AWS ParallelCluster 用户指南 (v3)

AWS ParallelCluster

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

AWS ParallelCluster: AWS ParallelCluster 用户指南 (v3)

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

Amazon 的商标和商业外观不得用于任何非 Amazon 的商品或服务，也不得以任何可能引起客户混淆、贬低或诋毁 Amazon 的方式使用。所有非 Amazon 拥有的其他商标均为各自所有者的财产，这些所有者可能附属于 Amazon、与 Amazon 有关联或由 Amazon 赞助，也可能不是如此。

Table of Contents

什么是 AWS ParallelCluster 1
如何 AWS ParallelCluster 运作 2
AWS ParallelCluster 进程 2
clustermgtd 2
clusterstatusmgtd 3
computemgtd 3
AWS 使用的服务 AWS ParallelCluster 3
Amazon API Gateway 4
AWS Batch 5
AWS CloudFormation 5
Amazon CloudWatch 5
亚马逊 CloudWatch 活动 5
Amazon CloudWatch 日志 6
AWS CodeBuild 6
Amazon DynamoDB 6
Amazon Elastic Block Store 6
Amazon Elastic Compute Cloud 7
Amazon Elastic Container Registry 7
Amazon EFS 7
亚马逊 f FSx or Lustre 7
FSx 适用于 NetApp ONTAP 的亚马逊 8
FSx 适用于 OpenZFS 的亚马逊 8
AWS Identity and Access Management 8
AWS Lambda 8
Amazon RDS 9
Amazon Route 53 9
Amazon Simple Notification Service 9
Amazon Simple Storage Service 9
Amazon VPC 10
Elastic Fabric Adapter 10
EC2 Image Builder 10
Amazon DCV 10
AWS ParallelCluster 内部目录 10
设置 AWS ParallelCluster 12

先决条件 12
设置一个 AWS 账户 12
创建密钥对 14
安装 AWS ParallelCluster CLI 14
AWS ParallelCluster 在虚拟环境中安装（推荐） 14
使用 pip AWS ParallelCluster 在非虚拟环境中安装 16
AWS ParallelCluster 作为独立应用程序安装 17
安装后需要执行的步骤 19
安装 AWS ParallelCluster 用户界面 19
安装 PCUI 20
堆栈参数 22
配置自定义域 26
Amazon Cognito 用户群体选项 27
识别 AWS ParallelCluster 和 PCUI 版本 29
PCUI 成本 30
入门 30
使用 AWS ParallelCluster CLI 配置和创建集群 31
使用 AWS ParallelCluster UI 配置和创建集群 40
连接到集群 41
集群的多用户访问 42
创建 Active Directory 43
使用 AD 域创建集群 43
登录到与 AD 域集成的集群 46
运行 MPI 作业 47
LDAP (S) 集群配置示例 AWS Managed Microsoft AD 47
最佳实践 51
最佳实践：头节点实例类型选择 51
最佳实践：网络性能 51
最佳实践：预算提醒 53
最佳实践：将集群移至新的 AWS ParallelCluster 次要版本或补丁版本 53
从 AWS ParallelCluster 2.x 升级到 3.x 54
自定义引导操作 54
AWS ParallelCluster 2.x 和 3.x 使用不同的配置文件语法 54
包容性语言 60
调度器支持 61
AWS ParallelCluster CLI 61

IMDS 配置更新 63
使用 AWS ParallelCluster 64
AWS ParallelCluster 用户界面 65
AWS Lambda 中的 VPC 配置 AWS ParallelCluster 66
AWS Identity and Access Management 中的权限 AWS ParallelCluster 67
AWS ParallelCluster Amazon EC2 实例角色 68
AWS ParallelCluster `pcluster`用户策略示例 69
AWS ParallelCluster 用于管理 IAM 资源的用户示例策略 83
AWS ParallelCluster 用于管理 IAM 权限的配置参数 89
网络配置 103
AWS ParallelCluster 在单个公有子网中 104
AWS ParallelCluster 使用两个子网 105
AWS ParallelCluster 在使用连接的单个私有子网中 AWS Direct Connect 107
AWS ParallelCluster 使用调 AWS Batch 度器 108
AWS ParallelCluster 在无法访问互联网的单个子网中 110
由配置的登录节点 AWS ParallelCluster 116
登录节点的安全性 116
登录节点联网 116
登录节点存储 117
登录节点的 Imds 属性 117
登录节点生命周期 118
运行登录节点池所需的权限 118
自定义引导操作 119
配置 121
参数 125
包含自定义引导操作的示例集群 125
如何更新自定义引导脚本的示例 IMDSv2 127
如何更新配置的示例 IMDSv1 127
使用 Amazon S3 128
示例 128
使用竞价型实例 129
情形 1：没有运行作业的竞价型实例被中断 130
情形 2：运行单节点作业的竞价型实例被中断 130
情形 3：运行多节点作业的竞价型实例被中断 130
支持的调度器 AWS ParallelCluster 130
Slurm Workload Manager 130

AWS Batch ... 188
共享存储 ... 195
Amazon EBS ... 197
Amazon EFS ... 198
FSx 为了光泽 ... 199
FSx 适用于 ONTAP、 FSx OpenZFS 和文件缓存 ... 199
使用共享存储 ... 200
限额 ... 203
标记 ... 204
查看标签 ... 204
监控 AWS ParallelCluster 和日志 ... 207
与 Amazon CloudWatch 日志集成 ... 208
亚马逊 CloudWatch 控制面板 ... 211
Amazon 针对集群指标的 CloudWatch 警报 ... 213
AWS ParallelCluster 配置的日志轮换 ... 216
`pcluster` CLI 日志 ... 217
Amazon EC2 控制台输出日志 ... 217
检索 PCUI 和 AWS ParallelCluster 运行时日志 ... 218
检索和保留日志 ... 220
AWS CloudFormation 自定义资源 ... 223
由托管的提供商堆栈 AWS ParallelCluster ... 224
集群资源 ... 225
集群操作 ... 228
对包含 AWS ParallelCluster 自定义资源的堆栈进行故障排除 ... 229
Elastic Fabric Adapter ... 229
启用 Intel MPI ... 230
AWS ParallelCluster API ... 231
AWS ParallelCluster API 文档 ... 232
使用以下方式部署 AWS ParallelCluster API AWS CLI ... 232
更新 API ... 235
正在调用 API AWS ParallelCluster ... 235
访问 API 日志和数据记录 ... 238
AWS ParallelCluster 适用于 Terraform ... 238
通过 Amazon DCV 连接到头节点和登录节点 ... 239
Amazon DCV HTTPS 证书 ... 239
许可 Amazon DCV ... 240

使用 pcluster update-cluster 240
更新策略：定义 240
pcluster update-cluster 示例 244
AWS ParallelCluster AMI 自定义 246
AWS ParallelCluster AMI 自定义注意事项 247
执行自定义组件验证测试 247
使用 pcluster 命令监控 Image Builder 进程以帮助进行调试 248
其他考虑因素 248
使用按需容量预留（ODCR）启动实例 249
将 ODCR 与 AWS ParallelCluster 249
使用容量块（CB）启动实例 257
将 CB 与 AWS ParallelCluster 257
AMI 修补和亚马逊 EC2 实例更换 259
头节点实例更新或替换 260
保存临时驱动器中的数据 260
停止和启动集群的头节点 260
操作系统 262
操作系统注意事项 262
的参考 AWS ParallelCluster 264
AWS ParallelCluster 版本 3 CLI 命令 264
pcluster 265
pcluster3-config-converter 309
配置文件 310
集群配置文件 310
构建映像配置文件 435
AWS ParallelCluster API 参考 444
buildImage 445
createCluster 450
deleteCluster 456
deleteClusterInstances 459
deleteImage 461
describeCluster 464
describeClusterInstances 472
describeComputeFleet 476
describeImage 477
getClusterLog活动 484

getClusterStack活动 488
getImageLog活动 492
getImageStack活动 496
listClusters 500
listClusterLog直播 504
listImageLog直播 508
listImages 512
listOfficialImages 515
updateCluster 518
updateComputeFleet 524
AWS ParallelCluster Python 库 API 526
AWS ParallelCluster Python 库授权 527
安装 AWS ParallelCluster Python 库 527
集群 API 操作 527
计算实例集 API 操作 531
集群和堆栈日志操作 533
映像 API 操作 536
映像和堆栈日志操作 538
示例 541
AWS Lambda 用于 AWS ParallelCluster Python 库 542
教程 544
在上运行你的第一份工作 AWS ParallelCluster 544
验证安装 545
创建您的第一个集群 545
登录到头节点 546
使用 Slurm 运行首个作业 547
构建自定义 AWS ParallelCluster AMI 548
如何自定义 AWS ParallelCluster AMI 549
构建自定义 AWS ParallelCluster AMI 549
修改 AWS ParallelCluster AMI 555
集成 Active Directory 557
创建 AD 基础设施 559
（可选）管理 AD 用户和组 574
创建集群 576
以用户身份连接到集群 582
清理 582

使用密 AWS KMS 钥配置共享存储加密 588
创建策略 588
配置和创建集群 589
在多队列模式集群中运行作业 591
配置集群 591
创建集群 593
登录到头节点 594
在多队列模式下运行作业 594
使用 AWS ParallelCluster API 598
使用创建集群 Slurm 会计 612
步骤 1：为创建 VPC 和子网 AWS ParallelCluster 613
步骤 2：创建数据库堆栈 613
步骤 3：使用创建集群 Slurm 已启用记账 613
使用外部集群创建集群 Slurmdbd 会计 614
步骤 1：创建 Slurmdbd 堆栈 615
步骤 2：使用外部集群创建集群 Slurmdbd 已启用 616
恢复到以前的 S AWS ystems Manager 文档版本 617
恢复到以前的 SSM 文档版本 617
使用创建集群 AWS CloudFormation 619
使用 CloudFormation 快速创建堆栈创建集群 619
使用 AWS CloudFormation 命令行界面 (CLI) 创建集群 622
查看 CloudFormation 集群输出 624
访问您的集群 624
清理 625
使用 Terraform 部署 ParallelCluster API 625
定义 Terraform 项目 625
部署 API 628
所需的权限 628
使用 Terraform 创建集群 631
定义 Terraform 项目 632
部署集群 637
所需的权限 639
使用 Terraform 创建自定义 AMI 639
定义 Terraform 项目 640
部署 AMI 643
所需的权限 643

AWS ParallelCluster 用户界面与身份中心集成 644
启用 IAM Identity Center 644
将应用程序添加到 IAM Identity Center 647
使用 Pyxis 运行容器化作业 654
创建集群 655
提交作业 657
创建具有启用 EFA 的 Lustre FSx 的集群 657
要求 657
创建安全组 658
创建文件系统 659
创建集群 660
FSx 使用 EFA 进行验证正在起作用 661
AWS ParallelCluster 故障排除 662
尝试创建集群 663
failureCode 是 OnNodeConfiguredExecutionFailure 663
failureCode 是 OnNodeConfiguredDownloadFailure 663
failureCode 是 OnNodeConfiguredFailure 664
failureCode 是 OnNodeStartExecutionFailure 664
failureCode 是 OnNodeStartDownloadFailure 664
failureCode 是 OnNodeStartFailure 665
failureCode 是 EbsMountFailure 665
failureCode 是 EfsMountFailure 665
failureCode 是 FsxMountFailure 666
failureCode 是 RaidMountFailure 666
failureCode 是 AmiVersionMismatch 666
failureCode 是 InvalidAmi 667
failureCode 是 HeadNodeBootstrapFailure，并且 failureReason 是“无法设置头节点”。 667
failureCode 是 HeadNodeBootstrapFailure，并且 failureReason是“集群创建超时”。 667
failureCode 是 HeadNodeBootstrapFailure，并且 failureReason 是“无法引导头节点”。 668
failureCode 是 ResourceCreationFailure 669
failureCode 是 ClusterCreationFailure 669
WaitCondition timed out...在 CloudFormation 堆栈里看 669
Resource creation cancelled在 CloudFormation 堆栈里看 669

在 AWS CloudFormation 堆栈中看到错误Failed to run cfn-init...或其他错误 669
看到 chef-client.log 以“INFO: Waiting for static fleet capacity provisioning”结束 670
看到Failed to run preinstall or postinstall in cfn-init.log 670
This AMI was created with xxx, but is trying to be used with xxx...在 CloudFormation 堆栈里看 670
This AMI was not baked by AWS ParallelCluster...在 CloudFormation 堆栈里看 670
看到 pcluster create-cluster 命令无法在本地运行 670
其他支持 670
尝试运行作业 670
srun 交互式作业失败并显示错误“srun: error: fwd_tree_thread: can't find address for <host>, check slurm.conf” 671
运行 squeue 命令时，作业卡在 CF 状态 671
运行大型作业并看到“nfsd: too many open connections, consider increasing the number of threads in /var/log/messages” 671
运行 MPI 作业 672
尝试更新集群 673
pcluster update-cluster 命令无法在本地运行 673
使用 pcluster describe-cluster 命令时看到 clusterStatus 为 UPDATE_FAILED 673
集群更新超时 673
尝试访问存储 673
使用外部 Amazon f FSx or Lustre 文件系统 673
使用外部 Amazon Elastic File System 文件系统 673
尝试删除集群 673
pcluster delete-cluster 命令无法在本地运行 674
无法删除集群堆栈 674
尝试升级 AWS ParallelCluster API 堆栈 674
在计算节点初始化过程中看到错误 674
在 clustermgtd.log 中看到“Node bootstrap error” 674
我配置了按需容量预留 (ODCRs) 或区域预留实例 675
运行作业失败时在 slurm_resume.log 中看到“An error occurred (VcpuLimitExceeded)”，或创建集群失败时在 clustermgtd.log 中看到该错误 676

运行作业失败时在 slurm_resume.log 中看到“An error occurred (InsufficientInstanceCapacity)”，或创建集群失败时在 clustermgtd.log 中看到该错误 676
看到节点处于 DOWN 状态并显示Reason (Code:InsufficientInstanceCapacity)... 676
在 slurm_resume.log 中看到“cannot change locale (en_US.utf-8) because it has an invalid name” 676
以上情形都不适用于我的情况 677
集群运行状况指标故障排除 677
看到实例预置错误图表 677
看到运行状况不佳的实例错误图表 679
看到计算实例集空闲时间图表 681
排查集群部署问题 681
查看上 AWS CloudFormation 发生的事件 CREATE_FAILED 682
使用 CLI 查看日志流 684
使用 rollback-on-failure 重新创建失败的集群 686
排查使用 Terraform 部署集群的问题 687
ParallelCluster 未找到 API 687
用户无权调用 ParallelCluster API 688
排查扩展问题 688
用于调试的关键日志 689
运行作业失败时在 slurm_resume.log 中看到“InsufficientInstanceCapacity”错误，或创建集群失败时在 clustermgtd.log 中看到该错误 690
排查节点初始化问题 691
排查意外节点替换和终止问题 693
替换、终止或关闭有问题的实例和节点 695
队列（分区）Inactive 状态 695
排查其他已知的节点和作业问题 695
置放群组和实例启动问题 696
替换目录 696
排查 Amazon DCV 中的问题 696
Amazon DCV 的日志 696
Ubuntu Amazon DCV 问题 697
通过 AWS Batch 集成对集群中的问题进行故障排除 697
头节点问题 697
计算问题 698

作业失败 ... 698
端点 URL 连接超时错误 ... 698
排查与 Active Directory 的多用户集成问题 ... 698
特定于 Active Directory 的问题排查 ... 699
启用调试模式 ... 700
如何从 LDAPS 迁移到 LDAP ... 700
如何禁用 LDAPS 服务器证书验证 ... 700
如何使用 SSH 密钥而不是密码进行登录 ... 701
如何重置用户密码和过期的密码 ... 701
如何验证加入的域 ... 701
如何排查证书问题 ... 702
如何验证与 Active Directory 的集成是否正常工作 ... 704
如何排查计算节点登录问题 ... 704
多用户环境中 SimCenter StarCCM+ 作业的已知问题 ... 705
用户名解析的已知问题 ... 705
如何解决主目录创建问题 ... 705
排查自定义 AMI 问题 ... 706
排查 `cfn-hup` 未运行时的集群更新超时问题 ... 707
网络问题排查 ... 708
集群位于单个公有子网的问题 ... 708
执行 `onNodeUpdated` 自定义操作时集群更新失败 ... 708
看到自定义 Slurm 配置错误 ... 708
集群警报 ... 709
AWS ParallelCluster 支持政策 ... 710
安全性 ... 711
所用服务的安全信息 AWS ParallelCluster ... 711
数据保护 ... 712
数据加密 ... 713
另请参阅 ... 714
身份和访问管理 ... 714
合规性验证 ... 715
强制执行 TLS 1.2 ... 715
确定当前支持的协议 ... 715
编译 OpenSSL 和 Python ... 717
支持的 AWS 区域 ... 719
发布说明和文档历史记录 ... 721

..... dcccxix

什么是 AWS ParallelCluster

AWS ParallelCluster 是一款 AWS 受支持的开源集群管理工具，可帮助您在中部署和管理高性能计算 (HPC) 集群。 AWS Cloud它自动设置所需的计算资源、调度器和共享文件系统。你可以 AWS ParallelCluster 与 AWS Batch 和一起使用 Slurm 调度器。

借 AWS ParallelCluster助，您可以快速构建和部署概念验证和生产 HPC 计算环境。您还可以在此基础上构建和部署高级工作流程 AWS ParallelCluster，例如可自动执行整个 DNA 测序工作流程的基因组学门户。

您可以使用以下 AWS ParallelCluster 方法进行访问：

- AWS ParallelCluster 命令行界面 (CLI)
- AWS ParallelCluster API
- PCUI（从版本 3.5.0 开始添加了此功能）
- AWS ParallelCluster Python 库 API（从版本 3.5.0 开始添加了此功能）
- 作为 AWS CloudFormation 自定义资源（从版本 3.6.0 开始添加了此功能）

定价

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

如何 AWS ParallelCluster 运作

AWS ParallelCluster 不仅是为了管理集群而构建的，也是作为如何使用 AWS 服务构建 HPC 环境的参考。以下主题描述了 AWS ParallelCluster 流程、 AWS ParallelCluster 使用方法和方式以及内部目录。 AWS 服务

主题

- AWS ParallelCluster 进程
- AWS 使用的服务 AWS ParallelCluster
- AWS ParallelCluster 内部目录

AWS ParallelCluster 进程

本节适用于使用部署的集群 Slurm。 与该调度器一起使用时，可与底层作业调度器 AWS ParallelCluster 交互以管理计算节点的配置和移除。

对于基于的 HPC 集群 AWS Batch，依 AWS ParallelCluster 靠提供的功能 AWS Batch 来管理计算节点。

`clustermgtd`

集群管理守护程序 (`clustermgtd`) 执行以下任务：

- 清理非活动分区
- 管理 Slurm 与容量块关联的预留和节点（请参阅以下部分）
- 管理静态容量，确保其始终处于正常运行状态
- 将计划程序与 Amazon EC2 同步。
- 清理孤立实例
- 在暂停工作流程之外发生 Amazon EC2 终止时恢复计划程序节点状态
- 管理运行状况不佳的 Amazon EC2 实例（未通过亚马逊运行 EC2 状况检查的实例）
- 管理定期维护事件
- 管理不健康的调度器节点（调度器运行状况检查失败的节点）

的管理 Slurm 与容量块关联的预留和节点

ParallelCluster 支持按需容量预留 (ODCR) 和 Machine Learning 容量块 (CB)。与 ODCR 不同，CB 可以有未来的开始时间，并且有时间限制。

`clustermgtd`在循环中搜索运行状况不佳的节点，终止所有已关闭的 Amazon EC2 实例，如果它们是静态节点，则将其替换为新实例。

AWS ParallelCluster 以不同的方式管理与容量块关联的静态节点 — 即使 CB 尚未处于活动状态，它也会创建集群，并在 CB 处于活动状态后自动启动实例。

这些区域有：Slurm 与尚未激活的计算资源对应的节点将保持维护状态，直到到达 CB 启动时间。 CBs 这些 Slurm 节点仍处于与之关联的预留/维护状态 Slurm 管理员用户，这意味着他们可以接受作业，但任务一直处于待处理状态，直到 Slurm 已取消预订。

`clustermgtd`自动创建或删除 Slurm 保留— 它根据 CB 状态将相关的 CB 节点置于维护状态。当 CB 处于活动状态时，Slurm 预留被移除，节点启动并可用于待处理的任务或提交的新作业。

当到达 CB 结束时间时，当 CB 不再处于活动状态并且实例终止时，节点将移回reservation/maintenance state. It's up to users to resubmit/requeue the jobs to a new queue/compute资源。

`clusterstatusmgtd`

集群状态管理守护程序 (`clusterstatusmgtd`) 管理计算队列状态更新。它每分钟获取一次存储在 DynamoDB 表中的实例集状态并管理所有停止/启动请求。

`computemgtd`

计算管理守护程序 (`computemgtd`) 进程在每个群集计算节点上运行。每隔五 (5) 分钟，计算管理进程守护程序就会确认头节点可以访问并且运行正常。如果在五 (5) 分钟内无法访问头节点或头节点运行状况不佳，则将关闭计算节点。

AWS 使用的服务 AWS ParallelCluster

使用以下亚马逊 Web Services (AWS) 服务 AWS ParallelCluster。

主题

- Amazon API Gateway
- AWS Batch

- AWS CloudFormation
- Amazon CloudWatch
- 亚马逊 CloudWatch 活动
- Amazon CloudWatch 日志
- AWS CodeBuild
- Amazon DynamoDB
- Amazon Elastic Block Store
- Amazon Elastic Compute Cloud
- Amazon Elastic Container Registry
- Amazon EFS
- 亚马逊 f FSx or Lustre
- FSx 适用于 NetApp ONTAP 的亚马逊
- FSx 适用于 OpenZFS 的亚马逊
- AWS Identity and Access Management
- AWS Lambda
- Amazon RDS
- Amazon Route 53
- Amazon Simple Notification Service
- Amazon Simple Storage Service
- Amazon VPC
- Elastic Fabric Adapter
- EC2 Image Builder
- Amazon DCV

Amazon API Gateway

Amazon API Gateway 是一项可以创建、发布、维护、监控和保护任何规模的 REST、HTTP WebSocket APIs 的 AWS 服务

AWS ParallelCluster 使用 API Gateway 来托管 AWS ParallelCluster API。

有关 Amazon API Gateway 的更多信息，请参阅 https://aws.amazon.com/api-gateway/和 https://docs.aws.amazon.com/apigateway/。

AWS Batch

AWS Batch 是一项 AWS 托管作业调度器服务。它可以动态配置集群中计算资源的最佳数量和类型（例如 CPU 或内存优化型实例）。 AWS Batch 这些资源是根据批处理作业的特定要求（包括卷要求）预置的。使用 AWS Batch，您无需安装或管理额外的批处理计算软件或服务器集群即可有效地运行作业。

AWS Batch 仅与 AWS Batch 群集一起使用。

有关的更多信息 AWS Batch，请参见https://aws.amazon.com/batch/和https://docs.aws.amazon.com/batch/。

AWS CloudFormation

AWS CloudFormation 是一项为云环境中的第三方应用程序资源建模 AWS 和配置的通用语言的 infrastructure-as-code服务。它是使用的主要服务 AWS ParallelCluster。中的每个集群 AWS ParallelCluster 都表示为一个堆栈，每个群集所需的所有资源都在 AWS ParallelCluster CloudFormation 模板中定义。在大多数情况下， AWS ParallelCluster CLI 命令直接对应 AWS CloudFormation 堆栈命令，例如创建、更新和删除。在集群内启动的实例会对启动集群的 AWS CloudFormation 终端节点进行 AWS 区域 HTTPS 调用。

有关的更多信息 AWS CloudFormation，请参见 https://aws.amazon.com/cloudformation/和 https://docs.aws.amazon.com/cloudformation/。

Amazon CloudWatch

Amazon CloudWatch (CloudWatch) 是一项监控和可观察性服务，可为您提供数据和可操作的见解。这些见解可用于监控您的应用程序、响应性能变化和服务异常以及优化资源利用率。中 AWS ParallelCluster， CloudWatch 用于仪表板，用于监视和记录 Docker 映像构建步骤和 AWS Batch 作业输出。

在 2.10.0 AWS ParallelCluster 版本之前 CloudWatch ，仅用于集群。 AWS Batch

有关的更多信息 CloudWatch，请参见 https://aws.amazon.com/cloudwatch/和 https://docs.aws.amazon.com/cloudwatch/。

亚马逊 CloudWatch 活动

Amazon CloudWatch Ev CloudWatch ents（Events）提供近乎实时的系统事件流，这些事件描述了亚马逊 Web Services (AWS) 资源的变化。通过使用可快速设置的简单规则，您可以匹配事件并将事件路由到一个或多个目标函数或流。在中 AWS ParallelCluster， CloudWatch 事件用于 AWS Batch 作业。

有关 CloudWatch 事件的更多信息，请参阅 https://docs.aws.amazon.com//eventbridge/latest/userguide/eb-cwe-now-eb。

Amazon CloudWatch 日志

亚马逊 CloudWatch 日CloudWatch 志（日志）是亚马逊的核心功能之一 CloudWatch。您可以使用它来监控、存储、查看和搜索 AWS ParallelCluster中使用的众多组件的日志文件。

在 2.6.0 AWS ParallelCluster 版本之前， CloudWatch 日志仅用于集群。 AWS Batch

有关更多信息，请参阅 与 Amazon CloudWatch 日志集成。

AWS CodeBuild

AWS CodeBuild (CodeBuild) 是一项 AWS 托管的持续集成服务，用于编译源代码、运行测试和生成可随时部署的软件包。中 AWS ParallelCluster， CodeBuild 用于在创建集群时自动透明地构建 Docker 镜像。

CodeBuild 仅与 AWS Batch 群集一起使用。

有关的更多信息 CodeBuild，请参见https://aws.amazon.com/codebuild/和https://docs.aws.amazon.com/codebuild/。

Amazon DynamoDB

Amazon DynamoDB (DynamoDB) 是一项快速灵活的 NoSQL 数据库服务。它用于存储集群的最小状态信息。头节点跟踪 DynamoDB 表中的预置实例。

DynamoDB 不适用于集群。 AWS Batch

有关 DynamoDB 的更多信息，请参阅和。https://aws.amazon.com/dynamodb/https://docs.aws.amazon.com/dynamodb/

Amazon Elastic Block Store

Amazon Elastic Block Store (Amazon EBS) 是一项高性能块存储服务，可为共享卷提供永久性存储。所有 Amazon EBS 设置都可以通过配置进行传递。Amazon EBS 卷可以初始化为空，也可以从现有的 Amazon EBS 快照进行初始化。

有关 Amazon EBS 的更多信息，请参阅https://aws.amazon.com/ebs/和。https://docs.aws.amazon.com/ebs/

Amazon Elastic Compute Cloud

亚马逊弹性计算云 (Amazon EC2) 为提供计算能力 AWS ParallelCluster。头节点和计算节点是 Amazon EC2 实例。可以选择任何支持硬件虚拟机 (HVM) 的实例类型。头节点和计算节点可以是不同的实例类型。此外，如果使用多个队列，则部分或全部计算节点也可以作为竞价型实例启动。在实例上找到的实例存储卷将作为条带逻辑卷管理器 (LVM) 卷装载。

有关 Amazon 的更多信息 EC2 ，请参阅https://aws.amazon.com/ec2/和https://docs.aws.amazon.com/ec2/。

Amazon Elastic Container Registry

Amazon Elastic Container Registry（Amazon ECR）是一个完全托管式 Docker 容器注册表，可让开发人员轻松地存储、管理和部署 Docker 容器映像。在中 AWS ParallelCluster，Amazon ECR 存储了创建集群时生成的 Docker 镜像。然后，使用 Docker 镜像为提交的作业运行容器。 AWS Batch

Amazon ECR 仅适用于 AWS Batch 集群。

有关更多信息，请参阅https://aws.amazon.com/ecr/和https://docs.aws.amazon.com/ecr/。

Amazon EFS

Amazon Elastic File System (Amazon EFS) 提供了一种简单、可扩展并且完全托管的弹性 NFS 文件系统，可用于 AWS Cloud 服务和本地资源。当指定了 `EfsSettings` 时，将会使用 Amazon EFS。2.1.0 AWS ParallelCluster 版本中增加了对 Amazon EFS 的支持。

有关 Amazon EFS 的更多信息，请参阅https://aws.amazon.com/efs/和https://docs.aws.amazon.com/efs/。

亚马逊 f FSx or Lustre

FSx for Lustre 提供了一个使用开源 Lustre 文件系统的高性能文件系统。 FSx for Lustre 在指定`FsxLustreSettings 属性`时使用。在 2.2.1 AWS ParallelCluster 版本中增加了对 Lustre 的支持。FSx

有关 Lustre FSx 的更多信息，请参阅 lust https://aws.amazon.com/fsx/re/ 和。https://docs.aws.amazon.com/fsx/

FSx 适用于 NetApp ONTAP 的亚马逊

FSx for ONTAP 提供了一个完全托管的共享存储系统，该系统基于广受欢迎 NetApp的 ONTAP 文件系统构建。 FSx for ONTAP 在指定`FsxOntapSettings` 属性时使用。3.2.0 FSx AWS ParallelCluster 版本中增加了对 ONTAP 的支持。

有关 ONTAP 的更多信息 FSx ，请参阅 n https://aws.amazon.com/fsx/etapp-ontap/ 和。https://docs.aws.amazon.com/fsx/

FSx 适用于 OpenZFS 的亚马逊

FSx for OpenZFS 提供了一个完全托管的共享存储系统，该系统基于流行的 OpenZFS 文件系统构建。 FSx for openZFS 在指定`FsxOpenZfsSettings` 属性时使用。3.2.0 FSx AWS ParallelCluster 版本中增加了对 OpenZFS 的支持。

有关 OpenZFS FSx 的更多信息，请参阅 https://aws.amazon.com/fsx/ openzfs/ 和。https://docs.aws.amazon.com/fsx/

AWS Identity and Access Management

AWS Identity and Access Management (IAM) 在中 AWS ParallelCluster 用于为 Amazon EC2 提供特定于每个集群的实例的最低权限的 IAM 角色。 AWS ParallelCluster 实例只能访问部署和管理集群所需的特定 API 调用。

对于 AWS Batch 集群，还会为创建集群时与 Docker 映像构建过程相关的组件创建 IAM 角色。这些组件包括允许在 Amazon ECR 存储库中添加和删除 Docker 映像的 Lambda 函数。它们还包括允许删除为集群和 CodeBuild 项目创建的 Amazon S3 存储桶的功能。还有 AWS Batch 资源、实例和作业的角色。

有关 IAM 的更多信息，请参阅https://aws.amazon.com/iam/和https://docs.aws.amazon.com/iam/。

AWS Lambda

AWS Lambda (Lambda) 运行编排 Docker 镜像创建的函数。Lambda 还管理自定义集群资源的清理，如 Amazon ECR 存储库中和 Amazon S3 上存储的 Docker 映像。

有关 Lambda 的更多信息，请参阅https://aws.amazon.com/lambda/和。https://docs.aws.amazon.com/lambda/

Amazon RDS

Amazon Relational Database Service (Amazon RDS) 是一项网络服务，可以更轻松地在 AWS 云中设置、操作和扩展关系数据库。

AWS ParallelCluster 将 Amazon RDS 用于 AWS Batch 和 Slurm.

有关 Amazon RDS 的更多信息，请参阅https://aws.amazon.com/rds/和https://docs.aws.amazon.com/rds/。

Amazon Route 53

Amazon Route 53 (Route 53) 用于使用每个计算节点的主机名和完全限定域名创建托管区。

有关 Route 53 的更多信息，请参阅https://aws.amazon.com/route53/和https://docs.aws.amazon.com/route53/。

Amazon Simple Notification Service

Amazon Simple Notification Service (Amazon SNS) 是一项托管服务，提供从发布者向订阅用户（也称为创建者和使用者）的消息传输。

AWS ParallelCluster 使用亚马逊 SNS 进行 API 托管。

有关 Amazon SNS 的更多信息，请参阅https://aws.amazon.com/sns/和。https://docs.aws.amazon.com/sns/

Amazon Simple Storage Service

亚马逊简单存储服务 (Amazon S3) Service 存储的模板位于 AWS ParallelCluster 每个服务中。 AWS 区域 AWS ParallelCluster 可以配置为允许 CLI/SDK 工具使用 Amazon S3。

AWS ParallelCluster 还会在您的 AWS 账户 中创建一个 Amazon S3 存储桶，用于存储集群使用的资源，例如集群配置文件。 AWS ParallelCluster 在您创建集群的每个存储桶中 AWS 区域 都保留一个 Amazon S3 存储桶。

当您使用 AWS Batch 集群时，将使用您账户中的 Amazon S3 存储桶来存储相关数据。例如，该存储桶会存储根据提交的作业创建 Docker 映像和脚本时创建的构件。

有关更多信息，请参阅https://aws.amazon.com/s3/和https://docs.aws.amazon.com/s3/。

Amazon VPC

亚马逊虚拟私有云 (VPC) Virtual Private Cloud 定义了集群中节点使用的网络。

有关 Amazon VPC 的更多信息，请参阅https://aws.amazon.com/vpc/和https://docs.aws.amazon.com/vpc/。

Elastic Fabric Adapter

Elastic Fabric Adapter (EFA) 是一个用于实例的网络接口，您可以使用它来大规模运行需要高水平节点间通信的应用程序。 AWS

有关弹性结构适配器的更多信息，请参阅 https://aws.amazon.com/hpc/efa/。

EC2 Image Builder

EC2 Image Builder 是一项完全托管的 AWS 服务，可帮助您自动创建、管理和部署自定义、安全的 up-to-date服务器映像。

AWS ParallelCluster 使用 Image Builder 创建和管理 AWS ParallelCluster 图像。

有关 EC2 Image Builder 的更多信息，请参见https://aws.amazon.com/image-builder/和https://docs.aws.amazon.com/imagebuilder/。

Amazon DCV

Amazon DCV 是一种高性能远程显示协议，它是一种可在不同网络条件下向任何设备提供远程桌面和应用程序流的安全方式。当指定了 `HeadNode` 部分/`Dcv` 设置时，将会使用 Amazon DCV。2.5.0 AWS ParallelCluster 版本中增加了对 Amazon DCV 的支持。

有关 Amazon DCV 的更多信息，请参阅 https://aws.amazon.com/hpc/dcv/ 和。https://docs.aws.amazon.com/dcv/

AWS ParallelCluster 内部目录

有几个内部目录 AWS ParallelCluster 用于在集群内共享数据。头节点、计算节点和登录节点之间共享以下目录：

- `/opt/slurm`
- `/opt/intel`

- `/opt/parallelcluster/shared (only with compute nodes)`
- `/opt/parallelcluster/shared_login_nodes (only with login nodes)`
- `/home (unless specified in SharedStorage)`

> **Note**
>
> 默认情况下，这些目录创建在头节点 EBS 卷上，并作为 NFS 导出共享给计算节点和登录节点。从 AWS ParallelCluster 3.8 开始，您可以通过 AWS ParallelCluster 将SharedStorageType参数设置为 efs 来创建和管理 Amazon EFS 文件系统来托管和共享这些目录。
>
> 当集群横向扩展时，通过 EBS 卷的 NFS 导出可能会造成性能瓶颈。使用 EFS，您可以在集群扩展时避免 NFS 导出，并避免与之相关的性能瓶颈。

设置 AWS ParallelCluster

以下主题描述了如何设置 AWS ParallelCluster。您将了解如何安装必要的工具、如何使用这些工具、如何实施和管理多个用户对集群的访问以及最佳实践。

主题

- 先决条件
- 安装 AWS ParallelCluster 命令行界面 (CLI)
- 安装后需要执行的步骤
- 安装 PCUI
- 入门 AWS ParallelCluster
- 集群的多用户访问
- 最佳实践
- 从 AWS ParallelCluster 2.x 升级到 3.x

先决条件

在开始设置和使用之前 AWS ParallelCluster，请确保您已完成以下先决条件。

设置一个 AWS 账户

设置要使用的 AWS 账户 AWS ParallelCluster。

注册获取 AWS 账户

如果您没有 AWS 账户，请完成以下步骤来创建一个。

要注册 AWS 账户

1. 打开https://portal.aws.amazon.com/billing/注册。
2. 按照屏幕上的说明操作。

   在注册时，将接到电话或收到短信，要求使用电话键盘输入一个验证码。

   当您注册时 AWS 账户，就会创建AWS 账户根用户一个。根用户有权访问该账户中的所有 AWS 服务 和资源。作为最佳安全实践，请为用户分配管理访问权限，并且只使用根用户来执行需要根用户访问权限的任务。

AWS 注册过程完成后会向您发送一封确认电子邮件。您可以随时前往 https://aws.amazon.com/并选择“我的账户”，查看您当前的账户活动并管理您的账户。

创建具有管理访问权限的用户

注册后，请保护您的安全 AWS 账户 AWS 账户根用户 AWS IAM Identity Center，启用并创建管理用户，这样您就不会使用 root 用户执行日常任务。

保护你的 AWS 账户根用户

1. 选择 Root 用户并输入您的 AWS 账户 电子邮件地址，以账户所有者的身份登录。AWS Management Console在下一页上，输入您的密码。

   要获取使用根用户登录方面的帮助，请参阅《AWS 登录 用户指南》中的 Signing in as the root user。
2. 为您的根用户启用多重身份验证（MFA）。

   有关说明，请参阅 I A M 用户指南中的为 AWS 账户 根用户启用虚拟 MFA 设备（控制台）。

创建具有管理访问权限的用户

1. 启用 IAM Identity Center。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Enabling AWS IAM Identity Center。
2. 在 IAM Identity Center 中，为用户授予管理访问权限。

   有关使用 IAM Identity Center 目录 作为身份源的教程，请参阅《用户指南》 IAM Identity Center 目录中的使用默认设置配置AWS IAM Identity Center 用户访问权限。

以具有管理访问权限的用户身份登录

- 要使用您的 IAM Identity Center 用户身份登录，请使用您在创建 IAM Identity Center 用户时发送到您的电子邮件地址的登录网址。

  有关使用 IAM Identity Center 用户登录的帮助，请参阅AWS 登录 用户指南中的登录 AWS 访问门户。

将访问权限分配给其他用户

1. 在 IAM Identity Center 中，创建一个权限集，该权限集遵循应用最低权限的最佳做法。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Create a permission set。
2. 将用户分配到一个组，然后为该组分配单点登录访问权限。

   有关说明，请参阅《AWS IAM Identity Center 用户指南》中的 Add groups。

创建密钥对

要部署集群，请 AWS ParallelCluster 启动 Amazon EC2 实例以创建集群头节点和计算节点。要执行集群任务，例如运行和监控作业或管理用户，您必须能够访问集群头节点。要验证您可以使用 SSH 访问头节点实例，必须使用 Amazon EC2 密钥对。要了解如何创建密钥对，请参阅 Amazon Elastic Compute Cloud 用户指南（适用于 Linux 实例）中的创建密钥对。

安装 AWS ParallelCluster 命令行界面 (CLI)

AWS ParallelCluster 作为 Python 包分发，使用 Python `pip` 包管理器进行安装。有关如何安装 Python 程序包的说明，请参阅 Python Packaging User Guide 中的 Installing packages。

安装方式 AWS ParallelCluster：

- AWS ParallelCluster 在虚拟环境中安装（推荐）
- 使用 pip AWS ParallelCluster 在非虚拟环境中安装
- AWS ParallelCluster 作为独立应用程序安装

您可以在的发布页面上找到最新 CLI 的版本号 GitHub。在本指南中，命令示例假定您安装的是高于版本 3.6 的 Python 版本。`pip` 命令示例使用 `pip3` 版本。

同时管理 AWS ParallelCluster 2 和 AWS ParallelCluster 3

如果您同时使用 AWS ParallelCluster 2 和 AWS ParallelCluster 3，并且想要管理这两个软件包，我们建议您在不同的虚拟环境中安装 AWS ParallelCluster 2 和 AWS ParallelCluster 3。 CLIs 这样可以确保您能够继续使用每个版本的 AWS ParallelCluster 及关联的任何集群资源。

AWS ParallelCluster 在虚拟环境中安装（推荐）

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

先决条件

- AWS ParallelCluster 需要 Python 3.7 或更高版本。如果尚未安装该软件，请在 python.org 上针对您的平台下载兼容版本。

AWS ParallelCluster 在虚拟环境中安装

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

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

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

2. 创建虚拟环境并命名它。

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

   或者，您也可以使用 `-p` 选项指定特定的 Python 版本。

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

3. 激活新虚拟环境。

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

4. 安装 AWS ParallelCluster 到您的虚拟环境中。

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

5. 安装 Node Version Manager 和最新的长期支持 (LTS) Node.js 版本。 AWS Cloud Development Kit (AWS CDK) 需要使用 Node.js CloudFormation 来生成模板。

> **Note**
>
> 如果 Node.js 安装无法在您的平台上运行，您可以安装最新 LTS 版本之前的 LTS 版本。有关更多信息，请参阅 Node.js 发布时间表和 AWS CDK 先决条件。
>
> Node.js 安装命令示例：

```
$  nvm install --lts=Hydrogen
```

```
$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
$ chmod ug+x ~/.nvm/nvm.sh
$ source ~/.nvm/nvm.sh
$ nvm install --lts
$ node --version
```

6. 验证 AWS ParallelCluster 是否已正确安装。

```
$ pcluster version
{
  "version": "3.13.2"
}
```

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

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

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

使用 pip AWS ParallelCluster 在非虚拟环境中安装

你也可以使用 pip（Python 包的包管理器）AWS ParallelCluster 在非虚拟环境中进行安装。

先决条件

- AWS ParallelCluster 需要 Python 3.7 或更高版本。如果尚未安装该软件，请在 python.org 上针对您的平台下载兼容版本。

安装 AWS ParallelCluster

1. `pip`用于安装 AWS ParallelCluster。

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

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

2. 安装 Node Version Manager 和最新的长期支持 (LTS) Node.js 版本。 AWS Cloud Development Kit (AWS CDK) 需要使用 Node.js CloudFormation 来生成模板。

> **Note**
>
> 如果 Node.js 安装无法在您的平台上运行，您可以安装最新 LTS 版本之前的 LTS 版本。有关更多信息，请参阅 Node.js 发布时间表和 AWS CDK 先决条件。
>
> ```
> $  nvm install --lts=Hydrogen
> ```

```
$ curl -o- https://raw.githubusercontent.com/nvm-sh/nvm/v0.38.0/install.sh | bash
$ chmod ug+x ~/.nvm/nvm.sh
$ source ~/.nvm/nvm.sh
$ nvm install --lts
$ node --version
```

3. 验证 AWS ParallelCluster 安装是否正确。

```
$ pcluster version
{
  "version": "3.13.2"
}
```

4. 要升级到最新版本，请重新运行安装命令。

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

AWS ParallelCluster 作为独立应用程序安装

AWS ParallelCluster 作为独立应用程序安装在您的环境中。按照下一节中有关在可用操作系统 AWS ParallelCluster 上安装的说明进行操作。

先决条件

- 操作系统与安装程序的可用版本兼容的环境。

> **Note**
>
> AWS ParallelCluster 需要 NodeJS。 AWS ParallelCluster 安装程序包含 NodeJS (v18) 的捆绑版本，如果尚不存在，则安装该版本。如果您的系统与 NodeJS v18 不兼容，则应在安装之前安装 NodeJS。 AWS ParallelCluster

Linux

Linux x86 (64-bit)

AWS ParallelCluster 在您的环境中安装。

1. 下载最新的 pcluster 安装程序（校验和）。
2. 解压缩安装包并使用以下命令 AWS ParallelCluster 进行安装：

```
$ unzip pcluster-installer-bundle-<VERSION>-Linux_x86_64-signed.zip -d pcluster-installer-bundle
$ cd pcluster-installer-bundle
$ chmod +x install_pcluster.sh
```

3. 运行以下安装脚本。

```
$ bash install_pcluster.sh
```

4. 验证 AWS ParallelCluster 是否已正确安装。

```
$ pcluster version
{
    "version": "3.13.2"
}
```

排查 **pcluster** 安装错误

- 如果步骤 4 中未返回 AWS ParallelCluster 版本，请重新启动终端或`sourcebash_profile`以更新`PATH`变量以包含新的二进制目录，如以下示例所示：

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

- 如果您使用`pcluster`安装来创建`CustomActions`指定为 HTTPS 资源（而不是 S3）的集群 URIs，则可能会看到一条`WARNING`消息，指出这些资源可能未经过验证 ([`SSL: CERTIFICATE_VERIFY_FAILED`])。这是由已知问题引起的，如果您信任指定资源的真实性，则可以忽略此警告。

以前的安装程序捆绑包版本

- 无

安装后需要执行的步骤

本节假设你已经安装了 AWS ParallelCluster。您将学习如何验证已正确 AWS ParallelCluster 安装、如何更新到最新版本以及如何卸载。 AWS ParallelCluster

您可以通过运行来 AWS ParallelCluster 验证安装是否正确`pcluster version`。

```
$ pcluster version
{
"version": "3.13.2"
}
```

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

```
$ pip3 install aws-parallelcluster --upgrade --user
```

要卸载 AWS ParallelCluster，请使用`pip3 uninstall`。

```
$ pip3 uninstall aws-parallelcluster
```

如果您没有 Python 和 `pip3`，则使用适合您的环境的过程。

安装 PCUI

用户界 AWS ParallelCluster 面 (PCUI) 是一个基于 Web 的用户界面，它镜像 AWS ParallelCluster `pcluster` CLI，同时提供类似于控制台的体验。您可以在中安装并访问 PCUI。 AWS 账户当你运行它时，PCU AWS ParallelCluster I 会访问你的 Amazon API Gateway 上托管的 API 实例。 AWS 账户有关 PCUI 的更多信息，请参阅 AWS ParallelCluster 用户界面。

先决条件：

- 你必须有一个 AWS 账户
- 您必须有权访问 AWS Management Console

有关更多信息，请参阅 设置一个 AWS 账户。

主题

- 安装 PCUI
- 堆栈参数
- 配置自定义域
- Amazon Cognito 用户群体选项
- 识别 AWS ParallelCluster 和 PCUI 版本
- PCUI 成本

安装 PCUI

要安装 AWS ParallelCluster 用户界面 (PCUI) 的实例，请选择用于 AWS CloudFormation 创建集群 AWS 区域 的快速创建链接。该快速创建 URL 会将您引导至创建堆栈向导，您可以在其中提供快速创建堆栈模板输入并部署堆栈。有关 CloudFormation 快速创建堆栈的更多信息，请参阅《用户指南》中的为堆栈创建快速创建链接。AWS CloudFormation

> **Note**
>
> 您只能使用与安装 PCUI 相同的 AWS ParallelCluster 版本创建和编辑集群或构建映像。

使用 AWS CloudFormation 快速创建链接部署包含嵌套的 Amazon Cognito、API Gateway 和 Amazon Systems EC2 Manager 堆栈的 PCUI 堆栈。

1. 登录到 AWS Management Console。
2. 通过从此处的列表中选择 AWS 区域 快速创建链接来部署 PCUI。这将带您进入控制台中的 “CloudFormation 创建堆栈向导”。
   - us-east-1
   - us-east-2

- us-west-1
- us-west-2
- eu-west-1
- eu-west-2
- eu-west-3
- eu-central-1
- eu-north-1
- me-south-1
- sa-east-1
- ca-central-1
- ap-northeast-1
- ap-northeast-2
- ap-south-1
- ap-southeast-1
- ap-southeast-2
- us-gov-west-1

3. 输入有效的管理员电子邮件地址和ParallelCluster 版本。

   部署成功完成后，PCUI 将向该电子邮件地址发送一个临时密码，您可以使用该密码访问 PCUI。如果您在保存或使用该临时密码之前删除了电子邮件，则必须删除堆栈并重新安装 PCUI。
4. 将表单的其余部分留空或输入（可选）参数的值以自定义 PCUI 构建。
5. 记下堆栈名称，以供后续步骤中使用。
6. 导航至功能。同意这些 CloudFormation 能力。
7. 选择创建。完成 AWS ParallelCluster API 和 PCUI 部署大约需要 15 分钟。
8. 创建堆栈后查看堆栈的详细信息。
9. 部署完成后，打开发送到您输入的地址且包含临时密码的管理员电子邮件。使用它来访问 PCUI。（请记住，如果您在登录 PCUI 之前永久删除了电子邮件，则必须删除您创建的 PCUI 堆栈并重新安装 PCUI。
10. 在 AWS CloudFormation 控制台堆栈列表中，选择您在上一步中记下的堆栈名称的链接。
11. 在堆栈详细信息中，选择输出，然后选择名为 ***Stackname***URL 的密钥的链接以打开 PCUI（其中***Stackname***是您在上一步中记下的名称）。
12. 输入临时密码。按照步骤创建自己的密码并重新登录。

13. 现在，您已进入所选的 PCUI AWS 区域 的主页。
14. 要开始使用 PCUI，请参阅使用 PCUI 配置和创建集群。

> **Note**
>
> PCUI 会话的默认持续时间为 5 分钟，这是 Cognito 在 PCUI 2023.12.0 之前提供的最小值。因此，预计在会话到期之前，从 Cognito 用户池中移除的用户仍能访问系统。

堆栈参数

AdditionalPoliciesPCAPI：

描述：（可选）要附加到 Lambda 函数 ParallelCluster 默认执行角色的其他 IAM 策略的 ARN。只能指定一个策略。

类型：字符串

默认：”

AllowedPattern: “^ (arn:. *: iam::。 *: policy\\/([a-za-z0-9_-] +)) | () $”

AdminUserEmail:

描述：默认情况下要设置的管理用户的电子邮件地址（仅适用于新的 Cognito 实例）。

类型：字符串

默认：”

CognitoCustomDomain:

描述：（可选）Cognito 的自定义域名。如果省略，则将使用默认 Cognito 域名。

类型：字符串

默认：”

CognitoCustomDomainCertificateArn:

描述：（可选）为 Cognito 自定义域名颁发的 ACM 证书的 ARN。只有在指定时`CognitoCustomDomain`才需要这样做。

类型：字符串

默认：”

CustomDomain:

描述：（可选）自定义域名。如果省略，则将使用默认域名。

类型：字符串

默认：”

CustomDomainCertificateArn:

描述：（可选）为自定义域颁发的 ACM 证书的 ARN。只有在指定时`CustomDomain`才需要这样做。

类型：字符串

默认：”

IAMRoleAndPolicyPrefix:

描述：应用于每个 IAM 角色和策略名称的前缀（最大长度：10）。[ParallelCluster >= 3.8.0]

类型：字符串

默认：”

MaxLength: 10

ImageBuilderSubnetId:

描述：（可选）选择用于构建容器镜像的子网。子网必须为公有子网并自动分配为公 IPs有子网。如果未选中，则将使用默认子网。

类型：字符串

默认：”

ImageBuilderVpcId:

描述：（可选）选择用于构建容器镜像的 VPC。如果未选中，则将使用默认 VPC。

类型：字符串

默认：”

InfrastructureBucket:

描述：（可选）存储 CloudFormation 文件的 S3 存储桶。只有在测试对基础架构本身所做的更改时才更改此参数。

类型：字符串

默认：”

LambdaSecurityGroupIds:

描述：要与 PCUI Lambda 函数关联的安全组列表，以逗号分隔。

类型： CommaDelimitedList

默认：”

LambdaSubnetIds:

描述：要与 PCUI Lambda 函数关联的子网列表， IDs 以逗号分隔。这些子网应为私有子网，并与您的 VPC 终端节点相关联。

类型： CommaDelimitedList

默认：”

PermissionsBoundaryPolicy:

描述：用作 ParallelCluster UI 基础设施创建的每个 IAM 角色的权限边界的 IAM 策略的 ARN。”

类型：字符串

默认：”

AllowedPattern: “^ (arn:. *: iam::。 *: policy\\/([a-za-z0-9_-] +)) | () $”

PermissionsBoundaryPolicyPCAPI：

描述：用作 ParallelCluster API 基础设施创建的每个 IAM 角色的权限边界的 IAM 策略的 ARN。[ParallelCluster >= 3.8.0]

类型：字符串

默认：”

AllowedPattern: “^ (arn:. *: iam::。 *: policy\\/([a-za-z0-9_-] +)) | () $”

PublicEcrImageUri:

描述：如果指定，则指定 UI 容器的 Lambda 的 Docker 镜像 ParallelCluster 的 URI。

类型：字符串

默认：public.ecr。 aws/pcm/parallelcluster-ui: 2024.11.0

SNSRole:

描述：之前部署的 PCUI Cognito 堆栈的 SNSRole ARN。留空可创建一个新的。

类型：字符串

默认：”

UserPoolAuthDomain:

描述：之前部署 UserPoolAuthDomain 的 PCUI Cognito 用户池。留空可创建一个新的。

类型：字符串

默认：”

UserPoolId:

描述：之前部署 UserPoolId 的 PCUI Cognito 用户池。留空可创建一个新的。

类型：字符串

默认：”

版本：

描述： AWS ParallelCluster 要部署的版本。

类型：字符串

默认值：3.11.1

VpcEndpointId:

描述：输入带有 execute-API 服务类型接口的 VPC 终端节点，以启用私有 PCUI 实现。启用后，API 将仅接受来自给定 VPC 内部的请求。

类型：字符串

默认：”

配置自定义域

了解如何为 AWS ParallelCluster 用户界面 (PCUI) 配置自定义网域。UI 托管在您的 AWS 账户中的 Amazon API Gateway 上。您可以在 API Gateway 控制台中配置自定义域。

先决条件：

- 你有一个 AWS 账户.
- 您拥有域。
- 域根目录是可解析的。例如，如果您想`xyz.example.com`用作自定义域，则根域`example.com`必须是可解析的。这意味着您必须至少有 A 记录才能`example.com`指向 IP。如果你没有这样的记录，请添加一个指向 0.0.0.0 的虚拟记录。
- 您已在 us-east-1 中为 Cognito 创建自定义域名证书或将其导入到 AWS Certificate Manager (ACM) 中。
- 您拥有更改域名系统（DNS）设置的必要权限。

主题

- 部署 PCUI
- 配置 DNS

部署 PCUI

完成以下步骤以创建或更新您的 AWS ParallelCluster UI (PCUI) 部署，使其使用您的自定义域。

在此示例中，我们假设您要使用自定义域 `xyz.example.com` 部署 PCUI，并使用自定义域 `auth-xyz.example.com` 部署 Amazon Cognito 接口。

> **Note**
>
> 请注意，自定义 Amazon Cognito 域不是必需的，可以保留默认值。为了完整起见，我们将此自定义设置包括在内。

> **Note**
>
> AWS GovCloud (US) Regions不支持 Amazon Cognito 的自定义域。

为此，请使用以下参数部署 PCUI 堆栈：

- CustomDomain: *xyz.example.com*.
- CustomDomainCertificateArn：Amazon 资源名称 (ARN) 的 ACM 证书。*xyz.example.com*
- CognitoCustomDomain: *auth-xyz.example.com*.
- CognitoCustomDomainCertificateArn转到 ACM 证书的 Amazon 资源名称 (ARN)。*xyz.example.com*

配置 DNS

完成以下步骤来配置您的域名服务 (DNS)，以便 AWS ParallelCluster 用户界面 (PCUI) 和 Amazon Cognito 对所需的自定义域名做出响应。

此示例假设您要使用自定义域 `xyz.example.com` 部署 PCUI，并使用自定义域 `auth-xyz.example.com` 部署 Amazon Cognito 接口。

1. 使用以下参数部署 PCUI 堆栈：
   - CustomDomainEndpoint：在您的 DNS 中创建 A 记录*xyz.example.com*，该记录指向输出中指定的别名。
   - CognitoCustomDomainEndpoint：在您的 DNS 中创建 A 记录*auth-xyz.example.com*，该记录指向输出中指定的别名。
2. 等待约 10 分钟，以便可以分发 DNS 更改。
3. 分发 DNS 更改后，PCUI 对 *xyz.example.com/pcui* 进行响应，Amazon Cognito 身份验证页面对 *auth-xyz.example.com* 进行响应。

Amazon Cognito 用户群体选项

以下各节涉及 CloudFormation 快速创建链接或快速 URLs创建。快速创建 URL 会将您带到创建堆栈向导，您可以在其中提供快速创建堆栈模板输入并部署堆栈。有关 CloudFormation 快速创建堆栈的更多信息，请参阅《用户指南》中的为堆栈创建快速创建链接。AWS CloudFormation

要维护可用于多个用户 AWS ParallelCluster 界面 (PCUI) 实例的 Amazon Cognito 用户池，请考虑以下选项：

- 使用链接到基于嵌套堆栈创建的 Amazon Cognito 用户池的现有 PCUI 实例。 CloudFormation 这是您使用快速创建链接部署 PCUI 并将所有 Amazon Cognito 参数留空时创建的内容。

- 首先，部署一个独立的 Amazon Cognito 用户池。然后，部署一个与您刚刚部署的独立 Amazon Cognito 用户池关联的新 PCUI 实例。这样，您可以将 Amazon Cognito 部署与 PCUI 部署分开。请注意，非嵌套的 PCUI CloudFormation 堆栈更易于更新。

将现有 Amazon Cognito 用户池与新 PCUI 实例结合使用

完成以下步骤将现有 Amazon Cognito 用户池与新 PCUI 实例结合使用

将现有 Amazon Cognito 用户池与新 PCUI 实例结合使用

1. 在CloudFormation 控制台中，选择包含要用于多个 PCUI 实例的 Amazon Cognito 用户池的 PCUI 堆栈。
2. 导航到创建 Amazon Cognito 用户群体的嵌套堆栈。
3. 选择输出选项卡。
4. 复制以下参数的值：
   - `UserPoolId`
   - `UserPoolAuthDomain`
   - `SNSRole`
5. 使用快速创建链接并用您复制的输出填充所有 `External PCUI Amazon Cognito` 参数，部署一个新 PCUI 实例。这可防止新 PCUI 堆栈创建新用户池，并将其链接到从嵌套堆栈创建的现有 Amazon Cognito 用户池。您可以随后部署具有相同参数值的新 PCUI 实例，并可以将它们链接到 Amazon Cognito 用户池。

创建独立的 Amazon Cognito 用户群体

创建独立的 Amazon Cognito 用户池

1. 从以下列表中选择一个标有相同 AWS 区域 标签的快速创建链接，即可启动仅限 Amazon Cognito 的堆栈：
   - us-east-1
   - us-east-2
   - us-west-1
   - us-west-2
   - eu-west-1

- eu-west-2
- eu-west-3
- eu-central-1
- eu-north-1
- me-south-1
- sa-east-1
- ca-central-1
- ap-northeast-1
- ap-northeast-2
- ap-south-1
- ap-southeast-1
- ap-southeast-2
- us-gov-west-1

2. 堆栈创建完成后，选择输出选项卡并复制以下参数的值：
   - `UserPoolId`
   - `UserPoolAuthDomain`
   - `SNSRole`
3. 通过选择 PCUI 快速启动链接并用您复制的值填充所有 `External PCUI Amazon Cognito` 参数，部署一个新 PCUI 实例。新的 PCUI 实例链接到独立的 Amazon Cognito 用户池，不会创建嵌套堆栈或新的用户池。您可以随后部署具有相同参数值的新 PCUI 实例，并可以将它们链接到独立的 Amazon Cognito 用户池。

识别 AWS ParallelCluster 和 PCUI 版本

要识别 AWS ParallelCluster 和 PCUI 版本，请执行以下操作：

1. 在 CloudFormation 控制台中，选择一个 PCUI 堆栈。
2. 选择参数选项卡。
3. AWS ParallelCluster 版本是参数 “版本” 的值。
4. PCUI 版本位于该PublicEcrImageUri值的末尾。例如，如果值为 `public.ecr.aws/pcui/parallelcluster-ui-awslambda:2023.02`，则版本为 `2023.02`。

> **Note**
>
> 要将 PCUI 更新到最新 AWS ParallelCluster 版本，请通过选择快速创建链接来启动新堆栈。

PCUI 成本

PCUI 基于无服务器架构构建，在大多数情况下，您可以在 AWS 免费套餐类别中使用它。下表列出了 PCUI AWS 服务 所依赖的及其免费套餐限制。通常情况下，使用成本估计低于每月一美元。

服务	AWS 免费套餐
Amazon Cognito	每月 50000 个活跃用户
Amazon API Gateway	100 万次 rest API 调用
AWS Lambda	每月 100 万次免费请求和每月 400000 GB-秒的计算时间
EC2 Image Builder	不收取任何费用，除了 EC2
Amazon Elastic Compute Cloud	15 分钟一次性容器映像构建
AWS CloudFormation	5 GB 数据（摄取、存档存储和通过 Logs Insights 查询扫描的数据）

入门 AWS ParallelCluster

要开始使用 AWS ParallelCluster 命令行界面 (CLI) 或基于 Web 的用户界面 (UI) AWS ParallelCluster 配置和创建集群。版本 3.5.0 中添加了 PCUI。

主题

- 使用 AWS ParallelCluster 命令行界面配置和创建集群
- 使用 PCUI 配置和创建集群
- 连接到集群

使用 AWS ParallelCluster 命令行界面配置和创建集群

安装后 AWS ParallelCluster，完成以下配置步骤。

1. 验证您的 AWS 账户是否具有包含运行 `pcluster` CLI 所需的权限的角色。有关更多信息，请参阅 AWS ParallelCluster `pcluster`用户策略示例。
2. 设置您的 AWS 凭证。有关更多信息，请参阅 AWS CLI 用户指南 中的配置 AWS CLI。

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

3. 集群的启动 AWS 区域 位置必须至少有一个 Amazon EC2 密钥对。有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的 Amazon Elastic Compute Cloud 密钥对。

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

配置和创建第一个集群

要创建您的第一个集群，请使用 `pcluster configure` CLI 命令启动向导，提示您提供配置和创建集群所需的所有信息。与使用 AWS Batch 作为调度器相比，序列的细节有所不同。Slurm

Slurm

```
$  pcluster configure --config cluster-config.yaml
```

从有效 AWS 区域 标识符列表中，选择您希望集群运行 AWS 区域 的位置。

> **Note**
>
> AWS 区域 显示的列表基于您的账户分区，仅包括为您的账户启 AWS 区域 用的分区。有关为您的账户启 AWS 区域 用的更多信息，请参阅 AWS 区域中的管理AWS 一般参考。显示的示例来自 AWS 全局分区。如果您的账户在 AWS GovCloud (US) 分区中，则只会列出该分区 AWS 区域 中的账户（`gov-us-east-1`和`gov-us-west-1`）。同样，如果您的账户

位于 AWS 中国分区，则`cn-north-1`仅显示`cn-northwest-1`和。有关 AWS 区域 支持的完整列表 AWS ParallelCluster，请参阅AWS 区域 支持 AWS ParallelCluster。

```
Allowed values for 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. ap-southeast-3
9. ap-southeast-5
10. ap-southeast-7
11. ca-central-1
12. eu-central-1
13. eu-north-1
14. eu-south-1
15. eu-west-1
16. eu-west-2
17. eu-west-3
18. il-central-1
19. me-south-1
20. sa-east-1
21. us-east-1
22. us-east-2
23. us-west-1
24. us-west-2
AWS ## ID [ap-northeast-1]:
```

在选定的选项中，从在亚马逊弹性计算云中注册的密钥对中选择密钥对 AWS 区域：

```
Allowed values for Amazon EC2 Key Pair Name:
1. your-key-1
2. your-key-2
Amazon EC2 Key Pair Name [your-key-1]:
```

选择要用于集群的计划程序。

```
Allowed values for Scheduler:
```

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

选择操作系统。

```
Allowed values for Operating System:
1. alinux2
2. alinux2023
3. ubuntu2404
4. ubuntu2204
5. ubuntu2004
6. rhel8
7. rhel9
Operating System [alinux2]:
```

选择头节点实例类型：

```
Head node instance type [t2.micro]:
```

选择队列配置。注意：不能为同一个队列中的多个计算资源指定实例类型。

```
Number of queues [1]:
Name of queue 1 [queue1]:
Number of compute resources for queue1 [1]: 2
Compute instance type for compute resource 1 in queue1 [t2.micro]:
Maximum instance count [10]:
```

启用 EFA 以大规模运行需要高水平实例间通信的应用程序，无需额外付费： AWS

- 选择支持 Elastic Fabric Adapter (EFA) 的实例类型。
- 启用 EFA。
- 指定现有置放群组名称。如果将其留空，则会为您 AWS ParallelCluster 创建一个。

```
Compute instance type for compute resource 2 in queue1 [t2.micro]: c5n.18xlarge
Enable EFA on c5n.18xlarge (y/n) [y]: y
Maximum instance count [10]:
Placement Group name []:
```

完成前面的步骤后，决定是使用现有 VPC 还是让你 AWS ParallelCluster 创建 VPC。如果您没有正确配置的 VPC， AWS ParallelCluster 可以为您创建一个新的 VPC。它会将头节点和计算节点置于同一公有子网中，或者仅将头节点置于公有子网中，而将所有计算节点置于私有子网中。如果让 AWS ParallelCluster 创建 VPC，则必须决定是否所有节点都应位于公有子网中。有关更多信息，请参阅 网络配置。

如果您将集群配置为使用具有多个网络接口或网卡的实例类型，请参阅网络配置以了解其他网络要求。

可以达到您的配额，以达到 VPCs 允许的数量 AWS 区域。a 的默认配额 VPCs 为五个 AWS 区域。有关此配额以及如何请求提高配额的更多信息，请参阅 Amazon VPC 用户指南 中的 VPC 和子网。

> **Important**
>
> VPCs 创建者默认情况下 AWS ParallelCluster 不启用 VPC 流日志。VPC 流日志使您能够捕获有关进出您的网络接口的 IP 流量的信息 VPCs。有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 流日志。

如果您允许 AWS ParallelCluster 创建 VPC，请务必决定是否所有节点都位于公有子网中。

> **Note**
>
> 如果您愿意`1. Head node in a public subnet and compute fleet in a private subnet`，则 AWS ParallelCluster 创建一个 NAT 网关，即使您指定了免费套餐资源，也会产生额外费用。

```
Automate VPC creation? (y/n) [n]: y
Allowed values for Availability Zone:
1. us-east-1a
2. us-east-1b
3. us-east-1c
4. us-east-1d
5. us-east-1e
6. us-east-1f
Availability Zone [us-east-1a]:
Allowed values for Network Configuration:
1. Head node in a public subnet and compute fleet in a private subnet
```

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

如果您不创建新的 VPC，则必须选择现有 VPC。

如果您选择 AWS ParallelCluster 创建 VPC，请记下 VPC ID，以便日 AWS CLI 后使用将其删除。

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

选择 VPC 后，决定是使用现有子网还是创建新子网。

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

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

AWS Batch

```
$  pcluster configure --config cluster-config.yaml
```

从有效 AWS 区域 标识符列表中，选择您希望集群运行 AWS 区域 的位置。

> **Note**
>
> AWS 区域 显示的列表基于您的账户分区。它仅包括 AWS 区域 为您的账户启用的功能。有关为您的账户启 AWS 区域 用的更多信息，请参阅 AWS 区域中的管理AWS 一般参考。显示的示例来自 AWS 全局分区。如果您的账户在 AWS GovCloud (US) 分区中，则只会列出该分区 AWS 区域 中的账户（`gov-us-east-1`和`gov-us-west-1`）。同样，如果您的账户位于 AWS 中国分区，则`cn-north-1`仅显示`cn-northwest-1`和。有关 AWS 区域 支持的完整列表 AWS ParallelCluster，请参阅AWS 区域 支持 AWS ParallelCluster。

```
Allowed values for 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. ap-southeast-3
9. ap-southeast-5
10. ap-southeast-7
11. ca-central-1
12. eu-central-1
13. eu-north-1
14. eu-south-1
15. eu-west-1
16. eu-west-2
17. eu-west-3
18. il-central-1
19. me-south-1
20. sa-east-1
21. us-east-1
22. us-east-2
23. us-west-1
24. us-west-2
AWS ## ID [us-east-1]:
```

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

```
Allowed values for Amazon EC2 Key Pair Name:
1. your-key-1
2. your-key-2
Amazon EC2 Key Pair Name [your-key-1]:
```

选择要用于集群的计划程序。

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

当选择 `awsbatch` 作为计划程序时，`alinux2` 将用作操作系统。输入头节点实例类型：

```
Head node instance type [t2.micro]:
```

选择队列配置。 AWS Batch 调度器仅包含一个队列。输入计算节点集群的最大大小。这是用 v 来衡量CPUs的。

```
Number of queues [1]:
Name of queue 1 [queue1]:
Maximum vCPU [10]:
```

决定是使用现有 VPCs 还是让自己 AWS ParallelCluster VPCs 创建。如果没有正确配置的 VPC，AWS ParallelCluster 可以创建新的 VPC。它将使用同一公有子网中的头节点和计算节点，或者仅使用公有子网中的头节点，所有节点都在私有子网中。一个地区 VPCs 允许的数量可以达到您的配额。默认数字 VPCs 为五。有关此配额以及如何请求提高配额的更多信息，请参阅 Amazon VPC 用户指南 中的 VPC 和子网。

> **Important**
>
> VPCs 创建者默认情况下 AWS ParallelCluster 不启用 VPC 流日志。VPC 流日志使您能够捕获有关进出您的网络接口的 IP 流量的信息 VPCs。有关更多信息，请参阅《Amazon VPC 用户指南》中的 VPC 流日志。

如果您允许 AWS ParallelCluster 创建 VPC，请务必决定是否所有节点都位于公有子网中。

> **Note**
>
> 如果您选择 `1. Head node in a public subnet and compute fleet in a private subnet`， AWS ParallelCluster 将会创建一个 NAT 网关，即使您指定了免费套餐资源，也会产生额外费用。

```
Automate VPC creation? (y/n) [n]: y
Allowed values for Availability Zone:
1. us-east-1a
2. us-east-1b
3. us-east-1c
4. us-east-1d
5. us-east-1e
6. us-east-1f
```

```
Availability Zone [us-east-1a]:
Allowed values for Network Configuration:
1. Head node in a public subnet and compute fleet in a private subnet
2. Head node and compute fleet in the same public subnet
Network Configuration [Head node in a public subnet and compute fleet in a private
 subnet]: *1*
Beginning VPC creation. Please do not leave the terminal until the creation is
 finalized
```

如果您不创建新的 VPC，则必须选择现有 VPC。

如果您选择 AWS ParallelCluster 创建 VPC，请记下 VPC ID，以便日后使用 AWS CLI 或 AWS Management Console 将其删除。

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

选择 VPC 后，确保决定是使用现有子网还是创建新子网。

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

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

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

- VPC 必须具有 `DNS Resolution = yes` 和 `DNS Hostnames = yes`。
- VPC 还必须具有带适用于 AWS 区域的正确 `domain-name` 的 DHCP 选项。默认 DHCP 选项集已经指定了所需的 AmazonProvidedDNS。如果指定多个域名服务器，请参阅 Amazon VPC 用户指南中的 DHCP 选项集。使用私有子网时，请使用 NAT 网关或内部代理为计算节点启用 Web 访问。有关更多信息，请参阅 网络配置。

但所有设置都包含有效值时，您可以通过运行创建命令来启动集群：

```
$ pcluster create-cluster --cluster-name test-cluster --cluster-configuration cluster-
config.yaml
{
  "cluster": {
    "clusterName": "test-cluster",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:xxx:stack/test-cluster/
abcdef0-f678-890a-5abc-021345abcdef",
    "region": "eu-west-1",
    "version": "3.13.2",
    "clusterStatus": "CREATE_IN_PROGRESS"
  },
  "validationMessages": []
}
```

关注集群进度：

```
$ pcluster describe-cluster --cluster-name test-cluster
```

或

```
$ pcluster list-clusters --query 'clusters[?clusterName==`test-cluster`]'
```

在集群达到`"clusterStatus": "CREATE_COMPLETE"`状态后，您可以使用常规 SSH 客户端设置连接到该集群。有关连接亚马逊 EC2 实例的更多信息，请参阅亚马逊 EC2 用户指南中的亚马逊 EC2 用户指南。或者，您可以通过以下方式连接到该集群

```
$ pcluster ssh --cluster-name test-cluster -i ~/path/to/keyfile.pem
```

要删除该集群，请运行以下命令。

```
$ pcluster delete-cluster --region us-east-1 --cluster-name test-cluster
```

删除集群后，您可以通过删除网络堆栈来删除 VPC 中的 CloudFormation 网络资源。堆栈名称以“parallelclusternetworking-”开头，并且包含“YYYYMMDDHHMMSS”格式的创建时间。您可以使用 `list-stacks` 命令列出堆栈。

```
$ aws --region us-east-1 cloudformation list-stacks \
   --stack-status-filter "CREATE_COMPLETE" \
```

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

可以使用 delete-stack 命令删除堆栈。

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

为您pcluster configure创建的 VPC 不是在 CloudFormation 网络堆栈中创建的。您可以在控制台中或者通过使用 AWS CLI，手动删除该 VPC。

```
$ aws --region us-east-1 Amazon EC2 delete-vpc --vpc-id vpc-0b4ad9c4678d3c7ad
```

使用 PCUI 配置和创建集群

PCUI 是一个基于 Web 的用户界面，它镜像 AWS ParallelCluster `pcluster` CLI，同时提供类似于控制台的体验。您可以在中安装并访问 PCUI。 AWS 账户当你运行它时，PCU AWS ParallelCluster I 会访问你的 Amazon API Gateway 上托管的 API 实例。 AWS 账户

> **Note**
>
> PCUI 向导可能没有适用于最新支持 AWS ParallelCluster 版本中所有支持功能的 UI 选项。您可以根据需要手动编辑配置文件或使用 AWS ParallelCluster CLI。

本节将指导您了解如何使用 PCUI 配置和创建集群。

先决条件：

- 可以访问正在运行的 PCUI 实例。有关更多信息，请参阅 安装 PCUI。

配置和创建集群

1. 在 PCUI 的集群视图中，选择创建集群、逐步。
2. 在集群、名称中，输入集群的名称。
3. 为您的集群选择一个带有公有子网的 VPC，然后选择 Next。
4. 在 Head 节点中，选择添加 SSM 会话，然后选择下一步。

5. 在队列、计算资源中，为静态节点选择 1。
6. 对于实例类型，删除选定的默认实例类型，选择 t2.micro，然后选择下一步。
7. 在存储中，选择下一步。
8. 在集群配置中，查看集群配置 YAML，然后选择试运行对其进行验证。
9. 选择创建，根据经过验证的配置创建集群。
10. 几秒钟后，PCUI 会自动导航回集群，您可以在其中监控集群创建状态和堆栈事件。
11. 选择详细信息以查看集群的详细信息，例如版本和状态。
12. 选择实例以查看 Amazon EC2 实例列表和状态。
13. 选择堆栈事件可查看集群堆栈事件，以及 AWS Management Console 指向创建集群的 CloudFormation 堆栈的链接。
14. 在详细信息中，集群创建完成后，选择查看 YAML 以查看或下载集群配置 YAML 文件。
15. 集群创建完成后，选择 Shell 以访问集群头节点。

> **Note**
>
> 当你选择 Shell 时，会 AWS ParallelCluster 打开 Amazon S EC2 ystems Manager 会话并`ssm-user`向中添加一个`/etc/sudoers`。有关更多信息，请参阅 Amazon S EC2 ystems Manager 用户指南中的开启或关闭*`ssm-user`*账户管理权限。

16. 要进行清理，请在集群视图中选择集群，然后选择操作、删除集群。

连接到集群

使用时 AWS ParallelCluster，您可以连接到集群头节点来运行作业、查看结果、管理用户以及监控集群和作业状态。要连接到集群头节点实例，请使用以下方法之一：

- 您可以使用密钥对（k ssh ey pair）进行登录。在集群配置中的 `HeadNode`/`KeyName` 中指定私有密钥。有关更多信息，请参阅《亚马逊 Linux 实例 EC2 用户指南》中的使用 SSH 连接您的 Linux 实例。
- 您可以使用`pcluster ssh`命令行界面 (CLI) 命令登录。在集群配置 `HeadNode`/`KeyName` 中指定私有密钥。有关更多信息，请参阅 `pcluster ssh`。
- 您可以使用 SSM 会话连接到群集头节点。您必须将 `AmazonSSMManagedInstanceCore` 托管策略添加到集群配置中的 `HeadNode`/`AdditionalIamPolicies` 才能使用 SSM 会话进行连接。有关更多信息，请参阅 SSM 用户指南中的 SSM 会话管理器。

- 您可以使用 Amazon DCV 连接到集群头节点。有关更多信息，请参阅 通过 Amazon DCV 连接到头节点和登录节点。
- 使用 PCUI 时，您还可以使用用户界面提供的 Amazon EC2 Connect 命令连接到集群头节点。

集群的多用户访问

了解如何实施和管理单个集群的多用户访问。

在本主题中， AWS ParallelCluster 用户是指计算实例的系统用户。一个例子是 Amazon EC2 实例。`ec2-user`

AWS ParallelCluster 目前所有可用 AWS 区域 的地方 AWS ParallelCluster 都提供多用户访问支持。它可以与其他公司配合使用 AWS 服务，包括亚马逊 for Lustre 和亚马逊 FSx Elast ic File System。

您可以使用 AWS Directory Service for Microsoft Active Directory 或 Simple AD 来管理集群访问权限。请确保检查这些服务的 AWS 区域 可用性。要设置集群，请指定AWS ParallelCluster DirectoryService配置。 AWS Directory Service 目录可以连接到多个集群。这样可以对多种环境中的身份进行集中管理，并提供统一的登录体验。

当您使用 AWS Directory Service AWS ParallelCluster 多用户访问时，您可以使用目录中定义的用户凭据登录集群。这些凭证包含用户名和密码。首次登录到集群后，会自动生成用户 SSH 密钥。您可以使用该密钥登录，而无需使用密码。

部署目录服务后，您可以创建、删除和修改集群的用户或组。使用 AWS Directory Service，您可以在 AWS Management Console 或中使用 A ctive Directory 用户和计算机工具执行此操作。该工具可从任何已加入您的活动目录的 Amazon EC2 实例访问。有关更多信息，请参阅安装 Active Directory 管理工具。

如果您计划 AWS ParallelCluster 在无法访问 Internet 的单个子网中使用，AWS ParallelCluster 在无法访问互联网的单个子网中请参阅了解其他要求。

主题

- 创建 Active Directory
- 使用 AD 域创建集群
- 登录到与 AD 域集成的集群
- 运行 MPI 作业
- LDAP (S) 集群配置示例 AWS Managed Microsoft AD

创建 Active Directory

请确保在创建集群之前创建 Active Directory (AD)。有关如何为集群选择 Active Directory 类型的信息，请参阅 AWS Directory Service Administration Guide 中的 Which to choose。

如果目录为空，请使用用户名和密码添加用户。有关更多信息，请参阅特定于 AWS Directory Service for Microsoft Active Directory 或 Simple AD 的文档。

> **Note**
>
> AWS ParallelCluster 要求每个 Active Directory 用户目录都位于该`/home/$user`目录中。

使用 AD 域创建集群

> **Warning**
>
> 本介绍性部分介绍如何 AWS ParallelCluster 通过轻型目录访问协议 (LDAP) 设置托管活动目录 (AD) 服务器。LDAP 是一种不安全的协议。对于生产系统，我们强烈建议使用 TLS 证书 (LDAPS)，如下文中的LDAP (S) 集群配置示例 AWS Managed Microsoft AD一节所述。

通过在集群配置文件的 `DirectoryService` 部分指定相关信息，将您的集群配置为与目录集成。有关更多信息，请参阅 `DirectoryService` 配置部分。

您可以使用以下示例将您的集群与轻量级目录访问协议 (LDAP) 集成。 AWS Managed Microsoft AD

AWS Managed Microsoft AD 通过 LDAP 配置所需的特定定义：

- 必须在 `DirectoryService`/`AdditionalSssdConfigs` 下面将 `ldap_auth_disable_tls_never_use_in_production` 参数设置为 `True`。
- 您可以为 `DirectoryService`/`DomainAddr` 指定控制器主机名或 IP 地址。
- `DirectoryService`/`DomainReadOnlyUser` 语法必须如下所示：

```
cn=ReadOnly,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
```

获取您的 AWS Managed Microsoft AD 配置数据：

```
$ aws ds describe-directories --directory-id "d-abcdef01234567890"
```

```
{
    "DirectoryDescriptions": [
        {
            "DirectoryId": "d-abcdef01234567890",
            "Name": "corp.example.com",
            "DnsIpAddrs": [
                "203.0.113.225",
                "192.0.2.254"
            ],
            "VpcSettings": {
                "VpcId": "vpc-021345abcdef6789",
                "SubnetIds": [
                    "subnet-1234567890abcdef0",
                    "subnet-abcdef01234567890"
                ],
                "AvailabilityZones": [
                    "region-idb",
                    "region-idd"
                ]
            }
        }
    ]
}
```

AWS Managed Microsoft AD的集群配置：

```
Region: region-id
Image:
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-1234567890abcdef0
  Ssh:
    KeyName: pcluster
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ComputeResources:
```

```
        - Name: t2micro
          InstanceType: t2.micro
          MinCount: 1
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
DirectoryService:
  DomainName: dc=corp,dc=example,dc=com
  DomainAddr: ldap://203.0.113.225,ldap://192.0.2.254
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:MicrosoftAD.Admin.Password-1234
  DomainReadOnlyUser: cn=ReadOnly,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  AdditionalSssdConfigs:
    ldap_auth_disable_tls_never_use_in_production: True
```

要对 Simple AD 使用此配置，请在 **DirectoryService** 部分更改 **DomainReadOnlyUser** 属性值：

```
DirectoryService:
  DomainName: dc=corp,dc=example,dc=com
  DomainAddr: ldap://203.0.113.225,ldap://192.0.2.254
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:SimpleAD.Admin.Password-1234
  DomainReadOnlyUser: cn=ReadOnlyUser,cn=Users,dc=corp,dc=example,dc=com
  AdditionalSssdConfigs:
    ldap_auth_disable_tls_never_use_in_production: True
```

注意事项：

- 我们建议您使用 LDAP，TLS/SSL (or LDAPS) rather than LDAP alone. TLS/SSL确保连接已加密。
- `DirectoryService`/`DomainAddr` 属性值与 `describe-directories` 输出的 `DnsIpAddrs` 列表中的条目相匹配。
- 我们建议集群使用的子网位于 `DirectoryService`/`DomainAddr` 指向的同一可用区内。如果您使用推荐用于目录的自定义动态主机配置协议 (DHCP) 配置， VPCs 并且您的子网不在 `DirectoryService`/`DomainAddr`可用区内，则可用区之间可能会出现交叉流量。使用多用户 AD 集成功能不 需要使用自定义 DHCP 配置。
- `DirectoryService`/`DomainReadOnlyUser` 属性值指定必须在目录中创建的用户。默认情况下不创建此用户。我们建议您不要 向此用户授予修改目录数据的权限。
- `DirectoryService`/`PasswordSecretArn`属性值指向一个 AWS Secrets Manager 密钥，其中包含您为 `DirectoryService`/`DomainReadOnlyUser`属性指定的用户密码。如果此用户

的密码发生更改，请更新密钥值并更新集群。要针对新密钥值更新集群，必须使用 `pcluster update-compute-fleet` 命令停止计算实例集。如果您将集群配置为使用 `LoginNodes`，请停止 `LoginNodes`/`Pools` 并在将 `LoginNodes`/`Pools`/`Count` 设置为 0 后更新集群。然后在集群头节点内运行以下命令。

```
sudo /opt/parallelcluster/scripts/directory_service/
update_directory_service_password.sh
```

有关其他示例，另请参阅集成 Active Directory。

登录到与 AD 域集成的集群

如果您启用了 Active Directory (AD) 域集成功能，则会在集群头节点上启用密码身份验证。AD 用户的主目录是在用户第一次登录头节点或第一次登录头节点时创建的 sudo-user 切换到头节点上的 AD 用户。

不会为集群计算节点启用密码身份验证。AD 用户必须使用 SSH 密钥登录计算节点。

默认情况下，首次 SSH 登录到头节点时，会在 AD 用户 `/${HOME}/.ssh` 目录中设置 SSH 密钥。通过在集群配置中将 `DirectoryService`/`GenerateSshKeysForUsers` 布尔属性设置为 `false`，可以禁用此行为。默认情况下，`DirectoryService`/`GenerateSshKeysForUsers` 设置为 `true`。

如果 AWS ParallelCluster 应用程序需要在群集节点之间使用无密码 SSH，请确保在用户的主目录中正确设置 SSH 密钥。

AWS Managed Microsoft AD 密码将在 42 天后过期。有关更多信息，请参阅 AWS Directory Service Administration Guide 中的 Manage password policies for AWS Managed Microsoft AD。如果您的密码过期，则必须重置密码才能恢复集群访问权限。有关更多信息，请参阅 如何重置用户密码和过期的密码。

> **Note**
>
> 如果 AD 集成功能无法正常运行，则 SSSD 日志可为问题排查提供有用的诊断信息。这些日志位于集群节点上的 `/var/log/sssd` 目录中。默认情况下，它们还存储在集群的 Amazon CloudWatch 日志组中。
> 有关更多信息，请参阅 排查与 Active Directory 的多用户集成问题。

运行 MPI 作业

正如 SchedMD 中所建议的那样，使用引导 MPI 作业 Slurm 作为 MPI 引导方法。欲了解更多信息，请咨询官方 Slurm 您的 MPI 库的文档或官方文档。

例如，在 IntelMPI 官方文档中，您了解到在运行 StarCCM 作业时，必须设置 Slurm 通过导出环境变量作为流程协调器。`I_MPI_HYDRA_BOOTSTRAP=slurm`

> **Note**
>
> 已知问题
>
> 如果您的 MPI 应用程序依赖于 SSH 作为生成 MPI 作业的机制，则 Slurm 中的已知错误可能会导致将目录用户名错误地解析为“nobody”。
>
> 要么配置要使用的应用程序 Slurm 作为 MPI 引导方法，或参阅 “故障排除” 部分，了解更多详细信息和可能的解决方法。用户名解析的已知问题

LDAP (S) 集群配置示例 AWS Managed Microsoft AD

AWS ParallelCluster 通过与轻量级目录访问协议 (LDAP) 或 AWS Directory Service 通过 TLS/SSL 的 LDAP (LDAPS) 集成，支持多用户访问。

以下示例显示了如何创建集群配置以便与基于 LDAP(S) 的 AWS Managed Microsoft AD 集成。

AWS Managed Microsoft AD 通过 LDAPS 进行证书验证

您可以使用此示例将您的集群与 ov AWS Managed Microsoft AD er LDAPS 集成，并进行证书验证。

带有证书配置的 ov AWS Managed Microsoft AD er LDAPS 的具体定义：

- 对于具有证书验证功能的 LDAPS，必须将 `DirectoryService`/`LdapTlsReqCert` 设置为 `hard`（默认值）。
- `DirectoryService`/`LdapTlsCaCert` 必须指定您的证书颁发机构 (CA) 证书的路径。

  CA 证书是一个证书捆绑包，其中包含为 AD 域控制器颁发证书的整个 CA 链的证书。

  您的 CA 证书必须安装在集群节点上。
- 必须为 `DirectoryService`/`DomainAddr` 指定控制器主机名，而不是 IP 地址。
- `DirectoryService`/`DomainReadOnlyUser` 语法必须如下所示：

```
cn=ReadOnly,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
```

使用基于 LDAPS 的 AD 时的集群配置文件示例：

```
Region: region-id
Image:
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-1234567890abcdef0
  Ssh:
    KeyName: pcluster
  Iam:
    AdditionalIamPolicies:
      - Policy: arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
  CustomActions:
    OnNodeConfigured:
      Script: s3://&example-s3-bucket;/scripts/pcluster-dub-msad-ldaps.post.sh
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ComputeResources:
        - Name: t2micro
          InstanceType: t2.micro
          MinCount: 1
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
      Iam:
        AdditionalIamPolicies:
          - Policy: arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
      CustomActions:
        OnNodeConfigured:
          Script: s3://&example-s3-bucket;/scripts/pcluster-dub-msad-ldaps.post.sh
DirectoryService:
  DomainName: dc=corp,dc=example,dc=com
  DomainAddr: ldaps://win-abcdef01234567890.corp.example.com,ldaps://win-
abcdef01234567890.corp.example.com
```

```
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:MicrosoftAD.Admin.Password-1234
  DomainReadOnlyUser: cn=ReadOnly,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  LdapTlsCaCert: /etc/openldap/cacerts/corp.example.com.bundleca.cer
  LdapTlsReqCert: hard
```

在安装后脚本中添加证书并配置域控制器：

```
*#!/bin/bash*
set -e

AD_CERTIFICATE_S3_URI="s3://amzn-s3-demo-bucket/bundle/corp.example.com.bundleca.cer"
AD_CERTIFICATE_LOCAL="/etc/openldap/cacerts/corp.example.com.bundleca.cer"

AD_HOSTNAME_1="win-abcdef01234567890.corp.example.com"
AD_IP_1="192.0.2.254"

AD_HOSTNAME_2="win-abcdef01234567890.corp.example.com"
AD_IP_2="203.0.113.225"

# Download CA certificate
mkdir -p $(dirname "${AD_CERTIFICATE_LOCAL}")
aws s3 cp "${AD_CERTIFICATE_S3_URI}" "${AD_CERTIFICATE_LOCAL}"
chmod 644 "${AD_CERTIFICATE_LOCAL}"

# Configure domain controllers reachability
echo "${AD_IP_1} ${AD_HOSTNAME_1}" >> /etc/hosts
echo "${AD_IP_2} ${AD_HOSTNAME_2}" >> /etc/hosts
```

您可以从加入域的实例中检索域控制器主机名，如以下示例所示。

来自 Windows 实例

```
$ nslookup 192.0.2.254
```

```
Server:  corp.example.com
Address:  192.0.2.254

Name:    win-abcdef01234567890.corp.example.com
Address:  192.0.2.254
```

来自 Linux 实例

```
$ nslookup 192.0.2.254
```

```
192.0.2.254.in-addr.arpa   name = corp.example.com
192.0.2.254.in-addr.arpa   name = win-abcdef01234567890.corp.example.com
```

AWS Managed Microsoft AD 在没有证书验证的情况下通过 LDAPS

您可以使用此示例将您的集群与 ov AWS Managed Microsoft AD er LDAPS 集成，无需证书验证。

没有证书验证配置的 ov AWS Managed Microsoft AD er LDAPS 的具体定义：

- 必须将 DirectoryService/LdapTlsReqCert 设置为 never。
- 可以为 DirectoryService/DomainAddr 指定控制器主机名或 IP 地址。
- DirectoryService/DomainReadOnlyUser 语法必须如下所示：

  ```
  cn=ReadOnly,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  ```

无需证书验证即可 AWS Managed Microsoft AD 通过 LDAPS 使用的集群配置文件示例：

```
Region: region-id
Image:
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-1234567890abcdef0
  Ssh:
    KeyName: pcluster
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ComputeResources:
        - Name: t2micro
          InstanceType: t2.micro
          MinCount: 1
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
```

```
DirectoryService:
  DomainName: dc=corp,dc=example,dc=com
  DomainAddr: ldaps://203.0.113.225,ldaps://192.0.2.254
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:MicrosoftAD.Admin.Password-1234
  DomainReadOnlyUser: cn=ReadOnly,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  LdapTlsReqCert: never
```

最佳实践

以下各节提供了最佳使用实践 AWS ParallelCluster，其中包括网络性能和预算提醒。如果您在遵循这些最佳做法后仍遇到问题，AWS ParallelCluster 故障排除请参见以了解可能的解决方案。

最佳实践：头节点实例类型选择

尽管头节点不运行作业，但其功能和大小对集群的整体性能至关重要。在选择用于头节点的实例类型时，请考虑以下特征：

集群大小：头节点编排集群的扩展逻辑，并负责将新节点附加到调度器。要纵向扩展和缩减具有大量节点的集群，请为头节点提供一些额外的计算容量。

共享文件系统：当您使用共享文件系统时，请选择具有足够网络带宽和足够的 Amazon EBS 带宽的实例类型来处理您的工作流程。确保头节点既能够为集群公开足够的 NFS 服务器目录，又能够处理需要在计算节点和头节点之间共享的构件。

最佳实践：网络性能

网络性能对于高性能计算 (HPC) 应用程序至关重要。如果没有可靠的网络性能，这些应用程序就无法正常运行。为了优化网络性能，请考虑以下最佳实践。

- 置放群组：如果您使用的是 Slurm，请考虑将每个 Slurm 队列配置为使用集群置放群组。集群置放群组 是单个可用区中的实例的逻辑分组。有关更多信息，请参阅 Amazon EC2 用户指南中的置放群组。您可以在队列的 `Networking` 部分指定 `PlacementGroup`，每个计算资源都将分配给队列的置放群组。在计算资源的 `Networking` 部分指定 `PlacementGroup` 时，该特定计算资源将分配给该置放群组。计算资源置放群组规范优先于计算资源的队列规范。有关更多信息，请参阅 `SlurmQueues`/`Networking`/`PlacementGroup` 和 `SlurmQueues`/`ComputeResources`/`Networking`/`PlacementGroup`。

  ```
  Networking:
  ```

```
  PlacementGroup:
    Enabled: true
    Id: your-placement-group-name
```

或者， AWS ParallelCluster 请为您创建一个置放群组。

```
Networking:
  PlacementGroup:
    Enabled: true
```

从 3.3.0 AWS ParallelCluster 版开始，对置放群组的创建和管理进行了修改。在指定要启用的置放群组时，如果没有指定 name 或 Id，则在队列中，会为每个计算资源分配自己的托管置放群组，而不是为整个队列分配一个托管群组。这有助于减少容量不足错误。如果您需要为整个队列设置一个置放群组，则可以使用命名置放群组。

添加了 SlurmQueues/Networking/PlacementGroup/Name 作为 SlurmQueues/Networking/PlacementGroup/Id 的首选替代方案。

有关更多信息，请参阅 Networking。

- 增强联网：考虑选择支持增强联网的实例类型。此建议适用于所有当前一代实例。有关更多信息，请参阅《Amazon EC2 用户指南》中的 Linux 增强联网功能。
- Elastic Fabric Adapter：要支持高水平可扩展实例间通信，请考虑为网络选择 EFA 网络接口。EFA 量身定制的操作系统 (OS) 旁路硬件可利用 AWS Cloud的按需弹性和灵活性增强实例间通信。您可以配置每个 Slurm 队列 ComputeResource 以使用 Efa。有关将 EFA 与配合使用的更多信息 AWS ParallelCluster，请参阅Elastic Fabric Adapter。

```
ComputeResources:
  - Name: your-compute-resource-name
    Efa:
      Enabled: true
```

有关 EFA 的更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的弹性结构适配器。

- 实例带宽：带宽随实例大小而扩展。有关不同实例类型的信息，请参阅《亚马逊用户指南》中的 Amazon EBS 优化实例和 Amazon EBS 卷类型。 EC2

最佳实践：预算提醒

要管理中的资源成本 AWS ParallelCluster，我们建议您使用 AWS Budgets 操作来创建预算。您还可以为选定 AWS 资源创建已定义的预算阈值提醒。有关更多信息，请参阅 AWS Budgets 用户指南 中的配置预算操作。同样，您也可以使用 Amazon CloudWatch 创建账单警报。有关更多信息，请参阅创建账单警报以监控 AWS 预估费用。

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

当前，每个 AWS ParallelCluster 次要版本及其 `pcluster` CLI都是独立的。要将集群迁移至新的次要版本或修补版本，必须使用新版本的 CLI 重新创建集群。

为了优化将集群迁移到新的次要版本或修补版本的过程，我们建议执行下列操作：

- 将个人数据保存在集群外部创建的外部卷中，例如 Amazon EFS 和 FSx For Lustre。这样，您以后便可以轻松地将数据从一个集群迁移到另一个集群。
- 使用以下类型创建共享存储系统。您可以使用 AWS CLI 或创建这些系统 AWS Management Console。
  - `SharedStorage` / `EbsSettings` / `VolumeId`
  - `SharedStorage` / `EfsSettings` / `FileSystemId`
  - `SharedStorage` / `FsxLustreSettings` / `FileSystemId`

  在集群配置中将某个文件系统或卷定义为现有文件系统或卷。这样，当您删除集群时，这些文件系统或卷会被保留下来，并且可以附加到新集群。

  我们建议您使用 Amazon EFS 或 FSx Lustre 文件系统。这两个系统可以同时附加到多个集群。此外，您可以在删除现有集群之前将其中任何一个系统附加到新集群。
- 使用自定义引导操作来自定义您的实例，而不是使用自定义 AMI。如果您改为使用自定义 AMI，则需要对每次新版本发布删除并重新创建该 AMI。
- 我们建议您按以下顺序应用上述建议：
  1. 更新现有集群配置以使用现有文件系统定义。
  2. 验证 `pcluster` 版本并在需要时进行更新。
  3. 创建并测试新集群。在测试新集群时，请检查以下内容：
     - 确保您的数据在新集群中可用。
     - 确保您的应用程序可以在新集群中正常运行。
  4. 在新集群经过全面测试并可正常运行，并且您不再需要现有集群后，将现有集群删除。

从 AWS ParallelCluster 2.x 升级到 3.x

以下各节描述了从 AWS ParallelCluster 2.x 迁移到 3.x 时会发生什么，包括从一个版本到另一个版本的变化。

自定义引导操作

使用 AWS ParallelCluster 3，您可以使用和 `Scheduling`/`SlurmQueues`部分中的（在版本 2 中）和`OnNodeStart`（`pre_install`在 AWS ParallelCluster 版本 2 中）参数为头节点和`OnNodeConfigured`计算节点指定不同的自定义引导操作脚本。`post_install` AWS ParallelCluster `HeadNode`有关更多信息，请参阅 自定义引导操作。

为 AWS ParallelCluster 2 开发的自定义引导操作脚本必须经过调整才能在 AWS ParallelCluster 3 中使用：

- 我们不建议使用 `/etc/parallelcluster/cfnconfig` 和 `cfn_node_type` 来区分头节点和计算节点。相反，我们建议您在 `HeadNode` 和 `Scheduling`/`SlurmQueues` 中指定两个不同的脚本。
- 如果您希望继续加载`/etc/parallelcluster/cfnconfig`以在引导操作脚本中使用，请注意的值已从 “” 更改`cfn_node_type`为 MasterServer “HeadNode”（请参阅：包容性语言）。
- 在 AWS ParallelCluster 2 上，引导操作脚本的第一个输入参数是脚本的 S3 URL，已被保留。在 AWS ParallelCluster 3 中，只有在配置中配置的参数才会传递给脚本。

> **Warning**
>
> 不正式支持使用通过 `/etc/parallelcluster/cfnconfig` 文件提供的内部变量。此文件可能会在未来版本中删除。

AWS ParallelCluster 2.x 和 3.x 使用不同的配置文件语法

AWS ParallelCluster 3.x 配置使用 YAML 语法。有关完整参考，请参阅配置文件。

除了需要 YAML 文件格式外， AWS ParallelCluster 3.x 中还更新了许多配置部分、设置和参数值。在本节中，我们记录了 AWS ParallelCluster 配置的关键更改，并 side-by-side举例说明了每个版本的 AWS ParallelCluster这些差异。

启用和禁用超线程的多个调度器队列配置示例

AWS ParallelCluster 2:

```
[cluster default]
queue_settings = ht-enabled, ht-disabled
...

[queue ht-enabled]
compute_resource_settings = ht-enabled-i1
disable_hyperthreading = false

[queue ht-disabled]
compute_resource_settings = ht-disabled-i1
disable_hyperthreading = true

[compute_resource ht-enabled-i1]
instance_type = c5n.18xlarge
[compute_resource ht-disabled-i1]
instance_type = c5.xlarge
```

AWS ParallelCluster 3:

```
...
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: ht-enabled
      Networking:
        SubnetIds:
          - compute_subnet_id
      ComputeResources:
        - Name: ht-enabled-i1
          DisableSimultaneousMultithreading: true
          InstanceType: c5n.18xlarge
    - Name: ht-disabled
      Networking:
        SubnetIds:
          - compute_subnet_id
      ComputeResources:
        - Name: ht-disabled-i1
          DisableSimultaneousMultithreading: false
          InstanceType: c5.xlarge
```

Lustre 文件 FSx 系统配置的新增示例

AWS ParallelCluster 2:

```
[cluster default]
fsx_settings = fsx
...

[fsx fsx]
shared_dir = /shared-fsx
storage_capacity = 1200
imported_file_chunk_size = 1024
import_path = s3://amzn-s3-demo-bucket
export_path = s3://amzn-s3-demo-bucket/export_dir
weekly_maintenance_start_time = 3:02:30
deployment_type = PERSISTENT_1
data_compression_type = LZ4
```

AWS ParallelCluster 3:

```
...
SharedStorage:
  - Name: fsx
    MountDir: /shared-fsx
    StorageType: FsxLustre
    FsxLustreSettings:
      StorageCapacity: 1200
      ImportedFileChunkSize: 1024
      ImportPath: s3://amzn-s3-demo-bucket
      ExportPath: s3://amzn-s3-demo-bucket/export_dir
      WeeklyMaintenanceStartTime: "3:02:30"
      DeploymentType: PERSISTENT_1
      DataCompressionType: LZ4
```

装载 Lustre 文件系统的现有 FSx 群集配置示例

AWS ParallelCluster 2:

```
[cluster default]
fsx_settings = fsx
...

[fsx fsx]
shared_dir = /shared-fsx
```

```
fsx_fs_id = fsx_fs_id
```

AWS ParallelCluster 3:

```
...
SharedStorage:
  - Name: fsx
    MountDir: /shared-fsx
    StorageType: FsxLustre
    FsxLustreSettings:
      FileSystemId: fsx_fs_id
```

使用 Intel HPC 平台规范软件堆栈的集群示例

AWS ParallelCluster 2:

```
[cluster default]
enable_intel_hpc_platform = true
...
```

AWS ParallelCluster 3:

```
...
AdditionalPackages:
  IntelSoftware:
    IntelHpcPlatform: true
```

注意:

- Intel HPC 平台规范软件的安装受适用的 Intel 最终用户许可协议条款和条件的约束。

自定义 IAM 配置的示例，包括：实例配置文件、实例角色、实例的其他策略以及与集群关联的 lambda 函数的角色

AWS ParallelCluster 2:

```
[cluster default]
additional_iam_policies = arn:aws:iam::aws:policy/
AmazonS3ReadOnlyAccess,arn:aws:iam::aws:policy/AmazonDynamoDBReadOnlyAccess
ec2_iam_role = ec2_iam_role
iam_lambda_role = lambda_iam_role
```

```
...
```

AWS ParallelCluster 3:

```
...
Iam:
  Roles:
    CustomLambdaResources: lambda_iam_role
HeadNode:
  ...
  Iam:
    InstanceRole: ec2_iam_role
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ...
      Iam:
        InstanceProfile: iam_instance_profile
    - Name: queue2
      ...
      Iam:
        AdditionalIamPolicies:
          - Policy: arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
          - Policy: arn:aws:iam::aws:policy/AmazonDynamoDBReadOnlyAccess
```

注意:

- 对于 AWS ParallelCluster 2，IAM 设置适用于集群的所有实例，`additional_iam_policies`不能与一起使用`ec2_iam_role`。
- 对于 AWS ParallelCluster 3，您可以为头节点和计算节点设置不同的 IAM 设置，甚至可以为每个计算队列指定不同的 IAM 设置。
- 对于 AWS ParallelCluster 3，您可以使用 IAM 实例配置文件作为 IAM 角色的替代方案。`InstanceProfile`，`InstanceRole`或者`AdditionalIamPolicies`无法一起配置。

自定义引导操作示例

AWS ParallelCluster 2:

```
[cluster default]
s3_read_resource = arn:aws:s3:::amzn-s3-demo-bucket/*
```

```
pre_install = s3://amzn-s3-demo-bucket/scripts/pre_install.sh
pre_install_args = 'R curl wget'
post_install = s3://amzn-s3-demo-bucket/scripts/post_install.sh
post_install_args = "R curl wget"
...
```

AWS ParallelCluster 3:

```
...
HeadNode:
  ...
  CustomActions:
    OnNodeStart:
      Script: s3://amzn-s3-demo-bucket/scripts/pre_install.sh
      Args:
        - R
        - curl
        - wget
    OnNodeConfigured:
      Script: s3://amzn-s3-demo-bucket/scripts/post_install.sh
      Args: ['R', 'curl', 'wget']
  Iam:
    S3Access:
      - BucketName: amzn-s3-demo-bucket
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ...
      CustomActions:
        OnNodeStart:
          Script: s3://amzn-s3-demo-bucket/scripts/pre_install.sh
          Args: ['R', 'curl', 'wget']
        OnNodeConfigured:
          Script: s3://amzn-s3-demo-bucket/scripts/post_install.sh
          Args: ['R', 'curl', 'wget']
      Iam:
        S3Access:
          - BucketName: amzn-s3-demo-bucket
```

对 S3 存储桶资源具有读写访问权限的集群示例

AWS ParallelCluster 2:

```
[cluster default]
s3_read_resource = arn:aws:s3:::amzn-s3-demo-bucket/read_only/*
s3_read_write_resource = arn:aws:s3:::amzn-s3-demo-bucket/read_and_write/*
...
```

AWS ParallelCluster 3:

```
...
HeadNode:
  ...
  Iam:
    S3Access:
      - BucketName: amzn-s3-demo-bucket
        KeyName: read_only/
        EnableWriteAccess: False
      - BucketName: amzn-s3-demo-bucket
        KeyName: read_and_write/
        EnableWriteAccess: True
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ...
      Iam:
        S3Access:
          - BucketName: amzn-s3-demo-bucket
            KeyName: read_only/
            EnableWriteAccess: False
          - BucketName: amzn-s3-demo-bucket
            KeyName: read_and_write/
            EnableWriteAccess: True
```

包容性语言

AWS ParallelCluster 3 在 AWS ParallelCluster 2 中使用 “主节点” 的地方使用 “头节点” 一词。这包括以下这些：

- 在 AWS Batch 作业环境中导出的变量已更改：从变`MASTER_IP`为`PCLUSTER_HEAD_NODE_IP`。
- 所有 AWS CloudFormation 输出都从变`Master*`为`HeadNode*`。
- 全部 NodeType 和标签从变`Master`为`HeadNode`。

调度器支持

AWS ParallelCluster 3.x 不支持 Son of Grid Engine (SGE) 和 Torque 调度器。

AWS Batch 命令`awsbhosts`、、`awsbkill`、`awsboutawsbqueuesawsbstat`、和作为单独`awsbsub`的 `aws-parallelcluster-awsbatch-cli` PyPI 包分发。此程序包由 AWS ParallelCluster 安装在头节点上。您仍然可以在集群的头节点上使用这些 AWS Batch 命令。但如果您希望从头节点以外的其他位置使用 AWS Batch 命令，则必须先安装 `aws-parallelcluster-awsbatch-cli` PyPI 程序包。

AWS ParallelCluster CLI

AWS ParallelCluster 命令行界面 (CLI) 已更改。AWS ParallelCluster CLI 命令中描述了新语法。CLI 的输出格式为 JSON 字符串。

配置新集群

该`pcluster configure`命令在 AWS ParallelCluster 3 中包含不同的参数，而 AWS ParallelCluster 2 中包含的参数不同。有关更多信息，请参阅 `pcluster configure`。

另请注意，配置文件语法已从 AWS ParallelCluster 2 改变。有关集群配置设置的完整参考，请参阅集群配置文件。

创建新集群

AWS ParallelCluster 2 的`pcluster create`命令已被`pcluster create-cluster`命令所取代。

请注意， AWS ParallelCluster 2.x 中没有该`-nw`选项的默认行为是等待集群创建事件，而 AWS ParallelCluster 3.x 命令会立即返回。可以使用 `pcluster describe-cluster` 监控集群创建进度。

AWS ParallelCluster 3 配置文件包含单个集群定义，因此不再需要该`-t`参数。

下面是一个配置文件示例。

```
# AWS ParallelCluster v2
$ pcluster create \
    -r REGION \
    -c V2_CONFIG_FILE \
    -nw \
    -t CLUSTER_TEMPLATE \
    CLUSTER_NAME

# AWS ParallelCluster v3
```

```
$ pcluster create-cluster \
    --region REGION \
    --cluster-configuration V3_CONFIG_FILE \
    --cluster-name CLUSTER_NAME
```

列出集群

必须将 `pcluster list` AWS ParallelCluster 2.x 命令替换为`pcluster list-clusters`命令。

注意：你需要 AWS ParallelCluster v2 CLI 才能列出使用 2.x 版本的创建的集群。 AWS ParallelCluster 请参阅AWS ParallelCluster 在虚拟环境中安装（推荐），了解如何使用虚拟环境安装 AWS ParallelCluster 的多个版本。

```
# AWS ParallelCluster v2
$ pcluster list -r REGION

# AWS ParallelCluster v3
$ pcluster list-clusters --region REGION
```

启动和停止集群

`pcluster start`和 `pcluster stop` AWS ParallelCluster 2.x 命令必须替换为`pcluster update-compute-fleet`命令。

启动计算实例集：

```
# AWS ParallelCluster v2
$ pcluster start \
    -r REGION \
    CLUSTER_NAME

# AWS ParallelCluster v3 - Slurm fleets
$ pcluster update-compute-fleet \
    --region REGION \
    --cluster-name CLUSTER_NAME \
    --status START_REQUESTED

# AWS ParallelCluster v3 - AWS Batch fleets
$ pcluster update-compute-fleet \
    --region REGION \
    --cluster-name CLUSTER_NAME \
    --status ENABLED
```

停止计算实例集：

```
# AWS ParallelCluster v2
$ pcluster stop \
    -r REGION \
    CLUSTER_NAME

# AWS ParallelCluster v3 - Slurm fleets
$ pcluster update-compute-fleet \
    --region REGION \
    --cluster-name CLUSTER_NAME \
    --status STOP_REQUESTED

# AWS ParallelCluster v3 - AWS Batch fleets
$ pcluster update-compute-fleet \
    --region REGION \
    --cluster-name CLUSTER_NAME \
    --status DISABLED
```

连接到集群

`pcluster ssh` AWS ParallelCluster 2.x 命令在 AWS ParallelCluster 3.x 中有不同的参数名称。请参阅`pcluster ssh`。

连接到集群：

```
# AWS ParallelCluster v2
$ pcluster ssh \
    -r REGION \
    CLUSTER_NAME \
    -i ~/.ssh/id_rsa

# AWS ParallelCluster v3
$ pcluster ssh \
    --region REGION \
    --cluster-name CLUSTER_NAME \
    -i ~/.ssh/id_rsa
```

IMDS 配置更新

从版本 3.0.0 开始， AWS ParallelCluster 引入了对默认情况下仅限一部分超级用户访问头节点 IMDS（和实例配置文件凭证）的支持。有关更多信息，请参阅 `Imds 属性`。

使用 AWS ParallelCluster

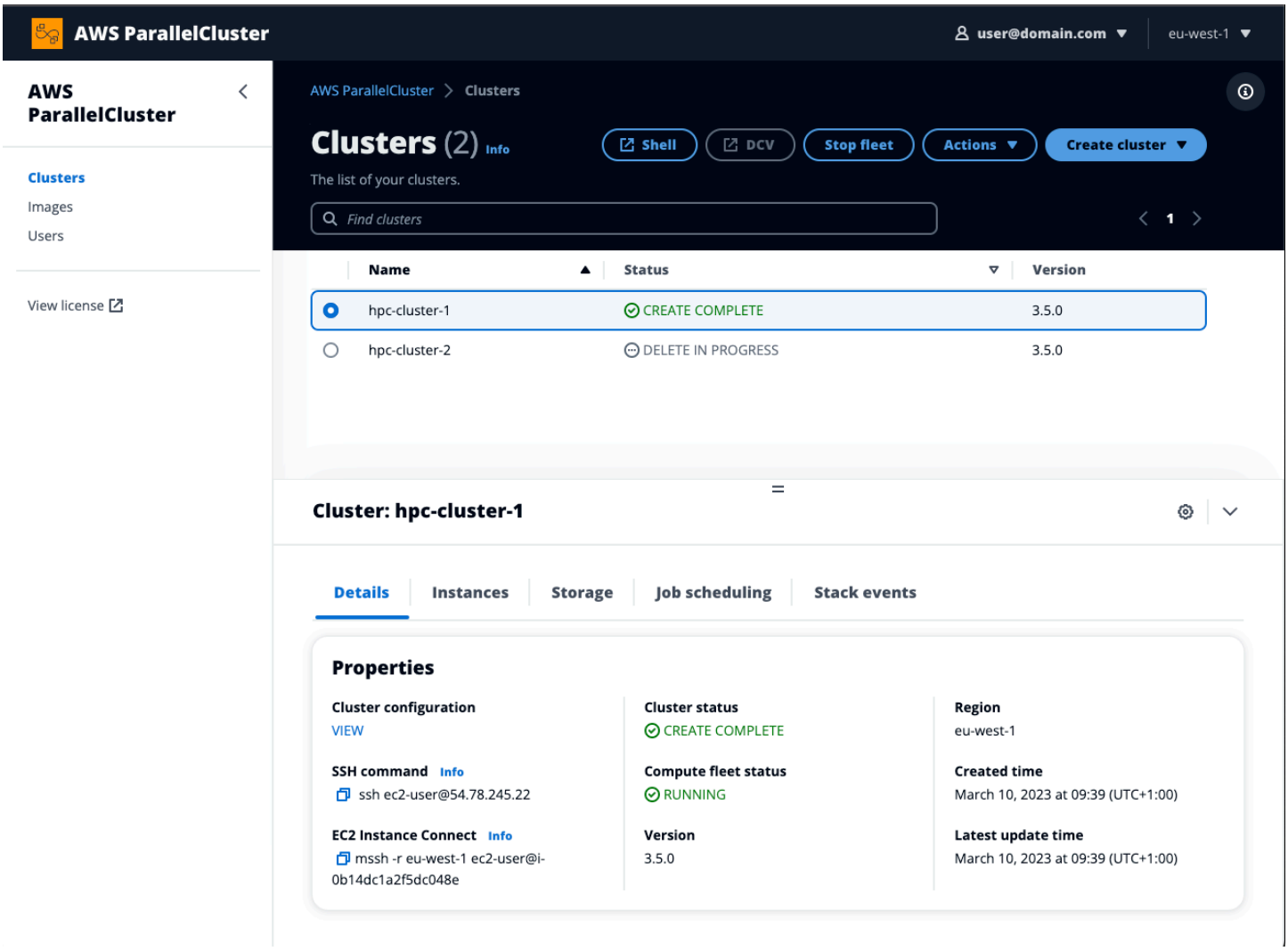
主题

- AWS ParallelCluster 用户界面
- AWS Lambda 中的 VPC 配置 AWS ParallelCluster
- AWS Identity and Access Management 中的权限 AWS ParallelCluster
- 网络配置
- 由配置的登录节点 AWS ParallelCluster
- 自定义引导操作
- 使用 Amazon S3
- 使用竞价型实例
- 支持的调度器 AWS ParallelCluster
- 共享存储
- AWS ParallelCluster 资源和标记
- 监控 AWS ParallelCluster 和日志
- AWS CloudFormation 自定义资源
- Elastic Fabric Adapter
- 启用 Intel MPI
- AWS ParallelCluster API
- AWS ParallelCluster 适用于 Terraform
- 通过 Amazon DCV 连接到头节点和登录节点
- 使用 pcluster update-cluster
- AWS ParallelCluster AMI 自定义
- 使用按需容量预留（ODCR）启动实例
- 使用容量块（CB）启动实例
- AMI 修补和亚马逊 EC2 实例更换
- 操作系统

AWS ParallelCluster 用户界面

用户 AWS ParallelCluster 界面 (PCUI) 是一个基于 Web 的用户界面，可用作用于创建、监控和管理集群的仪表板。您可以在中安装并访问 PCUI。 AWS 账户 PCUI 是在 3.5.0 AWS ParallelCluster 版本中添加的。

要安装并开始使用 PCUI，请参阅安装 PCUI和使用 PCUI 配置和创建集群。



PCUI 支持以下功能：

- 显示以下内容：
  - 您在使用中创建的集群列表 AWS ParallelCluster。 AWS 账户
  - 所列集群的可用状态和详细信息。
  - CloudFormation 堆栈事件和可用于监控的 AWS ParallelCluster 日志。

  - 您的集群上运行的作业的状态。
  - 可用于构建集群的自定义映像的列表。
  - UI 用于创建集群的官方映像的列表。
  - 有权访问 PCUI 的用户的列表。您可以添加和删除用户。
- 提供有关创建和编辑（更新）集群以及选择要添加、编辑或删除的支持的集群功能的 step-by-step指南。对于所编辑的集群配置，不能更改无法访问的输入字段。您可以选择在部署集群之前对集群配置进行试运行验证。
- 集群视图中具有用于访问头节点的直接 Shell 链接。在 step-by-step指导中选择 “添加 SSM 会话” 以添加直接 shell 访问权限，并在头节点上添加 SSM 托管实例核心策略。

在使用 PCUI 创建和管理集群时请考虑以下事项：

- 您只能使用与创建 PCUI 相同的 AWS ParallelCluster 版本创建和编辑集群或构建映像。只能查看早期版本的集群或映像。如果您管理多个版本的集群和映像，我们建议您创建支持每个版本的 PCUI 实例。
- PCUI 旨在镜像 `pcluster` CLI 功能。但存在一些区别。如果您符合 step-by-step指南，则说明您正在使用所有支持的功能。在部署之前，您可以选择手动编辑集群或映像配置。如果您这样做，我们建议您通过选择试运行来验证配置，以确认您的编辑完全受支持。

> **Note**
>
> PCUI 不支持 AWS Batch。

AWS Lambda 中的 VPC 配置 AWS ParallelCluster

AWS ParallelCluster AWS Lambda 用于在集群的生命周期内执行操作。AWS Lambda 函数始终在 Lambda 服务拥有的 VPC 内运行。也可以将此 Lambda 函数连接到虚拟私有云 (VPC) 中的私有子网以访问私有资源。

> **Note**
>
> Lambda 函数无法直接连接到具有专用实例租赁的 VPC。要连接到专用 VPC 中的资源，请将专用 VPC 对等连接到具有可以连接到专用 VPC 的默认租赁第二个 VPC。

有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的专用实例和如何将 Lambda 函数连接到专用 VPC？ 来自AWS 知识中心。

由创建的 Lambda 函数 AWS ParallelCluster 可以连接到私有 VPC。这些 Lambda 函数需要访问 AWS 服务。您可以使用以下方法通过互联网或 VPC 端点提供访问权限。

- 互联网访问

  要访问互联网和 AWS 服务，Lambda 函数需要网络地址转换 (NAT)。将来自您的私有子网的出站流量路由到公有子网中的 NAT 网关。
- VPC 端点

  有几项 AWS 服务提供 VPC 终端节点。您可以使用 VPC 终端节点 AWS 服务 从无法访问互联网的 VPC 进行连接。要查看 AWS ParallelCluster VPC 终端节点列表，请参阅网络。

> **Note**
>
> 子网和安全组的每种组合都必须 AWS 服务 使用其中一种方法提供访问权限。子网和安全组必须位于同一 VPC。

有关更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的 VPC 端点和 AWS Lambda 开发人员指南 中的 VPC 连接函数的 Internet 和服务访问。

要配置 Lambda 函数和的使用 VPCs，请参阅 `DeploymentSettings`/（有关集群）或 `DeploymentSettings`/（`LambdaFunctionsVpcConfig`有关图像）`LambdaFunctionsVpcConfig`。

AWS Identity and Access Management 中的权限 AWS ParallelCluster

AWS ParallelCluster 在创建和管理集群时使用 IAM 权限来控制对资源的访问权限。

要在 AWS 账户中创建和管理集群， AWS ParallelCluster 需要两个级别的权限：

- `pcluster` 用户调用 `pcluster` CLI 命令创建和管理集群所需的权限。
- 集群资源执行集群操作所需的权限。

AWS ParallelCluster 使用 A mazon EC2 实例配置文件和角色来提供集群资源权限。要管理集群资源权限， AWS ParallelCluster 还需要对 IAM 资源的权限。有关更多信息，请参阅 AWS ParallelCluster 用于管理 IAM 资源的用户示例策略。

pcluster 用户需要 IAM 权限才能使用 `pcluster` CLI 创建和管理集群及其资源。这些权限包含在可以添加到用户或角色的 IAM 策略中。有关 IAM 角色的更多信息，请参阅 AWS Identity and Access Management 用户指南 中的 创建用户角色。

您还可以使用 AWS ParallelCluster 用于管理 IAM 权限的配置参数。

以下各节包含所需的权限及示例。

要使用示例策略，请将 *<REGION>*、*<AWS ACCOUNT ID>* 和类似的字符串替换为相应的值。

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

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

主题

- AWS ParallelCluster Amazon EC2 实例角色
- AWS ParallelCluster pcluster用户策略示例
- AWS ParallelCluster 用于管理 IAM 资源的用户示例策略
- AWS ParallelCluster 用于管理 IAM 权限的配置参数

AWS ParallelCluster Amazon EC2 实例角色

使用默认配置设置创建集群时，会 AWS ParallelCluster 使用 Amazon EC2 实例配置文件自动创建默认集群 Amazon EC2 实例角色，该角色提供创建和管理集群及其资源所需的权限。

使用默认 AWS ParallelCluster 实例角色的替代方法

您可以使用`InstanceRole`集群配置设置来代替默认 AWS ParallelCluster 实例角色，为其指定自己的现有 IAM 角色 EC2。有关更多信息，请参阅 AWS ParallelCluster 用于管理 IAM 权限的配置参数。通常，您可以指定现有 IAM 角色来完全控制授予的权限 EC2。

如果您打算向默认实例角色添加额外的策略，我们建议您使用 `AdditionalIamPolicies` 配置设置而不是 `InstanceProfile` 或 `InstanceRole` 设置来传递其他 IAM 策略。您可以在更新集群时进行更新 `AdditionalIamPolicies`，但不能在更新集群时更新 `InstanceRole`。

AWS ParallelCluster **`pcluster`**用户策略示例

以下示例显示了使用 `pcluster` CLI 创建 AWS ParallelCluster 和管理其资源所需的用户策略。您可以将策略附加到用户或角色。

主题

- 基本 AWS ParallelCluster pcluster 用户策略
- 使用 AWS Batch 调度器时的其他 AWS ParallelCluster pcluster 用户策略
- 使用 Amazon FSx for Lustre 时的其他 AWS ParallelCluster pcluster用户政策
- AWS ParallelCluster 镜像构建pcluster用户政策

基本 AWS ParallelCluster **`pcluster`** 用户策略

以下策略显示了运行 AWS ParallelCluster `pcluster`命令所需的权限。

策略中列出的最后一个操作用于验证集群配置中指定的任何密钥。例如， AWS Secrets Manager 密钥用于配置集`DirectoryService`成。在这种情况下，只有当 `PasswordSecretArn` 中存在有效密钥时，才会创建集群。如果省略此操作，则会跳过密钥验证。为了改善您的安全状况，我们建议您通过仅添加集群配置中指定的密钥来缩小此策略声明的范围。

> **Note**
>
> 如果现有 Amazon EFS 文件系统是集群中使用的唯一文件系统，则可以将示例 Amazon EFS 策略声明的范围缩小到集群配置文件中 `SharedStorage` 部分 引用的特定文件系统。

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Action": [
              "ec2:Describe*"
          ],
          "Resource": "*",
          "Effect": "Allow",
```

```
            "Sid": "EC2Read"
        },
        {
            "Action": [
                "ec2:AllocateAddress",
                "ec2:AssociateAddress",
                "ec2:AttachNetworkInterface",
                "ec2:AuthorizeSecurityGroupEgress",
                "ec2:AuthorizeSecurityGroupIngress",
                "ec2:CreateFleet",
                "ec2:CreateLaunchTemplate",
                "ec2:CreateLaunchTemplateVersion",
                "ec2:CreateNetworkInterface",
                "ec2:CreatePlacementGroup",
                "ec2:CreateSecurityGroup",
                "ec2:CreateSnapshot",
                "ec2:CreateTags",
                "ec2:DeleteTags",
                "ec2:CreateVolume",
                "ec2:DeleteLaunchTemplate",
                "ec2:DeleteNetworkInterface",
                "ec2:DeletePlacementGroup",
                "ec2:DeleteSecurityGroup",
                "ec2:DeleteVolume",
                "ec2:DisassociateAddress",
                "ec2:ModifyLaunchTemplate",
                "ec2:ModifyNetworkInterfaceAttribute",
                "ec2:ModifyVolume",
                "ec2:ModifyVolumeAttribute",
                "ec2:ReleaseAddress",
                "ec2:RevokeSecurityGroupEgress",
                "ec2:RevokeSecurityGroupIngress",
                "ec2:RunInstances",
                "ec2:TerminateInstances"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2Write"
        },
        {
            "Action": [
                "dynamodb:DescribeTable",
                "dynamodb:ListTagsOfResource",
                "dynamodb:CreateTable",
```

```
                "dynamodb:DeleteTable",
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:Query",
                "dynamodb:TagResource"
            ],
            "Resource": "arn:aws:dynamodb:*:<AWS ACCOUNT ID>:table/parallelcluster-*",
            "Effect": "Allow",
            "Sid": "DynamoDB"
        },
        {
            "Action": [
                "route53:ChangeResourceRecordSets",
                "route53:ChangeTagsForResource",
                "route53:CreateHostedZone",
                "route53:DeleteHostedZone",
                "route53:GetChange",
                "route53:GetHostedZone",
                "route53:ListResourceRecordSets",
                "route53:ListQueryLoggingConfigs"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "Route53HostedZones"
        },
        {
            "Action": [
                "cloudformation:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "cloudwatch:PutDashboard",
                "cloudwatch:ListDashboards",
                "cloudwatch:DeleteDashboards",
                "cloudwatch:GetDashboard",
                "cloudwatch:PutMetricAlarm",
                "cloudwatch:DeleteAlarms",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutCompositeAlarm"
```

```
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudWatch"
        },
        {
            "Action": [
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetPolicy",
                "iam:SimulatePrincipalPolicy",
                "iam:GetInstanceProfile"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/*",
                "arn:aws:iam::<AWS ACCOUNT ID>:policy/*",
                "arn:aws:iam::aws:policy/*",
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/*"
            ],
            "Effect": "Allow",
            "Sid": "IamRead"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:DeleteInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "IamInstanceProfile"
        },
        {
            "Condition": {
                "StringEqualsIfExists": {
                    "iam:PassedToService": [
                        "lambda.amazonaws.com",
                        "ec2.amazonaws.com",
                        "spotfleet.amazonaws.com"
                    ]
                }
```

```
            },
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "IamPassRole"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
                "lambda:GetFunctionConfiguration",
                "lambda:GetFunction",
                "lambda:InvokeFunction",
                "lambda:AddPermission",
                "lambda:RemovePermission",
                "lambda:UpdateFunctionConfiguration",
                "lambda:TagResource",
                "lambda:ListTags",
                "lambda:UntagResource"
            ],
            "Resource": [
                "arn:aws:lambda:*:<AWS ACCOUNT ID>:function:parallelcluster-*",
                "arn:aws:lambda:*:<AWS ACCOUNT ID>:function:pcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Action": [
                "s3:*"
            ],
            "Resource": [
                "arn:aws:s3:::parallelcluster-*",
                "arn:aws:s3:::aws-parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "S3ResourcesBucket"
        },
        {
            "Action": [
```

```
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": "arn:aws:s3:::*-aws-parallelcluster*",
            "Effect": "Allow",
            "Sid": "S3ParallelClusterReadOnly"
        },
        {
            "Action": [
                "elasticfilesystem:*"
            ],
            "Resource": [
                "arn:aws:elasticfilesystem:*:<AWS ACCOUNT ID>:*"
            ],
            "Effect": "Allow",
            "Sid": "EFS"
        },
        {
            "Action": [
                "logs:DeleteLogGroup",
                "logs:PutRetentionPolicy",
                "logs:DescribeLogGroups",
                "logs:CreateLogGroup",
                "logs:TagResource",
                "logs:UntagResource",
                "logs:FilterLogEvents",
                "logs:GetLogEvents",
                "logs:CreateExportTask",
                "logs:DescribeLogStreams",
                "logs:DescribeExportTasks",
                "logs:DescribeMetricFilters",
                "logs:PutMetricFilter",
                "logs:DeleteMetricFilter"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "CloudWatchLogs"
        },
        {
            "Action": [
                "resource-groups:ListGroupResources"
            ],
            "Resource": "*",
            "Effect": "Allow",
```

```
            "Sid": "ResourceGroupRead"
        },
        {
            "Sid": "AllowDescribingFileCache",
            "Effect": "Allow",
            "Action": [
                "fsx:DescribeFileCaches"
            ],
            "Resource": "*"
        },
        {
            "Action": "secretsmanager:DescribeSecret",
            "Resource": "arn:aws:secretsmanager:<REGION>:<AWS ACCOUNT ID>:secret:<SECRET
 NAME>",
            "Effect": "Allow"
        }
    ]
}
```

使用 AWS Batch 调度器时的其他 AWS ParallelCluster `pcluster` 用户策略

如果您需要使用 AWS Batch 调度程序创建和管理集群，则需要以下附加策略。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Condition": {
                "StringEqualsIfExists": {
                    "iam:PassedToService": [
                        "ecs-tasks.amazonaws.com",
                        "batch.amazonaws.com",
                        "codebuild.amazonaws.com"
                    ]
                }
            },
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
            ],
            "Effect": "Allow",
```

```
            "Sid": "IamPassRole"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:AWSServiceName": [
                        "batch.amazonaws.com"
                    ]
                }
            },
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:DeleteServiceLinkedRole"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/aws-service-role/
batch.amazonaws.com/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "codebuild:*"
            ],
            "Resource": "arn:aws:codebuild:*:<AWS ACCOUNT ID>:project/pcluster-*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "ecr:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "ECR"
        },
        {
            "Action": [
                "batch:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "Batch"
        },
        {
```

```
            "Action": [
                "events:*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "AmazonCloudWatchEvents"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:ListContainerInstances"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "ECS"
        }
    ]
}
```

使用 Amazon FSx for Lustre 时的其他 AWS ParallelCluster `pcluster`用户政策

如果您需要使用 Amazon FSx for Lustre 创建和管理集群，则需要遵循以下附加策略。

> **Note**
>
> 如果现有的 Amazon FSx 文件系统是您的集群中唯一使用的文件系统，则可以将示例 Amazon FSx 策略声明的范围缩小到集群配置文件中引用的特定文件系统。SharedStorage 部分

```
{
  "Version": "2012-10-17",
  "Statement": [
      {
          "Condition": {
              "StringEquals": {
                  "iam:AWSServiceName": [
                      "fsx.amazonaws.com",
                      "s3.data-source.lustre.fsx.amazonaws.com"
                  ]
              }
          },
          "Action": [
```

```
                "iam:CreateServiceLinkedRole",
                "iam:DeleteServiceLinkedRole"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "fsx:*"
            ],
            "Resource": [
                "arn:aws:fsx:*:<AWS ACCOUNT ID>:*"
            ],
            "Effect": "Allow",
            "Sid": "FSx"
        },
        {
            "Action": [
                "iam:CreateServiceLinkedRole",
                "iam:AttachRolePolicy",
                "iam:PutRolePolicy"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/aws-service-role/s3.data-
source.lustre.fsx.amazonaws.com/*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "s3:Get*",
                "s3:List*",
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::<S3 NAME>",
            "Effect": "Allow"
        }
    ]
}
```

AWS ParallelCluster 镜像构建**pcluster**用户政策

打算使用其创建自定义 Amazon EC2 图片的用户 AWS ParallelCluster 必须具有以下一组权限。

```
{
```

```
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:DescribeSecurityGroups",
                "ec2:DescribeImages",
                "ec2:DescribeInstanceTypeOfferings",
                "ec2:DescribeInstanceTypes",
                "ec2:DeregisterImage",
                "ec2:DeleteSnapshot"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "EC2"
        },
        {
            "Action": [
                "iam:CreateInstanceProfile",
                "iam:AddRoleToInstanceProfile",
                "iam:GetRole",
                "iam:GetRolePolicy",
                "iam:GetInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/parallelcluster/*",
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/ParallelClusterImage*",
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "IAM"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "lambda.amazonaws.com",
                        "ec2.amazonaws.com"
                    ]
                }
            },
            "Action": [
                "iam:PassRole"
            ],
```

```
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/parallelcluster/*",
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "IAMPassRole"
        },
        {
            "Action": [
                "logs:GetLogEvents",
                "logs:CreateLogGroup",
                "logs:TagResource",
                "logs:UntagResource",
                "logs:DeleteLogGroup"
            ],
            "Resource": [
                "arn:aws:logs:*:<AWS ACCOUNT ID>:log-group:/aws/imagebuilder/
ParallelClusterImage-*",
                "arn:aws:logs:*:<AWS ACCOUNT ID>:log-group:/aws/lambda/
ParallelClusterImage-*"
            ],
            "Effect": "Allow",
            "Sid": "CloudWatch"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack"
            ],
            "Resource": [
                "arn:aws:cloudformation:*:<AWS ACCOUNT ID>:stack/*"
            ],
            "Effect": "Allow",
            "Sid": "CloudFormation"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "lambda:GetFunction",
                "lambda:AddPermission",
                "lambda:RemovePermission",
                "lambda:DeleteFunction",
                "lambda:TagResource",
```

```
                "lambda:ListTags",
                "lambda:UntagResource"
            ],
            "Resource": [
                "arn:aws:lambda:*:<AWS ACCOUNT ID>:function:ParallelClusterImage-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Action": [
                "imagebuilder:Get*"
            ],
            "Resource": "*",
            "Effect": "Allow",
            "Sid": "ImageBuilderGet"
        },
        {
            "Action": [
                "imagebuilder:CreateImage",
                "imagebuilder:TagResource",
                "imagebuilder:CreateImageRecipe",
                "imagebuilder:CreateComponent",
                "imagebuilder:CreateDistributionConfiguration",
                "imagebuilder:CreateInfrastructureConfiguration",
                "imagebuilder:DeleteImage",
                "imagebuilder:DeleteComponent",
                "imagebuilder:DeleteImageRecipe",
                "imagebuilder:DeleteInfrastructureConfiguration",
                "imagebuilder:DeleteDistributionConfiguration"
            ],
            "Resource": [
                "arn:aws:imagebuilder:*:<AWS ACCOUNT ID>:image/parallelclusterimage-*",
                "arn:aws:imagebuilder:*:<AWS ACCOUNT ID>:image-recipe/
parallelclusterimage-*",
                "arn:aws:imagebuilder:*:<AWS ACCOUNT ID>:component/
parallelclusterimage-*",
                "arn:aws:imagebuilder:*:<AWS ACCOUNT ID>:distribution-configuration/
parallelclusterimage-*",
                "arn:aws:imagebuilder:*:<AWS ACCOUNT ID>:infrastructure-configuration/
parallelclusterimage-*"
            ],
            "Effect": "Allow",
            "Sid": "ImageBuilder"
```

```
        },
        {
            "Action": [
                "s3:CreateBucket",
                "s3:ListBucket",
                "s3:ListBucketVersions"
            ],
            "Resource": [
                "arn:aws:s3:::parallelcluster-*"
            ],
            "Effect": "Allow",
            "Sid": "S3Bucket"
        },
        {
            "Action": [
                "sns:GetTopicAttributes",
                "sns:TagResource",
                "sns:CreateTopic",
                "sns:Subscribe",
                "sns:Publish",
                "SNS:DeleteTopic",
                "SNS:Unsubscribe"
            ],
            "Resource": [
                "arn:aws:sns:*:<AWS ACCOUNT ID>:ParallelClusterImage-*"
            ],
            "Effect": "Allow",
            "Sid": "SNS"
        },
        {
            "Action": [
                "s3:PutObject",
                "s3:GetObject",
                "s3:GetObjectVersion",
                "s3:DeleteObject",
                "s3:DeleteObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::parallelcluster-*/*"
            ],
            "Effect": "Allow",
            "Sid": "S3Objects"
        },
        {
```

```
            "Action": "iam:CreateServiceLinkedRole",
            "Effect": "Allow",
            "Resource": "arn:aws:iam::*:role/aws-service-role/
imagebuilder.amazonaws.com/AWSServiceRoleForImageBuilder",
            "Condition": {
                "StringLike": {
                    "iam:AWSServiceName": "imagebuilder.amazonaws.com"
                }
            }
        }
    ]
}
```

AWS ParallelCluster 用于管理 IAM 资源的用户示例策略

使用 AWS ParallelCluster 创建集群或自定义集群时 AMIs，必须提供包含向 AWS ParallelCluster 组件授予所需权限集的权限的 IAM 策略。在创建集群 AWS ParallelCluster 或自定义映像时，这些 IAM 资源可以由自动创建，也可以作为输入提供。

您可以使用以下模式通过在配置中使用其他 IAM 策略为 AWS ParallelCluster 用户提供访问 IAM 资源所需的权限。

主题

- 特权 IAM 访问模式
- 受限的 IAM 访问模式
- PermissionsBoundary 模式

特权 IAM 访问模式

在此模式下， AWS ParallelCluster 会自动创建所有必要的 IAM 资源。这些 IAM 策略的范围已缩小，仅允许访问集群资源。

要启用特权 IAM 访问模式，请向用户角色添加以下策略。

> **Note**
>
> 如果您配置 `HeadNode`/`Iam`/`AdditionalPolicies`或 `Scheduling`//`SlurmQueuesIam`/`AdditionalPolicies`参数，则必须向 AWS

ParallelCluster 用户提供为每个其他策略附加和分离角色策略的权限，如以下策略所示。将附加策略 ARNs 添加到附加和分离角色策略的条件中。

> **Warning**
>
> 此模式使用户能够在中拥有 IAM 管理员权限 AWS 账户

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
           "Action": [
               "iam:CreateServiceLinkedRole",
               "iam:DeleteRole",
               "iam:TagRole"
           ],
           "Resource": [
               "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
           ],
           "Effect": "Allow",
           "Sid": "IamRole"
       },
       {
           "Action": [
               "iam:CreateRole"
           ],
           "Resource": [
               "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
           ],
           "Effect": "Allow",
           "Sid": "IamCreateRole"
       },
       {
           "Action": [
               "iam:PutRolePolicy",
               "iam:DeleteRolePolicy"
           ],
           "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
           "Effect": "Allow",
           "Sid": "IamInlinePolicy"
```

```
        },
        {
            "Condition": {
                "ArnLike": {
                    "iam:PolicyARN": [
                        "arn:aws:iam::<AWS ACCOUNT ID>:policy/parallelcluster*",
                        "arn:aws:iam::<AWS ACCOUNT ID>:policy/parallelcluster/*",
                        "arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy",
                        "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
                        "arn:aws:iam::aws:policy/AWSBatchFullAccess",
                        "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess",
                        "arn:aws:iam::aws:policy/service-role/AWSBatchServiceRole",
                        "arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role",
                        "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy",
                        "arn:aws:iam::aws:policy/service-role/
AmazonEC2SpotFleetTaggingRole",
                        "arn:aws:iam::aws:policy/EC2InstanceProfileForImageBuilder",
                        "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"
                    ]
                }
            },
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
            "Effect": "Allow",
            "Sid": "IamPolicy"
        }
    ]
}
```

受限的 IAM 访问模式

如果没有向用户授予其他 IAM 策略，则集群或自定义映像构建所需的 IAM 角色需要由管理员手动创建，并作为集群配置的一部分进行传递。

创建集群时，需要使用以下参数：

- Iam / Roles / LambdaFunctionsRole

- HeadNode / Iam / InstanceRole | InstanceProfile
- Scheduling / SlurmQueues / Iam / InstanceRole | InstanceProfile

构建自定义映像时，需要使用以下参数：

- Build / Iam / InstanceRole | InstanceProfile
- Build / Iam / CleanupLambdaRole

作为上面所列参数的一部分传递的 IAM 角色必须以 `/parallelcluster/` 路径前缀进行创建。如果无法做到这一点，则需要更新用户策略以便对特定自定义角色授予 `iam:PassRole` 权限，如以下示例所示。

```
{
   "Condition": {
       "StringEqualsIfExists": {
           "iam:PassedToService": [
               "ecs-tasks.amazonaws.com",
               "lambda.amazonaws.com",
               "ec2.amazonaws.com",
               "spotfleet.amazonaws.com",
               "batch.amazonaws.com",
               "codebuild.amazonaws.com"
           ]
       }
   },
   "Action": [
       "iam:PassRole"
   ],
   "Resource": [
       <list all custom IAM roles>
   ],
   "Effect": "Allow",
   "Sid": "IamPassRole"
}
```

> **Warning**
>
> 目前，此模式不允许管理 AWS Batch 集群，因为并非所有 IAM 角色都可以在集群配置中传递。

PermissionsBoundary 模式

此模式委托创建绑 AWS ParallelCluster 定到配置的 IAM 权限边界的 IAM 角色。有关 IAM 权限边界的更多信息，请参阅 IAM 用户指南 中的 IAM 实体的权限边界。

需要将以下策略添加到用户角色。

在策略中，*<permissions-boundary-arn>*替换为要作为权限边界强制执行的 IAM 策略 ARN。

> **Warning**
>
> 如果您配置 HeadNode/Iam/AdditionalPolicies 或 Scheduling/SlurmQueues/Iam/ 参数，则必须向用户授予为每个其他策略附加和分离角色策略的权限，如以下策略所示。将附加策略 ARNs 添加到附加和分离角色策略的条件中。

```
{
   "Version": "2012-10-17",
   "Statement": [
       {
           "Action": [
               "iam:CreateServiceLinkedRole",
               "iam:DeleteRole",
               "iam:TagRole"
           ],
           "Resource": [
               "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
           ],
           "Effect": "Allow",
           "Sid": "IamRole"
       },
       {
           "Condition": {
               "StringEquals": {
                   "iam:PermissionsBoundary": [
                       <permissions-boundary-arn>
                   ]
               }
           },
           "Action": [
               "iam:CreateRole"
           ],
```

```
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*"
            ],
            "Effect": "Allow",
            "Sid": "IamCreateRole"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": [
                        <permissions-boundary-arn>
                    ]
                }
            },
            "Action": [
                "iam:PutRolePolicy",
                "iam:DeleteRolePolicy"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
            "Effect": "Allow",
            "Sid": "IamInlinePolicy"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PermissionsBoundary": [
                        <permissions-boundary-arn>
                    ]
                },
                "ArnLike": {
                    "iam:PolicyARN": [
                        "arn:aws:iam::<AWS ACCOUNT ID>:policy/parallelcluster*",
                        "arn:aws:iam::<AWS ACCOUNT ID>:policy/parallelcluster/*",
                        "arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy",
                        "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore",
                        "arn:aws:iam::aws:policy/AWSBatchFullAccess",
                        "arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess",
                        "arn:aws:iam::aws:policy/service-role/AWSBatchServiceRole",
                        "arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role",
                        "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy",
                        "arn:aws:iam::aws:policy/service-role/
AmazonEC2SpotFleetTaggingRole",
```

```
                        "arn:aws:iam::aws:policy/EC2InstanceProfileForImageBuilder",
                        "arn:aws:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole"
                    ]
                }
            },
            "Action": [
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
            "Effect": "Allow",
            "Sid": "IamPolicy"
        }
    ]
}
```

启用此模式后，创建或更新集群时必须在 `Iam`/`PermissionsBoundary` 配置参数中指定权限边界 ARN，在构建自定义映像时必须在 `Build`/`Iam`/`PermissionBoundary` 参数中指定权限边界 ARN。

AWS ParallelCluster 用于管理 IAM 权限的配置参数

AWS ParallelCluster 公开了一系列配置选项，用于自定义和管理集群中或自定义 AMI 创建过程中使用的 IAM 权限和角色。

主题

- 集群配置
- 自定义映像配置

集群配置

主题

- 头节点 IAM 角色
- Amazon S3 访问权限
- 其他 IAM 策略
- AWS Lambda 函数角色
- 计算节点 IAM 角色
- 权限边界

头节点 IAM 角色

HeadNode / Iam / InstanceRole | InstanceProfile

使用此选项，您可以覆盖分配给集群头节点的默认 IAM 角色。有关更多详细信息，请参阅 InstanceProfile 参考。

以下是当调度器为 Slurm 时作为该角色一部分使用的一组最少策略：

- `arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy` 托管的 IAM 策略 有关更多信息，请参阅 A mazon 用户指南中的创建用于 CloudWatch 代理的 IAM 角色和 CloudWatch 用户。
- `arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore` 托管的 IAM 策略。有关更多信息，请参阅 AWS Systems Manager 用户指南 中的用于 AWS Systems Manager的AWS 托管策略。
- 其他 IAM 策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*",
                "arn:aws:s3:::dcv-license.<REGION>/*",
                "arn:aws:s3:::parallelcluster-*-v1-do-not-delete/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "dynamodb:GetItem",
                "dynamodb:PutItem",
                "dynamodb:UpdateItem",
                "dynamodb:BatchWriteItem",
                "dynamodb:BatchGetItem"
            ],
            "Resource": "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/
parallelcluster-*",
            "Effect": "Allow"
        },
```

```
        {
            "Condition": {
                "StringEquals": {
                    "ec2:ResourceTag/parallelcluster:node-type": "Compute"
                }
            },
            "Action": "ec2:TerminateInstances",
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "ec2:RunInstances",
                "ec2:CreateFleet"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            },
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/parallelcluster/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeVolumes",
                "ec2:DescribeInstanceAttribute",
                "ec2:DescribeCapacityReservations"
            ],
            "Resource": "*",
```

```
            "Effect": "Allow"
        },
        {
            "Action": [
                "ec2:CreateTags",
                "ec2:AttachVolume"
            ],
            "Resource": [
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:instance/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:volume/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:DescribeStackResource",
                "cloudformation:SignalResource"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "route53:ChangeResourceRecordSets"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:<REGION>:<AWS ACCOUNT
 ID>:secret:<SECRET_ID>",
            "Effect": "Allow"
        }
    ]
}
```

请注意，如果使用 Scheduling/SlurmQueues/Iam/InstanceRole 来覆盖计算 IAM 角色，则上面报告的头节点策略需要在 `iam:PassRole` 权限的 `Resource` 部分中包含此类角色。

以下是当调度器为 AWS Batch时作为该角色一部分使用的一组最少策略：

- `arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy` 托管的 IAM 策略。有关更多信息，请参阅 A mazon 用户指南中的创建用于 CloudWatch 代理的 IAM 角色和 CloudWatch 用户。
- `arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore` 托管的 IAM 策略。有关更多信息，请参阅 AWS Systems Manager 用户指南 中的用于 AWS Systems Manager的AWS 托管策略。
- 其他 IAM 策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "s3:GetObject",
                "s3:PutObject",
                "s3:GetObjectVersion"
            ],
            "Resource": [
                "arn:aws:s3:::parallelcluster-*-v1-do-not-delete/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": "s3:GetObject",
            "Resource": [
                "arn:aws:s3:::dcv-license.<REGION>/*",
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
            ],
            "Effect": "Allow"
        },
        {
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "batch.amazonaws.com"
                    ]
                }
            },
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
```

```
                "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/parallelcluster/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "batch:DescribeJobQueues",
                "batch:DescribeJobs",
                "batch:ListJobs",
                "batch:DescribeComputeEnvironments"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "batch:SubmitJob",
                "batch:TerminateJob",
                "logs:GetLogEvents",
                "ecs:ListContainerInstances",
                "ecs:DescribeContainerInstances"
            ],
            "Resource": [
                "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:log-group:/aws/batch/job:log-
stream:PclusterJobDefinition*",
                "arn:aws:ecs:<REGION>:<AWS ACCOUNT ID>:container-instance/AWSBatch-
PclusterComputeEnviron*",
                "arn:aws:ecs:<REGION>:<AWS ACCOUNT ID>:cluster/AWSBatch-Pcluster*",
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-queue/
PclusterJobQueue*",
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job-definition/
PclusterJobDefinition*:*",
                "arn:aws:batch:<REGION>:<AWS ACCOUNT ID>:job/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "ec2:DescribeInstances",
                "ec2:DescribeInstanceStatus",
                "ec2:DescribeVolumes",
                "ec2:DescribeInstanceAttribute"
            ],
            "Resource": "*",
```

```
            "Effect": "Allow"
        },
        {
            "Action": [
                "ec2:CreateTags",
                "ec2:AttachVolume"
            ],
            "Resource": [
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:instance/*",
                "arn:aws:ec2:<REGION>:<AWS ACCOUNT ID>:volume/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": [
                "cloudformation:DescribeStackResource",
                "cloudformation:DescribeStacks",
                "cloudformation:SignalResource"
            ],
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:<REGION>:<AWS ACCOUNT
 ID>:secret:<SECRET_ID>",
            "Effect": "Allow"
        }
    ]
}
```

Amazon S3 访问权限

HeadNode/Iam/S3Access 或 Scheduling/SlurmQueues/S3Access

在这些配置部分中，您可以在 AWS ParallelCluster创建与集群的头节点或计算节点关联的 IAM 角色时向这些角色授予其他 Amazon S3 策略来自定义 Amazon S3 访问权限。有关更多信息，请参阅每个配置参数的参考文档。

只有在使用 特权 IAM 访问模式 或 PermissionsBoundary 模式 来配置用户时，才能使用此参数。

其他 IAM 策略

`HeadNode`/`Iam`/`AdditionalIamPolicies` 或 `SlurmQueues`/`Iam`/`AdditionalIamPolicies`

使用此选项将其他托管 IAM 策略附加到与集群的头节点或计算节点关联的 IAM 角色（如果这些角色由创建） AWS ParallelCluster。

> **Warning**
> 要使用此选项，请确保针对需要附加的 IAM 策略向 AWS ParallelCluster 用户授予 `iam:AttachRolePolicy` 和 `iam:DetachRolePolicy` 权限。

AWS Lambda 函数角色

`Iam` / `Roles` / `LambdaFunctionsRole`

此选项将覆盖集群创建过程中使用的所有 AWS Lambda 函数所附加的角色。 AWS Lambda 需要配置为允许担任该角色的委托人。

> **Note**
> 如果设置了 `DeploymentSettings`/`LambdaFunctionsVpcConfig`，则 `LambdaFunctionsRole` 必须包括用于设置 VPC 配置的 AWS Lambda 角色权限。

以下是作为该角色一部分使用的一组最少策略：

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "route53:ListResourceRecordSets",
        "route53:ChangeResourceRecordSets"
      ],
      "Resource": "arn:aws:route53:::hostedzone/*",
      "Effect": "Allow"
    },
    {
      "Action": ["logs:CreateLogStream", "logs:PutLogEvents"],
      "Effect": "Allow",
```

```
      "Resource": "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:log-group:/aws/lambda/
pcluster-*"
    },
    {
      "Action": "ec2:DescribeInstances",
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": "ec2:TerminateInstances",
      "Condition": {
        "StringEquals": {
          "ec2:ResourceTag/parallelcluster:node-type": "Compute"
        }
      },
      "Effect": "Allow",
      "Resource": "*"
    },
    {
      "Action": [
        "s3:DeleteObject",
        "s3:DeleteObjectVersion",
        "s3:ListBucket",
        "s3:ListBucketVersions"
      ],
      "Effect": "Allow",
      "Resource": [
        "arn:aws:s3:::parallelcluster-*-v1-do-not-delete",
        "arn:aws:s3:::parallelcluster-*-v1-do-not-delete/*"
      ]
    }
  ]
}
```

计算节点 IAM 角色

Scheduling / SlurmQueues / Iam / _InstanceRole | InstanceProfile

此选项允许覆盖分配给集群计算节点的 IAM 角色。有关更多信息，请参阅 InstanceProfile。

以下是作为该角色一部分使用的一组最少策略：

- `arn:aws:iam::aws:policy/CloudWatchAgentServerPolicy` 托管的 IAM 策略。有关更多信息，请参阅 A mazon 用户指南中的创建用于 CloudWatch代理的 IAM 角色和 CloudWatch 用户。

- `arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore` 托管的 IAM 策略。有关更多信息，请参阅 AWS Systems Manager 用户指南 中的用于 AWS Systems Manager的AWS 托管策略。
- 其他 IAM 策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "dynamodb:Query",
                "dynamodb:UpdateItem",
                "dynamodb:PutItem",
                "dynamodb:GetItem"
            ],
            "Resource": "arn:aws:dynamodb:<REGION>:<AWS ACCOUNT ID>:table/
parallelcluster-*",
            "Effect": "Allow"
        },
        {
            "Action": "s3:GetObject",
            "Resource": [
                "arn:aws:s3:::<REGION>-aws-parallelcluster/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": "ec2:DescribeInstanceAttribute",
            "Resource": "*",
            "Effect": "Allow"
        },

        {
            "Action": "cloudformation:DescribeStackResource",
            "Resource":
                [ "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/*/*" ],
            "Effect" "Allow"
        }
    ]
}
```

权限边界

`Iam` / `PermissionsBoundary`

此参数强制 AWS ParallelCluster 将给定的 IAM 策略作为 a 附加`PermissionsBoundary`到在集群部署中创建的所有 IAM 角色。

有关定义此设置后用户所需的策略的列表，请参阅 `PermissionsBoundary` 模式。

自定义映像配置

主题

- EC2 Image Builder 的实例角色
- AWS Lambda 清理角色
- 其他 IAM 策略
- 权限边界

EC2 Image Builder 的实例角色

`Build` / `Iam` / `InstanceRole` | `InstanceProfile`

使用此选项，您可以覆盖分配给 I EC2 mage Builder 启动的亚马逊 EC2 实例的 IAM 角色来创建自定义 AMI。

以下是作为该角色一部分使用的一组最少策略：

- `arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore` 托管的 IAM 策略。有关更多信息，请参阅 AWS Systems Manager 用户指南 中的用于 AWS Systems Manager的AWS 托管策略。
- `arn:aws:iam::aws:policy/EC2InstanceProfileForImageBuilder` 托管的 IAM 策略。有关更多信息，请参阅 Image Builder User Guide 中的 `EC2InstanceProfileForImageBuilder` policy。
- 其他 IAM 策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "ec2:CreateTags",
```

```
                "ec2:ModifyImageAttribute"
            ],
            "Resource": "arn:aws:ec2:<REGION>::image/*",
            "Effect": "Allow"
        }
    ]
}
```

AWS Lambda 清理角色

Build / Iam / CleanupLambdaRole

此选项将覆盖自定义映像构建过程中使用的所有 AWS Lambda 函数所附加的角色。 AWS Lambda 需要配置为允许担任该角色的委托人。

> **Note**
>
> 如果设置了 DeploymentSettings/LambdaFunctionsVpcConfig，则 CleanupLambdaRole 必须包括用于设置 VPC 配置的 AWS Lambda 角色权限。

以下是作为该角色一部分使用的一组最少策略：

- arn:aws:iam::aws:policy/service-role/AWSLambdaBasicExecutionRole 托管的 IAM 策略。有关更多信息，请参阅 AWS Lambda 开发人员指南 中的 Lambda 功能的AWS 托管策略。
- 其他 IAM 策略：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "iam:DetachRolePolicy",
                "iam:DeleteRole",
                "iam:DeleteRolePolicy"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:role/parallelcluster/*",
            "Effect": "Allow"
        },
        {
            "Action": [
```

```
                "iam:DeleteInstanceProfile",
                "iam:RemoveRoleFromInstanceProfile"
            ],
            "Resource": "arn:aws:iam::<AWS ACCOUNT ID>:instance-profile/
parallelcluster/*",
            "Effect": "Allow"
        },
        {
            "Action": "imagebuilder:DeleteInfrastructureConfiguration",
            "Resource": "arn:aws:imagebuilder:<REGION>:<AWS ACCOUNT
 ID>:infrastructure-configuration/parallelclusterimage-*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "imagebuilder:DeleteComponent"
            ],
            "Resource": [
                "arn:aws:imagebuilder:<REGION>:<AWS ACCOUNT ID>:component/
parallelclusterimage-*/*"
            ],
            "Effect": "Allow"
        },
        {
            "Action": "imagebuilder:DeleteImageRecipe",
            "Resource": "arn:aws:imagebuilder:<REGION>:<AWS ACCOUNT ID>:image-recipe/
parallelclusterimage-*/*",
            "Effect": "Allow"
        },
        {
            "Action": "imagebuilder:DeleteDistributionConfiguration",
            "Resource": "arn:aws:imagebuilder:<REGION>:<AWS ACCOUNT ID>:distribution-
configuration/parallelclusterimage-*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "imagebuilder:DeleteImage",
                "imagebuilder:GetImage",
                "imagebuilder:CancelImageCreation"
            ],
            "Resource": "arn:aws:imagebuilder:<REGION>:<AWS ACCOUNT ID>:image/
parallelclusterimage-*/*",
            "Effect": "Allow"
```

```
        },
        {
            "Action": "cloudformation:DeleteStack",
            "Resource": "arn:aws:cloudformation:<REGION>:<AWS ACCOUNT ID>:stack/*/*",
            "Effect": "Allow"
        },
        {
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:<REGION>::image/*",
            "Effect": "Allow"
        },
        {
            "Action": "tag:TagResources",
            "Resource": "*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "lambda:DeleteFunction",
                "lambda:RemovePermission"
            ],
            "Resource": "arn:aws:lambda:<REGION>:<AWS ACCOUNT
 ID>:function:ParallelClusterImage-*",
            "Effect": "Allow"
        },
        {
            "Action": "logs:DeleteLogGroup",
            "Resource": "arn:aws:logs:<REGION>:<AWS ACCOUNT ID>:log-group:/aws/
lambda/ParallelClusterImage-*:*",
            "Effect": "Allow"
        },
        {
            "Action": [
                "SNS:GetTopicAttributes",
                "SNS:DeleteTopic",
                "SNS:GetSubscriptionAttributes",
                "SNS:Unsubscribe"
            ],
            "Resource": "arn:aws:sns:<REGION>:<AWS ACCOUNT ID>:ParallelClusterImage-
*",
            "Effect": "Allow"
        }
    ]
```

```
}
```

其他 IAM 策略

`Build` / `Iam` / `AdditionalIamPolicies`

您可以使用此选项将其他托管 IAM 策略附加到与 I EC2 mage Builder 用来生成自定义 AMI 的亚马逊 EC2 实例关联的角色。

> **Warning**
> 要使用此选项，请确保针对需要附加的 IAM 策略向 AWS ParallelCluster用户授予 `iam:AttachRolePolicy` 和 `iam:DetachRolePolicy` 权限。

权限边界

`Build` / `Iam` / `PermissionsBoundary`

此参数强制 AWS ParallelCluster 将给定的 IAM 策略作为 a 附加`PermissionsBoundary`到在自定义 AMI 构建过程中创建的所有 IAM 角色。

有关使用此类功能所需的策略列表，请参阅 `PermissionsBoundary` 模式。

网络配置

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

VPC 必须有`DNS Resolution = yes`、`DNS Hostnames = yes`和 DHCP 选项以及该区域的正确域名。默认 DHCP 选项集已经指定了所需的 AmazonProvidedDNS。如果指定多个域名服务器，请参阅 Amazon VPC 用户指南 中的 DHCP 选项集。

AWS ParallelCluster 支持以下高级配置：

- 适用于头节点和计算节点的一个子网。
- 两个子网，头节点位于一个公有子网中，计算节点位于私有子网中。子网可以是新的子网，也可以是现有子网。

所有这些配置都可以在有或没有公有 IP 地址的情况下运行。 AWS ParallelCluster 也可以部署为对所有 AWS 请求使用 HTTP 代理。这些配置的组合会产生许多部署方案。例如，您可以配置一个公有子网，允许所有人通过 Internet 进行访问。或者，您可以使用所有流量的 HTTP 代理配置完全私有网络。 AWS Direct Connect

从 AWS ParallelCluster 3.0.0 开始，可以为每个队`SecurityGroups`列`AdditionalSecurityGroups`配置不同的`PlacementGroup`设置。有关更多信息，请参阅 `HeadNode`/`Networking`、`SlurmQueues`/`Networking` 和 `AwsBatchQueues`/`Networking`。

有关一些网络场景的插图，请参阅以下架构图。

主题

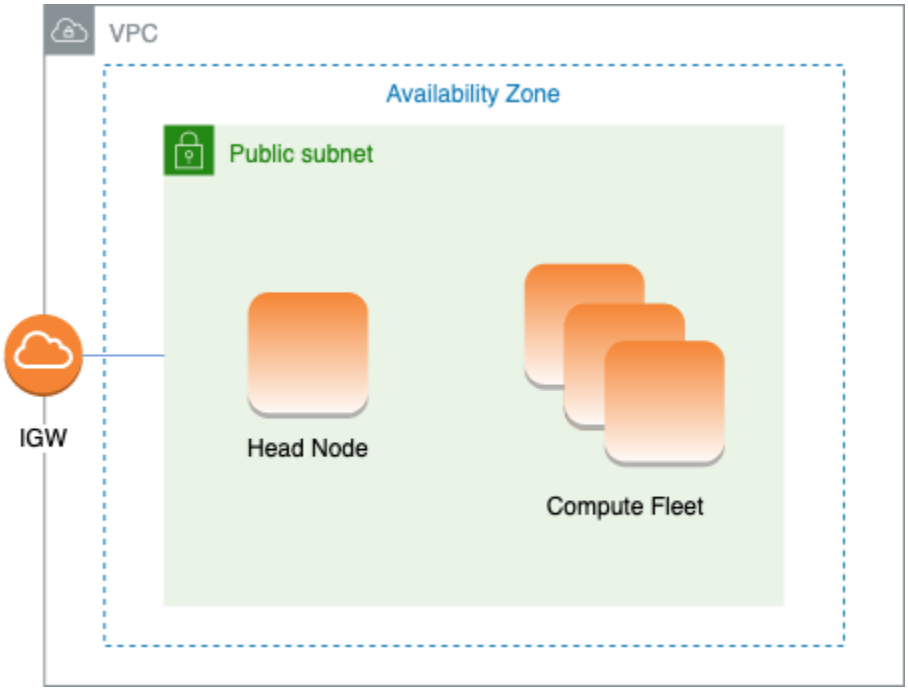
- AWS ParallelCluster 在单个公有子网中
- AWS ParallelCluster 使用两个子网
- AWS ParallelCluster 在使用连接的单个私有子网中 AWS Direct Connect
- AWS ParallelCluster 使用调 AWS Batch 度器
- AWS ParallelCluster 在无法访问互联网的单个子网中

AWS ParallelCluster 在单个公有子网中

在此配置中，必须为集群的所有实例分配一个公有 IP 才能访问互联网。为此，请执行以下操作：

- 通过为//中使用的子网启用 “启用自动分配公共 IPv4 地址” 设置`SubnetId`或在 `HeadNode`/`Networking`/中分配弹性 IP，确保为头节点分配了公有 IP 地址。`HeadNodeNetworkingElasticIp`
- 通过为///中使用的子网打开 “启用自动分配公有 IPv4 地址” 设置`SubnetIds`或在 `Scheduling`/`SlurmQueuesNetworking`/中设置`AssignPublicIp`:true，确保为计算节点分配了公有 IP 地址。`SchedulingSlurmQueuesNetworking`
- 如果您定义了一个p4d实例类型，或者另一个具有多个网络接口或头节点的网络接口卡的实例类型，则必须将 `HeadNode`/`Networking`/设置为`ElasticIp`true以提供公共访问权限。 AWS public IPs 只能分配给使用单个网络接口启动的实例。对于这种情况，我们建议您使用 NAT 网关为集群计算节点提供公有访问权限。有关 IP 地址的更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。
- 您无法定义p4d或hp6id实例类型，也无法定义具有多个网络接口或用于计算节点的网络接口卡的其他实例类型，因为 IPs 只能将 AWS 公共分配给使用单个网络接口启动的实例。有关 IP 地址的更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。

有关更多信息，请参阅 Amazon VPC 用户指南 中的启用互联网访问。



此架构的配置需要以下设置：

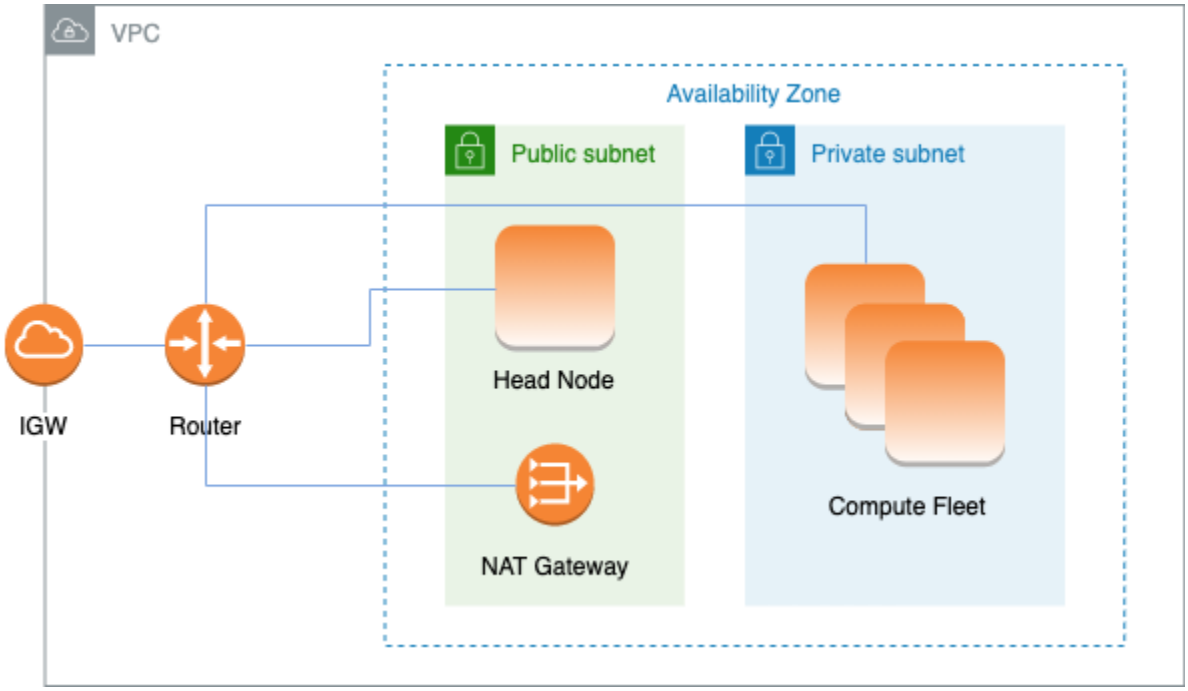
```
# Note that all values are only provided as examples
HeadNode:
  ...
  Networking:
    SubnetId: subnet-12345678 # subnet with internet gateway
    #ElasticIp: true | false | eip-12345678
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - ...
      Networking:
        SubnetIds:
          - subnet-12345678 # subnet with internet gateway
        #AssignPublicIp: true
```

AWS ParallelCluster 使用两个子网

在此配置中，只需要为集群的头节点分配公有 IP。您可以通过为//中使用的子网启用 “启用自动分配公有 IPv4地址” 设置SubnetId或在 HeadNode/Networking/中分配弹性 IP 来实现此目的。HeadNodeNetworkingElasticIp

如果您定义了 p4d 实例类型或其他具有多个网络接口的实例类型或头节点的网络接口卡，则必须将 HeadNode/Networking/设置为ElasticIptrue以提供公共访问权限。 AWS public IPs 只能分配给使用单个网络接口启动的实例。有关 IP 地址的更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。

此配置中用于队列的子网需要 NAT 网关或内部代理，以便为计算实例授予互联网访问权限。

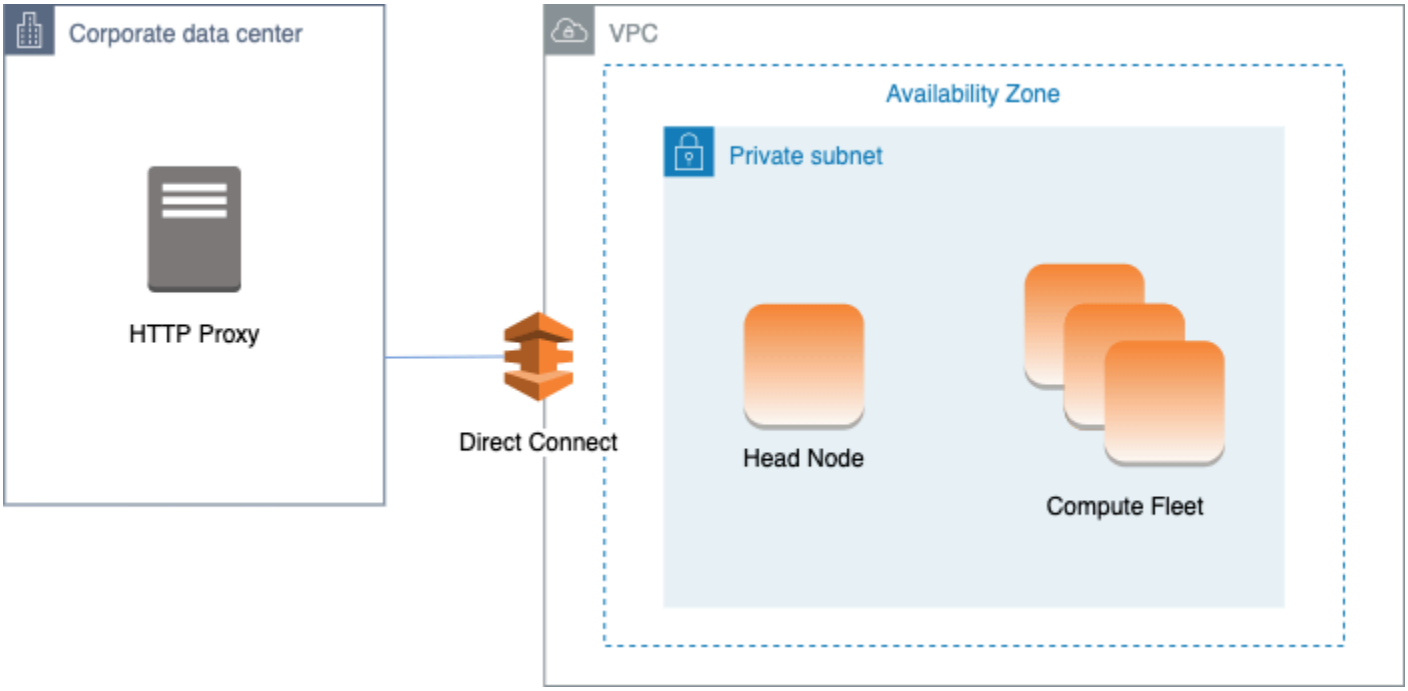


对计算实例使用现有私有子网的配置需要以下设置：

```
# Note that all values are only provided as examples
HeadNode:
  ...
  Networking:
    SubnetId: subnet-12345678 # subnet with internet gateway
    #ElasticIp: true | false | eip-12345678
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - ...
      Networking:
        SubnetIds:
          - subnet-23456789 # subnet with NAT gateway
        #AssignPublicIp: false
```

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

当 Scheduling/SlurmQueues/Networking/AssignPublicIp 设置为 false 时，必须正确设置子网以便对所有流量使用代理。头节点和计算节点都需要 Web 访问权限。



此架构的配置需要以下设置：

```
# Note that all values are only provided as examples
HeadNode:
  ...
  Networking:
    SubnetId: subnet-34567890 # subnet with proxy
    Proxy:
      HttpProxyAddress: http://proxy-address:port
  Ssh:
    KeyName: ec2-key-name
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - ...
      Networking:
        SubnetIds:
          - subnet-34567890 # subnet with proxy
        AssignPublicIp: false
        Proxy:
```

```
          HttpProxyAddress: http://proxy-address:port
```

AWS ParallelCluster 使用调 AWS Batch 度器

当您使用`awsbatch`作为调度器类型时， AWS ParallelCluster 会创建一个 AWS Batch 托管计算环境。 AWS Batch 环境管理 Amazon Elastic Container Service (Amazon ECS) 容器实例。这些实例在 `AwsBatchQueues`/`Networking`/`SubnetIds` 参数中配置的子网中启动。 AWS Batch 为了正常运行，Amazon ECS 容器实例需要访问外部网络才能与 Amazon ECS 服务终端节点通信。这会转换为以下情形：

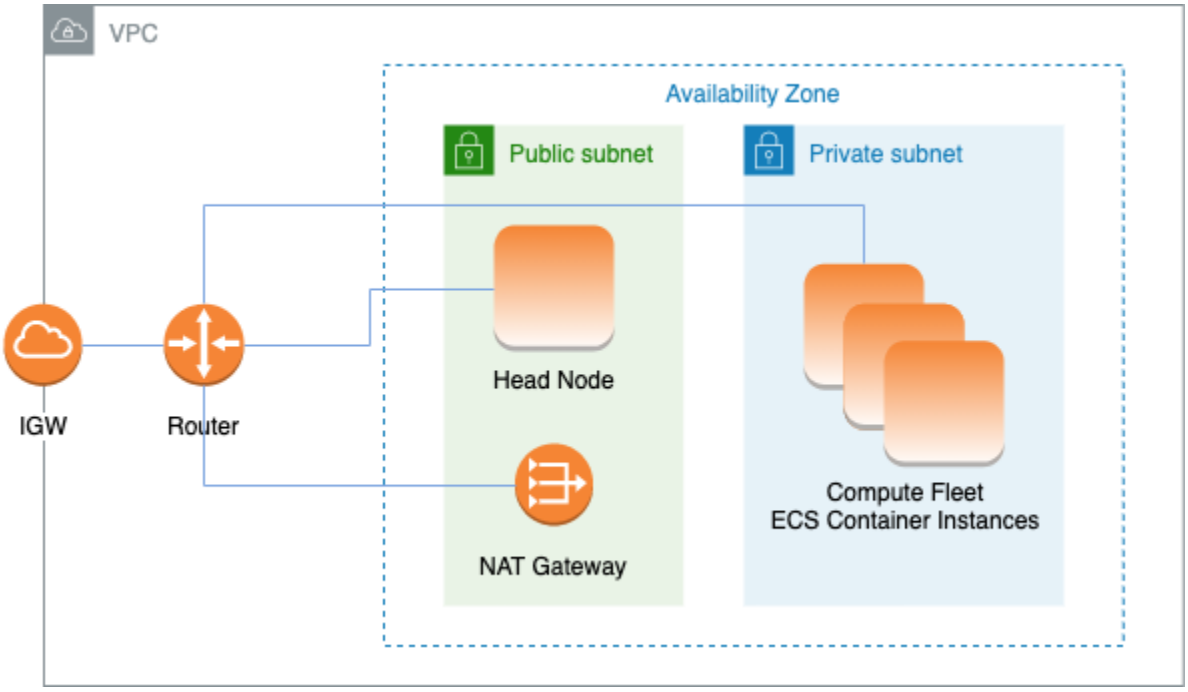
- 为队列指定的子网 ID 使用 NAT 网关访问互联网。我们建议采用此方法。
- 在队列子网中启动的实例具有公有 IP 地址，并可通过互联网网关访问互联网。

此外，如果您对多节点并行作业感兴趣（来自 AWS Batch 文档）：

AWS Batch 多节点并行任务使用 Amazon ECS `awsvpc` 网络模式。这为您的多节点并行任务容器提供了与 Amazon EC2 实例相同的联网属性。每个多节点并行作业容器都可获得自己的弹性网络接口、主要私有 IP 地址以及内部 DNS 主机名。在同一 Amazon VPC 子网中创建网络接口，作为其主机计算资源。适用于计算资源的任何安全组，也适用于该主机计算资源。

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

要使集群能够运行多节点并行作业，必须配置 NAT 网关。



之前的所有配置和注意事项也同样有效。 AWS Batch以下是 AWS Batch 网络配置的示例。

```
# Note that all values are only provided as examples
HeadNode:
  ...
  Networking:
    SubnetId: subnet-12345678 # subnet with internet gateway, NAT gateway or proxy
    #ElasticIp: true | false | eip-12345678
    #Proxy:
      #HttpProxyAddress: http://proxy-address:port
  Ssh:
    KeyName: ec2-key-name
Scheduling:
  Scheduler: awsbatch
  AwsBatchQueues:
    - ...
      Networking:
        SubnetIds:
          - subnet-23456789 # subnet with internet gateway, NAT gateway or proxy
        #AssignPublicIp: true | false
```

在 `Scheduling`/`AwsBatchQueues`/`Networking` 部分中，`SubnetIds` 是列表类型，但目前仅支持一个子网。

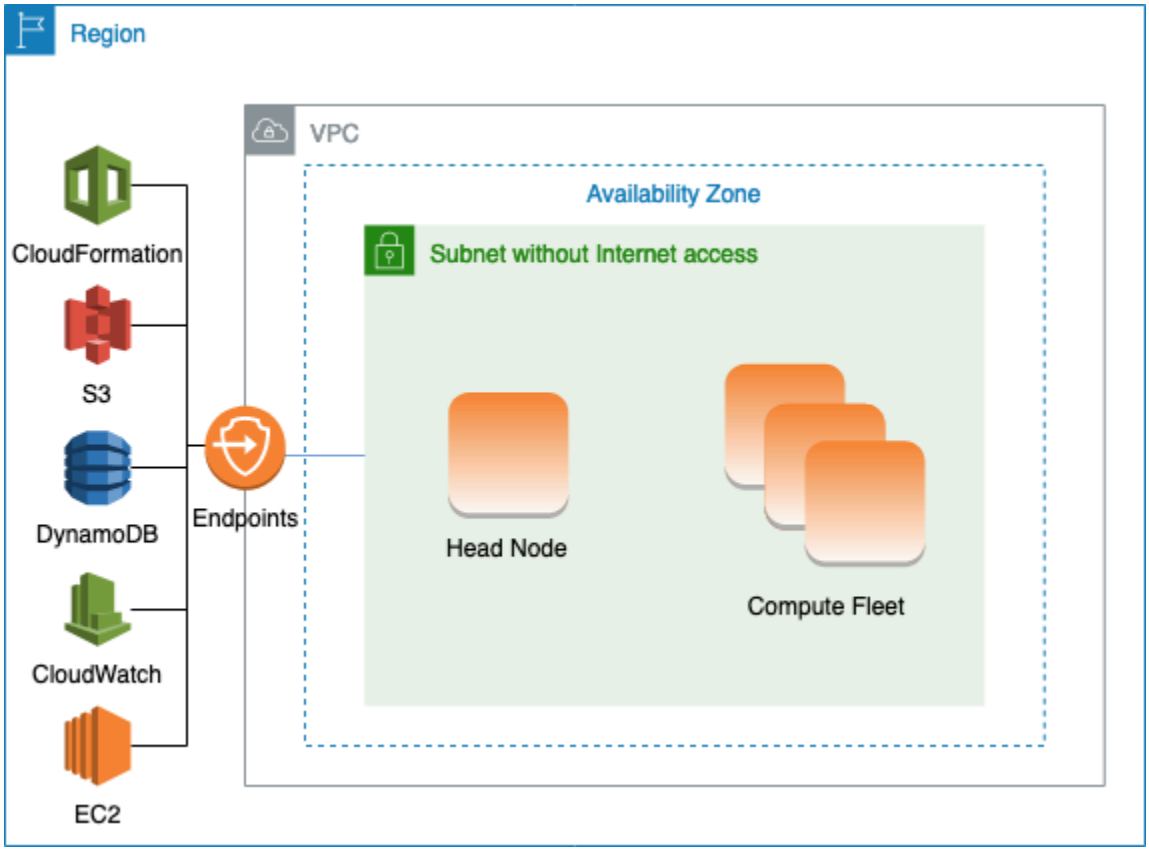
有关更多信息，请参阅以下主题：

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

AWS ParallelCluster 在无法访问互联网的单个子网中

没有互联网访问权限的子网不允许使用入站或出站互联网连接。此 AWS ParallelCluster 配置可以帮助关注安全的客户进一步增强其 AWS ParallelCluster 资源的安全性。 AWS ParallelCluster 节点 AWS ParallelCluster AMIs 的构建基础包括运行无法访问互联网的集群所需的所有软件。通过这种方式，AWS ParallelCluster 可以创建和管理带有无法访问互联网的节点的集群。

在本节中，您将了解如何配置集群。您还将了解运行无互联网访问权限的集群时的限制。



配置 VPC 端点

为确保群集正常运行，群集节点必须能够与多个 AWS 服务进行交互。

创建并配置以下 VPC 终端节点，以便集群节点无需访问互联网即可与 AWS 服务进行交互：

Commercial and AWS GovCloud (US) partitions

服务	服务名称	类型
Amazon CloudWatch	com.amazonaws。*region-id* .logs	接口
AWS CloudFormation	com.amazonaws。*region-id* .cloudfor	接口
Amazon EC2	com.amazonaws。*region-id* .ec2	接口
Amazon S3	com.amazonaws。*region-id* .s3	Gateway
Amazon DynamoDB	com.amazonaws。*region-id* .dynamodb	Gateway
AWS Secrets Manager**	com.amazonaws。*region-id* .secretsManag	接口
AWS 弹性负载平衡***	com.amazonaws。*region-id* .elasticload bal	接口
AWS 自动缩放***	com.amazonaws。*region-id* . 自动缩放	接口

China partition

服务	服务名称	类型
Amazon CloudWatch	com.amazonaws。*region-id* .logs	接口
AWS CloudFormation	cn.com.amazonaws。*region-id* .cloudfor	接口

服务	服务名称	类型
Amazon EC2	cn.com.amazonaws。*region-id* .ec2	接口
Amazon S3	com.amazonaws。*region-id* .s3	Gateway
Amazon DynamoDB	com.amazonaws。*region-id* .dynamodb	Gateway
AWS Secrets Manager**	com.amazonaws。*region-id* .secretsManag	接口
AWS 弹性负载平衡***	com.amazonaws。*region-id* .elasticload bal	接口
AWS 自动缩放***	cn.com.amazonaws。*region-id* . 自动缩放	接口

** 只有在启用了 `DirectoryService` 时才需要此端点，否则它是可选的。

*** 只有在启用时LoginNodes才需要这些端点，否则它们是可选的。

VPC 中的所有实例都必须具有适当的安全组才能与端点通信。您可以通过将安全组添加到 `HeadNode` 配置下面的 `AdditionalSecurityGroups` 和 `SlurmQueues` 配置下面的 `AdditionalSecurityGroups` 来实现这一目的。例如，如果创建了 VPC 端点而未显式指定安全组，则默认安全组将与端点关联。通过将默认安全组添加到 `AdditionalSecurityGroups`，即可启用集群与端点之间的通信。

> **Note**
>
> 当您使用 IAM 策略限制对 VPC 端点的访问时，必须将以下内容添加到 Amazon S3 VPC 端点：
>
> ```
> PolicyDocument:
>  Version: 2012-10-17
>  Statement:
>  - Effect: Allow
> ```

```
      Principal: "*"
      Action:
        - "s3:PutObject"
      Resource:
        - !Sub "arn:${AWS::Partition}:s3:::cloudformation-waitcondition-
${AWS::Region}/*"
```

禁用 Route 53 并使用 Amazon EC2 主机名

创建Slurm集群时， AWS ParallelCluster 会创建用于解析自定义计算节点主机名的私有 Route 53 托管区域，例如`{queue_name}-{st|dy}-{compute_resource}-{N}`。由于 Route 53 不支持 VPC 端点，因此必须禁用此功能。此外， AWS ParallelCluster 必须配置为使用默认 Amazon EC2 主机名，例如`ip-1-2-3-4`。将以下设置应用于您的集群配置：

```
...
Scheduling:
  ...
  SlurmSettings:
    Dns:
      DisableManagedDns: true
      UseEc2Hostnames: true
```

> **Warning**
>
> 对于在 `SlurmSettings`/`Dns`/`DisableManagedDns` 和 `UseEc2Hostnames` 设置为 `true` 的情况下创建的集群，DNS 无法解析 Slurm `NodeName`。可改用 Slurm `NodeHostName`。

> **Note**
>
> 从 3.3.0 AWS ParallelCluster 版本开始，本说明不相关。
> 对于 3.3.0 之前的 AWS ParallelCluster 支持版本：
> 如果设置`UseEc2Hostnames`为`true`，则使用 AWS ParallelCluster `prolog`和`epilog`脚本设置Slurm配置文件：
>
> - 分配了每个作业后，`prolog` 用于向计算节点上的 `/etc/hosts` 中添加节点信息。
> - `epilog` 用于清理 `prolog` 写入的内容。

要添加自定义 epilog 或 prolog 脚本，请分别将其添加到 /opt/slurm/etc/pcluster/prolog.d/ 或 /opt/slurm/etc/pcluster/epilog.d/ 文件夹。

集群配置

了解如何将集群配置为在没有互联网连接的子网中运行。

此架构的配置需要以下设置：

```
# Note that all values are only provided as examples
...
HeadNode:
  ...
  Networking:
    SubnetId: subnet-1234567890abcdef0 # the VPC of the subnet needs to have VPC
 endpoints
    AdditionalSecurityGroups:
      - sg-abcdef01234567890 # optional, the security group that enables the
 communication between the cluster and the VPC endpoints
LoginNodes: # optional, if enabled, requires creation and configuration of VPC
 endpoints for AWS Elastic Load Balancing (ELB) and Auto Scaling services
  Pools:
    - ...
      Networking:
        SubnetIds:
          - subnet-1234567890abcdef0 # the VPC of the subnet needs to have VPC
 endpoints attached
        AdditionalSecurityGroups:
          - sg-1abcdef01234567890 # optional, the security group that enables the
 communication between the cluster and the VPC endpoints
Scheduling:
  Scheduler: Slurm # Cluster in a subnet without internet access is supported only when
 the scheduler is Slurm.
  SlurmSettings:
    Dns:
      DisableManagedDns: true
      UseEc2Hostnames: true
  SlurmQueues:
    - ...
      Networking:
```

```
        SubnetIds:
          - subnet-1234567890abcdef0 # the VPC of the subnet needs to have VPC
endpoints attached
        AdditionalSecurityGroups:
          - sg-1abcdef01234567890 # optional, the security group that enables the
communication between the cluster and the VPC endpoints
```

- SubnetId(s)：无互联网访问权限的子网。

  要启用 AWS ParallelCluster 和 AWS 服务之间的通信，子网的 VPC 必须连接 VPC 终端节点。在创建集群之前，请确认子网中已禁用自动分配公有 IPv4 地址，以确保pcluster命令可以访问集群。
- AdditionalSecurityGroups：启用集群和 VPC 端点之间通信的安全组。

  可选：

  - 如果创建了 VPC 端点而未显式指定安全组，则会关联 VPC 的默认安全组。因此，请将默认安全组提供给 AdditionalSecurityGroups。
  - 如果在创建集群 and/or 时使用自定义安全组，AdditionalSecurityGroups则无需使用 VPC 终端节点，只要自定义安全组支持集群和 VPC 终端节点之间的通信。
- Scheduler：集群调度器。

  slurm 是唯一的有效值。只有 Slurm 调度器支持没有互联网访问权限的子网中的集群。
- SlurmSettings：Slurm 设置。

  请参阅上一节禁用 Route53 并使用 Ama EC2 zon 主机名。

限制

- 通过 SSH 或 Amazon DCV 连接到头节点：连接到集群时，请确保连接客户端可以通过私有 IP 地址访问集群的头节点。如果客户端与头节点不在同一 VPC 中，请在该 VPC 的公有子网中使用代理实例。此要求适用于 SSH 和 DCV 连接。如果子网没有互联网访问权限，则无法访问头节点的公有 IP。pcluster ssh 和 dcv-connect 命令使用公有 IP（如果存在）或私有 IP。在创建集群之前，请确认子网中已禁用自动分配公有 IPv4 地址，以确保pcluster命令可以访问集群。

  以下示例显示了如何连接到在集群头节点中运行的 DCV 会话。您通过 Amazon 代理 EC2 实例进行连接。该实例用作 PC 的 Amazon DCV 服务器，也用作私有子网中头节点的客户端。

  利用公有子网中的代理实例通过 DCV 进行连接：

  1. 在公有子网中创建 Amazon EC2 实例，该子网与集群的子网位于同一 VPC 中。

2. 确保在您的亚马逊 EC2 实例上安装了 Amazon DCV 客户端和服务器。
3. 将 AWS ParallelCluster 用户策略附加到代理 Amazon EC2 实例。有关更多信息，请参阅 AWS ParallelCluster `pcluster`用户策略示例。
4. 在 Amazon 代理 EC2 实例 AWS ParallelCluster 上安装。
5. 通过 DCV 连接到 Amazon 代理 EC2 实例。
6. 在代理实例上使用 `pcluster dcv-connect` 命令即可连接到没有互联网访问权限的子网中的集群。

- 与其他 AWS 服务互动：上面仅列出了严格要求 AWS ParallelCluster 的服务。如果您的集群必须与其他服务交互，请创建相应的 VPC 端点。

由配置的登录节点 AWS ParallelCluster

从版本 3.7.0 开始， AWS ParallelCluster 集群管理员可以配置登录节点，这些节点可用于向用户提供运行作业的访问权限，而不是直接访问集群头节点。 具有适当权限的集群用户可以使用 Active Directory 或其 ssh 凭证登录、提交和管理其作业。这可以改善集群管理，并最大限度地减小耗尽 Slurm 管理集群所需的头节点资源的几率。已登录用户还可以访问登录节点上挂载的集群的所有共享存储。如果需要停止登录节点，则登录的用户将通过他们正在使用的活动 shell 会话提前收到通知。

登录节点被指定为池，池定义了一组具有相同资源配置的登录节点。池中的所有登录节点都配置为网络负载均衡器的一部分，后者会以循环方式在各个登录节点之间分配会话。现有实施使用户能够指定多个登录节点池。

登录节点的安全性

除非为登录节点池指定了`AllowedIPs`设置 `AllowedIps`，否则 AllowedIps 登录节点将继承头节点的设置。通过这种方式，集群管理员可以通过指定源 CIDR 或允许在头节点或登录节点池上进行 SSH 连接的前缀列表来限制集群的安全状态。

在本实现中，启用登录节点时，不会自动限制对头节点的访问。如果需要，集群管理员可以通过使用标准 Linux 命令来更新头节点 ssh 配置，从而限制此访问权限。这也可以通过在头节点上指定自定义安全组来实现，方法是使用 ParallelCluster YAML 文件头节点部分中的`AdditionalSecurityGroups`设置来拒绝来自未经授权的用户的连接。

登录节点联网

登录节点使用为登录节点池配置的网络负载均衡器的单个连接地址进行预置。该地址的连接设置基于登录节点池配置中指定的子网类型。

- 如果子网是私有子网，则该地址将是私有地址，为了向登录节点授予访问权限，集群管理员必须预置堡垒主机。
- 如果子网是公有子网，则该地址将是公有地址

所有连接请求均由网络负载均衡器使用循环路由进行管理。

登录节点存储

在集群上配置的所有共享存储（ ParallelCluster 包括托管存储）都将安装在所有登录节点上。

检索登录节点信息

要检索为访问登录节点而预置的单个连接的地址，集群管理员可以运行 `describe-cluster` 命令。该命令还将提供有关登录节点状态的更多信息。

登录节点是一种支持的新节点类型 ParallelCluster ，在查询特定节点类型的状态时，可以使用`describe-cluster-instances`命令指定登录节点。

登录节点池的单个连接地址的可用性并不能阻止对特定登录节点的直接访问。但为了避免 ssh 客户端发出警告，不建议使用直接连接。ssh 客户端在本地存储每个目标地址的主机标识符。由于每个池的主机标识符是特定的，因此使用不同的 IPs 和/或单个连接地址可能具有相同的主机标识符与不同的目标地址相关联：这可能会导致 ssh 客户端发出警告，因为相同的主机标识符与多个目标相关联。

登录节点的 Imds 属性

登录节点的 IMDS（以及实例配置文件凭证）的访问权限仅限于根用户、集群管理用户（默认）和操作系统特定的`pc-cluster-admin`默认用户（`ec2-user`在 Amazon Linux 2 和 RedHat Ubuntu 18.04 `ubuntu` 上）。

要限制 IMDS 的访问权限，请 AWS ParallelCluster 管理一连串的。`iptables`

> **Note**
>
> 对 `iptables` 或 `ip6tables` 规则进行任何自定义都可能干扰登录节点上用于限制 IMDS 访问权限的机制。另请参阅 `Imds property setting`。

登录节点生命周期

目前没有用于停止和启动池中登录节点的专用命令。为了停止池中的登录节点，集群管理员必须更新集群配置，将登录节点数指定为零 (`Count: 0`)，然后运行 `pcluster.update-cluster-v3` 命令。

> **Note**
>
> 已登录用户会收到有关特定实例终止以及相关宽限期的通知。在宽限期内，除了来自集群默认用户的连接外，不允许任何新连接。集群管理员可以从头节点或登录节点上通过编辑 `/opt/parallelcluster/shared_login_nodes/loginmgtd_config.json` 文件来自定义显示的消息。当您使用AWS Systems Manager Session Manager会话管理器进行连接时，此终止消息不可见。

为了启动登录节点池，集群管理员必须在集群配置中还原先前的 `Count` 值，然后运行 `update-cluster` 命令。

运行登录节点池所需的权限

要管理登录节点池，集群管理员必须具有以下额外权限：

```
        - Action:
          - iam:CreateServiceLinkedRole
          - autoscaling:DeleteAutoScalingGroup
          - autoscaling:DeleteLifecycleHook
          - autoscaling:Describe*
          - autoscaling:PutLifecycleHook
          - autoscaling:UpdateAutoScalingGroup
          - elasticloadbalancing:CreateListener
          - elasticloadbalancing:CreateTargetGroup
          - elasticloadbalancing:DeleteListener
          - elasticloadbalancing:DeleteLoadBalancer
          - elasticloadbalancing:DeleteTargetGroup
          - elasticloadbalancing:Describe*
          - elasticloadbalancing:ModifyLoadBalancerAttributes
        Resource: '*'
        Condition:
          ForAllValues:StringEquals:
            aws:TagKeys: [ "parallelcluster:cluster-name" ]
```

```
            - Action:
              - autoscaling:CreateAutoScalingGroup
              - elasticloadbalancing:AddTags
              - elasticloadbalancing:CreateLoadBalancer
            Resource: '*'
            Effect: Allow
```

自定义引导操作

如果定义了 HeadNode/CustomActions/OnNodeStart配置设置，则会在节点启动后立即 AWS ParallelCluster 运行任意代码。如果定义了 HeadNode/CustomActions/OnNodeConfigured配置设置，则在正确完成节点配置后 AWS ParallelCluster 运行代码。

从 3.4.0 AWS ParallelCluster 版开始，如果您定义了 HeadNode/CustomActions/OnNodeUpdated配置设置，则可以在头节点更新后运行代码。

在大多数情况下，此代码存储在 Amazon Simple Storage Service (Amazon S3) 中并通过 HTTPS 连接进行访问。此代码将以 `root` 用户身份运行，可以采用集群操作系统支持的任何脚本语言。代码通常采用 Bash 或 Python 语言。

> **Note**
>
> 从 3.7.0 AWS ParallelCluster 版开始，集群 Imds/ImdsSupport默认设置为。`v2.0`
> 当您创建要升级到版本 3.7.0 及更高版本的新集群时，要么更新您的自定义引导操作脚本使其与之兼容，要 IMDSv2 么在集群配置文件`v1.0`中将 Imds/ImdsSupport设置为。

> **Warning**
>
> 您有责任按照责任共担模型中所述配置自定义脚本和参数。验证您的自定义引导脚本和参数是否来自您信任的可以完全访问集群节点的来源。

> **Warning**
>
> AWS ParallelCluster 不支持使用通过`/etc/parallelcluster/cfnconfig`文件提供的内部变量。此文件可能会在未来版本中删除。

在开始任何节点部署引导操作（例如配置 NAT、Amazon Elastic Block Store (Amazon EBS) 或调度器）之前，将会调用 OnNodeStart 操作。OnNodeStart 引导操作可能包括修改存储、添加额外的用户和添加程序包。

> **Note**
>
> 如果您为集群配置DirectoryService和 HeadNodeCustomActions//OnNodeStart脚本，则会在集群运行脚本sssd之前 AWS ParallelCluster 配置DirectoryService并重新启动。OnNodeStart

节点引导过程完成后将会调用 OnNodeConfigured 操作。OnNodeConfigured 操作是实例被视为完全配置并已完成之前执行的最后操作。某些 OnNodeConfigured 操作包括更改调度器设置、修改存储和修改程序包。您可以通过在配置期间指定参数来将参数传递给脚本。

在头节点更新完成并且调度器和共享存储与最新的集群配置更改保持一致之后，将会调用 OnNodeUpdated 操作。

当 OnNodeStart 或 OnNodeConfigured 自定义操作成功时，将使用退出代码零 (0) 来表示成功。任何其他退出代码都表示实例引导失败。

当 OnNodeUpdated 自定义操作成功时，将使用退出代码零 (0) 来表示成功。任何其他退出代码都表示更新失败。

> **Note**
>
> 如果配置 OnNodeUpdated，则在更新失败时，必须将 OnNodeUpdated 操作手动恢复到先前的状态。
> 如果 OnNodeUpdated 自定义操作失败，则更新将回滚到之前的状态。但 OnNodeUpdated 操作仅在更新时运行，而不是在堆栈回滚时运行。

在/和 i HeadNode/SchedulingSlurmQueues/CustomActions配置部分中，您可以为头节点CustomActions和每个队列指定不同的脚本。 OnNodeUpdated只能在该HeadNode部分中进行配置。

> **Note**
>
> 在 3.0 AWS ParallelCluster 版本之前，无法为头节点和计算节点指定不同的脚本。请参阅 从 AWS ParallelCluster 2.x 升级到 3.x。

主题

- 用于定义操作和参数的配置设置
- 参数
- 包含自定义引导操作的示例集群
- 如何更新自定义引导脚本的示例 IMDSv2
- 如何更新配置的示例 IMDSv1

用于定义操作和参数的配置设置

以下配置设置用于定义 HeadNode/CustomActions/OnNodeStart、OnNodeConfigured、OnNodeUpdated 以及 Scheduling/CustomActions/OnNodeStart、OnNodeConfigured 操作和参数。

```
HeadNode:
  [...]
  CustomActions:
    OnNodeStart:
      # Script URL. This is run before any of the bootstrap scripts are run
      Script: s3://amzn-s3-demo-bucket/on-node-start.sh
      Args:
        - arg1
    OnNodeConfigured:
      # Script URL. This is run after all the bootstrap scripts are run
      Script: s3://amzn-s3-demo-bucket/on-node-configured.sh
      Args:
        - arg1
    OnNodeUpdated:
      # Script URL. This is run after the head node update is completed.
      Script: s3://amzn-s3-demo-bucket/on-node-updated.sh
      Args:
        - arg1
  # Bucket permissions
  Iam:
```

```
    S3Access:
      - BucketName: bucket_name
        EnableWriteAccess: false
Scheduling:
  Scheduler: slurm
   [...]
  SlurmQueues:
    - Name: queue1
      [...]
      CustomActions:
        OnNodeStart:
          Script: s3://amzn-s3-demo-bucket/on-node-start.sh
          Args:
            - arg1
        OnNodeConfigured:
          Script: s3://amzn-s3-demo-bucket/on-node-configured.sh
          Args:
            - arg1
      Iam:
        S3Access:
          - BucketName: bucket_name
            EnableWriteAccess: false
```

使用Sequence设置（在 3.6.0 AWS ParallelCluster 版本中添加）：

```
HeadNode:
  [...]
  CustomActions:
    OnNodeStart:
      # Script URLs. The scripts are run in the same order as listed in the
 configuration, before any of the bootstrap scripts are run.
      Sequence:
        - Script: s3://amzn-s3-demo-bucket/on-node-start1.sh
          Args:
            - arg1
        - Script: s3://amzn-s3-demo-bucket/on-node-start2.sh
          Args:
            - arg1
        [...]
    OnNodeConfigured:
      # Script URLs. The scripts are run in the same order as listed in the
 configuration, after all the bootstrap scripts are run.
      Sequence:
```

```
        - Script: s3://amzn-s3-demo-bucket/on-node-configured1.sh
          Args:
            - arg1
        - Script: s3://amzn-s3-demo-bucket/on-node-configured2.sh
          Args:
            - arg1
        [...]
    OnNodeUpdated:
      # Script URLs. The scripts are run in the same order as listed in the
 configuration, after the head node update is completed.
      Sequence:
        - Script: s3://amzn-s3-demo-bucket/on-node-updated1.sh
          Args:
            - arg1
        - Script: s3://amzn-s3-demo-bucket/on-node-updated2.sh
          Args:
            - arg1
        [...]
  # Bucket permissions
  Iam:
    S3Access:
      - BucketName: bucket_name
        EnableWriteAccess: false
Scheduling:
  Scheduler: slurm
   [...]
  SlurmQueues:
    - Name: queue1
      [...]
      CustomActions:
        OnNodeStart:
          # Script URLs. The scripts are run in the same order as listed in the
 configuration, before any of the bootstrap scripts are run
          Sequence:
            - Script: s3://amzn-s3-demo-bucket/on-node-start1.sh
              Args:
                - arg1
            - Script: s3://amzn-s3-demo-bucket/on-node-start2.sh
              Args:
                - arg1
            [...]
        OnNodeConfigured:
          # Script URLs. The scripts are run in the same order as listed in the
 configuration, after all the bootstrap scripts are run
```

```
          Sequence:
            - Script: s3://amzn-s3-demo-bucket/on-node-configured1.sh
              Args:
                - arg1
            - Script: s3://amzn-s3-demo-bucket/on-node-configured2.sh
              Args:
                - arg1
            [...]
      Iam:
        S3Access:
          - BucketName: bucket_name
            EnableWriteAccess: false
```

该Sequence设置是从 3.6.0 AWS ParallelCluster 版本开始添加的。指定 Sequence 后，您可以列出自定义操作的多个脚本。在配置自定义操作时， AWS ParallelCluster 继续支持使用单个脚本而不包括 Sequence。

AWS ParallelCluster 不支持同时包含单个脚本和Sequence同一个自定义操作。例如，如果您指定以下配置，则会 AWS ParallelCluster 失败。

```
[...]
  CustomActions:
    OnNodeStart:
      # Script URL. This is run before any of the bootstrap scripts are run
      Script: s3://amzn-s3-demo-bucket/on-node-start.sh
          Args:
            - arg1
      # Script URLs. The scripts are run in the same order as listed in the
 configuration, before any of the bootstrap scripts are run.
      Sequence:
        - Script: s3://amzn-s3-demo-bucket/on-node-start1.sh
          Args:
            - arg1
        - Script: s3://amzn-s3-demo-bucket/on-node-start2.sh
          Args:
            - arg1
[...]
```

参数

在 AWS ParallelCluster 2.x 中，$1参数是保留的，用于存储自定义脚本的 URL。如果要重复使用为 AWS ParallelCluster 2.x和 AWS ParallelCluster 3.x创建的自定义引导脚本，则需要通过考虑参数的转移来调整它们。请参阅 从 AWS ParallelCluster 2.x 升级到 3.x。

包含自定义引导操作的示例集群

以下步骤创建一个要在节点配置完成后执行的简单脚本，该脚本将在集群的节点中安装 `R`, `curl` 和 `wget` 软件包。

1. 创建脚本。

```
#!/bin/bash
  echo "The script has $# arguments"
  for arg in "$@"
  do
      echo "arg: ${arg}"
  done
  yum -y install "${@:1}"
```

2. 使用正确的权限将脚本上传到 Amazon S3。如果公共读取权限不适合您，请使用 `HeadNode`/`Iam`/`S3Access` 和 `Scheduling`/`SlurmQueues` 配置部分。有关更多信息，请参阅 使用 Amazon S3。

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

> **Important**
>
> 如果脚本是在 Windows 上编辑的，则必须将行结尾从 CRLF 更改为 LF，然后才能将脚本上传到 Amazon S3。

3. 更新 AWS ParallelCluster 配置以包含新`OnNodeConfigured`操作。

```
CustomActions:
  OnNodeConfigured:
    Script: https://<amzn-s3-demo-bucket>.s3.<region>.amazonaws.com/myscript.sh
    Args:
      - "R"
```

```
      - "curl"
      - "wget"
```

如果存储桶没有公共读取权限，则使用 s3 作为 URL 协议。

```
CustomActions:
  OnNodeConfigured:
    Script: s3://amzn-s3-demo-bucket/myscript.sh
    Args:
      - "R"
      - "curl"
      - "wget"
```

4. 启动集群。

```
$ pcluster create-cluster --cluster-name mycluster \
  --region <region> --cluster-configuration config-file.yaml
```

5. 验证输出。

- 如果您在HeadNode配置中添加了自定义操作，请登录到头节点并运行以下命令/var/log/cfn-init.log来检查位于的cfn-init.log文件：

```
$ less /var/log/cfn-init.log
  2021-09-03 10:43:54,588 [DEBUG] Command run
  postinstall output: The script has 3 arguments
  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
```

- 如果您将自定义操作添加到了 SlurmQueues 设置中，请检查位于计算节点中 /var/log/cloud-init.log 处的 cloud-init.log。用于 CloudWatch 查看这些日志。

您可以在 Amazon CloudWatch 控制台中查看这两个日志。有关更多信息，请参阅 与 Amazon CloudWatch 日志集成。

如何更新自定义引导脚本的示例 IMDSv2

在以下示例中，我们更新了与一起 IMDSv1 使用的自定义 bootstrap 操作脚本。 IMDSv2该 IMDSv1 脚本检索亚马逊 EC2 实例 AMI ID 元数据。

```
#!/bin/bash
AMI_ID=$(curl http://169.254.169.254/latest/meta-data/ami-id)
echo $AMI_ID >> /home/ami_id.txt
```

下图显示了为兼容而修改的自定义引导操作脚本。 IMDSv2

```
#!/bin/bash
AMI_ID=$(TOKEN=`curl -X PUT "http://169.254.169.254/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600"` \
         && curl -H "X-aws-ec2-metadata-token: $TOKEN" -v http://169.254.169.254/
latest/meta-data/ami-id)
echo $AMI_ID >> /home/ami_id.txt
```

有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的检索实例元数据。

如何更新配置的示例 IMDSv1

以下是使用 3.7.0 及更早 AWS ParallelCluster 版本 IMDSv1 时支持的集群配置示例。

```
Region: us-east-1
Imds:
  ImdsSupport: v1.0
Image:
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-abcdef01234567890
  Ssh:
    KeyName: key-name
  CustomActions:
    OnNodeConfigured:
      Script: Script-path
Scheduling:
  Scheduler: slurm
  SlurmQueues:
```

```
  - Name: queue1
    CustomActions:
      OnNodeConfigured:
        Script: Script-path
    ComputeResources:
    - Name: t2micro
      Instances:
      - InstanceType: t2.micro
      MinCount: 11
    Networking:
      SubnetIds:
      - subnet-abcdef01234567890
```

使用 Amazon S3

您可以通过配置 AWS ParallelCluster中的// S3Access和 HeadNode//IamSchedulingSlurmQueues- NameIam/S3Access参数 AWS ParallelCluster 配置对 Amazon S3 的访问权限。

示例

以下示例配置对 *firstbucket/read_only/* 中所有对象的只读访问权限，对 *secondbucket/read_and_write/* 中所有对象的读/写访问权限。

```
...
HeadNode:
  ...
  Iam:
    S3Access:
      - BucketName: firstbucket
        KeyName: read_only/*
        EnableWriteAccess: false
      - BucketName: secondbucket
        KeyName: read_and_write/*
        EnableWriteAccess: true
...
```

下一个示例配置对账户任何存储桶 (*) 的 *read_only/* 文件夹中所有对象的只读访问权限。

```
...
```

```
HeadNode:
  ...
  Iam:
    S3Access:
      - BucketName: *
        KeyName: read_only/*
        EnableWriteAccess: false
...
```

最后一个示例配置对账户中所有存储桶和对象的 read_only 访问权限。

```
...
HeadNode:
  ...
  Iam:
    S3Access:
      - BucketName: *
...
```

使用竞价型实例

AWS ParallelCluster 如果您在集群配置文件SPOT中将 `SlurmQueues`/`CapacityType`或 `AwsBatchQueues`/设置为 `CapacityType`，则使用竞价型实例。竞价型实例比按需型实例更具成本效益，但它们可能会中断。利用竞价型实例中断通知可能会有所帮助，这些通知会在亚马逊 EC2 必须停止或终止您的竞价型实例之前提供两分钟的警告。有关更多信息，请参阅 Amazon EC2 用户指南中的竞价型实例中断。要了解 `AwsBatchQueues` 如何使用竞价型实例，请参阅 AWS Batch User Guide 中的 Compute Resources。

AWS ParallelCluster 配置的调度器将任务分配给带有竞价型实例的队列中的计算资源，就像将任务分配给带有按需实例的队列中的计算资源一样。

使用竞价型实例时，您的账户中必须存在 AWSServiceRoleForEC2竞价服务相关角色。要使用在您的账户中创建此角色 AWS CLI，请运行以下命令：

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

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

以下各部分介绍了使用 `SlurmQueues` 时竞价型实例可能被中断的三种情形。

情形 1：没有运行作业的竞价型实例被中断

发生这种中断时，如果调度器队列有需要额外实例的待处理任务，或者活动实例的数量小于 `SlurmQueues`/`ComputeResources`/`MinCount`，则 AWS ParallelCluster 尝试替换实例。如果 AWS ParallelCluster 无法配置新实例，则会定期重复请求新实例。

情形 2：运行单节点作业的竞价型实例被中断

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

情形 3：运行多节点作业的竞价型实例被中断

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

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

支持的调度器 AWS ParallelCluster

AWS ParallelCluster 支持 Slurm 和 AWS Batch 调度程序，它们使用设置进行`Scheduler`设置。以下主题将介绍每种调度器及其使用方法。

主题

- Slurm Workload Manager (slurm)
- 将 AWS Batch (awsbatch) 调度器与 AWS ParallelCluster

Slurm Workload Manager (**`slurm`**)

集群容量大小和更新

集群的容量由集群可以扩展的计算节点数量来定义。计算节点由 AWS ParallelCluster 配置中的计算资源中定义的 Amazon EC2 实例提供支持(`Scheduling/SlurmQueues/ComputeResources`)，并按照 1:1 映射到(`Scheduling/SlurmQueues`)的队列进行组织 Slurm 分区。

在计算资源中，可以配置集群中必须始终保持运行的最小计算节点（实例）数（MinCount），以及计算资源可以扩展到的最大实例数（MaxCount3）。

在创建集群时或群集更新时，为集群中定义的每个计算资源 (Scheduling/SlurmQueues/ComputeResources) AWS ParallelCluster 启动中配置MinCount的任意数量的 Amazon EC2 实例。为覆盖集群中计算资源的最小节点数量而启动的实例称为静态节点。静态节点一旦启动，就会在集群中持续存在，除非发生特定事件或情况，否则系统不会终止它们。例如，此类事件包括失败 Slurm 或者 Amazon EC2 健康检查和变更 Slurm 节点状态变为 “耗尽” 或 “关闭”。

为应对集群MaxCount 负载增加而*1*按需启动的 Amazon EC2 实例被称为动态节点，范围为*‘MaxCount - MinCount’*（减去 MinCount））。 它们的性质是短暂的，启动它们是为了处理待处理的任务，如果它们Scheduling/SlurmSettings/ScaledownIdletime在集群配置中定义的一段时间内保持闲置状态（默认值：10 分钟），它们就会被终止。

静态节点和动态节点符合以下命名方案：

- 静态节点 <Queue/Name>-st-<ComputeResource/Name>-<num>，其中 <num> = 1..ComputeResource/MinCount
- 动态节点 <Queue/Name>-dy-<ComputeResource/Name>-<num>，其中 <num> = 1..(ComputeResource/MaxCount - ComputeResource/MinCount)

例如，给定以下 AWS ParallelCluster 配置：

```
Scheduling:
    Scheduler: Slurm
    SlurmQueues:
        - Name: queue1
            ComputeResources:
                - Name: c5xlarge
                    Instances:
                        - InstanceType: c5.xlarge
                        MinCount: 100
                        MaxCount: 150
```

将在中定义以下节点 Slurm

```
$ sinfo
```

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

当计算资源的 `MinCount == MaxCount` 时，所有相应的计算节点都将是静态的，所有实例都将在集群创建/更新时启动并保持运行。例如：

```
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ComputeResources:
        - Name: c5xlarge
          Instances:
            - InstanceType: c5.xlarge
          MinCount: 100
          MaxCount: 100
```

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

集群容量更新

集群容量的更新包括添加或删除队列、计算资源或更改计算资源的 `MinCount/MaxCount`。从 AWS ParallelCluster 版本 3.9.0 开始，要缩小队列的大小，需要在集群更新之前停止计算队列或将其QueueUpdateStrategy设置为 TERMINATION for。在以下情况下，无需停止计算队列或将其设置QueueUpdateStrategy为 “终止”：

- 将新队列添加到 Scheduling/`SlurmQueues`
- 向队列中添加新的计算资源 `Scheduling/SlurmQueues/ComputeResources`
- 增加计算资源的 `MaxCount`
- 计算资源的增加 MaxCount 和相同计算资源的增加量至少相同 MinCount

注意事项和限制

本节旨在概述在调整集群容量大小时应考虑的任何重要因素、约束或限制。

- 从Scheduling/SlurmQueues所有具有静态和动态名称<Queue/Name>-*的计算节点中移除队列时，将从 Slurm 配置和相应的 Amazon EC2 实例将被终止。
- Scheduling/SlurmQueues/ComputeResources从队列中移除计算资源时，所有名<Queue/Name>-*-<ComputeResource/Name>-*为静态和动态的计算节点都将从队列中移除 Slurm 配置和相应的 Amazon EC2 实例将被终止。

在更改计算资源的 MinCount 参数时，我们可以区分两种不同的情况，一种是 MaxCount 与 MinCount 相等（仅静态容量），另一种是 MaxCount 大于 MinCount（静态和动态混合容量）。

只有静态节点的容量变化

- 如果在增加MinCount（和MaxCount）时MinCount == MaxCount，将通过将静态节点的数量扩展到新的值来配置集群，MinCount<Queue/Name>-st-<ComputeResource/Name>-<new_MinCount>并且系统将继续尝试启动 Amazon EC2 实例以满足新的所需静态容量。
- 如果在减少MinCount（和MaxCount）数量 N 时MinCount == MaxCount，将通过移除最后 N 个静态节点来配置集群，<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount - N>...<old_MinCount>]并且系统将终止相应的 Amazon EC2 实例。
  - 初始状态 MinCount = MaxCount = 100
  -
    ```
    $ sinfo
    PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
    queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
    ```
  - 更新 MinCount 和 MaxCount: MinCount = MaxCount = 70 上的 -30
  -
    ```
    $ sinfo
    PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
    queue1*      up   infinite     70   idle queue1-st-c5xlarge-[1-70]
    ```

混合节点的容量变化

如果在增加MinCount数量 N（假设保持不变）时MinCount < MaxCount，MaxCount将通过将静态节点的数量扩展到新的值 MinCount (old_MinCount + N): 来配置集群，<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount + N>并且系统将继续尝试启动 Amazon EC2 实例以满足新的所需静态容量。此外，为了满足计算资源的MaxCount容量，通过删除最后 N 个动态节点来更新集群配置：<Queue/Name>-dy-<ComputeResource/Name>-[<MaxCount - old_MinCount - N>...<MaxCount - old_MinCount>]系统将终止相应的 Amazon EC2 实例。

- 初始状态：MinCount = 100; MaxCount = 150
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```
- 将 +30 更新为 MinCount : MinCount = 130 (MaxCount = 150)
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     20  idle~ queue1-dy-c5xlarge-[1-20]
queue1*      up   infinite    130   idle queue1-st-c5xlarge-[1-130]
```

如果在增加MinCount且MaxCount数量相同 N 时MinCount < MaxCount，将通过将静态节点数扩展到新值 MinCount (old_MinCount + N): 来配置集群，<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount + N>并且系统将继续尝试启动 Amazon EC2 实例以满足新的所需静态容量。此外，动态节点的数量不会因新的

MaxCount 值。

- 初始状态：MinCount = 100; MaxCount = 150
-
```
$ sinfo
```

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 将 +30 更新为 MinCount : MinCount = 130 (MaxCount = 180)
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     20  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    130   idle queue1-st-c5xlarge-[1-130]
```

如果MinCount < MaxCount在减少MinCount数量 N（假设保持不变）时，MaxCount将通过移除最后 N 个静态节点的静态节点来配置集群，系统将终止相应的 Amazon EC2 实例。<Queue/Name>-st-<ComputeResource/Name>-[<old_MinCount - N>...<old_MinCount>此外，为了满足计算资源的MaxCount容量，通过扩展动态节点的数量来更新集群配置以填补空白。MaxCount - new_MinCount: <Queue/Name>-dy-<ComputeResource/Name>-[1..<MazCount - new_MinCount>]在这种情况下，由于这些是动态节点，因此除非调度程序在新节点上有待处理的任务，否则不会启动新的 Amazon EC2 实例。

- 初始状态：MinCount = 100; MaxCount = 150
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 更新 MinCount : MinCount = 70 (MaxCount = 120) 上的 -30
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     80  idle~ queue1-dy-c5xlarge-[1-80]
queue1*      up   infinite     70   idle queue1-st-c5xlarge-[1-70]
```

如果在减少MinCount且MaxCount数量相同的 N 时MinCount < MaxCount，将通过移除最后 N 个静态节点<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount - N>...<oldMinCount>]来配置集群，系统将终止相应的 Amazon EC2 实例。

此外，动态节点的数量不会因新的 MaxCount 值而发生变化。

- 初始状态：MinCount = 100; MaxCount = 150
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```
- 更新 MinCount : MinCount = 70 (MaxCount = 120) 上的 -30
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     80  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite     70   idle queue1-st-c5xlarge-[1-70]
```

如果在减少MaxCount数量 N（假设保持不变）时MinCount < MaxCount，MinCount将通过移除最后 N 个动态节点来配置集群，<Queue/Name>-dy-<ComputeResource/Name>-<old_MaxCount - N...<oldMaxCount>]并且系统将终止相应的 Amazon EC2 实例（如果它们正在运行）。预计不会对静态节点产生任何影响。

- 初始状态：MinCount = 100; MaxCount = 150
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 更新 `MaxCount : MinCount = 100 (MaxCount = 120)` 上的 -30
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     20  idle~ queue1-dy-c5xlarge-[1-20]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

对作业的影响

在移除节点和 Amazon EC2 实例终止的所有情况下，除非没有其他节点满足任务要求，否则在已移除的节点上运行的 sbatch 作业都将重新排队。在最后一种情况下，任务失败，状态为 NODE_FAIL，并从队列中消失，必须手动重新提交。

如果您计划执行集群大小调整更新，可以防止作业在计划更新期间将被移除的节点上运行。这可以通过将要移除的节点设置为维护状态来实现。请注意，将节点设置为维护状态不会影响最终已在该节点上运行的作业。

假设在计划的集群大小调整更新中，您将移除节点 `qeueu-st-computeresource-[9-10]`。你可以创建一个 Slurm 使用以下命令进行预订

```
sudo -i scontrol create reservation ReservationName=maint_for_update user=root
 starttime=now duration=infinite flags=maint,ignore_jobs nodes=qeueu-st-
computeresource-[9-10]
```

这将创建一个 Slurm 在节点`maint_for_update`上命名的预留`qeueu-st-computeresource-[9-10]`。从创建预留开始，`qeueu-st-computeresource-[9-10]` 节点上就不能再运行任何作业。请注意，预留不会阻止作业最终在 `qeueu-st-computeresource-[9-10]` 节点上分配。

集群大小更新后，如果 Slurm 仅在调整大小更新期间移除的节点上设置了预留，维护预留将自动删除。相反，如果你创建了一个 Slurm 在集群调整大小更新后仍然存在的节点上进行预留，我们可能需要在执行调整大小更新后使用以下命令删除节点上的维护预留

```
sudo -i scontrol delete ReservationName=maint_for_update
```

如需了解更多详情，请访问 Slurm 预订，请在此处查看 schedMD 官方文档。

容量变更时的集群更新过程

调度器配置更改后，将在集群更新过程中执行以下步骤：

- 停下来 AWS ParallelCluster `clustermgtd (supervisorctl stop clustermgtd)`
- 生成已更新 Slurm 配置中的分区 AWS ParallelCluster 配置
- 重启 `slurmctld`（通过 Chef 服务配方完成）
- 检查 `slurmctld` 状态 `(systemctl is-active --quiet slurmctld.service)`
- Reload Slurm 配置 `(scontrol reconfigure)`
- 启动 `clustermgtd (supervisorctl start clustermgtd)`

有关信息 Slurm，请参阅 https://slurm.schedmd.com。有关下载，请参阅 https://github.com/SchedMD/slur m/tags。有关源代码，请参阅 https://github.com/SchedMD/slurm。

支持的集群版本和 SLURM 版本

下表列出了 AWS ParallelCluster 和 Slurm AWS 支持的版本。

AWS ParallelCluster 版本	支持 Slurm version
3.13.0	24.05.07
3.12.0	23.11.10
3.11.0	23.11.10
3.9.2、3.9.3、3.10.0	23.11.7
3.9.0、3.9.1	23.11.4
3.8.0	23.02.7
3.7.2	23.02.6
3.7.1	23.02.5
3.7.0	23.02.4
3.6.0、3.6.1	23.02.2

AWS ParallelCluster 版本	支持 Slurm version
3.5.0、3.5.1	22.05.8
3.4.0、3.4.1	22.05.7
3.3.0、3.3.1	22.05.5
3.1.4、3.1.5、3.2.0、3.2.1	21.08.8-2
3.1.2、3.1.3	21.08.6
3.1.1	21.08.5
3.0.0	20.11.8

主题

- 多个队列的配置
- Slurm 多队列模式指南
- Slurm 集群保护模式
- Slurm 集群快速容量不足故障转移
- Slurm 基于内存的调度
- Slurm 的多实例类型分配
- 动态节点的集群扩展
- Slurm 用会计 AWS ParallelCluster
- Slurm 配置自定义
- Slurmprolog 和 epilog
- 集群容量大小和更新

多个队列的配置

在 AWS ParallelCluster 版本 3 中，您可以通过将设置为`slurm`并在配置文件`SlurmQueues`中为指定多个队列来配置多个队列。`Scheduler`在此模式下，不同的实例类型共存于配置文件的`ComputeResources` 部分中指定的计算节点中。具有不同实例类型的 `ComputeResources` 会根据`SlurmQueues` 的需要纵向扩展或缩减。

当工作负载共享相同的底层基础架构和资源（例如共享存储、网络或登录节点）时，单个集群中的多个队列通常优先于多个集群。如果工作负载具有相似的计算、存储和网络需求，则在单个集群中使用多个队列会更高效，因为它允许资源共享并避免不必要的重复。这种方法简化了管理，减少了开销，同时还能实现高效的作业调度和资源分配。另一方面，当工作负载之间存在很强的安全、数据或操作隔离要求时，应使用多个集群。例如，如果您需要使用不同的计划、更新周期或访问策略独立管理和操作工作负载，则多个集群更合适。

集群队列和计算资源配额

资源	限额
Slurm queues	每个集群 50 个队列
Compute resources	每个队列 50 个计算资源 每个集群 50 个计算资源

节点数

队列的 ComputeResources 中的每个计算资源都必须具有唯一的 Name、InstanceType、MinCount 和 MaxCount。MinCount 和 MaxCount 具有默认值，用于定义队列的 ComputeResources 中的计算资源的实例范围。您也可以为 MinCount 和 MaxCount 指定自己的值。ComputeResources 中的每个计算资源均由编号为 1 到 MinCount 值的静态节点和编号为 MinCount 值到 MaxCount 值的动态节点组成。

示例配置

下面是集群配置文件的 Scheduling 部分的示例。在此配置中，有两个名为 queue1 和 queue2 的队列，每个队列的 ComputeResources 都指定了 MaxCount。

```
Scheduling:
  Scheduler: slurm
  SlurmQueues:
  - Name: queue1
    ComputeResources:
    - InstanceType: c5.xlarge
      MaxCount: 5
      Name: c5xlarge
    - InstanceType: c4.xlarge
      MaxCount: 5
      Name: c4xlarge
```

```
  - Name: queue2
    ComputeResources:
    - InstanceType: c5.xlarge
      MaxCount: 5
      Name: c5xlarge
```

主机名

启动到计算实例集中的实例是动态分配的。将为每个节点生成主机名。默认情况下， AWS ParallelCluster 将使用以下格式的主机名：

$HOSTNAME=$QUEUE-$STATDYN-$COMPUTE_RESOURCE-$NODENUM

- $QUEUE 是队列的名称。例如，如果 SlurmQueues 部分中的条目将 Name 设置为“queue-name”，则“$QUEUE”为“queue-name”。
- 对于静态节点，$STATDYN 为 st，对于动态节点则为 dy。
- $COMPUTE_RESOURCE 是与此节点对应的 ComputeResources 计算资源的 Name。
- $NODENUM 是节点的编号。对于静态节点，$NODENUM 介于一 (1) 和 MinCount 的值之间，对于动态节点，则介于一 (1) 和 MaxCount-MinCount 之间。

根据上面的示例配置文件，queue1 和计算资源 c5xlarge 的给定节点的主机名为：queue1-dy-c5xlarge-1。

主机名和完全限定域名 (FQDN) 都是使用 Amazon Route 53 托管区创建的。FQDN 是 $HOSTNAME.$CLUSTERNAME.pcluster，其中 $CLUSTERNAME 是集群的名称。

请注意，Slurm 节点名称也将使用相同的格式。

用户可以选择使用为计算节点提供支持的实例的默认 Amazon EC2 主机名，而不是使用的默认主机名格式 AWS ParallelCluster。这可通过将 UseEc2Hostnames 参数设置为 true 来完成。但是，Slurm节点名称将继续使用默认 AWS ParallelCluster 格式。

Slurm 多队列模式指南

在这里，你可以了解如何 AWS ParallelCluster 以及 Slurm 管理队列（分区）节点以及如何监控队列和节点状态。

概览

扩展架构基于 Slurm的云调度指南和省电插件。有关省电插件的更多信息，请参阅 Slurm 省电指南。在架构中，可能可供群集使用的资源通常在 Slurm 配置为云节点。

云节点生命周期

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

- 处于 **POWER_SAVING** 状态的节点在 sinfo 中显示 ~ 后缀（例如 idle~）在此状态下，没有 EC2 实例支持该节点。但是，Slurm 仍然可以将任务分配给节点。
- 正在过渡到 **POWER_UP** 状态的节点将在 sinfo 中显示 # 后缀（例如 idle#）。在以下情况下，节点会自动过渡到POWER_UP状态 Slurm 将任务分配给POWER_SAVING处于状态的节点。

  或者，您可以以 su 根用户身份使用以下命令将节点手动转变为 POWER_UP 状态：

  ```
  $ scontrol update nodename=nodename state=power_up
  ```

  在此阶段，调ResumeProgram用、启动和配置 EC2实例，然后节点过渡到POWER_UP状态。
- 当前可供使用的节点在 sinfo 中不显示后缀（例如 idle）。节点设置完毕并加入集群后，即可用于运行作业。在此阶段，节点已正确配置并可供使用。

  一般而言，我们建议 Amazon EC2 实例的数量与可用节点的数量相同。在大多数情况下，在创建集群后，静态节点即可用。
- 正在过渡到 **POWER_DOWN** 状态的节点在 sinfo 中显示 % 后缀（例如 idle%）。经过 ScaledownIdletime 之后，动态节点会自动进入 POWER_DOWN 状态。相比之下，静态节点在大多数情况下不会关闭。但您可以以 su 根用户身份使用以下命令将节点手动置于 POWER_DOWN 状态：

  ```
  $ scontrol update nodename=nodename state=down reason="manual draining"
  ```

  在此状态下，与节点关联的实例将会终止，节点将设置回 POWER_SAVING 状态并在经过 ScaledownIdletime 之后可供使用。

  该ScaledownIdletime设置已保存到 Slurm 配置SuspendTimeout设置。
- 离线的节点将在 sinfo 中显示 * 后缀（例如 down*）。如果出现以下情况，则节点将脱机 Slurm 控制器无法联系节点，或者如果静态节点被禁用并且后备实例已终止。

请考虑以下 sinfo 示例中所示的节点状态。

```
$ sinfo
  PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
  efa          up   infinite      4  idle~ efa-dy-efacompute1-[1-4]
  efa          up   infinite      1   idle efa-st-efacompute1-1
  gpu          up   infinite      1  idle% gpu-dy-gpucompute1-1
  gpu          up   infinite      9  idle~ gpu-dy-gpucompute1-[2-10]
  ondemand     up   infinite      2   mix# ondemand-dy-ondemandcompute1-[1-2]
  ondemand     up   infinite     18  idle~ ondemand-dy-ondemandcompute1-
[3-10],ondemand-dy-ondemandcompute2-[1-10]
  spot*        up   infinite     13  idle~ spot-dy-spotcompute1-[1-10],spot-dy-
spotcompute2-[1-3]
  spot*        up   infinite      2   idle spot-st-spotcompute2-[1-2]
```

`spot-st-spotcompute2-[1-2]` 和 `efa-st-efacompute1-1` 节点已经设置了支持实例，并且可供使用。`ondemand-dy-ondemandcompute1-[1-2]` 节点处于 `POWER_UP` 状态，应会在几分钟内可用。`gpu-dy-gpucompute1-1` 节点处于 `POWER_DOWN` 状态，它将在 `ScaledownIdletime`（默认为 10 分钟）之后过渡到 `POWER_SAVING` 状态。

所有其他节点都处于`POWER_SAVING`状态，没有 EC2实例支持它们。

使用可用节点

可用节点由 Amazon EC2 实例提供支持。默认情况下，可以使用节点名称直接通过 SSH 加入到实例中（例如 `ssh efa-st-efacompute1-1`）。可以使用以下命令来检索实例的私有 IP 地址：

```
$ scontrol show nodes nodename
```

在返回的 `NodeAddr` 字段中检查 IP 地址。

对于不可用的节点，该`NodeAddr`字段不应指向正在运行的 Amazon EC2 实例。而是应与节点名称相同。

作业状态和提交

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

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

为作业分配的所有节点都可用后，该作业将进入 `RUNNING` (R) 状态。

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

所有节点都配置了以下特征，这些特征可以在作业提交命令中使用：

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

您可以使用以下命令查看特定节点的特征：

```
$ scontrol show nodes nodename
```

在返回的结果中，查看 AvailableFeatures 列表。

考虑集群的初始状态，可以通过运行 sinfo 命令来查看该状态。

```
$ sinfo
  PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
  efa          up   infinite      4  idle~ efa-dy-efacompute1-[1-4]
  efa          up   infinite      1   idle efa-st-efacompute1-1
  gpu          up   infinite     10  idle~ gpu-dy-gpucompute1-[1-10]
  ondemand     up   infinite     20  idle~ ondemand-dy-ondemandcompute1-
[1-10],ondemand-dy-ondemandcompute2-[1-10]
  spot*        up   infinite     13  idle~ spot-dy-spotcompute1-[1-10],spot-dy-
spotcompute2-[1-3]
  spot*        up   infinite      2   idle spot-st-spotcompute2-[1-2]
```

请注意，spot 是默认队列。它由 * 后缀表示。

向默认队列 (spot) 中的一个静态节点提交作业。

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

向 EFA 队列中的一个动态节点提交作业。

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

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

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

向 gpu 队列中的一个 GPU 节点提交作业。

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

考虑使用 squeue 命令的作业状态。

```
$ squeue
 JOBID PARTITION     NAME   USER   ST       TIME  NODES NODELIST(REASON)
  12   ondemand      wrap   ubuntu CF       0:36     10 ondemand-dy-ondemandcompute1-
[1-8],ondemand-dy-ondemandcompute2-[1-2]
  13        gpu      wrap   ubuntu CF       0:05      1 gpu-dy-gpucompute1-1
   7       spot      wrap   ubuntu  R       2:48      1 spot-st-spotcompute2-1
   8        efa      wrap   ubuntu  R       0:39      1 efa-dy-efacompute1-1
```

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

```
# Nodes states corresponds to state of running jobs
$ sinfo
 PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
 efa          up   infinite      3  idle~ efa-dy-efacompute1-[2-4]
 efa          up   infinite      1    mix efa-dy-efacompute1-1
 efa          up   infinite      1   idle efa-st-efacompute1-1
 gpu          up   infinite      1   mix~ gpu-dy-gpucompute1-1
 gpu          up   infinite      9  idle~ gpu-dy-gpucompute1-[2-10]
 ondemand     up   infinite     10   mix# ondemand-dy-ondemandcompute1-[1-8],ondemand-
dy-ondemandcompute2-[1-2]
 ondemand     up   infinite     10  idle~ ondemand-dy-ondemandcompute1-[9-10],ondemand-
dy-ondemandcompute2-[3-10]
 spot*        up   infinite     13  idle~ spot-dy-spotcompute1-[1-10],spot-dy-
spotcompute2-[1-3]
 spot*        up   infinite      1    mix spot-st-spotcompute2-1
 spot*        up   infinite      1   idle spot-st-spotcompute2-2
```

节点状态和特征

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

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

分区状态

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

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

集群 update-compute-fleet

- 停止计算队列-执行以下命令时，所有分区都将转换为INACTIVE状态， AWS ParallelCluster 进程将分区保持在INACTIVE状态。

```
$ pcluster update-compute-fleet --cluster-name testSlurm \
   --region eu-west-1 --status STOP_REQUESTED
```

- 启动计算实例集 - 执行以下命令时，所有分区最初都将转变为 UP 状态。但是， AWS ParallelCluster 进程不会使分区保持UP状态。您需要手动更改分区状态。所有静态节点将在几分钟后变为可用。请注意，将分区设置为 UP 不会增加任何动态容量。

```
$ pcluster update-compute-fleet --cluster-name testSlurm \
   --region eu-west-1 --status START_REQUESTED
```

当运行 `update-compute-fleet` 时，您可以通过运行 `pcluster describe-compute-fleet` 命令并检查 `Status` 来查看集群的状态。下面列出了可能的状态：

- STOP_REQUESTED：停止计算实例集请求已发送到集群。
- STOPPING：`pcluster` 进程当前正在停止计算实例集。
- STOPPED：`pcluster` 进程已完成停止进程，所有分区都处于 INACTIVE 状态，并且所有计算实例都已终止。
- START_REQUESTED：启动计算实例集请求已发送到集群。
- STARTING：`pcluster` 进程当前正在启动集群。
- RUNNING：`pcluster` 进程已完成启动过程，所有分区都处于 UP 状态，静态节点将在几分钟后可用。

- PROTECTED：此状态表示某些分区持续出现引导故障。受影响的分区处于非活动状态。请调查问题，然后运行 update-compute-fleet 以重新启用该实例集。

队列的手动控制

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

- 启动处于 **POWER_SAVING** 状态的动态节点

  su以根用户身份运行以下命令：

  ```
  $ scontrol update nodename=nodename state=power_up
  ```

  你也可以提交占位符sleep 1作业，请求一定数量的节点，然后依赖 Slurm 打开所需数量的节点的电源。

- 在 **ScaledownIdletime** 之前关闭动态节点

  我们建议您以 su 根用户身份使用以下命令将动态节点设置为 DOWN：

  ```
  $ scontrol update nodename=nodename state=down reason="manually draining"
  ```

  AWS ParallelCluster 自动终止并重置已关闭的动态节点。

  通常，我们不建议直接使用 scontrol update nodename=*nodename* state=power_down 命令将节点设置为 POWER_DOWN。这是因为 AWS ParallelCluster 会自动处理关闭过程，

- 禁用队列（分区）或停止特定分区中的所有静态节点

  以 su 根用户身份使用以下命令将特定队列设置为 INACTIVE：

  ```
  $ scontrol update partition=queuename state=inactive
  ```

  此操作会终止支持该分区中节点的所有实例。

- 启用队列（分区）

  以 su 根用户身份使用以下命令将特定队列设置为 UP：

  ```
  $ scontrol update partition=queuename state=up
  ```

扩展行为和调整

下面是正常扩展工作流程的示例：

- 调度器收到需要两个节点的作业。
- 调度器将两个节点转换为 POWER_UP 状态，并使用节点名称（例如 queue1-dy-spotcompute1-[1-2]）调用 ResumeProgram。
- ResumeProgram启动两个 Amazon EC2 实例并分配的私有 IP 地址和主机名queue1-dy-spotcompute1-[1-2]，等待ResumeTimeout（默认时间为 30 分钟），然后重置节点。
- 实例配置完成并加入集群。作业开始在实例上运行。
- 作业完成并停止运行。
- 经过配置的 SuspendTime（设置为 ScaledownIdletime）后，调度器将实例设置为 POWER_SAVING 状态。然后，调度器将 queue1-dy-spotcompute1-[1-2] 设置为 POWER_DOWN 状态并使用节点名称调用 SuspendProgram。
- 为两个节点调用 SuspendProgram。节点保持在 POWER_DOWN 状态，例如通过保持 idle% 状态持续 SuspendTimeout（默认时段为 120 秒（2 分钟））。在 clustermgtd 检测到节点正在关闭后，它会终止支持实例。然后，它将 queue1-dy-spotcompute1-[1-2] 转变到空闲状态并重置私有 IP 地址和主机名，使其能够启动以供将来的作业使用。

如果出现问题，特定节点的某个实例由于某种原因无法启动，则会发生以下情况：

- 调度器收到需要两个节点的作业。
- 调度器将两个云爆发节点转变为 POWER_UP 状态，并使用节点名称（例如 queue1-dy-spotcompute1-[1-2]）调用 ResumeProgram。
- ResumeProgram仅启动一 (1) 个 Amazon EC2 实例queue1-dy-spotcompute1-1，并配置一个 (1) 个实例无法启动。queue1-dy-spotcompute1-2
- queue1-dy-spotcompute1-1 未受影响，并在进入 POWER_UP 状态后上线。
- queue1-dy-spotcompute1-2转换到POWER_DOWN状态，任务会自动重新排队，因为 Slurm 检测到节点故障。
- 经过 SuspendTimeout（默认为 120 秒（2 分钟））之后 queue1-dy-spotcompute1-2 变为可用。在此期间，作业将重新排队，并可以开始在另一个节点上运行。
- 上述过程将会重复，直到作业可以在可用节点上运行而不发生故障。

有两个定时参数可以根据需要进行调整：

- **ResumeTimeout**（默认为 30 分钟）：ResumeTimeout控制时间 Slurm 在将节点转换到关闭状态之前会等待。
  - 如果您的安装前/安装后过程几乎需要那么长时间的话，延长 ResumeTimeout 可能会很有用。
  - ResumeTimeout 也是在出现问题的情况下 AWS ParallelCluster 在替换或重置节点之前等待的最长时间。如果在启动或设置过程中发生任何错误，计算节点会自行终止。 AWS ParallelCluster 检测到已终止的实例后，进程会替换节点。
- **SuspendTimeout**（默认为 120 秒（2 分钟））：SuspendTimeout 控制将节点放回系统并准备好再次使用的速率。
  - 越短SuspendTimeout意味着重置节点的速度越快，并且 Slurm 可以尝试更频繁地启动实例。
  - SuspendTimeout 越长，故障节点的重置就会越慢。同时，Slurm 尝试使用其他节点。SuspendTimeout如果超过几分钟 Slurm 尝试循环浏览系统中的所有节点。对于大型系统（超过 1,000 个节点）来说，时间更长SuspendTimeout可能有利于减轻 stress on Slurm 当它经常尝试将失败的作业重新排队时。
  - 请注意，SuspendTimeout这并不是指 AWS ParallelCluster 等待终止节点后备实例的时间。POWER_DOWN 节点的支持实例将会立即终止。终止过程通常在几分钟内完成。但在此期间，节点仍处于 POWER_DOWN 状态，无法供调度器使用。

架构的日志

以下列表包含关键日志。与 Amazon Logs 一起使用的 CloudWatch 日志流名称的格式*logIdentifier*为*{hostname}*.*{instance_id}*.*{logIdentifier}*，后面是日志名称。

- ResumeProgram: /var/log/parallelcluster/slurm_resume.log (slurm_resume)
- SuspendProgram: /var/log/parallelcluster/slurm_suspend.log (slurm_suspend)
- clustermgtd: /var/log/parallelcluster/clustermgtd.log (clustermgtd)
- computemgtd: /var/log/parallelcluster/computemgtd.log (computemgtd)
- slurmctld: /var/log/slurmctld.log (slurmctld)
- slurmd: /var/log/slurmd.log (slurmd)

常见问题以及调试方法：

无法启动、加电或加入集群的节点

- 动态节点：
  - 检查 ResumeProgram 日志，查看是否对该节点调用过 ResumeProgram。如果不是，请检查slurmctld日志以确定是否 Slurm 已尝试ResumeProgram与该节点通话。请注意，ResumeProgram 上不正确的权限可能会导致它静默失败。
  - 如果调用了 ResumeProgram，请查看是否为该节点启动了实例。如果未启动实例，则应有明确的错误消息，说明实例启动失败的原因。
  - 如果启动了实例，则可能在引导过程中出现了问题。从ResumeProgram日志中找到相应的私有 IP 地址和实例 ID，然后在 Logs 中查看特定实例的相应引导 CloudWatch 日志。
- 静态节点：
  - 检查 clustermgtd 日志，查看是否为该节点启动了实例。如果实例未启动，则应有明确的错误说明实例启动失败的原因。
  - 如果启动了实例，则引导过程出现了问题。从clustermgtd日志中找到相应的私有 IP 和实例 ID，然后在 Logs 中查看特定实例的相应引导 CloudWatch 日志。

节点意外替换或终止和节点故障

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

替换或终止实例时出现故障、关闭节点时出现故障

- 通常，clustermgtd 会处理所有预期的实例终止操作。在 clustermgtd 日志中查看其无法替换或终止节点的原因。

- 对于 ScaledownIdletime 失败的动态节点，请在 SuspendProgram 日志中查看 slurmctld 进程是否以特定节点作为参数进行了调用。请注意，SuspendProgram 实际上并不执行任何特定的操作，它只是记录被调用时的时间。所有实例终止和NodeAddr重置均由完成。clustermgtdSlurm 将节点过渡到IDLE之后SuspendTimeout。

其它问题：

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

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

Slurm 集群保护模式

当集群在启用保护模式的情况下运行时，会在启动计算节点时 AWS ParallelCluster 监控和跟踪计算节点引导失败。这样做是为了检测这些失败是否持续发生。

如果在队列（分区）中检测到以下情况，集群将进入受保护状态：

1. 持续发生连续的计算节点引导失败，没有成功的计算节点启动。
2. 失败计数达到预定义的阈值。

集群进入保护状态后， AWS ParallelCluster 禁用故障等于或高于预定义阈值的队列。

Slurm 集群保护模式已在 3.0.0 AWS ParallelCluster 版中添加。

您可以使用受保护模式来减少在计算节点引导失败循环上花费的时间和资源。

受保护模式参数

protected_failure_count

protected_failure_count 指定队列（分区）中激活集群受保护状态的连续失败次数。

默认的 protected_failure_count 为 10 并启用受保护模式。

如果 protected_failure_count 大于零，则启用受保护模式。

如果 protected_failure_count 小于或等于零，则禁用受保护模式。

通过在 HeadNode 中 /etc/parallelcluster/slurm_plugin/parallelcluster_clustermgtd.conf 处的 clustermgtd 配置文件中添加该参数，可以更改 protected_failure_count 值。

您可以随时更新此参数，并且无需停止计算实例集即可执行此操作。如果在失败计数达到 protected_failure_count 之前在队列中启动成功，则失败计数将重置为零。

在受保护状态下检查集群状态

当集群处于受保护状态时，您可以检查计算实例集状态和节点状态。

计算实例集状态

在受保护状态下运行的集群的计算实例集状态为 PROTECTED。

```
$ pcluster describe-compute-fleet --cluster-name <cluster-name> --region <region-id>
{
   "status": "PROTECTED",
   "lastStatusUpdatedTime": "2022-04-22T00:31:24.000Z"
}
```

节点状态

要了解哪些队列（分区）的引导失败已激活受保护状态，请登录集群并运行 sinfo 命令。引导失败次数达到或超过 protected_failure_count 的分区处于 INACTIVE 状态。引导失败次数未达到或超过 protected_failure_count 的分区处于 UP 状态并正常工作。

PROTECTED 状态不影响正在运行的作业。如果作业正在引导失败次数达到或超过 protected_failure_count 的分区上运行，则正在运行的作业完成后会将该分区设置为 INACTIVE。

请考虑以下示例中所示的节点状态。

```
$ sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
queue1* inact infinite 10 down% queue1-dy-c5xlarge-[1-10]
queue1* inact infinite 3490 idle~ queue1-dy-c5xlarge-[11-3500]
queue2 up infinite 10 idle~ queue2-dy-c5xlarge-[1-10]
```

分区 queue1 为 INACTIVE，因为检测到连续 10 次计算节点引导失败。

节点 queue1-dy-c5xlarge-[1-10] 后面的实例已启动，但由于运行状况不佳，未能加入集群。

集群处于受保护状态。

分区 queue2 不受 queue1 中的引导失败的影响。它处于 UP 状态并且仍然可以运行作业。

如何停用受保护状态

解决引导错误后，您可以运行以下命令使集群退出受保护状态。

```
$ pcluster update-compute-fleet --cluster-name <cluster-name> \
  --region <region-id> \
  --status START_REQUESTED
```

激活受保护状态的引导失败

激活受保护状态的引导错误细分为以下三种类型。要确定类型和问题，您可以检查是否 AWS ParallelCluster 生成的日志。如果生成了日志，则可以检查这些日志以查看错误详细信息。有关更多信息，请参阅 检索和保留日志。

1. 导致实例自行终止的引导错误。

   实例在引导过程的早期失败，例如由于 SlurmQueues\CustomActions\OnNodeStart|OnNodeConfigured 脚本中的错误而自行终止的实例。

   对于动态节点，请查找类似于下面的错误：

   ```
   Node bootstrap error: Node ... is in power up state without valid backing instance
   ```

   对于静态节点，请在 clustermgtd 日志 (/var/log/parallelcluster/clustermgtd) 中查找类似于下面的错误：

   ```
   Node bootstrap error: Node ... is in power up state without valid backing instance
   ```

2. 节点 **resume_timeout** 或 **node_replacement_timeout** 过期。

   实例无法在 resume_timeout 内（对于动态节点）或 node_replacement_timeout 内（对于静态节点）加入集群。它不会在超时之前自行终止。例如，群集的网络设置不正确，节点的DOWN状态设置为 Slurm 超时到期后。

对于动态节点，请查找类似于下面的错误：

```
Node bootstrap error: Resume timeout expires for node
```

对于静态节点，请在 `clustermgtd` 日志 (`/var/log/parallelcluster/clustermgtd`) 中查找类似于下面的错误：

```
Node bootstrap error: Replacement timeout expires for node ... in replacement.
```

3. 节点未通过运行状况检查。

   节点后面的实例未通过 Amazon EC2 运行状况检查或计划的事件运行状况检查，并且这些节点被视为引导失败节点。在这种情况下，实例因超出控制范围的原因而终止。 AWS ParallelCluster

   请在 `clustermgtd` 日志 (`/var/log/parallelcluster/clustermgtd`) 中查找类似于下面的错误：

```
Node bootstrap error: Node %s failed during bootstrap when performing health check.
```

4. 计算节点出现故障 Slurm 注册。

   将`slurmd`守护程序注册到 Slurm control daemon (`slurmctld`) 失败并导致计算节点状态更改为`INVALID_REG`状态。配置不正确 Slurm 计算节点可能导致此错误，例如配置了计算节点规格错误的`CustomSlurmSettings`计算节点。

   在头节点上的 `slurmctld` 日志文件 (`/var/log/slurmctld.log`) 或失败的计算节点上的 `slurmd` 日志文件 (`/var/log/slurmd.log`) 中查找类似于下面的错误：

```
Setting node %s to INVAL with reason: ...
```

如何调试受保护模式

如果您的集群处于受保护状态，并且 AWS ParallelCluster 生成了来自的`clustermgtd`日志`HeadNode`以及来自有问题的计算节点的`cloud-init-output`日志，则可以查看日志以了解错误详情。有关如何检索日志的更多信息，请参阅检索和保留日志。

头节点上的 **`clustermgtd`** 日志 (**`/var/log/parallelcluster/clustermgtd`**)

日志消息会显示哪些分区发生了引导失败以及相应的引导失败计数。

```
[slurm_plugin.clustermgtd:_handle_protected_mode_process] - INFO - Partitions
bootstrap failure count: {'queue1': 2}, cluster will be set into protected mode if
 protected failure count reach threshold.
```

在 `clustermgtd` 日志中，搜索 `Found the following bootstrap failure nodes` 以查找哪个节点引导失败。

```
[slurm_plugin.clustermgtd:_handle_protected_mode_process] - WARNING -
Found the following bootstrap failure nodes: (x2)  ['queue1-st-
c5large-1(192.168.110.155)',  'broken-st-c5large-2(192.168.65.215)']
```

在 `clustermgtd` 日志中，搜索 `Node bootstrap error` 以查找失败的原因。

```
[slurm_plugin.clustermgtd:_is_node_bootstrap_failure] - WARNING - Node bootstrap
 error:
Node broken-st-c5large-2(192.168.65.215) is currently in  replacement and no backing
 instance
```

计算节点上的 **`cloud-init-output`** 日志 (**`/var/log/cloud-init-output.log`**)

在 `clustermgtd` 日志中获取引导失败节点的私有 IP 地址后，您可以登录计算节点或按照检索和保留日志中的指导检索日志，找到相应的计算节点日志。在大多数情况下，有问题的节点的 `/var/log/cloud-init-output` 日志会显示导致计算节点引导失败的步骤。

Slurm 集群快速容量不足故障转移

从 AWS ParallelCluster 版本 3.2.0 开始，集群在默认情况下会启用快速容量不足故障转移模式。这样可以最大限度地减少在检测到 Amazon 容量 EC2 不足错误时重试排队任务所花费的时间。当您将队列配置为使用不同实例类型的多个计算资源时，这尤其有效。

Amazon EC2 检测到容量不足故障：

- `InsufficientInstanceCapacity`
- `InsufficientHostCapacity`
- `InsufficientReservedInstanceCapacity`
- `MaxSpotInstanceCountExceeded`
- `SpotMaxPriceTooLow`：竞价型请求价格低于要求的最低竞价型请求履行价格时激活。

- Unsupported：使用特定实例类型不支持的实例类型激活 AWS 区域。

在快速容量不足故障切换模式下，如果在将任务分配给 SlurmQueues/时检测到容量不足错误 compute resource，则 AWS ParallelCluster 执行以下操作：

1. 将计算资源设置为禁用 (DOWN) 状态，持续预定义的时间段。
2. 使用 POWER_DOWN_FORCE 取消计算资源失败的节点作业并暂停失败的节点。将失败的节点设置为 IDLE 和 POWER_DOWN (!) 状态，然后设置为 POWERING_DOWN (%)。
3. 将作业重新排队到另一个计算资源。

已禁用计算资源的静态和已启动的节点不受影响。作业可以在这些节点上完成。

此循环将会重复，直到将作业成功分配给一个或多个计算资源节点。有关节点状态的信息，请参阅 Slurm 多队列模式指南。

如果找不到运行该作业的计算资源，则将作业设置为 PENDING 状态，直到经过预定义的时间段。在这种情况下，您可以按照下一节所述修改预定义时间段。

容量不足超时参数

insufficient_capacity_timeout

insufficient_capacity_timeout 指定检测到容量不足错误时，计算资源保持在禁用 (down) 状态的时间段（以秒为单位）。

默认情况下，insufficient_capacity_timeout 处于启用状态。

默认的 insufficient_capacity_timeout 为 600 秒（10 分钟）。

如果 insufficient_capacity_timeout 值小于或等于零，则表示已禁用快速容量不足故障转移模式。

通过在 HeadNode 中 /etc/parallelcluster/slurm_plugin/parallelcluster_clustermgtd.conf 处的 clustermgtd 配置文件中添加该参数，可以更改 insufficient_capacity_timeout 值。

可以在不停止计算实例集的情况下随时更新该参数。

例如：

- `insufficient_capacity_timeout=600`:

  如果检测到容量不足错误，则会将计算资源设置为禁用 (DOWN)。10 分钟后，其失败节点将设置为 `idle~` (POWER_SAVING) 状态。
- `insufficient_capacity_timeout=60`:

  如果检测到容量不足错误，则计算资源将处于禁用状态 (DOWN)。1 分钟后，其失败节点将设置为 `idle~` 状态。
- `insufficient_capacity_timeout=0`:

  禁用了快速容量不足故障转移模式。未禁用计算资源。

> **Note**
>
> 从节点因容量不足错误而失败到集群管理进程守护程序检测到节点失败之间可能有长达一分钟的延迟。这是因为集群管理守护程序会检查节点容量不足故障，并每隔一分钟将计算资源设置为down状态。

快速容量不足故障转移模式状态

当集群处于快速容量不足故障转移模式时，您可以检查其状态和节点状态。

节点状态

当作业提交到计算资源动态节点并检测到容量不足错误时，该节点将被置于 down# 状态并显示原因：

```
(Code:InsufficientInstanceCapacity)Failure when resuming nodes.
```

然后，关闭的节点（处于 `idle~` 状态的节点）将被设置为 down~ 并显示原因：

```
(Code:InsufficientInstanceCapacity)Temporarily disabling node due to insufficient
 capacity.
```

作业重新排队到队列中的其他计算资源。

计算资源静态节点和处于 UP 状态的节点不受快速容量不足故障转移模式影响。

请考虑以下示例中所示的节点状态。

```
$ sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
queue1*   up    infinite    30  idle~ queue1-dy-c-1-[1-15],queue1-dy-c-2-[1-15]
queue2    up    infinite    30  idle~ queue2-dy-c-1-[1-15],queue2-dy-c-2-[1-15]
```

我们向 queue1 提交了需要一个节点的作业。

```
$ sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
queue1*   up   infinite  1   down# queue1-dy-c-1-1
queue1*   up   infinite  15  idle~ queue1-dy-c-2-[1-15]
queue1*   up   infinite  14  down~ queue1-dy-c-1-[2-15]
queue2    up   infinite  30  idle~ queue2-dy-c-1-[1-15],queue2-dy-c-2-[1-15]
```

节点 `queue1-dy-c-1-1` 将会启动以运行该作业。但由于容量不足错误，该实例启动失败。节点 `queue1-dy-c-1-1` 被设置为 down。该计算资源中已关闭的动态节点 (`queue2-dy-c-1`) 将被设置为 down。

您可以使用 `scontrol show nodes` 检查节点原因。

```
$ scontrol show nodes queue1-dy-c-1-1
NodeName=broken-dy-c-2-1 Arch=x86_64 CoresPerSocket=1
CPUAlloc=0 CPUTot=96 CPULoad=0.00
...
ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
Reason=(Code:InsufficientInstanceCapacity)Failure when resuming nodes
 [root@2022-03-10T22:17:50]

$ scontrol show nodes queue1-dy-c-1-2
NodeName=broken-dy-c-2-1 Arch=x86_64 CoresPerSocket=1
CPUAlloc=0 CPUTot=96 CPULoad=0.00
...
ExtSensorsJoules=n/s ExtSensorsWatts=0 ExtSensorsTemp=n/s
Reason=(Code:InsufficientInstanceCapacity)Temporarily disabling node due to
 insufficient capacity [root@2022-03-10T22:17:50]
```

作业排队到队列计算资源中的另一种实例类型。

经过 `insufficient_capacity_timeout` 之后，计算资源中的节点将重置为 `idle~` 状态。

```
$ sinfo
```

```
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
queue1*   up    infinite    30  idle~ queue1-dy-c-1-[1-15],queue1-dy-c-2-[1-15]
queue2    up    infinite    30  idle~ queue2-dy-c-1-[1-15],queue2-dy-c-2-[1-15]
```

经过 `insufficient_capacity_timeout` 并且计算资源中的节点重置为 `idle~` 状态后，Slurm 调度器会为这些节点分配较低的优先级。除非发生以下情况之一，否则调度器会继续从其他队列计算资源中选择权重较高的节点：

- 作业的提交要求与已恢复的计算资源相匹配。
- 没有其他计算资源可用，因为它们的容量已满。
- `slurmctld` 已重新启动。
- AWS ParallelCluster 计算队列已停止运行并开始关闭所有节点的电源。

相关日志

可以在头节点的 Slurm `resume` 日志和 `clustermgtd` 日志中找到与容量不足错误和快速容量不足故障转移模式相关的日志。

Slurm **`resume`** (**`/var/log/parallelcluster/slurm_resume.log`**)

节点由于容量不足而无法启动时的错误消息。

```
[slurm_plugin.instance_manager:_launch_ec2_instances] - ERROR - Failed RunInstances
 request: dcd0c252-90d4-44a7-9c79-ef740f7ecd87
[slurm_plugin.instance_manager:add_instances_for_nodes] - ERROR - Encountered
 exception when launching instances for nodes (x1) ['queue1-dy-c-1-1']: An error
 occurred
(InsufficientInstanceCapacity) when calling the RunInstances operation (reached max
 retries: 1): We currently do not have sufficient p4d.24xlarge capacity in the
Availability Zone you requested (us-west-2b). Our system will be working on
 provisioning additional capacity. You can currently get p4d.24xlarge capacity by
 not
specifying an Availability Zone in your request or choosing us-west-2a, us-west-2c.
```

Slurm **`clustermgtd`** (**`/var/log/parallelcluster/clustermgtd`**)

queue1 中的计算资源 c-1 因容量不足而被禁用。

```
[slurm_plugin.clustermgtd:_reset_timeout_expired_compute_resources] - INFO - The
 following compute resources are in down state
```

```
due to insufficient capacity: {'queue1': {'c-1':
 ComputeResourceFailureEvent(timestamp=datetime.datetime(2022, 4, 14, 23, 0, 4,
 769380, tzinfo=datetime.timezone.utc),
error_code='InsufficientInstanceCapacity')}}, compute resources are reset after
 insufficient capacity timeout (600 seconds) expired
```

容量不足超时到期后，将会重置该计算资源，并将计算资源中的节点设置为 `idle~`。

```
[root:_reset_insufficient_capacity_timeout_expired_nodes] - INFO - Reset the
 following compute resources because insufficient capacity
timeout expired: {'queue1': ['c-1']}
```

Slurm 基于内存的调度

从 3.2.0 版开始，支持 AWS ParallelCluster Slurm 使用 `SlurmSettings`/`EnableMemoryBasedScheduling`集群配置参数进行基于内存的调度。

> **Note**
>
> 从 3.7.0 AWS ParallelCluster 版开始，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`可以启用。
>
> 适用于 3.2.0 到 3.6 AWS ParallelCluster 版本。 *x*，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`无法启用。

> **Warning**
>
> 当您在中指定多个实例类型时 Slurm 队列计算资源`EnableMemoryBasedScheduling`启用后，该`RealMemory`值为所有实例类型可用的最小内存量。如果您指定的实例类型具有截然不同的内存容量，则可能会导致大量未使用的内存。

随`EnableMemoryBasedScheduling: true`着，Slurm 调度器跟踪每个任务在每个节点上所需的内存量。然后，Slurm 调度器使用此信息在同一个计算节点上调度多个作业。作业在节点上所需的内存总量不能大于可用的节点内存。调度器可防止作业使用的内存超过提交作业时请求的内存。

使用 `EnableMemoryBasedScheduling: false`，作业可能会争夺共享节点上的内存并导致作业失败和 `out-of-memory` 事件。

> **Warning**
> Slurm 对其标签使用 2 的乘方表示法，例如 MB 或 GB。将这些标签分别读作 MiB 和 GiB。

Slurm 基于配置和内存的调度

随着`EnableMemoryBasedScheduling: true`，Slurm 设置以下内容 Slurm 配置参数：

- `slurm.conf` 中的 `SelectTypeParameters=CR_CPU_Memory`。此选项将节点内存配置为中的可消耗资源 Slurm.
- `ConstrainRAMSpace=yes`在 Slurm `cgroup.conf`。 使用此选项，作业对内存的访问权限仅限于提交作业时请求的内存量。

> **Note**
> 其他几个 Slurm 配置参数可能会影响的行为 Slurm 调度程序和资源管理器（如果设置了这两个选项）。有关更多信息，请参阅 。Slurm 文档。

Slurm 调度程序和基于内存的调度

`EnableMemoryBasedScheduling: false`（默认值）

默认情况下，`EnableMemoryBasedScheduling` 设置为 false。如果为假，Slurm 在其调度算法中不包括内存作为资源，也不跟踪作业使用的内存。用户可以指定 `--mem MEM_PER_NODE` 选项来设置作业所需的每个节点的最小内存量。这会强制调度器在调度作业时选择 `RealMemory` 值至少为 `MEM_PER_NODE` 的节点。

例如，假设用户提交了两个使用 `--mem=5GB` 的作业。如果请求的资源（如 CPUs 或）可用，则 GPUs 这些作业可以在内存为 8 GiB 的节点上同时运行。不会调度这两个作业在 `RealMemory` 小于 5 GiB 的计算节点上运行。

> **Warning**
> 禁用基于内存的调度时，Slurm 不跟踪作业使用的内存量。在同一节点上运行的作业可能会争夺内存资源并导致其他作业失败。

在禁用基于内存的调度时，我们建议用户不要指定 `--mem-per-cpu` 或 `--mem-per-gpu` 选项。这些选项可能导致的行为与中描述的有所不同 Slurm 文档。

`EnableMemoryBasedScheduling: true`

当设置`EnableMemoryBasedScheduling`为 true 时，Slurm 使用`--mem`提交选项跟踪每个作业的内存使用情况，并防止作业使用的内存超过请求的内存量。

在上面的示例中，用户提交了两个使用 `--mem=5GB` 的作业。这些作业无法在内存为 8 GiB 的节点上同时运行。这是因为所需的总内存量大于节点上可用的内存量。

启用基于内存的调度后，其`--mem-per-gpu`行为`--mem-per-cpu`与中描述的内容保持一致 Slurm 文档中）。例如，使用 `--ntasks-per-node=2 -c 1 --mem-per-cpu=2GB` 提交作业。在这种情况下，Slurm 为每个节点分配总计 4 GiB 的任务。

> **Warning**
>
> 启用基于内存的调度后，我们建议用户在提交作业时包含 `--mem` 规范。使用默认值 Slurm 包含的配置 AWS ParallelCluster，如果不包含内存选项（`--mem`--`mem-per-cpu`、或--`mem-per-gpu`），Slurm 将分配的节点的全部内存分配给该作业，即使它只请求一部分其他资源，例如 CPUs 或 GPUs。这可在作业完成之前有效地防止节点共享，因为没有内存可用于其他作业。发生这种情况是因为 Slurm 将作业的每个节点的内存设置为在提交作业`DefMemPerNode`时未提供任何内存规格时。此参数的默认值为 0，指定的是对节点内存进行无限制访问。
>
> 如果同一个队列中存在多种具有不同内存量的计算资源，则在不同的节点上可能会为不带内存选项的已提交作业分配不同的内存量。这取决于调度器为作业提供了哪些节点。用户可以在集群或分区级别为选项（例如`DefMemPerNode`或 `DefMemPerCPU`）定义自定义值 Slurm 配置文件以防止这种行为。

Slurm RealMemory 和 AWS ParallelCluster SchedulableMemory

使用 Slurm 随附的配置 AWS ParallelCluster，Slurm 解释RealMemory为每个节点可用于作业的内存量。从版本 3.2.0 开始，默认 AWS ParallelCluster 设置`RealMemory`为亚马逊 EC2 实例类型中列出并由亚马逊 EC2 API DescribeInstanceTypes返回的内存的 95%。

禁用基于内存的调度时，Slurm 当用户提交`--mem`指定任务时`RealMemory`，调度器用于筛选节点。

启用基于内存的调度后，Slurm 调度器解释RealMemory为计算节点上运行的作业可用的最大内存量。

默认设置可能不是所有实例类型的最佳设置：

- 此设置可能高于节点实际可以访问的内存量。当计算节点是小型实例类型时，可能会发生这种情况。
- 此设置可能低于节点实际可以访问的内存量。当计算节点是大型实例类型时，可能会发生这种情况，并可能导致大量未使用的内存。

您可以使用 SlurmQueues/ComputeResources/SchedulableMemory微调计算节点 AWS ParallelCluster 的RealMemory配置值。要覆盖默认值，请针对您的集群配置专门为 SchedulableMemory 定义一个自定义值。

要检查计算节点的实际可用内存，请在该节点上运行 /opt/slurm/sbin/slurmd -C 命令。此命令可返回节点的硬件配置，包括 RealMemory 值。有关更多信息，请参阅 slurmd -C。

确保计算节点的操作系统进程有足够的内存。为此，请将 SchedulableMemory 值设置为低于 slurmd -C 命令返回的 RealMemory 值，从而限制作业可用的内存。

Slurm 的多实例类型分配

从 3.3.0 AWS ParallelCluster 版开始，您可以将集群配置为从计算资源的一组已定义的实例类型中进行分配。可以基于 Amazon EC2 机队的低成本或最佳容量策略进行分配。

这组已定义的实例类型必须全部具有相同数量的 v，CPUs 或者如果禁用了多线程，则必须具有相同数量的内核。此外，这组实例类型必须具有相同制造商的相同数量的加速器。如果 Efa/Enabled 设置为 true，则实例必须支持 EFA。有关更多信息和要求，请参阅 Scheduling/SlurmQueues/AllocationStrategy 和 ComputeResources/Instances。

capacity-optimized根据您的CapacityType配置 AllocationStrategy，可以设置为lowest-price或。

在 Instances 中，您可以配置一组实例类型。

> **Note**
>
> 从 3.7.0 AWS ParallelCluster 版开始，如果您在实例中配置了多个实例类型，则EnableMemoryBasedScheduling可以启用。
>
> 适用于 3.2.0 到 3.6 AWS ParallelCluster 版本。 *x*，如果您在实例中配置了多个实例类型，则EnableMemoryBasedScheduling无法启用。

以下示例显示了如何查询 v CPUs、EFA 支持和架构的实例类型。

Query InstanceTypes 采用 96 v CPUs 和 x86_64 架构。

```
$ aws ec2 describe-instance-types --region region-id \
  --filters "Name=vcpu-info.default-vcpus,Values=96" "Name=processor-info.supported-
architecture,Values=x86_64" \
  --query "sort_by(InstanceTypes[*].
{InstanceType:InstanceType,MemoryMiB:MemoryInfo.SizeInMiB,CurrentGeneration:CurrentGeneration,V
 &InstanceType)" \
  --output table
```

Query InstanceTypes 拥有 64 个内核、支持 EFA 和 arm64 架构。

```
$ aws ec2 describe-instance-types --region region-id \
  --filters "Name=vcpu-info.default-cores,Values=64" "Name=processor-
info.supported-architecture,Values=arm64" "Name=network-info.efa-
supported,Values=true" --query "sort_by(InstanceTypes[*].
{InstanceType:InstanceType,MemoryMiB:MemoryInfo.SizeInMiB,CurrentGeneration:CurrentGeneration,V
 &InstanceType)" \
  --output table
```

下一个集群配置示例片段显示了如何使用这些配置 InstanceType 以及 AllocationStrategy 属性。

```
...
 Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue-1
      CapacityType: ONDEMAND
      AllocationStrategy: lowest-price
      ...
      ComputeResources:
        - Name: computeresource1
          Instances:
            - InstanceType: r6g.2xlarge
            - InstanceType: m6g.2xlarge
            - InstanceType: c6g.2xlarge
          MinCount: 0
          MaxCount: 500
        - Name: computeresource2
          Instances:
```

```
            - InstanceType: m6g.12xlarge
            - InstanceType: x2gd.12xlarge
          MinCount: 0
          MaxCount: 500
...
```

动态节点的集群扩展

ParallelCluster 支持 Slurm的方法通过使用来动态缩放集群 Slurm的省电插件。有关更多信息，请参阅《云计划指南》和 Slurm中的 “省电指南” Slurm 文档中）。以下主题描述了 Slurm 每个版本的策略。

主题

- Slurm 3.8.0 版中的动态节点分配策略
- Slurm 3.7.x 版中的动态节点分配策略
- Slurm 3.6.x 及之前版本中的动态节点分配策略

Slurm 3.8.0 版中的动态节点分配策略

从 ParallelCluster 版本 3.8.0 开始， ParallelCluster 使用作业级恢复或作业级扩展作为默认的动态节点分配策略来扩展集群： ParallelCluster 根据每个作业的要求、分配给任务的节点数量以及需要恢复的节点数量来扩展集群。 ParallelCluster 从 SLURM_RESUME_FILE 环境变量中获取此信息。

动态节点的扩展分为两个步骤，包括启动 EC2实例和将启动的 Amazon EC2 实例分配给 Slurm 节点。这两个步骤中的每一个都可以使用all-or-nothing或尽力而为的逻辑来完成。

要启动 Amazon EC2 实例，请执行以下操作：

- all-or-nothing调用启动的 Amazon EC2 API，最小目标等于目标总容量
- 尽力调用启动 Amazon EC2 API，最低目标等于 1，总目标容量等于请求的容量

用于将 Amazon EC2 实例分配给 Slurm 节点：

- all-or-nothing将 Amazon EC2 实例分配给 Slurm 仅当可以为每个请求的节点分配一个 Amazon EC2 实例时才有节点
- 尽最大努力将 Ama EC2 zon 实例分配给 Slurm 节点，即使 Amazon EC2 实例容量未涵盖所有请求的节点

  上述策略的可能组合转化为 ParallelCluster发射策略。

Example

<caption>The available ParallelCluster 启动策略 that can be set into the ScalingStrategy cluster configuration to be used with 作业级扩展 are:</caption>

all-or-nothing缩放：

此策略包括 AWS ParallelCluster 为每个任务 EC2 启动 Amazon 启动实例 API 调用，这要求成功启动所请求的计算节点所需的所有实例。这可确保集群只在每个作业所需的容量可用时才进行扩展，避免在扩展过程结束时留下闲置实例。

该策略使用all-or-nothing逻辑来启动每项任务的 Amazon EC2 实例 plus，并使用all-or-nothing逻辑将亚马逊 EC2 实例分配给 Slurm 节点。

该策略将启动请求分为若干批次，每个请求的计算资源一个批次，每个批次最多 500 个节点。对于跨越多个计算资源或超过 500 个节点的请求， ParallelCluster 按顺序处理多个批次。

任何单一资源的批处理失败都会导致所有相关未使用容量的终止，从而确保在扩展过程结束时不会留下任何闲置实例。

限制

- 扩展所需的时间与每次执行提交的任务数量成正比 Slurm 恢复程序。
- 扩展操作受 RunInstances 资源账户限制的限制，默认情况下设置为 1000 个实例。此限制符合 AWS EC2 API 限制政策，有关更多详细信息，请参阅 A mazon EC2 API 限制文档
- 当您在具有单一实例类型的计算资源中提交任务时，在跨越多个可用区的队列中，只有在单个可用区中可以提供所有容量时，all-or-nothing EC2启动 API 调用才会成功。
- 当您在具有多种实例类型的计算资源中提交任务时，在具有单个可用区的队列中，只有当所有容量均可由单个实例类型提供时，all-or-nothingAmazon La EC2 unch API 调用才会成功。
- 当您在具有多种实例类型的计算资源中提交任务时，在跨越多个可用区的队列中，all-or-nothingAmazon la EC2 unch API 调用不受支持，而是 ParallelCluster 执行尽力扩展。

greedy-all-or-nothing缩放：

该 all-or-nothing策略的这种变体仍然可以确保集群仅在每个任务所需的容量可用时才进行扩展，从而避免在扩展过程结束时出现空闲实例，但它涉及 ParallelCluster 启动一个旨在实现最低目标容量为 1 的 Amazon EC2 启动实例 API 调用，尝试将启动的节点数量最大化到请求的容量。该策略使用尽力为所有任务启动 EC2实例的逻辑，以及将 Amazon EC2 实例分配给的all-or-nothing逻辑 Slurm 每个作业的节点。

该策略将启动请求分为若干批次，每个请求的计算资源一个批次，每个批次最多 500 个节点。对于跨越多个计算资源或超过 500 个节点的请求， ParellelCluster 按顺序处理多个批次。

该策略以在扩展过程中暂时过度扩展为代价，最大限度地提高吞吐量，从而确保在扩展过程结束时不会留下闲置实例。

限制

- 可能会出现临时的过度扩展，导致在扩展完成前过渡到运行状态的实例产生额外费用。
- 与 all-or-nothing策略中相同的实例限制适用，具体取决于 AWS的 RunInstances 资源账户限制。

最大努力扩展：

此策略调用 Amazon EC2 启动实例 API 的方法是，将最小容量定为 1，目标是实现请求的总容量，但代价是在扩展过程执行完毕后，如果不是所有请求的容量都可用，则在扩展过程执行后留下空闲的实例。该策略使用尽力逻辑启动所有任务的 Amazon EC2 实例，以及为每个任务将 Amazon EC2 实例分配给 Slurm 节点的最大努力逻辑。

该策略将启动请求分为若干批次，每个请求的计算资源一个批次，每个批次最多 500 个节点。对于跨越多个计算资源或超过 500 个节点的请求， ParallelCluster 按顺序处理多个批次。

这种策略允许在多次扩展过程执行中，扩展的实例数量远远超过默认的 1000 个实例限制，但代价是在不同扩展进程中出现闲置实例。

限制

- 在扩展过程结束时可能出现的闲置运行实例，用于无法分配作业要求的所有节点的情况。

以下示例显示了使用不同的ParallelCluster 启动策略扩展动态节点的行为。假设您提交了两个任务，每个任务请求了 20 个节点，总共有 40 个相同类型的节点，但只有 30 EC2 个 Amazon 实例可用于满足请求的容量 EC2。

all-or-nothing缩放：

- 对于第一个任务，将调用 A all-or-nothingmazon EC2 启动实例 API，请求了 20 个实例。一次成功的调用启动了 20 个实例
- all-or-nothing 将 20 个已启动的实例分配给 Slurm 第一个作业的节点成功
- 调用了另一个 all-or-nothingAmazon EC2 启动实例 API，为第二个任务请求 20 个实例。调用不成功，因为只有另外 10 个实例的容量。目前未启动任何实例

greedy-all-or-nothing缩放：

- 调用了尽力而为的 Amazon EC2 启动实例 API，请求了 40 个实例，这是所有任务请求的总容量。这会导致启动 30 个实例
- 将 20 个已启动的实例all-or-nothing分配给 Slurm 第一个作业的节点成功
- 将其余已启动实例的另一次all-or-nothing分配给 Slurm 已尝试第二个作业的节点，但由于该作业请求的总共 20 个实例中只有 10 个可用实例，因此分配不成功
- 10 个未分配的已启动实例被终止

最大努力扩展：

- 调用了尽力而为的 Amazon EC2 启动实例 API，请求了 40 个实例，这是所有任务请求的总容量。这会导致启动 30 个实例。
- 尽力将 20 个已启动的实例分配给 Slurm 第一个作业的节点成功。
- 在剩下的 10 个已启动实例中，另一项尽力分配给 Slurm 即使请求的总容量为 20，第二个任务的节点也会成功。但是，由于任务请求的是 20 个节点，并且只能将 Amazon EC2 实例分配给其中 10 个节点，因此任务无法启动，实例处于空闲状态，直到找到足够的容量在稍后调用扩展过程时启动缺少的 10 个实例，或者调度器将任务安排在其他已经运行的计算节点上。

Slurm 3.7.x 版中的动态节点分配策略

ParallelCluster 使用 2 种类型的动态节点分配策略来扩展集群：

- 根据可用的请求节点信息进行分配：
  - 所有节点恢复或节点列表扩展：

    ParallelCluster 仅根据以下条件扩大集群 Slurm请求的节点列表名称时 Slurm 的`ResumeProgram`跑步。它仅按节点名称向节点分配计算资源。节点名称列表可以跨越多个作业。
  - 作业级别恢复或作业级别扩展：

    ParallelCluster 根据每个作业的要求、当前分配给该任务的节点数以及需要恢复的节点来扩展集群。 ParallelCluster 从`SLURM_RESUME_FILE`环境变量中获取此信息。
- 使用 Amazon EC2 发布策略进行分配：
  - 最大努力扩展：

ParallelCluster 使用最小目标容量等于 1 的 Amazon EC2 启动实例 API 调用来扩展集群，启动支持请求的节点所需的部分实例，但不一定是全部实例。

- 缩ll-or-nothing放：

  ParallelCluster 使用 Amazon EC2 启动实例 API 调用扩展集群，只有在支持请求的节点所需的所有实例都启动后，该调用才会成功。在这种情况下，它调用 Amazon EC2 启动实例 API 的最小目标容量等于请求的总容量。

默认情况下， ParallelCluster 使用节点列表扩展和尽力而为 A EC2 mazon 启动策略来启动支持请求的节点所需的部分实例，但不一定是全部实例。它会尝试预置尽可能多的容量来处理所提交的工作负载。

从 3.7.0 ParallelCluster 版开始， ParallelCluster 使用作业级缩放和all-or-nothing EC2启动策略，用于在独占模式下提交的作业。当您在独占模式下提交作业时，该作业对其分配的节点拥有独占访问权限。有关更多信息，请参阅《独家》中的 Slurm 文档中）。

要以独占模式提交作业，请执行以下操作：

- 提交时通过专属标志 Slurm 任务到集群。例如 `sbatch ... --exclusive`。

  或

- 向 `JobExclusiveAllocation` 设置为 `true` 的已配置集群队列提交作业。

以独占模式提交作业时：

- ParallelCluster 目前批量启动请求最多包含 500 个节点。如果任务请求的节点超过 500 个， ParallelCluster 则为每组 500 个节点发出启动请求，为其余节点发出额外的启动请求。all-or-nothing
- 如果节点分配在单个计算资源中， ParallelCluster 则为每组 500 个节点发出启动请求，为其余节点发出额外的启动请求。all-or-nothing如果启动请求失败， ParallelCluster 将终止所有启动请求创建的未使用容量。
- 如果节点分配跨越多个计算资源，则 ParallelCluster 需要为每个计算资源发出all-or-nothing启动请求。这些请求也会进行批处理。如果其中一个计算资源的启动请求失败，则 ParallelCluster 终止所有计算资源启动请求所创建的未使用容量。

使用all-or-nothing启动策略进行@@ 作业级扩展已知局限性：

- 当您在具有单一实例类型的计算资源中提交任务时，在跨越多个可用区的队列中，只有在单个可用区中可以提供所有容量时，all-or-nothing EC2启动 API 调用才会成功。

- 当您在具有多种实例类型的计算资源中提交任务时，在具有单个可用区的队列中，只有当所有容量均可由单个实例类型提供时，all-or-nothingAmazon La EC2 unch API 调用才会成功。
- 当您在具有多种实例类型的计算资源中提交任务时，在跨越多个可用区的队列中，all-or-nothingAmazon la EC2 unch API 调用不受支持，而是 ParallelCluster执行尽力扩展。

Slurm 3.6.x 及之前版本中的动态节点分配策略

AWS ParallelCluster 仅使用一种类型的动态节点分配策略来扩展集群：

- 根据可用的请求节点信息进行分配：
  - 全节点恢复或节点列表扩展：仅 ParallelCluster 根据以下条件扩展集群 Slurm请求的节点列表名称时 Slurm的`ResumeProgram`跑步。它仅按节点名称向节点分配计算资源。节点名称列表可以跨越多个作业。
- 使用 Amazon EC2 发布策略进行分配：
  - 尽力扩 ParallelCluster 展：使用最小目标容量等于 1 的 Amazon EC2 启动实例 API 调用来扩展集群，启动支持请求的节点所需的部分但不一定是全部实例。

ParallelCluster 使用节点列表扩展和尽力而为 A EC2 mazon 启动策略来启动支持请求的节点所需的部分实例，但不一定是全部实例。它会尝试预置尽可能多的容量来处理所提交的工作负载。

限制

- 在扩展过程结束时可能出现的闲置运行实例，用于无法分配作业要求的所有节点的情况。

Slurm 用会计 AWS ParallelCluster

从 3.3.0 版开始，支持 AWS ParallelCluster Slurm 使用集群配置参数 SlurmSettings/数据库进行核算。

从 3.10.0 版开始，支持 AWS ParallelCluster Slurm 使用带有集群配置参数/的外部 Slurmdbd 进行核算。SlurmSettingsExternalSlurmdbd如果多个集群共享同一个数据库，建议使用外部 Slurmdbd。

With Slurm 会计，您可以集成外部会计数据库来执行以下操作：

- 管理集群用户或用户组和其他实体。有了这个功能，你可以使用 Slurm的更高级的功能，例如资源限制强制执行、公平共享和。 QOSs
- 收集并保存作业数据，例如运行作业的用户、作业的持续时间及其使用的资源。您可以使用 `sacct` 实用工具查看保存的数据。

> **Note**
>
> AWS ParallelCluster 支持 Slurm 会计 Slurm 支持的 MySQL 数据库服务器。

与 Slurm 使用外部进行会计 Slurmdbd 在 AWS ParallelCluster v3.10.0 及更高版本中

在配置之前 Slurm 会计，你必须有现有的外部账号 Slurmdbd 数据库服务器，它连接到现有的外部数据库服务器。

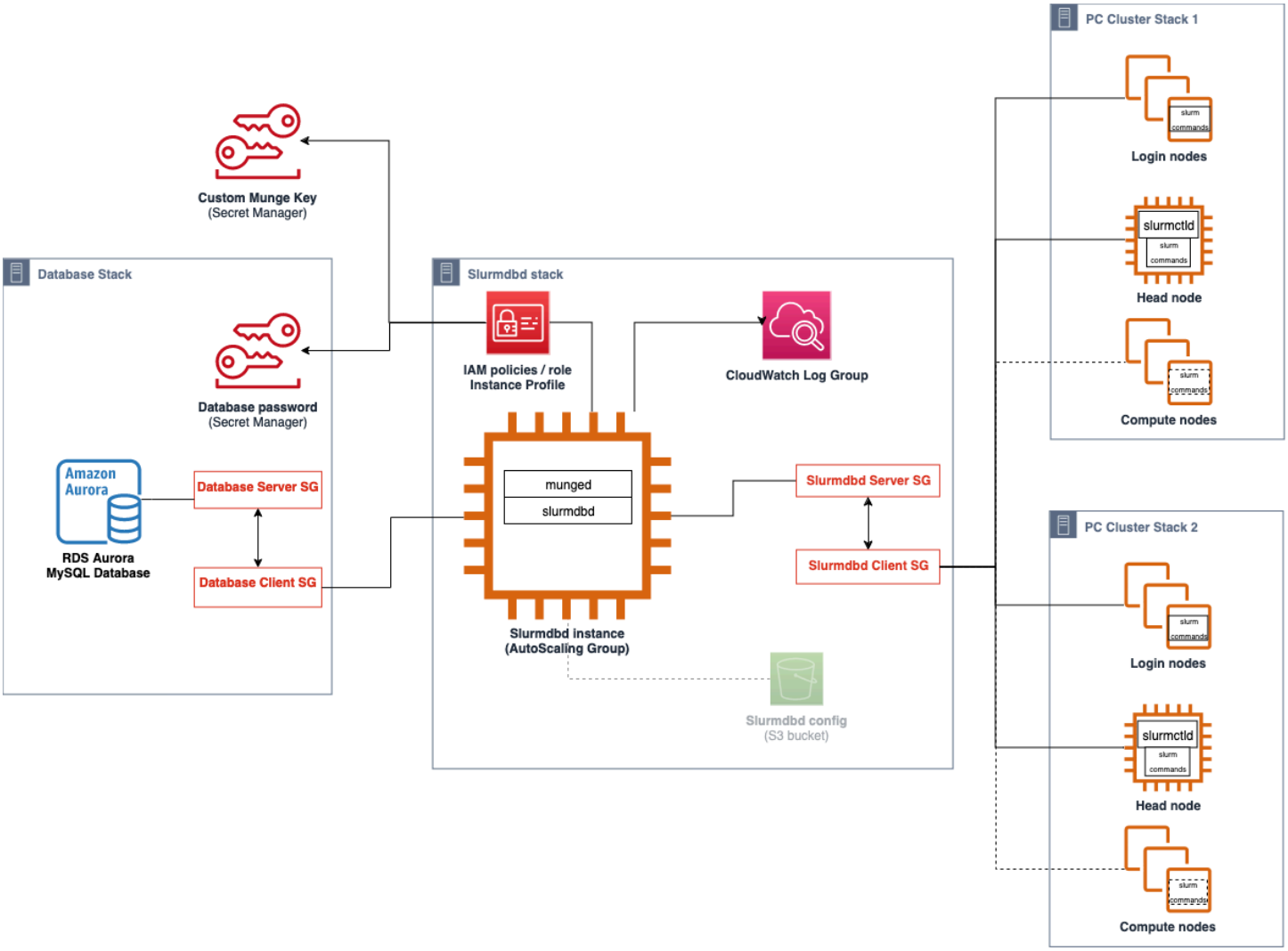
要对此进行配置，请定义以下内容：

- 外部的地址 Slurmdbd 服务器位于 ExternalSlurmdbd/Host 中。服务器必须存在并且可以从头节点访问。
- 用于与外部通信的 munge 密钥 Slurmdbd 服务器在MungeKeySecretArn。

要查看分步教程，请参阅使用外部集群创建集群 Slurmdbd 会计。

> **Note**
>
> 你有责任管理 Slurm 数据库记账实体。

AWS ParallelCluster 外部的架构 SlurmDB 支持功能允许多个集群共享相同的集群 SlurmDB 和同一个数据库。



> **Warning**
>
> 和外部 AWS ParallelCluster 之间的流量 SlurmDB 未加密。建议运行集群和外部集群 SlurmDB 在可信的网络中。

与 Slurm 使用头节点进行记账 Slurmdbd 在 AWS ParallelCluster v3.3.0 及更高版本中

在配置之前 Slurm 记账，您必须拥有现有的外部数据库服务器和使用`mysql`协议的数据库。

要配置 Slurm 使用记账 AWS ParallelCluster，您必须定义以下内容：

- 在 Database/Uri 中定义的外部数据库服务器的 URI。服务器必须存在并且可以从头节点访问。

- 访问数据库/ PasswordSecretArn和数据库/中定义的外部数据库的凭据UserName。 AWS ParallelCluster 使用此信息在上配置记账 Slurm 关卡和头节点上的`slurmdbd`服务。 `slurmdbd`是管理群集和数据库服务器之间通信的守护程序。

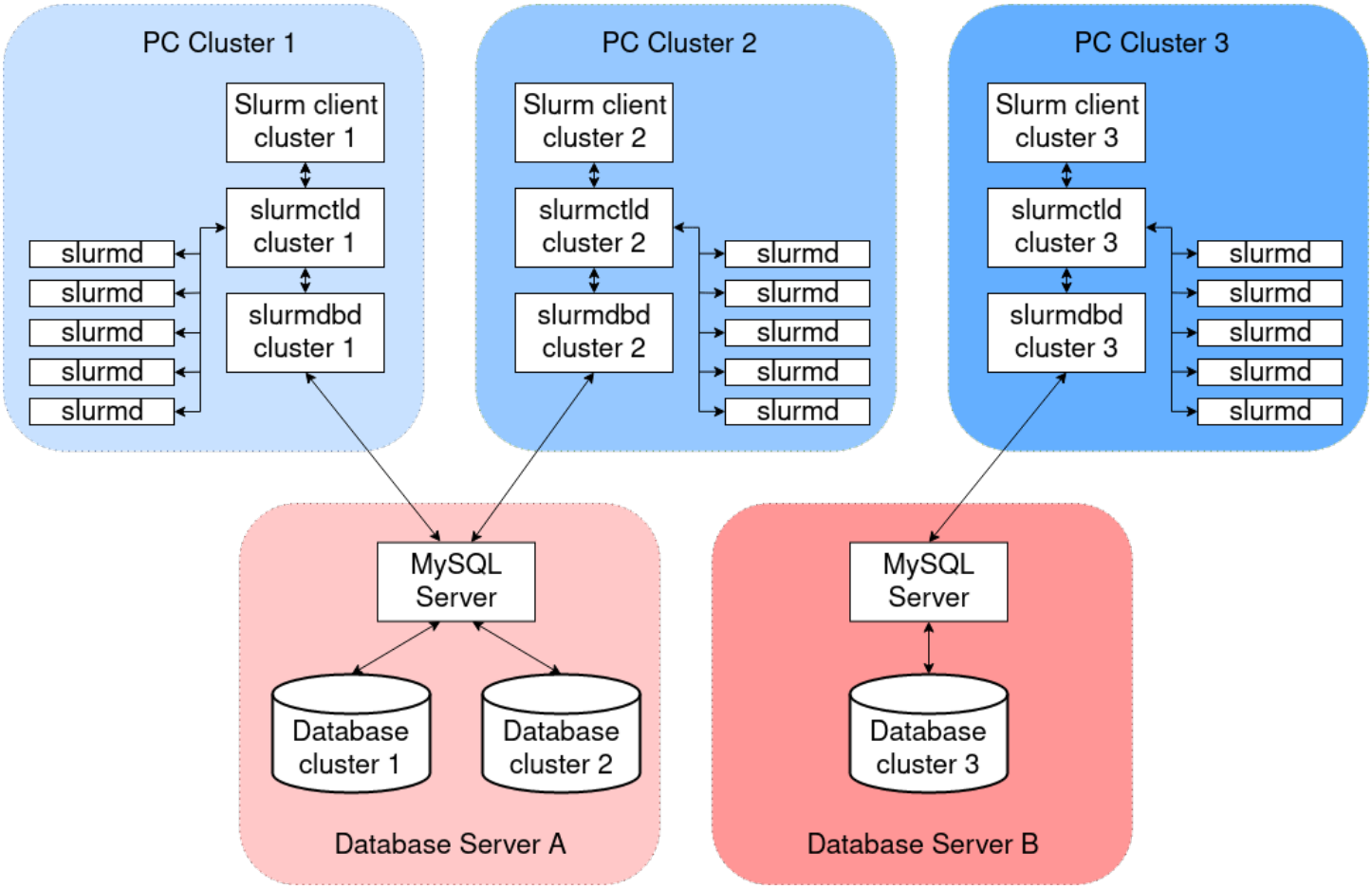
要查看分步教程，请参阅使用创建集群 Slurm 会计。

> **Note**
>
> AWS ParallelCluster 执行基本的引导程序 Slurm 通过在中将默认集群用户设置为数据库管理员来记账数据库 Slurm 数据库。 AWS ParallelCluster 不会向会计数据库添加任何其他用户。客户负责管理中的会计实体 Slurm 数据库。

AWS ParallelCluster 配置`slurmdbd`以确保集群有自己的集群 Slurm 数据库服务器上的数据库。同一台数据库服务器可以跨多个群集使用，但每个群集都有自己的独立数据库。 AWS ParallelCluster 使用集群名称在`slurmdbd`配置文件`StorageLoc`参数中定义数据库的名称。请考虑以下情况：数据库服务器上存在的数据库包括的集群名称未映射到有效的集群名称。在这种情况下，您可以使用该集群名称创建一个新集群以映射到该数据库。Slurm 为新集群重复使用数据库。

> **Warning**
>
> - 我们不建议设置多个集群同时使用同一个数据库。这样做可能会导致性能问题，甚至导致数据库死锁情况。
> - 如果 Slurm 在集群的头节点上启用了记账，我们建议使用具有强大 CPU、更多内存和更高网络带宽的实例类型。Slurm 记账可能会给集群的头节点增加压力。

在当前的架构中 AWS ParallelCluster Slurm 记账功能，每个集群都有自己的`slurmdbd`守护程序实例，如下图示例配置所示。



如果你要添加自定义 Slurm 您的集群环境具有多集群或联邦功能，所有集群都必须引用同一个`slurmdbd`实例。对于此替代方案，我们建议您启用 AWS ParallelCluster Slurm 对一个群集进行核算，然后手动配置其他群集以连接到第一个群集上托管的群集。`slurmdbd`

如果您使用 AWS ParallelCluster 的是 3.3.0 之前的版本，请参阅替代方法来实现 Slurm 这篇 HPC 博客文章中描述的会计。

Slurm 会计注意事项

数据库和集群在不同的地方 VPCs

要启用 Slurm 记账，需要数据库服务器作为`slurmdbd`守护程序执行的读取和写入操作的后端。在创建或更新集群以启用集群之前 Slurm 记账，头节点必须能够访问数据库服务器。

如果您需要在集群使用的 VPC 之外的不同 VPC 上部署数据库服务器，请考虑以下事项：

- 要启用群集端和数据库服务器之间的通信，必须设置两者之间的连接 VPCs。`slurmdbd`有关更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的 VPC 对等连接。

- 您必须在集群的 VPC 上创建要连接到头节点的安全组。两者建立 VPCs 对等关系后，即可在数据库端和集群端安全组之间进行交叉链接。有关更多信息，请参阅 Amazon Virtual Private Cloud 用户指南 中的安全组规则。

配置 **slurmdbd** 和数据库服务器之间的 TLS 加密

使用默认值 Slurm 如果服务器支持 TLS 加密，则 AWS ParallelCluster 提供与数据库服务器`slurmdbd`建立 TLS 加密连接的记账配置。 AWS 默认情况下，Amazon RDS 等数据库服务 Amazon Aurora 支持 TLS 加密。

通过在数据库服务器上设置 `require_secure_transport` 参数，可以在服务器端要求安全连接。这是在提供的 CloudFormation 模板中配置的。

根据安全性方面的最佳实践，我们建议您同时在 `slurmdbd` 客户端上启用服务器身份验证。为此，请在StorageParameters中配置`slurmdbd.conf`。将服务器 CA 证书上传到集群的头节点。接下来，将 `slurmdbd.conf` 中 `StorageParameters` 的 SSL_CA 选项设置为头节点上服务器 CA 证书的路径。这样做会在 `slurmdbd` 侧启用服务器身份验证。进行这些更改后，重启 `slurmdbd` 服务以便在启用身份验证的情况下重新建立与数据库服务器的连接。

更新数据库凭证

要更新数据库/UserName或的值 PasswordSecretArn，必须先停止计算队列。假设存储在密钥中的 AWS Secrets Manager 密钥值已更改，且其 ARN 未更改。在这种情况下，集群不会自动将数据库密码更新为新值。要针对新密钥值更新集群，请从头节点运行以下命令。

```
$ sudo /opt/parallelcluster/scripts/slurm/update_slurm_database_password.sh
```

> **Warning**
> 为避免会计数据丢失，我们建议仅在已停止计算实例集的情况下更改数据库密码。

数据库监控

我们建议您启用 AWS 数据库服务的监控功能。有关更多信息，请参阅 Amazon RDS 监控或 Amazon Aurora 监控文档。

Slurm 配置自定义

从 3.6.0 AWS ParallelCluster 版开始，您可以自定义 `slurm.conf` Slurm AWS ParallelCluster 集群配置中的配置。

在集群配置中，您可以自定义 Slurm 使用以下集群配置设置配置参数：

- 自定义 Slurm 使用 `SlurmSettings`/`CustomSlurmSettings`或参数为整个集群`CustomSlurmSettingsIncludeFile`设置参数。 AWS ParallelCluster 如果同时指定两者，则失败。
- 自定义 Slurm 使用 `SlurmQueues`/为队列设置参数 `CustomSlurmSettings`（映射到 Slurm 分区）。
- 自定义 Slurm 使用 `SlurmQueues`/`ComputeResources`/为计算资源设置参数 `CustomSlurmSettings`（映射到 Slurm 节点）。

Slurm 配置自定义限制和使用时的注意事项 AWS ParallelCluster

- 对于`CustomSlurmSettings`和`CustomSlurmSettingsIncludeFile`设置，您只能指定和更新包含在中的`slurm.conf`参数 Slurm 您用于配置集群的 AWS ParallelCluster 版本所支持的版本。
- 如果您指定自定义 Slurm 在任何`CustomSlurmSettings`参数中进行配置， AWS ParallelCluster 执行验证检查并阻止设置或更新 Slurm 与 AWS ParallelCluster 逻辑冲突的配置参数。这些区域有：Slurm 已知与 AWS ParallelCluster 之冲突的配置参数在拒绝列表中标识。如果有其他情况，拒绝列表可能会在 future AWS ParallelCluster 版本中发生变化 Slurm 功能已添加。有关更多信息，请参阅被拒登名单 Slurm 的配置参数 `CustomSlurmSettings`。
- AWS ParallelCluster 仅检查参数是否在拒绝列表中。 AWS ParallelCluster 无法验证您的自定义 Slurm 配置参数语法或语义。您有责任验证您的自定义 Slurm 配置参数。无效的自定义 Slurm 配置参数可能导致 Slurm 守护程序失败可能导致集群创建和更新失败。
- 如果您指定自定义 Slurm 中的配置`CustomSlurmSettingsIncludeFile`， AWS ParallelCluster 不执行任何验证。
- 您可以更新 `CustomSlurmSettings` 和 `CustomSlurmSettingsIncludeFile` 而不停止然后启动计算实例集。在这种情况下， AWS ParallelCluster 重新启动`slurmctld`守护程序并运行命令。`scontrol reconfigure`

  一段时间 Slurm 在整个集群中注册更改之前，可能需要对配置参数进行不同的操作。例如，它们可能需要重启集群中的所有进程守护程序。您有责任验证 AWS ParallelCluster 操作是否足以传播您的

自定义内容 Slurm 更新期间的配置参数设置。如果您发现 AWS ParallelCluster 操作还不够，则您有责任按照中的建议提供传播更新后的设置所需的额外操作 Slurm 文档。

被拒登名单 Slurm 的配置参数 **`CustomSlurmSettings`**

下表列出了拒绝使用参数的 AWS ParallelCluster 版本（从 3.6.0 版本开始）。`CustomSlurmSettings`3.6.0 之前的 AWS ParallelCluster 版本不支持。

集群级别列入拒绝列表的参数：

Slurm 参数	已在版本中列出拒绝 AWS ParallelCluster
CommunicationParameters	3.6.0
Epilog	3.6.0
GresTypes	3.6.0
LaunchParameters	3.6.0
Prolog	3.6.0
ReconfigFlags	3.6.0
ResumeFailProgram	3.6.0
ResumeProgram	3.6.0
ResumeTimeout	3.6.0
SlurmctldHost	3.6.0
SlurmctldLogFile	3.6.0
SlurmctldParameters	3.6.0
SlurmdLogfile	3.6.0
SlurmUser	3.6.0
SuspendExcNodes	3.6.0

Slurm 参数	已在版本中列出拒绝 AWS ParallelCluster
SuspendProgram	3.6.0
SuspendTime	3.6.0
TaskPlugin	3.6.0
TreeWidth	3.6.0

本机时在集群级别被拒绝列出的参数 Slurm 记账集成是在群集配置中配置的：

Slurm 参数	已在版本中列出拒绝 AWS ParallelCluster
AccountingStorageType	3.6.0
AccountingStorageHost	3.6.0
AccountingStoragePort	3.6.0
AccountingStorageUser	3.6.0
JobAcctGatherType	3.6.0

队列管理的队列的队列（分区）级别的拒绝列表参数： AWS ParallelCluster

Slurm 参数	已在版本中列出拒绝 AWS ParallelCluster
Nodes	3.6.0
PartitionName	3.6.0
ResumeTimeout	3.6.0
State	3.6.0
SuspendTime	3.6.0

由以下机构管理的计算资源（节点）级别的拒绝列出的计算资源的参数： AWS ParallelCluster

Slurm 参数	已在版本和更高 AWS ParallelCluster 版本中列出 “拒绝”
CPUs	3.6.0
Features	3.6.0
Gres	3.6.0
NodeAddr	3.6.0
NodeHostname	3.6.0
NodeName	3.6.0
权重	3.7.0

Slurm**prolog** 和 **epilog**

从 AWS ParallelCluster 版本 3.6.0 开始，Slurm 使用部署的配置 AWS ParallelCluster 包括`Prolog`和`Epilog`配置参数：

```
# PROLOG AND EPILOG
Prolog=/opt/slurm/etc/scripts/prolog.d/*
Epilog=/opt/slurm/etc/scripts/epilog.d/*
SchedulerParameters=nohold_on_prolog_fail
BatchStartTimeout=180
```

有关更多信息，请参阅 Prolog 和 Epilog 指南中的 Slurm 文档中）。

AWS ParallelCluster 包括以下 prolog 和 epilog 脚本：

- `90_plcuster_health_check_manager`（位于 `Prolog` 文件夹）
- `90_pcluster_noop`（位于 `Epilog` 文件夹）

> **Note**
>
> Prolog 和 Epilog 文件夹都必须至少包含一个文件。

您可以将自定义 prolog 或 epilog 脚本添加到相应的 Prolog 和 Epilog 文件夹中，从而使用自己的自定义脚本。

> **Warning**
>
> Slurm 按字母顺序反向运行文件夹中的所有脚本。

prolog 和 epilog 脚本的运行持续时间会影响运行作业所需的时间。当运行的 prolog 脚本数量较多或运行时间较长时，请更新 BatchStartTimeout 配置设置。默认值为 3 分钟。

如果您要使用自定义 prolog 和 epilog 脚本，请将这些脚本放置在相应的 Prolog 和 Epilog 文件夹中。我们建议您保留在每个自定义脚本之前运行的 90_plcuster_health_check_manager 脚本。有关更多信息，请参阅 Slurm 配置自定义。

集群容量大小和更新

集群的容量由集群可以扩展的计算节点数量来定义。计算节点由 AWS ParallelCluster 配置中的计算资源中定义的 Amazon EC2 实例提供支持(Scheduling/SlurmQueues/ ComputeResources)，并按照 1:1 映射到(Scheduling/SlurmQueues)的队列进行组织 Slurm 分区。

在计算资源中，可以配置集群中必须始终保持运行的计算节点（实例）的最小数量（MinCount），以及计算资源可扩展到的实例最大数量（MaxCount3 ）。

在创建集群时或群集更新时，为集群中定义的每个计算资源 (Scheduling/SlurmQueues/ ComputeResources) AWS ParallelCluster 启动中配置MinCount的任意数量的 Amazon EC2 实例。为覆盖集群中计算资源的最小节点数量而启动的实例称为静态节点。静态节点一旦启动，就会在集群中持续存在，除非发生特定事件或情况，否则系统不会终止它们。例如，此类事件包括失败 Slurm 或者 Amazon EC2 健康检查和变更 Slurm 节点状态变为 “耗尽” 或 “关闭”。

为应对集群MaxCount 负载增加而*1*按需启动的 Amazon EC2 实例被称为动态节点，范围为*‘MaxCount - MinCount’*（减去 MinCount））。 它们的性质是短暂的，启动它们是为了处理待处理的任务，如果它们Scheduling/SlurmSettings/ScaledownIdletime在集群配置中定义的一段时间内保持闲置状态（默认值：10 分钟），它们就会被终止。

静态节点和动态节点符合以下命名方案：

- 静态节点 `<Queue/Name>-st-<ComputeResource/Name>-<num>`，其中 `<num> = 1..ComputeResource/MinCount`
- 动态节点 `<Queue/Name>-dy-<ComputeResource/Name>-<num>`，其中 `<num> = 1..(ComputeResource/MaxCount - ComputeResource/MinCount)`

例如，给定以下 AWS ParallelCluster 配置：

```
Scheduling:
    Scheduler: Slurm
    SlurmQueues:
        - Name: queue1
            ComputeResources:
                - Name: c5xlarge
                    Instances:
                        - InstanceType: c5.xlarge
                        MinCount: 100
                        MaxCount: 150
```

将在中定义以下节点 Slurm

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

当计算资源的 `MinCount == MaxCount` 时，所有相应的计算节点都将是静态的，所有实例都将在集群创建/更新时启动并保持运行。例如：

```
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue1
      ComputeResources:
        - Name: c5xlarge
```

```
          Instances:
            - InstanceType: c5.xlarge
          MinCount: 100
          MaxCount: 100
```

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

集群容量更新

集群容量的更新包括添加或删除队列、计算资源或更改计算资源的 `MinCount/MaxCount`。从 AWS ParallelCluster 版本 3.9.0 开始，缩小队列大小需要在集群更新之前停止计算队列或将其QueueUpdateStrategy设置为 TERMINATION for。在以下情况下，无需停止计算队列或将其设置QueueUpdateStrategy为 “终止”：

- 将新队列添加到 Scheduling/`SlurmQueues`
- 向队列中添加新的计算资源 `Scheduling/SlurmQueues/ComputeResources`
- 增加计算资源的 `MaxCount`
- 计算资源的增加 MaxCount 和相同计算资源的增加量至少相等 MinCount

注意事项和限制

本节旨在概述在调整集群容量大小时应考虑的任何重要因素、约束或限制。

- 从`Scheduling/SlurmQueues`所有具有静态和动态名称`<Queue/Name>-*`的计算节点中移除队列时，将从 Slurm 配置和相应的 Amazon EC2 实例将被终止。
- `Scheduling/SlurmQueues/ComputeResources`从队列中移除计算资源时，所有名`<Queue/Name>-*-<ComputeResource/Name>-*`为静态和动态的计算节点都将从队列中移除 Slurm 配置和相应的 Amazon EC2 实例将被终止。

在更改计算资源的 `MinCount` 参数时，我们可以区分两种不同的情况，一种是 `MaxCount` 与 `MinCount` 相等（仅静态容量），另一种是 `MaxCount` 大于 `MinCount`（静态和动态混合容量）。

只有静态节点的容量变化

- 如果在增加MinCount（和MaxCount）时MinCount == MaxCount，将通过将静态节点的数量扩展到新的值来配置集群，MinCount<Queue/Name>-st-<ComputeResource/Name>-<new_MinCount>并且系统将继续尝试启动 Amazon EC2 实例以满足新的所需静态容量。
- 如果在减少MinCount（和MaxCount）数量 N 时MinCount == MaxCount，将通过移除最后 N 个静态节点来配置集群，<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount - N>...<old_MinCount>]并且系统将终止相应的 Amazon EC2 实例。
  - 初始状态 MinCount = MaxCount = 100
  - ```
    $ sinfo
    PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
    queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
    ```
  - 更新 MinCount 和 MaxCount: MinCount = MaxCount = 70 上的 -30
  - ```
    $ sinfo
    PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
    queue1*      up   infinite     70   idle queue1-st-c5xlarge-[1-70]
    ```

混合节点的容量变化

如果在增加MinCount数量 N（假设保持不变）时MinCount < MaxCount，MaxCount将通过将静态节点的数量扩展到新的值 MinCount (old_MinCount + N): 来配置集群，<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount + N>并且系统将继续尝试启动 Amazon EC2 实例以满足新的所需静态容量。此外，为了满足计算资源的MaxCount容量，通过删除最后 N 个动态节点来更新集群配置：<Queue/Name>-dy-<ComputeResource/Name>-[<MaxCount - old_MinCount - N>...<MaxCount - old_MinCount>]系统将终止相应的 Amazon EC2 实例。

- 初始状态：MinCount = 100; MaxCount = 150
- ```
  $ sinfo
  ```
  ```

```
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 将 +30 更新为 MinCount : MinCount = 130 (MaxCount = 150)
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     20  idle~ queue1-dy-c5xlarge-[1-20]
queue1*      up   infinite    130   idle queue1-st-c5xlarge-[1-130]
```

如果在增加MinCount且MaxCount数量相同 N 时MinCount < MaxCount，将通过将静态节点数扩展到新值 MinCount (old_MinCount + N): 来配置集群，<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount + N>并且系统将继续尝试启动 Amazon EC2 实例以满足新的所需静态容量。此外，动态节点的数量不会因新的

MaxCount 值。

- 初始状态：MinCount = 100; MaxCount = 150
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 将 +30 更新为 MinCount : MinCount = 130 (MaxCount = 180)
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     20  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    130   idle queue1-st-c5xlarge-[1-130]
```

如果MinCount < MaxCount在减少MinCount数量 N（假设保持不变）时，MaxCount将通过移除最后 N 个静态节点的静态节点来配置集群，系统将终止相应的 Amazon EC2 实例。<Queue/Name>-st-<ComputeResource/Name>-[<old_MinCount - N>...<old_MinCount>此外，为了满足计算资源的MaxCount容量，通过扩展动态节点的数量来更新集群配置以填补空白。MaxCount - new_MinCount: <Queue/Name>-dy-<ComputeResource/Name>-[1..<MazCount - new_MinCount>]在这种情况下，由于这些是动态节点，因此除非调度程序在新节点上有待处理的任务，否则不会启动新的 Amazon EC2 实例。

- 初始状态：MinCount = 100; MaxCount = 150
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```
- 更新 MinCount : MinCount = 70 (MaxCount = 120) 上的 -30
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     80  idle~ queue1-dy-c5xlarge-[1-80]
queue1*      up   infinite     70   idle queue1-st-c5xlarge-[1-70]
```

如果在减少MinCount且MaxCount数量相同的 N 时MinCount < MaxCount，将通过移除最后 N 个静态节点<Queue/Name>-st-<ComputeResource/Name>-<old_MinCount - N>...<oldMinCount>]来配置集群，系统将终止相应的 Amazon EC2 实例。

此外，动态节点的数量不会因新的 MaxCount 值而发生变化。

- 初始状态：MinCount = 100; MaxCount = 150
-
```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
```

```
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 更新 MinCount : MinCount = 70 (MaxCount = 120) 上的 -30
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     80  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite     70   idle queue1-st-c5xlarge-[1-70]
```

如果在减少MaxCount数量 N（假设保持不变）时MinCount < MaxCount，MinCount将通过移除最后 N 个动态节点来配置集群，<Queue/Name>-dy-<ComputeResource/Name>-<old_MaxCount - N...<oldMaxCount>]并且系统将终止相应的 Amazon EC2 实例（如果它们正在运行）。预计不会对静态节点产生任何影响。

- 初始状态：MinCount = 100; MaxCount = 150
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     50  idle~ queue1-dy-c5xlarge-[1-50]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

- 更新 MaxCount : MinCount = 100 (MaxCount = 120) 上的 -30
-

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     20  idle~ queue1-dy-c5xlarge-[1-20]
queue1*      up   infinite    100   idle queue1-st-c5xlarge-[1-100]
```

对作业的影响

在移除节点和 Amazon EC2 实例终止的所有情况下，除非没有其他节点满足任务要求，否则在已移除的节点上运行的 sbatch 作业都将重新排队。在最后一种情况下，任务失败，状态为 NODE_FAIL，并从队列中消失，必须手动重新提交。

如果您计划执行集群大小调整更新，可以防止作业在计划更新期间将被移除的节点上运行。这可以通过将要移除的节点设置为维护状态来实现。请注意，将节点设置为维护状态不会影响最终已在该节点上运行的作业。

假设在计划的集群大小调整更新中，您将移除节点 `qeueu-st-computeresource-[9-10]`。你可以创建一个 Slurm 使用以下命令进行预订

```
sudo -i scontrol create reservation ReservationName=maint_for_update user=root
 starttime=now duration=infinite flags=maint,ignore_jobs nodes=qeueu-st-
computeresource-[9-10]
```

这将创建一个 Slurm 在节点`maint_for_update`上命名的预留`qeueu-st-computeresource-[9-10]`。从创建预留开始，`qeueu-st-computeresource-[9-10]` 节点上就不能再运行任何作业。请注意，预留不会阻止作业最终在 `qeueu-st-computeresource-[9-10]` 节点上分配。

集群大小更新后，如果 Slurm 仅在调整大小更新期间移除的节点上设置了预留，维护预留将自动删除。相反，如果你创建了一个 Slurm 在集群调整大小更新后仍然存在的节点上进行预留，我们可能需要在执行调整大小更新后使用以下命令删除节点上的维护预留

```
sudo -i scontrol delete ReservationName=maint_for_update
```

欲了解更多详情，请访问 Slurm 预订，请在此处查看 schedMD 官方文档。

容量变更时的集群更新过程

调度器配置更改后，将在集群更新过程中执行以下步骤：

- 停下来 AWS ParallelCluster `clustermgtd` (`supervisorctl stop clustermgtd`)
- 生成已更新 Slurm 配置中的分区 AWS ParallelCluster 配置
- 重启 `slurmctld`（通过 Chef 服务配方完成）
- 检查 `slurmctld` 状态 (`systemctl is-active --quiet slurmctld.service`)
- Reload Slurm 配置 (`scontrol reconfigure`)

- 启动 `clustermgtd (supervisorctl start clustermgtd)`

将 AWS Batch (**awsbatch**) 调度器与 AWS ParallelCluster

AWS ParallelCluster 还支持 AWS Batch 调度程序。以下主题介绍如何使用 AWS Batch。有关的信息 AWS Batch，请参见AWS Batch。有关文档，请参阅 AWS Batch User Guide。

AWS ParallelCluster 适用于 CLI 命令 AWS Batch

使用`awsbatch`调度程序时，的 AWS ParallelCluster CLI 命令 AWS Batch 会自动安装在 AWS ParallelCluster 头节点中。CLI 使用 AWS Batch API 操作并允许执行以下操作：

- 提交和管理作业。
- 监控作业、队列和主机。
- 镜像传统调度器命令。

> **Important**
>
> AWS ParallelCluster 不支持 GPU 作业 AWS Batch。有关更多信息，请参阅 GPU 作业。

此 CLI 作为单独的软件包进行分发。有关更多信息，请参阅 调度器支持。

主题

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

awsbsub

向集群的作业队列提交作业。

```
awsbsub [-h] [-jn JOB_NAME] [-c CLUSTER] [-cf] [-w WORKING_DIR]
```

```
        [-pw PARENT_WORKING_DIR] [-if INPUT_FILE] [-p VCPUS] [-m MEMORY]
        [-e ENV] [-eb ENV_DENYLIST] [-r RETRY_ATTEMPTS] [-t TIMEOUT]
        [-n NODES] [-a ARRAY_SIZE] [-d DEPENDS_ON]
        [command] [arguments [arguments ...]]
```

> **Important**
>
> AWS ParallelCluster 不支持 GPU 作业 AWS Batch。有关更多信息，请参阅 GPU 作业。

定位参数

command

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

arguments

（可选）指定命令或命令文件的参数。

命名的参数

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

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

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

指定要使用的集群。

-cf, --command-file

指示命令是要传输到计算实例的文件。

默认值：False

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

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

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

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

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

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

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

指定要为容器保留的 v CPUs 数。与一起使用时`-nodes`，它会标识每个节点CPUs 的 v 数。

默认值：1

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

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

默认值：128

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

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

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

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

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

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

默认值：1

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

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

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

指定要为作业预留的节点数量。为此参数指定一个值，以启用多节点并行提交。

> **Note**
>
> 当 Scheduler/AwsBatchQueues/CapacityType 参数设置为 SPOT 时，不 支持多节点并行作业。此外，您的账户中必须有 AWSServiceRoleForEC2Spot 服务相关角色。您可以使用以下 AWS CLI 命令创建此角色：
>
> ```
> $ aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> 有关更多信息，请参阅《适用于 Linux 实例的 Amazon Elastic Compute Cloud 用户指南》中的竞价型实例请求的服务相关角色。

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

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

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

指定作业的依赖项的分号分隔的列表。一个作业可依赖于最多 20 个作业。您可以指定 SEQUENTIAL 类型依赖项，而不指定数组作业的作业 ID。顺序依赖项允许每个子数组作业按顺序完成，从索引 0 开始。您也可以使用数组作业的作业 ID 指定 N_TO_N 类型依赖项。N_TO_N 依赖项意味着此作业的每个子索引必须等待每个依赖项的相应子索引完成后才能开始。此参数的语法是 “jobid=*<string>*，type=*<string>*;...”。

awsbstat

显示集群的作业队列中提交的作业。

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

定位参数

job_ids

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

命名的参数

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

指示要使用的集群。

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

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

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

-e, --expand-children

展开具有子作业（数组和多节点并行）的作业。

默认值：False

-d, --details

显示作业详细信息。

默认值：False

awsbout

显示给定作业的输出。

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

定位参数

job_id

指定作业 ID。

命名的参数

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

指示要使用的集群。

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

获取任务输出的第一*HEAD*行。

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

获取作业输出的最后几个 <tail> 行。

-s, --stream

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

默认值：False

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

设置流式传输时段。

默认：5

awsbkill

取消或终止集群中提交的作业。

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

定位参数

job_ids

指定要取消或终止的 IDs 以空格分隔的作业列表。

命名的参数

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

指示要使用的集群的名称。

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

指示要附加到作业的消息，并说明取消作业的原因。

默认：”Terminated by the user”

awsbqueues

显示与集群关联的作业队列。

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

定位参数

job_queues

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

命名的参数

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

指定要使用的集群的名称。

-d, --details

指明是否显示队列的详细信息。

默认值：False

awsbhosts

显示属于集群的计算环境的主机。

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

定位参数

instance_ids

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

命名的参数

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

指定要使用的集群的名称。

-d, --details

指示是否显示主机的详细信息。

默认值：False

共享存储

AWS ParallelCluster 支持使用 Amazon EBS、FSx ONTAP 和 FSx OpenZFS 共享存储卷、Amazon EFS 和 FSx Lustre 共享存储文件系统或文件缓存。我们建议您遵循 AWS Well-Architected Framework 可靠性支柱指南，备份您的卷和文件系统。

选择符合 HPC 应用程序 I/O 要求的存储系统。您可以根据具体用例优化每个文件系统。有关更多信息，请参阅存储选项概述。

Amazon EBS 卷附加到头节点，并通过 NFS 与计算节点共享。此选项可能具有成本效益，但随着存储需求的扩展，性能将取决于头节点资源。随着添加到集群中的计算节点越来越多以及吞吐量需求的增加，这可能会成为瓶颈。

Amazon EFS 文件系统会随着存储需求的变化而扩展。您可以为各种用例配置这些文件系统。可以使用 Amazon EFS 文件系统在集群上运行并行化且对延迟敏感的应用程序。

FSx f@@ or Lustre 文件系统可以以高达每秒数百 GB 的吞吐量、数百万的 IOPS 和亚毫秒的延迟处理海量数据集。 FSx 用于要求苛刻的高性能计算环境的 Lustre 文件系统。

在中`SharedStorage` 部分，您可以定义外部存储或 AWS ParallelCluster 托管存储：

- 外部存储是指您管理的现有卷或文件系统。 AWS ParallelCluster 不会创建或删除此存储空间。
- 托管存储是指 AWS ParallelCluster 创建并可以删除的卷或文件系统。

外部存储

您可以配置 AWS ParallelCluster 为在创建或更新集群时将外部存储连接到集群。同样，您可以将其配置为在删除或更新集群时将外部存储与集群分离。您的数据将被保留，您可以在集群生命周期之外将其用于长期永久性共享存储。

> **Note**
>
> 3.8 AWS ParallelCluster 之前的版本不允许将外部管理的文件系统挂载到上。/home从 3.8 版开始， AWS ParallelCluster 允许使用 /home 作为外部托管文件系统的挂载点。/home通过将下方的MountDir参数指定/home为值，可以将外部托管的文件系统挂载到中。SharedStorage 部分
> Amazon File Cache 不适合用作系统 /home 目录，因此目前不支持挂载 /home。
> 在配置选项SharedStorage 部分下指定/home目录时，将覆盖SharedStorageType配置选项，这意味着SharedStorage 部分将改用下面的设置。
> 将外部文件系统挂载到/home目录时，会将头节点的/home内容 AWS ParallelCluster 复制到外部文件系统，而不会覆盖外部存储器上的现有文件。这包括为默认用户传输集群的 SSH 密钥（如果外部文件系统中没有）。有关更多信息，请参阅 AWS ParallelCluster 共享存储注意事项。

AWS ParallelCluster 托管存储

AWS ParallelCluster 默认情况下，在配置中，托管存储取决于集群的生命周期。默认情况下，SharedStorage DeletionPolicy 配置参数设置为 Delete。

默认情况下，如果满足以下条件之一，则会删除 AWS ParallelCluster 托管文件系统或卷及其数据。

- 您删除集群。
- 您更改托管共享存储配置 Name。
- 您从配置中删除托管共享存储。

将 DeletionPolicy 设置为 Retain 可保留您的托管共享文件系统或卷及数据。我们建议您定期备份数据，以避免数据丢失。您可以使用 AWS Backup 集中管理所有存储选项的备份。

您可以使用配置设置删除生命周期依赖关系。有关更多信息，请参阅 将 AWS ParallelCluster 托管存储转换为外部存储。

有关共享存储配额的信息，请参阅共享存储的配额。

有关共享存储和切换到新 AWS ParallelCluster 版本的更多信息，请参阅最佳实践：将集群移至新的 AWS ParallelCluster 次要版本或补丁版本。

您可以配置 AWS ParallelCluster 为在创建或更新集群时将外部存储连接到集群。同样，您可以将其配置为在删除或更新集群时将外部存储与集群分离。您的数据将被保留，您可以将其用于依赖于集群生命周期的长期永久性共享存储解决方案。

默认情况下，托管存储依赖于集群的生命周期。您可以使用将 AWS ParallelCluster 托管存储转换为外部存储中所述的配置设置删除这种依赖关系。

通过特定的设置，您可以针对自己的用例优化支持的每种存储解决方案。

有关共享存储配额，请参阅共享存储的配额。

有关共享存储和切换到新 AWS ParallelCluster 版本的更多信息，请参阅最佳实践：将集群移至新的 AWS ParallelCluster 次要版本或补丁版本。

以下主题介绍如何为 AWS ParallelCluster 支持的每种存储服务配置共享存储。

主题

- Amazon Elastic Block Store
- Amazon Elastic File System
- 亚马逊 f FSx or Lustre
- 针对 ONTAP、 FSx OpenZFS 和文件缓存共享存储进行配置 FSx
- 在中使用共享存储 AWS ParallelCluster
- 共享存储的配额

Amazon Elastic Block Store

要将现有的外部 Amazon EBS 卷用于独立于集群生命周期的长期永久性存储，请指定 `EbsSettings`/`VolumeId`。

如果不指定 `VolumeId`，则在创建集群时， AWS ParallelCluster 会默认根据 `EbsSettings` 创建托管 EBS 卷。在删除集群或从集群配置中删除卷时， AWS ParallelCluster 也会删除卷和数据。

对于 AWS ParallelCluster 托管的 EBS 卷，当集群被删除或从集群配置`Snapshot`中移除卷时，您可以使用 `EbsSettings`/`DeletionPolicy` AWS ParallelCluster 来`Delete`指示`Retain`、或该卷。默认情况下，将 `DeletionPolicy` 设置为 `Delete`。

> **Warning**
>
> 对于 AWS ParallelCluster 托管共享存储，`Delete`默认设置`DeletionPolicy`为。

这意味着如果满足以下条件之一，则会删除托管卷及其数据：

- 您删除集群。
- 您更改托管共享存储配置 `SharedStorage`/`Name`。
- 您从配置中删除托管共享存储。

我们建议您定期使用快照备份共享存储，以避免数据丢失。有关 Amazon EBS 快照的更多信息，请参阅 Amazon Elastic Compute Cloud 用户指南（适用于 Linux 实例）中的 Amazon EBS 快照。要了解如何管理数据备份 AWS 服务，请参阅《AWS Backup 开发人员指南》中的 AWS Backup。

Amazon Elastic File System

要将现有的外部 Amazon EFS 文件系统用于集群生命周期之外的长期永久存储，请指定 `EfsSettings`/`FileSystemId`，默认情况下，从 AWS ParallelCluster 创建集群`EfsSettings`时开始创建托管 Amazon EFS 文件系统。 AWS ParallelCluster 还会在删除群集或从群集配置中删除文件系统时删除文件系统和数据。

对于 AWS ParallelCluster 托管的 Amazon EFS 文件系统，您可以使用 `EfsSettings`/`DeletionPolicy`指示 AWS ParallelCluster 何时删除集群或何时将文件系统从集群配置中删除。`Delete` `Retain`默认情况下，将 `DeletionPolicy` 设置为 `Delete`。

> **Warning**
>
> 对于 AWS ParallelCluster 托管共享存储，`Delete`默认设置`DeletionPolicy`为。
> 这意味着如果满足以下条件之一，则会删除托管文件系统及其数据：
>
> - 您删除集群。
> - 您更改托管共享存储配置 `SharedStorage`/`Name`。
> - 您从配置中删除托管共享存储。
>
> 我们建议您定期备份共享存储，以避免数据丢失。有关如何备份单独 Amazon EFS 卷的更多信息，请参阅 Amazon Elastic File System User Guide 中的 Backing up your Amazon EFS file systems。要了解如何管理数据备份 AWS 服务，请参阅《AWS Backup 开发人员指南》中的 AWS Backup。

亚马逊 f FSx or Lustre

要将 Lustre 文件系统的现有外部 FSx 文件系统用于集群生命周期之外的长期永久存储，请指定 `FsxLustreSettings`/`FileSystemId`。

如果未指定 `FsxLustreSettings`/`FileSystemId`，则默认情况下， AWS ParallelCluster 会从创建集群`FsxLustreSettings`时开始创建 FSx 适用于 Lustre 的托管文件系统。 AWS ParallelCluster 还会在删除群集或从群集配置中删除文件系统时删除文件系统和数据。

FSx 对于 AWS ParallelCluster 托管的 Lustre 文件系统，当集群被删除`Delete`或`Retain`从群集配置中删除文件系统时，您可以使用 `FsxLustreSettings`/`DeletionPolicy`来指示 AWS ParallelCluster 或文件系统。默认情况下，将 `DeletionPolicy` 设置为 `Delete`。

> **Warning**
>
> 对于 AWS ParallelCluster 托管共享存储，`Delete`默认设置`DeletionPolicy`为。这意味着如果满足以下条件之一，则会删除托管文件系统及其数据：
>
> - 您删除集群。
> - 您更改托管共享存储配置 `SharedStorage`/`Name`。
> - 您从配置中删除托管共享存储。
>
> 我们建议您定期备份共享存储，以避免数据丢失。您可以在集群中使用 `SharedStorage`/`FsxLustreSettings`/`AutomaticBackupRetentionDays` 和 `DailyAutomaticBackupStartTime` 来定义备份。要了解如何管理数据备份 AWS 服务，请参阅《AWS Backup 开发人员指南》中的 AWS Backup。

针对 ONTAP、 FSx OpenZFS 和文件缓存共享存储进行配置 FSx

对 FSx 于 ONTAP、 FSx OpenZFS 和文件缓存，您可以使用 `FsxOntapSettingsFsxOpenZfsSettings`/`VolumeIdVolumeId`、/和 `FileCacheSettings`/`FileCacheId`来指定为集群安装外部现有卷或文件缓存。

AWS ParallelCluster ONTAP、 FSx OpenZFS 和文件缓存不支持托管共享存储。 FSx

在中使用共享存储 AWS ParallelCluster

在以下各节中，您将学习如何使用 AWS ParallelCluster 和共享存储，包括共享存储注意事项以及如何将托管存储转换为外部存储。

主题

- AWS ParallelCluster 共享存储注意事项
- 将 AWS ParallelCluster 托管存储转换为外部存储

AWS ParallelCluster 共享存储注意事项

在 AWS ParallelCluster中使用共享存储时，请注意以下几点。

- 使用 AWS Backup 或其他方法备份文件系统数据，以管理所有存储系统的备份。
- 要添加共享存储，可在配置文件中添加共享存储部分，然后创建或更新集群。
- 要删除共享存储，可从配置文件中删除共享存储部分并更新集群。
- 要将现有的 AWS ParallelCluster 托管共享存储替换为新的托管存储，请更改 `SharedStorage`/的值`Name`并更新群集。

> **Warning**
>
> 默认情况下，当您使用新`Name`参数执行集群更新时，会删除现有的 AWS ParallelCluster 托管存储空间和数据。如果您需要更改 `Name` 并保留现有的托管共享存储数据，请确保在更新集群之前将 `DeletionPolicy` 设置为 `Retain` 或备份数据。

- 如果您没有备份 AWS ParallelCluster 托管存储数据`Delete`，`DeletionPolicy`则当您的集群被删除或从集群配置中移除托管存储并更新集群时，您的数据就会被删除。
- 如果您没有备份 AWS ParallelCluster 托管存储数据`Retain`，`DeletionPolicy`则在删除集群之前您的文件系统将被分离，并且可以作为外部文件系统重新连接到另一个群集。将会保留您的数据。
- 如果 AWS ParallelCluster 托管存储已从群集配置中移除`Retain`，`DeletionPolicy`则可以将其作为外部文件系统重新连接到群集，同时保留您的群集数据。
- 从 AWS ParallelCluster 版本 3.4.0 开始，您可以通过配置`SharedStorage`/`EfsSettings`/`EncryptionInTransit`和`IamAuthorization`设置来增强 Amazon EFS 文件系统挂载的安全性。
- 将外部文件系统挂载到 /home 目录时， AWS ParallelCluster 复制 head nod s /home directory to the external filesystem. It copies existing data in the /home directory without overwriting existing

files or directories on the external storage. This includes the cluster's SSH key for the default user in case it does not already exist on the external filesystem. Consequently all other clusters that mount the same external filesystem to their respective /home e 目录的内容也将具有与集群默认用户相同的 SSH 密钥。

- 在将同一个外部文件系统挂载到群集的 /home 目录的多集群环境中，当第一个群集将外部文件系统挂载到 /home 时，仅生成一次授予对在头节点上创建的计算节点的访问权限的 SSH 密钥。 AWS ParallelCluster所有其它集群都使用相同的 SSH 密钥。因此，任何拥有这些共享集群默认用户的 SSH 密钥的人都可以访问任何集群。所有计算节点都允许使用最初生成的密钥进行连接。

将 AWS ParallelCluster 托管存储转换为外部存储

了解如何将 AWS ParallelCluster 托管存储转换为外部存储。

操作步骤基于下面的示例配置文件代码段。

```
...
  - MountDir: /fsx
    Name: fsx
    StorageType: FsxLustre
    FsxLustreSettings:
      StorageCapacity: 1200
      DeletionPolicy: Delete
...
```

将 AWS ParallelCluster 托管存储转换为外部存储

1. 在集群配置文件中将 `DeletionPolicy` 设置为 `Retain`。

   ```
   ...
      - MountDir: /fsx
        Name: fsx
        StorageType: FsxLustre
        FsxLustreSettings:
          StorageCapacity: 1200
          DeletionPolicy: Retain
   ...
   ```

2. 要设置 `DeletionPolicy` 更改，请运行以下命令。

   ```
   pcluster update-cluster -n cluster-name -c cluster-config.yaml
   ```

3. 从集群配置文件中删除 SharedStorage 部分。

```
...
...
```

4. 要将托管 SharedStorage 更改为外部 SharedStorage 并将其与集群分离，请运行以下命令。

```
pcluster update-cluster -n cluster-name -c cluster-config.yaml
```

5. 您的共享存储现在变为了外部共享存储，并且已与集群分离。
6. 要将外部文件系统附加到原始集群或其他集群，请按照以下步骤操作。

   a. 获取 L FSx ustre 文件系统 ID。

      i. 要使用，请 AWS CLI 运行以下命令并找到名称包含原始群集名称的文件系统并记下文件系统 ID。

      ```
      aws fsx describe-file-systems
      ```

      ii. 要使用 AWS Management Console，请登录并导航到https://console.aws.amazon.com/fsx/。在文件系统列表中，找到名称中包含原始集群名称的文件系统，并记下文件系统 ID。

   b. 更新文件系统安全组规则，以提供访问该文件系统和集群子网以及从该文件系统和集群子网进行访问的权限。您可以在 Amazon FSx 控制台中找到文件系统安全组的名称和 ID。

      向文件系统安全组中添加规则，允许针对头节点和计算节点 IP CIDR 范围或前缀的入站和出站 TCP 流量。为入站和出站 TCP 流量指定 TCP 端口 988、1021、1022 和 1023。

      有关更多信息，请参阅版本 2 AWS Command Line Interface 用户指南 EC2 中的 SharedStorageFsxLustreSettings//FileSystemId和为 Amazon 创建、配置和删除安全组。

   c. 将 SharedStorage 部分添加到集群配置中。

      ```
      ...
         - MountDir: /fsx
           Name: fsx-external
           StorageType: FsxLustre
           FsxLustreSettings:
             FileSystemId: fs-02e5b4b4abd62d51c
      ...
      ```

d. 要向集群中添加外部共享，请运行以下命令。

```
pcluster update-cluster -n cluster-name -c cluster-config.yaml
```

共享存储的配额

根据下表中所列的配额，可以将集群 `SharedStorage` 配置为挂载现有的共享文件存储和创建新的共享文件存储。

每个集群的挂载文件存储配额

文件共享存储类型	AWS ParallelCluster 托管存储	外部存储	净总配额
Amazon EBS	5	5	5
RAID	1	0	1
Amazon EFS	1	20	21
亚马逊 FSx †	1 代表 L FSx ustre	20	21

> **Note**
> 此配额表已在 3.2.0 AWS ParallelCluster 版本中添加。

† AWS ParallelCluster 仅支持安装 FSx 适用于 NetApp ONTAP 的现有亚马逊、适用 FSx 于 OpenZFS 的亚马逊和文件缓存系统。它不支持 FSx 为 ONTAP、 FSx OpenZFS 和文件缓存系统创建新的内容。

> **Note**
> 如果您 AWS Batch 用作调度程序， FSx 则 Lustre 仅在群集头节点上可用。
> 文件缓存不支持 AWS Batch 调度程序。

AWS ParallelCluster 资源和标记

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

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

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

> **Warning**
>
> 所有 AWS ParallelCluster 标签都是必不可少的，不得对其进行修改，以免影响系统功能。因此，您无法修改 AWS 系统标签。

以下是 AWS ParallelCluster 资源的 AWS 系统标签示例。

```
"aws:cloudformation:stack-name"="clustername"
```

以下是应用于资源的 AWS ParallelCluster 标签的示例。

```
"parallelcluster:cluster-name"="clustername"
```

您可以在的 Amazon EC2 部分查看这些标签 AWS Management Console。

查看标签

完成以下步骤，即可在的 Amazon EC2 部分查看标签 AWS Management Console。

查看标签

1. 浏览亚马逊 EC2 控制台，网址为https://console.aws.amazon.com/ec2/。
2. 要查看所有集群标签，请在导航窗格中选择标签。
3. 要按实例查看集群标签，请在导航窗格中选择实例。
4. 选择一个集群实例。
5. 在实例详细信息中选择管理标签选项卡并查看标签。

6. 在实例详细信息中选择存储选项卡。
7. 选择卷 ID。
8. 在卷中，选择该卷。
9. 在卷详细信息中选择标签选项卡并查看标签。

AWS ParallelCluster 头节点实例标签

键	标签值
parallelcluster:cluster-name	*clustername*
Name	HeadNode
aws:ec2launchtemplate:id	*lt-1234567890abcdef0*
aws:ec2launchtemplate:version	*1*
parallelcluster:node-type	HeadNode
aws:cloudformation:stack-name	*clustername*
aws:cloudformation:logical-id	HeadNode
aws:cloudformation:stack-id	arn:aws:cloudformation: *region-id* :*ACCOUNTID* :stack/*clustername* /*1234abcd-12ab-12ab-12ab-1234567890abcdef0*
parallelcluster:version	*3.13.2*

AWS ParallelCluster 头节点根卷标签

标签密钥	标签值
parallelcluster:cluster-name	*clustername*
parallelcluster:node-type	HeadNode
parallelcluster:version	*3.13.2*

AWS ParallelCluster 计算节点实例标签

键	标签值
parallelcluster:cluster-name	*clustername*
parallelcluster:compute-resource-name	*compute-resource-name*
aws:ec2launchtemplate:id	*lt-1234567890abcdef0*
aws:ec2launchtemplate:version	*1*
parallelcluster:node-type	Compute
parallelcluster:queue-name	*queue-name*
parallelcluster:version	*3.13.2*

AWS ParallelCluster 计算节点根卷标签

标签密钥	标签值
parallelcluster:cluster-name	*clustername*
parallelcluster:compute-resource-name	*compute-resource-name*
parallelcluster:node-type	Compute
parallelcluster:queue-name	*queue-name*
parallelcluster:version	*3.13.2*

PCUI 标签

标签密钥	标签值
parallelcluster-ui	true

监控 AWS ParallelCluster 和日志

监控是维护和其他 AWS 解决方案的可靠性、可用性和性能的重要组成部分。 AWS ParallelCluster AWS 提供以下监控工具 AWS ParallelCluster，供您监视、报告问题并在适当时自动采取措施：

- Amazon 会实时 CloudWatch监控您的 AWS 资源和您运行 AWS 的应用程序。您可以收集和跟踪指标，创建自定义的控制平面，以及设置警报以在指定的指标达到您指定的阈值时通知您或采取措施。例如，您可以 CloudWatch 跟踪您的 Amazon EC2 实例的 CPU 使用率或其他指标，并在需要时自动启动新实例。有关更多信息，请参阅 Amazon CloudWatch 用户指南。
- Amazon Lo CloudWatch gs 使您能够监控、存储和访问来自亚马逊 EC2 实例和其他来源的日志文件。 CloudTrail CloudWatch 日志可以监视日志文件中的信息，并在达到特定阈值时通知您。您还可以在高持久性存储中检索您的日志数据。有关更多信息，请参阅 Amazon CloudWatch 日志用户指南。
- AWS CloudTrail 捕获由您的 AWS 账户 或代表该账户发出的 API 调用和相关事件，并将日志文件传输到您指定的 Amazon S3 桶。您可以标识哪些用户和账户调用了 AWS、发出调用的源 IP 地址以及调用的发生时间。有关更多信息，请参阅 AWS CloudTrail 《用户指南》。
- Amazon EventBridge 是一项无服务器事件总线服务，可以轻松地将您的应用程序与来自各种来源的数据连接起来。 EventBridge 提供来自您自己的应用程序、 Software-as-a-Service (SaaS) 应用程序和 AWS 服务的实时数据流，并将这些数据路由到 Lambda 等目标。这使您能够监控服务中发生的事件，并构建事件驱动的架构。有关更多信息，请参阅 Amazon EventBridge 用户指南。

主题

- 与 Amazon CloudWatch 日志集成
- 亚马逊 CloudWatch 控制面板
- Amazon 针对集群指标的 CloudWatch 警报
- AWS ParallelCluster 配置的日志轮换
- pcluster CLI 日志
- Amazon EC2 控制台输出日志
- 检索 PCUI 和 AWS ParallelCluster 运行时日志
- 检索和保留日志

与 Amazon CloudWatch 日志集成

有关 CloudWatch 日志的更多信息，请参阅 Amazon CloudWatch 日志用户指南。要配置 CloudWatch 日志集成，请参阅一`Monitoring`节。要了解如何使用将自定义日志附加到 CloudWatch 配置中`append-config`，请参阅 Amazon CloudWatch 用户指南中的多个 CloudWatch 代理配置文件。

Amazon CloudWatch 日志集群日志

将为每个集群创建一个名为 `/aws/parallelcluster/`*`cluster-name-<timestamp>`* 的日志组（例如 `/aws/parallelcluster/testCluster-202202050215`）。每个节点上的每个日志（如果路径包含 `*`，则为一组日志）都有一个名为 *`{hostname}.{instance_id}.{logIdentifier}`* 的日志流。（例如 `ip-172-31-10-46.i-02587cf29cc3048f3.nodewatcher`。）日志数据 CloudWatch 由代理发送到，该CloudWatch 代理像`root`在所有集群实例上一样运行。

Amazon CloudWatch 控制面板是在创建集群时创建的。此仪表板使您能够查看存储在日志中的 CloudWatch 日志。有关更多信息，请参阅 亚马逊 CloudWatch 控制面板。

此列表包含平台*`logIdentifier`*、调度器和节点可用的日志流的和路径。

适用于平台、调度器和节点的日志流

平台	调度器	Nodes	日志流
amazon redhat ubuntu	awsbatc slurm	HeadNc	dcv-authenticator：`/var/log/parallelcluster/pcluster_dcv_authenticator.log` dcv-ext-authenticator: `/var/log/parallelcluster/pcluster_dcv_connect.log` dcv-agent：`/var/log/dcv/agent.*.log` dcv-xsession：`/var/log/dcv/dcv-xsession.*.log` dcv-server：`/var/log/dcv/server.log` dcv-session-launcher: `/var/log/dcv/sessionlauncher.log` Xdcv：`/var/log/dcv/Xdcv.*.log` cfn-init：`/var/log/cfn-init.log`

平台	调度器	Nodes	日志流
			chef-client：`/var/log/chef-client.log`
amazon redhat ubuntu	awsbatc slurm	Comput eet HeadNc	cloud-init：`/var/log/cloud-init.log` supervisord：`/var/log/supervisord.log`
amazon redhat ubuntu	slurm	Comput eet	cloud-init-output: `/var/log/cloud-init-output.log` computemgtd：`/var/log/parallelcluster/computemgtd` slurmd：`/var/log/slurmd.log` slurm_prolog_epilog：`/var/log/parallelcluster/slurm_prolog_epilog.log`

平台	调度器	Nodes	日志流
amazon redhat ubuntu	slurm	HeadNo	sssd：`/var/log/sssd/sssd.log` sssd_domain_default：`/var/log/sssd/sssd_default.log` pam_ssh_key_generator：`/var/log/parallelcluster/pam_ssh_key_generator.log` clusterstatusmgtd：`/var/log/parallelcluster/clusterstatusmgtd` clustermgtd：`/var/log/parallelcluster/clustermgtd` compute_console_output：`/var/log/parallelcluster/compute_console_output` slurm_resume：`/var/log/parallelcluster/slurm_resume.log` slurm_suspend：`/var/log/parallelcluster/slurm_suspend.log` slurmctld：`/var/log/slurmctld.log` slurm_fleet_status_manager：`/var/log/parallelcluster/slurm_fleet_status_manager.log`
amazon redhat	awsbato slurm	Comput eet HeadNo	system-messages：`/var/log/messages`
ubuntu	awsbato slurm	Comput eet HeadNo	syslog：`/var/log/syslog`

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

Amazon CloudWatch Logs 生成镜像日志

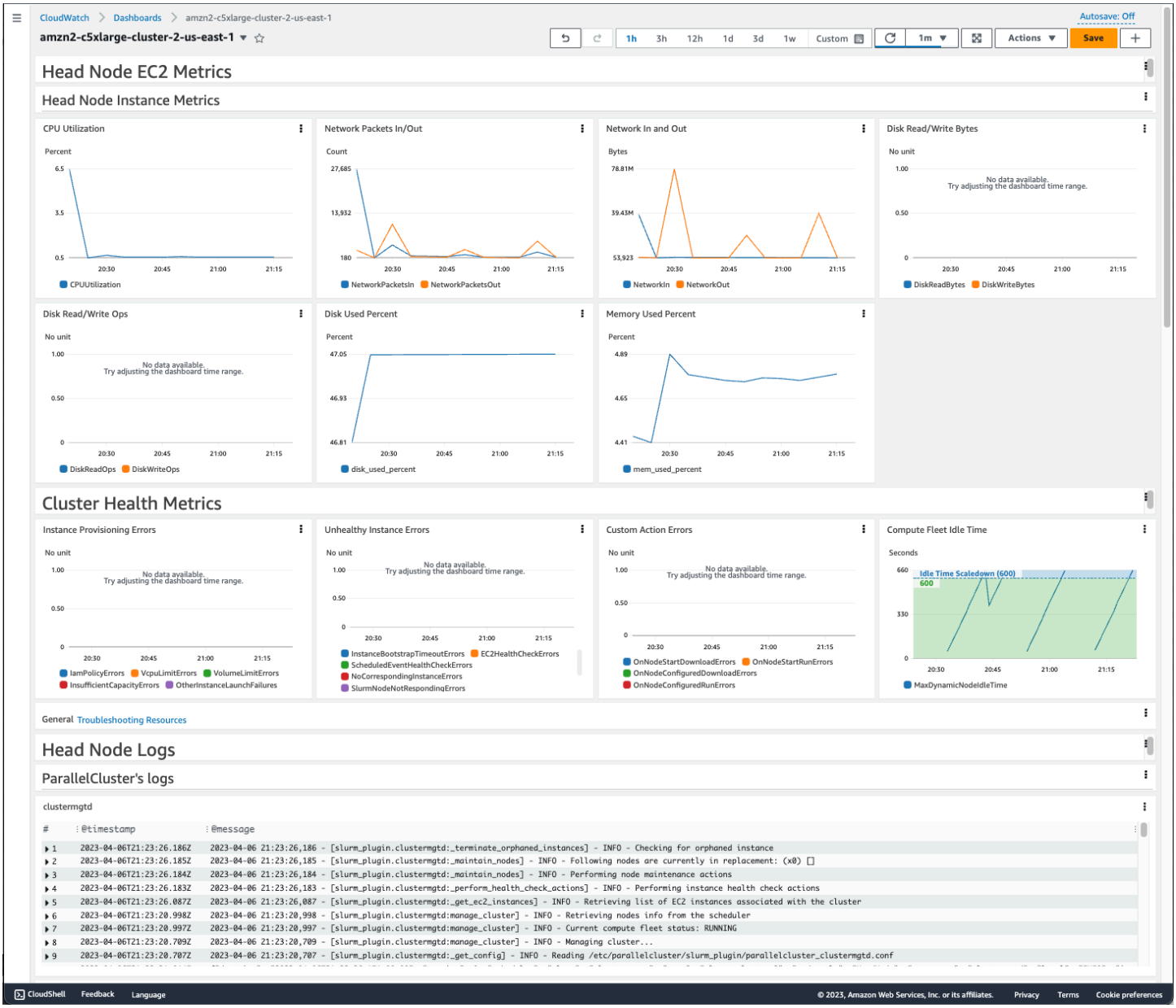
将为每个自定义构建映像创建名为 /aws/imagebuilder/ParallelClusterImage-*<image-id>* 的日志组。名为 *{pcluster-version}* /1 的唯一日志流包含构建映像过程的输出。

您可以使用 pcluster 映像命令访问这些日志。有关更多信息，请参阅 AWS ParallelCluster AMI 自定义。

亚马逊 CloudWatch 控制面板

Amazon CloudWatch 控制面板是在创建集群时创建的。这样可以更轻松地监控集群中的节点和查看 Amazon Logs 中存储的 CloudWatch 日志。仪表板的名称是*ClusterName*-*Region*。*ClusterName*是您的集群的名称，*Region*也是集群所在 AWS 区域 的名称。您可以在控制台中访问控制面板，也可以通过打开 https://console.aws.amazon.com/cloudwatch/home?region=*Region*#dashboards:name=*ClusterName*-*Region* 来访问控制面板。

下图显示了集群的示例 CloudWatch 仪表板。



头节点实例指标

控制面板的第一部分显示了头节点 Amazon EC2 指标的图表。

如果您的集群具有共享存储，则下一部分将显示共享存储指标。

集群运行状况指标

如果您的集群使用 Slurm 在调度方面，集群运行状况指标图表显示集群计算节点的实时错误。有关更多信息，请参阅 集群运行状况指标故障排除。从 3.6.0 AWS ParallelCluster 版开始，集群运行状况指标已添加到控制面板中。

头节点日志

最后一部分列出了按 AWS ParallelCluster日志、调度程序日志、Amazon DCV 集成日志和系统日志分组的主节点日志。

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

如果您不想创建亚马逊 CloudWatch 控制面板，可以通过将 `Monitoring`//`DashboardsCloudWatch`/设置`Enabled`为将其关闭`false`。

> **Note**
>
> 如果您禁用创建亚马逊 CloudWatch 控制面板，则还会禁用集群的亚马逊 CloudWatch `disk_used_percent`和`memory_used_percent`警报。有关更多信息，请参阅 Amazon 针对集群指标的 CloudWatch 警报。
>
> 从 3.6 AWS ParallelCluster 版开始添加`disk_used_percent`和`memory_used_percent`警报。

Amazon 针对集群指标的 CloudWatch 警报

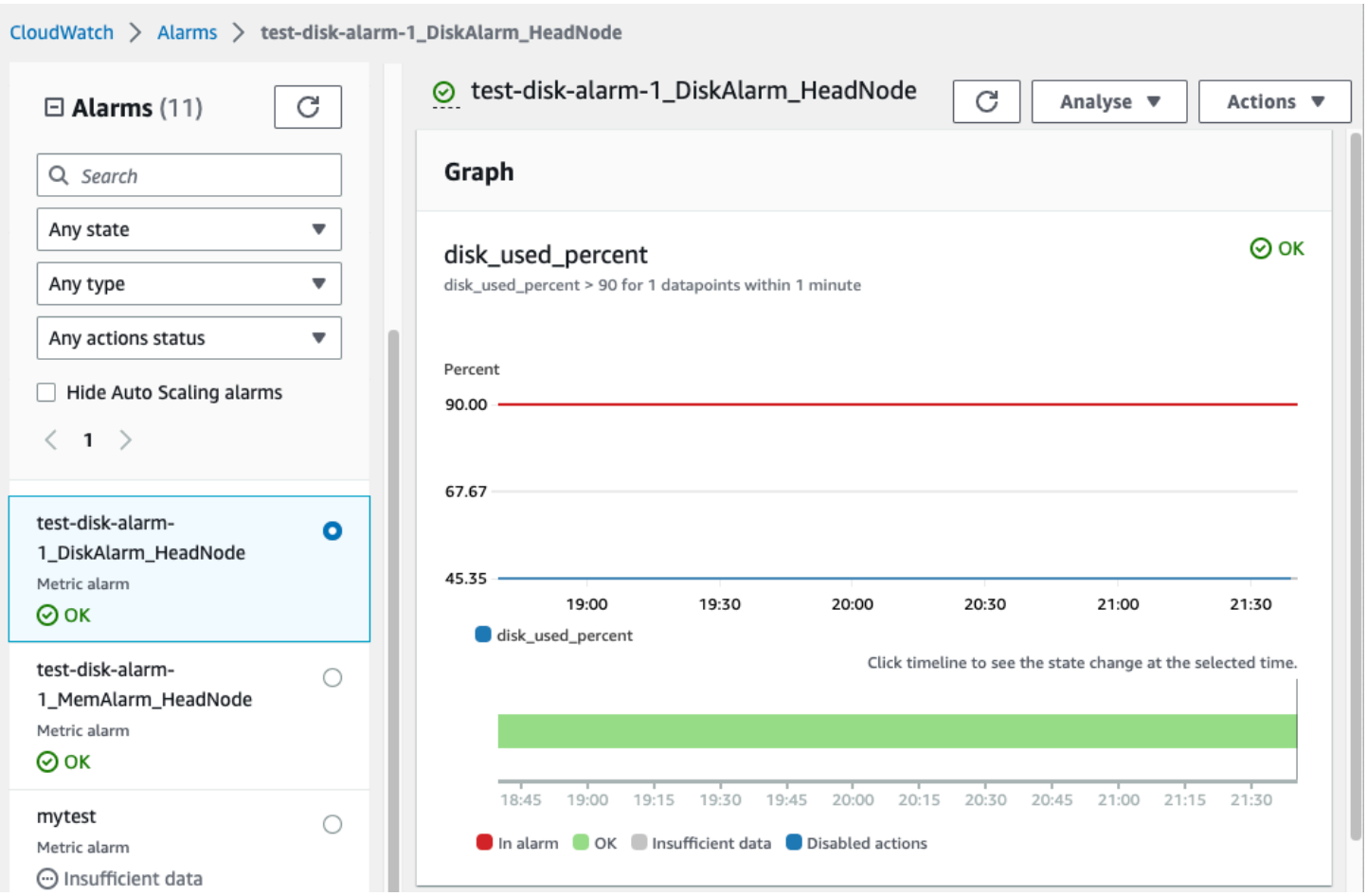
从 3.6 AWS ParallelCluster 版开始，您可以为集群配置用于监控头节点的 Amazon CloudWatch 警报。一个警报监控根卷 `disk_used_percent`，另一个警报监控 `mem_used_percent` 指标。有关更多信息，请参阅 Amazon CloudWatch 用户指南中的 CloudWatch 代理收集的指标。

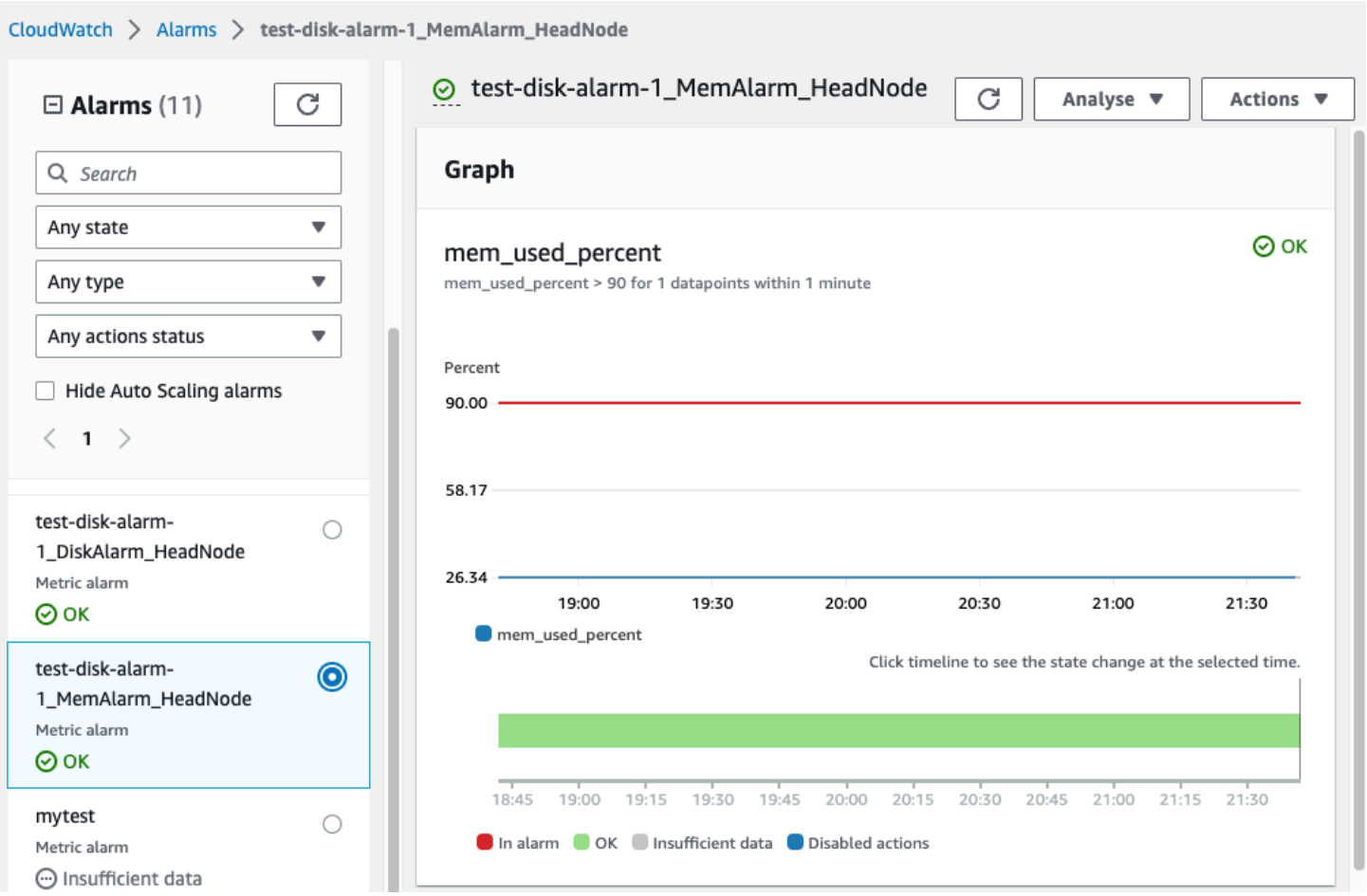
警报按以下方式命名：

- *`cluster-name`*`_DiskAlarm_HeadNode`
- *`cluster-name`*`_MemAlarm_HeadNode`

*`cluster-name`*是您的集群的名称。

在导航窗格中选择 AI CloudWatch arms，即可访问控制台中的警报。下图显示了集群的磁盘使用率警报和内存使用率警报。


CloudWatch > Alarms > test-disk-alarm-1_DiskAlarm_HeadNode
Alarms (11)
Search
Any state
Any type
Any actions status
Hide Auto Scaling alarms
1
test-disk-alarm-1_DiskAlarm_HeadNode
Metric alarm
OK
test-disk-alarm-1_MemAlarm_HeadNode
Metric alarm
OK
mytest
Metric alarm
Insufficient data
test-disk-alarm-1_DiskAlarm_HeadNode
Analyse
Actions
Graph
disk_used_percent
OK
disk_used_percent > 90 for 1 datapoints within 1 minute
Percent
90.00
67.67
45.35
19:00
19:30
20:00
20:30
21:00
21:30
disk_used_percent
Click timeline to see the state change at the selected time.
18:45
19:00
19:15
19:30
19:45
20:00
20:15
20:30
20:45
21:00
21:15
21:30
In alarm
OK
Insufficient data
Disabled actions



当 1 个数据点的磁盘使用率百分比在 1 分钟时间段内超过 90% 时，磁盘使用率警报就会处于 ALARM 状态。

当 1 个数据点的内存使用率百分比在 1 分钟时间段内超过 90% 时，内存使用率警报就会处于 ALARM 状态。

> **Note**
>
> AWS ParallelCluster 默认情况下不配置警报操作。有关如何设置警报操作（例如发送通知）的信息，请参阅警报操作。有关亚马逊 CloudWatch 警报的更多信息，请参阅亚马逊 CloudWatch 用户指南中的使用亚马逊 CloudWatch 警报。

如果您不想创建这些 Amazon CloudWatch 警报，请在集群配置`false`中将`Monitoring`/`DashboardsCloudWatch`/设置为 `Enabled`，将其停用。这也将禁用Amazon CloudWatch 控制面板的创建。有关更多信息，请参阅 亚马逊 CloudWatch 控制面板。

> **Note**
>
> 如果您停用了 Amazon CloudWatch 控制面板的创建，则还会停用集群的 Amazon CloudWatch `disk_used_percent` 和`memory_used_percent`警报。

AWS ParallelCluster 配置的日志轮换

AWS ParallelCluster 日志轮换配置位于`/etc/logrotate.d/parallelcluster_*_log_rotation`文件中。当配置的日志轮换时，当前日志内容将保留在单个备份中，清空的日志将恢复日志记录。

配置的每个日志仅保留 1 个备份。

AWS ParallelCluster 将快速增长的日志配置为在大小达到 50 MB 时进行轮换。快速增长的日志与扩展有关 Slurm，包括`/var/log/parallelcluster/clustermgtd`/`var/log/parallelcluster/slurm_resume.log`、和`/var/log/slurmctld.log`。

AWS ParallelCluster 将增长缓慢的日志配置为在大小达到 10 MB 时进行轮换。

在启用日志记录的情况下，您可以查看在集群配置 `Logs`/`CloudWatch`/`RetentionInDays`设置中定义的天数内保留的较早日 CloudFormation 志。检查 `RetentionInDays` 设置，查看您的用例是否需要增加天数。

AWS ParallelCluster 配置和轮换以下日志：

头节点日志

```
/var/log/cloud-init.log
/var/log/supervisord.log
/var/log/cfn-init.log
/var/log/chef-client.log
/var/log/dcv/server.log
/var/log/dcv/sessionlauncher.log
/var/log/dcv/agent.*.log
/var/log/dcv/dcv-xsession.*.log
/var/log/dcv/Xdcv.*.log
/var/log/parallelcluster/pam_ssh_key_generator.log
/var/log/parallelcluster/clustermgtd
/var/log/parallelcluster/clusterstatusmgtd
/var/log/parallelcluster/slurm_fleet_status_manager.log
/var/log/parallelcluster/slurm_resume.log
```

```
/var/log/parallelcluster/slurm_suspend.log
/var/log/slurmctld.log
/var/log/slurmdbd.log
/var/log/parallelcluster/compute_console_output.log
```

计算节点日志

```
/var/log/cloud-init.log
/var/log/supervisord.log
/var/log/cloud-init-output.log
/var/log/parallelcluster/computemgtd
/var/log/slurmd.log
```

登录节点日志

```
/var/log/cloud-init.log
/var/log/cloud-init.log
/var/log/cloud-init-output.log
/var/log/supervisord.log
/var/log/parallelcluster/pam_ssh_key_generator.log
```

`pcluster` CLI 日志

`pcluster` CLI 将您的命令的日志写入 `/home/user/.parallelcluster/` 下的 `pcluster.log.#` 文件中。

对于每条命令，日志通常都包括带输入的命令、用于发出该命令的 CLI API 版本的副本、响应以及信息和错误消息。对于创建和生成命令，日志还包括配置文件、配置文件验证操作、 CloudFormation 模板和堆栈命令。

您可以使用这些日志来验证错误、输入、版本和 `pcluster` CLI 命令。它们还可以记录发出命令的时间。

Amazon EC2 控制台输出日志

当 AWS ParallelCluster 检测到静态计算节点实例意外终止时，它会在一段时间后尝试从已终止的节点实例中检索 Amazon EC2 控制台输出。这样，如果计算节点无法与 Amazon 通信 CloudWatch，仍然可以从控制台输出中检索有关节点终止原因的有用故障排除信息。此控制台输出记录在头节点的 `/var/log/parallelcluster/compute_console_output` 日志中。有关亚马逊 EC2 控制台输出的更多信息，请参阅亚马逊 Linux 实例 EC2 用户指南中的实例控制台输出。

默认情况下， AWS ParallelCluster 仅从已终止节点的示例子集中检索控制台输出。在有大量终止导致多个控制台输出请求的情况下，这可防止集群头节点不堪重负。默认情况下， AWS ParallelCluster 在终止检测和控制台输出检索之间等待 5 分钟，让 Amazon 有 EC2 时间从节点检索最终的控制台输出。

您可以在头节点上的 `/etc/parallelcluster/slurm_plugin/parallelcluster_clustermgtd.conf` 文件中编辑样本量和等待时间参数值。

此功能已在 3.5.0 AWS ParallelCluster 版本中添加。

Amazon EC2 控制台输出参数

您可以在头节点`/etc/parallelcluster/slurm_plugin/parallelcluster_clustermgtd.conf`的文件中编辑以下 Amazon EC2 控制台输出参数的值。

`compute_console_logging_enabled`

要禁用控制台输出日志收集，请将 `compute_console_logging_enabled` 设置为 `false`。默认值为 `true`。

您可以随时更新此参数，而无需停止计算实例集。

`compute_console_logging_max_sample_size`

`compute_console_logging_max_sample_size`设置每次检测到意外终止时从中 AWS ParallelCluster 收集控制台输出的最大计算节点数。如果此值小于1，则从所有终止的节点 AWS ParallelCluster 检索控制台输出。默认值为 1。

您可以随时更新此参数，而无需停止计算实例集。

`compute_console_wait_time`

`compute_console_wait_time`设置从检测到节点故障到从该节点收集控制台输出之间 AWS ParallelCluster 等待的时间（以秒为单位）。如果您确定 Amazon EC2 需要更多时间来收集已终止节点的最终输出，则可以延长等待时间。默认值为 300 秒（5 分钟）。

您可以随时更新此参数，而无需停止计算实例集。

检索 PCUI 和 AWS ParallelCluster 运行时日志

了解如何检索 PCUI 和 AWS ParallelCluster 运行时日志以进行故障排除。首先，找到相关的 PCUI 和 AWS ParallelCluster 堆栈名称。使用堆栈名称找到安装日志组。最后，导出日志。这些日志特定于 AWS ParallelCluster 运行时系统。有关集群日志，请参阅检索和保留日志。

先决条件

- AWS CLI 已安装。
- 您拥有在 PCUI 开启 AWS 账户 的上运行 AWS CLI 命令的凭据。
- 您可以在 PCUI 处于开启状态 AWS 账户 的上访问亚马逊 CloudWatch 控制台。

步骤 1：找到相关堆栈的堆栈名称

在以下示例中，将红色突出显示的文本替换为实际值。

使用安装 PCUI 的 AWS 区域 位置列出堆栈：

```
$ aws cloudformation list-stacks --region aws-region-id
```

请注意以下堆栈的堆栈名称：

- 在您的账户中部署 PCUI 的堆栈的名称。您在安装 PCUI 时输入了该名称；例如 `pcluster-ui`。
- 以您输入的 AWS ParallelCluster 堆栈名称为前缀的堆栈；例如，`pcluster-ui-ParallelClusterApi-ABCD1234EFGH`。

步骤 2：找到日志组

列出 PCUI 堆栈的日志组，如以下示例所示：

```
$ aws cloudformation describe-stack-resources \
   --region aws-region-id \
   --stack-name pcluster-ui \
   --query "StackResources[?ResourceType == 'AWS::Logs::LogGroup' &&
 (LogicalResourceId == 'ApiGatewayAccessLog' || LogicalResourceId ==
 'ParallelClusterUILambdaLogGroup')].PhysicalResourceId" \
   --output text
```

列出 AWS ParallelCluster API 堆栈的日志组，如以下示例所示：

```
$ aws cloudformation describe-stack-resources \
   --region aws-region-id \
   --stack-name pcluster-ui-ParallelCluster-Api-ABCD1234EFGH \
   --query "StackResources[?ResourceType == 'AWS::Logs::LogGroup' && LogicalResourceId
 == 'ParallelClusterFunctionLogGroup'].PhysicalResourceId" \
```

```
    --output text
```

记下日志组列表，以便在下一个步骤中使用。

步骤 3：导出日志

使用以下步骤收集并导出日志：

1. 登录 AWS Management Console，然后在 PCUI 已开启的上导航到 Amazon CloudWatch 控制台。 AWS 账户
2. 在导航窗格上，依次选择日志和 日志见解。
3. 选择上一步中列出的所有日志组。
4. 选择时间范围，例如 12 小时。
5. 运行以下查询：

```
$ fields @timestamp, @message
| sort @timestamp desc
| limit 10000
```

6. 选择导出结果、下载表 (JSON)。

检索和保留日志

AWS ParallelCluster 为计算实例 HeadNode 和存储创建 Amazon EC2 指标。您可以在控制台的自定义 CloudWatch 控制面板中查看指标。 AWS ParallelCluster 还会在 CloudWatch 日志组中创建集群日志流。您可以在控制台的自定义 CloudWatch 控制面板或日志组中查看这些日志。监控集群配置部分描述了如何修改集群 CloudWatch日志和控制面板。有关更多信息，请参阅与 Amazon CloudWatch 日志集成和亚马逊 CloudWatch 控制面板。

日志是用于排查问题的有用资源。例如，如果您想要删除失败的集群，则首先创建该集群的日志存档可能会很有用。按照存档日志中的步骤创建存档。

主题

- 中的集群日志不可用 CloudWatch
- 存档日志
- 保留的日志
- 已终止节点日志

中的集群日志不可用 CloudWatch

如果中没有集群日志 CloudWatch，请检查以确保在向配置中添加自定义 AWS ParallelCluster CloudWatch 日志时没有覆盖日志配置。

要向 CloudWatch 配置中添加自定义日志，请确保附加到配置中，而不是获取和覆盖配置。有关`fetch-config`和的更多信息`append-config`，请参阅《CloudWatch 用户指南》中的多个 CloudWatch 代理配置文件。

要恢复 AWS ParallelCluster CloudWatch 日志配置，可以在 AWS ParallelCluster 节点内运行以下命令：

```
$ PLATFORM="$(ohai platform | jq -r ".[]")"
LOG_GROUP_NAME="$(cat /etc/chef/dna.json | jq -r ".cluster.log_group_name")"
SCHEDULER="$(cat /etc/chef/dna.json | jq -r ".cluster.scheduler")"
NODE_ROLE="$(cat /etc/chef/dna.json | jq -r ".cluster.node_type")"
CONFIG_DATA_PATH="/usr/local/etc/cloudwatch_agent_config.json"
/opt/parallelcluster/pyenv/versions/cookbook_virtualenv/bin/python /usr/local/bin/
write_cloudwatch_agent_json.py --platform $PLATFORM --config $CONFIG_DATA_PATH --log-
group $LOG_GROUP_NAME --scheduler $SCHEDULER --node-role $NODE_ROLE
/opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m ec2
 -c file:/opt/aws/amazon-cloudwatch-agent/etc/amazon-cloudwatch-agent.json -s
```

存档日志

您可以将日志存档到 Amazon S3 或本地文件中（具体取决于 `--output-file` 参数）。

> **Note**
>
> 从 AWS ParallelCluster 3.12.0 开始，您可以将日志导出到默认 AWS ParallelCluster 存储桶。在这种情况下，您无需设置存储桶权限。

> **Note**
>
> 向 Amazon S3 存储桶策略添加权限以授予 CloudWatch 访问权限。有关更多信息，请参阅CloudWatch 日志用户指南中的对 Amazon S3 存储桶设置权限。

```
$ pcluster export-cluster-logs --cluster-name mycluster --region eu-west-1 \
```

```
  --bucket bucketname --bucket-prefix logs
{
  "url": "https://bucketname.s3.eu-west-1.amazonaws.com/export-log/mycluster-
logs-202109071136.tar.gz?..."
}

# use the --output-file parameter to save the logs locally
$ pcluster export-cluster-logs --cluster-name mycluster --region eu-west-1 \
  --bucket bucketname --bucket-prefix logs --output-file /tmp/archive.tar.gz
{
  "path": "/tmp/archive.tar.gz"
}
```

除非在配置或`export-cluster-logs`命令参数中明确指定，否则该档案包含过去 14 天来自头节点和计算节点的 Amazon CloudWatch Logs 流和 AWS CloudFormation 堆栈事件。命令完成所需的时间取决于集群中的节点数量和日志中 CloudWatch 可用的日志流数量。有关可用日志流的更多信息，请参阅与 Amazon CloudWatch 日志集成。

保留的日志

从版本 3.0.0 开始，删除集群时会默认 AWS ParallelCluster 保留 CloudWatch 日志。如果您想要删除集群并保留其日志，请确保集群配置中的 `Monitoring`/`Logs`/`CloudWatch`/`DeletionPolicy` 未设置为 `Delete`。否则，请将此字段的值更改为 `Retain`，然后运行 `pcluster update-cluster` 命令。然后，运行`pcluster delete-cluster --cluster-name <cluster_name>`删除集群，但保留存储在 Amazon 中的日志组 CloudWatch。

已终止节点日志

如果静态计算节点意外终止并且 CloudWatch 没有日志，请检查`/var/log/parallelcluster/compute_console_output`日志中是否 AWS ParallelCluster 已将该计算节点的控制台输出记录在头节点上。有关更多信息，请参阅 用于调试的关键日志。

如果`/var/log/parallelcluster/compute_console_output`日志不可用或不包含节点的输出，请使用从故障节点检索控制台输出。 AWS CLI 登录到集群头节点并从 `/var/log/parallelcluster/slurm_resume.log` 文件中获取失败节点的 `instance-id`。

使用 `instance-id` 运行以下命令，检索控制台输出：

```
$ aws ec2 get-console-output --instance-id i-abcdef01234567890
```

如果动态计算节点在启动后自行终止并且 CloudWatch 没有日志，请提交激活集群扩展操作的作业。等待实例失败，然后检索实例控制台日志。

登录到集群头节点并从 `/var/log/parallelcluster/slurm_resume.log` 文件中获取计算节点的 `instance-id`。

要检索实例控制台日志，请使用以下命令：

```
$ aws ec2 get-console-output --instance-id i-abcdef01234567890
```

当计算节点日志不可用时，控制台输出日志可以帮助您调试计算节点失败的根本原因。

AWS CloudFormation 自定义资源

从 3.6.0 AWS ParallelCluster 版开始，您可以在堆栈中使用 AWS ParallelCluster CloudFormation 自定义资源。 AWS CloudFormation 自定义资源是 AWS ParallelCluster 托管堆栈。这样，您就可以使用 CloudFormation 来配置和管理您的集群。例如，您可以在 CloudFormation 堆栈中配置群集外部资源，例如网络、共享存储和安全组基础架构。此外，您可以使用 CloudFormation 基础设施即代码管道来管理集群。

通过执行以下操作向 CloudFormation 模板添加 AWS ParallelCluster 自定义资源：

1. 添加由其拥有和托管的自定义资源提供程序堆栈 AWS ParallelCluster。
2. 将 CloudFormation 模板中的提供程序堆栈作为自定义资源引用。

自定义资源提供程序堆栈处理和响应 CloudFormation 请求。例如，在部署 CloudFormation 堆栈时，还要配置和创建集群。要更新集群，您需要更新 CloudFormation 堆栈。在删除堆栈时，将会删除集群。有关 CloudFormation 自定义资源的更多信息，请参阅AWS CloudFormation 用户指南中的自定义资源。

> **Warning**
>
> CloudFormation 未检测到自定义资源偏差。仅 CloudFormation 用于更新集群配置和删除集群。
> 您可以使用 `pcluster` CLI 或 AWS ParallelCluster 用户界面 来监控集群的状态或更新计算实例集，但不得使用它们来更新集群配置或删除集群。

> **Note**
>
> 我们建议您为堆栈添加终止保护，以避免意外删除。

由托管的提供商堆栈 AWS ParallelCluster

自定义资源提供程序堆栈的格式如以下 CloudFormation 模板片段所示：

```
PclusterClusterProvider:
  Type: AWS::CloudFormation::Stack
  Properties:
    Parameters:
      CustomLambdaRole: # (Optional) RoleARN to override default
      AdditionalIamPolicies: # (Optional) comma-separated list of IAM policies to add
    TemplateURL: !Sub
      - https://${AWS::Region}-aws-parallelcluster.s3.${AWS::Region}.${AWS::URLSuffix}/
parallelcluster/${Version}/templates/custom_resource/cluster.yaml
      - { Version: 3.13.2 }
```

属性：

参数：

CustomLambdaRole（可选）：

具有运行 AWS Lambda（可创建和管理集群）的权限的自定义角色。默认情况下，该角色使用 AWS ParallelCluster 文档中默认定义的相同策略。

AdditionalIamPolicies（可选）：

以逗号分隔的其他 IAM 策略 Amazon 资源名称 (ARNs) 列表，以添加到 Lambda 使用的角色中。仅在未指定 `CustomLambdaRole` 时使用，并且可以留空。

如果需要对头节点、计算节点使用其他策略或需要其他策略以访问 Amazon S3 存储桶，请将其添加到 `CustomLambdaRole` 或 `AdditionalIamPolicy` 属性中。

如果您需要将其他策略附加到头节点，则还必须授予必要的权限，以便将这些策略附加或分离到与头节点关联的 IAM 角色。具体而言，您需要将 “ia AttachRolePolicy m:” 和 “ia DetachRolePolicy m:” 权限（或托管策略中的等效权限）附加到头节点使用的 IAM 角色。有关更多信息，请参阅 AWS ParallelCluster 用于管理 IAM 资源的用户示例策略。

有关默认策略的更多信息，请参阅 AWS Identity and Access Management 中的权限 AWS ParallelCluster。

TemplateURL（必需）：

AWS ParallelCluster 自定义资源文件 URL。

输出：

ServiceToken:

可用作自定义资源 `ServiceToken` 属性的值。自定义资源`ServiceToken`指定 AWS CloudFormation 将请求发送到何处。这是您在 AWS CloudFormation 模板中包含的群集资源的必填输入。

LogGroupArn:

底层资源登录到 CloudWatch `LogGroup`的 ARN。

LambdaLayerArn:

用于运行操作的 Lambda 层的 ARN。 AWS ParallelCluster

集群资源

CloudFormation 群集资源的格式如以下 CloudFormation 模板片段所示：

```
PclusterCluster:
  Type: Custom::PclusterCluster
  Properties:
    ServiceToken: !GetAtt [ PclusterClusterProvider , Outputs.ServiceToken ]
    ClusterName: !Sub 'c-${AWS::StackName}' # Must be different from StackName
    ClusterConfiguration:
    # Your Cluster Configuration
```

属性：

ServiceToken:

AWS ParallelCluster 提供程序堆栈`ServiceToken`输出。

ClusterName:

要创建和管理的集群的名称。该名称不得与 CloudFormation 堆栈的名称相匹配。创建集群后将无法更改该名称。

ClusterConfiguration:

集群配置 YAML 文件，如集群配置文件中所述。但是，您可以使用常用的 CloudFormation 结构，例如内部函数。

DeletionPolicy:

定义在删除根堆栈时是否删除集群。默认值为 `Delete`。

Retain：

如果删除了自定义资源，则保留集群。

> **Note**
> 为了使保留的集群继续正常运行，依赖于集群的资源（例如存储和网络）必须将删除策略设置为保留。

Delete：

如果删除了自定义资源，则删除集群。

`Fn::GetAtt` 返回值：

`Fn::GetAtt` 内置函数会为指定属性返回某种类型的值。有关使用该`Fn::GetAtt intrinsic`函数的更多信息，请参阅 Fn:: GetAtt。

ClusterProperties:

`pcluster describe-cluster` 操作返回的值。

validationMessages：

包含上次创建或更新操作期间出现的所有验证消息的字符串。

logGroupName:

用于记录 Lambda 集群操作的日志组的名称。日志事件保留 90 天，集群删除后日志组将保留。

示例：`Fn::GetAtt`：

```
# Provide the public IP address of the head node as an output of a stack
```

```
Outputs:
  HeadNodeIp:
    Description: The public IP address of the head node
    Value: !GetAtt [ PclusterCluster, headNode.publicIpAddress ]
```

示例：带有 AWS ParallelCluster 自定义资源的简单、完整的 CloudFormation 模板：

```
AWSTemplateFormatVersion: '2010-09-09'
Description: > AWS ParallelCluster CloudFormation Template

Parameters:
  HeadNodeSubnet:
    Description: Subnet where the HeadNode will run
    Type: AWS::EC2::Subnet::Id

  ComputeSubnet:
    Description: Subnet where the Compute Nodes will run
    Type: AWS::EC2::Subnet::Id

  KeyName:
    Description: KeyPair to login to the head node
    Type: AWS::EC2::KeyPair::KeyName

Resources:
  PclusterClusterProvider:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: !Sub
        - https://${AWS::Region}-aws-parallelcluster.s3.${AWS::Region}.
${AWS::URLSuffix}/parallelcluster/${Version}/templates/custom_resource/cluster.yaml
        - { Version: 3.13.2 }

  PclusterCluster:
    Type: Custom::PclusterCluster
    Properties:
      ServiceToken: !GetAtt [ PclusterClusterProvider , Outputs.ServiceToken ]
      ClusterName: !Sub 'c-${AWS::StackName}'
      ClusterConfiguration:
        Image:
          Os: alinux2
        HeadNode:
          InstanceType: t2.medium
          Networking:
```

```
            SubnetId: !Ref HeadNodeSubnet
          Ssh:
            KeyName: !Ref KeyName
        Scheduling:
          Scheduler: slurm
          SlurmQueues:
          - Name: queue0
            ComputeResources:
            - Name: queue0-cr0
              InstanceType: t2.micro
            Networking:
              SubnetIds:
              - !Ref ComputeSubnet

Outputs:
  HeadNodeIp:
    Description: The Public IP address of the HeadNode
    Value: !GetAtt [ PclusterCluster, headNode.publicIpAddress ]
  ValidationMessages:
    Description: Any warnings from cluster create or update operations.
    Value: !GetAtt PclusterCluster.validationMessages
```

要了解有关如何使用 CloudFormation AWS ParallelCluster 自定义资源的更多信息，请参阅使用创建集群 AWS CloudFormation。

集群操作

将群集自定义资源添加到 CloudFormation 堆栈后， CloudFormation 可以执行以下群集操作：

- CloudFormation 在部署包含 AWS ParallelCluster 自定义资源的堆栈时，会在新的单独堆栈中创建集群。
- 如果您更新堆栈中定义的群集配置，则会根据配置更新策略 CloudFormation 更新群集。 AWS ParallelCluster 自定义资源提供程序不会在更新集群之前停止计算队列。我们建议对集群更新使用 `QueueUpdateStrategy` 设置。这样，在使用 AWS ParallelCluster 自定义资源时，可以避免在更新前后进行显式`pcluster update-compute-fleet`调用。
- 如果删除堆栈，则会删除集群。

对包含 AWS ParallelCluster 自定义资源的堆栈进行故障排除

使用 AWS ParallelCluster 自定义资源，从新的独立堆栈 CloudFormation 部署群集。您可以通过执行以下步骤来监控集群创建：

1. 导航到， AWS Management Console 然后 CloudFormation 在导航窗格中选择 Stacks。
2. 选择名为您为集群名称定义的名称的堆栈。
3. 如果堆栈状态为 `ROLLBACK_COMPLETE`，则表明在创建集群过程中出现了错误。
4. 选择堆栈详细信息，然后选择事件选项卡。
5. 在逻辑 ID 上搜索事件，查找您为集群名称定义的名称。该事件包含一个 `Status reason`，给出问题的理由。
6. 您也可以选择堆栈下拉菜单，然后选择已删除以查看已删除堆栈的列表。选择包含该集群名称的堆栈并查看事件以了解更多详细信息。
7. 要查看管理群集的自定义资源提供程序的输出，请选择描述为 “AWS ParallelCluster 群集自定义资源” 的堆栈。选择资源选项卡，找到逻辑 ID 为 `PclusterCfnFunctionLogGroup` 的资源，然后点击提供的链接。查看显示 Lambda 调试输出的日志流。
8. 要对集群进行故障排除，请参阅 AWS ParallelCluster 故障排除。

Elastic Fabric Adapter

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

要将 EFA 与 AWS ParallelCluster 和一起使用 Slurm 调度器，将 `SlurmQueues`//`ComputeResourcesEfa`/设置为`Enabled`。`true`

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

我们建议您在置放群组中运行启用 EFA 的实例。这样，这些实例便可启动到单个可用区的低延迟组中。有关如何使用 AWS ParallelCluster配置置放群组的更多信息，请参阅 `SlurmQueues`/`Networking`/`PlacementGroup`。

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

> **Note**
> 不支持在不同的可用区之间使用 Elastic Fabric Adapter (EFA)。有关更多信息，请参阅日程安排/SlurmQueues/联网/SubnetIds。

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

启用 Intel MPI

英特尔 MPI 已在。 AWS ParallelCluster AMIs

> **Note**
> 要使用 Intel MPI，必须确认并接受 Intel 简化软件许可证的条款。

默认情况下，Open MPI 位于路径上。要启用 Intel MPI 而不是 Open MPI，必须先加载 Intel MPI 模块。然后需要使用 `module load intelmpi` 安装最新版本。模块的确切名称随每次更新发生变化。要查看哪些模块可用，请运行 `module avail`。输出如下所示。

```
$ module avail
-----------------------------/usr/share/Modules/modulefiles
 --------------------------------
dot                             modules
libfabric-aws/1.16.0~amzn3.0    null
module-git                      openmpi/4.1.4
module-info                     use.own

----------------------------/opt/intel/mpi/2021.6.0/modulefiles
 ---------------------------
intelmpi
```

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

```
$ module load intelmpi
```

要查看加载了哪些模块，请运行 `module list`。

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

要验证是否已启用 Intel MPI，请运行 `mpirun --version`。

```
$ mpirun --version
Intel(R) MPI Library for Linux* OS, Version 2021.6 Build 20220227 (id: 28877f3f32)
Copyright 2003-2022, Intel Corporation.
```

加载 Intel MPI 模块后，将更改多个路径以使用 Intel MPI 工具。要运行由 Intel MPI 工具编译的代码，请先加载 Intel MPI 模块。

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

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

AWS ParallelCluster API

什么是 AWS ParallelCluster API？

AWS ParallelCluster API 是一种无服务器应用程序，一旦部署到您的应用程序 AWS 账户，便可通过 API 提供对 AWS ParallelCluster 功能的编程访问。

AWS ParallelCluster API 以独立AWS CloudFormation模板的形式分发，其中包括一个公开 AWS ParallelCluster 功能的 Amazon API Gateway 终端节点和一个负责处理所调用AWS Lambda功能的函数。

下图显示了 AWS ParallelCluster API 基础架构的高级架构图。

AWS ParallelCluster API 文档

描述该 API 的 OpenAPI 规范文件可以从以下网址下载： AWS ParallelCluster

```
https://<REGION>-aws-parallelcluster.s3.<REGION>.amazonaws.com/
parallelcluster/<VERSION>/api/ParallelCluster.openapi.yaml
```

从 OpenAPI 规范文件开始，你可以使用 Swagger UI 或 Redoc 等众多可用工具中的一个来生成 API 的文档。 AWS ParallelCluster

如何部署 AWS ParallelCluster API

要部署 AWS ParallelCluster API，你需要成为 API 的管理员 AWS 账户。

用于部署 API 的模板可在以下 URL 中找到：

```
https://<REGION>-aws-parallelcluster.s3.<REGION>.amazonaws.com/
parallelcluster/<VERSION>/api/parallelcluster-api.yaml
```

哪里*<REGION>*是需要部署 API AWS 区域 的地方，*<VERSION>*是 AWS ParallelCluster 版本（例如 3.13.2）。

AWS Lambda 使用带有 Lambda 层接口AWS ParallelCluster Python 库 API来处理 API 调用的功能。

> **Warning**
>
> 中任何有权访问我们的 AWS 账户 Amazon API Gateway 服务的用户都会自动继承管理 AWS ParallelCluster API 资源的权限。 AWS Lambda

使用以下方式部署 AWS ParallelCluster API AWS CLI

在本节中，您将学习如何使用进行部署 AWS CLI。

如果您尚未配置要与 CLI 一起使用的 AWS 凭证，请执行此操作。

```
$ aws configure
```

运行以下命令以部署 API：

```
$ REGION=<region>
$ API_STACK_NAME=<stack-name>  # This can be any name
$ VERSION=3.13.2
$ aws cloudformation create-stack \
    --region ${REGION} \
    --stack-name ${API_STACK_NAME} \
    --template-url https://${REGION}-aws-parallelcluster.s3.${REGION}.amazonaws.com/
parallelcluster/${VERSION}/api/parallelcluster-api.yaml \
    --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND
$ aws cloudformation wait stack-create-complete --stack-name ${API_STACK_NAME} --region
 ${REGION}
```

自定义部署

您可以使用模板公开的 AWS CloudFormation 参数来自定义 API 部署。要在通过 CLI 部署时配置参数的值，可以使用以下选项：`--parameters ParameterKey=KeyName,ParameterValue=Value`。

以下参数可选：

- 区域-使用`Region`参数指定 API 是能够控制全部资源 AWS 区域 （默认）还是单个资源 AWS 区域。将此值设置 AWS 区域 为要部署的 API 以限制访问权限。
- ParallelClusterFunctionRole-这会覆盖分配给实现 AWS ParallelCluster 功能的 AWS Lambda 函数的 IAM 角色。该参数接受 IAM 角色的 ARN。需要将此角色配置为 AWS Lambda 作为 IAM 委托人。此外，由于此角色将取代 API Lambda 函数的默认角色，因此它必须至少具有 API 所需的默认权限，如中所列。AWS ParallelCluster `pcluster`用户策略示例
- ParallelClusterFunctionAdditionalPolicies-要附加到 AWS ParallelCluster API 函数角色的其他 IAM 策略的 ARN。只能指定一个策略。
- CustomDomainName，CustomDomainCertificate，CustomDomainHostedZoneId-使用这些参数为 Amazon API Gateway 终端节点设置自定义域。 `CustomDomainName`是要使用的域名，`CustomDomainCertificate`是该域名的 AWS 托管证书的 ARN，`CustomDomainHostedZoneId`也是您要在其中创建记录的 A mazon Route 53 托管区域的 ID。

> **Warning**
>
> 您可以配置自定义域设置，以对 API 强制使用最低版本的传输层安全性协议 (TLS)。有关更多信息，请参阅在 API Gateway 中为自定义域选择最低 TLS 版本。

- EnableIamAdminAccess-默认情况下，处理 AWS ParallelCluster API 操作的 AWS Lambda 函数配置为阻止任何特权 IAM 访问的 IAM 角色 (`EnableIamAdminAccess=false`)。这使得 API 无法处理需要创建 IAM 角色或策略的操作。因此，只有在资源配置过程中提供 IAM 角色作为输入时，才能成功创建集群或自定义映像。

  当设置`EnableIamAdminAccess`为`true`时， AWS ParallelCluster API 将获得管理部署集群或生成自定义集群所需的 IAM 角色创建的权限 AMIs。

  > **Warning**
  >
  > 如果将其设置为 true，则会向处理 AWS ParallelCluster 操作的 AWS Lambda 函数授予 IAM 管理员权限。

  有关启用此模式时可以解锁的功能的更多详细信息，请参阅。AWS ParallelCluster 用于管理 IAM 资源的用户示例策略

- PermissionsBoundaryPolicy-此可选参数接受现有 IAM 策略 ARN，该策略将被设置为 PC API 基础设施创建的所有 IAM 角色的权限边界，并设置为管理 IAM 权限的条件，因此 PC API 只能创建具有此策略的角色。

  有关此模式施加的限制的更多详细信息，请参阅 `PermissionsBoundary` 模式。

- CreateApiUserRole-默认情况下， AWS ParallelCluster API 的部署包括创建 IAM 角色，该角色被设置为唯一有权调用 API 的角色。 Amazon API Gateway 终端节点配置了基于资源的策略，仅向创建的用户授予调用权限。 要更改此设置，请设置`CreateApiUserRole=false`并向选定的 IAM 用户授予 API 访问权限。有关更多信息，请参阅 Amazon API Gateway 开发者指南中的控制调用 API 的访问权限。

  > **Warning**
  >
  > 如果`CreateApiUserRole=true`对 API 终端节点的访问不受到 Amazon API Gateway 资源策略的限制，则所有拥有不受限制`execute-api:Invoke`权限的 IAM 角色都可以访问

AWS ParallelCluster 功能。有关更多信息，请参阅《API Gat eway 开发者指南》中的使用 API Gateway 资源策略控制 API 的访问权限。

> **Warning**
>
> `ParallelClusterApiUserRole` 有权调用所有 AWS ParallelCluster API 操作。要限制对一部分 API 资源的访问，请参阅《API Gateway 开发者指南》中的使用 IAM 策略控制谁可以调用 API Gateway API 方法。

- IAMRoleAndPolicyPrefix-此可选参数接受一个最多包含 10 个字符的字符串，该字符串将用作作为 PC API 基础设施一部分创建的 IAM 角色和策略的前缀。

更新 API

在本节中，您将学习如何使用两个可用选项之一来更新 API。

升级到新 AWS ParallelCluster 版本

选项 1：要移除现有 API，请删除相应的 AWS CloudFormation 堆栈并部署新的 API，如上所示。

选项 2：要更新现有 API，请运行以下命令：

```
$ REGION=<region>
$ API_STACK_NAME=<stack-name>  # This needs to correspond to the existing API stack
 name
$ VERSION=3.13.2
$ aws cloudformation update-stack \
    --region ${REGION} \
    --stack-name ${API_STACK_NAME} \
    --template-url https://${REGION}-aws-parallelcluster.s3.${REGION}.amazonaws.com/
parallelcluster/${VERSION}/api/parallelcluster-api.yaml \
    --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND
$ aws cloudformation wait stack-update-complete --stack-name ${API_STACK_NAME} --region
 ${REGION}
```

正在调用 API AWS ParallelCluster

AWS ParallelCluster Amazon API Gateway 终端节点配置了`AWS_IAM`授权类型，要求所有请求都必须使用有效的 IAM 凭证进行 Sigv4 签名（API 参考：发出 http 请求）。

使用默认设置进行部署时，API 调用权限仅授予给使用 API 创建的默认 IAM 用户。

要检索默认 IAM 用户的 ARN，请运行：

```
$ REGION=<region>
$ API_STACK_NAME=<stack-name>
$ aws cloudformation describe-stacks --region ${REGION} --stack-name ${API_STACK_NAME}
 --query "Stacks[0].Outputs[?OutputKey=='ParallelClusterApiUserRole'].OutputValue" --
output text
```

要获取默认 IAM 用户的临时证书，请运行 STS AssumeRole 命令。

要检索 AWS ParallelCluster API 端点，请运行以下命令：

```
$ REGION=<region>
$ API_STACK_NAME=<stack-name>
$ aws cloudformation describe-stacks --region ${REGION} --stack-name ${API_STACK_NAME}
 --query "Stacks[0].Outputs[?OutputKey=='ParallelClusterApiInvokeUrl'].OutputValue" --
output text
```

该 AWS ParallelCluster API 可以由任何符合 OpenAPI 规范的 HTTP 客户端调用，可以在此处找到：

```
https://<REGION>-aws-parallelcluster.s3.<REGION>.amazonaws.com/
parallelcluster/<VERSION>/api/ParallelCluster.openapi.yaml
```

请求必须按照此处所述进行 SigV4 签名。

目前，我们不提供任何官方 API 客户端实现。但是，您可以使用 OpenAPI 生成器轻松地从 OpenAPI 模型生成 API 客户端。生成客户端后，如果没有开箱即用，则需要添加 Sigv4 签名。

可以在 AWS ParallelCluster 存储库中找到 Python API 客户端的参考实现。要详细了解如何使用 Python API 客户端，请参阅使用 AWS ParallelCluster API 教程。

要实施更高级的访问控制机制，例如亚马逊 Cognito 或 Lambda 授权者，或者要使用或 API 密钥进一步保护 API，请按照 Amazon API Gateway 文档进行 AWS WAF 操作。

> **Warning**
>
> 有权调用 AWS ParallelCluster API 的 IAM 用户可以间接控制 AWS ParallelCluster 中管理的所有 AWS 资源 AWS 账户。这包括创建由于用户 IAM 策略的限制而导致用户无法直接控制

的 AWS 资源。例如，根据集群的配置，创建 AWS ParallelCluster 集群可能包括部署亚马逊 EC2 实例、Amazon Route 53、Amazon Elastic File System FSx 文件系统、Amazon 文件系统、IAM 角色以及用户可能无法直接控制的其他 AWS 服务 资源。 AWS ParallelCluster

Warning

使用配置中`AdditionalIamPolicies`指定的方式创建集群时，其他策略必须与以下模式之一匹配：

```
- !Sub arn:${AWS::Partition}:iam::${AWS::AccountId}:policy/parallelcluster*
- !Sub arn:${AWS::Partition}:iam::${AWS::AccountId}:policy/parallelcluster/*
- !Sub arn:${AWS::Partition}:iam::aws:policy/CloudWatchAgentServerPolicy
- !Sub arn:${AWS::Partition}:iam::aws:policy/AmazonSSMManagedInstanceCore
- !Sub arn:${AWS::Partition}:iam::aws:policy/AWSBatchFullAccess
- !Sub arn:${AWS::Partition}:iam::aws:policy/AmazonS3ReadOnlyAccess
- !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/AWSBatchServiceRole
- !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role
- !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
- !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AmazonEC2SpotFleetTaggingRole
- !Sub arn:${AWS::Partition}:iam::aws:policy/EC2InstanceProfileForImageBuilder
- !Sub arn:${AWS::Partition}:iam::aws:policy/service-role/
AWSLambdaBasicExecutionRole
```

如果需要其他策略，则可以执行以下操作之一：

- 在以下文件中编辑 `DefaultParallelClusterIamAdminPolicy`：

  ```
  https://<REGION>-aws-parallelcluster.s3.<REGION>.amazonaws.com/
  parallelcluster/<VERSION>/api/parallelcluster-api.yaml
  ```

  将策略添加在 `ArnLike/iam:PolicyARN` 部分中。

- 不要在配置文件`AdditionalIamPolicies`中为指定策略，而是手动向在集群中创建的 AWS ParallelCluster 实例角色添加策略。

访问 API 日志和数据记录

API 日志发布到亚马逊 CloudWatch ，保留期为 30 天。要检索与 API 部署关联的 LogGroup 名称，请运行以下命令：

```
$ REGION=<region>
$ API_STACK_NAME=<stack-name>
$ aws cloudformation describe-stacks --region ${REGION} --
stack-name ${API_STACK_NAME} --query "Stacks[0].Outputs[?
OutputKey=='ParallelClusterLambdaLogGroup'].OutputValue" --output text
```

也可以通过 Lambda 控制台访问 Lambda 数据记录、日志和 AWS X-Ray 跟踪日志。要检索与 API 部署关联的 Lambda 函数的 ARN，请运行以下命令：

```
$ REGION=<region>
$ API_STACK_NAME=<stack-name>
$ aws cloudformation describe-stacks --region ${REGION} --stack-name ${API_STACK_NAME}
 --query "Stacks[0].Outputs[?OutputKey=='ParallelClusterLambdaArn'].OutputValue" --
output text
```

AWS ParallelCluster 适用于 Terraform

从 AWS ParallelCluster 3.8.0 开始，您可以使用 Terraform 部署集群和自定义镜像。要开始使用此功能，请参阅 Terraform Registry 中的 Terraform P AWS ParallelCluster rovid er。

> **Note**
>
> 您的账户中必须部署 ParallelCluster API 才能使用该提供商。

使用下表来确定提供程序和 AWS ParallelCluster 版本之间的兼容性：

提供程序版本	AWS ParallelCluster 版本
1.0.0	3.8.0-3.10.1
1.1.0	3.11.0+

查看如何使用提供程序的示例。

要获得更流畅的体验，请使用来 AWS ParallelCluster自 Terraform Registry 的官方 Terraform 模块。该模块让您能够部署：

1. ParallelCluster API
2. ParallelCluster 使用 YAML 配置文件和 HCL 定义的集群
3. ParallelCluster 集群所需的网络基础架构

查看如何使用该模块的示例。

通过 Amazon DCV 连接到头节点和登录节点

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

Amazon DCV 软件自动安装在头节点上，可以通过使用 `HeadNode` 中的 `Dcv` 部分来启用。

从 3.11 AWS ParallelCluster 版开始，可以使用 `LoginNodes` 部分 /池配置中的 Dcv 部分为登录节点池启用 Amazon DCV。

```
LoginNodes:
  Pools:
    Dcv:
      Enabled: true
```

AWS ParallelCluster 设置/home/*<DEFAULT_AMI_USER>*为 DCV 服务器存储文件夹。有关 Amazon DCV 配置参数的更多信息，请参阅 `HeadNode`/`Dcv`。

要连接到头节点上的 Amazon DCV 会话，请使用 `dcv-connect` 命令。要在登录节点上进行连接，请使用带有 `--login-node-ip` 参数的 `dcv-connect`，并传入要连接的登录节点的公有或私有 IP 地址。

Amazon DCV HTTPS 证书

Amazon DCV 会自动生成自签名证书，以确保 Amazon DCV 客户端和 Amazon DCV 服务器之间的流量安全。

要使用其它证书替换默认的自签名 Amazon DCV 证书，首先要连接到头节点。然后，在运行 `pcluster dcv-connect` 命令之前，将证书和密钥复制到 `/etc/dcv` 文件夹。

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

许可 Amazon DCV

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

AWS ParallelCluster 自动向头节点 IAM 策略添加所需的权限。使用自定义 IAM 实例策略时，请使用《亚马逊 DCV 管理员指南》中亚马逊的 Amazon DCV EC2 中描述的权限。

有关故障排除提示，请参阅 排查 Amazon DCV 中的问题。

使用 `pcluster update-cluster`

在 AWS ParallelCluster 3.x 中，`pcluster update-cluster`分析用于创建当前集群的设置以及配置文件中的设置是否存在问题。如果发现任何问题，就会进行报告并显示修复这些问题所要执行的步骤。例如，如果更改了计算 `InstanceType`，则必须先停止计算实例集，然后才能继续进行更新。此问题发现后即会报告。如果未发现阻止更新的问题，则会启动更新过程并报告更改。

在运行更新之前，您可以使用 `pcluster update-cluster` `--dryrun option` 来查看更改。有关更多信息，请参阅 `pcluster update-cluster` 示例。

有关故障排除指导，请参阅AWS ParallelCluster 故障排除。

更新策略：定义

更新策略：必须停止集群中的登录节点，才能更改此设置进行更新。

在使用集群中的登录节点时，无法更改这些设置。要么恢复更改，要么停止集群登录节点。（您可以通过将每个池的计数设置为 0 来停止群集中的登录节点）。集群的登录节点停止后，您可以更新集群（pcluster update-cluster）以激活更改。

> **Note**
>
> 从 3.7.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：可以添加登录节点池，但移除池需要停止集群中的所有登录节点。

要移除池，必须停止集群中的所有登录节点。（将每个池的计数设置为 0，即可停止集群中的登录节点）。集群的登录节点停止后，您可以更新集群（`pcluster update-cluster`）以激活更改。

> **Note**
> 从 3.11.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：必须停止池中的登录节点，才能更改此设置进行更新。

当池中的登录节点处于使用状态时，您无法更改这些设置。要么恢复更改，要么停止池登录节点。（将池的计数设置为 0，即可停止池中的登录节点）。池的登录节点停止后，您可以更新集群（`pcluster update-cluster`）以激活更改。

> **Note**
> 从 3.11.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：可以在更新期间更改此设置。

更改此设置后，可以使用 `pcluster update-cluster` 更新集群。

更新策略：如果更改此设置，则不允许更新。

更改此设置后，无法更新集群。您必须还原原始集群的设置，并使用更新的设置创建一个新集群。您可以在将来删除原始集群。要创建新集群，请使用 `pcluster create-cluster`。要删除原始集群，请使用 `pcluster delete-cluster`。

更新策略：在更新期间不分析此设置。

可以更改这些设置，并使用 `pcluster update-cluster` 更新集群。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

当存在计算实例集时，无法更改这些设置。必须还原更改，或者必须停止计算实例集（使用 `pcluster update-compute-fleet`）。停止计算实例集后，您可以更新集群 (`pcluster update-cluster`) 以激活更改。例如，如果您要在 `SlurmQueues`/`ComputeResources`/-`Name`/`MinCount` > 0 的情况下使用 Slurm 调度器，则会启动一个计算实例集。

更新策略：必须停止计算实例集和登录节点才能更改此设置以进行更新。

如果计算实例集存在，或登录节点正在使用中，则无法更改这些设置。要么恢复更改，要么停止计算实例集和登录节点（可使用 `pcluster update-compute-fleet` 停止计算实例集）。计算实例集和登录节点停止后，您可以更新集群（`pcluster update-cluster`）以激活更改。

更新策略：更新期间不能减小此设置。

这些设置可以更改，但不能减小。如果必须减小这些设置，您必须还原原始集群的设置，并使用更新的设置创建一个新集群。您可以在将来删除原始集群。要创建新集群，请使用 `pcluster create-cluster`。要删除原始集群，请使用 `pcluster delete-cluster`。

更新策略：如果更改此设置，则不允许更新。如果您强制更新，则将忽略新值并使用旧值。

更改此设置后，无法更新集群。您必须还原原始集群的设置，并使用更新的设置创建一个新集群。您可以在将来删除原始集群。要创建新集群，请使用 `pcluster create-cluster`。要删除原始集群，请使用 `pcluster delete-cluster`。

更新策略：必须停止计算实例集或必须设置 **`QueueUpdateStrategy`** 才能更改此设置以进行更新。

可以更改这些设置。必须停止计算实例集（使用 `pcluster update-compute-fleet`），或者必须设置 `QueueUpdateStrategy`。停止计算实例集或设置 `QueueUpdateStrategy` 后，您可以更新集群 (`pcluster update-cluster`) 以激活更改。

> **Note**
>
> 从 3.2.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：对于此列表值设置，可以在更新期间添加新值，或者在删除现有值时必须停止计算实例集。

可以在更新期间为这些设置添加新值。向列表中添加新值后，可以使用 (`pcluster update-cluster`) 更新集群。

要从列表中删除现有值，必须停止计算实例集（使用 `pcluster update-compute-fleet`）。

例如，如果您正在使用Slurm调度程序并向 Instances/ 添加新的实例类型InstanceType，则可以在不停止计算队列的情况下更新集群。要从 Instances/ 中移除现有实例类型InstanceType，必须先停止计算队列（使用 pcluster）。 update-compute-fleet

> **Note**
>
> 从 3.2.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：要缩小队列的大小，需要停止计算队列或者QueueUpdateStrategy必须将其设置为 TERMINATE 才能更改此设置以进行更新。

可以更改这些设置，但是如果更改会减小队列的大小，则必须停止计算队列（使用 pcluster update-compute-fleet）或QueueUpdateStrategy必须将其设置为 TERMINATION。在计算队列停止或设置QueueUpdateStrategy为终止后，您可以更新集群（pcluster update-cluster）以激活更改。

在调整集群容量时，设置 TERMINATE 只会终止节点列表后面的节点，而不会触及同一分区的所有其它节点。

例如，如果集群的初始容量为 `MinCount = 5` 和 `MaxCount = 10`，则节点为 `st-[1-5]; dy-[1-5]`。将集群大小调整为 `MinCount = 3` 和 `MaxCount = 5` 时，新的集群容量将由节点 `st-[1-3]; dy-[1-2]` 构成，更新期间不会触及这些节点。在更新过程中，只有 `st-[4-5]; dy-[3-5]` 节点会被终止。

支持以下更改，不需要停止计算队列，也不要求将其QueueUpdateStrategy设置为 TERMINATION：

- 添加了一个新SlurmQueue的
- 添加了一个新ComputeResource的
- MaxCount增加了
- MinCount增加MaxCount并且增加的金额至少相同

注意：从 3.9.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：对于此列表值设置，必须停止计算实例集或必须设置 **`QueueUpdateStrategy`** 才能添加新值；删除现有值时必须停止计算实例集。

可以在更新期间为这些设置添加新值。必须停止计算实例集（使用 `pcluster update-compute-fleet`），或者必须设置 `QueueUpdateStrategy`。停止计算实例集或设置 `QueueUpdateStrategy` 后，您可以更新集群 (`pcluster update-cluster`) 以激活更改。

要从列表中删除现有值，必须停止计算实例集（使用 `pcluster update-compute-fleet`）。

> **Note**
>
> 从 3.3.0 AWS ParallelCluster 版开始支持此更新政策。

更新策略：必须停止所有计算节点才能删除托管置放群组。必须停止计算实例集或必须设置 **QueueUpdateStrategy** 才能更改此设置以进行更新。

> 必须停止计算实例集（使用 pcluster update-compute-fleet）才能删除托管置放群组。如果您在停止计算实例集之前运行集群更新以删除托管置放群组，则会返回一条配置无效消息，并且不会继续更新。停止计算实例集可以保证没有实例在运行。

pcluster update-cluster 示例

可以更改这些设置，但是如果更改会减小队列的大小，则必须停止计算队列（使用 pcluster update-compute-fleet）或QueueUpdateStrategy必须将其设置为 TERMINATION。在计算队列停止或设置QueueUpdateStrategy为终止后，您可以更新集群（pcluster update-cluster）以激活更改。

- 此示例演示了一些允许的更改的更新，并且直接开始更新。

```
$ pcluster update-cluster --cluster-name cluster_name --cluster-config
 ~/.parallelcluster/test_cluster --region us-east-1
{
  "cluster": {
    "clusterName": cluster_name,
    "cloudformationStackStatus": "UPDATE_IN_PROGRESS",
    "cloudformationStackArn": stack_arn,
    "region": "us-east-1",
    "version": "3.13.2",
    "clusterStatus": "UPDATE_IN_PROGRESS"
  },
  "changeSet": [
    {
      "parameter": "HeadNode.Networking.AdditionalSecurityGroups",
      "requestedValue": [
        "sg-0cd61884c4ad11234"
      ],
      "currentValue": [
        "sg-0cd61884c4ad16341"
      ]
    }
  ]
}
```

- 此示例演示了一些允许的更改的试运行更新。试运行可用于在不启动更新的情况下报告更改集。

```
$ pcluster update-cluster --cluster-name cluster_name --cluster-config
 ~/.parallelcluster/test_cluster --region us-east-1 --dryrun true
{
  "message": "Request would have succeeded, but DryRun flag is set.",
  "changeSet": [
    {
      "parameter": "HeadNode.Networking.AdditionalSecurityGroups",
      "requestedValue": [
        "sg-0cd61884c4ad11234"
      ],
      "currentValue": [
        "sg-0cd61884c4ad16341"
      ]
    }
  ]
}
```

- 此示例演示了一些阻止更新的更改的更新。

```
$ pcluster update-cluster --cluster-name cluster_name --cluster-config
 ~/.parallelcluster/test_cluster --region us-east-1
{
  "message": "Update failure",
  "updateValidationErrors": [
    {
      "parameter": "HeadNode.Ssh.KeyName",
      "requestedValue": "mykey_2",
      "message": "Update actions are not currently supported for the 'KeyName'
 parameter. Restore 'KeyName' value to 'jenkinsjun'. If you need this change, please
 consider creating a new cluster instead of updating the existing one.",
      "currentValue": "mykey_1"
    },
    {
      "parameter": "Scheduling.SlurmQueues[queue1].ComputeResources[queue1-
t2micro].InstanceType",
      "requestedValue": "c4.xlarge",
      "message": "All compute nodes must be stopped. Stop the compute fleet with the
 pcluster update-compute-fleet command",
      "currentValue": "t2.micro"
    },
    {
      "parameter": "SharedStorage[ebs1].MountDir",
```

```
      "requestedValue": "/my/very/very/long/shared_dir",
      "message": "Update actions are not currently supported for the 'MountDir'
 parameter. Restore 'MountDir' value to '/shared'. If you need this change, please
 consider creating a new cluster instead of updating the existing one.",
      "currentValue": "/shared"
    }
  ],
  "changeSet": [
    {
      "parameter": "HeadNode.Networking.AdditionalSecurityGroups",
      "requestedValue": [
        "sg-0cd61884c4ad11234"
      ],
      "currentValue": [
        "sg-0cd61884c4ad16341"
      ]
    },
    {
      "parameter": "HeadNode.Ssh.KeyName",
      "requestedValue": "mykey_2",
      "currentValue": "mykey_1"
    },
    {
      "parameter": "Scheduling.SlurmQueues[queue1].ComputeResources[queue1-
t2micro].InstanceType",
      "requestedValue": "c4.xlarge",
      "currentValue": "t2.micro"
    },
    {
      "parameter": "SharedStorage[ebs1].MountDir",
      "requestedValue": "/my/very/very/long/shared_dir",
      "currentValue": "/shared"
    }
  ]
}
```

AWS ParallelCluster AMI 自定义

在某些情况下，需要为 AWS ParallelCluster 构建自定义 AMI。本节介绍构建自定义 AWS ParallelCluster AMI 时应考虑的事项。

您可以使用以下方法之一构建自定义 AWS ParallelCluster AMI：

1. 创建构建映像配置文件，然后使用 `pcluster` CLI 使用 Image Builder 构建 EC2 映像。此过程自动运行，可重复，并且支持监控。有关更多信息，请参阅 `pcluster` 映像命令。
2. 从 AWS ParallelCluster AMI 创建实例，然后登录该实例并进行手动修改。最后，使用 Amazon EC2 从修改后的实例创建新 AMI。此过程需要更少的时间。但过程不自动运行，也不可重复，而且不支持使用 `pcluster` CLI 映像监控命令。

有关这些方法的更多信息，请参阅构建自定义 AWS ParallelCluster AMI。

AWS ParallelCluster AMI 自定义注意事项

无论您如何创建自定义映像，我们都建议您执行初步验证测试，并包含相应的预置以监控所创建映像的状态。

要使用构建自定义 AMIpcluster，您需要创建一个带有`Build`和`Image`部分的构建映像配置文件，Im EC2 age B uilder 将使用该文件来构建您的自定义映像。`Build` 部分指定 Image Builder 在构建映像时需要的项目。这包括 `ParentImage`（基础映像）和 `Components`。Image Builder 组件定义一系列步骤，在创建映像之前自定义实例时或在测试由创建的映像启动的实例时，必须执行这些步骤。有关更多信息，请参阅 Image Builder 用户指南中的使用 Imag EC2 e Builder 创建自定义组件。

从 pcluster 调用`build-image`以创建自定义映像时，Image Builder 会使用带有 AWS ParallelCluster 食谱的构建映像配置来引导您 AWS ParallelCluster 。`ParentImage`Image Builder 将会下载组件、运行构建和验证阶段、创建 AMI、从 AMI 中启动实例并运行测试。该过程完成后，Image Builder 将生成新映像或停止消息。

执行自定义组件验证测试

在将 Image Builder 组件包含在配置中之前，请使用以下方法之一对其进行测试和验证。由于 Image Builder 进程可能需要长达 1 小时的时间，因此我们建议您事先测试这些组件。这可为您节省大量时间。

脚本案例

在构建映像过程之外，在正在运行的实例中测试脚本，并验证脚本是否以退出代码 0 退出。

Amazon 资源名称 (ARN) 案例

在构建映像过程之外，在正在运行的实例中测试组件文档。有关要求列表，请参阅 Image Builder User Guide 中的 Component manager。

成功验证后，将组件添加到构建映像配置中

确认自定义组件正常工作后，将其添加到构建映像配置文件中。

使用 pcluster 命令监控 Image Builder 进程以帮助进行调试

describe-image

使用此命令可监控构建映像状态。

list-image-log-streams

使用此命令获取可用于检索日志事件的日志流get-image-log-events。 IDs

get-image-log-events

使用此命令可获取构建映像进程事件的日志流。

例如，您可以使用以下命令来跟踪构建映像事件。

```
$ watch -n 1 'pcluster get-image-log-events -i <image-id> \
        --log-stream-name/1 <pcluster-version> \
        --query "events[*].message" | tail -n 50'
```

get-image-stack-events

使用此命令可检索 Image Builder 创建的堆栈的映像堆栈事件。

export-image-logs

使用此命令可保存映像日志。

有关 AWS ParallelCluster 日志和 Amazon 的更多信息 CloudWatch，请参阅Amazon CloudWatch Logs 生成镜像日志和亚马逊 CloudWatch 控制面板。

其他考虑因素

新 AWS ParallelCluster 版本和自定义 AMIs

如果构建并使用自定义 AMI，则必须重复执行用于随每个新的 AWS ParallelCluster 版本创建自定义 AMI 的步骤。

自定义引导操作

查看该自定义引导操作部分，确定您要进行的修改是否可以编写脚本并支持未来的 AWS ParallelCluster 版本。

使用自定义 AMIs

你可以在集群配置 AMIs 中的 `Image`/`CustomAmi`和//-`SchedulingSlurmQueuesNameImage`/`CustomAmi`部分中指定自定义。

要排查自定义 AMI 验证警告，请参阅排查自定义 AMI 问题。

使用按需容量预留（ODCR）启动实例

通过按需容量预留 (ODCR)，您可以在特定可用区为集群 Amazon EC2 实例预留容量。通过这种方式，您可以独立于由节省计划或区域性预留实例提供的账单账户创建和管理容量预留。

您可以配置`open`或`targeted` ODCR。开放式 ODCR 涵盖与 ODCR 属性匹配的所有实例。这些属性包括实例类型、平台和可用区。您必须在集群配置中显式定义定向 ODCR。要确定 ODC open R 是还是`targeted`，请运行 AWS CLI Amazon EC2 `describe-capacity-reservation`命令。

您还可以在集群置放群组中创建 ODCR，称为集群置放群组按需容量预留 (CPG ODCR)。

ODCRs 可以将多个资源分组到一个资源组中。这可以在集群配置文件中定义。有关资源组的更多信息，请参阅 Resource Groups and Tags User Guide 中的 What are resource groups?。

将 ODCR 与 AWS ParallelCluster

AWS ParallelCluster 支持开放式 ODCR。使用开放式 ODCR 时，您不需要在 AWS ParallelCluster中指定任何内容。系统会自动为集群选择实例。您可以指定现有的置放群组或为您 AWS ParallelCluster 创建一个新的置放群组。

集群配置中的 ODCR

从 AWS ParallelCluster 版本 3.3.0 开始，您可以在集群配置文件 ODCRs 中进行定义，而无需指定 Amazon EC2 运行实例替代。

您首先使用各自链接文档中描述的方法创建容量预留和资源组。您必须使用这些 AWS CLI 方法来创建容量预留组。如果您使用 AWS Management Console，则只能创建基于标签或基于堆栈的资源组。启动带有容量预留的实例 AWS CLI 时， AWS ParallelCluster 或不支持基于标签和基于堆栈的资源组。

创建容量预留和资源组后，在 SlurmQueues/CapacityReservationTarget 或 SlurmQueues/ComputeResources/CapacityReservationTarget 中指定它们，如以下集群配置示例所示。用您的有效值替换以红色*values*突出显示的内容。

```
Image:
  Os: os
HeadNode:
  InstanceType: head_node_instance
  Networking:
    SubnetId: public_subnet_id
  Ssh:
    KeyName: key_name
Scheduling:
  Scheduler: scheduler
  SlurmQueues:
    - Name: queue1
      Networking:
        SubnetIds:
          - private_subnet_id
      ComputeResources:
        - Name: cr1
          Instances:
            - InstanceType: instance
          MaxCount: max_queue_size
          MinCount: max_queue_size
          Efa:
            Enabled: true
          CapacityReservationTarget:
            CapacityReservationResourceGroupArn: capacity_reservation_arn
```

已过时/不推荐-带有 Amazon EC2 实例覆盖的目标 ODCR

> **Warning**
>
> - 从 3.3.0 AWS ParallelCluster 版开始，我们不建议使用这种方法。本节仍然是使用先前版本的实施参考。
> - 此方法与 Slurm 的多实例类型分配不兼容。

AWS ParallelCluster 3.1.1 中增加了`targeted` ODCRs 对 Support 的支持。在此版本中，引入了一种机制，该机制可以覆盖 `EC2RunInstances`参数并传递有关预留的信息，以用于中 AWS

ParallelCluster配置的每个计算资源。该机制与targeted ODCR 兼容。但在使用targeted ODCR 时，必须指定 run-instances 覆盖配置。 ODCRs 必须在 AWS CLI Amazon EC2 run-instances命令中明确定义@@ 目标。要确定 ODCR 是open还是targeted运行 AWS CLI Amazon EC2 命令describe-capacity-reservation。

ODCRs 可以将多个资源分组到一个资源组中。这可以用在运行实例覆盖中，以同时定位多个 ODCRs 实例。

如果您使用的是 targeted ODCR，则可以指定置放群组。但您还需要指定 run-instances 覆盖配置。

假设为您 AWS 创建了一个 targeted ODCR，或者您有一组特定的预留实例。则您无法指定置放群组。由配置的规则 AWS 可能与置放群组设置冲突。因此，如果您的应用程序需要置放群组，请使用 CPG ODCR。在任何一种情况下，您还必须指定 run-instances 覆盖配置。

如果您使用的是 CPG ODCR，则必须指定 run-instances 覆盖配置，并且必须在集群配置中指定相同的置放群组。

将预留实例与 AWS ParallelCluster

预留实例不同于容量预留 (ODCR)。预留实例有两种类型。区域性预留实例不预留容量。分区预留实例可在指定可用区中预留容量。

如果您有区域性预留实例，则没有容量预留，可能会出现容量不足错误。如果您有分区预留实例，则有容量预留，但没有 run-instances API 参数可以用来指定容量预留。

任何 AWS ParallelCluster 版本都支持预留实例。您无需在中指定任何内容 AWS ParallelCluster ，实例就会自动选中。

使用分区预留实例时，您可以不在集群配置中指定置放群组，从而避免潜在的容量不足错误。

已过时/不推荐-使用 AWS ParallelCluster 3 中的**RunInstances**自定义功能进行**targeted**按需容量预留 (ODCR)

> **Warning**
>
> - 从 3.3.0 AWS ParallelCluster 版开始，我们不建议使用这种方法。本节仍然是使用先前版本的实施参考。
> - 此方法与 Slurm 的多实例类型分配不兼容。

您可以为集群队列中配置的每个计算资源覆盖 Amazon EC2 `RunInstances` 参数。为此，请在集群的头节点上创建包含以下代码片段内容的 `/opt/slurm/etc/pcluster/run_instances_overrides.json` 文件：

- `${queue_name}` 是要对其应用覆盖的队列的名称。
- `${compute_resource_name}` 是要对其应用覆盖的计算资源。
- `${overrides}`是一个任意 JSON 对象，其中包含用于队列和实例类型特定组合的`RunInstances`替代列表。覆盖语法必须遵循与 run_instances boto3 调用中记录的规范相同的规范。

```
{
    "${queue_name}": {
        "${compute_resource_name}": {
            ${overrides}
        },
        ...
    },
    ...
}
```

例如，以下 JSON 将 ODCR 组 `group_arn` 配置为用于 `my-queue` 和 `my-compute-resource` 中配置的 `p4d.24xlarge` 实例。

```
{
    "my-queue": {
        "my-compute-resource": {
            "CapacityReservationSpecification": {
                "CapacityReservationTarget": {
                    "CapacityReservationResourceGroupArn": "group_arn"
                }
            }
        }
    }
}
```

生成此 JSON 文件后，负责集群扩展的 AWS ParallelCluster 守护程序会自动使用覆盖配置启动实例。要确认实例预置是否使用了指定的参数，请查看以下日志文件：

- `/var/log/parallelcluster/clustermgtd`（对于静态容量）
- `/var/log/parallelcluster/slurm_resume.log`（对于动态容量）

如果参数正确，您会发现包含以下内容的日志条目：

```
Found RunInstances parameters override. Launching instances with: <parameters_list>
```

过时/不推荐 - 创建包含 **targeted** 按需容量预留（ODCR）的集群

> **Warning**
>
> - 从 3.3.0 AWS ParallelCluster 版开始，我们不建议使用这种方法。本节仍然是使用先前版本的实施参考。
> - 此方法与 Slurm 的多实例类型分配不兼容。

1. 创建资源组以对容量分组。

```
$ aws resource-groups create-group --name EC2CRGroup \
    --configuration '{"Type":"AWS::EC2::CapacityReservationPool"}'
 '{"Type":"AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-
resource-types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

> **Note**
>
> 资源组不支持由其他账户共享的资源。
> 如果目标 ODCR 由其他账户共享，则无需创建资源组。在步骤 3 中使用 `CapacityReservationId` 代替资源组。

```
#!/bin/bash
set -e

# Override run_instance attributes
cat > /opt/slurm/etc/pcluster/run_instances_overrides.json << EOF
{
    "my-queue": {
        "my-compute-resource": {
            "CapacityReservationSpecification": {
                "CapacityReservationTarget": {
                    "CapacityReservationId": "cr-abcdef01234567890"
                }
            }
```

```
            }
        }
    }
EOF
```

向资源组中添加容量预留。每次创建新的 ODCR 时，请将其添加到群组预留中。*ACCOUNT_ID*替换为您的账户 ID、*PLACEHOLDER_CAPACITY_RESERVATION*容量预留 ID 和*REGION_ID*您的 AWS 区域 ID（例如 us-east-1）。

```
$ aws resource-groups group-resources --region REGION_ID --group EC2CRGroup \
    --resource-arns arn:aws:ec2:REGION_ID:ACCOUNT_ID:capacity-
reservation/PLACEHOLDER_CAPACITY_RESERVATION
```

在本地计算机中创建策略文档。*ACCOUNT_ID*替换为您的账户 ID 和 *REGION_ID* AWS 区域 ID（例如 us-east-1）。

```
cat > policy.json << EOF
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "RunInstancesInCapacityReservation",
            "Effect": "Allow",
            "Action": "ec2:RunInstances",
            "Resource": [
                "arn:aws:ec2:REGION_ID:ACCOUNT_ID:capacity-reservation/*",
                "arn:aws:resource-groups:REGION_ID:ACCOUNT_ID:group/*"
            ]
        }
    ]
}
EOF
```

2. AWS 账户 使用您创建的 json 文件在您的上创建 IAM 策略。

```
$ aws iam create-policy --policy-name RunInstancesCapacityReservation --policy-
document file://policy.json
```

3. 在实例上本地创建以下安装后脚本并将其命名为 **postinstall.sh**。

*ACCOUNT_ID*替换为你的 AWS 账户 身份证和*REGION_ID*你的 AWS 区域 身份证（例如 us-east-1）。

```
#!/bin/bash
set -e

# Override run_instance attributes
cat > /opt/slurm/etc/pcluster/run_instances_overrides.json << EOF
{
    "my-queue": {
        "my-compute-resource": {
            "CapacityReservationSpecification": {
                "CapacityReservationTarget": {
                    "CapacityReservationResourceGroupArn": "arn:aws:resource-
groups:REGION_ID:ACCOUNT_ID:group/EC2CRGroup"
                }
            }
        }
    }
}
EOF
```

将该文件上传到 Amazon S3 存储桶。*amzn-s3-demo-bucket*替换为您特定的 S3 存储桶名称。

```
$ aws s3 mb s3://amzn-s3-demo-bucket
aws s3 cp postinstall.sh s3://amzn-s3-demo-bucket/postinstall.sh
```

4. 创建本地集群配置，将占位符替换为您自己的值。

```
Region: REGION_ID
Image:
  Os: alinux2
HeadNode:
  InstanceType: c5.2xlarge
  Ssh:
    KeyName: YOUR_SSH_KEY
  Iam:
    S3Access:
      - BucketName: amzn-s3-demo-bucket
    AdditionalIamPolicies:
      - Policy: arn:aws:iam::ACCOUNT_ID:policy/RunInstancesCapacityReservation
```

```
  ## This post-install script is executed after the node is configured.
  ## It is used to install scripts at boot time and specific configurations
  ## In the script below we are overriding the calls to RunInstance to force
  ## the provisioning of our my-queue partition to go through
  ## the On-Demand Capacity Reservation
  CustomActions:
    OnNodeConfigured:
      Script: s3://amzn-s3-demo-bucket/postinstall.sh
  Networking:
    SubnetId: YOUR_PUBLIC_SUBNET_IN_TARGET_AZ

Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: my-queue
      ComputeResources:
        - MinCount: 0
          MaxCount: 100
          InstanceType: p4d.24xlarge
          Name: my-compute-resource
          Efa:
            Enabled: true
      Networking:
        ## PlacementGroup:
        ##   Enabled: true ## Keep PG disabled if using targeted ODCR
        SubnetIds:
          - YOUR_PRIVATE_SUBNET_IN_TARGET_AZ
```

5. 创建集群。

   使用以下命令来创建集群。*cluster-config.yaml*替换为配置文件名、*cluster-dl*集群*REGION_ID*名称和区域 ID（例如 us-east-1）。

```
$ pcluster create-cluster --cluster-configuration cluster-config.yaml --cluster-
name cluster-dl --region REGION_ID
```

   创建集群后，安装后脚本将在头节点中运行。该脚本将会创建 `run_instances_overrides.json` 文件并覆盖对 `RunInstances` 的调用，以将分区强制预置为完成按需容量预留。

   负责集群扩展的 AWS ParallelCluster 守护程序会自动将此配置用于启动的新实例。要确认是否将指定的参数用于预置实例，您可以查看以下日志文件：

- /var/log/parallelcluster/clustermgtd（对于静态容量 - MinCount > 0）
- /var/log/parallelcluster/slurm_resume.log（对于动态容量）

如果参数正确，您会发现包含以下内容的日志条目。

```
Found RunInstances parameters override. Launching instances with: <parameters_list>
```

更新 RunInstances 覆盖

您可以随时更新生成的 JSON 配置，而无需停止计算实例集。应用更改后，所有新实例都将使用更新的配置启动。如果您需要将更新的配置应用于正在运行的节点，请通过强制终止实例 AWS ParallelCluster 来回收节点，然后等待替换这些节点。为此，您可以从 Amazon EC2 控制台终止实例 AWS CLI，或者通过设置 Slurm 处于DOWN或DRAIN状态的节点。

使用以下命令设置 Slurm 节点到DOWN或DRAIN。

```
$ scontrol update nodename=my-queue-dy-my-compute-resource-1 state=down
 reason=your_reason
scontrol update nodename=my-queue-dy-my-compute-resource-1 state=drain
 reason=your_reason
```

使用容量块（CB）启动实例

AWS ParallelCluster 支持 Machine Learn ing 的按需容量预留 (ODCR) 和容量块 (CB)。与 ODCR 不同，CB 可以有未来的开始时间，并且有时间限制。有关使用 ODCR 启动的更多信息，请参阅使用按需容量预留（ODCR）启动实例。

将 CB 与 AWS ParallelCluster

要将您的新集群或现有集群配置为使用 CB，您的 AWS 账户中首先需要有一个有效的 CB。您可以按照官方文档使用 AWS Management Console AWS Command Line Interface、或 SDK 查找和购买可用的 CB。获得有效的 CB 后，您可以在 AWS ParallelCluster 配置文件中设置 CB Amazon 资源名称 (ARN) 和相关参数。有关更多信息，请参阅查找和购买容量块（CB）

集群配置中的 CB

要为特定队列使用 CB，需要使用 `CapacityReservationId` 参数。将此参数配置为现有的 CB ID。您可以从用于创建 CB 的 AWS Management Console AWS CLI、或 SDK 中获取 CB ARN。

您必须为要在其中使用 CB 的队列设置 `CapacityType = CAPACITY_BLOCK`。将其设置为计算资源的 `InstanceType`（与 CB 的 Amazon Elastic Compute Cloud 实例类型相同）。

在计算资源级别指定 `CapacityReservationId` 时，`InstanceType` 是可选的，因为它将自动从预留中进行检索。

当使用 `CapacityType = CAPACITY_BLOCK` 时，`MaxCount` 必须等于 `MinCount` 且大于 0，因为作为 CB 预留一部分的所有实例都是作为静态节点管理的。

在创建集群时，头节点会等待所有静态节点准备就绪，然后再发出集群创建成功的信号。但是，在使用 `CapacityType = CAPACITY_BLOCK` 时，作为计算资源关联的一部分的节点将不在此检查之列。即使并非所有配置都处于活动状态，也会创建集群。

以下配置文件片段显示了在 AWS ParallelCluster 配置文件中启用的必需参数。

```
SlurmQueues:
 - Name: string
   CapacityType: CAPACITY_BLOCK
   ComputeResources:
   - Name: string
     InstanceType: String (EC2 Instance type of the CB)
     MinCount: integer (<= total capacity of the CB)
     MaxCount: integer (equal to MinCount)
     CapacityReservationTarget:
        CapacityReservationId: String (CB id)
```

如何 AWS ParallelCluster 使用容量块 (CB)

AWS ParallelCluster 以一种特殊的方式管理与之关联的静态节点。 AWS ParallelCluster 即使 CB 尚未激活，也会创建集群，并且一旦 CB 处于活动状态，实例就会自动启动。

这些区域有：Slurm 与计算资源相对应、与之关联但尚未处于活动状态的节点将一直处于维护状态，直到它们到达 CB 开始时间。Slurm 节点保持预留/维护状态并与 slurm 管理员用户关联。这意味着这些节点可以接受作业，但作业一直处于 `pending` 状态，直至移除预留。

AWS ParallelCluster 自动更新 Slurm 保留并将相关的 CB 节点置于维护状态（对应于 CB 状态）。当 CB 处于活动状态时，Slurm 预留被移除，节点启动并可用于待处理的任务或提交的新作业。

当到达 CB 结束时间时，当 CB 不再处于活动状态并且实例终止时，节点将移回 reservation/maintenance state. It's up to users to resubmit/requeue the jobs to a new queue/compute-resource。

AMI 修补和亚马逊 EC2 实例更换

为确保所有动态启动的集群计算节点的行为方式一致，请 AWS ParallelCluster 禁用集群实例自动操作系统更新。此外，还会为每个版本 AWS ParallelCluster 及其关联 AWS ParallelCluster AMIs 的 CLI 构建一组特定的。这组特定的版本 AMIs 保持不变，只有它们所针对的 AWS ParallelCluster 版本支持。AWS ParallelCluster AMIs对于已发布的版本未更新。

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

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

```
$ pcluster version
$ pcluster list-official-images
```

AWS ParallelCluster 头节点是一个静态实例，你可以手动更新它。从 3.0.0 AWS ParallelCluster 版开始，完全支持重启和重启头节点。

如果您的实例具有临时实例存储，则必须记得在手动更新之前保存实例存储数据。有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的 *HeadNode LocalStorage* ???//*EphemeralVolume*???集群配置和带有实例存储卷的实例类型。

计算节点是临时实例。默认情况下，只能从头节点访问它们。从 AWS ParallelCluster 版本 3.0.0 开始，在使用以下命令停止计算队列后，您可以通过修改 `Scheduling`//`SlurmQueuesImage`/`CustomAmi`参数并运行`pcluster update-cluster`命令来更新与计算实例关联的 AMI：`pcluster update-compute-fleet`

```
$ pcluster update-compute-fleet-status --status STOP_REQUESTED
```

可以使用以下方法之一为计算节点自动创建更新的自定义 AMI：

- 将 `pcluster build-image` 命令与更新的 `Build`/`ParentImage` 结合使用。
- 使用 `Build`/`UpdateOsPackages`/`Enabled`:`true` 运行构建。

头节点实例更新或替换

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

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

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

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

保存临时驱动器中的数据

从 AWS ParallelCluster 版本 3.0.0 开始，每种实例类型都完全支持头节点重启和重启。但如果实例有临时驱动器，其数据将会丢失。在重启头节点之前，请按照以下步骤保留数据。

要检查是否有需要保留的数据，请查看 `EphemeralVolume`/`MountDir` 文件夹（默认情况下为 `/scratch`）中的内容。

您可以将数据传输到根卷或连接到集群的共享存储系统，例如 Amazon FSx、Amazon EFS 或 Amazon EBS。请注意，将数据传输到远程存储可能会产生额外费用。

保存数据后，继续停止和启动集群的头节点。

停止和启动集群的头节点

1. 确认集群中没有任何正在运行的作业。

   使用时 Slurm 调度器：

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

```
$ pcluster update-compute-fleet --cluster-name cluster-name --status STOP_REQUESTED
{
  "status": "STOP_REQUESTED",
```

```
  ...
}
```

3. 等到计算实例集状态变为 STOPPED：

```
$ pcluster update-compute-fleet --cluster-name cluster-name --status STOP_REQUESTED
{
  "status": "STOPPED",
  ...
}
```

4. 要通过操作系统重启或实例重启进行手动更新，您可以使用 AWS Management Console 或 AWS CLI。下面是使用 AWS CLI的示例。

```
# Retrieve head node instance id
$ pcluster describe-cluster --cluster-name cluster-name --status STOP_REQUESTED
{
  "headNode": {
  "instanceId": "i-1234567890abcdef0",
  ...
},
  ...
}
# stop and start the instance
$ aws ec2 stop-instances --instance-ids 1234567890abcdef0
{
  "StoppingInstances": [
    {
      "CurrentState": {
        "Name": "stopping"
        ...
      },
      "InstanceId": "i-1234567890abcdef0",
      "PreviousState": {
        "Name": "running"
        ...
      }
    }
  ]
}
$ aws ec2 start-instances --instance-ids 1234567890abcdef0
{
  "StartingInstances": [
```

```
    {
      "CurrentState": {
        "Name": "pending"
        ...
      },
      "InstanceId": "i-1234567890abcdef0",
      "PreviousState": {
        "Name": "stopped"
        ...
      }
    }
  ]
}
```

5. 启动集群计算实例集：

```
$ pcluster update-compute-fleet --cluster-name cluster-name --status
 START_REQUESTED
{
  "status": "START_REQUESTED",
  ...
}
```

操作系统

AWS ParallelCluster 支持亚马逊 Linux 2、亚马逊 Linux 2023、Ubuntu24.04、Ubuntu 22.04、Ubuntu 20.04、红帽企业 Linux 8 (RHEL8)、Rocky 8、红帽企业 Linux 9 () 和 Rocky 9。RHEL9
AWS ParallelCluster AMIs 为特定操作系统预先构建，有关 AMIs提供的更多详细信息， AWS ParallelCluster 请参阅。Image 部分

操作系统注意事项

Ubuntu 22.04 和 Ubuntu 24.04

Ubuntu 22.04 和 Ubuntu 24.04 需要更安全的密钥才能使用 SSH，并且默认情况下不支持 RSA 密钥。请生成 ed25519 密钥并使用该密钥进行集群创建。

Ubuntu 22.04 无法更新到最新的内核，因为该内核没有 FSx 客户端。

RHEL 8

RedHat 从版本 3.6.0 开始添加企业 Linux 8.7 (rhel8)。 AWS ParallelCluster 如果您将集群配置为使用 rhel8，则任何实例类型的按需成本都高于将集群配置为使用支持的其它操作系统时的按需成本。

有关定价的更多信息，请参阅按需定价和 Amazon Elastic Compute Cloud 上如何提供和定价 Red Hat Enterprise Linux？。

Rocky 8

AWS ParallelCluster 3.8.0 支持 Rocky Linux 8，但预先构建的 Rocky Linux 8 AMIs（适用于 x86 和 ARM 架构）不可用。 AWS ParallelCluster 3.8.0 支持使用属性使用自定义 AMIs 方式使用 Rocky Linux 8 创建集群。CustomAmi有关构建自定义的更多信息 AMIs，请参阅AWS ParallelCluster AMI 自定义。

要从基本 Rocky Linux 8 AMI 构建自定义 AMI，你可以考虑订阅 Mark AWS etplace 上线的 Rocky Linux 8 AMIs。请务必在 Mark AWS etplace AMIs 上查看 Rocky Linux 8 的定价和订阅费用。或者，你也可以使用官方的 Rocky Linux 8 AMIs 作为你的基础 AMI。

Rocky9

AWS ParallelCluster 3.9.0 支持 Rocky Linux 9，但预先构建的 Rocky Linux 9 AMIs（适用于 x86 和 ARM 架构）不可用。 AWS ParallelCluster 3.9.0 支持使用属性使用自定义 AMIs 方式使用 Rocky Linux 9 创建集群。CustomAmi有关构建自定义的更多信息 AMIs，请参阅 AWS ParallelCluster AMI 自定义。要从基本 Rocky Linux 9 AMI 构建自定义 AMI，你也可以使用官方的 Rocky Linux 9 AMIs 作为基础 AMI。如果基础 AMI 没有最新的内核，则自定义 Rocky Linux 9 AMI 的构建可能会失败。要在构建 AMI 之前升级内核，请执行以下操作：

- 通过 https://rockylinux.org/cloud-images/，使用 rocky9 AMI id 启动实例
- 通过 ssh 登录实例并运行以下命令：`sudo yum -y update`
- 从实例创建映像以用作 `ParentImage`

的参考 AWS ParallelCluster

主题

- AWS ParallelCluster CLI 命令
- 配置文件
- AWS ParallelCluster API 参考
- AWS ParallelCluster Python 库 API

AWS ParallelCluster CLI 命令

`pcluster`是主要 AWS ParallelCluster CLI 命令。您可以使用`pcluster`在和中启动和管理 HPC 集群来创建和管理自定义 AMI 映像。 AWS Cloud

`pcluster3-config-converter`用于将 AWS ParallelCluster 版本 2 格式的集群配置转换为 AWS ParallelCluster 版本 3 格式。

```
pcluster [-h] ( build-image | configure |
                create-cluster | dcv-connect |
                delete-cluster | delete-cluster-instances | delete-image |
                describe-cluster | describe-cluster-instances |
                describe-compute-fleet | describe-image |
                export-cluster-logs | export-image-logs |
                get-cluster-log-events | get-cluster-stack-events |
                get-image-log-events | get-image-stack-events |
                list-cluster-log-streams | list-clusters |
                list-images | list-image-log-streams | list-official-images |
                ssh | update-cluster |
                update-compute-fleet | version ) ...
pcluster3-config-converter [-h] [-t CLUSTER_TEMPLATE]
                [-c CONFIG_FILE]
                [--force-convert]
                [-o OUTPUT_FILE]
```

主题

- pcluster
- pcluster3-config-converter

pcluster

pcluster是主要 AWS ParallelCluster CLI 命令。使用 pcluster 可以在 AWS Cloud中启动和管理 HPC 集群。

pcluster 将您的命令的日志写入 /home/user/.parallelcluster/ 下的 pcluster.log.# 文件中。有关更多信息，请参阅 pcluster CLI 日志。

要使用 pcluster，您必须拥有具有运行该命令所需的权限的 IAM 角色。

```
pcluster [-h]
```

参数

pcluster ***command***

可能的选择：build-image| configure| | create-cluster| dcv-connect| delete-cluster| delete-cluster-instances| | delete-image| describe-cluster| describe-cluster-instances| | describe-compute-fleet| describe-image| export-cluster-logs| export-image-logs| | get-cluster-log-events| get-cluster-stack-events| get-image-log-events| | get-image-stack-events| list-clusters| list-cluster-log-streams| list-images| list-image-log-streamslist-official-images| ssh| update-cluster| update-compute-fleet| version

子命令：

主题

- pcluster build-image
- pcluster configure
- pcluster create-cluster
- pcluster dcv-connect
- pcluster delete-cluster
- pcluster delete-cluster-instances
- pcluster delete-image
- pcluster describe-cluster

- pcluster describe-cluster-instances
- pcluster describe-compute-fleet
- pcluster describe-image
- pcluster export-cluster-logs
- pcluster export-image-logs
- pcluster get-cluster-log-events
- pcluster get-cluster-stack-events
- pcluster get-image-log-events
- pcluster get-image-stack-events
- pcluster list-clusters
- pcluster list-cluster-log-streams
- pcluster list-images
- pcluster list-image-log-streams
- pcluster list-official-images
- pcluster ssh
- pcluster update-cluster
- pcluster update-compute-fleet
- pcluster version

pcluster build-image

在指定区域创建自定义 AWS ParallelCluster 镜像。

```
pcluster build-image [-h]
                 --image-configuration IMAGE_CONFIGURATION
                 --image-id IMAGE_ID
                [--debug]
                [--dryrun DRYRUN]
                [--query QUERY]
                [--region REGION]
                [--rollback-on-failure ROLLBACK_ON_FAILURE]
                [--suppress-validators SUPPRESS_VALIDATORS [SUPPRESS_VALIDATORS ...]]
                [--validation-failure-level {INFO,WARNING,ERROR}]
```

命名的参数

-h, --help

显示 pcluster build-image 的帮助文本。

--image-configuration, -c *IMAGE_CONFIGURATION*

将映像配置文件指定为 YAML 文档。

--image-id, -i *IMAGE_ID*

指定将要构建的映像的 ID。

--debug

启用调试登入

--dryrun *DRYRUN*

当为 true 时，该命令执行验证而不创建任何资源。您可以使用此参数来验证映像配置。（默认值为 false。）

--query *QUERY*

JMESPath 要对输出执行的查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用图像配置文件中的 “区域” 设置、AWS_DEFAULT_REGION环境变量、文件[default]部分中的region~/.aws/config设置或--region参数来指定。

--rollback-on-failure *ROLLBACK_ON_FAILURE*

当为 true 时，会在失败时自动启动映像堆栈回滚。（默认值为 false。）

--suppress-validators *SUPPRESS_VALIDATORS* [*SUPPRESS_VALIDATORS ...*]

标识一个或多个要禁止的配置验证器。

格式：(ALL|type:[A-Za-z0-9]+)

--validation-failure-level {INFO,WARNING,ERROR}

指定将导致创建失败的最低验证级别。（默认值为 ERROR。）

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster build-image --image-configuration image-config.yaml --image-id custom-alinux2-image
{
  "image": {
    "imageId": "custom-alinux2-image",
    "imageBuildStatus": "BUILD_IN_PROGRESS",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/custom-alinux2-image/1234abcd-56ef-78gh-90ij-abcd1234efgh",
    "region": "us-east-1",
    "version": "3.1.2"
  }
}
```

> **Warning**
>
> `pcluster build-image` 使用默认 VPC。如果默认 VPC 已被删除（可能使用 AWS Control Tower 或 AWS 着陆区域），则必须在映像配置文件中指定子网 ID。有关更多信息，请参阅 SubnetId。

`pcluster configure`

启动 AWS ParallelCluster 版本 3 的交互式配置向导。有关更多信息，请参阅 使用 AWS ParallelCluster 命令行界面配置和创建集群。

```
pcluster configure [-h]
                 --config CONFIG
                [--debug]
                [--region REGION]
```

命名的参数

`-h, --help`

显示 `pcluster configure` 的帮助文本。

`--config CONFIG`

生成的配置文件的输出路径。

--debug

启用调试登入

--region, -r *REGION*

指定 AWS 区域 要使用的。必须使用映像配置文件中的 Region 设置、AWS_DEFAULT_REGION 环境变量、~/.aws/config 文件 [default] 部分中的 region 设置或 --region 参数指定区域。

pcluster create-cluster

创建集 AWS ParallelCluster 群。

```
pcluster create-cluster [-h]
                 --cluster-configuration CLUSTER_CONFIGURATION
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--dryrun DRYRUN]
                [--query QUERY]
                [--region REGION]
                [--rollback-on-failure ROLLBACK_ON_FAILURE]
                [--suppress-validators SUPPRESS_VALIDATORS [SUPPRESS_VALIDATORS ...]]

                [--validation-failure-level {INFO,WARNING,ERROR}]
```

命名的参数

-h, --help

显示 pcluster create-cluster 的帮助文本。

--cluster-configuration, -c *CLUSTER_CONFIGURATION*

指定 YAML 集群配置文件。

--cluster-name, -n *CLUSTER_NAME*

指定要创建的集群的名称。

名称必须以字母字符开头。名称最多可以包含 60 个字符。如果启用 Slurm 会计，则名称最多可包含 40 个字符。

有效字符：A-Z、a-z、0-9 和 -（连字符）。

--debug

启用调试日志记录。

--dryrun ***DRYRUN***

当为 true 时，该命令执行验证而不创建任何资源。您可以使用此参数来验证集群配置。（默认值为 false。）

--query ***QUERY***

指定要对输出执行的 JMESPath 查询。

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

指定 AWS 区域 要使用的。 AWS 区域 必须使用群集配置文件中的Region设置、AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分的region设置或--region参数来指定。

--rollback-on-failure ***ROLLBACK_ON_FAILURE***

当为 true 时，会在失败时自动启动集群堆栈回滚。（默认值为 true。）

--suppress-validators ***SUPPRESS_VALIDATORS*** **[*SUPPRESS_VALIDATORS* ...]**

标识一个或多个要禁止的配置验证器。

格式：(ALL|type:[A-Za-z0-9]+)

--validation-failure-level {INFO,WARNING,ERROR}

指定将导致创建失败的最低验证级别。（默认值为 ERROR。）

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster create-cluster -c cluster-config.yaml -n cluster-v3
{
  "cluster": {
    "clusterName": "cluster-v3",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster-v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
    "region": "us-east-1",
```

```
    "version": "3.1.4",
    "clusterStatus": "CREATE_IN_PROGRESS"
  }
}
```

pcluster dcv-connect

允许使用 Amazon DCV 通过交互式会话连接到头节点。

```
pcluster dcv-connect [-h]
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--key-path KEY_PATH]
                [--login-node-ip LOGIN_NODE_IP]
                [--region REGION]
                [--show-url]
```

命名的参数

-h, --help

显示 `pcluster dcv-connect` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--key-path *KEY_PATH*

指定用于连接的 SSH 密钥的路径。

--login-node-ip

指定集群中登录节点的公有或私有 IP 地址。使用该参数可以连接到集群中已启用 DCV 的登录节点。

> **Note**
>
> 此参数已在 3.11.0 AWS ParallelCluster 版本中添加。

--region, -r *REGION*

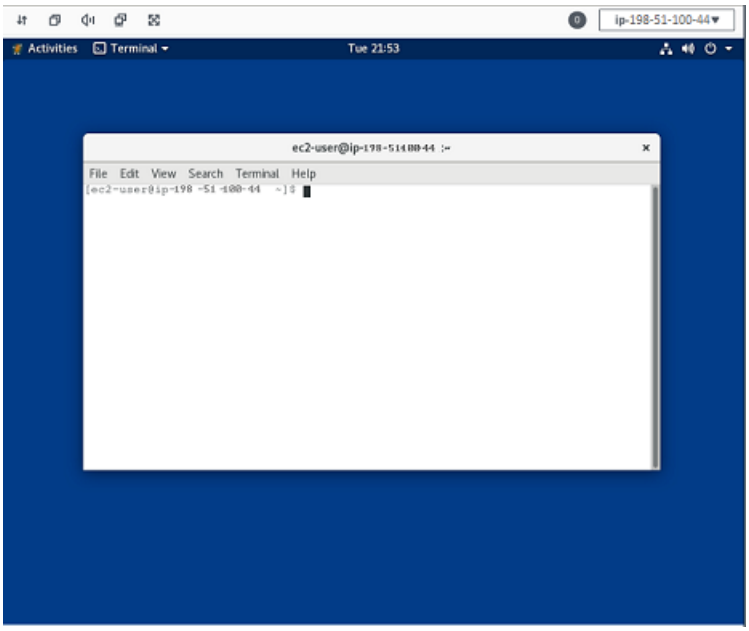
指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

--show-url

打印将用于 DCV 连接的 URL 并退出。

使用 AWS ParallelCluster 版本 3.11 的示例

```
$ pcluster dcv-connect -n cluster-3Dcv --login-node-ip 198.51.100.44 -r us-east-1 --
key-path /home/user/.ssh/key.pem
```



pcluster delete-cluster

开始删除集群。

```
pcluster delete-cluster [-h]
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster delete-cluster` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。必须使用 `AWS_DEFAULT_REGION` 环境变量、`~/.aws/config` 文件 `[default]` 部分中的 `region` 设置或 `--region` 参数指定区域。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster delete-cluster -n cluster-v3
   {
  "cluster": {
    "clusterName": "cluster-v3",
    "cloudformationStackStatus": "DELETE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster-v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
    "region": "us-east-1",
    "version": "3.1.4",
    "clusterStatus": "DELETE_IN_PROGRESS"
  }
}
```

pcluster delete-cluster-instances

开始强制终止所有集群计算节点。这不适用于集 AWS Batch 群。

```
pcluster delete-cluster-instances [-h]
                  --cluster-name CLUSTER_NAME
```

```
                [--debug]
                [--force FORCE]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 pcluster delete-cluster-instances 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--force *FORCE*

当为 true 时，通过忽略验证错误强制删除。（默认值为 false。）

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

```
$ pcluster delete-cluster-instances -n cluster-v3
```

pcluster delete-image

开始删除自定义 AWS ParallelCluster 镜像。

```
pcluster delete-image [-h]
                 --image-id IMAGE_ID
                [--debug]
                [--force FORCE]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 pcluster delete-image 的帮助文本。

--image-id, -i *IMAGE_ID*

指定将要删除的映像的 ID。

--debug

启用调试日志记录。

--force *FORCE*

当为 true 时，如果有使用 AMI 的实例或者共享了 AMI，则强制删除。（默认值为 false。）

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster delete-image --image-id custom-alinux2-image
{
  "image": {
    "imageId": "custom-alinux2-image",
    "imageBuildStatus": "DELETE_IN_PROGRESS",
    "region": "us-east-1",
    "version": "3.1.4"
  }
}
```

pcluster describe-cluster

获取有关集群的详细信息

```
pcluster describe-cluster [-h]
                  --cluster-name CLUSTER_NAME
```

```
                [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 pcluster describe-cluster 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

描述集群详细信息：

```
$ pcluster describe-cluster -n cluster-v3
{
  "creationTime": "2022-07-12T17:19:16.101Z",
  "headNode": {
    "launchTime": "2022-07-12T17:22:21.000Z",
    "instanceId": "i-1234567890abcdef0",
    "publicIpAddress": "198.51.100.44",
    "instanceType": "t2.micro",
    "state": "running",
    "privateIpAddress": "192.0.2.0.196"
  },
  "loginNodes": [
      {
          "status": "active",
```

```
            "poolName": "pool1",
            "address": "cluster-v3-eMr9BYRKZVDa-e5bb34f40b24f51d.elb.us-
east-1.amazonaws.com",
            "scheme": "internet-facing",
            "healthyNodes": 1,
            "unhealthyNodes": 0
        },
        {
            "status": "active",
            "poolName": "pool2",
            "address": "cluster-v3-PaQ7GgC27sic-aba10c890247b36b.elb.us-
east-1.amazonaws.com",
            "scheme": "internet-facing",
            "healthyNodes": 1,
            "unhealthyNodes": 0
        }
    ],
    "version": "3.1.4",
    "clusterConfiguration": {
      "url": "https://parallelcluster-e5ca74255d6c3886-v1-do-not-delete..."
    },
    "tags": [
      {
        "value": "3.11",
        "key": "parallelcluster:version"
      }
    ],
    "cloudFormationStackStatus": "CREATE_COMPLETE",
    "clusterName": "cluster-v3",
    "computeFleetStatus": "RUNNING",
    "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster-v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
    "lastUpdatedTime": "2022-07-12T17:19:16.101Z",
    "region": "us-east-1",
    "clusterStatus": "CREATE_COMPLETE"
}
```

使用 `describe-cluster` 检索集群配置：

```
$ curl -o - $(pcluster describe-cluster -n cluster-v3 --query clusterConfiguration.url
 | xargs echo)
Region: us-east-1
Image:
```

```
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-abcdef01234567890
  Ssh:
    KeyName: adpc
  Iam:
    S3Access:
      - BucketName: cluster-v3-bucket
        KeyName: logs
        EnableWriteAccess: true
Scheduling:
  Scheduler: slurm
  SlurmQueues:
  - Name: queue1
    ComputeResources:
    - Name: t2micro
      InstanceType: t2.micro
      MinCount: 0
      MaxCount: 10
    Networking:
      SubnetIds:
      - subnet-021345abcdef6789
```

pcluster describe-cluster-instances

描述集群中的实例。

```
pcluster describe-cluster-instances [-h]
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--next-token NEXT_TOKEN]
                [--node-type {HeadNode,ComputeNode,LoginNode}]
                [--query QUERY]
                [--queue-name QUEUE_NAME]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster describe-cluster-instances` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--node-type {HeadNode,ComputeNode,LoginNode}

指定要列出的节点类型。支持的值为 HeadNode、ComputeNode 和 LoginNode。如果未指定此参数，则描述 HeadNode、ComputeNode 和 LoginNode 实例。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--queue-name *QUEUE_NAME*

指定要列出的队列的名称。如果未指定此参数，则描述所有队列中的实例。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster describe-cluster-instances -n cluster-v3
{
  "instances": [
    {
      "launchTime": "2022-07-12T17:22:21.000Z",
      "instanceId": "i-1234567890abcdef0",
      "publicIpAddress": "198.51.100.44",
      "instanceType": "t2.micro",
      "state": "running",
      "nodeType": "HeadNode",
      "privateIpAddress": "192.0.2.0.196"
    },
    {
      "launchTime": "2022-07-12T17:37:42.000Z",
```

```
      "instanceId": "i-021345abcdef6789",
      "queueName": "queue1",
      "publicIpAddress": "198.51.100.44",
      "instanceType": "t2.micro",
      "state": "pending",
      "nodeType": "ComputeNode",
      "privateIpAddress": "192.0.2.0.196"
    },
    {
        "launchTime": "2022-07-12T17:37:42.000Z",
        "instanceId": "i-021345abcdef6789",
        "poolName": "pool1",
        "publicIpAddress": "198.51.100.44",
        "instanceType": "t2.micro",
        "state": "pending",
        "nodeType": "loginNode",
        "privateIpAddress": "192.0.2.0.196"
    }
  ]
}
```

pcluster describe-compute-fleet

描述计算实例集的状态。

```
pcluster describe-compute-fleet [-h]
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster describe-compute-fleet` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster describe-compute-fleet -n pcluster-v3
{
  "status": "RUNNING",
  "lastStatusUpdatedTime": "2022-07-12T17:24:26.000Z"
}
```

pcluster describe-image

获取有关映像的详细信息

```
pcluster describe-image [-h]
                 --image-id IMAGE_ID
                [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster describe-image` 的帮助文本。

--image-id, -i *IMAGE_ID*

指定映像的 ID。

--debug

启用调试日志记录。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster describe-image --image-id custom-alinux2-image
{
  "imageConfiguration: {
    "url": "https://parallelcluster-1234abcd5678-v1-do-not-delete.../configs/image-
config.yaml"
  },
  "imageId": "custom-alinux2-image",
  "creationTime": "2022-04-05T20:23:07.000Z"
  "imageBuildStatus": "BUILD_COMPLETE",
  "region": "us-east-1",
  "ec2AmiInfo": {
    "amiName": "custom-alinux2-image 2022-04-05T19-55-22.518Z",
    "amiId": "ami-1234abcd5678efgh",
    "description": "AWS ParallelCluster AMI for alinux2,
 kernel-4.14.268-205.500.amzn2.x86_64, lustre-2.10.8-5.amzn2.x86_64,
 efa-1.14.2-1.amzn2.x86_64, dcv-2021.3.11591-1.el7.x86_64, slurm-21-08-6-1",
    "state": "AVAILABLE",
  "tags": [
      {
        "value": "arn:aws:imagebuilder:us-east-1:123456789012:image/
parallelclusterimage-custom-alinux2-image/3.1.2/1",
        "key": "Ec2ImageBuilderArn"
      },
      {
        "value": "parallelcluster-1234abcd5678efgh-v1-do-not-delete",
        "key": "parallelcluster:amzn-s3-demo-bucket"
      },
      {
        "value": "custom-alinux2-image",
        "key": "parallelcluster:image_name"
      },
      {
        "value": "available",
        "key": "parallelcluster:build_status"
      },
      {
```

```
        "value": "s3://amzn-s3-demo-bucket/parallelcluster/3.1.2/images/custom-alinux2-
image-1234abcd5678efgh/configs/image-config.yaml",
        "key": "parallelcluster:build_config"
      },
      {
        "value": "Amazon EC2 Image Builder",
        "key": "CreatedBy"
      },
      {
        "value": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/imagebuilder/
ParallelClusterImage-custom-alinux2-image",
        "key": "parallelcluster:build_log"
      },
      {
        "value": "4.14.268-205.500.amzn2.x86_64",
        "key": "parallelcluster:kernel_version"
      },
      {
        "value": "arn:aws:imagebuilder:us-east-1:444455556666:image/amazon-linux-2-
x86/2022.3.16/1",
        "key": "parallelcluster:parent_image"
      },
      {
        "value": "3.1.2",
        "key": "parallelcluster:version"
      },
      {
        "value": "0.5.14",
        "key": "parallelcluster:munge_version"
      },
      {
        "value": "21-08-6-1",
        "key": "parallelcluster:slurm_version"
      },
      {
        "value": "2021.3.11591-1.el7.x86_64",
        "key": "parallelcluster:dcv_version"
      },
      {
        "value": "alinux2-image",
        "key": "parallelcluster:image_id"
      },
      {
        "value": "3.2.3",
```

```
        "key": "parallelcluster:pmix_version"
      },
      {
        "value": "parallelcluster/3.13.2/images/alinux2-image-abcd1234efgh56781234",
        "key": "parallelcluster:s3_image_dir"
      },
      {
        "value": "1.14.2-1.amzn2.x86_64",
        "key": "parallelcluster:efa_version"
      },
      {
        "value": "alinux2",
        "key": "parallelcluster:os"
      },
      {
        "value": "aws-parallelcluster-cookbook-3.1.2",
        "key": "parallelcluster:bootstrap_file"
      },
      {
        "value": "1.8.23-10.amzn2.1.x86_64",
        "key": "parallelcluster:sudo_version"
      },
      {
        "value": "2.10.8-5.amzn2.x86_64",
        "key": "parallelcluster:lustre_version"
      }
    ],
    "architecture": "x86_64"
  },
  "version": "3.1.2"
}
```

pcluster export-cluster-logs

通过 Amazon S3 存储桶，将集群的日志导出到本地 `tar.gz` 存档。

```
pcluster export-cluster-logs [-h]
                 --cluster-name CLUSTER_NAME
                [--bucket BUCKET_NAME]
                [--bucket-prefix BUCKET_PREFIX]
                [--debug]
                [--end-time END_TIME]
                [--filters FILTER [FILTER ...]]
```

```
                [--keep-s3-objects KEEP_S3_OBJECTS]
                [--output-file OUTPUT_FILE]
                [--region REGION]
                [--start-time START_TIME]
```

> **Note**
>
> 该`export-cluster-logs`命令会等待 CloudWatch Logs 完成日志的导出，因此预计会有一段时间没有任何输出。

命名的参数

`-h, --help`

显示 `pcluster export-cluster-logs` 的帮助文本。

`--bucket` *`BUCKET_NAME`*

指定要将集群日志数据导出到的 Amazon S3 存储桶的名称。它必须与集群位于相同的区域。

> **Note**
>
> - 您必须向 Amazon S3 存储桶策略添加权限才能授予 CloudWatch 访问权限。有关更多信息，请参阅CloudWatch 日志用户指南中的对 Amazon S3 存储桶设置权限。
> - 从 3.12.0 AWS ParallelCluster 版开始，该**`--bucket`**选项是可选的。如果未指定该选项，则要么使用 AWS ParallelCluster 区域默认存储桶 (`parallelcluster-hash-v1-DO-NOT-DELETE`)，要么如果在集群配置中指定了指向`CustomS3Bucket`的 Amazon S3 存储桶，则将使用该存储桶。如果您未指定该`--bucket`选项而使用默认 AWS ParallelCluster 存储桶，则无法将日志导出到该`parallelcluster/`文件夹，因为该文件夹是保留给内部使用的受保护文件夹。

> **Important**
>
> 如果使用 AWS ParallelCluster 默认存储桶，pcluster 将负责配置存储桶策略。如果您自定义了存储桶策略，然后升级到 AWS ParallelCluster 版本 3.12.0，则存储桶策略将被覆盖，您将需要重新应用更改。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--bucket-prefix *BUCKET_PREFIX*

指定导出的日志数据在 Amazon S3 存储桶中的存储位置的路径。

默认情况下，存储桶前缀为：

```
cluster-name-logs-202209061743.tar.gz
```

*202209061743*是时间`%Y%m%d%H%M`格式的示例。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.12.0 开始，如果您未指定该`--bucket`选项并使用默认 AWS ParallelCluster 存储桶，则无法将日志导出到该`parallelcluster/`文件夹，因为它是保留供内部使用的受保护文件夹。

--debug

启用调试日志记录。

--end-time *END_TIME*

指定用于收集日志事件的时间范围的结束时间，以 ISO 8601 格式（`YYYY-MM-DDThh:mm:ssZ`，例如 `2021-01-01T20:00:00Z`）表示。不包括时间戳等于或晚于该时间的事件。可以省略时间元素（例如分和秒）。默认值为当前时间。

--filters *FILTER* [*FILTER* ...]

为日志指定筛选器。格式：`Name=a,Values=1 Name=b,Values=2,3`。支持的筛选器为：

`private-dns-name`

指定实例私有 DNS 名称的短格式（例如 `ip-10-0-0-101`）。

`node-type`

指定节点类型，此筛选器唯一接受的值是 `HeadNode`。

--keep-s3-objects *KEEP_S3_OBJECTS*

如果为 `true`，则会保留导出到 Amazon S3 的导出对象。（默认值为 `false`。）

--output-file ***OUTPUT_FILE***

指定要将日志存档保存到的文件路径。如果提供此参数，则在本地保存日志。否则会通过输出中返回的 URL 将日志上传到 Amazon S3。默认为上传到 Amazon S3。

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

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

--start-time ***START_TIME***

指定时间范围的开始时间，以 ISO 8601 格式（`YYYY-MM-DDThh:mm:ssZ`，例如`2021-01-01T20:00:00Z`）表示。包括时间戳等于或晚于该时间的日志事件。如果未指定，则默认为集群的创建时间。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster export-cluster-logs --bucket cluster-v3-bucket -n cluster-v3
{
  "url": "https://cluster-v3-bucket..."
}
```

无法检索日志？

如果您无法使用`export-cluster-logs`命令检索日志，请执行以下操作之一：

- 从集群的 CloudWatch 日志组中手动检索日志。
- 如果日志组为空，请通过 SSH 进入群集节点并检索中列出的日志排查节点初始化问题。
- 如果由于集群创建失败而无法访问群集节点，则使用选项重新创建集群`--rollback-on-failure false`并从节点检索日志。

pcluster export-image-logs

通过 Amazon S3 存储桶，将映像生成器的日志导出到本地 `tar.gz` 存档。

```
pcluster export-image-logs [-h]
                 --image-id IMAGE_ID
                [--bucket BUCKET]
```

```
                [--bucket-prefix BUCKET_PREFIX]
                [--debug]
                [--end-time END_TIME]
                [--keep-s3-objects KEEP_S3_OBJECTS]
                [--output-file OUTPUT_FILE]
                [--region REGION]
                [--start-time START_TIME]
```

> **Note**
>
> 该`export-image-logs`命令会等待 CloudWatch Logs 完成日志的导出，因此预计会有一段时间没有任何输出。

命名的参数

`-h, --help`

显示 `pcluster export-image-logs` 的帮助文本。

`--bucket BUCKET_NAME`

指定要将映像构建日志导出到的 Amazon S3 存储桶的名称。它必须与映像位于相同的区域。

> **Note**
>
> - 您必须向 Amazon S3 存储桶策略添加权限才能授予 CloudWatch 访问权限。有关更多信息，请参阅CloudWatch 日志用户指南中的对 Amazon S3 存储桶设置权限。
> - 从 3.12.0 AWS ParallelCluster 版开始，该**`--bucket`**选项是可选的。如果未指定该选项，则要么使用 AWS ParallelCluster 区域默认存储桶 (`parallelcluster-hash-v1-DO-NOT-DELETE`)，要么如果在`CustomS3Bucket`映像配置中指定了，则使用该存储桶。

> **Important**
>
> 如果使用 AWS ParallelCluster 默认存储桶，pcluster 将负责配置存储桶策略。如果您在升级到 AWS ParallelCluster 版本 3.12.0 之前自定义存储桶策略，则存储桶策略将被覆盖，您需要重新应用更改。

--image-id, -i ***IMAGE_ID***

要导出其日志的映像 ID。

--bucket-prefix ***BUCKET_PREFIX***

指定导出的日志数据在 Amazon S3 存储桶中的存储位置的路径。

默认情况下，存储桶前缀为：

```
ami-id-logs-202209061743.tar.gz
```

202209061743 是 `%Y%m%d%H%M` 格式的当前时间。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.12.0 开始，如果您不指定该`--bucket`选项并使用默认 AWS ParallelCluster 存储桶，则无法将日志导出到该`parallelcluster/`文件夹，因为它是保留供内部使用的受保护文件夹。

--debug

启用调试日志记录。

--end-time ***END_TIME***

指定用于收集日志事件的时间范围的结束时间，以 ISO 8601 格式（`YYYY-MM-DDThh:mm:ssZ`，例如 `2021-01-01T20:00:00Z`）表示。不包括时间戳等于或晚于该时间的事件。可以省略时间元素（例如分和秒）。默认值为当前时间。

--keep-s3-objects ***KEEP_S3_OBJECTS***

如果为 `true`，则会保留导出到 Amazon S3 的导出对象。（默认值为 `false`。）

--output-file ***OUTPUT_FILE***

指定要将日志存档保存到的文件路径。如果提供此参数，则在本地保存日志。否则会通过输出中返回的 URL 将日志上传到 Amazon S3。默认为上传到 Amazon S3。

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

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

--start-time ***START_TIME***

指定时间范围的开始时间，以 ISO 8601 格式（`YYYY-MM-DDThh:mm:ssZ`，例如 `2021-01-01T20:00:00Z`）表示。包括时间戳等于或晚于该时间的日志事件。如果未指定，则默认为集群的创建时间。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster export-image-logs --bucket image-v3-bucket --image-id ami-1234abcd5678efgh
{
  "url": "https://image-v3-bucket..."
}
```

pcluster get-cluster-log-events

检索与日志流关联的事件。

```
pcluster get-cluster-log-events [-h]
                 --cluster-name CLUSTER_NAME
                 --log-stream-name LOG_STREAM_NAME
                [--debug]
                [--end-time END_TIME]
                [--limit LIMIT]
                [--next-token NEXT_TOKEN]
                [--query QUERY]
                [--region REGION]
                [--start-from-head START_FROM_HEAD]
                [--start-time START_TIME]
```

命名的参数

-h, --help

显示 `pcluster get-cluster-log-events` 的帮助文本。

--cluster-name, -n ***CLUSTER_NAME***

指定集群的名称。

--log-stream-name ***LOG_STREAM_NAME***

指定日志流的名称。您可以使用 `list-cluster-log-streams` 命令来检索与一个或多个事件关联的日志流。

--debug

启用调试日志记录。

--end-time ***END_TIME***

指定时间范围的结束时间，以 ISO 8601 格式（YYYY-MM-DDThh:mm:ssZ，例如 2021-01-01T20:00:00Z）表示。不包括时间戳等于或晚于该时间的事件。

--limit ***LIMIT***

指定返回的日志事件的最大数量。如果不指定值，则最大值为 1 MB 的响应大小所能容纳的日志事件数量，最多可达 10000 个日志事件。

--next-token ***NEXT_TOKEN***

用于获取下一组结果的令牌。

--query ***QUERY***

指定要对输出执行的 JMESPath 查询。

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

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

--start-from-head ***START_FROM_HEAD***

如果值为 true，则最先返回最早的日志事件。如果值为 false，则最先返回最近的日志事件。（默认值为 false。）

--start-time ***START_TIME***

指定时间范围的开始时间，以 ISO 8601 格式（YYYY-MM-DDThh:mm:ssZ，例如 2021-01-01T20:00:00Z）表示。包括时间戳等于或晚于该时间的事件。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster get-cluster-log-events \
    -c cluster-v3 \
    -r us-east-1 \
    --log-stream-name ip-198-51-100-44.i-1234567890abcdef0.clustermgtd \
    --limit 3
```

```
{
  "nextToken": "f/36966906399261933213029082268132291405859205452101451780/s",
  "prevToken": "b/36966906399239632467830551644990755687586557090595471362/s",
  "events": [
    {
      "message": "2022-07-12 19:16:53,379 - [slurm_plugin.clustermgtd:_maintain_nodes]
 - INFO - Performing node maintenance actions",
      "timestamp": "2022-07-12T19:16:53.379Z"
    },
    {
      "message": "2022-07-12 19:16:53,380 - [slurm_plugin.clustermgtd:_maintain_nodes]
 - INFO - Following nodes are currently in replacement: (x0) []",
      "timestamp": "2022-07-12T19:16:53.380Z"
    },
    {
      "message": "2022-07-12 19:16:53,380 -
 [slurm_plugin.clustermgtd:_terminate_orphaned_instances] - INFO - Checking for
 orphaned instance",
      "timestamp": "2022-07-12T19:16:53.380Z"
    }
  ]
}
```

pcluster get-cluster-stack-events

检索与指定集群的堆栈关联的事件。

> **Note**
>
> 从 3.6.0 版开始， AWS ParallelCluster 使用嵌套堆栈来创建与队列和计算资源关联的资源。`GetClusterStackEvents` API 和 `pcluster get-cluster-stack-events` 命令仅返回集群主堆栈事件。您可以在 CloudFormation 控制台中查看集群堆栈事件，包括与队列和计算资源相关的事件。

```
pcluster get-cluster-stack-events [-h]
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--next-token NEXT_TOKEN]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster get-cluster-stack-events` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster get-cluster-stack-events \
    -n cluster-v3 \
    -r us-east-1 \
    --query "events[0]"
{
  "eventId": "1234abcd-56ef-78gh-90ij-abcd1234efgh",
  "physicalResourceId": "arn:aws:cloudformation:us-east-1:123456789012:stack/cluster-
v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
  "resourceStatus": "CREATE_COMPLETE",
  "stackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/cluster-
v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
  "stackName": "cluster-v3",
  "logicalResourceId": "cluster-v3",
  "resourceType": "AWS::CloudFormation::Stack",
  "timestamp": "2022-07-12T18:29:12.140Z"
}
```

pcluster get-image-log-events

检索与映像构建相关的日志事件。

```
pcluster get-image-log-events [-h]
                 --image-id IMAGE_ID
                 --log-stream-name LOG_STREAM_NAME
                [--debug]
                [--end-time END_TIME]
                [--limit LIMIT]
                [--next-token NEXT_TOKEN]
                [--query QUERY]
                [--region REGION]
                [--start-from-head START_FROM_HEAD]
                [--start-time START_TIME]
```

命名的参数

-h, --help

显示 `pcluster get-image-log-events` 的帮助文本。

--image-id, -i *IMAGE_ID*

指定映像的 ID。

--log-stream-name *LOG_STREAM_NAME*

指定日志流的名称。您可以使用 `list-image-log-streams` 命令来检索与一个或多个事件关联的日志流。

--debug

启用调试日志记录。

--end-time *END_TIME*

指定时间范围的结束时间，以 ISO 8601 格式（`YYYY-MM-DDThh:mm:ssZ`，例如 `2021-01-01T20:00:00Z`）表示。不包括时间戳等于或晚于该时间的事件。

--limit *LIMIT*

指定返回的日志事件的最大数量。如果不指定值，则最大值为 1 MB 的响应大小所能容纳的日志事件数量，最多可达 10000 个日志事件。

--next-token ***NEXT_TOKEN***

用于获取下一组结果的令牌。

--query ***QUERY***

指定要对输出执行的 JMESPath 查询。

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

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

--start-from-head ***START_FROM_HEAD***

如果值为 `true`，则最先返回最早的日志事件。如果值为 `false`，则最先返回最近的日志事件。（默认值为 `false`。）

--start-time ***START_TIME***

指定时间范围的开始时间，以 ISO 8601 格式（`YYYY-MM-DDThh:mm:ssZ`，例如 `2021-01-01T20:00:00Z`）表示。包括时间戳等于或晚于该时间的事件。

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster get-image-log-events --image-id custom-alinux2-image --region us-east-1 --
log-stream-name 3.1.2/1 --limit 3
   {
  "nextToken": "f/36778317771100849897800729464621464113270312017760944178/s",
  "prevToken": "b/36778317766952911290874033560295820514557716777648586800/s",
  "events": [
    {
      "message": "ExecuteBash: FINISHED EXECUTION",
      "timestamp": "2022-04-05T22:13:26.633Z"
    },
    {
      "message": "Document arn:aws:imagebuilder:us-east-1:123456789012:component/
parallelclusterimage-test-1234abcd-56ef-78gh-90ij-abcd1234efgh/3.1.2/1",
      "timestamp": "2022-04-05T22:13:26.741Z"
    },
    {
      "message": "TOE has completed execution successfully",
      "timestamp": "2022-04-05T22:13:26.819Z"
    }
  ]
```

```
}
```

pcluster get-image-stack-events

检索与指定映像构建的堆栈关联的事件。

```
pcluster get-image-stack-events [-h]
                 --image-id IMAGE_ID
                [--debug]
                [--next-token NEXT_TOKEN]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster get-image-stack-events` 的帮助文本。

--image-id, -i *IMAGE_ID*

指定映像的 ID。

--debug

启用调试日志记录。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster get-image-stack-events --image-id custom-alinux2-image --region us-east-1 --query "events[0]"
   {
  "eventId": "ParallelClusterImage-CREATE_IN_PROGRESS-2022-04-05T21:39:24.725Z",
```

```
  "physicalResourceId": "arn:aws:imagebuilder:us-east-1:123456789012:image/
parallelclusterimage-custom-alinux2-image/3.1.2/1",
  "resourceStatus": "CREATE_IN_PROGRESS",
  "resourceStatusReason": "Resource creation Initiated",
  "resourceProperties": "{\"InfrastructureConfigurationArn\":
\"arn:aws:imagebuilder:us-east-1:123456789012:infrastructure-configuration/
parallelclusterimage-1234abcd-56ef-78gh-90ij-abcd1234efgh\",\"ImageRecipeArn
\":\"arn:aws:imagebuilder:us-east-1:123456789012:image-recipe/
parallelclusterimage-custom-alinux2-image/3.1.2\",\"DistributionConfigurationArn
\":\"arn:aws:imagebuilder:us-east-1:123456789012:distribution-
configuration/parallelclusterimage-1234abcd-56ef-78gh-90ij-abcd1234efgh\",
\"EnhancedImageMetadataEnabled\":\"false\",\"Tags\":{\"parallelcluster:image_name\":
\"custom-alinux2-image\",\"parallelcluster:image_id\":\"custom-alinux2-image\"}}",
  "stackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/custom-alinux2-
image/1234abcd-56ef-78gh-90ij-abcd1234efgh",
  "stackName": "custom-alinux2-image",
  "logicalResourceId": "ParallelClusterImage",
  "resourceType": "AWS::ImageBuilder::Image",
  "timestamp": "2022-04-05T21:39:24.725Z"
}
```

pcluster list-clusters

检索现有集群的列表。

```
pcluster list-clusters [-h]
                [--cluster-status {CREATE_IN_PROGRESS,CREATE_FAILED,CREATE_COMPLETE,
                           DELETE_IN_PROGRESS,DELETE_FAILED,UPDATE_IN_PROGRESS,
                           UPDATE_COMPLETE,UPDATE_FAILED}
                           [{CREATE_IN_PROGRESS,CREATE_FAILED,CREATE_COMPLETE,
                             DELETE_IN_PROGRESS,DELETE_FAILED,UPDATE_IN_PROGRESS,
                             UPDATE_COMPLETE,UPDATE_FAILED} ...]]
                [--debug]
                [--next-token NEXT_TOKEN]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster list-clusters` 的帮助文本。

--cluster-status {CREATE_IN_PROGRESS, CREATE_FAILED, CREATE_COMPLETE, DELETE_IN_PROGRESS, DELETE_FAILED, UPDATE_IN_PROGRESS, UPDATE_COMPLETE, UPDATE_FAILED} [{CREATE_IN_PROGRESS, CREATE_FAILED, CREATE_COMPLETE, DELETE_IN_PROGRESS, DELETE_FAILED, UPDATE_IN_PROGRESS, UPDATE_COMPLETE, UPDATE_FAILED} ...]

指定要筛选的集群状态的列表。（默认值为 all。）

--debug

启用调试日志记录。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster list-clusters
{
  "clusters": [
    {
      "clusterName": "cluster-v3",
      "cloudformationStackStatus": "CREATE_COMPLETE",
      "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster-v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
      "region": "us-east-1",
      "version": "3.1.4",
      "clusterStatus": "CREATE_COMPLETE"
    }
  ]
}
```

pcluster list-cluster-log-streams

检索与集群关联的日志流的列表。

```
pcluster list-cluster-log-streams [-h]
                 --cluster-name CLUSTER_NAME
                [--filters FILTERS [FILTERS ...]]
                [--next-token NEXT_TOKEN] [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster list-cluster-log-streams` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--filters *FILTERS* [*FILTERS* ...]

为日志流指定筛选器。格式：`Name=a,Values=1 Name=b,Values=2,3`。支持的筛选器为：

`private-dns-name`

指定实例私有 DNS 名称的短格式（例如 `ip-10-0-0-101`）。

`node-type`

指定节点类型，此筛选器唯一接受的值是 `HeadNode`。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster list-cluster-log-streams \
    -n cluster-v3 \
    -r us-east-1 \
    --query 'logStreams[*].logStreamName'
[
  "ip-172-31-58-205.i-1234567890abcdef0.cfn-init",
  "ip-172-31-58-205.i-1234567890abcdef0.chef-client",
  "ip-172-31-58-205.i-1234567890abcdef0.cloud-init",
  "ip-172-31-58-205.i-1234567890abcdef0.clustermgtd",
  "ip-172-31-58-205.i-1234567890abcdef0.slurmctld",
  "ip-172-31-58-205.i-1234567890abcdef0.supervisord",
  "ip-172-31-58-205.i-1234567890abcdef0.system-messages"
]
```

pcluster list-images

检索现有自定义映像的列表。

```
pcluster list-images [-h]
                 --image-status {AVAILABLE,PENDING,FAILED}
                [--debug]
                [--next-token NEXT_TOKEN]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster list-images` 的帮助文本。

--image-status {AVAILABLE,PENDING,FAILED}

按提供的状态筛选返回的映像。

--debug

启用调试日志记录。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster list-images --image-status AVAILABLE
{
  "images": [
    {
      "imageId": "custom-alinux2-image",
      "imageBuildStatus": "BUILD_COMPLETE",
      "ec2AmiInfo": {
        "amiId": "ami-1234abcd5678efgh"
      },
      "region": "us-east-1",
      "version": "3.1.2"
    }
  ]
}
```

pcluster list-image-log-streams

检索与映像关联的日志流的列表。

```
pcluster list-image-log-streams [-h]
                 --image-id IMAGE_ID
                [--next-token NEXT_TOKEN] [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster list-image-log-streams` 的帮助文本。

--image-id, -i *IMAGE_ID*

指定映像的 ID。

--debug

启用调试日志记录。

--next-token *NEXT_TOKEN*

用于获取下一组结果的令牌。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用`AWS_DEFAULT_REGION`环境变量、`~/.aws/config`文件`[default]`部分中的`region`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster list-image-log-streams --image-id custom-alinux2-image --region us-east-1 --
query 'logStreams[*].logStreamName'
[
  "3.0.0/1",
  "3.1.2/1"
]
```

pcluster list-official-images

描述官方 AWS ParallelCluster AMIs。

```
pcluster list-official-images [-h]
                [--architecture ARCHITECTURE]
                [--debug]
                [--os OS]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster list-official-images` 的帮助文本。

--architecture *ARCHITECTURE*

指定要用于筛选结果的架构。如果未指定此参数，则返回所有架构。

--debug

启用调试日志记录。

--os *OS*

指定要用于筛选结果的操作系统。如果未指定此参数，则返回所有操作系统。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用图像配置文件中的 “区域” 设置、`AWS_DEFAULT_REGION`环境变量、文件`[default]`部分中的`region`~`/.aws/config`设置或`--region`参数来指定。

使用 AWS ParallelCluster 版本 3.1.2 的示例：

```
$ pcluster list-official-images
{
  "images": [
    {
      "amiId": "ami-015cfeb4e0d6306b2",
      "os": "ubuntu2004",
      "name": "aws-parallelcluster-3.1.2-ubuntu-2004-lts-hvm-x86_64-202202261505
 2022-02-26T15-08-34.759Z",
      "version": "3.1.2",
      "architecture": "x86_64"
    },
    {
      "amiId": "ami-036f23237ce49d25b",
      "os": "ubuntu2204",
      "name": "aws-parallelcluster-3.1.2-ubuntu-1804-lts-hvm-x86_64-202202261505
 2022-02-26T15-08-17.558Z",
```

```
      "version": "3.1.2",
      "architecture": "x86_64"
    },
    {
      "amiId": "ami-09e5327e694d89ef4",
      "os": "ubuntu2004",
      "name": "aws-parallelcluster-3.1.2-ubuntu-2004-lts-hvm-arm64-202202261505
 2022-02-26T15-08-45.736Z",
      "version": "3.1.2",
      "architecture": "arm64"
    },
    {
      "amiId": "ami-0b9b0874c35f626ae",
      "os": "alinux2",
      "name": "aws-parallelcluster-3.1.2-amzn2-hvm-x86_64-202202261505
 2022-02-26T15-08-31.311Z",
      "version": "3.1.2",
      "architecture": "x86_64"
    },
    {
      "amiId": "ami-0d0de4f95f56374bc",
      "os": "alinux2",
      "name": "aws-parallelcluster-3.1.2-amzn2-hvm-arm64-202202261505
 2022-02-26T15-08-46.088Z",
      "version": "3.1.2",
      "architecture": "arm64"
    },
    {
      "amiId": "ami-0ebf7bc54b8740dc6",
      "os": "ubuntu2204",
      "name": "aws-parallelcluster-3.1.2-ubuntu-1804-lts-hvm-arm64-202202261505
 2022-02-26T15-08-45.293Z",
      "version": "3.1.2",
      "architecture": "arm64"
    }
  ]
}
```

pcluster ssh

运行预填充了集群用户名和 IP 地址的 ssh 命令。将任意参数附加到 ssh 命令行的结尾。

```
pcluster ssh [-h]
```

```
                 --cluster-name CLUSTER_NAME
                [--debug]
                [--dryrun DRYRUN]
                [--region REGION]
```

命名的参数

-h, --help

显示 pcluster ssh 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定要连接到的集群的名称。

--debug

启用调试日志记录。

--dryrun *DRYRUN*

当为 true 时，打印将要运行的命令行并退出。（默认值为 false。）

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

示例：

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

运行预填充了集群用户名和 IP 地址的 ssh 命令：

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

pcluster update-cluster

更新现有集群以匹配指定配置文件的设置。

```
pcluster update-cluster [-h]
                 --cluster-configuration CLUSTER_CONFIGURATION
                 --cluster-name CLUSTER_NAME
                [--debug]
```

```
                 [--dryrun DRYRUN]
                 [--force-update FORCE_UPDATE]
                 [--query QUERY]
                 [--region REGION]
                 [--suppress-validators SUPPRESS_VALIDATORS [SUPPRESS_VALIDATORS ...]]
                 [--validation-failure-level {INFO,WARNING,ERROR}]
```

命名的参数

-h, --help

显示 pcluster update-cluster 的帮助文本。

--cluster-configuration, -c *CLUSTER_CONFIGURATION*

指定 YAML 集群配置文件。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--debug

启用调试日志记录。

--dryrun *DRYRUN*

当为 true 时，执行验证而不更新集群和创建任何资源。它可用于验证映像配置和更新要求。（默认值为 false。）

--force-update *FORCE_UPDATE*

当为 true 时，通过忽略更新验证错误强制更新。（默认值为 false。）

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用群集配置文件中的Region设置、AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分的region设置或--region参数来指定。

--suppress-validators *SUPPRESS_VALIDATORS* [*SUPPRESS_VALIDATORS ...*]

标识一个或多个要禁止的配置验证器。

格式：(ALL|type:[A-Za-z0-9]+)

--validation-failure-level ***{INFO,WARNING,ERROR}***

指定为更新报告的验证失败级别。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster update-cluster -c cluster-config.yaml -n cluster-v3 -r us-east-1
{
  "cluster": {
    "clusterName": "cluster-v3",
    "cloudformationStackStatus": "UPDATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster-v3/1234abcd-56ef-78gh-90ij-abcd1234efgh",
    "region": "us-east-1",
    "version": "3.1.4",
    "clusterStatus": "UPDATE_IN_PROGRESS"
  },
  "changeSet": [
    {
      "parameter": "HeadNode.Iam.S3Access",
      "requestedValue": {
        "BucketName": "amzn-s3-demo-bucket1",
        "KeyName": "output",
        "EnableWriteAccess": false
      }
    },
    {
      "parameter": "HeadNode.Iam.S3Access",
      "currentValue": {
        "BucketName": "amzn-s3-demo-bucket2",
        "KeyName": "logs",
        "EnableWriteAccess": true
      }
    }
  ]
}
```

pcluster update-compute-fleet

更新集群计算实例集的状态。

```
pcluster update-compute-fleet [-h]
```

```
                 --cluster-name CLUSTER_NAME
                 --status {START_REQUESTED,STOP_REQUESTED,ENABLED,DISABLED}

                [--debug]
                [--query QUERY]
                [--region REGION]
```

命名的参数

-h, --help

显示 `pcluster update-compute-fleet` 的帮助文本。

--cluster-name, -n *CLUSTER_NAME*

指定集群的名称。

--status {START_REQUESTED,STOP_REQUESTED,ENABLED,DISABLED}

指定应用于集群计算实例集的状态。状态START_REQUESTED和STOP_REQUESTED对应于Slurm调度器，而状态ENABLED和DISABLED对应于调度器。 AWS Batch

--debug

启用调试日志记录。

--query *QUERY*

指定要对输出执行的 JMESPath 查询。

--region, -r *REGION*

指定 AWS 区域 要使用的。 AWS 区域 必须使用AWS_DEFAULT_REGION环境变量、~/.aws/config文件[default]部分中的region设置或--region参数来指定。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster update-compute-fleet -n cluster-v3 --status STOP_REQUESTED
{
  "status": "STOP_REQUESTED",
  "lastStatusUpdatedTime": "2022-07-12T20:19:47.653Z"
}
```

pcluster version

显示的版本 AWS ParallelCluster。

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

命名的参数

-h, --help

显示 `pcluster version` 的帮助文本。

--debug

启用调试日志记录。

使用 AWS ParallelCluster 版本 3.1.4 的示例：

```
$ pcluster version
{
  "version": "3.1.4"
}
```

pcluster3-config-converter

读取 AWS ParallelCluster 版本 2 配置文件并写入 AWS ParallelCluster 版本 3 配置文件。

```
pcluster3-config-converter [-h]
                [-t CLUSTER_TEMPLATE]
                [-c CONFIG_FILE]
                [--force-convert]
                [-o OUTPUT_FILE]
```

命名的参数

-h, --help

显示 `pcluster3-config-converter` 的帮助文本。

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

指定要转换的配置文件的 [cluster] 部分。如果未指定，则脚本将在 [global] 部分中查找 cluster-template 参数或搜索 [cluster default]。

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

指定要读取的 AWS ParallelCluster 版本 2 配置文件。

--force-convert

即使不支持或不建议使用一个或多个设置，也启用转换。

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

指定要写入的 AWS ParallelCluster 版本 3 配置文件。如果未指定此参数，则将配置写入 stdout。

> **Note**
>
> 该pcluster3-config-converter命令是在 3.0.1 AWS ParallelCluster 版本中添加的。

配置文件

AWS ParallelCluster 使用 YAML 1.1 文件作为配置参数。

主题

- 集群配置文件
- 构建映像配置文件

集群配置文件

AWS ParallelCluster 版本 3 使用单独的配置文件来控制群集基础架构的定义和自定义的定义 AMIs。所有配置文件都使用 YAML 1.1 文件。下面链接了每个配置文件的详细信息。有关一些示例配置，请参阅 a https://github.com/aws/w parallelcluster/tree/release-3.0/cli/tests/pcluster/example s-_configs。

这些对象用于 AWS ParallelCluster 版本 3 的集群配置。

主题

- 集群配置文件属性
- Imds 部分

- Image 部分
- HeadNode 部分
- Scheduling 部分
- SharedStorage 部分
- Iam 部分
- LoginNodes 部分
- Monitoring 部分
- Tags 部分
- AdditionalPackages 部分
- DirectoryService 部分
- DeploymentSettings 部分

集群配置文件属性

`Region`（可选，`String`）

AWS 区域 为群集指定。例如 `us-east-2`。

更新策略：如果更改此设置，则不允许更新。

`CustomS3Bucket`（可选，`String`）

指定在您的 AWS 账户中创建的 Amazon S3 存储桶的名称，该存储桶用于存储集群使用的资源（例如集群配置文件）和导出日志。 AWS ParallelCluster 在您创建集群的每个 AWS 区域中维护一个 Amazon S3 存储桶。默认情况下，这些 Amazon S3 存储桶命名为 `parallelcluster-hash-v1-DO-NOT-DELETE`。

更新策略：如果更改此设置，则不允许更新。如果您强制更新，则将忽略新值并使用旧值。

`AdditionalResources`（可选，`String`）

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

值必须是指向公有模板的 HTTPS URL，并提供所有参数。

没有默认值。

更新策略：可以在更新期间更改此设置。

Imds 部分

（可选）指定全局实例元数据服务 (IMDS) 配置。

```
Imds:
  ImdsSupport: string
```

Imds 属性

ImdsSupport（可选，String）

指定集群节点中支持的 IMDS 版本。支持的值为 v1.0 和 v2.0。默认值为 v2.0。

如果设置ImdsSupport为v1.0， IMDSv2 则同时支持 IMDSv1 和。

如果设置ImdsSupport为v2.0，则 IMDSv2 仅支持。

有关更多信息，请参阅《适用于 Linux 实例的 Amazon EC2 用户指南》 IMDSv2中的 “使用”。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 从 AWS ParallelCluster 3.7.0 开始，ImdsSupport默认值为。v2.0我们建议您在自定义操作调 IMDSv1 用 IMDSv2 中设置ImdsSupport为v2.0并替换为。
> 3.3.0 AWS ParallelCluster 版本ImdsSupport中增加了对 Imds/的支持。

Image 部分

> **Note**
>
> 官方 AMIs 分发的不支持版本 AWS ParallelCluster 将在闲置 18 个月后变为不可用。这些旧映像包含过时的软件，出现问题时无法获得支持。我们强烈建议改用最新的支持版本。

（必需）定义集群的操作系统。

```
Image:
  Os: string
  CustomAmi: string
```

Image 属性

Os（必需，String）

指定要对集群使用的操作系统。支持的值
为alinux2、、alinux2023、ubuntu2404、ubuntu2204、ubuntu2004、rhel8、rocky8、rhel9

> **Note**
>
> RedHat 从 AWS ParallelCluster 版本 3.6.0 开始添加企业 Linux 8.7 (rhel8)。
> 如果您将集群配置为使用 rhel，则任何实例类型的按需成本都高于将集群配置为使用支持的其它操作系统时的按需成本。有关定价的更多信息，请参阅按需定价和亚马逊上的红帽企业 Linux 是如何 EC2 提供和定价的？ 。
> RedHat 从 AWS ParallelCluster 版本 3.9.0 开始添加企业 Linux 9 (rhel9)。

所有 AWS 商业区域都支持以下所有操作系统。

分区（AWS 区域）	alinux2	ubuntu224 和 ubuntu204	ubuntu244	rhel8	rhel9	alinux2023
商业（均 AWS 区域未特别提及）	True	True	True	True	True	True
AWS GovCloud（美国东部）(us-gov-east-1)	True	True	True	True	True	True
AWS GovCloud（美国西部）(us-gov-west-1)	True	True	True	True	True	True
中国（北京）(cn-north-1)	True	True	True	True	True	True
中国（宁夏）(cn-northwest-1)	True	True	True	True	True	True

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> AWS ParallelCluster 3.8.0 支持 Rocky Linux 8，但预先构建的 Rocky Linux 8 AMIs（适用于 x86 和 ARM 架构）不可用。 AWS ParallelCluster 3.8.0 支持使用自定义方式使用 Rocky Linux 8 创建集群。 AMIs有关更多信息，请参阅操作系统注意事项。 AWS ParallelCluster 3.9.0 支持 Rocky Linux 9，但预先构建的 Rocky Linux 9 AMIs（适用于 x86 和 ARM 架构）不可用。 AWS ParallelCluster 3.9.0 支持使用自定义方式使用 Rocky Linux 9 创建集群。 AMIs有关更多信息，请参阅操作系统注意事项。

`CustomAmi`（可选，`String`）

指定要用于头节点和计算节点的自定义 AMI（而非默认 AMI）的 ID。有关更多信息，请参阅 AWS ParallelCluster AMI 自定义。

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

例如，如果自定义 AMI 具有与之关联的加密快照，则用户和头节点策略中都需要以下其他策略：

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

要构建 E RedHat nterprise Linux 自定义 AMI，必须配置操作系统以安装 RHUI (AWS) 存储库提供的软件包：`rhel-<version>-baseos-rhui-rpmsrhel-<version>-appstream-rhui-`

rpms、和codeready-builder-for-rhel-<version>-rhui-rpms。此外，自定义 AMI 上的存储库包含的 kernel-devel 程序包版本必须与正在运行的内核版本相同。

已知限制：

- 只有 RHEL 8.2 及更高版本支持 Lustre FSx 。
- RHEL 8.7 内核版本 4.18.0-425.3.1.el8 不支持 Lustre。 FSx
- 只有 RHEL 8.4 及更高版本支持 EFA。
- AL23 不支持 NICE DCV，因为它不包括运行 NICE DCV 所需的图形桌面环境。有关更多信息，请参阅官方的 NICE DCV 文档。

要排查自定义 AMI 验证警告，请参阅排查自定义 AMI 问题。

更新策略：如果更改此设置，则不允许更新。

HeadNode 部分

（必需）指定头节点的配置。

```
HeadNode:
  InstanceType: string
  Networking:
    SubnetId: string
    ElasticIp: string/boolean
    SecurityGroups:
      - string
    AdditionalSecurityGroups:
      - string
    Proxy:
      HttpProxyAddress: string
  DisableSimultaneousMultithreading: boolean
  Ssh:
    KeyName: string
    AllowedIps: string
  LocalStorage:
    RootVolume:
      Size: integer
      Encrypted: boolean
      VolumeType: string
      Iops: integer
      Throughput: integer
      DeleteOnTermination: boolean
```

```
    EphemeralVolume:
      MountDir: string
  SharedStorageType: string
  Dcv:
    Enabled: boolean
    Port: integer
    AllowedIps: string
  CustomActions:
    OnNodeStart:
      Sequence:
        - Script: string
          Args:
            - string
      Script: string
      Args:
        - string
    OnNodeConfigured:
      Sequence:
        - Script: string
          Args:
            - string
      Script: string
      Args:
        - string
    OnNodeUpdated:
      Sequence:
        - Script: string
          Args:
            - string
      Script: string
      Args:
        - string
  Iam:
    InstanceRole: string
    InstanceProfile: string
    S3Access:
      - BucketName: string
        EnableWriteAccess: boolean
        KeyName: string
    AdditionalIamPolicies:
      - Policy: string
  Imds:
    Secured: boolean
  Image:
```

```
    CustomAmi: string
```

HeadNode 属性

`InstanceType（必填，String）`

指定头节点的实例类型。

指定用于头节点的 Amazon EC2 实例类型。实例类型的架构必须与用于 AWS Batch `InstanceType`或Slurm`InstanceType`设置的架构相同。

> **Note**
>
> AWS ParallelCluster 不支持该`HeadNode`设置的以下实例类型。
>
> - hpc6id

如果您定义一个 p4d 实例类型或其他具有多个网络接口或网络接口卡的实例类型，则必须`ElasticIp`将设置为`true`以提供公共访问权限。 AWS public IPs 只能分配给使用单个网络接口启动的实例。对于这种情况，我们建议您使用 NAT 网关为集群计算节点提供公有访问权限。有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。

更新策略：如果更改此设置，则不允许更新。

`DisableSimultaneousMultithreading（可选，Boolean）`

如果`true`，则在头节点上禁用超线程。默认值为 `false`。

并非所有实例类型都可以禁用超线程。有关支持禁用超线程的实例类型列表，请参阅 A mazon EC2 用户指南中每种实例类型的 CPU 核心和每个 CPU 内核的线程。

更新策略：如果更改此设置，则不允许更新。

`SharedStorageType（可选，String）`

指定用于内部共享数据的存储类型。内部共享的数据包括 AWS ParallelCluster 用于管理群集的数据，以及用于挂载共享文件系统卷的默认共享数据（`/home`如果未在挂载目录中指定）。`SharedStorage` 部分有关内部共享数据的更多详细信息，请参阅 AWS ParallelCluster 内部目录。

如果使用 `Ebs`（默认存储类型），则头节点将使用 NFS 导出其根卷的部分内容，作为计算节点和登录节点的共享目录。

如果Efs， ParallelCluster 将创建一个 EFS 文件系统用于共享的内部数据，以及。/home

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 当集群横向扩展时，EBS 存储类型可能会出现性能瓶颈，因为头节点会使用 NFS 导出与计算节点共享根卷中的数据。使用 EFS，可以在集群横向扩展时避免 NFS 导出，并避免与之相关的性能瓶颈。建议选择 EBS，以最大限度地提高存放小文件和安装过程 read/write 的可能性。选择 EFS 进行扩展。

Networking

（必需）定义头节点的网络配置。

```
Networking:
  SubnetId: string
  ElasticIp: string/boolean
  SecurityGroups:
    - string
  AdditionalSecurityGroups:
    - string
  Proxy:
    HttpProxyAddress: string
```

更新策略：如果更改此设置，则不允许更新。

Networking 属性

SubnetId（必需，String）

指定要在其中预置头节点的现有子网的 ID。

更新策略：如果更改此设置，则不允许更新。

ElasticIp（可选，String）

创建弹性 IP 地址或将弹性 IP 地址分配给头节点。支持的值为 true、false 或现有弹性 IP 地址的 ID。默认值为 false。

更新策略：如果更改此设置，则不允许更新。

`SecurityGroups`（可选，`[String]`）

用于头节点的 Amazon VPC 安全组 ID 的列表。如果不包含此属性，则它们将替换 AWS ParallelCluster 创建的安全组。

验证您的SharedStorage系统是否正确配置了安全组。

更新策略：可以在更新期间更改此设置。

`AdditionalSecurityGroups`（可选，`[String]`）

用于头节点的其他 Amazon VPC 安全组 ID 的列表。

更新策略：可以在更新期间更改此设置。

`Proxy`（可选）

指定头节点的代理设置。

```
Proxy:
                                HttpProxyAddress:
                                string
```

`HttpProxyAddress`（可选，`String`）

定义 HTTP 或 HTTPS 代理服务器，通常为 `https://x.x.x.x:8080`。

没有默认值。

更新策略：如果更改此设置，则不允许更新。

Ssh

（可选）定义头节点的 SSH 访问配置。

```
Ssh:
      KeyName: string
      AllowedIps: string
```

更新策略：可以在更新期间更改此设置。

Ssh 属性

KeyName（可选，String）

命名现有 Amazon EC2 密钥对以启用 SSH 访问头节点。

更新策略：如果更改此设置，则不允许更新。

AllowedIps（可选，String）

指定头节点 SSH 连接的 CIDR 格式 IP 范围或前缀列表 ID。默认值为 0.0.0.0/0。

更新策略：可以在更新期间更改此设置。

LocalStorage

（可选）定义头节点的本地存储配置。

```
LocalStorage:
  RootVolume:
    Size: integer
    Encrypted: boolean
    VolumeType: string
    Iops: integer
    Throughput: integer
    DeleteOnTermination: boolean
  EphemeralVolume:
    MountDir: string
```

更新策略：可以在更新期间更改此设置。

LocalStorage 属性

RootVolume（必需）

指定头节点的根卷存储。

```
RootVolume:
  Size: integer
  Encrypted: boolean
  VolumeType: string
  Iops: integer
  Throughput: integer
```

```
DeleteOnTermination: boolean
```

更新策略：可以在更新期间更改此设置。

Size（可选，Integer）

指定头节点根卷大小，以吉字节 (GiB) 为单位。默认大小来自 AMI。如果使用不同的大小，则 AMI 必须支持 growroot。

更新策略：如果更改此设置，则不允许更新。

Encrypted（可选，Boolean）

指定是否对根卷进行加密。默认值为 true。

更新策略：如果更改此设置，则不允许更新。

VolumeType（可选，String）

指定 A mazon EBS 卷类型。支持的值为 gp2、gp3、io1、io2、sc1、st1 和 standard。默认值为 gp3。

有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 Amaz on EBS 卷类型。

更新策略：如果更改此设置，则不允许更新。

Iops（可选，Integer）

定义 io1、io2 和 gp3 类型卷的 IOPS 数。

默认值、支持的值以及 volume_size/volume_iops 比率因 VolumeType 和 Size 而异。

更新策略：如果更改此设置，则不允许更新。

VolumeType = io1

默认值：Iops = 100

支持的值：Iops = 100–64000 †

最大 Iops/Size 比率 = 50 IOPS/GiB。5000 IOPS 需要至少 100 GiB 的 Size。

VolumeType = io2

默认值：Iops = 100

支持的值：Iops = 100–64000（io2 Block Express 卷为 256000）†

最大 Iops/Size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 Size。

VolumeType = gp3

默认值：Iops = 3000

支持的值：Iops = 3000–16000

最大 Iops/Size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 Size。

† 只有在配置超过 32,000 IOPS 的 Nitro 系统上构建的实例才能保证最大 IOPS。其他实例保证最高为 32000 IOPS。除非您修改io1卷，否则较旧的卷可能无法达到最大性能。 io2Block Express 卷在 R5b 实例类型上支持高达 256000 的 Iops 值。有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 “io2屏蔽 Express 卷”。

更新策略：可以在更新期间更改此设置。

Throughput（可选，Integer）

定义 gp3 卷类型的吞吐量，以 MiB/s 为单位。此设置仅在 VolumeType 为 gp3 时有效。默认值为 125。支持的值：125–1000 MiB/s

Throughput 与 Iops 的比率不能超过 0.25。如果最大吞吐量为 1000，则该Iops设置 MiB/s 必须至少为 4000。

更新策略：如果更改此设置，则不允许更新。

DeleteOnTermination（可选，Boolean）

指定头节点终止时是否应删除根卷。默认值为 true。

更新策略：如果更改此设置，则不允许更新。

EphemeralVolume（可选）

指定任何实例存储卷的详细信息。有关更多信息，请参阅 Amazon EC2 用户指南中的实例存储卷。

```
EphemeralVolume:
  MountDir: string
```

更新策略：如果更改此设置，则不允许更新。

MountDir（可选，String）

指定实例存储卷的挂载目录。默认值为 /scratch。

更新策略：如果更改此设置，则不允许更新。

Dcv

（可选）定义在头节点上运行的 Amazon DCV 服务器的配置设置。

有关更多信息，请参阅 通过 Amazon DCV 连接到头节点和登录节点。

```
Dcv:
  Enabled: boolean
  Port: integer
  AllowedIps: string
```

> **Important**
>
> 默认情况下，设置的 Amazon DCV 端口对所有 IPv4 地址开放。 AWS ParallelCluster 但是，只有当您具有 Amazon DCV 会话的 URL 时，才能连接到 Amazon DCV 端口，并应在 `pcluster dcv-connect` 返回 URL 后的 30 秒内连接到 Amazon DCV 会话。请使用 `AllowedIps` 设置进一步限制对具有 CIDR 格式 IP 范围的 Amazon DCV 端口的访问，并使用 `Port` 设置来设置非标准端口。

更新策略：如果更改此设置，则不允许更新。

Dcv 属性

`Enabled`（必填，`Boolean`）

指定是否在头节点上启用 Amazon DCV。默认值为 `false`。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> Amazon DCV 会自动生成自签名证书，该证书用于保护 Amazon DCV 客户端与在头节点上运行的 Amazon DCV 服务器之间的流量。要配置您自己的证书，请参阅 Amazon DCV HTTPS 证书。

`Port`（可选，`Integer`）

指定 Amazon DCV 的端口。默认值为 8443。

更新策略：如果更改此设置，则不允许更新。

`AllowedIps`（可选，建议，`String`）

指定 Amazon DCV 连接的 CIDR 格式的 IP 范围。此设置仅在 AWS ParallelCluster 创建安全组时使用。默认值是 `0.0.0.0/0`，允许从任何 Internet 地址访问。

更新策略：可以在更新期间更改此设置。

CustomActions

（可选）指定要在头节点上运行的自定义脚本。

```
CustomActions:
  OnNodeStart:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
  OnNodeConfigured:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
  OnNodeUpdated:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
```

CustomActions 属性

`OnNodeStart`（可选）

指定要在启动任何节点部署引导操作之前在头节点上运行的单个脚本或一系列脚本。有关更多信息，请参阅自定义引导操作。

Sequence（可选）

要运行的脚本列表。 AWS ParallelCluster 按配置文件中列出的顺序运行脚本，从第一个脚本开始。

Script（必需，String）

指定要使用的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选，[String]）

要传递到脚本的参数的列表。

Script（必需，String）

指定用于单个脚本的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选，[String]）

要传递到单个脚本的参数的列表。

更新策略：如果更改此设置，则不允许更新。

OnNodeConfigured（可选）

指定要在节点引导操作完成后在头节点上运行的单个脚本或一系列脚本。有关更多信息，请参阅自定义引导操作。

Sequence（可选）

指定要运行的脚本的列表。

Script（必需，String）

指定要使用的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选，[String]）

要传递到脚本的参数的列表。

Script（必需，String）

指定用于单个脚本的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选，[String]）

要传递到单个脚本的参数的列表。

更新策略：如果更改此设置，则不允许更新。

OnNodeUpdated（可选）

指定要在节点更新操作完成后在头节点上运行的单个脚本或一系列脚本。有关更多信息，请参阅自定义引导操作。

Sequence（可选）

指定要运行的脚本的列表。

Script（必需，String）

指定要使用的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选，[String]）

要传递到脚本的参数的列表。

Script（必需，String）

指定用于单个脚本的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选，[String]）

要传递到单个脚本的参数的列表。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> OnNodeUpdated是从 AWS ParallelCluster 3.4.0 开始添加的。
> Sequence是从 3.6.0 AWS ParallelCluster 版本开始添加的。指定后Sequence，您可以列出一个自定义操作的多个脚本。 AWS ParallelCluster 继续支持使用单个脚本配置自定义操作，不包括脚本Sequence。
> AWS ParallelCluster 不支持同时包含单个脚本和Sequence同一个自定义操作。

Iam

（可选）指定要在头节点上使用的实例角色或实例配置文件，用于覆盖集群的默认实例角色或实例配置文件。

```
Iam:
  InstanceRole: string
  InstanceProfile: string
```

```
  S3Access:
    - BucketName: string
      EnableWriteAccess: boolean
      KeyName: string
  AdditionalIamPolicies:
    - Policy: string
```

更新策略：可以在更新期间更改此设置。

Iam 属性

InstanceProfile（可选，String）

指定用于覆盖默认头节点实例配置文件的实例配置文件。您不能同时指定 InstanceProfile 和 InstanceRole。格式为 arn:*Partition*:iam::*Account*:instance-profile/*InstanceProfileName*。

如果指定此设置，则不能指定 S3Access 和 AdditionalIamPolicies 设置。

我们建议您指定 S3Access 和 AdditionalIamPolicies 设置中的一个或两个，因为添加到 AWS ParallelCluster 中的功能通常需要新权限。

更新策略：如果更改此设置，则不允许更新。

InstanceRole（可选，String）

指定用于覆盖默认头节点实例角色的实例角色。您不能同时指定 InstanceProfile 和 InstanceRole。格式为 arn:*Partition*:iam::*Account*:role/*RoleName*。

如果指定此设置，则不能指定 S3Access 和 AdditionalIamPolicies 设置。

我们建议您指定 S3Access 和 AdditionalIamPolicies 设置中的一个或两个，因为添加到 AWS ParallelCluster 中的功能通常需要新权限。

更新策略：可以在更新期间更改此设置。

S3Access

S3Access（可选）

指定存储桶。此设置用于生成针对存储桶授予指定访问权限的策略。

如果指定此设置，则不能指定 InstanceProfile 和 InstanceRole 设置。

我们建议您指定 S3Access 和 AdditionalIamPolicies 设置中的一个或两个，因为添加到 AWS ParallelCluster 中的功能通常需要新权限。

```
S3Access:
  - BucketName: string
    EnableWriteAccess: boolean
    KeyName: string
```

更新策略：可以在更新期间更改此设置。

BucketName（必需，String）

存储桶的名称。

更新策略：可以在更新期间更改此设置。

KeyName（可选，String）

存储桶的密钥。默认值为“*”。

更新策略：可以在更新期间更改此设置。

EnableWriteAccess（可选，Boolean）

指示是否为存储桶启用写入权限。默认值为 false。

更新策略：可以在更新期间更改此设置。

AdditionalIamPolicies

AdditionalIamPolicies（可选）

指定适用于亚马逊 IAM 策略的亚马逊资源名称 (ARNs) 列表 EC2。除了所需的权限外，此列表还附在用于头节点的根角色上 AWS ParallelCluster。

IAM 策略名称及其 ARN 不相同。不能使用名称。

如果指定此设置，则不能指定 InstanceProfile 和 InstanceRole 设置。

我们建议您使用 AdditionalIamPolicies，因为 AdditionalIamPolicies 已经添加到 AWS ParallelCluster 所需的权限中，而 InstanceRole 必须包含所有必需的权限。随着功能的不断添加，所需权限通常会随版本发生变化。

没有默认值。

```
AdditionalIamPolicies:
  - Policy: string
```

更新策略：可以在更新期间更改此设置。

Policy（可选，[String]）

IAM 策略的列表。

更新策略：可以在更新期间更改此设置。

Imds

（可选）指定实例元数据服务 (IMDS) 的属性。有关更多信息，请参阅 Amazon EC2 用户指南中的实例元数据服务版本 2 的工作原理。

```
Imds:
    Secured: boolean
```

更新策略：如果更改此设置，则不允许更新。

Imds 属性

Secured（可选，Boolean）

如果为 `true`，则仅允许一部分超级用户访问头节点的 IMDS（以及实例配置文件凭证）。

如果为 `false`，则头节点中的每个用户都可以访问头节点的 IMDS。

允许以下用户访问头节点的 IMDS：

- 根用户
- 集群管理用户（默认为 `pc-cluster-admin`）
- 操作系统特定的默认用户（`ec2-user`在 Amazon Linux 2 和 RedHat Ubuntu 18.04 `ubuntu`上。

默认值为 `true`。

`default`用户负责确保集群拥有与 AWS 资源交互所需的权限。如果您禁用`default`用户 IMDS 访问权限，则 AWS ParallelCluster 无法管理计算节点并停止工作。请勿禁用 `default` 用户 IMDS 访问权限。

向用户授予头节点 IMDS 的访问权限后，他们可以使用头节点的实例配置文件中包含的权限。例如，他们可以使用这些权限启动 Amazon EC2 实例，或者读取集群配置为用于身份验证的 AD 域的密码。

要限制 IMDS 的访问权限，请 AWS ParallelCluster 管理一连串的。`iptables`

具有 `sudo` 访问权限的集群用户可以通过运行以下命令，对其他单独用户（包括 `default` 用户）有选择地启用或禁用对头节点 IMDS 的访问权限：

```
$ sudo /opt/parallelcluster/scripts/imds/imds-access.sh --allow <USERNAME>
```

您可以使用此命令的 `--deny` 选项禁用用户的 IMDS 访问权限。

如果您无意中禁用了 `default` 用户的 IMDS 访问权限，则可以使用 `--allow` 选项还原该权限。

> **Note**
>
> 对 `iptables` 或 `ip6tables` 规则进行任何自定义都可能干扰头节点上用于限制 IMDS 访问权限的机制。

更新策略：如果更改此设置，则不允许更新。

Image

（可选）为头节点定义自定义映像。

```
Image:
    CustomAmi: string
```

更新策略：如果更改此设置，则不允许更新。

Image 属性

`CustomAmi`（可选，`String`）

指定要用于头节点的自定义 AMI（而非默认 AMI）的 ID。有关更多信息，请参阅 AWS ParallelCluster AMI 自定义。

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

例如，如果自定义 AMI 具有与之关联的加密快照，则用户和头节点策略中都需要以下其他策略：

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

要排查自定义 AMI 验证警告，请参阅排查自定义 AMI 问题。

更新策略：如果更改此设置，则不允许更新。

Scheduling 部分

（必需）定义集群中使用的作业调度器以及该作业调度器管理的计算实例。您可以使用Slurm或 AWS Batch 调度程序。每个调度器支持一组不同的设置和属性。

主题

- Scheduling 属性
- AwsBatchQueues
- SlurmQueues
- SlurmSettings

```
Scheduling:
  Scheduler: slurm
  ScalingStrategy: string
  SlurmSettings:
```

```
    MungeKeySecretArn: string
    ScaledownIdletime: integer
    QueueUpdateStrategy: string
    EnableMemoryBasedScheduling: boolean
    CustomSlurmSettings: [dict]
    CustomSlurmSettingsIncludeFile: string
    Database:
      Uri: string
      UserName: string
      PasswordSecretArn: string
      DatabaseName: string
    ExternalSlurmdbd: boolean
      Host: string
      Port: integer
    Dns:
      DisableManagedDns: boolean
      HostedZoneId: string
      UseEc2Hostnames: boolean
  SlurmQueues:
    - Name: string
      ComputeSettings:
        LocalStorage:
          RootVolume:
            Size: integer
            Encrypted: boolean
            VolumeType: string
            Iops: integer
            Throughput: integer
          EphemeralVolume:
            MountDir: string
      CapacityReservationTarget:
        CapacityReservationId: string
        CapacityReservationResourceGroupArn: string
      CapacityType: string
      AllocationStrategy: string
      JobExclusiveAllocation: boolean
      CustomSlurmSettings: dict
      Tags:
        - Key: string
          Value: string
      HealthChecks:
        Gpu:
          Enabled: boolean
      Networking:
```

```
        SubnetIds:
          - string
        AssignPublicIp: boolean
        SecurityGroups:
          - string
        AdditionalSecurityGroups:
          - string
        PlacementGroup:
          Enabled: boolean
          Id: string
          Name: string
        Proxy:
          HttpProxyAddress: string
      ComputeResources:
        - Name: string
          InstanceType: string
          Instances:
            - InstanceType: string
          MinCount: integer
          MaxCount: integer
          DynamicNodePriority: integer
          StaticNodePriority: integer
          SpotPrice: float
          DisableSimultaneousMultithreading: boolean
          SchedulableMemory: integer
          HealthChecks:
            Gpu:
              Enabled: boolean
          Efa:
            Enabled: boolean
            GdrSupport: boolean
          CapacityReservationTarget:
            CapacityReservationId: string
            CapacityReservationResourceGroupArn: string
          Networking:
            PlacementGroup:
              Enabled: boolean
              Name: string
          CustomSlurmSettings: dict
          Tags:
            - Key: string
              Value: string
      CustomActions:
        OnNodeStart:
```

```
          Sequence:
            - Script: string
              Args:
                - string
          Script: string
          Args:
            - string
        OnNodeConfigured:
          Sequence:
            - Script: string
              Args:
                - string
          Script: string
          Args:
            - string
      Iam:
        InstanceProfile: string
        InstanceRole: string
        S3Access:
          - BucketName: string
            EnableWriteAccess: boolean
            KeyName: string
        AdditionalIamPolicies:
          - Policy: string
      Image:
        CustomAmi: string
```

```
Scheduling:
  Scheduler: awsbatch
  AwsBatchQueues:
    - Name: string
      CapacityType: string
      Networking:
        SubnetIds:
          - string
        AssignPublicIp: boolean
        SecurityGroups:
          - string
        AdditionalSecurityGroups:
          - string
      ComputeResources:  # this maps to a Batch compute environment (initially we
 support only 1)
        - Name: string
```

```
        InstanceTypes:
          - string
        MinvCpus: integer
        DesiredvCpus: integer
        MaxvCpus: integer
        SpotBidPercentage: float
```

Scheduling 属性

Scheduler（必需，String）

指定使用的调度器的类型。支持的值为 `slurm` 和 `awsbatch`。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> `awsbatch` 仅支持 `alinux2` 操作系统和 `x86_64` 平台。

ScalingStrategy（可选，String）

让您能够选择动态 Slurm 节点的纵向扩展方式。支持的值为 `all-or-nothing`、`greedy-all-or-nothing` 和 `best-effort`，默认值为 `all-or-nothing`。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 扩展策略仅适用于 Slurm 即将恢复的节点，而不适用于最终已在运行的节点。

- `all-or-nothing`此策略严格遵循 all-or-nothing-approach，旨在避免在扩展过程结束时出现空闲实例。它是在 all-or-nothing基础上运行的，这意味着它要么完全扩展，要么根本不扩展。请注意，当作业需要超过 500 个节点或跨多个计算资源时，临时启动的实例可能会产生额外费用。在三种可能的扩展策略中，该策略的吞吐量最低。扩展时间取决于每次执行 Slurm 恢复程序时提交的作业数量。此外，您的扩展不能远远超过每次执行的默认 RunInstances 资源帐户限制，默认情况下为 1000 个实例。更多详情可在亚马逊 EC2 API 限制文档中找到
- `greedy-all-or-nothing` 与该 all-or-nothing策略类似，它旨在避免在缩放后出现闲置实例。此策略允许在扩展过程中临时超额扩展，以实现比该 all-or-nothing方法更高的吞吐量，但也具有与 RunInstances 资源账户限制相同的扩展限制，即 1000 个实例。

- `best-effort` 该策略优先考虑高吞吐量，即使这意味着某些实例在扩展过程结束时可能处于闲置状态。该策略会尝试分配作业要求的尽可能多的节点，但有可能无法满足全部要求。与其他策略不同，尽力而为的方法可以积累比标准 RunInstances限制更多的实例，但代价是在执行多个扩展过程时会有闲置资源。

每种策略都旨在满足不同的扩展需求，您可以根据自己的具体要求和限制条件进行选择。

AwsBatchQueues

（可选）AWS Batch 队列设置。仅支持一个队列。如果 `Scheduler` 设置为 `awsbatch`，则此部分是必需的。有关 `awsbatch` 调度器的更多信息，请参阅联网设置和 将 AWS Batch (awsbatch) 调度器与 AWS ParallelCluster。

```
AwsBatchQueues:
  - Name: string
    CapacityType: string
    Networking:
      SubnetIds:
        - string
      AssignPublicIp: boolean
      SecurityGroups:
        - string
      AdditionalSecurityGroups:
        - string
    ComputeResources:  # this maps to a Batch compute environment (initially we support
 only 1)
      - Name: string
        InstanceTypes:
          - string
        MinvCpus: integer
        DesiredvCpus: integer
        MaxvCpus: integer
        SpotBidPercentage: float
```

更新策略：可以在更新期间更改此设置。

AwsBatchQueues 属性

Name（必需，**String**）

AWS Batch 队列的名称。

更新策略：如果更改此设置，则不允许更新。

CapacityType（可选，**String**）

AWS Batch 队列使用的计算资源的类型。支持的值为 ONDEMAND、SPOT 或 CAPACITY_BLOCK。默认值为 ONDEMAND。

> **Note**
>
> 如果将 CapacityType 设置为 SPOT，则您的账户必须包含 AWSServiceRoleForEC2Spot 服务相关角色。您可以使用以下 AWS CLI 命令创建此角色。
>
> ```
> $ aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```
>
> 有关更多信息，请参阅 Amazon Amazon Linux 实例 EC2 用户指南中的竞价型实例请求的服务相关角色。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

Networking

（必需）定义 AWS Batch 队列的网络配置。

```
Networking:
  SubnetIds:
    - string
  AssignPublicIp: boolean
  SecurityGroups:
    - string
  AdditionalSecurityGroups:
    - string
```

Networking 属性

SubnetIds（必需，**[String]**）

指定要在其中配置 AWS Batch 队列的现有子网的 ID。目前仅支持一个子网。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

AssignPublicIp（可选，**String**）

为 AWS Batch 队列中的节点创建或分配公有 IP 地址。支持的值为 `true` 和 `false`。默认值取决于您指定的子网。

更新策略：如果更改此设置，则不允许更新。

SecurityGroups（可选，**[String]**）

AWS Batch 队列使用的安全组列表。如果您未指定安全组，则 AWS ParallelCluster 会创建新的安全组。

更新策略：可以在更新期间更改此设置。

AdditionalSecurityGroups（可选，**[String]**）

AWS Batch 队列使用的安全组列表。

更新策略：可以在更新期间更改此设置。

ComputeResources

（必需）定义 AWS Batch 队列的 ComputeResources 配置。

```
ComputeResources:  # this maps to a Batch compute environment (initially we support
 only 1)
  - Name: string
    InstanceTypes:
      - string
    MinvCpus: integer
    DesiredvCpus: integer
    MaxvCpus: integer
    SpotBidPercentage: float
```

ComputeResources 属性

Name（必需，**String**）

AWS Batch 队列计算环境的名称。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

InstanceTypes（必需，**[String]**）

实例类型的 AWS Batch 计算环境数组。所有实例类型都必须使用 x86_64 架构。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

MinvCpus（可选，**Integer**）

AWS Batch 计算环境可以使用的最小数量。 VCPUs

更新策略：可以在更新期间更改此设置。

DesiredVcpus（可选，**Integer**）

AWS Batch 计算环境 VCPUs 中所需的数量。 AWS Batch `MaxvCpus`根据任务队列中的需求在`MinvCpus`和之间调整此值。

更新策略：在更新期间不分析此设置。

MaxvCpus（可选，**Integer**）

AWS Batch 计算环境的最大数量。 VCPUs 不能将此值设置为低于 `DesiredVcpus`。

更新策略：更新期间不能减小此设置。

SpotBidPercentage（可选，**Float**）

实例启动前，Amazon EC2 竞价型实例价格可以达到的实例类型按需价格的最大百分比。默认值为 `100` (100%)。支持的范围是 1-100。

更新策略：可以在更新期间更改此设置。

SlurmQueues

（可选）Slurm 队列的设置。如果 `Scheduler` 设置为 `slurm`，则此部分是必需的。

```
SlurmQueues:
  - Name: string
    ComputeSettings:
      LocalStorage:
        RootVolume:
          Size: integer
          Encrypted: boolean
          VolumeType: string
          Iops: integer
          Throughput: integer
        EphemeralVolume:
          MountDir: string
    CapacityReservationTarget:
      CapacityReservationId: string
```

```
      CapacityReservationResourceGroupArn: string
    CapacityType: string
    AllocationStrategy: string
    JobExclusiveAllocation: boolean
    CustomSlurmSettings: dict
    Tags:
      - Key: string
        Value: string
    HealthChecks:
      Gpu:
        Enabled: boolean
    Networking:
      SubnetIds:
        - string
      AssignPublicIp: boolean
      SecurityGroups:
        - string
      AdditionalSecurityGroups:
        - string
      PlacementGroup:
        Enabled: boolean
        Id: string
        Name: string
      Proxy:
        HttpProxyAddress: string
    ComputeResources:
      - Name: string
        InstanceType: string
        Instances:
          - InstanceType: string
        MinCount: integer
        MaxCount: integer
        DynamicNodePriority: integer
        StaticNodePriority: integer
        SpotPrice: float
        DisableSimultaneousMultithreading: boolean
        SchedulableMemory: integer
        HealthChecks:
          Gpu:
            Enabled: boolean
        Efa:
          Enabled: boolean
          GdrSupport: boolean
        CapacityReservationTarget:
```

```
          CapacityReservationId: string
          CapacityReservationResourceGroupArn: string
        Networking:
          PlacementGroup:
            Enabled: boolean
            Name: string
        CustomSlurmSettings: dict
        Tags:
          - Key: string
            Value: string
    CustomActions:
      OnNodeStart:
        Sequence:
          - Script: string
            Args:
              - string
        Script: string
        Args:
          - string
      OnNodeConfigured:
        Sequence:
          - Script: string
            Args:
              - string
        Script: string
        Args:
          - string
    Iam:
      InstanceProfile: string
      InstanceRole: string
      S3Access:
        - BucketName: string
          EnableWriteAccess: boolean
          KeyName: string
      AdditionalIamPolicies:
        - Policy: string
    Image:
      CustomAmi: string
```

更新策略：对于此列表值设置，可以在更新期间添加新值，或者在删除现有值时必须停止计算实例集。

SlurmQueues 属性

Name（必需，**String**）

Slurm 队列的名称。

> **Note**
>
> 在更新期间，集群大小可能会更改。有关更多信息，请参阅集群容量大小和更新

更新策略：如果更改此设置，则不允许更新。

CapacityReservationTarget

> **Note**
>
> `CapacityReservationTarget`已在 3.3.0 AWS ParallelCluster 版本中添加。

```
CapacityReservationTarget:
  CapacityReservationId: string
  CapacityReservationResourceGroupArn: string
```

指定队列计算资源的按需容量预留。

CapacityReservationId（可选，**String**）

要用于队列计算资源的现有容量预留的 ID。ID 可以指的是 ODCR 或适用于 ML 的容量块。

预留使用的平台必须与实例使用的平台相同。例如，如果您的实例在 `rhel8` 上运行，则您的容量预留必须在 Red Hat Enterprise Linux 平台上运行。有关更多信息，请参阅《Amazon Linux 实例 EC2 用户指南》中的支持的平台。

> **Note**
>
> 如果在集群配置中包含 `Instances`，则必须从配置中排除此队列级别 `CapacityReservationId` 设置。

更新策略：必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

CapacityReservationResourceGroupArn（可选，**String**）

用作队列计算资源的服务相关容量预留组的资源组的 Amazon 资源名称 (ARN)。 AWS ParallelCluster 根据以下条件确定并使用资源组中最适当的容量预留。

- 如果在 SlurmQueues/Networking或 SlurmQueues//中启用 Networking，PlacementGroup则 AWS ParallelCluster 选择以实例类型为目标的资源组，如果计算资源存在，则PlacementGroup为计算资源选择资源组。ComputeResources

  PlacementGroup 必须以 ComputeResources 中定义的实例类型之一为目标。
- 如果PlacementGroup未在 SlurmQueues/Networking或 SlurmQueuesComputeResources/中启用 Networking，则 AWS ParallelCluster 选择仅针对计算资源的实例类型的资源组（如果存在计算资源）。

在队列的所有计算资源和可用区中，资源组必须为可用区中的每种实例类型保留至少一个 ODCR。有关更多信息，请参阅 使用按需容量预留（ODCR）启动实例。

有关多子网配置要求的更多信息，请参阅 Networking/SubnetIds。

> **Note**
>
> 3.4.0 AWS ParallelCluster 版本中添加了多个可用区。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

CapacityType（可选，**String**）

Slurm 队列使用的计算资源的类型。支持的值为 ONDEMAND、SPOT 或 CAPACITY_BLOCK。默认值为 ONDEMAND。

> **Note**
>
> 如果将 CapacityType 设置为 SPOT，则您的账户必须具有 AWSServiceRoleForEC2Spot 服务相关角色。您可以使用以下 AWS CLI 命令来创建此角色。
>
> ```
> $ aws iam create-service-linked-role --aws-service-name spot.amazonaws.com
> ```

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

更新策略：必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

`AllocationStrategy`（可选，**`String`**）

为 `Instances` 中定义的所有计算资源指定分配策略。

有效值：`lowest-price` |`capacity-optimized` |`price-capacity-optimized`

默认值：`lowest-price`

`lowest-price`

- 如果您使用`CapacityType = ONDEMAND`，Amazon EC2 Fleet 会使用价格来确定订单，并首先启动价格最低的实例。
- 如果您使用`CapacityType = SPOT`，Amazon EC2 Fleet 会从具有可用容量的最低价格竞价型实例池中启动实例。如果池在满足您的所需容量之前已用完容量，Amazon F EC2 leet 会通过为您启动实例来满足您的请求。特别是，Amazon EC2 Fleet 会从具有可用容量的最低价格竞价型实例池中启动实例。Amazon EC2 Fleet 可能会从多个不同的池中启动竞价型实例。
- 如果设置 `CapacityType = CAPACITY_BLOCK`，则没有分配策略，因此无法配置 `AllocationStrategy` 参数。

`capacity-optimized`

- 如果设置 `CapacityType = ONDEMAND`，则 `capacity-optimized` 不可用。
- 如果您进行了设置`CapacityType = SPOT`，Amazon EC2 Fleet 将从竞价型实例池中启动具有最佳容量的实例，以适应要启动的实例数量。

`price-capacity-optimized`

- 如果设置 `CapacityType = ONDEMAND`，则 `capacity-optimized` 不可用。
- 如果您进行了设置`CapacityType = SPOT`，Amazon EC2 Fleet 会根据正在启动的实例数量确定容量可用性最高的池。这意味着我们将从我们认为短期内中断概率最低的池中请求竞价型实例。然后，Amazon EC2 Fleet 请求这些池中价格最低的竞价型实例。

更新策略：必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.3.0 开始支持 `AllocationStrategy`。

`JobExclusiveAllocation`（可选，**`String`**）

如果设置为 `true`，则 Slurm 分区 `OverSubscribe` 标志设置为 `EXCLUSIVE`。当 `OverSubscribe=EXCLUSIVE` 时，分区中的作业将对分配的所有节点具有独占访问权限。有关更多信息，请参阅 Slurm 文档中的 EXCLUSIVE。

有效值：`true` | `false`

默认值：`false`

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.7.0 开始支持 `JobExclusiveAllocation`。

`CustomSlurmSettings`（可选，**`Dict`**）

定义自定义 Slurm 分区（队列）配置设置。

指定应用于队列（分区）的自定义 Slurm 配置参数键值对的字典。

每个单独的键值对（例如 `Param1: Value1`）都以 `Param1=Value1` 格式单独添加到 Slurm 分区配置行的末尾。

您只能指定未在 `CustomSlurmSettings` 中列入拒绝列表的 Slurm 配置参数。有关列入拒绝列表的 Slurm 配置参数的信息，请参阅 被拒登名单 Slurm 的配置参数 `CustomSlurmSettings`。

AWS ParallelCluster 仅检查参数是否在拒绝列表中。 AWS ParallelCluster 不会验证您的自定义 Slurm配置参数语法或语义。您有责任验证您的自定义Slurm配置参数。无效的自定义 Slurm 配置参数可能会导致 Slurm 进程守护程序失败，从而导致集群创建和更新失败。

有关如何使用指定自定义Slurm配置参数的更多信息 AWS ParallelCluster，请参阅Slurm 配置自定义。

有关 Slurm 配置参数的更多信息，请参阅 Slurm 文档中的 slurm.conf。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.6.0 开始支持 `CustomSlurmSettings`。

Tags（可选，[字符串]）

标签键值对的列表。`ComputeResource` 标签覆盖 Tags 部分或 `SlurmQueues/Tags` 中指定的重复标签。

Key（可选，**String**）

标签键。

Value（可选，**String**）

标签值。

更新策略：必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

HealthChecks（可选）

指定队列中所有计算资源上的计算节点运行状况检查。

Gpu（可选）

指定队列中所有计算资源上的 GPU 运行状况检查。

> **Note**
>
> AWS ParallelCluster 在使用 `alinux2` ARM 操作系统的节点`Gpu`中不支持`HealthChecks/`。这些平台不支持 NVIDIA 数据中心 GPU 管理器 (DCGM)。

Enabled（可选，**Boolean**）

是否 AWS ParallelCluster 对计算节点执行 GPU 运行状况检查。默认值为 `false`。

Gpu 运行状况检查行为

- 如果 `Gpu/Enabled` 设置为 `true`，则 AWS ParallelCluster 对队列中的计算资源执行 GPU 运行状况检查。
- `Gpu` 运行状况检查会对计算资源执行 GPU 运行状况检查，以防止在 GPU 降级的节点上提交作业。
- 如果某个计算节点未通过 `Gpu` 运行状况检查，则该计算节点的状态将更改为 `DRAIN`。新作业不会在此节点上启动。现有作业将运行至完成。所有正在运行的作业完成后，如果该计算节点是动态节点，则会终止；如果是静态节点，则会被替换。
- `Gpu`运行状况检查的持续时间取决于所选实例类型、实例 GPUs 中的数量和`Gpu`运行状况检查目标的数量（等同于作业 GPU 目标的数量）。对于具有 8 的实例 GPUs，典型持续时间小于 3 分钟。
- 如果 `Gpu` 运行状况检查在不受支持的实例上运行，它将退出，作业将在计算节点上运行。例如，如果一个实例没有 GPU，或者一个实例有 GPU，但不是 NVIDIA GPU，则运行状况检查将会退出，作业将在计算节点上运行。仅支持 NVIDIA GPUs 。
- `Gpu` 运行状况检查使用 `dcgmi` 工具对节点执行运行状况检查，并采取以下步骤：

  当在节点中开始 `Gpu` 运行状况检查时：

  1. 它会检测 `nvidia-dcgm` 和 `nvidia-fabricmanager` 服务是否正在运行。
  2. 如果这些服务未运行，则 `Gpu` 运行状况检查将会启动这些服务。
  3. 它会检测是否启用了持久性模式。
  4. 如果未启用持久性模式，则 `Gpu` 运行状况检查将会启用该模式。

  在运行状况检查结束时，`Gpu` 运行状况检查会将这些服务和资源还原到其初始状态。
- 如果任务分配给一组特定的节点 GPUs，则运行`Gpu`状况检查仅在该特定节点上运行。否则，运行`Gpu`状况检查将在节点 GPUs 中的所有节点上运行。
- 如果计算节点同时收到 2 个或更多个 `Gpu` 运行状况检查请求，则仅运行第一个运行状况检查，并跳过其他运行状况检查。目标节点的运行状况检查也是如此 GPUs。您可以查看日志文件以获取有关此情况的更多信息。
- `/var/log/parallelcluster/slurm_health_check.log` 文件中提供了特定计算节点的运行状况检查日志。该文件可在 Amazon CloudWatch 的集群 CloudWatch 日志组中找到，您可以在其中找到：
  - 有关 `Gpu` 运行状况检查运行的操作的详细信息，包括启用和禁用服务以及持久性模式。
  - GPU 标识符、序列号和 UUID。
  - 运行状况检查输出。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> HealthChecks从 3.6.0 AWS ParallelCluster 版开始受支持。

Networking

（必需）定义 Slurm 队列的网络配置。

```
Networking:
  SubnetIds:
    - string
  AssignPublicIp: boolean
  SecurityGroups:
    - string
  AdditionalSecurityGroups:
    - string
  PlacementGroup:
    Enabled: boolean
    Id: string
    Name: string
  Proxy:
    HttpProxyAddress: string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Networking 属性

SubnetIds（必需，**[String]**）

您在 IDs 其中配置Slurm队列的现有子网。

如果您在 SlurmQueues/ComputeResources/InstanceType 中配置实例类型，则只能定义一个子网。

如果您在 SlurmQueues/ComputeResources/Instances 中配置实例类型，则可以定义单个子网或多个子网。

如果您使用多个子网，则为队列定义的所有子网都必须位于同一 VPC 中，每个子网位于单独的可用区 (AZ) 中。

例如，假设您为队列定义了 subnet-1 和 subnet-2。

则 `subnet-1` 和 `subnet-2` 不能都在 AZ-1 中。

`subnet-1` 可以在 AZ-1 中，`subnet-2` 可以在 AZ-2 中。

如果您只配置一种实例类型并想要使用多个子网，请在 `Instances` 而不是 `InstanceType` 中定义您的实例类型。

例如，定义 `ComputeResources/Instances/InstanceType=instance.type` 而不是 `ComputeResources/InstanceType=instance.type`。

> **Note**
>
> 不支持在不同的可用区之间使用 Elastic Fabric Adapter (EFA)。

使用多个可用区可能会导致存储网络延迟增加，并提高可用区间的数据传输成本。例如，当实例访问位于不同 AZ 的文件存储时，可能会发生这种情况。有关更多信息，请参阅同一 AWS 区域内的数据传输。

集群更新为从使用单个子网改为使用多个子网：

- 假设集群的子网定义是用单个子网和一个 Lustre AWS ParallelCluster 托管 FSx 文件系统定义的。则您无法使用更新的子网 ID 定义直接更新此集群。要更新该集群，必须先将托管文件系统更改为外部文件系统。有关更多信息，请参阅 将 AWS ParallelCluster 托管存储转换为外部存储。
- 假设集群的子网定义是用单个子网和一个外部 Amazon EFS 文件系统定义的，前提是定义为要添加的多个子网的所有子网都不存在 EFS 挂载目标。 AZs 则您无法使用更新的子网 ID 定义直接更新此集群。要更新集群或创建集群，必须先为已定义的多个子网的所有创建所有挂载目标。 AZs

可用区和集群容量预留定义于 CapacityReservationResourceGroupArn：

- 如果定义的容量预留资源组所涵盖的实例类型和可用区集合与为队列定义的实例类型和可用区集合之间没有重叠，则无法创建集群。
- 如果定义的容量预留资源组所涵盖的实例类型和可用区集与为该队列定义的一组实例类型和可用区之间存在部分重叠，则可以创建集群。 AWS ParallelCluster 会发送一条警告消息，说明这种情况存在部分重叠。
- 有关更多信息，请参阅 使用按需容量预留（ODCR）启动实例。

> **Note**
>
> 3.4.0 AWS ParallelCluster 版本中添加了多个可用区。

> **Warning**
>
> 此警告适用于 3.3.1 之前的所有 3.x.y AWS ParallelCluster 版本。 AWS ParallelCluster 如果更改此参数，版本 3.3.1 不会受到影响。
> 对于 3.3.1 之前的 AWS ParallelCluster 3 个版本：
> 如果您更改此参数并更新群集，则会创建一个新的 Lustre 托管 FSx 文件系统，并在不保留现有数据的情况下删除现有 FSx 的 Lustre 托管文件系统。这会导致数据丢失。在继续操作之前，如果要保留数据，请务必备份现有 FSx Lustre 文件系统的数据。有关更多信息，请参阅 fo r Lustre 用户指南中的使用备份。FSx

如果添加了新子网值，更新策略：可以在更新期间更改此设置。

如果删除了子网值，更新策略：必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

`AssignPublicIp`（可选，**`String`**）

为 Slurm 队列中的节点创建或分配公有 IP 地址。支持的值为 `true` 和 `false`。您指定的子网决定默认值。一个子网， IPs 默认分配公有 IP 地址。

如果您定义了p4d或hpc6id实例类型，或者其他具有多个网络接口或网络接口卡的实例类型，则必须将 `HeadNode`/`Networking`/设置为`ElasticIp``true`以提供公共访问权限。 AWS public IPs 只能分配给使用单个网络接口启动的实例。对于这种情况，我们建议您使用 NAT 网关为集群计算节点提供公有访问权限。在这种情况下，请将 `AssignPublicIp` 设置为 `false`。有关 IP 地址的更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。

更新策略：如果更改此设置，则不允许更新。

`SecurityGroups`（可选，**`[String]`**）

用于Slurm 队列的安全组的列表。如果未指定安全组，则会为您 AWS ParallelCluster 创建安全组。

验证您的SharedStorage系统是否正确配置了安全组。

> **Warning**
>
> 此警告适用于所有 3. *x*。 *y* AWS ParallelCluster 3.3.0 之前的版本。 AWS ParallelCluster 如果更改此参数，版本 3.3.0 不会受到影响。
> 对于 3.3.0 之前的 AWS ParallelCluster 3 个版本：
> 如果您更改此参数并更新群集，则会创建一个新的 Lustre 托管 FSx 文件系统，并在不保留现有数据的情况下删除现有 FSx 的 Lustre 托管文件系统。这会导致数据丢失。如果要保留数据，请务必备份现有 F FSx or Lustre 文件系统的数据。有关更多信息，请参阅 fo r Lustre 用户指南中的使用备份。FSx

> **Warning**
>
> 如果您为计算实例启用 Efa，请确保启用了 EFA 的实例是允许进出自身的所有入站和出站流量的安全组的成员。

更新策略：可以在更新期间更改此设置。

`AdditionalSecurityGroups`（可选，**`[String]`**）

用于Slurm 队列的其他安全组的列表。

更新策略：可以在更新期间更改此设置。

`PlacementGroup`（可选）

指定 Slurm 队列的置放群组设置。

```
PlacementGroup:
  Enabled: boolean
  Id: string
  Name: string
```

更新策略：必须停止所有计算节点才能删除托管置放群组。必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

`Enabled`（可选，**`Boolean`**）

指示是否对 Slurm 队列使用置放群组。默认值为 `false`。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Id（可选，**String**）

Slurm队列使用的现有集群置放群组的置放群组 ID。请务必提供置放群组 ID 而不是名称。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Name（可选，**String**）

Slurm 队列使用的现有集群置放群组的置放群组名称。确保提供置放群组名称 而不是 ID。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

> **Note**
>
> - 如果 PlacementGroup/Enabled 设置为 true 而未定义 Name 或 Id，则会为每个计算资源分配自己的托管置放群组，除非将 ComputeResources/Networking/PlacementGroup 定义为覆盖此设置。
> - 从 AWS ParallelCluster 版本3.3.0开始，添加Name了 SlurmQueuesNetworkingPlacementGroup///作为/ SlurmQueuesNetworkingPlacementGroup/Id的首选替代方案。
>
> PlacementGroup/Id 和 PlacementGroup/Name 是等效的。您可以使用任何一个。
>
> 如果同时包含 PlacementGroup/Id和 PlacementGroup/Name， AWS ParallelCluster 则失败。您只能选择其中一项。
>
> 您无需更新集群即可使用 PlacementGroup/Name。

Proxy（可选）

指定 Slurm 队列的代理设置。

```
Proxy:
  HttpProxyAddress: string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

HttpProxyAddress（可选，**String**）

为 Slurm 队列定义 HTTP 或 HTTPS 代理服务器。通常为 `https://x.x.x.x:8080`。

没有默认值。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Image

（可选）指定要用于 Slurm 队列的映像。要对所有节点使用相同的 AMI，请使用Image部分中的CustomAmi设置。

```
Image:
  CustomAmi: string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Image 属性

CustomAmi（可选，**String**）

用于Slurm队列的 AMI，而不是默认的 AMI AMIs。您可以使用 C pcluster LI 命令查看默认值列表 AMIs。

> **Note**
> AMI 必须基于头节点所使用的相同操作系统。

```
pcluster list-official-images
```

如果自定义 AMI 需要其他权限才能启动，则必须将这些权限添加到头节点策略中。

例如，如果自定义 AMI 具有与之关联的加密快照，则头节点策略中需要以下其他策略：

```
{
```

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

要排查自定义 AMI 验证警告，请参阅排查自定义 AMI 问题。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

ComputeResources

（必需）定义 Slurm 队列的 ComputeResources 配置。

> **Note**
>
> - 在更新期间，集群大小可能会更改。有关更多信息，请参阅集群容量大小和更新。
> - 只有将新的计算资源部署在属于创建集群时存在的 CIDR 块的子网中，才能将其添加到集群。

```
ComputeResources:
  - Name: string
    InstanceType: string
    Instances:
      - InstanceType: string
    MinCount: integer
    MaxCount: integer
```

```
    DynamicNodePriority: integer
    StaticNodePriority: integer
    SpotPrice: float
    DisableSimultaneousMultithreading: boolean
    SchedulableMemory: integer
    HealthChecks:
      Gpu:
        Enabled: boolean
    Efa:
      Enabled: boolean
      GdrSupport: boolean
    CapacityReservationTarget:
      CapacityReservationId: string
      CapacityReservationResourceGroupArn: string
    Networking:
      PlacementGroup:
        Enabled: boolean
        Name: string
    CustomSlurmSettings: dict
    Tags:
      - Key: string
        Value: string
```

更新策略：对于此列表值设置，可以在更新期间添加新值，或者在删除现有值时必须停止计算实例集。

ComputeResources 属性

Name（必需，**String**）

Slurm 队列计算环境的名称。名称最多可以包含 25 个字符。

更新策略：如果更改此设置，则不允许更新。

InstanceType（必需，**String**）

此 Slurm 计算资源中使用的实例类型。集群中的所有实例类型都必须使用相同的处理器架构。实例可以使用 x86_64 或 arm64 架构。

集群配置必须定义InstanceType或实例。如果两者都被定义， AWS ParallelCluster 则失败。

定义 InstanceType 时，不能定义多个子网。如果您只配置一种实例类型并想要使用多个子网，请在 Instances 而不是 InstanceType 中定义您的实例类型。有关更多信息，请参阅 Networking/SubnetIds。

如果您定义了p4d或hpc6id实例类型，或者其他具有多个网络接口或网络接口卡的实例类型，则必须按照中所述在私有子网中启动计算实例AWS ParallelCluster 使用两个子网。 AWS public IPs 只能分配给使用单个网络接口启动的实例。有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

Instances（必需）

指定计算资源的实例类型列表。要为实例类型列表指定分配策略，请参阅 `AllocationStrategy`。

集群配置必须定义 `InstanceType` 或 `Instances`。如果同时定义了两者，则 AWS ParallelCluster 将会失败。

有关更多信息，请参阅 Slurm 的多实例类型分配。

```
Instances:
  - InstanceType: string
```

> **Note**
>
> 从 3.7.0 AWS ParallelCluster 版开始，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`可以启用。
> 适用于 3.2.0 到 3.6 AWS ParallelCluster 版本。 *x*，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`无法启用。

更新策略：对于此列表值设置，可以在更新期间添加新值，或者在删除现有值时必须停止计算实例集。

InstanceType（必需，**String**）

要在此 Slurm 计算资源中使用的实例类型。集群中的所有实例类型都必须使用相同的处理器架构，即 `x86_64` 或 `arm64`。

`Instances` 中列出的实例类型必须：

- 相同数量的 vCPUs，或者，如果设置`DisableSimultaneousMultithreading`为`true`，则为相同数量的内核。
- 具有相同制造商的相同数量的加速器。

- 支持 EFA，如果 `Efa`/`Enabled` 设置为 `true`。

`Instances` 中列出的实例类型可以具有：

- 不同的内存量。

  在这种情况下，应将最小内存设置为可消耗的 Slurm 资源。

  > **Note**
  >
  > 从 3.7.0 AWS ParallelCluster 版开始，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`可以启用。
  > 适用于 3.2.0 到 3.6 AWS ParallelCluster 版本。 *x*，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`无法启用。

- 不同的网卡。

  在这种情况下，为计算资源配置的网络接口数量由网卡数量最少的实例类型定义。

- 不同的网络带宽。
- 不同的实例存储大小。

如果您定义了p4d或hpc6id实例类型，或者其他具有多个网络接口或网络接口卡的实例类型，则必须按照中所述在私有子网中启动计算实例AWS ParallelCluster 使用两个子网。 AWS public IPs 只能分配给使用单个网络接口启动的实例。有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的在实例启动期间分配公有 IPv4 地址。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

> **Note**
>
> `Instances`从 3.3.0 AWS ParallelCluster 版开始受支持。

`MinCount`（可选，**`Integer`**）

Slurm 计算资源使用的最小实例数量。默认值是 0。

> **Note**
>
> 在更新期间，集群大小可能会更改。有关更多信息，请参阅集群容量大小和更新

更新策略：必须停止计算实例集才能更改此设置以进行更新。

MaxCount（可选，**Integer**）

Slurm 计算资源使用的最大实例数量。默认值为 10。

使用时`CapacityType = CAPACITY_BLOCK`，`MaxCount`必须等于`MinCount`和大于 0，因为容量块预留的所有实例部分都作为静态节点进行管理。

在创建集群时，头节点会等待所有静态节点准备就绪，然后再发出集群创建成功的信号。但是，当您使用时`CapacityType = CAPACITY_BLOCK`，此检查将不考虑与容量块关联的计算资源的节点部分。即使不是所有配置的容量块都处于活动状态，也会创建集群。

> **Note**
>
> 在更新期间，集群大小可能会更改。有关更多信息，请参阅集群容量大小和更新

DynamicNodePriority（可选，**Integer**）

队列计算资源中动态节点的优先级。该优先级映射到计算资源动态节点的 Slurm 节点 `Weight` 配置参数。默认值为 `1000`。

Slurm 将 `Weight` 值最低的节点设置为最高优先级。

> **Warning**
>
> 在 Slurm 分区（队列）中使用许多不同的 `Weight` 值可能会减慢队列中作业调度的速度。在 AWS ParallelCluster 3.7.0 之前的版本中，静态节点和动态节点的默认权重相同。1在这种情况下，由于静态和动态节点的命名架构，Slurm 可能会将空闲的动态节点的优先级设置为高于空闲的静态节点。如果所有其他条件相同，Slurm 会按名称的字母顺序调度节点。

> **Note**
>
> `DynamicNodePriority`已在 3.7.0 AWS ParallelCluster 版本中添加。

更新策略：可以在更新期间更改此设置。

StaticNodePriority（可选，**Integer**）

队列计算资源中静态节点的优先级。该优先级映射到计算资源静态节点的 Slurm 节点 `Weight` 配置参数。默认值为 1。

Slurm 将 `Weight` 值最低的节点设置为最高优先级。

> **Warning**
>
> 在 Slurm 分区（队列）中使用许多不同的 `Weight` 值可能会减慢队列中作业调度的速度。

> **Note**
>
> `StaticNodePriority`已在 3.7.0 AWS ParallelCluster 版本中添加。

更新策略：可以在更新期间更改此设置。

SpotPrice（可选，**Float**）

在启动任何实例之前为 Amazon EC2 竞价型实例支付的最高价格。默认值为按需价格。

更新策略：必须停止计算实例集或必须设置 `QueueUpdateStrategy` 才能更改此设置以进行更新。

DisableSimultaneousMultithreading（可选，**Boolean**）

如果为 `true`，则禁用 Slurm 队列中节点上的多线程。默认值为 `false`。

并非所有实例类型都可禁用多线程。有关支持禁用多线程的实例类型列表，请参阅 A mazon EC2 用户指南中每种实例类型的 CPU 核心和每个 CPU 内核的线程。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

SchedulableMemory（可选，**Integer**）

在 Slurm 参数 `RealMemory` 中为计算资源的计算节点配置的内存量，以 MiB 为单位。如果启用了 `SlurmSettings`/`EnableMemoryBasedScheduling`，则此值为可供作业使用的节点内存的上限。默认值是亚马逊 EC2 实例类型中列出并由亚马逊 EC2 API 返回的内存的 95% `DescribeInstanceTypes`。确保将以 GiB 为单位给出的值转换为 MiB 单位。

支持的值：`1-EC2Memory`

EC2Memory是在亚马逊 EC2 实例类型中列出并由亚马 EC2 逊 API 返回的内存（以 MiB 为单位）。DescribeInstanceTypes确保将以 GiB 为单位给出的值转换为 MiB 单位。

当启用了 SlurmSettings/EnableMemoryBasedScheduling 时，此选项最相关。有关更多信息，请参阅 Slurm 基于内存的调度。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.2.0 开始支持 SchedulableMemory。从版本 3.2.0 开始，默认情况下， AWS ParallelCluster 将Slurm计算节点配置RealMemory为 Ama EC2 zon API 返回的内存的 95%。DescribeInstanceTypes此配置与 EnableMemoryBasedScheduling 的值无关。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

HealthChecks（可选）

指定计算资源上的运行状况检查。

Gpu（可选）

指定计算资源上的 GPU 运行状况检查。

Enabled（可选，**Boolean**）

是否 AWS ParallelCluster 对队列中的计算资源执行 GPU 运行状况检查。默认值为 false。

> **Note**
>
> AWS ParallelCluster 在使用 alinux2 ARM 操作系统的节点Gpu中不支持HealthChecks/。这些平台不支持 NVIDIA 数据中心 GPU 管理器 (DCGM)。

Gpu 运行状况检查行为

- 如果 Gpu /设置Enabled为true，则对计算资源 AWS ParallelCluster 执行 GPU 运行状况检查。
- Gpu 运行状况检查会对计算资源执行运行状况检查，以防止在 GPU 降级的节点上提交作业。

- 如果某个计算节点未通过 Gpu 运行状况检查，则该计算节点的状态将更改为 DRAIN。新作业不会在此节点上启动。现有作业将运行至完成。所有正在运行的作业完成后，如果该计算节点是动态节点，则会终止；如果是静态节点，则会被替换。
- Gpu运行状况检查的持续时间取决于所选实例类型、实例 GPUs 中的数量和Gpu运行状况检查目标的数量（等同于作业 GPU 目标的数量）。对于具有 8 的实例 GPUs，典型持续时间小于 3 分钟。
- 如果 Gpu 运行状况检查在不受支持的实例上运行，它将退出，作业将在计算节点上运行。例如，如果一个实例没有 GPU，或者一个实例有 GPU，但不是 NVIDIA GPU，则运行状况检查将会退出，作业将在计算节点上运行。仅支持 NVIDIA GPUs 。
- Gpu 运行状况检查使用 dcgmi 工具对节点执行运行状况检查，并采取以下步骤：

  当在节点中开始 Gpu 运行状况检查时：

  1. 它会检测 nvidia-dcgm 和 nvidia-fabricmanager 服务是否正在运行。
  2. 如果这些服务未运行，则 Gpu 运行状况检查将会启动这些服务。
  3. 它会检测是否启用了持久性模式。
  4. 如果未启用持久性模式，则 Gpu 运行状况检查将会启用该模式。

  在运行状况检查结束时，Gpu 运行状况检查会将这些服务和资源还原到其初始状态。
- 如果任务分配给一组特定的节点 GPUs，则运行Gpu状况检查仅在该特定节点上运行。否则，运行Gpu状况检查将在节点 GPUs 中的所有节点上运行。
- 如果计算节点同时收到 2 个或更多个 Gpu 运行状况检查请求，则仅运行第一个运行状况检查，并跳过其他运行状况检查。针对节点的运行状况检查也是如此 GPUs。您可以查看日志文件以获取有关此情况的更多信息。
- /var/log/parallelcluster/slurm_health_check.log 文件中提供了特定计算节点的运行状况检查日志。此文件可在 Amazon CloudWatch 的集群 CloudWatch 日志组中找到，您可以在其中找到：
  - 有关 Gpu 运行状况检查运行的操作的详细信息，包括启用和禁用服务以及持久性模式。
  - GPU 标识符、序列号和 UUID。
  - 运行状况检查输出。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> HealthChecks从 3.6.0 AWS ParallelCluster 版开始受支持。

Efa（可选）

为 Slurm 队列中的节点指定 Elastic Fabric Adapter (EFA) 设置。

```
Efa:
  Enabled: boolean
  GdrSupport: boolean
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Enabled（可选，**Boolean**）

指定 Elastic Fabric Adapter (EFA) 已启用。要查看支持 EFA 的亚马逊 EC2 实例列表，请参阅亚马逊 Linux 实例 EC2 用户指南中的支持的实例类型。有关更多信息，请参阅 Elastic Fabric Adapter。我们建议您使用集群 SlurmQueues/Networking/PlacementGroup 最大限度地缩短实例之间的延迟。

默认值为 false。

> **Note**
> 不支持在不同的可用区之间使用 Elastic Fabric Adapter (EFA)。有关更多信息，请参阅 SubnetIds。

> **Warning**
> 如果您在中定义自定义安全组 SecurityGroups，请确保您的启用 EFA 的实例是允许所有入站和出站流量进入自身的安全组的成员。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

GdrSupport（可选，**Boolean**）

（可选）从 AWS ParallelCluster 版本 3.0.2 开始，此设置无效。如果计算资源和操作系统的实例类型支持 GPUDirect RDMA（远程直接内存访问）的 Elastic Fabric Adapter (EFA)，则始终启用对 RDMA（远程直接内存访问）Slurm的支持。

> **Note**
>
> AWS ParallelCluster 版本 3.0.0 到 3.0.1：已为计算资源启用对 GPUDirect RDMA 的支持。Slurm特定操作系统（是alinux2、ubuntu1804或p4d.24xlargeubuntu2004）上的特定实例类型 () 支持对 GPUDirect RDMA 的支持。Os默认值为 False。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

CapacityReservationTarget

```
CapacityReservationTarget:
  CapacityReservationId: string
  CapacityReservationResourceGroupArn: string
```

指定要用于计算资源的按需容量预留。

CapacityReservationId（可选，**String**）

要用于队列计算资源的现有容量预留的 ID。ID 可以指的是 ODCR 或适用于 ML 的容量块。

如果在计算资源级别指定此参数 InstanceType是可选的，则将自动从预留中检索该参数。

CapacityReservationResourceGroupArn（可选，**String**）

指示用作计算资源的服务相关容量预留组的资源组的 Amazon 资源名称 (ARN)。 AWS ParallelCluster 确定并使用组中最适当的容量预留。对于为计算资源列出的每种实例类型，资源组必须至少有一个 ODCR。有关更多信息，请参阅 使用按需容量预留（ODCR）启动实例。

- 如果PlacementGroup在 SlurmQueues/Networking或 SlurmQueues//中启用 Networking，则 AWS ParallelCluster 选择以实例类型PlacementGroup为目标的资源组和计算资源（如果存在）。ComputeResources

  PlacementGroup 必须以 ComputeResources 中定义的实例类型之一为目标。
- 如果PlacementGroup未在 SlurmQueues/Networking或 SlurmQueuesComputeResources/中启用 Networking，则 AWS ParallelCluster 选择仅针对计算资源的实例类型（如果存在）的资源组。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.3.0 开始添加了 `CapacityReservationTarget`。

Networking

```
Networking:
  PlacementGroup:
    Enabled: boolean
    Name: string
```

更新策略：必须停止所有计算节点才能删除托管置放群组。必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

PlacementGroup（可选）

指定计算资源的置放群组设置。

Enabled（可选，**Boolean**）

指示是否对计算资源使用置放群组。

- 如果设置为 `true` 而未定义 `Name`，则无论 SlurmQueues/Networking/PlacementGroup 设置如何，都会为该计算资源分配自己的托管置放群组。
- 如果设置为 `true` 且定义了 `Name`，则无论 `SlurmQueues/Networking/PlacementGroup` 设置如何，都将为该计算资源分配命名的置放群组。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Name（可选，**String**）

用于计算资源的现有集群置放群组的置放群组名称。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

> **Note**
>
> - 如果 `PlacementGroup/Enabled` 和 `Name` 都未设置，则它们各自的值默认为 `SlurmQueues`/`Networking`/`PlacementGroup` 设置。
> - `ComputeResources/Networking/已PlacementGroup`在 3.3.0 AWS ParallelCluster 版本中添加。

`CustomSlurmSettings`（可选，**`Dict`**）

（可选）定义自定义 Slurm 节点（计算资源）配置设置。

指定应用于 Slurm 节点（计算资源）的自定义 Slurm 配置参数键值对的字典。

每个单独的键值对（例如 `Param1: Value1`）都以 `Param1=Value1` 格式单独添加到 Slurm 节点配置行的末尾。

您只能指定未在 `CustomSlurmSettings` 中列入拒绝列表的 Slurm 配置参数。有关列入拒绝列表的 Slurm 配置参数的信息，请参阅 被拒登名单 Slurm 的配置参数 `CustomSlurmSettings`。

AWS ParallelCluster 仅检查参数是否在拒绝列表中。 AWS ParallelCluster 不会验证您的自定义 Slurm配置参数语法或语义。您有责任验证您的自定义Slurm配置参数。无效的自定义 Slurm 配置参数可能会导致 Slurm 进程守护程序失败，从而导致集群创建和更新失败。

有关如何使用指定自定义Slurm配置参数的更多信息 AWS ParallelCluster，请参阅Slurm 配置自定义。

有关 Slurm 配置参数的更多信息，请参阅 Slurm 文档中的 slurm.conf。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.6.0 开始支持 `CustomSlurmSettings`。

`Tags`（可选，[字符串]）

标签键值对的列表。`ComputeResource` 标签覆盖 `Tags` 部分或 `SlurmQueues`/`Tags` 中指定的重复标签。

Key（可选，**String**）

标签键。

Value（可选，**String**）

标签值。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

ComputeSettings

（必需）定义 Slurm 队列的 ComputeSettings 配置。

ComputeSettings 属性

指定 Slurm 队列中节点的 ComputeSettings 的属性。

```
ComputeSettings:
  LocalStorage:
    RootVolume:
      Size: integer
      Encrypted: boolean
      VolumeType: string
      Iops: integer
      Throughput: integer
     EphemeralVolume:
      MountDir: string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

LocalStorage（可选）

指定 Slurm 队列中节点的 LocalStorage 的属性。

```
LocalStorage:
  RootVolume:
    Size: integer
    Encrypted: boolean
    VolumeType: string
    Iops: integer
    Throughput: integer
```

```
  EphemeralVolume:
    MountDir: string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

RootVolume（可选）

指定 Slurm 队列中节点的根卷的详细信息。

```
RootVolume:
  Size: integer
  Encrypted: boolean
  VolumeType: string
  Iops: integer
  Throughput: integer
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Size（可选，**Integer**）

指定 Slurm 队列中节点的根卷大小，以吉字节 (GiB) 为单位。默认大小来自 AMI。如果使用不同的大小，则 AMI 必须支持 `growroot`。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Encrypted（可选，**Boolean**）

如果为 `true`，则对 Slurm 队列中节点的根卷进行加密。默认值为 `false`。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

VolumeType（可选，**String**）

指定 Slurm 队列中节点的 Amazon EBS 卷类型。支持的值为 `gp2`、`gp3`、`io1`、`io2`、`sc1`、`st1` 和 `standard`。默认值为 `gp3`。

有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 Amaz on EBS 卷类型。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Iops（可选，**Boolean**）

定义 io1、io2 和 gp3 类型卷的 IOPS 数。

默认值、支持的值以及 volume_size/volume_iops 比率因 VolumeType 和 Size 而异。

VolumeType = io1

默认值：Iops = 100

支持的值：Iops = 100–64000 †

最大 volume_iops/volume_size 比率 = 50 IOPS/GiB。5000 IOPS 需要至少 100 GiB 的 volume_size。

VolumeType = io2

默认值：Iops = 100

支持的值：Iops = 100–64000（io2 Block Express 卷为 256000）†

最大 Iops/Size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 Size。

VolumeType = gp3

默认值：Iops = 3000

支持的值：Iops = 3000–16000 †

最大 Iops/Size 比率 = 500 IOPS/GiB（对于 IOPS 大于 3000 的卷）。

† 只有在 Nitro System 上构建的实例也配置超过 32000 IOPS 时，才能保证最大 IOPS。其他实例最高可具有 32000 IOPS。除非您修改卷，否则较早的 io1 卷可能无法实现完全性能。io2Block Express 卷在 R5b 实例类型上支持高达 256000 的 volume_iops 值。有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 “io2屏蔽 Express 卷”。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Throughput（可选，**Integer**）

定义 gp3 卷类型的吞吐量，以 MiB/s 为单位。此设置仅在 VolumeType 为 gp3 时有效。默认值为 125。支持的值：125–1000 MiB/s

Throughput 与 Iops 的比率不能超过 0.25。如果最大吞吐量为 1000，则该Iops设置 MiB/s 必须至少为 4000。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

EphemeralVolume（可选，**Boolean**）

指定临时卷的设置。临时卷是通过将所有实例存储卷合并到 ext4 文件系统格式的单个逻辑卷而创建的。默认值为 /scratch。如果实例类型没有任何实例存储卷，则不会创建临时卷。有关更多信息，请参阅 Amazon EC2 用户指南中的实例存储卷。

```
EphemeralVolume:
  MountDir: string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

MountDir（可选，**String**）

Slurm 队列中每个节点的临时卷的挂载目录。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

CustomActions

（可选）指定要在 Slurm 队列中的节点上运行的自定义脚本。

```
CustomActions:
  OnNodeStart:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
  OnNodeConfigured:
    Sequence:
      - Script: string
```

```
        Args:
          - string
    Script: string
    Args:
      - string
```

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

CustomActions 属性

OnNodeStart（可选，**String**）

指定在启动任何节点部署引导操作之前，要在 Slurm 队列中的节点上运行的脚本序列或单个脚本。AWS ParallelCluster 不支持同一个自定义操作同时包含单个脚本和 Sequence。有关更多信息，请参阅自定义引导操作。

Sequence（可选）

要运行的脚本的列表。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Script（必需，**String**）

要使用的文件。文件路径可以 `https://` 或 `s3://` 开头。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Args（可选，**[String]**）

要传递到脚本的参数的列表。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Script（必需，**String**）

用于单个脚本的文件。文件路径可以 `https://` 或 `s3://` 开头。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Args（可选，**[String]**）

要传递到单个脚本的参数的列表。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

OnNodeConfigured（可选，**String**）

指定在所有节点引导操作完成之后，要在 Slurm 队列中的节点上运行的脚本序列或单个脚本。AWS ParallelCluster 不支持同一个自定义操作同时包含单个脚本和 Sequence。有关更多信息，请参阅自定义引导操作。

Sequence（可选）

要运行的脚本的列表。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Script（必需，**String**）

要使用的文件。文件路径可以 `https://` 或 `s3://` 开头。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Args（可选，**[String]**）

要传递到脚本的参数的列表。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Script（必需，**String**）

用于单个脚本的文件。文件路径可以 `https://` 或 `s3://` 开头。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

Args（可选，**[String]**）

要传递到单个脚本的参数的列表。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

> **Note**
>
> Sequence是从 3.6.0 AWS ParallelCluster 版本开始添加的。指定后Sequence，您可以列出一个自定义操作的多个脚本。 AWS ParallelCluster 继续支持使用单个脚本配置自定义操作，不包括脚本Sequence。
>
> AWS ParallelCluster 不支持同时包含单个脚本和Sequence同一个自定义操作。

Iam

（可选）定义 Slurm 队列的可选 IAM 设置。

```
Iam:
  S3Access:
    - BucketName: string
      EnableWriteAccess: boolean
      KeyName: string
  AdditionalIamPolicies:
    - Policy: string
  InstanceProfile: string
  InstanceRole: string
```

更新策略：可以在更新期间更改此设置。

Iam 属性

InstanceProfile（可选，**String**）

指定用于覆盖 Slurm 队列默认实例角色或实例配置文件的实例配置文件。您不能同时指定 InstanceProfile 和 InstanceRole。格式为 arn:${Partition}:iam::${Account}:instance-profile/${InstanceProfileName}。

如果指定此设置，则不能指定 S3Access 和 AdditionalIamPolicies 设置。

我们建议您指定 S3Access 和 AdditionalIamPolicies 设置中的一个或两个，因为添加到 AWS ParallelCluster 中的功能通常需要新权限。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

InstanceRole（可选，**String**）

指定用于覆盖 Slurm 队列默认实例角色或实例配置文件的实例角色。您不能同时指定 InstanceProfile 和 InstanceRole。格式为 arn:${Partition}:iam::${Account}:role/${RoleName}。

如果指定此设置，则不能指定 S3Access 和 AdditionalIamPolicies 设置。

我们建议您指定 S3Access 和 AdditionalIamPolicies 设置中的一个或两个，因为添加到 AWS ParallelCluster 中的功能通常需要新权限。

更新策略：可以在更新期间更改此设置。

S3Access（可选）

为 Slurm 队列指定存储桶。此设置用于生成针对 Slurm 队列中的存储桶授予指定访问权限的策略。

如果指定此设置，则不能指定 InstanceProfile 和 InstanceRole 设置。

我们建议您指定 S3Access 和 AdditionalIamPolicies 设置中的一个或两个，因为添加到 AWS ParallelCluster 中的功能通常需要新权限。

```
S3Access:
  - BucketName: string
    EnableWriteAccess: boolean
    KeyName: string
```

更新策略：可以在更新期间更改此设置。

BucketName（必需，**String**）

存储桶的名称。

更新策略：可以在更新期间更改此设置。

KeyName（可选，**String**）

存储桶的密钥。默认值为 *。

更新策略：可以在更新期间更改此设置。

EnableWriteAccess（可选，**Boolean**）

指示是否为存储桶启用写入权限。

更新策略：可以在更新期间更改此设置。

AdditionalIamPolicies（可选）

指定适用于亚马逊 IAM 策略的亚马逊资源名称 (ARNs) 列表 EC2 。除了所需的权限外，此列表还附在用于Slurm队列的根角色上 AWS ParallelCluster。

IAM 策略名称及其 ARN 不相同。不能使用名称。

如果指定此设置，则不能指定 `InstanceProfile` 和 `InstanceRole` 设置。

我们建议您使用 `AdditionalIamPolicies`，因为 `AdditionalIamPolicies` 已经添加到 AWS ParallelCluster 所需的权限中，而 `InstanceRole` 必须包含所有必需的权限。随着功能的不断添加，所需权限通常会随版本发生变化。

没有默认值。

```
AdditionalIamPolicies:
  - Policy: string
```

更新策略：可以在更新期间更改此设置。

Policy（必需，**[String]**）

IAM 策略的列表。

更新策略：可以在更新期间更改此设置。

SlurmSettings

（可选）为 Slurm 定义应用到整个集群的设置。

```
SlurmSettings:
  ScaledownIdletime: integer
  QueueUpdateStrategy: string
  EnableMemoryBasedScheduling: boolean
```

```
  CustomSlurmSettings: [dict]
  CustomSlurmSettingsIncludeFile: string
  Database:
    Uri: string
    UserName: string
    PasswordSecretArn: string
  ExternalSlurmdbd:
    Host: string
    Port: integer
  Dns:
    DisableManagedDns: boolean
    HostedZoneId: string
    UseEc2Hostnames: boolean
```

SlurmSettings 属性

ScaledownIdletime（可选，**Integer**）

定义 Slurm 节点终止前没有作业的时间（以分钟为单位）。

默认值为 10。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

MungeKeySecretArn（可选，**String**）

纯文本 Secrets AWS Manager 密钥的亚马逊资源名称 (ARN)，其中包含要在集群中使用的 base64 编码的 munge 密钥。Slurm该 munge 密钥将用于验证 Slurm 客户端命令与作为远程服务器的 Slurm 进程守护程序之间的 RPC 调用。如果未提供， AWS ParallelCluster 将 MungeKeySecretArn 为集群生成一个随机 munge 密钥。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.8.0 开始支持 `MungeKeySecretArn`。

> **Warning**
>
> 如果 MungeKeySecretArn 是新添加到现有集群中，则在回滚或稍后移除时 ParallelCluster 不会恢复之前的 munge Key。 MungeKeySecretArn相反，将生成一个新的随机 munge 密钥。

如果 AWS ParallelCluster 用户有权 DescribeSecret访问该特定机密资源， MungeKeySecretArn 则会进行验证。 MungeKeySecretArn 在以下情况下有效：

- 指定的密钥存在；
- 密钥为明文，包含一个有效的 base64 编码字符串；
- 解码后的二进制 munge 密钥大小介于 256 和 8192 位之间。

如果 pcluster 用户 IAM 策略不包括 DescribeSecret， MungeKeySecretArn则不进行验证并显示警告消息。有关更多信息，请参阅 基本 AWS ParallelCluster `pcluster` 用户策略。

更新时 MungeKeySecretArn，必须停止计算队列和所有登录节点。

如果密钥 ARN 中的密钥值被修改，而 ARN 保持不变，则不会自动用新的 munge 密钥更新集群。要使用密钥 ARN 的新 munge 密钥，必须停止计算实例集和登录节点，然后从头节点运行以下命令。

```
sudo /opt/parallelcluster/scripts/slurm/update_munge_key.sh
```

运行命令后，您可以恢复计算队列和登录节点：新配置的计算和登录节点将自动开始使用新的 munge 密钥。

要生成 base64 编码的自定义 munge 密钥，可以使用 munge 软件随附的 mungekey 实用程序，然后使用操作系统中常见的 base64 实用程序对其进行编码。或者，也可以使用 bash（请将 bs 参数设置在 32 至 1024 之间）

```
dd if=/dev/random bs=128 count=1 2>/dev/null | base64 -w 0
```

或 Python，如下所示：

```
import random
import os
import base64

# key length in bytes
key_length=128

base64.b64encode(os.urandom(key_length)).decode("utf-8")
```

更新策略：NEW UPDATE POLICY WITH COMPUTE FLEET AND LOGIN NODES STOPPED（3.7.0 中错误地未添加）。

QueueUpdateStrategy（可选，**String**）

为具有以下更新策略的 SlurmQueues 部分参数指定替换策略：

更新策略：必须停止计算实例集或必须设置 QueueUpdateStrategy 才能更改此设置以进行更新。

仅在集群更新过程开始时使用 QueueUpdateStrategy 值。

有效值：COMPUTE_FLEET_STOP |DRAIN |TERMINATE

默认值：COMPUTE_FLEET_STOP

DRAIN

队列中更改了参数值的节点将设置为 DRAINING。处于此状态的节点不接受新作业，正在运行的作业会继续运行，直至完成。

在节点变成 idle (DRAINED) 之后，如果是静态节点，则将替换该节点；如果是动态节点，则将终止该节点。其他队列中未更改参数值的其他节点不受影响。

此策略替换更改了参数值的所有队列节点所需的时间取决于正在运行的工作负载。。

COMPUTE_FLEET_STOP

QueueUpdateStrategy 参数的默认值。使用此设置，更新该SlurmQueues部分下的参数要求您在执行集群更新之前停止计算队列：

```
$ pcluster update-compute-fleet --status STOP_REQUESTED
```

TERMINATE

在更改了参数值的队列中，将会终止正在运行的作业并立即关闭节点。

静态节点将被替换，动态节点将被终止。

其他队列中未更改参数值的其他节点不受影响。

更新策略：在更新期间不分析此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.2.0 开始支持 QueueUpdateStrategy。

EnableMemoryBasedScheduling（可选，**Boolean**）

如果为 `true`，则在 Slurm 中启用基于内存的调度。有关更多信息，请参阅 SlurmQueues/ComputeResources/SchedulableMemory。

默认值为 `false`。

> **Warning**
> 启用基于内存的调度会影响 Slurm 调度器处理作业和节点分配的方式。
> 有关更多信息，请参阅 Slurm 基于内存的调度。

> **Note**
> 从 AWS ParallelCluster 版本 3.2.0 开始支持 `EnableMemoryBasedScheduling`。

> **Note**
> 从 3.7.0 AWS ParallelCluster 版开始，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`可以启用。
> 适用于 3.2.0 到 3.6 AWS ParallelCluster 版本。 *x*，如果您在实例中配置了多个实例类型，则`EnableMemoryBasedScheduling`无法启用。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

CustomSlurmSettings（可选，**[Dict]**）

定义应用到整个集群的自定义 Slurm 设置。

指定要附加到 AWS ParallelCluster 生成的 `slurm.conf` 文件末尾的键值对的 Slurm 配置字典列表。

列表中的每个字典都显示为添加到 Slurm 配置文件中的单独一行。您可指定简单参数或复杂参数。

简单参数包含单个键值对，如以下示例所示：

```
- Param1: 100
- Param2: "SubParam1,SubParam2=SubValue2"
```

在 Slurm 配置中呈现的示例：

```
Param1=100
Param2=SubParam1,SubParam2=SubValue2
```

复杂的 Slurm 配置参数包含多个以空格分隔的键值对，如以下示例所示：

```
- NodeName: test-nodes[1-10]
  CPUs: 4
  RealMemory: 4196
  ... # other node settings
- NodeSet: test-nodeset
  Nodes: test-nodes[1-10]
  ... # other nodeset settings
- PartitionName: test-partition
  Nodes: test-nodeset
  ... # other partition settings
```

在 Slurm 配置中呈现的示例：

```
NodeName=test-nodes[1-10] CPUs=4 RealMemory=4196 ... # other node settings
NodeSet=test-nodeset Nodes=test-nodes[1-10] ... # other nodeset settings
PartitionName=test-partition Nodes=test-nodeset ... # other partition settings
```

> **Note**
>
> 自定义 Slurm 节点的名称中不得包含 `-st-` 或 `-dy-` 模式。这些模式是为 AWS ParallelCluster托管的节点预留的。

如果您在 `CustomSlurmSettings` 中指定自定义 Slurm 配置参数，则不得为 `CustomSlurmSettingsIncludeFile` 指定自定义 Slurm 配置参数。

您只能指定未在 `CustomSlurmSettings` 中列入拒绝列表的 Slurm 配置参数。有关列入拒绝列表的 Slurm 配置参数的信息，请参阅 被拒登名单 Slurm 的配置参数 `CustomSlurmSettings`。

AWS ParallelCluster 仅检查参数是否在拒绝列表中。 AWS ParallelCluster 不会验证您的自定义 Slurm配置参数语法或语义。您有责任验证您的自定义Slurm配置参数。无效的自定义 Slurm 配置参数可能会导致 Slurm 进程守护程序失败，从而导致集群创建和更新失败。

有关如何使用指定自定义Slurm配置参数的更多信息 AWS ParallelCluster，请参阅Slurm 配置自定义。

有关 Slurm 配置参数的更多信息，请参阅 Slurm 文档中的 slurm.conf。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.6.0 开始支持 `CustomSlurmSettings`。

`CustomSlurmSettingsIncludeFile`（可选，**`String`**）

定义应用到整个集群的自定义 Slurm 设置。

指定将包含自定义 Slurm 配置参数的自定义 Slurm 文件附加到 AWS ParallelCluster 生成的 `slurm.conf` 文件的末尾。

必须包括该文件的路径。路径可以 `https://` 或 `s3://` 开头。

如果您为 `CustomSlurmSettingsIncludeFile` 指定自定义 Slurm 配置参数，则不得为 `CustomSlurmSettings` 指定自定义 Slurm 配置参数。

> **Note**
>
> 自定义 Slurm 节点的名称中不得包含 `-st-` 或 `-dy-` 模式。这些模式是为 AWS ParallelCluster托管的节点预留的。

您只能指定未在 `CustomSlurmSettingsIncludeFile` 中列入拒绝列表的 Slurm 配置参数。有关列入拒绝列表的 Slurm 配置参数的信息，请参阅 被拒登名单 Slurm 的配置参数 `CustomSlurmSettings`。

AWS ParallelCluster 仅检查参数是否在拒绝列表中。 AWS ParallelCluster 不会验证您的自定义Slurm配置参数语法或语义。您有责任验证您的自定义Slurm配置参数。无效的自定义 Slurm 配置参数可能会导致 Slurm 进程守护程序失败，从而导致集群创建和更新失败。

有关如何使用指定自定义Slurm配置参数的更多信息 AWS ParallelCluster，请参阅Slurm 配置自定义。

有关 Slurm 配置参数的更多信息，请参阅 Slurm 文档中的 slurm.conf。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.6.0 开始支持 `CustomSlurmSettings`。

Database

（可选）定义用于在集群上启用 Slurm 会计的设置。有关更多信息，请参阅 Slurm 用会计 AWS ParallelCluster。

```
Database:
   Uri: string
   UserName: string
   PasswordSecretArn: string
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

Database 属性

Uri（必需，**String**）

用作 Slurm 会计后端的数据库服务器的地址。此 URI 必须采用 `host:port` 格式且不得包含架构，例如 `mysql://`。主机可以是 IP 地址，也可以是头节点可解析的 DNS 名称。如果未提供端口，AWS ParallelCluster 将使用 MySQL 默认端口 3306。

AWS ParallelCluster 将Slurm记账数据库引导到集群，并且必须访问该数据库。

在执行以下操作之前，必须可以访问该数据库：

- 创建集群。
- 通过集群更新启用 Slurm 会计。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

UserName（必需，**String**）

Slurm 用于连接数据库、写入会计日志和执行查询的身份。用户必须对数据库具有读取和写入权限。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

PasswordSecretArn（必需，**String**）

包含UserName纯文本密码的 AWS Secrets Manager 密钥的 Amazon 资源名称 (ARN)。此密码与 UserName 和 Slurm 会计一起使用，用于在数据库服务器上进行身份验证。

> **Note**
>
> - 使用 AWS Secrets Manager 控制台创建密钥时，请务必选择 “其他类型的密钥”，选择纯文本，并且仅在密钥中包含密码文本。
> - 你不能在数据库密码中使用 “#” 字符，因为 Slurm 在 slurmdbd.conf 中不支持它。
> - 有关如何使用创建密钥 AWS Secrets Manager 的更多信息，请参阅创建 AWS Secrets Manager 密钥。

如果用户拥有 DescribeSecret 的权限，则会验证 PasswordSecretArn。如果指定的密钥存在，则 PasswordSecretArn 有效。如果用户 IAM 策略不包括 DescribeSecret，则不验证 PasswordSecretArn 并显示警告消息。有关更多信息，请参阅 基本 AWS ParallelCluster pcluster 用户策略。

更新 PasswordSecretArn 时，必须停止计算实例集。如果更改了密钥值而未更改密钥 ARN，则不会使用新数据库密码自动更新集群。要针对新密钥值更新集群，您必须在停止计算实例集后从头节点内运行以下命令。

```
$ sudo /opt/parallelcluster/scripts/slurm/update_slurm_database_password.sh
```

> **Warning**
>
> 我们建议仅在已停止计算实例集的情况下更改数据库密码以避免会计数据丢失。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

DatabaseName（可选，**String**）

数据库服务器上用于 Slurm 会计的数据库名称（由 Uri 参数定义）。

数据库名称可以包含小写字母、数字和下划线。名称长度不得超过 64 个字符。

此参数映射到 slurmdbd.conf 的 `StorageLoc` 参数。

如果未提供，`DatabaseName`则 ParallelCluster 将使用集群的名称为定义值`StorageLoc`。

允许更新 `DatabaseName`，但要注意以下几点：

- 如果数据库服务器上尚 DatabaseName 不存在具有名称的数据库，slurmdbd 将创建该数据库。您有责任根据需要重新配置新数据库（例如，添加会计实体 — 集群、账户、用户 QOSs、关联等）。
- 如果数据库服务器上 DatabaseName 已经存在同名数据库，slurmdbd 会将其用于会计功能。Slurm

更新策略：必须停止计算实例集才能更改此设置以进行更新。

> **Note**
>
> 从版本 3.3.0 开始添加了 `Database`。

ExternalSlurmdbd

（可选）定义通过外部 slurmdbd 服务器启用 Slurm Accounting 的设置。有关更多信息，请参阅使用进行Slurm会计 AWS ParallelCluster。

```
ExternalSlurmdbd:
  Host: string
  Port: integer
```

`ExternalSlurmdbd` 属性

`Host`（必需，**`String`**）

用于 Slurm 会计的外部 slurmdbd 服务器的地址。主机可以是 IP 地址，也可以是头节点可解析的 DNS 名称。

更新策略：可以在更新期间更改此设置。

`Port`（可选，**`Integer`**）

slurmdbd 服务监听的端口。默认值为 6819。

更新策略：可以在更新期间更改此设置。

Dns

（可选）为 Slurm 定义应用到整个集群的设置。

```
Dns:
  DisableManagedDns: boolean
  HostedZoneId: string
  UseEc2Hostnames: boolean
```

Dns 属性

DisableManagedDns（可选，**Boolean**）

如果为 `true`，则不创建集群的 DNS 条目，并且无法解析 Slurm 节点名称。

默认情况下， AWS ParallelCluster 创建一个 Route 53 托管区域，启动时将在其中注册节点。默认值为 `false`。如果设置`DisableManagedDns`为`true`，则托管区域不是由创建的 AWS ParallelCluster。

要了解如何使用此设置在没有互联网访问权限的子网中部署集群，请参阅AWS ParallelCluster 在无法访问互联网的单个子网中。

> **Warning**
>
> 集群需要名称解析系统才能正常运行。如果 `DisableManagedDns` 设置为 `true`，则必须提供名称解析系统。要使用亚马逊 EC2 默认 DNS，请将设置`UseEc2Hostnames`为`true`。或者配置您自己的 DNS 解析程序，并确保在启动实例时注册节点名称。例如，您可以通过配置 `CustomActions`/`OnNodeStart` 来实现这一目标。

更新策略：如果更改此设置，则不允许更新。

HostedZoneId（可选，**String**）

定义要用于集群 DNS 名称解析的自定义 Route 53 托管区 ID。如果提供，则在指定的托管区域中 AWS ParallelCluster 注册集群节点，并且不会创建托管托管区域。

更新策略：如果更改此设置，则不允许更新。

UseEc2Hostnames（可选，**Boolean**）

如果是true，则使用默认 EC2主机名配置集群计算节点。还会使用此信息更新 Slurm NodeHostName。默认值为 false。

要了解如何使用此设置在没有互联网访问权限的子网中部署集群，请参阅AWS ParallelCluster 在无法访问互联网的单个子网中。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.3.0 开始，此备注不相关。
> 对于 3.3.0 之前的 AWS ParallelCluster 支持版本：
> 如果设置UseEc2Hostnames为true，则使用 AWS ParallelCluster prolog和epilog脚本设置 Slurm 配置文件：
>
> - 分配了每个作业后，prolog 用于向计算节点上的 /etc/hosts 中添加节点信息。
> - epilog 用于清理 prolog 写入的内容。
>
> 要添加自定义 epilog 或 prolog 脚本，请分别将其添加到 /opt/slurm/etc/pcluster/prolog.d/ 或 /opt/slurm/etc/pcluster/epilog.d/ 文件夹。

更新策略：如果更改此设置，则不允许更新。

SharedStorage 部分

（可选）集群的共享存储设置。

AWS ParallelCluster 支持使用 Amazon EBS、FSx ONTAP 和 FSx OpenZFS 共享存储卷、Amazon EFS 和 FSx Lustre 共享存储文件系统或文件缓存。

在 SharedStorage 部分中，您可以定义外部存储或托管存储：

- 外部存储是指您管理的现有卷或文件系统。 AWS ParallelCluster 不会创建或删除它。
- AWS ParallelCluster 托管存储是指 AWS ParallelCluster 创建并可以删除的卷或文件系统。

有关共享存储配额以及有关如何配置共享存储的更多信息，请参阅使用共享存储中的 AWS ParallelCluster。

> **Note**
>
> 如果 AWS Batch 用作调度程序，FSx 则 Lustre 仅在群集头节点上可用。

```
SharedStorage:
  - MountDir: string
    Name: string
    StorageType: Ebs
    EbsSettings:
      VolumeType: string
      Iops: integer
      Size: integer
      Encrypted: boolean
      KmsKeyId: string
      SnapshotId: string
      Throughput: integer
      VolumeId: string
      DeletionPolicy: string
      Raid:
        Type: string
        NumberOfVolumes: integer
  - MountDir: string
    Name: string
    StorageType: Efs
    EfsSettings:
      Encrypted: boolean
      KmsKeyId: string
      EncryptionInTransit: boolean
      IamAuthorization: boolean
      PerformanceMode: string
      ThroughputMode: string
      ProvisionedThroughput: integer
      FileSystemId: string
      DeletionPolicy: string
      AccessPointId: string
  - MountDir: string
    Name: string
    StorageType: FsxLustre
    FsxLustreSettings:
      StorageCapacity: integer
      DeploymentType: string
```

```
      ImportedFileChunkSize: integer
      DataCompressionType: string
      ExportPath: string
      ImportPath: string
      WeeklyMaintenanceStartTime: string
      AutomaticBackupRetentionDays: integer
      CopyTagsToBackups: boolean
      DailyAutomaticBackupStartTime: string
      PerUnitStorageThroughput: integer
      BackupId: string
      KmsKeyId: string
      FileSystemId: string
      AutoImportPolicy: string
      DriveCacheType: string
      StorageType: string
      DeletionPolicy: string
      DataRepositoryAssociations:
      - Name: string
        BatchImportMetaDataOnCreate: boolean
        DataRepositoryPath: string
        FileSystemPath: string
        ImportedFileChunkSize: integer
        AutoExportPolicy: string
        AutoImportPolicy: string
  - MountDir: string
    Name: string
    StorageType: FsxOntap
    FsxOntapSettings:
      VolumeId: string
  - MountDir: string
    Name: string
    StorageType: FsxOpenZfs
    FsxOpenZfsSettings:
      VolumeId: string
  - MountDir: string
    Name: string
    StorageType: FileCache
    FileCacheSettings:
      FileCacheId: string
```

SharedStorage 更新策略

- 对于 managed/external EBS、托管 EFS 和托管 FSx Lustre，更新策略为 更新策略：对于此列表值设置，必须停止计算实例集或必须设置 QueueUpdateStrategy 才能添加新值；删除现有值时必须停止计算实例集。
- 对于外部 EFS、 FSx Lustre、 FSx ONTAP FSx OpenZfs 和文件缓存，更新策略是，更新策略：可以在更新期间更改此设置。

SharedStorage 属性

`MountDir`（必需，`String`）

共享存储的挂载路径。

更新策略：如果更改此设置，则不允许更新。

`Name`（必需，`String`）

共享存储的名称。可以在更新设置时使用此名称。

> **Warning**
>
> 如果您指定了 AWS ParallelCluster 托管共享存储，并且更改了的值`Name`，则会删除现有的托管共享存储空间和数据，并创建新的托管共享存储。通过集群更新来更改 `Name` 的值等同于用新的托管共享存储替换现有的托管共享存储。`Name`如果您需要保留现有共享存储中的数据，请务必在更改之前备份数据。

更新策略：对于此列表值设置，必须停止计算实例集或必须设置 QueueUpdateStrategy 才能添加新值；删除现有值时必须停止计算实例集。

`StorageType`（必需，`String`）

共享存储的类型。支持的值为 `Ebs`、`Efs`、`FsxLustre`、`FsxOntap` 和 `FsxOpenZfs`。

有关更多信息，请参阅`FsxLustreSettings`、`FsxOntapSettings`和`FsxOpenZfsSettings`。

> **Note**
>
> 如果您 AWS Batch 用作调度程序， FSx 则 Lustre 仅在群集头节点上可用。

更新策略：如果更改此设置，则不允许更新。

EbsSettings

（可选）Amazon EBS 卷的设置。

```
EbsSettings:
  VolumeType: string
  Iops: integer
  Size: integer
  Encrypted: boolean
  KmsKeyId: string
  SnapshotId: string
  VolumeId: string
  Throughput: integer
  DeletionPolicy: string
  Raid:
    Type: string
    NumberOfVolumes: integer
```

更新策略：如果更改此设置，则不允许更新。

EbsSettings 属性

当设置DeletionPolicy为时Delete，如果集群被删除或通过群集更新移除该卷，则该托管卷及其数据将被删除。

有关更多信息，请参阅使用 AWS ParallelCluster 中的共享存储。

VolumeType（可选，String）

指定 Amazon EBS 卷类型。支持的值为 gp2、gp3、io1、io2、sc1、st1 和 standard。默认值为 gp3。

有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 Amaz on EBS 卷类型。

更新策略：如果更改此设置，则不允许更新。

Iops（可选，Integer）

定义 io1、io2 和 gp3 类型卷的 IOPS 数。

默认值、支持的值以及 volume_size/volume_iops 比率因 VolumeType 和 Size 而异。

VolumeType = io1

默认值：Iops = 100

支持的值：Iops = 100–64000 †

最大 volume_iops/volume_size 比率 = 50 IOPS/GiB。5000 IOPS 需要至少 100 GiB 的 volume_size。

VolumeType = io2

默认值：Iops = 100

支持的值：Iops = 100–64000（io2 Block Express 卷为 256000）†

最大 Iops/Size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 Size。

VolumeType = gp3

默认值：Iops = 3000

支持的值：Iops = 3000–16000

最大 Iops/Size 比率 = 500 IOPS/GiB。5000 IOPS 需要至少 10 GiB 的 Size。

† 只有在 Nitro System 上构建的实例配置超过 32000 IOPS 时，才能保证最大 IOPS。其他实例保证最高为 32000 IOPS。除非您修改卷，否则较早的 io1 卷可能无法实现完全性能。io2Block Express 卷在 R5b 实例类型上支持高达 256000 的 volume_iops 值。有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 “io2屏蔽 Express 卷”。

更新策略：可以在更新期间更改此设置。

Size（可选，Integer）

指定卷大小，以吉字节 (GiB) 为单位。默认值为 35。

更新策略：如果更改此设置，则不允许更新。

Encrypted（可选，Boolean）

指定是否对卷进行加密。默认值为 true。

更新策略：如果更改此设置，则不允许更新。

KmsKeyId（可选，String）

指定用于加密的自定义 AWS KMS 密钥。此设置要求将 Encrypted 设置设为 true。

更新策略：如果更改此设置，则不允许更新。

SnapshotId（可选，String）

指定 Amazon EBS 快照 ID（如果使用快照作为卷的来源）。

更新策略：如果更改此设置，则不允许更新。

VolumeId（可选，String）

指定 Amazon EBS 卷 ID。为 EbsSettings 实例指定此参数后，还可以并且只能指定 MountDir 参数。

必须在 HeadNode 所在的同一可用区中创建卷。

> **Note**
>
> 3.4.0 AWS ParallelCluster 版本中添加了多个可用区。

更新策略：如果更改此设置，则不允许更新。

Throughput（可选，Integer）

为卷配置的 MiB/s 吞吐量，最大为 1,000 MiB/s。

此设置仅在 VolumeType 为 gp3 时有效。支持的范围为 125 到 1000，默认值为 125。

更新策略：可以在更新期间更改此设置。

DeletionPolicy（可选，String）

指定删除集群或删除卷时是保留卷、删除卷还是创建快照。支持的值为 Delete、Retain 和 Snapshot。默认值为 Delete。

当DeletionPolicy设置为时Delete，如果集群被删除或通过群集更新移除该卷，则该托管卷及其数据将被删除。

有关更多信息，请参阅 共享存储。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> `DeletionPolicy`从 3.2.0 AWS ParallelCluster 版开始受支持。

Raid

（可选）定义 RAID 卷的配置。

```
Raid:
  Type: string
  NumberOfVolumes: integer
```

更新策略：如果更改此设置，则不允许更新。

Raid 属性

`Type`（必需，`String`）

定义 RAID 阵列的类型。支持的值为“0”（条带）和“1”（镜像）。

更新策略：如果更改此设置，则不允许更新。

`NumberOfVolumes`（可选，`Integer`）

定义用于创建 RAID 阵列的 Amazon EBS 卷的数量。支持的值范围为 2-5。默认值（定义了 `Raid` 设置时）为 2。

更新策略：如果更改此设置，则不允许更新。

EfsSettings

（可选）Amazon EFS 文件系统的设置。

```
EfsSettings:
  Encrypted: boolean
  KmsKeyId: string
  EncryptionInTransit: boolean
  IamAuthorization: boolean
  PerformanceMode: string
  ThroughputMode: string
  ProvisionedThroughput: integer
  FileSystemId: string
```

```
  DeletionPolicy: string
  AccessPointId: string
```

更新策略：如果更改此设置，则不允许更新。

EfsSettings 属性

当DeletionPolicy设置为时Delete，如果删除了群集，或者如果通过群集更新删除了文件系统，则会删除托管文件系统及其数据。

有关更多信息，请参阅使用 AWS ParallelCluster 中的共享存储。

Encrypted（可选，Boolean）

指定是否对 Amazon EFS 文件系统加密。默认值为 false。

更新策略：如果更改此设置，则不允许更新。

KmsKeyId（可选，String）

指定用于加密的自定义 AWS KMS 密钥。此设置要求将 Encrypted 设置设为 true。

更新策略：如果更改此设置，则不允许更新。

EncryptionInTransit（可选，Boolean）

如果设置为 true，则使用传输层安全性协议 (TLS) 挂载 Amazon EFS 文件系统。默认情况下，该选项设置为 false。

> **Note**
> AWS Batch 如果用作调度程序，则EncryptionInTransit不支持。

> **Note**
> 从 AWS ParallelCluster 版本 3.4.0 开始添加了 EncryptionInTransit。

更新策略：如果更改此设置，则不允许更新。

IamAuthorization（可选，Boolean）

IamAuthorization是从 3.4.0 AWS ParallelCluster 版本开始添加的。

如果设置为 `true`，则使用系统的 IAM 身份对 Amazon EFS 进行身份验证。默认情况下，该选项设置为 `false`。

> **Note**
>
> 如果 `IamAuthorization` 被设置为 `true`，则 `EncryptionInTransit` 也必须被设置为 `true`。

> **Note**
>
> AWS Batch 如果用作调度程序，则`IamAuthorization`不支持。

更新策略：如果更改此设置，则不允许更新。

`PerformanceMode`（可选，`String`）

指定 Amazon EFS 文件系统的性能模式。支持的值为 `generalPurpose` 和 `maxIO`。默认值为 `generalPurpose`。有关更多信息，请参阅 Amazon Elastic File System User Guide 中的 Performance modes。

对于大多数文件系统，我们推荐使用 `generalPurpose` 性能模式。

使用 `maxIO` 性能模式的文件系统可以扩展到更高级别的聚合吞吐量和每秒操作数。但是，对于大多数文件操作来说，代价是稍高的延迟。

更新策略：如果更改此设置，则不允许更新。

`ThroughputMode`（可选，`String`）

指定 Amazon EFS 文件系统的吞吐量模式。支持的值为 `bursting` 和 `provisioned`。默认值为 `bursting`。使用 `provisioned` 时，必须指定 `ProvisionedThroughput`。

更新策略：可以在更新期间更改此设置。

`ProvisionedThroughput`（`ThroughputMode` 为 `provisioned` 时必需，`Integer`）

定义预配置的吞吐量（在。MiB/s) of the Amazon EFS file system, measured in MiB/s这与《亚马逊 EFS API 参考》中的ProvisionedThroughputInMibps参数相对应。

如果您使用了此参数，则必须将 `ThroughputMode` 设置为 `provisioned`。

支持的范围是 1-1024。要请求提高限制，请联系 支持。

更新策略：可以在更新期间更改此设置。

`FileSystemId`（可选，`String`）

为现有文件系统定义 Amazon EFS 文件系统 ID。

如果集群配置为跨越多个可用区，则必须在集群使用的每个可用区中定义一个文件系统挂载目标。

指定了此参数时，只能指定 `MountDir`。不能指定其他 `EfsSettings`。

如果设置此选项，则定义的文件系统必须符合以下要求：

- 文件系统必须在集群的每个可用区中具有现有的挂载目标，允许来自 `HeadNode` 和 `ComputeNodes` 的入站和出站 NFS 流量。在计划//网络 SlurmQueues/中配置了多个可用区SubnetIds。

  为确保允许集群和文件系统之间的流量，您可以执行以下操作之一：

  - 配置挂载目标的安全组以允许进出集群子网的 CIDR 或前缀列表的流量。

    > **Note**
    > AWS ParallelCluster 验证端口是否已打开以及 CIDR 或前缀列表是否已配置。 AWS ParallelCluster 不验证 CIDR 块或前缀列表的内容。

  - 通过使用 `SlurmQueues`/`Networking`/`SecurityGroups` 和 `HeadNode`/`Networking`/`SecurityGroups`，设置集群节点的自定义安全组。必须将自定义安全组配置为允许集群和文件系统之间的流量。

    > **Note**
    > 如果所有集群节点都使用自定义安全组，则 AWS ParallelCluster 仅验证端口是否已打开。 AWS ParallelCluster 无法验证源和目标的配置是否正确。

> **Warning**
> OneZone 只有当所有计算节点和头节点都位于同一个可用区时，才支持 EFS。EFS OneZone 只能有一个挂载目标。

> **Note**
>
> 3.4.0 AWS ParallelCluster 版本中添加了多个可用区。

更新策略：如果更改此设置，则不允许更新。

`DeletionPolicy`（可选，`String`）

指定从集群中删除文件系统或删除集群时，是应保留还是删除文件系统。支持的值是 `Delete` 和 `Retain`。默认值为 `Delete`。

如果设置DeletionPolicy为`Delete`，则如果删除了群集，或者如果通过群集更新删除了文件系统，则会删除托管文件系统及其数据。

有关更多信息，请参阅 共享存储。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> `DeletionPolicy`从 3.3.0 AWS ParallelCluster 版开始受支持。

`AccessPointId`（可选，`String`）

如果指定了该选项，将挂载由`access point ID` 定义的文件系统入口点，而不是文件系统根目录。

有关更多信息，请参阅 共享存储。

更新策略：如果更改此设置，则不允许更新。

`FsxLustreSettings`

> **Note**
>
> 如果为 `StorageType` 指定了 `FsxLustre`，则必须定义 `FsxLustreSettings`。

（可选）适用 FSx 于 Lustre 文件系统的设置。

```
FsxLustreSettings:
  StorageCapacity: integer
  DeploymentType: string
  ImportedFileChunkSize: integer
  DataCompressionType: string
  ExportPath: string
  ImportPath: string
  WeeklyMaintenanceStartTime: string
  AutomaticBackupRetentionDays: integer
  CopyTagsToBackups: boolean
  DailyAutomaticBackupStartTime: string
  PerUnitStorageThroughput: integer
  BackupId: string # BackupId cannot coexist with some of the fields
  KmsKeyId: string
  FileSystemId: string # FileSystemId cannot coexist with other fields
  AutoImportPolicy: string
  DriveCacheType: string
  StorageType: string
  DeletionPolicy: string
```

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 如果 AWS Batch 用作调度程序， FSx 则 Lustre 仅在群集头节点上可用。

`FsxLustreSettings` 属性

如果设置DeletionPolicy为`Delete`，则如果删除了群集，或者如果通过群集更新删除了文件系统，则会删除托管文件系统及其数据。

有关更多信息，请参阅 共享存储。

`StorageCapacity`（必需，`Integer`）

设置 Lustre 文件系统的存储容量，以 GiB FSx 为单位。 `StorageCapacity`如果要创建新的文件系统，则为必填项。如果指定了 `BackupId` 或 `FileSystemId`，则不要包含 `StorageCapacity`。

- 对于 `SCRATCH_2`、`PERSISTENT_1` 和 `PERSISTENT_2` 部署类型，有效值为 1200 GiB、2400 GiB，并以 2400 GiB 为增量。

- 对于 SCRATCH_1 部署类型，有效值为 1200 GiB、2400 GiB，并以 3600 GiB 为增量。

更新策略：如果更改此设置，则不允许更新。

DeploymentType（可选，String）

指定 Lustre 文件系统的部署类型。 FSx 支持的值有 SCRATCH_1、SCRATCH_2、PERSISTENT_1 和 PERSISTENT_2。默认值为 SCRATCH_2。

当您需要临时存储和短期处理数据时，请选择 SCRATCH_1 和 SCRATCH_2 部署类型。SCRATCH_2 部署类型提供了数据的传输中加密，以及比 SCRATCH_1 更高的突发吞吐能力。

为长期存储和不对延迟敏感的以吞吐量为重点的工作负载选择PERSISTENT_1部署类型。PERSISTENT_1支持对传输中的数据进行加密。它在所有可用 Lustr FSx e AWS 区域 的地方都可用。

对于长期存储以及需要最高级别的 IOPS 和吞吐量的延迟敏感型工作负载，请选择 PERSISTENT_2 部署类型。PERSISTENT_2 支持 SSD 存储，并提供更高的 PerUnitStorageThroughput（最高达到 1000 MB/s/TiB）。PERSISTENT_2 适用于有限数量的 AWS 区域。有关部署类型和可用 AWS 区域 位置PERSISTENT_2列表的更多信息，请参阅《Amazon FSx for Lustre 用户指南》中的 Lustre 文件系统部署选项。 FSx

当您从支持此功能的 Amazon EC2 实例访问SCRATCH_2PERSISTENT_1、或PERSISTENT_2部署类型的文件系统时，会自动启用传输中数据的加密。

从受支持的 AWS 区域中受支持的实例类型进行访问时，支持 SCRATCH_2、PERSISTENT_1 和 PERSISTENT_2 部署类型的传输中数据加密。有关更多信息，请参阅 Amazon for Lustre 用户指南 FSx 中的加密传输中的数据。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.2.0 开始增加了对 PERSISTENT_2 部署类型的支持。

更新策略：如果更改此设置，则不允许更新。

ImportedFileChunkSize（可选，Integer）

对于从数据存储库导入的文件，此值决定单个物理磁盘上存储的每个文件的条带计数和最大数据量（以 MiB 为单位）。可以对单个文件进行条带化的最大磁盘数受构成文件系统的总磁盘数限制。

默认区块大小为 1024MiB（1GiB），最大值能够达到 512000MiB（500GiB）。Amazon S3 数据元的最大大小为 5 TB。

> **Note**
>
> 使用PERSISTENT_2部署类型的文件系统不支持此参数。有关如何配置数据存储库关联的说明，请参阅 Amazon FSx for Lustre 用户指南中的将文件系统关联到 S3 存储桶。

更新策略：如果更改此设置，则不允许更新。

`DataCompressionType`（可选，`String`）

为 Lustre 文件系统设置数据压缩配置。 FSx 支持的值为LZ4。 LZ4表示该 LZ4 算法已开启数据压缩。如果未指定 `DataCompressionType`，则在创建文件系统时关闭数据压缩。

有关更多信息，请参阅 Lustre 数据压缩。

更新策略：可以在更新期间更改此设置。

`ExportPath`（可选，`String`）

Amazon S3 中导出您 FSx 的 for Lustre 文件系统根目录的路径。仅当指定了 `ImportPath` 参数时支持此设置。该路径必须使用在 `ImportPath` 中指定的相同 Amazon S3 存储桶。您可以提供一个可选的前缀，用于从 for Lustre 文件系统中导出新数据和更改数据。 FSx 如果未提供ExportPath值，则 FSx 为 Lustre 设置默认导出路s3://*amzn-s3-demo-bucket*/FSxLustre[creation-timestamp]径。此时间戳采用 UTC 格式，例如 s3://*amzn-s3-demo-bucket*/FSxLustre20181105T222312Z。

Amazon S3 导出桶必须与 `ImportPath` 指定的导入桶相同。如果仅指定存储桶名称（例如 s3://*amzn-s3-demo-bucket*），则会获得文件系统对象与 Amazon S3 存储桶对象的 1:1 映射。此映射意味着 Amazon S3 中的输入数据会在导出时被覆盖。如果您在导出路径中提供了自定义前缀（例如s3://*amzn-s3-demo-bucket*/[custom-optional-prefix]），则 Lustre 会将文件系统的内容导出到 Amazon S3 存储桶中的该导出前缀。 FSx

> **Note**
>
> 使用PERSISTENT_2部署类型的文件系统不支持此参数。按照《Amaz FSx on for Lustre 用户指南》中的将文件系统链接到 S3 存储桶中所述配置数据存储库关联。

更新策略：如果更改此设置，则不允许更新。

`ImportPath`（可选，`String`）

您用作 for Lustre 文件系统数据存储库的 Amazon S3 存储桶的路径（包括可选前缀）。 FSx 您 FSx 的 for Lustre 文件系统的根目录将映射到您选择的 Amazon S3 存储桶的根目录。例如，`s3://amzn-s3-demo-bucket/optional-prefix`。如果您在 Amazon S3 名称后指定了前缀，则只将具有该前缀的对象键加载到文件系统。

> **Note**
>
> 使用`PERSISTENT_2`部署类型的文件系统不支持此参数。按照《Amaz FSx on for Lustre 用户指南》中的将文件系统链接到 S3 存储桶中所述配置数据存储库关联。

更新策略：如果更改此设置，则不允许更新。

`WeeklyMaintenanceStartTime`（可选，`String`）

执行每周维护的首选开始时间。它采用 UTC+0 时区的 `"d:HH:MM"` 格式。对于此格式，d 是从 1 到 7 的星期几数字，从星期一开始，以星期日结束。此字段必须使用引号。

更新策略：可以在更新期间更改此设置。

`AutomaticBackupRetentionDays`（可选，`Integer`）

保留自动备份的天数。将此值设置为 0 将禁用自动备份。支持的范围是 0-90。默认值是 0。此设置仅在与 `PERSISTENT_1` 和 `PERSISTENT_2` 部署类型一起使用时有效。有关更多信息，请参阅《Amazon FSx for Lustre 用户指南》中的使用备份。

更新策略：可以在更新期间更改此设置。

`CopyTagsToBackups`（可选，`Boolean`）

如果是`true`，请将 for Lustre 文件系统的标签复制到备份中。 FSx 此值默认为 `false`。如果设置为 `true`，则会将文件系统的所有标签复制到用户未指定标签的所有自动和用户启动的备份。如果此值为 `true`，并且指定了一个或多个标签，则仅将指定的标签复制到备份。如果您在创建用户启动的备份时指定了一个或多个标签，则无论此值如何，都不会从文件系统中复制任何标签。此设置仅在与 `PERSISTENT_1` 和 `PERSISTENT_2` 部署类型一起使用时有效。

更新策略：如果更改此设置，则不允许更新。

`DailyAutomaticBackupStartTime`（可选，`String`）

每天重复的时间，格式为 `HH:MM`。`HH` 是一天中的零填充小时 (00-23)。`MM` 是小时中的零填充分钟 (00-59)。例如，`05:00` 指定每天上午 5 点。此设置仅在与 `PERSISTENT_1` 和 `PERSISTENT_2` 部署类型一起使用时有效。

更新策略：可以在更新期间更改此设置。

`PerUnitStorageThroughput`（对 `PERSISTENT_1` 和 `PERSISTENT_2` 部署类型为必需，`Integer`）

描述每 1 TiB 存储的读取和写入吞吐量（以 MB/s/TiB 为单位）。文件系统吞吐容量的计算方法是将文件系统存储容量 (TiB) 乘以 `PerUnitStorageThroughput` (MB/s/TiB). For a 2.4 TiB file system, provisioning 50 MB/s/TiBof)，得出文件系统吞吐`PerUnitStorageThroughput`量的 120 MB/s 。您需要为预置的吞吐量付费。这与该PerUnitStorageThroughput属性相对应。

有效值：

PERSISTENT_1 SSD 存储：50、100、200 MB/s/TiB。

PERSISTENT_1 HDD 存储：12、40 MB/s/TiB。

PERSISTENT_2 SSD 存储：125、250、500、1000 MB/s/TiB。

更新策略：如果更改此设置，则不允许更新。

`BackupId`（可选，`String`）

指定用于从现有备份中恢复 for Lustre 文件系统的备份 ID。 FSx 指定了 `BackupId` 设置时，不得指定 `AutoImportPolicy`、`DeploymentType`、`ExportPath`、`KmsKeyId`、`ImportPath`、`ImportedFil` 和 `PerUnitStorageThroughput` 设置。这些设置将从备份中读取。此外，不得指定 `AutoImportPolicy`、`ExportPath`、`ImportPath` 和 `ImportedFileChunkSize` 设置。这与该BackupId属性相对应。

更新策略：如果更改此设置，则不允许更新。

`KmsKeyId`（可选，`String`）

AWS Key Management Service (AWS KMS) 密钥 ID 的 ID，用于加密 for Lustre 文件系统的数据，使静态的 Lustre 文件系统具有永久性 FSx 。 FSx 如果未指定，则使用 f FSx or Lustre 托管密钥。Lustre 文件系统的`SCRATCH_1`和`SCRATCH_2` FSx 始终使用 Lustre 托管密钥 FSx 进行静态加密。有关更多信息，请参阅 AWS Key Management Service API 参考 中的 Encrypt。

更新策略：如果更改此设置，则不允许更新。

`FileSystemId`（可选，`String`）

指定 Lustre 现有 FSx 文件系统的 ID。

如果指定了此选项，则仅使用 `FsxLustreSettings` 中的 `MountDir` 和 `FileSystemId` 设置。`FsxLustreSettings` 中的所有其他设置都将被忽略。

> **Note**
>
> 如果使用 AWS Batch 调度器，则 fo FSx r Lustre 仅在头节点上可用。

> **Note**
>
> 该文件系统必须关联到通过端口 988、1021、1022 和 1023 允许入站和出站 TCP 流量的安全组。

通过执行以下操作之一，确保允许集群和文件系统之间的流量：

- 配置文件系统的安全组以允许进出集群子网的 CIDR 或前缀列表的流量。

  > **Note**
  >
  > AWS ParallelCluster 验证端口是否已打开以及 CIDR 或前缀列表是否已配置。 AWS ParallelCluster 不验证 CIDR 块或前缀列表的内容。

- 通过使用 `SlurmQueues`/`Networking`/`SecurityGroups` 和 `HeadNode`/`Networking`/`SecurityGroups`，设置集群节点的自定义安全组。必须将自定义安全组配置为允许集群和文件系统之间的流量。

  > **Note**
  >
  > 如果所有集群节点都使用自定义安全组，则 AWS ParallelCluster 仅验证端口是否已打开。 AWS ParallelCluster 无法验证源和目标的配置是否正确。

更新策略：如果更改此设置，则不允许更新。

`AutoImportPolicy`（可选，`String`）

当您创建 f FSx or Lustre 文件系统时，您的现有 Amazon S3 对象将显示为文件和目录列表。使用此属性选择 Lustre 在关联的 Amazon S3 存储桶中添加或修改对象时如何 FSx 使您的文件和目录列表保持最新。 `AutoImportPolicy`可以有以下值：

- `NEW`-自动导入已开启。 FSx for Lustre 会自动导入添加到链接的 Amazon S3 存储桶中且当前不存在于 for Lustre 文件系统中的任何新对象 FSx 的目录列表。
- `NEW_CHANGED`-自动导入已开启。 FSx for Lustre 会自动导入添加到 Amazon S3 存储桶中的任何新对象以及选择此选项后在 Amazon S3 存储桶中更改的任何现有对象的文件和目录列表。
- `NEW_CHANGED_DELETED`-自动导入已开启。 FSx for Lustre 会自动导入添加到 Amazon S3 存储桶中的任何新对象、在 Amazon S3 存储桶中更改的任何现有对象以及您选择此选项后在 Amazon S3 存储桶中删除的所有对象的文件和目录列表。

> **Note**
>
> 在 AWS ParallelCluster 版本 3.1.1 中增加了对 `NEW_CHANGED_DELETED` 的支持。

如果`AutoImportPolicy`未指定，则关闭自动导入。 FSx for Lustre 仅在创建文件系统时更新链接的 Amazon S3 存储桶中的文件和目录列表。 FSx 选择此选项后，for Lustre 不会更新任何新对象或已更改对象的文件和目录列表。

有关更多信息，请参阅 Amaz FSx on for Lustre 用户指南中的自动从 S3 存储桶导入更新。

> **Note**
>
> 使用 `PERSISTENT_2` 部署类型的文件系统不支持此参数。有关如何配置数据存储库关联的说明，请参阅 Amazon FSx for Lustre 用户指南中的将文件系统关联到 S3 存储桶。

更新策略：如果更改此设置，则不允许更新。

`DriveCacheType`（可选，`String`）

指定文件系统具有 SSD 驱动器缓存。只有将 `StorageType` 设置设为 `HDD`，且将 `DeploymentType` 设置设为 `PERSISTENT_1`，才能设置此选项。这与该DriveCacheType属性相对应。有关更多信息，FSx 请参阅《Amazon FSx for Lustre 用户指南》中的 Lustre 部署选项。

唯一有效值为 `READ`。要禁用 SSD 驱动器缓存，请不要指定 `DriveCacheType` 设置。

更新策略：如果更改此设置，则不允许更新。

`StorageType`（可选，`String`）

为您正在创建的 for FSx Lustre 文件系统设置存储类型。有效值为 `SSD` 和 `HDD`。

- 设置为 `SSD` 以使用固态驱动器存储。
- 设置为 `HDD` 以使用硬盘驱动器存储。`PERSISTENT` 部署类型支持 `HDD`。

默认值为 `SSD`。有关更多信息，请参阅亚马逊 Windows 版用户指南中的存储类型选项和亚马逊 FSx for Lustre 用户指南中的多个存储选项。 FSx

更新策略：如果更改此设置，则不允许更新。

`DeletionPolicy`（可选，`String`）

指定从集群中删除文件系统或删除集群时，是应保留还是删除文件系统。支持的值是 `Delete` 和 `Retain`。默认值为 `Delete`。

如果设置DeletionPolicy为`Delete`，则如果删除了群集，或者如果通过群集更新删除了文件系统，则会删除托管文件系统及其数据。

有关更多信息，请参阅 共享存储。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> `DeletionPolicy`从 3.3.0 AWS ParallelCluster 版开始受支持。

`DataRepositoryAssociations`（可选，`String`）

列表 DRAs （每个文件系统最多 8 个）

每个数据存储库关联都必须有一个唯一的 Amazon FSx 文件系统目录以及与之关联的唯一 S3 存储桶或前缀。

不能ImportPath在使用ExportPath和的 FsxLustreSettings 同时使用 DRAs。

更新策略：可以在更新期间更改此设置。

`Name`（必需，`String`）

DRA 的名称。可以在更新设置时使用此名称。

更新策略：如果更改此设置，则不允许更新。

`BatchImportMetaDataOnCreate`（可选，`Boolean`）

一种布尔标志，指示是否应在创建数据存储库关联后运行用于导入元数据的导入数据存储库任务。如果将此标志设置为 `true`，则该任务将运行。

默认值：`false`

更新策略：如果更改此设置，则不允许更新。

`DataRepositoryPath`（必需，`String`）

要链接到文件系统的 Amazon S3 数据存储库的路径。该路径可以是 S3 存储桶或格式 `s3://amzn-s3-demo-bucket/myPrefix/` 的前缀。此路径指定 S3 数据存储库文件将从中导入或导出到的位置。

不能与其他重叠 DRAs

模式：`^[^\u0000\u0085\u2028\u2029\r\n]{3,4357}$`

最低：`3`

最高：`4357`

更新策略：如果更改此设置，则不允许更新。

`FileSystemPath`（必需，`String`）

Amazon fo FSx r Lustre 文件系统上的路径，它指向将与 1-1 映射的高级目录（例如 `/ns1//ns1/subdir/`）或子目录（例如）。`DataRepositoryPath`名称中的前导正斜杠必填。两个数据存储库关联不能具有重叠的文件系统路径。例如，如果数据存储库与文件系统路径 `/ns1/` 相关联，则您无法将另一个数据存储库与文件系统路径 `/ns1/ns2` 相关联。

此路径指定文件将在您的文件系统中的哪个位置导出或导入到哪个位置。只能将此文件系统目录链接到一个 Amazon S3 桶，而不能将其他 S3 桶链接到该目录。

不能与其他重叠 DRAs

> **Note**
>
> 如果您仅指定正斜杠 (/) 作为文件系统路径，则只能将一个数据存储库链接到文件系统。您只能将“/”指定为与文件系统相关联的第一个数据存储库的文件系统路径。

模式：^[^\u0000\u0085\u2028\u2029\r\n]{1,4096}$

最低：1

最高：4096

更新策略：如果更改此设置，则不允许更新。

ImportedFileChunkSize（可选，Integer）

对于从数据存储库导入的文件，此值决定单个物理磁盘上存储的每个文件的条带计数和最大数据量（以 MiB 为单位）。可以对单个文件进行条带化的最大磁盘数受构成文件系统或缓存的总磁盘数限制。

默认区块大小为 1024MiB（1GiB），最大值能够达到 512000MiB（500GiB）。Amazon S3 数据元的最大大小为 5 TB。

最低：1

最高：4096

更新策略：可以在更新期间更改此设置。

AutoExportPolicy（可选，Array of strings）

该列表可包含一个或多个以下值：

- NEW - 新文件和目录将在添加到文件系统时自动导出到数据存储库。
- CHANGED - 对文件系统上的文件和目录所做的更改将自动导出到数据存储库。
- DELETED - 在文件系统上删除文件和目录后，将在数据存储库中自动删除这些文件和目录。

您可以为您的 AutoExportPolicy 定义事件类型的任意组合。

最高：3

更新策略：可以在更新期间更改此设置。

AutoImportPolicy（可选，Array of strings）

该列表可包含一个或多个以下值：

- NEW-Amazon FSx 会自动导入添加到链接的 S3 存储桶且 FSx 文件系统中当前不存在的文件的元数据。

- CHANGED-当数据存储库中的文件更改时，Amazon FSx 会自动更新文件元数据并使文件系统中的现有文件内容失效。
- DELETED-当相应文件在数据存储库中被删除时，Amazon 会 FSx 自动删除文件系统上的文件。

您可以为您的 AutoImportPolicy 定义事件类型的任意组合。

最高：3

更新策略：可以在更新期间更改此设置。

FsxOntapSettings

> **Note**
>
> 如果为 StorageType 指定了 FsxOntap，则必须定义 FsxOntapSettings。

（可选）适用 FSx 于 ONTAP 的文件系统的设置。

```
FsxOntapSettings:
  VolumeId: string
```

FsxOntapSettings 属性

VolumeId（必需，String）

指定 ONTAP FSx 系统的现有卷 ID。

> **Note**
>
> - 如果使用 AWS Batch 调度程序， FSx 则 ONTAP 仅在头节点上可用。
> - 如果 FSx 适用于 ONTAP 的部署类型为Multi-AZ，请确保已正确配置头节点子网的路由表。
> - 3.2.0 FSx AWS ParallelCluster 版本中增加了对 ONTAP 的支持。
> - 该文件系统必须关联到通过端口 111、635、2049 和 4046 允许入站和出站 TCP 和 UDP 流量的安全组。

通过执行以下操作之一，确保允许集群和文件系统之间的流量：

- 配置文件系统的安全组以允许进出集群子网的 CIDR 或前缀列表的流量。

> **Note**
> AWS ParallelCluster 验证端口是否已打开以及 CIDR 或前缀列表是否已配置。 AWS ParallelCluster 不验证 CIDR 块或前缀列表的内容。

- 通过使用 `SlurmQueues`/`Networking`/`SecurityGroups` 和 `HeadNode`/`Networking`/`SecurityGroups`，设置集群节点的自定义安全组。必须将自定义安全组配置为允许集群和文件系统之间的流量。

> **Note**
> 如果所有集群节点都使用自定义安全组，则 AWS ParallelCluster 仅验证端口是否已打开。 AWS ParallelCluster 无法验证源和目标的配置是否正确。

更新策略：如果更改此设置，则不允许更新。

`FsxOpenZfsSettings`

> **Note**
> 如果为 `StorageType` 指定了 `FsxOpenZfs`，则必须定义 `FsxOpenZfsSettings`。

（可选）适用 FSx 于 OpenZFS 文件系统的设置。

```
FsxOpenZfsSettings:
  VolumeId: string
```

更新策略：如果更改此设置，则不允许更新。

`FsxOpenZfsSettings` 属性

`VolumeId`（必需，`String`）

指定 OpenZFS FSx 系统的现有卷 ID。

> **Note**
>
> - 如果使用 AWS Batch 调度程序， FSx 则 OpenZFS 只能在头节点上使用。
> - 3.2.0 FSx AWS ParallelCluster 版本中增加了对 OpenZFS 的支持。
> - 该文件系统必须关联到通过端口 111、2049、20001、20002 和 20003 允许入站和出站 TCP 和 UDP 流量的安全组。

通过执行以下操作之一，确保允许集群和文件系统之间的流量：

- 配置文件系统的安全组以允许进出集群子网的 CIDR 或前缀列表的流量。

  > **Note**
  >
  > AWS ParallelCluster 验证端口是否已打开以及 CIDR 或前缀列表是否已配置。 AWS ParallelCluster 不验证 CIDR 块或前缀列表的内容。

- 通过使用 `SlurmQueues`/`Networking`/`SecurityGroups` 和 `HeadNode`/`Networking`/`SecurityGroups`，设置集群节点的自定义安全组。必须将自定义安全组配置为允许集群和文件系统之间的流量。

  > **Note**
  >
  > 如果所有集群节点都使用自定义安全组，则 AWS ParallelCluster 仅验证端口是否已打开。 AWS ParallelCluster 无法验证源和目标的配置是否正确。

更新策略：如果更改此设置，则不允许更新。

`FileCacheSettings`

> **Note**
>
> 如果为 `StorageType` 指定了 `FileCache`，则必须定义 `FileCacheSettings`。

（可选）文件缓存的设置。

```
FileCacheSettings:
  FileCacheId: string
```

更新策略：如果更改此设置，则不允许更新。

FileCacheSettings 属性

FileCacheId（必需，String）

指定现有文件缓存的文件缓存 ID。

> **Note**
>
> - 文件缓存不支持 AWS Batch 调度程序。
> - 3.7.0 AWS ParallelCluster 版本中增加了对文件缓存的支持。
> - 该文件系统必须关联到通过端口 988 允许入站和出站 TCP 流量的安全组。

通过执行以下操作之一，确保允许集群和文件系统之间的流量：

- 配置文件缓存的安全组以允许进出集群子网的 CIDR 或前缀列表的流量。

  > **Note**
  >
  > AWS ParallelCluster 验证端口是否已打开以及 CIDR 或前缀列表是否已配置。 AWS ParallelCluster 不验证 CIDR 块或前缀列表的内容。

- 通过使用 SlurmQueues/Networking/SecurityGroups 和 HeadNode/Networking/SecurityGroups，设置集群节点的自定义安全组。必须将自定义安全组配置为允许集群和文件系统之间的流量。

  > **Note**
  >
  > 如果所有集群节点都使用自定义安全组，则 AWS ParallelCluster 仅验证端口是否已打开。 AWS ParallelCluster 无法验证源和目标的配置是否正确。

更新策略：如果更改此设置，则不允许更新。

Iam 部分

（可选）指定集群的 IAM 属性。

```
Iam:
  Roles:
    LambdaFunctionsRole: string
  PermissionsBoundary: string
  ResourcePrefix: string
```

更新策略：可以在更新期间更改此设置。

Iam 属性

PermissionsBoundary（可选，String）

要用作 AWS ParallelCluster创建的所有角色的权限边界的 IAM 策略的 ARN。有关更多信息，请参阅《IAM 用户指南》中的 IAM 实体的权限边界。格式为 `arn:${Partition}:iam::${Account}:policy/${PolicyName}`。

更新策略：可以在更新期间更改此设置。

Roles（可选）

指定集群使用的 IAM 角色的设置。

更新策略：可以在更新期间更改此设置。

LambdaFunctionsRole（可选，String）

要用于的 IAM 角色的 ARN。 AWS Lambda这会覆盖所有 AWS CloudFormation 支持自定义资源的 Lambda 函数的默认角色。需要将 Lambda 配置可以担任该角色的主体。这不会覆盖用于的 Lambda 函数的角色。 AWS Batch格式为 `arn:${Partition}:iam::${Account}:role/${RoleName}`。

更新策略：可以在更新期间更改此设置。

ResourcePrefix（可选）

为由创建的 IAM 资源指定路径或名称前缀 AWS ParallelCluster。

资源前缀必须遵循 IAM 指定的命名规则：

- 名称最多可以包含 30 个字符。
- 名称只能是没有斜杠 (/) 字符的字符串。
- 路径最多可以有 512 个字符。
- 路径必须以斜杠 (/) 开头和结尾。它可以在起始和结尾斜杠 (/) 之间包含多个斜杠 (/)。
- 您可以组合路径和名称 `/path/name`。

指定名称。

```
Iam:
  ResourcePrefix: my-prefix
```

指定路径。

```
Iam:
  ResourcePrefix: /org/dept/team/project/user/
```

指定路径和名称。

```
Iam:
  ResourcePrefix: /org/dept/team/project/user/my-prefix
```

如果指定 `/my-prefix`，则会返回错误。

```
Iam:
  ResourcePrefix: /my-prefix
```

返回配置错误。路径必须有两个 `/`。前缀本身不能有 `/`。

更新策略：如果更改此设置，则不允许更新。

`LoginNodes` 部分

> **Note**
>
> 3.7.0 AWS ParallelCluster 版本中增加了对 `LoginNodes` Support 的支持。

（可选）指定登录节点池的配置。

```
LoginNodes:
  Pools:
    - Name: string
      Count: integer
      InstanceType: string
      GracetimePeriod: integer
      Image:
        CustomAmi: string
      Ssh:
        KeyName: string
        AllowedIps: string
      Networking:
        SubnetIds:
          - string
        SecurityGroups:
          - string
        AdditionalSecurityGroups:
          - string
      Dcv:
        Enabled: boolean
        Port: integer
        AllowedIps: string
      CustomActions:
        OnNodeStart:
          Sequence:
            - Script: string
              Args:
                - string
          Script: string
          Args:
            - string
        OnNodeConfigured:
          Sequence:
            - Script: string
              Args:
                - string
          Script: string
          Args:
            - string
        OnNodeUpdated:
          Sequence:
            - Script: string
```

```
            Args:
              - string
          Script: string
          Args:
            - string
      Iam:
        InstanceRole: string
        InstanceProfile: string
        AdditionalIamPolicies:
          - Policy: string
```

更新策略：必须停止集群中的登录节点，才能更改此设置进行更新。

LoginNodes 属性

Pools 属性

定义具有相同资源配置的登录节点组。从 AWS ParallelCluster 3.11.0 开始，最多可以指定 10 个池。

```
Pools:
  - Name: string
    Count: integer
    InstanceType: string
    GracetimePeriod: integer
    Image:
      CustomAmi: string
    Ssh:
      KeyName: string
      AllowedIps: string
    Networking:
      SubnetIds:
        - string
      SecurityGroups:
        - string
      AdditionalSecurityGroups:
        - string
    Dcv:
      Enabled: boolean
      Port: integer
      AllowedIps: string
    CustomActions:
      OnNodeStart:
        Sequence:
```

```
          - Script: string
            Args:
              - string
        Script: string
        Args:
          - string
      OnNodeConfigured:
        Sequence:
          - Script: string
            Args:
              - string
        Script: string
        Args:
          - string
      OnNodeUpdated:
        Sequence:
          - Script: string
            Args:
              - string
        Script: string
        Args:
          - string
    Iam:
      InstanceRole: string
      InstanceProfile: string
      AdditionalIamPolicies:
        - Policy: string
```

更新策略：可以添加登录节点池，但移除池需要停止集群中的所有登录节点。

Name（必需 String）

指定 LoginNodes 池的名称。此参数用于标记 LoginNodes 资源。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.11.0 开始，更新策略是：必须停止池中的登录节点才能更改此设置以进行更新。

`Count`（必需 `Integer`）

指定要保持活动状态的登录节点的数量。

更新策略：可以在更新期间更改此设置。

`InstanceType`（必需 `String`）

指定用于登录节点的 Amazon EC2 实例类型。该实例类型的架构必须与用于 Slurm `InstanceType` 设置的架构相同。

更新策略：如果登录节点池已停止，则可以更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.11.0 开始，更新策略是：必须停止池中的登录节点才能更改此设置以进行更新。

`GracetimePeriod`（可选 `Integer`）

指定从向已登录用户发出登录节点即将停用的通知到实际停止事件之间经过的最短时间（以分钟为单位）。`GracetimePeriod` 的有效值为 3 到 120 分钟。默认值为 10 分钟。

> **Note**
>
> 触发事件涉及多个 AWS 服务之间的交互。有时，网络延迟和信息的传播可能需要一些时间，因此由于 AWS 服务内部延迟，宽限期可能需要比预期更长的时间。

更新策略：可以在更新期间更改此设置。

`Image`（可选）

定义登录节点的映像配置。

```
Image:
  CustomAmi: String
```

`CustomAmi`（可选 `String`）

指定用于配置登录节点的自定义 AMI。如果未指定，则该值默认为 HeadNode 部分 中指定的值。

更新策略：如果更改此设置，则不允许更新。

Ssh（可选）

定义登录节点的 ssh 配置。

```
Ssh:
  KeyName: string
  AllowedIps: string
```

> **Note**
>
> 从 AWS ParallelCluster 版本 3.11.0 开始，更新策略是：必须停止池中的登录节点才能更改此设置以进行更新。

KeyName（可选 String）

指定用于登录到登录节点的 ssh 密钥。如果未指定，则该值默认为 HeadNode 部分 中指定的值。

更新策略：必须停止池中的登录节点，才能更改此设置进行更新。

AllowedIps（可选 String）

指定池中登录节点 SSH 连接的 CIDR 格式 IP 范围或前缀列表 ID。默认值为头节点配置中AllowedIps定义的，或者0.0.0.0/0如果未指定。 HeadNode 部分。

更新策略：必须停止池中的登录节点，才能更改此设置进行更新。

> **Note**
>
> 3.11.0 AWS ParallelCluster 版本中增加了 AllowedIps 对登录节点的支持。

Networking（必填）

```
Networking:
  SubnetIds:
    - string
  SecurityGroups:
```

```
    - string
  AdditionalSecurityGroups:
    - string
```

> **Note**
>
> 从 AWS ParallelCluster 版本 3.11.0 开始，更新策略是：必须停止池中的登录节点才能更改此设置以进行更新。

`SubnetIds`（必需 `[String]`）

您在其中配置登录节点池的现有子网的 ID。您只能定义一个子网。

更新策略：如果更改此设置，则不允许更新。

`SecurityGroups`（可选 `[String]`）

用于登录节点池的安全组的列表。如果未指定安全组，则会为您 AWS ParallelCluster 创建安全组。

更新策略：如果更改此设置，则不允许更新。

`AdditionalSecurityGroups`（可选 `[String]`）

用于登录节点池的其他安全组的列表。

更新策略：如果更改此设置，则不允许更新。

`Dcv`（可选）

定义在登录节点上运行的 NICE DCV 服务器的配置设置。有关更多信息，请参阅 通过 Amazon DCV 连接到头节点和登录节点。

```
Dcv:
  Enabled: boolean
  Port: integer
  AllowedIps: string
```

> **Important**
>
> 默认情况下，设置的 NICE DCV 端口对所有 IPv4 地址开放。 AWS ParallelCluster 只有当您拥有 NICE DCV 会话的 URL 并在 pcluster dcv-connect 返回 URL 后 30 秒内连接到

NICE DCV 会话，才能连接到 NICE DCV 端口。请使用 `AllowedIps` 设置进一步限制对具有 CIDR 格式 IP 范围的 NICE DCV 端口的访问，并使用 Port 设置来设置非标准端口。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 3.11.0 AWS ParallelCluster 版本中增加了对登录节点上的 DCV 的支持。

`Enabled`（必需 `Boolean`）

指定是否在池中的登录节点上启用 NICE DCV。默认值为 `false`。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> NICE DCV 会自动生成自签名证书，该证书用于保护 NICE DCV 客户端和登录节点上运行的 NICE DCV 服务器之间的流量。要配置您自己的证书，请参阅 Amazon DCV HTTPS 证书。

`Port`（可选 `Integer`）

指定 NICE DCV 的端口。默认值为 8443。

更新策略：如果更改此设置，则不允许更新。

`AllowedIps`（可选 `String`）

指定 NICE DCV 连接的 CIDR 格式的 IP 范围。此设置仅在 AWS ParallelCluster 创建安全组时使用。默认值是 `0.0.0.0/0`，允许从任何 Internet 地址访问。

更新策略：如果更改此设置，则不允许更新。

`CustomActions`（可选）

指定在登录节点上运行的自定义脚本。

```
CustomActions:
```

```
  OnNodeStart:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
  OnNodeConfigured:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
  OnNodeUpdated:
    Sequence:
      - Script: string
        Args:
          - string
    Script: string
    Args:
      - string
```

> **Note**
>
> 3.11.0 AWS ParallelCluster 版本中增加了对登录节点上的自定义操作的支持。

`OnNodeStart`（可选）

指定在启动任何节点部署引导操作前在登录节点上运行的单个脚本或脚本序列。有关更多信息，请参阅自定义引导操作。

`Sequence`（可选）

要运行的脚本列表。 AWS ParallelCluster 按配置文件中列出的顺序运行脚本，从第一个脚本开始。

`Script`（必需 `String`）

指定要使用的文件。文件路径可以 `https://` 或 `s3://` 开头。

Args（可选 [String]）

要传递到脚本的参数的列表。

更新策略：如果更改此设置，则不允许更新。

Script（必需 String）

指定用于单个脚本的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选 [String]）

要传递到单个脚本的参数的列表。

OnNodeConfigured（可选）

指定节点引导过程完成后，在登录节点上运行的单个脚本或一系列脚本。有关更多信息，请参阅自定义引导操作。

Sequence（可选）

要运行的脚本列表。 AWS ParallelCluster 按配置文件中列出的顺序运行脚本，从第一个脚本开始。

Script（必需 String）

指定要使用的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选 [String]）

要传递到脚本的参数的列表。

更新策略：如果更改此设置，则不允许更新。

Script（必需 String）

指定用于单个脚本的文件。文件路径可以 https:// 或 s3:// 开头。

Args（可选 [String]）

要传递到单个脚本的参数的列表。

更新策略：如果更改此设置，则不允许更新。

OnNodeUpdated（可选）

指定在头节点更新完成并且调度程序和共享存储与最新的群集配置更改保持一致后要运行的单个脚本或一系列脚本。有关更多信息，请参阅自定义引导操作。

`Sequence`（可选）

要运行的脚本列表。 AWS ParallelCluster 按配置文件中列出的顺序运行脚本，从第一个脚本开始。

`Script`（必需 `String`）

指定要使用的文件。文件路径可以 `https://` 或 `s3://` 开头。

`Args`（可选 `[String]`）

要传递到脚本的参数的列表。

`Script`（必需 `String`）

指定用于单个脚本的文件。文件路径可以 `https://` 或 `s3://` 开头。

`Args`（可选 `[String]`）

要传递到单个脚本的参数的列表。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> AWS ParallelCluster 不支持同时包含单个脚本和Sequence同一个自定义操作。

`Iam`（可选）

指定要在登录节点上使用的实例角色或实例配置文件，用于覆盖集群的默认实例角色或实例配置文件。

```
Iam:
  InstanceRole: string
  InstanceProfile: string
  AdditionalIamPolicies:
    - Policy: string
```

> **Note**
>
> 从 AWS ParallelCluster 版本 3.11.0 开始，更新策略是：必须停止池中的登录节点才能更改此设置以进行更新。

`InstanceProfile`（可选 `String`）

指定用于覆盖默认登录节点实例配置文件的实例配置文件。您不能同时指定 `InstanceProfile` 和 `InstanceRole`。格式为 `arn:Partition:iam::Account:instance-profile/`*`InstanceProfileName`*。如果指定此设置，则不能指定 `InstanceRole` 和 `AdditionalIamPolicies` 设置。

更新策略：如果更改此设置，则不允许更新。

`InstanceRole`（可选 `String`）

指定用于覆盖默认登录节点实例角色的实例角色。您不能同时指定 `InstanceProfile` 和 `InstanceRole`。格式为 `arn:Partition:iam::Account:role/RoleName`。如果指定此设置，则不能指定 `InstanceProfile` 和 `AdditionalIamPolicies` 设置。

更新策略：如果更改此设置，则不允许更新。

`AdditionalIamPolicies`（可选）

```
AdditionalIamPolicies:
  - Policy: string
```

IAM 策略的 Amazon 资源名称 (ARN)。

指定适用于亚马逊 IAM 策略的亚马逊资源名称 (ARNs) 列表 EC2。除了所需的权限外，此列表还附在用于登录节点的根角色上 AWS ParallelCluster。

IAM 策略名称及其 ARN 不相同。不能使用名称。

如果指定此设置，则不能指定 `InstanceProfile` 和 `InstanceRole` 设置。我们建议您使用，`AdditionalIamPoliciesAdditionalIamPolicies`因为已添加到所需的权限中，并且`InstanceRole`必须包含所有必需的权限。 AWS ParallelCluster 随着功能的不断添加，所需权限通常会随版本发生变化。

没有默认值。

更新策略：如果更改此设置，则不允许更新。

`Policy`（必需 `[String]`）

更新策略：如果更改此设置，则不允许更新。

Monitoring 部分

（可选）指定集群的监控设置。

```
Monitoring:
  Logs:
    CloudWatch:
      Enabled: boolean
      RetentionInDays: integer
      DeletionPolicy: string
    Rotation:
      Enabled: boolean
  Dashboards:
    CloudWatch:
      Enabled: boolean
  DetailedMonitoring: boolean
  Alarms:
   Enabled: boolean
```

更新策略：在更新期间不分析此设置。

Monitoring 属性

`Logs`（可选）

集群的日志设置。

更新策略：如果更改此设置，则不允许更新。

`CloudWatch`（可选）

集群的 CloudWatch 日志设置。

更新策略：如果更改此设置，则不允许更新。

`Enabled`（必需，`Boolean`）

如果是`true`，则集群日志将流式传输到 CloudWatch 日志。默认值为 `true`。

更新策略：如果更改此设置，则不允许更新。

`RetentionInDays`（可选，`Integer`）

在日志中保留日志事件的 CloudWatch 天数。默认值为 180。支持的值有 0、1、3、5、7、14、30、60、90、120、150、180、365、400、545、731、1827 和 3653。值为 0 将使用默认的 CloudWatch 日志保留设置，即永不过期。

更新策略：可以在更新期间更改此设置。

DeletionPolicy（可选，String）

表示删除集群时是否删除 CloudWatch 日志上的日志事件。可能的值为 Delete 和 Retain。默认值为 Retain。

更新策略：可以在更新期间更改此设置。

Rotation（可选）

集群的日志轮换设置。

更新策略：如果更改此设置，则不允许更新。

Enabled（必需，Boolean）

如果为 true，则启用日志轮换。默认值为 true。当 AWS ParallelCluster 配置的日志文件达到一定大小时，将对其进行轮换并保留单个备份。有关更多信息，请参阅AWS ParallelCluster 配置的日志轮换。

更新策略：如果更改此设置，则不允许更新。

Dashboards（可选）

集群的控制面板设置。

更新策略：可以在更新期间更改此设置。

CloudWatch（可选）

集群的 CloudWatch 仪表板设置。

更新策略：可以在更新期间更改此设置。

Enabled（必需，Boolean）

如果启用true，则 CloudWatch 仪表板已启用。默认值为 true。

更新策略：可以在更新期间更改此设置。

DetailedMonitoring（可选，Boolean）

如果设置为true，则会对计算队列 Amazon EC2 实例启用详细监控。启用后，Amazon EC2 控制台会显示每隔 1 分钟监控实例的图表。启用此功能后，会产生额外费用。默认值为 false。

有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的启用或关闭实例的详细监控。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

> **Note**
>
> `DetailedMonitoring`是从 3.6.0 AWS ParallelCluster 版本开始添加的。

`Alarms`（可选）

CloudWatch 集群警报。

更新策略：可以在更新期间更改此设置。

`Enabled`（可选）

如果是`true`，则将为集群创建 CloudWatch 警报。默认值为 `true`。

更新策略：可以在更新期间更改此设置。

> **Note**
>
> 从 AWS ParallelCluster 版本 3.8.0 开始，将为头节点创建以下警报：Amazon Healt EC2 h Check、CPU/Memory/Disk利用率和包含所有其他警报的复合警报。

`Tags` 部分

（可选），Array 定义所有群集资源使用 AWS CloudFormation 并传播到所有群集资源的标签。有关更多信息，请参阅 AWS CloudFormation 用户指南 中的 AWS CloudFormation 资源标签。

```
Tags:
  - Key: string
    Value: string
```

更新策略：如果更改此设置，则不允许更新。

`Tags` 属性

`Key`（必需，`String`）

定义标签的名称。

更新策略：如果更改此设置，则不允许更新。

Value（必需，String）

定义标签的值。

更新策略：如果更改此设置，则不允许更新。

AdditionalPackages 部分

（可选）用于标识要安装的其他程序包。

```
AdditionalPackages:
  IntelSoftware:
    IntelHpcPlatform: boolean
```

更新策略：如果更改此设置，则不允许更新。

IntelSoftware

（可选）定义 Intel Select Solutions 的配置。

```
IntelSoftware:
  IntelHpcPlatform: boolean
```

更新策略：如果更改此设置，则不允许更新。

IntelSoftware 属性

IntelHpcPlatform（可选，Boolean）

如果为true，则表示英特尔 Parallel Studio 的最终用户许可协议已被接受。这将导致 Intel Parallel Studio 安装在头节点上并与计算节点共享。这使头节点进行引导的时间增加了几分钟。

更新策略：如果更改此设置，则不允许更新。

DirectoryService 部分

> **Note**
>
> 在 3.1.1 AWS ParallelCluster 版本中添加DirectoryService了对 Support 的支持。

（可选）支持多用户访问的集群的目录服务设置。

AWS ParallelCluster 管理支持多用户通过系统安全服务守护程序 (SSSD) 支持的轻型目录访问协议 (LDAP) 使用活动目录 (AD) 访问集群的权限。有关更多信息，请参阅什么是 AWS Directory Service？在《AWS Directory Service 管理指南》中。

我们建议您使用 LDAP TLS/SSL（简称 LDAPS），以确保任何潜在的敏感信息都通过加密通道传输。

```
DirectoryService:
  DomainName: string
  DomainAddr: string
  PasswordSecretArn: string
  DomainReadOnlyUser: string
  LdapTlsCaCert: string
  LdapTlsReqCert: string
  LdapAccessFilter: string
  GenerateSshKeysForUsers: boolean
  AdditionalSssdConfigs: dict
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

`DirectoryService` 属性

> **Note**
>
> 如果您计划 AWS ParallelCluster 在无法访问 Internet 的单个子网中使用，AWS ParallelCluster 在无法访问互联网的单个子网中请参阅了解其他要求。

`DomainName`（必需，`String`）

用于指示身份信息的 Active Directory (AD) 域。

`DomainName` 同时接受完全限定域名 (FQDN) 和 LDAP 可分辨名称 (DN) 格式。

- FQDN 示例：`corp.example.com`
- LDAP DN 示例：`DC=corp,DC=example,DC=com`

此属性对应于名为 `ldap_search_base` 的 sssd-ldap 参数。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

`DomainAddr`（必需，`String`）

URI 或 URIs 指向用作 LDAP 服务器的 AD 域控制器。该 URI 对应于名为 `ldap_uri` 的 SSSD-LDAP 参数。该值可以是逗号分隔的字符串。 URIs要使用 LDAP，必须在每个 URI 的开头添加 `ldap://`。

示例值：

```
ldap://192.0.2.0,ldap://203.0.113.0          # LDAP
ldaps://192.0.2.0,ldaps://203.0.113.0        # LDAPS without support for certificate
 verification
ldaps://abcdef01234567890.corp.example.com  # LDAPS with support for certificate
 verification
192.0.2.0,203.0.113.0                        # AWS ParallelCluster uses LDAPS by
 default
```

如果您使用 LDAPS 进行证书验证，则 URIs 必须是主机名。

如果您使用 LDAPS 而不进行证书验证或 LDAP，则 URIs 可以是主机名或 IP 地址。

使用 LDAP over TLS/SSL (LDAPS) 可避免通过未加密的通道传输密码和其他敏感信息。如果 AWS ParallelCluster 找不到协议，它会在每个 URI 或主机名的开头添加 `ldaps://`。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

`PasswordSecretArn`（必需，`String`）

包含`DomainReadOnlyUser`纯文本密码的 AWS Secrets Manager 密钥的 Amazon 资源名称 (ARN)。密钥的内容对应于名为 `ldap_default_authtok` 的 SSSD-LDAP 参数。

> **Note**
>
> 使用 AWS Secrets Manager 控制台创建密钥时，请务必选择 “其他类型的密钥”，选择纯文本，并且仅在密钥中包含密码文本。
> 有关如何使用 AWS Secrets Manager 创建密钥的更多信息，请参阅创建 AWS Secrets Manager 密钥

LDAP 客户端使用密码向 AD 域进行身份验证`DomainReadOnlyUser`，就像请求身份信息一样。

如果用户拥有 `DescribeSecret` 的权限，则会验证 `PasswordSecretArn`。如果指定的密钥存在，则 `PasswordSecretArn` 有效。如果用户 IAM 策略不包括 `DescribeSecret`，则不验

证 PasswordSecretArn 并显示警告消息。有关更多信息，请参阅 基本 AWS ParallelCluster pcluster 用户策略。

当密钥的值发生变化时，不会 自动更新集群。要针对新密钥值更新集群，必须使用 the section called “pcluster update-compute-fleet” 命令停止计算实例集，然后从头节点内运行以下命令。

```
$ sudo /opt/parallelcluster/scripts/directory_service/
update_directory_service_password.sh
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

DomainReadOnlyUser（必需，String）

在对集群用户登录进行身份验证时，用于查询 AD 域以获取身份信息的身份。它对应于名为 ldap_default_bind_dn 的 SSSD-LDAP 参数。对此值使用您的 AD 身份信息。

以节点上特定 LDAP 客户端要求的格式指定身份：

- MicrosoftAD：

```
cn=ReadOnlyUser,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
```

- SimpleAD：

```
cn=ReadOnlyUser,cn=Users,dc=corp,dc=example,dc=com
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

LdapTlsCaCert（可选，String）

证书包的绝对路径，其中包含证书链中为域控制器颁发证书的每个证书颁发机构的证书。它对应于名为 ldap_tls_cacert 的 SSSD-LDAP 参数。

证书捆绑包是一个由 PEM 格式（在 Windows 中也称为 DER Base64 格式）的不同证书串联组成的文件。它用于验证充当 LDAP 服务器的 AD 域控制器的身份。

AWS ParallelCluster 不负责将证书初始放置到节点上。作为集群管理员，您可以在创建集群后在头节点中手动配置证书，也可以使用引导脚本。或者，您可以使用包含头节点上配置的证书的亚马逊机器映像 (AMI)。

Simple AD 不提供 LDAPS 支持。要了解如何将 Simple AD 目录与集成 AWS ParallelCluster，请参阅AWS 安全博客中的如何为 Simple AD 配置 LDAPS 端点。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

LdapTlsReqCert（可选，String）

指定在 TLS 会话中对服务器证书执行哪些检查。它对应于名为 ldap_tls_reqcert 的 SSSD-LDAP 参数。

有效值：never、allow、try、demand 和 hard。

即使发现证书有问题，never、allow 和 try 也允许继续连接。

在未发现证书问题的情况下，demand 和 hard 允许继续进行通信。

如果集群管理员使用的值不需要证书验证成功，则会向管理员返回一条警告消息。出于安全考虑，我们建议您不要禁用证书验证。

默认值为 hard。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

LdapAccessFilter（可选，String）

指定用于将目录访问权限限制为一部分用户的筛选器。此属性对应于名为 ldap_access_filter 的 SSSD-LDAP 参数。您可以使用此属性将查询限制为支持大量用户的 AD。

此筛选器可阻止用户访问集群。但它不影响被阻止用户的可发现性。

如果设置了此属性，则 SSSD 参数 access_provider 将由 AWS ParallelCluster 在内部设置为 ldap 且不得被 DirectoryService/AdditionalSssdConfigs 设置修改。

如果省略此属性且未在 DirectoryService/AdditionalSssdConfigs 中指定自定义用户访问权限，则目录中的所有用户都可以访问集群。

示例：

```
"!(cn=SomeUser*)"  # denies access to every user with an alias that starts with
 "SomeUser"
"(cn=SomeUser*)"   # allows access to every user with alias that starts with
 "SomeUser"
"memberOf=cn=TeamOne,ou=Users,ou=CORP,dc=corp,dc=example,dc=com" # allows access
 only to users in group "TeamOne".
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

`GenerateSshKeysForUsers`（可选，Boolean）

定义集群用户在头节点上进行初始身份验证后是否立即为其 AWS ParallelCluster 生成 SSH 密钥。

如果设置为 `true`，则每个用户在头节点上进行首次身份验证后会为其生成一个 SSH 密钥并保存到 *USER_HOME_DIRECTORY*`/.ssh/id_rsa`（如果不存在）。

对于尚未在头节点上进行身份验证的用户，可能会在以下情况下进行首次身份验证：

- 用户使用自己的密码首次登录头节点。
- 在头节点中，sudoer 首次切换到用户：`su` *USERNAME*
- 在头节点中，sudoer 首次以用户身份运行命令：`su -u` *USERNAME COMMAND*

之后，用户可以使用 SSH 密钥登录到集群头节点和计算节点。使用 AWS ParallelCluster时，集群计算节点的密码登录在设计上是禁用的。如果用户未登录到头节点，则不会生成 SSH 密钥，用户将无法登录到计算节点。

默认值为 `true`。

更新策略：必须停止计算实例集才能更改此设置以进行更新。

`AdditionalSssdConfigs`（可选，Dict）

键值对的字典，其中包含要写入集群实例上的 SSSD 配置文件的 SSSD 参数和值。有关 SSSD 配置文件的完整描述，请参阅 SSSD 的 on-instance man 页面和相关配置文件。

SSSD 参数和值必须与下表中所述 AWS ParallelCluster的 SSSD 配置兼容。

- `id_provider`由`ldap`内部设置为 AWS ParallelCluster ，不得修改。
- `access_provider`指定 `DirectoryService`/ AWS ParallelCluster 时在`ldap`内部设置`LdapAccessFilter`为，且不得修改此设置。

  如果省略 `DirectoryService`/`LdapAccessFilter`，则也将省略其指定的 `access_provider`。例如，如果您在 `AdditionalSssdConfigs` 中将 `access_provider` 设置为 `simple`，则不得指定 `DirectoryService`/`LdapAccessFilter`。

以下配置代码段是 `AdditionalSssdConfigs` 有效配置的示例。

此示例启用 SSSD 日志的调试级别，将搜索库限制为特定的组织部门，并禁用凭证缓存。

```
DirectoryService:
  ...
```

```
  AdditionalSssdConfigs:
    debug_level: "0xFFF0"
    ldap_search_base: OU=Users,OU=CORP,DC=corp,DC=example,DC=com
    cache_credentials: False
```

此示例指定 SSSD simple access_provider 的配置。为 EngineeringTeam 中的用户提供了目录访问权限。在这种情况下，不得设置 DirectoryService/LdapAccessFilter。

```
DirectoryService:
  ...
  AdditionalSssdConfigs:
    access_provider: simple
    simple_allow_groups: EngineeringTeam
```

更新策略：必须停止计算实例集才能更改此设置以进行更新。

DeploymentSettings 部分

> **Note**
>
> DeploymentSettings是从 3.4.0 AWS ParallelCluster 版本开始添加的。

（可选）指定部署设置配置。

```
DeploymentSettings:
  LambdaFunctionsVpcConfig:
    SecurityGroupIds:
      - string
    SubnetIds:
      - string
  DisableSudoAccessForDefaultUser: Boolean
  DefaultUserHome: string # 'Shared' or 'Local'
```

DeploymentSettings 属性

LambdaFunctionsVpcConfig

（可选）指定 AWS Lambda 函数 VPC 配置。有关更多信息，请参阅 AWS Lambda 中的 VPC 配置 AWS ParallelCluster。

```
LambdaFunctionsVpcConfig:
  SecurityGroupIds:
    - string
  SubnetIds:
    - string
```

LambdaFunctionsVpcConfig properties

SecurityGroupIds（必需，[String]）

附加到 Lambda 函数 IDs 的 Amazon VPC 安全组列表。

更新策略：如果更改此设置，则不允许更新。

SubnetIds（必需，[String]）

附加 IDs 到 Lambda 函数的子网列表。

更新策略：如果更改此设置，则不允许更新。

> **Note**
> 子网和安全组必须位于同一 VPC。

DisableSudoAccessForDefaultUser 财产

> **Note**
> 只有 Slurm 集群支持此配置选项。

（可选）如果是 True，则将禁用默认用户的 sudo 权限。这适用于集群中的所有节点。

```
# Main DeploymentSettings section in config yaml(applies to HN, CF and LN)
DeploymentSettings:
  DisableSudoAccessForDefaultUser: True
```

要更新 DisableSudoAccessForDefaultUser 的值，必须停止计算实例集和所有登录节点。

更新策略：必须停止计算实例集和登录节点才能更改此设置以进行更新。

DefaultUserHome 属性

如果设置为 `Shared`，则集群将使用默认设置，并通过 `/home/<default user>` 在集群中共享默认用户的目录。

如果设置为 `Local`，则头节点、登录节点和计算节点都将有一个单独的本地默认用户目录，该目录存储在 `local/home/<default user>` 中。

构建映像配置文件

AWS ParallelCluster 版本 3 使用 YAML 1.1 文件作为构建镜像配置参数。请确认缩进正确以减少配置错误。有关更多信息，请参阅 YAML 1.1 规范 (https://yaml.org/spec/1.1/)。

这些配置文件用于定义如何使用 EC2 Image Builder 构建您的自定义 AWS ParallelCluster AMIs 文件。使用 `pcluster build-image` 命令即可触发自定义 AMI 构建过程。有关一些示例配置文件，请参阅 aw parallelcluster/tree/release-3.0/cli/tests/pcluster/schemas/test_imagebuilder_schema/tests-_imagebuild https://github.com/aws/ er_schema。

主题

- 构建映像配置文件属性
- Build 部分
- Image 部分
- DeploymentSettings 部分

构建映像配置文件属性

`Region`（可选，`String`）

AWS 区域 为`build-image`操作指定。例如 `us-east-2`。

`CustomS3Bucket`（可选，`String`）

指定在您的 AWS 账户中创建的 Amazon S3 存储桶的名称，该存储桶用于存储自定义 AMI 构建过程使用的资源和导出日志。图像使用的信息位于镜像配置的自定义存储桶中。 AWS ParallelCluster 在您创建集群的每个 AWS 区域中维护一个 Amazon S3 存储桶。默认情况下，这些 Amazon S3 存储桶命名为 `parallelcluster-hash-v1-DO-NOT-DELETE`。

Build 部分

（必需）指定用于构建映像的配置。

```
Build:
  Imds:
    ImdsSupport: string
  InstanceType: string
  SubnetId: string
  ParentImage: string
  Iam:
    InstanceRole: string
    InstanceProfile: string
    CleanupLambdaRole: string
    AdditionalIamPolicies:
      - Policy: string
    PermissionsBoundary: string
  Components:
    - Type: string
      Value: string
  Tags:
    - Key: string
      Value: string
  SecurityGroupIds:
    - string
  UpdateOsPackages:
    Enabled: boolean
  Installation:
    NvidiaSoftware:
      Enabled: boolean
    LustreClient:
      Enabled: boolean
```

Build 属性

`InstanceType`（必需，String）

指定用于构建映像的实例的实例类型。

`SubnetId`（可选，String）

指定要在其中预置用于构建映像的实例的现有子网的 ID。提供的子网需要互联网访问权限。请注意，如果构建失败，您可能需要修改子网的 IP 地址属性。

> **Warning**
>
> `pcluster build-image` 使用默认 VPC。如果默认 VPC 已被删除（可能使用 AWS Control Tower 或 AWS 着陆区域），则必须指定子网 ID。

指定时 SubnetId，建议同时指定该 SecurityGroupIds 属性。如果省略 SecurityGroupIds ， AWS ParallelCluster 将使用默认安全组或依赖指定子网内的默认行为。当你同时使用两者时，你会获得以下优势：

- 精细控制：当您明确定义两者时，可以确保在映像构建过程中启动的实例放置在正确的子网中，并且具有构建组件和任何必需服务所需的精确网络访问权限（例如访问构建脚本的 S3）。
- 安全最佳实践：当您定义适当的安全组时，这有助于将网络访问限制为仅限必要的端口和服务，从而增强构建环境的安全性。
- 避免潜在问题：如果您仅依赖默认设置，则可能会导致安全组过于开放或过于严格，从而在构建过程中导致问题。

`ParentImage`（必需，`String`）

指定基础映像。父映像可以是非 AWS ParallelCluster AMI，也可以是同一版本的官方 AWS ParallelCluster AMI。您不能使用不同版本的 AWS ParallelCluster 官方或自定义 AMI AWS ParallelCluster。格式必须是映像的 ARN `arn:Partition:imagebuilder:Region:Account:image/ImageName/ImageVersion` 或 AMI ID `ami-12345678`。

`SecurityGroupIds`（可选，`[String]`）

指定镜像的安全 IDs 组列表。

Imds

Imds 属性

（可选）指定 Amazon EC2 ImageBuilder 构建和测试实例元数据服务 (IMDS) 设置。

```
Imds:
  ImdsSupport: string
```

`ImdsSupport`（可选，`String`）

指定 Amazon EC2 ImageBuilder 构建和测试实例支持哪些 IMDS 版本。支持的值为 `v2.0` 和 `v1.0`。默认值为 `v2.0`。

如果设置`ImdsSupport`为`v1.0`， IMDSv2 则同时支持 IMDSv1 和。

如果设置`ImdsSupport`为`v2.0`，则 IMDSv2 仅支持。

有关更多信息，请参阅《适用于 Linux 实例的 Amazon EC2 用户指南》 IMDSv2中的 “使用”。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 从 3.7.0 AWS ParallelCluster 版开始，`ImdsSupport`默认值为。`v2.0`我们建议您在自定义操作调 IMDSv1 用 IMDSv2 中设置`ImdsSupport`为`v2.0`并替换为。
> 3.3.0 AWS ParallelCluster 版本`ImdsSupport`中增加了对 `Imds`/的支持。

Iam

Iam 属性

（可选）指定映像构建的 IAM 资源。

```
Iam:
  InstanceRole: string
  InstanceProfile: string
  CleanupLambdaRole: string
  AdditionalIamPolicies:
    - Policy: string
  PermissionsBoundary: string
```

`InstanceProfile`（可选，`String`）

指定实例配置文件以覆盖 EC2 Image Builder 实例的默认实例配置文件。`InstanceProfile`和`InstanceRole`和`AdditionalIamPolicies`不能一起指定。格式为 `arn:`*`Partition`*`:iam::`*`Account`*`:instance-profile/`*`InstanceProfileName`*。

`InstanceRole`（可选，`String`）

指定一个实例角色来覆盖 EC2 Image Builder 实例的默认实例角色。`InstanceProfile`和`InstanceRole`和`AdditionalIamPolicies`不能一起指定。格式为 `arn:Partition:iam::Account:role/RoleName`。

`CleanupLambdaRole`（可选，`String`）

用于支持 AWS CloudFormation 自定义资源的 AWS Lambda 函数的 IAM 角色的 ARN，该资源将在构建完成后移除构建工件。需要将 Lambda 配置可以担任该角色的主体。格式为 `arn:Partition:iam::Account:role/RoleName`。

`AdditionalIamPolicies`（可选）

指定要附加到用于生成自定义 AMI 的 I EC2 mage Builder 实例的其他 IAM 策略。

```
AdditionalIamPolicies:
  - Policy: string
```

`Policy`（可选，`[String]`）

IAM 策略的列表。格式为 `arn:Partition:iam::Account:policy/PolicyName`。

`PermissionsBoundary`（可选，`String`）

要用作 AWS ParallelCluster创建的所有角色的权限边界的 IAM 策略的 ARN。有关 IAM 权限边界的更多信息，请参阅 IAM 用户指南 中的 IAM 实体的权限边界。格式为 `arn:Partition:iam::Account:policy/PolicyName`。

Components

Components 属性

（可选）除了默认提供的 EC2 ImageBuilder 组件外，指定在 AMI 构建过程中要使用的 Amazon 组件 AWS ParallelCluster。此类组件可用于自定义 AMI 构建过程。有关更多信息，请参阅 AWS ParallelCluster AMI 自定义。

```
Components:
  - Type: string
    Value: string
```

`Type`（可选，`String`）

指定组件的类型-值对的类型。类型可以是 `arn` 或 `script`。

`Value`（可选，`String`）

指定组件的类型-值对的值。当类型为时`arn`，这是 I EC2 mage Builder 组件的 ARN。当 type 为时`script`，这就是 https or s3 link that points to the script to use when you create the EC2 Image Builder com重点。

Tags

Tags 属性

（可选）指定要在用于构建 AMI 的资源中设置的标签列表。

```
Tags:
  - Key: string
    Value: string
```

`Key`（可选，`String`）

定义标签的名称。

`Value`（可选，`String`）

定义标签的值。

UpdateOsPackages

UpdateOsPackages 属性

（可选）指定在安装 AWS ParallelCluster 软件堆栈之前是否更新操作系统。

```
UpdateOsPackages:
  Enabled: boolean
```

`Enabled`（可选，`Boolean`）

如果是`true`，则在安装软件之前更新并重新启动操作系统。 AWS ParallelCluster 默认值为`false`。

> **Note**
>
> 启用 `UpdateOsPackages` 后，将会更新所有可用的操作系统程序包，包括内核。作为客户，您有责任验证更新是否与更新中未包含的 AMI 依赖项兼容。
> 例如，假设您要为内核版本 Y.0 和某个组件 AWS ParallelCluster 版本 Z.0 附带的 X.0 版本构建一个 AMI。假设可用更新包括更新的内核版本 Y.1，但没有组件 Z.0 的更新。在启用 `UpdateOsPackages` 之前，您应负责验证组件 Z.0 是否支持内核 Y.1。

`Installation`

`Installation` 属性

（可选）指定要在映像上安装的其他软件。

```
Installation:
  NvidiaSoftware:
    Enabled: boolean
  LustreClient:
    Enabled: boolean
```

`NvidiaSoftware`属性（可选）

指定要安装的 Nvidia 软件。

```
NvidiaSoftware:
    Enabled: boolean
```

`Enabled`（可选，`boolean`）

如果是`true`，则将安装 Nvidia GPU 驱动程序和 CUDA。默认值为 `false`。

`LustreClient`属性（可选）

指定将安装 Amazon FSx Lustre 客户端。

```
LustreClient:
    Enabled: boolean
```

`Enabled`（可选，`boolean`）

如果是`true`，则将安装 Lustre 客户端。默认值为 `true`。

Image 部分

（可选）定义映像构建的映像属性。

```
Image:
  Name: string
  RootVolume:
    Size: integer
    Encrypted: boolean
    KmsKeyId: string
  Tags:
    - Key: string
      Value: string
```

Image 属性

Name（可选，String）

指定 AMI 的名称。如果未指定，则使用调用 pcluster build-image 命令时使用的名称。

Tags

Tags 属性

（可选）指定映像的键值对。

```
Tags:
  - Key: string
    Value: string
```

Key（可选，String）

定义标签的名称。

Value（可选，String）

定义标签的值。

RootVolume

RootVolume 属性

（可选）指定映像根卷的属性。

```
RootVolume:
  Size: integer
  Encrypted: boolean
  KmsKeyId: string
```

Size（可选，Integer）

指定映像根卷的大小，以 GiB 为单位。默认大小为 ParentImage 的大小加 27 GiB。

Encrypted（可选，Boolean）

指定是否对卷进行加密。默认值为 false。

KmsKeyId（可选，String）

指定用于加密卷的 AWS KMS 密钥的 ARN。格式为 arn:*Partition*:kms:*Region*:*Account*:key/*KeyId*。

DeploymentSettings 部分

> **Note**
>
> DeploymentSettings是从 3.4.0 AWS ParallelCluster 版本开始添加的。

（可选）指定部署设置配置。

```
DeploymentSettings:
  LambdaFunctionsVpcConfig:
    SecurityGroupIds:
      - string
    SubnetIds:
      - string
```

DeploymentSettings 属性

LambdaFunctionsVpcConfig

（可选）指定 AWS Lambda 函数 VPC 配置。有关更多信息，请参阅 AWS Lambda 中的 VPC 配置 AWS ParallelCluster。

```
LambdaFunctionsVpcConfig:
  SecurityGroupIds:
    - string
  SubnetIds:
    - string
```

LambdaFunctionsVpcConfig properties

`SecurityGroupIds`（必需，[String]）

附加到 Lambda 函数 IDs 的 Amazon VPC 安全组列表。

更新策略：如果更改此设置，则不允许更新。

`SubnetIds`（必需，[String]）

附加 IDs 到 Lambda 函数的子网列表。

更新策略：如果更改此设置，则不允许更新。

> **Note**
>
> 子网和安全组必须位于同一 VPC。

AWS ParallelCluster API 参考

本节提供每个 AWS ParallelCluster API 操作的描述、语法和用法示例。

主题

- buildImage
- createCluster
- deleteCluster

- deleteClusterInstances
- deleteImage
- describeCluster
- describeClusterInstances
- describeComputeFleet
- describeImage
- getClusterLog活动
- getClusterStack活动
- getImageLog活动
- getImageStack活动
- listClusters
- listClusterLog直播
- listImageLog直播
- listImages
- listOfficialImages
- updateCluster
- updateComputeFleet

buildImage

在中创建自定义 AWS ParallelCluster 镜像 AWS 区域。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
POST /v3/images/custom
```

```
{
  "imageConfiguration": "string",
  "imageId": "string",
  "dryrun": boolean,
  "region": "string",
  "rollbackOnFailure": boolean,
  "supressValidators": [ "string" ],
  "validationFailureLevel": "string"
}
```

请求正文

imageConfiguration

映像配置作为 YAML 文档。

类型：字符串。

必需：是

imageId

要构建的映像的 ID。

类型：字符串。

必需：是

dryrun

如果设置为 `true`，则仅执行请求验证而不创建任何资源。使用此参数可验证映像配置。默认值为 `false`。

类型：布尔值

必需：否

region

你 AWS 区域 在其中运行命令来构建镜像。

类型：字符串

必需：否

rollbackOnFailure

如果设置为 `true`，则在创建映像失败时会回滚映像堆栈。默认值为 `false`。

类型：布尔值

必需：否

suppressValidators

标识一个或多个要禁止的配置验证器。

类型：字符串列表

格式：`(ALL|type:[A-Za-z0-9]+)`

必需：否

validationFailureLevel

导致映像构建失败的最低验证级别。默认值为 `ERROR`。

类型：字符串

有效值：`INFO | WARNING | ERROR`

必需：否

响应语法

```
{
  "image": {
    "imageId": "string",
    "ec2AmiInfo": {
      "amiId": "string"
    },
    "region": "string",
    "version": "string",
    "cloudformationStackArn": "string",
    "imageBuildStatus": "BUILD_IN_PROGRESS",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS"
  },
  "validationMessages": [
```

```
    {
      "id": "string",
      "type": "string",
      "level": "INFO",
      "message": "string"
    }
  ]
}
```

响应正文

image

imageId

映像的 ID。

类型：字符串

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE`

ec2 AmiInfo

ami_id

亚马逊 EC2 AMI ID。

类型：字符串

imageBuildStatus

映像构建状态。

类型：字符串

有效值：`BUILD_IN_PROGRESS | BUILD_FAILED | BUILD_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE`

区域

AWS 区域 在其中构建镜像。

类型：字符串

version

用于构建镜像的 AWS ParallelCluster 版本。

类型：字符串

validationMessages

验证级别低于 `validationFailureLevel` 的消息的列表。消息列表是在配置验证期间收集的。

id

验证器 ID。

类型：字符串

level

验证级别。

类型：字符串

有效值：`INFO | WARNING | ERROR`

message

验证消息。

类型：字符串

type

验证器的类型。

类型：字符串

示例

Python

请求

```
$ build_image(custom-image-id, custom-image-config.yaml)
```

200 响应

```
{
  "image": {
    "cloudformation_stack_arn": "arn:aws:cloudformation:us-
east-1:123456789012:stack/custom-image-id/711b76b0-af81-11ec-a29f-0ee549109f1f",
    "cloudformation_stack_status": "CREATE_IN_PROGRESS",
    "image_build_status": "BUILD_IN_PROGRESS",
    "image_id": "custom-image-id",
    "region": "us-east-1",
    "version": "3.2.1"
  }
}
```

createCluster

在 AWS 区域中创建托管集群。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
POST /v3/clusters
{
  "clusterName": "string",
  "clusterConfiguration": "string",
  "dryrun": boolean,
  "region": "string",
  "rollbackOnFailure", boolean,
  "suppressValidators": [ "string" ],
  "validationFailureLevel": "string"
}
```

请求正文

clusterConfiguration

集群配置作为 YAML 文档。

类型：字符串。

必需：是

clusterName

要创建的集群的名称。

名称必须以字母字符开头。名称最多可以包含 60 个字符。如果 Slurm 启用记账功能后，名称最多可包含 40 个字符。

类型：字符串。

必需：是

dryrun

如果设置为`true`，则仅执行请求验证，而不创建任何资源。使用此参数可验证集群配置。默认值为`false`。

类型：布尔值

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

rollbackOnFailure

如果设置为`true`，则在集群创建失败时会进行集群堆栈回滚。默认值为 `true`。

类型：布尔值

必需：否

suppressValidators

标识一个或多个要禁止的配置验证器。

类型：字符串列表

格式：`(ALL|type:[A-Za-z0-9]+)`

必需：否

validationFailureLevel

导致集群创建失败的最低验证级别。默认值为 `ERROR`。

类型：字符串

有效值：`INFO | WARNING | ERROR`

必需：否

响应语法

```
{
  "cluster": {
    "clusterName": "string",
    "region": "string",
    "version": "string",
```

```
    "cloudformationStackArn": "string",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "clusterStatus": "CREATE_IN_PROGRESS",
    "scheduler": {
      "type": "string",
      "metadata": {
        "name": "string",
        "version": "string"
      }
    }
  },
  "validationMessages": [
    {
      "id": "string",
      "type": "string",
      "level": "INFO",
      "message": "string"
    }
  ]
}
```

响应正文

clusterName

集群的名称。

类型：字符串

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS |

UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE

clusterStatus

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED

区域

创建 AWS 区域 集群时使用的。

类型：字符串

scheduler

metadata

调度器元数据

名称

调度器的名称。

类型：字符串

version

调度器版本。

类型：字符串

type

调度器类型。

类型：字符串

version

用于创建集群的 AWS ParallelCluster 版本。

类型：字符串

validation_messages

验证级别低于 `validationFailureLevel` 的消息的列表。消息列表是在配置验证期间收集的。

id

验证器的 ID。

类型：字符串

level

类型：字符串

有效值：`INFO | WARNING | ERROR`

message

验证消息。

类型：字符串

type

验证器的类型。

类型：字符串

示例

Python

请求

```
$ create_cluster(cluster_name_3x, cluster-config.yaml)
```

200 响应

```
{
  "cluster": {
    "cloudformation_stack_arn": "arn:aws:cloudformation:us-
east-1:123456789012:stack/cluster-3x/e0462730-50b5-11ed-99a3-0a5ddc4a34c7",
    "cloudformation_stack_status": "CREATE_IN_PROGRESS",
```

```
    "cluster_name": "cluster-3x",
    "cluster_status": "CREATE_IN_PROGRESS",
    "region": "us-east-1",
    "scheduler": {
      "type": "slurm"
    },
    "version": "3.2.1"
  }
}
```

deleteCluster

开始删除集群。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
DELETE /v3/clusters/{clusterName}
{
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

区域

AWS 区域 在其中删除集群。

类型：字符串

必需：否

响应语法

```
{
   "cluster": {
       "clusterName": "string",
       "region": "string",
       "version": "string",
       "cloudformationStackArn": "string",
       "cloudformationStackStatus": "DELETE_IN_PROGRESS",
       "clusterStatus": "DELETE_IN_PROGRESS",
       "scheduler": {
           "type": "string",
           "metadata": {
               "name": "string",
               "version": "string"
           }
        }
    }
}
```

响应正文

cluster

集群实例列表。

clusterName

集群的名称。

类型：字符串

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE`

clusterStatus

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED`

区域

AWS 区域 在其中创建集群。

类型：字符串

scheduler

metadata

调度器元数据。

名称

调度器的名称。

类型：字符串

version

调度器版本。

类型：字符串

type

调度器类型。

类型：字符串

version

用于创建集群的 AWS ParallelCluster 版本。

类型：字符串

示例

Python

请求

```
$ delete_cluster(cluster_name_3x)
```

200 响应

```
{
  "cluster": {
    "cloudformation_stack_arn": "arn:aws:cloudformation:us-
east-1:123456789012:stack/cluster_name_3x/16b49540-aee5-11ec-8e18-0ac1d712b241",
    "cloudformation_stack_status": "DELETE_IN_PROGRESS",
    "cluster_name": "cluster_name_3x",
    "cluster_status": "DELETE_IN_PROGRESS",
    "region": "us-east-1",
    "version": "3.2.1"
  }
}
```

deleteClusterInstances

开始强制终止所有集群计算节点。此操作不支持 AWS Batch 集群。

主题

- 请求语法

- 请求正文
- 响应正文
- 示例

请求语法

```
DELETE /v3/clusters/{clusterName}/instances
{
  "force": boolean,
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

force

如果设置为 `true`，则在找不到具有给定名称的集群时强制删除。默认值为 `false`。

类型：布尔值

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

响应正文

无

示例

Python

请求

```
$ delete_cluster_instances(cluster_name_3x)
```

200 响应

无

deleteImage

开始删除自定义 AWS ParallelCluster 镜像。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
DELETE /v3/images/custom/{imageId}
{
  "force": boolean,
  "region": "string"
}
```

请求正文

imageId

映像的 ID。

类型：字符串。

必需：是

force

如果设置为 `true`，则强制删除 AMI。如果有使用 AMI 的实例或者共享了 AMI，则使用此参数。默认值为 `false`。

类型：布尔值

必需：否

region

图像 AWS 区域 是在其中创建的。

类型：字符串

必需：否

响应语法

```
{
  "image": {
    "imageId": "string",
    "ec2AmiInfo": {
      "amiId": "string"
    },
    "region": "string",
    "version": "string",
    "cloudformationStackArn": "string",
    "imageBuildStatus": "DELETE_IN_PROGRESS",
    "cloudformationStackStatus": "DELETE_IN_PROGRESS"
  }
}
```

响应正文

image

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE`

ec2 AmiInfo

amiId

亚马逊 EC2 AMI ID。

类型：字符串

imageBuildStatus

映像构建状态。

类型：字符串

有效值：`BUILD_IN_PROGRESS | BUILD_FAILED | BUILD_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE`

imageId

映像的 ID。

类型：字符串

区域

AWS 区域 在其中创建镜像。

类型：字符串

version

用于构建镜像的 AWS ParallelCluster 版本。

类型：字符串

示例

Python

请求

```
$ delete_image(custom-image-id)
```

200 响应

```
{
  "image": {
    "image_build_status": "DELETE_IN_PROGRESS",
    "image_id": "custom-image-id",
    "region": "us-east-1",
    "version": "3.2.1"
  }
}
```

describeCluster

获取有关现有集群的详细信息。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters/{clusterName}
{
```

```
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

区域

AWS 区域 集群所在的。

类型：字符串

必需：否

响应语法

> **Note**
>
> `failureReason`已更改为`failures`从 3.5.0 AWS ParallelCluster 版本开始。

```
{
  "clusterName": "string",
  "region": "string",
  "version": "string",
  "cloudFormationStackStatus": "CREATE_IN_PROGRESS",
  "clusterStatus": "CREATE_IN_PROGRESS",
  "scheduler": {
    "type": "string",
    "metadata": {
      "name": "string",
      "version": "string"
    }
  },
```

```
  "cloudformationStackArn": "string",
  "creationTime": "2019-08-24T14:15:22Z",
  "lastUpdatedTime": "2019-08-24T14:15:22Z",
  "clusterConfiguration": {
    "url": "string"
  },
  "computeFleetStatus": "START_REQUESTED",
  "tags": [
    {
      "key": "string",
      "value": "string"
    }
  ],
  "headNode": {
    "instanceId": "string",
    "instanceType": "string",
    "launchTime": "2019-08-24T14:15:22Z",
    "privateIpAddress": "string",
    "publicIpAddress": "string",
    "state": "pending"
  },
  "failures": [
    {
      "failureCode": "string",
      "failureReason": "string"
    }
  ],
  "loginNodes": {
    "status": "string",
    "address": "string",
    "poolName": "string",
    "scheme": "string",
    "healthyNodes": integer,
    "unhealthyNodes": integer
  }
}
```

响应正文

clusterName

集群的名称。

类型：字符串

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE`

clusterConfiguration

url

集群配置文件的 URL。

类型：字符串

clusterStatus

集群状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED`

computeFleetStatus

计算实例集状态。

类型：字符串

有效值：`START_REQUESTED | STARTING | RUNNING | PROTECTED | STOP_REQUESTED | STOPPING | STOPPED | UNKNOWN | ENABLED | DISABLED`

creationTime

创建集群时的时间戳。

类型：日期时间

lastUpdatedTime

上次更新集群时的时间戳。

类型：日期时间

区域

AWS 区域 在其中创建集群。

类型：字符串

tags

与集群关联的标签的列表。

key

标签名称。

类型：字符串

tag

标签值。

类型：字符串

version

用于创建集群的 AWS ParallelCluster 版本。

类型：字符串

failures

集群堆栈处于 `CREATE_FAILED` 状态时的故障列表。

failureCode

集群堆栈处于 `CREATE_FAILED` 状态时的故障代码。

类型：字符串

failureReason

集群堆栈处于 `CREATE_FAILED` 状态时的故障原因。

类型：字符串

head_node

集群头节点。

instanceId

亚马逊 EC2 实例 ID。

类型：字符串

instanceType

Amazon EC2 实例类型。

类型：字符串

launchTime

Amazon EC2 实例的启动时间。

类型：日期时间

privateIpAddress

集群私有 IP 地址。

类型：字符串

publicIpAddress

集群公共 IP 地址。

类型：字符串

状态

头节点实例状态。

类型：字符串

有效值：`pending | running | shutting-down | terminated | stopping | stopped`

scheduler

metadata

调度器元数据。

名称

调度器的名称。

类型：字符串

version

调度器版本。

类型：字符串

loginNodes

status

登录节点状态。

类型：字符串

有效值：`PENDING | FAILED | ACTIVE`

address

登录节点地址。

类型：字符串

poolName

登录节点池名称。

类型：字符串

scheme

登录节点方案。

类型：字符串

healthyNodes

健康节点的数量。

类型：整数

unhealthyNodes

不正常节点数。

类型：整数

type

调度器类型。

类型：字符串

示例

Python

请求

```
$ describe_cluster(cluster_name_3x)
```

200 响应

```
{
  "cloud_formation_stack_status": "CREATE_COMPLETE",
  "cloudformation_stack_arn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster_name_3x/16b49540-aee5-11ec-8e18-0ac1d712b241",
  "cluster_configuration": {
    "url": "https://parallelcluster-...."
  },
  "cluster_name": "cluster_name_3x",
  "cluster_status": "CREATE_COMPLETE",
  "compute_fleet_status": "RUNNING",
  "creation_time": datetime.datetime(2022, 3, 28, 22, 19, 9, 661000,
 tzinfo=tzlocal()),
  "head_node": {
    "instance_id": "i-abcdef01234567890",
    "instance_type": "t2.micro",
    "launch_time": datetime.datetime(2022, 3, 28, 22, 21, 56, tzinfo=tzlocal()),
    "private_ip_address": "172.31.56.3",
    "public_ip_address": "107.23.100.164",
    "state": "running"
```

```
  },
  "last_updated_time": datetime.datetime(2022, 3, 28, 22, 19, 9, 661000,
 tzinfo=tzlocal()),
  "region": "us-east-1",
  "tags": [
    {
      "key": "parallelcluster:version", "value": "3.2.1"
    }
  ],
  "version": "3.2.1"
}
```

describeClusterInstances

描述属于集群的实例。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters/{clusterName}/instances
{
  "nextToken": "string",
  "nodeType": "string",
  "queueName": "string",
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

nodeType

按节点类型筛选实例。

类型：字符串

有效值：HeadNode、ComputeNode、LoginNode

必需：否

queueName

按队列名称筛选实例。

类型：字符串

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

响应语法

```
{
  "nextToken": "string",
  "instances": [
    {
      "instanceId": "string",
```

```
      "instanceType": "string",
      "launchTime": "2019-08-24T14:15:22Z",
      "privateIpAddress": "string",
      "publicIpAddress": "string",
      "state": "pending",
      "nodeType": "HeadNode",
      "queueName": "string",
      "poolName": "string"
    }
  ]
}
```

响应正文

instances

集群实例列表。

instanceId

亚马逊 EC2 实例 ID。

类型：字符串

instanceType

Amazon EC2 实例类型。

类型：字符串

launchTime

Amazon EC2 实例的启动时间。

类型：日期时间

nodeType

节点类型。

类型：字符串

有效值：`HeadNode`、`ComputeNode`、`LoginNode`

publicIpAddress

集群公共 IP 地址。

类型：字符串

queueName

Amazon EC2 实例支持节点的队列名称。

类型：字符串

状态

节点 Amazon EC2 实例状态。

类型：字符串

有效值：`pending | running | shutting-down | terminated | stopping | stopped`

nextToken

用于检索下一组结果的令牌，没有更多结果时为 `null`。

类型：字符串

示例

Python

请求

```
$ describe_cluster_instances(cluster_name_3x)
```

200 响应

```
{
  "instances": [
    {
      "instance_id": "i-abcdef01234567890",
      "instance_type": "t2.micro",
      "launch_time": datetime.datetime(2022, 3, 30, 14, 2, 7, tzinfo=tzlocal()),
      "node_type": "HeadNode",
      "private_ip_address": "192.0.2.5",
      "public_ip_address": "198.51.100.180",
      "state": "running"
```

```
    }
  ]
}
```

describeComputeFleet

描述计算实例集的状态。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters/{clusterName}/computefleet
{
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

区域

AWS 区域 集群所在的。

类型：字符串

必需：否

响应语法

```
{
  "status": "START_REQUESTED",
  "lastStatusUpdatedTime": "2019-08-24T14:15:22Z"
}
```

响应正文

状态

类型：字符串

有效值：START_REQUESTED | STARTING | RUNNING | PROTECTED | STOP_REQUESTED | STOPPING | STOPPED | UNKNOWN | ENABLED | DISABLED

lastStatusUpdated时间

代表上次状态更新时间的时间戳。

类型：日期时间

示例

Python

请求

```
$ describe_compute_fleet(cluster_name_3x)
```

200 响应

```
{
  "last_status_updated_time": datetime.datetime(2022, 3, 28, 22, 27, 14,
 tzinfo=tzlocal()),
  "status": "RUNNING"
}
```

describeImage

获取有关现有映像的详细信息。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/images/custom/{imageId}
{
  "region": "string"
}
```

请求正文

imageId

映像的 ID。

类型：字符串。

必需：是

区域

图像 AWS 区域 是在其中创建的。

类型：字符串

必需：否

响应语法

```
{
  "imageId": "string",
  "region": "string",
  "version": "string",
```

```
  "imageBuildStatus": "BUILD_IN_PROGRESS",
  "imageBuildLogsArn": "string",
  "cloudformationStackStatus": "CREATE_IN_PROGRESS",
  "cloudformationStackStatusReason": "string",
  "cloudformationStackArn": "string",
  "creationTime": "2019-08-24T14:15:22Z",
  "cloudformationStackCreationTime": "2019-08-24T14:15:22Z",
  "cloudformationStackTags": [
    {
      "key": "string",
      "value": "string"
    }
  ],
  "imageConfiguration": {
    "url": "string"
  },
  "imagebuilderImageStatus": "PENDING",
  "imagebuilderImageStatusReason": "string",
  "ec2AmiInfo": {
    "amiId": "string",
    "tags": [
      {
        "key": "string",
        "value": "string"
      }
    ],
    "amiName": "string",
    "architecture": "string",
    "state": "PENDING",
    "description": "string"
  }
}
```

响应正文

imageId

要检索其详细信息的映像的 ID。

类型：字符串

imageBuildStatus

映像构建状态。

类型：字符串

有效值：`BUILD_IN_PROGRESS | BUILD_FAILED | BUILD_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE`

imageConfiguration

url

映像配置文件的 URL。

类型：字符串

区域

AWS 区域 在其中创建镜像。

类型：字符串

version

用于构建镜像的 AWS ParallelCluster 版本。

类型：字符串

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackCreation时间

CloudFormation 堆栈的创建时间。

类型：日期时间

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS |`

UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE

cloudformationStackStatus原因

CloudFormation 堆栈状态的原因。

类型：字符串

cloudformationStackTags

CloudFormation 堆栈的标签列表。

键

标签名称。

类型：字符串

value

标签值。

类型：字符串

creationTime

图像的创建时间。

类型：日期时间

ec2 AmiInfo

amiId

亚马逊 EC2 AMI ID。

类型：字符串

amiName

亚马逊 EC2 AMI 的名称。

类型：字符串

架构

亚马逊 EC2 AMI 架构。

类型：字符串

状态

亚马逊 EC2 AMI 的状态。

类型：字符串

有效值：`PENDING | AVAILABLE | INVALID | DEREGISTERED | TRANSIENT | FAILED | ERROR`

tags

亚马逊 EC2 AMI 标签列表。

key

标签名称。

类型：字符串

值

标签值。

类型：字符串

imagebuilderImageStatus

ImageBuilder 图像状态。

类型：字符串

有效值：`PENDING | CREATING | BUILDING | TESTING | DISTRIBUTING | INTEGRATING | AVAILABLE | CANCELLED | FAILED | DEPRECATED | DELETED`

imagebuilderImageStatus原因

ImageBuilder 图片处于该状态的原因。

类型：字符串

imageBuildLogsArn

映像构建过程日志的 Amazon 资源名称 (ARN)。

类型：字符串

示例

Python

请求

```
$ describe_image(custom-image-id)
```

200 响应

```
{
  "cloudformation_stack_arn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
custom-image-id/6accc570-b080-11ec-845e-0e2dc6386985",
  "cloudformation_stack_creation_time": datetime.datetime(2022, 3, 30, 23, 23, 33,
 731000, tzinfo=tzlocal()),
  "cloudformation_stack_status": "CREATE_IN_PROGRESS",
  "cloudformation_stack_tags": [
    {
      "key": "parallelcluster:version", "value": "3.2.1"
    },
    {
      "key": "parallelcluster:image_name",
      "value": 'custom-image-id"
    },
    {
      "key": "parallelcluster:custom-image-id",
      "value": "custom-image-id"
    },
    {
      "key": 'parallelcluster:amzn-s3-demo-bucket",
      "value": 'amzn-s3-demo-bucket"
    },
    {
      "key": "parallelcluster:s3_image_dir",
      "value": "parallelcluster/3.2.1/images/custom-image-id-1234567890abcdef0"
    },
    {
      "key": "parallelcluster:build_log",
      "value": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/imagebuilder/
ParallelClusterImage-custom-image-id"
```

```
    },
    {
      "key": "parallelcluster:build_config",
      "value": "s3://amzn-s3-demo-bucket/parallelcluster/3.2.1/images/custom-image-
id-1234567890abcdef0/configs/image-config.yaml"
    }
  ],
  "image_build_logs_arn": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/
imagebuilder/ParallelClusterImage-alinux2-image",
  "image_build_status": "BUILD_IN_PROGRESS",
  "image_configuration": {
    "url": "https://amzn-s3-demo-bucket.s3.amazonaws.com/parallelcluster/3.2.1/
images/custom-image-id-1234567890abcdef0/configs/image-config.yaml?..."
  },
  "image_id": 'custom-image-id',
  "imagebuilder_image_status": "PENDING",
  "region": "us-east-1",
  "version": "3.2.1"
}
```

getClusterLog活动

检索与日志流关联的事件。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters/{clusterName}/logstreams/{logStreamName}
{
  "endTime": datetime,
  "limit": float,
  "nextToken": "string",
  "region": "string",
```

```
  "startFromHead": boolean,
  "startTime": datetime
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

logStreamName

日志流的名称。

类型：字符串。

必需：是

endTime

时间范围的结束，以 ISO 8601 格式表示。不包括时间戳等于或晚于该时间的事件。

类型：日期时间

格式：`2021-01-01T20:00:00Z`

必需：否

limit

返回的日志事件的最大数目。如果您不指定值，则最大值是响应大小为 1 MB 或最多 10,000 个日志事件中可以容纳的日志事件数。

类型：浮点数

必需：否

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

startFromHead

如果设置为 `true`，则最先返回最早的日志事件。如果值为 `false`，则最先返回最新的日志事件。默认值为 `false`。

类型：布尔值

必需：否

startTime

时间范围的开始，以 ISO 8601 格式表示。包括时间戳等于或晚于该时间的事件。

类型：日期时间

格式：`2021-01-01T20:00:00Z`

必需：否

响应语法

```
{
  "nextToken": "string",
  "prevToken": "string",
  "events": [
    {
      "timestamp": "2019-08-24T14:15:22Z",
      "message": "string"
    }
  ]
}
```

响应正文

events

筛选的事件的列表。

message

事件消息。

类型：字符串

timestamp

事件时间戳。

类型：日期时间

nextToken

用于检索下一组结果的令牌，没有更多结果时为 null。

类型：字符串

prevToken

一种标记，可用于检索前一组结果，null或者在没有其他结果时使用。

类型：字符串

示例

Python

请求

```
$ get_cluster_log_events(cluster_name_3x, log_stream_name=ip-192-0-2-26.i-abcdef01234567890.cfn-init)
```

200 响应

```
"events": [
  {
```

```
    "message": "2022-09-22 16:40:15,127 [DEBUG] CloudFormation client initialized
 with endpoint https://cloudformation.us-east-1.amazonaws.com",
    "timestamp": "2022-09-22T16:40:15.127Z"
  },
  {
    "message": "2022-09-22 16:40:15,127 [DEBUG] Describing resource
 HeadNodeLaunchTemplate in stack cluster_name_3x",
    "timestamp": "2022-09-22T16:40:15.127Z"
  },
  ...
]
```

getClusterStack活动

检索与集群的堆栈关联的事件。

> **Note**
>
> 从 3.6.0 版开始， AWS ParallelCluster 使用嵌套堆栈来创建与队列和计算资源关联的资源。`GetClusterStackEvents` API 和 `pcluster get-cluster-stack-events` 命令仅返回集群主堆栈事件。您可以在 CloudFormation 控制台中查看集群堆栈事件，包括与队列和计算资源相关的事件。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters/{clusterName}/stackevents
{
  "nextToken": "string",
  "region": "string"
```

```
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

响应语法

```
{
  "nextToken": "string",
  "events": [
    {
      "stackId": "string",
      "eventId": "string",
      "stackName": "string",
      "logicalResourceId": "string",
      "physicalResourceId": "string",
      "resourceType": "string",
      "timestamp": "2019-08-24T14:15:22Z",
      "resourceStatus": "CREATE_IN_PROGRESS",
      "resourceStatusReason": "string",
```

```
      "resourceProperties": "string",
      "clientRequestToken": "string"
    }
  ]
}
```

响应正文

events

筛选的事件的列表。

clientRequestToken

传递给生成此事件的操作的令牌。

类型：字符串

eventId

此事件的唯一 ID。

类型：字符串

logicalResourceId

模板中指定资源的逻辑名称。

类型：字符串

physicalResourceId

与资源的物理实例关联的名称或唯一标识符。

类型：字符串

resourceProperties

用于创建资源的属性的 BLOB。

类型：字符串

resourceStatus

资源状态。

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | DELETE_SKIPPED | UPDATE_IN_PROGRESS | UPDATE_FAILED | UPDATE_COMPLETE | IMPORT_FAILED | IMPORT_COMPLETE | IMPORT_IN_PROGRESS | IMPORT_ROLLBACK_IN_PROGRESS | IMPORT_ROLLBACK_FAILED | IMPORT_ROLLBACK_COMPLETE

resourceStatusReason

与资源关联的成功或失败消息。

类型：字符串

resourceType

资源的类型。

类型：字符串

stackId

堆栈实例的唯一 ID 名称。

类型：字符串

stackName

与堆栈关联的名称。

类型：字符串

timestamp

上次更新状态的时间。

类型：日期时间

nextToken

用于检索下一组结果的令牌，没有更多结果时为 null。

类型：字符串

示例

Python

请求

```
$ get_cluster_stack_events(cluster_name_3x)
```

200 响应

```
{
  "events": [
    {
      "event_id": "590b3820-b081-11ec-985e-0a7af5751497",
      "logical_resource_id": "cluster_name_3x",
      "physical_resource_id": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster_name_3x/11a59710-b080-11ec-b8bd-129def1380e9",
      "resource_status": "CREATE_COMPLETE",
      "resource_type": "AWS::CloudFormation::Stack",
      "stack_id": "arn:aws:cloudformation:us-east-1:123456789012:stack/
cluster_name_3x/11a59710-b080-11ec-b8bd-129def1380e9",
      "stack_name": "cluster_name_3x",
      "timestamp": datetime.datetime(2022, 3, 30, 23, 30, 13, 268000,
 tzinfo=tzlocal())
    },
    ...
  ]
}
```

getImageLog活动

检索与映像构建关联的事件。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/images/custom/{imageId}/logstreams/{logStreamName}
{
```

```
  "endTime": datetime,
  "limit": float,
  "nextToken": "string",
  "region": "string",
  "startFromHead": boolean,
  "startTime": datetime
}
```

请求正文

imageId

映像的 ID。

类型：字符串。

必需：是

logStreamName

日志流的名称。

类型：字符串。

必需：是

endTime

时间范围的结束，以 ISO 8601 格式表示。不包括时间戳等于或晚于该时间的事件。

类型：日期时间

格式：`2021-01-01T20:00:00Z`

必需：否

limit

返回的日志事件的最大数目。如果不指定值，则最大值为 1 MB 的响应大小所能容纳的日志事件数量，最多可达 10000 个日志事件。

类型：浮点数

必需：否

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

AWS 区域 那张图片在里面。

类型：字符串

必需：否

startFromHead

如果设置为 `true`，则最先返回最早的日志事件。如果设置为 false，则最先返回最新的日志事件。默认值为 `false`。

类型：布尔值

必需：否

startTime

时间范围的开始，以 ISO 8601 格式表示。包括时间戳等于或晚于该时间的事件。

类型：日期时间

格式：`2021-01-01T20:00:00Z`

必需：否

响应语法

```
{
  "nextToken": "string",
  "prevToken": "string",
  "events": [
    {
      "timestamp": "2019-08-24T14:15:22Z",
      "message": "string"
```

```
    }
  ]
}
```

响应正文

events

筛选的事件的列表。

message

事件消息。

类型：字符串

timestamp

事件时间戳。

类型：日期时间

nextToken

用于检索下一组结果的令牌，没有更多结果时为 null。

类型：字符串

prevToken

一种标记，可用于检索前一组结果，null或者在没有其他结果时使用。

类型：字符串

示例

Python

请求

```
$ get_image_log_events(image_id, log_stream_name=3.2.1/1)
```

200 响应

```
"events": [
  {
    "message": "ExecuteBash: STARTED EXECUTION",
    "timestamp": 2022-04-05T15:51:20.228Z"
  },
  {
    "message": "ExecuteBash: Created temporary directory: /tmp/1234567890abcdef0",
    "timestamp": "2022-04-05T15:51:20.228Z"
  },
  ...
]
```

getImageStack活动

检索与映像构建的堆栈关联的事件。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/images/custom/{imageId}/stackevents
{
  "nextToken": "string",
  "region": "string"
}
```

请求正文

imageId

映像的 ID。

类型：字符串。

必需：是

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

图像所在 AWS 区域 的那个。

类型：字符串

必需：否

响应语法

```
{
  "nextToken": "string",
  "events": [
    {
      "stackId": "string",
      "eventId": "string",
      "stackName": "string",
      "logicalResourceId": "string",
      "physicalResourceId": "string",
      "resourceType": "string",
      "timestamp": "2019-08-24T14:15:22Z",
      "resourceStatus": "CREATE_IN_PROGRESS",
      "resourceStatusReason": "string",
      "resourceProperties": "string",
      "clientRequestToken": "string"
    }
  ]
}
```

响应正文

events

筛选的事件的列表。

clientRequestToken

传递给生成此事件的操作的令牌。

类型：字符串

eventId

此事件的唯一 ID。

类型：字符串

logicalResourceId

模板中指定资源的逻辑名称。

类型：字符串

physicalResourceId

与资源的物理实例关联的名称或唯一标识符。

类型：字符串

resourceProperties

用于创建资源的属性的 BLOB。

类型：字符串

resourceStatus

资源状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | DELETE_SKIPPED | UPDATE_IN_PROGRESS | UPDATE_FAILED | UPDATE_COMPLETE | IMPORT_FAILED | IMPORT_COMPLETE | IMPORT_IN_PROGRESS | IMPORT_ROLLBACK_IN_PROGRESS | IMPORT_ROLLBACK_FAILED | IMPORT_ROLLBACK_COMPLETE`

resourceStatusReason

与资源关联的成功或失败消息。

类型：字符串

resourceType

资源的类型。

类型：字符串

stackId

堆栈实例的唯一 ID 名称。

类型：字符串

stackName

与堆栈关联的名称。

类型：字符串

timestamp

上次更新状态的时间。

类型：日期时间

nextToken

用于检索下一组结果的令牌，没有更多结果时为 `null`。

类型：字符串

示例

Python

请求

```
$ get_image_stack_events(image_id)
```

200 响应

```
{
  'events': [
    {
      'event_id': 'ParallelClusterImage-
CREATE_IN_PROGRESS-2022-03-30T23:26:33.499Z',
      'logical_resource_id': 'ParallelClusterImage',
```

```
      'physical_resource_id': 'arn:aws:imagebuilder:us-east-1:123456789012:image/
parallelclusterimage-alinux2-image/3.2.1/1',
      'resource_properties': {
        "InfrastructureConfigurationArn":"arn:aws:imagebuilder:us-
east-1:123456789012:infrastructure-configuration/parallelclusterimage-6accc570-
b080-11ec-845e-0e2dc6386985",
        "ImageRecipeArn":"arn:aws:imagebuilder:us-east-1:123456789012:image-recipe/
parallelclusterimage-alinux2-image/3.2.1",
        "DistributionConfigurationArn":"arn:aws:imagebuilder:us-
east-1:123456789012:distribution-configuration/parallelclusterimage-6accc570-
b080-11ec-845e-0e2dc6386985",
        "EnhancedImageMetadataEnabled":"false",
        "Tags": {
          "parallelcluster:image_name":"alinux2-
image","parallelcluster:image_id":"alinux2-image"
        }
      },
      'resource_status': 'CREATE_IN_PROGRESS',
      'resource_status_reason': 'Resource creation Initiated',
      'resource_type': 'AWS::ImageBuilder::Image',
      'stack_id': 'arn:aws:cloudformation:us-east-1:123456789012:stack/alinux2-
image/6accc570-b080-11ec-845e-0e2dc6386985',
      'stack_name': 'alinux2-image',
      'timestamp': datetime.datetime(2022, 3, 30, 23, 26, 33, 499000,
 tzinfo=tzlocal())
    },
    ...
  ]
}
```

listClusters

检索现有集群的列表。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters
{
  "clusterStatus": "string",
  "nextToken": "string",
  "region": "string"
}
```

请求正文

clusterStatus

按集群状态筛选。默认为所有集群。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED`

必需：否

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

集 AWS 区域 群中的。

类型：字符串

必需：否

响应语法

```
{
```

```
  "nextToken": "string",
  "clusters": [
    {
      "clusterName": "string",
      "region": "string",
      "version": "string",
      "cloudformationStackArn": "string",
      "cloudformationStackStatus": "CREATE_IN_PROGRESS",
      "clusterStatus": "CREATE_IN_PROGRESS",
      "scheduler": {
        "type": "string",
        "metadata": {
          "name": "string",
          "version": "string"
        }
      }
    }
  ]
}
```

响应正文

clusters

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE`

clusterName

集群的名称。

类型：字符串

clusterStatus

集群状态。

类型：字符串

有效值：`CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED`

scheduler

metadata

调度器元数据。

名称

调度器的名称。

类型：字符串

version

调度器版本。

类型：字符串

type

调度器的类型。

类型：字符串

区域

AWS 区域 在其中创建集群。

类型：字符串

version

用于创建集群的 AWS ParallelCluster 版本。

类型：字符串

nextToken

用于检索下一组结果的令牌，没有更多结果时为 null。

类型：字符串

示例

Python

请求

```
$ list_clusters()
```

200 响应

```
{
  'clusters':
   [
      {
        'cloudformation_stack_arn': 'arn:aws:cloudformation:us-
east-1:123456789012:stack/cluster_name_3x/16b49540-aee5-11ec-8e18-0ac1d712b241',
        'cloudformation_stack_status': 'CREATE_COMPLETE',
        'cluster_name': 'cluster_name_3x',
        'cluster_status': 'CREATE_COMPLETE',
        'region': 'us-east-1',
        'version': '3.2.1'
      },
      ...
   ]
}
```

listClusterLog直播

检索与集群关联的日志流的列表。

主题

- 请求语法

- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/clusters/{clusterName}/logstreams
{
  "filters": [ "string" ],
  "nextToken": "string",
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

filters

筛选日志流。

可接受的筛选器包括：

- private-dns-name：实例私有 DNS 名称的缩写形式（例如 ip-10-0-0-0-101）。
- node-type：有效值：`HeadNode`。

类型：字符串数组，唯一

格式：`Name=a,Values=1 Name=b,Values=2,3`

必需：否

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

响应语法

```
{
  "nextToken": "string",
  "logStreams": [
    {
      "logStreamName": "string",
      "creationTime": "2019-08-24T14:15:22Z",
      "firstEventTimestamp": "2019-08-24T14:15:22Z",
      "lastEventTimestamp": "2019-08-24T14:15:22Z",
      "lastIngestionTime": "2019-08-24T14:15:22Z",
      "uploadSequenceToken": "string",
      "logStreamArn": "string"
    }
  ]
}
```

响应正文

logStreams

日志流的列表。

creationTime

创建流的时间。

类型：日期时间

firstEventTimestamp

流中第一个事件的时间。

类型：日期时间

lastEventTimestamp

流中最后一个事件的时间。该 lastEventTime 值以最终一致性为基础进行更新。它通常会在摄取后不到一小时内更新，但在极少数情况下可能需要更长时间。

类型：日期时间

lastIngestionTime

上次摄取时间。

类型：日期时间

logStreamArn

日志流的 Amazon 资源名称 (ARN)。

类型：字符串

logStreamName

日志流的名称。

类型：字符串

uploadSequenceToken

序列令牌。

类型：字符串

nextToken

用于检索下一组结果的令牌，没有更多结果时为 `null`。

类型：字符串

示例

Python

请求

```
$ list_cluster_log_streams(cluster_name_3x)
```

200 响应

```
{
  'log_streams': [
      {
        'creation_time': datetime.datetime(2022, 3, 30, 14, 7, 34, 354000,
 tzinfo=tzlocal()),
        'first_event_timestamp': datetime.datetime(2022, 3, 30, 14, 6, 41, 444000,
 tzinfo=tzlocal()),
        'last_event_timestamp': datetime.datetime(2022, 3, 30, 14, 25, 55, 462000,
 tzinfo=tzlocal()),
        'last_ingestion_time': datetime.datetime(2022, 3, 30, 14, 49, 50, 62000,
 tzinfo=tzlocal()),
        'log_stream_arn': 'arn:aws:logs:us-east-1:123456789012:log-group:/aws/
parallelcluster/cluster_name_3x:log-stream:ip-192-0-2-26.i-abcdef01234567890.cfn-
init',
        'log_stream_name': 'ip-192-0-2-26.i-abcdef01234567890.cfn-init',
         ...
        'upload_sequence_token': '####'
      },
      ...
   ]
}
```

listImageLog直播

检索与映像关联的日志流的列表。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/images/custom/{imageId}/logstreams
```

```
{
  "nextToken": "string",
  "region": "string"
}
```

请求正文

imageId

映像的 ID。

类型：字符串。

必需：是

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

图像所在 AWS 区域 的那个。

类型：字符串

必需：否

响应语法

```
{
  "nextToken": "string",
  "logStreams": [
    {
      "logStreamName": "string",
      "creationTime": "2019-08-24T14:15:22Z",
      "firstEventTimestamp": "2019-08-24T14:15:22Z",
      "lastEventTimestamp": "2019-08-24T14:15:22Z",
      "lastIngestionTime": "2019-08-24T14:15:22Z",
      "uploadSequenceToken": "string",
```

```
      "logStreamArn": "string"
    }
  ]
}
```

响应正文

logStreams

日志流的列表。

creationTime

创建流的时间。

类型：日期时间

firstEventTimestamp

流中第一个事件的时间。

类型：日期时间

lastEventTimestamp

流中最后一个事件的时间。该 lastEventTime 值以最终一致性为基础进行更新。它通常会在摄取后不到一小时内更新，但在极少数情况下可能需要更长时间。

类型：日期时间

lastIngestionTime

上次摄取时间。

类型：日期时间

logStreamArn

日志流的 Amazon 资源名称 (ARN)。

类型：字符串

logStreamName

日志流的名称。

类型：字符串

uploadSequenceToken

序列令牌。

类型：字符串

next_token

用于检索下一组结果的令牌，没有更多结果时为 null。

类型：字符串

示例

Python

请求

```
$ list_image_log_streams(custom-image-id)
```

200 响应

```
{
  'log_streams': [
    {
      'creation_time': datetime.datetime(2022, 3, 29, 20, 29, 24, 875000,
 tzinfo=tzlocal()),
      'first_event_timestamp': datetime.datetime(2022, 3, 29, 20, 29, 24, 775000,
 tzinfo=tzlocal()),
      'last_event_timestamp': datetime.datetime(2022, 3, 29, 20, 38, 23, 944000,
 tzinfo=tzlocal()),
      'last_ingestion_time': datetime.datetime(2022, 3, 29, 20, 51, 56, 26000,
 tzinfo=tzlocal()),
      'log_stream_arn': 'arn:aws:logs:us-east-1:123456789012:log-group:/aws/
imagebuilder/ParallelClusterImage-alinux2-image:log-stream:3.2.1/1',
      'log_stream_name': '3.2.1/1',
      'upload_sequence_token': '####'
    },
    ...
  ]
}
```

listImages

检索现有自定义映像的列表。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /images/custom
{
  "imageStatus": "string",
  "nextToken": "string",
  "region": "string"
}
```

请求正文

imageStatus

按提供的状态筛选映像。

类型：字符串

有效值：`AVAILABLE | PENDING | FAILED`

必需：是

nextToken

用于获取下一组结果的令牌。

类型：字符串

必需：否

region

AWS 区域 那张图片在里面。

类型：字符串

必需：否

响应语法

```
{
  "nextToken": "string",
  "images": [
    {
      "imageId": "string",
      "ec2AmiInfo": {
        "amiId": "string"
      },
      "region": "string",
      "version": "string",
      "cloudformationStackArn": "string",
      "imageBuildStatus": "BUILD_IN_PROGRESS",
      "cloudformationStackStatus": "CREATE_IN_PROGRESS"
    }
  ]
}
```

响应正文

images

映像的列表。

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE

ec2 AmiInfo

ami_id

亚马逊 EC2 AMI ID。

类型：字符串

imageBuildStatus

映像构建状态。

有效值：BUILD_IN_PROGRESS | BUILD_FAILED | BUILD_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE

类型：字符串

imageId

映像的 ID。

类型：字符串

区域

AWS 区域 在其中创建镜像。

类型：字符串

version

用于构建镜像的 AWS ParallelCluster 版本。

类型：字符串

nextToken

用于检索下一组结果的令牌，没有更多结果时为 null。

类型：字符串

示例

Python

请求

```
$ list_images("AVAILABLE")
```

200 响应

```
{
  'images': [
    {
      'ec2_ami_info': {
        'ami_id': 'ami-abcdef01234567890'
      },
      'image_build_status': 'BUILD_COMPLETE',
      'image_id': 'custom-image',
      'region': 'us-east-1',
      'version': '3.2.1'
    }
  ]
}
```

listOfficialImages

检索 AWS ParallelCluster 官方图片列表。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
GET /v3/images/official
```

```
{
  "architecture": "string",
  "os": "string",
  "region": "string"
}
```

请求正文

架构

按架构筛选。默认为不筛选。

类型：字符串

有效值：`x86_64 | arm64`

必需：否

os

按操作系统分发进行筛选。默认为不筛选。

类型：字符串

有效值：`alinux2 | alinux2023 | ubuntu2404 | ubuntu2204 | ubuntu2004 | rhel8 | rhel9`

必需：否

region

其中 AWS 区域 列出了官方图片。

类型：字符串

必需：否

响应语法

```
{
  "images": [
    {
      "architecture": "string",
      "amiId": "string",
```

```
      "name": "string",
      "os": "string",
      "version": "string"
    }
  ]
}
```

响应正文

images

amiId

AMI 的 ID。

类型：字符串

架构

AMI 架构。

类型：字符串

名称

AMI 的名称。

类型：字符串

os

AMI 操作系统。

类型：字符串

version

AWS ParallelCluster 版本。

类型：字符串

示例

Python

请求

```
$ list_official_images()
```

200 响应

```
{
  'images': [
    {
      'ami_id': 'ami-015cfeb4e0d6306b2',
      'architecture': 'x86_64',
      'name': 'aws-parallelcluster-3.2.1-ubuntu-2004-lts-hvm-x86_64-202202261505 '
      '2022-02-26T15-08-34.759Z',
      'os': 'ubuntu2004',
      'version': '3.2.1'
    },
    ...
  ]
}
```

updateCluster

更新集群。

主题

- 请求语法
- 请求正文
- 响应语法
- 响应正文
- 示例

请求语法

```
PUT /v3/clusters/{clusterName}
{
  "clusterConfiguration": "string",
  "dryrun": boolean,
  "forceUpdate": boolean,
  "region": "string",
```

```
  "suppressValidators": "string",
  "validationFailureLevel": "string"
}
```

请求正文

clusterConfiguration

集群配置作为 YAML 文档。

必需：是

clusterName

集群的名称。

类型：字符串。

必需：是

dryrun

如果设置为 `true`，则仅执行请求验证而不创建任何资源。使用此参数可验证集群配置和更新要求。默认值为 `false`。

类型：布尔值

必需：否

forceUpdate

如果设置为 `true`，则忽略更新验证错误并强制更新。默认值为 `false`。

类型：布尔值

必需：否

region

AWS 区域 集群所在的。

类型：字符串

必需：否

suppressValidators

标识一个或多个要禁止的配置验证器。

类型：字符串

格式：(ALL|type:[A-Za-z0-9]+)

必需：否

有效值示例：currentValue、requestedValue、message

validationFailureLevel

导致更新失败的最低验证级别。

类型：字符串

有效值：INFO | WARNING | ERROR

必需：否

响应语法

```
{
  "cluster": {
    "clusterName": "string",
    "region": "string",
    "version": "string",
    "cloudformationStackArn": "string",
    "cloudformationStackStatus": "UPDATE_IN_PROGRESS",
    "clusterStatus": "UPDATE_IN_PROGRESS",
    "scheduler": {
      "type": "string",
      "metadata": {
        "name": "string",
        "version": "string"
      }
    }
  },
  "validationMessages": [
    {
      "id": "string",
```

```
      "type": "string",
      "level": "INFO",
      "message": "string"
    }
  ],
  "changeSet": [
    {
      "parameter": "string",
      "currentValue": "string",
      "requestedValue": "string"
    }
  ]
}
```

响应正文

changeSet

集群更新的更改集。

currentValue

要更新的参数的当前值。

类型：字符串

参数

要更新的参数。

类型：字符串

requestedValue

要更新的参数的请求值。

类型：字符串

cluster

cloudformationStackArn

主 CloudFormation 堆栈的亚马逊资源名称 (ARN)。

类型：字符串

cloudformationStackStatus

堆 CloudFormation 栈状态。

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | ROLLBACK_IN_PROGRESS | ROLLBACK_FAILED | ROLLBACK_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_ROLLBACK_IN_PROGRESS | UPDATE_ROLLBACK_FAILED | UPDATE_ROLLBACK_COMPLETE_CLEANUP_IN_PROGRESS | UPDATE_ROLLBACK_COMPLETE

clusterName

集群的名称。

类型：字符串

clusterStatus

集群状态。

类型：字符串

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | DELETE_COMPLETE | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED

区域

AWS 区域 在其中创建集群。

类型：字符串

scheduler

metadata

调度器元数据。

名称

调度器的名称。

类型：字符串

version

调度器版本。

类型：字符串

type

调度器类型。

类型：字符串

version

AWS ParallelCluster 用于创建集群的版本。

类型：字符串

validationMessages

验证级别低于 `validationFailureLevel` 的消息的列表。消息列表是在配置验证期间收集的。

id

验证器的 ID。

类型：字符串

level

验证级别。

类型：字符串

有效值：`INFO | WARNING | ERROR`

message

验证消息。

类型：字符串

type

验证器的类型。

类型：字符串

示例

Python

请求

```
$ update_cluster(cluster_name_3x, path/config-file.yaml)
```

200 响应

```
{
  'change_set': [
    {
      'current_value': '10',
      'parameter':
 'Scheduling.SlurmQueues[queue1].ComputeResources[t2micro].MaxCount',
      'requested_value': '15'
    }
  ],
  'cluster': {
    'cloudformation_stack_arn': 'arn:aws:cloudformation:us-
east-1:123456789012:stack/test-api-cluster/e0462730-50b5-11ed-99a3-0a5ddc4a34c7',
    'cloudformation_stack_status': 'UPDATE_IN_PROGRESS',
    'cluster_name': 'cluster-3x',
    'cluster_status': 'UPDATE_IN_PROGRESS',
    'region': 'us-east-1',
    'scheduler': {
      'type': 'slurm'
    },
    'version': '3.2.1'
  }
}
```

updateComputeFleet

更新集群计算实例集的状态。

主题

- 请求语法
- 请求正文

- 响应语法
- 响应正文
- 示例

请求语法

```
PATCH /v3/clusters/{clusterName}/computefleet
{
  "status": "string",
  "region": "string"
}
```

请求正文

clusterName

集群的名称。

类型：字符串。

必需：是

状态

计算实例集状态。

类型：字符串

有效值：`START_REQUESTED | STOP_REQUESTED | ENABLED | DISABLED`

必需：是

区域

AWS 区域 集群所在的。

类型：字符串

必需：否

响应语法

```
{
```

```
  "status": "START_REQUESTED",
  "lastStatusUpdatedTime": "2019-08-24T14:15:22Z"
}
```

响应正文

状态

计算实例集状态。

类型：字符串

有效值：START_REQUESTED | STARTING | RUNNING | PROTECTED | STOP_REQUESTED | STOPPING | STOPPED | UNKNOWN | ENABLED | DISABLED

lastStatusUpdated时间

代表上次状态更新时间的时间戳。

类型：日期时间

示例

Python

请求

```
$ update_compute_fleet(cluster_name_3x,  "START_REQUESTED")
```

200 响应

```
{
  'last_status_updated_time': datetime.datetime(2022, 3, 28, 22, 27, 14,
 tzinfo=tzlocal()),
  'status': 'START_REQUESTED'
}
```

AWS ParallelCluster Python 库 API

从 3.5.0 AWS ParallelCluster 版本开始，你可以使用 AWS ParallelCluster Python 库 AWS ParallelCluster 进行访问。您可以在自己的`pcluster`环境中或在 AWS Lambda 运行时中访问该 AWS

ParallelCluster 库。了解如何使用 AWS ParallelCluster Python 库访问 AWS ParallelCluster 该 API。P AWS ParallelCluster ython 库提供的功能与 AWS ParallelCluster API 提供的功能相同。

AWS ParallelCluster Python 库的操作和参数在转换为不`snake_case`使用大写字母时会反映 API 参数的操作和参数。

主题

- AWS ParallelCluster Python 库授权
- 安装 AWS ParallelCluster Python 库
- 集群 API 操作
- 计算实例集 API 操作
- 集群和堆栈日志操作
- 映像 API 操作
- 映像和堆栈日志操作
- 示例
- AWS Lambda 用于 AWS ParallelCluster Python 库

AWS ParallelCluster Python 库授权

使用对 boto3 有效的任何标准方式指定凭证。有关更多信息，请参阅 boto3 文档。

安装 AWS ParallelCluster Python 库

1. 按照设置 AWS ParallelCluster 中的说明安装 `pcluster` CLI 版本 3.5.0 或更高版本。
2. 导入 `pcluster` 模块并开始使用库，如以下示例所示：

```
import pcluster.lib as pc
pc.create_cluster(cluster_name="mycluster", cluster_configuration="config.yaml"
```

集群 API 操作

主题

- list_clusters
- create_cluster

- delete_cluster
- describe_cluster
- update_cluster

list_clusters

```
list_clusters(region, next_token, cluster_status)
```

检索现有集群的列表。

参数：

region

列出部署到给定对象的集群 AWS 区域。

next_token

用于获取下一组结果的令牌。

cluster_status

按集群状态筛选。默认为列出所有集群。

有效值：CREATE_IN_PROGRESS | CREATE_FAILED | CREATE_COMPLETE | DELETE_IN_PROGRESS | DELETE_FAILED | UPDATE_IN_PROGRESS | UPDATE_COMPLETE | UPDATE_FAILED

create_cluster

```
create_cluster(cluster_name, cluster_configuration, region, suppress_validators,
 validation_failure_level, dry_run, rollback_on_failure, wait)
```

在指定区域内创建集群。

参数：

cluster_name（必需）

集群名称。

cluster_configuration（必需）

作为 Python 数据类型的集群配置。

region

集群 AWS 区域。

suppress_validators

标识一个或多个要禁止的集群配置验证器。

格式：`(ALL | type:[A-Za-z0-9]+)`

validation_failure_level

导致集群创建失败的最低验证级别。默认值为 `ERROR`。

有效值：`INFO` | `WARNING` | `ERROR`。

dry_run

执行请求验证而不创建任何资源。您可以使用此参数来验证集群配置。默认值为 `False`。

rollback_on_failure

如果设置为`True`，则在 AWS ParallelCluster 出现故障时自动启动集群堆栈回滚。默认值为 `True`。

wait

如果设置为 `True`，则 AWS ParallelCluster 等待操作完成。默认值为 `False`。

delete_cluster

```
delete_cluster(cluster_name, region, wait)
```

删除指定区域中的集群。

参数：

cluster_name（必需）

集群名称。

region

集群 AWS 区域。

wait

如果设置为 `True`，则等待操作完成。默认值为 `False`。

describe_cluster

```
describe_cluster(cluster_name, region)
```

获取有关现有集群的详细信息。

参数：

cluster_name（必需）

集群名称。

region

集群 AWS 区域。

update_cluster

```
update_cluster(cluster_name, cluster_configuration, suppress_validators,
 validation_failure_level, region, force_update, dry_run, wait)
```

更新指定区域中的集群。

参数：

cluster_name（必需）

集群名称。

cluster_configuration（必需）

作为 Python 数据类型的集群配置。

suppress_validators

标识一个或多个要禁止的集群配置验证器。

格式：(ALL | type:[A-Za-z0-9]+)

validation_failure_level

导致集群更新失败的最低验证级别。默认值为 ERROR。

有效值：INFO |WARNING |ERROR

region

集群 AWS 区域。

dry_run

执行请求验证而不创建或更新任何资源。您可以使用此参数来验证集群配置。默认值为 False。

force_update

如果设置为 True，则通过忽略更新验证错误强制更新。默认值为 False。

wait

如果设置为 True，则等待操作完成。默认值为 False。

计算实例集 API 操作

主题

- describe_compute_fleet
- update_compute_fleet
- delete_cluster_instances
- describe_cluster_instances

describe_compute_fleet

```
describe_compute_fleet(cluster_name, region)
```

描述指定集群的集群计算实例集的状态。

参数：

cluster_name（必需）

集群名称。

region

描述部署到给定集群的计算队列状态 AWS 区域。

update_compute_fleet

```
update_compute_fleet(cluster_name, status, region)
```

更新集群计算实例集的状态。

参数：

cluster_name（必需）

集群名称。

status（必需）

要更新到的状态。

有效值：START_REQUESTED | STOP_REQUESTED | ENABLED | DISABLED

region

集群 AWS 区域。

delete_cluster_instances

```
delete_cluster_instances(cluster_name, region, force)
```

开始强制终止所有集群计算节点。此操作不支持 AWS Batch 集群。

参数：

cluster_name（必需）

集群名称。

region

集群 AWS 区域。

force

如果设置为 `True`，则在找不到具有指定 `cluster_name` 的集群时强制删除。默认值为 `False`。

describe_cluster_instances

```
describe_cluster_instances(cluster_name, region, next_token, node_type, queue_name)
```

描述集群的实例。

参数：

cluster_name（必需）

集群名称。

region

集群 AWS 区域。

next_token

用于获取下一组结果的令牌。

node_type

按 `node_type` 筛选实例。

有效值：`HeadNode` | `ComputeNode`

queue_name

按队列名称筛选实例。

集群和堆栈日志操作

主题

- list_cluster_log_streams
- get_cluster_log_events
- get_cluster_stack_events

list_cluster_log_streams

```
list_cluster_log_streams(cluster_name, region, filters, next_token)
```

列出指定集群的日志流。

参数：

cluster_name（必需）

集群名称。

region

集群 AWS 区域。

filters

筛选集群日志流。

格式：`'Name=a,Values=1 Name=b,Values=2,3'`

接受的筛选器：

code-dns-name

实例私有 DNS 名称的短格式；例如 `ip-10-0-0-101`。

node-type

节点类型。

有效值：`HeadNode`

next_token

用于获取下一组结果的令牌。

get_cluster_log_events

```
get_cluster_log_events(cluster_name, log_stream_name, region, next_token,
 start_from_head, limit, start_time, end_time)
```

获取指定集群和日志流的日志事件。

参数：

cluster_name（必需）

集群名称。

log_stream_name（必需）

日志流名称。

region

集群 AWS 区域。

next_token

用于获取下一组结果的令牌。

start_from_head

如果设置为`True`，则首先 AWS ParallelCluster 返回最早的日志事件。如果设置为 `False`，则最先返回最新的日志事件。默认值为 `False`。

limit

返回的日志事件的最大数目。如果不指定值，则最大值为 1 MB 的响应大小所能容纳的日志数量，最多可达 10000 个日志事件。

start_time

日志事件时间范围的开始时间，以 ISO 8601 格式表示，例如 `'2021-01-01T20:00:00Z'`。包括时间戳等于或晚于该时间的事件。

end_time

日志事件时间范围的结束时间，以 ISO 8601 格式表示，例如 `'2021-01-01T20:00:00Z'`。不包括时间戳等于或晚于该时间的事件。

get_cluster_stack_events

```
get_cluster_stack_events(cluster_name, region, next_token)
```

获取指定集群的堆栈事件。

参数：

cluster_name（必需）

集群名称。

region

集群 AWS 区域。

next_token

用于获取下一组结果的令牌。

映像 API 操作

主题

- list_images
- build_image
- delete_image
- describe_image

list_images

```
list_images(image_status, region, next_token)
```

检索现有映像的列表。

参数：

image_status（必需）

按映像状态筛选。

有效值：AVAILABLE |PENDING |FAILED

region

列出在指定 AWS 区域中构建的映像。

next_token

用于获取下一组结果的令牌。

build_image

```
build_image(image_configuration, image_id, suppress_validators,
 validation_failure_level, dry_run, rollback_on_failure, region)
```

在给定区域创建自定义 AWS ParallelCluster 镜像。

参数：

image_configuration（必需）

作为 Python 数据的映像配置。

image_id（必需）

映像 ID。

suppress_validators

标识一个或多个要禁止的映像配置验证器。

格式：`(ALL | type:[A-Za-z0-9]+)`

validation_failure_level

导致映像创建失败的最低验证级别。默认值为 `ERROR`。

有效值：`INFO` |`WARNING` |`ERROR`

dry_run

如果设置为`True`，则在不创建任何资源的情况下 AWS ParallelCluster 执行请求验证。您可以使用此参数来验证映像配置。默认值为 `False`。

rollback_on_failure

如果设置为`True`，则在 AWS ParallelCluster 出现故障时自动启动映像堆栈回滚。默认值为 `False`。

region

映像 AWS 区域。

delete_image

```
delete_image(image_id, region, force)
```

删除指定区域中的映像。

参数：

image_id（必需）

映像 ID。

region

映像 AWS 区域。

force

如果设置为`True`，则如果实例正在使用 AMI 或 AMI 已共享，则 AWS ParallelCluster 强制删除。默认值为 `False`。

describe_image

```
describe_image(image_id, region)
```

获取有关现有映像的详细信息。

参数：

image_id（必需）

映像 ID。

region

映像 AWS 区域。

映像和堆栈日志操作

主题

- list_image_log_streams
- get_image_log_events

- get_image_stack_events
- list_official_images

list_image_log_streams

```
list_image_log_streams(image_id, region, next_token)
```

列出映像的日志流。

参数：

image_id（必需）

映像 ID。

region

映像 AWS 区域。

next_token

用于获取下一组结果的令牌。

get_image_log_events

```
get_image_log_events(image_id, log_stream_name, region, next_token, start_from_head,
 limit, start_time, end_time)
```

获取指定映像和日志流的日志事件。

参数：

image_id（必需）

映像 ID。

log_stream_name（必需）

日志流名称。

region

映像 AWS 区域。

next_token

用于获取下一组结果的令牌。

start_from_head

如果设置为`True`，则首先 AWS ParallelCluster 返回最早的日志事件。如果设置为 `False`，则最先返回最新的日志事件。默认值为 `False`。

limit

返回的日志事件的最大数目。如果不指定值，则最大值为 1 MB 的响应大小所能容纳的日志数量，最多可达 10000 个日志事件。

start_time

日志事件时间范围的开始时间，以 ISO 8601 格式表示，例如 `'2021-01-01T20:00:00Z'`。包括时间戳等于或晚于该时间的事件。

end_time

日志事件时间范围的结束时间，以 ISO 8601 格式表示，例如 `'2021-01-01T20:00:00Z'`。不包括时间戳等于或晚于该时间的事件。

get_image_stack_events

```
get_image_stack_events(image_id, region, next_token)
```

获取指定映像的堆栈事件。

参数：

image_id（必需）

映像 ID。

region

映像 AWS 区域。

next_token

用于获取下一组结果的令牌。

list_official_images

```
list_official_images(region,os, architecture)
```

检索官方 AWS ParallelCluster 图片列表。

参数：

region

映像 AWS 区域。

os

按操作系统分发进行筛选。默认为不筛选。

architecture

按架构筛选。默认为不筛选。

示例

主题

- 创建集群

创建集群

当您运行以下示例脚本时，在将指定的输入存储在您的环境中后，您就会创建一个集群。集群配置是根据集群配置文档作为一种 Python 数据类型创建的。

```
import os
import pprint
import pcluster.lib as pc
pp = pprint.PrettyPrinter()

HEAD_NODE_SUBNET = os.environ["HEAD_NODE_SUBNET"]
COMPUTE_NODE_SUBNET = os.environ["HEAD_NODE_SUBNET"]
KEY_NAME = os.environ["KEY_NAME"]
CONFIG = {'Image': {'Os': 'alinux2'},
          'HeadNode': {'InstanceType': 't2.large',
                       'Networking': {'SubnetId': HEAD_NODE_SUBNET},
```

```
                        'Ssh': {'KeyName': KEY_NAME}},

          'Scheduling': {'Scheduler': 'slurm',
                         'SlurmQueues':
                         [{'Name': 'queue0',
                           'ComputeResources':
                           [{'Name': 'queue0-i0', 'InstanceType': 't2.micro',
                             'MinCount': 0, 'MaxCount': 10}],
                           'Networking': {'SubnetIds': [COMPUTE_NODE_SUBNET]}}]}}


pp.pprint(pc.create_cluster(cluster_name="mycluster", cluster_configuration=CONFIG))
```

输出：

```
{'cluster': {'cloudformationStackArn': 'arn:aws:cloudformation:us-
east-2:123456789012:stack/mycluster/00000000-aaaa-1111-999-000000000000',
             'cloudformationStackStatus': 'CREATE_IN_PROGRESS',
             'clusterName': 'mycluster',
             'clusterStatus': 'CREATE_IN_PROGRESS',
             'region': 'us-east-2',
             'scheduler': {'type': 'slurm'},
             'version': '3.13.2'}}
```

AWS Lambda 用于 AWS ParallelCluster Python 库

您可以部署 Lambda 层和运行时来访问 Pyth AWS ParallelCluster on 库。我们托管 AWS ParallelCluster zip 文件，您可以按照以下步骤中的说明输入指向 zip 文件的链接来使用这些文件。Lambda 使用 zip 文件来准备运行时系统环境，以支持对 Python 库的访问。 AWS ParallelCluster Python 库是在 3.5.0 AWS ParallelCluster 版本中添加的。您只能对版本 3.5.0 和更高版本使用该库。

托管的 zip 文件 URL 的格式为：`s3://`*`aws-region-id`*`-aws-parallelcluster/parallelcluster/`*`3.13.2`*`/layers/aws-parallelcluster/lambda-layer.zip`。（*`3.13.2`*替换为下一步中要使用的 AWS ParallelCluster 版本。）

使用以下命令开始访问 P AWS ParallelCluster ython 库 AWS Lambda

创建 Lambda 层

1. 登录 AWS Management Console 并导航到 AWS Lambda 控制台。
2. 在导航窗格中选择层，然后选择创建层。

3. 输入层的名称，然后选择从 Amazon S3 上传文件。
4. 输入 zip 文件的 URL：s3://*aws-region-id*-aws-parallelcluster/parallel *3.13.2* cluster//-layer.zip。layers/aws-parallelcluster/lambda
5. 对于兼容架构，选择 x86_64 架构。
6. 对于兼容运行时，请选择 Python 3.12 运行时。
7. 选择创建。

使用 Lambda 层

1. 在 Lambda 控制台导航窗格中，依次选择函数、创建函数。
2. 输入您的函数的名称。
3. 在 “运行时” 中，选择 Python 3.12 运行时。
4. 对于架构，选择 x86_64 架构。
5. 选择创建函数。
6. 创建函数后，选择层，然后选择添加层。
7. 选择自定义层，然后选择您在之前的步骤中创建的层。
8. 选择层版本。
9. 选择添加。
10. 您的 Lambda 需要权限才能管理使用创建的集群。 AWS ParallelCluster创建具有基本 AWS ParallelCluster `pcluster` 用户策略中所列权限的 Lambda 角色。

现在，您可以 AWS ParallelCluster 从 Python 库中进行访问，如中所述AWS ParallelCluster Python 库 API。

关于如何使用的教程 AWS ParallelCluster

以下教程向您展示了如何开始使用 AWS ParallelCluster 版本 3，并为一些常见任务提供了最佳实践指导。

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

主题

- 在上运行你的第一份工作 AWS ParallelCluster
- 构建自定义 AWS ParallelCluster AMI
- 集成 Active Directory
- 使用密 AWS KMS 钥配置共享存储加密
- 在多队列模式集群中运行作业
- 使用 AWS ParallelCluster API
- 使用创建集群 Slurm 会计
- 使用外部集群创建集群 Slurmdbd 会计
- 恢复到以前的 S AWS ystems Manager 文档版本
- 使用创建集群 AWS CloudFormation
- 使用 Terraform 部署 ParallelCluster API
- 使用 Terraform 创建集群
- 使用 Terraform 创建自定义 AMI
- AWS ParallelCluster 用户界面与身份中心集成
- 使用 Pyxis 运行容器化作业
- 创建具有启用 EFA 的 Lustre FSx 的集群

在上运行你的第一份工作 AWS ParallelCluster

本教程将引导你完成在上面运行第一个 Hello World 作业 AWS ParallelCluster

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

先决条件

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 你有一个 A mazon EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

验证安装

首先，我们验证安装和配置 AWS ParallelCluster 是否正确，包括 Node.js 依赖项。

```
$ node --version
v16.8.0
$ pcluster version
{
  "version": "3.13.2"
}
```

这将返回的运行版本 AWS ParallelCluster。

创建您的第一个集群

现在应该创建您的第一个集群了。由于本教程的工作负载不是性能密集型的，因此，我们可以使用 `t2.micro` 的默认实例大小。（对于生产工作负载，您需要选择最适合您的需求的实例大小。）我们将您的集群命名为 `hello-world`。

```
$ pcluster create-cluster \
    --cluster-name hello-world \
    --cluster-configuration hello-world.yaml
```

> **Note**
>
> 必须 AWS 区域 为大多数`pcluster`命令指定要使用的。如果未在 `AWS_DEFAULT_REGION` 环境变量或 `~/.aws/config` 文件 `[default]` 部分的 `region` 设置中指定，则必须在 `pcluster` 命令行中提供 `--region` 参数。

如果输出为您提供有关配置的消息，您将需要运行以下命令来配置 AWS ParallelCluster：

```
$ pcluster configure --config hello-world.yaml
```

如果 pcluster create-cluster 命令成功，则将显示类似于以下内容的输出：

```
{
  "cluster": {
    "clusterName": "hello-world",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:xxx:stack/xxx",
    "region": "...",
    "version": "...",
    "clusterStatus": "CREATE_IN_PROGRESS"
  }
}
```

您可以使用以下命令监控集群的创建：

```
$ pcluster describe-cluster --cluster-name hello-world
```

正在创建集群时，clusterStatus 会报告“CREATE_IN_PROGRESS”。成功创建集群后，clusterStatus 将转变为“CREATE_COMPLETE”。输出还为我们提供头节点的 publicIpAddress 和 privateIpAddress。

登录到头节点

使用您的 OpenSSH pem 文件登录到头节点。

```
$ pcluster ssh --cluster-name hello-world -i /path/to/keyfile.pem
```

登录后，请运行命令 sinfo 以验证您的计算节点是否已设置和配置。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
queue1*      up   infinite     10  idle~ queue1-dy-queue1t2micro-[1-10]
```

输出显示我们的集群中有一个队列，最多包含十个节点。

使用 Slurm 运行首个作业

接下来，我们将创建一个作业，该作业睡眠一小段时间，然后输出它自己的主机名。使用以下内容创建名为 `hellojob.sh` 的文件。

```
#!/bin/bash
sleep 30
echo "Hello World from $(hostname)"
```

接下来，使用 `sbatch` 提交作业，并验证其是否运行。

```
$ sbatch hellojob.sh
Submitted batch job 2
```

现在，您可以查看您的队列并检查该作业的状态。新 Amazon EC2 实例的配置将在后台启动。您可以使用 `sinfo` 命令监控集群实例的状态。

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
                 2    queue1 hellojob ec2-user CF       3:30      1 queue1-dy-
queue1t2micro-1
```

输出显示作业已提交给 `queue1`。请等候 30 秒，以便作业完成，然后再次运行 `squeue`。

```
$ squeue
             JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
```

现在，队列中没有作业，我们可以检查当前目录中的输出。

```
$ ls -l
total 8
-rw-rw-r-- 1 ec2-user ec2-user 57 Sep  1 14:25 hellojob.sh
-rw-rw-r-- 1 ec2-user ec2-user 43 Sep  1 14:30 slurm-2.out
```

在输出中，我们看到一个“out”文件。我们可以查看作业的输出：

```
$ cat slurm-2.out
Hello World from queue1-dy-queue1t2micro-1
```

输出还显示我们的作业已在实例 `queue1-dy-queue1t2micro-1` 上成功运行。

在刚创建的集群中，只有主目录在集群的所有节点之间共享。

要了解有关创建和使用集群的更多信息，请参阅最佳实践。

如果您的应用程序需要共享软件、库或数据，请考虑以下选项：

- 如中所述，构建包含您的软件的 AWS ParallelCluster 已启用的自定义 AMI 构建自定义 AWS ParallelCluster AMI。
- 使用 AWS ParallelCluster 配置文件中的StorageSettings选项来指定共享文件系统并将已安装的软件存储在指定的装载位置。
- 使用自定义引导操作自动执行集群中每个节点的引导过程。

构建自定义 AWS ParallelCluster AMI

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

> **Important**
>
> 如果构建自定义 AMI，则必须重复执行用于随每个新的 AWS ParallelCluster 版本创建自定义 AMI 的步骤。

在进一步阅读之前，我们建议您首先查看自定义引导操作一节。确定未来 AWS ParallelCluster 版本是否可以脚本化并支持您要进行的修改。

尽管通常构建自定义 AMI 并不理想，但在某些特定场景中，需要 AWS ParallelCluster 为其构建自定义 AMI。本教程介绍如何为这些场景构建自定义 AMI。

先决条件

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 你有一个 A mazon EC2 密钥 pair。
- 您拥有具有运行 `pcluster` CLI 和构建映像所需的权限的 IAM 角色。

如何自定义 AWS ParallelCluster AMI

有两种方法可以构建自定义 AWS ParallelCluster AMI。这两种方法之一是使用 CLI 构建新 AM AWS ParallelCluster I。另一种方法需要进行手动修改以构建可在您的 AWS 账户下使用的新 AMI。

构建自定义 AWS ParallelCluster AMI

如果您有自定义 AMI 和软件，则可以在其 AWS ParallelCluster 上应用所需的更改。 AWS ParallelCluster 依靠 EC2 Image Builder 服务进行自定义构建 AMIs。有关更多信息，请参阅 Image Builder User Guide。

关键点：

- 该过程需要 1 小时左右的时间。如果在构建时需要安装其他`Build`/`Components`，则此时间可能会有所不同。
- AMI 标有主要组件的版本。其中包括内核、调度器和 EFA 驱动程序。还会在 AMI 描述中报告一部分组件版本。
- 从 AWS ParallelCluster 3.0.0 开始，可以使用一组新的 CLI 命令来管理映像的生命周期。这包括 `build-image`、`list-images`、`describe-image` 和 `delete-image`。
- 这种方法是可重复的。您可以重新运行它以保持 AMIs 更新（例如，操作系统更新），然后在更新现有集群时使用它们。

> **Note**
>
> 如果你 AWS 在中国分区中使用这种方法，你可能会遇到网络错误。例如，当`pcluster build-image`命令从操作系统存储库 GitHub 或从操作系统存储库下载软件包时，您可能会看到这些错误。如果发生这种情况，我们建议使用以下替代方法之一：
>
> 1. 按照修改 AWS ParallelCluster AMI 方法绕过此命令。
> 2. 在另一个分区和区域（例如）中构建映像`us-east-1`，然后 store/restore 将其移动到中国区域。有关更多信息，请参阅 Amazon EC2 用户指南中的使用 S3 存储和恢复 AMI。

步骤：

1. 配置您的 AWS 账户 证书，以便 AWS ParallelCluster 客户端可以代表您调用 AWS API 操作。有关所需权限的列表，请参阅AWS Identity and Access Management 中的权限 AWS ParallelCluster。

2. 创建基本的构建映像 配置文件。为此，请指定用于构建映像以及 `ParentImage` 的 `InstanceType`。这些步骤用作创建 AMI 的起点。有关可选构建参数的更多信息，请参阅映像配置。

```
Build:
 InstanceType: <BUILD_INSTANCE_TYPE>
 ParentImage: <BASE_AMI_ID>
```

3. 使用 CLI 命令`pcluster build-image`从您作为基础提供的 AMI 开始构建 AMI。 AWS ParallelCluster

```
$ pcluster build-image --image-id IMAGE_ID --image-configuration IMAGE_CONFIG.yaml --
region REGION
    {
 "image": {
   "imageId": "IMAGE_ID",
   "imageBuildStatus": "BUILD_IN_PROGRESS",
   "cloudformationStackStatus": "CREATE_IN_PROGRESS",
   "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
IMAGE_ID/abcd1234-ef56-gh78-ij90-1234abcd5678",
   "region": "us-east-1",
   "version": "3.13.2"
 }
}
```

> **Warning**
>
> `pcluster build-image` 使用默认 VPC。如果您使用 AWS Control Tower 或 AWS 着陆区删除默认 VPC，则必须在映像配置文件中指定子网 ID。有关更多信息，请参阅 `SubnetId`。

有关其他参数的列表，请参阅 `pcluster build-image` 命令参考页面。前面命令的结果如下所述：

- 堆 CloudFormation 栈是根据映像配置创建的。该堆栈包含构建所需的所有 EC2 Image Builder 资源。
- 创建的资源包括可以向其添加自定义图像生成器 AWS ParallelCluster 组件的官方 Image Builder 组件。有关更多信息，请参阅 Image Builder 用户指南中的使用 Imag EC2 e Builder 创建自定义组件。

- EC2 Image Builder 启动构建实例、应用 AWS ParallelCluster 食谱、安装 AWS ParallelCluster 软件堆栈并执行必要的配置任务。 AWS ParallelCluster 这本食谱用于构建和引导 AWS ParallelCluster。
- 停止该实例，并在其基础上创建新的 AMI。
- 从新创建的 AMI 启动另一个实例。在测试阶段， EC2 Image Builder 会运行在 Image Builder 组件中定义的测试。
- 如果构建成功，则会删除堆栈。如果构建失败，则会保留堆栈以供检查。

4. 您可以通过运行以下命令来监控构建过程的状态。构建完成后，您可以运行该 AMI 以便检索响应中给出的 AMI ID。

```
$ pcluster describe-image --image-id IMAGE_ID --region REGION

# BEFORE COMPLETE
{
 "imageConfiguration": {
   "url": "https://parallelcluster-1234abcd5678efgh-v1-do-not-
delete.s3.amazonaws.com/parallelcluster/3.13.2/images/IMAGE_ID-abcd1234efgh5678/
configs/image-config.yaml?...",
 },
 "imageId": "IMAGE_ID",
 "imagebuilderImageStatus": "BUILDING",
 "imageBuildStatus": "BUILD_IN_PROGRESS",
 "cloudformationStackStatus": "CREATE_IN_PROGRESS",
 "cloudformationStackArn": "arn:aws:cloudformation:us-east-1:123456789012:stack/
IMAGE_ID/abcd1234-ef56-gh78-ij90-1234abcd5678",
 "region": "us-east-1",
 "version": "3.13.2",
 "cloudformationStackTags": [
   {
     "value": "3.13.2",
     "key": "parallelcluster:version"
   },
   {
     "value": "IMAGE_ID",
     "key": "parallelcluster:image_name"
   },
   ...
 ],
 "imageBuildLogsArn": "arn:aws:logs:us-east-1:123456789012:log-group:/aws/
imagebuilder/ParallelClusterImage-IMAGE_ID",
 "cloudformationStackCreationTime": "2022-04-05T21:36:26.176Z"
}
```

```
# AFTER COMPLETE
{
 "imageConfiguration": {
   "url": "https://parallelcluster-1234abcd5678efgh-v1-do-not-delete.s3.us-
east-1.amazonaws.com/parallelcluster/3.13.2/images/IMAGE_ID-abcd1234efgh5678/configs/
image-config.yaml?Signature=..."
 },
 "imageId": "IMAGE_ID",
 "imageBuildStatus": "BUILD_COMPLETE",
 "region": "us-east-1",
 "ec2AmiInfo": {
     "amiName": "IMAGE_ID 2022-04-05T21-39-24.020Z",
     "amiId": "ami-1234stuv5678wxyz",
     "description": "AWS ParallelCluster AMI for alinux2,
 kernel-4.14.238-182.422.amzn2.x86_64, lustre-2.10.8-5.amzn2.x86_64,
 efa-1.13.0-1.amzn2.x86_64, dcv-2021.1.10598-1.el7.x86_64, slurm-20-11-8-1",
     "state": "AVAILABLE",
     "tags": [
      {
        "value": "2021.3.11591-1.el7.x86_64",
        "key": "parallelcluster:dcv_version"
      },
      ...
     ],
   "architecture": "x86_64"
 },
 "version": "3.13.2"
}
```

5. 要创建集群，请在集群配置内的 `CustomAmi` 字段中输入该 AMI ID。

对 AMI 创建过程进行故障排除和监控

映像创建将在大约一个小时后完成。您可以通过运行 `pcluster describe-image` 命令或日志检索命令来监控该过程。

```
$ pcluster describe-image --image-id IMAGE_ID --region REGION
```

该`build-image`命令使用构建映像所需的所有 Amazon EC2 资源创建一个 CloudFormation 堆栈，并启动 Image Builder 进程。 EC2

运行该build-image命令后，可以使用来检索 CloudFormation 堆栈事件pcluster get-image-stack-events。您可以使用 --query 参数来筛选结果，以查看最新事件。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的筛选 AWS CLI 输出。

```
$ pcluster get-image-stack-events --image-id IMAGE_ID --region REGION --query
 "events[0]"
{
 "eventId": "ParallelClusterImage-CREATE_IN_PROGRESS-2022-04-05T21:39:24.725Z",
 "physicalResourceId": "arn:aws:imagebuilder:us-east-1:123456789012:image/
parallelclusterimage-IMAGE_ID/3.13.2/1",
 "resourceStatus": "CREATE_IN_PROGRESS",
 "resourceStatusReason": "Resource creation Initiated",
 "resourceProperties": "{\"InfrastructureConfigurationArn\":
\"arn:aws:imagebuilder:us-east-1:123456789012:infrastructure-configuration/
parallelclusterimage-abcd1234-ef56-gh78-ij90-1234abcd5678\",\"ImageRecipeArn\":
\"arn:aws:imagebuilder:us-east-1:123456789012:image-recipe/parallelclusterimage-
IMAGE_ID/3.13.2\",\"DistributionConfigurationArn\":\"arn:aws:imagebuilder:us-
east-1:123456789012:distribution-configuration/parallelclusterimage-abcd1234-ef56-
gh78-ij90-1234abcd5678\",\"Tags\":{\"parallelcluster:image_name\":\"IMAGE_ID\",
\"parallelcluster:image_id\":\"IMAGE_ID\"}}",
 "stackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/IMAGE_ID/abcd1234-
ef56-gh78-ij90-1234abcd5678",
 "stackName": "IMAGE_ID",
 "logicalResourceId": "ParallelClusterImage",
 "resourceType": "AWS::ImageBuilder::Image",
 "timestamp": "2022-04-05T21:39:24.725Z"
}
```

大约 15 分钟后，堆栈事件将出现在与 Image Builder 创建相关的日志事件条目中。您现在可以使用 pcluster list-image-log-streams 和 pcluster get-image-log-events 命令列出映像日志流并监控 Image Builder 步骤。

```
$ pcluster list-image-log-streams --image-id IMAGE_ID --region REGION \
    --query 'logStreams[*].logStreamName'

 "3.13.2/1"
]

$ pcluster get-image-log-events --image-id IMAGE_ID --region REGION \
--log-stream-name 3.13.2/1 --limit 3
{
 "nextToken": "f/36295977202298886557255241372854078762600452615936671762",
```

```
 "prevToken": "b/36295977196879805474012299949460899223469007699834306 72",
 "events": [
   {
     "message": "ExecuteBash: FINISHED EXECUTION",
     "timestamp": "2022-04-05T22:13:26.633Z"
   },
   {
     "message": "Document arn:aws:imagebuilder:us-east-1:123456789012:component/
parallelclusterimage-test-abcd1234-ef56-gh78-ij90-1234abcd5678/3.13.2/1",
     "timestamp": "2022-04-05T22:13:26.741Z"
   },
   {
     "message": "TOE has completed execution successfully",
     "timestamp": "2022-04-05T22:13:26.819Z"
   }
 ]
}
```

继续使用 describe-image 命令进行检查，直到看到 BUILD_COMPLETE 状态。

```
$ pcluster describe-image --image-id IMAGE_ID --region REGION
{
 "imageConfiguration": {
   "url": "https://parallelcluster-1234abcd5678efgh-v1-do-not-delete.s3.us-
east-1.amazonaws.com/parallelcluster/3.13.2/images/IMAGE_ID-abcd1234efgh5678/configs/
image-config.yaml?Signature=..."
 },
 "imageId": "IMAGE_ID",
 "imageBuildStatus": "BUILD_COMPLETE",
 "region": "us-east-1",
 "ec2AmiInfo": {
     "amiName": "IMAGE_ID 2022-04-05T21-39-24.020Z",
     "amiId": "ami-1234stuv5678wxyz",
     "description": "AWS ParallelCluster AMI for alinux2,
 kernel-4.14.238-182.422.amzn2.x86_64, lustre-2.10.8-5.amzn2.x86_64,
 efa-1.13.0-1.amzn2.x86_64, dcv-2021.1.10598-1.el7.x86_64, slurm-20-11-8-1",
     "state": "AVAILABLE",
     "tags": [
      {
        "value": "2021.3.11591-1.el7.x86_64",
        "key": "parallelcluster:dcv_version"
      },
      ...
```

```
      ],
    "architecture": "x86_64"
  },
  "version": "3.13.2"
}
```

如果您需要排查自定义 AMI 创建问题，请按照以下步骤所述创建映像日志存档。

可以将日志存档到 Amazon S3 存储桶或本地文件中，具体取决于 `--output` 参数。

```
$ pcluster export-image-logs --image-id IMAGE_ID --region REGION \
--bucket BUCKET_NAME --bucket-prefix BUCKET_FOLDER
{
 "url": "https://BUCKET_NAME.s3.us-east-1.amazonaws.com/BUCKET-FOLDER/IMAGE_ID-
logs-202209071136.tar.gz?AWSAccessKeyId=..."
}

$ pcluster export-image-logs --image-id IMAGE_ID \
--region REGION --bucket BUCKET_NAME --bucket-prefix BUCKET_FOLDER --output-file /tmp/
archive.tar.gz
{
 "path": "/tmp/archive.tar.gz"
}
```

该档案包含与 Image Builder 进程和 AWS CloudFormation 堆栈事件相关的 CloudWatch 日志流。该命令的运行可能需要几分钟才能完成。

管理自定义 AMIs

从 AWS ParallelCluster 3.0.0 开始，CLI 中添加了一组用于构建、监控和管理映像生命周期的新命令。有关这些命令的更多信息，请参阅 pcluster 命令。

修改 AWS ParallelCluster AMI

此方法包括通过在官方 AWS ParallelCluster AMI 上添加自定义来修改官方 AMI。基础版 AWS ParallelCluster AMIs 已更新为新版本。它们 AMIs 具有安装和配置后运行所需 AWS ParallelCluster 的所有组件。您可以以其中一个 AMI 作为起点。

关键点：

- 此方法比 `build-image` 命令快。但它是一个手动过程，不会自动重复。

- 使用此方法，您无权使用通过 CLI 提供的日志检索和映像生命周期管理命令。

步骤：

New Amazon EC2 console

1. 找到与您使用的具体内容相对应 AWS 区域 的 AMI。要找到它，请使用带有`--region`参数的`pcluster list-official-images`命令来选择特定的 AWS 区域 和`--os`和`--architecture`参数，筛选出具有您要使用的操作系统和架构的所需 AMI。从输出中检索 Amazon EC2 图片编号。
2. 登录 AWS Management Console 并打开 Amazon EC2 控制台，网址为https://console.aws.amazon.com/ec2/。
3. 在导航窗格中，选择 “图像”，然后AMIs。搜索检索到的 EC2映像 ID，选择 AMI，然后从 AMI 中选择启动实例。
4. 向下滚动并选择您的实例类型。
5. 选择您的密钥对，然后选择启动实例。
6. 使用操作系统用户和您的 SSH 密钥登录您的实例。
7. 手动自定义实例以满足您的要求。
8. 运行以下命令以准备实例来创建 AMI。

   ```
   sudo /usr/local/sbin/ami_cleanup.sh
   ```

9. 在控制台中，选择实例状态和停止实例。

   导航到实例，选择新实例，然后依次选择实例状态和停止实例。

10. 使用 Amazon EC2 控制台或创建镜像从实例 AWS CLI 创建新 AMI。

    从亚马逊 EC2 控制台

    a. 在导航窗格中选择实例。

    b. 选择您创建和修改的实例。

    c. 在操作中，依次选择映像和创建映像。

    d. 选择创建映像。

11. 在集群配置内的 `CustomAmi` 字段中输入新 AMI ID，并创建集群。

Old Amazon EC2 console

1. 找到与您使用的具体内容相对应 AWS 区域 的 AWS ParallelCluster AMI。要找到它，你可以使用带有`--region`参数的`pcluster list-official-images`命令来选择特定的 AWS 区域和`--os`和`--architecture`参数，然后根据你要使用的操作系统和架构筛选出所需的 AMI。您可以从输出中检索 Amazon EC2 图片编号。
2. 登录 AWS Management Console 并打开 Amazon EC2 控制台，网址为https://console.aws.amazon.com/ec2/。
3. 在导航窗格中，选择 “图像”，然后AMIs。为公共图像设置过滤器并搜索检索到的 EC2映像 ID，选择 AMI，然后选择 Launch。
4. 选择您的实例类型，然后选择下一步：配置实例详细信息或查看并启动以启动您的实例。
5. 选择启动，选择您的密钥对，然后选择启动实例。
6. 使用操作系统用户和您的 SSH 密钥登录您的实例。有关更多信息，请导航至实例，选择新实例，然后选择连接。
7. 手动自定义实例以满足您的要求。
8. 运行以下命令以准备实例来创建 AMI：

   ```
   sudo /usr/local/sbin/ami_cleanup.sh
   ```

9. 在 Amazon EC2 控制台中，在导航窗格中选择实例，选择您的新实例，然后选择操作、实例状态和停止。
10. 使用 Amazon EC2 控制台或创建镜像从实例 AWS CLI 创建新 AMI。

    从亚马逊 EC2 控制台

    a. 在导航窗格中选择实例。

    b. 选择您创建和修改的实例。

    c. 在操作中，依次选择映像和创建映像。

    d. 选择创建映像。
11. 在集群配置内的 `CustomAmi` 字段中输入新 AMI ID，并创建集群。

集成 Active Directory

在本教程中，您将创建一个多用户环境。此环境包括与 AWS Managed Microsoft AD （A AWS ParallelCluster ctive Directory）集成的环境`corp.example.com`。您将配置一个 `Admin` 用户来管理

目录，一个 `ReadOnly` 用户来读取目录，以及一个 `user000` 用户来登录到集群。您可以使用自动路径或手动路径来创建网络资源、活动目录 (AD) 和用于配置 AD 的 Amazon EC2 实例。无论采用何种路径，您创建的基础架构都已预先配置为 AWS ParallelCluster 使用以下方法之一进行集成：

- 具有证书验证功能的 LDAPS（最安全的选项，建议使用）
- 没有证书验证功能的 LDAPS
- LDAP

LDAP 本身不 提供加密。为确保安全传输潜在敏感信息，我们强烈建议您在与集成的集群中使用 LDAPS（基于 TLS/SSL 的 LDAP）。 ADs有关更多信息，请参阅《 AWS Directory Service 管理指南》 AWS Managed Microsoft AD中的使用启用服务器端 LDAPS。

创建这些资源后，继续配置和创建与 Active Directory (AD) 集成的集群。创建集群之后，以您创建的用户身份登录。有关在本教程中创建的配置的更多信息，请参阅集群的多用户访问和 `DirectoryService` 配置部分。

本教程介绍如何创建支持多用户访问集群的环境。本教程不介绍如何创建和使用 AWS Directory Service 广告。本教程 AWS Managed Microsoft AD 中提供的设置步骤仅用于测试目的。它们不能 取代 AWS Directory Service Administration Guide 的 AWS Managed Microsoft AD 和 Simple AD 章节中所述的官方文档和最佳实践。

> **Note**
>
> 目录用户密码根据目录密码策略属性定义过期。要使用重置目录密码 AWS ParallelCluster，请参阅如何重置用户密码和过期的密码。

> **Note**
>
> 目录域控制器 IP 地址可能会因域控制器更改和目录维护而更改。如果您选择自动快速创建方法来创建目录基础架构，则当目录 IP 地址更改时，必须手动使目录控制器前面的负载均衡器保持一致。使用快速创建方法时，目录 IP 地址不会自动与负载均衡器保持一致。

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

先决条件

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 你有一个 A mazon EC2 密钥 pair。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

在学习本教程时，请将诸如*region-id*和*d-abcdef01234567890*之类的名称替换*inputs highlighted in red*为自己的名字和 IDs。*0123456789012*用你的 AWS 账户 电话号码替换。

创建 AD 基础设施

选择 “自动” 选项卡，使用 AWS CloudFormation 快速创建模板创建 Active Directory (AD) 基础架构。

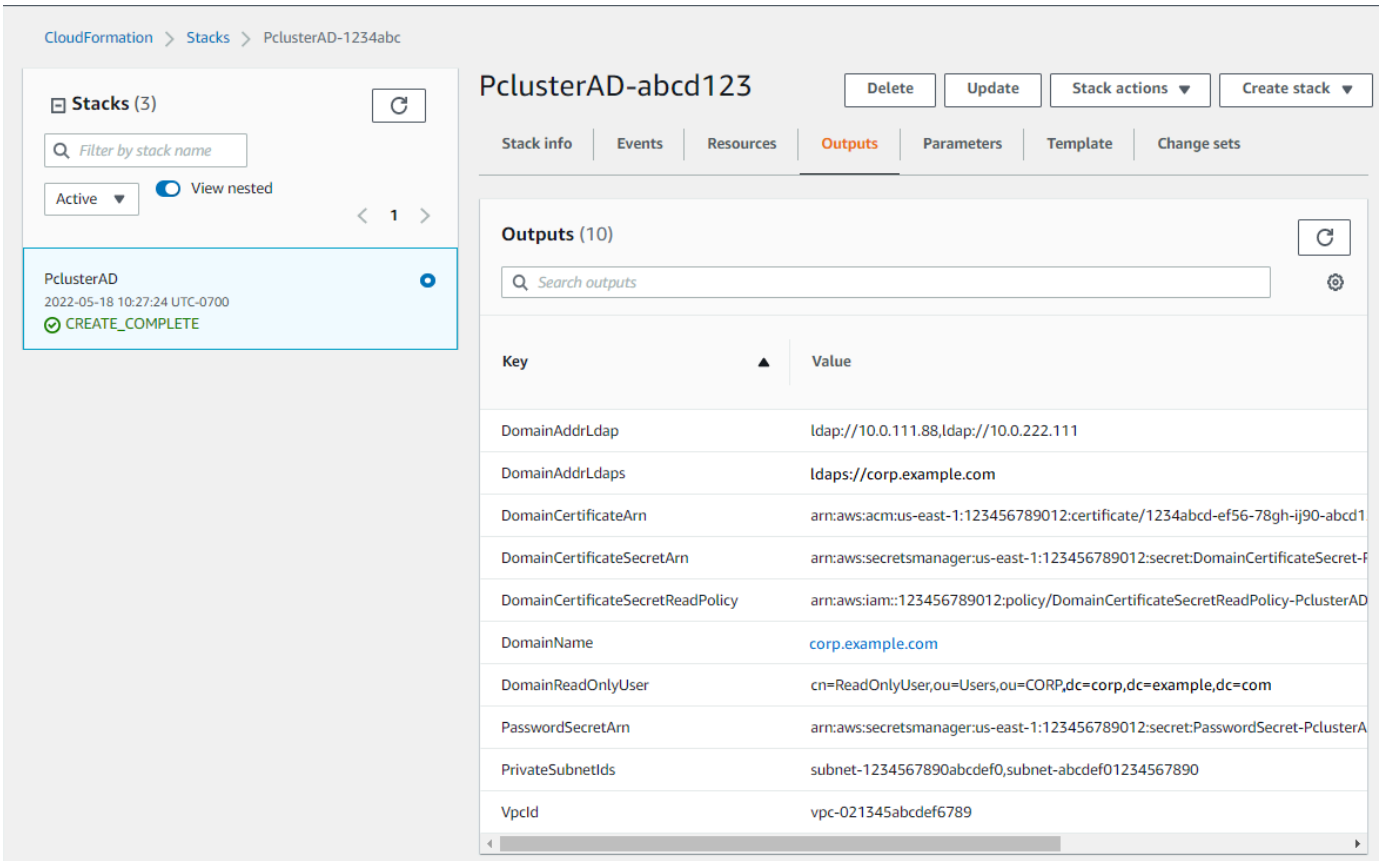
选择手动选项卡可手动创建 AD 基础架构。

自动

1. 登录到 AWS Management Console。
2. 打开CloudFormation 快速创建（区域 us-east-1），在控制台中创建以下资源： CloudFormation
   - 具有两个子网的 VPC 和公有访问路由（如果未指定 VPC）。
   - 一个 AWS Managed Microsoft AD。
   - 已加入 AD 的 Amazon EC2 实例，可用于管理目录。
3. 在快速创建堆栈页面的参数部分，输入以下参数的密码：
   - AdminPassword
   - ReadOnlyPassword
   - UserPassword

   记下这些密码。本教程后面将会用到这些密码。
4. 对于 DomainName，输入 **`corp.example.com`**。
5. 对于 Keypair，请输入亚马逊 EC2 密钥对的名称。
6. 在页面底部选中各个框以确认各项访问功能。
7. 选择创建堆栈。

8. CloudFormation 堆栈达到CREATE_COMPLETE状态后，选择堆栈的 “输出” 选项卡。记下输出资源名称， IDs 因为您需要在以后的步骤中使用它们。输出提供了创建集群所需的信息。



9. 要完成练习（可选）管理 AD 用户和组，您需要目录 ID。选择资源并向下滚动，记下目录 ID。
10. 继续（可选）管理 AD 用户和组或创建集群。

手动

为目录服务创建在不同可用区中具有两个子网的 VPC 以及 AWS Managed Microsoft AD。

创建 AD

> **Note**
>
> - 目录和域名是 `corp.example.com`。短名称是 `CORP`。
> - 在脚本中更改 `Admin` 密码。
> - 创建 Active Directory (AD) 至少需要 15 分钟。

使用以下 Python 脚本在您的本地 AWS 区域创建 VPC、子网和 AD 资源。将此文件另存为 `ad.py` 并运行。

```
import boto3
import time
from pprint import pprint

vpc_name = "PclusterVPC"
ad_domain = "corp.example.com"
admin_password = "asdfASDF1234"

ec2 = boto3.client("ec2")
ds = boto3.client("ds")
region = boto3.Session().region_name

# Create the VPC, Subnets, IGW, Routes
vpc = ec2.create_vpc(CidrBlock="10.0.0.0/16")["Vpc"]
vpc_id = vpc["VpcId"]
time.sleep(30)
ec2.create_tags(Resources=[vpc_id], Tags=[{"Key": "Name", "Value": vpc_name}])
subnet1 = ec2.create_subnet(VpcId=vpc_id, CidrBlock="10.0.0.0/17",
 AvailabilityZone=f"{region}a")["Subnet"]
subnet1_id = subnet1["SubnetId"]
time.sleep(30)
ec2.create_tags(Resources=[subnet1_id], Tags=[{"Key": "Name", "Value": f"{vpc_name}/
subnet1"}])
ec2.modify_subnet_attribute(SubnetId=subnet1_id, MapPublicIpOnLaunch={"Value": True})
subnet2 = ec2.create_subnet(VpcId=vpc_id, CidrBlock="10.0.128.0/17",
 AvailabilityZone=f"{region}b")["Subnet"]
subnet2_id = subnet2["SubnetId"]
time.sleep(30)
ec2.create_tags(Resources=[subnet2_id], Tags=[{"Key": "Name", "Value": f"{vpc_name}/
subnet2"}])
ec2.modify_subnet_attribute(SubnetId=subnet2_id, MapPublicIpOnLaunch={"Value": True})
igw = ec2.create_internet_gateway()["InternetGateway"]
ec2.attach_internet_gateway(InternetGatewayId=igw["InternetGatewayId"], VpcId=vpc_id)
route_table = ec2.describe_route_tables(Filters=[{"Name": "vpc-id", "Values":
 [vpc_id]}])["RouteTables"][0]
ec2.create_route(RouteTableId=route_table["RouteTableId"],
 DestinationCidrBlock="0.0.0.0/0", GatewayId=igw["InternetGatewayId"])
ec2.modify_vpc_attribute(VpcId=vpc_id, EnableDnsSupport={"Value": True})
ec2.modify_vpc_attribute(VpcId=vpc_id, EnableDnsHostnames={"Value": True})
```

```
# Create the Active Directory
ad = ds.create_microsoft_ad(
    Name=ad_domain,
    Password=admin_password,
    Description="ParallelCluster AD",
    VpcSettings={"VpcId": vpc_id, "SubnetIds": [subnet1_id, subnet2_id]},
    Edition="Standard",
)
directory_id = ad["DirectoryId"]

# Wait for completion
print("Waiting for the directory to be created...")
directories = ds.describe_directories(DirectoryIds=[directory_id])
["DirectoryDescriptions"]
directory = directories[0]
while directory["Stage"] in {"Requested", "Creating"}:
    time.sleep(3)
    directories = ds.describe_directories(DirectoryIds=[directory_id])
["DirectoryDescriptions"]
    directory = directories[0]

dns_ip_addrs = directory["DnsIpAddrs"]

pprint({"directory_id": directory_id,
        "vpc_id": vpc_id,
        "subnet1_id": subnet1_id,
        "subnet2_id": subnet2_id,
        "dns_ip_addrs": dns_ip_addrs})
```

下面是该 Python 脚本的示例输出。

```
{
  "directory_id": "d-abcdef01234567890",
  "dns_ip_addrs": ["192.0.2.254", "203.0.113.237"],
  "subnet1_id": "subnet-021345abcdef6789",
  "subnet2_id": "subnet-1234567890abcdef0",
  "vpc_id": "vpc-021345abcdef6789"
}
```

记下输出资源名称和 IDs。您将在后面的步骤中用到它们。

脚本完成后，继续执行下一步。

创建一个 Amazon EC2 实例

New Amazon EC2 console

1. 登录到 AWS Management Console。
2. 如果您没有附加步骤 4 中列出的策略的角色，请打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。否则，请跳至步骤 5。
3. 创建`ResetUserPassword`政策，将红色突出显示的内容替换为您的 AWS 区域 ID、账户 ID 和您为创建 AD 而运行的脚本输出中的目录 ID。

   ResetUserPassword

```
{
       "Statement": [
        {
            "Action": [
                "ds:ResetUserPassword"
            ],
            "Resource": "arn:aws:ds:region-id:123456789012:directory/d-abcdef01234567890",
            "Effect": "Allow"
        }
    ]
}
```

4. 创建附加了以下策略的 IAM 角色。
   - AWS 托管策略 Amazon SSMManaged InstanceCore
   - AWS 托管策略 Amazon SSMDirectory ServiceAccess
   - ResetUserPassword 政策
5. 打开 Amazon EC2 控制台，网址为https://console.aws.amazon.com/ec2/。
6. 在 Amazon EC2 控制面板中，选择启动实例。
7. 在应用程序和操作系统映像中，选择最近的 Amazon Linux 2 AMI。
8. 对于实例类型，选择 t2.micro。
9. 对于密钥对，选择一个密钥对。
10. 对于网络设置，选择编辑。
11. 对于 VPC，选择目录 VPC。
12. 向下滚动并选择高级详细信息。

13. 在高级详细信息中的域加入目录中，选择 **corp.example.com**。
14. 对于 IAM 实例配置文件，选择您在步骤 1 中创建的角色或附加了步骤 4 中所列策略的角色。
15. 在摘要中，选择启动实例。
16. 记下实例 ID（例如 i-1234567890abcdef0），然后等待实例完成启动。
17. 在实例启动后，继续进行下一步操作。

Old Amazon EC2 console

1. 登录到 AWS Management Console。
2. 如果您没有附加步骤 4 中列出的策略的角色，请打开 IAM 控制台，网址为https://console.aws.amazon.com/iam/。否则，请跳至步骤 5。
3. 创建 `ResetUserPassword` 策略。将红色突出显示的内容替换为您的 AWS 区域 ID、 AWS 账户 ID 和您为创建 Active Directory (AD) 而运行的脚本输出中的目录 ID。

   ResetUserPassword

```
{
        "Statement": [
            {
                "Action": [
                    "ds:ResetUserPassword"
                ],
                "Resource": "arn:aws:ds:region-id:123456789012:directory/d-abcdef01234567890",
                "Effect": "Allow"
            }
        ]
    }
```

4. 创建附加了以下策略的 IAM 角色。
   - AWS 托管策略 Amazon SSMManaged InstanceCore
   - AWS 托管策略 Amazon SSMDirectory ServiceAccess
   - ResetUserPassword 策略
5. 打开 Amazon EC2 控制台，网址为https://console.aws.amazon.com/ec2/。
6. 在 Amazon EC2 控制面板中，选择启动实例。
7. 在应用程序和操作系统映像中，选择最近的 Amazon Linux 2 AMI。

8. 对于实例类型，选择 t2.micro。
9. 对于密钥对，选择一个密钥对。
10. 在网络设置中，选择编辑。
11. 对于网络设置中的 VPC，选择目录 VPC。
12. 向下滚动并选择高级详细信息。
13. 在高级详细信息中的域加入目录中，选择 **corp.example.com**。
14. 对于高级详细信息中的实例配置文件，选择您在步骤 1 中创建的角色或附加了步骤 4 中所列策略的角色。
15. 在摘要中，选择启动实例。
16. 记下实例 ID（例如 i-1234567890abcdef0），然后等待实例完成启动。
17. 在实例启动后，继续进行下一步操作。

将您的实例加入到 AD

1. 以 **admin** 身份连接到您的实例并加入 AD 领域。

   运行以下命令连接到实例。

```
$ INSTANCE_ID="i-1234567890abcdef0"
```

```
$ PUBLIC_IP=$(aws ec2 describe-instances \
--instance-ids $INSTANCE_ID \
--query "Reservations[0].Instances[0].PublicIpAddress" \
--output text)
```

```
$ ssh -i ~/.ssh/keys/keypair.pem ec2-user@$PUBLIC_IP
```

2. 安装必要的软件并加入该领域。

```
$ sudo yum -y install sssd realmd oddjob oddjob-mkhomedir adcli samba-common samba-
common-tools krb5-workstation openldap-clients policycoreutils-python
```

3. 将管理员密码替换为您的 **admin** 密码。

```
$ ADMIN_PW="asdfASDF1234"
```

```
$ echo $ADMIN_PW | sudo realm join -U Admin corp.example.com
Password for Admin:
```

如果上述操作成功，您就加入到了该领域，并可以继续下一步操作。

向 AD 中添加用户

1. 创建 ReadOnlyUser 和其他用户。

   在此步骤中，您将使用在前一步中安装的 adcli 和 openldap-clients 工具。

```
$ echo $ADMIN_PW | adcli create-user -x -U Admin --domain=corp.example.com --
display-name=ReadOnlyUser ReadOnlyUser
```

```
$ echo $ADMIN_PW | adcli create-user -x -U Admin --domain=corp.example.com --
display-name=user000 user000
```

2. 验证是否创建了用户：

   目录 DNS IP 地址是 Python 脚本的输出。

```
$ DIRECTORY_IP="192.0.2.254"
```

```
$ ldapsearch -x -h $DIRECTORY_IP -D Admin -w $ADMIN_PW -b
 "cn=ReadOnlyUser,ou=Users,ou=CORP,dc=corp,dc=example,dc=com"
```

```
$ ldapsearch -x -h $DIRECTORY_IP -D Admin -w $ADMIN_PW -b
 "cn=user000,ou=Users,ou=CORP,dc=corp,dc=example,dc=com"
```

   默认情况下，当您使用 `ad-cli` 创建用户时，该用户将处于禁用状态。

3. 在本地计算机上重置并激活用户密码：

   注销您的 Amazon EC2 实例。

> **Note**
>
> - `ro-p@ssw0rd`是的密码`ReadOnlyUser`，从中检索 AWS Secrets Manager。
> - `user-p@ssw0rd` 是集群用户的密码，在您连接 (`ssh`) 到集群时提供。

`directory-id` 是 Python 脚本的输出。

```
$ DIRECTORY_ID="d-abcdef01234567890"
```

```
$ aws ds reset-user-password \
--directory-id $DIRECTORY_ID \
--user-name "ReadOnlyUser" \
--new-password "ro-p@ssw0rd" \
--region "region-id"
```

```
$ aws ds reset-user-password \
--directory-id $DIRECTORY_ID \
--user-name "user000" \
--new-password "user-p@ssw0rd" \
--region "region-id"
```

4. 将密码添加到 Secrets Manager 密钥中。

   现在，您已经创建`ReadOnlyUser`并设置了密码，请将其存储在 AWS ParallelCluster 用于验证登录名的密钥中。

   使用 Secrets Manager 创建新密钥以将 `ReadOnlyUser` 的密码作为值。密钥值格式必须仅为纯文本（而不是 JSON 格式）。记下密钥的 ARN，以在后面的步骤中使用。

```
$ aws secretsmanager create-secret --name "ADSecretPassword" \
--region region_id \
--secret-string "ro-p@ssw0rd" \
--query ARN \
--output text
arn:aws:secretsmanager:region-id:123456789012:secret:ADSecretPassword-1234
```

具有证书验证功能的 LDAPS（推荐）的设置

记下资源 IDs。您将在后面的步骤中用到它们。

1. 在本地生成域证书。

```
$ PRIVATE_KEY="corp-example-com.key"
CERTIFICATE="corp-example-com.crt"
printf ".\n.\n.\n.\n.\ncorp.example.com\n.\n" | openssl req -x509 -sha256 -nodes -
newkey rsa:2048 -keyout $PRIVATE_KEY -days 365 -out $CERTIFICATE
```

2. 将证书存储到 Secrets Manager，以便以后可以从集群内进行检索。

```
$ aws secretsmanager create-secret --name example-cert \
  --secret-string file://$CERTIFICATE \
  --region region-id
{
  "ARN": "arn:aws:secretsmanager:region-id:123456789012:secret:example-
cert-123abc",
  "Name": "example-cert",
  "VersionId": "14866070-092a-4d5a-bcdd-9219d0566b9c"
}
```

3. 将以下策略添加到您为将 Amazon EC2 实例加入 AD 域而创建的 IAM 角色中。

   `PutDomainCertificateSecrets`

```
{
    "Statement": [
        {
            "Action": [
                "secretsmanager:PutSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:region-id:123456789012:secret:example-
cert-123abc"
            ],
            "Effect": "Allow"
        }
    ]
}
```

4. 将证书导入到 AWS Certificate Manager (ACM)。

```
$ aws acm import-certificate --certificate fileb://$CERTIFICATE \
  --private-key fileb://$PRIVATE_KEY \
  --region region-id
{
  "CertificateArn": "arn:aws:acm:region-
id:123456789012:certificate/343db133-490f-4077-b8d4-3da5bfd89e72"
}
```

5. 创建放置在 Active Directory 端点前面的负载均衡器。

```
$ aws elbv2 create-load-balancer --name CorpExampleCom-NLB \
  --type network \
  --scheme internal \
  --subnets subnet-1234567890abcdef0 subnet-021345abcdef6789 \
  --region region-id
{
  "LoadBalancers": [
    {
      "LoadBalancerArn": "arn:aws:elasticloadbalancing:region-
id:123456789012:loadbalancer/net/CorpExampleCom-NLB/3afe296bf4ba80d4",
      "DNSName": "CorpExampleCom-NLB-3afe296bf4ba80d4.elb.region-id.amazonaws.com",
      "CanonicalHostedZoneId": "Z2IFOLAFXWLO4F",
      "CreatedTime": "2022-05-05T12:56:55.988000+00:00",
      "LoadBalancerName": "CorpExampleCom-NLB",
      "Scheme": "internal",
      "VpcId": "vpc-021345abcdef6789",
      "State": {
        "Code": "provisioning"
       },
       "Type": "network",
       "AvailabilityZones": [
         {
           "ZoneName": "region-idb",
           "SubnetId": "subnet-021345abcdef6789",
           "LoadBalancerAddresses": []
         },
         {
           "ZoneName": "region-ida",
           "SubnetId": "subnet-1234567890abcdef0",
           "LoadBalancerAddresses": []
         }
```

```
        ],
        "IpAddressType": "ipv4"
    }
  ]
}
```

6. 创建以 Active Directory 端点为目标的目标组。

```
$ aws elbv2 create-target-group --name CorpExampleCom-Targets --protocol TCP \
  --port 389 \
  --target-type ip \
  --vpc-id vpc-021345abcdef6789 \
  --region region-id
{
  "TargetGroups": [
    {
      "TargetGroupArn": "arn:aws:elasticloadbalancing:region-
id:123456789012:targetgroup/CorpExampleCom-Targets/44577c583b695e81",
      "TargetGroupName": "CorpExampleCom-Targets",
      "Protocol": "TCP",
      "Port": 389,
      "VpcId": "vpc-021345abcdef6789",
      "HealthCheckProtocol": "TCP",
      "HealthCheckPort": "traffic-port",
      "HealthCheckEnabled": true,
      "HealthCheckIntervalSeconds": 30,
      "HealthCheckTimeoutSeconds": 10,
      "HealthyThresholdCount": 3,
      "UnhealthyThresholdCount": 3,
      "TargetType": "ip",
      "IpAddressType": "ipv4"
    }
  ]
}
```

7. 将 Active Directory (AD) 端点注册到目标组。

```
$ aws elbv2 register-targets --target-group-arn
 arn:aws:elasticloadbalancing:region-id:123456789012:targetgroup/CorpExampleCom-
Targets/44577c583b695e81 \
  --targets Id=192.0.2.254,Port=389 Id=203.0.113.237,Port=389 \
  --region region-id
```

8. 对证书创建 LB 侦听器。

```
$ aws elbv2 create-listener --load-balancer-arn
 arn:aws:elasticloadbalancing:region-id:123456789012:loadbalancer/net/
CorpExampleCom-NLB/3afe296bf4ba80d4 \
  --protocol TLS \
  --port 636 \
  --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region-
id:123456789012:targetgroup/CorpExampleCom-Targets/44577c583b695e81 \
  --ssl-policy ELBSecurityPolicy-TLS-1-2-2017-01 \
  --certificates CertificateArn=arn:aws:acm:region-
id:123456789012:certificate/343db133-490f-4077-b8d4-3da5bfd89e72 \
  --region region-id
  "Listeners": [
  {
    "ListenerArn": "arn:aws:elasticloadbalancing:region-id:123456789012:listener/
net/CorpExampleCom-NLB/3afe296bf4ba80d4/a8f9d97318743d4b",
    "LoadBalancerArn": "arn:aws:elasticloadbalancing:region-
id:123456789012:loadbalancer/net/CorpExampleCom-NLB/3afe296bf4ba80d4",
    "Port": 636,
    "Protocol": "TLS",
    "Certificates": [
      {
        "CertificateArn": "arn:aws:acm:region-
id:123456789012:certificate/343db133-490f-4077-b8d4-3da5bfd89e72"
       }
     ],
     "SslPolicy": "ELBSecurityPolicy-TLS-1-2-2017-01",
     "DefaultActions": [
       {
         "Type": "forward",
         "TargetGroupArn": "arn:aws:elasticloadbalancing:region-
id:123456789012:targetgroup/CorpExampleCom-Targets/44577c583b695e81",
         "ForwardConfig": {
           "TargetGroups": [
             {
                "TargetGroupArn": "arn:aws:elasticloadbalancing:region-
id:123456789012:targetgroup/CorpExampleCom-Targets/44577c583b695e81"
              }
            ]
          }
        }
```

```
      ]
    }
  ]
}
```

9. 创建托管区以便可以在集群 VPC 内发现该域。

```
$ aws route53 create-hosted-zone --name corp.example.com \
  --vpc VPCRegion=region-id,VPCId=vpc-021345abcdef6789 \
  --caller-reference "ParallelCluster AD Tutorial"
{
  "Location": "https://route53.amazonaws.com/2013-04-01/hostedzone/
Z09020002B5MZQNXMSJUB",
  "HostedZone": {
    "Id": "/hostedzone/Z09020002B5MZQNXMSJUB",
    "Name": "corp.example.com.",
    "CallerReference": "ParallelCluster AD Tutorial",
    "Config": {
         "PrivateZone": true
    },
    "ResourceRecordSetCount": 2
  },
  "ChangeInfo": {
    "Id": "/change/C05533343BF3IKSORW1TQ",
    "Status": "PENDING",
    "SubmittedAt": "2022-05-05T13:21:53.863000+00:00"
  },
  "VPC": {
    "VPCRegion": "region-id",
    "VPCId": "vpc-021345abcdef6789"
  }
}
```

10. 创建名为 **`recordset-change.json`** 并包含以下内容的文件。**`HostedZoneId`** 是负载均衡器的规范托管区 ID。

```
{
  "Changes": [
    {
      "Action": "CREATE",
      "ResourceRecordSet": {
        "Name": "corp.example.com",
        "Type": "A",
```

```
        "Region": "region-id",
        "SetIdentifier": "example-active-directory",
        "AliasTarget": {
          "HostedZoneId": "Z2IFOLAFXWLO4F",
          "DNSName": "CorpExampleCom-NLB-3afe296bf4ba80d4.elb.region-
id.amazonaws.com",
          "EvaluateTargetHealth": true
        }
      }
    }
  ]
}
```

11. 将记录集更改提交到托管区，这次使用托管区 ID。

```
$ aws route53 change-resource-record-sets --hosted-zone-id Z09020002B5MZQNXMSJUB \
  --change-batch file://recordset-change.json
{
  "ChangeInfo": {
    "Id": "/change/C0137926I56R3GC7XW2Y",
    "Status": "PENDING",
    "SubmittedAt": "2022-05-05T13:40:36.553000+00:00"
  }
}
```

12. 创建包含以下内容的策略文档 **policy.json**。

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:region-id:123456789012:secret:example-cert-abc123"
      ],
      "Effect": "Allow"
    }
  ]
}
```

13. 创建名为 **policy.json** 并包含以下内容的策略文档。

```
$ aws iam create-policy --policy-name ReadCertExample \
  --policy-document file://policy.json
{
  "Policy": {
    "PolicyName": "ReadCertExample",
    "PolicyId": "ANPAUUXUVBC42VZSI4LDY",
    "Arn": "arn:aws:iam::123456789012:policy/ReadCertExample-efg456",
    "Path": "/",
    "DefaultVersionId": "v1",
    "AttachmentCount": 0,
    "PermissionsBoundaryUsageCount": 0,
    "IsAttachable": true,
    "CreateDate": "2022-05-05T13:42:18+00:00",
    "UpdateDate": "2022-05-05T13:42:18+00:00"
  }
}
```

14. 继续按照 （可选）管理 AD 用户和组 或 创建集群 中的步骤操作。

（可选）管理 AD 用户和组

在此步骤中，您将管理已加入有效交付 (AD) 域的 EC2 Amazon Linux 2 实例中的用户和群组。

如果您使用的是自动 方法，请重启并登录到在自动操作过程中创建并加入 AD 的实例。

如果您使用的是手动 方法，请重启并登录到您在前面的步骤中创建并加入 AD 的实例。

在这些步骤中，您将使用前一步在实例中安装的 adcli 和 openldap-clients 工具。

登录已加入 AD 域的 Amazon EC2 实例

1. 在亚马逊 EC2 控制台中，选择在之前的步骤中创建的无标题的亚马逊 EC2 实例。该实例的状态可能是已停止。
2. 如果实例状态为已停止，请选择实例状态，然后选择启动实例。
3. 状态检查通过后，选择该实例，然后选择连接并 SSH 登录到该实例。

登录已加入 AD 的 Amazon A EC2 mazon Linux 2 实例时管理用户和群组

当您使用 `-U "Admin"` 选项运行 `adcli` 命令时，系统会提示您输入 AD `Admin` 密码。您可以将 AD `Admin` 密码作为 `ldapsearch` 命令的一部分。

1. 创建用户。

```
$ adcli create-user "clusteruser" --domain "corp.example.com" -U "Admin"
```

2. 设置用户密码。

```
$ aws --region "region-id" ds reset-user-password --directory-id "d-
abcdef01234567890" --user-name "clusteruser" --new-password "new-p@ssw0rd"
```

3. 创建组。

```
$ adcli create-group "clusterteam" --domain "corp.example.com" -U "Admin"
```

4. 将用户添加到组。

```
$ adcli add-member "clusterteam" "clusteruser" --domain "corp.example.com" -U
 "Admin"
```

5. 描述用户和组。

   描述所有用户。

```
$ ldapsearch "(&(objectClass=user))" -x -h "192.0.2.254" -b
 "DC=corp,DC=example,DC=com" -D
 "CN=Admin,OU=Users,OU=CORP,DC=corp,DC=example,DC=com" -w "p@ssw0rd"
```

   描述特定用户。

```
$ ldapsearch "(&(objectClass=user)(cn=clusteruser))"
 -x -h "192.0.2.254" -b "DC=corp,DC=example,DC=com" -D
 "CN=Admin,OU=Users,OU=CORP,DC=corp,DC=example,DC=com" -w "p@ssw0rd"
```

   使用名称模式描述所有用户。

```
$ ldapsearch "(&(objectClass=user)(cn=user*))" -x -h "192.0.2.254" -b
 "DC=corp,DC=example,DC=com" -D
 "CN=Admin,OU=Users,OU=CORP,DC=corp,DC=example,DC=com" -w "p@ssw0rd"
```

描述属于特定组的所有用户。

```
$ ldapsearch "(&(objectClass=user)
(memberOf=CN=clusterteam,OU=Users,OU=CORP,DC=corp,DC=example,DC=com))"
 -x -h "192.0.2.254" -b "DC=corp,DC=example,DC=com" -D
 "CN=Admin,OU=Users,OU=CORP,DC=corp,DC=example,DC=com" -w "p@ssw0rd"
```

描述所有组

```
$ ldapsearch "objectClass=group" -x -h "192.0.2.254" -b "DC=corp,DC=example,DC=com"
 -D "CN=Admin,OU=Users,OU=CORP,DC=corp,DC=example,DC=com" -w "p@ssw0rd"
```

描述特定组

```
$ ldapsearch "(&(objectClass=group)(cn=clusterteam))"
 -x -h "192.0.2.254" -b "DC=corp,DC=example,DC=com" -D
 "CN=Admin,OU=Users,OU=CORP,DC=corp,DC=example,DC=com" -w "p@ssw0rd"
```

6. 从组中删除用户。

```
$ adcli remove-member "clusterteam" "clusteruser" --domain "corp.example.com" -U
 "Admin"
```

7. 删除用户。

```
$ adcli delete-user "clusteruser" --domain "corp.example.com" -U "Admin"
```

8. 删除组。

```
$ adcli delete-group "clusterteam" --domain "corp.example.com" -U "Admin"
```

创建集群

如果您尚未退出 Amazon EC2 实例，请立即退出。

设置该环境是为了创建可以针对 Active Directory (AD) 对用户进行身份验证的集群。

创建简单的集群配置并提供与连接到 AD 相关的设置。想要了解更多信息，请参阅`DirectoryService`部分。

选择以下集群配置之一，然后将其复制到名为 `ldaps_config.yaml`、`ldaps_nocert_config.yaml` 或 `ldap_config.yaml` 的文件中。

建议您选择具有证书验证功能的 LDAPS 配置。如果选择此配置，则还必须将引导脚本复制到名为 `active-directory.head.post.sh` 的文件中。此外，您必须将其存储在配置文件中指示的 Amazon S3 存储桶中。

具有证书验证功能的 LDAPS 配置（推荐）

> **Note**
>
> 必须更改以下组件。
>
> - `KeyName`: 您的一个 Amazon EC2 密钥对。
> - `SubnetId` / `SubnetIds`： CloudFormation 快速创建堆栈（自动教程）或 python 脚本（手动教程）输出中 IDs 提供的子网之一。
> - `Region`：您在其中创建 AD 基础架构的区域。
> - `DomainAddr`：此 IP 地址是您的 AD 服务的 DNS 地址之一。
> - `PasswordSecretArn`：包含 `DomainReadOnlyUser` 密码的密钥的 Amazon 资源名称 (ARN)。
> - `BucketName`：保存引导脚本的存储桶的名称。
> - `AdditionalPolicies/Policy`：读取域名认证政策的亚马逊资源名称 (ARN)。ReadCertExample
> - `CustomActions/OnNodeConfigured/Args`：保存域名认证策略的密钥的 Amazon 资源名称 (ARN)。
>
> 为了提高安全性，我们建议使用 `HeadNode/Ssh/AllowedIps` 配置来限制对头节点的 SSH 访问。

```
Region: region-id
Image:
```

```
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-abcdef01234567890
  Ssh:
    KeyName: keypair
  Iam:
    AdditionalIamPolicies:
      - Policy: arn:aws:iam::123456789012:policy/ReadCertExample
    S3Access:
      - BucketName: amzn-s3-demo-bucket
        EnableWriteAccess: false
        KeyName: bootstrap/active-directory/active-directory.head.post.sh
  CustomActions:
    OnNodeConfigured:
      Script: s3://amzn-s3-demo-bucket/bootstrap/active-directory/active-
directory.head.post.sh
      Args:
        - arn:aws:secretsmanager:region-id:123456789012:secret:example-cert-123abc
        - /opt/parallelcluster/shared/directory_service/domain-certificate.crt
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue0
      ComputeResources:
        - Name: queue0-t2-micro
          InstanceType: t2.micro
          MinCount: 1
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
DirectoryService:
  DomainName: corp.example.com
  DomainAddr: ldaps://corp.example.com
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:ADSecretPassword-1234
  DomainReadOnlyUser: cn=ReadOnlyUser,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  LdapTlsCaCert: /opt/parallelcluster/shared/directory_service/domain-certificate.crt
  LdapTlsReqCert: hard
```

引导脚本

创建引导程序文件后，在将其上传到 S3 存储桶之前，请运行`chmod +x active-directory.head.post.sh`以授予 AWS ParallelCluster 运行权限。

```
#!/bin/bash
set -e

CERTIFICATE_SECRET_ARN="$1"
CERTIFICATE_PATH="$2"

[[ -z $CERTIFICATE_SECRET_ARN ]] && echo "[ERROR] Missing CERTIFICATE_SECRET_ARN" &&
 exit 1
[[ -z $CERTIFICATE_PATH ]] && echo "[ERROR] Missing CERTIFICATE_PATH" && exit 1

source /etc/parallelcluster/cfnconfig
REGION="${cfn_region:?}"

mkdir -p $(dirname $CERTIFICATE_PATH)
aws secretsmanager get-secret-value --region $REGION --secret-id
 $CERTIFICATE_SECRET_ARN --query SecretString --output text > $CERTIFICATE_PATH
```

没有证书验证功能的 LDAPS 配置

> **Note**
>
> 必须更改以下组件。
>
> - `KeyName`: 您的一个 Amazon EC2 密钥对。
> - `SubnetId` / `SubnetIds`： CloudFormation 快速创建堆栈（自动教程）或 python 脚本（手动教程）输出中的子网 IDs 之一。
> - `Region`：您在其中创建 AD 基础架构的区域。
> - `DomainAddr`：此 IP 地址是您的 AD 服务的 DNS 地址之一。
> - `PasswordSecretArn`：包含 `DomainReadOnlyUser` 密码的密钥的 Amazon 资源名称 (ARN)。
>
> 为了更好的安全状况，我们建议使用 HeadNode /Ssh/ AllowedIps 配置来限制对头节点的 SSH 访问。

```
Region: region-id
Image:
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-abcdef01234567890
  Ssh:
    KeyName: keypair
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue0
      ComputeResources:
        - Name: queue0-t2-micro
          InstanceType: t2.micro
          MinCount: 1
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
DirectoryService:
  DomainName: corp.example.com
  DomainAddr: ldaps://corp.example.com
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:ADSecretPassword-1234
  DomainReadOnlyUser: cn=ReadOnlyUser,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  LdapTlsReqCert: never
```

LDAP 配置

> **Note**
>
> 必须更改以下组件。
>
> - `KeyName`: 您的一个 Amazon EC2 密钥对。
> - `SubnetId` / `SubnetIds`： CloudFormation 快速创建堆栈（自动教程）或 python 脚本（手动教程）输出中 IDs 提供的子网之一。
> - `Region`：您在其中创建 AD 基础架构的区域。
> - `DomainAddr`：此 IP 地址是您的 AD 服务的 DNS 地址之一。

- `PasswordSecretArn`：包含 `DomainReadOnlyUser` 密码的密钥的 Amazon 资源名称 (ARN)。

为了更好的安全状况，我们建议使用 HeadNode /Ssh/ AllowedIps 配置来限制对头节点的 SSH 访问。

```
Region: region-id
Image:
  Os: alinux2
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-abcdef01234567890
  Ssh:
    KeyName: keypair
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: queue0
      ComputeResources:
        - Name: queue0-t2-micro
          InstanceType: t2.micro
          MinCount: 1
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
DirectoryService:
  DomainName: dc=corp,dc=example,dc=com
  DomainAddr: ldap://192.0.2.254,ldap://203.0.113.237
  PasswordSecretArn: arn:aws:secretsmanager:region-
id:123456789012:secret:ADSecretPassword-1234
  DomainReadOnlyUser: cn=ReadOnlyUser,ou=Users,ou=CORP,dc=corp,dc=example,dc=com
  AdditionalSssdConfigs:
    ldap_auth_disable_tls_never_use_in_production: True
```

使用以下命令创建集群。

```
$ pcluster create-cluster --cluster-name "ad-cluster" --cluster-configuration "./
ldaps_config.yaml"
```

```
{
  "cluster": {
    "clusterName": "pcluster",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:region-id:123456789012:stack/ad-
cluster/1234567-abcd-0123-def0-abcdef0123456",
    "region": "region-id",
    "version": 3.13.2,
    "clusterStatus": "CREATE_IN_PROGRESS"
  }
}
```

以用户身份连接到集群

您可以使用以下命令确定集群的状态。

```
$ pcluster describe-cluster -n ad-cluster --region "region-id" --query "clusterStatus"
```

输出如下所示。

```
"CREATE_IN_PROGRESS" / "CREATE_COMPLETE"
```

达到 `"CREATE_COMPLETE"` 状态后，使用创建的用户名和密码登录。

```
$ HEAD_NODE_IP=$(pcluster describe-cluster -n "ad-cluster" --region "region-id" --query
 headNode.publicIpAddress | xargs echo)
```

```
$ ssh user000@$HEAD_NODE_IP
```

您可以通过提供 `/home/user000@HEAD_NODE_IP/.ssh/id_rsa` 中为新用户创建的 SSH 密钥，而不使用密码进行登录。

如果 `ssh` 命令成功，则表示您已成功以用户身份连接到集群，并通过了身份验证，可以使用 Active Directory (AD)。

清理

1. 从本地计算机上删除集群。

   ```
   $ pcluster delete-cluster --cluster-name "ad-cluster" --region "region-id"
   ```

```
{
  "cluster": {
    "clusterName": "ad-cluster",
    "cloudformationStackStatus": "DELETE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:region-id:123456789012:stack/
ad-cluster/1234567-abcd-0123-def0-abcdef0123456",
    "region": "region-id",
    "version": "3.13.2",
    "clusterStatus": "DELETE_IN_PROGRESS"
  }
}
```

2. 检查集群删除进度。

```
$ pcluster describe-cluster --cluster-name "ad-cluster" --region "region-id" --
query "clusterStatus"
"DELETE_IN_PROGRESS"
```

成功删除集群后，继续执行下一步。

自动

删除 Active Directory 资源

1. 来自https://console.aws.amazon.com/cloudformation/。
2. 在导航窗格中，选择堆栈。
3. 从堆栈列表中选择 AD 堆栈（例如 `pcluster-ad`）。
4. 选择删除。

手动

1. 删除 Amazon EC2 实例。
   a. 从中 https://console.aws.amazon.com/ec2/，选择导航窗格中的实例。
   b. 从实例列表中，选择您为将用户添加到目录而创建的实例。
   c. 选择实例状态，然后选择终止实例。

2. 删除托管区。

   a. 创建包含以下内容的 `recordset-delete.json`。在此示例中，HostedZoneId 是负载均衡器的规范托管区域 ID。

```
{
  "Changes": [
    {
      "Action": "DELETE",
      "ResourceRecordSet": {
        "Name": "corp.example.com",
        "Type": "A",
        "Region": "region-id",
        "SetIdentifier": "pcluster-active-directory",
        "AliasTarget": {
          "HostedZoneId": "Z2IFOLAFXWLO4F",
          "DNSName": "CorpExampleCom-NLB-3afe296bf4ba80d4.elb.region-
id.amazonaws.com",
          "EvaluateTargetHealth": true
        }
      }
    }
  ]
}
```

   b. 使用托管区 ID 将记录集更改提交到托管区。

```
$ aws route53 change-resource-record-sets --hosted-zone-
id Z09020002B5MZQNXMSJUB \
  --change-batch file://recordset-delete.json
{
 "ChangeInfo": {
     "Id": "/change/C04853642A0TH2TJ5NLNI",
     "Status": "PENDING",
     "SubmittedAt": "2022-05-05T14:25:51.046000+00:00"
 }
}
```

   c. 删除托管区。

```
$ aws route53 delete-hosted-zone --id Z09020002B5MZQNXMSJUB
{
 "ChangeInfo": {
```

```
        "Id": "/change/C0468051QFABTVHMDEG9",
        "Status": "PENDING",
        "SubmittedAt": "2022-05-05T14:26:13.814000+00:00"
    }
}
```

3. 删除 LB 侦听器。

```
$ aws elbv2 delete-listener \
  --listener-arn arn:aws:elasticloadbalancing:region-id:123456789012:listener/net/
CorpExampleCom-NLB/3afe296bf4ba80d4/a8f9d97318743d4b --region region-id
```

4. 删除目标组。

```
$ aws elbv2 delete-target-group \
  --target-group-arn arn:aws:elasticloadbalancing:region-
id:123456789012:targetgroup/CorpExampleCom-Targets/44577c583b695e81 --
region region-id
```

5. 删除负载均衡器。

```
$ aws elbv2 delete-load-balancer \
  --load-balancer-arn arn:aws:elasticloadbalancing:region-
id:123456789012:loadbalancer/net/CorpExampleCom-NLB/3afe296bf4ba80d4 --
region region-id
```

6. 删除集群用于从 Secrets Manager 读取证书的策略。

```
$ aws iam delete-policy --policy-arn arn:aws:iam::123456789012:policy/
ReadCertExample
```

7. 删除包含域证书的密钥。

```
$ aws secretsmanager delete-secret \
  --secret-id arn:aws:secretsmanager:region-id:123456789012:secret:example-
cert-123abc \
  --region region-id
{
 "ARN": "arn:aws:secretsmanager:region-id:123456789012:secret:example-cert-123abc",
 "Name": "example-cert",
 "DeletionDate": "2022-06-04T16:27:36.183000+02:00"
}
```

8. 从 ACM 中删除证书。

```
$ aws acm delete-certificate \
  --certificate-arn arn:aws:acm:region-
id:123456789012:certificate/343db133-490f-4077-b8d4-3da5bfd89e72 --region region-id
```

9. 删除 Active Directory (AD) 资源。

   a. IDs 从 python 脚本的输出中获取以下资源`ad.py`：

      - AD ID
      - AD 子网 IDs
      - AD VPC ID

   b. 通过运行以下命令删除目录。

```
$ aws ds delete-directory --directory-id d-abcdef0123456789 --region region-id
{
   "DirectoryId": "d-abcdef0123456789"
}
```

   c. 列出 VPC 中的安全组。

```
$ aws ec2 describe-security-groups --filters '[{"Name":"vpc-id","Values":
["vpc-07614ade95ebad1bc"]}]' --region region-id
```

   d. 删除自定义安全组。

```
$ aws ec2 delete-security-group --group-id sg-021345abcdef6789 --region region-
id
```

   e. 删除子网。

```
$ aws ec2 delete-subnet --subnet-id subnet-1234567890abcdef --region region-id
```

```
$ aws ec2 delete-subnet --subnet-id subnet-021345abcdef6789 --region region-id
```

   f. 描述互联网网关。

```
$ aws ec2 describe-internet-gateways \
  --filters Name=attachment.vpc-id,Values=vpc-021345abcdef6789 \
```

```
  --region region-id
{
  "InternetGateways": [
    {
      "Attachments": [
        {
          "State": "available",
          "VpcId": "vpc-021345abcdef6789"
        }
      ],
      "InternetGatewayId": "igw-1234567890abcdef",
      "OwnerId": "123456789012",
      "Tags": []
    }
  ]
}
```

g. 分离互联网网关。

```
$ aws ec2 detach-internet-gateway \
  --internet-gateway-id igw-1234567890abcdef \
  --vpc-id vpc-021345abcdef6789 \
  --region region-id
```

h. 删除互联网网关。

```
$ aws ec2 delete-internet-gateway \
  --internet-gateway-id igw-1234567890abcdef \
  --region region-id
```

i. 删除 VPC。

```
$ aws ec2 delete-vpc \
  --vpc-id vpc-021345abcdef6789 \
  --region region-id
```

j. 删除包含 `ReadOnlyUser` 密码的密钥。

```
$ aws secretsmanager delete-secret \
  --secret-id arn:aws:secretsmanager:region-
id:123456789012:secret:ADSecretPassword-1234" \
  --region region-id
```

使用密 AWS KMS 钥配置共享存储加密

了解如何设置客户托管 AWS KMS 密钥来加密和保护您在为其配置的集群文件存储系统中的数据 AWS ParallelCluster。

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

AWS ParallelCluster 支持以下共享存储配置选项：

- `SharedStorage` / `EbsSettings` / `KmsKeyId`
- `SharedStorage` / `EfsSettings` / `KmsKeyId`
- `SharedStorage` / `FsxLustreSettings` / `KmsKeyId`

您可以使用这些选项为 Amazon EBS、Amazon EFS 和 Lustre 共享存储系统 FSx 加密提供客户托管 AWS KMS 密钥。要使用这些选项，您必须为以下各项创建并配置 IAM 策略：

- `HeadNode` / `Iam` / `AdditionalIamPolicies` / `Policy`
- `Scheduler` / `SlurmQueues` / `Iam` / `AdditionalIamPolicies` / `Policy`

先决条件

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 你有一个 A mazon EC2 密钥 pair。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

主题

- 创建策略
- 配置和创建集群

创建策略

在本教程中，您将创建一个使用密 AWS KMS 钥配置共享存储加密的策略。

创建策略。

1. 前往 IAM 控制台：https://console.aws.amazon.com/iam/主页。
2. 选择策略。
3. 选择创建策略。
4. 选择 JSON 选项卡，然后粘贴以下策略。请务必将所有出现的*123456789012*替换为您的 AWS 账户 ID 和密钥 Amazon 资源名称 (ARN) 以及 AWS 区域 您自己的名称。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:DescribeKey",
                "kms:ReEncrypt*",
                "kms:CreateGrant",
                "kms:Decrypt"
            ],
            "Resource": [
                "arn:aws:kms:region-id:123456789012:key/abcd1234-ef56-gh78-ij90-
abcd1234efgh5678"
            ]
        }
    ]
}
```

5. 对于本教程，将策略命名为 `ParallelClusterKmsPolicy`，然后选择创建策略。
6. 记下策略 ARN。您需要用它来配置您的集群。

配置和创建集群

下面是一个示例集群配置，其中包括带加密功能的 Amazon Elastic Block Store 共享文件系统。

```
Region: eu-west-1
Image:
  Os: alinux2
```

```
HeadNode:
  InstanceType: t2.micro
  Networking:
    SubnetId: subnet-abcdef01234567890
  Ssh:
    KeyName: my-ssh-key
  Iam:
    AdditionalIamPolicies:
      - Policy: arn:aws:iam::123456789012:policy/ParallelClusterKmsPolicy
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: q1
      ComputeResources:
        - Name: t2micro
          InstanceType: t2.micro
          MinCount: 0
          MaxCount: 10
      Networking:
        SubnetIds:
          - subnet-abcdef01234567890
      Iam:
        AdditionalIamPolicies:
          - Policy: arn:aws:iam::123456789012:policy/ParallelClusterKmsPolicy
SharedStorage:
  - MountDir: /shared/ebs1
    Name: shared-ebs1
    StorageType: Ebs
    EbsSettings:
      Encrypted: True
      KmsKeyId: abcd1234-ef56-gh78-ij90-abcd1234efgh5678
```

将红色文本项目替换为您自己的值。然后，创建一个使用您的 AWS KMS 密钥对 Amazon EBS 中的数据进行加密的集群。

Amazon EFS 和 FSx Lustre 文件系统的配置类似。

Amazon EFS `SharedStorage` 配置如下。

```
...
SharedStorage:
  - MountDir: /shared/efs1
    Name: shared-efs1
```

```
    StorageType: Efs
    EfsSettings:
      Encrypted: True
      KmsKeyId: abcd1234-ef56-gh78-ij90-abcd1234efgh5678
```

fo FSx r Lustre 的`SharedStorage`配置如下所示。

```
...
SharedStorage:
  - MountDir: /shared/fsx1
    Name: shared-fsx1
    StorageType: FsxLustre
    FsxLustreSettings:
      StorageCapacity: 1200
      DeploymentType: PERSISTENT_1
      PerUnitStorageThroughput: 200
      KmsKeyId: abcd1234-ef56-gh78-ij90-abcd1234efgh5678
```

在多队列模式集群中运行作业

本教程介绍如何在多队列模式下运行第一个 Hello World “” 作业。 AWS ParallelCluster

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

先决条件

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 你有一个 A mazon EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

配置集群

首先，通过运行以下命令来验证安装 AWS ParallelCluster 是否正确。

```
$ pcluster version
```

有关 `pcluster version`的更多信息，请参阅`pcluster version`。

此命令返回的运行版本 AWS ParallelCluster。

接下来，运行 `pcluster configure` 以生成基本配置文件。按照运行此命令后的所有提示进行操作。

```
$ pcluster configure --config multi-queue-mode.yaml
```

有关 `pcluster configure` 命令的更多信息，请参阅pcluster configure。

完成此步骤后，将出现一个名为 `multi-queue-mode.yaml` 的基本配置文件。此文件包含基本集群配置。

在下一步中，您将修改新配置文件并启动包含多个队列的集群。

> **Note**
>
> 本教程中使用的某些实例不符合免费套餐资格。

在本教程中，请修改您的配置文件以匹配以下配置。以红色突出显示的项目代表您的配置文件值。请使用您自己的值。

```
Region: region-id
Image:
 Os: alinux2
HeadNode:
 InstanceType: c5.xlarge
 Networking:
   SubnetId: subnet-abcdef01234567890
 Ssh:
   KeyName: yourkeypair
Scheduling:
 Scheduler: slurm
 SlurmQueues:
 - Name: spot
   ComputeResources:
   - Name: c5xlarge
     InstanceType: c5.xlarge
     MinCount: 1
     MaxCount: 10
   - Name: t2micro
     InstanceType: t2.micro
```

```
      MinCount: 1
      MaxCount: 10
    Networking:
      SubnetIds:
      - subnet-abcdef01234567890
  - Name: ondemand
    ComputeResources:
    - Name: c52xlarge
      InstanceType: c5.2xlarge
      MinCount: 0
      MaxCount: 10
    Networking:
      SubnetIds:
      - subnet-021345abcdef6789
```

创建集群

根据您的配置文件，创建一个名为 `multi-queue-cluster` 的集群。

```
$ pcluster create-cluster --cluster-name multi-queue-cluster --cluster-configuration
 multi-queue-mode.yaml
{
 "cluster": {
   "clusterName": "multi-queue-cluster",
   "cloudformationStackStatus": "CREATE_IN_PROGRESS",
   "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:123456789012:stack/
multi-queue-cluster/1234567-abcd-0123-def0-abcdef0123456",
   "region": "eu-west-1",
   "version": "3.13.2",
   "clusterStatus": "CREATE_IN_PROGRESS"
 }
}
```

有关 `pcluster create-cluster` 命令的更多信息，请参阅pcluster create-cluster。

要检查集群的状态，请运行以下命令。

```
$ pcluster list-clusters
{
 "cluster": {
   "clusterName": "multi-queue-cluster",
   "cloudformationStackStatus": "CREATE_IN_PROGRESS",
```

```
   "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:123456789012:stack/
multi-queue-cluster/1234567-abcd-0123-def0-abcdef0123456",
   "region": "eu-west-1",
   "version": "3.13.2",
   "clusterStatus": "CREATE_IN_PROGRESS"
 }
}
```

创建集群后，clusterStatus 字段将显示 CREATE_COMPLETE。

登录到头节点

使用您的私有 SSH 密钥文件登录到头节点。

```
$ pcluster ssh --cluster-name multi-queue-cluster -i ~/path/to/yourkeyfile.pem
```

有关 pcluster ssh的更多信息，请参阅pcluster ssh。

登录后，运行命令 sinfo 以验证是否已设置和配置调度器队列。

有关 sinfo 的更多信息，请参阅 Slurm 文档 中的 sinfo。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
spot*        up   infinite     18  idle~ spot-dy-c5xlarge-[1-9],spot-dy-t2micro-[1-9]
spot*        up   infinite      2  idle  spot-st-c5xlarge-1,spot-st-t2micro-1
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
```

输出显示您的集群中有一个 t2.micro 和一个 c5.xlarge 计算节点处于 idle 状态。

其他节点都处于节能状态，由节点状态的~后缀表示，没有 Amazon EC2 实例支持它们。默认队列由队列名称后面的 * 后缀指示。spot 是默认作业队列。

在多队列模式下运行作业

接下来，尝试将作业运行到睡眠模式一段时间。该作业稍后将输出自己的主机名。确保当前用户可以运行此脚本。

```
$ tee <<EOF hellojob.sh
#!/bin/bash
sleep 30
echo "Hello World from \$(hostname)"
```

```
EOF

$ chmod +x hellojob.sh
$ ls -l hellojob.sh
-rwxrwxr-x 1 ec2-user ec2-user 57 Sep 23 21:57 hellojob.sh
```

使用 sbatch 命令提交作业。使用 -N 2 选项为该作业请求两个节点，然后验证作业是否成功提交。有关 sbatch 的更多信息，请参阅 Slurm 文档 中的 sbatch。

```
$ sbatch -N 2 --wrap "srun hellojob.sh"
Submitted batch job 1
```

您可以使用 squeue 命令查看您的队列并检查该作业的状态。由于您未指定特定队列，因此使用默认队列 (spot)。有关 squeue 的更多信息，请参阅 Slurm 文档 中的 squeue。

```
$ squeue
JOBID PARTITION     NAME     USER  ST       TIME  NODES NODELIST(REASON)
   1      spot     wrap ec2-user  R       0:10      2 spot-st-c5xlarge-1,spot-st-
t2micro-1
```

输出显示此作业目前处于运行状态。等待作业完成。这大约需要 30 秒。然后，再次运行 squeue。

```
$ squeue
JOBID PARTITION     NAME     USER          ST       TIME  NODES NODELIST(REASON)
```

现在，队列中的作业已全部完成，请在当前目录中查找名为 slurm-1.out 的输出文件。

```
$ cat slurm-1.out
Hello World from spot-st-t2micro-1
Hello World from spot-st-c5xlarge-1
```

输出显示该作业已在 spot-st-t2micro-1 和 spot-st-c5xlarge-1 节点上成功运行。

现在，通过使用以下命令为特定实例指定约束条件来提交相同的作业。

```
$ sbatch -N 3 -p spot -C "[c5.xlarge*1&t2.micro*2]" --wrap "srun hellojob.sh"
Submitted batch job 2
```

您对 sbatch 使用了以下参数：

- -N 3：请求三个节点。

- `-p spot`：将作业提交到 `spot` 队列。您也可以通过指定 `-p ondemand`，将作业提交到 `ondemand` 队列。
- `-C "[c5.xlarge*1&t2.micro*2]"`：指定该作业的特定节点约束条件。这将请求对该作业使用一个 `c5.xlarge` 节点和两个 `t2.micro` 节点。

运行 `sinfo` 命令查看节点和队列。中的队 AWS ParallelCluster 列称为中的分区Slurm。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
spot*        up   infinite      1  alloc# spot-dy-t2micro-1
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[2-10],spot-dy-t2micro-[2-9]
spot*        up   infinite      1  mix   spot-st-c5xlarge-1
spot*        up   infinite      1  alloc spot-st-t2micro-1
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
```

节点正在启动。这由节点状态上的 # 后缀指示。运行 squeue 命令查看集群中作业的信息。

```
$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
   2      spot     wrap ec2-user CF       0:04      3 spot-dy-c5xlarge-1,spot-dy-
t2micro-1,spot-st-t2micro-1
```

您的作业处于 `CF` (CONFIGURING) 状态，正在等待实例纵向扩展并加入集群。

大约三分钟后，节点可用，并且作业进入 R (RUNNING) 状态。

```
$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
   2      spot     wrap ec2-user  R       0:07      3 spot-dy-t2micro-1,spot-st-
c5xlarge-1,spot-st-t2micro-1
```

作业完成，所有三个节点都处于 `idle` 状态。

```
$ squeue
JOBID PARTITION     NAME     USER ST       TIME  NODES NODELIST(REASON)
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[1-9],spot-dy-t2micro-[2-9]
spot*        up   infinite      3  idle  spot-dy-t2micro-1,spot-st-c5xlarge-1,spot-st-
t2micro-1
```

```
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
```

然后，当队列中没有剩余作业后，在本地目录中查看 slurm-2.out。

```
$ cat slurm-2.out
Hello World from spot-st-t2micro-1
Hello World from spot-dy-t2micro-1
Hello World from spot-st-c5xlarge-1
```

以下是集群的最终状态。

```
$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
spot*        up   infinite     17  idle~ spot-dy-c5xlarge-[1-9],spot-dy-t2micro-[2-9]
spot*        up   infinite      3  idle  spot-dy-t2micro-1,spot-st-c5xlarge-1,spot-st-
t2micro-1
ondemand     up   infinite     10  idle~ ondemand-dy-c52xlarge-[1-10]
```

注销集群后，您可以通过运行 pcluster delete-cluster 来进行清理。有关更多信息，请参阅pcluster list-clusters和pcluster delete-cluster。

```
$ pcluster list-clusters
{
 "clusters": [
   {
     "clusterName": "multi-queue-cluster",
     "cloudformationStackStatus": "CREATE_COMPLETE",
     "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:123456789012:stack/
multi-queue-cluster/1234567-abcd-0123-def0-abcdef0123456",
     "region": "eu-west-1",
     "version": "3.1.4",
     "clusterStatus": "CREATE_COMPLETE"
   }
 ]
}
$ pcluster delete-cluster -n multi-queue-cluster
{
 "cluster": {
   "clusterName": "multi-queue-cluster",
   "cloudformationStackStatus": "DELETE_IN_PROGRESS",
   "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:123456789012:stack/
multi-queue-cluster/1234567-abcd-0123-def0-abcdef0123456",
```

```
    "region": "eu-west-1",
    "version": "3.1.4",
    "clusterStatus": "DELETE_IN_PROGRESS"
  }
}
```

使用 AWS ParallelCluster API

在本教程中，您将使用 Amazon API Gateway 和 AWS ParallelCluster CloudFormation 模板来构建和测试 API。然后，您可以使用上提供的示例客户端 GitHub 来使用 API。有关使用 API 的更多信息，请参阅 AWS ParallelCluster API。

有关更多信息，请参阅 Image Builder 用户指南中的使用 Imag EC2 e Builder 创建自定义组件。

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

先决条件

- 已在您的计算环境中安装和配置。 AWS CLI
- AWS ParallelCluster 安装在虚拟环境中。有关更多信息，请参阅 AWS ParallelCluster 在虚拟环境中安装（推荐）。
- 你有一个 A mazon EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

步骤 1：使用 Amazon API Gateway 构建 API

留在您的主用户目录中并激活您的虚拟环境：

1. 安装有用的 JSON 命令行处理器。

```
$ sudo yum groupinstall -y "Development Tools"
 sudo yum install -y jq python3-devel
```

2. 运行以下命令获取您的 AWS ParallelCluster 版本并将其分配给环境变量。

```
$ PCLUSTER_VERSION=$(pcluster version | jq -r '.version')
 echo "export PCLUSTER_VERSION=${PCLUSTER_VERSION}" |tee -a ~/.bashrc
```

3. 创建环境变量并将您的区域 ID 分配给该变量。

```
$ export AWS_DEFAULT_REGION="us-east-1"
 echo "export AWS_DEFAULT_REGION=${AWS_DEFAULT_REGION}" |tee -a ~/.bashrc
```

4. 运行以下命令以部署 API。

```
API_STACK_NAME="pc-api-stack"
 echo "export API_STACK_NAME=${API_STACK_NAME}" |tee -a ~/.bashrc
```

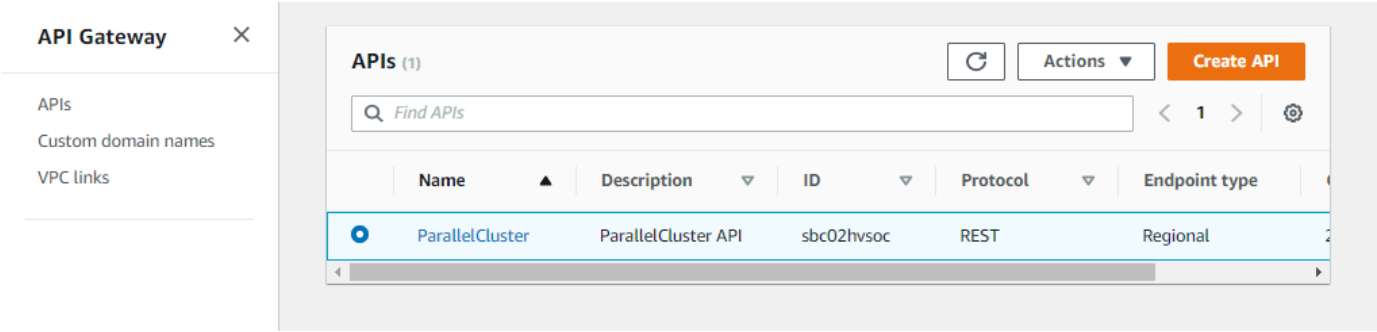
```
aws cloudformation create-stack \
   --region ${AWS_DEFAULT_REGION} \
   --stack-name ${API_STACK_NAME} \
   --template-url https://${AWS_DEFAULT_REGION}-aws-parallelcluster.s3.
${AWS_DEFAULT_REGION}.amazonaws.com/parallelcluster/${PCLUSTER_VERSION}/api/
parallelcluster-api.yaml \
   --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND \
   --parameters ParameterKey=EnableIamAdminAccess,ParameterValue=true

    {
       "StackId": "arn:aws:cloudformation:us-east-1:123456789012:stack/my-api-
stack/abcd1234-ef56-gh78-ei90-1234abcd5678"
    }
```

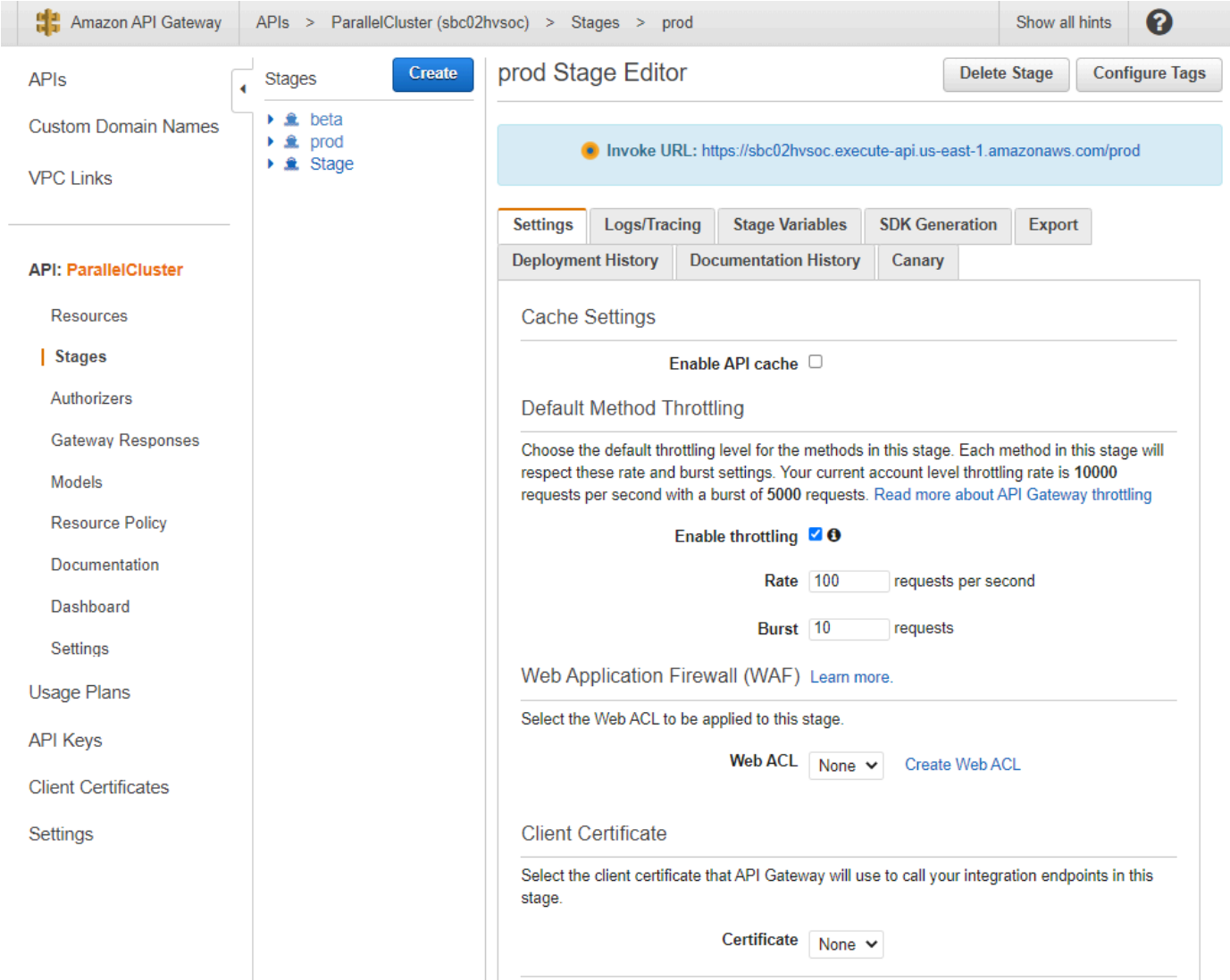
过程完成后，继续执行下一步。

步骤 2：在 Amazon API Gateway 控制台中测试 API

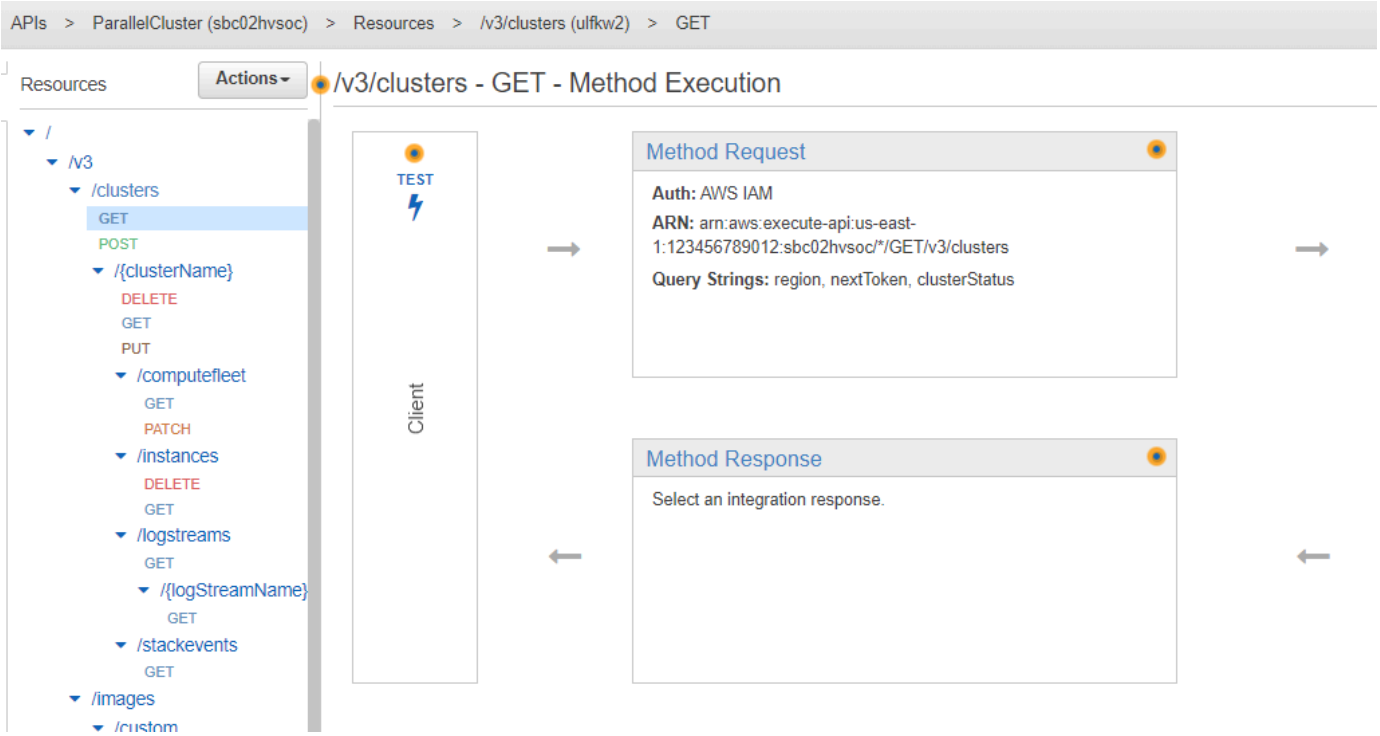
1. 登录到 AWS Management Console。
2. 导航到 Amazon API Gateway 控制台。
3. 选择您的 API 部署。



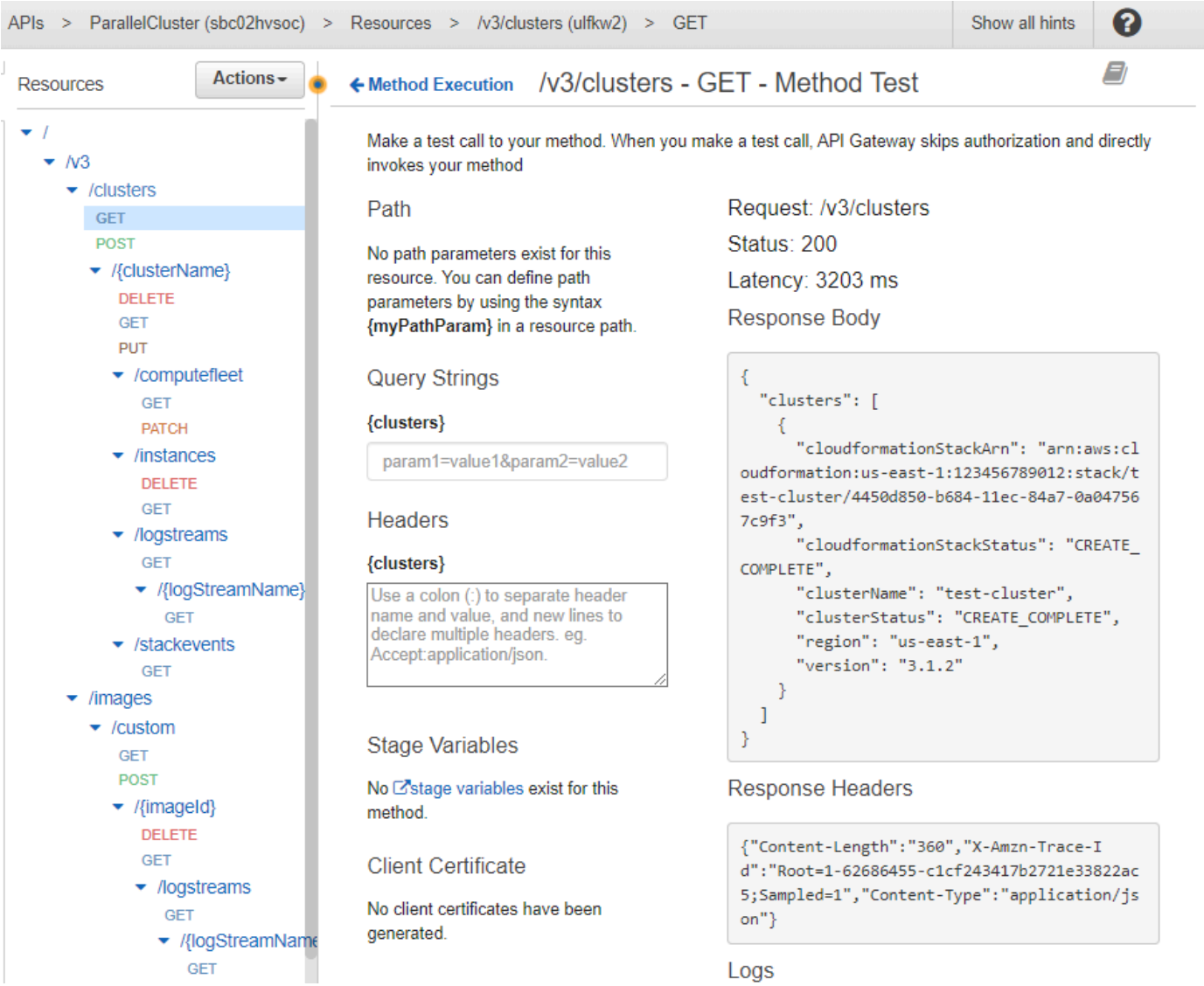
4. 选择阶段，然后选择一个阶段。



5. 记下 API Gateway 提供的用于访问或调用您的 API 的 URL。它以蓝色突出显示。
6. 选择资源，然后选择 **/clusters** 下面的 **GET**。
7. 选择测试图标，然后向下滚动并选择测试图标。



将显示 `/clusters GET` 的响应。



步骤 3：准备并测试用于调用 API 的示例客户端

将 AWS ParallelCluster 源代码克隆`cd`到`api`目录中，然后安装 Python 客户端库。

1.
```
$ git clone -b v${PCLUSTER_VERSION} https://github.com/aws/aws-parallelcluster aws-parallelcluster-v${PCLUSTER_VERSION}
 cd aws-parallelcluster-v${PCLUSTER_VERSION}/api
```

```
$ pip3 install client/src
```

2. 导航回您的主用户目录。

3. 导出客户端在运行时使用的 API Gateway 基本 URL。

```
$ export PCLUSTER_API_URL=$( aws cloudformation describe-stacks
 --stack-name ${API_STACK_NAME} --query 'Stacks[0].Outputs[?
OutputKey==`ParallelClusterApiInvokeUrl`].OutputValue' --output text )
 echo "export PCLUSTER_API_URL=${PCLUSTER_API_URL}" |tee -a ~/.bashrc
```

4. 导出客户端用于创建集群的集群名称。

```
$ export CLUSTER_NAME="test-api-cluster"
 echo "export CLUSTER_NAME=${CLUSTER_NAME}" |tee -a ~/.bashrc
```

5. 运行以下命令以存储示例客户端用于访问 API 的凭证。

```
$ export PCLUSTER_API_USER_ROLE=$( aws cloudformation describe-
stacks --stack-name ${API_STACK_NAME} --query 'Stacks[0].Outputs[?
OutputKey==`ParallelClusterApiUserRole`].OutputValue' --output text )
 echo "export PCLUSTER_API_USER_ROLE=${PCLUSTER_API_USER_ROLE}" |tee -a ~/.bashrc
```

步骤 4：复制客户端代码脚本并运行集群测试

1. 将以下示例客户端代码复制到您的主用户目录中的 `test_pcluster_client.py`。客户端代码发出执行以下操作的请求：

   - 创建集群。
   - 描述集群。
   - 列出集群。
   - 描述计算实例集。
   - 描述集群实例。

```
# Copyright 2021 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
```

```
# without restriction, including without limitation the rights to use, copy,
 modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and
 to
# permit persons to whom the Software is furnished to do so.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
 COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
#
#  Author: Evan F. Bollig (Github: bollig)

import time, datetime
import os
import pcluster_client
from pprint import pprint
from pcluster_client.api import (
    cluster_compute_fleet_api,
    cluster_instances_api,
    cluster_operations_api
)
from pcluster_client.model.create_cluster_request_content import
 CreateClusterRequestContent
from pcluster_client.model.cluster_status import ClusterStatus
region=os.environ.get("AWS_DEFAULT_REGION")

# Defining the host is optional and defaults to http://localhost
# See configuration.py for a list of all supported configuration parameters.
configuration = pcluster_client.Configuration(
    host = os.environ.get("PCLUSTER_API_URL")
)
cluster_name=os.environ.get("CLUSTER_NAME")

# Enter a context with an instance of the API client
with pcluster_client.ApiClient(configuration) as api_client:
    cluster_ops = cluster_operations_api.ClusterOperationsApi(api_client)
    fleet_ops = cluster_compute_fleet_api.ClusterComputeFleetApi(api_client)
    instance_ops = cluster_instances_api.ClusterInstancesApi(api_client)
```

```
    # Create cluster
    build_done = False
    try:
        with open('cluster-config.yaml', encoding="utf-8") as f:
            body = CreateClusterRequestContent(cluster_name=cluster_name,
cluster_configuration=f.read())
            api_response = cluster_ops.create_cluster(body, region=region)
    except pcluster_client.ApiException as e:
        print("Exception when calling create_cluster: %s\n" % e)
        build_done = True
    time.sleep(60)

    # Confirm cluster status with describe_cluster
    while not build_done:
        try:
            api_response = cluster_ops.describe_cluster(cluster_name,
region=region)
            pprint(api_response)
            if api_response.cluster_status == ClusterStatus('CREATE_IN_PROGRESS'):
                print('. . . working . . .', end='', flush=True)
                time.sleep(60)
            elif api_response.cluster_status == ClusterStatus('CREATE_COMPLETE'):
                print('READY!')
                build_done = True
            else:
                print('ERROR!!!!')
                build_done = True
        except pcluster_client.ApiException as e:
            print("Exception when calling describe_cluster: %s\n" % e)

    # List clusters
    try:
        api_response = cluster_ops.list_clusters(region=region)
        pprint(api_response)
    except pcluster_client.ApiException as e:
        print("Exception when calling list_clusters: %s\n" % e)

    # DescribeComputeFleet
    try:
        api_response = fleet_ops.describe_compute_fleet(cluster_name,
region=region)
        pprint(api_response)
    except pcluster_client.ApiException as e:
        print("Exception when calling compute fleet: %s\n" % e)
```

```
    # DescribeClusterInstances
    try:
        api_response = instance_ops.describe_cluster_instances(cluster_name,
 region=region)
        pprint(api_response)
    except pcluster_client.ApiException as e:
        print("Exception when calling describe_cluster_instances: %s\n" % e)
```

2. 创建集群配置。

```
$ pcluster configure --config cluster-config.yaml
```

3. API 客户端库将自动从您的环境变量（例如 AWS_ACCESS_KEY_ID、AWS_SECRET_ACCESS_KEY 或 AWS_SESSION_TOKEN）或 $HOME/.aws 中检测配置详细信息。以下命令可将您当前的 IAM 角色切换到指定的 ParallelClusterApiUserRole。

```
$  eval $(aws sts assume-role --role-arn ${PCLUSTER_API_USER_ROLE} --role-
session-name ApiTestSession | jq -r '.Credentials | "export AWS_ACCESS_KEY_ID=
\(.AccessKeyId)\nexport AWS_SECRET_ACCESS_KEY=\(.SecretAccessKey)\nexport
 AWS_SESSION_TOKEN=\(.SessionToken)\n"')
```

需要注意的错误：

如果您看到类似于下面的错误，则表示您已经假定 ParallelClusterApiUserRole 和您的 AWS_SESSION_TOKEN 已过期。

```
An error occurred (AccessDenied) when calling the AssumeRole operation:
User: arn:aws:sts::XXXXXXXXXXXX:assumed-role/ParallelClusterApiUserRole-XXXXXXXX-
XXXX-XXXX-XXXX-XXXXXXXXXXXX/ApiTestSession
is not authorized to perform: sts:AssumeRole on resource:
 arn:aws:iam::XXXXXXXXXXXX:role/ParallelClusterApiUserRole-XXXXXXXX-XXXX-XXXX-XXXX-
XXXXXXXXXXXX
```

删除该角色，然后重新运行 aws sts assume-role 命令以使用 ParallelClusterApiUserRole。

```
$ unset AWS_SESSION_TOKEN
unset AWS_SECRET_ACCESS_KEY
unset AWS_ACCESS_KEY_ID
```

要向您当前的用户提供 API 访问权限，您必须扩展资源策略。

4. 运行以下命令以测试示例客户端。

```
$ python3 test_pcluster_client.py
{'cluster_configuration': 'Region: us-east-1\n'
                          'Image:\n'
                          '  Os: alinux2\n'
                          'HeadNode:\n'
                          '  InstanceType: t2.micro\n'
                          '  Networking . . . :\n'
                          '    SubnetId: subnet-1234567890abcdef0\n'
                          '  Ssh:\n'
                          '    KeyName: adpc\n'
                          'Scheduling:\n'
                          '  Scheduler: slurm\n'
                          '  SlurmQueues:\n'
                          '  - Name: queue1\n'
                          '    ComputeResources:\n'
                          '    - Name: t2micro\n'
                          '      InstanceType: t2.micro\n'
                          '      MinCount: 0\n'
                          '      MaxCount: 10\n'
                          '    Networking . . . :\n'
                          '      SubnetIds:\n'
                          '      - subnet-1234567890abcdef0\n',
 'cluster_name': 'test-api-cluster'}
{'cloud_formation_stack_status': 'CREATE_IN_PROGRESS',
 'cloudformation_stack_arn': 'arn:aws:cloudformation:us-east-1:123456789012:stack/
test-api-cluster/abcd1234-ef56-gh78-ij90-1234abcd5678',
 'cluster_configuration': {'url': 'https://parallelcluster-021345abcdef6789-v1-do-
not-delete...},
 'cluster_name': 'test-api-cluster',
 'cluster_status': 'CREATE_IN_PROGRESS',
 'compute_fleet_status': 'UNKNOWN',
 'creation_time': datetime.datetime(2022, 4, 28, 16, 18, 47, 972000,
 tzinfo=tzlocal()),
 'last_updated_time': datetime.datetime(2022, 4, 28, 16, 18, 47, 972000,
 tzinfo=tzlocal()),
 'region': 'us-east-1',
 'tags': [{'key': 'parallelcluster:version', 'value': '3.1.3'}],
 'version': '3.1.3'}
        .
        .
        .
```

```
. . . working . . . {'cloud_formation_stack_status': 'CREATE_COMPLETE',
 'cloudformation_stack_arn': 'arn:aws:cloudformation:us-east-1:123456789012:stack/
test-api-cluster/abcd1234-ef56-gh78-ij90-1234abcd5678',
 'cluster_configuration': {'url': 'https://parallelcluster-021345abcdef6789-v1-do-
not-delete...},
 'cluster_name': 'test-api-cluster',
 'cluster_status': 'CREATE_COMPLETE',
 'compute_fleet_status': 'RUNNING',
 'creation_time': datetime.datetime(2022, 4, 28, 16, 18, 47, 972000,
 tzinfo=tzlocal()),
 'head_node': {'instance_id': 'i-abcdef01234567890',
               'instance_type': 't2.micro',
               'launch_time': datetime.datetime(2022, 4, 28, 16, 21, 46,
 tzinfo=tzlocal()),
               'private_ip_address': '172.31.27.153',
               'public_ip_address': '52.90.156.51',
               'state': 'running'},
 'last_updated_time': datetime.datetime(2022, 4, 28, 16, 18, 47, 972000,
 tzinfo=tzlocal()),
 'region': 'us-east-1',
 'tags': [{'key': 'parallelcluster:version', 'value': '3.1.3'}],
 'version': '3.1.3'}
READY!
```

步骤 5：复制客户端代码脚本并删除集群

1. 将以下示例客户端代码复制到 `delete_cluster_client.py`。客户端代码发出删除集群的请求。

```
# Copyright 2021 Amazon.com, Inc. or its affiliates. All Rights Reserved.
# SPDX-License-Identifier: MIT-0
#
# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy,
 modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and
 to
# permit persons to whom the Software is furnished to do so.
#
```

```
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR
 COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
#
#  Author: Evan F. Bollig (Github: bollig)

import time, datetime
import os
import pcluster_client
from pprint import pprint
from pcluster_client.api import (
    cluster_compute_fleet_api,
    cluster_instances_api,
    cluster_operations_api
)
from pcluster_client.model.create_cluster_request_content import
 CreateClusterRequestContent
from pcluster_client.model.cluster_status import ClusterStatus
region=os.environ.get("AWS_DEFAULT_REGION")

# Defining the host is optional and defaults to http://localhost
# See configuration.py for a list of all supported configuration parameters.
configuration = pcluster_client.Configuration(
    host = os.environ.get("PCLUSTER_API_URL")
)
cluster_name=os.environ.get("CLUSTER_NAME")

# Enter a context with an instance of the API client
with pcluster_client.ApiClient(configuration) as api_client:
    cluster_ops = cluster_operations_api.ClusterOperationsApi(api_client)

    # Delete the cluster
    gone = False
    try:
        api_response = cluster_ops.delete_cluster(cluster_name, region=region)
    except pcluster_client.ApiException as e:
        print("Exception when calling delete_cluster: %s\n" % e)
    time.sleep(60)
```

```
    # Confirm cluster status with describe_cluster
    while not gone:
        try:
            api_response = cluster_ops.describe_cluster(cluster_name,
 region=region)
            pprint(api_response)
            if api_response.cluster_status == ClusterStatus('DELETE_IN_PROGRESS'):
                print('. . . working . . .', end='', flush=True)
                time.sleep(60)
        except pcluster_client.ApiException as e:
            gone = True
            print("DELETE COMPLETE or Exception when calling describe_cluster: %s
\n" % e)
```

2. 运行以下命令以删除集群。

```
$ python3 delete_cluster_client.py
{'cloud_formation_stack_status': 'DELETE_IN_PROGRESS',
'cloudformation_stack_arn': 'arn:aws:cloudformation:us-east-1:123456789012:stack/
test-api-cluster/abcd1234-ef56-gh78-ij90-1234abcd5678',
'cluster_configuration': {'url': 'https://parallelcluster-021345abcdef6789-v1-do-
not-delete...},
'cluster_name': 'test-api-cluster',
'cluster_status': 'DELETE_IN_PROGRESS',
'compute_fleet_status': 'UNKNOWN',
'creation_time': datetime.datetime(2022, 4, 28, 16, 50, 47, 943000,
 tzinfo=tzlocal()),
'head_node': {'instance_id': 'i-abcdef01234567890',
              'instance_type': 't2.micro',
              'launch_time': datetime.datetime(2022, 4, 28, 16, 53, 48,
 tzinfo=tzlocal()),
              'private_ip_address': '172.31.17.132',
              'public_ip_address': '34.201.100.37',
              'state': 'running'},
'last_updated_time': datetime.datetime(2022, 4, 28, 16, 50, 47, 943000,
 tzinfo=tzlocal()),
'region': 'us-east-1',
'tags': [{'key': 'parallelcluster:version', 'value': '3.1.3'}],
'version': '3.1.3'}
       .
       .
       .
. . . working . . . {'cloud_formation_stack_status': 'DELETE_IN_PROGRESS',
```

```
'cloudformation_stack_arn': 'arn:aws:cloudformation:us-east-1:123456789012:stack/
test-api-cluster/abcd1234-ef56-gh78-ij90-1234abcd5678',
'cluster_configuration': {'url': 'https://parallelcluster-021345abcdef6789-v1-do-
not-delete...},
'cluster_name': 'test-api-cluster',
'cluster_status': 'DELETE_IN_PROGRESS',
'compute_fleet_status': 'UNKNOWN',
'creation_time': datetime.datetime(2022, 4, 28, 16, 50, 47, 943000,
 tzinfo=tzlocal()),
'last_updated_time': datetime.datetime(2022, 4, 28, 16, 50, 47, 943000,
 tzinfo=tzlocal()),
'region': 'us-east-1',
'tags': [{'key': 'parallelcluster:version', 'value': '3.1.3'}],
'version': '3.1.3'}
. . . working . . . DELETE COMPLETE or Exception when calling describe_cluster:
 (404)
Reason: Not Found
        .
        .
        .
HTTP response body: {"message":"Cluster 'test-api-cluster' does not exist or
 belongs to an incompatible ParallelCluster major version."}
```

3. 测试完成后，取消设置环境变量。

```
$ unset AWS_SESSION_TOKEN
unset AWS_SECRET_ACCESS_KEY
unset AWS_ACCESS_KEY_ID
```

步骤 6：清理

您可以使用 AWS Management Console 或 AWS CLI 删除您的 API。

1. 在 AWS CloudFormation 控制台中，选择 API 堆栈，然后选择删除。
2. 如果使用的是 AWS CLI，请运行以下命令。

   使用 AWS CloudFormation。

```
$ aws cloudformation delete-stack --stack-name ${API_STACK_NAME}
```

使用创建集群 Slurm 会计

学习如何使用配置和创建集群 Slurm 会计。有关更多信息，请参阅 Slurm 用会计 AWS ParallelCluster。

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

在本教程中，您将使用CloudFormation 快速创建的模板 (us-east-1) 来创建适用于 MySQL 的无服务器数据库。Amazon Aurora该模板指示 CloudFormation 创建所有必要的组件，以便在集群所在的 VPC 上部署 Amazon Aurora 无服务器数据库。该模板还会为集群与数据库之间的连接创建基本的网络和安全配置。

> **Note**
>
> 从 3.3.0 版开始，支持 AWS ParallelCluster Slurm 使用集群配置参数 SlurmSettings/数据库进行核算。

> **Note**
>
> 快速创建模板用作一个示例。此模板并未涵盖所有可能的用例 Slurm 会计数据库服务器。您负责创建配置和容量适合您的生产工作负载的数据库服务器。

先决条件：

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 你有一个 A mazon EC2 密钥对。
- 您拥有具有运行 `pcluster` CLI 所需的权限的 IAM 角色。
- 您在其中部署快速创建模板的区域支持 Amazon Aurora MySQL Serverless v2。有关更多信息，请参阅适用于 Aurora MySQL 的 Aurora Serverless v2。

步骤 1：为创建 VPC 和子网 AWS ParallelCluster

要将提供的 CloudFormation 模板用于 Slurm 记账数据库，您必须准备好集群的 VPC。您可以手动设置，也可以在使用 AWS ParallelCluster 命令行界面配置和创建集群的过程中进行设置。如果您已经使用 AWS ParallelCluster，则可能已经具有可用来部署集群和数据库服务器的 VPC。

步骤 2：创建数据库堆栈

使用CloudFormation 快速创建模板 (us-east-1) 为以下对象创建数据库堆栈 Slurm 会计。该模板需要以下输入：

- 数据库服务器凭证，特别是管理员用户名和密码。
- Amazon Aurora 无服务器集群的大小。这取决于预期的集群负载。
- 网络参数，特别是目标 VPC 和子网或用于创建子网的 CIDR 块。

为您的数据库服务器选择适当的凭证和大小。对于网络选项，您必须使用 AWS ParallelCluster 集群部署到的同一个 VPC。您可以为数据库创建子网并将其作为输入传递给模板。或者，为两个子网提供两个不相交的 CIDR 块，然后让 CloudFormation 模板为 CIDR 块创建两个子网。确保 CIDR 块不与现有子网重叠。如果 CIDR 块与现有子网重叠，则无法创建堆栈。

创建数据库服务器需要几分钟时间。

步骤 3：使用创建集群 Slurm 已启用记账

提供的 CloudFormation 模板生成一个包含一些已定义输出的 CloudFormation 堆栈。从中 AWS Management Console，您可以在 CloudFormation 堆栈视图的 “输出” 选项卡中查看输出。要启用 Slurm 记账，其中一些输出必须在 AWS ParallelCluster 集群配置文件中使用：

- `DatabaseHost`：用于 `SlurmSettings`/`Database`/`Uri` 集群配置参数。
- `DatabaseAdminUser`：用于 `SlurmSettings`/`Database`/`UserName` 集群配置参数值。
- `DatabaseSecretArn`：用于 `SlurmSettings`/`Database`/`PasswordSecretArn` 集群配置参数。
- `DatabaseClientSecurityGroup`：这是 `HeadNode`/`Networking`/`SecurityGroups` 配置参数中定义的附加到集群头节点的安全组。

使用输出值更新您的集群配置文件 `Database` 参数。使用 `pcluster` CLI 创建集群。

```
$ pcluster create-cluster -n cluster-3.x -c path/to/cluster-config.yaml
```

创建集群后，您可以开始使用 Slurm 记账命令，例如`sacctmgr`或`sacct`。

使用外部集群创建集群 Slurmdbd 会计

学习如何使用外部配置和创建集群 Slurmdbd 会计。有关更多信息，请参阅 。Slurm 会计 AWS ParallelCluster。

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

AWS ParallelCluster 用户界面基于无服务器架构构建，你可以在其中使用它 AWS Free Tier 大多数情况下的类别。有关更多信息，请参阅 AWS ParallelCluster UI 成本。

在本教程中，您将使用 AWS CloudFormation 快速创建模板来创建必要的组件，以便在与集群相同的 VPC 上部署 Slurmdbd 实例。该模板会为集群与数据库之间的连接创建基本的联网和安全配置。

> **Note**
>
> 从开始`version 3.10.0`，AWS ParallelCluster 支持带有集群配置参数的外部 Slurmdbd。`SlurmSettings / ExternelSlurmdbd`

> **Note**
>
> 快速创建模板用作一个示例。此模板并不涵盖所有可能的使用案例。您负责创建一个外部 Slurmdbd，其配置和容量应适合您的生产工作负载。

先决条件：

- AWS ParallelCluster 已安装。
- AWS CLI 已安装并配置。
- 您有一个 Amazon Elastic Compute Cloud 密钥对。
- 您的 AWS Identity and Access Management 角色具有运行 `pcluster` CLI 所需的权限。
- 你有一个 Slurm 会计数据库。逐步完成创建教程 Slurm 会计数据库，请按照创建 Slurm 会计数据库堆栈中的步骤 1 和步骤 2 进行操作。

步骤 1：创建 Slurmdbd 堆栈

在本教程中，使用CloudFormation 快速创建模板 (us-east-1) 创建 Slurmdbd 堆栈。该模板需要以下输入：

网络连接

- VPCId：用于启动 Slurmdbd 实例的 VPC ID。
- SubnetId：用于启动 Slurmdbd 实例的子网 ID。
- PrivatePrefix：VPC 的 CIDR 前缀。
- PrivateIp：要分配给 Slurmdbd 实例的辅助私有 IP。

数据库连接

- DBMSClientSG：要附加到 Slurmdbd 实例的安全组。该安全组应允许数据库服务器和 Slurmdbd 实例之间的连接。
- DBMSDatabase名称：数据库的名称。
- DBMSUsername：数据库的用户名。
- DBMSPasswordSecretArn：包含数据库密码的密钥。
- DBMSUri：数据库服务器的 URI。

实例设置

- InstanceType：用于 slurmdbd 实例的实例类型。
- KeyName：用于 slurmdbd 实例的亚马逊密 EC2 钥对。

Slurmdbd 设置

- AMIID：Slurmdbd 实例的 AMI。AMI 应该是 ParallelCluster AMI。 ParallelCluster AMI 的版本决定了 Slurmdbd 的版本。
- MungeKeySecretArn: 包含 munge 密钥的密钥，用于对 Slurmdbd 和集群之间的通信进行身份验证。
- SlurmdbdPort: slurmdbd 使用的端口号。
- EnableSlurmdbdSystemService：启用 slurmdbd 作为系统服务，让它在实例启动时运行。

> **Warning**
>
> 如果数据库是由不同版本的创建的 SlurmDB，请勿使用 Slurmdbd 作为系统服务。
> 如果数据库包含大量条目，则 Slurm Database Daemon (SlurmDBD) 可能需要几十分钟才能更新数据库，并且在此时间间隔内没有响应。
> 升级之前 SlurmDB，对数据库进行备份。有关更多信息，请参阅 。Slurm 文档。

步骤 2：使用外部集群创建集群 Slurmdbd 已启用

提供的 AWS CloudFormation 模板会生成一个包含一些已定义输出的 AWS CloudFormation 堆栈。

从中 AWS Management Console，查看 AWS CloudFormation 堆栈中的 “输出” 选项卡，查看创建的实体。要启用 Slurm 记账，其中一些输出必须在 AWS ParallelCluster 配置文件中使用：

- SlurmdbdPrivateIp: 用于 SlurmSettings/ExternalSlurmdbd/主机集群配置参数。
- SlurmdbdPort：用于 SlurmSettings/ ExternalSlurmdbd/P ort 集群配置参数值。
- AccountingClientSecurityGroup：这是附加到群集头节点的安全组，在 /Networking HeadNode/配置参数中定义AdditionalSecurityGroups。

此外，在 AWS CloudFormation 堆栈视图的 “参数” 选项卡中：

- MungeKeySecretArn: 用于 SlurmSettings/cl MungeKeySecretArnuster 配置参数值。

使用输出值更新集群配置文件数据库参数。使用 pcluster AWS CLI 创建集群。

```
$  pcluster create-cluster -n cluster-3.x-c path/to/cluster-config.yaml
```

创建集群后，您可以开始使用 Slurm 记账命令，例如`sacctmgr`或`sacct`。

> **Warning**
>
> 和外部`ParallelCluster`之间的流量 SlurmDB 未加密。建议运行集群和外部集群 SlurmDB 在可信的网络中。

恢复到以前的 S AWS ystems Manager 文档版本

了解如何恢复到以前的 S AWS ystems Manager 文档版本。有关 SSM 文档的更多信息，请参阅 AWS Systems Manager 用户指南 中的 AWS Systems Manager 文档。

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

先决条件：

- AWS 账户 具有管理 SSM 文档的权限。
- AWS CLI 已安装并配置。

恢复到以前的 SSM 文档版本

1. 在您的终端中，运行以下命令以获取您拥有的现有 SSM 文档的列表。

```
$ aws ssm list-documents --document-filter "key=Owner,value=Self"
```

2. 将一个 SSM 文档恢复到以前的版本。在此示例中，我们将 `SessionManagerRunShell` 文档恢复到以前版本。您可以使用 SSM `SessionManagerRunShell` 文档自定义您启动的每个 SSM Shell会话。

   a. 通过运行以下命令找到 `SessionManagerRunShell` 的 `DocumentVersion` 参数：

```
$ aws ssm describe-document --name "SSM-SessionManagerRunShell"
{
    "Document": {
        "Hash": "...",
        "HashType": "Sha256",
        "Name": "SSM-SessionManagerRunShell",
        "Owner": "123456789012",
        "CreatedDate": "2023-02-20T19:04:32.390000+00:00",
        "Status": "Active",
        "DocumentVersion": "1",
        "Parameters": [
            {
                "Name": "linuxcmd",
                "Type": "String",
                "Description": "The command to run on connection...",
```

```
                "DefaultValue": "if [ -d '/opt/parallelcluster' ]; then
 source /opt/parallelcluster/cfnconfig; sudo su - $cfn_cluster_user; fi; /bin/
bash"
            }
        ],
        "PlatformTypes": [
            "Windows",
            "Linux",
            "MacOS"
        ],
        "DocumentType": "Session",
        "SchemaVersion": "1.0",
        "LatestVersion": "2",
        "DefaultVersion": "1",
        "DocumentFormat": "JSON",
        "Tags": []
    }
}
```

最新版本为 2。

b. 通过运行以下命令，恢复到以前的版本：

```
$ aws ssm delete-document --name "SSM-SessionManagerRunShell" --document-
version 2
```

3. 再次运行 `describe-document` 命令，验证文档版本是否已恢复：

```
$ aws ssm describe-document --name "SSM-SessionManagerRunShell"
{
    "Document": {
        "Hash": "...",
        "HashType": "Sha256",
        "Name": "SSM-SessionManagerRunShell",
        "Owner": "123456789012",
        "CreatedDate": "2023-02-20T19:04:32.390000+00:00",
        "Status": "Active",
        "DocumentVersion": "1",
        "Parameters": [
            {
                "Name": "linuxcmd",
                "Type": "String",
                "Description": "The command to run on connection...",
```

```
                "DefaultValue": "if [ -d '/opt/parallelcluster' ]; then source /
opt/parallelcluster/cfnconfig; sudo su - $cfn_cluster_user; fi; /bin/bash"
            }
        ],
        "PlatformTypes": [
            "Windows",
            "Linux",
            "MacOS"
        ],
        "DocumentType": "Session",
        "SchemaVersion": "1.0",
        "LatestVersion": "1",
        "DefaultVersion": "1",
        "DocumentFormat": "JSON",
        "Tags": []
    }
}
```

最新版本为 1。

使用创建集群 AWS CloudFormation

学习如何使用 AWS ParallelCluster CloudFormation 自定义资源创建集群。有关更多信息，请参阅 AWS CloudFormation 自定义资源。

使用时 AWS ParallelCluster，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 AWS 使用的服务 AWS ParallelCluster。

先决条件：

- AWS CLI 已安装并配置。
- 亚马逊 EC2 密钥对。
- 具有运行 `pcluster` CLI 所需的权限的 IAM 角色。

使用 CloudFormation 快速创建堆栈创建集群

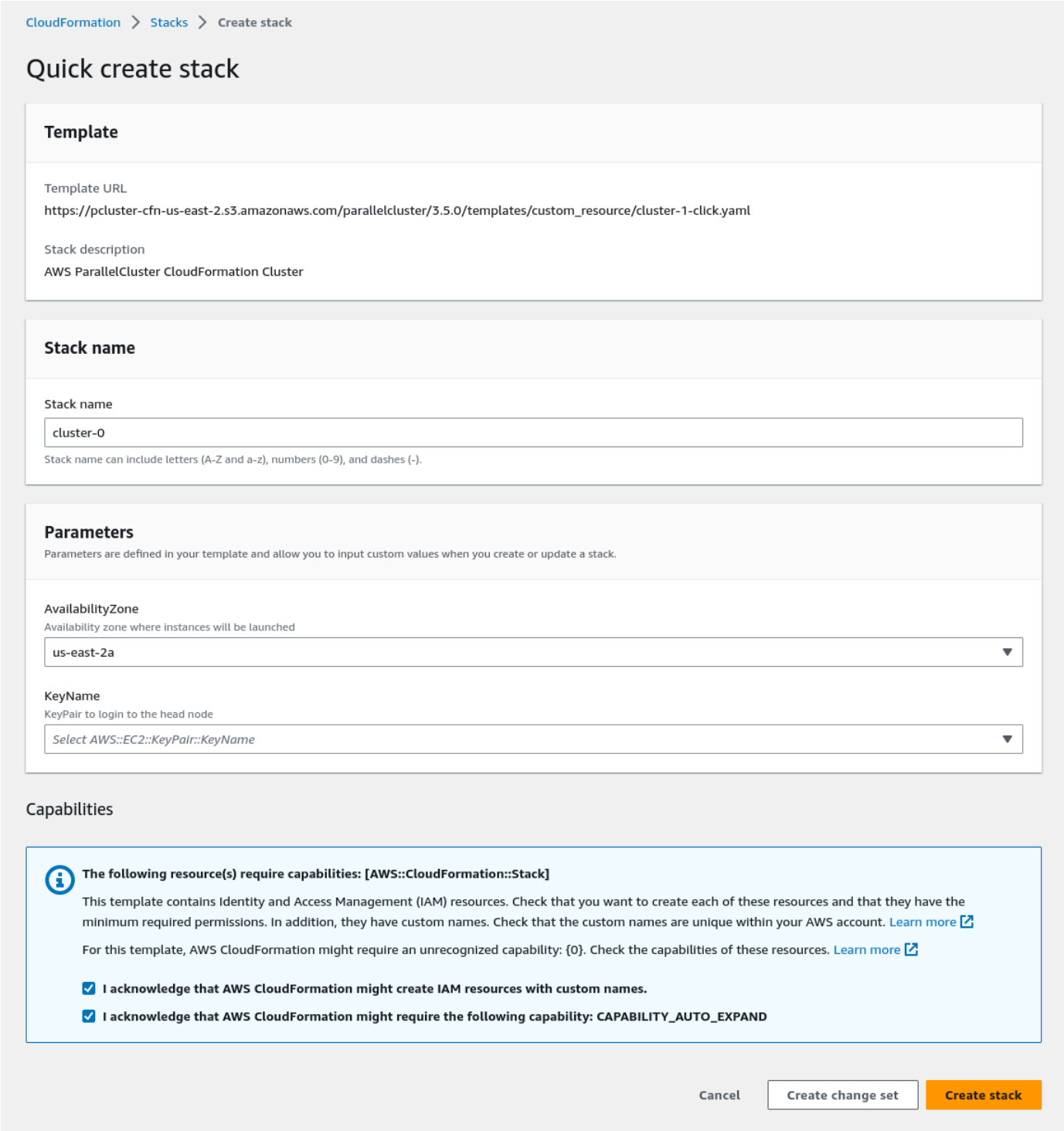
在本教程中，您将使用快速创建堆栈来部署用于创建集群的 CloudFormation 模板和以下 AWS 资源：

- 使用 CloudFormation 快速创建 CloudFormation 堆栈创建的根堆栈。

- 嵌套 CloudFormation 堆栈，包括默认策略、默认 VPC 设置和自定义资源提供程序。
- 示例 AWS ParallelCluster 集群堆栈和您可以登录并运行作业的集群。

使用创建集群 AWS CloudFormation

1. 登录到 AWS Management Console。
2. 打开 CloudFormation 快速创建链接，在 CloudFormation 控制台中创建以下资源：
   - 带有 VPC 的嵌套 CloudFormation 堆栈，其中包含公有子网和私有子网，分别用于运行集群头节点和计算节点。
   - 带有用于管理群集的 AWS ParallelCluster 自定义资源的嵌套 CloudFormation 堆栈。
   - 带有用于管理集群的默认策略的嵌套 CloudFormation 堆栈。
   - 嵌套 CloudFormation 堆栈的根堆栈。
   - 具有Slurm调度程序和已定义数量的计算节点的 AWS ParallelCluster 集群。



3. 在快速创建堆栈参数部分，输入以下参数的值：

   a. 对于 KeyName，请输入您的 Amazon EC2 密钥对的名称。

   b. 对于 AvailabilityZone，为集群节点选择可用区，例如`us-east-1a`。

4. 在页面底部选中各个框以确认各项访问功能。
5. 选择创建堆栈。
6. 等待 CloudFormation 堆栈达到`CREATE_COMPLETE`状态。

使用 AWS CloudFormation 命令行界面 (CLI) 创建集群

在本教程中，您将使用用于的 AWS 命令行界面 (CLI) CloudFormation 来部署用于创建集群的 CloudFormation 模板。

创建以下 AWS 资源：

- 使用 CloudFormation 快速创建 CloudFormation 堆栈创建的根堆栈。
- 嵌套 CloudFormation 堆栈，包括默认策略、默认 VPC 设置和自定义资源提供程序。
- 示例 AWS ParallelCluster 集群堆栈和您可以登录并运行作业的集群。

用您自己的值替换*inputs highlighted in red*（例如*keypair*）。

使用创建集群 AWS CloudFormation

1. 使用以下内容创建`cluster_template.yaml`名为的 CloudFormation 模板：

```
AWSTemplateFormatVersion: '2010-09-09'
Description: > AWS ParallelCluster CloudFormation Template

Parameters:
  KeyName:
    Description: KeyPair to login to the head node
    Type: AWS::EC2::KeyPair::KeyName

  AvailabilityZone:
    Description: Availability zone where instances will be launched
    Type: AWS::EC2::AvailabilityZone::Name
    Default: us-east-2a

Mappings:
  ParallelCluster:
    Constants:
      Version: 3.13.2

Resources:
```

```
  PclusterClusterProvider:
    Type: AWS::CloudFormation::Stack
    Properties:
      TemplateURL: !Sub
        - https://${AWS::Region}-aws-parallelcluster.s3.${AWS::Region}.
${AWS::URLSuffix}/parallelcluster/${Version}/templates/custom_resource/cluster.yaml
        - { Version: !FindInMap [ParallelCluster, Constants, Version] }

  PclusterVpc:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        PublicCIDR: 10.0.0.0/24
        PrivateCIDR: 10.0.16.0/20
        AvailabilityZone: !Ref AvailabilityZone
      TemplateURL: !Sub
        - https://${AWS::Region}-aws-parallelcluster.s3.${AWS::Region}.
${AWS::URLSuffix}/parallelcluster/${Version}/templates/networking/public-private-
${Version}.cfn.json
        - { Version: !FindInMap [ParallelCluster, Constants, Version ] }

  PclusterCluster:
    Type: Custom::PclusterCluster
    Properties:
      ServiceToken: !GetAtt [ PclusterClusterProvider , Outputs.ServiceToken ]
      ClusterName: !Sub 'c-${AWS::StackName}'
      ClusterConfiguration:
        Image:
          Os: alinux2
        HeadNode:
          InstanceType: t2.medium
          Networking:
            SubnetId: !GetAtt [ PclusterVpc , Outputs.PublicSubnetId ]
          Ssh:
            KeyName: !Ref KeyName
        Scheduling:
          Scheduler: slurm
          SlurmQueues:
          - Name: queue0
            ComputeResources:
            - Name: queue0-cr0
              InstanceType: t2.micro
            Networking:
              SubnetIds:
```

```
              -  !GetAtt [ PclusterVpc , Outputs.PrivateSubnetId ]
Outputs:
  HeadNodeIp:
    Description: The Public IP address of the HeadNode
    Value: !GetAtt [ PclusterCluster, headNode.publicIpAddress ]
```

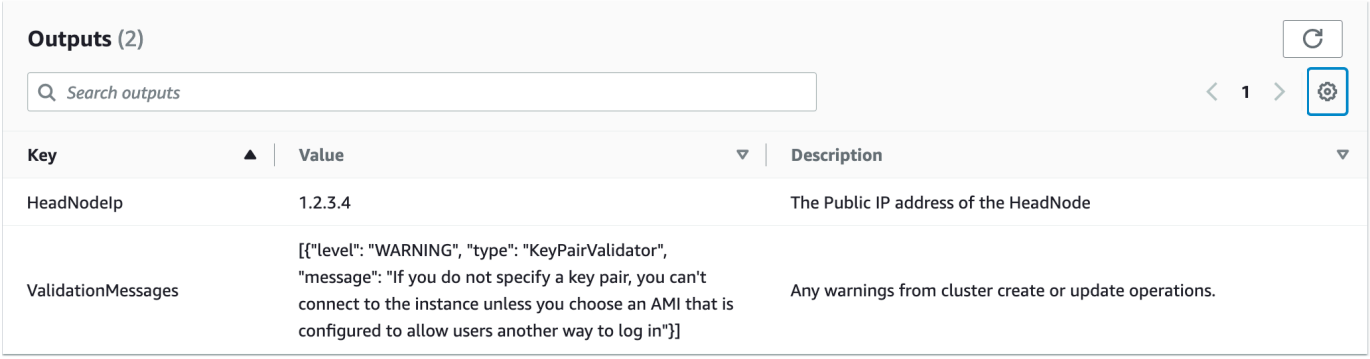
2. 运行以下 AWS CLI 命令部署 CloudFormation 堆栈以进行集群创建和管理。

```
$ aws cloudformation deploy --template-file ./cluster_template.yaml \
  --stack-name mycluster \
  --parameter-overrides KeyName=keypair \
                        AvailabilityZone=us-east-2b \
  --capabilities CAPABILITY_NAMED_IAM CAPABILITY_AUTO_EXPAND
```

查看 CloudFormation 集群输出

查看集 CloudFormation 群输出以获取有用的集群详细信息。添加的 `ValidationMessages` 属性允许访问集群创建和更新操作中的验证消息。

1. 导航到CloudFormation 控制台并选择包含您的 AWS ParallelCluster 自定义资源的堆栈。
2. 选择堆栈详细信息，然后选择输出选项卡。



验证消息可能会被截断。有关如何检索日志的更多信息，请参阅AWS ParallelCluster 故障排除。

访问您的集群

访问集群。

ssh 登录到集群头节点

1. CloudFormation 堆栈部署完成后，使用以下命令获取头节点的 IP 地址：

```
$ HEAD_NODE_IP=$(aws cloudformation describe-stacks --stack-name=mycluster --query
 "Stacks|[0].Outputs[?OutputKey=='HeadNodeIp']|[0].OutputValue" --output=text)
```

您也可以从 CloudFormation 控制台集群堆栈输出选项卡中的HeadNodeIp参数中检索头节点 IP 地址。

您可以在此处找到头节点 IP 地址，因为它是在集群 CloudFormation 模板的`Outputs`部分中添加的，专门针对此示例集群。

2. 通过运行以下命令，连接集群头节点：

```
$ ssh -i keyname.pem ec2-user@$HEAD_NODE_IP
```

清理

请删除集群。

1. 运行以下 AWS CLI 命令删除 CloudFormation 堆栈和集群。

```
$ aws cloudformation delete-stack --stack-name=mycluster
```

2. 通过运行以下命令，检查堆栈删除状态。

```
$ aws cloudformation describe-stacks --stack-name=mycluster
```

使用 Terraform 部署 ParallelCluster API

在本教程中，您将定义一个简单的 Terraform 项目来部署 API。 ParallelCluster

先决条件

- Terraform v1.5.7+ 已安装。
- 具有部署 ParallelCluster API 权限的 IAM 角色。请参阅the section called “所需的权限”。

定义 Terraform 项目

在本教程中，您将定义一个 Terraform 项目。

1. 创建名为 `my-pcluster-api` 的目录。

   您创建的所有文件都将位于此目录中。

2. 创建用于配置`provider.tf` AWS 提供程序的文件。

```
provider "aws" {
  region  = var.region
  profile = var.profile
}
```

3. 使用 ParallelCluster模块创建文件`main.tf`以定义资源。

```
module "parallelcluster_pcluster_api" {
  source = "aws-tf/parallelcluster/aws//modules/pcluster_api"
  version = "1.1.0"

  region                = var.region
  api_stack_name        = var.api_stack_name
  api_version           = var.api_version

  parameters = {
    EnableIamAdminAccess = "true"
  }
}
```

4. 创建 `variables.tf` 文件来定义可以为此项目注入的变量。

```
variable "region" {
  description = "The region the ParallelCluster API is deployed in."
  type        = string
  default     = "us-east-1"
}

variable "profile" {
  type        = string
  description = "The AWS profile used to deploy the clusters."
  default     = null
}

variable "api_stack_name" {
  type        = string
  description = "The name of the CloudFormation stack used to deploy the
 ParallelCluster API."
```

```
  default     = "ParallelCluster"
}

variable "api_version" {
  type        = string
  description = "The version of the ParallelCluster API."
}
```

5. 创建 `terraform.tfvars` 文件来设置变量的任意值。

   下面的文件`us-east-1`使用堆栈名称部署了 ParallelCluster API 3.11.1。`MyParallelClusterAPI-3111`您将能够使用其堆栈名称来引用此 ParallelCluster API 部署。

   > **Note**
   >
   > 以下代码中的`api_version`赋值可以替换为任何支持的 AWS ParallelCluster 版本。

```
region = "us-east-1"
api_stack_name = "MyParallelClusterAPI-3111"
api_version = "3.11.1"
```

6. 创建 `outputs.tf` 文件来定义此项目返回的输出。

```
output "pcluster_api_stack_outputs" {
  value = module.parallelcluster_pcluster_api.stack_outputs
}
```

   项目目录为：

```
my-pcluster-api
### main.tf - Terraform entrypoint to define the resources using the
 ParallelCluster module.
### outputs.tf - Defines the outputs returned by Terraform.
### providers.tf - Configures the AWS provider.
### terraform.tfvars - Set the arbitrary values for the variables, i.e. region,
 PCAPI version, PCAPI stack name
### variables.tf - Defines the variables, e.g. region, PCAPI version, PCAPI stack
 name.
```

部署 API

要部署 API，请按顺序运行标准的 Terraform 命令。

1. 构建项目：

```
terraform init
```

2. 定义部署计划：

```
terraform plan -out tfplan
```

3. 部署计划：

```
terraform apply tfplan
```

所需的权限

你需要以下权限才能使用 Terraform 部署 ParallelCluster API：

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": [
                "cloudformation:DescribeStacks",
                "cloudformation:GetTemplate"
            ],
            "Resource": "arn:PARTITION:cloudformation:REGION:ACCOUNT:stack/*",
            "Effect": "Allow",
            "Sid": "CloudFormationRead"
        },
        {
            "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:CreateChangeSet"
            ],
            "Resource": "arn:PARTITION:cloudformation:REGION:ACCOUNT:stack/
MyParallelClusterAPI*",
            "Effect": "Allow",
```

```
            "Sid": "CloudFormationWrite"
        },
        {
            "Action": [
                "cloudformation:CreateChangeSet"
            ],
            "Resource": [
                "arn:PARTITION:cloudformation:REGION:aws:transform/Include",
                "arn:PARTITION:cloudformation:REGION:aws:transform/
Serverless-2016-10-31"
            ],
            "Effect": "Allow",
            "Sid": "CloudFormationTransformWrite"
        },
        {
            "Action": [
                "s3:GetObject"
            ],
            "Resource": [
                "arn:PARTITION:s3:::*-aws-parallelcluster/parallelcluster/*/api/
ParallelCluster.openapi.yaml",
                "arn:PARTITION:s3:::*-aws-parallelcluster/parallelcluster/*/layers/aws-
parallelcluster/lambda-layer.zip"
            ],
            "Effect": "Allow",
            "Sid": "S3ParallelClusterArtifacts"
        },
        {
            "Action": [
                "iam:CreateRole",
                "iam:DeleteRole",
                "iam:GetRole",
                "iam:CreatePolicy",
                "iam:DeletePolicy",
                "iam:GetPolicy",
                "iam:GetRolePolicy",
                "iam:AttachRolePolicy",
                "iam:DetachRolePolicy",
                "iam:PutRolePolicy",
                "iam:DeleteRolePolicy",
                "iam:ListPolicyVersions"
            ],
            "Resource": [
                "arn:PARTITION:iam::ACCOUNT:role/*",
```

```
                "arn:PARTITION:iam::ACCOUNT:policy/*"
            ],
            "Effect": "Allow",
            "Sid": "IAM"
        },
        {
            "Action": [
                "iam:PassRole"
            ],
            "Resource": [
                "arn:PARTITION:iam::ACCOUNT:role/ParallelClusterLambdaRole-*",
                "arn:PARTITION:iam::ACCOUNT:role/APIGatewayExecutionRole-*"
            ],
            "Effect": "Allow",
            "Sid": "IAMPassRole"
        },
        {
            "Action": [
                "lambda:CreateFunction",
                "lambda:DeleteFunction",
                "lambda:GetFunction",
                "lambda:PublishLayerVersion",
                "lambda:DeleteLayerVersion",
                "lambda:GetLayerVersion",
                "lambda:TagResource",
                "lambda:UntagResource"
            ],
            "Resource": [
                "arn:PARTITION:lambda:REGION:ACCOUNT:layer:PCLayer-*",
                "arn:PARTITION:lambda:REGION:ACCOUNT:function:*-
ParallelClusterFunction-*"
            ],
            "Effect": "Allow",
            "Sid": "Lambda"
        },
        {
            "Action": [
                "logs:CreateLogGroup",
                "logs:DeleteLogGroup",
                "logs:DescribeLogGroups",
                "logs:PutRetentionPolicy",
                "logs:TagLogGroup",
                "logs:UntagLogGroup"
            ],
```

```
            "Resource": [
                "arn:PARTITION:logs:REGION:ACCOUNT:log-group:/aws/lambda/*-
ParallelClusterFunction-*"
            ],
            "Effect": "Allow",
            "Sid": "Logs"
        },
        {
            "Action": [
                "apigateway:DELETE",
                "apigateway:GET",
                "apigateway:PATCH",
                "apigateway:POST",
                "apigateway:PUT",
                "apigateway:UpdateRestApiPolicy"
            ],
            "Resource": [
                "arn:PARTITION:apigateway:REGION::/restapis",
                "arn:PARTITION:apigateway:REGION::/restapis/*",
                "arn:PARTITION:apigateway:REGION::/tags/*"
            ],
            "Effect": "Allow",
            "Sid": "APIGateway"
        }
    ]
}
```

使用 Terraform 创建集群

使用时 AWS ParallelCluster，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 the section called “AWS 使用的服务 AWS ParallelCluster”。

先决条件

- Terraform v1.5.7+ 已安装。
- the section called “AWS ParallelCluster API” v3.8.0+ 已部署在您的账户中。请参阅the section called “使用 Terraform 部署 ParallelCluster API”。
- 具有调用 ParallelCluster API 权限的 IAM 角色。参见 [所需权限]

定义 Terraform 项目

在本教程中，您将定义一个简单的 Terraform 项目来部署集群。

1. 创建名为 `my-clusters` 的目录。

   您创建的所有文件都将位于此目录中。

2. 创建文件`terraform.tf`以导入 ParallelCluster 提供程序。

```
terraform {
  required_version = ">= 1.5.7"
  required_providers {
    aws-parallelcluster = {
      source  = "aws-tf/aws-parallelcluster"
      version = "~> 1.0"
    }
  }
}
```

3. 创建用于配置 ParallelCluster 和 AWS 提供程序的文件`providers.tf`。

```
provider "aws" {
  region  = var.region
  profile = var.profile
}

provider "aws-parallelcluster" {
  region         = var.region
  profile        = var.profile
  api_stack_name = var.api_stack_name
  use_user_role  = true
}
```

4. 使用 ParallelCluster模块创建文件`main.tf`以定义资源。

```
module "pcluster" {
  source  = "aws-tf/parallelcluster/aws"
  version = "1.1.0"

  region                = var.region
  api_stack_name        = var.api_stack_name
  api_version           = var.api_version
```

```
  deploy_pcluster_api   = false

  template_vars         = local.config_vars
  cluster_configs       = local.cluster_configs
  config_path           = "config/clusters.yaml"
}
```

5. 创建 `clusters.tf` 文件来将多个集群定义为 Terraform 局部变量。

> **Note**
>
> 可以在 `cluster_config` 元素中定义多个集群。对于每个集群，您都可以在局部变量中显式定义集群属性（见 `DemoCluster01`）或引用外部文件（见 `DemoCluster02`）。

要查看可在配置元素中设置的集群属性，请参阅the section called “集群配置文件”。

要查看创建集群时可设置的选项，请参阅 the section called “`pcluster create-cluster`”。

```
locals {
  cluster_configs = {
    DemoCluster01 : {
      region : local.config_vars.region
      rollbackOnFailure : false
      validationFailureLevel : "WARNING"
      suppressValidators : [
        "type:KeyPairValidator"
      ]
      configuration : {
        Region : local.config_vars.region
        Image : {
          Os : "alinux2"
        }
        HeadNode : {
          InstanceType : "t3.small"
          Networking : {
            SubnetId : local.config_vars.subnet
          }
          Iam : {
            AdditionalIamPolicies : [
              { Policy : "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" }
            ]
```

```
          }
        }
        Scheduling : {
          Scheduler : "slurm"
          SlurmQueues : [{
            Name : "queue1"
            CapacityType : "ONDEMAND"
            Networking : {
              SubnetIds : [local.config_vars.subnet]
            }
            Iam : {
              AdditionalIamPolicies : [
                { Policy : "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore" }
              ]
            }
            ComputeResources : [{
              Name : "compute"
              InstanceType : "t3.small"
              MinCount : "1"
              MaxCount : "4"
            }]
          }]
          SlurmSettings : {
            QueueUpdateStrategy : "TERMINATE"
          }
        }
      }
    }
    DemoCluster02 : {
      configuration : "config/cluster_config.yaml"
    }
  }
}
```

6. 创建 `config/clusters.yaml` 文件来将多个集群定义为 YAML 配置。

```
DemoCluster03:
  region: ${region}
  rollbackOnFailure: true
  validationFailureLevel: WARNING
  suppressValidators:
    - type:KeyPairValidator
  configuration: config/cluster_config.yaml
DemoCluster04:
```

```
  region: ${region}
  rollbackOnFailure: false
  configuration: config/cluster_config.yaml
```

7. 创建文件`config/cluster_config.yaml`，这是一个标准 ParallelCluster 配置文件，可以在其中注入 Terraform 变量。

   要查看可在配置元素中设置的集群属性，请参阅the section called “集群配置文件”。

```
Region: ${region}
Image:
 Os: alinux2
HeadNode:
 InstanceType: t3.small
 Networking:
   SubnetId: ${subnet}
 Iam:
   AdditionalIamPolicies:
     - Policy: arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
Scheduling:
 Scheduler: slurm
 SlurmQueues:
   - Name: queue1
     CapacityType: ONDEMAND
     Networking:
       SubnetIds:
         - ${subnet}
     Iam:
       AdditionalIamPolicies:
         - Policy: arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
     ComputeResources:
       - Name: compute
         InstanceType: t3.small
         MinCount: 1
         MaxCount: 5
 SlurmSettings:
   QueueUpdateStrategy: TERMINATE
```

8. 创建 `clusters_vars.tf` 文件来定义可以注入到集群配置中的变量。

   此文件使您能够定义可在集群配置中使用的动态值，例如区域和子网。

   此示例直接从项目变量中检索值，但您可能需要使用自定义逻辑来确定它们。

```
locals {
  config_vars = {
    subnet = var.subnet_id
    region = var.cluster_region
  }
}
```

9. 创建 `variables.tf` 文件来定义可以为此项目注入的变量。

```
variable "region" {
  description = "The region the ParallelCluster API is deployed in."
  type        = string
  default     = "us-east-1"
}

variable "cluster_region" {
  description = "The region the clusters will be deployed in."
  type        = string
  default     = "us-east-1"
}

variable "profile" {
  type        = string
  description = "The AWS profile used to deploy the clusters."
  default     = null
}

variable "subnet_id" {
  type        = string
  description = "The id of the subnet to be used for the ParallelCluster
 instances."
}

variable "api_stack_name" {
  type        = string
  description = "The name of the CloudFormation stack used to deploy the
 ParallelCluster API."
  default     = "ParallelCluster"
}

variable "api_version" {
  type        = string
```

```
  description = "The version of the ParallelCluster API."
}
```

10. 创建 `terraform.tfvars` 文件来设置变量的任意值。

    以下文件使用现有 ParallelCluster API 3.11.1 在子网`eu-west-1`中部署集群`subnet-123456789`，该API 3.11.1 已使用堆栈名称部署在子网中`us-east-1`。`MyParallelClusterAPI-3111`

```
region = "us-east-1"
api_stack_name = "MyParallelClusterAPI-3111"
api_version = "3.11.1"

cluster_region = "eu-west-1"
subnet_id = "subnet-123456789"
```

11. 创建 `outputs.tf` 文件来定义此项目返回的输出。

```
output "clusters" {
  value = module.pcluster.clusters
}
```

    项目目录为：

```
my-clusters
### config
#   ### cluster_config.yaml - Cluster configuration, where terraform variables can
 be injected..
#   ### clusters.yaml - File listing all the clusters to deploy.
### clusters.tf - Clusters defined as Terraform local variables.
### clusters_vars.tf - Variables that can be injected into cluster configurations.
### main.tf - Terraform entrypoint where the ParallelCluster module is configured.
### outputs.tf - Defines the cluster as a Terraform output.
### providers.tf - Configures the providers: ParallelCluster and AWS.
### terraform.tf - Import the ParallelCluster provider.
### terraform.tfvars - Defines values for variables, e.g. region, PCAPI stack name.
### variables.tf - Defines the variables, e.g. region, PCAPI stack name.
```

部署集群

要部署集群，请按顺序运行标准的 Terraform 命令。

> **Note**
>
> 此示例假设您已经在账户中部署了 ParallelCluster API。

1. 构建项目：

```
terraform init
```

2. 定义部署计划：

```
terraform plan -out tfplan
```

3. 部署计划：

```
terraform apply tfplan
```

使用集群部署 ParallelCluster API

如果您尚未部署 ParallelCluster API，但想将其与集群一起部署，请更改以下文件：

- `main.tf`

```
module "pcluster" {
  source  = "aws-tf/aws/parallelcluster"
  version = "1.0.0"

  region                = var.region
  api_stack_name        = var.api_stack_name
  api_version           = var.api_version
  deploy_pcluster_api   = true
  parameters = {
    EnableIamAdminAccess = "true"
  }

  template_vars         = local.config_vars
  cluster_configs       = local.cluster_configs
  config_path           = "config/clusters.yaml"
}
```

- `providers.tf`

```
provider "aws-parallelcluster" {
  region   = var.region
  profile  = var.profile
  endpoint = module.pcluster.pcluster_api_stack_outputs.ParallelClusterApiInvokeUrl
  role_arn = module.pcluster.pcluster_api_stack_outputs.ParallelClusterApiUserRole
}
```

所需的权限

使用 Terraform 部署集群需要以下权限：

- 担任 ParallelCluster API 角色，该角色负责与 ParallelCluster API 交互
- 描述 ParallelCluster API 的 AWS CloudFormation 堆栈以验证其存在并检索其参数和输出

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Action": "sts:AssumeRole",
            "Resource": "arn:PARTITION:iam::ACCOUNT:role/PCAPIUserRole-*",
            "Effect": "Allow",
            "Sid": "AssumePCAPIUserRole"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks"
            ],
            "Resource": "arn:PARTITION:cloudformation:REGION:ACCOUNT:stack/*",
            "Effect": "Allow",
            "Sid": "CloudFormation"
        }
    ]
}
```

使用 Terraform 创建自定义 AMI

使用时 AWS ParallelCluster，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 the section called “AWS 使用的服务 AWS ParallelCluster”。

先决条件

- Terraform v1.5.7+ 已安装。
- the section called “AWS ParallelCluster API” v3.8.0+ 已部署在您的账户中。请参阅the section called “使用 Terraform 创建集群”。
- 具有调用 ParallelCluster API 权限的 IAM 角色。请参阅the section called “所需的权限”。

定义 Terraform 项目

在本教程中，您将定义一个简单的 Terraform 项目来部署自定义 ParallelCluster AMI。

1. 创建名为 `my-amis` 的目录。

   您创建的所有文件都将位于此目录中。

2. 创建文件`terraform.tf`以导入 ParallelCluster 提供程序。

```
terraform {
  required_version = ">= 1.5.7"
  required_providers {
    aws-parallelcluster = {
      source  = "aws-tf/aws-parallelcluster"
      version = "~> 1.0"
    }
  }
}
```

3. 创建用于配置 ParallelCluster 和 AWS 提供程序的文件`providers.tf`。

```
provider "aws" {
  region  = var.region
  profile = var.profile
}

provider "aws-parallelcluster" {
  region         = var.region
  profile        = var.profile
  api_stack_name = var.api_stack_name
  use_user_role  = true
}
```

4. 使用 ParallelCluster模块创建文件`main.tf`以定义资源。

要查看可在 `image_configuration` 元素中设置的映像属性，请参阅the section called “构建映像配置文件”。

要查看创建映像时可设置的选项（如 `image_id` 和 `rollback_on_failure`），请参阅 the section called “`pcluster build-image`”。

```
data "aws-parallelcluster_list_official_images" "parent_image" {
  region = var.region
  os = var.os
  architecture = var.architecture
}

resource "aws-parallelcluster_image" "demo01" {
  image_id            = "demo01"
  image_configuration = yamlencode({
    "Build":{
      "InstanceType": "c5.2xlarge",
      "ParentImage": data.aws-
parallelcluster_list_official_images.parent_image.official_images[0].amiId,
      "UpdateOsPackages": {"Enabled": false}
    }
  })
  rollback_on_failure = false
}
```

5. 创建 `variables.tf` 文件来定义可以为此项目注入的变量。

```
variable "region" {
  description = "The region the ParallelCluster API is deployed in."
  type        = string
  default     = "us-east-1"
}

variable "profile" {
  type        = string
  description = "The AWS profile used to deploy the clusters."
  default     = null
}

variable "api_stack_name" {
  type        = string
```

```
  description = "The name of the CloudFormation stack used to deploy the
 ParallelCluster API."
  default     = "ParallelCluster"
}

variable "api_version" {
  type        = string
  description = "The version of the ParallelCluster API."
}

variable "os" {
  type        = string
  description = "The OS of the ParallelCluster image."
}

variable "architecture" {
  type        = string
  description = "The architecture of the ParallelCluster image."
}
```

6. 创建文件`terraform.tfvars`以设置变量的任意值。

   在下面的文件中，使用已经使用堆栈名称部署的现有 ParallelCluster API 3.11.1，`us-east-1`基于适用于 x86_64 架构的 Amazon Linux 2 部署自定义 AMI。`us-east-1` `MyParallelClusterAPI-3111`

```
region = "us-east-1"
api_stack_name = "MyParallelClusterAPI-3111"
api_version = "3.11.1"

os = "alinux2"
architecture = "x86_64"
```

7. 创建 `outputs.tf` 文件来定义此项目返回的输出。

```
output "parent_image" {
  value = data.aws-
parallelcluster_list_official_images.parent_image.official_images[0]
}

output "custom_image" {
  value = aws-parallelcluster_image.demo01
```

```
}
```

项目目录为：

```
my-amis
### main.tf - Terraform entrypoint where the ParallelCluster module is configured.
### outputs.tf - Defines the cluster as a Terraform output.
### providers.tf - Configures the providers: ParallelCluster and AWS.
### terraform.tf - Import the ParallelCluster provider.
### terraform.tfvars - Defines values for variables, e.g. region, PCAPI stack name.
### variables.tf - Defines the variables, e.g. region, PCAPI stack name.
```

部署 AMI

要部署 AMI，请按顺序运行标准的 Terraform 命令。

1. 构建项目：

```
terraform init
```

2. 定义部署计划：

```
terraform plan -out tfplan
```

3. 部署计划：

```
terraform apply tfplan
```

所需的权限

使用 Terraform 部署自定义 AMI 需要以下权限：

- 担任 ParallelCluster API 角色，该角色负责与 ParallelCluster API 交互
- 描述 ParallelCluster API 的 AWS CloudFormation 堆栈，以验证其存在并检索其参数和输出

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

```
        {
            "Action": "sts:AssumeRole",
            "Resource": "arn:PARTITION:iam::ACCOUNT:role/PCAPIUserRole-*",
            "Effect": "Allow",
            "Sid": "AssumePCAPIUserRole"
        },
        {
            "Action": [
                "cloudformation:DescribeStacks"
            ],
            "Resource": "arn:PARTITION:cloudformation:REGION:ACCOUNT:stack/*",
            "Effect": "Allow",
            "Sid": "CloudFormation"
        }
    ]
}
```

AWS ParallelCluster 用户界面与身份中心集成

本教程的目标是演示如何将用户 AWS ParallelCluster 界面与 IAM Identity Center 集成，以实现单点登录解决方案，该解决方案可统一可与 AWS ParallelCluster 集群共享的 Active Directory 中的用户。

使用时 AWS ParallelCluster，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 AWS 使用的服务 AWS ParallelCluster。

先决条件：

- 现有的 AWS ParallelCluster UI，可按照此处的说明进行安装。
- 现有的托管 Active Directory，最好是同时用于与之集成的目录 AWS ParallelCluster。

启用 IAM Identity Center

如果您已经将身份中心连接到您的 AWS Managed Microsoft AD（Active Directory），则可以使用该身份中心，并且可以跳至将您的应用程序添加到 IAM 身份中心部分。

如果您尚未将身份中心连接到 AWS Managed Microsoft AD，请按照以下步骤进行设置。

启用 Identity Center

1. 在控制台中，导航至 IAM Identity Center。（请确保您位于您所在的地区 AWS Managed Microsoft AD。）

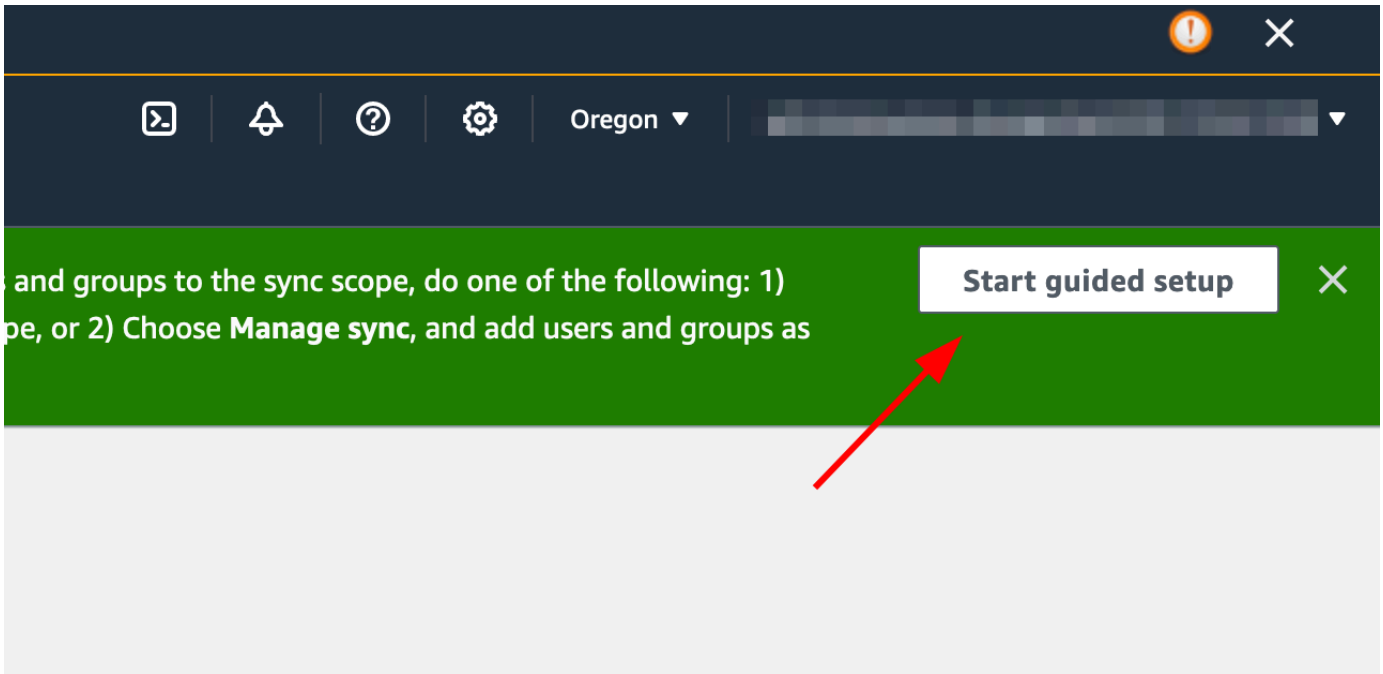
2. 单击启用按钮，此时系统可能会询问您是否要启用组织，这是一项要求，因此您可以选择启用。注：这将给您的账户管理员发送一封确认电子邮件，您应点击链接进行确认。

将 Identity Center 连接到 Managed AD

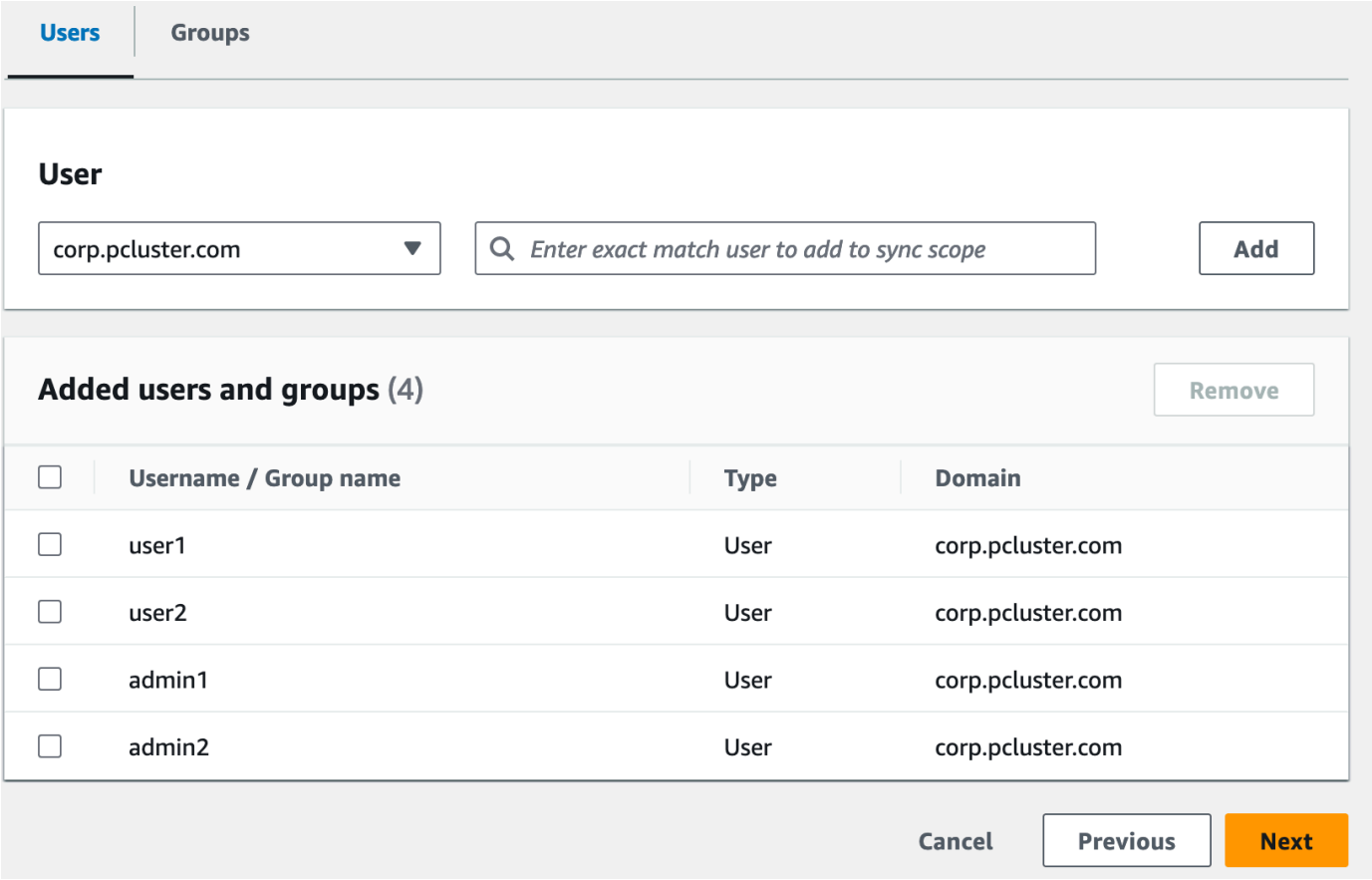
1. 在启用 Identity Center 后的下一页，您会看到“建议设置步骤”，在步骤 1 下，选择选择您的身份源。
2. 在“身份源”部分，单击操作下拉菜单（右上角），然后选择更改身份源。
3. 选择 Active Directory。
4. 在现有目录下，选择您的目录。
5. 单击下一步。
6. 查看您的更改，滚动到底部，在文本框中键入 ACCEPT 进行确认，然后单击更改身份源。
7. 等待更改完成，然后您应该会在顶部看到一个绿色横幅。

将用户和组同步到 Identity Center

1. 在绿色横幅中，单击开始引导式设置（右上角的按钮）

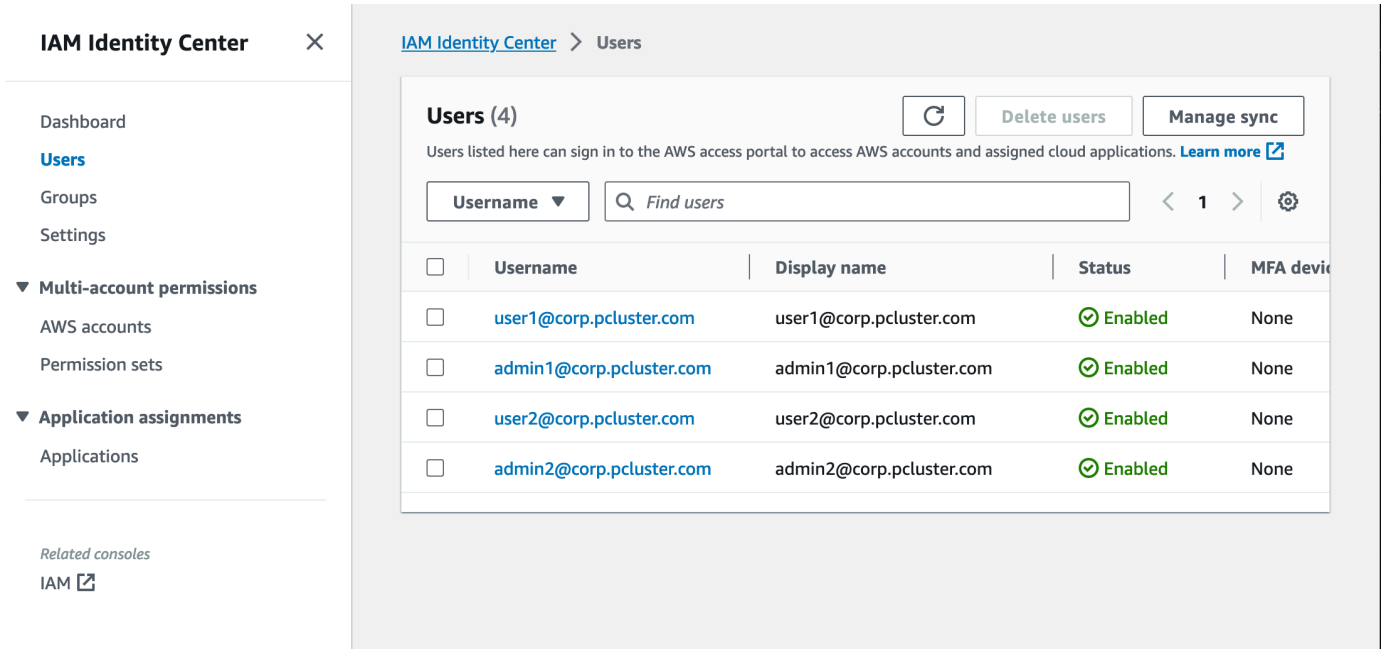


2. 在配置属性映射中，单击下一步
3. 在“配置同步范围”部分，键入要同步到 Identity Center 的用户的名称，然后单击添加
4. 添加完用户和组后，单击下一步



5. 查看您的更改，然后单击保存配置
6. 如果您在下一个屏幕中看到有关用户未同步的警告，请选择右上角的恢复同步按钮。
7. 接下来，要启用用户，请在左侧的用户选项卡中，选择一个用户，然后单击启用用户访问 > 启用用户访问

   注：如果顶部出现警告横幅，则可能需要选择“恢复同步”，然后等待用户进行同步（尝试使用刷新按钮查看用户是否已同步）。



将应用程序添加到 IAM Identity Center

将用户与 IAM Identity Center 同步后，您需要添加新的应用程序。这将配置可从 IAM Identity Center 门户使用哪些启用了 SSO 的应用程序。在本例中，我们将添加 AWS ParallelCluster UI 作为应用程序，而 IAM Identity Center 将作为身份提供者。

下一步将 AWS ParallelCluster 用户界面作为应用程序添加到 IAM 身份中心。 AWS ParallelCluster UI 是一个 Web 门户，可帮助用户管理其集群。有关更多信息，请参阅 AWS ParallelCluster UI。

在 Identity Center 中设置应用程序

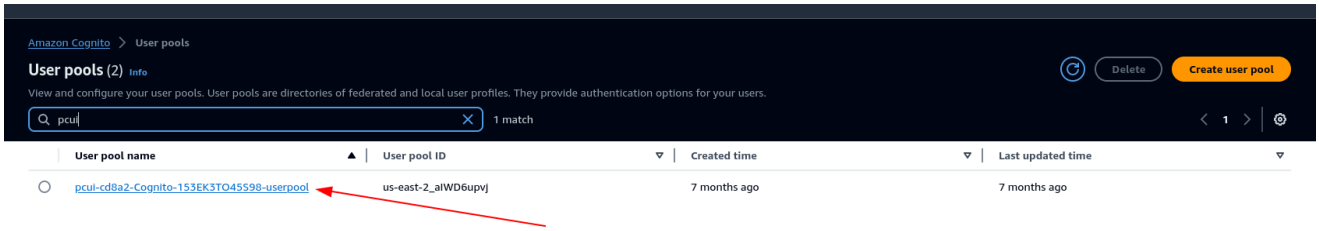
1. 在 IAM Identity Center > 应用程序下（位于左侧菜单栏上，单击“应用程序”）
2. 单击添加应用程序
3. 选择添加自定义 SAML 2.0 应用程序
4. 单击下一步
5. 选择您要使用的显示名称和描述（例如 PC AWS ParallelCluster UI 和 UI）
6. 在 IAM Identity Center 元数据下，复制 IAM Identity Center SAML 元数据文件的链接并保存备用，这将在 Web 应用程序上配置 SSO 时使用
7. 在应用程序属性下的“应用程序启动 URL”中，输入您的 PCUI 地址。这可以通过进入 CloudFormation 控制台，选择与 PCUI 对应的堆栈（例如 parallelcluster-ui），然后转到 “输出” 选项卡进行查找 ParallelCluster UIUrl

例如 https://m2iwazsi1j.execute-api.us-east-1.amazonaws.com

8. 在应用程序元数据下，选择手动键入您的元数据值。然后提供以下值。

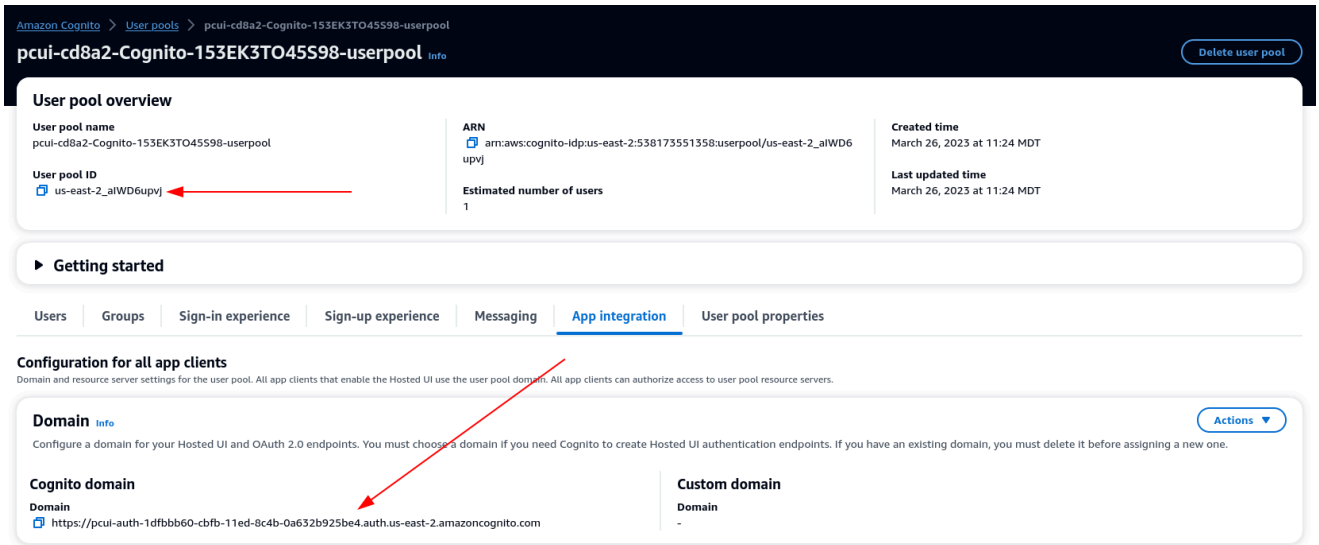
   a. 重要提示：请确保将域前缀、区域和用户池 ID 值替换为与您的环境相关的信息。

   b. 打开 Amazon Cognito > 用户池控制台，即可获取域前缀、区域和用户池 ID



   c. 选择与 PCUI 对应的用户池（其用户池名称将像 pcui-cd8a2-cognito-153 s98-UserP EK3 TO45 ool）

   d. 导航到应用程序集成



9. 应用程序断言消费者服务 (ACS) 网址：https://<domain-prefix>.auth。 <region>.amazoncognito。com/saml2/idpresponse

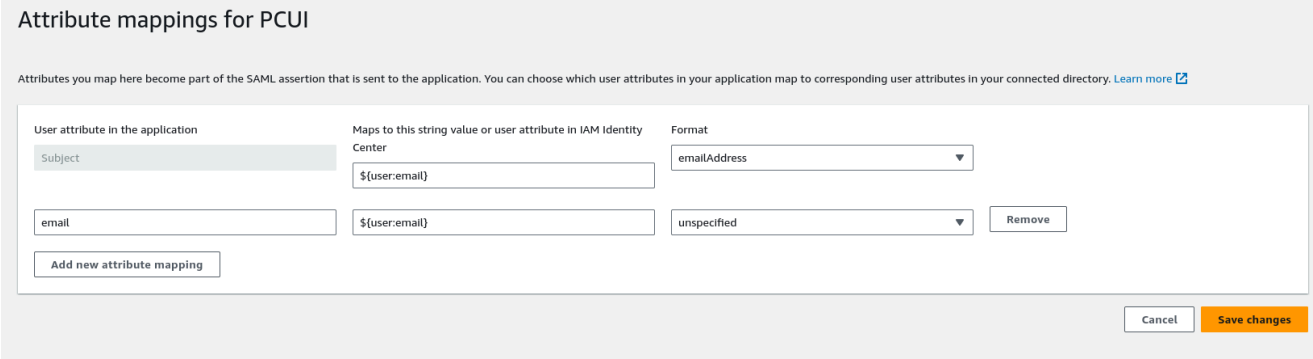
   应用程序 SAML 受众：urn:amazon:cognito:sp:<userpool-id>

10. 选择提交。然后，转到您添加的应用程序的详细信息页面。

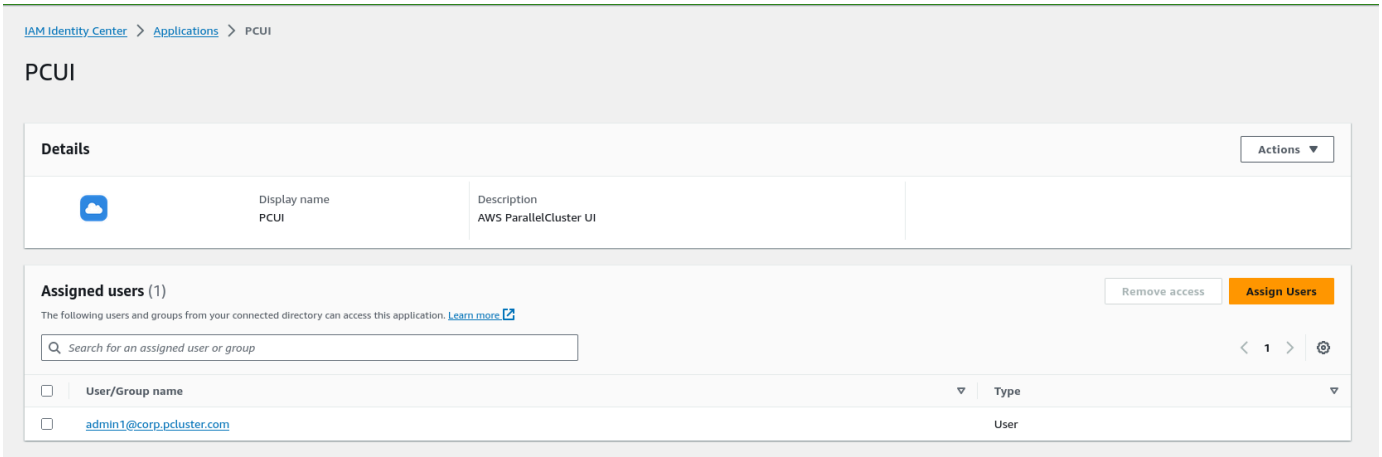
11. 选择操作下拉列表并选择编辑属性映射。然后，提供以下属性。

   a. 应用程序中的用户属性：subject（注：subject 已预填写。）→ 映射到 IAM Identity Center 中的这个字符串值或用户属性：${user:email}，格式：emailAddress

b. 应用程序中的用户属性：email → 映射到 IAM Identity Center 中的这个字符串值或用户属性：${user:email}，格式：unspecified

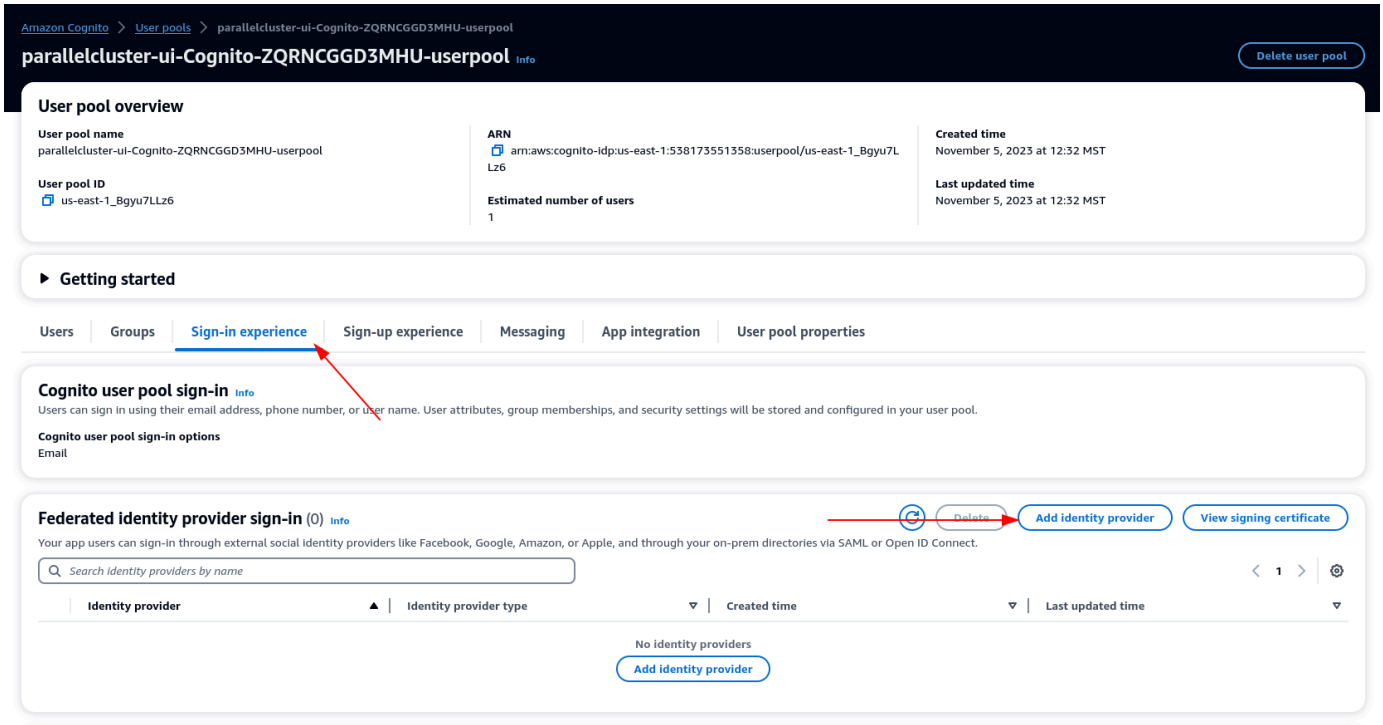


12. 保存您的更改。
13. 选择分配用户按钮，然后将您的用户分配给应用程序。这些是您的 Active Directory 中有权访问 PCUI 界面的用户。

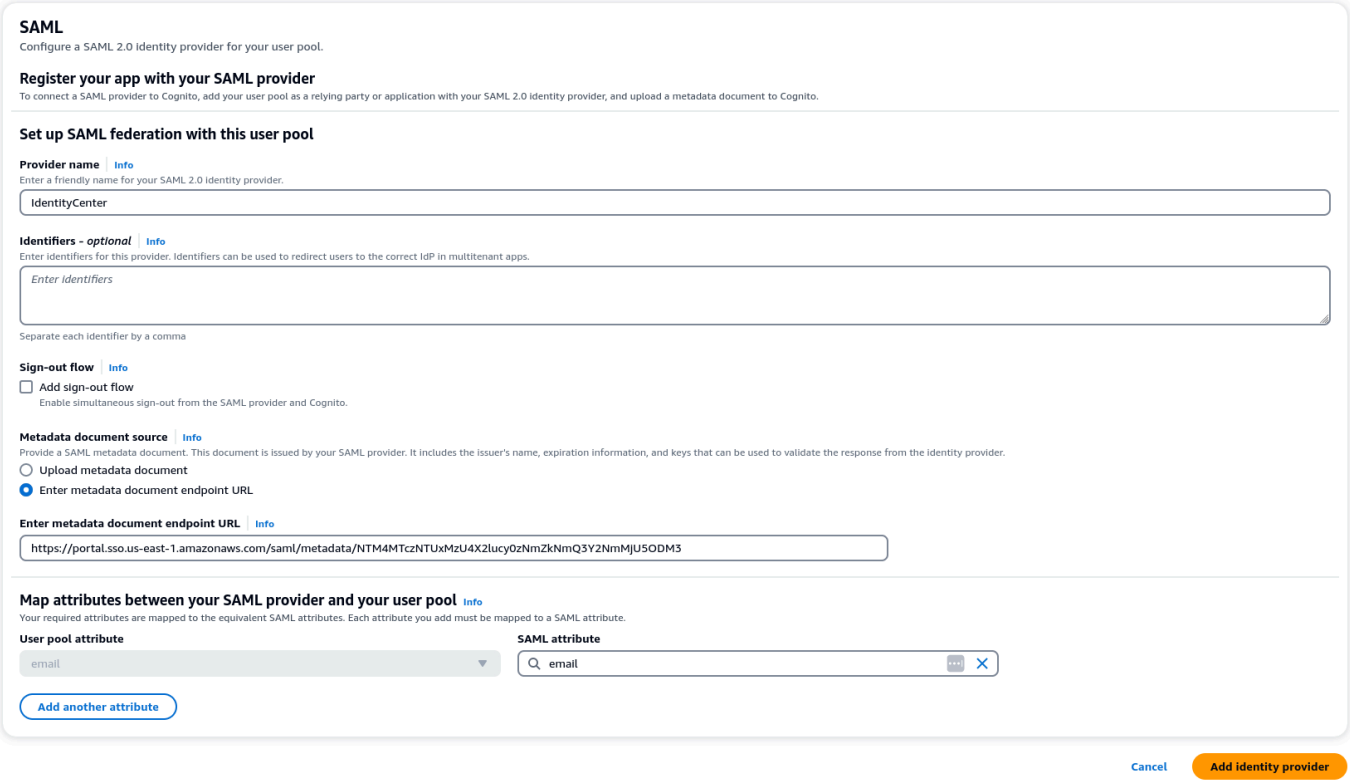


将 IAM Identity Center 配置为用户池中的 SAML IdP

1. 在您的用户池设置中，选择登录体验 > 添加身份提供者



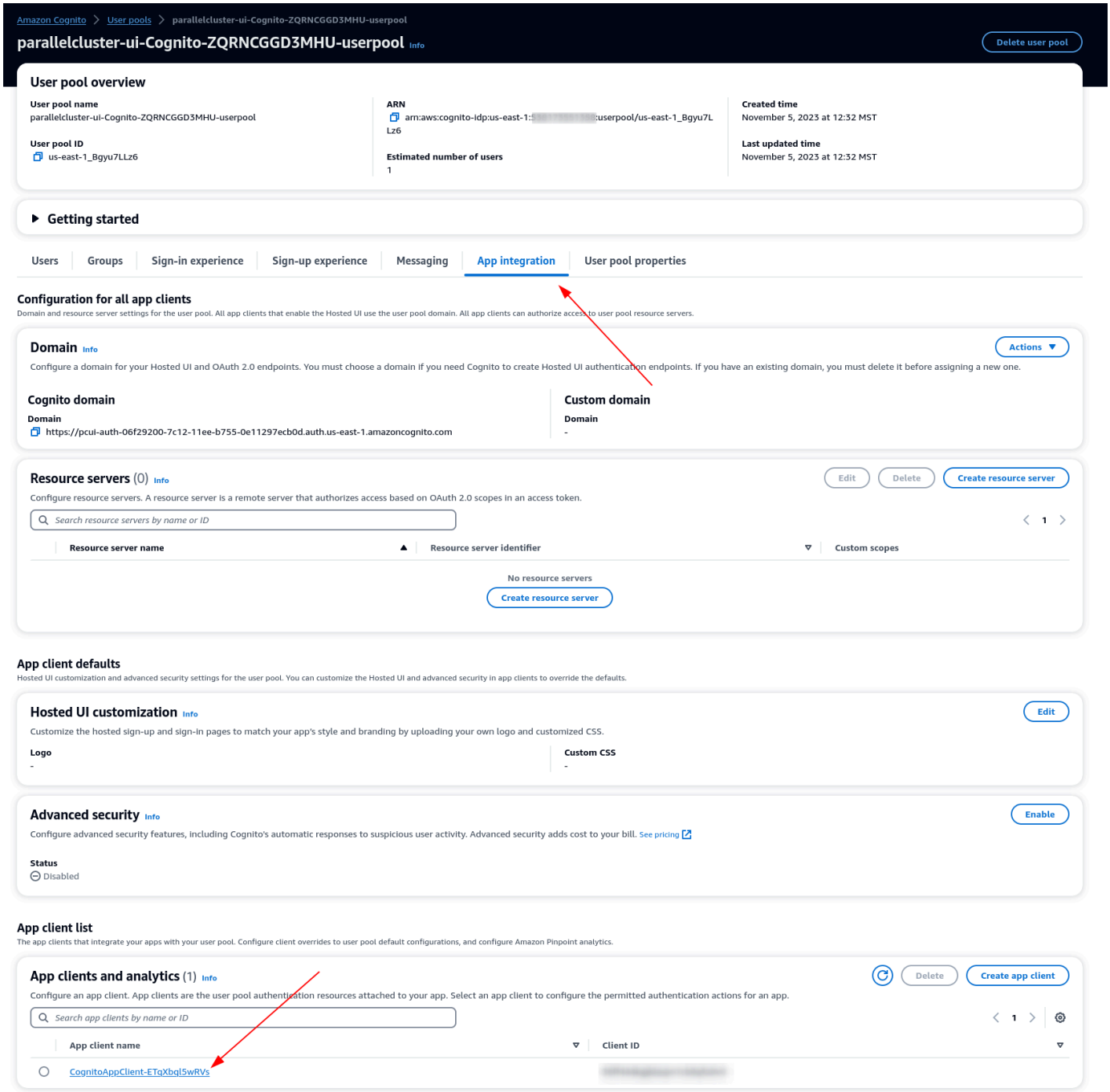
2. 选择一个 SAML IdP
3. 如需提供商名称，请提供 IdentityCenter
4. 在元数据文档源下，选择输入元数据文档端点 URL，然后提供在 Identity Center 的应用程序设置期间复制的 URL
5. 在属性下，如果是电子邮件，则选择 email



6. 选择添加身份提供者。

将 IdP 与用户池应用程序客户端集成

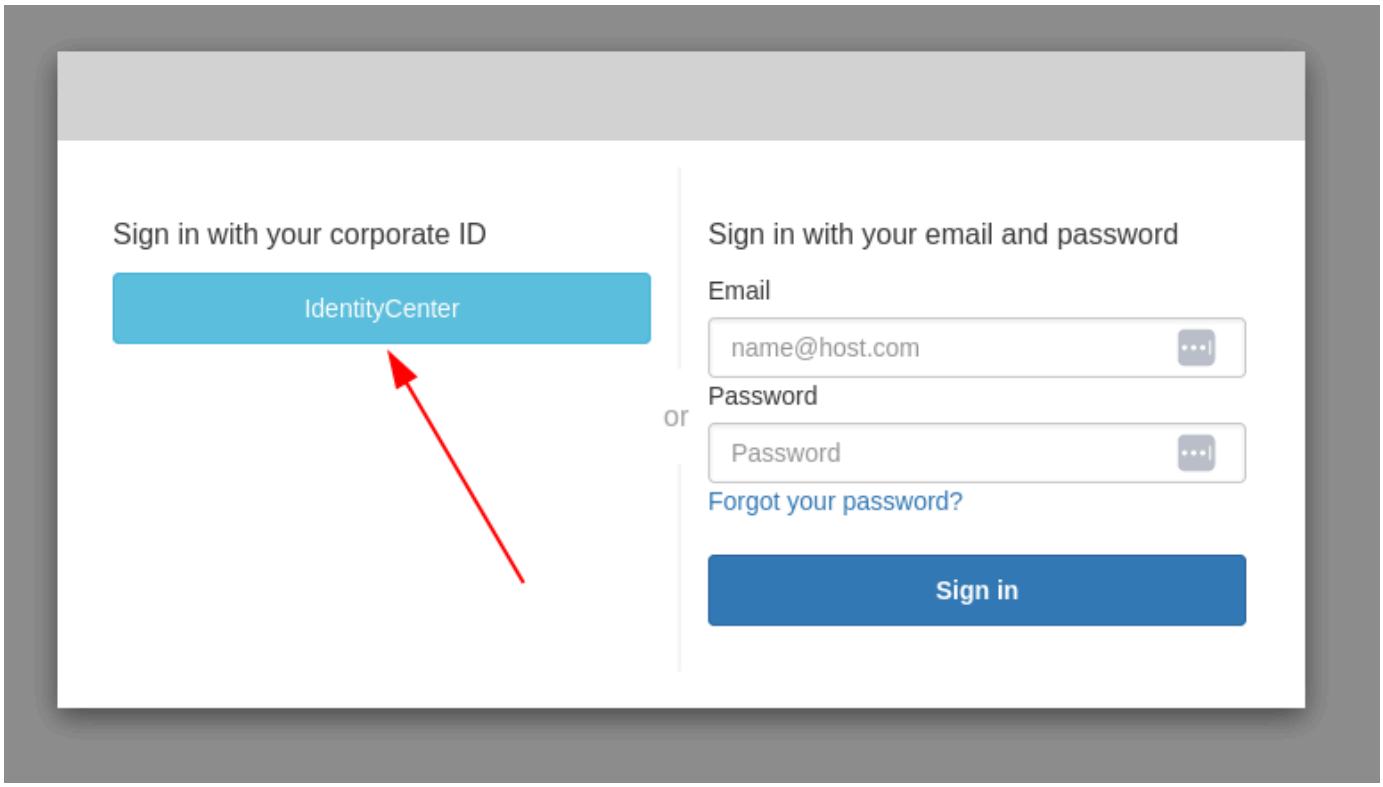
1. 接下来，在用户池的应用程序集成部分下，选择应用程序客户端列表下列出的客户端



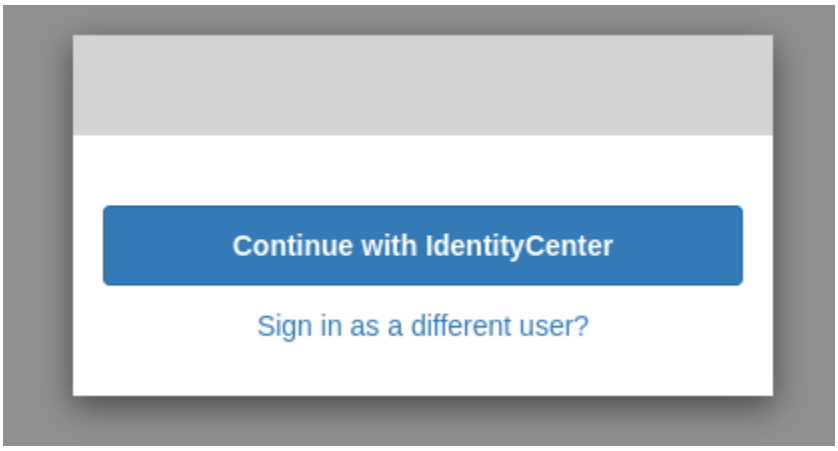
2. 在托管 UI 下，选择编辑
3. 在 “身份提供者” 下IdentityCenter也选择。
4. 选择 Save changes（保存更改）

验证您的设置

1. 接下来，我们将通过登录 PCUI 来验证刚刚创建的设置。登录 PCUI 门户，您现在应该可以看到使用企业 ID 登录的选项：

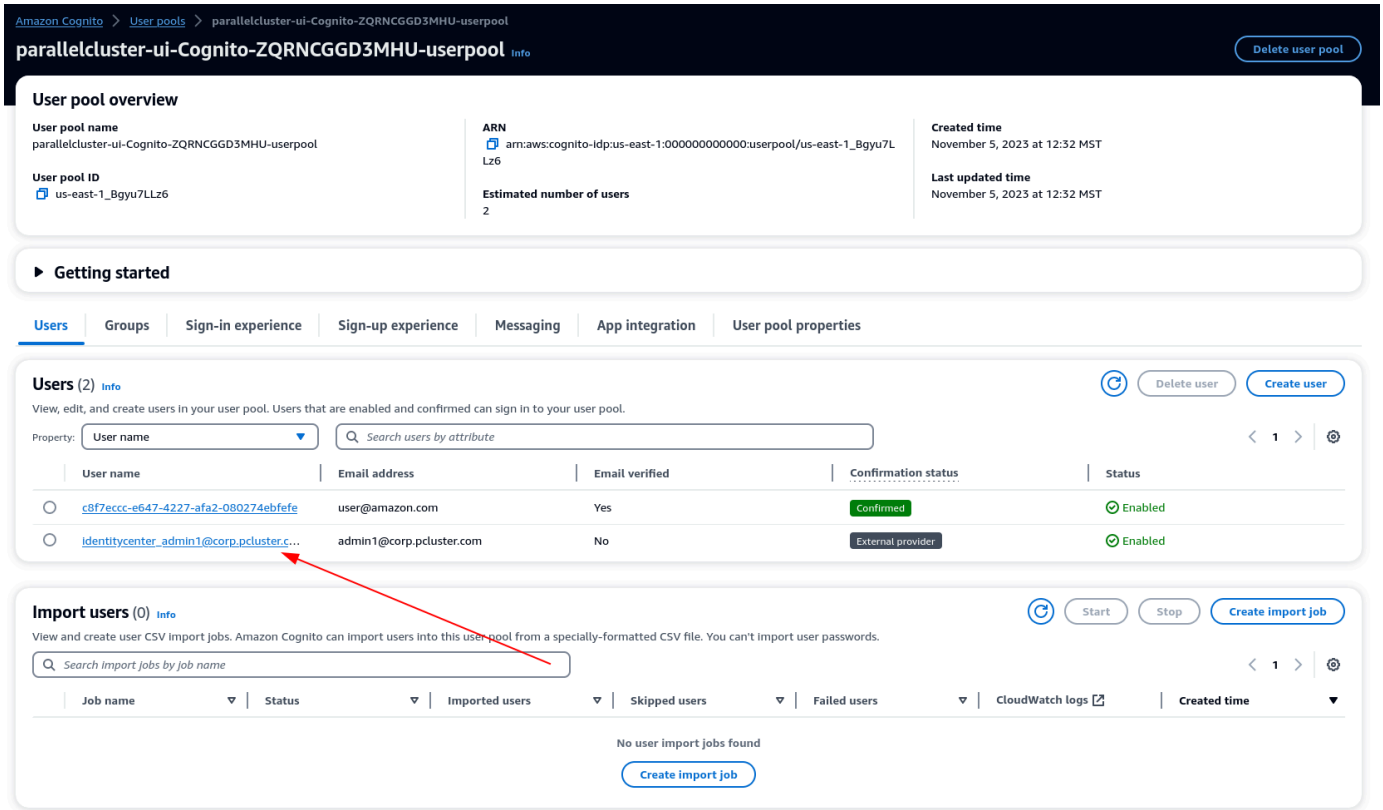


2. 单击该IdentityCenter按钮将带您进入 IAM Identity Center IdP 登录页面，然后打开一个包含您的应用程序（包括 PCUI）的页面，然后打开该应用程序。
3. 进入以下屏幕后，可看到您的用户应该已经添加到 Cognito 用户池中。



将用户设为管理员

1. 现在导航到 Amazon Cognito > 用户池控制台，选择新创建的用户，其前缀应为 identitycenter



2. 在组成员资格下，选择将用户添加到组，选择管理员，然后单击添加。
3. 现在，当您单击 “继续” 时， IdentityCenter您将被导航到 AWS ParallelCluster 用户界面页面。

使用 Pyxis 运行容器化作业

了解如何创建一个能够使用 Pyxis 运行容器化作业的集群，Pyxis 是一个 SPANK 插件，用于在 SLURM 中管理容器化作业。Pyxis 中的容器由 Enroot 管理，Enroot 是将传统 container/OS 图像转换为非特权沙箱的工具。有关更多信息，请参阅 NVIDIA Pyxis 和 NVIDIA Enroot。

> **Note**
>
> - 此功能在 AWS ParallelCluster v3.11.1 中可用
> - 本教程中的脚本移动 (mv) 一些文件，这会将它们从其原始位置删除。如果要将这些文件的副本保存在其原始位置，请更改脚本以改用 copy (cp) 命令。

使用时 AWS ParallelCluster，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 AWS 使用的服务 AWS ParallelCluster。

先决条件：

- AWS CLI 已安装并配置。
- 亚马逊 EC2 密钥对。 。
- 具有运行 pcluster CLI 所需的权限的 IAM 角色。

创建集群

从 AWS ParallelCluster 3.11.1 开始，所有官方版本都预 AMIs 装了 Pyxis 和 Enroot。特别是，SLURM 在 Pyxis 支持下重新编译，Enroot 作为二进制文件安装到系统中。不过，您必须根据自己的具体需求进行配置。Enroot 和 Pyxis 使用的文件夹对集群性能有重要影响。有关更多信息，请参阅 Pyxis documentation 和 Enroot documentation。

为方便起见，您可在 `/opt/parallelcluster/examples/` 中找到 Pyxis、Enroot 和 SPANK 的配置示例。

要使用我们提供的示例配置部署集群，请完成以下教程。

使用示例配置创建集群

必须在头节点上配置 Pyxis 和 Enroot，方法是先为 Enroot 创建持久和易失性目录，然后为 Pyxis 创建运行时目录，最后将 Pyxis 作为 SPANK 插件在整个集群中启用。

1. 在头节点中执行以下脚本作为OnNodeConfigured自定义操作，在头节点上配置 Pyxis 和 Enroot。

```
#!/bin/bash
set -e

echo "Executing $0"

# Configure Enroot
ENROOT_PERSISTENT_DIR="/var/enroot"
ENROOT_VOLATILE_DIR="/run/enroot"

sudo mkdir -p $ENROOT_PERSISTENT_DIR
sudo chmod 1777 $ENROOT_PERSISTENT_DIR
sudo mkdir -p $ENROOT_VOLATILE_DIR
sudo chmod 1777 $ENROOT_VOLATILE_DIR
sudo mv /opt/parallelcluster/examples/enroot/enroot.conf /etc/enroot/enroot.conf
sudo chmod 0644 /etc/enroot/enroot.conf
```

```
# Configure Pyxis
PYXIS_RUNTIME_DIR="/run/pyxis"

sudo mkdir -p $PYXIS_RUNTIME_DIR
sudo chmod 1777 $PYXIS_RUNTIME_DIR

sudo mkdir -p /opt/slurm/etc/plugstack.conf.d/
sudo mv /opt/parallelcluster/examples/spank/plugstack.conf /opt/slurm/etc/
sudo mv /opt/parallelcluster/examples/pyxis/pyxis.conf /opt/slurm/etc/
plugstack.conf.d/
sudo -i scontrol reconfigure
```

2. 必须在计算实例集上配置 Pyxis 和 Enroot，方法是为 Enroot 创建持久和易失性目录，为 Pyxis 创建运行时目录。在计算节点中将以下脚本作为OnNodeStart自定义操作执行，以便在计算队列上配置 Pyxis 和 Enroot。

```
#!/bin/bash
set -e

echo "Executing $0"

# Configure Enroot
ENROOT_PERSISTENT_DIR="/var/enroot"
ENROOT_VOLATILE_DIR="/run/enroot"
ENROOT_CONF_DIR="/etc/enroot"

sudo mkdir -p $ENROOT_PERSISTENT_DIR
sudo chmod 1777 $ENROOT_PERSISTENT_DIR
sudo mkdir -p $ENROOT_VOLATILE_DIR
sudo chmod 1777 $ENROOT_VOLATILE_DIR
sudo mkdir -p $ENROOT_CONF_DIR
sudo chmod 1777 $ENROOT_CONF_DIR
sudo mv /opt/parallelcluster/examples/enroot/enroot.conf /etc/enroot/enroot.conf
sudo chmod 0644 /etc/enroot/enroot.conf

# Configure Pyxis
PYXIS_RUNTIME_DIR="/run/pyxis"

sudo mkdir -p $PYXIS_RUNTIME_DIR
sudo chmod 1777 $PYXIS_RUNTIME_DIR
```

```
# In Ubuntu24.04 Apparmor blocks the creation of unprivileged user namespaces,
# which is required by Enroot. So to run Enroot, it is required to disable this
 restriction.
# See https://ubuntu.com/blog/ubuntu-23-10-restricted-unprivileged-user-namespaces
source /etc/os-release
if [ "${ID}${VERSION_ID}" == "ubuntu24.04" ]; then
    echo "kernel.apparmor_restrict_unprivileged_userns = 0" | sudo tee /etc/
sysctl.d/99-pcluster-disable-apparmor-restrict-unprivileged-userns.conf
    sudo sysctl --system
fi
```

提交作业

现在，Pyxis 已配置到集群中，您可以使用 sbatch 和 srun 命令提交容器化作业了，这些命令现在增加了特定于容器的选项。

```
# Submitting an interactive job
srun -N 2 --container-image docker://ubuntu:22.04 hostname

# Submitting a batch job
sbatch -N 2 --wrap='srun --container-image docker://ubuntu:22.04 hostname'
```

创建具有启用 EFA 的 Lustre FSx 的集群

在本教程中，您将创建一个使用支持 EFA 的 FSx Lustre 文件系统作为共享存储的集群。使用启用 EFA 的 FSx Lustre 文件系统可以将性能提高多达 8 倍。要验证是否需要启用 EFA 的文件系统，请查看 for Lustre 用户指南中的使用启用 EFA 的文件系统。FSx

使用时 AWS ParallelCluster，您只需为创建或更新 AWS ParallelCluster 映像和集群时创建的 AWS 资源付费。有关更多信息，请参阅 AWS 使用的服务 AWS ParallelCluster。

要求

- C AWS LI 已安装并配置完毕。
- C ParallelCluster LI 已安装并配置完毕。
- 用于登录集群的 Amazon EC2 密钥对。

- 具有运行 ParallelCluster CLI 所需权限的 IAM 角色。

创建安全组

在部署集群和文件系统的同一 VPC 中创建两个安全组：一个用于在群集节点上运行的客户端，另一个用于文件系统。

```
# Create security group for the FSx client
aws ec2 create-security-group \
    --group-name Fsx-Client-SecurityGroup \
    --description "Allow traffic for the FSx Lustre client" \
    --vpc-id vpc-cluster \
    --region region

# Create security group for the FSx file system
aws ec2 create-security-group \
    --group-name Fsx-FileSystem-SecurityGroup \
    --description "Allow traffic for the FSx Lustre File System" \
    --vpc-id vpc-cluster \
    --region region
```

在本教程的其余部分中，我们将假设`sg-client`和分别`sg-file-system`是客户端和文件系统的安全组 ID。

按照 EFA 的要求，为客户端配置安全组以允许所有出站流量进入文件系统。

```
# Allow all outbound traffic from the client to the file system
aws ec2 authorize-security-group-egress \
 --group-id sg-client \
 --protocol -1 \
 --port -1 \
 --source-group sg-file-system \
 --region region
```

按照 EFA 的要求，为文件系统配置安全组，使其允许其内部的所有入站/出站流量以及来自客户端的所有入站流量。

```
# Allow all inbound traffic within this security group
aws ec2 authorize-security-group-ingress \
    --group-id sg-file-system \
    --protocol -1 \
```

```
    --port -1 \
    --source-group sg-file-system \
    --region region

# Allow all outbound traffic within this security group
aws ec2 authorize-security-group-egress \
    --group-id sg-file-system \
    --protocol -1 \
    --port -1 \
    --source-group sg-file-system \
    --region region

# Allow all inbound traffic from the client
aws ec2 authorize-security-group-ingress \
    --group-id sg-file-system \
    --protocol -1 \
    --port -1 \
    --source-group sg-client \
    --region region

# Allow all outbound traffic to the client
aws ec2 authorize-security-group-egress \
    --group-id sg-file-system \
    --protocol -1 \
    --port -1 \
    --source-group sg-client \
    --region region
```

创建文件系统

在计算节点所在的可用区 (AZ) 中创建文件系统；并在以下代码中*subnet-compute-nodes*替换为其 ID。这是允许 EFA 使用您的文件系统所必需的。请注意，作为创建文件系统的一部分，我们使用 EfaEnable 属性启用 EFA。

```
aws fsx create-file-system \
    --file-system-type LUSTRE \
    --storage-capacity 38400 \
    --storage-type SSD \
    --subnet-ids subnet-compute-nodes \
    --security-group-ids sg-file-system \
    --lustre-configuration
 DeploymentType=PERSISTENT_2,PerUnitStorageThroughput=125,EfaEnabled=true,MetadataConfiguration
 \
```

```
    --region region
```

记下上一个命令返回的文件系统 ID。在本教程的其余部分中，*fs-id*用此文件系统 ID 替换。

创建集群

1. 使用 AWS ParallelCluster YAML 配置文件中设置的以下配置创建集群：
   a. AMI 基于支持的操作系统，例如 Ubuntu 22.04。
   b. 计算节点必须使用支持 EFA 且具有 Nitro v4+ 的实例类型，例如 g6.16xlarge。
      - 计算节点必须位于文件系统所在的同一个可用区中。
      - 计算节点必须将 Efa/Enabled 设置为 true。
      - 计算节点必须将配置脚本`configure-efa-fsx-lustre-client.sh`作为OnNodeStart自定义操作运行。为了方便起见，该脚本在FSx 官方文档中提供，并在我们的公共存储桶中提供，旨在在计算节点上配置 FSx Lustre 客户端，让它们使用 EFA。
2. 创建集群配置文件`config.yaml`：

```
Region: region
Image:
  Os: ubuntu2204
HeadNode:
  InstanceType: c5.xlarge
  Networking:
    SubnetId: subnet-xxxxxxxxxx
    AdditionalSecurityGroups:
        - sg-client
  Ssh:
    KeyName: my-ssh-key
Scheduling:
  Scheduler: slurm
  SlurmQueues:
    - Name: q1
      ComputeResources:
        - Name: cr1
          Instances:
            - InstanceType: g6.16xlarge
          MinCount: 1
          MaxCount: 3
          Efa:
            Enabled: true
      Networking:
```

```
        SubnetIds:
          - subnet-xxxxxxxxxx # Subnet in the same AZ where the file system is
        AdditionalSecurityGroups:
          - sg-client
        PlacementGroup:
          Enabled: false
      CustomActions:
        OnNodeStart:
          Script: https://us-east-1-aws-parallelcluster.s3.us-east-1.amazonaws.com/
scripts/fsx-lustre-efa/configure-efa-fsx-lustre-client.sh
SharedStorage:
  - MountDir: /fsx
    Name: my-fsxlustre-efa-external
    StorageType: FsxLustre
    FsxLustreSettings:
      FileSystemId: fs-id
```

然后使用该配置创建集群：

```
pcluster create-cluster \
    --cluster-name fsx-efa-tutorial \
    --cluster-configuration config.yaml \
    --region region
```

FSx 使用 EFA 进行验证正在起作用

要验证 Lustre 网络流量是否在使用 EFA，请使用可以显示给定网络接口的网络流量的 Lustre `lnetctl` 工具。为此，请在计算节点中执行以下命令：

```
# Take note of the number of packets flowing through the interface,
# which are specified in statistics:send_count and statistics:recv_count
sudo lnetctl net show --net efa -v

# Generate traffic to the file system
echo 'Hello World' > /fsx/hello-world.txt

# Take note of the number of packets flowing through the interface,
# which are specified in statistics:send_count and statistics:recv_count
sudo lnetctl net show --net efa -v
```

如果该功能起作用，则流经该接口的数据包数量预计会增加。

AWS ParallelCluster 故障排除

以下各节针对使用 AWS ParallelCluster时可能出现的问题提供问题排查提示。 AWS ParallelCluster 社区维护着一个 Wiki 页面，在 AWS ParallelCluster GitHub Wi ki 上提供了许多疑难解答技巧。有关已知问题的列表，请参阅已知问题。

主题

- 尝试创建集群
- 尝试运行作业
- 尝试更新集群
- 尝试访问存储
- 尝试删除集群
- 尝试升级 AWS ParallelCluster API 堆栈
- 在计算节点初始化过程中看到错误
- 集群运行状况指标故障排除
- 排查集群部署问题
- 排查使用 Terraform 部署集群的问题
- 排查扩展问题
- 置放群组和实例启动问题
- 替换目录
- 排查 Amazon DCV 中的问题
- 通过 AWS Batch 集成对集群中的问题进行故障排除
- 排查与 Active Directory 的多用户集成问题
- 排查自定义 AMI 问题
- 排查 cfn-hup 未运行时的集群更新超时问题
- 网络问题排查
- 执行 onNodeUpdated 自定义操作时集群更新失败
- 看到自定义 Slurm 配置错误
- 集群警报

尝试创建集群

使用 3.5.0 及更高 AWS ParallelCluster 版本创建集群，且集群创建失败且`--rollback-on-failure`设置为时`false`，请使用 `pcluster describe-cluster` CLI 命令获取状态和失败信息。在这种情况下，`pcluster describe-cluster` 输出的预期 `clusterStatus` 为 `CREATE_FAILED`。检查输出中的 `failures` 部分以找到 `failureCode` 和 `failureReason`。然后在后面的部分中找到匹配的 `failureCode` 以获取更多故障排除帮助。有关更多信息，请参阅 `pcluster describe-cluster`。

在后面的部分中，我们建议您查看头节点上的日志，例如 `/var/log/cfn-init.log` 和 `/var/log/chef-client.log` 文件。有关 AWS ParallelCluster 日志以及如何查看日志的更多信息，请参阅用于调试的关键日志和检索和保留日志。

如果您没有`failureCode`，请导航到 AWS CloudFormation 控制台以查看集群堆栈。查看 `HeadNodeWaitCondition` 的 `Status Reason` 或其他资源上的失败，以了解更多失败详细信息。有关更多信息，请参阅 查看上 AWS CloudFormation 发生的事件 `CREATE_FAILED`。检查头节点上的 `/var/log/cfn-init.log` 和 `/var/log/chef-client.log` 文件。如果由于头节点创建失败而导致集群创建失败，并且集群日志组中没有集群日志，则必须在失败时保留集群，指定 `--rollback-on-failure = True` 并从头节点本身检索日志。

`failureCode` 是 `OnNodeConfiguredExecutionFailure`

- 为什么失败？

  您在配置的头节点部分的 `OnNodeConfigured` 中提供了用于创建集群的自定义脚本。但该自定义脚本未能运行。

- 如何解决？

  检查 `/var/log/cfn-init.log` 文件以更详细地了解失败以及如何解决自定义脚本中的问题。在该日志快要结束时，您可能会在 `Running command runpostinstall` 消息后看到与 `OnNodeConfigured` 脚本相关的运行信息。

`failureCode` 是 `OnNodeConfiguredDownloadFailure`

- 为什么失败？

  您在配置的头节点部分的 `OnNodeConfigured` 中提供了用于创建集群的自定义脚本。但该自定义脚本未能下载。

- 如何解决？

  确保 URL 有效并且正确配置了访问权限。有关自定义引导脚本配置的更多信息，请参阅自定义引导操作。

  检查 /var/log/cfn-init.log 文件。在该日志快要结束时，您可能会在 Running command runpostinstall 消息后看到与 OnNodeConfigured 脚本处理（包括下载）相关的运行信息。

failureCode 是 OnNodeConfiguredFailure

- 为什么失败？

  您在配置的头节点部分的 OnNodeConfigured 中提供了用于创建集群的自定义脚本。但在集群部署中使用自定义脚本失败。无法确定直接原因，需要进一步调查。

- 如何解决？

  检查 /var/log/cfn-init.log 文件。在该日志快要结束时，您可能会在 Running command runpostinstall 消息后看到与 OnNodeConfigured 脚本处理相关的运行信息。

failureCode 是 OnNodeStartExecutionFailure

- 为什么失败？

  您在配置的头节点部分的 OnNodeStart 中提供了用于创建集群的自定义脚本。但该自定义脚本未能运行。

- 如何解决？

  检查 /var/log/cfn-init.log 文件以更详细地了解失败以及如何解决自定义脚本中的问题。在该日志快要结束时，您可能会在 Running command runpreinstall 消息后看到与 OnNodeStart 脚本相关的运行信息。

failureCode 是 OnNodeStartDownloadFailure

- 为什么失败？

  您在配置的头节点部分的 OnNodeStart 中提供了用于创建集群的自定义脚本。但该自定义脚本未能下载。

- 如何解决？

  确保 URL 有效并且正确配置了访问权限。有关自定义引导脚本配置的更多信息，请参阅自定义引导操作。

  检查 `/var/log/cfn-init.log` 文件。在该日志快要结束时，您可能会在 `Running command runpreinstall` 消息后看到与 `OnNodeStart` 脚本处理（包括下载）相关的运行信息。

failureCode 是 OnNodeStartFailure

- 为什么失败？

  您在配置的头节点部分的 `OnNodeStart` 中提供了用于创建集群的自定义脚本。但在集群部署中使用自定义脚本失败。无法确定直接原因，需要进一步调查。

- 如何解决？

  检查 `/var/log/cfn-init.log` 文件。在该日志快要结束时，您可能会在 `Running command runpreinstall` 消息后看到与 `OnNodeStart` 脚本处理相关的运行信息。

failureCode 是 EbsMountFailure

- 为什么失败？

  集群配置中定义的 EBS 卷无法挂载。

- 如何解决？

  检查 `/var/log/chef-client.log` 文件以查看失败详细信息。

failureCode 是 EfsMountFailure

- 为什么失败？

  集群配置中定义的 Amazon EFS 卷无法挂载。

- 如何解决？

  如果您定义了现有的 Amazon EFS 文件系统，请确保允许集群和该文件系统之间的流量。有关更多信息，请参阅 `SharedStorage`/`EfsSettings`/`FileSystemId`。

检查 `/var/log/chef-client.log` 文件以查看失败详细信息。

failureCode 是 FsxMountFailure

- 为什么失败？

  集群配置中定义的 Amazon FSx 文件系统无法装载。

- 如何解决？

  如果您定义了现有 Amazon FSx 文件系统，请确保允许集群和文件系统之间的流量。有关更多信息，请参阅 `SharedStorage`/`FsxLustreSettings`/`FileSystemId`。

  检查 `/var/log/chef-client.log` 文件以查看失败详细信息。

failureCode 是 RaidMountFailure

- 为什么失败？

  集群配置中定义的 RAID 卷无法挂载。

- 如何解决？

  检查 `/var/log/chef-client.log` 文件以查看失败详细信息。

failureCode 是 AmiVersionMismatch

- 为什么失败？

  用于创建自定义 AMI 的 AWS ParallelCluster 版本与用于配置集群的 AWS ParallelCluster 版本不同。在 CloudFormation 控制台中，查看集群 CloudFormation 堆栈详细信息并查看`HeadNodeWaitCondition`，以获取`Status Reason`有关 AWS ParallelCluster 版本和 AMI 的更多详细信息。有关更多信息，请参阅 查看上 AWS CloudFormation 发生的事件 `CREATE_FAILED`。

- 如何解决？

  确保用于创建自定义 AMI 的 AWS ParallelCluster 版本与用于配置集群的 AWS ParallelCluster 版本相同。您可以更改自定义 AMI 版本或 `pcluster` CLI 版本以使其相同。

failureCode 是 InvalidAmi

- 为什么失败？

  自定义 AMI 无效，因为它不是使用构建的 AWS ParallelCluster。

- 如何解决？

  使用 `pcluster build-image` 命令，通过将您的 AMI 设置为父映像来创建 AMI。有关更多信息，请参阅 pcluster build-image。

failureCode 是 HeadNodeBootstrapFailure，并且 failureReason 是“无法设置头节点”。

- 为什么失败？

  无法确定直接原因，需要进一步调查。例如，可能是因为集群处于受保护状态，这可能是由于无法预置静态计算实例集造成的。

- 如何解决？

  检查 `/var/log/chef-client.log.` 文件以查看失败详细信息。

> **Note**
>
> 如果您看到 `RuntimeError` 异常“`Cluster state has been set to PROTECTED mode due to failures detected in static node provisioning`”，则表示集群处于受保护状态。有关更多信息，请参阅 如何调试受保护模式。

failureCode 是 HeadNodeBootstrapFailure，并且 failureReason是“集群创建超时”。

- 为什么失败？

  默认情况下，集群创建的完成时间限制为 30 分钟。如果集群创建未在此时间范围内完成，则集群创建将失败并显示超时错误。集群创建可能由于不同的原因而超时。例如，头节点创建失败、网络问题、头节点中的自定义脚本运行时间过长、计算节点中运行的自定义脚本中的错误或计算节点预置等待时间过长均可导致超时失败。无法确定直接原因，需要进一步调查。

- 如何解决？

  检查 /var/log/cfn-init.log 和 /var/log/chef-client.log 文件以查看失败详细信息。有关 AWS ParallelCluster 日志以及如何获取这些日志的更多信息，请参阅用于调试的关键日志和检索和保留日志。

  在这些日志中，您可能会发现以下内容。

  - 在 **chef-client.log** 快要结束时看到“**Waiting for static fleet capacity provisioning**”

    这表示在等待静态节点启动时，集群创建超时。有关更多信息，请参阅 在计算节点初始化过程中看到错误。

  - 在 **cfn-init.log** 的末尾看到 **OnNodeConfigured** 或 **OnNodeStart** 头节点脚本未完成

    这表示 OnNodeConfigured 或 OnNodeStart 自定义脚本的运行需要很长时间并导致了超时错误。检查您的自定义脚本是否存在导致其运行很长时间的问题。如果您的自定义脚本需要运行很长时间，请考虑在集群配置文件中添加一个 DevSettings 部分来更改超时限制，如以下示例所示：

    ```
    DevSettings:
      Timeouts:
        HeadNodeBootstrapTimeout: 1800 # default setting: 1800 seconds
    ```

  - 找不到日志，或者未成功创建头节点

    可能未成功创建头节点，因此找不到日志。在 CloudFormation 控制台中，查看集群堆栈详细信息以查看其他故障详细信息。

failureCode 是 HeadNodeBootstrapFailure，并且 failureReason 是“无法引导头节点”。

- 为什么失败？

  无法确定直接原因，需要进一步调查。

- 如何解决？

  检查 /var/log/cfn-init.log 和 /var/log/chef-client.log 文件。

failureCode 是 ResourceCreationFailure

- 为什么失败？

  在集群创建过程中，创建某些资源失败。导致失败的原因多种多样。例如，容量问题或 IAM 策略配置错误可导致资源创建失败。

- 如何解决？

  在 CloudFormation 控制台中，查看群集堆栈以查看其他资源创建失败的详细信息。

failureCode 是 ClusterCreationFailure

- 为什么失败？

  无法确定直接原因，需要进一步调查。

- 如何解决？

  在 CloudFormation 控制台中，查看集群堆栈并查看，`Status Reason``HeadNodeWaitCondition`以查找其他故障详细信息。

  检查 `/var/log/cfn-init.log` 和 `/var/log/chef-client.log` 文件。

WaitCondition timed out...在 CloudFormation 堆栈里看

有关更多信息，请参阅 failureCode 是 HeadNodeBootstrapFailure，并且 failureReason是“集群创建超时”。。

Resource creation cancelled在 CloudFormation 堆栈里看

有关更多信息，请参阅 failureCode 是 ResourceCreationFailure。

在 AWS CloudFormation 堆栈中看到错误Failed to run cfn-init...或其他错误

检查 `/var/log/cfn-init.log` 和 `/var/log/chef-client.log` 以查看其他失败详细信息。

看到 `chef-client.log` 以“`INFO: Waiting for static fleet capacity provisioning`”结束

这与等待静态节点启动时集群创建超时有关。有关更多信息，请参阅 在计算节点初始化过程中看到错误。

看到`Failed to run preinstall or postinstall in cfn-init.log`

集群配置 `HeadNode` 部分中有 `OnNodeConfigured` 或 `OnNodeStart` 脚本。该脚本未正常运行。检查 `/var/log/cfn-init.log` 文件以查看自定义脚本的错误详细信息。

`This AMI was created with xxx, but is trying to be used with xxx...`在 CloudFormation 堆栈里看

有关更多信息，请参阅 `failureCode` 是 `AmiVersionMismatch`。

`This AMI was not baked by AWS ParallelCluster...`在 CloudFormation 堆栈里看

有关更多信息，请参阅 `failureCode` 是 `InvalidAmi`。

看到 `pcluster create-cluster` 命令无法在本地运行

检查本地文件系统中的 `~/.parallelcluster/pcluster-cli.log` 以查看失败详细信息。

其他支持

按照排查集群部署问题中的故障排除指南进行操作。

查看您的场景是否包含在 “GitHub 已知问题 AWS ParallelCluster ” 中 GitHub。

尝试运行作业

如果您在尝试运行作业时遇到问题，下一节将提供可能的问题排查解决方案。

srun 交互式作业失败并显示错误“srun: error: fwd_tree_thread: can't find address for <host>, check slurm.conf”

- 为什么失败？

  您运行了 srun 命令以提交作业，然后使用 pcluster update-cluster 命令提高了队列大小，而没有在更新完成后重启 Slurm 进程守护程序。

  Slurm 以树状层次结构组织 Slurm 进程守护程序以优化通信。只有在进程守护程序启动时才会更新此层次结构。

  假设您使用 srun 启动一个作业，然后运行 pcluster update-cluster 命令提高队列的大小。新计算节点在更新过程中启动。然后，Slurm 将您的作业排队到其中一个新计算节点。在这种情况下，Slurm 进程守护程序和 srun 都不会检测到新计算节点。srun 返回错误，因为它未检测到新节点。

- 如何解决？

  在所有计算节点上重启 Slurm 进程守护程序，然后使用 srun 提交作业。您可以通过运行重启计算节点的 scontrol reboot 命令来调度 Slurm 进程守护程序重启。有关更多信息，请参阅 Slurm 文档中的 scontrol reboot。您也可以通过请求重启相应的 systemd 服务来手动重启计算节点上的 Slurm 进程守护程序。

运行 squeue 命令时，作业卡在 CF 状态

这可能是动态节点启动的问题。有关更多信息，请参阅 在计算节点初始化过程中看到错误。

运行大型作业并看到“nfsd: too many open connections, consider increasing the number of threads in /var/log/messages”

对于网络文件系统，当达到网络限制时， I/O 等待时间也会增加。这可能会导致软锁定，因为网络用于写入网络和 I/O 指标的数据。

对于第 5 代实例，我们使用 ENA 驱动程序来公开数据包计数器。这些计数器计算网络达到实例带宽限制 AWS 时形成的数据包。您可以检查这些计数器以查看它们是否大于 0。如果是，则说明您已超出带宽限制。您可以通过运行 ethtool -S eth0 | grep exceeded 来查看这些计数器。

超出网络限制通常是由于支持过多的 NFS 连接所致。当达到或超过网络限制时，这是首先要检查的因素之一。

例如，以下输出显示已丢弃程序包：

```
$ ethtool -S eth0 | grep exceeded
  bw_in_allowance_exceeded: 38750610
  bw_out_allowance_exceeded: 1165693
  pps_allowance_exceeded: 103
  conntrack_allowance_exceeded: 0
  linklocal_allowance_exceeded: 0
```

为避免收到此消息，请考虑将头节点实例类型更改为性能更高的实例类型。考虑将您的数据存储移至未导出为 NFS 共享的共享存储文件系统，例如 Amazon EFS 或 Amaz FSx on。有关更多信息，请参阅共享存储 AWS ParallelCluster Wiki 上的 “最佳实践” GitHub。

运行 MPI 作业

启用调试模式

要启用 OpenMPI 调试模式，请参阅 Open MPI 有哪些控件可以帮助调试。

要启用 IntelMPI 调试模式，请参阅其他环境变量。

在作业输出中看到 `MPI_ERRORS_ARE_FATAL` 和 `OPAL ERROR`

这些错误代码来自应用程序中的 MPI 层。要了解如何从应用程序获取 MPI 调试日志，请参阅启用调试模式。

导致此错误的一个可能原因是，您的应用程序已针对特定的 MPI 实现（例如 OpenMPI）进行编译，而您正在尝试对不同的 MPI 实现（例如 IntelMPI）运行该应用程序。确保针对相同的 MPI 实现编译和运行应用程序。

在禁用托管 DNS 的情况下使用 `mpirun`

对于在 SlurmSettings/Dn s/DisableManagedDns和 UseEc2Hostnames 设置为的情况下创建的集群`true`，DNS 不会解析Slurm节点名称。 Slurm如果nodenames未启用且 MPI 作业是在上下文中运行的，则可以引导 MPI 进程。Slurm我们建议按照 Slurm MPI 用户指南中的指导来运行使用 Slurm 的 MPI 作业。

尝试更新集群

下一节提供在尝试更新集群时可能发生的问题的问题排查解决方案。

`pcluster update-cluster` 命令无法在本地运行

检查本地文件系统中的 `~/.parallelcluster/pcluster-cli.log` 以查看失败详细信息。

使用 `pcluster describe-cluster` 命令时看到 `clusterStatus` 为 `UPDATE_FAILED`

如果集群堆栈更新回滚，请检查 `/var/log/chef-client.log` 文件以查看错误详细信息。

查看在 “GitHub 已知问题” 中是否提到了您的问题 GitHub。 AWS ParallelCluster

集群更新超时

这可能是与 `cfn-hup` 未运行有关的问题。如果 `cfn-hup` 进程守护程序因外部原因终止，它不会自动重启。如果`cfn-hup`未运行，则在集群更新期间， CloudFormation 堆栈会按预期启动更新过程，但更新过程未在头节点上激活，堆栈部署最终会超时。有关更多信息，请参阅排查 `cfn-hup` 未运行时的集群更新超时问题以排除故障并从问题中恢复。

尝试访问存储

了解尝试访问存储时的问题排查提示。

使用外部 Amazon f FSx or Lustre 文件系统

确保允许集群和该文件系统之间的流量。该文件系统必须与通过端口 988、1021、1022 和 1023 允许入站和出站 TCP 流量的安全组关联。有关如何设置安全组的更多信息，请参阅FileSystemId。

使用外部 Amazon Elastic File System 文件系统

确保允许集群和该文件系统之间的流量。该文件系统必须与通过端口 988、1021、1022 和 1023 允许入站和出站 TCP 流量的安全组关联。有关如何设置安全组的更多信息，请参阅FileSystemId。

尝试删除集群

如果您在尝试删除集群时遇到错误，以下各节提供常见场景的问题排查提示。

`pcluster delete-cluster` 命令无法在本地运行

检查本地文件系统中的 `~/.parallelcluster/pcluster-cli.log` 文件。

无法删除集群堆栈

如果集群堆栈删除失败，请查看 CloudFormation 堆栈事件消息。

检查您的问题是否在 “GitHub 已知问题” 中提及 GitHub。 AWS ParallelCluster

尝试升级 AWS ParallelCluster API 堆栈

如果您遇到错误，例如尝试升级 AWS ParallelCluster API 堆栈`UPDATE_FAILED`时，我们建议您在 AWS ParallelCluster Wiki 的 “已知问题” 部分中查看解决方案 GitHub。例如，请参阅 ECR 资源 ParallelCluster 的 API 堆栈升级失败，其中确定了一个可能的问题并提供了缓解选项。

在计算节点初始化过程中看到错误

以下各节提供了计算节点初始化过程中出现错误时的问题排查提示。这包括引导错误、在日志中看到错误，以及如果这些场景都不适用于您的具体情况，该如何处理。

主题

- 在 clustermgtd.log 中看到“Node bootstrap error”
- 我配置了按需容量预留 (ODCRs) 或区域预留实例
- 运行作业失败时在 slurm_resume.log 中看到“An error occurred (VcpuLimitExceeded)”，或创建集群失败时在 clustermgtd.log 中看到该错误
- 运行作业失败时在 slurm_resume.log 中看到“An error occurred (InsufficientInstanceCapacity)”，或创建集群失败时在 clustermgtd.log 中看到该错误
- 看到节点处于 DOWN 状态并显示Reason (Code:InsufficientInstanceCapacity)...
- 在 slurm_resume.log 中看到“cannot change locale (en_US.utf-8) because it has an invalid name”
- 以上情形都不适用于我的情况

在 `clustermgtd.log` 中看到“`Node bootstrap error`”

该问题与计算节点无法引导有关。有关如何调试集群受保护模式问题的信息，请参阅如何调试受保护模式。

我配置了按需容量预留 (ODCRs) 或区域预留实例

ODCRs 其中包括具有多个网络接口的实例，例如 p4d、p4de 和 AWS Trainium (Trn)

在集群配置文件中，检查 HeadNode 是否位于公有子网中，以及计算节点是否位于私有子网中。

ODCRs 是针对 ODCR

尽管我已经按照使用按需容量预留（ODCR）启动实例中的说明准备好了 **/opt/slurm/etc/pcluster/run_instances_overrides.json**，但仍看到“**Unable to read file '/opt/slurm/etc/pcluster/run_instances_overrides.json'.**”

如果您使用的是目标 AWS ParallelCluster 版本3.1.1至3.2.1 ODCRs，并且还使用运行实例覆盖 JSON 文件，则可能您的 JSON 文件格式不正确。您可能会在 clustermgtd.log 中看到错误，例如下面的错误：

```
Unable to read file '/opt/slurm/etc/pcluster/run_instances_overrides.json'.
Using default: {} in  /var/log/parallelcluster/clustermgtd.
```

通过运行以下命令验证 JSON 文件格式是否正确：

```
$ echo /opt/slurm/etc/pcluster/run_instances_overrides.json | jq
```

集群创建失败时在 **clustermgtd.log** 中看到“**Found RunInstances parameters override.**”，或运行作业失败时在 **slurm_resume.log** 中看到该错误

如果您使用的是运行实例覆盖 JSON 文件，请检查是否在 /opt/slurm/etc/pcluster/run_instances_overrides.json 文件中正确设置了队列名称和计算资源名称。

运行作业失败时在 **slurm_resume.log** 中看到“**An error occurred (InsufficientInstanceCapacity)**”，或创建集群失败时在 **clustermgtd.log** 中看到该错误

使用 PG-ODCR（置放群组 ODCR）

创建具有关联置放群组的 ODCR 时，必须在配置文件中使用相同的置放群组名称。在集群配置中设置相应的置放群组名称。

使用区域预留实例

如果您使用区域预留实例并在集群配置中将 PlacementGroup/Enabled 设置为 true，则可能会看到错误，例如下面的错误：

```
We currently do not have sufficient trn1.32xlarge capacity in the Availability Zone
 you requested (us-east-1d). Our system will be working on provisioning additional
 capacity.
You can currently get trn1.32xlarge capacity by not specifying an Availability Zone in
 your request or choosing us-east-1a, us-east-1b, us-east-1c, us-east-1e, us-east-1f.
```

您可能会看到这种情况，因为区域预留实例未放置在同一 UC（或主干）中，这可能会在使用置放群组时导致容量不足错误 (ICEs)。您可以通过在集群配置中禁用 `PlacementGroup` 群组设置来检查这种情况，以确定集群是否可以分配实例。

运行作业失败时在 `slurm_resume.log` 中看到“`An error occurred (VcpuLimitExceeded)`”，或创建集群失败时在 `clustermgtd.log` 中看到该错误

查看您账户上的 vCPU 限制，了解您正在使用的特定 Amazon EC2 实例类型。如果您看到的 v 为零或 CPUs 小于您请求的 v，请申请提高限额。有关如何查看当前限制和申请新限制的信息，请参阅亚马逊 EC2 用户指南中的亚马逊 EC2 服务配额。

运行作业失败时在 `slurm_resume.log` 中看到“`An error occurred (InsufficientInstanceCapacity)`”，或创建集群失败时在 `clustermgtd.log` 中看到该错误

您遇到了容量不足问题。按照 Kn https://aws.amazon.com/premiumsupport/owledge-center/ec2-insufficient-capacity-errors /进行问题故障排除。

看到节点处于 `DOWN` 状态并显示`Reason (Code:InsufficientInstanceCapacity)...`

您遇到了容量不足问题。按照 Kn https://aws.amazon.com/premiumsupport/owledge-center/ec2-insufficient-capacity-errors /进行问题故障排除。有关快速容量不足故障转移模式 AWS ParallelCluster 的更多信息，请参阅。Slurm 集群快速容量不足故障转移

在 `slurm_resume.log` 中看到“`cannot change locale (en_US.utf-8) because it has an invalid name`”

如果 yum 安装过程失败，导致区域设置处于不一致状态，则可能会发生这种情况。例如，当用户终止安装过程时可能会导致这种情况。

要验证原因，请执行下列操作：

- 运行 `su - pcluster-admin`。

  Shell 显示错误，例如“`cannot change locale...no such file or directory`”。
- 运行 `localedef --list`。

  返回空列表或不包含默认区域设置。
- 使用 `yum history` 和 `yum history info #ID` 检查最后的 `yum` 命令。最后 ID 是否包含`Return-Code: Success`？

  如果最后 ID 不包含`Return-Code: Success`，则表明安装后脚本可能未成功运行。

要解决此问题，请尝试使用 `yum reinstall glibc-all-langpacks` 重建区域设置。重建后，如果解决了该问题，则 `su - pcluster-admin` 不会显示错误或警告。

以上情形都不适用于我的情况

要排查计算节点初始化问题，请参阅排查节点初始化问题。

查看您的场景是否包含在 “GitHub 已知问题 AWS ParallelCluster ” 中 GitHub。

集群运行状况指标故障排除

从 3.6.0 AWS ParallelCluster 版开始，集群运行状况指标已添加到 AWS ParallelCluster Amazon CloudWatch 控制面板中。在以下各节中，您可以了解控制面板运行状况指标以及可用于排除和解决问题的操作。

主题

- 看到实例预置错误图表
- 看到运行状况不佳的实例错误图表
- 看到计算实例集空闲时间图表

看到实例预置错误图表

如果您在`Instance Provisioning Errors`图表中看到非零值，则表示用于支持 slurm 节点的 Amazon EC2 实例无法在或 API 上`CreateFleet`启动。`RunInstance`

看到 IAMPolicyErrors

- 发生了什么？

  权限不足导致许多实例启动失败，错误代码为 `UnauthorizedOperation`。

- 如何解决？

  如果您配置了自定义 `InstanceRole` 或 `InstanceProfile`，请检查 IAM 策略并验证使用的凭证是否正确。

  检查 `clustermgtd` 文件以查看静态节点错误详细信息。检查 `slurm_resume.log` 文件以查看动态节点错误详细信息。通过详细信息进一步了解必须添加的缺失权限。

看到 VcpuLimitErrors

- 发生了什么？

  AWS ParallelCluster 无法启动实例，因为它已达到您 AWS 账户 为集群计算节点配置的特定 Amazon EC2 实例类型的 vCPU 限制。

- 如何解决？

  在静态节点的 `clustermgtd` 文件和动态节点的 `slurm_resume.log` 文件中检查 `VcpuLimitExceeded` 错误，以获取更多详细信息。要解决此问题，您可以请求提高 vCPU 限制。有关如何查看当前限制和请求新限制的更多信息，请参阅《Amazon Elastic Compute Cloud 用户指南（适用于 Linux 实例）》中的 Amazon Elastic Compute Cloud 服务限额。

看到 VolumeLimitErrors

- 发生了什么？

  您已达到自己的 Amazon EBS 容量限制 AWS 账户， AWS ParallelCluster 并且无法启动带有错误代码`InsufficientVolumeCapacity`或`VolumeLimitExceeded`的实例。

- 如何解决？

  对静态节点检查 `clustermgtd` 文件，对动态节点检查 `slurm_resume.log` 文件，以获取更多卷限制详细信息。要解决此问题，您可以使用其他卷清理现有卷 AWS 区域，或者联系 AWS 支持中心提交提高 Amazon EBS 卷限制的请求。

看到 `InsufficientCapacityErrors`

- 发生了什么？

  AWS ParallelCluster 没有足够的容量启动 Amazon EC2 实例来支持节点。

- 如何解决？

  对静态节点检查 `clustermgtd` 文件，对动态节点检查 `slurm_resume.log` 文件，以获取容量不足错误的详细信息。要对问题进行故障排除，请按照https://aws.amazon.com/premiumsupport/知识中心/ec2-/中的指导进行操作。insufficient-capacity-errors

`OtherInstanceLaunchFailures`

- 发生了什么？

  用于支持计算节点的 Amazon EC2 实例无法使用`CreateFleet`或 `RunInstance` API 启动。

- 如何解决？

  对静态节点检查 `clustermgtd` 文件，对动态节点检查 `slurm_resume.log` 文件，以获取错误的详细信息。

看到运行状况不佳的实例错误图表

- 发生了什么？

  许多计算实例已启动，但随后因运行状况不佳而终止。

- 如何解决？

  有关排查运行状况不佳的节点的更多信息，请参阅排查意外节点替换和终止问题。

看到 `InstanceBootstrapTimeoutError`

- 发生了什么？

  实例无法在 `resume_timeout` 内（对于动态节点）或 `node_replacement_timeout` 内（对于静态节点）加入集群。如果没有为计算节点正确配置网络，则可能会发生这种情况，或者，如果在计算节点上运行的自定义脚本需要太长时间才能完成，则可能会发生这种情况。

- 如何解决？

  对于动态节点，检查 `clustermgtd` 日志 (`/var/log/parallelcluster/clustermgtd`) 以查看计算节点 IP 地址和错误，例如以下内容：

  ```
  Node bootstrap error: Resume timeout expires for node
  ```

  对于静态节点，检查 `clustermgtd` 日志 (`/var/log/parallelcluster/clustermgtd`) 以查看计算节点 IP 地址和错误，例如以下内容：

  ```
  Node bootstrap error: Replacement timeout expires for node ... in replacement.
  ```

  有关更多详细信息，请检查 `/var/log/cloud-init-output.log` 文件中的错误。您可以从 `clustermgtd` 和 `slurm_resume` 日志文件中检索有问题的计算节点的 IP 地址。

看到 `EC2HealthCheckErrors`

- 发生了什么？

  一个实例未通过 Amazon EC2 运行状况检查。

- 如何解决？

  有关如何排查此问题的信息，请参阅通过故障状态检查来排查实例问题。

看到 `ScheduledEventHealthCheckErrors`

- 发生了什么？

  某个实例未通过 Amazon EC2 预定事件运行状况检查，并且运行状况不佳。

- 如何解决？

  有关如何排查此问题的信息，请参阅实例的计划事件。

看到 `NoCorrespondingInstanceErrors`

- 发生了什么？

AWS ParallelCluster 找不到支持节点的实例。这些节点可能已在引导操作期间自行终止。SlurmQueues/CustomActions/OnNodeStart | OnNodeConfigured 脚本或网络错误可能会产生 NoCorrespondingInstanceErrors。

- 如何解决？

  有关更多详细信息，请检查 /var/log/cloud-init-output.log 以查看计算节点。

看到计算实例集空闲时间图表

看到 MaxDynamicNodeIdleTime 远长于空闲时间缩减阈值

- 发生了什么？

  您的实例未正确终止。 MaxDynamicNodeIdleTime显示由 Amazon EC2 实例支持的动态节点处于空闲状态的最长时间（以秒为单位）。空闲时间缩减阈值源自集群配置 ScaledownIdletime 参数。当计算节点的空闲时间超过空闲时间 Scaledown 秒时，Slurm 关闭节点的电源并 AWS ParallelCluster 终止后备实例。在这种情况下，某些因素会阻止实例终止。

- 如何解决？

  有关此问题的更多信息，请参阅排查扩展问题中的替换、终止或关闭有问题的实例和节点。

排查集群部署问题

如果您的集群创建失败并回滚堆栈创建，则可以通过查看日志文件来诊断问题。失败消息可能类似于以下输出：

```
$ pcluster create-cluster --cluster-name mycluster --region eu-west-1 \
 --cluster-configuration cluster-config.yaml
{
  "cluster": {
    "clusterName": "mycluster",
    "cloudformationStackStatus": "CREATE_IN_PROGRESS",
    "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:xxx:stack/
mycluster/1bf6e7c0-0f01-11ec-a3b9-024fcc6f3387",
    "region": "eu-west-1",
    "version": "3.13.2",
    "clusterStatus": "CREATE_IN_PROGRESS"
  }
```

```
}

$ pcluster describe-cluster --cluster-name mycluster --region eu-west-1
{
  "creationTime": "2021-09-06T11:03:47.696Z",
  ...
  "cloudFormationStackStatus": "ROLLBACK_IN_PROGRESS",
  "clusterName": "mycluster",
  "computeFleetStatus": "UNKNOWN",
  "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:xxx:stack/
mycluster/1bf6e7c0-0f01-11ec-a3b9-024fcc6f3387",
  "lastUpdatedTime": "2021-09-06T11:03:47.696Z",
  "region": "eu-west-1",
  "clusterStatus": "CREATE_FAILED"
}
```

主题

- 查看上 AWS CloudFormation 发生的事件 CREATE_FAILED
- 使用 CLI 查看日志流
- 使用 rollback-on-failure 重新创建失败的集群

查看上 AWS CloudFormation 发生的事件 `CREATE_FAILED`

您可以使用控制台或 AWS ParallelCluster CLI 查看`CREATE_FAILED`错误 CloudFormation 事件，以帮助找到根本原因。

主题

- 在 CloudFormation 控制台中查看事件
- 使用 CLI 查看和筛选 CloudFormation 以下事件 CREATE_FAILED

在 CloudFormation 控制台中查看事件

要查看有关导致该`"CREATE_FAILED"`状态的原因的更多信息，您可以使用 CloudFormation 控制台。

从控制台查看 CloudFormation 错误消息。

1. 登录 AWS Management Console 并导航到 https://console.aws.amazon.com/cloudformation。
2. 选择名为 *`cluster_name`* 的堆栈。

3. 选择事件选项卡。
4. 通过按逻辑 ID 滚动浏览资源事件列表，查看创建失败的资源的状态。如果子任务创建失败，请向后移动，找到失败的资源事件。
5. 例如，如果您看到以下状态消息，则必须使用不会超过当前 vCPU 限制或请求更多 vCPU 容量的实例类型。

```
2022-02-04 16:09:44 UTC-0800 HeadNode CREATE_FAILED 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).
```

使用 CLI 查看和筛选 CloudFormation 以下事件 **CREATE_FAILED**

要诊断集群创建问题，您可以通过筛选 CREATE_FAILED 状态来使用 pcluster get-cluster-stack-events 命令。有关更多信息，请参阅《AWS Command Line Interface 用户指南》中的筛选 AWS CLI 输出。

```
$ pcluster get-cluster-stack-events --cluster-name mycluster --region eu-west-1 \
    --query 'events[?resourceStatus==`CREATE_FAILED`]'
  [
    {
      "eventId": "3ccdedd0-0f03-11ec-8c06-02c352fe2ef9",
      "physicalResourceId": "arn:aws:cloudformation:eu-west-1:xxx:stack/
mycluster/1bf6e7c0-0f02-11ec-a3b9-024fcc6f3387",
      "resourceStatus": "CREATE_FAILED",
      "resourceStatusReason": "The following resource(s) failed to create: [HeadNode].
 ",
      "stackId": "arn:aws:cloudformation:eu-west-1:xxx:stack/
mycluster/1bf6e7c0-0f02-11ec-a3b9-024fcc6f3387",
      "stackName": "mycluster",
      "logicalResourceId": "mycluster",
      "resourceType": "AWS::CloudFormation::Stack",
      "timestamp": "2021-09-06T11:11:51.780Z"
    },
    {
      "eventId": "HeadNode-CREATE_FAILED-2021-09-06T11:11:50.127Z",
      "physicalResourceId": "i-04e91cc1f4ea796fe",
```

```
      "resourceStatus": "CREATE_FAILED",
      "resourceStatusReason": "Received FAILURE signal with UniqueId
 i-04e91cc1f4ea796fe",
      "resourceProperties": "{\"LaunchTemplate\":{\"Version\":\"1\",\"LaunchTemplateId
\":\"lt-057d2b1e687f05a62\"}}",
      "stackId": "arn:aws:cloudformation:eu-west-1:xxx:stack/
mycluster/1bf6e7c0-0f02-11ec-a3b9-024fcc6f3387",
      "stackName": "mycluster",
      "logicalResourceId": "HeadNode",
      "resourceType": "AWS::EC2::Instance",
      "timestamp": "2021-09-06T11:11:50.127Z"
    }
  ]
```

在前面的示例中，失败发生在头节点设置中。

使用 CLI 查看日志流

要调试此类问题，您可以通过筛选 node-type 来使用 pcluster list-cluster-log-streams 列出头节点中的可用日志流，然后分析日志流内容。

```
$ pcluster list-cluster-log-streams --cluster-name mycluster --region eu-west-1 \
--filters 'Name=node-type,Values=HeadNode'
{
  "logStreams": [
    {
      "logStreamArn": "arn:aws:logs:eu-west-1:xxx:log-group:/aws/parallelcluster/
mycluster-202109061103:log-stream:ip-10-0-0-13.i-04e91cc1f4ea796fe.cfn-init",
      "logStreamName": "ip-10-0-0-13.i-04e91cc1f4ea796fe.cfn-init",
      ...
    },
    {
      "logStreamArn": "arn:aws:logs:eu-west-1:xxx:log-group:/aws/parallelcluster/
mycluster-202109061103:log-stream:ip-10-0-0-13.i-04e91cc1f4ea796fe.chef-client",
      "logStreamName": "ip-10-0-0-13.i-04e91cc1f4ea796fe.chef-client",
      ...
    },
    {
      "logStreamArn": "arn:aws:logs:eu-west-1:xxx:log-group:/aws/parallelcluster/
mycluster-202109061103:log-stream:ip-10-0-0-13.i-04e91cc1f4ea796fe.cloud-init",
      "logStreamName": "ip-10-0-0-13.i-04e91cc1f4ea796fe.cloud-init",
      ...
```

```
    },
    ...
  ]
}
```

可用于查找初始化错误的两个主要日志流如下：

- cfn-init 是 cfn-init 脚本的日志。首先检查此日志流。在此日志中，您可能会看到“Command chef failed”错误。查看此行前面的几行，了解与该错误消息相关的更多细节。有关更多信息，请参阅 cfn-init。
- cloud-init 是 cloud-init 的日志。如果您在 cfn-init 中没有看到任何内容，请接下来尝试查看此日志。

您可以使用 pcluster get-cluster-log-events 来检索日志流的内容（请注意使用 --limit 5 选项来限制检索到的事件数量）：

```
$ pcluster get-cluster-log-events --cluster-name mycluster \
  --region eu-west-1 --log-stream-name ip-10-0-0-13.i-04e91cc1f4ea796fe.cfn-init \
  --limit 5
{
  "nextToken": "f/36370880979637159565202782352491087067973952362220945409/s",
  "prevToken": "b/36370880752972385367337528725601470541902663176996585497/s",
  "events": [
    {
      "message": "2021-09-06 11:11:39,049 [ERROR] Unhandled exception during build:
 Command runpostinstall failed",
      "timestamp": "2021-09-06T11:11:39.049Z"
    },
    {
      "message": "Traceback (most recent call last):\n  File \"/opt/aws/bin/
cfn-init\", line 176, in <module>\n    worklog.build(metadata, configSets)\n
 File \"/usr/lib/python3.7/site-packages/cfnbootstrap/construction.py\", line
 135, in build\n    Contractor(metadata).build(configSets, self)\n  File \"/
usr/lib/python3.7/site-packages/cfnbootstrap/construction.py\", line 561, in
 build\n    self.run_config(config, worklog)\n  File \"/usr/lib/python3.7/
site-packages/cfnbootstrap/construction.py\", line 573, in run_config\n
 CloudFormationCarpenter(config, self._auth_config).build(worklog)\n  File \"/usr/
lib/python3.7/site-packages/cfnbootstrap/construction.py\", line 273, in build\n
   self._config.commands)\n  File \"/usr/lib/python3.7/site-packages/cfnbootstrap/
command_tool.py\", line 127, in apply\n    raise ToolError(u\"Command %s failed\" %
 name)",
```

```
      "timestamp": "2021-09-06T11:11:39.049Z"
    },
    {
      "message": "cfnbootstrap.construction_errors.ToolError: Command runpostinstall
 failed",
      "timestamp": "2021-09-06T11:11:39.049Z"
    },
    {
      "message": "2021-09-06 11:11:49,212 [DEBUG] CloudFormation client initialized
 with endpoint https://cloudformation.eu-west-1.amazonaws.com",
      "timestamp": "2021-09-06T11:11:49.212Z"
    },
    {
      "message": "2021-09-06 11:11:49,213 [DEBUG] Signaling resource HeadNode in stack
 mycluster with unique ID i-04e91cc1f4ea796fe and status FAILURE",
      "timestamp": "2021-09-06T11:11:49.213Z"
    }
  ]
}
```

在前面的示例中，失败是由 `runpostinstall` 失败导致的，因此它与 `CustomActions` 的 `OnNodeConfigured` 配置参数中使用的自定义引导脚本的内容紧密相关。

使用 `rollback-on-failure` 重新创建失败的集群

AWS ParallelCluster 在 CloudWatch 日志组中创建集群日志流。您可以在控制台的 “自定义 CloudWatch 控制面板” 或 “日志” 组中查看这些日志。有关更多信息，请参阅与 Amazon CloudWatch 日志集成和 亚马逊 CloudWatch 控制面板：如果没有可用的日志流，则失败可能是由 `CustomActions` 自定义引导脚本或 AMI 相关问题导致的。要在这种情况下诊断创建问题，请使用 `pcluster create-cluster`（包含设置为 `false` 的 `--rollback-on-failure` 参数）重新创建集群。然后，使用 SSH 查看集群，如以下所示：

```
$ pcluster create-cluster --cluster-name mycluster --region eu-west-1 \
   --cluster-configuration cluster-config.yaml --rollback-on-failure false
 {
   "cluster": {
     "clusterName": "mycluster",
     "cloudformationStackStatus": "CREATE_IN_PROGRESS",
     "cloudformationStackArn": "arn:aws:cloudformation:eu-west-1:xxx:stack/
mycluster/1bf6e7c0-0f01-11ec-a3b9-024fcc6f3387",
     "region": "eu-west-1",
     "version": "3.13.2",
```

```
    "clusterStatus": "CREATE_IN_PROGRESS"
  }
}
$ pcluster ssh --cluster-name mycluster
```

登录到头节点后，您应该可以找到三个主要的日志文件，可以用它们来查找错误。

- `/var/log/cfn-init.log` 是 `cfn-init` 脚本的日志。首先查看此日志。在此日志中，您可能会看到类似“`Command chef failed`”的错误。查看此行前面的几行，了解与该错误消息相关的更多细节。有关更多信息，请参阅 cfn-init。
- `/var/log/cloud-init.log` 是 cloud-init 的日志。如果您在 `cfn-init.log` 中没有看到任何内容，请接下来尝试查看此日志。
- `/var/log/cloud-init-output.log` 是 cloud-init 运行的命令的输出。这包括 `cfn-init` 的输出。在大多数情况下，排查此类问题无需查看此日志。

排查使用 Terraform 部署集群的问题

本节与使用 Terraform 部署的集群相关。

ParallelCluster 未找到 API

由于找不到 ParallelCluster API，计划可能会失败。在这种情况下，返回的错误信息如下：

```
Planning failed. Terraform encountered an error while generating this plan.

#
# Error: Unable to retrieve ParallelCluster API cloudformation stack.
#
#   with provider["registry.terraform.io/aws-tf/aws-parallelcluster"],
#   on providers.tf line 6, in provider "aws-parallelcluster":
#    6: provider "aws-parallelcluster" {
#
# operation error CloudFormation: DescribeStacks, https response error StatusCode: 400,
 RequestID: REQUEST_ID, api error ValidationError: Stack with id PCAPI_STACK_NAME does
 not exist
```

要解决此错误，请在要创建集群的账户中部署 ParallelCluster API。请参阅the section called “使用 Terraform 创建集群”。

用户无权调用 ParallelCluster API

计划可能会失败，因为 role/user 您假设用于部署 Terraform 项目的 IAM 没有与 API 交互的 ParallelCluster 权限。在这种情况下，返回的错误信息如下：

```
Planning failed. Terraform encountered an error while generating this plan.

# Error: 403 Forbidden
#
#   with
 module.parallelcluster_clusters.module.clusters[0].pcluster_cluster.managed_configs["DemoClust
#   on .terraform/modules/parallelcluster_clusters/modules/clusters/main.tf line 35, in
 resource "pcluster_cluster" "managed_configs":
#   35: resource "pcluster_cluster" "managed_configs" {
#
# {{"Message":"User: USER_ARN is not authorized to perform: execute-api:Invoke on
 resource: PC_API_REST_RESOURCE with an explicit deny"}
# }
```

要解决此错误，请配置 ParallelCluster 提供商，使其使用 ParallelCluster API 角色与 API 交互。

```
provider "aws-parallelcluster" {
  region         = var.region
  profile        = var.profile
  api_stack_name = var.api_stack_name
  **use_user_role** **= true**
}
```

排查扩展问题

本节与使用 3.0.0 及更高 AWS ParallelCluster 版本使用Slurm作业调度程序安装的集群相关。有关配置多个队列的更多信息，请参阅多个队列的配置。

如果您的一个正在运行的集群遇到问题，请在开始排查问题之前，通过运行以下命令将该集群置于 STOPPED 状态。这样可以防止产生任何意外成本。

```
$ pcluster update-compute-fleet --cluster-name mycluster \
   --status STOP_REQUESTED
```

您可以使用 `pcluster list-cluster-log-streams` 命令，并通过使用其中一个失败节点或头节点的 `private-dns-name` 进行筛选，列出集群节点中可用的日志流。

```
$ pcluster list-cluster-log-streams --cluster-name mycluster --region eu-west-1 \
 --filters 'Name=private-dns-name,Values=ip-10-0-0-101'
```

然后，您可以通过使用 `pcluster get-cluster-log-events` 命令并传递与下一节中提及的其中一个关键日志相对应的 `--log-stream-name`，检索日志流的内容并进行分析：

```
$ pcluster get-cluster-log-events --cluster-name mycluster \
--region eu-west-1 --log-stream-name ip-10-0-0-13.i-04e91cc1f4ea796fe.cfn-init
```

AWS ParallelCluster 在 CloudWatch 日志组中创建集群日志流。您可以在控制台的 “自定义 CloudWatch 控制面板” 或 “日志” 组中查看这些日志。有关更多信息，请参阅与 Amazon CloudWatch 日志集成和亚马逊 CloudWatch 控制面板。

主题

- 用于调试的关键日志
- 运行作业失败时在 slurm_resume.log 中看到“InsufficientInstanceCapacity”错误，或创建集群失败时在 clustermgtd.log 中看到该错误
- 排查节点初始化问题
- 排查意外节点替换和终止问题
- 替换、终止或关闭有问题的实例和节点
- 队列（分区）Inactive 状态
- 排查其他已知的节点和作业问题

用于调试的关键日志

下表概述了头节点的关键日志：

- `/var/log/cfn-init.log`- AWS CloudFormation 这是初始化日志。其中包含设置实例时运行的所有命令。可以用它来排查初始化问题。
- `/var/log/chef-client.log`：这是 Chef 客户端日志。其中包含通过 Chef/CINC 运行的所有命令。可以用它来排查初始化问题。
- `/var/log/parallelcluster/slurm_resume.log`：这是 `ResumeProgram` 日志。它启动动态节点的实例。可以用它来排查动态节点启动问题。

- /var/log/parallelcluster/slurm_suspend.log：这是 SuspendProgram 日志。在终止动态节点的实例时会调用该日志。可以用它来排查动态节点终止问题。查看此日志时，还应检查 clustermgtd 日志。
- /var/log/parallelcluster/clustermgtd：这是 clustermgtd 日志。它作为集中式进程守护程序运行，用于管理大多数集群操作。可以用它来排查任何启动、终止或集群操作问题。
- /var/log/slurmctld.log-这是Slurm控制守护程序日志。 AWS ParallelCluster 不会做出扩展决策。相反，它只会尝试启动资源来满足 Slurm 的要求。它可用于排查扩展和分配问题、与作业相关的问题以及与调度器相关的任何启动和终止问题。
- /var/log/parallelcluster/compute_console_output：此日志记录意外终止的静态计算节点样本子集的控制台输出。如果静态计算节点终止并且计算节点日志在中不可用，则使用此日志 CloudWatch。当您使用 Amazon EC2 控制台或检索实例控制台输出时 AWS CLI ，您收到的compute_console_output log内容是相同的。

以下是计算节点的关键日志：

- /var/log/cloud-init-output.log：这是 cloud-init 日志。其中包含设置实例时运行的所有命令。可以用它来排查初始化问题。
- /var/log/parallelcluster/computemgtd：这是 computemgtd 日志。它在每个计算节点上运行，用于在头节点上的 clustermgtd 进程守护程序离线的不常见事件中监控节点。可以用它来排查意外终止问题。
- /var/log/slurmd.log - 这是 Slurm 计算进程守护程序日志。可以用它来排查初始化和计算失败问题。

运行作业失败时在 slurm_resume.log 中看到“InsufficientInstanceCapacity”错误，或创建集群失败时在 clustermgtd.log 中看到该错误

如果集群使用 Slurm 调度器，则会遇到容量不足问题。如果发出实例启动请求时没有足够的可用实例，则会返回 InsufficientInstanceCapacity 错误。

对于静态实例容量，您可以在 /var/log/parallelcluster/clustermgtd 中的 clustermgtd 日志中找到该错误。

对于动态实例容量，您可以在 /var/log/parallelcluster/slurm_resume.log 中的 ResumeProgram 日志中找到该错误。

消息与以下示例类似：

```
An error occurred (InsufficientInstanceCapacity) when calling the RunInstances/
CreateFleet operation...
```

根据您的用例，请考虑使用以下方法之一来避免收到这些类型的错误消息：

- 如果启用了置放群组，则将其禁用。有关更多信息，请参阅 置放群组和实例启动问题。
- 为实例预留容量并使用 ODCR（按需容量预留）启动这些实例。有关更多信息，请参阅 使用按需容量预留（ODCR）启动实例。
- 配置多个具有不同实例类型的计算资源。如果您的工作负载不需要特定的实例类型，则可以对多个计算资源利用快速容量不足故障转移。有关更多信息，请参阅 Slurm 集群快速容量不足故障转移。
- 在同一个计算资源中配置多种实例类型，并利用多实例类型分配。有关配置多个实例的更多信息，请参阅 Slurm 的多实例类型分配和 `Scheduling`/`SlurmQueues`/`ComputeResources`/`Instances`。
- 通过在集群配置 `Scheduling`/`SlurmQueues`/`Networking`/`SubnetIds` 中更改子网 ID，将队列移动到不同的可用区。
- 如果您的工作负载没有紧密耦合，则将队列分散到不同的可用区。有关配置多个子网的更多信息，请参阅 `Scheduling`/`SlurmQueues`/`Networking`/`SubnetIds`。

排查节点初始化问题

本节介绍如何排查节点初始化问题。这包括节点无法启动、开机或加入集群的问题。

主题

- 头节点
- 计算节点

头节点

适用日志：

- `/var/log/cfn-init.log`
- `/var/log/chef-client.log`
- `/var/log/parallelcluster/clustermgtd`
- `/var/log/parallelcluster/slurm_resume.log`

- /var/log/slurmctld.log

检查 /var/log/cfn-init.log 和 /var/log/chef-client.log 日志或相应的日志流。这些日志包含设置头节点时运行的所有操作。设置过程中发生的大多数错误的错误消息应该都包含在 /var/log/chef-client.log 日志中。如果在集群的配置中指定了 OnNodeStart 或 OnNodeConfigured 脚本，请通过日志消息仔细检查脚本是否成功运行。

创建集群时，头节点必须等待计算节点加入集群，然后才能加入集群。因此，如果计算节点加入集群失败，则头节点也会失败。根据您使用的计算节点的类型，您可以按照其中一组过程来排查此类问题：

计算节点

- 适用日志：
  - /var/log/cloud-init-output.log
  - /var/log/slurmd.log
- 如果启动了计算节点，请先检查 /var/log/cloud-init-output.log，其中应包含类似于头节点 /var/log/chef-client.log 日志的设置日志。设置过程中发生的大多数错误的错误消息应该都包含在 /var/log/cloud-init-output.log 日志中。如果在集群配置中指定了预安装或安装后脚本，请检查它们是否成功运行。
- 如果您使用的是修改了 Slurm 配置的自定义 AMI，则可能存在阻止计算节点加入集群的 Slurm 相关错误。对于与调度器相关的错误，请检查 /var/log/slurmd.log 日志。

动态计算节点：

- 搜索计算节点名称的 ResumeProgram 日志 (/var/log/parallelcluster/slurm_resume.log) 以查看是否对该节点调用过 ResumeProgram。（如果ResumeProgram从未被调用，则可以查看slurmctld日志 (/var/log/slurmctld.log) 以确定是否尝试Slurm过ResumeProgram使用该节点进行调用）。
- 请注意，ResumeProgram 的权限不正确可能会导致 ResumeProgram 静默失败。如果您使用的是修改了 ResumeProgram 设置的自定义 AMI，请检查该 ResumeProgram 是否由 slurm 用户拥有并具有 744 (rwxr--r--) 权限。
- 如果调用了 ResumeProgram，请查看是否为该节点启动了实例。如果未启动任何实例，则会看到一条描述启动失败的错误消息。
- 如果启动了实例，则在设置过程中可能出现了问题。您应该会从 ResumeProgram 日志中看到相应的私有 IP 地址和实例 ID。此外，您可以查看特定实例的相应设置日志。有关排查计算节点设置错误的更多信息，请参阅下一节。

静态计算节点：

- 检查 `clustermgtd` (`/var/log/parallelcluster/clustermgtd`) 日志，查看是否为该节点启动了实例。如果它们没有启动，则应该有明确的错误消息详细说明启动失败的原因。
- 如果启动了实例，则表示设置过程中出现了问题。您应该会从 `ResumeProgram` 日志中看到相应的私有 IP 地址和实例 ID。此外，您可以查看特定实例的相应设置日志。

由竞价型实例支持的计算节点：

- 如果这是您第一次使用竞价型实例，并且作业保持在 PD（待处理状态），请仔细检查 `/var/log/parallelcluster/slurm_resume.log` 文件。您可能会发现类似下面的错误：

  ```
  2022-05-20 13:06:24,796 - [slurm_plugin.common:add_instances_for_nodes] - ERROR -
   Encountered exception when launching instances for nodes (x1) ['spot-dy-t2micro-2']:
   An error occurred (AuthFailure.ServiceLinkedRoleCreationNotPermitted) when calling
   the RunInstances operation: The provided credentials do not have permission to
   create the service-linked role for Amazon EC2 Spot Instances.
  ```

  使用竞价型实例时，您的账户中必须存在 `AWSServiceRoleForEC2Spot` 服务相关角色。要使用在您的账户中创建此角色 AWS CLI，请运行以下命令：

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

  有关更多信息，请参阅 Amazon AWS ParallelCluster EC2 用户指南使用竞价型实例中的用户指南和竞价型实例请求的服务相关角色。

排查意外节点替换和终止问题

本节继续探讨如何排查节点相关问题，特别是在节点意外替换或终止时。

- 适用日志：
  - `/var/log/parallelcluster/clustermgtd`（头节点）
  - `/var/log/slurmctld.log`（头节点）
  - `/var/log/parallelcluster/computemgtd`（计算节点）

节点意外替换或终止

- 检查 clustermgtd 日志 (/var/log/parallelcluster/clustermgtd) 以查看 clustermgtd 是否替换或终止了节点。请注意，clustermgtd 处理所有正常的节点维护操作。
- 如果 clustermgtd 替换或终止了该节点，则应该会有详细说明为何对该节点执行此操作的消息。如果原因与调度器有关（例如，因为节点处于 DOWN 状态），请查看 slurmctld 日志以获取更多信息。如果原因 EC2 与亚马逊有关，则应附上信息性消息，详细说明需要更换的亚马逊 EC2 相关问题。
- 如果clustermgtd没有终止节点，请先检查这是否是 Amazon 的预期终止 EC2 ，更具体地说是即时终止。 computemgtd，在计算节点上运行，如果节点clustermgtd被确定为运行状况不佳，也可以终止节点。检查 computemgtd 日志 (/var/log/parallelcluster/computemgtd) 以查看 computemgtd 是否终止了该节点。

节点失败

- 检查 slurmctld 日志 (/var/log/slurmctld.log) 以查看作业或节点失败的原因。请注意，如果节点失败，作业会自动重新排队。
- 如果slurm_resume报告该节点已启动，并在几分钟后clustermgtd报告说 Amazon EC2 中没有该节点的相应实例，则该节点可能会在设置过程中失败。要从计算 (/var/log/cloud-init-output.log) 中检索日志，请执行以下步骤：
  - 提交一项作业以便让 Slurm 启动一个新节点。
  - 等待计算节点启动。
  - 修改实例启动的关闭行为，以便停止而不是终止出现故障的计算节点。

```
$ aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --instance-initiated-shutdown-behavior "{\"Value\": \"stop\"}"
```

  - 启用终止保护。

```
$ aws ec2 modify-instance-attribute \
    --instance-id i-1234567890abcdef0 \
    --disable-api-termination
```

  - 将节点标记为易于识别。

```
$ aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=Name,Value=QUARANTINED-Compute
```

- 通过更改 parallelcluster:cluster-name 标签将节点与集群分离。

```
$ aws ec2 create-tags \
    --resources i-1234567890abcdef0 \
    --tags Key=parallelcluster:clustername,Value=QUARANTINED-ClusterName
```

- 使用以下命令从该节点检索控制台输出。

```
$ aws ec2 get-console-output --instance-id i-1234567890abcdef0 --output text
```

替换、终止或关闭有问题的实例和节点

- 适用日志：
  - /var/log/parallelcluster/clustermgtd（头节点）
  - /var/log/parallelcluster/slurm_suspend.log（头节点）
- 在大多数情况下，clustermgtd 会处理所有预期的实例终止操作。检查 clustermgtd 日志以查看其无法替换或终止节点的原因。
- 对于 SlurmSettings 属性 失败的动态节点，请检查 SuspendProgram 日志以查看 slurmctld 是否以特定节点作为参数调用了 SuspendProgram。请注意，SuspendProgram 实际上并不执行任何操作，它只是记录被调用时的时间。所有实例终止和 NodeAddr 重置均由 clustermgtd 完成。经过 SuspendTimeout 之后，Slurm 会自动将节点置回 POWER_SAVING 状态。
- 如果计算节点由于引导失败而持续失败，请验证它们是否在启用了 Slurm 集群保护模式的情况下启动。如果未启用受保护模式，请修改受保护模式设置以启用受保护模式。排查并修复引导脚本问题。

队列（分区）Inactive 状态

如果您运行 sinfo 并且输出显示队列的 AVAIL 状态为 inact，则您的集群可能启用了 Slurm 集群保护模式，并且队列已在预定义的时间段内设置为 INACTIVE 状态。

排查其他已知的节点和作业问题

另一种已知问题是 AWS ParallelCluster 可能无法分配工作岗位或做出扩展决策。对于此类问题， AWS ParallelCluster 只按照 Slurm 的指示启动、终止或维护资源。对于这些问题，请查看 slurmctld 日志以排查问题。

置放群组和实例启动问题

为了获得最低的节点间延迟，请使用置放群组。置放群组可确保您的实例位于同一网络主干中。如果发出请求时没有足够的可用实例，则会返回 `InsufficientInstanceCapacity` 错误。要减少在使用集群置放群组时收到此错误的可能性，请将 `SlurmQueues`/`Networking`/`PlacementGroup`/`Enabled` 参数设置为 `false`。

要进一步控制容量访问权限，请考虑使用 ODCR（按需容量预留）启动实例。

有关更多信息，请参阅 Amazon Linux 实例 EC2 用户指南中的排查实例启动问题和置放群组角色和限制。

替换目录

有些目录无法替换。如果您在替换目录时遇到问题，可能就是这种情况。以下目录在节点之间共享，无法替换。

- `/opt/intel`：这包括 Intel MPI、Intel Parallel Studio 和相关文件。
- `/opt/slurm` - 这包括 Slurm 工作负载管理器和相关文件。（有条件，仅当 `Scheduler: slurm` 时。）

排查 Amazon DCV 中的问题

主题

- Amazon DCV 的日志
- Ubuntu Amazon DCV 问题

Amazon DCV 的日志

Amazon DCV 的日志将写入 `/var/log/dcv/` 目录的文件中。查看这些日志有助于排查问题。

实例类型应至少有 1.7 吉字节（GiB）的 RAM 才能运行 Amazon DCV。Nano 和 Micro 实例类型没有足够的内存来运行 Amazon DCV。

AWS ParallelCluster 在日志组中创建 Amazon DCV 日志流。您可以在控制台的 “自定义 CloudWatch 控制面板” 或 “日志” 组中查看这些日志。有关更多信息，请参阅与 Amazon CloudWatch 日志集成和亚马逊 CloudWatch 控制面板。

Ubuntu Amazon DCV 问题

在 Ubuntu 上对 Amazon DCV 会话运行 Gnome 终端时，您可能不会自动拥有访问 AWS ParallelCluster 通过登录 Shell 提供的用户环境的权限。该用户环境提供 openmpi 或 intelmpi 等环境模块以及其他用户设置。

Gnome 终端的默认设置会阻止 Shell 作为登录 Shell 启动。这意味着 shell 配置文件不会自动获取，也不会加载 AWS ParallelCluster 用户环境。

要正确获取外壳配置文件并访问 AWS ParallelCluster 用户环境，请执行以下操作之一：

- 更改默认终端设置：
  1. 在 Gnome 终端中选择编辑菜单。
  2. 选择首选项，然后选择配置文件。
  3. 选择命令，然后选择作为登录 Shell 运行命令。
  4. 打开新终端。
- 使用命令行获取可用的配置文件：

```
$ source /etc/profile && source $HOME/.bashrc
```

通过 AWS Batch 集成对集群中的问题进行故障排除

本节为集成了 AWS Batch 调度程序的集群提供了可能的故障排除技巧，特别是头节点问题、计算问题、作业失败和超时错误。

主题

- 头节点问题
- 计算问题
- 作业失败
- 端点 URL 连接超时错误

头节点问题

您可以像排查 Slurm 集群问题一样排查头节点设置问题（Slurm 特定日志除外）。有关这些问题的更多信息，请参阅头节点。

计算问题

AWS Batch 管理服务的扩展和计算方面。如果您遇到与计算相关的问题，请参阅 AWS Batch 故障排除文档以获取帮助。

作业失败

如果作业失败，您可以运行 `awsbout` 命令来检索作业输出。您也可以运行`awsbstat`命令以获取指向 Amazon 存储的任务日志的链接 CloudWatch。

端点 URL 连接超时错误

如果多节点并行作业失败并显示错误：`Connect timeout on endpoint URL`：

- 在 `awsbout` 输出日志中，从输出中检查作业是否为多节点并行作业：`Detected 3/3 compute nodes. Waiting for all compute nodes to start.`
- 验证计算节点子网是否为公有子网。

在中使用时，多节点 p AWS Batch arallel 作业不支持使用公有子网。 AWS ParallelCluster请为计算节点和作业使用私有子网。有关更多信息，请参阅 AWS Batch User Guide 中的 Compute environment considerations。要为您的计算节点配置私有子网，请参阅AWS ParallelCluster 使用调 AWS Batch 度器。

排查与 Active Directory 的多用户集成问题

本节内容涉及与 Active Directory 集成的集群。

如果 Active Directory 集成功能未按预期运行，则 SSSD 日志可以提供有用的诊断信息。这些日志位于集群节点上的 `/var/log/sssd` 中。默认情况下，它们还存储在集群的 Amazon CloudWatch 日志组中。

主题

- 特定于 Active Directory 的问题排查
- 启用调试模式
- 如何从 LDAPS 迁移到 LDAP
- 如何禁用 LDAPS 服务器证书验证

- 如何使用 SSH 密钥而不是密码进行登录
- 如何重置用户密码和过期的密码
- 如何验证加入的域
- 如何排查证书问题
- 如何验证与 Active Directory 的集成是否正常工作
- 如何排查计算节点登录问题
- 多用户环境中 SimCenter StarCCM+ 作业的已知问题
- 用户名解析的已知问题
- 如何解决主目录创建问题

特定于 Active Directory 的问题排查

本节与特定于 Active Directory 类型的问题排查有关。

Simple AD

- `DomainReadOnlyUser` 值必须与用户的 Simple AD 目录基础搜索相匹配：

  `cn=ReadOnlyUser,cn=Users,dc=corp,dc=example,dc=com`

  注意 `Users` 的 `cn`。
- 默认管理员用户是 `Administrator`。
- `Ldapsearch` 要求在用户名前面加上 NetBIOS 名称。

  `Ldapsearch` 语法必须如下所示：

```
$ ldapsearch -x -D "corp\\Administrator" -w "Password" -H ldap://192.0.2.103 \
   -b "cn=Users,dc=corp,dc=example,dc=com"
```

AWS Managed Microsoft AD

- 该`DomainReadOnlyUser`值必须与基于 AWS Managed Microsoft AD 目录的用户搜索相匹配：

  `cn=ReadOnlyUser,ou=Users,ou=CORP,dc=corp,dc=example,dc=com`
- 默认管理员用户是 `Admin`。

- Ldapsearch 语法必须如下所示：

```
$ ldapsearch -x -D "Admin" -w "Password" -H ldap://192.0.2.103 \
   -b "ou=Users,ou=CORP,dc=corp,dc=example,dc=com"
```

启用调试模式

SSSD 提供的调试日志可用于排查问题。要启用调试模式，您必须在对集群配置进行以下更改后更新集群：

```
DirectoryService:
  AdditionalSssdConfigs:
    debug_level: "0x1ff"
```

如何从 LDAPS 迁移到 LDAP

不鼓励从 LDAPS（采用 TLS/SSL 的 LDAP）迁移到 LDAP，因为 LDAP 本身不提供任何加密。但可以使用它来进行测试和问题排查。

您可以使用先前的配置定义更新集群，从而将集群还原到以前的配置。

要从 LDAPS 迁移到 LDAP，必须在集群配置中进行以下更改，然后更新集群：

```
DirectoryService:
  LdapTlsReqCert: never
  AdditionalSssdConfigs:
    ldap_auth_disable_tls_never_use_in_production: True
```

如何禁用 LDAPS 服务器证书验证

出于测试或问题排查目的，在头节点上暂时禁用 LDAPS 服务器证书验证可能很有用。

您可以使用先前的配置定义更新集群，从而将集群还原到以前的配置。

要禁用 LDAPS 服务器证书验证，必须在集群配置中进行以下更改，然后更新集群：

```
DirectoryService:
  LdapTlsReqCert: never
```

如何使用 SSH 密钥而不是密码进行登录

SSH 密钥是在您首次使用密码登录后在 /home/$user/.ssh/id_rsa 中创建的。要使用 SSH 密钥登录，必须先使用密码登录，在本地复制 SSH 密钥，然后像往常一样用它进行无密码的 SSH 登录：

```
$ ssh -i $LOCAL_PATH_TO_SSH_KEY $username@$head_node_ip
```

如何重置用户密码和过期的密码

如果用户无法访问集群，其 AWS Managed Microsoft AD 密码可能已过期。

要重置密码，请使用对目录具有写入权限的用户和角色运行以下命令：

```
$ aws ds reset-user-password \
  --directory-id "d-abcdef01234567890" \
  --user-name "USER_NAME" \
  --new-password "NEW_PASSWORD" \
  --region "region-id"
```

如果您重置 DirectoryService/DomainReadOnlyUser 的密码：

1. 确保使用新密码更新 DirectoryService/PasswordSecretArn 密钥。
2. 更新集群以获取新密钥值：
   a. 使用 pcluster update-compute-fleet 命令停止计算实例集。
   b. 在集群头节点中运行以下命令。

   ```
   $ sudo /opt/parallelcluster/scripts/directory_service/
   update_directory_service_password.sh
   ```

密码重置和集群更新后，用户的集群访问权限应会恢复。

有关更多信息，请参阅 AWS Directory Service Administration Guide 中的 Reset a user password。

如何验证加入的域

必须从加入到域的实例（而非头节点）中运行以下命令。

```
$ realm list corp.example.com \
```

```
  type: kerberos \
  realm-name: CORP.EXAMPLE.COM \
  domain-name: corp.example.com \
  configured: kerberos-member \
  server-software: active-directory \
  client-software: sssd \
  required-package: oddjob \
  required-package: oddjob-mkhomedir \
  required-package: sssd \
  required-package: adcli \
  required-package: samba-common-tools \
  login-formats: %U \
  login-policy: allow-realm-logins
```

如何排查证书问题

当 LDAPS 通信不起作用时，可能是由于 TLS 通信中的错误造成的，这反过来可能是由于证书问题造成的。

有关证书的说明：

- 集群配置 `LdapTlsCaCert` 中指定的证书必须是一个 PEM 证书捆绑包，其中包含为域控制器颁发证书的整个证书颁发机构 (CA) 链的证书。
- PEM 证书捆绑包是由 PEM 证书串联而成的文件。
- PEM 格式的证书（通常用于 Linux）等同于 base64 DER 格式的证书（通常由 Windows 导出）。
- 如果域控制器的证书由从属 CA 颁发，则证书捆绑包必须同时包含从属 CA 和根 CA 的证书。

问题排查验证步骤：

以下验证步骤假定从集群头节点内运行命令，并且可以在 *SERVER*:*PORT* 访问域控制器。

要排查与证书有关的问题，请按照以下验证步骤操作：

验证步骤：

1. 检查与 Active Directory 域控制器的连接：

   验证是否可以连接到域控制器。如果此步骤成功，则与域控制器的 SSL 连接成功并验证了证书。您的问题与证书无关。

   如果此步骤失败，请继续进行下一步验证。

```
$ openssl s_client -connect SERVER:PORT -CAfile PATH_TO_CA_BUNDLE_CERTIFICATE
```

2. 检查证书验证：

   验证本地 CA 证书捆绑包是否可以验证域控制器提供的证书。如果此步骤成功，则您的问题与证书无关，而是与其他网络问题有关。

   如果此步骤失败，请继续进行下一步验证。

```
$ openssl verify -verbose -
CAfile PATH_TO_CA_BUNDLE_CERTIFICATE PATH_TO_A_SERVER_CERTIFICATE
```

3. 检查 Active Directory 域控制器提供的证书：

   验证域控制器提供的证书的内容是否符合预期。如果此步骤成功，则用于验证控制器的 CA 证书可能有问题，请转到下一个问题排查步骤。

   如果此步骤失败，则必须更正为域控制器颁发的证书并重新执行问题排查步骤。

```
$ openssl s_client -connect SERVER:PORT -showcerts
```

4. 检查证书的内容：

   验证域控制器提供的证书的内容是否符合预期。如果此步骤成功，则用于验证控制器的 CA 证书可能有问题，请转到下一个问题排查步骤。

   如果此步骤失败，则必须更正为域控制器颁发的证书并重新运行问题排查步骤。

```
$ openssl s_client -connect SERVER:PORT -showcerts
```

5. 查看本地 CA 证书捆绑包的内容：

   验证用于验证域控制器证书的本地 CA 证书捆绑包的内容是否符合预期。如果此步骤成功，则域控制器提供的证书可能有问题。

   如果此步骤失败，则必须更正为域控制器颁发的 CA 证书捆绑包并重新运行问题排查步骤。

```
$ openssl x509 -in PATH_TO_A_CERTIFICATE -text
```

如何验证与 Active Directory 的集成是否正常工作

如果以下两项检查成功，则表示与 Active Directory 的集成工作正常。

检查：

1. 您可以发现目录中定义的用户：

   从集群头节点中以 `ec2-user` 身份运行以下命令：

   ```
   $ getent passwd $ANY_AD_USER
   ```

2. 您可以通过提供用户密码 SSH 登录到头节点：

   ```
   $ ssh $ANY_AD_USER@$HEAD_NODE_IP
   ```

如果第一项检查失败，预计第二项检查也会失败。

其他问题排查检查：

- 验证目录中是否存在该用户。
- 启用调试日志记录。
- 考虑通过从 LDAPS 迁移到 LDAP暂时禁用加密，以排除 LDAPS 问题。

如何排查计算节点登录问题

本节内容涉及登录到与 Active Directory 集成的集群中的计算节点。

使用 AWS ParallelCluster时，集群计算节点的密码登录在设计上是禁用的。

所有用户都必须使用自己的 SSH 密钥登录到计算节点。

如果在集群配置中启用了 `GenerateSshKeysForUsers`，则用户可以在首次身份验证（例如登录）后在头节点中检索到其 SSH 密钥。

当用户首次在头节点上进行身份验证时，他们可以检索到作为目录用户自动为他们生成的 SSH 密钥。还会创建该用户的主目录。当 sudo-user 第一次切换到头节点中的用户时，也可能发生这种情况。

如果用户未登录到头节点，则不会生成 SSH 密钥，用户将无法登录到计算节点。

多用户环境中 SimCenter StarCCM+ 作业的已知问题

本节内容与在 Siemens 的 Simcenter StarCCM+ 计算流体动力学软件提供的多用户环境中启动的作业有关。

如果您运行配置为使用嵌入式 IntelMPI 的 StarCCM+ v16 作业，则默认使用 SSH 引导 MPI 进程。

由于已知的导致用户名解析错误的 Slurm 错误，作业可能会失败并显示类似于“`error setting up the bootstrap proxies`”的错误。此错误仅影响 AWS ParallelCluster 版本 3.1.1 和 3.1.2。

为了防止这种情况发生，请强制 IntelMPI 使用 Slurm 作为 MPI 引导方法。按照 IntelMPI 官方文档中所述，将环境变量 `I_MPI_HYDRA_BOOTSTRAP=slurm` 导出到启动 StarCCM+ 的作业脚本中。

用户名解析的已知问题

本节内容与在作业中检索用户名有关。

由于已知的 Slurm 错误，如果您运行作业时不使用 `srun`，则在作业进程中检索到的用户名可能是 nobody。此错误仅影响 AWS ParallelCluster 版本 3.1.1 和 3.1.2。

例如，如果您以目录用户身份运行命令 `sbatch --wrap 'srun id'`，则会返回正确的用户名。但如果您以目录用户身份运行 `sbatch --wrap 'id'`，则可能会返回 nobody 作为用户名。

您可以使用以下解决方法。

1. 在可能的情况下使用 `'srun'` 而不是 `'sbatch'` 来启动作业。
2. 通过按如下方式设置集群AdditionalSssdConfigs中的配置来启用 SSSD 枚举。

```
AdditionalSssdConfigs:
  enumerate: true
```

如何解决主目录创建问题

本节内容与主目录创建问题有关。

如果您看到类似以下示例所示的错误，则说明您首次登录到头节点时未为您创建主目录，或您在头节点中首次从 sudoer 切换到 Active Directory 用户时未为您创建主目录。

```
$ ssh AD_USER@$HEAD_NODE_IP
```

```
/opt/parallelcluster/scripts/generate_ssh_key.sh failed: exit code 1

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
Could not chdir to home directory /home/PclusterUser85: No such file or directory
```

主目录创建失败可能是由集群头节点中安装的 oddjob 和 oddjob-mkhomedir 程序包造成的。

如果没有主目录和 SSH 密钥，用户就无法向集群节点提交作业或进行 SSH 登录。

如果您的系统中需要 oddjob 程序包，请验证 oddjobd 服务是否正在运行，然后刷新 PAM 配置文件以确保已创建主目录。为此，请在头节点中运行以下示例所示的命令。

```
sudo systemctl start oddjobd
sudo authconfig --enablemkhomedir --updateall
```

如果您的系统中不需要 oddjob 程序包，请将其卸载，然后刷新 PAM 配置文件以确保已创建主目录。为此，请在头节点中运行以下示例所示的命令。

```
sudo yum remove -y oddjob oddjob-mkhomedir
sudo authconfig --enablemkhomedir --updateall
```

排查自定义 AMI 问题

本节提供可能的自定义 AMI 问题排查提示。

当您使用自定义 AMI 时，您会看到以下警告：

```
"validationMessages": [
  {
    "level": "WARNING",
    "type": "CustomAmiTagValidator",
    "message": "The custom AMI may not have been created by pcluster. You can ignore
 this warning if the AMI is shared or copied from another pcluster AMI. If the
 AMI is indeed not created by pcluster, cluster creation will fail. If the cluster
 creation fails, please go to https://docs.aws.amazon.com/parallelcluster/latest/ug/
troubleshooting.html#troubleshooting-stack-creation-failures for troubleshooting."
  },
```

```
  {
   "level": "WARNING",
   "type": "AmiOsCompatibleValidator",
   "message": "Could not check node AMI ami-0000012345 OS and cluster OS alinux2
 compatibility, please make sure they are compatible before cluster creation and update
 operations."
  }
]
```

如果您确定使用的是正确的 AMI，则可以忽略这些警告。

如果您不想将来看到这些警告，请使用以下标签标记自定义 AMI，其中*my-os*是alinux2、、alinux2023、ubuntu2404、ubuntu2204ubuntu2004rhel8rhel9、或中的一个，*"3.13.2"*是正在使用的pcluster版本：

```
$ aws ec2 create-tags \
  --resources ami-yourcustomAmi \
  --tags Key="parallelcluster:version",Value="3.13.2"
 Key="parallelcluster:os",Value="my-os"
```

排查 cfn-hup 未运行时的集群更新超时问题

cfn-hup 帮助程序作为一个进程守护程序，旨在检测资源元数据中出现的变更，并在检测到变更时运行用户指定的操作。这就是您通过 UpdateStack API 操作对正在运行的 Amazon EC2 实例进行配置更新的方式。

目前，cfn-hup 进程守护程序由 supervisord 启动。但在启动后，cfn-hup 进程将脱离 supervisord 控制。如果 cfn-hup 进程守护程序因外部因素终止，它不会自动重启。如果cfn-hup未运行，则在集群更新期间， CloudFormation 堆栈会按预期启动更新过程，但更新过程未在头节点上激活，堆栈最终会进入超时状态。从集群日志 /var/log/chef-client 中，您可以看到从未调用过更新食谱。

失败时检查并重启 **cfn-hup**

1. 在头节点上，检查 cfn-hup 是否正在运行：

   ```
   $ ps aux | grep cfn-hup
   ```

2. 在头节点上检查 cfn-hup 日志 /var/log/cfn-hup.log 和 /var/log/supervisord.log。
3. 如果 cfn-hup 未运行，请尝试通过运行以下命令进行重启：

```
$ sudo /opt/parallelcluster/pyenv/versions/cookbook_virtualenv/bin/supervisorctl
 start cfn-hup
```

网络问题排查

本节提供了遇到网络问题时的问题排查提示，特别是在处理单个公有子网中的集群问题时。

集群位于单个公有子网的问题

从其中一个计算节点中检查 `cloud-init-output.log`。如果发现类似以下指示节点卡在 Slurm 初始化状态的问题，则很可能是由于缺少 DynamoDB VPC 端点所致。添加 DynamoDB 端点。有关更多信息，请参阅 AWS ParallelCluster 在无法访问互联网的单个子网中。

```
ruby_block[retrieve compute node info] action run[2022-03-11T17:47:11+00:00] INFO:
 Processing ruby_block[retrieve compute node info] action run (aws-parallelcluster-
slurm::init line 31)
```

执行 **onNodeUpdated** 自定义操作时集群更新失败

当 `HeadNode`/`CustomActions`/`OnNodeUpdated` 脚本失败时，更新将失败，并且在回滚时不会运行该脚本。您负责在回滚完成后手动执行所需的清理。例如，如果 `OnNodeUpdated` 脚本更改了配置文件中某个字段的状态（例如，从 `true` 更改为 `false`），然后失败，则您需要将该字段值手动还原到更新前的状态（例如，从 `false` 还原为 `true`）。有关更多信息，请参阅 自定义引导操作。

看到自定义 Slurm 配置错误

从 AWS ParallelCluster 版本 3.6.0 开始，您不能再通过将单个`prolog`或`epilog`脚本包含在自定义Slurm配置中来定位它们。在 AWS ParallelCluster 版本 3.6.0 及更高版本中，您必须在相应的`prolog`和文件夹中找到自定义`epilog`脚本`Prolog`和`Epilog`脚本。这些文件夹默认配置为指向以下位置：

- `Prolog` 指向 `/opt/slurm/etc/scripts/prolog.d/`。
- `Epilog` 指向 `/opt/slurm/etc/scripts/epilog.d/`。

我们建议您将 `90_plcuster_health_check_manager` prolog 脚本和 `90_pcluster_noop` epilog 脚本保留在原处。

Slurm 按字母倒序运行这些脚本。Prolog 和 Epilog 文件夹都必须至少包含一个文件。有关更多信息，请参阅Slurmprolog 和 epilog和Slurm 配置自定义。

集群警报

集群运行状况监控对于确保最佳性能至关重要。 AWS ParallelCluster 使您能够监控集群头节点 CloudWatch 的多个警报。

本节详细介绍了每种类型的头节点集群警报，包括其命名约定、触发警报的特定条件和建议的问题排查步骤。

例如，集群警报的命名约定是 CLUSTER_NAME-COMPONENT-METRIC，例如 mycluster-HeadNode-Cpu。

- CLUSTER_NAME-HeadNode：表示头节点的整体状态。如果以下警报中至少有一个是红色的，它就是红色的。
- CLUSTER_NAME-HeadNode-Health: 如果至少有一次 Amazon Healt EC2 h Check 失败，则为红色。如果出现警报，建议查看排查状态检查失败的实例的问题。
- CLUSTER_NAME-HeadNode-Cpu：如果 CPU 利用率超过 90%，则显示红色。如果出现警报，请使用 ps -aux --sort=-%cpu | head -n 10 查看占用 CPU 资源最多的进程。
- CLUSTER_NAME-HeadNode-Mem：如果内存利用率超过 90%，则显示红色。如果出现警报，请使用 ps -aux --sort=-%mem | head -n 10 查看占用内存资源最多的进程。
- CLUSTER_NAME-HeadNode-Disk：如果路径 / 上占用的磁盘空间大于 90%，则显示红色。如果出现警报，请使用 du -h --max-depth=2 / 2> /dev/null | sort -hr 检查占用大部分空间的文件夹。

AWS ParallelCluster 支持政策

AWS ParallelCluster 支持同时发布多个版本。每个 AWS ParallelCluster 版本都有预定的 Support 生命周期终止 (EOSL) 日期。在 EOSL 日期之后，不再为该版本提供进一步的支持或维护。

AWS ParallelCluster 使用`major.minor.`*`patch`*版本方案。新功能、性能改进、安全更新和错误修复包含在最新主要版本的新次要版本中。次要版本在主要版本中向后兼容。对于关键问题， AWS 通过发布补丁来提供修复，但仅适用于尚未到达 EOSL 的最新次要版本。如果要使用新版本的更新，则需要升级到新的次要版本或补丁版本。

AWS ParallelCluster 版本	支持生命周期结束 (EOSL) 日期
3.0。*x*	2023 年 3 月 31 日
3.1。*x*	8/31/2023
3.2。*x*	2024 年 1 月 31 日
3.3。*x*	5/31/2024
3.4。*x*	2024 年 6 月 28 日
3.5。*x*	8/31/2024
3.6。*x*	11/30/2024
3.7。*x*	2025 年 2 月 28 日
3.8。*x*	6/30/2025
3.9。*x*	09/05/2025
3.10。*x*	12/27/2025
3.11。*x*	2026 年 3 月 25 日
3.12。*x*	06/30/2026
3.13。*x*	09/30/2026

安全性 AWS ParallelCluster

云安全 AWS 是重中之重。作为 AWS 客户，您可以受益于专为满足最安全敏感的组织的要求而构建的数据中心和网络架构。

安全是双方共同承担 AWS 的责任。责任共担模式 将其描述为云的安全性和云中的安全性：

- 云安全 — AWS 负责保护在 AWS 云中运行 AWS 服务的基础架构。 AWS 还为您提供可以安全使用的服务。作为AWS 合规计划合规计划合规计划合的一部分，第三方审计师定期测试和验证我们安全的有效性。要了解适用的合规计划 AWS ParallelCluster，请参阅按合规计划划分的范围内的 AWSAWS 服务按合规计划。
- 云端安全 — 您的责任由您使用的特定 AWS 服务或服务决定。您还需要对多种其它相关因素负责，包括您的数据的敏感性、您公司的要求以及适用的法律法规。

本文档介绍在使用分担责任模型时应如何应用 AWS ParallelCluster。以下主题向您介绍如何进行配置 AWS ParallelCluster 以满足您的安全和合规性目标。您还将学习如何 AWS ParallelCluster 以一种有助于监控和保护 AWS 资源的方式使用。

主题

- 所用服务的安全信息 AWS ParallelCluster
- 中的数据保护 AWS ParallelCluster
- Identity and Access Management AWS ParallelCluster
- 合规性验证 AWS ParallelCluster
- 强制实施最低版本 TLS 1.2

所用服务的安全信息 AWS ParallelCluster

- 亚马逊的安全 EC2
- Amazon API Gateway 中的安全性
- 安全性 AWS Batch
- AWS CloudFormation中的安全性
- 亚马逊的安全 CloudWatch
- AWS CodeBuild中的安全性

- Amazon DynamoDB 中的安全性
- Amazon ECR 中的安全性
- Amazon ECS 中的安全性
- Amazon EFS 中的安全性
- Lustr FSx e 的安全保障
- AWS Identity and Access Management (IAM) 中的安全性
- EC2 Image Builder 中的安全
- 安全性 AWS Lambda
- Amazon Route 53 中的安全性
- Amazon SNS 中的安全性
- 亚马逊 SQS 中的安全性（适用于 2. AWS ParallelCluster x 版）
- Amazon S3 中的安全性
- Amazon VPC 中的安全性

中的数据保护 AWS ParallelCluster

分 AWS 担责任模型适用于中的数据保护 AWS ParallelCluster。如本模型所述 AWS ，负责保护运行所有内容的全球基础架构 AWS Cloud。您负责维护对托管在此基础结构上的内容的控制。您还负责您所使用的 AWS 服务 的安全配置和管理任务。有关数据隐私的更多信息，请参阅数据隐私常见问题。有关欧洲数据保护的信息，请参阅 AWS Security Blog 上的 AWS Shared Responsibility Model and GDPR 博客文章。

出于数据保护目的，我们建议您保护 AWS 账户 凭证并使用 AWS IAM Identity Center 或 AWS Identity and Access Management (IAM) 设置个人用户。这样，每个用户只获得履行其工作职责所需的权限。还建议您通过以下方式保护数据：

- 对每个账户使用多重身份验证（MFA）。
- 使用 SSL/TLS 与资源通信。 AWS 我们要求使用 TLS 1.2，建议使用 TLS 1.3。
- 使用设置 API 和用户活动日志 AWS CloudTrail。有关使用 CloudTrail 跟踪捕获 AWS 活动的信息，请参阅AWS CloudTrail 用户指南中的使用跟 CloudTrail 踪。
- 使用 AWS 加密解决方案以及其中的所有默认安全控件 AWS 服务。
- 使用高级托管安全服务（例如 Amazon Macie），它有助于发现和保护存储在 Amazon S3 中的敏感数据。

- 如果您在 AWS 通过命令行界面或 API 进行访问时需要经过 FIPS 140-3 验证的加密模块，请使用 FIPS 端点。有关可用的 FIPS 端点的更多信息，请参阅《美国联邦信息处理标准（FIPS）第 140-3 版》。

强烈建议您切勿将机密信息或敏感信息（如您客户的电子邮件地址）放入标签或自由格式文本字段（如名称字段）。这包括您使用控制台、API AWS ParallelCluster 或以其他 AWS 服务 方式使用控制台 AWS CLI、API 或时 AWS SDKs。在用于名称的标签或自由格式文本字段中输入的任何数据都可能会用于计费或诊断日志。如果您向外部服务器提供网址，强烈建议您不要在网址中包含凭证信息来验证对该服务器的请求。

数据加密

所有安全服务均具有一项重要功能，即信息在未处于活动使用状态时都会加密。

静态加密

AWS ParallelCluster 除了代表用户与 AWS 服务进行交互所需的凭据外，它本身不存储任何客户数据。

对于集群中节点上的数据，可以对数据进行静态加密。

对于 Amazon EBS 卷，使用 `EbsSettings` 部分中的 `EbsSettings`/`Encrypted` 和 `EbsSettings`/`KmsKeyId` 设置来配置加密。有关更多信息，请参阅《亚马逊 EC2 用户指南》中的 Amazon EBS 加密。

对于 Amazon EFS 卷，使用 `EfsSettings` 部分中的 `EfsSettings`/`Encrypted` 和 `EfsSettings`/`KmsKeyId` 设置来配置加密。有关更多信息，请参阅 Amazon Elastic File System User Guide 中的 How encryption at rest works。

对 FSx 于 Lustre 文件系统，创建 Amazon FSx 文件系统时会自动启用静态数据加密。有关更多信息，请参阅 Amazon for Lustre 用户指南 FSx 中的加密静态数据。

对于带有 NVMe 卷的实例类型， NVMe 实例存储卷上的数据使用在实例的硬件模块上实现的 XTS-AES-256 密码进行加密。加密密钥使用硬件模块生成，并且对于每个 NVMe 实例存储设备都是唯一的。当实例停止或终止并且无法恢复时，将销毁所有加密密钥。无法禁用此加密，并且无法提供自己的加密密钥。有关更多信息，请参阅 Amazon EC2 用户指南中的静态加密。

如果您使用 AWS ParallelCluster 调用将客户数据传输到本地计算机进行存储的 AWS 服务，请参阅该服务的《用户指南》中的 “安全与合规性” 一章，了解如何存储、保护和加密这些数据的信息。

传输中加密

默认情况下，通过通过 HTTPS/TLS 连接发送所有内容来加密从正在运行的客户端计算机 AWS ParallelCluster 和 AWS 服务端点传输的所有数据。可以自动加密集群中节点之间的流量，具体取决于所选的实例类型。有关更多信息，请参阅 Amazon EC2 用户指南中的传输中加密。

另请参阅

- 亚马逊的数据保护 EC2
- EC2 Image Builder 中的数据保护
- 中的数据保护 AWS CloudFormation
- Amazon EFS 中的数据保护
- Amazon S3 中的数据保护
- FSx 为 Lustre 提供数据保护

Identity and Access Management AWS ParallelCluster

AWS ParallelCluster 使用角色来访问您的 AWS 资源及其服务。 AWS ParallelCluster 用于授予权限的实例和用户策略记录在中AWS Identity and Access Management 中的权限 AWS ParallelCluster。

唯一的主要区别在于使用标准用户和长期凭证时如何进行身份验证。尽管用户需要密码才能访问 AWS 服务的控制台，但该用户需要访问密钥对才能使用执行相同的操作 AWS ParallelCluster。所有其他短期凭证的使用方式与在控制台中使用时相同。

使用的凭证存储 AWS ParallelCluster 在纯文本文件中，未加密。

- 该`$HOME/.aws/credentials`文件存储访问您的 AWS 资源所需的长期证书。这包括访问密钥 ID 和秘密访问密钥。
- 短期凭证（例如您承担的角色或用于 AWS IAM Identity Center 服务的角色的凭证）也存储在 `$HOME/.aws/cli/cache` 和 `$HOME/.aws/sso/cache` 文件夹中。

风险防范

- 我们强烈建议您在 `$HOME/.aws` 文件夹及其子文件夹和文件上配置文件系统权限，仅限授权用户访问。
- 尽可能使用具有临时凭证的角色，以减少凭证泄露时造成损坏的机会。仅使用长期凭证来请求和刷新短期角色凭证。

合规性验证 AWS ParallelCluster

作为多个合规计划的一部分，第三方审计师评估 AWS 服务的安全 AWS 性和合规性。使用 AWS ParallelCluster 访问服务不会改变该服务的合规性。

有关特定合规计划范围内的 AWS 服务列表，请参阅AWS 按合规计划划分的范围内AWS 服务（按合分）。有关一般信息，请参阅AWS 合规计划AWS。

您可以使用下载第三方审计报告 AWS Artifact。有关更多信息，请参阅在 AWS Artifact中下载报告。

您在使用 AWS ParallelCluster 时的合规责任取决于您的数据的敏感性、贵公司的合规目标以及适用的法律和法规。 AWS 提供了以下资源来帮助实现合规性：

- 安全与合规性快速入门指南 — 这些部署指南讨论了架构注意事项，并提供了在上面部署以安全性和合规性为重点的基准环境的步骤。 AWS
- 在 A@@ mazon Web Services 上构建 HIPAA 安全与合规架构 AWS 白皮书 — 本白皮书描述了各公司如何使用来 AWS 创建符合 HIPAA 标准的应用程序。
- AWS 合AWS 规资源 — 此工作簿和指南集合可能适用于您的行业和所在地区。
- 使用AWS Config 开发人员指南中的规则评估资源 — 该 AWS Config 服务评估您的资源配置在多大程度上符合内部实践、行业准则和法规。
- AWS Security Hub— 此 AWS 服务可全面了解您的安全状态 AWS ，帮助您检查是否符合安全行业标准和最佳实践。

强制实施最低版本 TLS 1.2

为了提高与 AWS 服务通信时的安全性，应将您的配置 AWS ParallelCluster 为使用 TLS 1.2 或更高版本。当你使用时 AWS ParallelCluster，Python 用于设置 TLS 版本。

为了确保不 AWS ParallelCluster 使用早于 TLS 1.2 的 TLS 版本，您可能需要重新编译 OpenSSL 以强制执行此最低限度，然后重新编译 Python 以使用新构建的 OpenSSL。

确定当前支持的协议

首先，使用 OpenSSL 创建一个自签名证书，以用于测试服务器和 Python 开发工具包。

```
$ openssl req -subj '/CN=localhost' -x509 -newkey rsa:4096 -nodes -keyout key.pem -out
 cert.pem -days 365
```

然后，使用 OpenSSL 启动测试服务器。

```
$ openssl s_server -key key.pem -cert cert.pem -www
```

在新的终端窗口中，创建虚拟环境并安装 Python 开发工具包。

```
$ python3 -m venv test-env
source test-env/bin/activate
pip install botocore
```

创建一个名为 `check.py` 的新 Python 脚本，该脚本使用此开发工具包的底层 HTTP 库。

```
$ import urllib3
URL = 'https://localhost:4433/'

http = urllib3.PoolManager(
ca_certs='cert.pem',
cert_reqs='CERT_REQUIRED',
)
r = http.request('GET', URL)
print(r.data.decode('utf-8'))
```

运行您的新脚本。

```
$ python check.py
```

这将显示有关所建立的连接的详细信息。在输出中搜索“协议：”。如果输出为 “TLSv1.2” 或更高版本，则软件开发工具包默认为 TLS v1.2 或更高版本。如果它是较早的版本，则需要重新编译 OpenSSL 并重新编译 Python。

但是，即使 Python 的安装默认为 TLS v1.2 或更高版本，如果服务器不支持 TLS v1.2 或更高版本，则 Python 仍可能重新协商到 TLS v1.2 之前的版本。要检查 Python 是否不会自动重新协商到较早版本，请使用以下命令重新启动测试服务器。

```
$ openssl s_server -key key.pem -cert cert.pem -no_tls1_3 -no_tls1_2 -www
```

如果您使用的是较早版本的 OpenSSL，则可能没有可用的 `-no_tls_3` 标志。如果是这种情况，请删除该标志，因为您使用的 OpenSSL 版本不支持 TLS v1.3。然后，运行 Python 脚本。

```
$ python check.py
```

如果您正确安装了 Python 但未针对 TLS 1.2 之前的版本进行重新协商，则应收到 SSL 错误。

```
$ urllib3.exceptions.MaxRetryError: HTTPSConnectionPool(host='localhost',
 port=4433): Max retries exceeded with url: / (Caused by SSLError(SSLError(1, '[SSL:
 UNSUPPORTED_PROTOCOL] unsupported protocol (_ssl.c:1108)')))
```

如果您能够建立连接，则需要重新编译 OpenSSL 和 Python 以禁用对早于 TLS v1.2 的协议进行协商。

编译 OpenSSL 和 Python

为了确保在 TLS 1.2 之前的版本中 AWS ParallelCluster 不会进行协商，你需要重新编译 OpenSSL 和 Python。要执行此操作，请复制以下内容以创建脚本并运行脚本。

```
#!/usr/bin/env bash
set -e

OPENSSL_VERSION="1.1.1d"
OPENSSL_PREFIX="/opt/openssl-with-min-tls1_2"
PYTHON_VERSION="3.8.1"
PYTHON_PREFIX="/opt/python-with-min-tls1_2"


curl -O "https://www.openssl.org/source/openssl-$OPENSSL_VERSION.tar.gz"
tar -xzf "openssl-$OPENSSL_VERSION.tar.gz"
cd openssl-$OPENSSL_VERSION
./config --prefix=$OPENSSL_PREFIX no-ssl3 no-tls1 no-tls1_1 no-shared
make > /dev/null
sudo make install_sw > /dev/null


cd /tmp
curl -O "https://www.python.org/ftp/python/$PYTHON_VERSION/Python-$PYTHON_VERSION.tgz"
tar -xzf "Python-$PYTHON_VERSION.tgz"
cd Python-$PYTHON_VERSION
./configure --prefix=$PYTHON_PREFIX --with-openssl=$OPENSSL_PREFIX --disable-shared > /
dev/null
make > /dev/null
sudo make install > /dev/null
```

这会编译具有静态链接的 OpenSSL 的 Python 版本，该版本不会自动协商早于 TLS 1.2 的任何版本。这也会在 `/opt/openssl-with-min-tls1_2` 目录中安装 OpenSSL，并在 `/opt/python-with-min-tls1_2` 目录中安装 Python。运行此脚本后，请确认安装新版本的 Python。

```
$ /opt/python-with-min-tls1_2/bin/python3 --version
```

这应该打印出以下内容。

```
Python 3.8.1
```

要确认此新版本的 Python 不协商早于 TLS 1.2 的版本，请使用新安装的 Python 版本（即 `/opt/python-with-min-tls1_2/bin/python3`）重新运行 确定当前支持的协议 中的步骤。

AWS 区域 支持 AWS ParallelCluster

AWS ParallelCluster 版本 3 有以下版本 AWS 区域：

区域名称	区域	第一个支持该地区的 AWS ParallelCluster 版本
美国东部（俄亥俄州）	us-east-2	3.0.0
美国东部（弗吉尼亚州北部）	us-east-1	3.0.0
美国西部（加利福尼亚北部）	us-west-1	3.0.0
美国西部（俄勒冈州）	us-west-2	3.0.0
非洲（开普敦）	af-south-1	3.0.0
亚太地区（香港）	ap-east-1	3.0.0
亚太地区（孟买）	ap-south-1	3.0.0
亚太地区（首尔）	ap-northeast-2	3.0.0
亚太地区（新加坡）	ap-southeast-1	3.0.0
亚太地区（悉尼）	ap-southeast-2	3.0.0
亚太地区（雅加达）	ap-southeast-3	3.10.0
亚太地区（马来西亚）	ap-southeast-5	3.13.0
亚太地区（泰国）	ap-southeast-7	3.13.0
亚太地区（东京）	ap-northeast-1	3.0.0
加拿大（中部）	ca-central-1	3.0.0
中国（北京）	cn-north-1	3.0.0
中国（宁夏）	cn-northwest-1	3.0.0

区域名称	区域	第一个支持该地区的 AWS ParallelCluster 版本
欧洲（法兰克福）	eu-central-1	3.0.0
欧洲地区（爱尔兰）	eu-west-1	3.0.0
欧洲（伦敦）	eu-west-2	3.0.0
欧洲（米兰）	eu-south-1	3.0.0
欧洲（巴黎）	eu-west-3	3.0.0
欧洲地区（斯德哥尔摩）	eu-north-1	3.0.0
中东（巴林）	me-south-1	3.0.0
南美洲（圣保罗）	sa-east-1	3.0.0
AWS GovCloud （美国东部）	us-gov-east-1	3.0.0
AWS GovCloud （美国西部）	us-gov-west-1	3.0.0
以色列（特拉维夫）	il-central-1	3.8.0

发布说明和文档历史记录

下表描述了《AWS ParallelCluster 用户指南》的主要更新和新功能。我们还经常更新文档来处理发送给我们的反馈意见。

AWS ParallelCluster

更改	描述	日期
AWS ParallelCluster 3.13.2 版本已发布	要升级，请输入以下内容：`sudo pip install --upgrade aws-parallelcluster` 。 错误修复： • 修复了当计算资源引用已过期`update-cluster` 且无法再通过访问的容量预留时，可能会导致和`update-compute-fleet` 失败的错误 EC2 APIs。 • 修复了 Rocky 9 上父映像未发布最新内核版本时出现的`build-image` 故障。请参阅 https://github.com/aws/aws-parallelcluster/issues/6874。	2025 年 6 月 24 日
AWS ParallelCluster 3.13.1 版本已发布	要升级，请输入以下内容：`sudo pip install --upgrade aws-parallelcluster` 。 更改： • 将 Slurm 升级到版本 24.05.8。	2025年6月4日

更改	描述	日期

	• 将 EFA 安装程序升级到 1.41.0（从 1.38.1 开始）。
	  • efa-driver：efa-2.15.0-1
	  • efa-config：efa-con fig-1.18-1
	  • EFA 简介：efa-profile-1.7 -1
	  • libfabric-aws：libfabric-aws-2.1.0-1
	  • rdma-core：rdma-cor e-57.0-1
	  • 打开 MPI：openm pi40-aws-4.1.7-2 和 openmpi50-aws-5.0.6
	• 对于非亚马逊 Linux AMI，amazon-efs-utils请升级到 2.3.1 版（从 v2.1.0 起）。
	• 在 us-isob-east -1 和 us-iso-east -1 中支持 DCV。
	• 在 -1 和 us-isob-east -1 中支持 Lustre 和 Ontap 的 FSX。 us-iso-east
	• 通过在 ParallelCluster 映像构建开始时固定，在完成时取消固定，确保整个映像构建过程中的内核一致性。
	
	错误修复：
	• 修复了 ARM 性能库安装过程中导致构建映像在隔离环境中失败的错误。	

更改	描述	日期
	• 修复了导致脚本 “update_directory_service_password.sh” 无法更新 AD 密码的错误。	

更改	描述	日期
AWS ParallelCluster 3.13.0 版本已发布	要升级，请输入以下内容：`sudo pip install --upgrade aws-parallelcluster` 。 弃用： • 这是最后一个支持 Ubuntu 20.04 的 ParallelCluster 版本，因为 Ubuntu 20.04 将于 2025 年 5 月推出。 End-Of-Standard-Support 增强功能： • 添加对 Ubuntu 24.04 的支持。 • 添加对 ap-southeast-7 区域的支持。 • 从官方禁用未使用的服务 cups 和 wpa_supplicant 以提高安全性。 ParallelCluster AMIs 更改： • 将 Slurm 升级到版本 24.05.7。 • 将 NVIDIA 驱动程序升级到 570.86.15 版本（从 550.127.08 起），但除外。 OSs AL2	2025年4月1日

更改	描述	日期

	• 将 CUDA Toolkit 升级到 12.8.0 版（从 12.4.1 起），但除外。 OSs AL2
	• 将 Python 升级到 3.12.8，所有版本 OSs 除外 AL2（从 3.9.20 开始）。
	• 在 Ubuntu 22.04 上，安装与编译内核时使用的相同编译器版本的 Nvidia 驱动程序。
	• 升级`aws-cfn-bootstrap` 到 2.0-33 版本。
	• 将 EFA 安装程序升级到 1.38.0（从 1.36.0 开始）。
	  • efa-driver：efa-2.13.0-1
	  • efa-config：efa-config-1.17-1
	  • EFA 简介：efa-profile-1.7-1
	  • `Libfabric-aws`：`libfabric-aws-1.22.0-1`
	  • rdma-core：rdma-core-54.0-1
	  • Open MPI：`openmpi40-aws-4.1.7-1` 和 `openmpi50-aws-5.0.5`
	• 升级 amazon-efs-utils到 2.1.0 版。
	• 移除第三方食谱：apt-7.5.22 和 pyenv-4.2.3。	

更改	描述	日期

	• 升级第三方说明书依赖项：
	  • 第 4.5.21 行（来自第 4.5.13 行）
	  • nfs-5.1.5（来自 nfs-5.1.2）
	  • openssh-2.11.14（来自 openssh-2.11.12）
	  • yum-7.4.20（来自 yum-7.4.13）
	  • yum-epel-5.0.8（来自 yum-epel-5.0.2）
	• 将 Pmix 升级到 5.0.6（从 5.0.3 开始）。
	• 将 ARM PL 升级到版本 24.10（从 23.10 开始）。
	• 在 Lambda 层和安装程序中将 Python 升级到版本 3.12.8（从 3.9.17 开始）。
	• 在 Lambda 层和安装程序中将 NodeJS 升级到版本 20.18.3（从 18.20.3 开始）。
	• 删除以 DSA 形式为登录节点生成 DSA 密钥，OpenSSH 9.7+ 版本不支持这种密钥。
	• 计算节点启动后，在 Slurm 中设置实例 ID 和实例类型信息。
	• 安装不带 “no-cc-version-check” 选项的 NVIDIA 驱动程序，该选项现已在 NVIDIA 安装程序中弃用。	

更改	描述	日期
	• 添加验证器以强制使用最多 10 个登录节点池。 • 将默认根卷大小更新为 45 GB。 错误修复： • 取消使用 cfn-init 进行计算节点引导，以缩短节点扩展时间。 • 修复了使用代理时导致计算节点引导失败的问题。 • 在 Ubuntu 22.04 上，使用与编译内核相同的编译器版本安装 Nvidia 驱动程序，以防止安装失败。-修复更新期间仅在头节点上执行覆盖 aws-parallelcluster-node包的问题。 • 修复了在多用户环境（与 Active Directory 集成）Pyxis/Enroot 中执行的容器化作业会失败的问题。 • 修复使用目录服务时，authselect 的使用会导致在 Rocky 9.5+ 上启动节点失败的问题。	

更改	描述	日期
AWS ParallelCluster 3.12.0 版本已发布	要升级，请输入以下内容：`sudo pip install --upgrade aws-parallelcluster` 。 增强功能： • 添加新的构建映像配置部分`Build/Installation` 以打开/关闭 NVIDIA 软件和 Lustre 客户端安装。默认情况下，NVIDIA 软件虽然包含在官方版本中 ParallelCluster AMIs，但并未由安装`build-image` 。默认情况下，已安装 Lustre 客户端。 • 现在，CLI 命令`export-cluster-logs` 和`export-image-logs` 可以默认将日志导出到默认 ParallelCluster 存储桶或 customs3Bucket（如果在配置中指定）。 • 在 ARM 实例上将 Amazon DCV 支持扩展到 Ubuntu2204。 更改： • 将 NVIDIA 驱动程序升级到版本 550.127.08（从 550.90.07 开始）。这解决了 NVIDIA 的一个已知问题。有关更多信息，请参阅	2024 年 12 月 19 日

更改	描述	日期

	NVIDIA 数据中心文档中的已知问题。
	• 将 Amazon DCV 升级到版本`2024.0-18131` 。
	  • server：`2024.0-18131-1`
	  • xdcv：`2024.0.631-1`
	  • gl：`2024.0.1078-1`
	  • web_viewer：`2024.0-18131-1`
	• 将 EFA 安装程序升级到 `1.36.0`。
	  • Efa-driver：`efa-2.13.0-1`
	  • Efa-config：`efa-config-1.17-1`
	  • Efa-profile：`efa-profile-1.7-1`
	  • Libfabric-aws：`libfabric-aws-1.22.0-1`
	  • Rdma-core：`rdma-core-54.0-1`
	  • Open MPI：`openmpi40-aws-4.1.7-1` 和 `openmpi50-aws-5.0.5`
	• 失败时自动重启 slurmctld。
	• 升级 mysql-community-client到 8.0.39 版。
	• 移除对 Python 3.7 和 3.8 的支持，它们已经过时了。	

更改	描述	日期
	错误修复： • 修复了在集群更新期间未检测到自定义操作脚本序列更改的问题。 • 为 AWS ParallelCluster API 添加缺少的权限，以便为 Elastic Load Balancing 和 Auto Scaling 创建服务关联角色，这些角色是部署登录节点所必需的。 • 修复了我们在管理卷时获取区域的方式中的一个问题，使其能够正确处理本地区域。 • 修复了在更新`AccessPointIds` 期间添加 EFS 文件系统会失败的问题。 • 修复了在使用 PCAPI 时，更新非类型的参数`String`（例如）时群集更新可能会失败的问题。`MaxCount` • 挂载外部 OpenZFS 时，不再需要为端口 111、2049、20001、20002、20003 设置出站规则。	

更改	描述	日期
AWS ParallelCluster 3.11.1 版本已发布	设施特点： • Pyxis 现在默认为禁用状态，因此必须按照产品文档中的说明手动启用。 • 在 Lambd ParallelCluster a 层中将 Python 运行时升级到版本 3.12。 • 不再将 setuptools 的版本固定为 70.0.0 之前的版本。 • 将 libjwt 升级到 1.17.0 版本。 • 完整更改日志 错误修复 • 修复了我们在中配置 Pyxis Slurm 插件的方式中可能 ParallelCluster 导致作业提交失败的问题。 • 通过在公共策略模板中添加登录节点所需的缺少权限，修复了导致登录节点配置中部署失败的问题。 https://github.com/aws/aws-parallelcluster/issues/6483	2024 年 10 月 21 日

更改	描述	日期
AWS ParallelCluster 3.11.0 版本已发布	增强功能 • 添加对登录节点上的自定义操作的支持。 • 允许 DCV 连接到登录节点。 • 添加对 ap-southeast-3 区域的支持。 • 向登录节点网络负载均衡器添加安全组。 • 为登录节点添加`AllowedIps` 配置。 • 添加新配置`SharedStorage/EfsSettings/AccessPointId` 以为装载指定可选的 EFS 接入点 • 允许最多 10 个登录节点池。 • 在官方 pcluster 中安装 enroot 和 pyxis AMIs 更改 • [B REAKING] API `DescribeCluster` 和 CLI 命令返回的`loginNodes` 字段`describe-cluster` 已从字典更改为数组，以支持多个登录节点池。此更改破坏了向后兼容性，使这些操作与使用旧版本部署的集群不兼容。 • 将 Slurm 升级到 23.11.10（从 23.11.7 开始）。	2024 年 9 月 26 日

更改	描述	日期

	• 将 Pmix 升级到 5.0.3（从 5.0.2 开始）。
	• 将 EFA 安装程序升级到 `1.34.0`。
	  • Efa-driver：`efa-2.10.0-1`
	  • Efa-config：`efa-config-1.17-1`
	  • Efa-profile：`efa-profile-1.7-1`
	  • Libfabric-aws：`libfabric-aws-1.22.0-1`
	  • Rdma-core：`rdma-core-52.0-1`
	  • Open MPI：`openmpi40-aws-4.1.6-3` 和 `openmpi50-aws-5.0.3-11`
	• 将 NVIDIA 驱动程序升级到版本 550.90.07（从 535.183.01 开始）。
	• 将 CUDA 工具包升级到版本 12.4.1（从 12.2.2 开始）。
	• 将 Python 升级到 3.9.20（从 3.9.19 开始）。
	• 将英特尔 MPI 库升级到 2021.13.1.769（从 2021.12.1.8 开始）。
	
	错误修复	

更改	描述	日期
	• 修复验证器，`EfaPlacementGroupValidator` 使其在使用容量块时不建议配置置放群组。 • 确保 FSx 按照安全组规则创建 For Lustre 文件系统，修复偶尔出现的集群创建失败。 • 修复启用置放群组时集群删除失败的问题。 • 修复了在限制 SSH 访问时登录节点被标记为不健康的问题。 • 修复后`retrieve_supported_regions`，它可以获得正确的 S3 网址。 • 修复`describe_images` 为使用分页。 • `No route tables found`修复了在/ LoginNodesNetworking/ SubnetIds 中指定默认 VPC 子网时的错误。	
AWS ParallelCluster 3.10.1 版本已发布	错误修复 • 修复中国区域的镜像构建失败。	2024 年 7 月 8 日

更改	描述	日期

AWS ParallelCluster 3.10.0 版本已发布	增强功能：
	• 添加新的配置部分`Scheduling/SlurmSettings/ExternalSlurmdbd` 以将集群连接到外部 Slurmdbd。
	• 允许在隔离的网络中运行构建映像。
	• 添加对亚马逊 Linux 2023 的支持。
	• 添加对 `price-capacity-optimized` as 的支持`AllocationStrategy` 。
	• 添加验证器以防止使用带有容量块的置放组。
	
	更改：
	• 不再支持 CentOS 7。
	• 将 Cinc Client 从 18.2.7 升级到 18.4.12 版本。
	• 将 munge 升级到版本 0.5.16（从 0.5.15 开始）。
	• 将 Pmix 升级到 5.0.2（从 4.2.9 开始）。
	• 升级第三方说明书依赖项：
	  • apt-7.5.22（来自 apt-7.5.14）
	  • openssh-2.11.12（来自 openssh-2.11.3）
		2024 年 6 月 27 日

更改	描述	日期

	• 移除第三方食谱：selinux-6.1.12。
	• 将 EFA 安装程序升级到 `1.32.0`。
	  • Efa-driver：`efa-2.8.0-1`
	  • Efa-config：`efa-config-1.16-1`
	  • Efa-profile：`efa-profile-1.7-1`
	  • Libfabric-aws：`libfabric-aws-1.21.0-1`
	  • Rdma-core：`rdma-core-50.0-1`
	  • Open MPI：`openmpi40-aws-4.1.6-3` 和 `openmpi50-aws-5.0.2-12`
	• 将 NVIDIA 驱动程序升级到版本 535.183.01（从 535.154.05 开始）。
	• 将 Python 升级到 3.9.19（从 3.9.17 开始）。
	• 将英特尔 MPI 库升级到 2021.12.1.8（从 2021.9.0.43482 起）。
	
	错误修复：
	• 将数据存储库关联配置修复为`AutoExpor`	

更改	描述	日期
	tPolicy 可AutoImportPolicy 选配置。 • 修复了集群删除期间的一个问题，该问题现在可以在实例处于关闭或终止状态时完成计算队列清理。这是为了避免终止周期较长的实例类型的集群删除失败。 • 允许在集群配置Monitoring 部分启用 cloudwatch 控制面板并禁用警报。 • 允许 ParallelCluster 自定义资源使用禁止验证器。PclusterCluster/SuppressValidators • 将其删除，/etc/profile.d/pcluster.sh 这样就不会在每次用户登录时执行该操作，cfn_bootstrap_virtualenv 也不会将其添加到 PATH 环境变量中。 • 通过将字段failureReason 替换为DescribeCluster 响应来修复 ParallelCluster API 规范。failures • 通过添加缺失的 CloudFormation 堆栈状态来修复 ParallelCluster API 规范：IMPORT_*、REVIEW_IN_PROGRESS 和UPDATE_FAILED 。	

更改	描述	日期
	• 修复了阻止集群更新包含传输中加密的 EFS 文件系统的问题。 • 修复了在将 EFS 用于共享内部数据时，slurmctld 和 slurmdbd 服务无法在头节点重启时重新启动的问题。 • 在 Ubuntu 系统上，删除与来自的配置冲突的 cloud-init 日志文件的默认 logrotate 配置。 ParallelCluster • 使用 RHEL 8.10 或更高版本修复图像构建失败。	
AWS ParallelCluster 3.9.3 版本已发布	要进行升级，请键入 `sudo pip install --upgrade aws-parallelcluster` 设施特点： • 在 `us-iso-east-1` 中增加了对 `FSx Lustre` 作为共享存储类型的支持。 错误修复： • 从 Slurm 配置的 `SlurmctldParameters` 中移除 `cloud_dns` ，以避免出现 Slurm 扇出问题。 这不是必需的，因为我们是在实例启动时设置 IP 地址的。	2024 年 6 月 19 日

更改	描述	日期
AWS ParallelCluster 3.9.2 版本已发布	设施特点： • 将 Slurm 升级到 23.11.7（从 23.11.4 起）。 • 有关更多详细信息，请参阅CHANGELOG 3.9.2上的 GitHub。	2024 年 5 月 28 日
AWS ParallelCluster 3.9.1 版本已发布	要进行升级，请输入以下内容：`sudo pip install --upgrade aws-parallelcluster` 错误修复 • 在更新集群操作中卸载文件系统时，移除对共享存储 mountdir 的递归删除。	2024 年 4 月 11 日

更改	描述	日期
AWS ParallelCluster 3.9.0 版本已发布	要进行升级，请输入以下内容：`sudo pip install --upgrade aws-parallelcluster` 增强功能： • 添加配置参数 `DeploymentSettings/DefaultUserHome` ，使用户能够将默认用户的主目录移到 `/local/home` 而不是 `/home`（默认）。 • 允许更新 `MinCount`、`MaxCount`、`Que` 和 `ComputeResource` 配置参数，而无需停止计算实例集。现在可以通过将 `Scheduling/SlurmSettings/QueueUpdateStrategy` 设置为 TERMINATE 来更新它们。 AWS ParallelCluster 只终止通过集群更新调整集群容量时移除的节点。 • 允许在 FileCache 不替换计算和登录队列的情况下更新 Efs FsxLustre FsxOntap、、 FsxOpenZfs 和类型的外部共享存储。 • 添加对的支持 RHEL9。 • 增加了对 Rocky Linux 9 的支持，作为通过 `build-image` 过程创建的	2024 年 3 月 5 日

更改	描述	日期
	`CustomAmi` 。目前还没有官方的 AWS ParallelCluster Rocky9 Linux AMI 可用。 • `CommunicationParameters` 从 “自定义 Slurm 设置” 拒绝列表中移除。 • 在支持 OSes中添加`DeploymentSettings/DisableSudoAccessForDefaultUser` 参数以禁用默认用户的 sudo 访问权限。 • 对 FSx 于 Lustre 文件系统的更改由 ParallelCluster以下人员创建：将 Lustre 服务器版本更改为 `2.15.` • 通过 `['cluster']['nvidia']['kernel_open']` cookbook 节点属性，增加在构建 AMI 时选择开放源代码和封闭源代码 Nvidia 驱动程序的可能性。 • * 添加 clustermgtd 配置选项`ec2_instance_missing_max_count` 以允许可配置的重试次数，以实现最终的 A EC2 mazon 描述实例与运行的实例保持一致。 更改	

更改	描述	日期

	• 将 Slurm 升级到 23.11.4（从 23.02.7 开始）。
	• 将 NVIDIA 驱动程序升级到版本 535.154.05。
	• 在 pcluster CLI 中添加对 Python 3.11、3.12 的支持，以及。 aws-parallelcluster-batch-cli
	• 使用 Amazon EC2 DescribeInstances 响应`NetworkCardIndex` 列表中的网卡索引构建网络接口，而不是遍历`MaximumNetworkCards` 范围。
	• 使用 P3、G3、P2 和 G2 类型实例时，集群创建失败，因为其 GPU 架构与 3.8.0 版中引入的开源 Nvidia 驱动程序（OpenRM）不兼容。
	• 升级第三方 cookbook 依赖项：nfs-5.1.2（从 nfs-5.0.0 升级）
	• 将 EFA 安装程序升级到 `1.30.0.`
	  • Efa-driver：`efa-2.6.0-1`
	  • Efa-config：`efa-config-1.15-1`
	  • Efa-profile：`efa-profile-1.6-1`	

更改	描述	日期
	• Libfabric-aws：`libfabric-aws-1.19.0` • Rdma-core：`rdma-core-46.0-1` • Open MPI：`openmpi40-aws-4.1.6-2` 和 `openmpi50-aws-5.0.0-11` • 将 NICE DCV 升级到版本 `2023.1-16388.` • server：`2023.1.16388-1` • xdcv：`2023.1.565-1` • gl：`2023.1.1047-1` • web_viewer：`2023.1.16388-1` 错误修复 • 修复以 Active Directory 用户身份从登录节点提交作业时会失败的问题。该问题是由于在头节点上与外部 Active Directory 集成的配置不完整造成的。 • 重构在 CloudFormation 模板 parallelclusser-policies.yaml 中定义的 IAM 策略，以防止 ParallelCluster 因策略超过 IAM 限制而导致 API 部署失败。	

更改	描述	日期
	• 修复当头节点写入密钥的时间超过预期时，登录节点启动失败的问题。 有关更改的详细信息，请参阅上的aws-parallelcluster-ui软件包CHANGELOG 文件 GitHub。	

更改	描述	日期
AWS ParallelCluster 3.8.0 版本已发布	AWS ParallelCluster 3.8.0 版本已发布。 增强功能： • 添加对 Amazon 机器学习 EC2 容量块的支持。 • 增加了对 Rocky Linux 8 的支持，作为通过 `build-image` 过程创建的 `CustomAmi` 。目前还没有官方的 AWS ParallelCluster Rocky8 Linux AMI 可用。 • 添加`Scheduling/ScalingStrategy` 参数以控制在为 Slurm 计算节点启动 Amazon EC2 实例时要使用的集群扩展策略。可能的值为 `all-or-nothing` 、`greedy-all-or-nothing` 、`best-effort` ，默认值为 `all-or-nothing` 。 • 添加`HeadNode/SharedStorageType` 参数以使用 EFS 存储，而不是从头节点根卷导出 NFS 来存储集群内共享文件系统资源：Intel ParallelCluster、Slurm 和数据。`/home`此增强功能减少了头节点联网的负载。 • 允许通过配置文件`SharedStorage` 部分/	2023 年 12 月 19 日

更改	描述	日期
	home作为 EFS 或 FSx 外部共享存储进行挂载。 • 添加新参数`SlurmSettings/MungeKeySecretArn` 以允许使用来自 Secrets Manager 的外部用户定义的 MUNGE AWS 密钥。 • 添加`Monitoring/Alarms/Enabled` 参数以切换集群的 Amazon CloudWatch 警报。 • 添加头节点警报以监控 Amazon 运行 EC2 状况检查、CPU 利用率和头节点的整体状态，并将其添加到使用集群创建的 CloudWatch 控制面板中。 • 将 as 用于托管 Lustre 时`PERSISTENT_2` ，添加`DeploymentType` 对数据存储库关联 FSx 的支持。 • 添加`Scheduling/SlurmSettings/Database/DatabaseName` 参数以允许用户为数据库服务器上用于 Slurm 记账的数据库指定自定义名称。 • 在计算资源中配置`CapacityReservationTarget/CapacityReservationId` 时，将	

更改	描述	日期
	`InstanceType` 作为可选配置参数。 • 增加为 AWS ParallelCluster API 创建的 IAM 角色和策略指定前缀的可能性。 • 增加了为由 AWS ParallelCluster API 创建的 IAM 角色和策略指定要应用的权限边界的可能性。 更改 • 将 Slurm 升级到 23.02.7（从 23.02.6 开始）。 • 将 NVIDIA 驱动程序升级到版本 535.129.03。 • 将 CUDA 工具包升级到版本 12.2.2。 • 使用开源 NVIDIA GPU 驱动程序（OpenRM）作为 Linux 的 NVIDIA 内核模块，而不是 NVIDIA 封闭源代码模块。 • 移除 Slurm 恢复程序中对`all_or_nothing_batch` 配置参数的支持，转而使用新的`Scheduling/ScalingStrategy` 集群配置。 • 将集群警报命名约定更改为“[cluster-name]-[component-name]-[metric]”。	

更改	描述	日期

	• 将 ADC 区域中根卷和附加卷的默认 EBS 卷类型从 gp2 更改为 gp3。
	• API 的可选权限边界现已应用于 AWS ParallelCluster API 基础设施创建的每个 IAM 角色。
	  • 将 EFA 安装程序升级到 `1.29.1`。
	  • Efa-driver：`efa-2.6.0-1`
	  • Efa-config：`efa-config-1.15-1`
	  • Efa-profile：`efa-profile-1.5-1`
	  • Libfabric-aws：`libfabric-aws-1.19.0-1`
	  • Rdma-core：`rdma-core-46.0-1`
	  • Open MPI：`openmpi40-aws-4.1.6-1`
	• 除了使用 2.3.1 版本的 Centos 7 之外 OSes，所有支持版本都升级 GDRCopy 到 2.4 版。
	• 将 `aws-cfn-bootstrap` 升级到版本 2.0-28。
	• 在中添加对 Python 3.10 的 aws-parallelcluster-batch-cli 支持。	

更改	描述	日期
	错误修复 • 修复了在修改计算资源中声明的实例类型列表时，集群更新回滚后扩展配置不一致的问题。 • 修复了通过集群配置文件与外部 LDAP 服务器集成的集群中切换无 root 权限用户时的用户 SSH 密钥生成问题。 • 修复了在设置时禁用 Slurm 省电模式的问题。`Scaledown Idletime = -1` • 修复 Slurm Accounting `update_slurm_database_password.sh` 脚本中指向 Slurm 安装目录的硬编码路径。	
AWS ParallelCluster 3.7.2 版本已发布	AWS ParallelCluster 3.7.2 版本已发布。 更改： • 将 Slurm 升级到 23.02.6。	2023 年 10 月 25 日

更改	描述	日期

AWS ParallelCluster 3.7.1 版本已发布	AWS ParallelCluster 3.7.1 版本已发布。
	
	更改：
	
	• 将 Slurm 升级到 23.02.5（从 23.02.4 开始）。
	  • 将 Pmix 升级到 4.2.6（从 3.2.3 升级）。
	  • 将 libjwt 升级到 1.15.3（从 1.12.0 升级）。
	• 将 EFA 安装程序升级到 `1.26.1`，修复 P5 中的 RDMA 写入数据问题。
	  • Efa-driver：`efa-2.5.0-1` 。
	  • Efa-config：`efa-config-1.15-1` 。
	  • Efa-profile：`efa-profile-1.5-1` 。
	  • Libfabric-aws：`libfabric-aws-1.18.2-1` 。
	  • ERdma-核心：`rdma-core-46.0-1` 。
	  • Open MPI：`openmpi40-aws-4.1.5-4` 。
		2023 年 9 月 22 日

更改	描述	日期
AWS ParallelCluster 3.7.0 版本已发布	AWS ParallelCluster 3.7.0 版本已发布。 增强功能： • Support 使用配置 YAML 文件在计算资源中 AWS ParallelCluster 配置静态和动态节点优先级。 • 增加了对 Ubuntu 22 的支持。默认情况下不支持 RSA 密钥。 • 添加了队列配置设置 `JobExclusiveAllocation` ，用于在任何给定时间在分区中以独占模式将节点分配给单个作业。 • 允许在创建集群和更新集群时覆盖 `aws-parallelcluster-node` 程序包。对于头节点，这适用于集群更新。仅用于开发目的。 • 避免在计算节点上启动 NFS 服务器。 • 添加了对登录节点的支持。 • 当为 Slurm 计算资源指定了多种实例类型时，允许进行基于内存的调度。 • 添加了支持，允许将现有 Amazon 文件缓存作为共享存储进行挂载。	2023 年 8 月 30 日

更改	描述	日期

	更改：
	• 默认情况下，将 Slurm 动态节点的优先级（权重）分配为 1000。通过这样做，Slurm 可以将闲置的静态节点优先于空闲的动态节点。
	• 使`aws-parallelcluster-node` 守护程序只能处理 AWS ParallelCluster 托管的 Slurm 分区。
	• 将 `EFS-utils` 监视器轮询间隔增加到 10 秒。当 `EncryptionInTransit` 设置为 `true` 时（这是导致监视器运行的唯一条件），此更改适用。
	• 将 EFA 安装程序升级到 `1.25.1`。
	  • Efa-driver：`efa-2.5.0-1`（从 `efa-2.1.1g`）
	  • Efa-config：`efa-config-1.15-1`（从 `efa-config-1.13-1`）
	  • Efa-profile：`efa-profile-1.5-1`（无变化）
	  • Libfabric-aws：`libfabric-aws-1.18.1-0`（从 `libfabric-aws-1.17.1-1`）	

更改	描述	日期
	• Rdma-core：`rdma-core-46.0-1`（从 `rdma-core-43.0-1`） • Open MPI：`openmpi40-aws-4.1.5-4`（从 `openmpi40-aws-4.1.5-1`） • 将 Slurm 升级到 23.02.4 版。 • 将的默认值 lmds/lmds Support 从 v1.0 更改为 2.0。 • 弃用 Ubuntu 18。 • 将默认根卷大小更新为 40 GB，以补偿对 Centos 7 的限制。 • 限制头节点内文件 /tmp/wait_condition_handle.txt 的权限，只有根用户才能读取。 • 创建一个 Slurm 分区节点列表映射 JSON 文件，供节点程序包进程守护程序用来识别 PC 托管的 Slurm 分区和节点列表。 • 将 NVIDIA 驱动程序升级到版本 535.54.03。 • 将 CUDA 库升级到版本 12.2.0。 • 将 NVIDIA Fabric Manager 升级到 nvidia-fabricmanager-535。	

更改	描述	日期

	• 将 ARM PL 升级到版本 23.04.1，仅适用于 Ubuntu 22.04。
	• 将 NICE DCV 升级到版本 `2023.0-15487` 。
	  • Server：`2023.0.15487-1`
	  • xdcv：`2023.0.551-1`
	  • gl：`2023.0.1039-1`
	  • web_viewer：`2023.0.15487-1`
	
	错误修复：
	• 为 `ScaledownIdletime` 值添加验证功能，以防止设置的值低于 -1。
	• 修复了在启用 DCV 的 GPU 实例上使用 Ubuntu 深度学习 AMI 创建集群失败的问题。
	• 修复了使用创建 ParallelCluster CloudFormation 自定义资源提供商时导致创建悬而未决的 IAM 策略的问题 CustomLambdaRole。
	• 修复了使用等于 `True` 的 `SlurmSettings/Dns/UseEc2Hostnames` 时导致具有多个网络接口的实例上的计算节点 DNS 名称不一致的问题	

更改	描述	日期
	有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	
仅文档发布	AWS ParallelCluster 第 3 版特定用户指南已发布。 仅文档发布： • AWS ParallelCluster 版本 3 有自己的单独用户指南。	2023 年 7 月 17 日

更改	描述	日期
AWS ParallelCluster 版本 3.6.1 已发布	AWS ParallelCluster 3.6.1 版本已发布。 更改： • `clustermgtd` 如果将计算节点添加到多个 Slurm 分区，请避免出现重复的节点。 错误修复： • 删除根卷设备名称（`/dev/sda1` 和`/dev/xvda` ）的硬编码，然后从期间 AMIs 使用的中检索它`create-cluster` 。 • 修复使用`ElasticIp` 设置为的 CloudFormation 自定义资源时集群创建失败的问题`True`。 • 修复使用包含大型配置文件的 AWS CloudFormation 自定义资源时集群创建和更新失败的问题。 • 修复了无法在 Ubuntu 上禁用 `ptrace` 保护并且不允许在 libfabric 中进行跨内存附加 (CMA) 的问题。 • 修复了使用多个实例类型且未返回任何实例时的快速容量不足故障转移逻辑。	2023 年 7 月 5 日

更改	描述	日期
	有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 3.6.0 版本已发布	AWS ParallelCluster 3.6.0 版本已发布。 文档： • 为 AWS ParallelCluster Python 库 API 添加文档。 增强功能： • 添加对的支持 RHEL8。 • 添加用于创建和管理集群的AWS CloudFormation 自定义资源 CloudFormation。 • 在配置 YAML 文件中添加对自定义集群 Slurm AWS ParallelCluster 配置的支持。 • 在支持 LUA 的情况下构建 Slurm。 • 将每个集群的最大队列数限制从 10 增加到 50。每个队列最多可以有 50 个计算资源。每个集群最多可以有 50 个计算资源。 • 添加了支持，允许为 `OnNodeStart` 、`OnNodeConfigured` 和 `OnNodeUpdated` 参数中配置的事件指定一系列多个自定义操作脚本。 • 添加了新的配置部分 `HealthChecks /Gpu`，用	2023 年 5 月 22 日

更改	描述	日期
	于在运行作业之前在计算节点上应用 GPU 运行状况检查。 • 在 `SlurmQueues` 和`SlurmQueues /ComputeResources` 配置中增加了对 `Tags` 的支持。 • 在 `Monitoring` 配置中增加了对 DetailedMonitoring 的支持。 • 在 AWS ParallelCluster CloudWatch 仪表板中添加`mem_used_percent` 头节点内存和根卷磁盘利用率跟踪`disk_used_percent` 指标，并设置警报以监控这些指标。 • 对 AWS ParallelCluster 托管的 日志添加了日志轮换支持。 • 在CloudWatch 控制面板中跟踪常见的计算节点错误和动态节点最长空闲时间。 • 在创建 SSL 套接字时，强制 DCV Authenticator Server 至少使用 `TLS-1.2` 协议。 • 在除 `aarch64 centos7` 和 `alinux2` 之外的所有支持的操作系统上安装 NVIDIA Data Center GPU Manager (DCGM) 程序包。	

更改	描述	日期
	• 默认加载内核模块 nvidia-uvm，为 CUDA 驱动程序提供统一虚拟内存 (UVM) 功能。 • 安装 NVIDIA 持久性进程守护程序作为一项系统服务。 更改： • 将 Slurm 升级到版本`23.02.2`（从版本`22.05.8`开始）。 • 将 munge 升级到版本 `0.5.15`（从版本 `0.5.14`）。 • 将 Slurm 设置为 30 `TreeWidth` 。 • 将 Slurm `prolog` 和`epilog`配置分别设置为目标目录`/opt/slurm/etc/scripts/prolog.d/` 和。`/opt/slurm/etc/scripts/epilog.d/` • 将 Slurm 设置为 3 分钟`BatchStartTimeout` ，以便在注册计算节点期间运行`Prolog`脚本。 • 将 CloudWatch 日志`RetentionInDays` 的默认值从 14 天增加到 180 天。	

更改	描述	日期

	• 将 EFA 安装程序升级到 `1.22.1`。
	  • Dkms：`2.8.3-2`
	  • Efa-driver：`efa-2.1.1g` （无变化）
	  • Efa-config：`efa-config-1.13-1` （无变化）
	  • Efa-profile：`efa-profile-1.5-1` （无变化）
	  • Libfabric-aws：`libfabric-aws-1.17.1-1` （从 `libfabric-aws-1.17.0-1` ）
	  • Rdma-core：`rdma-core-43.0-1` （无变化）
	  • Open MPI：`openmpi40-aws-4.1.5-1` （无变化）
	• 在 Amazon Linux 2 上将 Lustre 客户端版本升级到 `2.12`。Lustre 客户端 `2.12` 已经安装在 Ubuntu 20.04、18.04 和 CentOS >= 7.7 上。
	• 在 CentOS 7.6 上将 Lustre 客户端版本升级到 `2.10.8`。
	• 将 NVIDIA 驱动程序升级到版本 `470.182.03` （从版本 `470.141.03` ）。	

更改	描述	日期

	• 将 NVIDIA Fabric Manager 升级到版本 470.182.03 （从版本 470.141.03 ）。
	• 将 NVIDIA CUDA Toolkit 升级到版本 11.8.0（从版本 11.7.1）。
	• 将 NVIDIA CUDA 示例升级到版本 11.8.0。
	• 将 Intel MPI Library 升级到 2021 年版更新 9（从 2021 年版更新 6）。有关更多信息，请参阅 Intel® MPI Library 2019 更新 9。
	• 将 NICE DCV 升级到版本 2023.0-15022 （从版本 2022.2-14521 ）。
	  • server：2023.0.15022-1 （从版本 2022.2-14521-1 ）。
	  • xdcv：2023.0.547-1 （从版本 2022.2.519-1 ）。
	  • gl：2023.0.1027-1 （从版本 2022.2.1012-1 ）。
	  • web_viewer：2023.0.15022-1 （从版本 2022.2.14521-1 ）。
	• 将 aws-cfn-bootstrap 升级到版本 2.0-24。	

更改	描述	日期

	• 升级 CodeBuild 环境在为 Batc AWS h 集群构建容器镜像时使用的镜像：
	  • `aws/codebuild/amazonlinux2-x86_64-standard:4.0`（从 `aws/codebuild/amazonlinux2-x86_64-standard:3.0`）。
	  • `aws/codebuild/amazonlinux2-aarch64-standard:2.0`（从 `aws/codebuild/amazonlinux2-aarch64-standard:1.0`）。
	
	错误修复：
	
	• 修复 Amazon EFS 和亚马逊 FSx 网络安全组验证器，以避免报告虚假错误。
	• 修复了 Image Builder 在 `build-image` 操作期间创建的资源缺少标记的问题。
	• 修复了 `MaxCount` 的更新策略，使其始终对 `MaxCount` 属性进行数值比较。
	• 修复了具有多个网卡的计算节点实例上的 IP 一致性问题。
	• 修复了`StoragePass` 在执行队列参数更新且未更新 Slurm 记账配置`slurm_par`	

更改	描述	日期
	allelcluster_slurmdbd.conf 时替换的问题。 • 修复了使用现有 EFS 文件系统创建集群时导致创建虚安全组的问题。 • 修复了重启 cfn-hup 进程守护程序时导致其失败的问题。 • 在 Slurm 保护模式下，将带有INVALID_REG 标记的动态节点视为引导失败。在 Slurm 注册失败之后，静态节点已被视为引导失败。node_replacement_timeout 有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期

AWS ParallelCluster 3.5.1 版本已发布	AWS ParallelCluster 3.5.1 版本已发布。
	
	增强功能：
	
	• 添加了独立的 `pcluster` CLI 安装程序可执行文件。
	
	更改：
	
	• 将 EFA 安装程序升级到 `1.22.0`。
	  • Efa-driver：`efa-2.1.1g` （从 `efa-2.1.1-1` ）
	  • Efa-config：`efa-config-1.13-1` （从 efa-config-1.12-1 ）
	  • Efa-profile：`efa-profile-1.5-1` （无变化）
	  • Libfabric-aws：`libfabric-aws-1.17.0-1` （从 `libfabric-aws-1.16.1amzn3.0-1` ）
	  • Rdma-core：`rdma-core-43.0-1` （无变化）
	  • Open MPI：`openmpi40-aws-4.1.5-1` （从 `openmpi40-aws-4.1.4-3` ）
	
	将 NICE DCV 升级到版本 `2022.2-14521` 。
		2023 年 3 月 29 日

更改	描述	日期
	• server：2022.2.14521-1 • xdcv：2022.2.519-1 • gl：2022.2.1012-1 • web_viewer：2022.2.14521-1 错误修复： • 修复了在集群更新过程中删除共享 Amazon EBS 卷时因 MountDir 和 /etc/exports 之间的模式匹配而导致的潜在节点启动失败问题。 • 修复了每次 clustermgtd 迭代时 compute_console_output 日志文件被截断的问题。 有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 3.5.0 版本已发布	AWS ParallelCluster 3.5.0 版本已发布。 增强功能： • 使用 AWS ParallelCluster UI 访问和管理集群。 • 在 CloudFormation 模板中添加版本化 AWS ParallelCluster 策略，供您在工作负载中引用。 • 添加可与自己的代码一起使用的 AWS ParallelCluster Python 库。 • 在计算节点引导失败时向 Amaz CloudWatch on 添加计算节点控制台输出的日志记录。 • 集群创建失败时向 `describe-cluster` 输出中添加了包含失败代码和原因的失败字段。 • 添加了验证器以防止在调用子进程模块时注入恶意字符串。 • 在配置静态节点时，如果集群状态更改为 `PROTECTED`，则集群创建将失败。 更改： • 升级到 Slurm 版本`22.05.8`（从版本开始）`22.05.7`	2023 年 2 月 20 日

更改	描述	日期

	• 将 EFA 安装程序升级到 `1.21.0`。
	  • Efa-driver：`efa-2.1.1-1` （从 `efa-2.1`）
	  • Efa-config：`efa-config-1.12-1` （从 efa-config-1.11-1）
	  • Efa-profile：`efa-profile-1.5-1` （无变化）
	  • Libfabric-aws：`libfabric-aws-1.16.1amzn3.0-1` （从 `libfabric-aws-1.16.1` ）
	  • Rdma-core：`rdma-core-43.0-1` （从 `rdma-core-43.0-2` ）
	  • Open MPI：`openmpi40-aws-4.1.4-3` （无变化）
	• 让 Slurm 控制器日志更加详细，并为 Slurm 省电插件启用额外的日志记录。
	
	错误修复：
	• 启用 Slurm 记账后，通过验证集群名称是否不超过 40 个字符来修复集群数据库创建问题。
	• 修复了在 `clustermgtd` Amazon EC2 实例状态检查	

更改	描述	日期
	失败时导致通过 Slurm 重启的计算节点被替换的问题。 • 修复了由于头节点上的 IAM 策略不正确而导致与其他账户共享容量预留的计算节点无法启动的问题。 有关更改的详细信息，请参阅上的 aws-parallelcluster、aws-parallelcluster-cookbookaws-parallelcluster-node、和软件包的`CHANGELOG` 文件。aws-parallelcluster-ui GitHub	
AWS ParallelCluster 3.4.1 版本已发布	AWS ParallelCluster 3.4.1 版本已发布。 错误修复： • 修复了 Slurm 调度器问题，该问题可能导致计算节点内部注册表的更新应用不正确。因此，如果出现此问题， EC2实例可能会变得不可用，或者可能由不正确的实例类型提供支持。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	2023 年 1 月 13 日

更改	描述	日期
AWS ParallelCluster 3.4.0 版本已发布	AWS ParallelCluster 3.4.0 版本已发布。 增强功能： • 增加了对跨多个可用区启动节点的支持，以提高容量可用性。 • 增加了对为每个队列指定多个子网的支持，以提高容量可用性。 • 在 I am/中添加新的配置参数 ResourcePrefix，为创建的 IAM 资源的路径和名称指定前缀 AWS ParallelCluster。 • 添加新的配置部分 DeploymentSettings/，LambdaFunctionsVpcConfig用于指定 AWS ParallelCluster Lambda 函数使用的 Vpc 配置。 • 添加了指定要在集群更新期间在头节点中运行的自定义脚本的功能。使用 Slurm 作为调度器OnNodeUpdated时，可以用 CustomActions/HeadNode/指定脚本。 更改： • 取消为现有文件系统创建 Amazon EFS 挂载目标。 • 使用 `amazon-efs-utils` 挂载 EFS 文件系统。可以使	2022 年 12 月 22 日

更改	描述	日期

	用传输中加密和 IAM 授权用户来挂载 EFS 文件系统。
	• 在 Cent OS7 和 Ubuntu 上安装 stunnel 5.67 以支持 EFS 传输中加密。
	• 将 EFA 安装程序升级到 `1.20.0`（从 `1.18.0`）。
	  • Efa-driver：`efa-2.1`（从 `efa-1.16.0-1`）
	  • Efa-config：`efa-config-1.11-1`（无变化）
	  • Efa-profile：`efa-profile-1.5-1`（无变化）
	  • Libfabric-aws：`libfabric-aws-1.16.1`（从 `libfabric-aws-1.16.0~amzn4.0-1`）
	  • Rdma-core：`rdma-core-43.0-2`（从 `rdma-core-41.0-2`）
	  • Open MPI：`openmpi40-aws-4.1.4-3`（从 `openmpi40-aws-4.1.4-2`）
	• 将 Slurm 升级到版本 `22.05.7`（从 `22.05.5`）。
	• 将 Python 升级到 `3.9.16` 和 `3.7.16`（从 `3.9.15` 和 `3.7.13`）。
	• 使用 Slurm`22.05.7`，处于`IDLE+CLOUD+COMPLET`	

更改	描述	日期
	ING+POWER_DOWN+NOT_RESPONDING 状态的动态节点不会被视为不健康。 有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 3.3.1 版本已发布	AWS ParallelCluster 3.3.1 版本已发布。 更改： • AMIs 在亚马逊 EC2 弃用两年后，官方 AWS ParallelCluster 产品现已上市。 • 将 AWS ParallelCluster API Lambda 的内存大小增加到 2048，以减少冷启动惩罚并避免超时。 错误修复： • 防止替换托管 FSx 的 Lustre 文件系统以及包括计算队列子网 ID 更改在内的集群更新数据丢失。 • SharedStorage `DeletionPolicy` 适用于集群更新操作。 有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包`CHANGELOG` 文件。 GitHub	2022 年 12 月 2 日
AWS ParallelCluster 仅限文档 hpc6id 注意	AWS ParallelCluster 仅限文档的更新 • AWS ParallelCluster 不支持 /设置的 hpc6id 实例类型。HeadNodeInstanceType	2022 年 12 月 2 日

更改	描述	日期

AWS ParallelCluster 3.1.5 版本已发布	AWS ParallelCluster 3.1.5 版本已发布。
	
	增强功能：
	
	• 修复了阻止空闲节点终止的 Slurm 问题。
	• 将 EFA 安装程序升级到 1.18.0
	  • Efa-driver：`efa-1.16.0-1`
	  • Efa-config：`efa-config-1.11-1` （从 `efa-config-1.9-1` ）
	  • Efa-profile：`efa-profile-1.5-1` （无变化）
	  • Libfabric-aws：`libfabric-aws-1.16.0~amzn4.0-1` （从 `libfabric-1.13.2` ）
	  • Rdma-core：`rdma-core-41.0-2` （从 `rdma-core-37.0` ）
	  • Open MPI：`openmpi40-aws-4.1.4-2` （从 `openmpi40-aws-4.1.1-2` ）
	
	更改：
	
	• 将`lambda:ListTags` 和`lambda:UntagResource` 添加
		2022 年 11 月 16 日

更改	描述	日期
	到`ParallelClusterUserRole` 用于集群更新的 AWS ParallelCluster API 堆栈中。 • 将 Intel MPI Library 升级到 2021 年版更新 6（从 2021 年版更新 4）。有关更多信息，请参阅 Intel® MPI Library 2021 更新 6。 • 将 NVIDIA 驱动程序升级到版本 470.141.03（从 470.103.01）。 • 将 NVIDIA Fabric Manager 升级到版本 470.141.03（从 470.103.01）。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 3.3.0 版本已发布	AWS ParallelCluster 3.3.0 版本已发布。 增强功能： • 使用 Slurm 作为调度器时，添加对计算资源的多实例分配配置的支持。有关更多信息，请参阅使用 Slurm 进行多实例类型分配。 • 增加了对使用更新配置进行集群更新时添加和删除 SharedStorage 的支持。有关更多信息，请参阅共享存储。 • 为 E fs 和FsxLustre共享存储设置添加新的配置参数`DeletionPolicy` 以支持存储保留。 • 使用新的配置参数 Scheduling/SlurmSettings/Database 添加对 Slurm 记账的支持。有关更多信息，请参阅 Slurm 使用记账。 AWS ParallelCluster • 增加了对按需容量预留（ODCR）和容量预留资源组的支持。有关更多信息，请参阅使用按需容量预留 (ODCR) 启动实例。 • 添加新的配置参数以指定要在集群中支持的 IMDS 版本或在集群中构建映像基础架构，即 I	2022 年 11 月 2 日

更改	描述	日期
	mds/和 build ImdsSupport、Imds/ImdsSupport、配置。 • 在/ComputeResources部分添加对网络 SlurmQueues/PlacementGroup的支持。 • 增加了对具有多个网络接口并且每个设备仅限一个 ENI 的实例的支持。 • 通过检查附加的安全组中的 CIDR 块，改进了外部 Amazon EFS 文件系统的网络验证。 • 添加了用于检查配置的实例类型是否支持置放群组的验证器。 • 将 NFS 线程数配置为 min(256, max(8, num_cores * 4)) 以确保更好的稳定性和性能。 • 将 NFS 安装移至构建时以减少配置时间。 • 为部署 AWS ParallelCluster API 时创建的、用于通知 docker 镜像构建事件的 EcrImageBuilder SNS 主题启用服务器端加密。 更改： • 更改 SlurmQueues/联网 //的行为`Enabled`。Placement	

更改	描述	日期
	Group现在，它会为每个计算资源创建一个唯一的托管置放群组，而不是为所有计算资源创建一个托管置放群组。 • 添加对 SlurmQueues/网络/PlacementGroup/名称作为首选命名方法的支持。 • 将头节点标签从启动模板移动到了实例定义中，以避免在标签更新时替换头节点。 • 通过 `cloud-init` 执行的脚本而不是通过启动模板中设置的 `CpuOptions` 禁用多线程处理。 • 在 API 基础架构、API Docker 容器和集群 Lambda 资源中将 Python 升级到版本 3.9，将 NodeJS 升级到版本 16。 • 在 `aws-parallelcluster-batch-cli` 中删除了对 Python 3.6 的支持。 • 将 Slurm 升级到版本 `22.05.5`（从 `21.08.8-2`）。 • 将 NVIDIA 驱动程序升级到版本 `470.141.03`（从 `470.129.06`）。 • 将 NVIDIA Fabric Manager 升级到版本 `470.141.03`（从 `470.129.06`）。	

更改	描述	日期

	• 将 NVIDIA CUDA Toolkit 升级到版本 11.7.1（from 11.4.4）。
	• 将 v AWS ParallelCluster irtualenvs 中使用的 Python 从3.7.13升级到。3.9.15
	• 将 EFA 安装程序升级到版本 1.18.0。
	  • Efa-driver：efa-1.16.0-1 （无变化）
	  • Efa-config：efa-config-1.11-1 （from efa-config-1.10-1 ）
	  • Efa-profile：efa-profile-1.5-1 （无变化）
	  • Libfabric-aws：libfabric-aws-1.16.0~amzn4.0-1 （从 libfabric-aws-1.16.0~amzn2.0-1 ）
	  • Rdma-core：rdma-core-41.0-2 （从 rdma-core-37.0 ）
	  • Open MPI：openmpi40-aws-4.1.4-2 （从 openmpi40-aws-4.1.1-2 ）
	• 将 NICE DCV 升级到版本 2022.1-13300 （从 2022.0-12760 ）。	

更改	描述	日期
	• 为 `Queues` 启用 `SingleSubnetValidator` 抑制。 • 当节点处于 `COMPLETING` 状态时不替换 `DRAIN` 节点，因为 Epilog 可能仍在运行。 错误修复： • 修复了 AWS ParallelCluster `ListClusterLogStreams` 命令中过滤器参数的验证失败的问题，即当传递的过滤器不正确时。 • 修复了与EfsSettings其他 SharedStorageSharedStorage/参数一起指定`FileSystemId` 时无法验证EfsSettings参数/的问题。以前不包括 `FileSystemId` 。 • 修复了在配置中更改 SharedStorage 的顺序以及进行其他更改时的集群更新问题。 • 修复`UpdateParallelClusterLambdaRole` 了 AWS ParallelCluster 用于上传日志的 API CloudWatch。 • 修复了在执行任何说明书之前安装程序包时 Cinc 不使用本地 CA 证书捆绑包的问题。	

更改	描述	日期
	• 修复了在设置 `Build:UpdateOsPackages:Enabled:true` 后使用 `pcluster build-image` 升级 ubuntu 时出现的挂起问题。 • 修复了 YAML 集群配置分析在遇到重复密钥时失败的问题。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	
AWS ParallelCluster 添加了仅限文档的 API 参考。	AWS ParallelCluster 仅限文档的更新 • 在文档中添加了版本 3 AWS ParallelCluster API 参考。	2022 年 10 月 27 日

更改	描述	日期
AWS ParallelCluster 3.2.1 版本已发布	AWS ParallelCluster 3.2.1 版本已发布。 增强功能： • 改进逻辑，将主机路由表关联到不同的网卡，从而更好地支持带有多个网卡的 Amazon EC2 实例 NICs。 更改： • 将 NVIDIA 驱动程序升级到版本 470.141.03。 • 将 NVIDIA Fabric Manager 升级到版本 470.141.03。 • 禁用可能对节点性能产生负面影响的 `cron` 作业任务 `man-db` 和 `mlocate`。 • 将 Intel MPI Library 升级到 2021.6.0.602。 • 将 Python 从 3.7.10 升级到 3.7.13 以应对这种安全风险。 错误修复： • 避免集群配置不可用时 `DescribeCluster` 失败。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软	2022 年 10 月 3 日

更改	描述	日期
	件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期

AWS ParallelCluster 3.2.0 版本已发布	AWS ParallelCluster 3.2.0 版本已发布。
	
	增强功能：
	
	• 在 Slurm 中添加了对基于内存的调度的支持。
	  • 在 Slurm 集群配置中配置计算节点实际内存。
	  • 添加新的配置参数 Schedul ing/SlurmSett ings/EnableMem oryBasedScheduling以在 Slurm 中启用基于内存的调度。
	  • 添加新的配置参数 Schedulin Schedulab leMemoryg ComputeRe sources/SlurmQueues/以覆盖调度器在计算节点上看到的默认内存值。
	• 提高了集群配置更新的灵活性，以便尽可能避免停止和启动整个集群。添加新的配置参数 S ch ed SlurmSett ingsulin QueueUpda teStrategyg//以设置在计算节点需要更新和替换配置时使用的首选策略。
	• 在 Ama EC2 zon 实例遇到容量不足问题时，改进可用计算资源的故障转移机制。当节点由于容量不足而启动
		2022 年 7 月 27 日

更改	描述	日期

	失败时，将计算节点禁用一段可配置的时间。
	• 添加对挂载 ONTAP 和 FSx OpenZFS 文件系统的现有 FSx 支持。
	• 添加对挂载现有 Amazon 弹性文件系统、Lustre、 FSx FSx ONTAP 和 OpenZFS 文件系统的多个实例 FSx 的支持。
	• 在创建新文件系统时 FSx ，添加对 Lustre Persistent_2 部署类型的支持。
	• 使用 pcluster 配置向导时，提示用户为支持的实例类型启用 EFA。
	• 添加了对使用 Slurm 重启计算节点的支持。
	• 改进了对 Slurm 电源状态的处理，也考虑节点的手动关闭。
	• 在产品中安装 NVIDIA GDRCopy 2.3 AMIs 以启用低延迟 GPU 内存复制。
	
	更改：
	• 将 EFA 安装程序升级到版本 1.17.2。
	  • EFA 驱动程序：`efa-1.16.0-1`
	  • EFA 配置：`efa-config-1.10-1`	

更改	描述	日期

	• EFA 配置文件：`efa-profile-1.5-1`
	• Libfabric：`libfabric-aws-1.16.0~amzn2.0-1`
	• RDMA 内核：`rdma-core-41.0-2`
	• Open MPI：`openmpi40-aws-4.1.4-2`
	• 将 NICE DCV 升级到版本 2022.0-12760。
	• 将 NVIDIA 驱动程序升级到版本 470.129.06。
	• 将 NVIDIA Fabric Manager 升级到版本 470.129.06。
	• 将根卷和其他卷中的默认 EBS 卷类型从 gp2 更改为 gp3。
	• 对 FSx 于 Lustre 文件系统的更改由以下人员创建： AWS ParallelCluster
	  • 将默认部署类型更改为 `Scratch_2` 。
	  • 将 Lustre 服务器版本更改为 `2.12`。
	• 传递现有的 Placement Group/`true`时，不需要将`PlacementGroup /Enable` d 设置为`Id`。
	• 当 `Placement Group /Enabled` 显式设置	

更改	描述	日期
	为 `false` 时，不允许设置 `PlacementGroup /Id`。 • 为 AWS ParallelCluster 创建的所有资源添加标签 `parallelcluster:cluster-name` 。 • 添加`lambda:ListTags` 和`lambda:UntagResource` ，由 AWS ParallelCluster API 堆栈`ParallelClusterUserRole` 用于集群更新。 • 启用配置参数`HeadNode/Imds/Secured`后仅限根用户和集群管理员用户 IPv6 访问权限。`IMDS` • 对于自定义 AMI，请使用 AMI 根卷大小，而不是ParallelCluster默认的 35 GiB。可以在集群配置文件中更改该值。 • 当配置参数 `Scheduling /SlurmQueues /ComputeResources /SpotPrice` 低于所需的最低竞价型请求履行价格时，自动禁用计算实例集。 • 在更新期间添加或删除某个部分时，在更改集中显示 `requested_value` 和 `current_value` 值。	

更改	描述	日期
	• 禁用深度学习中提供的aws-ubuntu-eni-helper 服务 AMIs，以避免在配置具有多个网卡的实例configure_nw_interface.sh 时发生冲突。 • 删除了对 Python 3.6 的支持。 • 在配置具有多个网卡的实例时，将所有网络接口的 MTU 设置为 9001。 • 配置计算节点 FQDN 时，删除结尾圆点。 • 在 POWERING_DOWN 中管理静态节点。 • 不替换 POWER_DOWN 中的动态节点，因为作业可能仍在运行。 • 只有在更新了集群配置中的 Scheduling 参数时，才会在集群更新时重启 clustermgtd 和 slurmctld 进程守护程序。 • 更新 slurmctld 和 slurmd systemd 服务文件。 • 启用配置参数HeadNode/Imds/Secured后仅允许根用户和集群管理员用户 IPv6 访问 IMDS。 • 设置 Slurm 配置 AuthInfo=cred_expi	

更改	描述	日期

	`re=70` 以缩短在节点不可用时重新排队的作业在重启之前必须等待的时间。
	• 升级第三方说明书依赖项：
	  • apt-7.4.2（从 apt-7.4.0）
	  • line-4.5.2（从 line-4.0.1）
	  • openssh-2.10.3（从 openssh-2.9.1）
	  • pyenv-3.5.1（从 pyenv-3.4.2）
	  • selinux-6.0.4（从 selinux-3.1.1）
	  • yum-7.4.0（从 yum-6.1.1）
	  • yum-epel-4.5.0（从 yum-epel-4.1.2）
	
	错误修复：
	• 修复构建自定义 AMI 时跳过 AWS ParallelCluster 验证和测试步骤的默认行为。
	• 修复了 `computemgtd` 中的文件句柄泄漏问题。
	• 修复了由于响应中尚不可用而偶尔导致已启动的实例立即终止的争用条件。 EC2 DescribeInstances
	• 对于使用 Arm 处理器的实例类型，修复了对 `DisableSimultaneousMultithreading` 参数的支持。	

更改	描述	日期
	• 修复从先前版本升级时的 AWS ParallelCluster API 堆栈更新失败。在 `EcrImageDeletionLambdaRole` 中添加了用于 `ListImage` `PipelineImages` 操作的资源模式。 • 修复 AWS ParallelCluster API 在创建 for Lustre 文件系统时添加了从 Amazon S3 导入或导出所需的权限缺失的问题。 FSx 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 今年迄今为止仅限文档的更新	AWS ParallelCluster 仅限文档的更新。 新章节： • 最佳实践：预算提醒 V3 • 最佳实践：将集群移至新的 AWS ParallelCluster 次要版本或补丁版本 V3 • 使用亚马逊 S3 V3 • 使用竞价型实例 V3 • Slurm 集群保护模式 V3 • AWS ParallelCluster 资源和标记 V3 • 亚马逊 CloudWatch 控制面板 V3 • 与 Amazon L CloudWatch ogs V3 集成 • 弹性织物适配器 V3 • AWS ParallelCluster AMI 定制 V3 • 使用按需容量预留 (ODCR) V3 启动实例 • AMI 修补和亚马逊 EC2 实例更换 V3 • V3 AWS ParallelCluster 工作原理 • 使用 KMS 密@@ AWS 钥 V3 配置共享存储加密 • 在多队列模式集群中运行作业 V3	2022 年 7 月 6 日

更改	描述	日期
	• 使用 AWS ParallelCluster API V3 章节更新： • 最佳实践：网络性能 V3：添加了使用 Elastic Fabric Adaptor 的最佳实践。 • AWS AWS ParallelCluster V3 中的 Identity and Access Management 权限：使用 Amazon FSx for Lu AWS ParallelCluster stre 时进行了各种更新并添加了其他 pcluster 用户策略。 • AWS ParallelCluster 疑难解答 V3：各种更新。	

更改	描述	日期
AWS ParallelCluster 3.1.4 版本已发布	AWS ParallelCluster 3.1.4 版本已发布。 增强功能： • 为 Directory Service/PasswordSecretArn 添加了验证功能，如果不存在密钥，则会失败。 增加了对启用 JWT 身份验证 Slurm 的支持。 更改： • 将 Slurm 升级到版本 21.08.8-2。 • 借助 JWT 支持构建 Slurm。 • 传递现有的 Placement Group/`true`时，不需要将`PlacementGroup` /Enable d 设置为`Id`。 • 添加`lambda:TagResource` 到 ParallelCluster API 堆栈中`ParallelClusterUserRole` 用于创建集群和创建映像。 错误修复： • 修复了使用带 `--filters` 选项的 `export-cluster-logs` 命令时导出集群日志的功能。	2022 年 5 月 16 日

更改	描述	日期
	• 修复 B AWS atch Docker 入口点以使用/home共享目录来协调 Multi-node-Parallel作业执行。 • 将 Slurm 不健康的静态节点设置为关闭时重置节点地址，以避免将容量不足的静态节点视为引导失败节点。 有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 3.1.3 版本已发布	AWS ParallelCluster 3.1.3 版本已发布。 增强功能： • 当切换到其他用户以及在以其他用户身份执行命令时，例如在 SSH 登录期间，将会执行 SSH 密钥创建并创建主目录。 • 在配置参数 Directory Service/DomainName 中增加了对 FQDN 和 LDAP 可分辨名称的支持。新验证器现在会检查这两种语法。 • 头节点上部署的新 `update_directory_service_password.sh` 脚本支持手动更新 SSSD 配置中的 Active Directory 密码。密码由 S AWS ecrets Manager 从集群配置中检索。 • 增加了对在没有默认 VPC 的环境中部署 API 基础架构的支持。 更改： • 在 x86_64 官方版本中禁用更深层次的 C 状态， AMIs 并通过`build-image` 命令 AMIs 创建，以保证高性能和低延迟。	2022 年 4 月 20 日

更改	描述	日期
	• 操作系统程序包更新和安全修复。 • 将 Amazon Linux 2 基础映像更改为 AMIs 与内核 5.10 配合使用。 错误修复： • 由于新的 Image Builder 政策，在成功构建映像`DELETE_FAILED` 后修复构建 EC2 映像堆栈。 • 修复了配置参数 Directory Service/DomainAddr 在包含多个域地址时转换为 ldap_uri SSSD 属性的问题。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件包。aws-parallelcluster-cookbook GitHub	

更改	描述	日期
AWS ParallelCluster 3.1.2 版本已发布	AWS ParallelCluster 3.1.2 版本已发布。 更改： • 将 Slurm 升级到版本 `21.08.6`（从 `21.08.5`）。 错误修复： • 修复了在没有互联网访问权限的子网中部署集群时在计算节点上更新 `/etc/hosts` 文件的问题。 • 修复了计算节点引导，在加入集群之前将等待临时驱动器初始化完成。 有关更改的详细信息，请参阅上的 aws-parallel cluster 软件包的`CHANGELOG` 文件。GitHub	2022 年 3 月 2 日

更改	描述	日期
AWS ParallelCluster 3.1.1 版本已发布	AWS ParallelCluster 3.1.1 版本已发布。 • 通过与通过 AWS Directory Service 管理的 Active Directory (AD) 域集成，增加对多用户群集环境的支持。 • 在集群配置文件中添加UseEc对 2Hostnames 的支持。如果设置为 true，则使用亚马逊的 EC2 默认主机名（例如 ip-1-2-3-4）作为计算节点。 • 增加了对在没有互联网访问权限的子网中创建集群的支持。 • 添加了对每个队列包含多种计算实例类型的支持。 • 在使用 NVIDIA 卡的 ARM 实例上添加了对使用 Slurm 进行 GPU 调度的支持。 • 在 AWS ParallelCluster CLI 中添加 `cluster-name` (-n)、`region` (-r)、`image-id` (-i) 和`cluster-configuration` /`image-configuration` (-c) 的缩写标志。 • 添加对 Lustre AutoImportPolicy参数`NEW_CHANGED_DELETED` 选项 FSx 的支持。	2022 年 2 月 10 日

更改	描述	日期
	• 为计算节点使用的 `EC2LaunchTemplates` 资源添加`parallelcluster:compute-resource-name` 标签。 • 改进在集群中创建的安全组，以便在为某些头节点 and/or 队列指定`SecurityGroups` 参数时允许来自自定义安全组的入站连接。 • 为 ARM 安装 NVIDIA 驱动程序和 CUDA 库。 更改： • 将 Slurm 升级到版本 `21.08.5`（从 `20.11.8`）。 • 将 Slurm 插件升级到版本 `21.08`（从 `20.11`）。 • 将 NICE DCV 升级到版本 `2021.3-11591`（从 `2021.1-10851`）。 • 将 NVIDIA 驱动程序升级到版本 `470.103.01`（从 `470.57.02`）。 • 将 NVIDIA Fabric Manager 升级到版本 `470.103.01`（从 `470.57.02`）。 • 将 CUDA 升级到版本 `11.4.4`（从 `11.4.0`）。 • Intel MPI 更新至 2021 年版更新 4（从 2019 年版更新 8 进行更新）。有关更	

更改	描述	日期
	多信息，请参阅 Intel® MPI Library 2021 更新 4。 • 升级 PMIx 到版本`3.2.3`（从`3.1.5`）。 • 删除了将失败的计算节点转储到 `/home/logs/compute` 。计算节点日志文件在 Amazon EC2 控制台日志 CloudWatch 和控制台日志中均可用。 • 启用潜在抑制 `SlurmQueues` 和 `ComputeResources` 长度验证器。 • 在 Amazon Linux 2 上禁用实例启动时的程序包更新。 • 在构建 AWS ParallelCluster 自定义图像时禁用 Amazon EC2 ImageBuilder 增强版图像元数据。 • 将`cloud-init` 数据源明确设置为。 EC2这可节省 Ubuntu 和 CentOS 平台的启动时间。 • 在计算实例集启动模板名称中使用计算资源名称而不是实例类型。 • 将 stderr 和 stdout 重定向到 CLI 日志文件，以防止 pcluster CLI 输出中出现不需要的文本。 • 移动 configure/install 食谱以将调用的食谱与主食谱	

更改	描述	日期
	分开。现有的入口点保持不变，并且向后兼容。 • 在 AMI 构建期间下载 Intel HPC 平台的依赖项，以避免在集群创建期间联系互联网。 • 配置 Slurm 节点时不从计算资源名称中删除 `-`。 • 未安装 NVIDIA 驱动程序时，请勿 GPUs 在 Slurm 中进行配置。 • 修复了 `BatchUserRole` 中的 `ecs:ListContainerInstances` 权限。 • 修复了未指定前缀时的集群日志导出问题，以前导出为 `None` 前缀。 • 修复了集群更新失败时不执行回滚的问题。 • 修复了 `BatchUserRole` 中的 `ecs:ListContainerInstances` 权限。 • 修复了 `HeadNode` 的 `RootVolume` 架构，如果指定了不支持的 `KmsKeyId`，则会引发错误。 • 修复 Amazon FSx 缺少要在 CloudWatch 控制面板中显示的指标。	

更改	描述	日期
	• 修复了 `EfaSecurityGroupValidator` 。以前，在提供自定义安全组并启用 EFA 的情况下，它可能会产生假失败。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件aws-parallelcluster-cookbook包。aws-parallelcluster-node GitHub	
AWS ParallelCluster 3.0.3 版本已发布	AWS ParallelCluster 3.0.3 版本已发布。 • 在 Amazon Linux 2 上禁用 `log4j-cve-2021-44228-hotpatch` 代理 (`Log4jHotPatch`) 以避免潜在的性能降低。有关更多信息，请参阅适用于 Apache Log4j 的 Amazon Linux 热补丁公告。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件包。aws-parallelcluster-cookbook GitHub	2022 年 1 月 17 日

更改	描述	日期
AWS ParallelCluster 3.0.2 版本已发布	AWS ParallelCluster 3.0.2 版本已发布。 将弹性结构适配器安装程序升级到 1.14.1 • EFA 配置：`efa-config-1.9-1` （从 `efa-config-1.9` ） • EFA 配置文件：`efa-profile-1.5-1` （从 `efa-profile-1.5` ） • EFA 内核模块：`efa-1.14.2` （从 `efa-1.13.0` ） • RDMA 内核：`rdma-core-37.0` （从 `rdma-core-35` ） • libfabric：`libfabric-1.13.2` （从 `libfabric-1.13.0` ） • Open MPI：`openmpi40-aws-4.1.1-2` （无变化） GPUDirect 如果实例类型支持 RDMA，则始终处于启用状态。GdrSupport配置选项无效。 有关更改的详细信息，请参阅 aws-parallelcluster `CHANGELOG` 的文件和上的软件包。aws-parallelcluster-cookbookaws-parallelcluster-node GitHub	2021 年 11 月 5 日

更改	描述	日期
AWS ParallelCluster 3.0.1 版本已发布	AWS ParallelCluster 3.0.1 版本已发布。 集群配置迁移工具 • 客户现在可以将其集群配置从 AWS ParallelCluster 版本 2 格式迁移到基于 YAML 的 AWS ParallelCluster 版本 3 格式。有关更多信息，请参阅 pclust er3-config-convert er。 可以停止头节点 • 停止计算队列后，可以使用 Amazon EC2 控制台或 st op-instances AWS CLI 命令停止头节点，然后再重新启动。 从`~/.aws/config` 文件中读取的默认 AWS 区域 • 对于 pcluster 命令，如果未在配置文件、环境或命令行中指定 AWS 区域，则使用`~/.aws/co nfig` 文件`[default]` 部分的`region`设置中指定的默认区域。 AWS 有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的	2021 年 10 月 27 日

更改	描述	日期
	软件包。aws-parallelcluster-cookbookaws-parallelcluster-node GitHub	

更改	描述	日期
AWS ParallelCluster 3.0.0 版本已发布	AWS ParallelCluster 3.0.0 版本已发布。 支持通过 Amazon API Gateway 进行集群管理 • 现在，客户可以使用 Amazon API Gateway 通过 HTTP 端点管理和部署集群。这为脚本化或事件驱动的工作流程开辟了新的可能性。 为了与此 API 兼容， AWS ParallelCluster 命令行界面 (CLI) 也进行了重新设计，并包括一个新的 JSON 输出选项。这项新功能使客户也可以使用 CLI 实现类似的构造块功能。 改进了自定义 AMI 的创建 • 现在，客户可以使用更强大的流程来 AMIs 使用 EC2 Image Builder 创建和管理自定义内容。现在 AMIs 可以通过单独的 AWS ParallelCluster 配置文件管理自定义，也可以在命令行界面中使用 pcluster build-image 命令进行创建。 AWS ParallelCluster	2021 年 9 月 10 日

更改	描述	日期
	有关更改的详细信息，请参阅 aws-parallelcluster CHANGELOG 的文件和上的软件包。aws-parallelcluster-cookbookaws-parallelcluster-node GitHub	

PCUI

更改	描述	日期
PCUI 版本 2025.04.0 已发布	PCUI 版本 2025.04.0 已发布 重大变化： • 删除 PC 版本的默认值。现在，用户必须指定要使用的电脑版本。 设施特点： • 添加新的堆栈参数 “AdditionalPolicies PCAPI ”，除了默认权限外，还可为 ParallelCluster API Lambda 角色添加自定义权限。 错误修复： • 通过使用 PCUI 模板修复私有子网中的 PCUI 部署问题，然后返回正确的模板。URLs	2025 年 4 月 16 日

更改	描述	日期
	• 修复了无法在 Job 状态选项卡中加载 200 多个作业的问题。（参见 https://github.com/aws/aws-parallelcluster-ui/issues/376）。 安全性： • 将 Python 从 3.9 升级到 3.12。 • 将交叉生成从 7.0.3 升级到 7.0.6 以解决 CVE-2024-21538 漏洞。 • 将请求从 2.31.0 升级到 2.32.0 以解决 CVE-2024-35195。 • 将 urllib3 从 1.26.18 升级到 1.26.19 以解决 CVE-2024-37891。 • 将密码学从 42.0.4 升级到 44.0.1 以解决 CWE-1395。 • 将证书从 2023.7.22 升级到 2024.7.4 以解决 CVE-2024-39689。 • 将 jinja2 从 3.1.3 升级到 3.1.6 以解决 CVE-2024-56201 和 CVE-2024-56326。 • 将 serverless_wsgi.py 升级到 3.0.5 版。 • 将 Werkzeug 从 2.3.8 升级到 3.0.6 版，以解决 CVE-2024-34069、CVE	

更改	描述	日期
	-2024-49766 和 CVE-2024-49767。 • 将 Axios 从 1.6.7 升级到 1.8.2 版本以解决 CVE-2024-39338。 • 将 Next.js 从 14.1.1 升级到版本 14.2.25 以解决 CVE-2024-51479、CVE-2024-46982 和 CVE-2025-29927。 • 将 idna 从 3.4 升级到版本 3.7 以解决 CVE-2024-3651。 • 将 nanoid 从 3.3.7 升级到 3.3.8 版本以解决 CVE-2024-55565。 • 将 python-jose 从 3.3.0 升级到 3.4.0 版本以解决 CVE-2022-29217。	
PCUI 版本 2024.11.0 已发布	PCUI 版本 2024.11.0 已发布 错误修复： • 明确设置 ECR 私有存储库的策略，以防止在影响到 Lambda 函数的堆栈更新时删除策略。该策略包括 Lambda 函数获取代码所需的权限。	2024 年 11 月 22 日

更改	描述	日期
PCUI 版本 2024.10.0 已发布	PCUI 版本 2024.10.0 已发布 更改： • 添加对 AWS ParallelCluster 3.11.1 的支持。 • 在向导中添加对按需容量预留和容量块的支持。 • 将 g6、m7 和 p5 系列添加到向导中支持的实例类型列表中。 • 添加新的堆栈可选参数，为 PCUI 和 Cognito 配置自定义域。 错误修复： • 修复了破坏自定义域名设置的错误。 安全性： • 将 Flask-cors 从 3.0.10 升级到 4.0.2 以解决 CVE-2024-6221 漏洞。 • 将 lint-staged 从 13.0.3 升级到 15.2.5，以解决 CVE-2024-4068 漏洞。 • 完整更改日志	2024 年 10 月 22 日

更改	描述	日期
发布了 PCUI 版本 2024.05.0	发布了 PCUI 版本 2024.05.0。 错误修复： • 修复了用户打开“作业状态”面板时前端屏蔽 UI 的错误。 • 完整更改日志	2024 年 5 月 14 日
发布了 PCUI 版本 2024.04.0	发布了 PCUI 版本 2024.04.0。 设施特点： • 增加了对 AWS ParallelCluster 版本 3.9.1 的支持 • 完整更改日志	2024 年 4 月 17 日
发布了 PCUI 版本 2024.03.0	发布了 PCUI 版本 2024.03.0。 设施特点： • 增加了对 AWS ParallelCluster 版本 3.9.0 的支持 • 增加了对 Ubuntu 22.04 和 Red Hat Enterprise Linux 9 的支持 • 已弃用 Ubuntu 18.04 错误修复 • 修复了在使用多个集群时导致某些集群不显示的问题 有关更改的详细信息，请参阅上的`aws-parallelcluster-ui` 软件包`CHANGELOG` 文件 GitHub。	2024 年 3 月 12 日

更改	描述	日期
发布了 PCUI 版本 2024.02.0	发布了 PCUI 版本 2024.02.0 更改： • 将 Lambda 运行时环境更新为 Python v3.9 有关更改的详细信息，请参阅上的aws-parallelcluster-ui 软件包CHANGELOG 文件 GitHub。	2024 年 2 月 8 日
发布了 PCUI 版本 2023.12.0	发布了 PCUI 版本 2023.12.0。 设施特点： • 增加了对使用私有联网部署 PCUI 的支持。 • 增加了选择对 PCUI 和 PCAPI 基础设施创建的每个 IAM 角色应用权限边界的可能性 • 增加了选择对 PCUI 和 PCAPI 基础设施创建的每个 IAM 角色和策略应用前缀的可能性。 • 增加了对 ParallelCluster 版本 3.8.0 的支持，向导中没有功能对等。 有关更改的详细信息，请参阅上的aws-parallelcluster-ui软件包CHANGELOG 文件 GitHub。	2023 年 12 月 21 日

更改	描述	日期
发布了 PCUI 版本 2023.10.0	发布了 PCUI 版本 2023.10.0。 设施特点： • 增加了对 ParallelCluster 3.7.2 的支持，向导中的功能对等仅限于 FSx 文件缓存和与多种实例类型的基于内存的调度兼容。 错误修复： • 修复了当 PCUI 没有与 Cost Explorer 交互的权限时导致 UI 错误的问题。 改进 • 通过将访问令牌 TTL 从 10 分钟缩短为 5 分钟，提高了安全性。 有关更改的详细信息，请参阅上的aws-parallelcluster-ui软件包`CHANGELOG` 文件 GitHub。	2023 年 10 月 20 日

更改	描述	日期
发布了 PCUI 版本 2023.06.0	发布了 PCUI 版本 2023.06.0。 更改： • 已将默认 AWS ParallelCluster API 版本升级到 3.6.0。 错误修复： • 修复了 AWS GovCloud（美国西部）区域部署中断的问题。 • 现在，在创建开始后，拆分面板可以正确加载集群详细信息。 注意: • 成本监控功能不在 AWS GovCloud（美国）地区提供。 有关更改的详细信息，请参阅上的aws-parallelcluster-ui软件包`CHANGELOG` 文件 GitHub。	2023 年 6 月 7 日

更改	描述	日期
发布了 PCUI 版本 2023.05.0	发布了 PCUI 版本 2023.05.0。 增强功能： • 从 3.6.0 AWS ParallelCluster 版本开始，添加对 RHEL 8 的支持。 • 添加了集群成本监控功能。 • 从 3.6.0 AWS ParallelCluster 版开始，增加队列和计算资源配额。 更改： • 改进了集群创建向导的用户界面。 • 提高了 PCUI 部署的速度。 • 改进了添加新用户的界面。 • 队列默认位于头节点子网中。 错误修复： • 集群创建完成后，切换到正确的区域。 • 修复了“编辑集群”功能中的加载指示器显示问题。 • 修复移除 EBS SnapshotId 属性时创建集群的问题。 有关更改的详细信息，请参阅上的aws-parallelcluster-ui软件包`CHANGELOG` 文件 GitHub。	2023 年 5 月 16 日

更改	描述	日期
发布了 PCUI 版本 2023.04.0	发布了 PCUI 版本 2023.04.0。 增强功能： • 重新设计了集群创建向导。 • 重新设计了集群日志页面。 • 为共享存储添加了自定义名称设置。 • 在向集群添加存储时添加了多个存储选项。 • 添加对 Amazon EFS 和 Lustre FSx 的`DeletionPolicy` 支持。 • 在集群配置中添加了 `ImdsSupport` 设置。 • 增加了对 C7 实例类型的支持。 • 添加了教程 “恢复到以前的 SAWS ystems Manager 文档版本”。 更改： • 集群配置 YAML 的大小最大可达到 1MB。 • 用户不会因为使用 Boto3 IAM 临时凭证进行授权而注销。 • 选择 HPC 实例时禁用了多线程选项。 • 删除了集群创建页面上的禁用回滚功能。	2023 年 4 月 17 日

更改	描述	日期
	• 在提供必需的信息之前，用户无法使用 PCUI。 • 最多可以添加 10 个队列。 • 在 PCUI 安装过程中不覆盖 `SSM-SessionManager RunShell` 文档。 错误修复： • 修复了损坏的重置密码链接。 • 修复了因 `EcrPrivateRepository` 不为空而导致 `delete stack` 损坏的问题 • 修复了“多用户管理属性”部分中“生成 SSH 密钥”复选框的初始化问题。 • 修复了因作业具有未定义属性而导致崩溃的问题。 • 修复了刮刮 FSx 设置。 • 修复了“启动和停止实例”按钮，单击一次后仍处于启用状态。 有关更改的详细信息，请参阅上的aws-parallelcluster-ui软件包`CHANGELOG` 文件 GitHub。	

Terraform

更改	描述	日期
1.1.0 的 Terraform Provider 已发布 AWS ParallelCluster	错误修复： • 修复了使用 ParallelCluster API 3.11.x 部署带有登录节点的集群时导致 terraform-apply 失败的问题。	2024 年 12 月 6 日
1.1.0 版的 Terraform 模块已发布 AWS ParallelCluster	更改： • 在所有模块示例中使用 AWS ParallelCluster Terraform Provider 1.x。 • 在所有带有堆栈名称 ParallelCluster的示例中使用 ParallelCluster API 3.11.1 API。 • 在所有模块示例中部署登录节点。	2024 年 12 月 6 日
1.0.0 的 Terraform Provider 已发布 AWS ParallelCluster	设施特点： • 完整更改日志	2024 年 6 月 26 日
1.0.0 版的 Terraform 模块已发布 AWS ParallelCluster	设施特点： • 完整更改日志	2024 年 6 月 26 日

本文属于机器翻译版本。若本译文内容与英语原文存在差异，则一律以英文原文为准。