en rafale supérieur au débit de base et le chiffrement des données en transit.

Les valeurs valides sont `SCRATCH_1`, `SCRATCH_2` et `PERSISTENT_1`.

`SCRATCH_1`

Type de déploiement par défaut FSx pour Lustre. Avec ce type de déploiement, le paramètre `storage_capacity` peut avoir pour valeur 1 200, 2 400 et n'importe quel multiple de 3 600. Support pour `SCRATCH_1` a été ajouté dans la AWS ParallelCluster version 2.4.0.

`SCRATCH_2`

La dernière génération de systèmes de fichiers Scratch. Il prend en charge un débit jusqu'à six fois supérieur à celui de base pour les charges de travail exigeantes. Il prend également en charge le chiffrement des données en transit pour les types d'instances pris en charge Régions AWS. Pour plus d'informations, consultez la section Chiffrement des données en transit dans le guide de l'utilisateur d'Amazon FSx for Lustre. Avec ce type de déploiement, le paramètre `storage_capacity` peut avoir pour valeur 1 200 et n'importe quel multiple de 2 400. Support pour `SCRATCH_2` a été ajouté dans la AWS ParallelCluster version 2.6.0.

`PERSISTENT_1`

Conçu pour un stockage à plus long terme. Les serveurs de fichiers sont hautement disponibles et les données sont répliquées dans la zone de AWS disponibilité des systèmes de fichiers. Il prend en charge le chiffrement des données en transit pour les types d'instances pris en charge. Avec ce type de déploiement, le paramètre `storage_capacity` peut avoir pour valeur 1 200 et n'importe quel multiple de 2 400. Support pour `PERSISTENT_1` a été ajouté dans la AWS ParallelCluster version 2.6.0.

La valeur par défaut est `SCRATCH_1`.

```
deployment_type = SCRATCH_2
```

> **Note**
>
> Support pour `deployment_type` a été ajouté dans la AWS ParallelCluster version 2.6.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`drive_cache_type`

(Facultatif) Spécifie que le système de fichiers possède un cache de disque SSD. Cela ne peut être défini que si le `storage_type` paramètre est défini sur`HDD`. Cela correspond à la DriveCacheTypepropriété. Pour plus d'informations, consultez FSx les options de déploiement de Lustre dans le guide de l'utilisateur d'Amazon FSx pour Lustre.

La seule valeur valide est `READ`. Pour désactiver le cache du disque SSD, ne spécifiez pas le `drive_cache_type` paramètre.

```
drive_cache_type = READ
```

> **Note**
>
> Support pour `drive_cache_type` a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`export_path`

(Facultatif) Spécifie le chemin Amazon S3 où la racine de votre système de fichiers est exportée. Lorsque le `export_path` paramètre est spécifié, les `fsx_backup_id` paramètres `automatic_backup_retention_dayscopy_tags_to_backups`,`daily_automatic_backup_start_` et ne doivent pas être spécifiés. Cela correspond à la ExportPathpropriété. Les données et métadonnées des fichiers ne sont pas automatiquement exportées vers le`export_path`. Pour plus d'informations sur l'exportation de données et de métadonnées, consultez la section Exportation des modifications apportées au référentiel de données dans le guide de l'utilisateur d'Amazon FSx for Lustre.

La valeur par défaut est `s3://`*`import-bucket`*`/FSxLustre`*`[creation-timestamp]`*, où *`import-bucket`* représente le compartiment fourni dans le paramètre `import_path`.

```
export_path = s3://bucket/folder
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`fsx_backup_id`

(Facultatif) Spécifie l'ID de la sauvegarde à utiliser pour restaurer le système de fichiers à partir d'une sauvegarde existante. Lorsque le `fsx_backup_id` paramètre est spécifié, les `per_unit_storage_throughput` paramètres `auto_import_policydeployment_type`,`export_path`,`fsx_kms_key_id`,`import_path`,`imported_` et ne doivent pas être spécifiés. Ces paramètres sont lus à partir de la sauvegarde. En outre`auto_import_policy`, les `imported_file_chunk_size` paramètres `export_pathimport_path`,, et ne doivent pas être spécifiés.

Cela correspond à la BackupIdpropriété.

```
fsx_backup_id = backup-fedcba98
```

> **Note**
>
> Support pour `fsx_backup_id` a été ajouté dans la AWS ParallelCluster version 2.8.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

fsx_fs_id

(Facultatif) Attache un système de fichiers existant FSx pour Lustre.

Si cette option est spécifiée, seuls les `fsx_fs_id` paramètres `shared_dir` et de la `[fsx]`section sont utilisés et tous les autres paramètres de la `[fsx]`section sont ignorés.

```
fsx_fs_id = fs-073c3803dca3e28a6
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

fsx_kms_key_id

(Facultatif) Spécifie l'ID de clé de votre AWS Key Management Service (AWS KMS) clé gérée par le client.

Cette clé est utilisée pour chiffrer les données dans votre système de fichiers au repos.

Il doit être utilisé avec un `ec2_iam_role` personnalisé. Pour de plus amples informations, veuillez consulter Chiffrement du disque avec une clé KMS personnalisée. Cela correspond au KmsKeyIdparamètre indiqué dans le Amazon FSx API Reference.

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

> **Note**
>
> Support pour `fsx_kms_key_id` a été ajouté dans la AWS ParallelCluster version 2.6.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

import_path

(Facultatif) Spécifie le compartiment S3 à partir duquel les données seront chargées dans le système de fichiers et qui servira de compartiment d'exportation. Pour de plus amples informations, veuillez consulter `export_path`. Si vous spécifiez le `import_path` paramètre`automatic_backup_retention_days`, les `fsx_backup_id` paramètres`copy_tags_to_backups`,`daily_automatic_backup_start_time`, et ne doivent pas être spécifiés. Cela correspond au ImportPathparamètre indiqué dans le Amazon FSx API Reference.

L'importation se produit à la création du cluster. Pour plus d'informations, consultez la section Importation de données depuis votre référentiel de données dans le guide de l'utilisateur d'Amazon FSx for Lustre. Lors de l'importation, seules les métadonnées du fichier (nom, propriétaire, horodatage et autorisations) sont importées. Les données du fichier ne sont importées depuis le compartiment S3 qu'après le premier accès au fichier. Pour plus d'informations sur le préchargement du contenu des fichiers, consultez la section Préchargement de fichiers dans votre système de fichiers dans le guide de l'utilisateur d'Amazon FSx for Lustre.

Si aucune valeur n'est fournie, le système de fichiers est vide.

```
import_path =  s3://bucket
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

imported_file_chunk_size

(Facultatif) Détermine le nombre de bandes et la quantité maximale de données pour chaque fichier (en MiB) stocké sur un seul disque physique pour les fichiers importés depuis un référentiel de données (en utilisant`import_path`). Le nombre maximal de disques sur lesquels un fichier unique peut être agrégé par bandes est limité au nombre total de disques qui composent le système de fichiers. Lorsque le `imported_file_chunk_size` paramètre est spécifié, les `fsx_backup_id` paramètres `automatic_backup_retention_dayscopy_tags_to_backups`,`daily_automatic_backup_start_` et ne doivent pas être spécifiés. Cela correspond à la ImportedFileChunkSizepropriété.

La taille de bloc par défaut est `1024` (1 GiB), et elle peut atteindre 512 000 MiB (500 GiB). Les objets Amazon S3 ont une taille maximale de 5 To.

```
imported_file_chunk_size = 1024
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

per_unit_storage_throughput

(Obligatoire pour les types de déploiement **PERSISTENT_1**) Pour le type de déploiement `deployment_type = PERSISTENT_1`, décrit le débit en lecture et en écriture pour chaque téraoctet (To) de stockage, en Mo/s/To. La capacité de débit du système de fichiers est calculée en multipliant la capacité de stockage du système de fichiers (TiB) par le `per_unit_storage_throughput` (pour `per_unit_storage_throughput` obtenir 120 Mo/s de débit MB/s/TiB). For a 2.4 TiB file system, provisioning 50 MB/s/TiB du système de fichiers). Vous payez le débit que vous allouez. Cela correspond à la PerUnitStorageThroughputpropriété.

Les valeurs possibles dépendent de la valeur du `storage_type` paramètre.

`storage_type = SSD`

Les valeurs possibles sont 50, 100 et 200.

`storage_type = HDD`

Les valeurs possibles sont 12, 40.

```
per_unit_storage_throughput = 200
```

> **Note**
>
> Support pour `per_unit_storage_throughput` a été ajouté dans la AWS ParallelCluster version 2.6.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

shared_dir

(Obligatoire) Définit le point de montage du système de fichiers FSx for Lustre sur les nœuds de tête et de calcul.

N'utilisez pas `NONE` ou `/NONE` comme répertoire partagé.

L'exemple suivant monte le système de fichiers à l'emplacement `/fsx`.

```
shared_dir = /fsx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

storage_capacity

(Obligatoire) Spécifie la capacité de stockage du système de fichiers en Gio. Cela correspond à la StorageCapacitypropriété.

Les valeurs de capacité de stockage possibles varient en fonction du paramètre `deployment_type`.

`SCRATCH_1`

Les valeurs possibles sont 1 200, 2 400 et n'importe quel multiple de 3 600.

`SCRATCH_2`

Les valeurs possibles sont 1 200 et n'importe quel multiple de 2 400.

`PERSISTENT_1`

Les valeurs possibles varient en fonction des valeurs des autres paramètres.

`storage_type = SSD`

Les valeurs possibles sont 1 200 et n'importe quel multiple de 2 400.

`storage_type = HDD`

Les valeurs possibles varient en fonction du `per_unit_storage_throughput` réglage.

`per_unit_storage_throughput = 12`

Les valeurs possibles sont un multiple de 6 000.

`per_unit_storage_throughput = 40`

Les valeurs possibles sont un multiple de 1800.

```
storage_capacity = 7200
```

> **Note**
>
> Pour les AWS ParallelCluster versions 2.5.0 et 2.5.1, les valeurs possibles de 1200, 2400 et tout multiple de 3600 étaient prises en `storage_capacity` charge. Pour les versions antérieures à AWS ParallelCluster la version 2.5.0, la taille minimale `storage_capacity` était de 3600.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`storage_type`

(Facultatif) Spécifie le type de stockage du système de fichiers. Cela correspond à la StorageTypepropriété. Les valeurs possibles sont `SSD` et `HDD`. La valeur par défaut est `SSD`.

Le type de stockage modifie les valeurs possibles des autres paramètres.

`storage_type = SSD`

Spécifie un type de stockage SSD (Solid State Drive).

`storage_type = SSD`modifie les valeurs possibles de plusieurs autres paramètres.

`drive_cache_type`

Ce paramètre ne peut pas être spécifié.

`deployment_type`

Ce paramètre peut être défini sur `SCRATCH_1``SCRATCH_2`, ou`PERSISTENT_1`.

`per_unit_storage_throughput`

Ce paramètre doit être spécifié si `deployment_type` est défini sur `PERSISTENT_1`. Les valeurs possibles sont 50, 100 ou 200.

`storage_capacity`

Ce paramètre doit être spécifié. Les valeurs possibles varient en fonction de`deployment_type`.

`deployment_type = SCRATCH_1`

`storage_capacity`peut être 1200, 2400 ou n'importe quel multiple de 3600.

`deployment_type = SCRATCH_2` ou `deployment_type = PERSISTENT_1`

`storage_capacity`peut être 1200 ou n'importe quel multiple de 2400.

`storage_type = HDD`

Spécifie un type de stockage sur disque dur (HDD).

`storage_type = HDD`modifie les valeurs possibles des autres paramètres.

`drive_cache_type`

Ce paramètre peut être spécifié.

`deployment_type`

Ce paramètre doit être défini sur`PERSISTENT_1`.

`per_unit_storage_throughput`

Ce paramètre doit être spécifié. Les valeurs possibles sont 12 ou 40.

`storage_capacity`

Ce paramètre doit être spécifié. Les valeurs possibles varient en fonction du `per_unit_storage_throughput` réglage.

`storage_capacity = 12`

`storage_capacity`peut être un multiple de 6000.

`storage_capacity = 40`

`storage_capacity`peut être un multiple de 1800.

```
storage_type = SSD
```

> **Note**
>
> Support pour ce `storage_type`paramètre a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`weekly_maintenance_start_time`

(Facultatif) Spécifie l'heure privilégiée d'exécution de la maintenance hebdomadaire, dans le fuseau horaire de l'heure universelle coordonnée (UTC). Cela correspond à la WeeklyMaintenanceStartTimepropriété.

Le format est [jour de la semaine]:[heure du jour]:[minute de l'heure]. Par exemple, le lundi à minuit est le suivant.

```
weekly_maintenance_start_time = 1:00:00
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

`[queue]` Section

Définit les paramètres de configuration pour une file d'attente unique. `[queue]`les sections ne sont prises en charge que lorsqu'elles `scheduler` sont définies sur`slurm`.

Le format est`[queue <queue-name>]`. *`queue-name`*doit commencer par une lettre minuscule, ne pas contenir plus de 30 caractères et contenir uniquement des lettres minuscules, des chiffres et des tirets (-).

```
[queue q1]
compute_resource_settings = i1,i2
placement_group = DYNAMIC
enable_efa = true
disable_hyperthreading = false
compute_type = spot
```

> **Note**
>
> Support pour `[queue]`cette section a été ajouté dans la AWS ParallelCluster version 2.9.0.

Rubriques

- compute_resource_settings
- compute_type
- disable_hyperthreading

- enable_efa
- enable_efa_gdr
- placement_group

compute_resource_settings

(Obligatoire) Identifie les `[compute_resource]`sections contenant les configurations des ressources de calcul pour cette file d'attente. Les noms des sections doivent commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

Jusqu'à trois (3) `[compute_resource]`sections sont prises en charge pour chaque `[queue]`section.

Par exemple, le paramètre suivant indique que les sections qui démarrent `[compute_resource cr1]` et `[compute_resource cr2]` sont utilisées.

```
compute_resource_settings = cr1, cr2
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

compute_type

(Facultatif) Définit le type d'instances à lancer pour cette file d'attente. Ce paramètre remplace le paramètre `cluster_type`.

Les options valides sont `ondemand` et `spot`.

La valeur par défaut est `ondemand`.

Pour plus d'informations sur les instances Spot, consultez Utilisation de instances Spot.

> **Note**
>
> L'utilisation d'instances Spot nécessite que le rôle `AWSServiceRoleForEC2Spot` lié au service existe dans votre compte. Pour créer ce rôle dans votre compte à l'aide de AWS CLI, exécutez la commande suivante :
>
> ```
> aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```

Pour plus d'informations, consultez la section Rôle lié au service pour les demandes d'instance Spot dans le guide de EC2 l'utilisateur Amazon.

L'exemple suivant utilise SpotInstances pour les nœuds de calcul de cette file d'attente.

```
compute_type = spot
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

disable_hyperthreading

(Facultatif) Désactive l'hyperthreading sur les nœuds de cette file d'attente. Tous les types d'instance ne peuvent pas désactiver l'hyper-threading. Pour obtenir la liste des types d'instances qui prennent en charge la désactivation de l'hyperthreading, consultez la section Cœurs et threads de processeur pour chaque cœur de processeur par type d'instance dans le guide de l'utilisateur Amazon EC2 . Si le `disable_hyperthreading` paramètre de la `[cluster]`section est défini, il ne peut pas être défini.

La valeur par défaut est `false`.

```
disable_hyperthreading = true
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

enable_efa

(Facultatif) Si ce paramètre est défini sur`true`, indique qu'Elastic Fabric Adapter (EFA) est activé pour les nœuds de cette file d'attente. Pour consulter la liste des EC2 instances compatibles EFA, consultez la section Types d'instances pris en charge dans le Guide de EC2 l'utilisateur Amazon pour les instances Linux. Si le `enable_efa` paramètre de la `[cluster]`section est défini, il ne peut pas être défini. Un groupe de placement de cluster doit être utilisé pour minimiser les latences entre les instances. Pour plus d’informations, consultez `placement` et `placement_group`.

```
enable_efa = true
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

enable_efa_gdr

(Facultatif) À partir de AWS ParallelCluster la version 2.11.3, ce paramètre n'a aucun effet. Le support d'Elastic Fabric Adapter (EFA) GPUDirect pour le RDMA (accès direct à distance à la mémoire) est activé pour les nœuds de calcul. Il est toujours activé s'il est pris en charge par le type d'instance.

> **Note**
>
> AWS ParallelCluster versions 2.10.0 à 2.11.2 : si`true`, indique qu'Elastic Fabric Adapter (EFA) GPUDirect RDMA (accès direct à la mémoire à distance) est activé pour les nœuds de cette file d'attente. La définition de cette `true` valeur nécessite que le `enable_efa` paramètre soit défini sur `true` .EFA GPUDirect RDMA est pris en charge par les types d'instances suivants (`p4d.24xlarge`) sur ces systèmes d'exploitation (`alinux2`, `centos7ubuntu1804`, ou`ubuntu2004`). Si le `enable_efa_gdr` paramètre de la `[cluster]`section est défini, il ne peut pas être défini. Un groupe de placement de cluster doit être utilisé pour minimiser les latences entre les instances. Pour plus d'informations, consultez `placement` et `placement_group`.

La valeur par défaut est `false`.

```
enable_efa_gdr = true
```

> **Note**
>
> Support pour `enable_efa_gdr` a été ajouté dans la AWS ParallelCluster version 2.10.0.

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

placement_group

(Facultatif) Le cas échéant, définit le groupe de placement pour cette file d'attente. Ce paramètre remplace le paramètre `placement_group`.

Les options valides sont les valeurs suivantes :

- `DYNAMIC`
- Nom d'un groupe de placement de EC2 clusters Amazon existant

Lorsque ce paramètre est défini sur`DYNAMIC`, un groupe de placement unique pour cette file d'attente est créé et supprimé dans le cadre de la pile de clusters.

Pour plus d'informations sur les groupes de placement, consultez la section Groupes de placement dans le guide de EC2 l'utilisateur Amazon. Si le même groupe de placement est utilisé pour différents types d'instances, il est plus probable que la demande échoue en raison d'une erreur de capacité insuffisante. Pour plus d'informations, consultez la section Capacité d'instance insuffisante dans le guide de EC2 l'utilisateur Amazon.

Il n'existe aucune valeur par défaut.

Tous les types d'instance ne prennent pas en charge les groupes de placement de cluster. Par exemple, `t2.micro` ne prend pas en charge les groupes de placement en cluster. Pour plus d'informations sur la liste des types d'instances qui prennent en charge les groupes de placement de clusters, consultez la section Règles et limites des groupes de placement de clusters dans le guide de EC2 l'utilisateur Amazon. Consultez Problèmes liés aux groupes de placement et au lancement d'instances pour obtenir des conseils pour l'utilisation des groupes de placement.

```
placement_group = DYNAMIC
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

`[raid]` Section

Définit les paramètres de configuration d'une matrice RAID construite à partir d'un certain nombre de volumes Amazon EBS identiques. Le disque RAID est monté sur le nœud principal et est exporté vers des nœuds de calcul avec NFS.

Le format est`[raid raid-name]`. *`raid-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

```
[raid rs]
shared_dir = raid
raid_type = 1
num_of_raid_volumes = 2
encrypted = true
```

Rubriques

- shared_dir
- ebs_kms_key_id
- encrypted
- num_of_raid_volumes
- raid_type
- volume_iops
- volume_size
- volume_throughput
- volume_type

shared_dir

(Obligatoire) Définit le point de montage de la matrice RAID sur la tête et les nœuds de calcul.

Le lecteur RAID est créé uniquement si ce paramètre est spécifié.

N'utilisez pas `NONE` ou `/NONE` comme répertoire partagé.

L'exemple suivant monte le tableau à `/raid`.

```
shared_dir = raid
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ebs_kms_key_id

(Facultatif) Spécifie une AWS KMS clé personnalisée à utiliser pour le chiffrement.

Ce paramètre doit être utilisé en même temps que `encrypted = true`, et doit avoir un `ec2_iam_role` personnalisé.

Pour de plus amples informations, veuillez consulter Chiffrement du disque avec une clé KMS personnalisée.

```
ebs_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

encrypted

(Facultatif) Spécifie si le système de fichiers est chiffré.

La valeur par défaut est `false`.

```
encrypted = false
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

num_of_raid_volumes

(Facultatif) Définit le nombre de volumes Amazon EBS à partir desquels assembler la matrice RAID.

Le nombre minimum de volumes est de2.

Le nombre maximum de volumes est de5.

La valeur par défaut est 2.

```
num_of_raid_volumes = 2
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

raid_type

(Obligatoire) Définit le type RAID de la matrice RAID.

Le lecteur RAID est créé uniquement si ce paramètre est spécifié.

Les options valides sont les valeurs suivantes :

- `0`

- 1

Pour plus d'informations sur les types de RAID, consultez les informations RAID dans le guide de EC2 l'utilisateur Amazon.

L'exemple suivant crée un tableau RAID `0` :

```
raid_type = 0
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

volume_iops

(Facultatif) Définit le nombre d'IOPS pour `io1io2`, et le `gp3` type de volumes.

La valeur par défaut, les valeurs prises en charge et `volume_iops` le `volume_size` ratio de production varient selon `volume_type` et`volume_size`.

`volume_type` = `io1`

Valeur par défaut `volume_iops` = 100

Valeurs prises en charge `volume_iops` = 100—64 000 †

`volume_size`Rapport `volume_iops` maximum = 50 IOPS par GiB. 5000 IOPS nécessitent au `volume_size` moins 100 GiB.

`volume_type` = `io2`

Valeur par défaut `volume_iops` = 100

Valeurs prises en charge `volume_iops` = 100 à 64 000 (256 000 pour les volumes `io2` Block Express) †

`volume_size`Rapport `volume_iops` maximum = 500 IOPS par GiB. 5000 IOPS nécessitent au `volume_size` moins 10 GiB.

`volume_type` = `gp3`

Valeur par défaut `volume_iops` = 3000

Valeurs prises en charge `volume_iops` = 3000 à 16 000

`volume_size`Rapport `volume_iops` maximum = 500 IOPS par GiB. 5000 IOPS nécessitent au `volume_size` moins 10 GiB.

```
volume_iops = 3000
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

† Le maximum d'IOPS est garanti uniquement sur les instances basées sur le système Nitro et approvisionnées avec plus de 32 000 IOPS. Les autres instances garantissent jusqu'à 32 000 IOPS. Les anciens `io1` volumes risquent de ne pas atteindre leurs performances optimales à moins que vous ne modifiiez le volume. `io2` Les volumes Block Express prennent en charge `volume_iops` des valeurs allant jusqu'à 256 000. Pour plus d'informations, consultez les volumes `io2` Block Express (en version préliminaire) dans le guide de EC2 l'utilisateur Amazon.

`volume_size`

(Facultatif) Définit la taille du volume à créer, en GiB.

La valeur par défaut et les valeurs prises en charge varient de`volume_type`.

`volume_type` = `standard`

Par défaut `volume_size` = 20 GiB

Valeurs prises en charge `volume_size` = 1 à 1024 GiB

`volume_type`= `gp2io1`,`io2`, et `gp3`

Par défaut `volume_size` = 20 GiB

Valeurs prises en charge `volume_size` = 1 à 16384 GiB

`volume_type`= `sc1` et `st1`

Par défaut `volume_size` = 500 GiB

Valeurs prises en charge `volume_size` = 500—16384 GiB

```
volume_size = 20
```

> **Note**
>
> Avant AWS ParallelCluster la version 2.10.1, la valeur par défaut pour tous les types de volumes était de 20 GiB.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`volume_throughput`

(Facultatif) Définit le débit pour les types de `gp3` volumes, en Mbits/s.

La valeur par défaut est `125`.

Valeurs prises en charge `volume_throughput` = 125-1000 MiB/s

Le rapport de `volume_throughput` à ne `volume_iops` peut pas être supérieur à 0,25. Le débit maximal de 1 000 Mbits/s nécessite que le `volume_iops` paramètre soit d'au moins 4 000.

```
volume_throughput = 1000
```

> **Note**
>
> Support pour `volume_throughput` a été ajouté dans la AWS ParallelCluster version 2.10.1.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`volume_type`

(Facultatif) Définit le type de volume à créer.

Les options valides sont les valeurs suivantes :

`gp2`, `gp3`

SSD à usage général

`io1`, `io2`

Provisioned IOPS SSD

`st1`

Disque dur à débit optimisé

`sc1`

HDD à froid

`standard`

Magnétique de génération précédente

Pour plus d'informations, consultez les types de volumes Amazon EBS dans le guide de l' EC2 utilisateur Amazon.

La valeur par défaut est `gp2`.

```
volume_type = io2
```

> **Note**
>
> Support pour `gp3` et `io2` a été ajouté dans la AWS ParallelCluster version 2.10.1.

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

`[scaling]` Section

Rubriques

- scaledown_idletime

Spécifie les paramètres qui définissent la façon dont les nœuds de calcul sont mis à l'échelle.

Le format est`[scaling scaling-name]`. *`scaling-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

```
[scaling custom]
```

```
scaledown_idletime = 10
```

scaledown_idletime

(Facultatif) Spécifie la durée en minutes sans tâche, après laquelle le nœud de calcul se termine.

Ce paramètre n'est pas utilisé s'il s'`awsbatch`agit du planificateur.

La valeur par défaut est `10`.

```
scaledown_idletime = 10
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

[vpc] Section

Spécifie les paramètres de configuration d'Amazon VPC. Pour plus d'informations VPCs, consultez Qu'est-ce qu'Amazon VPC ? et les meilleures pratiques de sécurité pour votre VPC dans le guide de l'utilisateur Amazon VPC.

Le format est`[vpc vpc-name]`. *`vpc-name`*doit commencer par une lettre, ne pas contenir plus de 30 caractères et contenir uniquement des lettres, des chiffres, des traits d'union (-) et des traits de soulignement (_).

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

Rubriques

- additional_sg
- compute_subnet_cidr
- compute_subnet_id
- master_subnet_id
- ssh_from
- use_public_ips

- vpc_id
- vpc_security_group_id

additional_sg

(Facultatif) Fournit un identifiant de groupe de sécurité Amazon VPC supplémentaire pour toutes les instances.

Il n'existe aucune valeur par défaut.

```
additional_sg = sg-xxxxxx
```

compute_subnet_cidr

(Facultatif) Spécifie un bloc CIDR (Classless Inter-Domain Routing). Utilisez ce paramètre si vous souhaitez AWS ParallelCluster créer un sous-réseau de calcul.

```
compute_subnet_cidr = 10.0.100.0/24
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

compute_subnet_id

(Facultatif) Spécifie l'ID d'un sous-réseau existant dans lequel approvisionner les nœuds de calcul.

S'il n'est pas spécifié, `compute_subnet_id` utilise la valeur de `master_subnet_id`.

Si le sous-réseau est privé, vous devez configurer NAT pour l'accès à Internet.

```
compute_subnet_id = subnet-xxxxxx
```

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

master_subnet_id

(Obligatoire) Spécifie l'ID d'un sous-réseau existant dans lequel le nœud principal doit être provisionné.

```
master_subnet_id = subnet-xxxxxx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

ssh_from

(Facultatif) Spécifie une plage d'adresses IP au format CIDR à partir de laquelle autoriser l'accès SSH.

Ce paramètre est utilisé uniquement lors de la AWS ParallelCluster création du groupe de sécurité.

La valeur par défaut est `0.0.0.0/0`.

```
ssh_from = 0.0.0.0/0
```

Politique de mise à jour : ce paramètre peut être modifié lors d'une mise à jour.

use_public_ips

(Facultatif) Définit s'il faut attribuer des adresses IP publiques aux instances de calcul.

Si ce paramètre est défini sur`true`, une adresse IP élastique est associée au nœud principal.

S'il est défini sur`false`, le nœud principal possède une adresse IP publique (ou non) conformément à la valeur du paramètre de configuration du sous-réseau « Attribuer automatiquement une adresse IP publique ».

Pour obtenir des exemples, consultez Configuration de mise en réseau.

La valeur par défaut est `true`.

```
use_public_ips = true
```

> **Important**
>
> Par défaut, Comptes AWS elles sont toutes limitées à cinq (5) adresses IP Elastic pour chacune Région AWS. Pour plus d'informations, consultez la section Limite d'adresses IP élastique dans le guide de EC2 l'utilisateur Amazon.

Politique de mise à jour : le parc informatique doit être arrêté pour que ce paramètre soit modifié pour une mise à jour.

vpc_id

(Obligatoire) Spécifie l'ID du VPC Amazon dans lequel le cluster doit être provisionné.

```
vpc_id = vpc-xxxxxx
```

Politique de mise à jour : si ce paramètre est modifié, la mise à jour n'est pas autorisée.

vpc_security_group_id

(Facultatif) Spécifie l'utilisation d'un groupe de sécurité existant pour toutes les instances.

Il n'existe aucune valeur par défaut.

```
vpc_security_group_id = sg-xxxxxx
```

Le groupe de sécurité créé par AWS ParallelCluster autorise l'accès SSH via le port 22 à partir des adresses spécifiées dans le `ssh_from` paramètre, ou de toutes les IPv4 adresses (`0.0.0.0/0`) si le `ssh_from` paramètre n'est pas spécifié. Si Amazon DCV est activé, le groupe de sécurité autorise l'accès à Amazon DCV en utilisant le port 8443 (ou tout autre `port` paramètre spécifié) à partir des adresses spécifiées dans le `access_from` paramètre, ou de toutes les IPv4 adresses (`0.0.0.0/0`) si le `access_from` paramètre n'est pas spécifié.

> **Warning**
>
> Vous pouvez modifier la valeur de ce paramètre et mettre à jour le cluster s'il `[cluster]fsx_settings`n'est pas spécifié, ou les deux `fsx_settings` et un système de fichiers externe existant FSx pour Lustre est spécifié `fsx-fs-id`dans `[fsx fs]`.
> Vous ne pouvez pas modifier la valeur de ce paramètre si un système de fichiers AWS ParallelCluster géré FSx pour Lustre est spécifié dans `fsx_settings` et`[fsx fs]`.

Politique de mise à jour : si les systèmes de fichiers Amazon FSx for Lustre AWS ParallelCluster gérés ne sont pas spécifiés dans la configuration, ce paramètre peut être modifié lors d'une mise à jour.

Exemples

Les exemples de configuration suivants illustrent AWS ParallelCluster les configurations utilisant Slurm, Torque, et des AWS Batch planificateurs.

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Table des matières

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

Slurm Workload Manager (`slurm`)

L'exemple suivant lance un cluster avec le planificateur `slurm`. L'exemple de configuration lance 1 cluster avec 2 files d'attente de tâches. La première file d'`spot`attente dispose initialement de 2 instances `t3.micro` Spot disponibles. Il peut être redimensionné jusqu'à 10 instances au maximum et réduit à un minimum d'une instance lorsqu'aucune tâche n'a été exécutée pendant 10 minutes (réglable à l'aide du `scaledown_idletime` paramètre). La deuxième file d'attente démarre sans aucune instance et peut être étendue jusqu'à un maximum de 5 instances `t3.micro` à la demande. `ondemand`

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

[aws]
aws_region_name = <your Région 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`) et Torque Resource Manager (`torque`)

> **Note**
>
> Cet exemple s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

L'exemple suivant lance un cluster avec le `torque` `sge` planificateur or. Pour utiliser SGE, remplacez `scheduler = torque` par`scheduler = sge`. L'exemple de configuration autorise un maximum de 5 nœuds simultanés et est réduit à deux lorsqu'aucune tâche n'a été exécutée pendant 10 minutes.

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

```
cluster_template = torque

[aws]
aws_region_name = <your Région 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**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs. Si vous utilisez ces versions, vous pouvez continuer à les utiliser ou à obtenir l'assistance des équipes de AWS service et de AWS support pour la résolution des problèmes.

AWS Batch (`awsbatch`)

L'exemple suivant lance un cluster avec le planificateur `awsbatch`. Il est configuré pour sélectionner le meilleur type d'instance en fonction des besoins en ressources de votre travail.

L'exemple de configuration autorise un maximum de 40 v CPUs simultanés et est réduit à zéro lorsqu'aucune tâche n'a été exécutée pendant 10 minutes (réglable à l'aide du `scaledown_idletime` paramètre).

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

```
[aws]
aws_region_name = <your Région AWS>

[vpc public]
master_subnet_id = <your subnet>
vpc_id = <your VPC>

[cluster awsbatch]
scheduler = awsbatch
compute_instance_type = optimal # optional, defaults to optimal
min_vcpus = 0                   # optional, defaults to 0
desired_vcpus = 0               # optional, defaults to 4
max_vcpus = 40                  # optional, defaults to 20
base_os = alinux2               # optional, defaults to alinux2, controls the base_os
 of
                                # the head node and the docker image for the compute
 fleet
key_name = <your EC2 keypair name>
vpc_settings = public
```

Comment AWS ParallelCluster fonctionne

AWS ParallelCluster a été conçu non seulement comme un moyen de gérer les clusters, mais aussi comme une référence sur la manière d'utiliser les AWS services pour créer votre environnement HPC.

Rubriques

- AWS ParallelCluster processus
- AWS services utilisés par AWS ParallelCluster
- AWS ParallelCluster Auto Scaling

AWS ParallelCluster processus

Cette section s'applique uniquement aux clusters HPC déployés avec l'un des planificateurs de tâches traditionnels pris en charge (SGE, Slurm, ou Torque). Lorsqu'il est utilisé avec ces planificateurs, il AWS ParallelCluster gère le provisionnement et la suppression des nœuds de calcul en interagissant à la fois avec le groupe Auto Scaling et le planificateur de tâches sous-jacent.

Pour les clusters HPC basés sur AWS Batch, AWS ParallelCluster s'appuie sur les fonctionnalités fournies par le AWS Batch pour la gestion des nœuds de calcul.

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs. Vous pouvez continuer à les utiliser dans les versions antérieures à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.

Rubriques

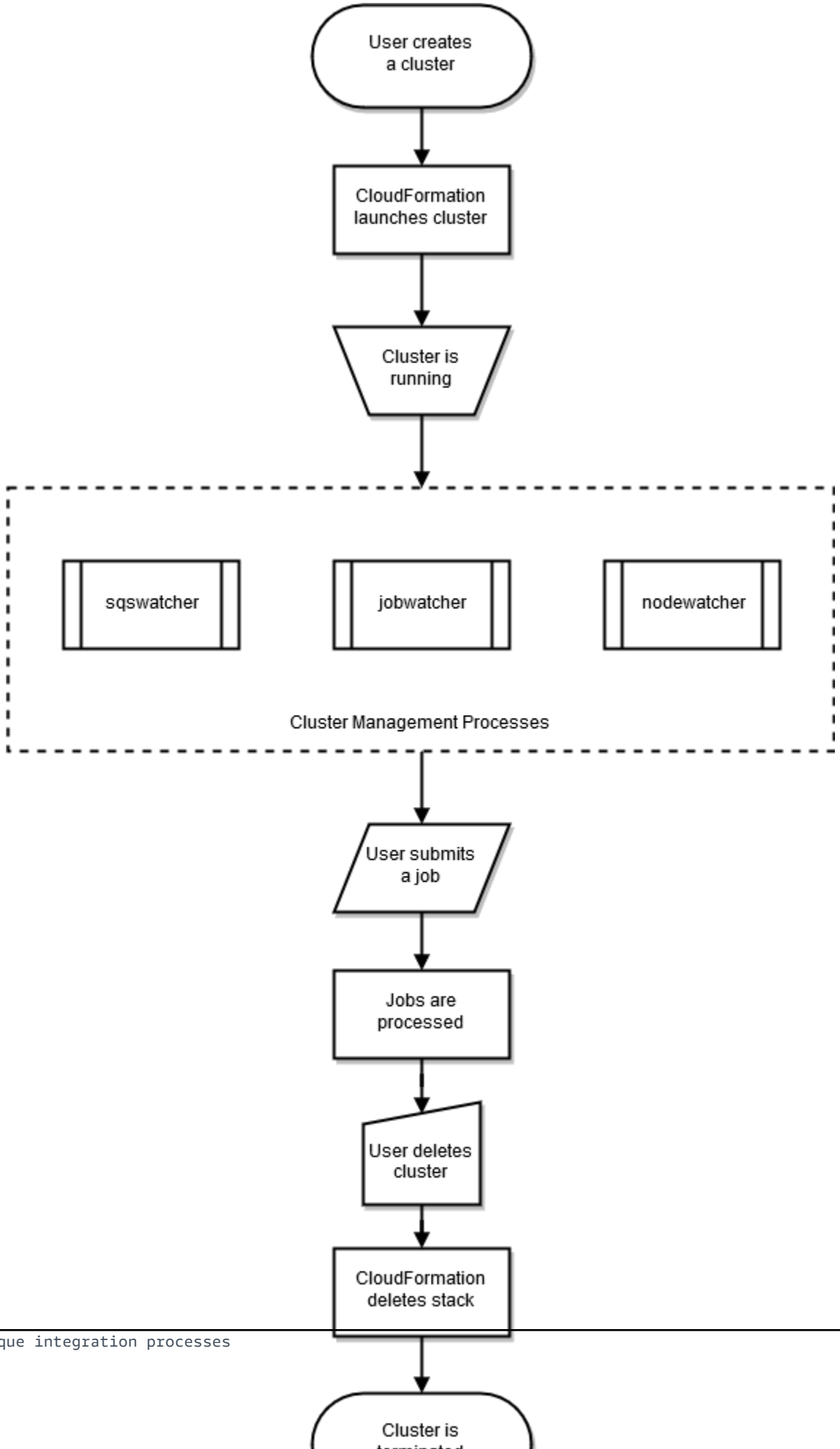
- SGE and Torque integration processes
- Slurm integration processes

SGE and Torque integration processes

> **Note**
>
> Cette section s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE and Torque planificateurs, Amazon SNS et Amazon SQS.

Vue d'ensemble générale

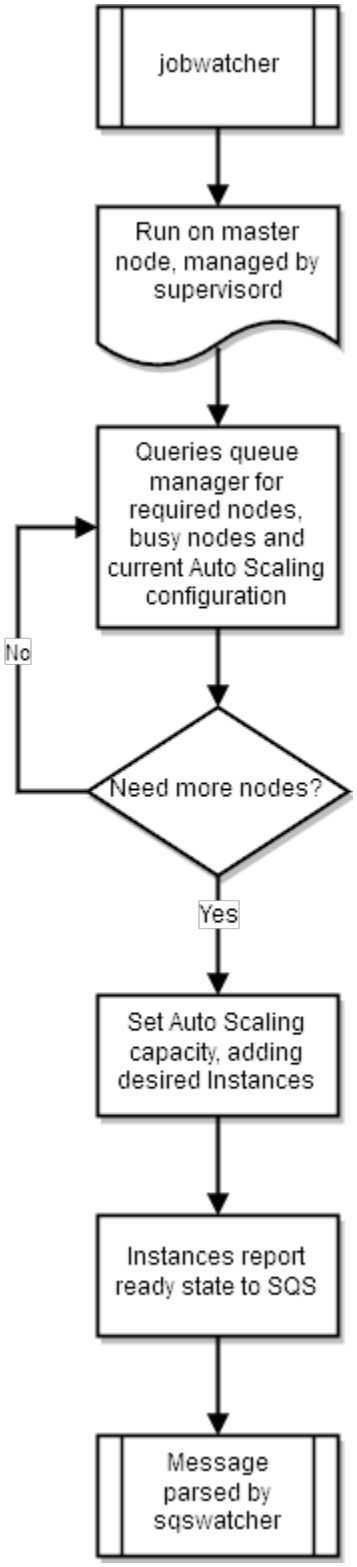
Le cycle de vie d'un cluster commence après sa création par un utilisateur. En règle générale, un cluster est créé à partir de l'interface de ligne de commande (CLI). Une fois créé, un cluster existe jusqu'à ce qu'il soit supprimé. AWS ParallelCluster les démons s'exécutent sur les nœuds du cluster, principalement pour gérer l'élasticité du cluster HPC. Le schéma suivant illustre un flux de travail utilisateur et le cycle de vie du cluster. Les sections suivantes décrivent les AWS ParallelCluster démons utilisés pour gérer le cluster.


User creates a cluster
CloudFormation launches cluster
Cluster is running
sqswatcher
jobwatcher
nodewatcher
Cluster Management Processes
User submits a job
Jobs are processed
User deletes cluster
CloudFormation deletes stack
Cluster is

Avec SGE and Torque planificateurs `nodewatcherjobwatcher`, AWS ParallelCluster utilisations et `sqswatcher` processus.

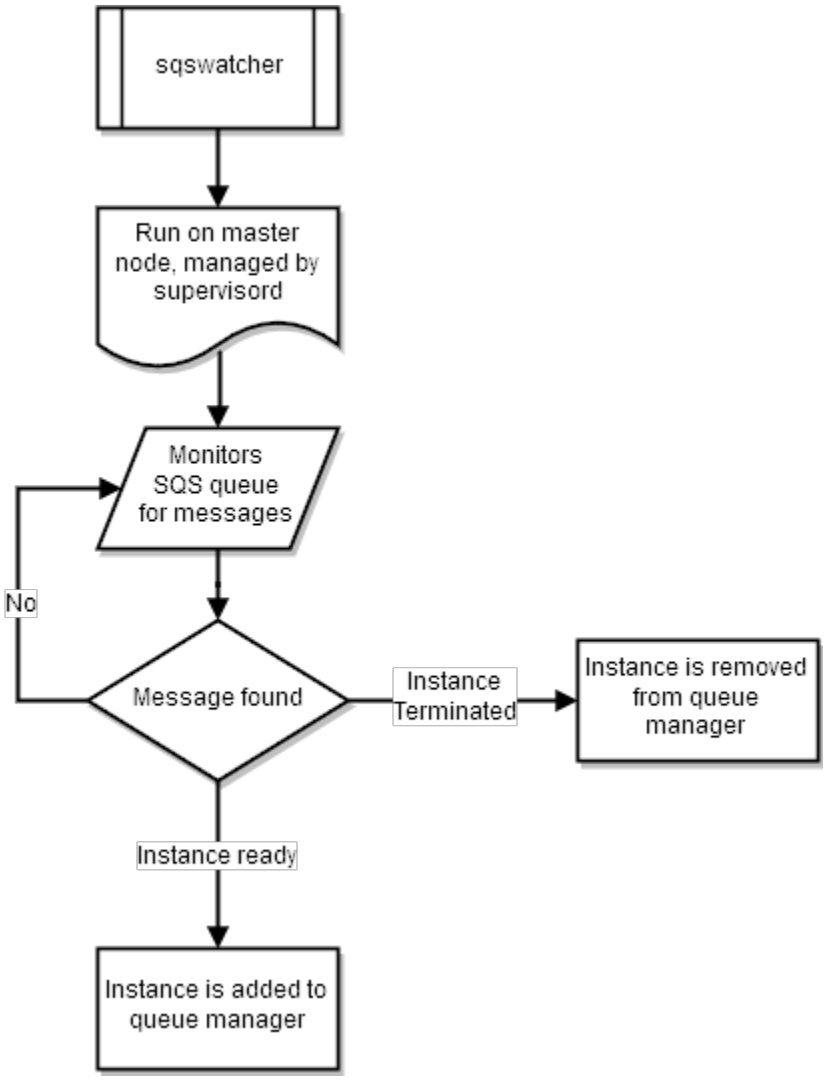
`jobwatcher`

Lorsqu'un cluster est en cours d'exécution, un processus appartenant à l'utilisateur root surveille le planificateur configuré (SGE or Torque). Chaque minute, il évalue la file d'attente afin de décider quand l'augmenter.


jobwatcher
Run on master node, managed by supervisord
Queries queue manager for required nodes, busy nodes and current Auto Scaling configuration
No
Need more nodes?
Yes
Set Auto Scaling capacity, adding desired Instances
Instances report ready state to SQS
Message parsed by sqswatcher

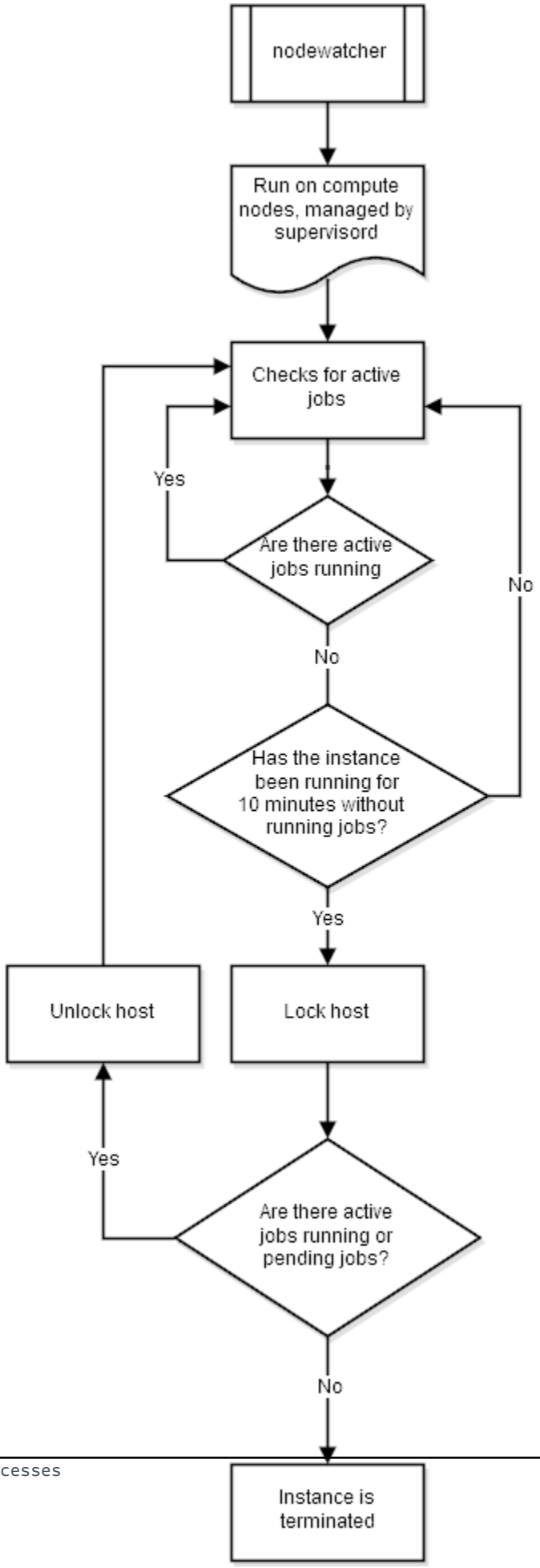
sqswatcher

Le `sqswatcher` processus surveille les messages Amazon SQS envoyés par Auto Scaling pour vous informer des changements d'état au sein du cluster. Lorsqu'une instance est mise en ligne, elle envoie un message « instance prête » à Amazon SQS. Ce message est capté par`sqs_watcher`, exécuté sur le nœud principal. Ces messages permettent d'informer le gestionnaire de file d'attente lorsque de nouvelles instances sont mises en ligne ou résiliées, pour qu'il puisse les ajouter à la file d'attente ou les en supprimer.



nodewatcher

Le processus `nodewatcher` s'exécute sur chaque nœud dans le parc d'instances de calcul. Une fois la période `scaledown_idletime` écoulée, telle que définie par l'utilisateur, l'instance est mise hors service.


nodewatcher
Run on compute nodes, managed by supervisord
Checks for active jobs
Yes
Are there active jobs running
No
No
Has the instance been running for 10 minutes without running jobs?
Yes
Unlock host
Lock host
Yes
Are there active jobs running or pending jobs?
No
Instance is terminated

Slurm integration processes

Avec Slurm les planificateurs, les AWS ParallelCluster usages `clustermgtd` et les `computemgt` processus.

clustermgtd

Les clusters qui s'exécutent en mode hétérogène (indiqué en spécifiant une `queue_settings` valeur) possèdent un processus daemon de gestion de cluster (`clustermgtd`) qui s'exécute sur le nœud principal. Ces tâches sont effectuées par le démon de gestion du cluster.

- Nettoyage de partitions inactives
- Gestion de la capacité statique : assurez-vous que la capacité statique est toujours active et saine
- Synchronisez le planificateur avec Amazon. EC2
- Nettoyage des instances orphelines
- Restaurez l'état du nœud du planificateur lors de la EC2 résiliation d'Amazon en dehors du flux de travail de suspension
- Gestion des EC2 instances Amazon défaillante (échec des bilans EC2 de santé Amazon)
- Gestion des événements de maintenance planifiés
- Gestion des nœuds du planificateur défaillante (échec des contrôles de santé du planificateur)

computemgtd

Les clusters qui s'exécutent en mode hétérogène (indiqué par la spécification d'une `queue_settings` valeur) possèdent des processus de gestion de calcul (daemon`computemgtd`) qui s'exécutent sur chacun des nœuds de calcul. Toutes les cinq (5) minutes, le démon de gestion du calcul confirme que le nœud principal est joignable et qu'il est sain. Si cinq (5) minutes s'écoulent pendant lesquelles le nœud principal ne peut pas être atteint ou n'est pas sain, le nœud de calcul est arrêté.

AWS services utilisés par AWS ParallelCluster

Les services Amazon Web Services (AWS) suivants sont utilisés par AWS ParallelCluster.

Rubriques

- AWS Auto Scaling
- AWS Batch
- AWS CloudFormation
- Amazon CloudWatch
- Amazon CloudWatch Logs
- AWS CodeBuild
- Amazon DynamoDB
- Amazon Elastic Block Store
- Amazon Elastic Compute Cloud
- Amazon Elastic Container Registry
- Amazon EFS
- Amazon FSx pour 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**
>
> Cette section s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5 AWS ParallelCluster , l'utilisation de. AWS Auto Scaling

AWS Auto Scaling est un service qui surveille vos applications et ajuste automatiquement la capacité en fonction de vos besoins de service spécifiques et changeants. Ce service gère vos ComputeFleet

instances en tant que groupe Auto Scaling. Le groupe peut être piloté de manière élastique par l'évolution de votre charge de travail ou fixé de manière statique par les configurations initiales de votre instance.

AWS Auto Scaling est utilisé avec des ComputeFleet instances mais n'est pas utilisé avec des AWS Batch clusters.

Pour plus d'informations sur AWS Auto Scaling, voir https://aws.amazon.com/autoscaling/et https://docs.aws.amazon.com/autoscaling/.

AWS Batch

AWS Batch est un service de planification de tâches AWS géré. Il fournit de manière dynamique la quantité et le type optimaux de ressources de calcul (par exemple, des instances optimisées pour le processeur ou la mémoire) dans AWS Batch les clusters. Ces ressources sont provisionnées en fonction des exigences spécifiques de vos tâches par lots, y compris les exigences en matière de volume. Grâce à AWS Batch cela, vous n'avez pas besoin d'installer ou de gérer des logiciels de traitement par lots ou des clusters de serveurs supplémentaires pour exécuter vos tâches efficacement.

AWS Batch est utilisé uniquement avec les AWS Batch clusters.

Pour plus d'informations sur AWS Batch, voir https://aws.amazon.com/batch/et https://docs.aws.amazon.com/batch/.

AWS CloudFormation

AWS CloudFormation est un infrastructure-as-code service qui fournit un langage commun pour modéliser AWS et fournir des ressources d'applications tierces dans votre environnement cloud. C'est le principal service utilisé par AWS ParallelCluster. Chaque cluster AWS ParallelCluster est représenté sous la forme d'une pile, et toutes les ressources requises par chaque cluster sont définies dans le AWS ParallelCluster AWS CloudFormation modèle. Dans la plupart des cas, les commandes AWS ParallelCluster CLI correspondent directement aux commandes de AWS CloudFormation pile, telles que les commandes de création, de mise à jour et de suppression. Les instances lancées au sein d'un cluster envoient des appels HTTPS au AWS CloudFormation point de terminaison dans Région AWS lequel le cluster est lancé.

Pour plus d'informations sur AWS CloudFormation, voir https://aws.amazon.com/cloudformation/et https://docs.aws.amazon.com/cloudformation/.

Amazon CloudWatch

Amazon CloudWatch (CloudWatch) est un service de surveillance et d'observabilité qui vous fournit des données et des informations exploitables. Ces informations peuvent être utilisées pour surveiller vos applications, répondre aux changements de performances et aux exceptions de service, et optimiser l'utilisation des ressources. In AWS ParallelCluster, CloudWatch est utilisé comme tableau de bord, pour surveiller et enregistrer les étapes de création de l'image Docker et le résultat des AWS Batch tâches.

Avant AWS ParallelCluster la version 2.10.0, CloudWatch il n'était utilisé qu'avec des AWS Batch clusters.

Pour plus d'informations sur CloudWatch, voir https://aws.amazon.com/cloudwatch/et https://docs.aws.amazon.com/cloudwatch/.

Amazon CloudWatch Logs

Amazon CloudWatch Logs (CloudWatch Logs) est l'une des fonctionnalités principales d'Amazon CloudWatch. Vous pouvez l'utiliser pour surveiller, stocker, afficher et rechercher dans les fichiers journaux de nombreux composants utilisés par AWS ParallelCluster.

Avant AWS ParallelCluster la version 2.6.0, CloudWatch Logs n'était utilisé qu'avec des AWS Batch clusters.

Pour de plus amples informations, veuillez consulter Intégration à Amazon CloudWatch Logs.

AWS CodeBuild

AWS CodeBuild (CodeBuild) est un service AWS géré d'intégration continue qui respecte le code source, exécute des tests et produit des progiciels prêts à être déployés. In AWS ParallelCluster, CodeBuild est utilisé pour créer automatiquement et de manière transparente des images Docker lors de la création de clusters.

CodeBuild est utilisé uniquement avec les AWS Batch clusters.

Pour plus d'informations sur CodeBuild, voir https://aws.amazon.com/codebuild/et https://docs.aws.amazon.com/codebuild/.

Amazon DynamoDB

Amazon DynamoDB (DynamoDB) est un service de base de données NoSQL rapide et flexible. Il est utilisé pour stocker les informations d'état minimales du cluster. Le nœud principal suit les instances provisionnées dans une table DynamoDB.

DynamoDB n'est pas utilisé avec les clusters. AWS Batch

Pour plus d'informations sur DynamoDB, reportez-vous aux sections et. https://aws.amazon.com/dynamodb/https://docs.aws.amazon.com/dynamodb/

Amazon Elastic Block Store

Amazon Elastic Block Store (Amazon EBS) est un service de stockage par blocs à hautes performances qui fournit un stockage persistant pour les volumes partagés. Tous les paramètres Amazon EBS peuvent être transmis via la configuration. Les volumes Amazon EBS peuvent être initialisés vides ou à partir d'un instantané Amazon EBS existant.

Pour plus d'informations sur Amazon EBS, consultez https://aws.amazon.com/ebs/et https://docs.aws.amazon.com/ebs/.

Amazon Elastic Compute Cloud

Amazon Elastic Compute Cloud (Amazon EC2) fournit la capacité de calcul pour AWS ParallelCluster. La tête et les nœuds de calcul sont des EC2 instances Amazon. N'importe quel type d'instance qui prend en charge HVM peut être sélectionné. Les nœuds de tête et de calcul peuvent être de types d'instances différents. De plus, si plusieurs files d'attente sont utilisées, certains ou tous les nœuds de calcul peuvent également être lancés en tant qu'instance ponctuelle. Les volumes de stockage d'instance disponibles sur les instances sont montés sous la forme d'un volume LVM agrégé par bandes.

Pour plus d'informations sur Amazon EC2, consultez https://aws.amazon.com/ec2/et https://docs.aws.amazon.com/ec2/.

Amazon Elastic Container Registry

Amazon Elastic Container Registry (Amazon ECR) est un registre de conteneurs Docker entièrement géré qui facilite le stockage, la gestion et le déploiement d'images de conteneurs Docker. Dans AWS ParallelCluster, Amazon ECR stocke les images Docker créées lors de la création de clusters.

Les images Docker sont ensuite utilisées AWS Batch pour exécuter les conteneurs pour les tâches soumises.

Amazon ECR est utilisé uniquement avec des AWS Batch clusters.

Pour plus d'informations, consultez https://aws.amazon.com/ecr/ et https://docs.aws.amazon.com/ecr/.

Amazon EFS

Amazon Elastic File System (Amazon EFS) fournit un système de fichiers NFS élastique simple, évolutif et entièrement géré à utiliser avec les AWS Cloud services et les ressources sur site. Amazon EFS est utilisé lorsque le `efs_settings` paramètre est spécifié et fait référence à une `[efs]`section. Support pour Amazon EFS a été ajouté dans la AWS ParallelCluster version 2.1.0.

Pour plus d'informations sur Amazon EFS, consultez https://aws.amazon.com/efs/et https://docs.aws.amazon.com/efs/.

Amazon FSx pour Lustre

FSx for Lustre fournit un système de fichiers performant qui utilise le système de fichiers open source Lustre. FSx for Lustre est utilisé lorsque le `fsx_settings` paramètre est spécifié et fait référence à une `[fsx]`section. Support FSx pour Lustre a été ajouté dans la AWS ParallelCluster version 2.2.1.

Pour plus d'informations sur FSx Lustre, voir https://aws.amazon.com/fsx/lustre/ et. https://docs.aws.amazon.com/fsx/

AWS Identity and Access Management

AWS Identity and Access Management (IAM) est utilisé dans le cadre AWS ParallelCluster pour fournir le rôle IAM le moins privilégié à Amazon EC2 pour l'instance spécifique à chaque cluster individuel. AWS ParallelCluster les instances n'ont accès qu'aux appels d'API spécifiques nécessaires au déploiement et à la gestion du cluster.

Avec les AWS Batch clusters, des rôles IAM sont également créés pour les composants impliqués dans le processus de création d'images Docker lors de la création de clusters. Ces composants incluent les fonctions Lambda qui sont autorisées à ajouter et à supprimer des images Docker dans et depuis le référentiel Amazon ECR. Elles incluent également les fonctions permettant de supprimer le compartiment Amazon S3 créé pour le cluster et le CodeBuild projet. Il existe également des rôles pour les AWS Batch ressources, les instances et les tâches.

Pour plus d'informations sur IAM, reportez-vous aux sections https://aws.amazon.com/iam/et https://docs.aws.amazon.com/iam/.

AWS Lambda

AWS Lambda (Lambda) exécute les fonctions qui orchestrent la création d'images Docker. Lambda gère également le nettoyage des ressources de cluster personnalisées, telles que les images Docker stockées dans le référentiel Amazon ECR et sur Amazon S3.

Pour plus d'informations sur Lambda, consultez https://aws.amazon.com/lambda/et. https://docs.aws.amazon.com/lambda/

Amazon DCV

Amazon DCV est un protocole d'affichage à distance hautes performances qui fournit un moyen sécurisé de diffuser des postes de travail distants et des applications en streaming sur n'importe quel appareil, quelles que soient les conditions du réseau. Amazon DCV est utilisé lorsque le `dcv_settings` paramètre est spécifié et fait référence à une `[dcv]`section. Support pour Amazon DCV a été ajouté dans la AWS ParallelCluster version 2.5.0.

Pour plus d'informations sur Amazon DCV, consultez https://aws.amazon.com/hpc/dcv/ et. https://docs.aws.amazon.com/dcv/

Amazon Route 53

Amazon Route 53 (Route 53) est utilisé pour créer des zones hébergées avec des noms d'hôtes et des noms de domaine complets pour chacun des nœuds de calcul.

Pour plus d'informations sur la Route 53, reportez-vous https://aws.amazon.com/route53/aux sections et https://docs.aws.amazon.com/route53/.

Amazon Simple Notification Service

> **Note**
>
> Cette section s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation d'Amazon Simple Notification Service AWS ParallelCluster n'est pas prise en charge.

Amazon Simple Notification Service (Amazon SNS) reçoit des notifications d'Auto Scaling. Ces événements sont appelés événements du cycle de vie et sont générés lorsqu'une instance démarre ou se termine dans un groupe Auto Scaling. Dans AWS ParallelCluster ce cadre, la rubrique Amazon SNS pour le groupe Auto Scaling est abonnée à une file d'attente Amazon SQS.

Amazon SNS n'est pas utilisé avec AWS Batch les clusters.

Pour plus d'informations sur Amazon SNS, consultez https://aws.amazon.com/sns/et. https://docs.aws.amazon.com/sns/

Amazon Simple Queue Service

> **Note**
>
> Cette section s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, l'utilisation d'Amazon Simple Queue Service AWS ParallelCluster n'est pas prise en charge.

Amazon Simple Queue Service (Amazon SQS) conserve les notifications envoyées depuis Auto Scaling, les notifications envoyées via Amazon SNS et les notifications envoyées depuis les nœuds de calcul. Amazon SQS dissocie l'envoi de notifications de la réception de notifications. Cela permet au nœud principal de gérer les notifications par le biais d'un processus de sondage. Dans ce processus, le nœud principal exécute Amazon SQSwatcher et interroge la file d'attente. Auto Scaling et les nœuds de calcul publient des messages dans la file d'attente.

Amazon SQS n'est pas utilisé avec AWS Batch les clusters.

Pour plus d'informations sur Amazon SQS, consultez https://aws.amazon.com/sqs/et. https://docs.aws.amazon.com/sqs/

Amazon Simple Storage Service

Amazon Simple Storage Service (Amazon S3) AWS ParallelCluster stocke les modèles situés dans chacun d'entre eux. Région AWS AWS ParallelCluster peut être configuré pour permettre aux outils CLI/SDK d'utiliser Amazon S3.

Lorsque vous utilisez un AWS Batch cluster, un compartiment Amazon S3 de votre compte est utilisé pour stocker les données associées. Par exemple, le bucket stocke les artefacts créés lorsqu'une image Docker et des scripts sont créés à partir de tâches soumises.

Pour plus d'informations, consultez https://aws.amazon.com/s3/ et https://docs.aws.amazon.com/s3/.

Amazon VPC

Amazon VPC définit un réseau utilisé par les nœuds de votre cluster. Les paramètres VPC du cluster sont définis dans la `[vpc]` section.

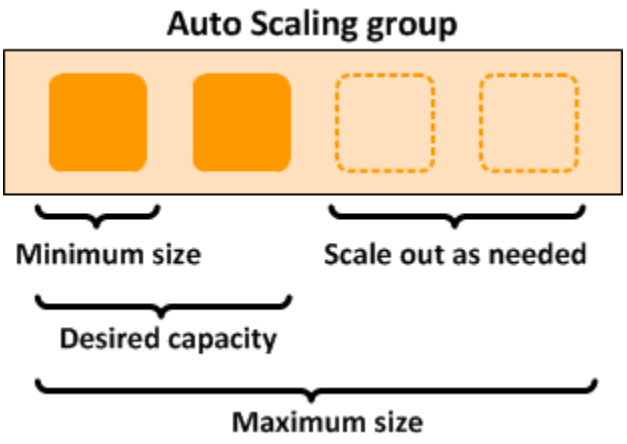
Pour plus d'informations sur Amazon VPC, consultez https://aws.amazon.com/vpc/et. https://docs.aws.amazon.com/vpc/

AWS ParallelCluster Auto Scaling

> **Note**
>
> Cette section s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs. Vous pouvez continuer à les utiliser dans les versions antérieures à la version 2.11.4, mais ils ne sont pas éligibles aux futures mises à jour ou à l'assistance en matière de résolution des problèmes de la part des équipes de AWS service et de AWS support.
> À partir de AWS ParallelCluster la version 2.9.0, Auto Scaling n'est plus compatible avec Slurm Workload Manager (Slurm). Pour en savoir plus sur Slurm et la mise à l'échelle de plusieurs files d'attente, consultez leTutoriel en mode file d'attente multiple.

La stratégie d'autoscaling décrite dans cette rubrique s'applique aux clusters HPC déployés avec Son of Grid Engine (SGE) ou Torque Resource Manager (Torque). En cas de déploiement avec l'un de ces planificateurs, AWS ParallelCluster implémente les fonctionnalités de dimensionnement en gérant le groupe Auto Scaling des nœuds de calcul, puis en modifiant la configuration du planificateur selon les besoins. Pour les clusters HPC basés sur AWS Batch, AWS ParallelCluster s'appuie sur les capacités de mise à l'échelle élastique fournies par le planificateur de tâches AWS géré. Pour plus d'informations, consultez What is Amazon EC2 Auto Scaling dans le guide de l'utilisateur d'Amazon EC2 Auto Scaling.

Les clusters déployés avec AWS ParallelCluster sont élastiques de plusieurs manières. La définition de `initial_queue_size` spécifie la valeur de taille minimale du groupe ComputeFleet Auto Scaling, ainsi que la valeur de capacité souhaitée. La définition de `max_queue_size` spécifie la valeur de taille maximale du groupe ComputeFleet Auto Scaling.



Mise à l'échelle

Chaque minute, un processus appelé `jobwatcher`s'exécute sur le nœud principal. Il évalue le nombre d'instances requis par les tâches en attente dans la file d'attente. Si le nombre total de nœuds occupés et de nœuds demandés est supérieur à la valeur actuellement souhaitée dans le groupe Auto Scaling, d'autres instances sont ajoutées. Si vous soumettez d'autres jobs, la file d'attente est réévaluée et le groupe Auto Scaling est mis à jour, jusqu'à la valeur spécifiée`max_queue_size`.

Avec un SGE planificateur, chaque tâche nécessite un certain nombre d'emplacements pour s'exécuter (un emplacement correspond à une unité de traitement, par exemple un vCPU). Lors de l'évaluation du nombre d'instances requises pour traiter les tâches actuellement en attente, `jobwatcher` divise le nombre total d'emplacements demandés par la capacité d'un nœud de calcul individuel. La capacité d'un nœud de calcul correspondant au nombre de v disponibles CPUs dépend du type d' EC2 instance Amazon spécifié dans la configuration du cluster.

Avec Slurm (avant AWS ParallelCluster la version 2.9.0) et Torque planificateurs, chaque tâche peut nécessiter à la fois un certain nombre de nœuds et un certain nombre de slots pour chaque nœud, selon les circonstances. Pour chaque demande, le `jobwatcher` détermine le nombre de nœuds de calcul qui sont nécessaires pour traiter les nouvelles exigences en matière de calcul. Par exemple, imaginons un cluster avec `c5.2xlarge` (8 vCPU) en tant que type d'instance de calcul en attente, et trois tâches mises en file d'attente avec les exigences suivantes :

- job1 : 2 nœuds / 4 emplacements chacun
- job2 : 3 nœuds / 2 emplacements chacun
- job3 : 1 nœud / 4 emplacements chacun

Dans cet exemple, trois nouvelles instances de calcul sont nécessaires dans le groupe Auto Scaling pour exécuter les trois tâches. `jobwatcher`

Limitation actuelle : la logique de mise à l'échelle automatique ne prend pas en compte les nœuds occupés partiellement chargés. Par exemple, un nœud qui exécute une tâche est considéré comme occupé même s'il y a des emplacements vides.

Diminution

Sur chaque nœud de calcul, un processus appelé `nodewatcher` s'exécute et évalue le temps d'inactivité du nœud. Une instance est mise hors service lorsque les deux conditions suivantes soient réunies :

- Une instance n'a pas de tâches pour une période plus longue que le paramètre `scaledown_idletime` (la valeur par défaut est 10 minutes)
- Il n'y a pas de tâches en attente dans le cluster

Pour mettre fin à une instance, `nodewatcher` appelle l'opération TerminateInstanceInAutoScalingGroupAPI, qui supprime une instance si la taille du groupe Auto Scaling est au moins égale à la taille minimale du groupe Auto Scaling. Ce processus diminue un cluster sans affecter l'exécution de tâches. Cela permet également de créer un cluster élastique avec un nombre de base fixe d'instances.

Cluster statique

La valeur de dimensionnement automatique est la même pour les charges de travail HPC que pour les autres charges de travail. La seule différence est que AWS ParallelCluster possède le code qui permet d'interagir plus intelligemment. Par exemple, si un cluster statique est requis, vous définissez les `max_queue_size` paramètres `initial_queue_size` et à la taille exacte du cluster requis, puis vous définissez le `maintain_initial_size` paramètre sur true. Cela donne au groupe ComputeFleet Auto Scaling la même valeur pour la capacité minimale, maximale et souhaitée.

Didacticiels

Les didacticiels suivants vous montrent comment démarrer et fournissent des conseils sur les meilleures pratiques pour certaines tâches courantes. AWS ParallelCluster

Rubriques

- Exécution de votre premier job sur AWS ParallelCluster
- Création d'une AWS ParallelCluster AMI personnalisée
- Exécution d'une tâche MPI avec un AWS ParallelCluster planificateur awsbatch
- Chiffrement du disque avec une clé KMS personnalisée
- Tutoriel en mode file d'attente multiple

Exécution de votre premier job sur AWS ParallelCluster

Ce didacticiel vous explique comment exécuter votre première tâche Hello World sur AWS ParallelCluster.

Prérequis

- AWS ParallelCluster est installé.
- Le AWS CLI est installé et configuré.
- Vous avez une paire de EC2 clés.
- Vous disposez d'un rôle IAM doté des autorisations requises pour exécuter la `pcluster` CLI.

Vérification de votre installation

Tout d'abord, nous vérifions qu' AWS ParallelCluster il est correctement installé et configuré.

```
$ pcluster version
```

Cela renvoie la version en cours d'exécution de AWS ParallelCluster. Si le résultat affiche un message concernant la configuration, vous devez exécuter ce qui suit pour configurer AWS ParallelCluster :

```
$ pcluster configure
```

Création de votre premier cluster

Il est temps de créer votre premier cluster. Comme la charge de travail pour ce didacticiel n'est pas exigeante, nous pouvons utiliser la taille d'instance par défaut, à savoir `t2.micro`. (Pour les charges de travail en production, vous choisissez une taille d'instance qui répond le mieux à vos besoins.)

Appelons votre cluster hello-world.

```
$ pcluster create hello-world
```

Lorsque le cluster est créé, vous obtenez une sortie similaire à ce qui suit :

```
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/
```

Le message `CREATE_COMPLETE` indique que le cluster a été créé avec succès. La sortie nous fournit également les adresses IP publiques et privées de notre nœud principal. Nous aurons besoin de cette adresse IP pour nous connecter.

Connexion à votre nœud principal

Utilisez votre fichier pem OpenSSH pour vous connecter à votre nœud principal.

```
pcluster ssh hello-world -i /path/to/keyfile.pem
```

Une fois connecté, exécutez la commande `qhost` pour vérifier que vos nœuds de calcul sont créés et configurés.

```
$ qhost
HOSTNAME                ARCH         NCPU NSOC NCOR NTHR  LOAD  MEMTOT  MEMUSE  SWAPTO
 SWAPUS
```

```
---------------------------------------------------------------------------------------------
global                  -               -    -    -    -     -       -       -       -
      -
ip-192-168-1-125        lx-amd64        2    1    2    2  0.15    3.7G  130.8M 1024.0M
    0.0
ip-192-168-1-126        lx-amd64        2    1    2    2  0.15    3.7G  130.8M 1024.0M
    0.0
```

La sortie montre que nous avons deux nœuds de calcul en cluster, tous deux avec deux threads disponibles.

Exécution de votre première tâche en utilisant SGE

> **Note**
>
> Cet exemple s'applique uniquement aux AWS ParallelCluster versions antérieures à la version 2.11.4 incluse. À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Ensuite, nous créons une tâche qui dort pendant un certain temps, puis génère son propre nom d'hôte.

Créez un fichier nommé `hellojob.sh` avec le contenu suivant :

```
#!/bin/bash
sleep 30
echo "Hello World from $(hostname)"
```

Ensuite, soumettez la tâche avec `qsub`, et vérifiez qu'elle s'exécute.

```
$ qsub hellojob.sh
Your job 1 ("hellojob.sh") has been submitted
```

Maintenant, vous pouvez afficher votre file d'attente et vérifier le statut de la tâche.

```
$ 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
```

La sortie indique que la tâche est actuellement en cours d'exécution. Attendez 30 secondes que la tâche se termine et exécutez `qstat` à nouveau.

```
$ qstat
$
```

Maintenant qu'il n'y a pas de tâche dans la file d'attente, nous pouvons rechercher la sortie dans notre répertoire actuel.

```
$ 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
```

Dans la sortie, nous constatons un « `e1` » et un fichier « `o1` » fichier dans notre script de tâche. Le `e1` fichier étant vide, aucune sortie n'a été envoyée à stderr. Si nous affichons le fichier `o1`, nous pouvons voir la sortie de notre tâche.

```
$ cat hellojob.sh.o1
Hello World from ip-192-168-1-125
```

La sortie indique également que notre tâche a été exécutée avec succès sur l'instance `ip-192-168-1-125`.

Pour en savoir plus sur la création et l'utilisation de clusters, consultezBonnes pratiques.

Création d'une AWS ParallelCluster AMI personnalisée

> **Important**
>
> Nous ne recommandons pas de créer une AMI personnalisée comme approche de personnalisation AWS ParallelCluster.
> En effet, une fois que vous avez créé votre propre AMI, vous ne recevrez plus de mises à jour ni de corrections de bogues dans les futures versions de AWS ParallelCluster. De plus,

si vous créez une AMI personnalisée, vous devez répéter les étapes que vous avez utilisées pour créer votre AMI personnalisée à chaque nouvelle AWS ParallelCluster version.

Avant de poursuivre votre lecture, nous vous recommandons de consulter la section Actions Bootstrap personnalisées pour déterminer si les modifications que vous souhaitez apporter peuvent être scriptées et prises en charge dans les futures AWS ParallelCluster versions.

Même si la création d'une AMI personnalisée n'est pas idéale (pour les raisons mentionnées précédemment), il existe toujours des scénarios dans lesquels il AWS ParallelCluster est nécessaire de créer une AMI personnalisée pour. Ce didacticiel vous guide tout au long du processus de création d'une AMI personnalisée pour ces scénarios.

> **Note**
>
> À partir de AWS ParallelCluster la version 2.6.1, la plupart des recettes d'installation sont ignorées par défaut lors du lancement des nœuds. Cela permet d'améliorer les temps de démarrage. Pour exécuter toutes les recettes d'installation afin d'améliorer la rétrocompatibilité au détriment des temps de démarrage, ajoutez `"skip_install_recipes" : "no"` la `cluster` clé dans le `extra_json` paramètre. Par exemple :
>
> ```
> extra_json = { "cluster" : { "skip_install_recipes" : "no" } }
> ```

Prérequis

- AWS ParallelCluster est installé.
- Le AWS CLI est installé et configuré.
- Vous avez une paire de EC2 clés.
- Vous disposez d'un rôle IAM doté des autorisations requises pour exécuter la `pcluster` CLI.

Comment personnaliser l' AWS ParallelCluster AMI

Il existe trois manières d'utiliser une AWS ParallelCluster AMI personnalisée décrites dans les sections suivantes. Deux de ces trois méthodes nécessitent que vous créiez une nouvelle AMI disponible sous votre nom Compte AWS. La troisième méthode (utiliser une AMI personnalisée au

moment de l'exécution) ne vous oblige pas à créer quoi que ce soit à l'avance, mais elle ajoute des risques au déploiement. Choisissez la méthode qui répond le mieux à vos besoins.

Modification d'une AMI

Il s'agit de la méthode la plus sûre et la plus recommandée. Comme l' AWS ParallelCluster AMI de base est souvent mise à jour avec les nouvelles versions, elle possède tous les composants nécessaires AWS ParallelCluster pour fonctionner lors de son installation et de sa configuration. Vous pouvez démarrer avec cette base.

New EC2 console

1. Dans la liste des AWS ParallelCluster AMI, recherchez l'AMI correspondant à l'AMI spécifique Région AWS que vous utilisez. La liste d'AMI que vous choisissez doit correspondre à la version AWS ParallelCluster que vous utilisez. Exécutez `pcluster version` pour vérifier la version. Pour AWS ParallelCluster la version 2.11.9, accédez à https://github.com/aws/aws-parallelcluster/blob/v2.11.9/amis .txt. Pour sélectionner une autre version, utilisez le même lien, cliquez sur le bouton Tag : 2.11.9, sélectionnez l'onglet Tags, puis sélectionnez la version appropriée.
2. Connectez-vous à la EC2 console Amazon AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/ec2/.
3. Dans le tableau de EC2 bord Amazon, choisissez Launch instance.
4. Dans Images de l'application et du système d'exploitation, choisissez Parcourir davantage AMIs, accédez à Communauté AMIs et entrez l'ID AWS ParallelCluster AMI correspondant à votre identifiant AMI Région AWS dans le champ de recherche.
5. Sélectionnez l'AMI, choisissez le type et les propriétés de votre instance, sélectionnez votre paire de clés et lancez l'instance.
6. Connectez-vous à votre instance à l'aide de l'utilisateur du système d'exploitation et de votre clé SSH. Pour plus d'informations, accédez à Instances, sélectionnez la nouvelle instance et Connect.
7. Personnalisez votre instance selon vos besoins.
8. Exécutez la commande suivante pour préparer votre instance pour la création d'une AMI :

   ```
   sudo /usr/local/sbin/ami_cleanup.sh
   ```

9. Accédez à Instances, choisissez la nouvelle instance, sélectionnez État de l'instance et Arrêtez l'instance.

10. Créez une nouvelle AMI à partir de l'instance à l'aide de la EC2 console ou de la commande AWS CLI create-image.

    Depuis la EC2 console

    a. Choisissez Instances dans le volet de navigation.
    b. Choisissez l'instance que vous avez créée et modifiée.
    c. Dans Actions, sélectionnez Image et modèles, puis Créer une image.
    d. Choisissez Créer une image.

11. Entrez le nouvel identifiant d'AMI dans le champ custom_ami de la configuration de votre cluster.

Old EC2 console

1. Dans la liste des AWS ParallelCluster AMI, recherchez l'AMI correspondant à l'AMI spécifique Région AWS que vous utilisez. La liste d'AMI que vous choisissez doit correspondre à la version AWS ParallelCluster que vous utilisez. Exécutez `pcluster version` pour vérifier la version. Pour AWS ParallelCluster la version 2.11.9, accédez à https://github.com/aws/aws-parallelcluster/blob/v2.11.9/amis .txt. Pour sélectionner une autre version, utilisez le même lien, cliquez sur le bouton Tag : 2.11.9, sélectionnez l'onglet Tags, puis sélectionnez la version appropriée.
2. Connectez-vous à la EC2 console Amazon AWS Management Console et ouvrez-la à l'adresse https://console.aws.amazon.com/ec2/.
3. Dans le tableau de EC2 bord Amazon, choisissez Launch instance.
4. Choisissez Communauté AMIs, recherchez l'ID AWS ParallelCluster AMI, puis sélectionnez-le.
5. Choisissez votre type d'instance et sélectionnez Suivant : Configurer les détails de l'instance ou Vérifier et lancer pour lancer votre instance.
6. Choisissez Launch, sélectionnez votre paire de clés et lancez les instances.
7. Connectez-vous à votre instance à l'aide de l'utilisateur du système d'exploitation et de votre clé SSH. Pour plus d'informations, accédez à Instances, sélectionnez la nouvelle instance et Connect.
8. Personnalisez votre instance selon vos besoins.
9. Exécutez la commande suivante pour préparer votre instance pour la création d'une AMI :

```
sudo /usr/local/sbin/ami_cleanup.sh
```

10. Accédez à Instances, choisissez la nouvelle instance, sélectionnez État de l'instance et Arrêtez
11. Créez une nouvelle AMI à partir de l'instance à l'aide de la EC2 console ou de la commande AWS CLI create-image.

    Depuis la EC2 console

    a. Choisissez Instances dans le volet de navigation.
    b. Choisissez l'instance que vous avez créée et modifiée.
    c. Dans Actions, choisissez Image, puis Créer une image.
    d. Choisissez Créer une image.
12. Entrez le nouvel identifiant d'AMI dans le champ custom_ami de la configuration de votre cluster.

Création d'une AWS ParallelCluster AMI personnalisée

Si vous avez une AMI personnalisée et des logiciels déjà en place, vous pouvez appliquer les modifications nécessitées par AWS ParallelCluster par dessus celui-ci.

1. Installez ce qui suit dans votre système local, avec la AWS ParallelCluster CLI :
   - Packer: recherchez la dernière version de votre système d'exploitation sur le site Web de Packer et installez-la. La version doit être au moins 1.4.0, mais la dernière version est recommandée. Vérifiez que la `packer` commande est disponible dans votre PATH.

   > **Note**
   >
   > Avant AWS ParallelCluster la version 2.8.0, Berkshelf (qui est installé à l'aide de`gem install berkshelf`) devait être utilisé. `pcluster createami`

2. Configurez vos Compte AWS informations d'identification afin que Packer puisse appeler les opérations AWS d'API en votre nom. L'ensemble minimal d'autorisations requises pour que Packer fonctionne est documenté dans la section Rôle de tâche ou d'instance IAM de la rubrique Amazon AMI Builder de la documentation Packer.
3. Utilisez la commande `createami` de la AWS ParallelCluster CLI pour créer une AWS ParallelCluster AMI à partir de celle que vous avez fournie comme base :

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

> **Important**
>
> Vous ne devez pas utiliser une AWS ParallelCluster AMI provenant d'un cluster en cours d'exécution comme `<BASE_AMI>` pour la `createami` commande. Dans le cas contraire, la commande échoue.

Pour les autres paramètres, voir`pcluster createami`.

4. La commande de l'étape 4 exécute Packer, qui effectue spécifiquement les opérations suivantes :
   a. Lance une instance à l'aide de l'AMI de base fournie.
   b. Applique le AWS ParallelCluster livre de recettes à l'instance pour installer le logiciel approprié et effectuer les autres tâches de configuration nécessaires.
   c. Arrête l'instance.
   d. Crée une nouvelle AMI à partir de l'instance.
   e. Résilie l'instance une fois que l'AMI a été créée.
   f. Fournit en sortie la chaîne d'ID de la nouvelle AMI à utiliser pour créer votre cluster.
5. Pour créer votre cluster, entrez l'ID de l'AMI dans le champ custom_ami de la configuration de votre cluster.

> **Note**
>
> Le type d'instance utilisé pour créer une AWS ParallelCluster AMI personnalisée `estt2.xlarge`. Ce type d'instance n'est pas éligible au niveau AWS gratuit. Par conséquent, toutes les instances créées lors de la création de cette AMI vous sont facturées.

Utilisation d'une AMI personnalisée lors de l'exécution

> **Warning**
>
> Pour éviter le risque d'utiliser une AMI incompatible avec AWS ParallelCluster, nous vous recommandons d'éviter d'utiliser cette méthode.

Lorsque des nœuds de calcul sont lancés alors qu'ils n'ont potentiellement pas été testés AMIs au moment de l'exécution, des incompatibilités avec l'installation AWS ParallelCluster du logiciel requis peuvent AWS ParallelCluster entraîner l'arrêt du fonctionnement.

Si vous ne souhaitez rien créer à l'avance, vous pouvez utiliser votre AMI et en créer une AWS ParallelCluster à partir de cette AMI.

Avec cette méthode, la création prend plus AWS ParallelCluster de temps car tous les logiciels nécessaires au AWS ParallelCluster moment de la création du cluster doivent être installés. De plus, la mise à l'échelle prend également plus de temps.

- Entrez l'ID d'AMI dans le champ custom_ami de la configuration de votre cluster.

Exécution d'une tâche MPI avec un AWS ParallelCluster planificateur `awsbatch`

Ce didacticiel vous explique comment exécuter une tâche MPI avec `awsbatch` comme planificateur.

Prérequis

- AWS ParallelCluster est installé.
- Le AWS CLI est installé et configuré.
- Vous avez une paire de EC2 clés.
- Vous disposez d'un rôle IAM doté des autorisations requises pour exécuter la `pcluster` CLI.

Création du cluster

Tout d'abord, nous allons créer une configuration pour un cluster qui utilise `awsbatch` comme planificateur. Veillez à remplacer les données manquantes dans la section `vpc` et le champ `key_name` avec les ressources que vous avez créées lors de la configuration.

```
[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-#######
```

Vous pouvez maintenant commencer à créer le cluster. Appelons notre cluster*awsbatch-tutorial*.

```
$ pcluster create -c /path/to/the/created/config/aws_batch.config -t awsbatch awsbatch-tutorial
```

Lorsque le cluster est créé, vous obtenez une sortie similaire à ce qui suit :

```
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
```

Connexion à votre nœud principal

Les commandes AWS ParallelCluster Batch CLI sont toutes disponibles sur la machine cliente sur laquelle AWS ParallelCluster elle est installée. Cependant, nous allons entrer en SSH dans le nœud principal et soumettre les tâches à partir de là. Cela nous permet de tirer parti du volume NFS partagé entre le responsable et toutes les instances Docker qui exécutent AWS Batch des tâches.

Utilisez votre fichier PEM SSH pour vous connecter à votre nœud principal.

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

Lorsque vous êtes connecté, exécutez les commandes `awsbqueues` `awsbhosts` pour afficher la AWS Batch file d'attente configurée et les instances Amazon ECS en cours d'exécution.

```
[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
```

Comme vous pouvez le voir à partir de la sortie, nous avons une seule exécution hôte. Cela est dû à la valeur que nous avons choisie pour `min_vcpus` dans la configuration. Si vous souhaitez afficher des informations supplémentaires sur la AWS Batch file d'attente et les hôtes, ajoutez l'`-d`indicateur à la commande.

Exécution de votre première tâche en utilisant AWS Batch

Avant de passer à MPI, nous allons créer une tâche factice simple qui restera en veille pendant un court instant, puis fournira en sortie son propre nom d'hôte, en saluant le nom transmis comme paramètre.

Créez un fichier nommé « hellojob.sh » avec le contenu suivant.

```
#!/bin/bash

sleep 30
echo "Hello $1 from $HOSTNAME"
echo "Hello $1 from $HOSTNAME" > "/shared/secret_message_for_${1}_by_
${AWS_BATCH_JOB_ID}"
```

Ensuite, soumettez la tâche avec `awsbsub` et vérifiez qu'elle s'exécute.

```
$ awsbsub -jn hello -cf hellojob.sh Luca
Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2 (hello) has been submitted.
```

Affichez votre file d'attente et vérifiez le statut de la tâche.

```
$ awsbstat
jobId                                    jobName      status    startedAt
 stoppedAt    exitCode
------------------------------------  -----------  --------  -------------------
 -----------  ----------
6efe6c7c-4943-4c1a-baf5-edbfeccab5d2  hello        RUNNING   2018-11-12 09:41:29  -
        -
```

La sortie fournit des informations détaillées pour la tâche.

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

Notez que la tâche est actuellement dans l'état `RUNNING`. Attendez 30 secondes que la tâche se termine et exécutez `awsbstat` à nouveau.

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

À présent, vous pouvez voir que la tâche se trouve dans l'état `SUCCEEDED`.

```
$ awsbstat -s SUCCEEDED
jobId                                    jobName      status     startedAt
 stoppedAt              exitCode
------------------------------------  -----------  ---------  -------------------
 -------------------  ----------
6efe6c7c-4943-4c1a-baf5-edbfeccab5d2  hello        SUCCEEDED  2018-11-12 09:41:29
 2018-11-12 09:42:00           0
```

Maintenant qu'il n'y a pas de tâche dans la file d'attente, nous pouvons rechercher la sortie via la commande `awsbout`.

```
$ awsbout 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
2018-11-12 09:41:29: Starting Job 6efe6c7c-4943-4c1a-baf5-edbfeccab5d2
download: s3://amzn-s3-demo-bucket/batch/job-hellojob_sh-1542015680924.sh to tmp/batch/
job-hellojob_sh-1542015680924.sh
2018-11-12 09:42:00: Hello Luca from ip-172-31-4-234
```

Nous pouvons voir que notre tâche s'est exécutée correctement sur l'instance « ip-172-31-4-234 ».

De plus, si vous examinez le répertoire `/shared`, vous y trouverez un message secret qui vous est destiné :

Pour explorer toutes les fonctionnalités qui ne sont pas disponibles dans le cadre de ce didacticiel, consultez la documentation de l'AWS ParallelCluster interface de ligne de commande Batch. Lorsque vous êtes prêt à poursuivre le didacticiel, continuons et voyons comment soumettre une tâche MPI.

Exécution d'une tâche MPI dans un environnement parallèle à plusieurs nœuds

Tout en restant connecté au nœud principal, créez un fichier dans le `/shared` répertoire nommé`mpi_hello_world.c`. Ajoutez le programme MPI suivante au fichier :

```
// 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();
}
```

Maintenant, enregistrez le code suivant en tant que `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
```

Nous sommes désormais prêts à soumettre notre première tâche MPI et à l'exécuter simultanément sur 3 nœuds :

```
$ awsbsub -n 3 -cf submit_mpi.sh
```

Nous allons maintenant surveiller le statut de la tâche et attendre qu'elle présente le statut `RUNNING` :

```
$ watch awsbstat -d
```

Une fois que la tâche est passée à l'état `RUNNING`, nous pouvons examiner sa sortie. Pour afficher la sortie du nœud principal, ajoutez `#0` à l'ID de la tâche. Pour afficher la sortie des nœuds de calcul, utilisez #1 et #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!
```

Nous pouvons désormais confirmer que la tâche s'est terminée avec succès :

```
[ec2-user@ip-10-0-0-111 ~]$ awsbstat -s ALL
jobId                                   jobName        status     startedAt
 stoppedAt            exitCode
------------------------------------  -------------  ---------  -------------------
 -------------------  ----------
5b4d50f8-1060-4ebf-ba2d-1ae868bbd92d  submit_mpi_sh  SUCCEEDED  2018-11-27 15:50:10
 2018-11-27 15:51:26  -
```

Remarque : si vous voulez résilier une tâche avant qu'elle ne se termine, utilisez la commande `awsbkill`.

Chiffrement du disque avec une clé KMS personnalisée

AWS ParallelCluster prend en charge les options de configuration `ebs_kms_key_id` et `fsx_kms_key_id`. Ces options vous permettent de fournir une AWS KMS clé personnalisée pour le chiffrement des disques Amazon EBS ou FSx pour Lustre. Pour les utiliser, vous spécifiez un `ec2_iam_role`.

Pour que le cluster puisse être créé, la AWS KMS clé doit connaître le nom du rôle du cluster. Cela vous empêche d'utiliser le rôle créé à la création du cluster, nécessitant un `ec2_iam_role` personnalisé.

Prérequis

- AWS ParallelCluster est installé.
- Le AWS CLI est installé et configuré.
- Vous avez une paire de EC2 clés.
- Vous disposez d'un rôle IAM doté des autorisations requises pour exécuter la `pcluster` CLI.

Création du rôle

Tout d'abord, vous devez créer une stratégie :

1. Accédez à la console IAM : https://console.aws.amazon.com/iam/accueil.
2. Sous Policies (Stratégies), Create policy (Créer une stratégie), cliquez sur l'onglet JSON.
3. Comme stratégie du corps, collez dans Instance Policy (Stratégie d'instance). Assurez-vous de remplacer toutes les occurrences de *<AWS ACCOUNT ID>* et *<REGION>*.
4. Nommez la stratégie `ParallelClusterInstancePolicy`, puis cliquez sur Create Policy (Créer une stratégie).

Ensuite, créez un rôle :

1. Sous Rôles, créez un rôle.
2. Cliquez sur `EC2` comme entité de confiance.
3. Sous Autorisations, recherchez le rôle `ParallelClusterInstancePolicy` que vous venez de créer et attachez-le.
4. Vérifiez le rôle `ParallelClusterInstanceRole`, puis cliquez sur Create Role (Créer un rôle).

Donnez vos autorisations clés

Dans la AWS KMS console **>** Clés gérées par le client **>** cliquez sur l'alias ou l'ID de clé de votre clé.

Cliquez sur le bouton Ajouter dans la zone Utilisateurs clés, sous l'onglet Politique clé, et recherchez le nom ParallelClusterInstanceRoleque vous venez de créer. Attachez-le.

Création du cluster

Maintenant créez un cluster. Voici un exemple de cluster avec des disques `Raid 0` chiffrés :

```
[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
```

Voici un exemple avec le système de fichiers FSx for Lustre :

```
[cluster default]
...
fsx_settings = fs
ec2_iam_role = ParallelClusterInstanceRole

[fsx fs]
shared_dir = /fsx
storage_capacity = 3600
imported_file_chunk_size = 1024
export_path = s3://bucket/folder
import_path = s3://bucket
weekly_maintenance_start_time = 1:00:00
fsx_kms_key_id = xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx
```

Des configurations similaires s'appliquent à Amazon EBS et aux systèmes de fichiers FSx basés sur Amazon.

Tutoriel en mode file d'attente multiple

Exécution de vos tâches AWS ParallelCluster en mode file d'attente multiple

Ce didacticiel vous explique comment exécuter votre première tâche Hello World AWS ParallelCluster avecMode de file d'attente multiple.

Prérequis

- AWS ParallelCluster est installé.
- Le AWS CLI est installé et configuré.
- Vous avez une paire de EC2 clés.
- Vous disposez d'un rôle IAM doté des autorisations requises pour exécuter la `pcluster` CLI.

> **Note**
>
> Le mode de file d'attente multiple n'est pris en charge que pour AWS ParallelCluster la version 2.9.0 ou ultérieure.

Configuration de votre cluster

Tout d'abord, vérifiez qu' AWS ParallelCluster il est correctement installé en exécutant la commande suivante.

```
$ pcluster version
```

Pour plus d'informations sur `pcluster version`, consultez `pcluster version`.

Cette commande renvoie la version en cours d'exécution de AWS ParallelCluster.

Exécutez ensuite `pcluster configure` pour générer un fichier de configuration de base. Suivez toutes les instructions qui suivent cette commande.

```
$ pcluster configure
```

Pour plus d'informations sur la commande `pcluster configure`, consultez `pcluster configure`.

Une fois cette étape terminée, vous devriez avoir un fichier de configuration de base sous`~/.parallelcluster/config`. Ce fichier doit contenir une configuration de base du cluster et une section VPC.

La partie suivante du didacticiel explique comment modifier la configuration que vous venez de créer et lancer un cluster avec plusieurs files d'attente.

> **Note**
>
> Certaines instances utilisées dans ce didacticiel ne sont pas éligibles au niveau gratuit.

Pour ce didacticiel, utilisez la configuration suivante.

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue

[aws]
aws_region_name = <Your Région 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
```

Création de votre cluster

Cette section explique comment créer le cluster en mode files d'attente multiples.

Tout d'abord, nommez votre cluster `multi-queue-hello-world` et créez-le conformément à la section du `multi-queue` cluster définie dans la section précédente.

```
$ pcluster create multi-queue-hello-world -t multi-queue
```

Pour plus d'informations sur `pcluster create`, consultez `pcluster create`.

Lorsque le cluster est créé, le résultat suivant s'affiche :

```
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
```

Le message `CREATE_COMPLETE` indique que le cluster a été créé avec succès. La sortie fournit également les adresses IP publiques et privées du nœud principal.

Connexion à votre nœud principal

Utilisez votre fichier de clé SSH privée pour vous connecter à votre nœud principal.

```
$ pcluster ssh multi-queue-hello-world -i ~/path/to/keyfile.pem
```

Pour plus d'informations sur `pcluster ssh`, consultez `pcluster ssh`.

Une fois connecté, exécutez la `sinfo` commande pour vérifier que les files d'attente de votre planificateur sont installées et configurées.

Pour plus d'informations sur`sinfo`, voir sinfo dans le Slurm documentation.

```
$ 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
```

La sortie indique que vous disposez de deux nœuds de `t2.micro` calcul dans l'`idle`état qui sont disponibles dans votre cluster.

> **Note**
>
> - `spot-st-t2micro-1`est un nœud statique dont le nom `st` figure dans son nom. Ce nœud est toujours disponible et correspond `min_count = 1` à la configuration de votre cluster.
> - `spot-dy-t2micro-1`est un nœud dynamique dont le nom `dy` figure dans son nom. Ce nœud est actuellement disponible car il correspond à `initial_count - min_count = 1` la configuration de votre cluster. Ce nœud est réduit au bout `scaledown_idletime` de cinq minutes, comme vous le souhaitez.

Les autres nœuds sont tous en état d'économie d'énergie, comme le montre le ~ suffixe dans l'état du nœud, sans qu'aucune EC2 instance ne les sauvegarde. La file d'attente par défaut est désignée par un `*` suffixe après son nom, tout comme votre file d'attente de tâches par défaut. `spot`

Exécution d'une tâche en mode file d'attente multiple

Ensuite, essayez d'exécuter une tâche pour dormir pendant un moment. La tâche produira ultérieurement son propre nom d'hôte. Assurez-vous que ce script peut être exécuté par l'utilisateur actuel.

```
$ 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
```

Soumettez le travail à l'aide de la `sbatch` commande. Demandez deux nœuds pour cette tâche avec l'`-N 2`option, et vérifiez que la tâche est correctement soumise. Pour plus d'informations sur `sbatch`, consultez .sbatchdans la documentation de Slurm.

```
$ sbatch -N 2 --wrap "srun hellojob.sh"
Submitted batch job 2
```

Vous pouvez consulter votre file d'attente et vérifier l'état de la tâche à l'aide de la `squeue` commande. Notez que, comme vous n'avez pas spécifié de file d'attente spécifique, la file d'attente par défaut (`spot`) est utilisée. Pour plus d'informations sur `squeue`, consultez .squeue dans le .Slurmdocumentation.

```
$ 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
```

La sortie indique que la tâche est actuellement en cours d'exécution. Attendez 30 secondes que la tâche se termine et exécutez `squeue` à nouveau.

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

Maintenant que les tâches de la file d'attente sont toutes terminées, recherchez le fichier de sortie `slurm-2.out` dans votre répertoire actuel.

```
$ cat slurm-2.out
Hello World from spot-dy-t2micro-1
Hello World from spot-st-t2micro-1
```

Le résultat indique également que notre tâche s'est exécutée avec succès sur les `spot-st-t2micro-2` nœuds `spot-st-t2micro-1` et.

Soumettez maintenant le même travail en spécifiant des contraintes pour des instances spécifiques à l'aide des commandes suivantes.

```
$ sbatch -N 3 -p spot -C "[c5.xlarge*1&t2.micro*2]" --wrap "srun hellojob.sh"
Submitted batch job 3
```

Vous avez utilisé ces paramètres pour`sbatch`.

- `-N 3`— demande trois nœuds
- `-p spot`— soumet la tâche à la `spot` file d'attente. Vous pouvez également soumettre une tâche à la `ondemand` file d'attente en spécifiant`-p ondemand`.
- `-C "[c5.xlarge*1&t2.micro*2]"`— spécifie les contraintes de nœud spécifiques pour cette tâche. Cela demande qu'un (1) `c5.xlarge` nœud et deux (2) `t2.micro` nœuds soient utilisés pour cette tâche.

Exécutez la `sinfo` commande pour afficher les nœuds et les files d'attente. (Les files d'attente AWS ParallelCluster sont appelées partitions dans 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
```

Les nœuds s'allument. Cela est indiqué par le # suffixe sur l'état du nœud. Exécutez le squeue commande pour afficher les informations relatives aux tâches du 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
```

Votre travail est dans le `CF` (CONFIGURING), en attente de la montée en puissance des instances et de leur intégration au cluster.

Au bout de trois minutes environ, les nœuds devraient être disponibles et la tâche devrait entrer dans le champ `R` (RUNNING) état.

```
$ 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
```

La tâche est terminée et les trois nœuds sont dans l'`idle`état.

```
$ 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
```

Ensuite, une fois qu'il ne reste aucune tâche dans la file d'attente, vous pouvez la rechercher `slurm-3.out` dans votre répertoire local.

```
$ cat slurm-3.out
Hello World from spot-dy-c5xlarge-1
Hello World from spot-st-t2micro-1
Hello World from spot-dy-t2micro-1
```

La sortie indique également que la tâche s'est exécutée avec succès sur les nœuds correspondants.

Vous pouvez observer le processus de réduction. Dans la configuration de votre cluster, vous avez spécifié une durée personnalisée `scaledown_idletime` de 5 minutes. Après cinq minutes d'inactivité, vos nœuds `spot-dy-c5xlarge-1` dynamiques `spot-dy-t2micro-1` sont automatiquement réduits et passent en `POWER_DOWN` mode. Notez que le nœud statique `spot-st-t2micro-1` n'est pas réduit.

```
$ 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
```

D'après le code ci-dessus, vous pouvez le voir `spot-dy-c5xlarge-1` et vous `spot-dy-t2micro-1` êtes en `POWER_DOWN` mode. Cela est indiqué par le `%` suffixe. Les instances correspondantes sont immédiatement mises hors service, mais les nœuds restent dans `POWER_DOWN` cet état et ne peuvent pas être utilisés pendant 120 secondes (deux minutes). Passé ce délai, les nœuds recommencent à économiser de l'énergie et peuvent être réutilisés. Pour de plus amples informations, veuillez consulter Slurmguide pour le mode de file d'attente multiple.

Cela devrait être l'état final du 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
```

Après vous être déconnecté du cluster, vous pouvez le nettoyer en exécutant`pcluster delete`. Pour plus d'informations, sur `pcluster list` et`pcluster delete`, voir `pcluster list` et`pcluster delete`.

```
$ pcluster list
multi-queue CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue
Deleting: multi-queue
...
```

Exécution de tâches sur un cluster avec des instances EFA et GPU

Cette partie du didacticiel explique comment modifier la configuration et lancer un cluster avec plusieurs files d'attente contenant des instances dotées d'un réseau EFA et de ressources GPU. Notez que les instances utilisées dans ce didacticiel sont des instances dont le prix est plus élevé.

Vérifiez les limites de votre compte pour vous assurer que vous êtes autorisé à utiliser ces instances avant de suivre les étapes décrites dans ce didacticiel.

Modifiez le fichier de configuration à l'aide de ce qui suit.

```
[global]
update_check = true
sanity_check = true
cluster_template = multi-queue-special

[aws]
aws_region_name = <Your Région 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
```

Créer le cluster

```
$ pcluster create multi-queue-special -t multi-queue-special
```

Une fois le cluster créé, utilisez votre fichier de clé SSH privé pour vous connecter à votre nœud principal.

```
$ pcluster ssh multi-queue-special -i ~/path/to/keyfile.pem
```

Il doit s'agir de l'état initial du 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]
```

Cette section décrit comment soumettre certaines tâches pour vérifier que les nœuds disposent de ressources EFA ou GPU.

Rédigez d'abord les scripts de travail. `efa_job.sh`dormira pendant 30 secondes. Ensuite, recherchez EFA dans la sortie de la `lspci` commande. `gpu_job.sh`dormira pendant 30 secondes. Ensuite, exécutez `nvidia-smi` pour afficher les informations du GPU sur le nœud.

```
$ 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
```

Soumettez l'offre d'emploi avec`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]
```

Au bout de quelques minutes, vous devriez voir les nœuds en ligne et les tâches en cours d'exécution.

```
[ec2-user@ip-172-31-15-251 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
efa*         up   infinite      4  idle~ efa-dy-c5n18xlarge-[2-5]
efa*         up   infinite      1    mix efa-dy-c5n18xlarge-1
gpu          up   infinite      9  idle~ gpu-dy-g38xlarge-[2-10]
gpu          up   infinite      1    mix gpu-dy-g38xlarge-1
[ec2-user@ip-172-31-15-251 ~]$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 4       gpu     wrap ec2-user  R       0:06      1 gpu-dy-g38xlarge-1
                 5       efa     wrap ec2-user  R       0:01      1 efa-dy-
c5n18xlarge-1
```

Une fois le travail terminé, vérifiez le résultat. À partir de la sortie du `slurm-2.out` fichier, vous pouvez voir que l'EFA est présent sur le `efa-dy-c5n18xlarge-1` nœud. À partir de la sortie du `slurm-3.out` fichier, vous pouvez voir que la `nvidia-smi` sortie contient des informations GPU pour le `gpu-dy-g38xlarge-1` nœud.

```
$ 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                                                 |
+-----------------------------------------------------------------------------+
```

Vous pouvez observer le processus de réduction. Dans la configuration du cluster, vous avez précédemment spécifié une durée `scaledown_idletime` maximale de cinq minutes. Par conséquent, après cinq minutes d'inactivité, vos nœuds dynamiques diminuent automatiquement et passent en `POWER_DOWN` mode. `spot-dy-c5xlarge-1` `spot-dy-t2micro-1` Finalement, les nœuds passent en mode économie d'énergie et peuvent être réutilisés.

```
$ 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]
```

Après vous être déconnecté du cluster, vous pouvez le nettoyer en exécutant`pcluster delete <cluster name>`.

```
$ pcluster list
multi-queue-special CREATE_COMPLETE 2.11.9
$ pcluster delete multi-queue-special
Deleting: multi-queue-special
...
```

Pour de plus amples informations, veuillez consulter Slurmguide pour le mode de file d'attente multiple.

Développement

Vous pouvez utiliser les sections suivantes pour démarrer le développement de AWS ParallelCluster.

> **Important**
>
> Les sections suivantes incluent les instructions pour l'utilisation d'une version personnalisée du livre de recettes et d'un package de nœuds AWS ParallelCluster personnalisé. Ces informations concernent une méthode avancée de personnalisation AWS ParallelCluster, avec des problèmes potentiels qui peuvent être difficiles à résoudre. L' AWS ParallelCluster équipe recommande vivement d'utiliser les scripts des actions Bootstrap personnalisées à des fins de personnalisation, car les hooks post-installation sont généralement plus faciles à déboguer et plus portables d'une version à l'autre de. AWS ParallelCluster

Rubriques

- Configuration d'un livre de AWS ParallelCluster recettes personnalisé
- Configuration d'un package de AWS ParallelCluster nœuds personnalisé

Configuration d'un livre de AWS ParallelCluster recettes personnalisé

> **Important**
>
> Vous trouverez ci-dessous des instructions pour utiliser une version personnalisée des recettes du AWS ParallelCluster livre de recettes. Il s'agit d'une méthode avancée de personnalisation AWS ParallelCluster, avec des problèmes potentiels qui peuvent être difficiles à corriger. L' AWS ParallelCluster équipe recommande vivement d'utiliser les scripts des actions Bootstrap personnalisées à des fins de personnalisation, car les hooks post-installation sont généralement plus faciles à déboguer et plus portables d'une version à l'autre de. AWS ParallelCluster

Étapes

1. Identifiez le répertoire de travail du AWS ParallelCluster livre de recettes dans lequel vous avez cloné le code du livre de AWS ParallelCluster recettes.

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

2. Détectez la version actuelle du AWS ParallelCluster livre de recettes.

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

3. Créez une archive du AWS ParallelCluster Cookbook et calculez son md5.

```
cd "${_cookbookDir}"
_stashName=$(git stash create)
git archive --format tar --prefix="aws-parallelcluster-cookbook-${_version}/"
 "${_stashName:-HEAD}" | gzip > "aws-parallelcluster-cookbook-${_version}.tgz"
md5sum "aws-parallelcluster-cookbook-${_version}.tgz" > "aws-parallelcluster-
cookbook-${_version}.md5"
```

4. Créez un compartiment Amazon S3 et chargez l'archive, son md5 et sa date de dernière modification dans le compartiment. Accordez une autorisation en lecture au public via une liste de contrôle d'accès (ACL) définie sur public-read.

```
_bucket=<the bucket name>
aws s3 cp --acl public-read aws-parallelcluster-cookbook-${_version}.tgz s3://
${_bucket}/cookbooks/aws-parallelcluster-cookbook-${_version}.tgz
aws s3 cp --acl public-read aws-parallelcluster-cookbook-${_version}.md5 s3://
${_bucket}/cookbooks/aws-parallelcluster-cookbook-${_version}.md5
aws s3api head-object --bucket ${_bucket} --key cookbooks/aws-parallelcluster-
cookbook-${_version}.tgz --output text --query LastModified > aws-parallelcluster-
cookbook-${_version}.tgz.date
aws s3 cp --acl public-read aws-parallelcluster-cookbook-${_version}.tgz.date s3://
${_bucket}/cookbooks/aws-parallelcluster-cookbook-${_version}.tgz.date
```

5. Ajoutez les variables suivantes au fichier AWS ParallelCluster de configuration, sous la `[cluster]`section.

```
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**
>
> À partir de AWS ParallelCluster la version 2.6.1, la plupart des recettes d'installation sont ignorées par défaut lors du lancement des nœuds afin d'améliorer les temps de démarrage. Pour ignorer la plupart des recettes d'installation afin de réduire les temps de démarrage au détriment de la rétrocompatibilité, `"skip_install_recipes" : "no"` supprimez la `cluster` clé du `extra_json` paramètre.

Configuration d'un package de AWS ParallelCluster nœuds personnalisé

> **Warning**
>
> Vous trouverez ci-dessous les instructions relatives à l'utilisation d'une version personnalisée du package de AWS ParallelCluster nœuds. Il s'agit d'une méthode avancée de personnalisation AWS ParallelCluster, avec des problèmes potentiels qui peuvent être difficiles à corriger. L' AWS ParallelCluster équipe recommande vivement d'utiliser les scripts des actions Bootstrap personnalisées à des fins de personnalisation, car les hooks post-installation sont généralement plus faciles à déboguer et plus portables d'une version à l'autre de. AWS ParallelCluster

Étapes

1. Identifiez le répertoire de travail du AWS ParallelCluster nœud dans lequel vous avez cloné le code du AWS ParallelCluster nœud.

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

2. Détecte la version actuelle du AWS ParallelCluster nœud.

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

3. Créez une archive du AWS ParallelCluster nœud.

```
cd "${_nodeDir}"
```

```
_stashName=$(git stash create)
git archive --format tar --prefix="aws-parallelcluster-node-${_version}/"
 "${_stashName:-HEAD}" | gzip > "aws-parallelcluster-node-${_version}.tgz"
```

4. Créez un compartiment Amazon S3 et chargez l'archive dans le compartiment. Accordez une autorisation en lecture au public via une liste de contrôle d'accès (ACL) définie sur public-read.

```
_bucket=<the bucket name>
aws s3 cp --acl public-read aws-parallelcluster-node-${_version}.tgz s3://${_bucket}/
node/aws-parallelcluster-node-${_version}.tgz
```

5. Ajoutez la variable suivante au fichier AWS ParallelCluster de configuration, sous la `[cluster]`section.

```
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**
>
> À partir de AWS ParallelCluster la version 2.6.1, la plupart des recettes d'installation sont ignorées par défaut lors du lancement des nœuds afin d'améliorer les temps de démarrage. Pour ignorer la plupart des recettes d'installation afin de réduire les temps de démarrage au détriment de la rétrocompatibilité, `"skip_install_recipes" : "no"` supprimez la `cluster` clé du `extra_json` paramètre.

AWS ParallelCluster résolution des problèmes

La AWS ParallelCluster communauté gère une page Wiki qui fournit de nombreux conseils de résolution des problèmes sur le AWS ParallelCluster GitHub Wiki. Pour obtenir la liste des problèmes connus, consultez la section Problèmes connus.

Rubriques

- Récupération et conservation des journaux
- Résolution des problèmes de déploiement des piles
- Résolution des problèmes dans plusieurs clusters en mode file d'attente
- Résolution des problèmes dans les clusters en mode file d'attente unique
- Problèmes liés aux groupes de placement et au lancement d'instances
- Répertoires qui ne peuvent pas être remplacés
- Résolution des problèmes dans Amazon DCV
- Résolution des problèmes dans les clusters avec AWS Batch intégration
- Résolution des problèmes en cas d'échec de création d'une ressource
- Résolution des problèmes liés à la taille des politiques IAM
- Support supplémentaire

Récupération et conservation des journaux

Les journaux constituent une ressource utile pour résoudre les problèmes. Avant de pouvoir utiliser les journaux pour résoudre les problèmes liés à vos AWS ParallelCluster ressources, vous devez d'abord créer une archive des journaux du cluster. Suivez les étapes décrites dans la rubrique Création d'une archive des journaux d'un cluster sur le AWS ParallelCluster GitHub Wiki pour démarrer ce processus.

Si l'un de vos clusters en cours d'exécution rencontre des problèmes, vous devez le placer dans un `STOPPED` état en exécutant la `pcluster stop <cluster_name>` commande avant de commencer le dépannage. Cela évite d'encourir des coûts imprévus.

S'il `pcluster` cesse de fonctionner ou si vous souhaitez supprimer un cluster tout en préservant ses journaux, exécutez la `pcluster delete –keep-logs <cluster_name>` commande. L'exécution de cette commande supprime le cluster tout en conservant le groupe de journaux stocké

sur Amazon CloudWatch. Pour plus d'informations sur cette commande, consultez la `pcluster delete` documentation.

Résolution des problèmes de déploiement des piles

Si votre cluster ne parvient pas à être créé et annule la création de la pile, vous pouvez consulter les fichiers journaux suivants pour diagnostiquer le problème. Vous souhaitez rechercher le résultat de `ROLLBACK_IN_PROGRESS` dans ces journaux. Le message d'échec doit se présenter comme suit :

```
$ 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
```

Pour diagnostiquer le problème, créez à nouveau le cluster en utilisant`pcluster create`, y compris le `--norollback` drapeau. Ensuite, connectez-vous au cluster en SSH :

```
$ pcluster create mycluster --norollback
...
$ pcluster ssh mycluster
```

Une fois connecté au nœud principal, vous devriez trouver trois fichiers journaux principaux que vous pouvez utiliser pour identifier l'erreur.

- `/var/log/cfn-init.log`est le journal du `cfn-init` script. Vérifiez d'abord ce journal. Vous êtes susceptible de voir une erreur comme `Command chef failed` dans ce journal. Regardez les lignes juste avant cette ligne pour plus de détails liés au message d'erreur. Pour plus d'informations, consultez cfn-init.
- `/var/log/cloud-init.log`est le journal de cloud-init. Si rien ne s'y trouve`cfn-init.log`, essayez ensuite de consulter ce journal.
- `/var/log/cloud-init-output.log`est le résultat des commandes exécutées par cloud-init. Cela inclut la sortie de`cfn-init`. Dans la plupart des cas, il n'est pas nécessaire de consulter ce journal pour résoudre ce type de problème.

Résolution des problèmes dans plusieurs clusters en mode file d'attente

Cette section concerne les clusters installés à l'aide de AWS ParallelCluster la version 2.9.0 ou ultérieure avec Slurm planificateur de tâches. Pour plus d'informations sur le mode de file d'attente multiple, consultezMode de file d'attente multiple.

Rubriques

- Journaux clés
- Résolution des problèmes d'initialisation des nœuds
- Résolution des problèmes de remplacement et de terminaison inattendus de nœuds
- Remplacement, arrêt ou mise hors tension des instances et des nœuds problématiques
- Résolution d'autres problèmes connus liés aux nœuds et aux tâches

Journaux clés

Le tableau suivant fournit une vue d'ensemble des journaux clés du nœud principal :

`/var/log/cfn-init.log`

Il s'agit du journal AWS CloudFormation d'initialisation. Il contient toutes les commandes exécutées lors de la configuration d'une instance. C'est utile pour résoudre les problèmes d'initialisation.

`/var/log/chef-client.log`

Il s'agit du journal du client Chef. Il contient toutes les commandes qui ont été exécutées via CHEF/CINC. C'est utile pour résoudre les problèmes d'initialisation.

`/var/log/parallelcluster/slurm_resume.log`

C'est un `ResumeProgram` journal. Il lance des instances pour les nœuds dynamiques et est utile pour résoudre les problèmes de lancement des nœuds dynamiques.

`/var/log/parallelcluster/slurm_suspend.log`

Voici le `SuspendProgram` journal. Il est appelé lorsque des instances sont résiliées pour des nœuds dynamiques et est utile pour résoudre les problèmes de terminaison des nœuds

dynamiques. Lorsque vous consultez ce journal, vous devez également le `clustermgtd` consulter.

`/var/log/parallelcluster/clustermgtd`

Voici le `clustermgtd` journal. Il s'exécute en tant que démon centralisé qui gère la plupart des actions des opérations de cluster. C'est utile pour résoudre tout problème de lancement, de terminaison ou de fonctionnement du cluster.

`/var/log/slurmctld.log`

C'est le Slurm journal du démon de contrôle. AWS ParallelCluster ne prend pas de décisions de mise à l'échelle. Au contraire, il tente uniquement de lancer des ressources pour satisfaire aux Slurm exigences. Il est utile pour les problèmes de dimensionnement et d'allocation, les problèmes liés au travail et les problèmes de lancement et de résiliation liés au planificateur.

Voici les remarques clés concernant les nœuds de calcul :

`/var/log/cloud-init-output.log`

Il s'agit du journal cloud-init. Il contient toutes les commandes exécutées lors de la configuration d'une instance. C'est utile pour résoudre les problèmes d'initialisation.

`/var/log/parallelcluster/computemgtd`

Voici le `computemgtd` journal. Il s'exécute sur chaque nœud de calcul pour surveiller le nœud dans les rares cas où le `clustermgtd` démon du nœud principal est hors ligne. C'est utile pour résoudre les problèmes de résiliation inattendus.

`/var/log/slurmd.log`

C'est le Slurm journal des démons de calcul. C'est utile pour résoudre les problèmes liés à l'initialisation et aux pannes de calcul.

Résolution des problèmes d'initialisation des nœuds

Cette section explique comment résoudre les problèmes d'initialisation des nœuds. Cela inclut les problèmes où le nœud ne parvient pas à démarrer, à démarrer ou à rejoindre un cluster.

Nœud principal :

Journaux applicables :

- `/var/log/cfn-init.log`
- `/var/log/chef-client.log`
- `/var/log/parallelcluster/clustermgtd`
- `/var/log/parallelcluster/slurm_resume.log`
- `/var/log/slurmctld.log`

Vérifiez les `/var/log/chef-client.log` journaux `/var/log/cfn-init.log` et. Ces journaux doivent contenir toutes les actions exécutées lors de la configuration du nœud principal. La plupart des erreurs survenant au cours de l'installation doivent contenir un message d'erreur dans le `/var/log/chef-client.log` journal. Si des scripts de pré-installation ou de post-installation sont spécifiés dans la configuration du cluster, vérifiez que le script s'exécute correctement via les messages du journal.

Lorsqu'un cluster est créé, le nœud principal doit attendre que les nœuds de calcul rejoignent le cluster avant de pouvoir le rejoindre. Ainsi, si les nœuds de calcul ne parviennent pas à rejoindre le cluster, le nœud principal échoue également. Vous pouvez suivre l'un de ces ensembles de procédures, en fonction du type de notes de calcul que vous utilisez, pour résoudre ce type de problème :

Nœuds de calcul dynamiques :

- Recherchez dans le `ResumeProgram` journal (`/var/log/parallelcluster/slurm_resume.log`) le nom de votre nœud de calcul pour voir s'il `ResumeProgram` a déjà été appelé avec le nœud. (Si vous `ResumeProgram` n'avez jamais été appelé, vous pouvez consulter le `slurmctld` log (`/var/log/slurmctld.log`) pour déterminer si Slurm J'ai déjà essayé d'appeler `ResumeProgram` avec le nœud.)
- Notez que des autorisations incorrectes pour `ResumeProgram` peuvent `ResumeProgram` entraîner un échec silencieux. Si vous utilisez une AMI personnalisée dont la `ResumeProgram` configuration a été modifiée, vérifiez qu'elle `ResumeProgram` appartient à l'`slurm`utilisateur et qu'elle dispose de l'autorisation `744` (`rwxr--r--`).
- En cas `ResumeProgram` d'appel, vérifiez si une instance est lancée pour le nœud. Si aucune instance n'a été lancée, un message d'erreur décrivant l'échec du lancement devrait s'afficher.
- Si l'instance est lancée, il se peut qu'un problème se soit produit pendant le processus de configuration. Vous devriez voir l'adresse IP privée et l'ID d'instance correspondants dans le `ResumeProgram` journal. De plus, vous pouvez consulter les journaux de configuration

correspondants pour l'instance en question. Pour plus d'informations sur le dépannage d'une erreur de configuration avec un nœud de calcul, consultez la section suivante.

Nœuds de calcul statiques :

- Consultez le journal `clustermgtd` (`/var/log/parallelcluster/clustermgtd`) pour voir si des instances ont été lancées pour le nœud. S'ils n'ont pas été lancés, un message d'erreur clair devrait s'afficher détaillant l'échec du lancement.
- Si l'instance est lancée, il y a un problème pendant le processus de configuration. Vous devriez voir l'adresse IP privée et l'ID d'instance correspondants dans le `ResumeProgram` journal. De plus, vous pouvez consulter les journaux de configuration correspondants pour l'instance en question.

- Nœuds de calcul :
  - Journaux applicables :
    - `/var/log/cloud-init-output.log`
    - `/var/log/slurmd.log`
  - Si le nœud de calcul est lancé`/var/log/cloud-init-output.log`, vérifiez d'abord, qui doit contenir les journaux de configuration similaires à ceux du nœud principal. `/var/log/chef-client.log` La plupart des erreurs survenant lors de l'installation doivent contenir des messages d'erreur dans le `/var/log/cloud-init-output.log` journal. Si des scripts de pré-installation ou de post-installation sont spécifiés dans la configuration du cluster, vérifiez qu'ils ont été exécutés correctement.
  - Si vous utilisez une AMI personnalisée avec modification de Slurm configuration, alors il pourrait y avoir un Slurm erreur associée qui empêche le nœud de calcul de rejoindre le cluster. Pour les erreurs liées au planificateur, consultez le `/var/log/slurmd.log` journal.

Résolution des problèmes de remplacement et de terminaison inattendus de nœuds

Cette section continue à explorer comment résoudre les problèmes liés aux nœuds, en particulier lorsqu'un nœud est remplacé ou arrêté de manière inattendue.

- Journaux applicables :
  - `/var/log/parallelcluster/clustermgtd`(nœud principal)
  - `/var/log/slurmctld.log`(nœud principal)

  - `/var/log/parallelcluster/computemgtd`(nœud de calcul)
- Nœuds remplacés ou interrompus de façon inattendue
  - Consultez le `clustermgtd` journal (`/var/log/parallelcluster/clustermgtd`) pour voir si l'action de remplacement ou de résiliation d'un nœud `clustermgtd` a été entreprise. Notez que cela `clustermgtd` gère toutes les actions de maintenance normales du nœud.
  - En cas de `clustermgtd` remplacement ou de résiliation du nœud, un message doit s'afficher expliquant pourquoi cette action a été entreprise sur le nœud. Si la raison est liée au planificateur (par exemple, parce que le nœud est dedans`DOWN`), consultez le `slurmctld` journal pour plus d'informations. Si la raison est EC2 liée à Amazon, il doit y avoir un message informatif détaillant le problème EC2 lié à Amazon qui a nécessité le remplacement.
  - Si cela `clustermgtd` n'a pas mis fin au nœud, vérifiez d'abord s'il s'agit d'une résiliation prévue par Amazon EC2, plus précisément d'une résiliation ponctuelle. `computemgtd`, exécuté sur un nœud de calcul, peut également prendre des mesures pour mettre fin à un nœud s'il `clustermgtd` est déterminé qu'il ne fonctionne pas correctement. Vérifiez `computemgtd` log (`/var/log/parallelcluster/computemgtd`) pour voir si le `computemgtd` nœud a été arrêté.
- Les nœuds ont échoué
  - Consultez `slurmctld` log (`/var/log/slurmctld.log`) pour savoir pourquoi une tâche ou un nœud a échoué. Notez que les tâches sont automatiquement mises en file d'attente en cas de défaillance d'un nœud.
  - Si vous `slurm_resume` signalez que le nœud est lancé et que vous `clustermgtd` signalez au bout de quelques minutes qu'il n'existe aucune instance correspondante dans Amazon EC2 pour ce nœud, le nœud risque de tomber en panne lors de la configuration. Pour récupérer le journal depuis un compute (`/var/log/cloud-init-output.log`), procédez comme suit :
    - Soumettre une offre d'emploi à louer Slurm créez un nouveau nœud.
    - Après le démarrage du nœud, activez la protection de terminaison à l'aide de cette commande.

      ```
      aws ec2 modify-instance-attribute --instance-id i-xyz --disable-api-termination
      ```

    - Récupérez la sortie de console depuis le nœud à l'aide de cette commande.

      ```
      aws ec2 get-console-output --instance-id i-xyz --output text
      ```

Remplacement, arrêt ou mise hors tension des instances et des nœuds problématiques

- Journaux applicables :
  - `/var/log/parallelcluster/clustermgtd`(nœud principal)
  - `/var/log/parallelcluster/slurm_suspend.log`(nœud principal)
- Dans la plupart des cas, `clustermgtd` gère toutes les actions de fermeture d'instance attendues. Consultez le `clustermgtd` journal pour savoir pourquoi il n'a pas réussi à remplacer ou à mettre fin à un nœud.
- En cas d'échec d'un nœud dynamique`scaledown_idletime`, consultez le `SuspendProgram` journal pour voir s'`SuspendProgram`il a été appelé `slurmctld` avec le nœud spécifique comme argument. Notez qu'`SuspendProgram`aucune action n'est réellement exécutée. Au contraire, il ne se connecte que lorsqu'il est appelé. La résiliation et la `NodeAddr` réinitialisation de toutes les instances sont effectuées par`clustermgtd`. Slurm remet `SuspendTimeout` automatiquement les nœuds dans un `POWER_SAVING` état ultérieur.

Résolution d'autres problèmes connus liés aux nœuds et aux tâches

Un autre type de problème connu est celui qui AWS ParallelCluster peut ne pas permettre d'allouer des tâches ou de prendre des décisions de dimensionnement. Dans ce type de problème, AWS ParallelCluster seuls le lancement, la résiliation ou la maintenance des ressources conformément à Slurm instructions. Pour ces problèmes, consultez le `slurmctld` journal pour les résoudre.

Résolution des problèmes dans les clusters en mode file d'attente unique

> **Note**
>
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Cette section s'applique aux clusters qui ne disposent pas du mode de file d'attente multiple avec l'une des deux configurations suivantes :

- Lancé à l'aide d'une AWS ParallelCluster version antérieure à 2.9.0 et SGE, Torque, ou Slurm planificateurs de tâches.
- Lancé avec AWS ParallelCluster la version 2.9.0 ou ultérieure et SGE or Torque planificateurs de tâches.

Rubriques

- Journaux clés
- Résolution des problèmes d'échec des opérations de lancement et de jointure
- Résolution des problèmes de dimensionnement
- Résolution d'autres problèmes liés au cluster

Journaux clés

Les fichiers journaux suivants sont les journaux clés du nœud principal.

Pour AWS ParallelCluster la version 2.9.0 ou ultérieure :

`/var/log/chef-client.log`

Il s'agit du journal du client CINC (chef). Il contient toutes les commandes exécutées via CINC. C'est utile pour résoudre les problèmes d'initialisation.

Pour toutes les AWS ParallelCluster versions :

`/var/log/cfn-init.log`

Voici le `cfn-init` journal. Il contient toutes les commandes qui ont été exécutées lors de la configuration d'une instance et est donc utile pour résoudre les problèmes d'initialisation. Pour plus d'informations, consultez cfn-init.

`/var/log/clustermgtd.log`

Il s'agit du `clustermgtd` journal de Slurm planificateurs. `clustermgtd`fonctionne en tant que démon centralisé qui gère la plupart des actions des opérations de cluster. C'est utile pour résoudre tout problème de lancement, de terminaison ou de fonctionnement du cluster.

`/var/log/jobwatcher`

Il s'agit du `jobwatcher` journal de SGE and Torque planificateurs. `jobwatcher`surveille la file d'attente du planificateur et met à jour le groupe Auto Scaling. C'est utile pour résoudre les problèmes liés à la mise à l'échelle des nœuds.

`/var/log/sqswatcher`

Il s'agit du `sqswatcher` journal de SGE and Torque planificateurs. `sqswatcher`traite l'événement Instance Ready envoyé par une instance de calcul après une initialisation réussie. Il ajoute également des nœuds de calcul à la configuration du planificateur. Ce journal est utile pour résoudre les problèmes liés à l'échec d'un ou de plusieurs nœuds à rejoindre un cluster.

Les journaux clés des nœuds de calcul sont les suivants.

AWS ParallelCluster version 2.9.0 ou ultérieure

`/var/log/cloud-init-output.log`

Il s'agit du journal d'initialisation du Cloud. Il contient toutes les commandes exécutées lors de la configuration d'une instance. C'est utile pour résoudre les problèmes d'initialisation.

AWS ParallelCluster versions antérieures à 2.9.0

`/var/log/cfn-init.log`

Il s'agit du journal CloudFormation d'initialisation. Il contient toutes les commandes exécutées lors de la configuration d'une instance. C'est utile pour résoudre les problèmes d'initialisation

Toutes les versions

`/var/log/nodewatcher`

Voici le `nodewatcher` journal. `nodewatcher`démons qui s'exécutent sur chaque nœud de calcul lors de l'utilisation SGE and Torque planificateurs. Ils réduisent la taille d'un nœud s'il est inactif. Ce journal est utile pour tout problème lié à la réduction des ressources.

Résolution des problèmes d'échec des opérations de lancement et de jointure

- Journaux applicables :
  - `/var/log/cfn-init-cmd.log`(nœud principal et nœud de calcul)
  - `/var/log/sqswatcher`(nœud principal)
- Si les nœuds n'ont pas pu être lancés, consultez le `/var/log/cfn-init-cmd.log` journal pour voir le message d'erreur spécifique. Dans la plupart des cas, les échecs de lancement des nœuds sont dus à un échec de configuration.
- Si les nœuds de calcul n'ont pas réussi à rejoindre la configuration du planificateur malgré une configuration réussie, consultez le `/var/log/sqswatcher` journal pour voir si l'événement a `sqswatcher` été traité. Dans la plupart des cas, ces problèmes sont dus au fait que l'événement `sqswatcher` n'a pas été traité.

Résolution des problèmes de dimensionnement

- Journaux applicables :
  - `/var/log/jobwatcher`(nœud principal)
  - `/var/log/nodewatcher`(nœud de calcul)
- Problèmes de mise à l'échelle : pour le nœud principal, consultez le `/var/log/jobwatcher` journal pour voir si le `jobwatcher` daemon a calculé le nombre approprié de nœuds requis et a mis à jour le groupe Auto Scaling. Notez qu'il `jobwatcher` surveille la file d'attente du planificateur et met à jour le groupe Auto Scaling.
- Problèmes de réduction : pour les nœuds de calcul, consultez le `/var/log/nodewatcher` journal du nœud problématique pour savoir pourquoi le nœud a été réduit. Notez que les `nodewatcher` démons réduisent la taille d'un nœud de calcul s'il est inactif.

Résolution d'autres problèmes liés au cluster

L'un des problèmes connus est l'échec des notes de calcul aléatoires sur les clusters à grande échelle, en particulier ceux comportant 500 nœuds de calcul ou plus. Ce problème est lié à une limitation de l'architecture de dimensionnement d'un cluster de files d'attente unique. Si vous souhaitez utiliser un cluster à grande échelle, si vous utilisez AWS ParallelCluster la version v2.9.0 ou ultérieure, utilisez Slurm, et pour éviter ce problème, vous devez effectuer une mise à niveau et

passer à un cluster compatible avec le mode de files d'attente multiples. Vous pouvez le faire en courant`pcluster-config convert`.

Pour les ultra-large-scale clusters, un réglage supplémentaire de votre système peut être nécessaire. Pour plus d'informations, contactez Support.

Problèmes liés aux groupes de placement et au lancement d'instances

Pour obtenir le plus faible temps de latence entre nœuds, utilisez un groupe de placement. Un groupe de placement garantit que vos instances sont situées sur la même structure de mise en réseau. S'il n'y a pas suffisamment d'instances disponibles lorsqu'une demande est faite, une `InsufficientInstanceCapacity` erreur est renvoyée. Pour réduire le risque de recevoir cette erreur lors de l'utilisation de groupes de placement de clusters, définissez le `placement_group` paramètre sur `DYNAMIC` et définissez le `placement` paramètre sur`compute`.

Si vous avez besoin d'un système de fichiers partagé performant, pensez à l'utiliser FSx pour Lustre.

Si le nœud principal doit se trouver dans le groupe de placement, utilisez le même type d'instance et le même sous-réseau pour le nœud principal et pour tous les nœuds de calcul. Ce faisant, le `compute_instance_type` paramètre a la même valeur que le `master_instance_type` paramètre, le `placement` paramètre est défini sur et le `compute_subnet_id` paramètre n'est pas spécifié. `cluster` Avec cette configuration, la valeur du `master_subnet_id` paramètre est utilisée pour les nœuds de calcul.

Pour plus d'informations, consultez les sections Résolution des problèmes de lancement d'instance et Groupes de placement, rôles et limites dans le Guide de EC2 l'utilisateur Amazon.

Répertoires qui ne peuvent pas être remplacés

Les répertoires suivants sont partagés entre les nœuds et ne peuvent pas être remplacés.

`/home`

Cela inclut le dossier personnel par défaut de l'utilisateur (`/home/ec2_user`sur Amazon Linux, `/home/centos` sur CentOS, et ainsi de `/home/ubuntu` suite Ubuntu).

`/opt/intel`

Ce répertoire inclut les fichiers Intel MPI, Intel Parallel Studio et les fichiers associés.

`/opt/sge`

> **Note**
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Cela inclut Son of Grid Engine et les fichiers associés. (Conditionnel, seulement si `scheduler = sge`.)

`/opt/slurm`

Cela inclut Slurm Workload Manager et les fichiers associés. (Conditionnel, seulement si `scheduler = slurm`.)

`/opt/torque`

> **Note**
> À partir de la version 2.11.5, AWS ParallelCluster ne prend pas en charge l'utilisation de SGE or Torque planificateurs.

Cela inclut Torque Resource Manager et les fichiers associés. (Conditionnel, seulement si `scheduler = torque`.)

Résolution des problèmes dans Amazon DCV

Rubriques

- Journaux pour Amazon DCV
- Mémoire de type d'instance Amazon DCV
- Problèmes liés à Ubuntu Amazon DCV

Journaux pour Amazon DCV

Les journaux d'Amazon DCV sont écrits dans des fichiers du `/var/log/dcv/` répertoire. L'examen de ces journaux peut aider à résoudre les problèmes.

Mémoire de type d'instance Amazon DCV

Le type d'instance doit disposer d'au moins 1,7 Gibioctet (GiB) de RAM pour exécuter Amazon DCV. Nano and micro les types d'instance ne disposent pas de suffisamment de mémoire pour exécuter Amazon DCV.

Problèmes liés à Ubuntu Amazon DCV

Lorsque vous exécutez Gnome Terminal sur une session DCV sur Ubuntu, il se peut que vous n'ayez pas automatiquement accès à l'environnement utilisateur mis à disposition AWS ParallelCluster via le shell de connexion. L'environnement utilisateur fournit des modules d'environnement tels que openmpi ou intelmpi, ainsi que d'autres paramètres utilisateur.

Les paramètres par défaut de Gnome Terminal empêchent le shell de démarrer en tant que shell de connexion. Cela signifie que les profils shell ne sont pas générés automatiquement et que l'environnement AWS ParallelCluster utilisateur n'est pas chargé.

Pour obtenir correctement le profil shell et accéder à l'environnement AWS ParallelCluster utilisateur, effectuez l'une des opérations suivantes :

- Modifiez les paramètres par défaut du terminal :
  1. Choisissez le menu Edition dans le terminal Gnome.
  2. Sélectionnez Préférences, puis Profils.
  3. Choisissez Command, puis sélectionnez Exécuter la commande en tant que shell de connexion.
  4. Ouvrez un nouveau terminal.
- Utilisez la ligne de commande pour rechercher les profils disponibles :

```
$ source /etc/profile && source $HOME/.bashrc
```

Résolution des problèmes dans les clusters avec AWS Batch intégration

Cette section concerne les clusters avec intégration d'un AWS Batch planificateur.

Problèmes liés au nœud principal

Les problèmes de configuration liés au nœud principal peuvent être résolus de la même manière qu'un cluster de files d'attente unique. Pour de plus amples informations sur ces problèmes, veuillez consulter Résolution des problèmes dans les clusters en mode file d'attente unique.

AWS Batch problèmes de soumission de tâches parallèles sur plusieurs nœuds

Si vous rencontrez des problèmes pour soumettre des tâches parallèles sur plusieurs nœuds lorsque vous les utilisez AWS Batch comme planificateur de tâches, vous devez passer à AWS ParallelCluster la version 2.5.0. Si cela n'est pas faisable, vous pouvez utiliser la solution décrite dans la rubrique : Appliquer un correctif automatique à un cluster utilisé pour soumettre des tâches parallèles sur plusieurs nœuds via. AWS Batch

Problèmes de calcul

AWS Batch gère les aspects de dimensionnement et de calcul de vos services. Si vous rencontrez des problèmes liés au calcul, consultez la documentation de AWS Batch dépannage pour obtenir de l'aide.

Échecs au travail

Si une tâche échoue, vous pouvez exécuter la `awsbout` commande pour récupérer le résultat de la tâche. Vous pouvez également exécuter la `awsbstat -d` commande pour obtenir un lien vers les journaux des tâches stockés par Amazon CloudWatch.

Résolution des problèmes en cas d'échec de création d'une ressource

Cette section concerne les ressources du cluster lorsque leur création échoue.

Lorsqu'une ressource ne parvient pas à être créée, ParallelCluster renvoie un message d'erreur comme celui-ci.

```
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)
}
```

Par exemple, si le message d'état s'affiche dans la réponse de commande précédente, vous devez utiliser des types d'instance qui ne dépasseront pas votre limite de vCPU actuelle ou qui ne demanderont pas une capacité de vCPU accrue.

Vous pouvez également utiliser la CloudFormation console pour consulter les informations relatives à l'`"Cluster creation failed"`état.

Afficher les messages CloudFormation d'erreur depuis la console.

1. Connectez-vous au AWS Management Console et naviguez vers https://console.aws.amazon.com/cloudformation.
2. Sélectionnez la pile nommée parallelcluster-. *`cluster_name`*
3. Choisissez l'onglet Événements.

4. Vérifiez l'état de la ressource dont la création a échoué en faisant défiler la liste des événements de ressource par ID logique. Si la création d'une sous-tâche a échoué, revenez en arrière pour trouver l'événement de ressource ayant échoué.
5. Exemple de message d' AWS CloudFormation erreur :

```
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].
```

Résolution des problèmes liés à la taille des politiques IAM

Reportez-vous à l'IAM et aux AWS STS quotas, aux exigences relatives aux noms et aux limites de caractères pour vérifier les quotas sur les politiques gérées associées aux rôles. Si la taille d'une politique gérée dépasse le quota, divisez-la en deux politiques ou plus. Si vous dépassez le quota du nombre de politiques associées à un rôle IAM, créez des rôles supplémentaires et répartissez les politiques entre eux pour atteindre le quota.

Support supplémentaire

Pour une liste des problèmes connus, consultez la page principale du GitHubWiki ou la page des problèmes. Pour les problèmes plus urgents, contactez Support ou ouvrez un nouveau GitHub numéro.

AWS ParallelCluster politique de support

AWS ParallelCluster prend en charge plusieurs versions en même temps. Une date de fin de vie de support (EOSL) est prévue pour chaque AWS ParallelCluster version. Après la date EOSL, aucun support ou maintenance supplémentaire n'est fourni pour cette version.

AWS ParallelCluster utilise un schéma de `major.minor.`*`patch`* version. Les nouvelles fonctionnalités, les améliorations des performances, les mises à jour de sécurité et les corrections de bogues sont incluses dans les nouvelles versions mineures de la dernière version majeure. Les versions mineures sont rétrocompatibles au sein d'une version majeure. Pour les problèmes critiques, AWS fournit des correctifs par le biais de mises à jour, mais uniquement pour les dernières versions mineures des versions qui n'ont pas atteint l'EOSL. Si vous souhaitez utiliser les mises à jour d'une nouvelle version, vous devez effectuer une mise à niveau vers la nouvelle version mineure ou corrective.

AWS ParallelCluster versions	Date de fin de vie prise en charge (EOSL)
2.10.4 et versions antérieures	31/12/2021
2,11. *x*	31 décembre 2022

Sécurité dans AWS ParallelCluster

La sécurité du cloud AWS est la priorité absolue. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des entreprises les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre vous AWS et vous. Le modèle de responsabilité partagée décrit ceci comme la sécurité du cloud et la sécurité dans le cloud :

- Sécurité du cloud : AWS est chargée de protéger l'infrastructure qui exécute les AWS services dans le AWS cloud. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Des auditeurs tiers testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre des programmes de AWS conformité Programmes de de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent à AWS ParallelCluster, voir AWS Services concernés par programme de conformitéAWS .
- Sécurité dans le cloud — Votre responsabilité est déterminée par le ou les AWS services spécifiques que vous utilisez. Vous êtes également responsable de plusieurs autres facteurs connexes, notamment la sensibilité de vos données, les exigences de votre entreprise et les lois et réglementations applicables.

Cette documentation décrit comment appliquer le modèle de responsabilité partagée lors de son utilisation AWS ParallelCluster. Les rubriques suivantes expliquent comment procéder à la configuration AWS ParallelCluster pour atteindre vos objectifs de sécurité et de conformité. Vous apprenez également à les utiliser AWS ParallelCluster de manière à surveiller et à sécuriser vos AWS ressources.

Rubriques

- Informations de sécurité pour les services utilisés par AWS ParallelCluster
- Protection des données dans AWS ParallelCluster
- Identity and Access Management pour AWS ParallelCluster
- Validation de conformité pour AWS ParallelCluster
- Application d'une version minimale de TLS 1.2

Informations de sécurité pour les services utilisés par AWS ParallelCluster

- Sécurité sur Amazon EC2
- Sécurité dans Amazon API Gateway
- Sécurité dans AWS Batch
- Sécurité dans AWS CloudFormation
- Sécurité sur Amazon CloudWatch
- Sécurité dans AWS CodeBuild
- Sécurité dans Amazon DynamoDB
- Sécurité dans Amazon ECR
- Sécurité dans Amazon ECS
- Sécurité dans Amazon EFS
- La sécurité au cœur FSx de Lustre
- Sécurité dans AWS Identity and Access Management (IAM)
- Sécurité dans EC2 Image Builder
- Sécurité dans AWS Lambda
- Sécurité dans Amazon Route 53
- Sécurité sur Amazon SNS
- Sécurité dans Amazon SQS (pour la AWS ParallelCluster version 2.x.)
- Sécurité dans Amazon S3
- Sécurité dans Amazon VPC

Protection des données dans AWS ParallelCluster

Le modèle de responsabilité AWS partagée de s'applique à la protection des données dans AWS ParallelCluster. Comme décrit dans ce modèle, AWS est chargé de protéger l'infrastructure mondiale qui gère tous les AWS Cloud. La gestion du contrôle de votre contenu hébergé sur cette infrastructure relève de votre responsabilité. Vous êtes également responsable des tâches de configuration et de gestion de la sécurité des Services AWS que vous utilisez. Pour plus d'informations sur la confidentialité des données, consultez Questions fréquentes (FAQ) sur la confidentialité des données. Pour en savoir plus sur la protection des données en Europe, consultez

le billet de blog Modèle de responsabilité partagée AWS et RGPD (Règlement général sur la protection des données) sur le Blog de sécuritéAWS .

À des fins de protection des données, nous vous recommandons de protéger les Compte AWS informations d'identification et de configurer les utilisateurs individuels avec AWS IAM Identity Center ou AWS Identity and Access Management (IAM). Ainsi, chaque utilisateur se voit attribuer uniquement les autorisations nécessaires pour exécuter ses tâches. Nous vous recommandons également de sécuriser vos données comme indiqué ci-dessous :

- Utilisez l'authentification multifactorielle (MFA) avec chaque compte.
- Utilisez le protocole SSL/TLS pour communiquer avec les ressources. AWS Nous exigeons TLS 1.2 et recommandons TLS 1.3.
- Configurez l'API et la journalisation de l'activité des utilisateurs avec AWS CloudTrail. Pour plus d'informations sur l'utilisation des CloudTrail sentiers pour capturer AWS des activités, consultez la section Utilisation des CloudTrail sentiers dans le guide de AWS CloudTrail l'utilisateur.
- Utilisez des solutions de AWS chiffrement, ainsi que tous les contrôles de sécurité par défaut qu'ils contiennent Services AWS.
- Utilisez des services de sécurité gérés avancés tels qu'Amazon Macie, qui contribuent à la découverte et à la sécurisation des données sensibles stockées dans Amazon S3.
- Si vous avez besoin de modules cryptographiques validés par la norme FIPS 140-3 pour accéder AWS via une interface de ligne de commande ou une API, utilisez un point de terminaison FIPS. Pour plus d'informations sur les points de terminaison FIPS disponibles, consultez Norme FIPS (Federal Information Processing Standard) 140-3.

Nous vous recommandons fortement de ne jamais placer d'informations confidentielles ou sensibles, telles que les adresses e-mail de vos clients, dans des balises ou des champs de texte libre tels que le champ Nom. Cela inclut lorsque vous travaillez avec AWS ParallelCluster ou d'autres Services AWS utilisateurs de la console, de l'API ou AWS SDKs. AWS CLI Toutes les données que vous entrez dans des balises ou des champs de texte de forme libre utilisés pour les noms peuvent être utilisées à des fins de facturation ou dans les journaux de diagnostic. Si vous fournissez une adresse URL à un serveur externe, nous vous recommandons fortement de ne pas inclure d'informations d'identification dans l'adresse URL permettant de valider votre demande adressée à ce serveur.

Chiffrement des données

Une caractéristique clé de tout service sécurisé est que les informations sont chiffrées lorsqu'elles ne sont pas utilisées activement.

Chiffrement au repos

AWS ParallelCluster ne stocke elle-même aucune donnée client autre que les informations d'identification dont elle a besoin pour interagir avec les AWS services au nom de l'utilisateur.

Pour les données relatives aux nœuds du cluster, les données peuvent être chiffrées au repos.

Pour les volumes Amazon EBS, le chiffrement est configuré à l'aide des `ebs_kms_key_id` paramètres de la `[ebs]`section relative à la AWS ParallelCluster version 2.x.) Pour plus d'informations, consultez la section relative au chiffrement Amazon EBS dans le guide de EC2 l'utilisateur Amazon.

Pour les volumes Amazon EFS, le chiffrement est configuré à l'aide `encrypted` des `efs_kms_key_id` paramètres et (dans la `[efs]`section de la AWS ParallelCluster version 2.x). Pour plus d'informations, consultez Comment fonctionne le chiffrement au repos dans le guide de l'utilisateur d'Amazon Elastic File System.

FSx Pour les systèmes de fichiers Lustre, le chiffrement des données au repos est automatiquement activé lors de la création d'un système de FSx fichiers Amazon. Pour plus d'informations, consultez la section Chiffrer les données au repos dans le guide de l'utilisateur d'Amazon FSx for Lustre.

Pour les types d'instance dotés de NVMe volumes, les données des volumes de stockage d' NVMe instance sont chiffrées à l'aide d'un chiffrement XTS-AES-256 implémenté sur un module matériel de l'instance. Les clés de chiffrement sont générées à l'aide du module matériel et sont uniques à chaque périphérique de stockage d' NVMe instance. Toutes les clés de chiffrement sont détruites lorsque l’instance est arrêtée ou résiliée et ne peuvent pas être récupérées. Vous ne pouvez pas désactiver le chiffrement et vous ne pouvez pas fournir votre propre clé de chiffrement. Pour plus d'informations, consultez la section Encryption at rest dans le guide de EC2 l'utilisateur Amazon.

Si vous invoquez un AWS service qui transmet les données des clients à votre ordinateur local à des fins de stockage, reportez-vous au chapitre Sécurité et conformité du guide de l'utilisateur de ce service pour obtenir des informations sur la manière dont ces données sont stockées, protégées et cryptées. AWS ParallelCluster

Chiffrement en transit

Par défaut, toutes les données transmises depuis l'ordinateur client en cours d'exécution AWS ParallelCluster et les points de terminaison du AWS service sont cryptées en envoyant le tout via une connexion HTTPS/TLS. Le trafic entre les nœuds du cluster peut être automatiquement

chiffré, en fonction des types d'instances sélectionnés. Pour plus d'informations, consultez la section Chiffrement en transit dans le guide de EC2 l'utilisateur Amazon.

Consultez aussi

- Protection des données sur Amazon EC2
- Protection des données dans EC2 Image Builder
- Protection des données dans AWS CloudFormation
- Protection des données dans Amazon EFS
- Protection des données dans Amazon S3
- Protection des données FSx pour Lustre

Identity and Access Management pour AWS ParallelCluster

AWS ParallelCluster utilise des rôles pour accéder à vos AWS ressources et à leurs services. Les politiques d'instance et d'utilisateur AWS ParallelCluster utilisées pour accorder des autorisations sont documentées surAWS Identity and Access Management rôles dans AWS ParallelCluster.

La seule différence majeure réside dans la façon dont vous vous authentifiez lorsque vous utilisez un utilisateur standard et des informations d'identification à long terme. Bien qu'un utilisateur ait besoin d'un mot de passe pour accéder à la console d'un AWS service, ce même utilisateur a besoin d'une paire de clés d'accès pour effectuer les mêmes opérations AWS ParallelCluster. Toutes les autres informations d'identification à court terme sont utilisées de la même manière qu'avec la console.

Les informations d'identification utilisées par AWS ParallelCluster sont stockées dans des fichiers en texte brut et ne sont pas cryptées.

- Le fichier `$HOME/.aws/credentials` stocke les informations d'identification à long terme requises pour accéder à vos ressources AWS . Vos ID de clé d'accès et clé d'accès secrète sont inclus.
- Les informations d'identification à court terme, telles que celles relatives aux rôles que vous assumez ou aux AWS IAM Identity Center services, sont également stockées dans le fichier `$HOME/.aws/cli/cache` et $HOME/. aws/sso/cachedossiers, respectivement.

Atténuation des risques

- Nous vous recommandons fortement de configurer vos autorisations de système de fichiers sur le dossier `$HOME/.aws` et ses dossiers et fichiers enfants, afin de restreindre l'accès aux seuls utilisateurs autorisés.
- Utilisez des rôles avec des informations d'identification temporaires dans la mesure du possible pour réduire les risques de dommages si les informations d'identification sont compromises. Utilisez les informations d'identification à long terme uniquement pour demander et actualiser les informations d'identification de rôle à court terme.

Validation de conformité pour AWS ParallelCluster

Des auditeurs tiers évaluent la sécurité et la conformité des AWS services dans le cadre de multiples programmes de AWS conformité. L'utilisation AWS ParallelCluster pour accéder à un service ne modifie pas la conformité de ce service.

Pour obtenir la liste des AWS services concernés par des programmes de conformité spécifiques, voir AWS Services concernés par programme de conformitéAWS . Pour des informations générales, consultez AWS la section Programmes de conformité Programmes AWS .

Vous pouvez télécharger des rapports d'audit tiers à l'aide du AWS Artifact. Pour plus d'informations, voir Téléchargement de rapports sur AWS Artifact.

Votre responsabilité en matière de conformité lors de l'utilisation AWS ParallelCluster est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. AWS fournit les ressources suivantes pour faciliter la mise en conformité :

- Guides de démarrage rapide sur la sécurité et la conformité Guides sur la sécurité et la conformité : ces guides de déploiement abordent les considérations architecturales et indiquent les étapes à suivre pour déployer des environnements de base axés sur la sécurité et la conformité sur. AWS
- Livre blanc sur l'architecture pour la sécurité et la conformité HIPAA sur Amazon Web Services — Ce AWS livre blanc décrit comment les entreprises peuvent créer des applications conformes à la loi HIPAA. AWS
- AWS ressources de https://aws.amazon.com/compliance/resources/ de conformité — Cette collection de classeurs et de guides peut s'appliquer à votre secteur d'activité et à votre région.
- Évaluation des ressources à l'aide des règles énoncées dans le guide du AWS Config développeur : le AWS Config service évalue dans quelle mesure les configurations de

vos ressources sont conformes aux pratiques internes, aux directives du secteur et aux réglementations.

- AWS Security Hub— Ce AWS service fournit une vue complète de l'état de votre sécurité interne, AWS ce qui vous permet de vérifier votre conformité aux normes et aux meilleures pratiques du secteur de la sécurité.

Application d'une version minimale de TLS 1.2

Pour renforcer la sécurité lors de la communication avec les AWS services, vous devez configurer votre système AWS ParallelCluster pour utiliser le protocole TLS 1.2 ou version ultérieure. Lorsque vous l'utilisez AWS ParallelCluster, Python est utilisé pour définir la version TLS.

Pour garantir qu'aucune version de TLS antérieure à TLS 1.2 n'est AWS ParallelCluster utilisée, vous devrez peut-être recompiler OpenSSL pour appliquer ce minimum, puis recompiler Python pour utiliser le nouvel OpenSSL.

Déterminez vos protocoles actuellement pris en charge

Tout d'abord, créez un certificat auto-signé à utiliser pour le serveur de test et le SDK Python à l'aide d'OpenSSL.

```
$ openssl req -subj '/CN=localhost' -x509 -newkey rsa:4096 -nodes -keyout key.pem -out
 cert.pem -days 365
```

Ensuite, faites tourner un serveur de test à l’aide d’OpenSSL.

```
$ openssl s_server -key key.pem -cert cert.pem -www
```

Dans une nouvelle fenêtre de terminal, créez un environnement virtuel et installez le SDK Python.

```
$ python3 -m venv test-env
source test-env/bin/activate
pip install botocore
```

Créez un nouveau script Python nommé `check.py` qui utilise la bibliothèque HTTP sous-jacente du 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'))
```

Exécutez votre nouveau script.

```
$ python check.py
```

Des détails sur la connexion effectuée s'affichent. Recherchez « Protocole : » dans le résultat. Si le résultat est « TLSv1 .2" ou version ultérieure, le SDK utilise par défaut TLS v1.2 ou version ultérieure. S'il s'agit d'une version antérieure, vous devez recompiler OpenSSL et recompiler Python.

Cependant, même si votre installation de Python est par défaut TLS v1.2 ou version ultérieure, il est toujours possible pour Python de renégocier vers une version antérieure à TLS v1.2 si le serveur ne prend pas en charge TLS v1.2 ou une version ultérieure. Pour vérifier que Python ne renégocie pas automatiquement des versions antérieures, redémarrez le serveur de test avec ce qui suit.

```
$ openssl s_server -key key.pem -cert cert.pem -no_tls1_3 -no_tls1_2 -www
```

Si vous utilisez une version antérieure d'OpenSSL, vous n'avez peut-être pas l'indicateur `-no_tls_3` disponible. Si c'est le cas, supprimez l'indicateur car la version d'OpenSSL que vous utilisez ne prend pas en charge TLS v1.3. Exécutez à nouveau le script Python.

```
$ python check.py
```

Si votre installation de Python ne renégocie pas correctement des versions antérieures à TLS 1.2, vous devriez recevoir une erreur SSL.

```
$ urllib3.exceptions.MaxRetryError: HTTPSConnectionPool(host='localhost',
 port=4433): Max retries exceeded with url: / (Caused by SSLError(SSLError(1, '[SSL:
 UNSUPPORTED_PROTOCOL] unsupported protocol (_ssl.c:1108)')))
```

Si vous êtes en mesure d'établir une connexion, vous devez recompiler OpenSSL et Python pour désactiver la négociation des protocoles antérieurs à TLS v1.2.

Compiler OpenSSL et Python

Pour vous assurer que AWS ParallelCluster cela ne négocie rien de antérieur à TLS 1.2, vous devez recompiler OpenSSL et Python. Pour ce faire, copiez le contenu suivant pour créer un script et exécutez-le.

```
#!/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
```

On compile ainsi une version de Python qui a un OpenSSL lié statiquement qui ne négocie pas automatiquement quoi que ce soit d'antérieur à TLS 1.2. On installe également OpenSSL dans le répertoire `/opt/openssl-with-min-tls1_2` et Python dans le répertoire `/opt/python-with-min-tls1_2`. Après avoir exécuté ce script, confirmez l'installation de la nouvelle version de Python.

```
$ /opt/python-with-min-tls1_2/bin/python3 --version
```

Ce qui suit devrait s'imprimer.

```
Python 3.8.1
```

Pour confirmer que cette nouvelle version de Python ne négocie pas une version antérieure à TLS 1.2, exécutez à nouveau les étapes à partir de Déterminez vos protocoles actuellement pris en chargeà l'aide de la version Python nouvellement installée (c'est-à-dire `/opt/python-with-min-tls1_2/bin/python3`).

Notes de publication et historique du document

Le tableau suivant décrit les principales mises à jour et les nouvelles fonctions pour le Guide de l'utilisateur AWS ParallelCluster . Nous mettons aussi la documentation à jour régulièrement pour prendre en compte les commentaires qui nous sont envoyés.

Modification	Description	Date
Publication de documentation uniquement	AWS ParallelCluster publication du guide de l'utilisateur spécifique à la version 2. Publication de documentation uniquement : • AWS ParallelCluster la version 2 possède son propre guide d'utilisation distinct.	17 juillet 2023
AWS ParallelCluster version 2.11.9 publiée	AWS ParallelCluster version 2.11.9 publiée. Correctifs de bogue : • Empêchez le remplacement des systèmes de fichiers gérés FSx pour Lustre et la perte de données lors des mises à jour de clusters qui incluent des modifications `devpc_security_group_id` .	2 décembre 2022

Pour plus de détails sur les modifications, consultez le `CHANGELOG` fichier du package aws-parallelcluster sur. GitHub

AWS ParallelCluster version 2.11.8 publiée	AWS ParallelCluster version 2.11.8 publiée.
	
	Changements :
	
	• Mettez à niveau la bibliothè que Intel MPI vers la version 2021 Update 6 (mise à jour à partir de la version 2021 Update 4). Pour plus d'informations, consultez la mise à jour 6 de la bibliothè que Intel® MPI 2021.
	• Mettre à jour le programme d'installation d'EFA vers la version 1.19.
	  • Pilote EFA : `efa-1.16.0-1`
	  • EFA-Config : `efa-config-1.11-1` (à partir de) `efa-config-1.9-1`
	  • Profil EFA : `efa-profile-1.5-1` (pas de changement)
	  • libFabric-aws : `libfabric-aws-1.16.0-1` (depuis) `libfabric-1.13.2`
	  • RDMA-Core : `rdma-core-41.0-2` (à partir de) `rdma-core-37.0`
	  • Ouvrez MPI : `openmpi40-aws-4.1.4-3` (à
		14 novembre 2022

	partir de `openmpi4.0-aws-4.1.1-2`) • Mettez à niveau le runtime Python, utilisé par les fonctions Lambda lors de l' AWS Batch intégration, vers python3.9. Correctifs de bogue : • Empêchez la modification des balises de cluster lors d'une mise à jour, car elles ne sont pas prises en charge. Pour plus de détails sur les modifications, consultez les `CHANGELOG` fichiers du package aws-parallelcluster sur. GitHub	
AWS ParallelCluster version 2.11.7 publiée	AWS ParallelCluster version 2.11.7 publiée. Changements : • Mettez à jour Slurm vers la version 20.11.9. Pour plus de détails sur les modifications, consultez les `CHANGELOG` fichiers du package aws-parallelcluster sur. GitHub	13 mai 2022

AWS ParallelCluster version 2.11.6 publiée	AWS ParallelCluster version 2.11.6 publiée. Améliorations : • Améliorez la gestion des exceptions en cas d'absence de réseau. Changements : • Mises à jour des packages du système d'exploitation et correctifs de sécurité. Pour plus de détails sur les modifications, consultez les `CHANGELOG` fichiers du package aws-parallelcluster sur. GitHub	19 avril 2022

AWS ParallelCluster version 2.11.5 publiée	AWS ParallelCluster version 2.11.5 publiée. Améliorations : • Ajoutez la prise en charge de l'`AutoImportPolicy` option `NEW_CHANGED_DELETED` as value of FSx for Lustre. • Supprimez le support pour les planificateurs SGE et Torque. • Désactivez le `log4j-cve-2021-44228-hotpatch` service sur Amazon Linux pour éviter toute dégradation potentielle des performances. Changements : • Mettez à niveau le pilote NVIDIA vers la version `470.103.01` (à partir de`470.82.01`). • Mettez à niveau le gestionnaire NVIDIA Fabric vers la version `470.103.01` (à partir de`470.82.01`). • Mettez à niveau la bibliothèque CUDA vers la version `11.4.4` (à partir de`11.4.3`).	1er mars 2022

- Intel MPI mis à jour vers la version 2021 Update 4 (mise à jour à partir de la version 2019 Update 8). Pour plus d'informations, consultez la mise à jour 4 de la bibliothèque Intel® MPI 2021.
- Prolongez le délai de création du nœud principal à une heure.

Correctifs de bogue :

- Corrigez la connexion DCV via les navigateurs.
- Corrigez les guillemets YAML pour empêcher les balises personnalisées d'être analysées sous forme de nombres.

Pour plus de détails sur les modifications, consultez les `CHANGELOG` fichiers du package aws-parallelcluster sur. GitHub

AWS ParallelCluster version 2.11.4 publiée	AWS ParallelCluster version 2.11.4 publiée. Les modifications incluent : • CentOS8 supports retirés. CentOS8 atteint la fin de vie (EOL) le 31 décembre 2021. • Passez Slurm Workload Manager à la version 20.11.8. • Mettez à niveau le client Cinc vers. `17.2.29` • Amazon DCV mis à jour vers Amazon DCV 2021.2-11190. Pour plus d'informations, consultez la section DCV 2021.2-11190 du 11 octobre 2021 dans le manuel Amazon DCV Administrator Guide. • Mettez à niveau le pilote NVIDIA vers la version `470.82.01` (à partir de`460.73.01`). • Mettez à niveau la bibliothèque CUDA vers la version `11.4.3` (à partir de`11.3.0`). • Mettez à niveau NVIDIA Fabric Manager vers`470.82.01` . • Désactivez la mise à jour du package au moment du	20 décembre 2021

lancement de l'instance sur Amazon Linux 2.

- Désactivez la mise à jour des packages sans surveillance sur Amazon Linux 2 Ubuntu et sur Amazon.
- Installez la version Python 3 des scripts d'AWS CloudFormation assistance sur les versions CentOS 7 et Ubuntu 18.04. (Ils étaient déjà utilisés sur Amazon Linux 2 et Ubuntu 20.04.)

Les correctifs incluent :

- Désactive la mise à jour du `ec2_iam_role` paramètre .
- Corrigez la `CpuOptions` configuration dans le modèle de lancement pour les T2 instances.

Pour plus de détails sur les modifications, consultez les `CHANGELOG` fichiers du cluster aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub

AWS ParallelCluster version 2.11.3 publiée	AWS ParallelCluster version 2.11.3 publiée. • Corrigez un `pcluster createami` échec dû à l'indisponibilité des Son of Grid Engine sources `surarc.liv.ac.uk` . Mettre à niveau le Elastic Fabric Adapter programme d'installation vers la version 1.14.1 (à partir de la version 1.13.0) • Configuration EFA : `efa-config-1.9-1` (à partir de `efa-config-1.9`) • Profil EFA : `efa-profile-1.5-1` (pas de changement) • Module EFA Kernel : `efa-1.14.2` (depuis `efa-1.13.0`) • Noyau RDMA : `rdma-core-37.0` (à partir de `rdma-core-35.0amzn`) • Libfabric : `libfabric-1.13.2` (de `libfabric-1.13.0amzn1.0`) • Ouvrez MPI : `openmpi40-aws-4.1.1-2` (pas de changement)	03 novembre 2021

GPUDirect Le RDMA est toujours activé s'il est pris en charge par le type d'instance.

- Les options `enable_efa_gdr` de configuration `enable_efa_gdr` et n'ont aucun effet.

Pour plus de détails sur les modifications, consultez les `CHANGELOG` fichiers du cluster aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub

AWS ParallelCluster version 2.11.2 publiée	AWS ParallelCluster version 2.11.2 publiée.
	
	Les modifications incluent :
	
	• N'installez pas EFA avec GPUDirect RDMA (GDR) activé au moment du démarrage si EFA est installé dans l'AMI de base.
	• Verrouillez la version du `nvidia-fabricmanager` package pour rester synchronisée avec la version du pilote NVIDIA installée par AWS ParallelCluster.
	• Slurm: correction d'un problème qui se produisait lorsque le cluster était arrêté puis redémarré alors qu'un nœud était en cours de démarrage.
	• Elastic Fabric Adapterprogramme d'installation mis à jour vers la version 1.13.0 :
	  • Configuration EFA : `efa-config-1.9` (pas de changement)
	  • Profil EFA : `efa-profile-1.5-1` (pas de changement)
	  • Module EFA Kernel : `efa-1.13.0` (aucun changement)
		27 août 2021

  - Noyau RDMA : `rdma-core-35.0amzn` (à partir de `rdma-core-32.1amzn`)
  - Libfabric : `libfabric-1.13.0amzn1.0` (de `libfabric-1.11.2amzn1.1`)
  - Ouvrez MPI : `openmpi40-aws-4.1.1-2` (pas de changement)
- Lorsque vous utilisez une AMI personnalisée avec un package EFA préinstallé, aucune modification n'est apportée à EFA au moment du démarrage du nœud. Le déploiement du package EFA d'origine est préservé.

Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour le cluster aws-parallelcluster et les packages qui s'y trouvent. aws-parallelcluster-cookbook GitHub

AWS ParallelCluster version 2.11.1 publiée	AWS ParallelCluster version 2.11.1 publiée.
	
	Les modifications incluent :
	
	• Montez les systèmes de fichiers à l'aide de l'option `noatime` mount pour arrêter d'enregistrer l'heure du dernier accès lors de la lecture d'un fichier. Cela améliore les performan ces du système de fichiers distant.
	• Elastic Fabric Adapterprogramme d'install ation mis à jour vers la version 1.12.3 :
	  • Configuration EFA : `efa-config-1.9` (à partir de `efa-config-1.8-1`)
	  • Profil EFA : `efa-profile-1.5-1` (pas de changement)
	  • Module EFA Kernel : `efa-1.13.0` (depuis `efa-1.12.3`)
	  • Cœur RDMA : `rdma-core-32.1amzn` (pas de changement)
	  • Libfabric : `libfabric-1.11.2amzn1.1` (pas de changement)
		23 juillet 2021

  - Ouvrez MPI : `openmpi40-aws-4.1.1-2` (pas de changement)
- Réessayez d'installer le `aws-parallelcluster` package sur le nœud principal lorsque vous l'utilisez AWS Batch comme planificateur.
- Évitez les défaillances lorsque vous créez SGE un type d'instance de plus de 31 CPUs v.
- Épinglé à la version 1.247347.6 de l' CloudWatch agent Amazon pour éviter les problèmes rencontrés dans la version 1.247348.0.

Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour le cluster aws-parallelcluster et les packages qui s'y trouvent. aws-parallelcluster-cookbook GitHub

AWS ParallelCluster version 2.11.0 publiée	AWS ParallelCluster version 2.11.0 publiée. Les modifications incluent : • Ajout du support pour Ubuntu 20.04 (`ubuntu2004`) et suppression du support pour Ubuntu 16.04 (`ubuntu1604`) et Amazon Linux (`alinux`). Amazon Linux 2 (`alinux2`) reste entièrement compatible. Pour de plus amples informations, veuillez consulter `base_os`. • Suppression du support pour les versions de Python inférieures à 3.6. • La taille du volume racine par défaut a été augmentée à 35 Gibioctets (GiB). Pour plus d'informations, consultez `compute_root_volume_size` et `master_root_volume_size`. • Elastic Fabric Adapterprogramme d'installation mis à jour vers la version 1.12.2 : • Configuration EFA : `efa-config-1.8-1` (à partir de `efa-config-1.7`)	1er juillet 2021

  - Profil EFA : `efa-profile-1.5-1` (de `efa-profile-1.4`)
  - Module EFA Kernel : `efa-1.12.3` (depuis `efa-1.10.2`)
  - Noyau RDMA : `rdma-core-32.1amzn` (à partir de `rdma-core-31.2amzn`)
  - Libfabric : `libfabric-1.11.2amzn1.1` (de `libfabric-1.11.1amzn1.0`)
  - Ouvrez MPI : `openmpi40-aws-4.1.1-2` (à partir de `openmpi40-aws-4.1.0`)
- Mise Slurm à niveau vers la version `20.11.7` (à partir de `20.02.7`).
- Installez l'agent SSM sur `centos7` et. `centos8` (L'agent SSM est préinstallé dans `alinux2` `ubuntu1804`, et `ubuntu2004`.)
- SGE: utilisez toujours le shortname comme filtre de nom d'hôte avec. `qstat`
- Utilisez le service de métadonnées d'instance version 2 (IMDSv2) au lieu du service de métadonnées d'instance version 1

(IMDSv1) pour récupérer les métadonnées d'instance. Pour plus d'informations, consultez la section Métadonnées de l'instance et données utilisateur dans le guide de EC2 l'utilisateur Amazon.

- Mettez à niveau le pilote NVIDIA vers la version `460.73.01` (à partir de`450.80.02`).
- Mettez à niveau la bibliothèque CUDA vers la version `11.3.0` (à partir de`11.0`).
- Mettez à niveau NVIDIA Fabric Manager vers`nvidia-fabricmanager-460` .
- Mettez à niveau le Python utilisé dans AWS ParallelCluster virtualenvs vers `3.7.10` (from). `3.6.13`
- Mettez à niveau le client Cinc vers. `16.13.16`
- Mettez à niveau les dépendances tierces de aws-parallelcluster-cookbook:
  - `apt-7.4.0` (à partir de`apt-7.3.0`).
  - `iptables-8.0.0` (à partir de`iptables-7.1.0`).

- `line-4.0.1` (à partir `deline-2.9.0`).
- `openssh-2.9.1` (à partir `deopenssh-2.8.1`).
- `pyenv-3.4.2` (à partir `depyenv-3.1.1`).
- `selinux-3.1.1` (à partir `deselinux-2.1.1`).
- `ulimit-1.1.1` (à partir `deulimit-1.0.0`).
- `yum-6.1.1` (à partir `deyum-5.1.0`).
- `yum-epel-4.1.2` (à partir `deyum-epel-3.3.0`).

Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub

AWS ParallelCluster version 2.10.4 publiée	AWS ParallelCluster version 2.10.4 publiée. Les modifications incluent : • Mise Slurm à niveau vers la version `20.02.7` (à partir de`20.02.4`). Pour plus de détails sur les modifications, consultez le fichier CHANGELOG du package aws-parallelcluster sur. GitHub	15 mai 2021

AWS ParallelCluster version 2.10.3 publiée	AWS ParallelCluster version 2.10.3 publiée.
	
	Les modifications incluent :
	
	• Ajout de la prise en charge de la Ubuntu version 18.04 et d'Amazon Linux 2 sur les instances AWS Graviton basées sur ARM en Chine AWS et. AWS GovCloud (US) Régions AWS
	• Elastic Fabric Adapterprogramme d'installation mis à jour vers la version 1.11.2 :
	  • Configuration EFA : `efa-config-1.7` (pas de changement)
	  • Profil EFA : `efa-profile-1.4` (de`efa-profile-1.3`)
	  • Module EFA Kernel : `efa-1.10.2` (aucun changement)
	  • Cœur RDMA : `rdma-core-31.2amzn` (pas de changement)
	  • Libfabric : `libfabric-1.11.1amzn1.0` (pas de changement)
	  • Ouvrez MPI : `openmpi40-aws-4.1.0` (pas de changement)
		18 mars 2021

Pour plus de détails sur les modifications, consultez le fichier CHANGELOG du package aws-parallelcluster sur. GitHub

AWS ParallelCluster version 2.10.2 publiée	AWS ParallelCluster version 2.10.2 publiée. Les modifications incluent : • Améliorez la validation de la configuration du cluster pour utiliser l'AMI cible du cluster lors de l'appel de l'opération EC2 RunInstances d'API Amazon en `--dry-run` mode. • Mettez à jour la version de Python utilisée dans les environnements AWS ParallelCluster virtuels vers la version 3.6.13. • `sanity_check` Correctif pour les types d'instances Arm. • `enable_efa` Corrigé lors de l'utilisation `centos8` avec le Slurm planificateur ou les types d'instances Arm. • Exécuter `apt update` en mode non interactif (`-y`). • Fix `encrypted_ephemeral` = vrai avec `alinux2` et `centos8`. Pour plus de détails sur les modifications, consultez le fichier CHANGELOG du	2 mars 2021

package aws-parallelcluster
sur. GitHub

AWS ParallelCluster version 2.10.1 publiée	AWS ParallelCluster sortie de la version 2.10.1.
	
	Les modifications incluent :
	
	• Ajout du support pour l'Afrique (Le Cap) (`af-south-1`), l'Europe (Milan`me-south-1`) () et le Moyen-Orient (Bahreïn) `me-south-1` (Régions AWS). Au lancement, le support est limité de la manière suivante :
	  • FSx pour Lustre et les instances Graviton basées sur ARM ne sont prises en charge dans aucun de ces systèmes. Régions AWS
	  • AWS Batch n'est pas pris en charge en Afrique (Cape Town).
	  • Amazon EBS `io2` et les types de `gp3` volume ne sont pas pris en charge en Afrique (Le Cap) et en Europe (Milan). Régions AWS
	• Ajout du support pour Amazon EBS `io2` et les types de `gp3` volume. Pour plus d'informations, consultez les `[ebs]`sections et `[raid]`sections.
		22 décembre 2020

- Ajout de la prise en charge des instances Graviton2 basées Elastic Fabric Adapter sur ARM en cours d'exécution`alinux2`, `ubuntu1804` ou. `ubuntu2004` Pour de plus amples informations, veuillez consulter Elastic Fabric Adapter.
- Installez les bibliothèques de performance Arm 20.2.1 sur Arm AMIs (`alinux2centos8`, et`ubuntu1804`). Pour de plus amples informations, veuillez consulter Bibliothèques de performances Arm.
- Intel MPI mis à jour vers la version 2019 Update 8 (mise à jour à partir de la version 2019 Update 7). Pour plus d'informations, consultez la mise à jour 8 de la bibliothèque Intel® MPI 2019.
- Suppression de l'appel AWS CloudFormation `DescribeStacks` d'opération d'API du point d'entrée AWS Batch Docker pour mettre fin aux échecs de tâches provoqués par le throttling by. AWS CloudFormation

- Amélioration des appels à l'appel d'opération de EC2 `DescribeInstanceTypes` l'API Amazon lors de la validation d'une configuration de cluster.
- Les images Docker d'Amazon Linux 2 sont extraites d'Amazon ECR Public lors de la création de l'image Docker pour le planificateur. `awsbatch`
- Le type d'instance par défaut est passé du type d'`t2.micro`instance codé en dur au type d'instance de niveau gratuit pour le Région AWS (`t2.micro`ou `t3.micro`, selon le Région AWS). Régions AWS qui ne disposent pas d'un niveau gratuit par défaut pour le type d'`t3.micro`instance.
- Elastic Fabric Adapterprogramme d'installation mis à jour vers la version 1.11.1 :
  - Configuration EFA : `efa-config-1.7` (à partir de `efa-config-1.5`)
  - Profil EFA : `efa-profile-1.3` (de `efa-profile-1.1`)

  - Module EFA Kernel : `efa-1.10.2` (aucun changement)
  - Noyau RDMA : `rdma-core-31.2amzn` (à partir de `rdma-core-31.amzn0`)
  - Libfabric : `libfabric-1.11.1amzn1.0` (de `libfabric-1.10.1amzn1.1`)
  - Ouvrez MPI : `openmpi40-aws-4.1.0` (à partir de `openmpi40-aws-4.0.5`)
- Les `master_subnet_id` paramètres `vpc_settings.vpc_id`, et sont désormais obligatoires.
- Le `nfsd` démon du nœud principal est désormais configuré pour utiliser au moins 8 threads. S'il y a plus de 8 cœurs, il utilisera autant de threads qu'il y a de cœurs. Lorsqu'il `ubuntu1604` est utilisé, le paramètre ne change qu'après le redémarrage du nœud.
- Amazon DCV mis à jour vers Amazon DCV 2020.2-9662. Pour plus d'informations, consultez la section DCV 2020.2-9662

du 4 décembre 2020 dans le manuel Amazon DCV Administrator Guide.

- Les packages Intel MPI et HPC pour AWS ParallelCluster sont extraits d'Amazon S3. Ils ne sont plus extraits des dépôts Intel yum.
- Le systemd niveau d'exécution par défaut a été changé en « `multi-user.target` on all » OSs lors de la création de l'officiel AWS ParallelCluster AMIs. Le niveau d'exécuti on est défini `graphical.target` sur le nœud principal uniquement lorsque le DCV est activé. Cela empêche les services graphiques (tels que `x/gdm`) de s'exécuter lorsqu'ils ne sont pas nécessaires.
- Support activé pour les `p4d.24xlarge` instances sur le nœud principal.
- Augmentez le nombre maximum de tentatives lors de l'enregistrement de Slurm nœuds dans Amazon Route 53.

Pour plus de détails sur les modifications, consultez les

fichiers CHANGELOG pour aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub

AWS ParallelCluster version 2.10.0 publiée	AWS ParallelCluster version 2.10.0 publiée.
	
	Les modifications incluent :
	
	• Ajout du support pour CentOS 8 personnes en tout Régions AWS (en dehors de la AWS Chine et AWS GovCloud des États-Unis). Suppression du support pour CentOS 6.
	• Ajout de la prise en charge `p4d.24xlarge` des instances pour les nœuds de calcul.
	• Ajout du support pour NVIDIA GPUDirect RDMA sur EFA en utilisant le nouveau `enable_efa_gdr` paramètre.
	• Ajout du support pour les fonctionnalités FSx d'Amazon for Lustre.
	  • Configurez votre système de fichiers Amazon FSx for Lustre pour importer les préférences à l'aide de ce `auto_import_policy` paramètre.
	  • Ajout de la prise en charge des systèmes de fichiers Amazon FSx for Lustre basés sur des disques durs à l'aide des paramètres `storage_t`
		18 novembre 2020

ype et drive_cache_type .

- Ajout d'un CloudWatch tableau de bord Amazon, incluant les métriques du nœud principal et un accès facile aux journaux du cluster. Pour de plus amples informations, veuillez consulter Tableau de CloudWatch bord Amazon.
- Ajout de la prise en charge de l'utilisation d'un compartiment Amazon S3 existant pour stocker les informations de configuration du cluster, à l'aide du cluster_resource_bucket paramètre.
- Amélioration de la pcluster createami commande.
  - `--post-install` Paramètre ajouté pour utiliser un script de post-installation lors de la création d'une AMI.
  - Ajout d'une étape de validation pour échouer lors de l'utilisation d'une AMI de base créée par une version différente de AWS ParallelCluster.
  - Ajout d'une étape de validation pour échouer si

le système d'exploitation sélectionné est différent du système d'exploitation de l'AMI de base.

  - Ajout de la prise en charge de l'utilisation d'une AMI AWS ParallelCluster de base.
- Amélioration de la `pcluster update` commande.
  - Le `tags` paramètre peut désormais être modifié lors d'une mise à jour.
  - Les files d'attente peuvent désormais être redimensionnées lors d'une mise à jour sans arrêter le parc informatique
- Paramètre `all_or_nothing_batch` de configuration ajouté pour `slurm_resume` le script. When`True`, ne `slurm_resume` réussira que si toutes les instances requises par toutes les tâches en attente Slurm seront disponibles. Pour plus d'informations, consultez la section Présentation des `all_or_nothing_batch` lancements dans le AWS ParallelCluster Wiki sur GitHub.

- Elastic Fabric Adapterprogramme d'installation mis à jour vers la version 1.10.1 :
  - Configuration EFA : `efa-config-1.5` (à partir de `efa-config-1.4`)
  - Profil EFA : `efa-profile-1.1` (de `efa-profile-1.0.0`)
  - Module EFA Kernel : `efa-1.10.2` (depuis `efa-1.6.0`)
  - Noyau RDMA : `rdma-core-31.amzn0` (à partir de `rdma-core-28.amzn0`)
  - Libfabric : `libfabric-1.11.1amzn1.0` (de `libfabric-1.10.1amzn1.1`)
  - Ouvrez MPI : `openmpi40-aws-4.0.5` (à partir de `openmpi40-aws-4.0.3`)
- Dans les AWS GovCloud (US) régions, activez le support pour Amazon DCV et AWS Batch.
- Dans les régions de AWS Chine, activez le support pour Amazon FSx pour Lustre.
- Mettez à niveau le pilote NVIDIA vers la version

450.80.02 (à partir de 450.51.05).

- Installez NVIDIA Fabric Manager pour activer NVIDIA NVSwitch sur les plateformes prises en charge.
- Suppression de Région AWS la valeur par défaut `deus-east-1` . Par défaut, cet ordre de recherche est utilisé.
  - Région AWS spécifié dans `-r` notre `--region` argument.
  - `AWS_DEFAULT_REGION` variable d'environnement.
  - `aws_region_name` paramètre dans la `[aws]`section du fichier de AWS ParallelCluster configuration (par défaut`~/.parallelcluster/config`).
  - `region`paramètre dans la `[default]` section du fichier de AWS CLI configuration (par défaut`~/aws/config`).

Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour aws-parallelcluster et les

packages correspondants.

aws-parallelcluster-cookbook

aws-parallelcluster-node

GitHub

AWS ParallelCluster version 2.9.0 publiée	AWS ParallelCluster version 2.9.0 publiée. Les modifications incluent : • Ajout de la prise en charge de plusieurs files d'attente et de plusieurs types d'instances dans le parc informatique en cas d'utilisation avecSlurm Workload Manager. Lorsque vous utilisez des files d'attente , les groupes Auto Scaling ne sont plus utilisés. Slurm Une zone hébergée Amazon Route 53 est désormais créée avec le cluster et est utilisée pour la résolution DNS des nœuds de calcul lorsque le Slurm planificateur est utilisé. Pour de plus amples informations, veuillez consulter Mode de file d'attente multiple. • Ajout de la prise en charge d'Amazon DCV sur les instances basées sur AWS Graviton basées sur ARM. • Ajout de la prise en charge de la désactivation de l'hyperthreading sur les types d'instances qui ne prennent pas en charge les options de processeur dans les modèles de lancement	11 septembre 2020

(par exemple les types d'`*.metal`instance).

- Ajout de la prise en charge de NFS 4 pour les systèmes de fichiers partagés depuis le nœud principal.
- Suppression de la dépendance à l'égard de cfn-init lors du démarrage des nœuds de calcul afin d'éviter les ralentissements liés à l'adhésion d'un grand nombre de AWS CloudFormation nœuds au cluster.
- Elastic Fabric Adapterprogramme d'installation mis à jour vers la version 1.9.5 :
  - Configuration EFA : `efa-config-1.4` (à partir de`efa-config-1.3`)
  - Profil EFA : `efa-profile-1.0.0` (nouveau)
  - Module noyau : `efa-1.6.0` (pas de changement)
  - Cœur RDMA : `rdma-core-28.amzn0` (pas de changement)
  - Libfabric : `libfabric-1.10.1amzn1.1` (pas de changement)
  - Ouvrez MPI : `openmpi40-aws-4.0.3` (pas de changement)

- Mise Slurm à niveau vers la version `20.02.4` (à partir de`19.05.5`).
- Amazon DCV mis à jour vers Amazon DCV 2020.1-9012. Pour plus d'informations, consultez les notes de mise à jour de DCV 2020.1-9012 — 24 août 2020 dans le manuel Amazon DCV Administrator Guide.
- Lors du montage de lecteurs NFS partagés, utilisez l'adresse IP privée du nœud principal au lieu du nom d'hôte.
- De nouveaux flux de CloudWatch journaux ont été ajoutés à Logs : `chef-client` `clustermgtd` `computemgtd` ,`slurm_resume` ,, et`slurm_suspend` .
- Ajout de la prise en charge des noms de files d'attente dans les scripts de pré-installation et de post-installation.
- Dans le AWS GovCloud (US) Régions AWS, utilisez l'option de facturation à la demande Amazon DynamoDB. Pour plus d'informations, consultez la section Mode à la demande

	dans le guide du développeur Amazon DynamoDB. Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub	
AWS ParallelCluster version 2.8.1 publiée	AWS ParallelCluster version 2.8.1 publiée. Les modifications incluent : • Désactivez le verrouillage d'écran pour les sessions Amazon DCV afin d'éviter que les utilisateurs ne soient verrouillés. • Correctif `pcluster configure` lors de l'inclusion d'un type d'instance basé sur AWS Graviton basé sur ARM. Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookbook aws-parallelcluster-node GitHub	4 août 2020

AWS ParallelCluster version 2.8.0 publiée	AWS ParallelCluster version 2.8.0 publiée. Les modifications incluent : • Ajout de la prise en charge des instances basées sur ARM AWS basées sur Graviton (comme le `A1` et). `C6g` • Ajout de la prise en charge des fonctionnalités de sauvegarde quotidienne automatique d'Amazon FSx for Lustre. Pour plus d'informations, consultez `automatic_backup_retention_days`, `copy_tags_to_backups`, `daily_automatic_backup_start_time` et `fsx_backup_id`. • Suppression de la dépendance à Berkshelf depuis. `pcluster createami` • Amélioration de la robustesse et de l'expérience utilisateur de. `pcluster update` Pour de plus amples informations, veuillez consulter Utiliser `pcluster update`. • Elastic Fabric Adapterprogramme d'install	23 juillet 2020

ation mis à jour vers la version 1.9.4 :

- Module noyau : `efa-1.6.0` (mis à jour depuis `efa-1.5.1`)
- Noyau RDMA : `rdma-core-28.amzn0` (mis à jour depuis `rdma-core-25.0`)
- Libfabric : `libfabric-1.10.1amzn1.1` (mis à jour depuis `libfabric-aws-1.9.0amzn1.1`)
- Ouvrez MPI : `openmpi40-aws-4.0.3` (pas de changement)

- Mettez à niveau le pilote NVIDIA vers la version 440.95.01 de Tesla sur CentOS 6 et la version 450.51.05 sur toutes les autres distributions.
- Mettez à niveau la bibliothèque CUDA vers la version 11.0 sur toutes les distributions autres que CentOS 6.

Pour plus de détails sur les modifications, consultez les fichiers CHANGELOG pour aws-parallelcluster et les packages correspondants. aws-parallelcluster-cookboo

kaws-parallelcluster-node

GitHub

AWS ParallelCluster version 2.7.0 publiée	AWS ParallelCluster version 2.7.0 publiée.
	
	Les modifications incluent :
	
	• `base_os` est maintenant un paramètre obligatoire.
	• `scheduler` est maintenant un paramètre obligatoire.
	• Amazon DCV mis à jour vers Amazon DCV 2020.0. Pour plus d'informations, consultez Amazon DCV publie la version 2020.0 avec son Surround 7.1 et prise en charge du stylet.
	
	Intel MPI mis à jour vers la version 2019 Update 7 (mise à jour depuis la version 2019 Update 6). Pour de plus amples informations, veuillez consulter la Mise à jour 7 de la bibliothèque Intel® MPI 2019.
	
	Elastic Fabric Adapter mis à jour vers la version 1.8.4 :
	  • Module noyau : `efa-1.5.1` (pas de changement)
	  • Cœur RDMA : `rdma-core-25.0` (pas de changement)
		19 mai 2020

	• Libfabric : `libfabric-aws-1.9.0amzn1.1` (pas de changement) • Open MPI : `openmpi40-aws-4.0.3` (mis à jour à partir de `openmpi40-aws-4.0.2`) • Mise à niveau CentOS 7 AMI vers la version 7.8-2003 (mise à jour du 7.7-1908) . Pour plus d'informations, consultez les notes de mise à jour de CentOS -7 (2003).	
AWS ParallelCluster version 2.6.1 publiée	AWS ParallelCluster sortie de la version 2.6.1. Les modifications incluent : • Supprimé `cfn-init-cmd` et `cfn-wire` présent dans les journaux stockés dans Amazon CloudWatch Logs. Pour de plus amples informations, veuillez consulter Intégration à Amazon CloudWatch Logs.	17 avril 2020

AWS ParallelCluster version 2.6.0 publiée	AWS ParallelCluster version 2.6.0 publiée. Les modifications incluent : • Ajout de la prise en charge d'Amazon Linux 2 • Amazon CloudWatch Logs est désormais utilisé pour collecter les journaux du cluster et du planifica teur. Pour de plus amples informations, veuillez consulter Intégration à Amazon CloudWatch Logs. • Ajout de la prise en charge des nouveaux types de déploiement d'Amazon FSx for Lustre `SCRATCH_2` et`PERSISTENT_1` . Support FSx pour Lustre les Ubuntu 18.04 et Ubuntu 16.04. Pour de plus amples informations, veuillez consulter fsx. • Ajout du support pour Amazon DCV le Ubuntu 18.04. Pour de plus amples informations, veuillez consulter Connectez-vous au nœud principal via Amazon DCV.	27 février 2020
AWS ParallelCluster version 2.5.1 publiée	AWS ParallelCluster version 2.5.1 publiée.	13 décembre 2019
AWS ParallelCluster version 2.5.0 publiée	AWS ParallelCluster version 2.5.0 publiée.	18 novembre 2019

AWS ParallelCluster introduit le support pour Intel MPI	AWS ParallelCluster la version 2.4.1 introduit le support pour Intel MPI.	29 juillet 2019
AWS ParallelCluster introduit le support pour l'EFA	AWS ParallelCluster la version 2.4.0 introduit le support pour Elastic Fabric Adapter (EFA).	11 juin 2019
AWS ParallelCluster documentation publiée sur le site de AWS documentation	La AWS ParallelCluster documentation est désormais disponible en 10 langues et aux formats HTML et PDF.	24 mai 2018

Les traductions sont fournies par des outils de traduction automatique. En cas de conflit entre le contenu d'une traduction et celui de la version originale en anglais, la version anglaise prévaudra.