사용 설명서

AWS PCS

AWS PCS: 사용 설명서

Table of Contents

AWS 병렬 컴퓨팅 서비스란 무엇입니까?

AWS 병렬 컴퓨팅 서비스(AWS PCS)는 고성능 컴퓨팅(HPC) 워크로드를 더 쉽게 실행 및 확장하고 Slurm을 AWS 사용하여에서 과학 및 엔지니어링 모델을 구축할 수 있는 관리형 서비스입니다. AWS PCS를 사용하여 동급 최고의 컴퓨팅, 스토리지, 네트워킹 및 시각화를 통합하는 AWS 컴퓨팅 클러스터를 구축합니다. 시뮬레이션을 실행하거나 과학 및 엔지니어링 모델을 구축합니다. 내장된 관리 및 관찰 기능을 사용하여 클러스터 작업을 간소화하고 간소화합니다. 사용자가 익숙한 환경에서 애플리케이션과 작업을 실행할 수 있도록 하여 사용자가 연구 및 혁신에 집중할 수 있도록 지원합니다.

주제

AWS PCS의 개념

AWS PCS의 클러스터에는 하나 이상의 컴퓨팅 노드 그룹과 연결된 대기열이 하나 이상 있습니다. 작업은 대기열에 제출되고 컴퓨팅 노드 그룹에서 정의한 EC2 인스턴스에서 실행됩니다. 이러한 기반을 사용하여 정교한 HPC 아키텍처를 구현할 수 있습니다.

클러스터

클러스터는 리소스를 관리하고 워크로드를 실행하기 위한 리소스입니다. 클러스터는 컴퓨팅, 네트워킹, 스토리지, 자격 증명 및 작업 스케줄러 구성의 어셈블리를 정의하는 AWS PCS 리소스입니다. 사용할 작업 스케줄러(현재 Slurm), 원하는 스케줄러 구성, 클러스터를 관리할 서비스 컨트롤러, 클러스터 리소스를 시작할 VPC를 지정하여 클러스터를 생성합니다. 스케줄러는 작업을 수락 및 예약하고 해당 작업을 처리하는 컴퓨팅 노드(EC2 인스턴스)도 시작합니다.

컴퓨팅 노드 그룹

컴퓨팅 노드 그룹은 AWS PCS가 작업을 실행하거나 클러스터에 대한 대화형 액세스를 제공하는 데 사용하는 컴퓨팅 노드 모음입니다. 컴퓨팅 노드 그룹을 정의할 때 Amazon EC2 인스턴스 유형, 최소 및 최대 인스턴스 수, 대상 VPC 서브넷, Amazon Machine Image(AMI), 구매 옵션 및 사용자 지정 시작 구성과 같은 일반적인 특성을 지정합니다. AWS PCS는 이러한 설정을 사용하여 컴퓨팅 노드 그룹에서 컴퓨팅 노드를 효율적으로 시작, 관리 및 종료합니다.

대기열

특정 클러스터에서 작업을 실행하려는 경우 특정 대기열(파티션이라고도 함)에 작업을 제출합니다. 작업은 AWS PCS가 컴퓨팅 노드 그룹에서 실행되도록 예약할 때까지 대기열에 남아 있습니다. 각 대기

열에 하나 이상의 컴퓨팅 노드 그룹을 연결합니다. 작업 스케줄러에서 제공하는 다양한 예약 정책을 사용하여 기본 컴퓨팅 노드 그룹 리소스에서 작업을 예약하고 실행하려면 대기열이 필요합니다. 사용자는 컴퓨팅 노드 또는 컴퓨팅 노드 그룹에 직접 작업을 제출하지 않습니다.

시스템 관리자

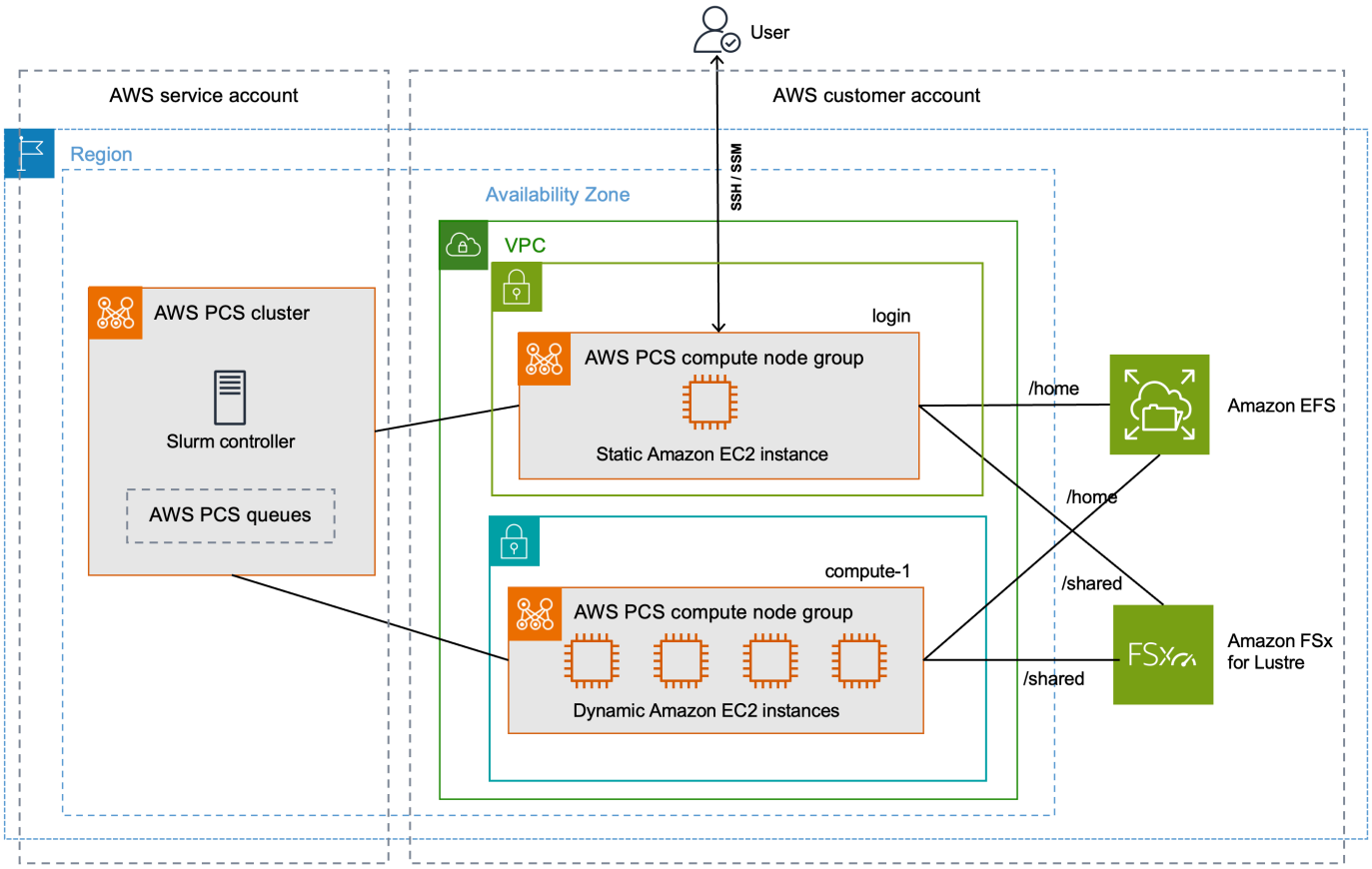
시스템 관리자는 클러스터를 배포, 유지 관리 및 운영합니다. AWS Management Console, AWS PCS API 및 AWS SDK를 통해 AWS PCS에 액세스할 수 있습니다. SSH를 통해 또는 관리 작업을 실행하고, 작업을 실행하고 AWS Systems Manager, 데이터를 관리하고, 기타 쉘 기반 활동을 수행할 수 있는 특정 클러스터에 액세스할 수 있습니다. 자세한 내용은 AWS Systems Manager 설명서를 참조하세요.

최종 사용자

최종 사용자는 클러스터를 배포하거나 운영할 day-to-day 책임이 없습니다. 터미널 인터페이스(예: SSH)를 사용하여 클러스터 리소스에 액세스하고, 작업을 실행하고, 데이터를 관리하고, 기타 쉘 기반 활동을 수행합니다.

AWS 병렬 컴퓨팅 서비스 시작하기

이 자습서에서는 AWS PCS를 사용해 볼 수 있는 간단한 클러스터를 생성합니다. 다음 그림은 클러스터의 설계를 보여줍니다.



자습서 클러스터 설계에는 다음과 같은 주요 구성 요소가 있습니다.

- AWS PCS 네트워킹 요구 사항을 충족하는 VPC 및 서브넷입니다.
- 공유 홈 디렉터리로 사용되는 Amazon EFS 파일 시스템입니다.
- 공유 고성능 디렉터리를 제공하는 Amazon FSx for Lustre 파일 시스템입니다.
- Slurm 컨트롤러를 제공하는 AWS PCS 클러스터입니다.
- 2 AWS PCS 컴퓨팅 노드 그룹.
  - 시스템에 대한 셸 기반 대화형 액세스를 제공하는 `login` 노드 그룹입니다.
  - `compute-1` 노드 그룹은 작업을 실행할 수 있도록 탄력적으로 규모를 조정하는 인스턴스를 제공합니다.
- `compute-1` 노드 그룹의 EC2 인스턴스로 작업을 전송하는 대기열 1개.

클러스터에는 보안 그룹, IAM 역할 및 EC2 시작 템플릿과 같은 추가 AWS 리소스가 필요하며, 이는 다이어그램에 표시되지 않습니다.

> **Note**
>
> Bash 셸에서이 주제의 명령줄 단계를 완료하는 것이 좋습니다. Bash 셸을 사용하지 않는 경우 줄 연속 문자 및 변수 설정 및 사용 방식과 같은 일부 스크립트 명령을 통해 셸이 조정되어야 합니다. 또한 쉘의 인용 및 이스케이프 규칙이 다를 수 있습니다. 자세한 내용은 AWS Command Line Interface 버전 2 사용 설명서의에서 문자열이 있는 따옴표 및 리터럴 AWS CLI을 참조하세요.

주제

AWS PCS를 시작하기 위한 사전 조건

다음 주제를 참조하여 AWS 계정 및 로컬 개발 환경을 AWS PCS용으로 준비합니다.

주제

- AWS PCS AWS CLI 용 설치
- AWS PCS에 필요한 IAM 권한

에 가입 AWS 하고 관리 사용자를 생성합니다.

다음 작업을 완료하여 AWS 병렬 컴퓨팅 서비스(AWS PCS)를 설정합니다.

주제

- 에 가입 AWS 계정
- 관리자 액세스 권한이 있는 사용자 생성

에 가입 AWS 계정

이 없는 경우 다음 단계를 AWS 계정완료하여 생성합니다.

에 가입하려면 AWS 계정

1. https://portal.aws.amazon.com/billing/signup을 엽니다.
2. 온라인 지시 사항을 따릅니다.

   등록 절차 중 전화 또는 텍스트 메시지를 받고 전화 키패드로 확인 코드를 입력하는 과정이 있습니다.

   에 가입하면 AWS 계정AWS 계정 루트 사용자이 생성됩니다. 루트 사용자에게는 계정의 모든 AWS 서비스 및 리소스에 액세스할 권한이 있습니다. 보안 모범 사례는 사용자에게 관리 액세스 권한을 할당하고, 루트 사용자만 사용하여 루트 사용자 액세스 권한이 필요한 작업을 수행하는 것입니다.

AWS 는 가입 프로세스가 완료된 후 확인 이메일을 보냅니다. 언제든지 https://aws.amazon.com/으로 이동하고 내 계정을 선택하여 현재 계정 활동을 보고 계정을 관리할 수 있습니다.

관리자 액세스 권한이 있는 사용자 생성

에 가입한 후 일상적인 작업에 루트 사용자를 사용하지 않도록 관리 사용자를 AWS 계정보호 AWS IAM Identity Center, AWS 계정 루트 사용자활성화 및 생성합니다.

보안 AWS 계정 루트 사용자

1. 루트 사용자를 선택하고 AWS 계정 이메일 주소를 입력하여 계정 소유자AWS Management Console로에 로그인합니다. 다음 페이지에서 비밀번호를 입력합니다.

   루트 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS 로그인 User Guide의 루트 사용자로 로그인을 참조하세요.
2. 루트 사용자의 다중 인증(MFA)을 활성화합니다.

   지침은 IAM 사용 설명서의 AWS 계정 루트 사용자(콘솔)에 대한 가상 MFA 디바이스 활성화를 참조하세요.

관리자 액세스 권한이 있는 사용자 생성

1. IAM Identity Center를 활성화합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 AWS IAM Identity Center설정을 참조하세요.
2. IAM Identity Center에서 사용자에게 관리 액세스 권한을 부여합니다.

   를 자격 증명 소스 IAM Identity Center 디렉터리 로 사용하는 방법에 대한 자습서는 사용 AWS IAM Identity Center 설명서의 기본값으로 사용자 액세스 구성을 IAM Identity Center 디렉터리 참조하세요.

관리 액세스 권한이 있는 사용자로 로그인

- IAM IDentity Center 사용자로 로그인하려면 IAM Identity Center 사용자를 생성할 때 이메일 주소로 전송된 로그인 URL을 사용합니다.

  IAM Identity Center 사용자를 사용하여 로그인하는 데 도움이 필요하면 AWS 로그인 사용 설명서의 AWS 액세스 포털에 로그인을 참조하세요.

추가 사용자에게 액세스 권한 할당

1. IAM Identity Center에서 최소 권한 적용 모범 사례를 따르는 권한 세트를 생성합니다.

   지침은AWS IAM Identity Center 사용 설명서의 Create a permission set를 참조하세요.
2. 사용자를 그룹에 할당하고, 그룹에 Single Sign-On 액세스 권한을 할당합니다.

   지침은 AWS IAM Identity Center 사용 설명서의 Add groups를 참조하세요.

AWS PCS AWS CLI 용 설치

최신 버전의를 사용해야 합니다 AWS CLI. 자세한 내용은 버전 2 사용 설명서의 최신 버전의에 설치 또는 업데이트를 AWS CLI AWS Command Line Interface 참조하세요.

를 구성해야 합니다 AWS CLI. 자세한 내용은 버전 2 사용 설명서의 구성을 AWS CLI 참조하세요. AWS Command Line Interface

명령 프롬프트에 다음 명령을 입력하여를 확인합니다. 그러면 도움말 정보가 AWS CLI표시됩니다.

```
aws pcs help
```

AWS PCS에 필요한 IAM 권한

사용 중인 IAM 보안 주체는 AWS PCS IAM 역할, 서비스 연결 역할 AWS CloudFormation, VPC 및 관련 리소스로 작업할 수 있는 권한이 있어야 합니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 및 서비스 연결 역할 AWS 병렬 컴퓨팅 서비스를 위한 ID 및 액세스 관리생성을 참조하세요. https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_create-service-linked-role.html 이 가이드의 모든 단계를 동일한 사용자로 완료해야 합니다. 현재 사용자를 확인하려면 다음 명령을 실행합니다.

```
aws sts get-caller-identity
```

AWS PCS 자습서와 AWS CloudFormation 함께 사용

AWS PCS 자습서에는 여러 단계가 있으며, 이는 AWS PCS 클러스터의 부분과 이를 생성하는 데 필요한 절차를 이해하는 데 도움이 되도록 마련된 것입니다. 자습서를 한 번 이상 진행하는 것이 좋습니다. 관련된 내용을 잘 이해한 후에는 AWS CloudFormation 를 사용하여 자동화를 통해 샘플 클러스터를 빠르게 생성할 수 있습니다.

CloudFormation 는 AWS 인프라 배포를 예측 가능하고 반복적으로 생성하고 프로비저닝할 수 있는 AWS 서비스입니다. CloudFormation 템플릿을 사용하여 샘플 클러스터에 대한 AWS 리소스를 스택이라는 단일 단위로 자동으로 프로비저닝할 수 있습니다. 스택 작업을 마치면 스택을 삭제할 수 있습니다.

자세한 내용은 CloudFormation 및 AWS PCS 시작하기 단원을 참조하십시오.

AWS PCS용 VPC 및 서브넷 생성

CloudFormation 템플릿을 사용하여 VPC 및 서브넷을 생성할 수 있습니다. 다음 URL을 사용하여 CloudFormation 템플릿을 다운로드한 다음 CloudFormation 콘솔에 템플릿을 업로드하여 새 CloudFormation 스택을 생성합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 콘솔 사용을 참조하세요.

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

CloudFormation 콘솔에서 템플릿을 연 상태에서 다음 옵션을 입력합니다. 템플릿에 제공된 기본값을 사용할 수 있습니다.

- 스택 이름 제공에서 다음을 수행합니다.
  - 스택 이름에 다음을 입력합니다.

    ```
    hpc-networking
    ```
- 파라미터에서:
  - VPC에서:
    - CidrBlock에서 다음을 입력합니다.

      ```
      10.3.0.0/16
      ```
  - 서브넷 A에서:
    - CidrPublicSubnetA에서 다음을 입력합니다.

      ```
      10.3.0.0/20
      ```
    - CidrPrivateSubnetA에서 다음을 입력합니다.

      ```
      10.3.128.0/20
      ```
  - 서브넷 B에서:
    - CidrPublicSubnetB에서 다음을 입력합니다.

      ```
      10.3.16.0/20
      ```
    - CidrPrivateSubnetB에서 다음을 입력합니다.

```
10.3.144.0/20
```

- 서브넷 C에서:
  - ProvisionSubnetsC에서 True를 선택합니다.
  - CidrPublicSubnetC에서 다음을 입력합니다.

```
10.3.32.0/20
```

  - CidrPrivateSubnetC에서 다음을 입력합니다.

```
10.3.160.0/20
```

- 기능에서:
  - 가 IAM 리소스를 생성할 AWS CloudFormation 수 있음을 승인합니다 확인란을 선택합니다.

CloudFormation 스택의 상태를 모니터링합니다. 에 도달하면 새 VPC에서 기본 보안 그룹의 ID를 `CREATE_COMPLETE`찾습니다. 자습서의 뒷부분에서 ID를 사용합니다.

클러스터 VPC의 기본 보안 그룹 찾기

새 VPC에서 기본 보안 그룹의 ID를 찾으려면 다음 절차를 따릅니다.

- Amazon VPC 콘솔로 이동하세요.
- VPC 대시보드에서 VPC로 필터링을 선택합니다.
  - 이름이 로 시작하는 VPC를 선택합니다`hpc-networking`.
  - 보안에서 보안 그룹을 선택합니다.
- 라는 그룹의 보안 그룹 ID를 찾습니다`default`. 설명가 있습니다`default VPC security group`. 나중에 ID를 사용하여 EC2 시작 템플릿을 구성합니다.

AWS PCS에 대한 보안 그룹 생성

AWS PCS는 보안 그룹을 사용하여 클러스터 및 해당 컴퓨팅 노드 그룹으로 들어오고 나가는 네트워크 트래픽을 관리합니다. 이 주제에 대한 자세한 내용은 섹션을 참조하세요보안 그룹 요구 사항 및 고려 사항.

이 단계에서는 CloudFormation 템플릿을 사용하여 두 개의 보안 그룹을 생성합니다.

- AWS PCS 컨트롤러, 컴퓨팅 노드 및 로그인 노드 간의 통신을 지원하는 클러스터 보안 그룹입니다.
- SSH 액세스를 지원하기 위해 선택적으로 로그인 노드에 추가할 수 있는 인바운드 SSH 보안 그룹

AWS PCS에 대한 보안 그룹 생성

CloudFormation 템플릿을 사용하여 보안 그룹을 생성할 수 있습니다. 다음 URL을 사용하여 CloudFormation 템플릿을 다운로드한 다음 CloudFormation 콘솔에 템플릿을 업로드하여 새 CloudFormation 스택을 생성합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 콘솔 사용을 참조하세요.

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/getting_started/assets/pcs-cluster-sg.yaml
```

AWS CloudFormation 콘솔에서 템플릿을 연 상태에서 다음 옵션을 입력합니다. 일부 옵션은 템플릿에 미리 채워집니다. 기본값으로 두면 됩니다.

- 스택 이름 제공에서
  - 스택 이름에 다음을 입력합니다.

    ```
    getstarted-sg
    ```
- 파라미터 아래
  - VpcId에서 이름이 로 시작하는 VPC를 선택합니다`hpc-networking`.
  - (선택 사항) ClientIpCidr에서 인바운드 SSH 보안 그룹에 대한 보다 제한적인 IP 범위를 입력합니다. 자체 IP/서브넷(자체 ip의 경우 x.x.x.x.x/32, 범위의 경우 x.x.x.x/24로 제한하는 것이 좋습니다. x.x.x.x를 자체 PUBLIC IP로 바꿉니다. https://ifconfig.co/ 같은 도구를 사용하여 퍼블릭 IP를 가져올 수 있습니다.)

CloudFormation 스택의 상태를 모니터링합니다. 보안 그룹 리소스`CREATE_COMPLETE`에 도달하면 준비됩니다.

두 가지 보안 그룹이 생성되며 이름은 다음과 같습니다.

- `cluster-getstarted-sg` - 클러스터 보안 그룹입니다.
- `inbound-ssh-getstarted-sg` - 인바운드 SSH 액세스를 허용하는 보안 그룹입니다.

AWS PCS에서 클러스터 생성

AWS PCS에서 클러스터는 리소스를 관리하고 워크로드를 실행하기 위한 영구 리소스입니다. 새 VPC 또는 기존 VPC의 서브넷에서 특정 스케줄러(AWS PCS는 현재 Slurm 지원)에 대한 클러스터를 생성합니다. 클러스터는 작업을 수락하고 예약하며 해당 작업을 처리하는 컴퓨팅 노드(EC2 인스턴스)도 시작합니다.

클러스터 생성

1. AWS PCS 콘솔을 열고 클러스터 생성을 선택합니다.
2. 클러스터 세부 정보 섹션에 다음 필드를 입력합니다.
   - 클러스터 이름 -를 입력합니다. `get-started`
   - 스케줄러 - Slurm 버전 25.05 선택
   - 컨트롤러 크기 - 스몰 선택
3. 네트워킹 섹션에서 다음 필드의 값을 선택합니다.
   - VPC - 라는 VPC를 선택합니다. `hpc-networking:Large-Scale-HPC`
   - 서브넷 - 이름이 로 시작하는 서브넷을 선택합니다. `hpc-networking:PrivateSubnetA`
   - 보안 그룹 - 라는 클러스터 보안 그룹을 선택합니다. `cluster-getstarted-sg`
4. 클러스터 생성을 선택합니다.

> **Note**
>
> 상태 필드에는 클러스터가 프로비저닝되는 동안 생성이 표시됩니다. 클러스터 생성에는 몇 분 정도 걸릴 수 있습니다.

Amazon Elastic File System에서 AWS PCS용 공유 스토리지 생성

Amazon Elastic File System(Amazon EFS)은 스토리지 용량 및 성능을 프로비저닝하거나 관리하지 않고도 파일 데이터를 공유할 수 있도록 서버리스의 완전 탄력적 파일 스토리지를 제공하는 AWS 서비스입니다. 자세한 내용은 Amazon Elastic File System 사용 설명서에서 Amazon Elastic File System이란?을 참조하세요.

AWS PCS 데모 클러스터는 EFS 파일 시스템을 사용하여 클러스터 노드 간에 공유 홈 디렉터리를 제공합니다. 클러스터와 동일한 VPC에 EFS 파일 시스템을 생성합니다.

Amazon EFS 파일 시스템 생성

1. Amazon EFS 콘솔로 이동합니다.
2. AWS PCS를 시도할 AWS 리전 위치와 동일하게 설정되어 있는지 확인합니다.
3. 파일 시스템 생성을 선택합니다.
4. 파일 시스템 생성 페이지에서 다음 파라미터를 설정합니다.
   - 이름에 `getstarted-efs`을 입력합니다.
   - Virtual Private Cloud(VPC)에서 라는 VPC를 선택합니다. `hpc-networking:Large-Scale-HPC`
   - 생성(Create)을 선택합니다. 그러면 파일 시스템 페이지로 돌아갑니다.
5. 파일 시스템의 파일 시스템 ID를 기록해 둡니다. `getstarted-efs` 나중에 이 정보가 필요합니다.

Amazon FSx for Lustre에서 AWS PCS용 공유 스토리지 생성

Amazon FSx for Lustre를 사용하면 인기 있는 고성능 Lustre 파일 시스템을 쉽고 비용 효율적으로 시작하고 실행할 수 있습니다. Lustre는 기계 학습, 고성능 컴퓨팅 (HPC), 비디오 처리, 금융 모델링 등 속도가 중요한 워크로드에 사용합니다. 자세한 내용은 Amazon FSx for Lustre 사용 설명서의 Amazon FSx for Lustre란 무엇입니까?를 참조하세요. FSx

AWS PCS 데모 클러스터는 FSx for Lustre 파일 시스템을 사용하여 클러스터 노드 간에 고성능 공유 디렉터리를 제공할 수 있습니다. 클러스터와 동일한 VPC에 FSx for Lustre 파일 시스템을 생성합니다.

FSx for Lustre 파일 시스템을 생성하려면

1. Amazon FSx 콘솔로 이동합니다.
2. 콘솔이 클러스터 AWS 리전 와 동일하게 사용하도록 설정되어 있는지 확인합니다.
3. 파일 시스템 생성을 선택합니다.
   - 파일 시스템 유형 선택에서 Amazon FSx for Lustre를 선택한 후 다음을 선택합니다.
4. 파일 시스템 세부 정보 지정 페이지에서 다음 파라미터를 설정합니다.
   - 파일 시스템 세부 정보에서
     - 이름에 `getstarted-fsx`을 입력합니다.
     - 배포 및 스토리지 유형에서 영구, SSD를 선택합니다.
     - 스토리지 단위당 처리량의 경우 125MB/s/TiB를 선택합니다.

        - 스토리지 용량에 1.2TiB를 입력합니다.
        - 메타데이터 구성에서 자동을 선택합니다.
        - 데이터 압축 유형에서 LZ4를 선택합니다.
    - 네트워크 및 보안에서
        - Virtual Private Cloud(VPC)에서 라는 VPC를 선택합니다. `hpc-networking:Large-Scale-HPC`
        - VPC 보안 그룹의 경우 라는 보안 그룹을 그대로 둡니다. `default`
        - 서브넷에서 이름이 로 시작하는 서브넷을 선택합니다. `hpc-networking:PrivateSubnetA`
    - 다른 옵션은 기본값으로 설정된 상태로 둡니다.
    - Next(다음)를 선택합니다.
5. 검토 및 생성 페이지에서 파일 시스템 생성을 선택합니다. 그러면 파일 시스템 페이지로 돌아갑니다.
6. 생성한 FSx for Lustre 파일 시스템의 세부 정보 페이지로 이동합니다.
7. 파일 시스템 ID와 탑재 이름을 기록해 둡니다. 나중에 이 정보가 필요합니다.

> **Note**
>
> 상태 필드에 파일 시스템이 프로비저닝되는 동안 생성이 표시됩니다. 파일 시스템 생성에는 몇 분 정도 걸릴 수 있습니다. 자습서의 나머지 부분을 진행하기 전에 완료될 때까지 기다립니다.

AWS PCS에서 컴퓨팅 노드 그룹 생성

컴퓨팅 노드 그룹은 AWS PCS가 시작하고 관리하는 컴퓨팅 노드(EC2 인스턴스)의 가상 컬렉션입니다. 컴퓨팅 노드 그룹을 정의할 때 EC2 인스턴스 유형, 최소 및 최대 인스턴스 수, 대상 VPC 서브넷, 기본 구매 옵션, 사용자 지정 시작 구성과 같은 일반적인 특성을 지정합니다. AWS PCS는 이러한 설정에 따라 컴퓨팅 노드 그룹에서 컴퓨팅 노드를 효율적으로 시작, 관리 및 종료합니다. 데모 클러스터는 컴퓨팅 노드 그룹을 사용하여 사용자 액세스를 위한 로그인 노드를 제공하고 별도의 컴퓨팅 노드 그룹을 사용하여 작업을 처리합니다. 다음 주제에서는 클러스터에서 이러한 컴퓨팅 노드 그룹을 설정하는 절차를 설명합니다.

주제

- AWS PCS용 인스턴스 프로파일 생성

- AWS PCS용 시작 템플릿 생성
- AWS PCS에서 로그인 노드에 대한 컴퓨팅 노드 그룹 생성
- AWS PCS에서 컴퓨팅 작업을 실행하기 위한 컴퓨팅 노드 그룹 생성

AWS PCS용 인스턴스 프로파일 생성

컴퓨팅 노드 그룹은 생성 시 인스턴스 프로파일이 필요합니다. AWS Management Console 을 사용하여 Amazon EC2 역할을 생성하는 경우, 콘솔이 자동으로 인스턴스 프로파일을 생성하여 해당 역할과 동일한 이름을 부여합니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 인스턴스 프로파일 사용을 참조하세요.

다음 절차에서는 AWS Management Console 를 사용하여 컴퓨팅 노드 그룹에 대한 인스턴스 프로파일도 생성하는 Amazon EC2에 대한 역할을 생성합니다.

역할 및 인스턴스 프로파일을 생성하려면

- IAM 콘솔로 이동합니다.
- 액세스 관리(Access management)에서 정책(Policies)을 선택합니다.
  - 정책 생성을 선택합니다.
  - 권한 지정의 정책 편집기에서 JSON을 선택합니다.
  - 텍스트 편집기의 내용을 다음과 같이 바꿉니다.

    JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "pcs:RegisterComputeNodeGroupInstance"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

  - 다음을 선택합니다.
  - 검토 및 생성에서 정책 이름에를 입력합니다`AWSPCS-getstarted-policy`.

    - 정책 생성을 선택합니다.
- 액세스 관리에서 역할을 선택합니다.
- 역할 생성을 선택합니다.
- 신뢰할 수 있는 엔터티 선택에서:
    - 신뢰할 수 있는 엔터티 유형에서 AWS 서비스를 선택합니다.
    - 사용 사례에서 EC2를 선택합니다.
        - 그런 다음 지정된 서비스의 사용 사례 선택에서 EC2를 선택합니다.
    - 다음을 선택합니다.
- 권한 추가에서:
    - 권한 정책에서 AWSPCS-getstarted-policy를 검색합니다.
    - AWSPCS-getstarted-policy 옆의 확인란을 선택하여 역할에 추가합니다.
    - 권한 정책에서 AmazonSSMManagedInstanceCore를 검색합니다.
    - AmazonSSMManagedInstanceCore 옆의 확인란을 선택하여 역할에 추가합니다.
    - 다음을 선택합니다.
- 이름, 검토 및 생성에서 다음을 수행합니다.
    - 역할 세부 정보에서:
        - [역할 이름(Role name)]에 `AWSPCS-getstarted-role`을 입력합니다.
    - 역할 생성을 선택합니다.

AWS PCS용 시작 템플릿 생성

컴퓨팅 노드 그룹을 생성할 때 AWS PCS가 시작하는 EC2 인스턴스를 구성하는 데 사용하는 EC2 시작 템플릿을 제공합니다. 여기에는 인스턴스가 시작될 때 실행되는 보안 그룹 및 스크립트와 같은 설정이 포함됩니다.

이 단계에서는 하나의 CloudFormation 템플릿을 사용하여 두 개의 EC2 시작 템플릿을 생성합니다. 템플릿 하나는 로그인 노드를 생성하는 데 사용되고 다른 하나는 컴퓨팅 노드를 생성하는 데 사용됩니다. 이들 간의 주요 차이점은 인바운드 SSH 액세스를 허용하도록 로그인 노드를 구성할 수 있다는 것입니다.

CloudFormation 템플릿 액세스

다음 URL을 사용하여 CloudFormation 템플릿을 다운로드한 다음 CloudFormation 콘솔에 템플릿을 업로드하여 새 CloudFormation 스택을 생성합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 콘솔 사용을 참조하세요.

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/getting_started/assets/pcs-lt-efs-fsxl.yaml
```

CloudFormation 템플릿을 사용하여 EC2 시작 템플릿 생성

다음 절차에 따라 CloudFormation 콘솔에서 CloudFormation 템플릿을 완료합니다.

- 스택 이름 제공에서 다음을 수행합니다.
  - 스택 이름에를 입력합니다`getstarted-lt`.
- 파라미터에서:
  - 보안에서
    - VpcSecurityGroupId에서 클러스터 VPC`default`에서 라는 보안 그룹을 선택합니다.
    - ClusterSecurityGroupId에서 라는 그룹을 선택합니다. `cluster-getstarted-sg`
    - SshSecurityGroupId에서 라는 이름의 그룹을 선택합니다. `inbound-ssh-getstarted-sg`
    - SshKeyName에서 원하는 SSH 키 페어를 선택합니다.
  - 파일 시스템에서
    - EfsFilesystemId에 자습서의 앞부분에서 생성한 EFS 파일 시스템의 파일 시스템 ID를 입력합니다.
    - FSxLustreFilesystemId에 자습서의 앞부분에서 생성한 FSx for Lustre 파일 시스템의 파일 시스템 ID를 입력합니다.
    - FSxLustreFilesystemMountName에 동일한 FSx for Lustre 파일 시스템의 탑재 이름을 입력합니다.
- 다음을 선택한 후 다음을 다시 선택합니다.
- 제출을 선택합니다.

CloudFormation 스택의 상태를 모니터링합니다. 시작 템플릿`CREATE_COMPLETE`에 도달하면 사용할 준비가 된 것입니다.

> **Note**
>
> CloudFormation 템플릿이 생성한 모든 리소스를 보려면 CloudFormation 콘솔을 엽니다. `getstarted-lt` 스택을 선택한 다음 리소스(Resources) 탭을 선택합니다.

AWS PCS에서 로그인 노드에 대한 컴퓨팅 노드 그룹 생성

컴퓨팅 노드 그룹은 AWS PCS가 시작하고 관리하는 컴퓨팅 노드(EC2 인스턴스)의 가상 컬렉션입니다. 컴퓨팅 노드 그룹을 정의할 때 EC2 인스턴스 유형, 최소 및 최대 인스턴스 수, 대상 VPC 서브넷, 기본 구매 옵션, 사용자 지정 시작 구성과 같은 일반적인 특성을 지정합니다. AWS PCS는 이러한 설정에 따라 컴퓨팅 노드 그룹에서 컴퓨팅 노드를 효율적으로 시작, 관리 및 종료합니다.

이 단계에서는 클러스터에 대한 대화형 액세스를 제공하는 정적 컴퓨팅 노드 그룹을 시작합니다. SSH 또는 Amazon EC2 Systems Manager(SSM)를 사용하여 로그인한 다음 셸 명령을 실행하고 Slurm 작업을 관리할 수 있습니다.

컴퓨팅 노드 그룹을 생성하려면

- AWS PCS 콘솔을 열고 클러스터로 이동합니다.
- 라는 클러스터를 선택합니다. `get-started`
- 컴퓨팅 노드 그룹으로 이동하여 생성을 선택합니다.
- 컴퓨팅 노드 그룹 설정 섹션에서 다음을 제공합니다.
  - 컴퓨팅 노드 그룹 이름 -를 입력합니다`login`.
- 컴퓨팅 구성에서 다음 값을 입력하거나 선택합니다.
  - EC2 시작 템플릿 - 이름이 인 시작 템플릿을 선택합니다. `login-getstarted-lt`
  - IAM 인스턴스 프로파일 - 라는 인스턴스 프로파일을 선택합니다. `AWSPCS-getstarted-role`
  - 서브넷 - 이름이 로 시작하는 서브넷을 선택합니다`hpc-networking:PublicSubnetA`.
  - 인스턴스 -를 선택합니다`c6i.xlarge`.
  - 조정 구성 - 최소 인스턴스 수에를 입력합니다`1`. 최대 인스턴스 수에 를 입력합니다. `1`
- 추가 설정에서 다음을 지정합니다.
  - AMI ID - 다음 형식의 이름을 가진 사용하려는 AMI를 선택합니다.

```
aws-pcs-sample_ami-amzn2-platform-slurm-version
```

샘플 AMIsAWS PCS에서 샘플 Amazon Machine Image(AMIs) 사용.

- 컴퓨팅 노드 그룹 생성을 선택합니다.

상태 필드에는 컴퓨팅 노드 그룹이 프로비저닝되는 동안 생성이 표시됩니다. 진행 중인 동안 자습서의 다음 단계로 진행할 수 있습니다.

AWS PCS에서 컴퓨팅 작업을 실행하기 위한 컴퓨팅 노드 그룹 생성

이 단계에서는 클러스터에 제출된 작업을 실행하기 위해 탄력적으로 확장되는 컴퓨팅 노드 그룹을 시작합니다.

컴퓨팅 노드 그룹을 생성하려면

- AWS PCS 콘솔을 열고 클러스터로 이동합니다.
- 라는 클러스터를 선택합니다. `get-started`
- 컴퓨팅 노드 그룹으로 이동하여 생성을 선택합니다.
- 컴퓨팅 노드 그룹 설정 섹션에서 다음을 제공합니다.
  - 컴퓨팅 노드 그룹 이름 -를 입력합니다`compute-1`.
- 컴퓨팅 구성에서 다음 값을 입력하거나 선택합니다.
  - EC2 시작 템플릿 - 이름이 인 시작 템플릿을 선택합니다. `compute-getstarted-lt`
  - IAM 인스턴스 프로파일 - 라는 인스턴스 프로파일을 선택합니다. `AWSPCS-getstarted-role`
  - 서브넷 - 이름이 로 시작하는 서브넷을 선택합니다`hpc-networking:PrivateSubnetA`.
  - 인스턴스 -를 선택합니다`c6i.xlarge`.
  - 조정 구성 - 최소 인스턴스 수에를 입력합니다`0`. 최대 인스턴스 수에 를 입력합니다. `4`
- 추가 설정에서 다음을 지정합니다.
  - AMI ID - 다음 형식의 이름을 가진 사용하려는 AMI를 선택합니다.

```
aws-pcs-sample_ami-amzn2-platform-slurm-version
```

샘플 AMIsAWS PCS에서 샘플 Amazon Machine Image(AMIs) 사용.

- 컴퓨팅 노드 그룹 생성을 선택합니다.

상태 필드에는 컴퓨팅 노드 그룹이 프로비저닝되는 동안 생성이 표시됩니다.

> **Important**
>
> 이 자습서의 다음 단계로 진행하기 전에 상태 필드에 활성이 표시될 때까지 기다립니다.

AWS PCS에서 작업을 관리하기 위한 대기열 생성

대기열에 작업을 제출하여 실행합니다. 작업은 AWS PCS가 컴퓨팅 노드 그룹에서 실행하도록 예약할 때까지 대기열에 남아 있습니다. 각 대기열은 처리를 수행하는 데 필요한 EC2 인스턴스를 제공하는 하나 이상의 컴퓨팅 노드 그룹과 연결됩니다.

이 단계에서는 컴퓨팅 노드 그룹을 사용하여 작업을 처리하는 대기열을 생성합니다.

대기열 생성

- AWS PCS 콘솔을 엽니다.
- 이름이 `get-started`인 클러스터를 선택합니다.
- 컴퓨팅 노드 그룹으로 이동하여 `compute-1` 그룹의 상태가 활성인지 확인합니다.

  > **Important**
  >
  > 다음 단계를 진행하기 전에 `compute-1` 그룹의 상태가 활성이어야 합니다.

- 대기열로 이동하여 대기열 생성을 선택합니다.
  - 대기열 구성 섹션에서 다음 값을 제공합니다.
    - 대기열 이름 - 다음을 입력합니다. `demo`
    - 컴퓨팅 노드 그룹 - 라는 컴퓨팅 노드 그룹을 선택합니다`compute-1`.
- 대기열 생성을 선택합니다.

상태 필드에는 대기열이 생성되는 동안 생성이 표시됩니다.

> **Important**
>
> 이 자습서의 다음 단계로 진행하기 전에 상태 필드에 활성이 표시될 때까지 기다립니다.

AWS PCS 클러스터에 연결

`login` 컴퓨팅 노드 그룹의 상태가 활성이 되면 생성한 EC2 인스턴스에 연결할 수 있습니다.

로그인 노드에 연결하려면

- AWS PCS 콘솔을 열고 클러스터로 이동합니다.
- 이름이 `get-started`인 클러스터를 선택합니다.
- 컴퓨팅 노드 그룹을 선택합니다.
- 라는 컴퓨팅 노드 그룹으로 이동합니다`login`.
- 컴퓨팅 노드 그룹 ID를 찾습니다.
- 다른 브라우저 창 또는 탭에서 Amazon EC2 콘솔을 엽니다.
  - 인스턴스를 선택합니다.
  - 다음 태그가 있는 EC2 인스턴스를 검색합니다. *`node-group-id`*를 이전 단계의 컴퓨팅 노드 그룹 ID 값으로 바꿉니다. 인스턴스는 1개여야 합니다.

    ```
    aws:pcs:compute-node-group-id=node-group-id
    ```

  - EC2 인스턴스에 연결합니다. 세션 관리자 또는 SSH를 사용할 수 있습니다.

    Session Manager

    - 인스턴스를 선택합니다.
    - 연결을 선택합니다.
    - 인스턴스에 연결에서 세션 관리자를 선택합니다.
    - 연결을 선택합니다.
    - 연결을 선택합니다. 브라우저에서 대화형 터미널이 시작됩니다.

    SSH

    - 인스턴스를 선택합니다.
    - 연결을 선택합니다.
    - 인스턴스에 연결에서 SSH 클라이언트를 선택합니다.
    - 콘솔에서 제공하는 지침을 따릅니다.

      > **Note**
      >
      > 인스턴스의 사용자 이름이이 **`ec2-user`** 아닙니다`root`.

AWS PCS에서 클러스터 환경 탐색

클러스터에 로그인한 후 셸 명령을 실행할 수 있습니다. 예를 들어 사용자를 변경하고, 공유 파일 시스템의 데이터를 작업하고, Slurm과 상호 작용할 수 있습니다.

사용자 변경

세션 관리자를 사용하여 클러스터에 로그인한 경우 로 연결될 수 있습니다`ssm-user`. 세션 관리자용으로 생성된 특수 사용자입니다. 다음 명령을 사용하여 Amazon Linux 2의 기본 사용자로 전환합니다. SSH를 사용하여 연결한 경우이 작업을 수행할 필요가 없습니다.

```
sudo su - ec2-user
```

공유 파일 시스템 작업

명령에서 EFS 파일 시스템과 FSx for Lustre 파일 시스템을 사용할 수 있는지 확인할 수 있습니다`df -h`. 클러스터의 출력은 다음과 유사해야 합니다.

```
[ec2-user@ip-10-3-6-103 ~]$ df -h
Filesystem                  Size  Used Avail Use% Mounted on
devtmpfs                    3.8G     0  3.8G   0% /dev
tmpfs                       3.9G     0  3.9G   0% /dev/shm
tmpfs                       3.9G  556K  3.9G   1% /run
tmpfs                       3.9G     0  3.9G   0% /sys/fs/cgroup
/dev/nvme0n1p1               24G   18G  6.6G  73% /
127.0.0.1:/                 8.0E     0  8.0E   0% /home
10.3.132.79@tcp:/zlshxbev  1.2T  7.5M  1.2T   1% /shared
tmpfs                       780M     0  780M   0% /run/user/0
tmpfs                       780M     0  780M   0% /run/user/1000
```

/home 파일 시스템은 127.0.0.1을 탑재하며 용량이 매우 큽니다. 자습서 앞부분에서 생성한 EFS 파일 시스템입니다. 여기에 작성된 모든 파일은 클러스터의 모든 노드에서 /home에서 사용할 수 있습니다.

/shared 파일 시스템은 프라이빗 IP를 탑재하며 용량은 1.2TB입니다. 자습서 앞부분에서 생성한 FSx for Lustre 파일 시스템입니다. 여기에 작성된 모든 파일은 클러스터의 모든 노드에서 /shared에서 사용할 수 있습니다.

Slurm과 상호 작용

주제

- 대기열 및 노드 나열
- 작업 표시

대기열 및 노드 나열

를 사용하여 대기열과 연결된 노드를 나열할 수 있습니다`sinfo`. 클러스터의 출력은 다음과 유사해야 합니다.

```
[ec2-user@ip-10-3-6-103 ~]$ sinfo
PARTITION AVAIL  TIMELIMIT  NODES  STATE NODELIST
demo         up   infinite      4  idle~ compute-1-[1-4]
[ec2-user@ip-10-3-6-103 ~]$
```

라는 파티션을 기록해 둡니다`demo`. 상태는 up 이고 노드는 최대 4개입니다. 노드 `compute-1` 그룹의 노드와 연결됩니다. 컴퓨팅 노드 그룹을 편집하고 최대 인스턴스 수를 8개로 늘리면 노드 수가 읽히8고 노드 목록에가 표시됩니다`compute-1-[1-8]`. 4개의 노드`test`로 라는 두 번째 컴퓨팅 노드 그룹을 생성하고 demo 대기열에 추가한 경우 해당 노드도 노드 목록에 표시됩니다.

작업 표시

를 사용하여 시스템의 모든 작업을 모든 상태로 나열할 수 있습니다`squeue`. 클러스터의 출력은 다음과 유사해야 합니다.

```
[ec2-user@ip-10-3-6-103 ~]$ squeue
JOBID PARTITION NAME USER ST TIME NODES NODELIST(REASON)
```

Slurm 작업이 보류 중이거나 실행 중인 경우 나중에 `squeue` 다시 실행해 보십시오.

AWS PCS에서 단일 노드 작업 실행

Slurm을 사용하여 작업을 실행하려면 작업 요구 사항을 지정하는 제출 스크립트를 준비하고 `sbatch` 명령을 사용하여 대기열에 제출합니다. 일반적으로 공유 디렉터리에서 수행되므로 로그인 및 컴퓨팅 노드에는 파일에 액세스할 수 있는 공통 공간이 있습니다.

클러스터의 로그인 노드에 연결하고 쉘 프롬프트에서 다음 명령을 실행합니다.

- 기본 사용자가 됩니다. 공유 디렉터리로 변경합니다.

```
sudo su - ec2-user
```

```
cd /shared
```

- 다음 명령을 사용하여 예제 작업 스크립트를 생성합니다.

```
cat << EOF > job.sh
#!/bin/bash
#SBATCH -J single
#SBATCH -o single.%j.out
#SBATCH -e single.%j.err

echo "This is job \${SLURM_JOB_NAME} [\${SLURM_JOB_ID}] running on \
${SLURMD_NODENAME}, submitted from \${SLURM_SUBMIT_HOST}" && sleep 60 && echo "Job
 complete"
EOF
```

- Slurm 스케줄러에 작업 스크립트를 제출합니다.

```
sbatch -p demo job.sh
```

- 작업이 제출되면 작업 ID가 숫자로 반환됩니다. 해당 ID를 사용하여 작업 상태를 확인합니다. 다음 명령의 *job-id*를에서 반환된 번호로 바꿉니다sbatch.

```
squeue --job job-id
```

Example

```
squeue --job 1
```

squeue 명령은 다음과 유사한 출력을 반환합니다.

```
JOBID PARTITION NAME USER     ST TIME NODES NODELIST(REASON)
1     demo      test ec2-user CF 0:47 1     compute-1
```

- R (실행 중) 상태에 도달할 때까지 작업 상태를 계속 확인합니다. 가 아무것도 반환squeue하지 않으면 작업이 완료됩니다.
- /shared 디렉터리의 내용을 검사합니다.

```
ls -alth /shared
```

명령 출력은 다음과 유사합니다.

```
-rw-rw-r- 1 ec2-user ec2-user 107 Mar 19 18:33 single.1.out
-rw-rw-r- 1 ec2-user ec2-user 0 Mar 19 18:32 single.1.err
-rw-rw-r- 1 ec2-user ec2-user 381 Mar 19 18:29 job.sh
```

`single.1.out` 및 라는 파일이 클러스터의 컴퓨팅 노드 중 하나에서 작성`single.1.err`되었습니다. 작업은 공유 디렉터리(`/shared`)에서 실행되었으므로 로그인 노드에서도 사용할 수 있습니다. 따라서이 클러스터에 대해 FSx for Lustre 파일 시스템을 구성했습니다.

- `single.1.out` 파일의 내용을 검사합니다.

```
cat /shared/single.1.out
```

출력은 다음과 유사합니다.

```
This is job test [1] running on compute-1, submitted from ip-10-3-13-181
Job complete
```

AWS PCS에서 Slurm을 사용하여 다중 노드 MPI 작업 실행

이 지침은 Slurm을 사용하여 AWS PCS에서 MPI(메시지 전달 인터페이스) 작업을 실행하는 방법을 보여줍니다.

로그인 노드의 셸 프롬프트에서 다음 명령을 실행합니다.

- 기본 사용자가 됩니다. 홈 디렉터리로 변경합니다.

```
sudo su - ec2-user
cd ~/
```

- C 프로그래밍 언어로 소스 코드를 생성합니다.

```
cat > hello.c << EOF
// * mpi-hello-world - https://www.mpitutorial.com
// Released under MIT License
//
// Copyright (c) 2014 MPI Tutorial.
//
// 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, subject to the following conditions:
// The above copyright notice and this permission notice shall be included in
// all copies or substantial portions of the Software.
//
// 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.

#include <mpi.h>
#include <stdio.h>
#include <stddef.h>

int main(int argc, char** argv) {
  // Initialize the MPI environment. The two arguments to MPI Init are not
  // currently used by MPI implementations, but are there in case future
  // implementations might need the arguments.
  MPI_Init(NULL, NULL);

  // Get the number of processes
  int world_size;
  MPI_Comm_size(MPI_COMM_WORLD, &world_size);

  // Get the rank of the process
  int world_rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &world_rank);

  // Get the name of the processor
  char processor_name[MPI_MAX_PROCESSOR_NAME];
  int name_len;
  MPI_Get_processor_name(processor_name, &name_len);

  // Print off a hello world message
  printf("Hello world from processor %s, rank %d out of %d processors\n",
         processor_name, world_rank, world_size);

  // Finalize the MPI environment. No more MPI calls can be made after this
  MPI_Finalize();
```

```
}
EOF
```

- OpenMPI 모듈을 로드합니다.

```
module load openmpi
```

- C 프로그램을 컴파일합니다.

```
mpicc -o hello hello.c
```

- Slurm 작업 제출 스크립트를 작성합니다.

```
cat > hello.sh << EOF
#!/bin/bash
#SBATCH -J multi
#SBATCH -o multi.out
#SBATCH -e multi.err
#SBATCH --exclusive
#SBATCH --nodes=4
#SBATCH --ntasks-per-node=1

srun $HOME/hello
EOF
```

- 공유 디렉터리로 변경합니다.

```
cd /shared
```

- 작업 스크립트를 제출합니다.

```
sbatch -p demo ~/hello.sh
```

- 작업이 완료될 때까지 squeue를 사용하여 모니터링합니다.
- 의 내용을 확인합니다multi.out.

```
cat multi.out
```

출력 결과는 다음과 비슷합니다. 각 순위에는 다른 노드에서 실행되었으므로 고유한 IP 주소가 있습니다.

```
Hello world from processor ip-10-3-133-204, rank 0 out of 4 processors
Hello world from processor ip-10-3-128-219, rank 2 out of 4 processors
Hello world from processor ip-10-3-141-26, rank 3 out of 4 processors
Hello world from processor ip-10-3-143-52, rank 1 out of 4 processor
```

AWS PCS에 대한 AWS 리소스 삭제

이 자습서를 위해 생성한 클러스터 및 노드 그룹을 완료한 후에는 생성한 리소스를 삭제해야 합니다.

> **Important**
>
> 에서 실행 중인 모든 리소스에 대한 청구 요금이 발생합니다. AWS 계정

이 자습서에서 생성한 AWS PCS 리소스를 삭제하려면

- AWS PCS 콘솔을 엽니다.
- get-started라는 클러스터로 이동합니다.
- 대기열 섹션으로 이동합니다.
- 데모라는 대기열을 선택합니다.
- Delete(삭제)를 선택합니다.

> **Important**
>
> 계속하기 전에 대기열이 삭제될 때까지 기다립니다.

- 컴퓨팅 노드 그룹 섹션으로 이동합니다.
- compute-1이라는 컴퓨팅 노드 그룹을 선택합니다.
- Delete(삭제)를 선택합니다.
- 로그인이라는 컴퓨팅 노드 그룹을 선택합니다.
- Delete(삭제)를 선택합니다.

> **Important**
>
> 계속하기 전에 두 컴퓨팅 노드 그룹이 모두 삭제될 때까지 기다립니다.

- 시작하기에 대한 클러스터 세부 정보 페이지에서 삭제를 선택합니다.

> **Important**
> 클러스터가 삭제될 때까지 기다렸다가 후속 단계를 진행합니다.

이 자습서에서 생성한 다른 AWS 리소스를 삭제하려면

- IAM 콘솔을 엽니다.
  - 역할을 선택합니다.
  - AWSPCS-getstarted-role이라는 역할을 선택한 다음 삭제를 선택합니다.
  - 역할이 삭제된 후 정책을 선택합니다.
  - AWSPCS-getstarted-policy라는 정책을 선택한 다음 삭제를 선택합니다.
- CloudFormation 콘솔을 엽니다.
  - getstarted-lt라는 스택을 선택합니다.
  - Delete(삭제)를 선택합니다.

> **Important**
> 계속하기 전에 스택이 삭제될 때까지 기다립니다.

- Amazon EFS 콘솔을 엽니다.
  - 파일 시스템을 선택합니다.
  - getstarted-efs라는 파일 시스템을 선택합니다.
  - Delete(삭제)를 선택합니다.

> **Important**
> 계속하기 전에 파일 시스템이 삭제될 때까지 기다립니다.

- Amazon FSx 콘솔을 엽니다.
  - 파일 시스템을 선택합니다.
  - getstarted-fsx라는 파일 시스템을 선택합니다.
  - Delete(삭제)를 선택합니다.

> **Important**
>
> 계속하기 전에 파일 시스템이 삭제될 때까지 기다립니다.

- CloudFormation 콘솔을 엽니다.
  - getstarted-sg라는 스택을 선택합니다.
  - Delete(삭제)를 선택합니다.
- CloudFormation 콘솔을 엽니다.
  - hpc-networking이라는 스택을 선택합니다.
  - Delete(삭제)를 선택합니다.

CloudFormation 및 AWS PCS 시작하기

AWS CloudFormation 를 사용하여 AWS PCS 클러스터를 생성할 수 있습니다.를 CloudFormation 사용하면 AWS 인프라 배포를 예측 가능하고 반복적으로 생성하고 프로비저닝할 수 있습니다. CloudFormation 를 사용하면 기본 AWS 인프라를 생성 및 구성 AWS 클라우드 하지 않고도에서 매우 안정적이고 확장 가능하며 비용 효율적인 애플리케이션을 빌드하기 위해 많은 AWS 서비스에서 리소스를 자동으로 프로비저닝할 수 있습니다.를 CloudFormation 사용하면 템플릿 파일을 사용하여 스택이라고 하는 리소스 모음을 단일 단위로 생성 및 삭제할 수 있습니다. 에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 란 무엇입니까 CloudFormation?를 CloudFormation참조하세요. 의 AWS PCS 리소스 유형에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS PCS 리소스 유형 참조를 CloudFormation참조하세요.

주제

- CloudFormation 를 사용하여 샘플 AWS PCS 클러스터 생성
- 를 사용하여 생성된 AWS PCS 클러스터에 연결 CloudFormation
- 에서 AWS PCS 클러스터 정리 CloudFormation
- AWS PCS용 CloudFormation 템플릿의 일부
- CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿

CloudFormation 를 사용하여 샘플 AWS PCS 클러스터 생성

다음 절차에서는의 CloudFormation 템플릿을 사용하여 샘플 AWS PCS 클러스터를 AWS Management Console 생성합니다. 에 대한 자세한 내용은 란 무엇입니까 CloudFormation?를 CloudFormation참조하십시오. AWS CloudFormation 사용 설명서의 . 의 AWS PCS 리소스 유형에 대한 자세한 내용은 AWS CloudFormation 사용 설명서의 AWS PCS 리소스 유형 참조를 CloudFormation참조하세요.

샘플 클러스터를 생성하려면

1. AWS 리전 를 선택하여에서 클러스터를 생성합니다(링크는 템플릿이 있는 CloudFormation 콘솔을 엽니다).

   - 미국 동부(버지니아 북부)(us-east-1)
   - 미국 동부(오하이오)(us-east-2)
   - 미국 서부(오레곤)(us-west-2)

- 아시아 태평양(뭄바이)(ap-south-1)
- 아시아 태평양(싱가포르)(ap-southeast-1)
- 아시아 태평양(시드니)(ap-southeast-2)
- 아시아 태평양(도쿄)(ap-northeast-1)
- 유럽(프랑크푸르트)(eu-central-1)
- 유럽(아일랜드)(eu-west-1)
- 유럽(런던)(eu-west-2)
- 유럽(파리)(eu-west-3)
- 유럽(밀라노)(eu-south-1)
- 유럽(스톡홀름)(eu-north-1)
- AWS GovCloud(미국 동부)(us-gov-east-1)
- AWS GovCloud(미국 서부)(us-gov-west-1)

2. 스택 이름 제공에 설명이 포함된 이름을 입력합니다. CloudFormation 스택의 이름입니다. 템플릿은이 값을 AWS PCS 클러스터의 이름으로 사용합니다.
3. 파라미터에서:
   a. SlurmVersion에서 클러스터에서 사용할 Slurm 버전을 선택합니다.
   b. NodeArchitecture에서 x86을 선택하여 x86_64 호환 인스턴스를 사용하는 클러스터를 배포하거나 Graviton을 선택하여 Arm64 인스턴스를 사용합니다.
   c. KeyName에서 SSH 키 페어를 선택하여 클러스터 로그인 노드에 액세스합니다. 선택한 키 페어에 대한 PEM 파일이 있는지 확인합니다.
   d. ClientIpCidr에 CIDR 형식의 IP 범위를 입력하여 로그인 노드에 대한 액세스를 제어합니다.

      > **Warning**
      > 의 기본값은 모든 IP 주소에서 액세스를 `0.0.0.0/0` 허용합니다.

   e. HpcRecipesS3Bucket 및 HpcRecipesBranch 값을 기본값으로 둡니다.
4. 기능 및 변환에서:
   a. 확인란을 선택하여가 IAM 리소스를 생성함을 CloudFormation 확인합니다.
   b. 확인란을 선택하여가 사용자 지정 이름으로 IAM 리소스를 생성함을 CloudFormation 확인합니다.

c. 확인란을 선택하여 새 스택CAPABILITY_AUTO_EXPAND을 확인합니다. 자세한 내용은 API 참조의 CreateStack을 참조하세요. AWS CloudFormation

5. 스택 생성을 선택합니다.
6. 스택의 상태를 모니터링합니다. 스택 상태가 이면 클러스터에 연결할 수 있습니다CREATE_COMPLETE.

를 사용하여 생성된 AWS PCS 클러스터에 연결 CloudFormation

CloudFormation 템플릿에서 AWS PCS 클러스터를 생성한 후 AWS PCS 콘솔(AWS Management Console)을 사용하여 클러스터를 관리할 수 있습니다. 클러스터의 로그인 노드 중 하나에 연결하여 클러스터를 관리하고, 작업을 실행하고, 데이터를 관리할 수도 있습니다. CloudFormation 스택은 클러스터에 연결하는 데 사용할 수 있는 링크를 제공합니다.

클러스터에 연결하려면

1. CloudFormation 콘솔을 엽니다
2. 생성한 스택을 선택합니다.
3. 스택의 출력 탭을 선택합니다.

   스택은 다음 링크를 제공합니다.

   - PcsConsoleUrl - 클러스터가 선택된 AWS 상태로 PCS 콘솔을 열려면이 링크를 선택합니다. 이를 사용하여 클러스터, 노드 그룹 및 대기열 구성을 탐색할 수 있습니다.
   - Ec2ConsoleUrl -이 링크를 선택하여 클러스터의 로그인 노드 그룹이 관리하는 인스턴스를 표시하도록 필터링된 Amazon EC2 콘솔을 엽니다.

     이 보기에서 인스턴스를 선택하고 연결을 선택할 수 있습니다. 샘플 클러스터의 인스턴스는 웹 브라우저에서 인바운드 SSH 및 AWS Systems Manager 연결을 지원합니다. 자세한 내용은 AWS PCS 클러스터에 연결 단원을 참조하십시오.

     로그인 인스턴스에 연결한 후의 자습서를 따를 수 있습니다AWS PCS에서 클러스터 환경 탐색.

에서 AWS PCS 클러스터 정리 CloudFormation

CloudFormation 를 사용하여 AWS PCS 클러스터를 생성한 경우 CloudFormation 콘솔을 열고 스택을 삭제하여 클러스터와 모든 관련 리소스를 삭제할 수 있습니다.

> **Important**
>
> 샘플 클러스터의 경우 클러스터에 추가 컴퓨팅 노드 그룹 또는 대기열을 생성한 경우(샘플 CloudFormation 템플릿이 생성한 `login` 및 `compute-1` 그룹 이상) CloudFormation 스택을 삭제 AWS CLI 하기 전에 AWS PCS 콘솔 또는를 사용하여 해당 리소스를 삭제해야 합니다. 자세한 내용은 AWS PCS에서 클러스터 삭제 단원을 참조하십시오.

AWS PCS용 CloudFormation 템플릿의 일부

CloudFormation 템플릿에는 각각 특정 목적에 맞는 섹션이 1개 이상 있습니다.는 템플릿에서 표준 형식, 구문 및 언어를 CloudFormation 정의합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 템플릿으로 작업을 참조하세요.

CloudFormation 템플릿은 사용자 지정이 가능하므로 형식이 다를 수 있습니다. CloudFormation 템플릿에서 AWS PCS 클러스터를 생성하는 데 필요한 부분을 이해하려면 제공된 샘플 템플릿을 검토하여 샘플 클러스터를 생성하는 것이 좋습니다. 이 주제에서는 해당 샘플 템플릿의 섹션을 간략하게 설명합니다.

> **Important**
>
> 이 주제의 코드 샘플이 완료되지 않았습니다. 줄임표(`[...]`)가 있으면 표시되지 않는 추가 코드가 있음을 나타냅니다. 전체 YAML 형식의 CloudFormation 템플릿을 다운로드하려면 섹션을 참조하세요CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿.

목차

헤더

```
AWSTemplateFormatVersion: '2010-09-09'
Transform: AWS::Serverless-2016-10-31
Description: AWS Parallel Computing Service "getting started" cluster
```

`AWSTemplateFormatVersion`는 템플릿이 준수하는 템플릿 형식 버전을 식별합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 템플릿 형식 버전 구문을 참조하세요.

`Transform`는 CloudFormation이 템플릿을 처리하는 데 사용하는 매크로를 지정합니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 템플릿 변환 섹션을 참조하세요. `AWS::Serverless-2016-10-31` 변환 CloudFormation 을 통해는 AWS Serverless Application Model (AWS SAM) 구문으로 작성된 템플릿을 처리할 수 있습니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 `AWS::Serverless` 변환을 참조하세요.

Metadata

```
### Stack metadata
Metadata:
  AWS::CloudFormation::Interface:
    ParameterGroups:
      - Label:
          default: PCS Cluster configuration
        Parameters:
          - SlurmVersion
          - ManagedAccounting
          - AccountingPolicyEnforcement
      - Label:
          default: PCS ComputeNodeGroups configuration
        Parameters:
          - NodeArchitecture
          - KeyName
          - ClientIpCidr
      - Label:
          default: HPC Recipes configuration
        Parameters:
          - HpcRecipesS3Bucket
          - HpcRecipesBranch
```

CloudFormation 템플릿의 `metadata` 섹션에서는 템플릿 자체에 대한 정보를 제공합니다. 샘플 템플릿은 AWS PCS를 사용하는 완전한 고성능 컴퓨팅(HPC) 클러스터를 생성합니다. 샘플 템플릿의

메타데이터 섹션은가 해당 스택을 CloudFormation 시작(프로비저닝)하는 방법을 제어하는 파라미터를 선언합니다. 아키텍처 선택(NodeArchitecture), Slurm 버전() 및 액세스 제어KeyName(및 SlurmVersion)를 제어하는 파라미터가 있습니다ClientIpCidr.

Parameters

이 Parameters 섹션에서는 템플릿의 사용자 지정 파라미터를 정의합니다.는 이러한 파라미터 정의를 CloudFormation 사용하여이 템플릿에서 스택을 시작할 때 상호 작용하는 양식을 구성하고 검증합니다.

```
Parameters:

  NodeArchitecture:
    Type: String
    Default: x86
    AllowedValues:
      - x86
      - Graviton
    Description: Processor architecture for the login and compute node instances

  SlurmVersion:
    Type: String
    Default: 25.05
    Description: Version of Slurm to use
    AllowedValues:
         - 24.11
         - 25.05

  ManagedAccounting:
    Type: String
    Default: 'disabled'
    AllowedValues:
      - 'enabled'
      - 'disabled'
    Description: Monitor cluster usage, manage access control, and enforce resource
 limits with Slurm accounting. Requires Slurm 24.11 or newer.

  AccountingPolicyEnforcement:
    Description: Specify which Slurm accounting policies to enforce
    Type: String
    Default: none
    AllowedValues:
      - none
```

```
      - 'associations,limits,safe'

  KeyName:
    Description: SSH keypair to log in to the head node
    Type: AWS::EC2::KeyPair::KeyName
    AllowedPattern: ".+"  # Required

  ClientIpCidr:
    Description: IP(s) allowed to access the login node over SSH. We recommend that
you restrict it with your own IP/subnet (x.x.x.x/32 for your own ip or x.x.x.x/24 for
range. Replace x.x.x.x with your own PUBLIC IP. You can get your public IP using tools
such as https://ifconfig.co/)
    Default: 127.0.0.1/32
    Type: String
    AllowedPattern: (\d{1,3})\.(\d{1,3})\.(\d{1,3})\.(\d{1,3})/(\d{1,2})
    ConstraintDescription: Value must be a valid IP or network range of the form
x.x.x.x/x.

  HpcRecipesS3Bucket:
    Type: String
    Default: aws-hpc-recipes
    Description: HPC Recipes for AWS S3 bucket
    AllowedValues:
         - aws-hpc-recipes
         - aws-hpc-recipes-dev
  HpcRecipesBranch:
    Type: String
    Default: main
    Description: HPC Recipes for AWS release branch
    AllowedPattern: '^(?!.*/\.git$)(?!.*/\.)(?!.*\\.\.)[a-zA-Z0-9-_\.]+$'
```

매핑

이 Mappings 섹션에서는 특정 조건 또는 종속성을 기반으로 값을 지정하는 키-값 페어를 정의합니다.

```
Mappings:

  Architecture:
    AmiArchParameter:
      Graviton: arm64
      x86: x86_64
    LoginNodeInstances:
      Graviton: c7g.xlarge
```

```
      x86: c6i.xlarge
    ComputeNodeInstances:
      Graviton: c7g.xlarge
      x86: c6i.xlarge
```

리소스

이 Resources 섹션에서는 스택의 일부로 프로비저닝하고 구성할 AWS 리소스를 선언합니다.

```
Resources:

  [...]
```

템플릿은 샘플 클러스터 인프라를 계층으로 프로비저닝합니다. VPC 구성Networking의 경우 로 시작합니다. 스토리지는 공유 스토리지용 및 FSxLStorage 고성능 스토리지EfsStorage용 이중 시스템에서 제공합니다. 코어 클러스터는를 통해 설정됩니다PCSCluster.

```
  Networking:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        ProvisionSubnetsC: "False"
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/net/hpc_large_scale/assets/main.yaml'

  EfsStorage:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        SubnetIds: !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
        SubnetCount: 1
        VpcId: !GetAtt [ Networking, Outputs.VPC ]
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/storage/efs_simple/assets/main.yaml'

  FSxLStorage:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        PerUnitStorageThroughput: 125
```

```
        SubnetId: !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
        VpcId: !GetAtt [ Networking, Outputs.VPC ]
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/storage/fsx_lustre/assets/persistent.yaml'

  [...]

  # Cluster
  PCSCluster:
    Type: AWS::PCS::Cluster
    Properties:
      Name: !Sub '${AWS::StackName}'
      Size: SMALL
      Scheduler:
        Type: SLURM
        Version: !Ref SlurmVersion
      Networking:
        SubnetIds:
          - !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
        SecurityGroupIds:
          - !GetAtt [ PCSSecurityGroup, Outputs.ClusterSecurityGroupId ]
```

컴퓨팅 리소스의 경우 템플릿은 PCSNodeGroupLogin 단일 로그인 노드와 최대 4개의 컴퓨팅 노드PCSNodeGroupCompute라는 두 개의 노드 그룹을 생성합니다. 이러한 노드 그룹은 권한 및 인스턴스 구성PCSInstanceProfile에 대해 PCSLaunchTemplate에서 지원됩니다.

```
  # Compute Node groups
  PCSInstanceProfile:
    Type: AWS::CloudFormation::Stack
    Properties:
      Parameters:
        # We have to regionalize this in case CX use the template in more than one
 region. Otherwise,
        # the create action will fail since instance-role-${AWS::StackName} already
 exists!
        RoleName: !Sub '${AWS::StackName}-${AWS::Region}'
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/pcs/getting_started/assets/pcs-iip-minimal.yaml'

  PCSLaunchTemplate:
    Type: AWS::CloudFormation::Stack
    Properties:
```

```
      Parameters:
        VpcDefaultSecurityGroupId: !GetAtt [ Networking, Outputs.SecurityGroup ]
        ClusterSecurityGroupId: !GetAtt [ PCSSecurityGroup,
 Outputs.ClusterSecurityGroupId ]
        SshSecurityGroupId: !GetAtt [ PCSSecurityGroup,
 Outputs.InboundSshSecurityGroupId ]
        EfsFilesystemSecurityGroupId: !GetAtt [ EfsStorage, Outputs.SecurityGroupId ]
        FSxLustreFilesystemSecurityGroupId: !GetAtt [ FSxLStorage,
 Outputs.FSxLustreSecurityGroupId ]
        SshKeyName: !Ref KeyName
        EfsFilesystemId: !GetAtt [ EfsStorage, Outputs.EFSFilesystemId ]
        FSxLustreFilesystemId: !GetAtt [ FSxLStorage, Outputs.FSxLustreFilesystemId ]
        FSxLustreFilesystemMountName: !GetAtt [ FSxLStorage,
 Outputs.FSxLustreMountName ]
      TemplateURL: !Sub 'https://${HpcRecipesS3Bucket}.s3.amazonaws.com/
${HpcRecipesBranch}/recipes/pcs/getting_started/assets/cfn-pcs-lt-efs-fsxl.yaml'

  # Compute Node groups - Login Nodes
  PCSNodeGroupLogin:
    Type: AWS::PCS::ComputeNodeGroup
    Properties:
      ClusterId: !GetAtt [PCSCluster, Id]
      Name: login
      ScalingConfiguration:
        MinInstanceCount: 1
        MaxInstanceCount: 1
      IamInstanceProfileArn: !GetAtt [ PCSInstanceProfile, Outputs.InstanceProfileArn ]
      CustomLaunchTemplate:
        TemplateId: !GetAtt [ PCSLaunchTemplate, Outputs.LoginLaunchTemplateId ]
        Version: 1
      SubnetIds:
        - !GetAtt [ Networking, Outputs.DefaultPublicSubnet ]
      AmiId: !GetAtt [PcsSampleAmi, AmiId]
      InstanceConfigs:
        - InstanceType: !FindInMap [ Architecture, LoginNodeInstances, !Ref
 NodeArchitecture ]

  # Compute Node groups - Compute Nodes
  PCSNodeGroupCompute:
    Type: AWS::PCS::ComputeNodeGroup
    Properties:
      ClusterId: !GetAtt [PCSCluster, Id]
      Name: compute-1
      ScalingConfiguration:
```

```
        MinInstanceCount: 0
        MaxInstanceCount: 4
      IamInstanceProfileArn: !GetAtt [ PCSInstanceProfile, Outputs.InstanceProfileArn ]
      CustomLaunchTemplate:
        TemplateId: !GetAtt [ PCSLaunchTemplate, Outputs.ComputeLaunchTemplateId ]
        Version: 1
      SubnetIds:
        - !GetAtt [ Networking, Outputs.DefaultPrivateSubnet ]
      AmiId: !GetAtt [PcsSampleAmi, AmiId]
      InstanceConfigs:
        - InstanceType: !FindInMap [ Architecture, ComputeNodeInstances, !Ref
NodeArchitecture ]
```

작업 예약은를 통해 처리됩니다PCSQueueCompute.

```
PCSQueueCompute:
  Type: AWS::PCS::Queue
  Properties:
    ClusterId: !GetAtt [PCSCluster, Id]
    Name: demo
    ComputeNodeGroupConfigurations:
      - ComputeNodeGroupId: !GetAtt [PCSNodeGroupCompute, Id]
```

AMI 선택은 PcsAMILookupFn Lambda 함수 및 관련 리소스를 통해 자동으로 수행됩니다.

```
PcsAMILookupRole:
  Type: AWS::IAM::Role
  [...]

PcsAMILookupFn:
  Type: AWS::Lambda::Function
  Properties:
    Runtime: python3.12
    Handler: index.handler
    Role: !GetAtt PcsAMILookupRole.Arn
    Code:
      [...]
    Timeout: 30
    MemorySize: 128
```

```
  # Example of using the custom resource to look up an AMI
  PcsSampleAmi:
    Type: Custom::AMILookup
    Properties:
      ServiceToken: !GetAtt PcsAMILookupFn.Arn
      OperatingSystem: 'amzn2'
      Architecture: !FindInMap [ Architecture, AmiArchParameter, !Ref
 NodeArchitecture ]
      SlurmVersion: !Ref SlurmVersion
```

출력

템플릿은 `ClusterId`, 및를 통해 클러스터 식별 `PcsConsoleUrl`및 관리 URLs`Ec2ConsoleUrl`.

```
Outputs:
  ClusterId:
    Description: The Id of the PCS cluster
    Value: !GetAtt [ PCSCluster, Id ]

  PcsConsoleUrl:
    Description: URL to access the cluster in the PCS console
    Value: !Sub
      - https://${ConsoleDomain}/pcs/home?region=${AWS::Region}#/clusters/${ClusterId}
      - { ConsoleDomain: !If [ GovCloud, 'console.amazonaws-us-gov.com', !If [ China,
 'console.amazonaws.cn', !Sub '${AWS::Region}.console.aws.amazon.com']],
          ClusterId: !GetAtt [ PCSCluster, Id ]
        }
    Export:
      Name: !Sub ${AWS::StackName}-PcsConsoleUrl

  Ec2ConsoleUrl:
    Description: URL to access instance(s) in the login node group via Session Manager
    Value: !Sub
      - https://${ConsoleDomain}/ec2/home?region=
${AWS::Region}#Instances:instanceState=running;tag:aws:pcs:compute-node-group-id=
${NodeGroupLoginId}
      - { ConsoleDomain: !If [ GovCloud, 'console.amazonaws-us-gov.com', !If [ China,
 'console.amazonaws.cn', !Sub '${AWS::Region}.console.aws.amazon.com']],
          NodeGroupLoginId: !GetAtt [ PCSNodeGroupLogin, Id ]
        }
    Export:
      Name: !Sub ${AWS::StackName}-Ec2ConsoleUrl
```

CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿

AWS 리전 이름	AWS 리전	소스 보기	스택 시작
미국 동부(버지니아 북부)	us-east-1	YAML 다운로드	
미국 동부(오하이오)	us-east-2	YAML 다운로드	
미국 서부(오리건)	us-west-2	YAML 다운로드	
아시아 태평양(뭄바이)	ap-south-1	YAML 다운로드	
아시아 태평양(싱가포르)	ap-southeast-1	YAML 다운로드	
아시아 태평양(시드니)	ap-southeast-2	YAML 다운로드	
아시아 태평양(도쿄)	ap-northeast-1	YAML 다운로드	
유럽(프랑크푸르트)	eu-central-1	YAML 다운로드	
유럽(아일랜드)	eu-west-1	YAML 다운로드	
Europe (London)	eu-west-2	YAML 다운로드	
유럽(파리)	eu-west-3	YAML 다운로드	
Europe (Milan)	eu-south-1	YAML 다운로드	
유럽(스톡홀름)	eu-north-1	YAML 다운로드	

AWS 리전 이름	AWS 리전	소스 보기	스택 시작
AWS GovCloud(미국 동부)	us-gov-east-1	YAML 다운로드	
AWS GovCloud(미국 서부)	us-gov-west-1	YAML 다운로드	

AWS PCS 클러스터

AWS PCS 클러스터는 다음 구성 요소로 구성됩니다.

- Slurm 제어 데몬()과 같은 HPC 시스템 스케줄러 소프트웨어의 관리형 인스턴스입니다`slurmctld`.
- HPC 시스템 스케줄러와 통합되어 Amazon EC2 인스턴스를 프로비저닝하고 관리하는 구성 요소입니다.
- HPC 시스템 스케줄러와 통합되어 로그 및 지표를 Amazon CloudWatch로 전송하는 구성 요소입니다.

이러한 구성 요소는에서 관리하는 계정에서 실행됩니다 AWS. 이들은 함께 작동하여 고객 계정의 Amazon EC2 인스턴스를 관리합니다. AWS PCS는 Amazon VPC 서브넷에 탄력적 네트워크 인터페이스를 프로비저닝하여 스케줄러 소프트웨어에서 Amazon EC2 인스턴스로의 연결을 제공합니다(예: 해당 인스턴스에서 배치 작업을 예약하고 사용자가 스케줄러 명령을 실행하여 해당 작업을 나열하고 관리할 수 있도록 지원).

주제

AWS PCS에서 클러스터 생성

이 주제에서는 사용 가능한 옵션에 대한 개요를 제공하고 AWS 병렬 컴퓨팅 서비스(AWS PCS)에서 클러스터를 생성할 때 고려해야 할 사항에 대해 설명합니다. AWS PCS 클러스터를 처음 생성하는 경우를 따르는 것이 좋습니다AWS 병렬 컴퓨팅 서비스 시작하기. 이 자습서는 가능한 모든 사용 가능한 옵션과 시스템 아키텍처로 확장하지 않고도 작동하는 HPC 시스템을 생성하는 데 도움이 될 수 있습니다.

> **Note**
>
> 클러스터를 생성한 후 인프라를 다시 빌드하지 않고도 여러 구성 설정을 수정할 수 있습니다. 자세한 내용은 AWS PCS에서 클러스터 업데이트 단원을 참조하십시오.

> **Note**
>
> 고급 예약 정책 및 리소스 관리를 구현하도록 사용자 지정 Slurm 설정을 구성할 수 있습니다. 자세한 내용은 AWS PCS에서 사용자 지정 Slurm 설정 구성 단원을 참조하십시오.

사전 조건

- AWS PCS 네트워킹 요구 사항을 충족하는 기존 VPC 및 서브넷입니다. 프로덕션 용도로 클러스터를 배포하기 전에 VPC 및 서브넷 요구 사항을 충분히 이해하는 것이 좋습니다. VPC 및 서브넷을 생성하려면 섹션을 참조하세요AWS PCS 클러스터에 대한 VPC 생성.
- AWS PCS 리소스를 생성하고 관리할 수 있는 권한이 있는 IAM 보안 주체입니다. 자세한 내용은 AWS 병렬 컴퓨팅 서비스를 위한 ID 및 액세스 관리 단원을 참조하십시오.

AWS PCS 클러스터 생성

AWS Management Console 또는를 사용하여 클러스터 AWS CLI 를 생성할 수 있습니다.

AWS Management Console

클러스터 생성

1. https://console.aws.amazon.com/pcs/home#/clusters AWS PCS 콘솔을 열고 클러스터 생성을 선택합니다.
2. 클러스터 설정 섹션에서 다음 필드를 입력합니다.
   - 클러스터 이름 - 클러스터의 이름입니다. 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하며 40자를 초과할 수 없습니다. 이름은 내에서 고유해야 하며 클러스터를 생성할 AWS 리전 AWS 계정 때 고유해야 합니다.
   - 스케줄러 - 스케줄러 및 버전을 선택합니다. 자세한 내용은 AWS PCS의 Slurm 버전 단원을 참조하십시오.
   - 컨트롤러 크기 - 컨트롤러의 크기를 선택합니다. 이렇게 하면 AWS PCS 클러스터에서 관리할 수 있는 동시 작업 및 컴퓨팅 노드 수가 결정됩니다. 클러스터가 생성될 때만 컨트롤러 크기를 설정할 수 있습니다. 크기 조정에 대한 자세한 내용은 섹션을 참조하세요AWS PCS의 클러스터 크기.
3. 네트워킹 섹션에서 다음 필드의 값을 선택합니다.

- 네트워크 유형 - 클러스터의 IP 주소 유형을 선택합니다. 클러스터는 IPv4 또는 IPv6를 사용할 수 있지만 둘 다 사용할 수는 없습니다. VPC와 서브넷은 동일한 네트워크 주소 유형을 사용해야 합니다. 각 서브넷에 사용하는 IP 주소 블록에는 사용 가능한 주소가 1개 이상 있어야 합니다. AWS 는 각 서브넷의 일부 주소를 예약합니다. 자세한 내용을 알아보려면 Amazon VPC 사용 설명서의 서브넷 CIDR 블록을 참조하세요.
- VPC - AWS PCS 요구 사항을 충족하는 기존 VPC를 선택합니다. 자세한 내용은 AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항 단원을 참조하십시오. 클러스터를 생성한 후에는 VPC를 변경할 수 없습니다. 나열된 VPCs 없는 경우 먼저 VPC를 생성해야 합니다.
- 서브넷 - 선택한 VPC에서 사용 가능한 모든 서브넷이 나열됩니다. AWS PCS 서브넷 요구 사항을 충족하는 서브넷을 선택합니다. 자세한 내용은 AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항 단원을 참조하십시오. 스케줄러 엔드포인트가 퍼블릭 인터넷에 노출되지 않도록 프라이빗 서브넷을 선택하는 것이 좋습니다.
- 보안 그룹 - AWS PCS가 클러스터에 대해 생성하는 네트워크 인터페이스와 연결할 보안 그룹(들)을 지정합니다. 클러스터와 컴퓨팅 노드 간의 통신을 허용하는 보안 그룹을 하나 이상 선택해야 합니다. 보안 그룹 빠른 생성을 선택하여 AWS PCS가 선택한 VPC에서 필요한 구성으로 보안 그룹을 생성하도록 하거나 기존 보안 그룹을 선택할 수 있습니다. 자세한 내용은 보안 그룹 요구 사항 및 고려 사항 단원을 참조하십시오.

4. (선택 사항) Slurm 회계 구성 섹션에서 Slurm 회계를 활성화하고 회계 파라미터를 설정할 수 있습니다. 자세한 내용은 AWS PCS의 Slurm 회계 단원을 참조하십시오.
5. (선택 사항) Slurm 구성 섹션에서 파라미터 이름과 값 페어를 추가하여 추가 Slurm 설정을 구성할 수 있습니다. 지원되는 파라미터의 전체 목록은 섹션을 참조하세요AWS PCS 클러스터에 대한 사용자 지정 Slurm 설정.
6. (선택 사항) 태그에서 AWS PCS 클러스터에 태그를 추가합니다.
7. 클러스터 생성을 선택합니다. AWS PCS가 클러스터를 생성하는 `Creating` 동안 상태 필드가 표시됩니다. 이 프로세스는 몇 분 정도 걸릴 수 있습니다.

> **Important**
>
> 당 `Creating` AWS 리전 상태에는 클러스터가 1개만 있을 수 있습니다 AWS 계정. AWS 클러스터를 생성하려고 할 때 `Creating` 상태에 이미 클러스터가 있는 경우 PCS는 오류를 반환합니다.

AWS CLI

클러스터 생성

1. 다음 명령을 사용하여 클러스터를 생성합니다. 명령을 실행하기 전에 다음과 같은 바꾸기를 합니다.

   - *region*을와 같이 클러스터를 생성 AWS 리전 하려는의 ID로 바꿉니다us-east-1.
   - *my-cluster*를 클러스터의 이름으로 바꿉니다. 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하며 40자를 초과할 수 없습니다. 이름은 클러스터를 생성하는 AWS 리전 및 AWS 계정 내에서 고유해야 합니다.
   - *25.05*를 지원되는 Slurm 버전으로 바꿉니다.

     > **Note**
     >
     > AWS PCS는 현재 Slurm 25.05 및 24.11을 지원합니다.

   - *SMALL*을 지원되는 클러스터 크기로 바꿉니다. 이렇게 하면 AWS PCS 클러스터에서 관리할 수 있는 동시 작업 및 컴퓨팅 노드 수가 결정됩니다. 클러스터가 생성될 때만 설정할 수 있습니다. 크기 조정에 대한 자세한 내용은 섹션을 참조하세요AWS PCS의 클러스터 크기.
   - 의 값을 자신의 값으로 바꿉subnetIds니다. 스케줄러 엔드포인트가 퍼블릭 인터넷에 노출되지 않도록 프라이빗 서브넷을 선택하는 것이 좋습니다.
   - AWS PCSsecurityGroupIds가 클러스터에 대해 생성하는 네트워크 인터페이스와 연결할를 지정합니다. 보안 그룹은 클러스터와 동일한 VPC에 있어야 합니다. 클러스터와 컴퓨팅 노드 간의 통신을 허용하는 보안 그룹을 하나 이상 선택해야 합니다. 자세한 내용은 보안 그룹 요구 사항 및 고려 사항 단원을 참조하십시오.

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM,version=25.05 \
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1
```

   - IPv6를 사용하려면 --networking 구성에 networkType=IPV6를 추가합니다.

```
  --networking networkType=IPV6,subnetIds=subnet-
ExampleId1,securityGroupIds=sg-ExampleId1
```

- 선택적으로 Slurm 동작을 사용자 지정하고 Slurm 구성 옵션을 지정하는 `--slurm-configration` 옵션을 추가할 수 있습니다. 다음 예제에서는 축소 유휴 시간을 60분(3600초)으로 설정하고, Slurm 회계를 활성화하고, `slurm.conf` 설정을의 값으로 지정합니다`slurmCustomSettings`. 자세한 내용은 AWS PCS의 Slurm 회계 단원을 참조하십시오.

> **Note**
>
> 회계는 Slurm 24.11 이상에서 지원됩니다.

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM,version=25.05 \
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1
    --slurm-configuration
 scaleDownIdleTimeInSeconds=3600,accounting='{mode=STANDARD}',slurmCustomSettings='[{p
```

2. 클러스터를 프로비저닝하는 데 몇 분 정도 걸릴 수 있습니다. 다음 명령을 사용하여 클러스터의 상태를 쿼리할 수 있습니다. 클러스터의 상태 필드가가 될 때까지 대기열 또는 컴퓨팅 노드 그룹 생성을 진행하지 마십시오`ACTIVE`.

```
aws pcs get-cluster --region region --cluster-identifier my-cluster
```

> **Important**
>
> 당 `Creating` AWS 리전 상태에는 클러스터가 1개만 있을 수 있습니다 AWS 계정. AWS 클러스터를 생성하려고 할 때 `Creating` 상태에 이미 클러스터가 있는 경우 PCS는 오류를 반환합니다.

클러스터에 권장되는 다음 단계

- 컴퓨팅 노드 그룹을 추가합니다.
- 대기열을 추가합니다.
- 로깅을 활성화합니다.

AWS PCS에서 클러스터 업데이트

AWS PCS를 사용하면 UpdateCluster API 또는 콘솔을 통해 생성된 클러스터 구성을 업데이트할 수 있습니다. 인프라를 재구축하지 않고도 클러스터 설정을 수정하여 운영 오버헤드를 줄이고 중단을 최소화할 수 있습니다.

클러스터 업데이트의 이점

AWS PCS 클러스터를 업데이트하면 서비스 중단 없이 HPC 인프라를 새로운 요구 사항에 맞게 조정할 수 있습니다. 클러스터를 다시 빌드하는 데 필요한 시간 이상이 아닌 몇 분 정도 걸립니다. 이 기능은 가동 중지 시간을 최소화해야 하는 프로덕션 환경과 워크로드 패턴 변화에 따라 클러스터 설정을 조정해야 하는 팀에 중요합니다.

지원되는 구성 변경 사항

세 가지 주요 설정 범주를 수정할 수 있습니다.

- 회계 구성 - 관리형 회계를 활성화 또는 비활성화하고 보존 설정을 구성합니다.
- 축소 동작 - `scaleDownIdleTime` 파라미터를 조정하여 동적 인스턴스가 유휴 상태로 유지되는 시간을 제어한 후 AWS PCS가 자동으로 종료합니다.
- Slurm 사용자 지정 설정 - Prolog, Epilog 및 SelectTypeParameters.

제한 사항

클러스터 생성 후에는 특정 구성을 수정할 수 없습니다. 다음이 포함됩니다.

- 보안 그룹 구성
- VPC 서브넷 선택
- 클러스터 크기
- Slurm 버전
- 클러스터 이름

이러한 설정은 클러스터 아키텍처의 기본이며 수정하려면 새 클러스터를 생성해야 합니다.

클러스터 업데이트를 위한 사전 조건

클러스터를 업데이트하기 전에 다음 조건이 충족되는지 확인합니다.

- 클러스터는 ACTIVE, UPDATE_FAILED또는 SUSPENDED 상태여야 합니다.
- 연결된 모든 리소스(대기열, 컴퓨팅 노드 그룹)는 ACTIVE 상태여야 합니다.
- UpdateCluster 작업에 대한 적절한 IAM 권한이 있어야 합니다.
- 진행 중인 다른 업데이트 작업은 없습니다.

프로세스 및 작업 영향 업데이트

업데이트 작업 중에 클러스터 컨트롤러에 잠시 연결할 수 없게 되더라도 컴퓨팅 노드는 기존 작업을 계속 실행합니다. 그러나 시스템은이 기간 동안 새 작업 제출을 수락하거나 일정을 결정할 수 없습니다.

콘솔 및 API 인터페이스를 통해 클러스터 업데이트를 모니터링할 수 있습니다. 클러스터는 업데이트 중에 다음 상태를 통해 전환됩니다.

- UPDATING - 업데이트 진행 중
- ACTIVE - 업데이트가 성공적으로 완료되었습니다.
- UPDATE_FAILED - 업데이트에 오류가 발생했습니다.

업데이트 중 결제

업데이트 작업 중에 AWS 도 PCS 클러스터에 대한 표준 시간당 요금이 계속 부과됩니다. 회계를 비활성화하도록 클러스터를 업데이트하면 클러스터가 UPDATING 상태가 되는 즉시 회계 기능에 대한 청구가 중지됩니다. 회계를 활성화하면 클러스터가 업데이트를 성공적으로 완료하고 ACTIVE 상태로 돌아갈 때까지 결제가 시작되지 않습니다.

주제

- AWS PCS 클러스터 업데이트
- AWS PCS에서 클러스터 업데이트에 대해 자주 묻는 질문
- AWS PCS 클러스터 업데이트 문제 해결

AWS PCS 클러스터 업데이트

다음 단계에 따라 클러스터에서 스케줄러 설정, 회계 구성 및 Slurm 사용자 지정 설정을 수정합니다. 자세한 내용은 AWS PCS 클러스터에 대한 사용자 지정 Slurm 설정 단원을 참조하십시오.

사전 조건

- 클러스터는 `ACTIVE`, `UPDATE_FAILED`또는 `SUSPENDED` 상태여야 합니다.
- 연결된 모든 리소스(대기열, 컴퓨팅 노드 그룹)는 `ACTIVE` 상태여야 합니다.
- 진행 중인 다른 업데이트 작업은 없습니다.

절차

AWS Management Console

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 업데이트할 클러스터를 선택합니다.
4. 편집을 선택합니다.
5. 클러스터 편집 페이지에서 원하는 설정을 수정합니다.
   - 스케줄러 구성에서 스케일 다운 유휴 시간을 업데이트하여 자동 종료 전에 동적 인스턴스가 유휴 상태로 유지되는 기간을 제어합니다.
   - 필요에 따라 Prolog, Epilog 및 Select 유형 파라미터 설정을 수정합니다.
   - 관리형 회계의 보존 기간을 활성화, 비활성화 또는 구성합니다.
   - 추가 스케줄러 설정에서 Slurm 사용자 지정 설정을 추가, 편집 또는 제거합니다. 지원되는 파라미터에 대한 자세한 내용은 섹션을 참조하세요AWS PCS 클러스터에 대한 사용자 지정 Slurm 설정.

   > **Note**
   >
   > 편집할 수 없는 필드는 읽기 전용으로 표시되며 현재 값을 표시합니다.

6. 업데이트를 선택하여 변경 사항을 제출합니다.
7. 프로세스 중에 "업데이트"로 표시되는 클러스터 상태를 모니터링합니다. 업데이트가 성공적으로 완료되면 상태가 변경됩니다.

AWS CLI

1. 터미널 또는 명령 프롬프트를 엽니다.

2. 다음 명령을 사용하여 클러스터 상태를 확인합니다.

```
aws pcs get-cluster --cluster-identifier my-cluster
```

3. 다음 예제 중 하나를 사용하여 업데이트 요청을 제출합니다.

   - 관리형 회계를 활성화하려면:

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration 'accounting={mode=STANDARD}'
```

   - Slurm Prolog 설정을 업데이트하려면:

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'SlurmCustomSettings=[{parameterName=Prolog,parameterValue="/path/to/
prolog.sh"}]'
```

   - 축소 유휴 시간을 업데이트하려면:

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration 'scaleDownIdleTimeInSeconds=300'
```

4. 클러스터 상태를 확인하여 업데이트 진행 상황을 모니터링합니다.

```
aws pcs get-cluster --cluster-identifier my-cluster
```

업데이트 요청이 성공하면 명령은 모든 변경 사항과 함께 클러스터 객체를 반환합니다. 클러스터 상태는 `ACTIVE` 완료되면에서 `UPDATING`로 변경됩니다.

AWS PCS에서 클러스터 업데이트에 대해 자주 묻는 질문

AWS PCS에서 클러스터 구성 업데이트에 대한 일반적인 질문에 대한 답변을 얻습니다.

어떤 설정을 수정할 수 있나요?

회계 구성(관리형 회계 활성화/비활성화), 축소 동작(scaleDownIdleTime 파라미터) 및 클러스터 수준에서 적용되는 지원되는 Slurm 사용자 지정 설정을 수정할 수 있습니다. 보안 그룹, VPC 서브넷, 클러스터 크기, Slurm 버전 또는 클러스터 이름은 수정할 수 없습니다.

여러 업데이트를 대기열에 넣을 수 있나요?

아니요. 다른 업데이트를 제출하기 전에 클러스터가 ACTIVE 상태로 돌아갈 때까지 기다려야 합니다. 연결된 모든 리소스(대기열, 컴퓨팅 노드 그룹)도 ACTIVE 상태여야 합니다.

클러스터 업데이트 작업을 취소할 수 있나요?

아니요, 진행 중인 클러스터 업데이트 작업은 취소할 수 없습니다.

클러스터가 업데이트되는 동안 작업을 제출할 수 있나요?

클러스터 업데이트 중에는 작업을 제출하지 않는 것이 좋습니다. 업데이트 프로세스 중에 Slurm 컨트롤러를 사용하지 못할 수 있습니다.

클러스터 업데이트 중에 작업이 계속 실행되나요?

예. 업데이트 프로세스 중에 클러스터 컨트롤러에 잠시 연결할 수 없게 되더라도 실행 중인 작업은 컴퓨팅 노드에서 계속 실행됩니다. 그러나 컨트롤러를 다시 사용할 수 있을 때까지 작업 상태가 업데이트되지 않을 수 있습니다.

업데이트 중에 결제는 어떻게 영향을 받나요?

표준 시간당 요금은 업데이트 작업 중에 계속됩니다. 회계를 비활성화하면 클러스터가 UPDATING 상태가 되면 결제가 중지됩니다. 회계를 활성화하면 클러스터가 성공적으로 ACTIVE 상태로 돌아가면 결제가 시작됩니다.

AWS PCS 클러스터 업데이트 문제 해결

이 주제는 클러스터 구성을 업데이트할 때 발생할 수 있는 일반적인 문제를 식별하고 해결하는 데 도움이 됩니다.

계정 구성 오류와 함께 업데이트 실패

일반적인 원인

클러스터가 UPDATE_FAILED 상태로 전환되고 오류 메시지는 회계 구성 문제를 나타냅니다. 이는 일반적으로 회계 구성이 현재 Slurm 버전과 호환되지 않거나 잘못된 설정을 포함하는 경우에 발생합니다.

해결 방법

회계 설정에서 클러스터의 Slurm 버전과의 호환성을 검토하고 유효한 구성 파라미터와 함께 수정된 업데이트 요청을 제출합니다.

사용자 지정 설정 오류와 함께 업데이트 실패

일반적인 원인

클러스터가 UPDATE_FAILED 상태로 전환되고 오류 메시지는 Slurm 사용자 지정 설정 문제를 나타냅니다. 이는 잘못된 Slurm 파라미터 값 또는 지원되지 않는 파라미터 조합을 제공할 때 발생합니다.

해결 방법

지원되는 파라미터와 비교하여 Slurm 사용자 지정 설정을 검증하고 유효한 파라미터 값 및 조합으로 수정된 업데이트 요청을 제출합니다.

업데이트 요청을 제출할 수 없음

일반적인 원인

콘솔에서 업데이트 버튼이 비활성화되었거나 API가 400 수준 오류를 반환합니다. 이는 클러스터가 적절한 상태가 아니거나, 연결된 리소스가 활성 상태가 아니거나, 구성에 검증 실패가 있을 때 발생합니다.

해결 방법

클러스터 및 연결된 모든 리소스가 ACTIVE 상태에 도달할 때까지 기다린 다음 업데이트 요청을 다시 제출하기 전에 구성에서 검증 오류를 검토합니다.

유효성 검사 오류

일반적인 원인

명령은 400레벨 HTTP 오류와 설명 메시지와 함께 즉시 반환됩니다. 이는 잘못된 클러스터 상태, 리소스 상태 또는 구성 파라미터로 인해 발생합니다.

해결 방법

응답에 언급된 특정 검증 오류를 해결하고 업데이트 작업을 다시 시도합니다.

AWS PCS에서 클러스터 삭제

이 주제에서는 AWS PCS 클러스터를 삭제하는 방법에 대한 개요를 제공합니다.

AWS PCS 클러스터 삭제 시 고려 사항

- 클러스터와 연결된 모든 대기열을 삭제해야 클러스터를 삭제할 수 있습니다. 자세한 내용은 AWS PCS에서 대기열 삭제 단원을 참조하십시오.
- 클러스터와 연결된 모든 컴퓨팅 노드 그룹을 삭제해야 클러스터를 삭제할 수 있습니다. 자세한 내용은 AWS PCS에서 컴퓨팅 노드 그룹 삭제 단원을 참조하십시오.

클러스터 삭제

AWS Management Console 또는를 사용하여 클러스터 AWS CLI 를 삭제할 수 있습니다.

AWS Management Console

클러스터 삭제

1. AWS PCS 콘솔을 엽니다.
2. 삭제할 클러스터를 선택합니다.
3. 삭제를 선택합니다.
4. 클러스터 상태 필드에가 표시됩니다`Deleting`. 완료되는 데 몇 분 정도 걸릴 수 있습니다.

AWS CLI

클러스터 삭제

1. 다음 명령을 사용하여 클러스터를 삭제하고 다음과 같이 바꿉니다.
   - *region-code*를 클러스터가 있는 로 바꿉 AWS 리전 니다.
   - *my-cluster*를 클러스터의 이름 또는 ID로 바꿉니다.

```
aws pcs delete-cluster --region region-code --cluster-identifier my-cluster
```

2. 클러스터를 삭제하는 데 몇 분 정도 걸릴 수 있습니다. 다음 명령을 사용하여 클러스터의 상태를 확인할 수 있습니다.

```
aws pcs get-cluster --region region-code --cluster-identifier my-cluster
```

AWS PCS의 클러스터 크기

AWS PCS는 패치 적용, 노드 프로비저닝 및 업데이트와 같은 주요 작업을 자동화하는 동시에 가용성과 보안이 뛰어난 클러스터를 제공합니다.

클러스터를 생성할 때 다음 두 가지 요인에 따라 클러스터의 크기를 선택합니다.

- 관리할 컴퓨팅 노드 수
- 지정된 시간에 컨트롤러가 추적하는 작업 수

> **Note**
>
> 작업 수에는 실행 중, 보류 중 및 최근에 완료된 작업이 포함됩니다. 완료된 작업은 제거되기 전에 컨트롤러에 의해 짧은 기간 동안 추적됩니다. 작업 처리량이 높은 동안 총 추적 작업 수가 관찰한 활성 작업 수를 초과할 수 있습니다.

> **Important**
>
> 클러스터를 생성한 후에는 클러스터 크기를 변경할 수 없습니다. 크기를 변경해야 하는 경우 새 클러스터를 생성해야 합니다.

Slurm 클러스터 크기	관리형 인스턴스 수	컨트롤러가 추적하는 작업 수
작은	최대 32개	최대 256
중간	최대 512	최대 8192
대형	최대 2048	최대 16384

예제

- 클러스터에 최대 24개의 관리형 인스턴스가 있고 최대 100개의 작업을 실행하는 경우 소형을 선택합니다.
- 클러스터에 최대 24개의 관리형 인스턴스가 있고 최대 1,000개의 작업을 실행하는 경우 중간을 선택합니다.

- 클러스터에 최대 1,000개의 관리형 인스턴스가 있고 최대 100개의 작업을 실행하는 경우 대규모를 선택합니다.
- 클러스터에 최대 1,000개의 관리형 인스턴스가 있고 최대 10,000개의 작업을 실행하는 경우 대규모를 선택합니다.

AWS PCS에서 클러스터 보안 암호 작업

클러스터를 생성하는 과정에서 AWS PCS는 클러스터의 작업 스케줄러에 연결하는 데 필요한 클러스터 보안 암호를 생성합니다. 또한 조정 이벤트에 대한 응답으로 시작할 인스턴스 세트를 정의하는 AWS PCS 컴퓨팅 노드 그룹을 생성합니다. AWS PCS는 클러스터 보안 암호로 해당 컴퓨팅 노드 그룹에서 시작한 인스턴스를 구성하여 작업 스케줄러에 연결할 수 있도록 합니다. Slurm 클라이언트를 수동으로 구성하려는 경우가 있습니다. 영구 로그인 노드를 빌드하거나 작업 관리 기능을 사용하여 워크플로 관리자를 설정하는 것이 그 예입니다.

AWS PCS는 클러스터 보안 암호를 접두사가 인 관리형 보안 암호로 저장합니다 AWS Secrets Manager. `pcs!` 보안 암호 비용은 AWS PCS 사용 요금에 포함됩니다. 를 통해 클러스터 보안 암호를 교체 AWS Secrets Manager 하여 보안 규정 준수를 유지하고 잠재적 보안 침해를 해결할 수 있습니다.

주제

- AWS Secrets Manager 를 사용하여 클러스터 보안 암호 찾기
- AWS PCS를 사용하여 클러스터 보안 암호 찾기
- Slurm 클러스터 보안 암호 가져오기
- AWS PCS에서 클러스터 보안 암호 교체

AWS Secrets Manager 를 사용하여 클러스터 보안 암호 찾기

AWS Management Console

1. Secrets Manager 콘솔로 이동합니다.
2. 보안 암호를 선택한 다음 `pcs!` 접두사를 검색합니다.

> **Note**
>
> AWS PCS 클러스터 보안 암호에는가 AWS PCS 클러스터 IDpcs!slurm-secret-*cluster-idcluster-id*인 형식의 이름이 있습니다.

AWS CLI

각 AWS PCS 클러스터 보안 암호에도 태그가 지정됩니다`aws:pcs:`*`cluster-id`*. 다음 명령을 사용하여 클러스터의 보안 암호 ID를 가져올 수 있습니다. 명령을 실행하기 전에 다음과 같이 대체합니다.

- 를 *`region`*로 바꾸 AWS 리전 어와 같은에서 클러스터를 생성합니다`us-east-1`.
- 를 클러스터 보안 암호를 찾을 AWS PCS 클러스터의 ID*`cluster-id`*로 바꿉니다.

```
aws secretsmanager list-secrets \
    --region region  \
    --filters Key=tag-key,Values=aws:pcs:cluster-id \
            Key=tag-value,Values=cluster-id
```

AWS PCS를 사용하여 클러스터 보안 암호 찾기

AWS CLI 를 사용하여 AWS PCS 클러스터 보안 암호의 ARN을 찾을 수 있습니다. 다음 명령을 입력하여 다음과 같이 대체합니다.

- 를 *`region`*로 바꾸 AWS 리전 어와 같은에서 클러스터를 생성합니다`us-east-1`.
- 를 클러스터의 이름 또는 식별자*`my-cluster`*로 바꿉니다.

```
aws pcs get-cluster --region region --cluster-identifier my-cluster
```

다음 예제 출력은 `get-cluster` 명령에서 가져온 것입니다. `secretArn` 및를 `secretVersion` 함께 사용하여 보안 암호를 가져올 수 있습니다.

```
{
    "cluster": {
        "name": "get-started",
        "id": "pcs_123456abcd",
        "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_123456abcd",
        "status": "ACTIVE",
        "createdAt": "2024-12-17T21:03:52+00:00",
        "modifiedAt": "2024-12-17T21:03:52+00:00",
        "scheduler": {
            "type": "SLURM",
            "version": "25.05"
```

```
        },
        "size": "SMALL",
        "slurmConfiguration": {
            "authKey": {
                "secretArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:pcs!
slurm-secret-pcs_123456abcd-a12ABC",
                "secretVersion": "ef232370-d3e7-434c-9a87-ec35c1987f75"
            }
        },
        "networking": {
            "subnetIds": [
                "subnet-0123456789abcdef0"
            ],
            "securityGroupIds": [
                "sg-0123456789abcdef0"
            ]
        },
        "endpoints": [
            {
                "type": "SLURMCTLD",
                "privateIpAddress": "10.3.149.220",
                "port": "6817"
            }
        ]
    }
}
```

Slurm 클러스터 보안 암호 가져오기

Secrets Manager를 사용하여 Slurm 클러스터 보안 암호의 현재 base64 인코딩 버전을 가져올 수 있습니다. 다음 예제에서는를 사용합니다 AWS CLI. 명령을 실행하기 전에 다음과 같이 대체합니다.

- 를 *region*로 바꾸 AWS 리전 어와 같은에서 클러스터를 생성합니다`us-east-1`.
- *secret-arn*를 AWS PCS 클러스터`secretArn`의 로 바꿉니다.

```
aws secretsmanager get-secret-value \
    --region region \
    --secret-id 'secret-arn' \
    --version-stage AWSCURRENT \
    --query 'SecretString' \
    --output text
```

Slurm 클러스터 보안 암호를 사용하는 방법에 대한 자세한 내용은 섹션을 참조하세요독립 실행형 인스턴스를 AWS PCS 로그인 노드로 사용.

권한

IAM 보안 주체를 사용하여 Slurm 클러스터 보안 암호를 가져옵니다. IAM 보안 주체는 보안 암호를 읽을 수 있는 권한이 있어야 합니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 역할 용어 및 개념을 참조하세요.

다음 샘플 IAM 정책은 예제 클러스터 보안 암호에 대한 액세스를 허용합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSecretValueRetrievalAndVersionListing",
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue",
                "secretsmanager:ListSecretVersionIds"
            ],
            "Resource": "arn:aws:secretsmanager:us-east-1:012345678901:secret:pcs!
slurm-secret-s3431v9rx2-FN7tJF"
        }
    ]
}
```

AWS PCS에서 클러스터 보안 암호 교체

AWS Secrets Manager 관리형 교체를 사용하여 AWS PCS에서 클러스터 보안 암호를 교체합니다. 정기적인 보안 암호 교체는 HPC 환경에서 강력한 보안 태세를 유지하기 위한 보안 모범 사례입니다. 이 기능을 사용하면 정기적인 자격 증명 교체를 요구하는 HIPAA 및 FedRAMP를 비롯한 업계 규정 준수 표준을 충족할 수 있습니다.

클러스터 보안 암호는 클러스터에 조인하는 컴퓨팅 노드 인증과 Slurm REST API 인증을 위한 JWT 키로 사용되는 이중 용도입니다. 교체하면 두 측면이 동시에 영향을 받습니다.

클러스터 보안 암호 교체 작동 방식

보안 암호 교체 중에 클러스터 안정성을 유지하기 위해 수동으로 준비합니다.

1. 준비 - 모든 컴퓨팅 노드 그룹을 0 용량으로 확장하고 실행 중인 작업이 없는지 확인합니다.

2. 교체 - Secrets Manager 콘솔 또는 API를 통해 교체 시작
3. 모니터링 - CloudTrail 이벤트를 통한 진행 상황 추적
4. 복구 - 컴퓨팅 노드 그룹을 원하는 용량으로 다시 확장

교체 중에 클러스터는 ACTIVE 상태를 유지하고 결제는 정상적으로 계속됩니다. 이 프로세스는 일반적으로 몇 분 정도 걸립니다.

요구 사항 및 제한 사항

클러스터 보안 암호를 교체하기 전에 다음 요구 사항을 완료합니다.

- 클러스터는 ACTIVE 또는 UPDATE_FAILED 상태여야 합니다.
- IAM 역할에 secretsmanager:RotateSecret 권한이 있어야 합니다.
- 모든 컴퓨팅 노드 그룹을 0 용량으로 확장해야 합니다.
- 교체 전에 모든 작업 중지

제한:

- 각 교체에 필요한 수동 준비
- 기존 JWT 토큰이 무효화되어 다시 발급해야 함
- BYO 로그인 노드는 교체 후 수동 보안 암호 업데이트가 필요합니다.

항목

- AWS PCS에서 클러스터 보안 암호 교체
- AWS PCS의 클러스터 보안 암호 교체에 대해 자주 묻는 질문
- AWS PCS에서 클러스터 보안 암호 교체 문제 해결

AWS PCS에서 클러스터 보안 암호 교체

클러스터 보안 암호를 교체하여 보안 요구 사항을 준수하고 잠재적 침해를 해결합니다. 이 프로세스를 수행하려면 클러스터를 유지 관리 모드로 전환해야 합니다.

사전 조건

- secretsmanager:RotateSecret 권한이 있는 IAM 역할

- ACTIVE 또는 UPDATE_FAILED 상태의 클러스터

절차

1. 클러스터 사용자에게 예정된 유지 관리 기간을 알립니다.
2. 모든 컴퓨팅 노드 그룹을 0 용량으로 조정하여 클러스터를 유지 관리 모드로 전환합니다.
   a. UpdateComputeNodeGroup API를 사용하여 모든 컴퓨팅 노드 그룹에 대해 minInstanceCount와 maxInstanceCount를 모두 0으로 설정합니다.
   b. 모든 노드가 중지될 때까지 기다립니다.
   c. 선택 사항: 정상적인 작업 처리를 위해 용량을 종료하기 전에 Slurm 명령을 사용하여 스케줄러 대기열을 드레이닝합니다.
3. Secrets Manager를 통해 교체를 시작합니다.
   - 콘솔 메서드:
     - Secrets Manager로 이동하여 클러스터 보안 암호를 선택하고 보안 암호 교체를 선택합니다.
   - API 메서드:
     - Secrets Manager `rotate-secret` API를 사용합니다.
4. 교체 진행 상황을 모니터링합니다.
   a. CloudTrail 이벤트를 통해 진행 상황을 추적합니다.
   b. Secrets Manager 콘솔 또는 `secretsmanager:describeSecret` API를 `lastRotatedDate` 통해 확인합니다.
   c. `RotationSucceeded` 또는 `RotationFailed` CloudTrail 이벤트를 기다립니다.
5. 교체에 성공하면 클러스터 용량을 복원합니다.
   a. UpdateComputeNodeGroup API를 사용하여 노드 그룹을 원하는 최소/최대 용량으로 재설정합니다.
   b. AWS PCS 관리형 로그인 노드의 경우: 추가 작업이 필요하지 않습니다.
   c. BYO 로그인 노드의 경우:
      i. 로그인 노드에 연결합니다.
      ii. Secrets Manager의 새 보안 암호`/etc/slurm/slurm.key`로 업데이트합니다.
      iii. Slurm Auth 및 Cred Kiosk Daemon(sackd)을 다시 시작합니다.

AWS PCS의 클러스터 보안 암호 교체에 대해 자주 묻는 질문

AWS PCS에서 클러스터 보안 암호 교체에 대한 일반적인 질문에 대한 답변을 찾습니다.

클러스터 보안 암호란 무엇입니까?

클러스터 보안 암호는 Slurm 컨트롤러와 AWS PCS 컴퓨팅 노드 간의 보안 통신을 지원하는 보안 자격 증명입니다. 또한 Slurm REST API 인증을 위한 JSON 웹 토큰(JWT) 키 역할을 합니다.

클러스터 보안 암호와 JWT 키의 차이점은 무엇인가요?

AWS PCS에서 클러스터 보안 암호와 JWT 키는 서로 다른 목적을 제공하는 동일한 리소스입니다. 클러스터 보안 암호는 Slurm 내부 통신을 인증하는 반면 JWT 키는 REST API 인증을 위한 토큰에 서명합니다. 교체하면 두 측면이 동시에 영향을 받습니다.

교체에는 시간이 얼마나 걸리나요?

교체 프로세스는 일반적으로 몇 분 정도 걸립니다. 클러스터는 ACTIVE 상태로 유지되며 교체 중에 결제가 정상적으로 계속됩니다.

자동 교체를 예약할 수 있나요?

Secrets Manager에서 예약된 교체를 활성화할 수 있습니다. 그러나 초기 릴리스에서는 각 교체 전에 수동 준비(노드 그룹을 0으로 크기 조정)가 필요합니다.

교체 후에도 기존 JWT 토큰이 계속 작동하나요?

아니요. 교체 후 기존 JWT 토큰이 무효화됩니다. REST API 클라이언트에 대한 새 토큰을 발행합니다.

클러스터 보안 암호는 어디에서 찾을 수 있나요?

클러스터 보안 암호는 Secrets Manager 콘솔 또는 AWS PCS 콘솔을 통해 찾을 수 있습니다. 자세한 지침은 AWS Secrets Manager 를 사용하여 클러스터 보안 암호 찾기 및 단원을 참조하십시오AWS PCS를 사용하여 클러스터 보안 암호 찾기.

교체 시 노드 그룹을 0으로 조정해야 하는 이유는 무엇입니까?

교체 시 보안 암호 업데이트 프로세스 중에 클러스터 안정성을 보장하기 위해 실행 중인 인스턴스가 필요하지 않습니다. 이렇게 하면 이전 보안 암호와 새 보안 암호 간의 인증 충돌을 방지할 수 있습니다.

이 기능은 어떤 규정 준수 요구 사항을 지원하나요?

이 기능을 사용하면 AWS PCS가 보안 제어의 일부로 정기적인 자격 증명 교체를 요구하는 HIPAA 및 FedRAMP를 비롯한 업계 규정 준수 표준을 충족할 수 있습니다.

AWS PCS에서 클러스터 보안 암호 교체 문제 해결

환경이 제대로 준비되지 않으면 클러스터 보안 암호 교체가 실패합니다. 가장 일반적인 원인은 클러스터의 활성 인스턴스입니다. 실패를 방지하려면:

1. 모든 노드 그룹을 0 용량으로 설정합니다.
2. 노드가 중지될 때까지 기다립니다.
3. 클러스터가 `CREATE_FAILED`, , `DELETE_FAILED`, `RESUMING` `SUSPENDING`또는 상태가 아닌지 확인합니다`SUSPENDED`.

교체에 실패하는 경우:

- RotationFailed CloudTrail 이벤트가 나타납니다.
- 클러스터 보안 암호는 변경되지 않습니다.
- 자세한 내용은 CloudTrail에서 RotationFailed 이벤트를 확인하세요.
- 성공적인 교체를 위한 모든 준비 단계 완료

AWS PCS 컴퓨팅 노드 그룹

AWS PCS 컴퓨팅 노드 그룹은 노드(Amazon EC2 인스턴스)의 논리적 모음입니다. 이러한 노드는 컴퓨팅 작업을 실행하는 데 사용할 수 있을 뿐만 아니라 HPC 시스템에 대한 대화형 쉘 기반 액세스를 제공하는 데도 사용할 수 있습니다. 컴퓨팅 노드 그룹은 사용할 Amazon EC2 인스턴스 유형, 실행할 인스턴스 수, 스팟 인스턴스 또는 온디맨드 인스턴스 사용 여부, 사용할 서브넷 및 보안 그룹, 시작할 때 각 인스턴스를 구성하는 방법 등 노드를 생성하는 규칙으로 구성됩니다. 이러한 규칙이 업데이트되면 AWS PCS는 컴퓨팅 노드 그룹과 연결된 리소스를 일치하도록 업데이트합니다.

주제

AWS PCS에서 컴퓨팅 노드 그룹 생성

이 주제에서는 사용 가능한 옵션에 대한 개요를 제공하고 AWS 병렬 컴퓨팅 서비스(AWS PCS)에서 컴퓨팅 노드 그룹을 생성할 때 고려해야 할 사항을 설명합니다. AWS PCS에서 컴퓨팅 노드 그룹을 처음 생성하는 경우의 자습서를 따르는 것이 좋습니다AWS 병렬 컴퓨팅 서비스 시작하기. 이 자습서는 가능한 모든 사용 가능한 옵션과 시스템 아키텍처로 확장하지 않고도 작동하는 HPC 시스템을 생성하는 데 도움이 될 수 있습니다.

> **Note**
>
> 컴퓨팅 노드 그룹에서 사용자 지정 Slurm 설정을 구성하여 리소스 사용률 및 노드 수준 동작을 제어할 수 있습니다. 자세한 내용은 AWS PCS에서 사용자 지정 Slurm 설정 구성 단원을 참조하십시오.

> **Important**
>
> AWS 현재는 IPv6 전용 네트워크에서 PCS를 사용하는 경우에도 로컬 노드 통신에 대해 IPv4를 지원하는 커널이 AWS 필요합니다. IPv6-only 자세한 내용은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs) 단원을 참조하십시오.

사전 조건

- 에서 원하는 수의 EC2 인스턴스를 시작하기에 충분한 서비스 할당량입니다 AWS 리전. AWS Management Console를 사용하여 서비스 할당량 증가를 확인하고 요청할 수 있습니다.
- AWS PCS 네트워킹 요구 사항을 충족하는 기존 VPC 및 서브넷(들)입니다. 프로덕션용으로 클러스터를 배포하기 전에 이러한 요구 사항을 철저히 이해하는 것이 좋습니다. 자세한 내용은 AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항 단원을 참조하십시오. 또한 CloudFormation 템플릿을 사용하여 VPC 및 subnets를 생성할 수 있습니다.는 CloudFormation 템플릿에 대한 HPC 레시피를 AWS 제공합니다. 자세한 내용은 GitHub의 aws-hpc-recipes를 참조하세요.
- AWS PCS `RegisterComputeNodeGroupInstance` API 작업을 호출하고 노드 그룹 인스턴스에 필요한 다른 AWS 리소스에 액세스할 수 있는 권한이 있는 IAM 인스턴스 프로파일입니다. 자세한 내용은 AWS 병렬 컴퓨팅 서비스를 위한 IAM 인스턴스 프로파일 단원을 참조하십시오.
- 노드 그룹 인스턴스의 시작 템플릿입니다. 자세한 내용은 AWS PCS에서 Amazon EC2 시작 템플릿 사용 단원을 참조하십시오.
- Amazon EC2 스팟 인스턴스를 사용하는 컴퓨팅 노드 그룹을 생성하려면에 AWSServiceRoleForEC2Spot 서비스 연결 역할이 있어야 합니다 AWS 계정. 자세한 내용은 AWS PCS에 대한 Amazon EC2 스팟 역할 단원을 참조하십시오.

AWS PCS에서 컴퓨팅 노드 그룹 생성

AWS Management Console 또는를 사용하여 컴퓨팅 노드 그룹을 생성할 수 있습니다 AWS CLI.

AWS Management Console

콘솔을 사용하여 컴퓨팅 노드 그룹을 생성하려면

1. AWS PCS 콘솔을 엽니다.
2. 컴퓨팅 노드 그룹을 생성할 클러스터를 선택합니다. 컴퓨팅 노드 그룹으로 이동하여 생성을 선택합니다.

3. 컴퓨팅 노드 그룹 설정 섹션에서 노드 그룹의 이름을 입력합니다. 이름에는 대소문자를 구분하는 영숫자와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하며 25자를 초과할 수 없습니다. 이름은 클러스터 내에서 고유해야 합니다.
4. 컴퓨팅 구성에서 다음 값을 입력하거나 선택합니다.

   a. EC2 시작 템플릿 -이 노드 그룹에 사용할 사용자 지정 시작 템플릿을 선택합니다. 시작 템플릿을 사용하여 서브넷, 보안 그룹, 모니터링 구성 및 인스턴스 수준 스토리지와 같은 네트워크 설정을 사용자 지정할 수 있습니다. 시작 템플릿이 준비되지 않은 경우 섹션을 AWS PCS에서 Amazon EC2 시작 템플릿 사용 참조하여 템플릿을 생성하는 방법을 알아봅니다.

   > **Important**
   >
   > AWS PCS는 각 컴퓨팅 노드 그룹에 대한 관리형 시작 템플릿을 생성합니다. 이름은 입니다`pcs-identifier-do-not-delete`. 컴퓨팅 노드 그룹을 생성하거나 업데이트할 때 이러한 항목을 선택하지 마십시오. 그렇지 않으면 노드 그룹이 올바르게 작동하지 않습니다.

   b. EC2 시작 템플릿 버전 - 사용자 지정 시작 템플릿의 버전을 선택해야 합니다. 나중에 버전을 변경하는 경우 시작 템플릿의 변경 사항을 감지하도록 컴퓨팅 노드 그룹을 업데이트해야 합니다. 자세한 내용은 AWS PCS 컴퓨팅 노드 그룹 업데이트 단원을 참조하십시오.

   c. AMI ID - 시작 템플릿에 AMI ID가 포함되어 있지 않거나 시작 템플릿의 값을 재정의하려면 여기에 AMI ID를 입력합니다. 노드 그룹에 사용되는 AMI는 AWS PCS와 호환되어야 합니다. 에서 제공하는 샘플 AMI를 선택할 수도 있습니다 AWS. 이 주제에 대한 자세한 내용은 단원을 참조하십시오AWS PCS용 Amazon Machine Image(AMIs).

   d. IAM 인스턴스 프로파일 - 노드 그룹의 인스턴스 프로파일을 선택합니다. 인스턴스 프로파일은 인스턴스에 리소스 및 서비스에 안전하게 액세스할 AWS 수 있는 권한을 부여합니다. 준비된 프로필이 없는 경우 기본 프로필 생성을 선택하여 최소 정책으로 AWS PCS가 자동으로 프로필을 생성하도록 하거나 섹션을 참조하세요AWS 병렬 컴퓨팅 서비스를 위한 IAM 인스턴스 프로파일.

   e. 서브넷 - AWS PCS 클러스터가 배포된 VPC에서 하나 이상의 서브넷을 선택합니다. 여러 서브넷을 선택하면 노드 간에 EFA 통신을 사용할 수 없으며 다른 서브넷의 노드 간 통신으로 인해 지연 시간이 늘어날 수 있습니다. 여기서 지정하는 서브넷이 EC2 시작 템플릿에서 정의한 서브넷과 일치하는지 확인합니다.

   f. 인스턴스 - 노드 그룹에서 조정 요청을 이행할 인스턴스 유형을 하나 이상 선택합니다. 모든 인스턴스 유형은 동일한 프로세서 아키텍처(x86_64 또는 arm64)와 vCPUs 가져야 합

니다. 인스턴스에 GPUs가 있는 경우 모든 인스턴스 유형에 동일한 수의 GPUs가 있어야 합니다.

g. 조정 구성 - 노드 그룹의 최소 및 최대 인스턴스 수를 지정합니다. 고정된 수의 노드가 실행되는 정적 구성 또는 최대 수의 노드가 실행될 수 있는 동적 구성을 정의할 수 있습니다. 정적 구성의 경우 최소값과 최대값을 0보다 큰 동일한 숫자로 설정합니다. 동적 구성의 경우 최소 인스턴스를 0으로 설정하고 최대 인스턴스를 0보다 큰 수로 설정합니다. AWS PCS는 정적 인스턴스와 동적 인스턴스가 혼합된 컴퓨팅 노드 그룹을 지원하지 않습니다.

5. (선택 사항) 추가 설정에서 다음을 지정합니다.

   a. 구매 옵션 - 온디맨드 인스턴스, 스팟 인스턴스 또는 기존 용량 블록을 선택합니다. 또한 온디맨드 용량 예약(ODCR)을 사용하려는 경우 온디맨드를 선택합니다. 자세한 내용은 AWS PCSODCRs 사용 단원을 참조하십시오. ML 예약에 기존 Amazon EC2 용량 블록을 사용하려면 용량 블록을 선택합니다. 자세한 내용은 AWS PCS에서 ML에 Amazon EC2 용량 블록 사용 단원을 참조하십시오.

   b. 할당 전략 - 스팟 구매 옵션을 선택한 경우 노드 그룹에서 인스턴스를 시작할 때 스팟 용량 풀을 선택하는 방법을 지정할 수 있습니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 스팟 인스턴스에 대한 할당 전략을 참조하세요. 온디맨드 구매 옵션을 선택한 경우에는이 옵션이 적용되지 않습니다.

6. (선택 사항) Slurm 사용자 지정 설정 섹션에서 파라미터 이름과 값 페어를 추가하여 추가 Slurm 설정을 구성할 수 있습니다. 지원되는 파라미터의 전체 목록은 섹션을 참조하세요AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정.

7. (선택 사항) 태그에서 컴퓨팅 노드 그룹에 태그를 추가합니다.

8. 컴퓨팅 노드 그룹 생성을 선택합니다. AWS PCS가 노드 그룹을 프로비저닝하는 `Creating` 동안 상태 필드가 표시됩니다. 몇 분 정도 걸릴 수 있습니다.

권장 다음 단계

- 작업을 처리할 수 있도록 노드 그룹을 AWS PCS의 대기열에 추가합니다.

AWS CLI

를 사용하여 컴퓨팅 노드 그룹을 생성하려면 AWS CLI

다음 명령을 사용하여 대기열을 생성합니다. 명령을 실행하기 전에 다음과 같은 바꾸기를 합니다.

1. ##을와 같이 클러스터 AWS 리전 를 생성할의 ID로 바꿉니다`us-east-1`.

2. *my-cluster*를 클러스터의 이름 또는 로 바꿉clusterId니다.
3. *my-node-group*을 컴퓨팅 노드 그룹의 이름으로 바꿉니다. 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하며 25자를 초과할 수 없습니다. 이름은 클러스터 내에서 고유해야 합니다.
4. *subnet-ExampleID1*을 클러스터 VPC의 하나 이상의 서브넷 IDs로 바꿉니다.
5. *lt-ExampleID1*을 사용자 지정 시작 템플릿의 ID로 바꿉니다. 준비된 항목이 없는 경우 섹션을 AWS PCS에서 Amazon EC2 시작 템플릿 사용 참조하여 생성 방법을 알아보세요.

   > **Important**
   >
   > AWS PCS는 각 컴퓨팅 노드 그룹에 대한 관리형 시작 템플릿을 생성합니다. 이름은 입니다pcs-*identifier*-do-not-delete. 컴퓨팅 노드 그룹을 생성하거나 업데이트할 때 이러한 항목을 선택하지 마십시오. 그렇지 않으면 노드 그룹이 올바르게 작동하지 않습니다.

6. *launch-template-version*을 특정 시작 템플릿 버전으로 바꿉니다. AWS PCS는 노드 그룹을 시작 템플릿의 특정 버전과 연결합니다.
7. *arn:InstanceProfile*을 IAM 인스턴스 프로파일의 ARN으로 바꿉니다. 준비된 항목이 없는 경우에서 지침을 참조AWS PCS에서 Amazon EC2 시작 템플릿 사용하세요.
8. *min-instances* 및 *max-instances*를 정수 값으로 바꿉니다. 고정된 수의 노드가 실행되는 정적 구성 또는 최대 수의 노드가 실행될 수 있는 동적 구성을 정의할 수 있습니다. 정적 구성의 경우 최소값과 최대값을 0보다 큰 동일한 숫자로 설정합니다. 동적 구성의 경우 최소 인스턴스를 0으로 설정하고 최대 인스턴스를 0보다 큰 숫자로 설정합니다. AWS PCS는 정적 인스턴스와 동적 인스턴스가 혼합된 컴퓨팅 노드 그룹을 지원하지 않습니다.
9. *t3.large*를 다른 인스턴스 유형으로 바꿉니다. instanceType 설정 목록을 지정하여 인스턴스 유형을 더 추가할 수 있습니다. 예: *--instance-configs instanceType=c6i.16xlarge instanceType=c6a.16xlarge*. 모든 인스턴스 유형은 동일한 프로세서 아키텍처(x86_64 또는 arm64)와 vCPUs 가져야 합니다. 인스턴스에 GPUs가 있는 경우 모든 인스턴스 유형에 동일한 수의 GPUs가 있어야 합니다.

```
aws pcs create-compute-node-group --region region \
    --cluster-identifier my-cluster \
    --compute-node-group-name my-node-group \
    --subnet-ids subnet-ExampleID1 \
    --custom-launch-template id=lt-ExampleID1,version='launch-template-version' \
    --iam-instance-profile-arn=arn:InstanceProfile \
```

```
    --scaling-config minInstanceCount=min-instances,maxInstanceCount=max-instance \
    --instance-configs instanceType=t3.large
```

Example- 사용자 지정 Slurm 설정을 사용하여 컴퓨팅 노드 그룹 생성

```
aws pcs create-compute-node-group --region region \
    --cluster-identifier my-cluster \
    --compute-node-group-name my-node-group \
    --subnet-ids subnet-ExampleID1 \
    --custom-launch-template id=lt-ExampleID1,version='launch-template-version' \
    --iam-instance-profile-arn=arn:InstanceProfile \
    --scaling-config minInstanceCount=min-instances,maxInstanceCount=max-instance \
    --instance-configs instanceType=t3.large \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Features,parameterValue="gpu,nvme"}]'
```

자세한 내용은 AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정 단원을 참조하십시오.

`create-compute-node-group` 명령에 추가할 수 있는 몇 가지 선택적 구성 설정이 있습니다.

- 사용자 지정 시작 템플릿에 AMI에 대한 참조가 포함되어 있지 않은`--amiId`지 또는 해당 값을 재정의하려는지 지정할 수 있습니다. 노드 그룹에 사용되는 AMI는 AWS PCS와 호환되어야 합니다. 에서 제공하는 샘플 AMI를 선택할 수도 있습니다 AWS. 이 주제에 대한 자세한 내용은 단원을 참조하십시오AWS PCS용 Amazon Machine Image(AMIs).
- `--purchase-option`를 사용하여 AWS PCS가 컴퓨팅 노드 그룹의 EC2 인스턴스를 구매하는 방법을 선택합니다. 온디맨드가 기본값입니다.
  - `ONDEMAND` - 온디맨드 인스턴스를 사용합니다. 또한 온디맨드 용량 예약(ODCR)을 사용하려는 경우이 옵션을 선택합니다. 자세한 내용은 AWS PCSODCRs 사용 단원을 참조하십시오.
  - `SPOT` - 스팟 인스턴스를 사용합니다. 스팟 인스턴스를 선택하는 경우 `--allocation-strategy`를 사용하여 노드 그룹에서 인스턴스를 시작할 때 AWS PCS가 스팟 용량 풀을 선택하는 방법을 정의할 수도 있습니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 스팟 인스턴스에 대한 할당 전략을 참조하세요.
  - `CAPACITY_BLOCK` - ML 예약에 기존 Amazon EC2 용량 블록을 사용합니다. 자세한 내용은 AWS PCS에서 ML에 Amazon EC2 용량 블록 사용 단원을 참조하십시오.
- 를 사용하여 노드 그룹의 노드에 대한 Slurm 구성 옵션을 제공할 수 있습니다`--slurm-configuration`. 가중치(예약 우선 순위)와 실제 메모리를 설정할 수 있습니다. 가중치가 낮은 노드는 우선 순위가 높고 단위는 임의입니다. 자세한 내용은 Slurm 설명서의 가중치를 참조하세요. 실제 메모리는 노드 그룹의 노드에서 실제 메모리의 크기(GB)를 설정합니다. 구성에서 AWS

PCS의 클러스터에 대한 CR_CPU_Memory 옵션과 함께 사용하기 위한 것입니다Slurm. 자세한 내용은 Slurm 설명서의 RealMemory를 참조하세요.

> **Important**
>
> 컴퓨팅 노드 그룹을 생성하는 데 몇 분 정도 걸릴 수 있습니다.

다음 명령을 사용하여 노드 그룹의 상태를 쿼리할 수 있습니다. 상태가에 도달할 때까지 노드 그룹을 대기열과 연결할 수 없습니다ACTIVE.

```
aws pcs get-compute-node-group --region region \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group
```

AWS PCS 컴퓨팅 노드 그룹 업데이트

이 주제에서는 사용 가능한 옵션에 대한 개요를 제공하고 AWS PCS 컴퓨팅 노드 그룹을 업데이트할 때 고려해야 할 사항을 설명합니다. Slurm 사용자 지정 설정에 대한 자세한 내용은 섹션을 참조하세요AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정.

AWS PCS 컴퓨팅 노드 그룹 업데이트 옵션

AWS PCS 컴퓨팅 노드 그룹을 업데이트하면 AWS PCS에서 시작한 인스턴스의 속성과 해당 인스턴스가 시작되는 방식에 대한 규칙을 변경할 수 있습니다. 예를 들어 노드 그룹 인스턴스의 AMI를 다른 소프트웨어가 설치된 인스턴스로 바꿀 수 있습니다. 또는 보안 그룹을 업데이트하여 인바운드 또는 아웃바운드 네트워크 연결을 변경할 수 있습니다. 조정 구성과 기본 구매 옵션을 변경할 수도 있습니다.

다음 노드 그룹 설정은 생성 후 변경할 수 없습니다.

- 이름
- 인스턴스

AWS PCS 컴퓨팅 노드 그룹 업데이트 시 고려 사항

컴퓨팅 노드 그룹은 작업을 처리하고 대화형 셸 액세스 및 기타 작업을 제공하는 데 사용되는 EC2 인스턴스를 정의합니다. 종종 하나 이상의 AWS PCS 대기열과 연결됩니다. 컴퓨팅 노드 그룹을 업데이트하여 동작(또는 노드의 동작)을 변경할 때 다음 사항을 고려하세요.

- 컴퓨팅 노드 그룹 속성에 대한 변경 사항은 컴퓨팅 노드 그룹 상태가 업데이트 중에서 활성으로 변경될 때 적용됩니다. 업데이트된 속성으로 새 인스턴스가 시작됩니다.
- 특정 노드의 구성에 영향을 주지 않는 업데이트는 실행 중인 노드에 영향을 주지 않습니다. 서브넷 추가 및 할당 전략 변경을 예로 들 수 있습니다.
- 컴퓨팅 노드 그룹의 시작 템플릿을 업데이트하는 경우 새 버전을 사용하도록 컴퓨팅 노드 그룹을 업데이트해야 합니다.
- 컴퓨팅 노드 그룹의 노드에서 보안 그룹을 추가하거나 제거하려면 시작 템플릿을 편집하고 컴퓨팅 노드 그룹을 업데이트합니다. 업데이트된 보안 그룹 세트와 함께 새 인스턴스가 시작됩니다.
- 컴퓨팅 노드 그룹에서 사용하는 보안 그룹을 직접 편집하면 실행 중인 인스턴스와 향후 인스턴스에 즉시 적용됩니다.
- 컴퓨팅 노드 그룹에서 사용하는 IAM 인스턴스 프로파일에서 권한을 추가하거나 제거하면 실행 중인 인스턴스와 향후 인스턴스에 즉시 적용됩니다.
- 컴퓨팅 노드 그룹의 인스턴스에서 사용하는 AMI를 변경하려면 새 AMI를 사용하도록 컴퓨팅 노드 그룹(또는 시작 템플릿)을 업데이트하고 AWS PCS가 인스턴스를 교체할 때까지 기다립니다.
- AWS PCS는 노드 그룹 업데이트 작업 후 노드 그룹의 기존 인스턴스를 대체합니다. 노드에서 실행 중인 작업이 있는 경우 AWS PCS가 노드를 교체하기 전에 해당 작업을 완료할 수 있습니다. 대화형 사용자 프로세스(예: 로그인 노드 인스턴스)가 종료됩니다. 노드 그룹 상태는 AWS PCS가 교체를 위해 인스턴스를 표시`Active`하면 로 돌아가지만 인스턴스가 유휴 상태일 때는 실제 교체가 발생합니다.
- 컴퓨팅 노드 그룹에 허용되는 최대 인스턴스 수를 줄이면 AWS PCS는 새 최대값을 충족하기 위해 Slurm에서 노드를 제거합니다. AWS PCS는 제거된 Slurm 노드와 연결된 실행 중인 인스턴스를 종료합니다. 제거된 노드에서 실행 중인 작업이 실패하고 대기열로 돌아갑니다.
- AWS PCS는 각 컴퓨팅 노드 그룹에 대한 관리형 시작 템플릿을 생성합니다. 이름은 입니다`pcs-`*`identifier`*`-do-not-delete`. 컴퓨팅 노드 그룹을 생성하거나 업데이트할 때 선택하지 마십시오. 그렇지 않으면 노드 그룹이 올바르게 작동하지 않습니다.
- 구매 옵션에 스팟을 사용하도록 컴퓨팅 노드 그룹을 업데이트하는 경우 계정에 AWSServiceRoleForEC2Spot 서비스 연결 역할이 있어야 합니다. 자세한 내용은 AWS PCS에 대한 Amazon EC2 스팟 역할 단원을 참조하십시오.

AWS PCS 컴퓨팅 노드 그룹을 업데이트하려면

AWS Management Console 또는 AWS CLI를 사용하여 노드 그룹을 업데이트할 수 있습니다.

AWS Management Console

컴퓨팅 노드 그룹을 업데이트하려면

1. 에서 AWS PCS 콘솔을 엽니다. `https://console.aws.amazon.com/pcs/home#/clusters`
2. 컴퓨팅 노드 그룹을 업데이트하려는 클러스터를 선택합니다.
3. 컴퓨팅 노드 그룹으로 이동하여 업데이트하려는 노드 그룹으로 이동한 다음 편집을 선택합니다.
4. 컴퓨팅 구성, 추가 설정 및 Slurm 사용자 지정 설정 섹션에서 다음을 제외한 모든 값을 업데이트합니다.

   - 인스턴스 - 컴퓨팅 노드 그룹의 인스턴스는 변경할 수 없습니다.

   Slurm 사용자 지정 설정에 대한 자세한 내용은 섹션을 참조하세요AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정.
5. 업데이트를 선택합니다. 변경 사항이 적용되는 동안 상태 필드에 업데이트 중이 표시됩니다.

> **Important**
>
> 컴퓨팅 노드 그룹 업데이트에는 몇 분 정도 걸릴 수 있습니다.

AWS CLI

컴퓨팅 노드 그룹을 업데이트하려면

1. 다음 명령을 사용하여 컴퓨팅 노드 그룹을 업데이트합니다. 명령을 실행하기 전에 다음과 같은 바꾸기를 합니다.

   a. *region-code*를 클러스터를 생성하려는 AWS 리전으로 바꿉니다.

   b. *my-node-group*을 컴퓨팅 노드 그룹의 이름 또는 `computeNodeGroupId` 로 바꿉니다.

   c. *my-cluster*를 클러스터의 이름 또는 로 바꿉`clusterId`니다.

```
aws pcs update-compute-node-group --region region-code \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group
```

Example- 사용자 지정 Slurm 설정으로 컴퓨팅 노드 그룹 업데이트

```
aws pcs update-compute-node-group --region region-code \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Features,parameterValue="gpu,nvme"}]'
```

자세한 내용은 AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정 단원을 참조하십시오.

2. 를 제외한 모든 노드 그룹 파라미터를 업데이트합니다`--instance-configs`. 예를 들어 새 AMI ID를 설정하려면 *my-custom-ami-id*가 선택한 AMI로 대체`--amiId my-custom-ami-id`되는 위치를 전달합니다.

> **Important**
>
> 컴퓨팅 노드 그룹을 업데이트하는 데 몇 분 정도 걸릴 수 있습니다.

다음 명령을 사용하여 노드 그룹의 상태를 쿼리할 수 있습니다.

```
aws pcs get-compute-node-group --region region-code \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-node-group
```

AWS PCS에서 컴퓨팅 노드 그룹 삭제

이 주제에서는 사용 가능한 옵션에 대한 개요를 제공하고 AWS PCS에서 컴퓨팅 노드 그룹을 삭제할 때 고려해야 할 사항을 설명합니다.

컴퓨팅 노드 그룹 삭제 시 고려 사항

컴퓨팅 노드 그룹은 작업을 처리하고 대화형 셸 액세스 및 기타 작업을 제공하는 데 사용되는 EC2 인스턴스를 정의합니다. 종종 하나 이상의 AWS PCS 대기열과 연결됩니다. 컴퓨팅 노드 그룹을 삭제하기 전에 다음 사항을 고려하세요.

- 컴퓨팅 노드 그룹에서 시작한 모든 EC2 인스턴스가 종료됩니다. 이렇게 하면 이러한 인스턴스에서 실행 중인 작업이 취소되고 실행 중인 대화형 프로세스가 종료됩니다.
- 컴퓨팅 노드 그룹을 삭제하려면 먼저 모든 대기열에서 연결을 해제해야 합니다. 자세한 내용은 AWS PCS 대기열 업데이트 단원을 참조하십시오.

컴퓨팅 노드 그룹 삭제

AWS Management Console 또는 AWS CLI 를 사용하여 컴퓨팅 노드 그룹을 삭제할 수 있습니다.

AWS Management Console

컴퓨팅 노드 그룹을 삭제하려면

1. AWS PCS 콘솔을 엽니다.
2. 컴퓨팅 노드 그룹의 클러스터를 선택합니다.
3. 컴퓨팅 노드 그룹으로 이동하여 삭제할 컴퓨팅 노드 그룹을 선택합니다.
4. 삭제를 선택합니다.
5. 상태 필드에가 표시됩니다`Deleting`. 완료되는 데 몇 분 정도 걸릴 수 있습니다.

> **Note**
>
> 스케줄러의 기본 명령을 사용하여 컴퓨팅 노드 그룹이 삭제되었는지 확인할 수 있습니다. 예를 들어 Slurm의 경우 `sinfo` 또는 `squeue`를 사용합니다.

AWS CLI

컴퓨팅 노드 그룹을 삭제하려면

- 다음 명령을 사용하여 컴퓨팅 노드 그룹을 삭제하고 다음과 같이 대체합니다.
  - *`region-code`*를 클러스터가 있는 로 바꿉 AWS 리전 니다.

- *my-node-group*을 컴퓨팅 노드 그룹의 이름 또는 ID로 바꿉니다.
- *my-cluster*를 클러스터의 이름 또는 ID로 바꿉니다.

```
aws pcs delete-compute-node-group --region region-code \
      --compute-node-group-identifier my-node-group \
      --cluster-identifier my-cluster
```

컴퓨팅 노드 그룹을 삭제하는 데 몇 분 정도 걸릴 수 있습니다.

> **Note**
>
> 스케줄러의 기본 명령을 사용하여 컴퓨팅 노드 그룹이 삭제되었는지 확인할 수 있습니다. 예를 들어 Slurm의 경우 `sinfo` 또는 `squeue`를 사용합니다.

AWS PCS에서 컴퓨팅 노드 그룹 세부 정보 가져오기

AWS Management Console 또는를 사용하여 컴퓨팅 노드 그룹 ID, Amazon 리소스 이름(ARN) 및 Amazon Machine Image(AMI) ID와 같은 컴퓨팅 노드 그룹에 대한 세부 정보를 AWS CLI 가져올 수 있습니다. 이러한 세부 정보는 AWS PCS API 작업 및 구성에 필요한 값인 경우가 많습니다.

AWS Management Console

컴퓨팅 노드 그룹 세부 정보를 가져오는 방법

1. AWS PCS 콘솔을 엽니다.
2. 클러스터를 선택합니다.
3. 컴퓨팅 노드 그룹을 선택합니다.
4. 목록 창에서 컴퓨팅 노드 그룹을 선택합니다.

AWS CLI

컴퓨팅 노드 그룹 세부 정보를 가져오는 방법

1. ListClusters API 작업을 사용하여 클러스터 이름 또는 ID를 찾습니다.

```
aws pcs list-clusters
```

출력 예제:

```
{
    "clusters": [
        {
            "name": "get-started-cfn",
            "id": "pcs_abc1234567",
            "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567",
            "createdAt": "2025-04-01T20:11:22+00:00",
            "modifiedAt": "2025-04-01T20:11:22+00:00",
            "status": "ACTIVE"
        }
    ]
}
```

2. ListComputeNodeGroups API 작업을 사용하여 클러스터의 컴퓨팅 노드 그룹을 나열합니다.

```
aws pcs list-compute-node-groups --cluster-identifier cluster-name-or-id
```

호출 예제:

```
aws pcs list-compute-node-groups --cluster-identifier get-started-cfn
```

출력 예제:

```
{
    "computeNodeGroups": [
        {
            "name": "compute-1",
            "id": "pcs_abc123abc1",
            "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567/
computenodegroup/pcs_abc123abc1",
            "clusterId": "pcs_abc1234567",
            "createdAt": "2025-04-01T20:19:25+00:00",
            "modifiedAt": "2025-04-01T20:19:25+00:00",
            "status": "ACTIVE"
        },
        {
```

```
            "name": "login",
            "id": "pcs_abc456abc7",
            "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567/
computenodegroup/pcs_abc456abc7",
            "clusterId": "pcs_abc1234567",
            "createdAt": "2025-04-01T20:19:31+00:00",
            "modifiedAt": "2025-04-01T20:19:31+00:00",
            "status": "ACTIVE"
        }
    ]
}
```

3. GetComputeNodeGroup API 작업을 사용하여 컴퓨팅 노드 그룹에 대한 추가 세부 정보를 가져옵니다.

```
aws pcs get-compute-node-group --cluster-identifier cluster-name-or-id --
compute-node-group-identifier compute-node-group-name-or-id
```

호출 예제:

```
aws pcs get-compute-node-group --cluster-identifier get-started-cfn --compute-
node-group-identifier compute-1
```

출력 예제:

```
{
    "computeNodeGroup": {
        "name": "compute-1",
        "id": "pcs_abc123abc1",
        "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_abc1234567/
computenodegroup/pcs_abc123abc1",
        "clusterId": "pcs_abc1234567",
        "createdAt": "2025-04-01T20:19:25+00:00",
        "modifiedAt": "2025-04-01T20:19:25+00:00",
        "status": "ACTIVE",
        "amiId": "ami-0123456789abcdef0",
        "subnetIds": [
            "subnet-abc012345789abc12"
        ],
        "purchaseOption": "ONDEMAND",
        "customLaunchTemplate": {
            "id": "lt-012345abcdef01234",
```

```
            "version": "1"
        },
        "iamInstanceProfileArn": "arn:aws:iam::111122223333:instance-profile/
AWSPCS-get-started-cfn-us-east-1",
        "scalingConfiguration": {
            "minInstanceCount": 0,
            "maxInstanceCount": 4
        },
        "instanceConfigs": [
            {
                "instanceType": "c6i.xlarge"
            }
        ]
    }
}
```

AWS PCS에서 컴퓨팅 노드 그룹 인스턴스 찾기

각 AWS PCS 컴퓨팅 노드 그룹은 공유 구성으로 EC2 인스턴스를 시작할 수 있습니다. EC2 태그를 사용하여 AWS Management Console 또는의 컴퓨팅 노드 그룹에서 인스턴스를 찾을 수 있습니다 AWS CLI.

AWS Management Console

컴퓨팅 노드 그룹 인스턴스를 찾으려면

1. AWS PCS 콘솔을 엽니다.
2. 클러스터를 선택합니다.
3. 컴퓨팅 노드 그룹을 선택합니다.
4. 생성한 로그인 노드 그룹의 ID를 찾습니다.
5. EC2 콘솔로 이동하여 인스턴스를 선택합니다.
6. 다음 태그가 있는 인스턴스를 검색합니다. *node-group-id*를 컴퓨팅 노드 그룹의 ID(이름 아님)로 바꿉니다.

   ```
   aws:pcs:compute-node-group-id=node-group-id
   ```

7. (선택 사항) 검색 필드에서 인스턴스 상태 값을 변경하여 구성 중이거나 최근에 종료된 인스턴스를 찾을 수 있습니다.

8. 태그가 지정된 인스턴스 목록에서 각 인스턴스의 인스턴스 ID와 IP 주소를 찾습니다.

AWS CLI

노드 그룹 인스턴스를 찾으려면 다음 명령을 사용합니다. 명령을 실행하기 전에 다음을 대체합니다.

- 를 클러스터 AWS 리전 의 *region-code*로 바꿉니다. 예시: us-east-1
- 를 컴퓨팅 노드 그룹의 ID(이름 아님)*node-group-id*로 바꿉니다. 컴퓨팅 노드 그룹의 ID를 찾으려면 섹션을 참조하세요AWS PCS에서 컴퓨팅 노드 그룹 세부 정보 가져오기.
- 다른 상태에서 EC2 인스턴스running를 찾으terminated려면 pending 또는와 같은 다른 인스턴스 상태로를 바꿉니다.

```
aws ec2 describe-instances \
     --region region-code --filters \
     "Name=tag:aws:pcs:compute-node-group-id,Values=node-group-id" \
     "Name=instance-state-name,Values=running" \
     --query 'Reservations[*].Instances[*].
{InstanceID:InstanceId,State:State.Name,PublicIP:PublicIpAddress,PrivateIP:PrivateIpAddress}
```

이 명령은 다음과 비슷한 출력을 반환합니다. 인스턴스가 프라이빗 서브넷에 있는 null 경우의 값은 PublicIP 입니다.

```
[
    [
        {
            "InstanceID": "i-0123456789abcdefa",
            "State": "running",
            "PublicIP": "18.189.32.188",
            "PrivateIP": "10.0.0.1"
        }
    ]
]
```

> **Note**
>
> 많은 수의 인스턴스를 반환`describe-instances`할 것으로 예상되는 경우 여러 페이지에 대한 옵션을 사용해야 합니다. 자세한 내용은 Amazon Elastic Compute Cloud API 참조의 DescribeInstances를 참조하세요.

AWS PCS에서 Amazon EC2 시작 템플릿 사용

Amazon EC2에서 시작 템플릿은 기본 설정 세트를 저장할 수 있으므로 인스턴스를 시작할 때 개별적으로 지정할 필요가 없습니다. AWS PCS는 컴퓨팅 노드 그룹을 구성하는 유연한 방법으로 시작 템플릿을 통합합니다. 노드 그룹을 생성할 때 시작 템플릿을 제공합니다. AWS PCS는 서비스에서 작동하도록 변환을 포함하는 파생된 시작 템플릿을 생성합니다.

사용자 지정 시작 템플릿을 작성할 때 옵션과 고려 사항을 이해하면 AWS PCS에 사용할 옵션을 작성하는 데 도움이 될 수 있습니다. 시작 템플릿에 대한 자세한 내용은 Amazon EC2 사용 설명서의 시작 템플릿에서 인스턴스 시작을 참조하세요.

주제

AWS PCS의 시작 템플릿 개요

EC2 시작 템플릿에 포함할 수 있는 파라미터는 30개가 넘으며 인스턴스 구성 방법의 여러 측면을 제어합니다. 대부분은 AWS PCS와 완벽하게 호환되지만 몇 가지 예외가 있습니다.

다음 EC2 시작 템플릿 파라미터는 서비스에서 직접 관리해야 하므로 AWS PCS에서 무시됩니다.

- 인스턴스 유형/인스턴스 유형 속성 지정(`InstanceRequirements`) - AWS PCS는 속성 기반 인스턴스 선택을 지원하지 않습니다.
- 인스턴스 유형(`InstanceType`) - 노드 그룹을 생성할 때 인스턴스 유형을 지정합니다.
- 고급 세부 정보/IAM 인스턴스 프로파일(`IamInstanceProfile`) - 노드 그룹을 생성하거나 업데이트할 때 이를 제공합니다.
- 고급 세부 정보/API 종료 비활성화(`DisableApiTermination`) - AWS PCS는 시작하는 노드 그룹 인스턴스의 수명 주기를 제어해야 합니다.
- 고급 세부 정보/API 중지 비활성화(`DisableApiStop`) - AWS PCS는 시작하는 노드 그룹 인스턴스의 수명 주기를 제어해야 합니다.

- 고급 세부 정보/중지 - 최대 절전 모드 동작(HibernationOptions) - AWS PCS는 인스턴스 최대 절전 모드를 지원하지 않습니다.
- 고급 세부 정보/Elastic GPU(ElasticGpuSpecifications) - Amazon Elastic Graphics는 2024년 1월 8일에 수명이 종료되었습니다.
- 고급 세부 정보/탄력적 추론(ElasticInferenceAccelerators) - 신규 고객은 더 이상 Amazon Elastic Inference를 사용할 수 없습니다.
- AAdvanced 세부 정보/CPU 옵션 지정/코어당 스레드(ThreadsPerCore) – AWS PCS는 코어당 스레드 수를 1로 설정합니다.

이러한 파라미터에는 AWS PCS와의 호환성을 지원하는 특별한 요구 사항이 있습니다.

- 사용자 데이터(UserData) - 멀티파트 인코딩되어야 합니다. AWS PCS용 Amazon EC2 사용자 데이터 작업을(를) 참조하세요.
- 애플리케이션 및 OS 이미지(ImageId) - 이를 포함할 수 있습니다. 그러나 노드 그룹을 생성하거나 업데이트할 때 AMI ID를 지정하면 시작 템플릿의 값이 재정의됩니다. 제공하는 AMI는 AWS PCS와 호환되어야 합니다. 자세한 내용은 "를 참조하십시오AWS PCS용 Amazon Machine Image(AMIs).
- 네트워크 설정/방화벽(보안 그룹)(SecurityGroups) - 보안 그룹 이름 목록은 AWS PCS 시작 템플릿에서 설정할 수 없습니다. 시작 템플릿에서 네트워크 인터페이스를 정의하지 않는 한 보안 그룹 IDs 목록(SecurityGroupIds)을 설정할 수 있습니다. 그런 다음 각 인터페이스에 대한 보안 그룹 IDs 지정해야 합니다. 자세한 내용은 AWS PCS의 보안 그룹 단원을 참조하십시오.
- 네트워크 설정/고급 네트워크 구성(NetworkInterfaces) - EC2 인스턴스를 단일 네트워크 카드와 함께 사용하고 특수 네트워킹 구성이 필요하지 않은 경우 AWS PCS는 인스턴스 네트워킹을 구성할 수 있습니다. 여러 네트워크 카드를 구성하거나 인스턴스에서 Elastic Fabric Adapter를 활성화하려면를 사용합니다NetworkInterfaces. 각 네트워크 인터페이스에는 아래에 보안 그룹 IDs 목록이 있어야 합니다Groups. 자세한 내용은 AWS PCS의 여러 네트워크 인터페이스 단원을 참조하십시오.
- 고급 세부 정보/용량 예약(CapacityReservationSpecification) -이 설정은 설정할 수 있지만 AWS PCS로 작업할 CapacityReservationId 때 특정를 참조할 수는 없습니다. 그러나 해당 그룹에 하나 이상의 용량 예약이 포함된 용량 예약 그룹을 참조할 수 있습니다. 자세한 내용은 AWS PCS의 용량 예약 단원을 참조하십시오.

기본 시작 템플릿 생성

AWS Management Console 또는를 사용하여 시작 템플릿을 생성할 수 있습니다 AWS CLI.

AWS Management Console

시작 템플릿 생성

1. Amazon EC2 콘솔을 열고 시작 템플릿을 선택합니다.
2. Create launch template(시작 템플릿 생성)을 선택합니다.
3. 시작 템플릿 이름 및 설명에서 시작 템플릿 이름에 고유하고 고유한 이름을 입력합니다.
4. 키 페어 이름의 키 페어(로그인)에서 AWS PCS에서 관리하는 EC2 인스턴스에 로그인하는 데 사용할 SSH 키 페어를 선택합니다. 이는 선택 사항이며, 권장 사항은 아닙니다.
5. 네트워크 설정에서 방화벽(보안 그룹)을 선택하고 네트워크 인터페이스에 연결할 보안 그룹을 선택합니다. 시작 템플릿의 모든 보안 그룹은 AWS PCS 클러스터 VPC의 보안 그룹이어야 합니다. 최소한 다음을 선택합니다.
   - AWS PCS 클러스터와의 통신을 허용하는 보안 그룹
   - AWS PCS에서 시작한 EC2 인스턴스 간의 통신을 허용하는 보안 그룹
   - (선택 사항) 대화형 인스턴스에 대한 인바운드 SSH 액세스를 허용하는 보안 그룹
   - (선택 사항) 컴퓨팅 노드가 인터넷에 발신 연결을 할 수 있도록 허용하는 보안 그룹
   - (선택 사항) 공유 파일 시스템 또는 데이터베이스 서버와 같은 네트워크 리소스에 대한 액세스를 허용하는 보안 그룹(들).
6. 새 시작 템플릿 ID는 Amazon EC2 콘솔의 시작 템플릿에서 액세스할 수 있습니다. 시작 템플릿 ID의 형식은 입니다`lt-0123456789abcdef01`.

권장 다음 단계

- 새 시작 템플릿을 사용하여 AWS PCS 컴퓨팅 노드 그룹을 생성하거나 업데이트합니다.

AWS CLI

시작 템플릿 생성

다음 명령을 사용하여 시작 템플릿을 생성합니다.

- 명령을 실행하기 전에 다음과 같은 바꾸기를 합니다.
  a. *`region-code`*를 AWS PCS로 작업하는 AWS 리전 로 바꿉니다.
  b. *`my-launch-template-name`*을 템플릿 이름으로 바꿉니다. AWS 계정 와 사용 AWS 리전 중인에 고유해야 합니다.

c. *my-ssh-key-name*을 기본 SSH 키의 이름으로 바꿉니다.

d. *sg-ExampleID1* 및 *sg-ExampleID2*를 EC2 인스턴스와 스케줄러 간의 통신과 EC2 인스턴스 간의 통신을 허용하는 보안 그룹 IDs로 바꿉니다. EC2 이 모든 트래픽을 활성화하는 보안 그룹이 하나만 있는 경우 `sg-ExampleID2` 및 이전 쉼표 문자를 제거할 수 있습니다. 보안 그룹 IDs. 시작 템플릿에 포함하는 모든 보안 그룹은 AWS PCS 클러스터 VPC에서 가져와야 합니다.

```
aws ec2 create-launch-template --region region-code \
    --launch-template-name my-template-name \
    --launch-template-data '{"KeyName":"my-ssh-key-name","SecurityGroupIds":
 ["sg-ExampleID1","sg-ExampleID2"]}'
```

AWS CLI 는 다음과 유사한 텍스트를 출력합니다. 시작 템플릿 ID는에서 찾을 수 있습니다`LaunchTemplateId`.

```
{
    "LaunchTemplate": {
        "LatestVersionNumber": 1,
        "LaunchTemplateId": "lt-0123456789abcdef01",
        "LaunchTemplateName": "my-launch-template-name",
        "DefaultVersionNumber": 1,
        "CreatedBy": "arn:aws:iam::123456789012:user/Bob",
        "CreateTime": "2019-04-30T18:16:06.000Z"
    }
}
```

권장 다음 단계

- 새 시작 템플릿을 사용하여 AWS PCS 컴퓨팅 노드 그룹을 생성하거나 업데이트합니다.

AWS PCS용 Amazon EC2 사용자 데이터 작업

인스턴스가 시작될 때 `cloud-init` 실행되는 시작 템플릿에 EC2 사용자 데이터를 제공할 수 있습니다. 콘텐츠 유형의 사용자 데이터 블록은 인스턴스가 AWS PCS API에 등록되기 전에 `cloud-config` 실행되고 콘텐츠 유형의 사용자 데이터 블록은 등록이 완료된 후 Slurm 데몬이 시작되기 전에 `text/x-shellscript` 실행됩니다. 콘텐츠 유형에 대한 자세한 내용은 cloud-init 문서를 참조하세요.

사용자 데이터는 다음을 포함하되 이에 국한되지 않는 일반적인 구성 시나리오를 수행할 수 있습니다.

- 사용자 또는 그룹 포함
- 패키지 설치
- 파티션 및 파일 시스템 생성
- 네트워크 파일 시스템 탑재

시작 템플릿의 사용자 데이터는 MIME 멀티파트 아카이브 형식이어야 합니다. 이는 사용자 데이터가 노드 그룹의 노드를 구성하는 데 필요한 다른 AWS PCS 사용자 데이터와 병합되기 때문입니다. 여러 사용자 데이터 블록을 단일 MIME 멀티파트 파일로 결합할 수 있습니다.

MIME 멀티파트 파일은 다음과 같은 구성 요소로 이루어집니다.

- 콘텐츠 유형 및 부분 경계 선언: `Content-Type: multipart/mixed; boundary="==BOUNDARY=="`
- MIME 버전 선언: `MIME-Version: 1.0`
- 다음 구성 요소를 포함하는 하나 이상의 사용자 데이터 블록:
  - 사용자 데이터 블록의 시작을 나타내는 시작 경계: `--==BOUNDARY==` 이 경계 앞의 라인은 비워 두어야 합니다.
  - 블록의 콘텐츠 유형 선언: `Content-Type: text/cloud-config; charset="us-ascii"` 또는 `Content-Type: text/x-shellscript; charset="us-ascii"`. 콘텐츠 유형 선언 뒤의 라인은 비워 두어야 합니다.
  - 쉘 명령 또는 `cloud-config` 지시어 목록 등의 사용자 데이터 콘텐츠
- MIME 멀티파트 파일의 끝을 나타내는 종료 경계: `--==BOUNDARY==--` 종료 경계 앞의 라인은 비워 두어야 합니다.

> **Note**
>
> Amazon EC2 콘솔의 시작 템플릿에 사용자 데이터를 추가하는 경우 일반 텍스트로 붙여 넣을 수 있습니다. 또는 파일에서 업로드할 수도 있습니다. AWS CLI 또는 AWS SDK를 사용하는 경우 먼저 사용자 데이터를 base64로 인코딩하고이 JSON 파일에 표시된 대로 CreateLaunchTemplate을 호출할 때 해당 문자열을 `UserData` 파라미터 값으로 제출해야 합니다.

```
{
    "LaunchTemplateName": "base64-user-data",
    "LaunchTemplateData": {
        "UserData":
 "ewogICAgIkxhdW5jaFRlbXBsYXRlTmFtZSI6ICJpbmNyZWFzZS1jb250YWluZXItdm9sdW..."
    }
}
```

예제

- 예: 패키지 리포지토리에서 소프트웨어 설치
- 예: S3 버킷에서 스크립트 실행
- 예: 전역 환경 변수 설정
- AWS PCS에서 네트워크 파일 시스템 사용
- 예: EFS 파일 시스템을 공유 홈 디렉터리로 사용

예: 패키지 리포지토리에서 AWS PCS용 소프트웨어 설치

시작 템플릿"userData"에이 스크립트를의 값으로 제공합니다. 자세한 내용은 AWS PCS용 Amazon EC2 사용자 데이터 작업 단원을 참조하십시오.

이 스크립트는 Cloud-config를 사용하여 시작 시 노드 그룹 인스턴스에 소프트웨어 패키지를 설치합니다. 자세한 내용은 cloud-init 설명서의 사용자 데이터 형식을 참조하세요. 이 예제에서는 `curl` 및를 설치합니다`llvm`.

> **Note**
>
> 인스턴스는 구성된 패키지 리포지토리에 연결할 수 있어야 합니다.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
```

```
- python3-devel
- rust
- golang

--==MYBOUNDARY==--
```

예: S3 버킷에서 AWS PCS에 대한 추가 스크립트 실행

시작 템플릿"userData"에서이 스크립트를의 값으로 제공합니다. 자세한 내용은 AWS PCS용 Amazon EC2 사용자 데이터 작업 단원을 참조하십시오.

다음 사용자 데이터 스크립트는 cloud-config를 사용하여 S3 버킷에서 스크립트를 가져와서 시작 시 노드 그룹 인스턴스에서 실행합니다. 자세한 내용은 cloud-init 설명서의 사용자 데이터 형식을 참조하세요.

다음 값을 자체 세부 정보로 바꿉니다.

- *amzn-s3-demo-bucket* – 계정이 읽을 수 있는 S3 버킷의 이름입니다.
- *object-key* - 가져올 스크립트의 S3 객체 키입니다. 여기에는 스크립트의 이름과 버킷의 폴더 구조에 있는 위치가 포함됩니다. 예를 들어 `scripts/script.sh`입니다. 자세한 내용은 Amazon Simple Storage Service 사용 설명서의 폴더를 사용하여 Amazon S3 콘솔에서 객체 구성을 참조하세요.
- *shell* -와 같은 스크립트를 실행하는 데 사용할 Linux 셸입니다`bash`.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- aws s3 cp s3://amzn-s3-demo-bucket/object-key /tmp/script.sh
- /usr/bin/shell /tmp/script.sh

--==MYBOUNDARY==--
```

노드 그룹의 IAM 인스턴스 프로파일에는 버킷에 대한 액세스 권한이 있어야 합니다. 다음 IAM 정책은 위의 사용자 데이터 스크립트에 있는 버킷의 예입니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

예: AWS PCS에 대한 전역 환경 변수 설정

시작 템플릿"userData"에이 스크립트를의 값으로 제공합니다. 자세한 내용은 AWS PCS용 Amazon EC2 사용자 데이터 작업 단원을 참조하십시오.

다음 예제에서는 /etc/profile.d를 사용하여 노드 그룹 인스턴스에서 전역 변수를 설정합니다.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
touch /etc/profile.d/awspcs-userdata-vars.sh
echo MY_GLOBAL_VAR1=100 >> /etc/profile.d/awspcs-userdata-vars.sh
echo MY_GLOBAL_VAR2=abc >> /etc/profile.d/awspcs-userdata-vars.sh

--==MYBOUNDARY==--
```

예: EFS 파일 시스템을 AWS PCS용 공유 홈 디렉터리로 사용

시작 템플릿"userData"에이 스크립트를의 값으로 제공합니다. 자세한 내용은 AWS PCS용 Amazon EC2 사용자 데이터 작업 단원을 참조하십시오.

이 예제에서는에서 예제 EFS 탑재를 확장AWS PCS에서 네트워크 파일 시스템 사용하여 공유 홈 디렉터리를 구현합니다. EFS 파일 시스템을 탑재하기 전에 /home의 콘텐츠가 백업됩니다. 그런 다음 탑재가 완료되면 콘텐츠가 공유 스토리지에 빠르게 복사됩니다.

이 스크립트의 다음 값을 자체 세부 정보로 바꿉니다.

- *`/mount-point-directory`* - EFS 파일 시스템을 탑재하려는 인스턴스의 경로입니다.
- *`filesystem-id`* - EFS 파일 시스템의 파일 시스템 ID입니다.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
  - amazon-efs-utils

runcmd:
  - mkdir -p /tmp/home
  - rsync -a /home/ /tmp/home
  - echo "filesystem-id:/ /mount-point-directory efs tls,_netdev" >> /etc/fstab
  - mount -a -t efs defaults
  - rsync -a --ignore-existing /tmp/home/ /home
  - rm -rf /tmp/home/

--==MYBOUNDARY==--
```

예: 암호 없는 SSH 활성화

공유 홈 디렉터리 예제를 기반으로를 구축하여 SSH 키를 사용하여 클러스터 인스턴스 간에 SSH 연결을 구현할 수 있습니다. 공유 홈 파일 시스템을 사용하는 각 사용자에 대해 다음과 유사한 스크립트를 실행합니다.

```
#!/bin/bash
```

```
mkdir -p $HOME/.ssh && chmod 700 $HOME/.ssh
touch $HOME/.ssh/authorized_keys
chmod 600 $HOME/.ssh/authorized_keys

if [ ! -f "$HOME/.ssh/id_rsa" ]; then
    ssh-keygen -t rsa -b 4096 -f $HOME/.ssh/id_rsa -N ""
    cat ~/.ssh/id_rsa.pub >> $HOME/.ssh/authorized_keys
fi
```

> **Note**
>
> 인스턴스는 클러스터 노드 간의 SSH 연결을 허용하는 보안 그룹을 사용해야 합니다.

AWS PCS의 용량 예약

온디맨드 용량 예약 또는 ML용 Amazon EC2 용량 블록을 사용하여 특정 가용 영역에서 특정 기간 동안 Amazon EC2 용량을 예약하여 필요할 때 필요한 컴퓨팅 용량을 사용할 수 있도록 할 수 있습니다.

온디맨드 용량 예약(ODCRs)을 사용하면 언제든지 특정 가용 영역의 Amazon EC2 인스턴스에 대한 컴퓨팅 용량을 예약할 수 있습니다. 장기 약정이나 선결제 없이 언제든지 예약을 생성하고 취소할 수 있습니다. ODCRs은 요구 사항이 변경될 때 수정할 수 있는 유연한 용량 예약이 필요한 경우에 적합합니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 온디맨드 용량 예약을 참조하세요.

ML용 Amazon EC2 용량 블록을 사용하면 향후 사용을 위해 최대 8주 전에 GPU 기반 가속 컴퓨팅 인스턴스를 예약할 수 있습니다. 1~64개의 인스턴스 블록을 1일에서 6개월까지 예약할 수 있습니다. 용량 블록은 특정 시간에 GPU 용량에 대한 액세스를 보장해야 하는 기계 학습 워크로드에 적합합니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 ML용 용량 블록을 참조하세요.

주제

- AWS PCSODCRs 사용
- AWS PCS에서 ML에 Amazon EC2 용량 블록 사용

AWS PCSODCRs 사용

AWS PCS가 예약 인스턴스를 사용하는 방법을 선택할 수 있습니다. 열려 있는 ODCR을 생성하면 AWS PCS 또는 계정의 다른 프로세스에서 시작한 일치하는 인스턴스가 예약에 포함됩니다. 대상

ODCR의 경우 특정 예약 ID로 시작된 인스턴스만 예약에 포함됩니다. 시간에 민감한 워크로드의 경우 대상 ODCRs이 더 일반적입니다.

시작 템플릿에 추가하여 대상 ODCR을 사용하도록 AWS PCS 컴퓨팅 노드 그룹을 구성할 수 있습니다. 이 작업을 수행하는 단계는 다음과 같습니다.

1. Amazon EC2 용량 예약 생성 사용 설명서를 사용하여 대상 온디맨드 용량 예약(ODCR)을 생성합니다.
2. ODCR을 시작 템플릿과 연결합니다. 이를 수행하는 방법에는 두 가지가 있습니다.

    a. 직접 ODCR 연결: 시작 템플릿에서 ODCR ID를 직접 참조합니다. 이 접근 방식은 엄격한 용량 제어를 제공하며 인스턴스 채우기를 지원하지 않습니다(컴퓨팅 노드 그룹이 ODCR에서 사용할 수 있는 것보다 많은 인스턴스를 요청하는 경우 추가 인스턴스가 시작되지 않음).

    b. 용량 예약 그룹 연결: 용량 예약 그룹에 ODCR을 추가하고 시작 템플릿에서 그룹을 참조합니다. 이 접근 방식은 인스턴스 채우기를 지원하므로 예약 용량이 초과되면 AWS PCS가 추가 온디맨드 인스턴스를 시작할 수 있습니다.
3. 시작 템플릿을 사용하도록 AWS PCS 컴퓨팅 노드 그룹을 생성하거나 업데이트합니다. 자세한 내용은 AWS PCS 컴퓨팅 노드 그룹 사용 설명서를 참조하세요.

    - 컴퓨팅 노드 그룹의 `purchaseOption`를 로 설정합니다`ONDEMAND`.

예: 대상 ODCR을 사용하여 hpc6a.48xlarge 인스턴스 예약 및 사용

이 예제 명령은 32hpc6a.48xlarge 인스턴스에 대한 대상 ODCR을 생성합니다. 배치 그룹에서 예약 인스턴스를 시작하려면 명령에 `--placement-group-arn`를 추가합니다. `--end-date` 및를 사용하여 중지 날짜를 정의할 수 있습니다. `--end-date-type`그렇지 않으면 수동으로 종료될 때까지 예약이 계속됩니다.

```
aws ec2 create-capacity-reservation \
    --instance-type hpc6a.48xlarge \
    --instance-platform Linux/UNIX \
    --availability-zone us-east-2a \
    --instance-count 32 \
    --instance-match-criteria targeted
```

이 명령의 결과는 새 ODCR의 ARN이 됩니다. ODCR ID는 ARN에서 검색`"arn:aws:ec2:us-east-2:123456789012:capacity-reservation/ODCR-ID"`하거나 Amazon EC2 DescribeCapacityReservations를 사용하여 검색할 수 있습니다.

직접 ODCR 연결: 시작 템플릿에 ODCR ID를 추가합니다. 다음은 ODCR ID를 참조하는 시작 템플릿의 예입니다.

```
{
  "CapacityReservationSpecification": {
    "CapacityReservationTarget": {
      "CapacityReservationId": "cr-1234567890abcdef1"
    }
  }
}
```

용량 예약 그룹 연결: 용량 예약 그룹을 생성하고 시작 템플릿에 그룹을 추가합니다. 다음 명령은 라는 용량 예약 그룹을 생성합니다EXAMPLE-CR-GROUP.

```
aws resource-groups create-group \
    --name EXAMPLE-CR-GROUP \
    --configuration \
        '{"Type": "AWS::EC2::CapacityReservationPool"}' \
        '{"Type": "AWS::ResourceGroups::Generic", "Parameters": [{"Name": "allowed-
resource-types", "Values": ["AWS::EC2::CapacityReservation"]}]}'
```

다음 명령은 용량 예약 그룹에 ODCR을 추가합니다.

```
aws resource-groups group-resources --group EXAMPLE-CR-GROUP \
    --resource-arns arn:aws:ec2:us-east-2:123456789012:capacity-reservation/
cr-1234567890abcdef1
```

ODCR을 생성하여 용량 예약 그룹에 추가하면 이제 시작 템플릿에 추가하여 AWS PCS 컴퓨팅 노드 그룹에 연결할 수 있습니다. 다음은 용량 예약 그룹을 참조하는 시작 템플릿의 예입니다.

```
{
  "CapacityReservationSpecification": {
    "CapacityReservationResourceGroupArn": "arn:aws:resource-groups:us-
east-2:123456789012:group/EXAMPLE-CR-GROUP"
  }
}
```

마지막으로 hpc6a.48xlarge 인스턴스를 사용하고 ODCR을 참조하는 시작 템플릿을 사용하도록 AWS PCS 컴퓨팅 노드 그룹을 생성하거나 업데이트합니다. 정적 노드 그룹의 경우 최소 및 최대 인스턴스를 예약 크기(32)로 설정합니다. 동적 노드 그룹의 경우 최소 인스턴스를 0으로 설정하고 최대 인스턴스를 원하는 인스턴스 크기로 설정합니다.

이 예제는 하나의 컴퓨팅 노드 그룹에 프로비저닝된 단일 ODCR의 간단한 구현입니다. 그러나 AWS PCS는 다른 많은 설계를 지원합니다. 예를 들어 대규모 ODCR 또는 용량 예약 그룹을 여러 컴퓨팅 노드 그룹으로 세분화할 수 있습니다. 또는 다른 AWS 계정이 생성하여 사용자와 공유한 ODCRs을 사용할 수 있습니다.

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 온디맨드 용량 예약 및 ML용 용량 블록을 참조하세요.

AWS PCS에서 ML에 Amazon EC2 용량 블록 사용

ML용 Amazon EC2 용량 블록은 특정 날짜 및 시간 범위 내에 GPU 기반 가속 컴퓨팅 인스턴스를 미리 예약하여 단기 워크로드를 지원할 수 있는 Amazon EC2 구매 옵션입니다. 용량 블록 내부에서 실행되는 인스턴스는 지연 시간이 짧은 페타비트 규모의 비차단 네트워킹을 위해 Amazon EC2 UltraClusters 내부에 자동으로 서로 가깝게 배치됩니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 ML용 용량 블록을 참조하세요.

시작 템플릿을 사용하여 컴퓨팅 노드 그룹의 인스턴스를 시작할 때 AWS PCS가 용량 블록을 사용하도록 할 수 있습니다.

> **Note**
>
> AWS PCS는 Slurm 버전 24.05 이후 용량 블록에 대한 지원을 도입했습니다.

제한 사항

- AWS PCS는 P6-B300, P6-B200, P5en, P5e, P5 및 P4d 인스턴스 패밀리가 있는 용량 블록만 지원합니다.
- 컴퓨팅 노드 그룹은 한 번에 하나의 용량 블록에만 연결할 수 있습니다.
- 컴퓨팅 노드 그룹을 여러 용량 블록을 결합하는 용량 예약 그룹과 연결할 수 없습니다.
- AWS PCS와 함께 사용하려면 용량 블록이 `scheduled` 또는 `active` 상태여야 합니다. 와 같은 다른 상태에서는 용량 블록을 사용할 수 없습니다`payment-failed`. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 용량 블록 보기를 참조하세요.
- P6 및 P5 인스턴스 유형의 경우 관련 AWS 설명서: P6 인스턴스의 소프트웨어 요구 사항, 여러 네트워크 카드를 사용하여 Amazon EC2 인스턴스의 네트워크 대역폭 최대화를 참조하세요.

용량 블록 만료

용량 블록은 특정 날짜 및 시간 범위로 제한됩니다. 용량 블록이 만료되는 경우:

- 해당 용량 블록과 연결된 컴퓨팅 노드 그룹은 계속 존재하며 동일한 대기열과 연결된 상태로 유지됩니다.
- Slurm 설정에 따라 컴퓨팅 노드 그룹의 모든 인스턴스가 종료되고 활성 작업이 실패할 수 있습니다.
- AWS PCS는 컴퓨팅 노드 그룹에서 새 인스턴스를 시작할 수 없습니다.
- 대기열에 다른 컴퓨팅 노드 그룹이 연결되거나 새 용량 블록을 지정하는 새 시작 템플릿을 사용하도록 컴퓨팅 노드 그룹을 업데이트할 때까지 대기 중이거나 새로 제출된 모든 작업은 보류 상태로 유지됩니다.

용량 블록을 사용하도록 AWS PCS 컴퓨팅 노드 그룹 구성

용량 블록을 컴퓨팅 노드 그룹과 연결하려면

1. 용량 블록을 지정하는 AWS PCS용 Amazon EC2 시작 템플릿을 생성합니다. AWS PCS용 시작 템플릿 생성에 대한 자세한 내용은 섹션을 참조하세요AWS PCS에서 Amazon EC2 시작 템플릿 사용.

   시작 템플릿에는 다음이 포함되어야 합니다.

   - `MarketType`의 값은 로 설정해야 `InstanceMarketOptions` 합니다`capacity-block`.
   - `CapacityReservationSpecification`가 유효한 `CapacityReservationId`
   - 구매한 용량 블록의 인스턴스 유형`InstanceType`과 일치하는 유효한 입니다.
2. 시작 템플릿을 사용하는 컴퓨팅 노드 그룹을 생성합니다. 자세한 내용은 AWS PCS에서 컴퓨팅 노드 그룹 생성 단원을 참조하십시오. 시작 템플릿을 사용하도록 기존 컴퓨팅 노드 그룹을 업데이트할 수도 있습니다. 자세한 내용은 AWS PCS 컴퓨팅 노드 그룹 업데이트 단원을 참조하십시오.

   컴퓨팅 노드 그룹을 생성하거나 업데이트할 때:

   - 컴퓨팅 노드 그룹을 생성하거나 업데이트하는 데 사용하는 IAM 자격 증명에는 다음 권한이 있어야 합니다.

     ```
     ec2:DescribeCapacityReservations
     ```

     자세한 내용은 AWS PCS에 대한 최소 권한 단원을 참조하십시오.

- 용량 블록은 `scheduled` 또는 `active` 상태여야 합니다.
- 컴퓨팅 노드 그룹의 `purchaseOption`를 로 설정합니다`CAPACITY_BLOCK`.
- `maxInstanceCount` 컴퓨팅 노드 그룹의는 용량 블록의 크기를 초과해서는 안 됩니다.
- 컴퓨팅 노드 그룹의 가용 영역은 컴퓨팅 노드 그룹의 서브넷 가용 영역 중 하나와 일치해야 합니다.

> **Important**
>
> 컴퓨팅 노드 그룹을 업데이트할 때는 해당 인스턴스 유형을 변경할 수 없습니다. 컴퓨팅 노드 그룹과 동일한 인스턴스 유형의 용량 블록만 사용할 수 있습니다. 다른 인스턴스 유형의 용량 블록을 사용하려면 새 컴퓨팅 노드 그룹을 생성해야 합니다.

AWS PCS에서 용량 블록 사용에 대해 자주 묻는 질문

방금 용량 블록 비용을 지불하고 AWS PCS에서 즉시 사용하려고 했지만 컴퓨팅 노드 그룹 생성에 실패했습니다. 어떻게 된 걸까요?

용량 블록이 `scheduled` 또는 `active` 상태가 아닐 수 있습니다. 용량 블록이 `scheduled` 또는가 된 후 다시 시도하세요`active`.

AWS PCS에서 용량 블록을 사용하고 있는데 익스텐션이 만료되기 전에 구입했습니다. AWS PCS에서 계속 사용하려면 어떻게 해야 하나요?

AWS PCS에서 용량 블록을 계속 사용하기 위해 아무 작업도 수행할 필요가 없습니다. 확장 결제가 성공한 후 용량 블록의 종료 날짜가 업데이트됩니다. 용량 블록이 만료되지 않는 한 컴퓨팅 노드 그룹은 계속 작동합니다. 확장 결제에 실패하면 용량 블록은 유지`active`되고 컴퓨팅 노드 그룹은 원래 종료 날짜에 용량 블록이 만료될 때까지 작동합니다.

용량 블록이 만료되면 대기 중인 작업과 실행 중인 작업은 어떻게 되나요?

용량 블록이 만료되기 전에 시작되지 않은 대기 중인 작업은 다른 컴퓨팅 노드 그룹을 대기열에 연결하거나 컴퓨팅 노드 그룹을 새 용량 블록으로 업데이트할 때까지 보류 상태로 유지됩니다. 여전히 대기열에 작업을 제출할 수 있습니다. Slurm 설정은 활성 작업에 영향을 미칩니다. 기본적으로 활성 작업은 자동으로 다시 대기열에 추가되지만 오류가 발생하거나 실패할 수 있습니다.

내 용량 블록이 만료되었습니다. 어떻게 해야 하나요?

아무 작업도 할 필요가 없습니다. Amazon EC2 콘솔에서 EC2 용량 예약 상태를 확인할 수 있습니다. 용량 블록이 만료되면 해당 용량 블록과 연결된 컴퓨팅 노드 그룹이 계속 존재하고 동일한 대기

열을 처리합니다. 컴퓨팅 노드 그룹에는 작업을 실행할 인스턴스가 없습니다. 사용자가 실행되지 않는 작업을 제출하지 못하도록 컴퓨팅 노드 그룹을 삭제하거나 대기열에서 연결을 해제할 수 있습니다.

새 용량 블록을 AWS PCS 컴퓨팅 노드 그룹에 사용하려고 합니다. 어떻게 해야 합니까?

새 용량 블록을 사용하려면 새 컴퓨팅 노드 그룹을 생성하는 것이 좋습니다. 자세한 내용은 용량 블록을 사용하도록 AWS PCS 컴퓨팅 노드 그룹 구성 단원을 참조하십시오.

클러스터와 서비스 간에 용량 블록 1개를 공유하려면 어떻게 해야 하나요?

용량 블록을 여러 클러스터와 서비스로 분할할 수 있습니다. 예를 들어, PCS-Cluster-1의 노드가 20개, PCS-Cluster-2의 노드가 16개PCS-Cluster-2, 다른 서비스의 나머지 노드가 있는 `p5.48xlarge` 인스턴스 64개로 용량 블록을 분할하려면 PCS-Cluster-1의 경우 `minInstanceCount` 및 `maxInstanceCount`를 모두 20으로 설정하고 PCS-Cluster-2의 경우 16으로 설정합니다.

용량 블록을 1개 이상 사용하거나 컴퓨팅 노드 그룹을 1개 이상 사용할 수 있나요?

아니요. 용량 블록 1개만 단일 컴퓨팅 노드 그룹과 연결할 수 있습니다. AWS PCS는 여러 용량 블록을 결합하는 용량 예약 그룹을 지원하지 않습니다.

용량 블록이 시작되거나 만료되는 시기를 어떻게 알 수 있나요?

AWS PCS와 독립적으로 Amazon EC2는 용량 블록 예약이 시작될 때 EventBridge를 통해 `Capacity Block Reservation Delivered` 이벤트를 전송하고 용량 블록 예약이 만료되기 40분 전에 `Capacity Block Reservation Expiration Warning` 이벤트를 전송합니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 EventBridge를 사용하여 용량 블록 모니터링을 참조하세요.

Slurm은 용량 블록의 상태를 어떻게 추적하나요?

를 실행`sinfo`하여 AWS PCS가 용량 블록을 사용하는 방법을 이해할 수 있습니다. 다음 예제 출력에서 대기열은 `active` 용량 블록에서 4개의 인스턴스를 실행하는 컴퓨팅 노드 그룹과 연결됩니다. 노드는 `idle` Slurm 상태(사용 가능하지만 아직 작업에 할당되지 않음)입니다.

```
$ sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
fanout up infinite 4 idle node-fanout-[1-4]
```

노드가 대신 `maint` 상태인 경우를 실행`scontrol show res`하여이 상태를 제어하는 Slurm 예약에 대한 세부 정보를 볼 수 있습니다. 다음 예제 출력에서 용량 블록은 미래 시작 날짜를 `scheduled` 갖습니다.

```
$ scontrol show res

ReservationName=node-fanout-scheduled StartTime=2025-10-14T13:09:17
 EndTime=2025-10-14T13:11:17 Duration=00:02:00
   Nodes=node-fanout-[1-4] NodeCnt=4 CoreCnt=16 Features=(null) PartitionName=(null)
 Flags=MAINT,SPEC_NODES
   TRES=cpu=16

   Users=root Groups=(null) Accounts=(null) Licenses=(null) State=ACTIVE
 BurstBuffer=(null)
   MaxStartDelay=(null)

   Comment=node-fanout Scheduled
```

용량 블록이 공유되기 때문에 용량을 시작하는 동안 발생하는 오류가 인지 어떻게 알 수 있습니까?

Amazon EC2 콘솔에서 용량 예약을 확인하여 용량 블록에서 활발하게 프로비저닝된 인스턴스 수를 확인합니다. 각 인스턴스의 태그를 확인하여 인스턴스를 사용하는 서비스 또는 클러스터를 찾습니다. 예를 들어 AWS , PCS의 모든 인스턴스에는 인스턴스`aws:pcs:cluster-id = pcs_l0mizqyk5o | aws:pcs:compute-node-group-id = pcs_ic7onkmfqk`가 속한 클러스터 및 컴퓨팅 노드 그룹을 나타내는와 같은 AWS PCS 태그가 있습니다. 그런 다음 용량 블록이 최대 용량인지 확인할 수 있습니다.

`scontrol show nodes`를 사용하여 AWS PCS 클러스터의 용량 블록 노드가를 트리거하고 있는지 확인합니다`ReservationCapacityExceeded`.

```
[root@ip-172-16-10-54 ~]# scontrol show nodes test-node-8-gamma-cb-2
NodeName=test-8-gamma-cb-2 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=8 CPUTot=8 CPULoad=0.00
   AvailableFeatures=test-8-gamma-cb,gpu
   ActiveFeatures=test-8-gamma-cb,gpu
   Gres=gpu:H100:1
   NodeAddr=test-8-gamma-cb-2 NodeHostName=test-8-gamma-cb-2
   RealMemory=249036 AllocMem=0 FreeMem=N/A Sockets=8 Boards=1
   State=IDLE+CLOUD+POWERING_DOWN ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A
 MCS_label=N/A
   Partitions=my-q
   BootTime=None SlurmdStartTime=None
   LastBusyTime=Unknown ResumeAfterTime=None
   CfgTRES=cpu=8,mem=249036M,billing=8
   AllocTRES=
   CurrentWatts=0 AveWatts=0
```

```
   Reason=Failed to launch backing instance (Error Code:
 ReservationCapacityExceeded) [root@2025-08-28T15:15:33]
```

여러 컴퓨팅 노드 그룹이 동일한 대기열에 연결된 경우 용량 블록 지원 인스턴스에서 작업을 강제로 실행하려면 어떻게 해야 합니까?

Slurm 기능 및 제약 조건을 사용하여 작업을 특정 노드 세트에 잠글 수 있습니다. 상태가 아닌 노드에서만 작동하므로 각 컴퓨팅 노드 그룹에 대해 Slurm 가중치를 설정하지 않는 것이 좋습니다`maint`.

유용한 시작 템플릿 파라미터

이 섹션에서는 AWS PCS에서 광범위하게 유용할 수 있는 몇 가지 시작 템플릿 파라미터에 대해 설명합니다.

자세한 CloudWatch 모니터링 켜기

시작 템플릿 파라미터를 사용하여 더 짧은 간격으로 CloudWatch 지표 수집을 활성화할 수 있습니다.

AWS Management Console

시작 템플릿을 생성하거나 편집하기 위한 콘솔 페이지의 고급 세부 정보 섹션에서이 옵션을 찾을 수 있습니다. 세부 CloudWatch 모니터링을 활성화로 설정합니다.

YAML

```
Monitoring:
    Enabled: True
```

JSON

```
{"Monitoring": {"Enabled": "True"}}
```

자세한 내용은 Linux 인스턴스용 Amazon Elastic Compute Cloud 사용 설명서의 인스턴스에 대한 세부 모니터링 활성화 또는 비활성화를 참조하세요.

인스턴스 메타데이터 서비스 버전 2(IMDS v2)

EC2 인스턴스와 함께 IMDS v2를 사용하면 보안이 크게 향상되고 AWS 환경의 인스턴스 메타데이터 액세스와 관련된 잠재적 위험을 완화할 수 있습니다.

AWS Management Console

시작 템플릿을 생성하거나 편집하기 위한 콘솔 페이지의 고급 세부 정보 섹션에서이 옵션을 찾을 수 있습니다. 액세스 가능한 메타데이터를 활성화됨으로 설정하고, 메타데이터 버전을 V2 전용(토큰 필요)으로 설정하고, 메타데이터 응답 홉 제한을 4로 설정합니다.

YAML

```
MetadataOptions:
  HttpEndpoint: enabled
  HttpTokens: required
  HttpPutResponseHopLimit: 4
```

JSON

```
{
    "MetadataOptions": {
        "HttpEndpoint": "enabled",
        "HttpPutResponseHopLimit": 4,
        "HttpTokens": "required"
    }
}
```

.

AWS PCS 대기열

AWS PCS 대기열은 스케줄러의 기본 작업 대기열 구현에 대한 간단한 추상화입니다. Slurm의 경우 AWS PCS 대기열은 Slurm 파티션과 동일합니다.

사용자는 하나 이상의 컴퓨팅 노드 그룹에서 제공하는 노드에서 실행되도록 예약할 수 있을 때까지 작업이 있는 대기열에 작업을 제출합니다. AWS PCS 클러스터에는 여러 작업 대기열이 있을 수 있습니다. 예를 들어 우선 순위가 높은 작업에 Amazon EC2 온디맨드 인스턴스를 사용하는 대기열과 우선 순위가 낮은 작업에 Amazon EC2 스팟 인스턴스를 사용하는 다른 대기열을 생성할 수 있습니다.

주제

AWS PCS에서 대기열 생성

이 주제에서는 사용 가능한 옵션에 대한 개요를 제공하고 AWS PCS에서 대기열을 생성할 때 고려해야 할 사항을 설명합니다.

> **Note**
>
> 대기열에서 사용자 지정 Slurm 설정을 구성하여 파티션별 예약 정책 및 리소스 관리를 구현할 수 있습니다. 자세한 내용은 AWS PCS에서 사용자 지정 Slurm 설정 구성 단원을 참조하십시오.

사전 조건

- AWS PCS 클러스터 - 대기열은 특정 AWS PCS 클러스터와 연결된 경우에만 생성할 수 있습니다.
- 하나 이상의 AWS PCS 컴퓨팅 노드 그룹 - 대기열은 하나 이상의 AWS PCS 컴퓨팅 노드 그룹과 연결되어야 합니다.

AWS PCS에서 대기열을 생성하려면

AWS Management Console 또는를 사용하여 대기열을 생성할 수 있습니다 AWS CLI.

AWS Management Console

콘솔을 사용하여 대기열을 생성하려면

1. AWS PCS 콘솔을 엽니다.
2. 대기열의 클러스터를 선택합니다. 대기열로 이동하여 대기열 생성을 선택합니다.
3. 대기열 구성 섹션에서 다음 값을 제공합니다.
   a. 대기열 이름 - 대기열의 이름입니다. 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하며 25자를 초과할 수 없습니다. 이름은 클러스터 내에서 고유해야 합니다.
   b. 컴퓨팅 노드 그룹 -이 대기열에 서비스를 제공할 컴퓨팅 노드 그룹을 하나 이상 선택합니다. 컴퓨팅 노드 그룹은 둘 이상의 대기열과 연결할 수 있습니다.
4. (선택 사항) 추가 스케줄러 설정 섹션에서 파라미터 이름과 값 페어를 추가하여 추가 Slurm 설정을 구성할 수 있습니다. 지원되는 파라미터의 전체 목록은 섹션을 참조하세요AWS PCS 대기열에 대한 사용자 지정 Slurm 설정.
5. (선택 사항) 태그에서 AWS PCS 대기열에 태그를 추가합니다.
6. 대기열 생성을 선택합니다. 상태 필드에는 AWS PCS가 대기열을 생성하는 동안 생성 중이 표시됩니다. 대기열 생성에는 몇 분 정도 걸릴 수 있습니다.

권장 다음 단계

- 새 대기열에 작업을 제출합니다.

AWS CLI

를 사용하여 대기열을 생성하려면 AWS CLI

다음 명령을 사용하여 대기열을 생성합니다. 다음과 같이 교체합니다.

1. *region-code*를 클러스터의 AWS 리전으로 바꿉니다. 예를 들어 `us-east-1`입니다.
2. *my-queue*를 대기열의 이름으로 바꿉니다. 이름에는 영숫자(대소문자 구분)와 하이픈만 사용할 수 있습니다. 영문자로 시작해야 하며 25자를 초과할 수 없습니다. 이름은 클러스터 내에서 고유해야 합니다.
3. *my-cluster*를 클러스터의 이름 또는 ID로 바꿉니다.

4. *compute-node-group-id*를 대기열에 서비스할 컴퓨팅 노드 그룹의 ID로 바꿉니다. 예를 들어 pcs_abcdef12345입니다.

> **Note**
>
> 대기열을 생성할 때 이름이 아닌 컴퓨팅 노드 그룹의 ID를 제공해야 합니다.

```
aws pcs create-queue --region region-code \
    --queue-name my-queue \
    --cluster-identifier my-cluster \
    --compute-node-group-configurations \
    computeNodeGroupId=compute-node-group-id
```

Example- 사용자 지정 Slurm 설정을 사용하여 대기열 생성

```
aws pcs create-queue --region region-code \
    --queue-name my-queue \
    --cluster-identifier my-cluster \
    --compute-node-group-configurations \
    computeNodeGroupId=compute-node-group-id \
    --slurm-configuration \
    'slurmCustomSettings=[{parameterName=Default,parameterValue=YES}]'
```

자세한 내용은 AWS PCS 대기열에 대한 사용자 지정 Slurm 설정 단원을 참조하십시오.

대기열을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 다음 명령을 사용하여 대기열의 상태를 쿼리할 수 있습니다. 상태가에 도달할 때까지 대기열에 작업을 제출할 수 없습니다ACTIVE.

```
aws pcs get-queue --region region-code \
    --cluster-identifier my-cluster \
    --queue-identifier my-queue
```

권장 다음 단계

- 새 대기열에 작업 제출

AWS PCS 대기열 업데이트

이 주제에서는 사용 가능한 옵션에 대한 개요를 제공하고 AWS PCS 대기열을 업데이트할 때 고려해야 할 사항을 설명합니다. Slurm 사용자 지정 설정에 대한 자세한 내용은 섹션을 참조하세요AWS PCS 대기열에 대한 사용자 지정 Slurm 설정.

AWS PCS 대기열 업데이트 시 고려 사항

대기열 업데이트는 실행 중인 작업에 영향을 주지 않지만 대기열이 업데이트되는 동안 클러스터가 새 작업을 수락하지 못할 수 있습니다.

AWS PCS 대기열을 업데이트하려면

AWS Management Console 또는를 사용하여 대기열 AWS CLI 을 업데이트할 수 있습니다.

AWS Management Console

대기열을 업데이트하려면

1. 에서 AWS PCS 콘솔을 엽니다. `https://console.aws.amazon.com/pcs/home#/clusters`
2. 대기열을 업데이트할 클러스터를 선택합니다.
3. 대기열로 이동하여 업데이트하려는 대기열로 이동한 다음 편집을 선택합니다.
4. 대기열 구성 섹션에서 다음 값 중 하나를 업데이트합니다.
   - 노드 그룹 - 대기열과의 연결에서 컴퓨팅 노드 그룹을 추가하거나 제거합니다.
   - 추가 스케줄러 설정 - 대기열에 대한 사용자 지정 Slurm 설정을 추가, 수정 또는 제거합니다. 자세한 내용은 AWS PCS 대기열에 대한 사용자 지정 Slurm 설정 단원을 참조하십시오.
   - 태그 - 대기열에 대한 태그를 추가하거나 제거합니다.
5. 업데이트를 선택합니다. 변경 사항이 적용되는 동안 상태 필드에 업데이트 중이 표시됩니다.

> **Important**
> 대기열 업데이트에는 몇 분 정도 걸릴 수 있습니다.

AWS CLI

대기열을 업데이트하려면

1. 다음 명령을 사용하여 대기열을 업데이트합니다. 명령을 실행하기 전에 다음과 같은 바꾸기를 합니다.

   a. *region-code*를 클러스터를 생성할 AWS 리전 로 바꿉니다.

   b. *my-queue*를 대기열의 이름 또는 computeNodeGroupId 로 바꿉니다.

   c. *my-cluster*를 클러스터의 이름 또는 로 바꿉clusterId니다.

   d. 컴퓨팅 노드 그룹 연결을 변경하려면에 대한 업데이트된 목록을 제공합니다--compute-node-group-configurations.

      - 예를 들어 두 번째 컴퓨팅 노드 그룹를 추가하려면computeNodeGroupExampleID2:

        ```
        --compute-node-group-configurations
         computeNodeGroupId=computeNodeGroupExampleID1,computeNodeGroupId=computeNodeGro
        ```

   ```
   aws pcs update-queue --region region-code \
       --queue-identifier my-queue \
       --cluster-identifier my-cluster \
       --compute-node-group-configurations \
       computeNodeGroupId=computeNodeGroupExampleID1
   ```

   Example- 사용자 지정 Slurm 설정으로 대기열 업데이트

   ```
   aws pcs update-queue --region region-code \
       --queue-identifier my-queue \
       --cluster-identifier my-cluster \
       --slurm-configuration \
       'slurmCustomSettings=[{parameterName=Default,parameterValue=YES}]'
   ```

   자세한 내용은 AWS PCS 대기열에 대한 사용자 지정 Slurm 설정 단원을 참조하십시오.

2. 대기열을 업데이트하는 데 몇 분 정도 걸릴 수 있습니다. 다음 명령을 사용하여 대기열의 상태를 쿼리할 수 있습니다. 상태가에 도달할 때까지 대기열에 작업을 제출할 수 없습니다ACTIVE.

   ```
   aws pcs get-queue --region region-code \
   ```

```
    --cluster-identifier my-cluster \
    --queue-identifier my-queue
```

권장 다음 단계

- 업데이트된 대기열에 작업을 제출합니다.

AWS PCS에서 대기열 삭제

이 주제에서는 AWS PCS에서 대기열을 삭제하는 방법에 대한 개요를 제공합니다.

대기열 삭제 시 고려 사항

- 대기열에 실행 중인 작업이 있는 경우 대기열이 삭제되면 스케줄러에 의해 작업이 종료됩니다. 대기열에서 보류 중인 작업은 취소됩니다. 스케줄러의 기본 명령(예: Slurm`scancel`용)을 사용하여 대기열의 작업이 완료되거나 수동으로 중지/취소될 때까지 기다리는 것이 좋습니다.

대기열 삭제

AWS Management Console 또는를 사용하여 대기열 AWS CLI 을 삭제할 수 있습니다.

AWS Management Console

대기열 삭제

1. AWS PCS 콘솔을 엽니다.
2. 대기열의 클러스터를 선택합니다.
3. 대기열로 이동하여 삭제할 대기열을 선택합니다.
4. 삭제를 선택합니다.
5. 상태 필드에가 표시됩니다`Deleting`. 완료되는 데 몇 분 정도 걸릴 수 있습니다.

> **Note**
>
> 스케줄러의 기본 명령을 사용하여 대기열이 삭제되었는지 확인할 수 있습니다. 예를 들어 Slurm의 경우 `sinfo` 또는 `squeue`를 사용합니다.

AWS CLI

대기열 삭제

- 다음 명령을 사용하여 대기열을 삭제하고 다음과 같이 바꿉니다.
  - *region-code*를 클러스터가 있는 로 바꿉 AWS 리전 니다.
  - *my-queue*를 대기열의 이름 또는 ID로 바꿉니다.
  - *my-cluster*를 클러스터의 이름 또는 ID로 바꿉니다.

```
aws pcs delete-queue --region region-code \
       --queue-identifier my-queue \
       --cluster-identifier my-cluster
```

대기열을 삭제하는 데 몇 분 정도 걸릴 수 있습니다.

> **Note**
>
> 스케줄러의 기본 명령을 사용하여 대기열이 삭제되었는지 확인할 수 있습니다. 예를 들어 Slurm의 경우 `sinfo` 또는 `squeue`를 사용합니다.

AWS PCS 로그인 노드

대화형 액세스 및 작업 관리를 지원하려면 일반적으로 AWS PCS 클러스터에 하나 이상의 로그인 노드가 필요합니다. 이를 수행하는 방법은 로그인 노드 기능을 위해 구성된 정적 AWS PCS 컴퓨팅 노드 그룹을 사용하는 것입니다. 로그인 노드 역할을 하도록 독립 실행형 EC2 인스턴스를 구성할 수도 있습니다.

주제

AWS PCS 컴퓨팅 노드 그룹을 사용하여 로그인 노드 제공

이 주제에서는 제안된 구성 옵션에 대한 개요를 제공하고 AWS PCS 컴퓨팅 노드 그룹을 사용하여 클러스터에 대한 지속적인 대화형 액세스를 제공할 때 고려해야 할 사항을 설명합니다.

로그인 노드에 대한 AWS PCS 컴퓨팅 노드 그룹 생성

운영상 이는 일반 컴퓨팅 노드 그룹을 생성하는 것과 크게 다르지 않습니다. 그러나 몇 가지 주요 구성 선택 사항이 있습니다.

- 컴퓨팅 노드 그룹에서 하나 이상의 EC2 인스턴스에 대한 정적 조정 구성을 설정합니다.
- 온디맨드 구매 옵션을 선택하여 인스턴스(들)를 회수하지 않도록 합니다.
- 로그인과 같은 컴퓨팅 노드 그룹의 정보 이름을 선택합니다.
- VPC 외부에서 로그인 노드 인스턴스(들)에 액세스하려면 퍼블릭 서브넷을 사용하는 것이 좋습니다.
- SSH 액세스를 허용하려는 경우 시작 템플릿에는 SSH 포트를 선택한 IP 주소에 노출하는 보안 그룹이 있어야 합니다.
- IAM 인스턴스 프로파일에는 최종 사용자에게 부여할 AWS 권한만 있어야 합니다. 세부 정보는 AWS 병렬 컴퓨팅 서비스를 위한 IAM 인스턴스 프로파일 섹션을 참조하세요.
- AWS Systems Manager Session Manager가 로그인 인스턴스를 관리하도록 허용하는 것이 좋습니다.
- 인스턴스 AWS 자격 증명에 대한 액세스를 관리 사용자로만 제한하는 것이 좋습니다.

- 로그인 노드(들)가 지속적으로 실행되므로 일반 컴퓨팅 노드 그룹보다 저렴한 인스턴스 유형을 선택합니다.
- 모든 인스턴스에 동일한 소프트웨어가 설치되어 있는지 확인하려면 다른 컴퓨팅 노드 그룹과 동일한(또는 파생) AMI를 사용합니다. AMIs. AWS PCS용 Amazon Machine Image(AMIs)
- 로그인 노드에서 컴퓨팅 인스턴스와 동일한 네트워크 파일 시스템(Amazon EFS, Amazon FSx for Lustre 등) 탑재를 구성합니다. 자세한 내용은 AWS PCS에서 네트워크 파일 시스템 사용 단원을 참조하십시오.

로그인 노드에 액세스

새 컴퓨팅 노드 그룹이 ACTIVE 상태에 도달하면 생성한 EC2 인스턴스(들)를 찾아 로그인할 수 있습니다. 자세한 내용은 AWS PCS에서 컴퓨팅 노드 그룹 인스턴스 찾기 단원을 참조하십시오.

로그인 노드에 대한 AWS PCS 컴퓨팅 노드 그룹 업데이트

UpdateComputeNodeGroup을 사용하여 로그인 노드 그룹을 업데이트할 수 있습니다. 노드 그룹 업데이트 프로세스의 일부로 실행 중인 인스턴스가 교체됩니다. 이렇게 하면 인스턴스의 활성 사용자 세션 또는 프로세스가 중단됩니다. 실행 중이거나 대기 중인 Slurm 작업은 영향을 받지 않습니다. 자세한 내용은 AWS PCS 컴퓨팅 노드 그룹 업데이트 단원을 참조하십시오.

컴퓨팅 노드 그룹에서 사용하는 시작 템플릿을 편집할 수도 있습니다. UpdateComputeNodeGroup을 사용하여 업데이트된 시작 템플릿을 컴퓨팅 노드 그룹에 적용해야 합니다. 컴퓨팅 노드 그룹에서 시작된 새 EC2 인스턴스는 업데이트된 시작 템플릿을 사용합니다. 자세한 내용은 AWS PCS에서 Amazon EC2 시작 템플릿 사용 단원을 참조하십시오.

로그인 노드에 대한 AWS PCS 컴퓨팅 노드 그룹 삭제

AWS PCS에서 컴퓨팅 노드 그룹 삭제 메커니즘을 사용하여 로그인 노드 그룹을 업데이트할 수 있습니다. 실행 중인 인스턴스는 노드 그룹 삭제의 일부로 종료됩니다. 이렇게 하면 인스턴스의 활성 사용자 세션 또는 프로세스가 중단됩니다. 실행 중이거나 대기 중인 Slurm 작업은 영향을 받지 않습니다. 자세한 내용은 AWS PCS에서 컴퓨팅 노드 그룹 삭제 단원을 참조하십시오.

독립 실행형 인스턴스를 AWS PCS 로그인 노드로 사용

AWS PCS 클러스터의 Slurm 스케줄러와 상호 작용하도록 독립 EC2 인스턴스를 설정할 수 있습니다. 이는 AWS PCS 클러스터에서 작동하지만 AWS PCS 관리 외부에서 작동하는 로그인 노드, 워크스테이션 또는 전용 워크플로 관리 호스트를 생성하는 데 유용합니다. 이렇게 하려면 각 독립 실행형 인스턴스가 다음을 수행해야 합니다.

1. 호환되는 Slurm 소프트웨어 버전이 설치되어 있어야 합니다.
2. AWS PCS 클러스터의 Slurmctld 엔드포인트에 연결할 수 있어야 합니다.
3. Slurm Auth 및 Cred Kiosk Daemon(sackd)이 AWS PCS 클러스터의 엔드포인트 및 보안 암호로 올바르게 구성되도록 합니다. 자세한 내용은 Slurm 설명서의 sackd를 참조하세요.

이 자습서는 AWS PCS 클러스터에 연결하는 독립 인스턴스를 구성하는 데 도움이 됩니다.

목차

1단계 - 대상 AWS PCS 클러스터의 주소 및 보안 암호 검색

다음 명령과 AWS CLI 함께를 사용하여 대상 AWS PCS 클러스터에 대한 세부 정보를 검색합니다. 명령을 실행하기 전에 다음과 같은 바꾸기를 합니다.

- *region-code*를 대상 클러스터가 실행 AWS 리전 중인 로 바꿉니다.
- *cluster-ident*를 대상 클러스터의 이름 또는 식별자로 바꿉니다.

```
aws pcs get-cluster --region region-code --cluster-identifier cluster-ident
```

명령은이 예제와 유사한 출력을 반환합니다.

```
{
    "cluster": {
        "name": "get-started",
        "id": "pcs_123456abcd",
        "arn": "arn:aws:pcs:us-east-1:111122223333:cluster/pcs_123456abcd",
        "status": "ACTIVE",
        "createdAt": "2024-12-17T21:03:52+00:00",
        "modifiedAt": "2024-12-17T21:03:52+00:00",
        "scheduler": {
```

```
            "type": "SLURM",
            "version": "25.05"
        },
        "size": "SMALL",
        "slurmConfiguration": {
            "authKey": {
                "secretArn": "arn:aws:secretsmanager:us-east-1:111122223333:secret:pcs!
slurm-secret-pcs_123456abcd-a12ABC",
                "secretVersion": "ef232370-d3e7-434c-9a87-ec35c1987f75"
            }
        },
        "networking": {
            "subnetIds": [
                "subnet-0123456789abcdef0"
            ],
            "securityGroupIds": [
                "sg-0123456789abcdef0"
            ]
        },
        "endpoints": [
            {
                "type": "SLURMCTLD",
                "privateIpAddress": "10.3.149.220",
                "port": "6817"
            }
        ]
    }
}
```

이 샘플에서 클러스터 Slurm 컨트롤러 엔드포인트의 IP 주소는 `10.3.149.220` 이고 포트에서 실행 중입니다6817. `secretArn`는 이후 단계에서 클러스터 보안 암호를 검색하는 데 사용됩니다. IP 주소와 포트는 이후 단계에서 `sackd` 서비스를 구성하는 데 사용됩니다.

2단계 - EC2 인스턴스 시작

EC2 인스턴스 시작

1. Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 Instances(인스턴스)를 선택한 다음에 Launch Instances(인스턴스 시작)를 선택하여 새 인스턴스 시작 마법사를 엽니다.
3. (선택 사항) 이름 및 태그 섹션에서와 같은 인스턴스의 이름을 제공합니다`PCS-LoginNode`. 이름은 인스턴스에 리소스 태그(`Name=PCS-LoginNode`)로 할당됩니다.

4. 애플리케이션 및 OS 이미지 섹션에서 AWS PCS에서 지원하는 운영 체제 중 하나의 AMI를 선택합니다. 자세한 내용은 지원되는 운영 체제 단원을 참조하십시오.
5. 인스턴스 유형 섹션에서 지원되는 인스턴스 유형을 선택합니다. 자세한 내용은 지원되는 인스턴스 유형 단원을 참조하십시오.
6. 키 페어 섹션에서 인스턴스에 사용할 SSH 키 페어를 선택합니다.
7. 네트워크 설정 섹션에서:
   - 편집을 선택합니다.
     i. AWS PCS 클러스터의 VPC를 선택합니다.
     ii. 방화벽(보안 그룹)에서 기존 보안 그룹 선택을 선택합니다.
       A. 인스턴스와 대상 AWS PCS 클러스터의 Slurm 컨트롤러 간의 트래픽을 허용하는 보안 그룹을 선택합니다. 자세한 내용은 보안 그룹 요구 사항 및 고려 사항 단원을 참조하십시오.
       B. (선택 사항) 인스턴스에 대한 인바운드 SSH 액세스를 허용하는 보안 그룹을 선택합니다.
8. 스토리지 섹션에서 필요에 따라 스토리지 볼륨을 구성합니다. 애플리케이션과 라이브러리를 설치하여 사용 사례를 활성화할 수 있도록 충분한 공간을 구성해야 합니다.
9. 고급에서 클러스터 보안 암호에 대한 액세스를 허용하는 IAM 역할을 선택합니다. 자세한 내용은 Slurm 클러스터 보안 암호 가져오기 단원을 참조하십시오.
10. 요약 창에서 인스턴스 시작을 선택합니다.

3단계 - 인스턴스에 Slurm 설치

인스턴스가 시작되고 활성화되면 원하는 메커니즘을 사용하여 인스턴스에 연결합니다. 에서 제공하는 Slurm 설치 관리자 AWS 를 사용하여 인스턴스에 Slurm을 설치합니다. 자세한 내용은 Slurm 설치 관리자 단원을 참조하십시오.

Slurm 설치 관리자를 다운로드하고 압축을 푼 다음 `installer.sh` 스크립트를 사용하여 Slurm을 설치합니다. 자세한 내용은 3단계 - Slurm 설치 단원을 참조하십시오.

4단계 - 클러스터 보안 암호 검색 및 저장

이 지침에는가 필요합니다 AWS CLI. 자세한 내용은 버전 2 사용 설명서의의 최신 버전 설치 또는 업데이트를 AWS CLI AWS Command Line Interface 참조하세요.

다음 명령을 사용하여 클러스터 보안 암호를 저장합니다.

- Slurm에 대한 구성 디렉터리를 생성합니다.

```
sudo mkdir -p /etc/slurm
```

- 클러스터 보안 암호를 검색, 디코딩 및 저장합니다. 이 명령을 실행하기 전에 *region-code*를 대상 클러스터가 실행 중인 리전으로 바꾸고 *secret-arn*을 1단계에서 secretArn 검색된 값으로 바꿉니다.

```
aws secretsmanager get-secret-value \
 --region region-code \
 --secret-id 'secret-arn' \
 --version-stage AWSCURRENT \
 --query 'SecretString' \
 --output text | base64 -d | sudo tee /etc/slurm/slurm.key
```

> **Warning**
>
> 다중 사용자 환경에서는 인스턴스에 액세스할 수 있는 모든 사용자가 인스턴스 메타데이터 서비스(IMDS)에 액세스할 수 있는 경우 클러스터 보안 암호를 가져올 수 있습니다. 이렇게 하면 다른 사용자를 가장할 수 있습니다. IMDS에 대한 액세스를 루트 또는 관리 사용자로만 제한하는 것이 좋습니다. 또는 인스턴스 프로파일에 의존하지 않는 다른 메커니즘을 사용하여 보안 암호를 가져오고 구성하는 것이 좋습니다.

- Slurm 키 파일에 대한 소유권 및 권한을 설정합니다.

```
sudo chmod 0600 /etc/slurm/slurm.key
sudo chown slurm:slurm /etc/slurm/slurm.key
```

> **Note**
>
> Slurm 키는 sackd 서비스가 실행되는 사용자 및 그룹이 소유해야 합니다.

5단계 - AWS PCS 클러스터에 대한 연결 구성

AWS PCS 클러스터에 대한 연결을 설정하려면 다음 단계에 따라 sackd를 시스템 서비스로 시작합니다.

> **Note**
>
> Slurm 25.05 이상을 사용하는 경우 스크립트를 사용하여 로그인 노드를 설정하여 대신 여러 클러스터에 연결할 수 있습니다. 자세한 내용은 독립 실행형 로그인 노드를 AWS PCS의 여러 클러스터에 연결 단원을 참조하십시오.

1. 다음 명령을 사용하여 sackd 서비스에 대한 환경 파일을 설정합니다. 명령을 실행하기 전에 *ip-address* 및 *port*를 1단계의 엔드포인트에서 검색된 값으로 바꿉니다.

```
sudo echo "SACKD_OPTIONS='--conf-server=ip-address:port'" > /etc/sysconfig/sackd
```

2. sackd 프로세스 관리를 위한 systemd 서비스 파일을 생성합니다.

```
sudo cat << EOF > /etc/systemd/system/sackd.service
[Unit]
Description=Slurm auth and cred kiosk daemon
After=network-online.target remote-fs.target
Wants=network-online.target
ConditionPathExists=/etc/sysconfig/sackd

[Service]
Type=notify
EnvironmentFile=/etc/sysconfig/sackd
User=slurm
Group=slurm
RuntimeDirectory=slurm
RuntimeDirectoryMode=0755
ExecStart=/opt/aws/pcs/scheduler/slurm-25.05/sbin/sackd --systemd \$SACKD_OPTIONS
ExecReload=/bin/kill -HUP \$MAINPID
KillMode=process
LimitNOFILE=131072
LimitMEMLOCK=infinity
LimitSTACK=infinity

[Install]
WantedBy=multi-user.target
EOF
```

3. sackd 서비스 파일의 소유권을 설정합니다.

```
sudo chown root:root /etc/systemd/system/sackd.service && \
```

```
    sudo chmod 0644 /etc/systemd/system/sackd.service
```

4. sackd 서비스를 활성화합니다.

```
sudo systemctl daemon-reload && sudo systemctl enable sackd
```

5. sackd 서비스를 시작합니다.

```
sudo systemctl start sackd
```

6단계 - (선택 사항) 연결 테스트

sackd 서비스가 실행 중인지 확인합니다. 샘플 출력은 다음과 같습니다. 오류가 있는 경우 일반적으로 여기에 표시됩니다.

```
[root@ip-10-3-27-112 ~]# systemctl status sackd
[x] sackd.service - Slurm auth and cred kiosk daemon
   Loaded: loaded (/etc/systemd/system/sackd.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2024-12-17 16:34:55 UTC; 8s ago
 Main PID: 9985 (sackd)
   CGroup: /system.slice/sackd.service
           ##9985 /opt/aws/pcs/scheduler/slurm-25.05/sbin/sackd --systemd --conf-
server=10.3.149.220:6817

Dec 17 16:34:55 ip-10-3-27-112.ec2.internal systemd[1]: Starting Slurm auth and cred
 kiosk daemon...
Dec 17 16:34:55 ip-10-3-27-112.ec2.internal systemd[1]: Started Slurm auth and cred
 kiosk daemon.
Dec 17 16:34:55 ip-10-3-27-112.ec2.internal sackd[9985]: sackd: running
```

sinfo 및와 같은 Slurm 클라이언트 명령을 사용하여 클러스터에 대한 연결이 작동하는지 확인합니다squeue. 다음은의 출력 예제입니다sinfo.

```
[root@ip-10-3-27-112 ~]# /opt/aws/pcs/scheduler/slurm-25.05/bin/sinfo
PARTITION AVAIL TIMELIMIT NODES STATE NODELIST
all up infinite 4 idle~ compute-[1-4]
```

작업을 제출할 수도 있어야 합니다. 예를 들어이 예제와 유사한 명령은 클러스터의 노드 1개에서 대화형 작업을 시작합니다.

```
/opt/aws/pcs/scheduler/slurm-25.05/bin/srun --nodes=1 -p all --pty bash -i
```

독립 실행형 로그인 노드를 AWS PCS의 여러 클러스터에 연결

`pcs-multi-cluster-login-configure.sh` 스크립트는 단일 독립 실행형 로그인 노드에서 여러 Slurm `sackd` 데몬을 구성하는 자동화된 방법을 제공합니다. 이를 통해 로그인 노드가 여러 클러스터와 통신할 수 있습니다. 스크립트는 다음 작업을 자동화합니다.

- AWS PCS API 작업을 사용하여 클러스터 정보 가져오기
- base64로 인코딩된 Slurm 인증 키에 대한 프롬프트
- 클러스터 인증 키를 사용하여 Slurm JWKS 파일을 생성합니다.
- 클러스터 엔드포인트 및 포트로 `sackd` 서비스를 구성합니다.
- 클러스터별 데몬에 대한 `systemd` 서비스 파일을 생성합니다`sackd`.
- 클러스터 환경 설정을 위한 활성화 스크립트를 생성합니다.
- `sackd` 서비스를 활성화하고 시작합니다.

> **Note**
>
> 이 스크립트에는 Slurm 버전 25.05 이상이 필요합니다.

Slurm은 인스턴스에 이미 설치되어 있어야 합니다(수동 프로세스의 3단계에 해당). 인스턴스는 대상 클러스터의 엔드포인트에 도달할 수 있어야 합니다. 스크립트는 수동 구성 프로세스에서 4단계와 5단계의 동일한 작업을 수행합니다. 클러스터 정보를 자동으로 가져오고, `sackd` 서비스를 구성하고, 필요한 `systemd` 서비스 파일을 생성하고, 사용자가 클러스터 상호 작용을 위해 셸 환경을 구성하는 데 사용할 수 있는 활성화 스크립트를 생성합니다.

주제

- AWS PCS 다중 클러스터 로그인 노드 구성 스크립트의 사전 조건
- AWS PCS 다중 클러스터 로그인 노드 구성 스크립트 코드
- AWS PCS 다중 클러스터 로그인 노드 구성 스크립트 사용

AWS PCS 다중 클러스터 로그인 노드 구성 스크립트의 사전 조건

시스템 요구 사항

- 가 `systemd` 지원되는 Linux OS
- 시스템 구성에 대한 루트 권한

필수 명령 및 패키지

- `bash` – 쉘 인터프리터(버전 4.0 이상)
- `curl` - AWS IMDS v2 메타데이터 검색의 경우
- `jq` - AWS API 응답을 구문 분석하기 위한 JSON 프로세서
- `aws` – AWS CLI v2 - AWS PCS API 작업 실행 및 Secrets Manager 액세스
- `systemctl` - `systemd` 서비스 관리
- `find` - 파일 시스템 검색 유틸리티
- `grep` - 텍스트 패턴 일치
- `sed` - 텍스트 조작을 위한 스트림 편집기
- `sort` - 텍스트 정렬 유틸리티
- `tail` - 파일의 마지막 줄을 표시합니다.
- `mkdir` - 디렉터리 생성
- `chmod` - 파일 권한 변경
- `chown` - 파일 소유권 변경
- `ldconfig` - 동적 링커 구성

AWS 요구 사항

- Slurm 버전 25.05 이상을 실행하는 AWS PCS 클러스터
- AWS 구성된 자격 증명(IAM 역할, 자격 증명 파일 또는 환경 변수를 통해)
- 다음에 대한 권한:
  - `pcs:GetCluster`
  - `secretsmanager:GetSecretValue` (대체 보안 암호를 사용하는 경우)

시스템 사용자 및 그룹

- `slurm` 사용자와 그룹이 시스템에 있어야 합니다.

Slurm 설치

- Slurm은 AWS PCS Slurm 설치 관리자 패키지와 동일한 위치에 설치해야 합니다.

```
/opt/aws/pcs/scheduler/slurm-version
```

AWS PCS 다중 클러스터 로그인 노드 구성 스크립트 코드

다음 소스 코드를 다음 이름의 파일에 저장합니다.

```
pcs-multi-cluster-login-configure.sh
```

스크립트 소스 코드

```
#!/bin/bash
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

# AWS PCS Multi-Cluster Standalone Login Node Configuration Script
#
# This script configures AWS Parallel Computing Service (PCS) multi-cluster stand alone
 login nodes
# by setting up the Slurm authentication and credential kiosk daemon (sackd)
# for connecting to remote PCS clusters.
#
# Prerequisites:
# - AWS CLI configured with appropriate permissions
# - Slurm version 25.05 or later
# - Root privileges for system configuration
# - Network connectivity to AWS PCS endpoints


set -eo pipefail

# Function to display usage
usage() {
```

```
    echo "Usage: $0 --cluster-identifier <cluster-identifier> [--endpoint-url
 <endpoint-url>]"
    echo "       $0 -h|--help"
}

# Function to display help
help() {
    echo "AWS PCS Multi-Cluster Standalone Login Node Configuration Script"
    echo "================================================="
    echo
    echo "This script configures multi-cluster standalone login node for AWS Parallel
 Computing Service (PCS)"
    echo "by setting up the Slurm authentication and credential kiosk daemon (sackd)."
    echo
    usage
    echo
    echo "Options:"
    echo "  --cluster-identifier <id>    AWS PCS cluster identifier (required)"
    echo "  --endpoint-url <url>         Custom PCS endpoint URL (optional)"
    echo "  -h, --help                   Show this help message"
    echo
    echo "Examples:"
    echo "  $0 --cluster-identifier my-pcs-cluster"
    echo
    echo "Note: This script requires root privileges and Slurm version 25.05 or later."
}

# Function to retrieve authentication key
get_auth_key() {
    if [ "$ALTERNATE_SECRET_RETRIEVAL" = "true" ]; then
        echo "Retrieving authentication key from AWS Secrets Manager..." >&2
        local auth_key_arn=$(echo "$CLUSTER_INFO" | jq -r
 '.cluster.slurmConfiguration.authKey.secretArn')
        local auth_key_version=$(echo "$CLUSTER_INFO" | jq -r
 '.cluster.slurmConfiguration.authKey.secretVersion')

        if [ "$auth_key_arn" = "null" ] || [ "$auth_key_version" = "null" ]; then
            echo "Error: Auth key information not found in cluster configuration" >&2
            exit 1
        fi

        if ! aws secretsmanager get-secret-value --secret-id "$auth_key_arn" --version-
id "$auth_key_version" --query SecretString --output text --region "$REGION" 2>/dev/
null; then
```

```
            echo "Error: Failed to retrieve auth key from Secrets Manager" >&2
            exit 1
        fi
    else
        echo "Please enter the base64-encoded Slurm authentication key:" >&2
        echo -n "Base64 of the Slurm secret key: " >&2
        local key
        read -rs key
        echo >&2
        echo "$key"
    fi
}

# Function to get next available SACKD port
get_next_sackd_port() {
    local exclude_file="$1"
    local port=6918
    local used_ports=()

    # Get all currently used SACKD ports into an array
    while IFS= read -r line; do
        used_ports+=("$line")
    done < <(find /etc/sysconfig -name "sackd-pcs-*" ! -path "$exclude_file" \
             -exec grep SACKD_PORT= '{}' ';' 2>/dev/null | \
             sed 's/.*SACKD_PORT=//' | sort -n)

    # Loop through used ports to find first available port
    for used_port in "${used_ports[@]}"; do
        if [ "$port" -lt "$used_port" ]; then
            break
        elif [ "$port" -eq "$used_port" ]; then
            ((port++))
        fi
    done

    echo "$port"
}

# Function to configure cluster
configure_cluster() {
    mkdir -p /etc/slurm
    SLURM_JWKS_FILE="/etc/slurm/slurm-${CLUSTER_NAME}.jwks"
```

```
    echo '{"keys":
[{"alg":"HS256","kty":"oct","kid":"key-'"${CLUSTER_ID}"'","k":"'"${BASE64_SLURM_KEY}"'"}]}'
 | jq -c '.' > "${SLURM_JWKS_FILE}"

    chmod 0600 "$SLURM_JWKS_FILE"
    chown slurm:slurm "$SLURM_JWKS_FILE"

    SLURM_INSTALL_PATH="/opt/aws/pcs/scheduler/slurm-${SLURM_VERSION}"

    SACKD_RUNTIME_DIRECTORY="/run/slurm-${CLUSTER_NAME}"
    mkdir -p "${SACKD_RUNTIME_DIRECTORY}"
    chown slurm:slurm "${SACKD_RUNTIME_DIRECTORY}"

    mkdir -p /etc/sysconfig
    SACKD_SERVICE_NAME="sackd-pcs-${CLUSTER_NAME}"
    SACKD_SERVICE_ENV="/etc/sysconfig/${SACKD_SERVICE_NAME}"
    SACKD_PORT=$(get_next_sackd_port "$SACKD_SERVICE_ENV")
    cat > "${SACKD_SERVICE_ENV}" << EOF
SACKD_OPTIONS='--conf-server=$ENDPOINTS'
SLURM_SACK_JWKS='$SLURM_JWKS_FILE'
RUNTIME_DIRECTORY='$SACKD_RUNTIME_DIRECTORY'
SACKD_PORT=$SACKD_PORT
EOF

    SACKD_SERVICE_PATH="/etc/systemd/system/${SACKD_SERVICE_NAME}.service"

    cat << EOF > "$SACKD_SERVICE_PATH"
[Unit]
Description=Slurm auth and cred kiosk daemon
After=network-online.target remote-fs.target
Wants=network-online.target
ConditionPathExists=${SACKD_SERVICE_ENV}

[Service]
Type=notify
EnvironmentFile=${SACKD_SERVICE_ENV}
User=slurm
Group=slurm
RuntimeDirectory=slurm-${CLUSTER_NAME}
RuntimeDirectoryMode=0755
ExecStart=${SLURM_INSTALL_PATH}/sbin/sackd --systemd \$SACKD_OPTIONS
ExecReload=/bin/kill -HUP \$MAINPID
KillMode=process
LimitNOFILE=131072
```

```
LimitMEMLOCK=infinity
LimitSTACK=infinity

[Install]
WantedBy=multi-user.target
EOF

    chown root:root "$SACKD_SERVICE_PATH"
    chmod 0644 "$SACKD_SERVICE_PATH"
    systemctl daemon-reload && systemctl enable "$SACKD_SERVICE_NAME"
    systemctl restart "$SACKD_SERVICE_NAME"

    ACTIVATE_SCRIPT="activate-pcs-${CLUSTER_NAME}"
    cat > "$ACTIVATE_SCRIPT" << EOF
# Activate script for Slurm cluster ${CLUSTER_NAME}

# Add Slurm paths
export PATH="${SLURM_INSTALL_PATH}/bin:\$PATH"
export MANPATH="${SLURM_INSTALL_PATH}/share/man:\$MANPATH"
export LD_LIBRARY_PATH="${SLURM_INSTALL_PATH}/lib:\$LD_LIBRARY_PATH"
ldconfig

# Set Slurm configuration
export SLURM_CONF="/run/slurm-${CLUSTER_NAME}/conf/slurm.conf"
export PCS_CLUSTER_NAME="${CLUSTER_NAME}"
export PCS_CLUSTER_IDENTIFIER="${CLUSTER_IDENTIFIER}"
export PCS_CLUSTER_ID="${CLUSTER_ID}"

echo "Activated PCS cluster environment: ${CLUSTER_NAME}"

# Deactivate function
function deactivate-pcs-${CLUSTER_NAME}() {
    export PATH="\$(echo "\$PATH" | sed -e "s|${SLURM_INSTALL_PATH}/bin:||g" -e "s|:
${SLURM_INSTALL_PATH}/bin||g" -e "s|^${SLURM_INSTALL_PATH}/bin\$||")"
    export MANPATH="\$(echo "\$MANPATH" | sed -e "s|${SLURM_INSTALL_PATH}/share/man:||
g" -e "s|:${SLURM_INSTALL_PATH}/share/man||g" -e "s|^${SLURM_INSTALL_PATH}/share/man\
$||")"
    export LD_LIBRARY_PATH="\$(echo "\$LD_LIBRARY_PATH" | sed -e "s|
${SLURM_INSTALL_PATH}/lib:||g" -e "s|:${SLURM_INSTALL_PATH}/lib||g" -e "s|^
${SLURM_INSTALL_PATH}/lib\$||")"
    unset SLURM_CONF
    unset PCS_CLUSTER_NAME
    unset PCS_CLUSTER_IDENTIFIER
    unset PCS_CLUSTER_ID
```

```
    unset -f deactivate-pcs-${CLUSTER_NAME}
    ldconfig
    echo "Deactivated PCS cluster environment: ${CLUSTER_NAME}"
}

export -f deactivate-pcs-${CLUSTER_NAME}

EOF
}

# Main function
main() {
    # Parse arguments
    CLUSTER_IDENTIFIER=""
    PCS_ENDPOINT_URL=""

    while [ "$1" != "" ]; do
        case $1 in
            --cluster-identifier)
                shift
                CLUSTER_IDENTIFIER="$1"
                ;;
            --endpoint-url)
                shift
                PCS_ENDPOINT_URL="--endpoint-url $1"
                ;;
            -h|--help)
                help
                exit 0
                ;;
            *)
                echo "Invalid argument: $1" >&2
                usage >&2
                exit 1
                ;;
        esac
        shift
    done

    # Validate required arguments
    if [ -z "$CLUSTER_IDENTIFIER" ]; then
        echo "Error: --cluster-identifier is required" >&2
        usage >&2
        exit 1
```

```
    fi

    # Validate running as root
    if [ "$EUID" -ne 0 ]; then
        echo "Error: This script must be run as root" >&2
        exit 1
    fi

    # Validate required commands are available
    for cmd in aws jq curl; do
        if ! command -v "$cmd" &> /dev/null; then
            echo "Error: Required command '$cmd' not found" >&2
            exit 1
        fi
    done

    # Get the region name from IMDS v2 with error handling (try IPv6 first, fallback to
 IPv4)
    echo "Retrieving AWS region from instance metadata..."
    # Try IPv6 IMDS endpoint first (fd00:ec2::254) with fast timeout (1s connect, 2s
 total)
    # If IPv6 fails, fallback to IPv4 IMDS endpoint (169.254.169.254)
    IMDS_ENDPOINT="http://[fd00:ec2::254]"
    if ! TOKEN=$(curl -s -X PUT "${IMDS_ENDPOINT}/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600" --connect-timeout 1 --max-time 2 2>/dev/null); then
        IMDS_ENDPOINT="http://169.254.169.254"
        if ! TOKEN=$(curl -s -X PUT "${IMDS_ENDPOINT}/latest/api/token" -H "X-aws-ec2-
metadata-token-ttl-seconds: 21600" --max-time 5); then
            echo "Error: Failed to retrieve IMDS token. Ensure this script is running
 on an EC2 instance." >&2
            exit 1
        fi
    fi

    if ! REGION=$(curl -s -H "X-aws-ec2-metadata-token: $TOKEN" "${IMDS_ENDPOINT}/
latest/dynamic/instance-identity/document" --max-time 5 | jq -r '.region'); then
        echo "Error: Failed to retrieve AWS region from instance metadata" >&2
        exit 1
    fi

    echo "Detected AWS region: $REGION"

    # Retrieve cluster information from AWS PCS
    echo "Retrieving cluster information for: $CLUSTER_IDENTIFIER"
```

```
   # shellcheck disable=SC2086
   if ! CLUSTER_INFO=$(aws pcs get-cluster --region "$REGION" --cluster-identifier
"$CLUSTER_IDENTIFIER" $PCS_ENDPOINT_URL 2>/dev/null); then
       echo "Error: Failed to retrieve cluster information. Check cluster identifier
and AWS permissions." >&2
       exit 1
   fi

   CLUSTER_ID=$(echo "$CLUSTER_INFO" | jq -r '.cluster.id')
   CLUSTER_NAME="$(echo "$CLUSTER_INFO" | jq -r '.cluster.name')"
   SLURM_VERSION=$(echo "$CLUSTER_INFO" | jq -r '.cluster.scheduler.version')
   SLURM_VERSION=${SLURM_VERSION#Slurm_}

   # Check if Slurm version is >= 25.05
   # shellcheck disable=SC2072
   if [[ "$SLURM_VERSION" < "25.05" ]]; then
       echo "Error: This script requires Slurm version 25.05 or later. Found version:
$SLURM_VERSION" >&2
       exit 1
   fi

   ENDPOINTS=$(echo "$CLUSTER_INFO" | jq -r '.cluster.endpoints[] | select(.type
== "SLURMCTLD") | (if .privateIpAddress != "" then .privateIpAddress else "["
+ .ipv6Address + "]" end) + ":" + .port' | tr '\n' ',' | sed 's/,$//')

   # Get BASE64_SLURM_KEY
   BASE64_SLURM_KEY=$(get_auth_key)

   if [ -z "$BASE64_SLURM_KEY" ]; then
       echo "Error: base64 Slurm key cannot be empty" >&2
       exit 1
   fi

   configure_cluster

   # Final configuration summary
   echo "========================================"
   echo "Configuration completed successfully!"
   echo "========================================"
   echo "Cluster Name: $CLUSTER_NAME"
   echo "Cluster ID: $CLUSTER_ID"
   echo "Slurm Version: $SLURM_VERSION"
   echo "Service Name: $SACKD_SERVICE_NAME"
   echo "SACKD Port: $SACKD_PORT"
```

```
    echo
    echo "To activate this cluster environment, run:"
    echo "  source ./$ACTIVATE_SCRIPT"
    echo
    echo "To deactivate this cluster environment, run:"
    echo "  deactivate-pcs-${CLUSTER_NAME}"
    echo
    echo "To check service status:"
    echo "  systemctl status $SACKD_SERVICE_NAME"
    echo
    echo "To view service logs:"
    echo "  journalctl -u $SACKD_SERVICE_NAME -f"
}

# Exit if being sourced for testing
[[ "${BASH_SOURCE[0]}" != "${0}" ]] && return

# Execute main function
main "$@"
```

AWS PCS 다중 클러스터 로그인 노드 구성 스크립트 사용

스크립트 실행

구성 스크립트를 실행하려면

1. 스크립트의 내용을 다음과 같은 파일에 저장합니다.

   ```
   pcs-multi-cluster-login-configure.sh
   ```

2. 실행 가능하도록 설정합니다.

   ```
   chmod +x pcs-multi-cluster-login-configure.sh
   ```

3. 스크립트를 실행합니다.

   ```
   ./pcs-multi-cluster-login-configure.sh --cluster-identifier cluster-name
   ```

클러스터 상호 작용 환경

구성에 성공하면 스크립트는 현재 디렉터리에 클러스터별 활성화 스크립트를 생성합니다. 스크립트의 이름은 입니다`activate-pcs-`*`cluster-name`*. 활성화 스크립트는 대상 클러스터와 상호 작용하는 데 필요한 환경 변수 및 경로를 구성합니다.

클러스터 환경을 활성화하려면

- `source` 명령을 사용하여 활성화 스크립트 실행

```
source ./activate-pcs-cluster-name
```

Example

```
# Activate cluster environment for cluster 'my-cluster'
source ./activate-pcs-my-cluster

# Now you can use Slurm commands
sinfo
squeue
sbatch my-job.sh
```

활성화 스크립트가 수행하는 작업

- 클러스터의 구성을 가리키도록 `SLURM_CONF` 환경 변수를 설정합니다.
- 클러스터의 Slurm 바이너리`PATH`를 포함하도록를 업데이트합니다.
- 기타 필요한 Slurm 환경 변수(`MANPATH`, `LD_LIBRARY_PATH`)를 구성합니다.
- AWS PCS 클러스터 식별 변수를 설정합니다.
- 대상 AWS PCS 클러스터와의 원활한 상호 작용을 활성화합니다.

클러스터 환경을 비활성화하려면

- 비활성화 명령을 실행합니다.

```
deactivate-pcs-cluster-name
```

Example

```
# After activating a cluster
source ./activate-pcs-my-cluster

# Work with the cluster
sinfo

# Deactivate when done
deactivate-pcs-my-cluster
```

비활성화 명령이 수행하는 작업

- 원래 PATH 환경 변수를 복원합니다.
- 클러스터별 Slurm 환경 변수를 설정 해제합니다.
- 쉘 환경을 사전 활성화 상태로 되돌립니다.

> **Note**
>
> 활성화는 세션별로 다르며 클러스터와 상호 작용하려는 쉘 세션에서 소싱되어야 합니다.

AWS PCS 네트워킹

AWS PCS 클러스터는 Amazon VPC에서 생성됩니다. 이 장에는 클러스터의 스케줄러 및 노드를 위한 네트워킹에 대한 다음 주제가 포함되어 있습니다.

인스턴스를 시작할 서브넷을 선택하는 경우를 제외하고 EC2 시작 템플릿을 사용하여 AWS PCS 컴퓨팅 노드 그룹에 대한 네트워킹을 구성해야 합니다. 시작 템플릿에 대한 자세한 내용은 AWS PCS에서 Amazon EC2 시작 템플릿 사용의 내용을 참조하세요.

주제

- AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항
- AWS PCS 클러스터에 대한 VPC 생성
- AWS PCS의 보안 그룹
- AWS PCS의 여러 네트워크 인터페이스
- AWS PCS의 EC2 인스턴스에 대한 배치 그룹
- AWS PCS에서 EFA(Elastic Fabric Adapter) 사용

AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항

AWS PCS 클러스터를 생성할 때 VPC에 서브넷을 지정합니다. 이 주제에서는 클러스터에 사용하는 VPC 및 서브넷(들)에 대한 AWS PCS 관련 요구 사항 및 고려 사항에 대한 개요를 제공합니다. AWS PCS와 함께 사용할 VPC가 없는 경우 AWS제공 CloudFormation 템플릿을 사용하여 VPC를 생성할 수 있습니다. VPCs에 대한 자세한 내용은 Amazon VPC 사용 설명서의 Virtual Private Clouds(VPC)를 참조하세요.

VPC 요구 사항 및 고려 사항

클러스터를 생성하는 경우 지정하는 VPC는 다음 요구 사항 및 고려 사항을 충족해야 합니다.

- VPC에는 생성하려는 클러스터, 노드 및 기타 클러스터 리소스에 사용할 수 있는 충분한 수의 IP 주소가 있어야 합니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC 및 서브넷의 IP 주소 지정을 참조하세요.
- 클러스터가 IPv6를 사용하는 경우:
  - IPv6 CIDR 블록을 VPC와 연결 자세한 내용은 Amazon VPC 사용 설명서의 VPC 생성을 참조하세요.

> **Important**
>
> IPv4와 IPv6 모두로 VPC를 구성할 수 있지만 클러스터에 대해 하나의 네트워크 유형만 선택할 수 있습니다.

  - 서브넷에 IPvIPv6 주소 자동 할당을 활성화합니다.
  - 자세한 내용은 다음을 참조하세요.
    - 의 IPv6 AWS
    - AWS의 IPv6 주소 지정 이해 및 확장 가능한 주소 지정 계획 설계
- VPC에는 DNS 호스트 이름과 DNS 확인 지원이 있어야 합니다. 그렇지 않으면 노드가 고객 클러스터를 등록할 수 없습니다. 자세한 내용은 Amazon VPC 사용 설명서의 VPC에 대한 DNS 속성을 참조하세요.
- VPC가 AWS PCS API에 연결할 수 AWS PrivateLink 있으려면를 사용하는 VPC 엔드포인트가 필요할 수 있습니다. 자세한 내용은 Amazon VPC 사용 설명서의를 사용하여 VPC를 서비스에 연결을 참조하세요 AWS PrivateLink.

> **Important**
>
> AWS PCS는 전용 인스턴스 테넌시가 있는 VPC를 지원하지 않습니다. AWS PCS에 사용하는 VPC는 `default` 인스턴스 테넌시를 사용해야 합니다. 기존 VPC의 인스턴스 테넌시를 변경할 수 있습니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 VPC의 인스턴스 테넌시 변경을 참조하세요.

서브넷 요구 사항 및 고려 사항

Slurm 클러스터를 생성하면 AWS PCS는 지정한 서브넷에 탄력적 네트워크 인터페이스(ENI)를 생성합니다. 이 네트워크 인터페이스를 사용하면 스케줄러 컨트롤러와 고객 VPC 간의 통신이 가능합니다. 또한 네트워크 인터페이스를 사용하면 Slurm이 계정에 배포된 구성 요소와 통신할 수 있습니다. 생성 시에만 클러스터의 서브넷을 지정할 수 있습니다.

클러스터의 서브넷 요구 사항

클러스터를 생성할 때 지정하는 서브넷은 다음 요구 사항을 충족해야 합니다.

- 서브넷에는 AWS PCS에서 사용할 IP 주소가 1개 이상 있어야 합니다.

- 클러스터가 IPv6를 사용하는 경우 클러스터의 모든 서브넷이 IPv6를 사용해야 합니다.

> **Important**
>
> 서브넷이 IPv6만 사용하도록 구성된 경우 AWS PCS 샘플 AMIs 및 여러 네트워크 인터페이스로 구성된 컴퓨팅 노드 그룹은 현재 작동하지 않습니다. 대신 듀얼 스택 서브넷(IPv4 및 IPv6) 또는 IPv4-only 서브넷을 사용합니다. 자세한 내용은 AWS PCS에서 샘플 Amazon Machine Image(AMIs) 사용 단원을 참조하십시오.

- 서브넷은 AWS Outposts AWS Wavelength또는 AWS 로컬 영역에 상주할 수 없습니다.
- 서브넷은 퍼블릭 또는 프라이빗일 수 있습니다. 가능하면 프라이빗 서브넷을 지정하는 것이 좋습니다. 퍼블릭 서브넷은 인터넷 게이트웨이에 대한 경로를 포함하는 라우팅 테이블이 있는 서브넷이고, 프라이빗 서브넷은 인터넷 게이트웨이에 대한 경로를 포함하지 않는 라우팅 테이블이 있는 서브넷입니다.

노드의 서브넷 요구 사항

노드 및 기타 클러스터 리소스를 AWS PCS 클러스터를 생성할 때 지정한 서브넷과 동일한 VPC의 다른 서브넷에 배포할 수 있습니다.

노드와 클러스터 리소스를 배포하는 모든 서브넷은 다음 요구 사항을 충족해야 합니다.

- 서브넷에 모든 노드와 클러스터 리소스를 배포할 수 있는 충분한 IP 주소가 있는지 확인해야 합니다.
- 클러스터가 IPv4를 사용하고 퍼블릭 서브넷에 노드를 배포하려는 경우 해당 서브넷은 IPv4 퍼블릭 주소를 자동 할당해야 합니다.

> **Note**
>
> 퍼블릭 서브넷의 인스턴스는 퍼블릭 IP 주소의 트래픽을 허용하는 인바운드 규칙이 있는 보안 그룹을 사용해야 합니다. 특정 소스 주소 제한이 없는 한, 이는 IPv4 소스 주소가 0.0.0.0/0이거나 IPv6 소스 주소가 ::/0임을 의미합니다.

- 노드를 배포하는 서브넷이 프라이빗 서브넷이고 라우팅 테이블에 네트워크 주소 변환(NAT) 디바이스(IPv4)에 대한 경로가 포함되지 않은 경우를 사용하여 VPC 엔드포인트를 고객 VPC AWS PrivateLink 에 추가합니다. VPC 엔드포인트는 노드가 접촉하는 모든 AWS 서비스에 필요합니다. 유일한 필수 엔드포인트는 노드가 `RegisterComputeNodeGroupInstance`

API 작업을 호출하도록 허용하는 AWS PCS입니다. 자세한 내용은 PCS API 참조의 RegisterComputeNodeGroupInstance를 참조하세요. AWS

- 퍼블릭 또는 프라이빗 서브넷 상태는 AWS PCS에 영향을 주지 않습니다. 필요한 엔드포인트에 연결할 수 있어야 합니다.

AWS PCS 클러스터에 대한 VPC 생성

AWS 병렬 컴퓨팅 서비스(AWS PCS) 내에서 클러스터에 대한 Amazon Virtual Private Cloud(VPC)를 생성할 수 있습니다.

Amazon VPC를 사용하여 정의한 가상 네트워크로 VPC 리소스를 시작합니다. 이 가상 네트워크는 사용자의 자체 데이터 센터에서 운영하는 기존 네트워크와 매우 유사합니다. 그러나 Amazon Web Services Services의 확장 가능한 인프라를 사용하면 얻을 수 있는 이점이 있습니다. 프로덕션 VPC 클러스터를 배포하기 전에 Amazon VPC 서비스를 철저히 이해하는 것이 좋습니다. 자세한 내용은 Amazon VPC란 무엇입니까?를 참조하세요. - 작성자 시각적 객체 모드.Amazon VPC 사용 설명서.

PCS 클러스터, 노드 및 지원 리소스(예: 파일 시스템 및 디렉터리 서비스)는 Amazon VPC 내에 배포됩니다. 기존 Amazon VPC를 PCS와 함께 사용하려면 AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항에 설명된 요구 사항을 충족해야 합니다. 이 주제에서는 AWS제공 CloudFormation 템플릿을 사용하여 PCS 요구 사항을 충족하는 VPC를 생성하는 방법을 설명합니다. 템플릿을 배포했다면 템플릿에서 생성된 리소스를 보면서 생성된 리소스와 해당 리소스의 구성을 정확히 알 수 있습니다.

사전 조건

PCS용 Amazon VPC를 생성하려면 Amazon VPC 리소스를 생성하는 데 필요한 IAM 권한이 있어야 합니다. 이러한 리소스는 VPC, 서브넷, 보안 그룹, 라우팅 테이블 및 경로, 인터넷 및 NAT 게이트웨이입니다. 자세한 내용은 Amazon VPC 사용 설명서의 퍼블릭 서브넷이 있는 VPC 생성을 참조하세요. Amazon EC2의 전체 목록을 검토하려면 서비스 승인 참조의 Amazon EC2에 사용되는 작업, 리소스 및 조건 키를 참조하세요.

Amazon VPC 생성

복사하여 PCS를 AWS 리전 사용할에 적합한 URL을 붙여 넣어 VPC를 생성합니다. CloudFormation 템플릿을 다운로드하여 CloudFormation 콘솔에 직접 업로드할 수도 있습니다.

- 미국 동부(버지니아 북부)(us-east-1)

```
https://console.aws.amazon.com/cloudformation/home?region=us-east-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 미국 동부(오하이오)(us-east-2)

```
https://console.aws.amazon.com/cloudformation/home?region=us-east-2#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 미국 서부(오레곤)(us-west-2)

```
https://console.aws.amazon.com/cloudformation/home?region=us-west-2#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 아시아 태평양(뭄바이)(ap-south-1)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-south-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 아시아 태평양(싱가포르)(ap-southeast-1)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-southeast-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 아시아 태평양(시드니)(ap-southeast-2)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-southeast-2#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 아시아 태평양(도쿄)(ap-northeast-1)

```
https://console.aws.amazon.com/cloudformation/home?region=ap-northeast-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 유럽(프랑크푸르트)(eu-central-1)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-central-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 유럽(아일랜드)(eu-west-1)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-west-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 유럽(런던)(eu-west-2)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-west-2#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 유럽(파리)(eu-west-3)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-west-3#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 유럽(밀라노)(eu-south-1)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-south-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 유럽(스톡홀름)(eu-north-1)

```
https://console.aws.amazon.com/cloudformation/home?region=eu-north-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- AWS GovCloud(미국 동부)(us-gov-east-1)

```
https://console.aws.amazon.com/cloudformation/home?region=us-gov-east-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- AWS GovCloud(미국 서부)(us-gov-west-1)

```
https://console.aws.amazon.com/cloudformation/home?region=us-gov-west-1#/stacks/
create/review?stackName=hpc-networking&templateURL=https://aws-hpc-recipes.s3.us-
east-1.amazonaws.com/main/recipes/net/hpc_large_scale/assets/main.yaml
```

- 템플릿만

```
https://aws-hpc-recipes.s3.us-east-1.amazonaws.com/main/recipes/net/hpc_large_scale/
assets/main.yaml
```

PCS용 Amazon VPC를 생성하려면

1. CloudFormation 콘솔에서 템플릿을 엽니다.

   > **Note**
   >
   > 템플릿에 미리 채워져 있으므로 기본값으로 두기만 하면 됩니다.

2. 스택 이름 제공에서 스택 이름을 입력하고를 입력합니다`hpc-networking`.
3. 파라미터에 다음 세부 정보를 입력합니다.

   a. VPC에서 CidrBlock을 입력하고 `10.3.0.0/16`

   b. 서브넷 A에서:

      i. 그런 다음 CidrPublicSubnetA를 입력하고 `10.3.0.0/20`

      ii. 그런 다음 CidrPrivateSubnetA를 입력하고 `10.3.128.0/20`

   c. 서브넷 B에서:

      i. 그런 다음 CidrPublicSubnetB를 입력하고 `10.3.16.0/20`

      ii. 그런 다음 CidrPrivateSubnetA를 입력하고 `10.3.144.0/20`

   d. 서브넷 C에서:

      i. ProvisionSubnetsC에서를 선택합니다`True`.

> **Note**
>
> 가용 영역이 3개 미만인 리전에서 VPC를 생성하는 경우 로 설정하면이 옵션이 무시됩니다`True`.

ii. 그런 다음 CidrPublicSubnetB를 입력하고 `10.3.32.0/20`

iii. 그런 다음 CidrPrivateSubnetA를 입력하고 `10.3.160.0/20`

4. 기능에서 AWS CloudFormation이 IAM 리소스를 생성할 수 있음을 승인합니다 확인란을 선택합니다.

CloudFormation 스택의 상태를 모니터링합니다. `CREATE_COMPLETE`에 도달하면 VPC 리소스를 사용할 준비가 된 것입니다.

> **Note**
>
> CloudFormation 템플릿이 생성한 모든 리소스를 보려면 CloudFormation 콘솔을 엽니다. `hpc-networking` 스택을 선택한 다음 리소스(Resources) 탭을 선택합니다.

AWS PCS의 보안 그룹

Amazon EC2의 보안 그룹은 인스턴스에 대한 인바운드 및 아웃바운드 트래픽을 제어하는 가상 방화벽 역할을 합니다. AWS PCS 컴퓨팅 노드 그룹에 대한 시작 템플릿을 사용하여 인스턴스에 보안 그룹을 추가하거나 제거합니다. 시작 템플릿에 네트워크 인터페이스가 없는 경우 `SecurityGroupIds`를 사용하여 보안 그룹 목록을 제공합니다. 시작 템플릿이 네트워크 인터페이스를 정의하는 경우 `Groups` 파라미터를 사용하여 각 네트워크 인터페이스에 보안 그룹을 할당해야 합니다. 시작 템플릿에 대한 자세한 내용은 AWS PCS에서 Amazon EC2 시작 템플릿 사용의 내용을 참조하세요.

> **Note**
>
> 시작 템플릿에서 보안 그룹 구성을 변경하면 컴퓨팅 노드 그룹이 업데이트된 후에 시작된 새 인스턴스에만 영향을 줍니다.

보안 그룹 요구 사항 및 고려 사항

AWS PCS는 클러스터를 생성할 때 지정하는 서브넷에 교차 계정 탄력적 네트워크 인터페이스(ENI)를 생성합니다. 이렇게 하면에서 관리하는 계정에서 실행되는 HPC 스케줄러가 AWS PCS에서 시작한 EC2 인스턴스와 통신할 수 있는 AWS경로를 제공합니다. 스케줄러 ENI와 클러스터 EC2 인스턴스 간의 양방향 통신을 허용하는 해당 ENI에 대한 보안 그룹을 제공해야 합니다.

이를 수행하는 간단한 방법은 그룹의 모든 멤버 간의 모든 포트에서 TCP/IP 트래픽을 허용하는 허용적 자체 참조 보안 그룹을 생성하는 것입니다. 클러스터와 노드 그룹 EC2 인스턴스 모두에 연결할 수 있습니다.

허용 보안 그룹 구성의 예

IPv4

규칙 타입	프로토콜	포트	소스	Destination
인바운드	모두	모두	본인	
아웃바운드	모두	모두		0.0.0.0/0
아웃바운드	모두	모두		본인

IPv6

규칙 타입	프로토콜	포트	소스	Destination
인바운드	모두	모두	본인	
아웃바운드	모두	모두		::/0
아웃바운드	모두	모두		본인

이러한 규칙은 모든 트래픽이 Slurm 컨트롤러와 노드 간에 자유롭게 흐르도록 허용하고, 모든 대상에 대한 모든 아웃바운드 트래픽을 허용하며, EFA 트래픽을 활성화합니다.

제한적인 보안 그룹 구성의 예

클러스터와 해당 컴퓨팅 노드 간의 열린 포트를 제한할 수도 있습니다. Slurm 스케줄러의 경우 클러스터에 연결된 보안 그룹은 다음 포트를 허용해야 합니다.

- 6817 - EC2 인스턴스`slurmctld`에서에 대한 인바운드 연결 활성화
- 6818 - EC2 인스턴스에서 `slurmd` 실행 중인에서 `slurmctld`로의 아웃바운드 연결 활성화

컴퓨팅 노드에 연결된 보안 그룹은 다음 포트를 허용해야 합니다.

- 6817 - EC2 인스턴스`slurmctld`에서 로 아웃바운드 연결을 활성화합니다.
- 6818 - 노드 그룹 인스턴스`slurmd`에서 `slurmctld`와 `slurmd`의 인바운드 및 아웃바운드 연결 활성화
- 60001~63000 - 지원할 노드 그룹 인스턴스 간의 인바운드 및 아웃바운드 연결 `srun`
- 노드 그룹 인스턴스 간의 EFA 트래픽입니다. 자세한 내용은 Linux 인스턴스용 사용 설명서의 EFA 지원 보안 그룹 준비를 참조하세요.
- 워크로드에 필요한 기타 노드 간 트래픽

AWS PCS의 여러 네트워크 인터페이스

일부 EC2 인스턴스에는 여러 개의 네트워크 카드가 있습니다. 이를 통해 100Gbps를 초과하는 대역폭 기능과 향상된 패킷 처리 등 더 높은 네트워크 성능을 제공할 수 있습니다. 여러 네트워크 카드가 있는 인스턴스에 대한 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 탄력적 네트워크 인터페이스를 참조하세요.

EC2 시작 템플릿에 네트워크 인터페이스를 추가하여 AWS PCS 컴퓨팅 노드 그룹의 인스턴스에 대한 추가 네트워크 카드를 구성합니다. 다음은 `hpc7a.96xlarge` 인스턴스에서 찾을 수 있는와 같은 두 개의 네트워크 카드를 활성화하는 시작 템플릿의 예입니다. 다음의 세부 정보를 적어 둡니다.

- 각 네트워크 인터페이스의 서브넷은 시작 템플릿을 사용할 AWS PCS 컴퓨팅 노드 그룹을 구성할 때 선택한 것과 동일해야 합니다.
- SSH 및 HTTPS 트래픽과 같은 일상적인 네트워크 통신이 발생하는 기본 네트워크 디바이스는 `DeviceIndex`를 로 설정하여 설정됩니다`0`. 다른 네트워크 인터페이스에는 `DeviceIndex`의이 있습니다`1`. 기본 네트워크 인터페이스는 하나만 있을 수 있습니다. 다른 모든 인터페이스는 보조 인터페이스입니다.

- 모든 네트워크 인터페이스에는 고유한이 있어야 합니다NetworkCardIndex. 권장되는 방법은 시작 템플릿에 정의된 대로 순차적으로 번호를 매기는 것입니다.
- 각 네트워크 인터페이스의 보안 그룹은를 사용하여 설정됩니다Groups. 이 예제에서는 인바운드 SSH 보안 그룹(sg-*SshSecurityGroupId*)이 기본 네트워크 인터페이스와 클러스터 내 통신을 활성화하는 보안 그룹()에 추가됩니다sg-*ClusterSecurityGroupId*. 마지막으로 인터넷(sg-*InternetOutboundSecurityGroupId*)에 대한 아웃바운드 연결을 허용하는 보안 그룹이 기본 인터페이스와 보조 인터페이스 모두에 추가됩니다.

```
{
    "NetworkInterfaces": [
        {
            "DeviceIndex": 0,
            "NetworkCardIndex": 0,
            "SubnetId": "subnet-SubnetId",
            "Groups": [
               "sg-SshSecurityGroupId",
               "sg-ClusterSecurityGroupId",
               "sg-InternetOutboundSecurityGroupId"
            ]
        },
        {
            "DeviceIndex": 1,
            "NetworkCardIndex": 1,
            "SubnetId": "subnet-SubnetId",
            "Groups": ["sg-InternetOutboundSecurityGroupId"]
        }
    ]
}
```

AWS PCS의 EC2 인스턴스에 대한 배치 그룹

배치 그룹을 사용하여 EC2 인스턴스에서 실행되는 워크로드의 요구 사항에 맞게 EC2 인스턴스 배치에 영향을 미칠 수 있습니다.

배치 그룹 유형

- 클러스터 - 인스턴스를 가용 영역에 가깝게 패키징하여 지연 시간이 짧은 통신을 최적화합니다.
- 파티션 - 논리적 파티션에 인스턴스를 분산하여 복원력을 극대화합니다.

- 분산 - 소수의 인스턴스가 고유한 하드웨어에서 시작되도록 엄격하게 적용하므로 복원력에도 도움이 될 수 있습니다.

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2 인스턴스에 대한 배치 그룹을 참조하세요.

Elastic Fabric Adapter(EFA)를 사용하도록 AWS PCS 컴퓨팅 노드 그룹을 구성할 때 클러스터 배치 그룹을 포함하는 것이 좋습니다.

EFA에서 작동하는 클러스터 배치 그룹을 생성하려면

1. 컴퓨팅 노드 그룹의 유형 클러스터를 사용하여 배치 그룹을 생성합니다.
   - 다음 AWS CLI 명령을 사용합니다.

     ```
     aws ec2 create-placement-group --strategy cluster --group-name PLACEMENT-GROUP-NAME
     ```

   - CloudFormation 템플릿을 사용하여 배치 그룹을 생성할 수도 있습니다. 자세한 내용은 AWS CloudFormation 사용 설명서의 CloudFormation 템플릿으로 작업을 참조하세요. 다음 URL에서 템플릿을 다운로드하여 CloudFormation 콘솔에 업로드합니다.

     ```
     https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/enable_efa/assets/efa-placement-group.yaml
     ```

2. 배치 그룹을 AWS PCS 컴퓨팅 노드 그룹의 EC2 시작 템플릿에 포함합니다.

AWS PCS에서 EFA(Elastic Fabric Adapter) 사용

Elastic Fabric Adapter(EFA)는 EC2 인스턴스에 연결하여 HPC(고성능 컴퓨팅) 및 기계 학습 애플리케이션을 가속화할 수 있는의 AWS 고성능 고급 네트워킹 인터커넥트입니다. EFA를 사용하여 AWS PCS 클러스터에서 실행되는 애플리케이션을 활성화하려면 다음과 같이 EFA를 사용하도록 AWS PCS 컴퓨팅 노드 그룹 인스턴스를 구성해야 합니다.

> **Note**
>
> AWS PCS 호환 AMI에 EFA 설치 - AWS PCS 컴퓨팅 노드 그룹에 사용되는 AMI에는 EFA 드라이버가 설치되어 있어야 합니다. EFA 소프트웨어가 설치된 사용자 지정 AMI를 빌드하는 방법에 대한 자세한 내용은 섹션을 참조하세요AWS PCS용 사용자 지정 Amazon Machine Image(AMIs).

목차

EFA 지원 EC2 인스턴스 식별

EFA를 사용하려면 AWS PCS 컴퓨팅 그룹에 허용되는 모든 인스턴스 유형이 EFA를 지원해야 하며, vCPUs(및 해당하는 경우 GPUs) 수가 동일해야 합니다. EFA 지원 인스턴스 목록은 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2의 HPC 및 ML 워크로드용 Elastic Fabric Adapter를 참조하세요. AWS CLI 를 사용하여 EFA를 지원하는 인스턴스 유형 목록을 볼 수도 있습니다. *region-code*를와 같이 AWS PCS AWS 리전 를 사용하는 로 바꿉니다us-east-1.

```
aws ec2 describe-instance-types \
   --region region-code \
   --filters Name=network-info.efa-supported,Values=true \
   --query "InstanceTypes[*].[InstanceType]" \
   --output text | sort
```

> **Note**
>
> 사용 가능한 네트워크 인터페이스 수 확인 - 일부 EC2 인스턴스에는 여러 네트워크 카드가 있습니다. 이렇게 하면 여러 EFAs. 자세한 내용은 AWS PCS의 여러 네트워크 인터페이스 단원을 참조하십시오.

EFA 통신을 지원하는 보안 그룹 생성

AWS CLI

다음 AWS CLI 명령을 사용하여 EFA를 지원하는 보안 그룹을 생성할 수 있습니다. 명령은 보안 그룹 ID를 출력합니다. 다음과 같이 교체합니다.

- *region-code* -와 같이 AWS PCS를 AWS 리전 사용하는를 지정합니다us-east-1.
- *vpc-id* - AWS PCS에 사용하는 VPC의 ID를 지정합니다.
- *efa-group-name* - 보안 그룹에 대해 선택한 이름을 입력합니다.

```
aws ec2 create-security-group \
    --group-name efa-group-name \
    --description "Security group to enable EFA traffic" \
    --vpc-id vpc-id \
    --region region-code
```

다음 명령을 사용하여 인바운드 및 아웃바운드 보안 그룹 규칙을 연결합니다. 다음과 같이 교체합니다.

- *efa-secgroup-id* - 방금 생성한 EFA 보안 그룹의 ID를 제공합니다.

```
aws ec2 authorize-security-group-ingress \
    --group-id efa-secgroup-id \
    --protocol -1 \
    --source-group efa-secgroup-id

aws ec2 authorize-security-group-egress \
    --group-id efa-secgroup-id \
    --protocol -1 \
    --source-group efa-secgroup-id
```

CloudFormation template

CloudFormation 템플릿을 사용하여 EFA를 지원하는 보안 그룹을 생성할 수 있습니다. 다음 URL에서 템플릿을 다운로드한 다음 AWS CloudFormation 콘솔에 업로드합니다.

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/enable_efa/assets/efa-sg.yaml
```

AWS CloudFormation 콘솔에서 템플릿을 연 상태에서 다음 옵션을 입력합니다.

- 스택 이름 제공에서
  - 스택 이름에와 같은 이름을 입력합니다efa-sg-stack.
- 파라미터에서

- SecurityGroupName에와 같은 이름을 입력합니다`efa-sg`.
- VPC에서 AWS PCS를 사용할 VPC를 선택합니다.

CloudFormation 스택 생성을 완료하고 상태를 모니터링합니다. `CREATE_COMPLETE` EFA 보안 그룹에 도달하면 사용할 준비가 된 것입니다.

(선택 사항) 배치 그룹 생성

클러스터 배치 그룹에서 EFA를 사용하는 모든 인스턴스를 시작하여 인스턴스 간의 물리적 거리를 최소화하는 것이 좋습니다. EFA를 사용할 각 컴퓨팅 노드 그룹에 대한 배치 그룹을 생성합니다. 컴퓨팅 노드 그룹에 대한 배치 그룹을 생성AWS PCS의 EC2 인스턴스에 대한 배치 그룹하려면 섹션을 참조하세요.

EC2 시작 템플릿 생성 또는 업데이트

EFA 네트워크 인터페이스는 AWS PCS 컴퓨팅 노드 그룹에 대한 EC2 시작 템플릿에서 설정됩니다. 네트워크 카드가 여러 개 있는 경우 여러 EFAs 구성할 수 있습니다. EFA 보안 그룹과 선택적 배치 그룹도 시작 템플릿에 포함됩니다.

다음은 hpc7a.96xlarge와 같은 두 개의 네트워크 카드가 있는 인스턴스의 시작 템플릿 예제입니다. 인스턴스는 클러스터 배치 그룹 `subnet-`*`SubnetID1`*에서 시작됩니다`pg-`*`PlacementGroupId1`*.

보안 그룹은 각 EFA 인터페이스에 특별히 추가해야 합니다. 모든 EFA에는 EFA 트래픽()을 활성화하는 보안 그룹이 필요합니다`sg-`*`EfaSecGroupId`*. 다른 보안 그룹, 특히 SSH 또는 HTTPS와 같은 일반 트래픽을 처리하는 보안 그룹은 기본 네트워크 인터페이스(`DeviceIndex`의 로 지정됨)에만 연결하면 됩니다`0`. 네트워크 인터페이스가 정의된 시작 템플릿은 `SecurityGroupIds` 파라미터를 사용한 보안 그룹 설정을 지원하지 않습니다. 구성하는 각 네트워크 인터페이스`Groups`에서에 대한 값을 설정해야 합니다.

```
{
    "Placement": {
        "GroupId": "pg-PlacementGroupId1"
    },
    "NetworkInterfaces": [
        {
            "DeviceIndex": 0,
            "InterfaceType": "efa",
            "NetworkCardIndex": 0,
            "SubnetId": "subnet-SubnetId1",
```

```
            "Groups": [
                "sg-SecurityGroupId1",
                "sg-EfaSecGroupId"
            ]
        },
        {
            "DeviceIndex": 1,
            "InterfaceType": "efa",
            "NetworkCardIndex": 1,
            "SubnetId": "subnet-SubnetId1"
            "Groups": ["sg-EfaSecGroupId"]
        }
    ]
}
```

EFA용 컴퓨팅 노드 그룹 생성 또는 업데이트

AWS PCS 컴퓨팅 노드 그룹에는 vCPUs, 프로세서 아키텍처 및 EFA 지원 수가 동일한 인스턴스가 포함되어야 합니다. EFA 소프트웨어가 설치된 AMI를 사용하고 EFA 지원 네트워크 인터페이스를 구성하는 시작 템플릿을 사용하도록 컴퓨팅 노드 그룹을 구성합니다.

(선택 사항) EFA 테스트

EFA 소프트웨어 설치에 포함된 `fi_pingpong` 프로그램을 실행하여 컴퓨팅 노드 그룹의 두 노드 간 EFA 지원 통신을 시연할 수 있습니다. 이 테스트가 성공하면 EFA가 제대로 구성된 것일 수 있습니다.

시작하려면 컴퓨팅 노드 그룹에 실행 중인 인스턴스 2개가 필요합니다. 컴퓨팅 노드 그룹에서 정적 용량을 사용하는 경우 사용 가능한 인스턴스가 이미 있어야 합니다. 동적 용량을 사용하는 컴퓨팅 노드 그룹의 경우 `salloc` 명령을 사용하여 두 개의 노드를 시작할 수 있습니다. 다음은 라는 대기열과 `hpc7g` 연결된 라는 동적 노드 그룹이 있는 클러스터의 예입니다`all`.

```
% salloc --nodes 2 -p all
salloc: Granted job allocation 6
salloc: Waiting for resource configuration
... a few minutes pass ...
salloc: Nodes hpc7g-[1-2] are ready for job
```

를 사용하여 할당된 두 노드의 IP 주소를 찾습니다`scontrol`. 다음 예제에서 주소는 `10.3.140.69`의 `hpc7g-1` 경우 ,의 `10.3.132.211` 경우 입니다`hpc7g-2`.

```
% scontrol show nodes hpc7g-[1-2]
```

```
NodeName=hpc7g-1 Arch=aarch64 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=64 CPUTot=64 CPULoad=0.00
   AvailableFeatures=hpc7g
   ActiveFeatures=hpc7g
   Gres=(null)
   NodeAddr=10.3.140.69 NodeHostName=ip-10-3-140-69 Version=25.05.5
   OS=Linux 5.10.218-208.862.amzn2.aarch64 #1 SMP Tue Jun 4 16:52:10 UTC 2024
   RealMemory=124518 AllocMem=0 FreeMem=110763 Sockets=64 Boards=1
   State=IDLE+CLOUD ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A
   Partitions=efa
   BootTime=2024-07-02T19:00:09 SlurmdStartTime=2024-07-08T19:33:25
   LastBusyTime=2024-07-08T19:33:25 ResumeAfterTime=None
   CfgTRES=cpu=64,mem=124518M,billing=64
   AllocTRES=
   CapWatts=n/a
   CurrentWatts=0 AveWatts=0
   ExtSensorsJoules=n/a ExtSensorsWatts=0 ExtSensorsTemp=n/a
   Reason=Maintain Minimum Number Of Instances [root@2024-07-02T18:59:00]
   InstanceId=i-04927897a9ce3c143 InstanceType=hpc7g.16xlarge

NodeName=hpc7g-2 Arch=aarch64 CoresPerSocket=1
   CPUAlloc=0 CPUEfctv=64 CPUTot=64 CPULoad=0.00
   AvailableFeatures=hpc7g
   ActiveFeatures=hpc7g
   Gres=(null)
   NodeAddr=10.3.132.211 NodeHostName=ip-10-3-132-211 Version=25.05.5
   OS=Linux 5.10.218-208.862.amzn2.aarch64 #1 SMP Tue Jun 4 16:52:10 UTC 2024
   RealMemory=124518 AllocMem=0 FreeMem=110759 Sockets=64 Boards=1
   State=IDLE+CLOUD ThreadsPerCore=1 TmpDisk=0 Weight=1 Owner=N/A MCS_label=N/A
   Partitions=efa
   BootTime=2024-07-02T19:00:09 SlurmdStartTime=2024-07-08T19:33:25
   LastBusyTime=2024-07-08T19:33:25 ResumeAfterTime=None
   CfgTRES=cpu=64,mem=124518M,billing=64
   AllocTRES=
   CapWatts=n/a
   CurrentWatts=0 AveWatts=0
   ExtSensorsJoules=n/a ExtSensorsWatts=0 ExtSensorsTemp=n/a
   Reason=Maintain Minimum Number Of Instances [root@2024-07-02T18:59:00]
   InstanceId=i-0a2c82623cb1393a7 InstanceType=hpc7g.16xlarge
```

SSH(또는 SSMhpc7g-1)를 사용하여 노드 중 하나(이 예제에서는)에 연결합니다. 이는 내부 IP 주소이므로 SSH를 사용하는 경우 로그인 노드 중 하나에서 연결해야 할 수 있습니다. 또한 컴퓨팅 노드 그룹 시작 템플릿을 통해 SSH 키로 인스턴스를 구성해야 합니다.

```
% ssh ec2-user@10.3.140.69
```

이제 서버 모드에서 fi_pingpong를 시작합니다.

```
/opt/amazon/efa/bin/fi_pingpong -p efa
```

두 번째 인스턴스(hpc7g-2)에 연결합니다.

```
% ssh ec2-user@10.3.132.211
```

fi_pingpong 클라이언트 모드에서 실행하여의 서버에 연결합니다hpc7g-1. 아래 예제와 유사한 출력이 표시되어야 합니다.

```
% /opt/amazon/efa/bin/fi_pingpong -p efa 10.3.140.69

bytes   #sent   #ack     total       time     MB/sec    usec/xfer   Mxfers/sec
64      10      =10      1.2k        0.00s      3.08      20.75       0.05
256     10      =10      5k          0.00s     21.24      12.05       0.08
1k      10      =10      20k         0.00s     82.91      12.35       0.08
4k      10      =10      80k         0.00s    311.48      13.15       0.08
[error] util/pingpong.c:1876: fi_close (-22) fid 0
```

(선택 사항) CloudFormation 템플릿을 사용하여 EFA 지원 시작 템플릿 생성

EFA를 설정하는 데 몇 가지 종속성이 있으므로 컴퓨팅 노드 그룹을 구성하는 데 사용할 수 있는 CloudFormation 템플릿이 제공됩니다. 최대 4개의 네트워크 카드가 있는 인스턴스를 지원합니다. 여러 네트워크 카드가 있는 인스턴스에 대한 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 탄력적 네트워크 인터페이스를 참조하세요.

다음 URL에서 CloudFormation 템플릿을 다운로드한 다음 AWS PCS를 사용하는 AWS 리전 의 CloudFormation 콘솔에 업로드합니다.

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/enable_efa/assets/pcs-lt-efa.yaml
```

CloudFormation 콘솔에서 템플릿을 연 상태에서 다음 값을 입력합니다. 템플릿은 몇 가지 기본 파라미터 값을 제공합니다. 기본값으로 그대로 둘 수 있습니다.

- 스택 이름 제공에서

  - 스택 이름에 설명이 포함된 이름을 입력합니다. 와 같이 AWS PCS 컴퓨팅 노드 그룹에 대해 선택할 이름을 통합하는 것이 좋습니다*NODEGROUPNAME*`-efa-lt`.
- 파라미터에서
  - NumberOfNetworkCards에서 노드 그룹에 있을 인스턴스의 네트워크 카드 수를 선택합니다.
  - VpcId에서 AWS PCS 클러스터가 배포되는 VPC를 선택합니다.
  - NodeGroupSubnetId에서 EFA 지원 인스턴스가 시작될 클러스터 VPC의 서브넷을 선택합니다.
  - PlacementGroupName에서 필드를 비워 두고 노드 그룹에 대한 새 클러스터 배치 그룹을 생성합니다. 사용하려는 기존 배치 그룹이 있는 경우 여기에 해당 이름을 입력합니다.
  - ClusterSecurityGroupId에서 클러스터의 다른 인스턴스와 AWS PCS API에 대한 액세스를 허용하는 데 사용할 보안 그룹을 선택합니다. 많은 고객이 클러스터 VPC에서 기본 보안 그룹을 선택합니다.
  - SshSecurityGroupId에서 클러스터의 노드에 대한 인바운드 SSH 액세스를 허용하는 데 사용하는 보안 그룹의 ID를 제공합니다.
  - SshKeyName에서 클러스터의 노드에 액세스할 SSH 키 페어를 선택합니다.
  - LaunchTemplateName에와 같은 시작 템플릿의 설명이 포함된 이름을 입력합니다*NODEGROUPNAME*`-efa-lt`. 이름은 AWS PCS를 사용할 AWS 리전 AWS 계정 의에 고유해야 합니다.
- 기능 아래
  - 가 IAM 리소스를 생성할 AWS CloudFormation 수 있음을 승인합니다 확인란을 선택합니다.

CloudFormation 스택의 상태를 모니터링합니다. 시작 템플릿`CREATE_COMPLETE`에 도달하면 사용할 준비가 된 것입니다. 위의에 설명된 대로 AWS PCS 컴퓨팅 노드 그룹과 함께 사용합니다EFA용 컴퓨팅 노드 그룹 생성 또는 업데이트.

AWS PCS에서 네트워크 파일 시스템 사용

AWS 병렬 컴퓨팅 서비스(AWS PCS) 컴퓨팅 노드 그룹에서 시작된 노드에 네트워크 파일 시스템을 연결하여 데이터와 파일을 작성하고 액세스할 수 있는 영구 위치를 제공할 수 있습니다. Amazon Elastic File System(Amazon EFS), Amazon FSx for Lustre, Amazon FSx for NetApp ONTAP, Amazon FSx for OpenZFS, Amazon File Cache 등 AWS 서비스에서 제공하는 파일 시스템을 사용할 수 있습니다. FSx OpenZFS NFS 서버와 같은 자체 관리형 파일 시스템을 사용할 수도 있습니다.

이 주제에서는 AWS PCS에서 네트워크 파일 시스템을 사용하기 위한 고려 사항과 예제를 다룹니다.

네트워크 파일 시스템 사용에 대한 고려 사항

다양한 파일 시스템의 구현 세부 정보는 다르지만 몇 가지 일반적인 고려 사항이 있습니다.

- 인스턴스에 관련 파일 시스템 소프트웨어를 설치해야 합니다. 예를 들어 Amazon FSx for Lustre를 사용하려면 적절한 Lustre 패키지가 있어야 합니다. 컴퓨팅 노드 그룹 AMI에 포함하거나 인스턴스 부팅 시 실행되는 스크립트를 사용하여이 작업을 수행할 수 있습니다.
- 공유 네트워크 파일 시스템과 컴퓨팅 노드 그룹 인스턴스 사이에 네트워크 경로가 있어야 합니다.
- 공유 네트워크 파일 시스템과 컴퓨팅 노드 그룹 인스턴스 모두에 대한 보안 그룹 규칙은 관련 포트에 대한 연결을 허용해야 합니다.
- 파일 시스템에 액세스하는 리소스 간에 일관된 POSIX 사용자 및 그룹 네임스페이스를 유지해야 합니다. 그렇지 않으면 PCS 클러스터에서 실행되는 작업 및 대화형 프로세스에 권한 오류가 발생할 수 있습니다.
- 파일 시스템 탑재는 EC2 시작 템플릿을 사용하여 수행됩니다. 네트워크 파일 시스템을 탑재할 때 발생하는 오류 또는 제한 시간으로 인해 인스턴스를 작업 실행에 사용할 수 없게 될 수 있습니다. 이로 인해 예기치 않은 비용이 발생할 수 있습니다. 시작 템플릿 디버깅에 대한 자세한 내용은 섹션을 참조하세요AWS PCS에서 Amazon EC2 시작 템플릿 사용.

네트워크 탑재 예제

Amazon EFS, Amazon FSx for Lustre, Amazon FSx for NetApp ONTAP, Amazon FSx for OpenZFS 및 Amazon File Cache를 사용하여 파일 시스템을 생성할 수 있습니다. 아래 관련 섹션을 확장하여 각 네트워크 탑재의 예를 확인하세요.

Amazon EFS

파일 시스템 설정

Amazon EFS 파일 시스템을 생성합니다. PCS 컴퓨팅 노드 그룹 인스턴스를 시작할 각 가용 영역에 탑재 대상이 있는지 확인합니다. 또한 각 탑재 대상이 PCS 컴퓨팅 노드 그룹 인스턴스에서 인바운드 및 아웃바운드 액세스를 허용하는 보안 그룹과 연결되어 있는지 확인합니다. 자세한 내용은 Amazon Elastic File System 사용 설명서의 탑재 대상 및 보안 그룹을 참조하세요.

시작 템플릿

파일 시스템 설정의 보안 그룹(들)을 컴퓨팅 노드 그룹에 사용할 시작 템플릿에 추가합니다.

`cloud-config` 메커니즘을 사용하여 Amazon EFS 파일 시스템을 탑재하는 사용자 데이터를 포함합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- *`mount-point-directory`* - Amazon EFS를 탑재할 각 인스턴스의 경로
- *`filesystem-id`* - EFS 파일 시스템의 파일 시스템 ID

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
  - amazon-efs-utils

runcmd:
  - mkdir -p /mount-point-directory
  - echo "filesystem-id:/ /mount-point-directory efs tls,_netdev" >> /etc/fstab
  - mount -a -t efs defaults

--==MYBOUNDARY==--
```

Amazon FSx for Lustre

파일 시스템 설정

AWS PCS를 사용할 VPC에서 FSx for Lustre 파일 시스템을 생성합니다. 영역 간 전송을 최소화하려면 대부분의 PCS 컴퓨팅 노드 그룹 인스턴스를 시작할 동일한 가용 영역의 서브넷에 배포합니다. 파일 시

스템이 PCS 컴퓨팅 노드 그룹 인스턴스에서 인바운드 및 아웃바운드 액세스를 허용하는 보안 그룹과 연결되어 있는지 확인합니다. 보안 그룹에 대한 자세한 내용은 Amazon FSx for Lustre 사용 설명서의 Amazon VPC를 사용한 파일 시스템 액세스 제어를 참조하세요. FSx

시작 템플릿

가 FSx for Lustre 파일 시스템을 탑재하는 `cloud-config` 데 사용하는 사용자 데이터를 포함합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- *`mount-point-directory`* - FSx for Lustre를 탑재하려는 인스턴스의 경로
- *`filesystem-id`* - FSx for Lustre 파일 시스템의 파일 시스템 ID
- *`mount-name`* - FSx for Lustre 파일 시스템의 탑재 이름
- *`region-code`* - FSx for Lustre 파일 시스템이 배포되는 AWS 리전 입니다(AWS PCS 시스템과 동일해야 함).
- (선택 사항) *`latest`*- FSx for Lustre에서 Lustre 지원하는 모든 버전

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- amazon-linux-extras install -y lustre=latest
- mkdir -p /mount-point-directory
- mount -t lustre filesystem-id.fsx.region-code.amazonaws.com@tcp:/mount-name /mount-
point-directory

--==MYBOUNDARY==--
```

Amazon FSx for NetApp ONTAP

파일 시스템 설정

AWS PCS를 사용할 VPC에서 Amazon FSx for NetApp ONTAP 파일 시스템을 생성합니다. 영역 간 전송을 최소화하려면 대부분의 AWS PCS 컴퓨팅 노드 그룹 인스턴스를 시작할 동일한 가용 영역의 서브넷에 배포합니다. 파일 시스템이 AWS PCS 컴퓨팅 노드 그룹 인스턴스에서 인바운드 및 아웃바운드 액세스를 허용하는 보안 그룹과 연결되어 있는지 확인합니다. 보안 그룹에 대한 자세한 내용은 FSx for ONTAP 사용 설명서의 Amazon VPC를 사용한 파일 시스템 액세스 제어를 참조하세요.

시작 템플릿

가 FSx for ONTAP 파일 시스템의 루트 볼륨을 탑재하는 `cloud-config` 데 사용하는 사용자 데이터를 포함합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- *`mount-point-directory`* - FSx for ONTAP 볼륨을 탑재하려는 인스턴스의 경로
- *`svm-id`* - FSx for ONTAP 파일 시스템의 SVM ID
- *`filesystem-id`* - FSx for ONTAP 파일 시스템의 파일 시스템 ID
- *`region-code`* - FSx for ONTAP 파일 시스템이 배포되는 입니다 AWS 리전 (AWS PCS 시스템과 동일해야 함).
- *`volume-name`* - FSx for ONTAP 볼륨 이름

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- mkdir -p /mount-point-directory
- mount -t nfs svm-id.filesystem-id.fsx.region-code.amazonaws.com:/volume-name /mount-point-directory

--==MYBOUNDARY==--
```

Amazon FSx for OpenZFS

파일 시스템 설정

AWS PCS를 사용할 VPC에서 FSx for OpenZFS 파일 시스템을 생성합니다. 영역 간 전송을 최소화하려면 대부분의 AWS PCS 컴퓨팅 노드 그룹 인스턴스를 시작할 동일한 가용 영역의 서브넷에 배포합니다. 파일 시스템이 AWS PCS 컴퓨팅 노드 그룹 인스턴스에서 인바운드 및 아웃바운드 액세스를 허용하는 보안 그룹과 연결되어 있는지 확인합니다. 보안 그룹에 대한 자세한 내용은 FSx for OpenZFS 사용 설명서의 Amazon VPC를 사용한 파일 시스템 액세스 관리를 참조하세요.

시작 템플릿

가 FSx for OpenZFS 파일 시스템의 루트 볼륨을 탑재하는 `cloud-config` 데 사용하는 사용자 데이터를 포함합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- *mount-point-directory* - FSx for OpenZFS 공유를 탑재하려는 인스턴스의 경로
- *filesystem-id* - FSx for OpenZFS 파일 시스템의 파일 시스템 ID
- *region-code* - FSx for OpenZFS 파일 시스템이 배포되는 AWS 리전 입니다(AWS PCS 시스템과 동일해야 함).

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- mkdir -p /mount-point-directory
- mount -t nfs -o noatime,nfsvers=4.2,sync,rsize=1048576,wsize=1048576 filesystem-
id.fsx.region-code.amazonaws.com:/fsx/ /mount-point-directory

--==MYBOUNDARY==--
```

Amazon File Cache

파일 시스템 설정

AWS PCS를 사용할 VPC에서 Amazon File Cache를 생성합니다. 영역 간 전송을 최소화하려면 대부분의 PCS 컴퓨팅 노드 그룹 인스턴스를 시작할 동일한 가용 영역에서 서브넷을 선택합니다. 파일 캐시가 포트 988에서 PCS 인스턴스와 파일 캐시 간의 인바운드 및 아웃바운드 트래픽을 허용하는 보안 그룹과 연결되어 있는지 확인합니다. 보안 그룹에 대한 자세한 내용은 Amazon File Cache 사용 설명서의 Amazon VPC를 사용한 캐시 액세스 제어를 참조하세요.

시작 템플릿

파일 시스템 설정의 보안 그룹(들)을 컴퓨팅 노드 그룹에 사용할 시작 템플릿에 추가합니다.

가 Amazon File Cache를 탑재하는 `cloud-config` 데 사용하는 사용자 데이터를 포함합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- *mount-point-directory* - FSx for Lustre를 탑재하려는 인스턴스의 경로
- *cache-dns-name* - 파일 캐시의 도메인 이름 시스템(DNS) 이름
- *mount-name* - 파일 캐시의 탑재 이름

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

runcmd:
- amazon-linux-extras install -y lustre=2.12
- mkdir -p /mount-point-directory
- mount -t lustre -o relatime,flock cache-dns-name@tcp:/mount-name /mount-point-
directory

--==MYBOUNDARY==--
```

AWS PCS용 Amazon Machine Image(AMIs)

AWS PCS는 사용자가 제공하는 AMIs와 함께 작동하므로 클러스터의 노드에 있는 소프트웨어 및 구성에 뛰어난 유연성을 제공합니다. AWS PCS를 시도하는 경우에서 제공하고에서 유지 관리하는 샘플 AMI를 사용할 수 있습니다 AWS. 프로덕션 환경에서 AWS PCS를 사용하는 경우 자체 AMIs를 빌드하는 것이 좋습니다. 이 주제에서는 샘플 AMIs를 검색하고 사용하는 방법과 사용자 지정 AMIs를 직접 빌드하고 사용하는 방법을 다룹니다.

주제

AWS PCS에서 샘플 Amazon Machine Image(AMIs) 사용

AWS는 AWS PCS 작업의 시작점으로 사용할 수 있는 샘플 AMIs를 제공합니다.

> **Important**
>
> 샘플 AMIs는 데모용이며 프로덕션 워크로드에는 권장되지 않습니다.

> **Important**
>
> 서브넷이 IPv6만 사용하도록 구성된 경우 AWS PCS 샘플 AMIs 및 여러 네트워크 인터페이스로 구성된 컴퓨팅 노드 그룹은 현재 작동하지 않습니다. 대신 듀얼 스택 서브넷(IPv4 및 IPv6) 또는 IPv4-only 서브넷을 사용합니다.

현재 AWS PCS 샘플 AMIs 찾기

AWS Management Console

AWS PCS 샘플 AMIs

```
aws-pcs-sample_ami-OS-architecture-scheduler-scheduler-major-version
```

허용되는 값

- *OS* - amzn2
- *####* - x86_64 또는 arm64
- *####* - slurm
- *scheduler-major-version* – 25.05

AWS PCS 샘플 AMIs 찾으려면

1. Amazon EC2 콘솔을 엽니다.
2. AMIs로 이동합니다.
3. 퍼블릭 이미지를 선택합니다.
4. 속성 또는 태그별로 AMI 찾기에서 템플릿 이름을 사용하여 AMI를 검색합니다.

   예시

   - Arm64 인스턴스의 Slurm 25.05용 샘플 AMI

     ```
     aws-pcs-sample_ami-amzn2-arm64-slurm-25.05
     ```

   - x86 인스턴스의 Slurm 25.05용 샘플 AMI

     ```
     aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05
     ```

   > **Note**
   >
   > AMIs가 여러 개 있는 경우 최신 타임스탬프가 있는 AMI를 사용합니다.

5. 컴퓨팅 노드 그룹을 생성하거나 업데이트할 때 AMI ID를 사용합니다.

AWS CLI

다음 명령을 사용하여 최신 AWS PCS 샘플 AMI를 찾을 수 있습니다. *region-code*를와 같이 AWS PCS를 AWS 리전 사용하는 로 바꿉니다us-east-1.

- x86_64

```
aws ec2 describe-images --region region-code --owners amazon \
--filters 'Name=name,Values=aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05*' \
            'Name=state,Values=available' \
--query 'sort_by(Images, &CreationDate)[-1].[Name,ImageId]' --output text
```

- Arm64

```
aws ec2 describe-images --region region-code --owners amazon \
--filters 'Name=name,Values=aws-pcs-sample_ami-amzn2-arm64-slurm-25.05*' \
            'Name=state,Values=available' \
--query 'sort_by(Images, &CreationDate)[-1].[Name,ImageId]' --output text
```

컴퓨팅 노드 그룹을 생성하거나 업데이트할 때 AMI ID를 사용합니다.

AWS PCS 샘플 AMIs 대해 자세히 알아보기

AWS PCS 샘플 AMIsAWS PCS 샘플 AMIs 릴리스 정보.

AWS PCS와 호환되는 자체 AMIs 구축

AWS PCS로 작동하는 자체 AMIs를 빌드하는 방법을 알아보려면 섹션을 참조하세요AWS PCS용 사용자 지정 Amazon Machine Image(AMIs).

AWS PCS용 사용자 지정 Amazon Machine Image(AMIs)

AWS PCS는 서비스에 가져오는 Amazon Machine Image(AMI)와 함께 작동하도록 설계되었습니다. 이러한 AMIs에는 임의의 소프트웨어 및 구성이 설치되어 있을 수 있습니다. 단, AWS PCS 에이전트와 호환되는 버전의 Slurm이 올바르게 설치 및 구성되어 있어야 합니다. 사용자 지정 AMI에 AWS PCS 소프트웨어를 설치하려면 AWS제공 설치 관리자를 사용해야 합니다. AWS제공 설치 관리자를 사용하여 사용자 지정 AMI에 Slurm을 설치하는 것이 좋지만 원하는 경우 직접 Slurm을 설치할 수 있습니다(권장되지 않음).

> **Note**
>
> 사용자 지정 AMI를 빌드하지 않고 AWS PCS를 시도하려는 경우에서 제공하는 샘플 AMI를 사용할 수 있습니다 AWS. 자세한 내용은 AWS PCS에서 샘플 Amazon Machine Image(AMIs) 사용 단원을 참조하십시오.

> **Important**
>
> AWS 현재는 IPv6 전용 네트워크에서 PCS를 사용하는 경우에도 로컬 노드 통신에 대해 IPv4를 지원하는 커널이 AWS 필요합니다. IPv6-only

이 자습서는 PCS 컴퓨팅 노드 그룹과 함께 사용하여 HPC 및 AI/ML 워크로드를 구동할 수 있는 AMI를 생성하는 데 도움이 됩니다.

주제

1단계 - 임시 인스턴스 시작

AWS PCS 소프트웨어 및 Slurm 스케줄러를 설치하고 구성하는 데 사용할 수 있는 임시 인스턴스를 시작합니다. 이 인스턴스를 사용하여 AWS PCS와 호환되는 AMI를 생성합니다.

임시 인스턴스를 실행합니다.

1. Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택한 다음 인스턴스 시작을 선택하여 새 인스턴스 시작 마법사를 엽니다.

3. (선택 사항) 이름 및 태그 섹션에서와 같은 인스턴스의 이름을 제공합니다`PCS-AMI-instance`. 이름은 인스턴스에 리소스 태그(`Name=PCS-AMI-instance`)로 할당됩니다.
4. Application and OS Images(애플리케이션 및 OS 이미지) 섹션에서 지원되는 운영 체제 중 하나에 해당하는 AMI를 선택합니다.
5. 인스턴스 유형(Instance type) 섹션에서 지원되는 인스턴스 유형(supported instance type)을 선택합니다.
6. 키 페어(Key pair) 섹션에서 인스턴스에 사용할 키 페어를 선택합니다.
7. 네트워크 설정 섹션에서:
   - 방화벽(보안 그룹)에서 기존 보안 그룹 선택을 선택한 다음 인스턴스에 대한 인바운드 SSH 액세스를 허용하는 보안 그룹을 선택합니다.
8. 스토리지(Storage) 섹션에서 필요에 따라 볼륨을 구성합니다. 자체 애플리케이션과 라이브러리를 설치하기에 충분한 공간을 구성해야 합니다.
9. 요약(Summary) 패널에서 인스턴스 실행(Launch instance)을 선택합니다.

2단계 - AWS PCS 에이전트 설치

Slurm과 함께 사용하도록 AWS PCS에서 시작한 인스턴스를 구성하는 에이전트를 설치합니다. AWS PCS 에이전트에 대한 자세한 내용은 섹션을 참조하세요AWS PCS 에이전트 버전.

AWS PCS 에이전트를 설치하려면

1. 시작한 인스턴스에 연결합니다. 자세한 내용은 Linux 인스턴스에 연결을 참조하세요.
2. (선택 사항) 모든 소프트웨어 패키지가 최신 상태인지 확인하려면 인스턴스에서 빠른 소프트웨어 업데이트를 수행합니다. 이 프로세스는 몇 분 정도 걸릴 수 있습니다.
   - Amazon Linux 2, Amazon Linux 2023, RHEL 9, RHEL 8, Rocky Linux 9 및 Rocky Linux 8

     ```
     sudo yum update -y
     ```

   - Ubuntu 22.04 및 Ubuntu 24.04

     ```
     sudo apt-get update && sudo apt-get upgrade -y
     ```

3. 인스턴스를 재부팅하고 다시 연결합니다.

4. AWS PCS 에이전트 설치 파일을 다운로드합니다. 설치 파일은 압축된 tarball(`.tar.gz`) 파일로 패키징됩니다. 최신 안정 버전을 다운로드하려면 다음 명령을 사용하십시오. *##*을와 같이 임시 인스턴스 AWS 리전 를 시작한 로 대체합니다`us-east-1`.

```
curl https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.2-1.tar.gz -o aws-pcs-agent-v1.3.2-1.tar.gz
```

`latest` 이전 명령에서 버전 번호를 로 대체하여 최신 버전을 가져올 수도 있습니다(예: `aws-pcs-agent-v1-latest.tar.gz`).

> **Note**
>
> 이는 향후 AWS PCS 에이전트 소프트웨어 릴리스에서 변경될 수 있습니다.

5. (선택 사항) AWS PCS 소프트웨어 tarball의 신뢰성과 무결성을 확인합니다. 이 작업을 수행하여 소프트웨어 게시자의 자격 증명을 확인하고 파일이 게시된 이후 변경되거나 손상되지 않았는지 확인하는 것이 좋습니다.

   a. AWS PCS용 퍼블릭 GPG 키를 다운로드하여 키링으로 가져옵니다. *##*을 임시 인스턴스를 시작한 AWS 리전 로 대체합니다. 명령에서 키 값이 반환됩니다. 키 값을 기록합니다. 다음 단계에서 사용합니다.

   ```
   wget https://aws-pcs-repo-public-keys-region.s3.region.amazonaws.com/aws-pcs-public-key.pub && \
       gpg --import aws-pcs-public-key.pub
   ```

   b. 다음 명령을 실행하여 GPG 키의 지문을 확인합니다.

   ```
   gpg --fingerprint 7EEF030EDDF5C21C
   ```

   명령은 다음과 동일한 지문을 반환해야 합니다.

   ```
   1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
   ```

   > **Important**
   >
   > 지문이 일치하지 않는 경우 AWS PCS 에이전트 설치 스크립트를 실행하지 마십시오. AWS Support에 문의하세요.

c. 서명 파일을 다운로드하고 AWS PCS 소프트웨어 tarball 파일의 서명을 확인합니다. *region*을와 같이 임시 인스턴스 AWS 리전 를 시작한 로 바꿉니다us-east-1.

```
wget https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.2-1.tar.gz.sig && \
    gpg --verify ./aws-pcs-agent-v1.3.2-1.tar.gz.sig
```

다음과 유사하게 출력됩니다.

```
gpg: assuming signed data in './aws-pcs-agent-v1.3.2-1.tar.gz'
gpg: Signature made Thu 06 Nov 2025 11:10:36 AM CET using RSA key ID ECC0AE5C
gpg: Good signature from "AWS PCS Packages (AWS PCS Packages)"
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
   Subkey fingerprint: B7E1 8788 3517 6A74 C3D5  EAF5 6088 136D ECC0 AE5C
```

결과에가 포함되어 `Good signature` 있고 지문이 이전 단계에서 반환된 지문과 일치하는 경우 다음 단계로 진행합니다.

> **Important**
>
> 지문이 일치하지 않는 경우 AWS PCS 소프트웨어 설치 스크립트를 실행하지 마십시오. AWS Support에 문의하세요.

6. 압축된 파일에서 `.tar.gz` 파일을 추출하고 추출된 디렉터리로 이동합니다.

```
tar -xf aws-pcs-agent-v1.3.2-1.tar.gz && \
    cd aws-pcs-agent
```

7. AWS PCS 소프트웨어를 설치합니다.

```
sudo ./installer.sh
```

8. AWS PCS 소프트웨어 버전 파일을 확인하여 설치가 성공했는지 확인합니다.

```
cat /opt/aws/pcs/version
```

다음과 유사하게 출력됩니다.

```
AGENT_INSTALL_DATE='Fri Dec 13 12:28:43 UTC 2024'
AGENT_VERSION='1.3.2'
AGENT_RELEASE='1'
```

3단계 - Slurm 설치

AWS PCS와 호환되는 Slurm 버전을 설치합니다. 자세한 내용은 AWS PCS의 Slurm 버전 단원을 참조하십시오.

> **Note**
>
> 이전 버전의 Slurm 소프트웨어가 설치된 AMI가 있는 경우 다음 단계를 수행하여 새 버전의 Slurm을 설치해야 합니다. AWS PCS 에이전트는 클러스터 생성 시 구성된 Slurm 버전에 따라 런타임 시 올바른 버전의 Slurm 바이너리를 활성화합니다.

Slurm을 설치하려면

1. AWS PCS 소프트웨어를 설치한 동일한 임시 인스턴스에 연결합니다.
2. Slurm 설치 관리자 소프트웨어를 다운로드합니다. Slurm 설치 프로그램은 압축된 tarball(`.tar.gz`) 파일로 패키징됩니다. 최신 안정 버전을 다운로드하려면 다음 명령을 사용하십시오. *##*을와 같은 임시 인스턴스 AWS 리전 의 로 대체합니다`us-east-1`.

   ```
   curl https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz \
        -o aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz
   ```

   `latest` 이전 명령에서 버전 번호를 로 대체하여 최신 버전을 가져올 수도 있습니다(예: `aws-pcs-slurm-25.05-installer-latest.tar.gz`). 체크섬이 있는 사용 가능한 버전의 전체 목록은 섹션을 참조하세요AWS PCS의 Slurm 버전.

   > **Note**
   >
   > 이는 Slurm 설치 프로그램 소프트웨어의 향후 릴리스에서 변경될 수 있습니다.

3. (선택 사항) Slurm 설치 관리자 tarball의 신뢰성과 무결성을 확인합니다. 이 작업을 수행하여 소프트웨어 게시자의 자격 증명을 확인하고 파일이 게시된 이후 변경되거나 손상되지 않았는지 확인하는 것이 좋습니다.

   a. AWS PCS용 퍼블릭 GPG 키를 다운로드하여 키링으로 가져옵니다. *##*을 임시 인스턴스를 시작한 AWS 리전 로 대체합니다. 명령에서 키 값이 반환됩니다. 키 값을 기록합니다. 다음 단계에서 사용합니다.

   ```
   wget https://aws-pcs-repo-public-keys-region.s3.region.amazonaws.com/aws-pcs-public-key.pub && \
       gpg --import aws-pcs-public-key.pub
   ```

   b. 다음 명령을 실행하여 GPG 키의 지문을 확인합니다.

   ```
   gpg --fingerprint 7EEF030EDDF5C21C
   ```

   명령은 다음과 동일한 지문을 반환해야 합니다.

   ```
   1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
   ```

   > **Important**
   >
   > 지문이 일치하지 않는 경우 Slurm 설치 스크립트를 실행하지 마십시오. AWS Support에 문의하세요.

   c. 서명 파일을 다운로드하고 Slurm 설치 관리자 tarball 파일의 서명을 확인합니다. *region*을와 같이 임시 인스턴스 AWS 리전 를 시작한 로 바꿉니다`us-east-1`.

   ```
   wget https://aws-pcs-repo-region.s3.region.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz.sig && \
       gpg --verify ./aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz.sig
   ```

   다음과 유사하게 출력됩니다.

   ```
   gpg: assuming signed data in './aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz'
   gpg: Signature made Fri 14 Nov 2025 11:35:15 AM UTC using RSA key ID ECC0AE5C
   gpg: Good signature from "AWS PCS Packages (AWS PCS Packages)"
   gpg: WARNING: This key is not certified with a trusted signature!
   gpg:          There is no indication that the signature belongs to the owner.
   ```

```
Primary key fingerprint: 1C24 32C1 862F 64D1 F90A  239A 7EEF 030E DDF5 C21C
     Subkey fingerprint: B7E1 8788 3517 6A74 C3D5  EAF5 6088 136D ECC0 AE5C
```

결과에가 포함되어 `Good signature` 있고 지문이 이전 단계에서 반환된 지문과 일치하는 경우 다음 단계로 진행합니다.

> **Important**
>
> 지문이 일치하지 않는 경우 Slurm 설치 스크립트를 실행하지 마십시오. AWS Support에 문의하세요.

4. 압축 `.tar.gz` 파일에서 파일을 추출하고 추출된 디렉터리로 이동하십시오.

```
tar -xf aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz && \
    cd aws-pcs-slurm-25.05-installer
```

5. Slurm을 설치합니다. 설치 관리자는 Slurm 및 해당 종속 항목을 다운로드, 컴파일 및 설치합니다. 선택한 임시 인스턴스의 사양에 따라 몇 분 정도 걸립니다.

```
sudo ./installer.sh -y
```

6. 스케줄러 버전 파일을 확인하여 설치를 확인합니다.

```
cat /opt/aws/pcs/scheduler/slurm-25.05/version
```

다음과 유사하게 출력됩니다.

```
SLURM_INSTALL_DATE='Fri Nov 14 15:15:37 UTC 2025'
SLURM_VERSION='25.05.5'
PCS_SLURM_RELEASE='1'
```

4단계 - (선택 사항) 추가 드라이버, 라이브러리 및 애플리케이션 소프트웨어 설치

임시 인스턴스에 추가 드라이버, 라이브러리 및 애플리케이션 소프트웨어를 설치합니다. 설치 절차는 특정 애플리케이션 및 라이브러리에 따라 달라집니다. 이전에 AWS PCS용 사용자 지정 AMI를 빌드하지 않은 경우 먼저 AWS PCS 소프트웨어 및 Slurm만 설치된 AMI를 빌드하고 테스트한 다음 초기 성공을 확인한 후 자체 소프트웨어 및 구성을 점진적으로 추가하는 것이 좋습니다.

예제

- Elastic Fabric Adapter(EFA) 소프트웨어. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2의 HPC 워크로드용 EFA 및 MPI 시작하기를 참조하세요.
- Amazon Elastic File System(Amazon EFS) 클라이언트. 자세한 내용은 Amazon Elastic File System 사용 설명서의 Amazon EFS 클라이언트 수동 설치를 참조하세요. Amazon Elastic File System
- Lustre 클라이언트: Amazon FSx for Lustre 및 Amazon File Cache를 사용합니다. 자세한 내용은 FSx for Lustre 사용 설명서의 Lustre 클라이언트 설치를 참조하세요.
- Amazon CloudWatch CloudWatch 에이전트. 자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트 설치를 참조하세요. Amazon CloudWatch
- AWS Neuron: trn* 및 inf* 인스턴스 유형을 사용합니다. 자세한 내용은 AWS Neuron 설명서를 참조하세요.
- p* 또는 g* 인스턴스 유형을 사용하기 위한 NVIDIA 드라이버, CUDA 및 DCGM.

5단계 - AWS PCS와 호환되는 AMI 생성

필요한 소프트웨어 구성 요소를 설치한 후 AWS PCS 컴퓨팅 노드 그룹에서 인스턴스를 시작하는 데 재사용할 수 있는 AMI를 생성합니다.

> **Important**
>
> AWS 현재는 IPv6 전용 네트워크에서 PCS를 사용하는 경우에도 로컬 노드 통신에 대해 IPv4를 지원하는 커널이 AWS 필요합니다. IPv6-only

임시 인스턴스에서 AMI를 생성하려면

1. Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택합니다.
3. 생성한 임시 인스턴스를 선택합니다. 작업, 이미지, 이미지 생성을 선택합니다.
4. [이미지 생성(Create image)]에서 다음을 수행합니다.
   a. [이미지 이름(Image name)]에 AMI를 설명하는 이름을 입력합니다.
   b. (선택 사항) [이미지 설명(Image description)]에 AMI의 용도에 대한 간략한 설명을 입력합니다.

c. 이미지 생성(Create image)을 선택합니다.

5. 탐색 창에서 AMI를 선택합니다.
6. 목록에서 생성한 AMI를 찾습니다. 상태가 보류 중에서 사용 가능으로 변경될 때까지 기다린 다음 AWS PCS 컴퓨팅 노드 그룹과 함께 사용합니다.

6단계 - AWS PCS 컴퓨팅 노드 그룹에 사용자 지정 AMI 사용

사용자 지정 AMI를 새 또는 기존 AWS PCS 컴퓨팅 노드 그룹과 함께 사용할 수 있습니다.

> **Important**
>
> AWS 현재는 IPv6 전용 네트워크에서 PCS를 사용하는 경우에도 로컬 노드 통신에 대해 IPv4를 지원하는 커널이 AWS 필요합니다. IPv6-only

New compute node group

사용자 지정 AMI를 사용하려면

1. AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 사용자 지정 AMI를 사용할 클러스터를 선택한 다음 컴퓨팅 노드 그룹을 선택합니다.
4. 새 컴퓨팅 노드 그룹을 생성합니다. 자세한 내용은 AWS PCS에서 컴퓨팅 노드 그룹 생성 단원을 참조하십시오. AMI ID에서 사용하려는 사용자 지정 AMI의 이름 또는 ID를 검색합니다. 컴퓨팅 노드 그룹 구성을 완료한 다음 컴퓨팅 노드 그룹 생성을 선택합니다.
5. (선택 사항) AMI가 인스턴스 시작을 지원하는지 확인합니다. 컴퓨팅 노드 그룹에서 인스턴스를 시작합니다. 단일 정적 인스턴스를 갖도록 컴퓨팅 노드 그룹을 구성하거나 컴퓨팅 노드 그룹을 사용하는 대기열에 작업을 제출할 수 있습니다.

   a. 새 컴퓨팅 노드 그룹 ID로 태그가 지정된 인스턴스가 나타날 때까지 Amazon EC2 콘솔을 확인합니다. 이에 대한 자세한 내용은 단원을 참조하십시오AWS PCS에서 컴퓨팅 노드 그룹 인스턴스 찾기.

   b. 인스턴스 시작이 표시되고 부트스트랩 프로세스가 완료되면 인스턴스가 예상 AMI를 사용하고 있는지 확인합니다. 이렇게 하려면 인스턴스를 선택한 다음 세부 정보에서 AMI ID를 검사합니다. 컴퓨팅 노드 그룹 설정에서 구성한 AMI와 일치해야 합니다.

   c. (선택 사항) 컴퓨팅 노드 그룹 조정 구성을 원하는 값으로 업데이트합니다.

Existing compute node group

사용자 지정 AMI를 사용하려면

1. AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 사용자 지정 AMI를 사용할 클러스터를 선택한 다음 컴퓨팅 노드 그룹을 선택합니다.
4. 구성할 노드 그룹을 선택하고 편집을 선택합니다. AMI ID에서 사용하려는 사용자 지정 AMI의 이름 또는 ID를 검색합니다. 컴퓨팅 노드 그룹 구성을 완료한 다음 업데이트를 선택합니다. 컴퓨팅 노드 그룹에서 시작된 새 인스턴스는 업데이트된 AMI ID를 사용합니다. 기존 인스턴스는 AWS PCS가 교체할 때까지 이전 AMI를 계속 사용합니다. 자세한 내용은 AWS PCS 컴퓨팅 노드 그룹 업데이트 단원을 참조하십시오.
5. (선택 사항) AMI가 인스턴스 시작을 지원하는지 확인합니다. 컴퓨팅 노드 그룹에서 인스턴스를 시작합니다. 단일 정적 인스턴스를 갖도록 컴퓨팅 노드 그룹을 구성하거나 컴퓨팅 노드 그룹을 사용하는 대기열에 작업을 제출할 수 있습니다.

   a. 새 컴퓨팅 노드 그룹 ID로 태그가 지정된 인스턴스가 나타날 때까지 Amazon EC2 콘솔을 확인합니다. 이에 대한 자세한 내용은 단원을 참조하십시오AWS PCS에서 컴퓨팅 노드 그룹 인스턴스 찾기.

   b. 인스턴스 시작이 표시되고 부트스트랩 프로세스가 완료되면 인스턴스가 예상 AMI를 사용하고 있는지 확인합니다. 이렇게 하려면 인스턴스를 선택한 다음 세부 정보에서 AMI ID를 검사합니다. 컴퓨팅 노드 그룹 설정에서 구성한 AMI와 일치해야 합니다.

   c. (선택 사항) 컴퓨팅 노드 그룹 조정 구성을 원하는 값으로 업데이트합니다.

7단계 - 임시 인스턴스 종료

AMI가 AWS PCS에서 의도한 대로 작동하는지 확인한 후 임시 인스턴스를 종료하여 요금 발생을 중지할 수 있습니다.

임시 인스턴스 종료

1. Amazon EC2 콘솔을 엽니다.
2. 탐색 창에서 인스턴스를 선택합니다.
3. 생성한 임시 인스턴스를 선택하고 작업, 인스턴스 상태, 인스턴스 종료를 선택합니다.
4. 확인 메시지가 표시되면 종료를 선택합니다.

AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자

AWS 는 인스턴스에 AWS PCS 소프트웨어를 설치할 수 있는 다운로드 가능한 파일을 제공합니다.는 관련 버전의 Slurm 및 해당 종속 항목을 다운로드, 컴파일 및 설치할 수 있는 소프트웨어 AWS 도 제공합니다. 이 지침을 사용하여 AWS PCS와 함께 사용할 사용자 지정 AMIs를 사용할 수 있습니다.

목차

AWS PCS 에이전트 소프트웨어 설치 관리자

AWS PCS 에이전트 소프트웨어 설치 관리자는 인스턴스 부트스트랩 프로세스 중에 AWS PCS와 함께 작동하도록 인스턴스를 구성합니다. 사용자 지정 AMI에 AWS PCS 에이전트를 설치하려면 AWS제공 설치 관리자를 사용해야 합니다.

AWS PCS 에이전트 소프트웨어에 대한 자세한 내용은 섹션을 참조하세요AWS PCS 에이전트 버전.

Slurm 설치 관리자

Slurm 설치 관리자는 관련 버전의 Slurm 및 해당 종속 항목을 다운로드, 컴파일 및 설치합니다. Slurm 설치 관리자를 사용하여 AWS PCS용 사용자 지정 AMIs를 빌드할 수 있습니다. Slurm 설치 관리자가 제공하는 소프트웨어 구성과 일치하는 경우 자체 메커니즘을 사용할 수도 있습니다. Slurm에 대한 AWS PCS 지원에 대한 자세한 내용은 섹션을 참조하세요AWS PCS의 Slurm 버전.

AWS제공 소프트웨어는 다음을 설치합니다.

- 요청된 메이저 및 유지 관리 버전의 Slurm(현재 버전 25.05.x) - 라이선스 GPL 2
  - Slurm은를 로 `--sysconfdir` 설정하여 빌드됩니다. `/etc/slurm`
  - Slurm은 `--enable-pam` 및 옵션을 사용하여 빌드됩니다. `--without-munge`
  - Slurm은 옵션으로 빌드됩니다. `--sharedstatedir=/run/slurm/`

  - Slurm은 PMIX 및 JWT 지원으로 구축되었습니다.
  - Slurm은에 설치됩니다. `/opt/aws/pcs/schedulers/slurm-25.05`
- OpenPMIX(버전 4.2.6) – 라이선스
  - OpenPMIX는의 하위 디렉터리로 설치됩니다. `/opt/aws/pcs/scheduler/`
- libjwt(버전 1.17.0) – 라이선스 MPL-2.0
  - libjwt는의 하위 디렉터리로 설치됩니다. `/opt/aws/pcs/scheduler/`

AWS제공 소프트웨어는 다음과 같이 시스템 구성을 변경합니다.

- 빌드에서 생성된 Slurm `systemd` 파일은 파일 이름이 `/etc/systemd/system/` 인에 복사됩니다`slurmd-25.05.service`.
- 존재하지 않으면 UID/GID가 인 Slurm 사용자 및 그룹(`slurm:slurm`)이 생성됩니다`401`.
- 폴더`/etc/aws/pcs/scheduler/slurm-25.05/plugstack.conf.d/`가 생성되어 SPANK 플러그인을 사용하여 AWS PCS에서 Slurm 기능 확장 구성을 저장합니다.
- Amazon Linux 2 및 Rocky Linux 9에서 설치는 EPEL 리포지토리를 추가하여 Slurm 또는 해당 종속성을 빌드하는 데 필요한 소프트웨어를 설치합니다.
- RHEL9에서 설치는 `codeready-builder-for-rhel-9-rhui-rpms` 및 `epel-release-latest-9`가 Slurm 또는 해당 종속성을 빌드하는 데 필요한 소프트웨어를 설치할 수 `fedoraproject` 있도록 합니다.

지원되는 운영 체제

AWS PCS에서 지원되는 운영 체제을(를) 참조하세요.

> **Note**
>
> AWS Deep Learning AMIs Amazon Linux 2 및 Ubuntu 22.04 기반 (DLAMI) 버전은 AWS PCS 소프트웨어 및 Slurm 설치 프로그램과 호환되어야 합니다. 자세한 내용은 AWS Deep Learning AMIs 개발자 안내서의 DLAMI 선택을 참조하세요.

지원되는 인스턴스 유형

AWS PCS 소프트웨어 및 Slurm 설치 관리자는 지원되는 운영 체제 중 하나를 실행할 수 있는 x86_64 또는 arm64 인스턴스 유형을 지원합니다.

지원되는 Slurm 버전

AWS PCS의 Slurm 버전을(를) 참조하세요.

체크섬을 사용하여 설치 관리자 확인

SHA256 체크섬을 사용하여 설치 관리자 tarball(.tar.gz) 파일을 확인할 수 있습니다. 이 작업을 수행하여 소프트웨어 게시자의 자격 증명을 확인하고 애플리케이션이 게시된 이후 변경되거나 손상되지 않았는지 확인하는 것이 좋습니다.

tarball을 확인하려면

SHA256 체크섬에 SHA256sum 유틸리티를 사용하고 tarball 파일 이름을 지정합니다. tarball 파일을 저장한 디렉터리에서 명령을 실행해야 합니다.

- SHA256

```
$ sha256sum tarball_filename.tar.gz
```

명령은 체크섬 값을 다음 형식으로 반환해야 합니다.

```
checksum_value tarball_filename.tar.gz
```

명령에서 반환된 체크섬 값을 다음 표에 제공된 체크섬 값과 비교합니다. 체크섬이 일치하면 설치 스크립트를 실행하는 것이 안전합니다.

> **Important**
>
> 체크섬이 일치하지 않으면 설치 스크립트를 실행하지 마십시오. 지원에 문의하세요.

예를 들어 다음 명령은 Slurm 25.05.5-1 tarball에 대한 SHA256 체크섬을 생성합니다.

```
$ sha256sum aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz
```

출력 예시:

```
3b0f93bce441d4f4f6935175f2c1e81cd961cb923adb416fa6689f5592047a7d aws-pcs-slurm-25.05-
installer-25.05.5-1.tar.gz
```

다음 표에는 최신 버전의 설치 관리자에 대한 체크섬이 나열되어 있습니다. *us-east-1*을 AWS PCS를 사용하는 AWS 리전 로 바꿉니다.

AWS PCS 에이전트

설치 관리자	다운로드 URL	SHA256 체크섬
AWS PCS 에이전트 1.3.2-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.2-1.tar.gz`	`06b32a952a1c849e3442e35c28ac2e4d6962b09286cad748f3c83d561b52ec6f`
AWS PCS 에이전트 1.3.1-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.1-1.tar.gz`	`5b7f1eb7b3a86bd2d331b5cb0138d868dc9452da34b480becd86af892c7e8d19`
AWS PCS 에이전트 1.3.0-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.3.0-1.tar.gz`	`eadc9b65c3db248bdde2a6c41814dfb1b97239f24ad55e03d8526dd9ab4a8d16`
AWS PCS 에이전트 1.2.2-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.2.2-1.tar.gz`	`fd7b6ea5442db75d723fc4971781ce6ae511baa21b87c4286fc1df8127b282b8`
AWS PCS 에이전트 1.2.1-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-agent/aws-pcs`	`2b784643ca01ccca1baa64fbfb34bb41efe8bdca69470998b74ce3962bc271d4`

설치 관리자	다운로드 URL	SHA256 체크섬
	-agent-v1.2.1-1.ta r.gz	
AWS PCS 에이전트 1.2.0-1	https://aws-pcs-repo- *us-east-1* .s3.*us-east-1* .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.2.0-1.tar.gz	470db8c4fc9e50277b6317f98584b6b547e73523043e34f018eecae767846805
AWS PCS 에이전트 1.1.1-1	https://aws-pcs-repo- *us-east-1* .s3.*us-east-1* .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.1.1-1.tar.gz	bef078bf60a6d8ecde2e6c49cd34d088703f02550279e3bf483d57a235334dc6
AWS PCS 에이전트 1.1.0-1	https://aws-pcs-repo- *us-east-1* .s3.*us-east-1* .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.1.0-1.tar.gz	594c32194c71bccc5d66e5213213ae38dd2c6d2f9a950bb01accea0bbab0873a
AWS PCS 에이전트 1.0.1-1	https://aws-pcs-repo- *us-east-1* .s3.*us-east-1* .amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.0.1-1.tar.gz	04e22264019837e3f42d8346daf5886eaacecd21571742eb505ea8911786bcb2

설치 관리자	다운로드 URL	SHA256 체크섬
AWS PCS 에이전트 1.0.0-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-agent/aws-pcs-agent-v1.0.0-1.tar.gz	d2d3d68d00c685435c38af471d7e2492dde5ce9eb222d7b6ef0042144b134ce0

Slurm 설치 관리자

설치 관리자	다운로드 URL	SHA256 체크섬
Slurm 25.05.5-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.5-1.tar.gz	e7bc84db4e71b8c7174e2f581a31233f839affb5306c76a8adba23204dcc703b
Slurm 25.05.4-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.4-1.tar.gz	3b0f93bce441d4f4f6935175f2c1e81cd961cb923adb416fa6689f5592047a7d
Slurm 25.05.3-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-25.05-installer-25.05.3-1.tar.gz	851bb5815b6700ceb30cc4a3fda204ca8ce362c14528c339908983255a936cf0
Slurm 24.11.7-1	https://aws-pcs-repo-*us-east-1*.s3.*us-east-1*.amazonaws.com/aws-pcs-slurm/aws-pcs	73d75be82c6f88f6e248fd0cc779a5630c62

설치 관리자	다운로드 URL	SHA256 체크섬
	-slurm-24.11-insta ller-24.11.7-1.tar.gz	d91ebabdd9cf0f61b1 943b6d7d09
Slurm 24.11.6-2	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.11-insta ller-24.11.6-2.tar.gz	f17cd78e0bc6b9c818 b794d9d2685cceabdc 73f4fbb12f7566ae5b 86a5abc32b
Slurm 24.11.6-1	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.11-insta ller-24.11.6-1.tar.gz	225de9fc18206f5f65 f412effe1fd457614a c97ee9822b3ff804a4 52b0fae522
Slurm 24.11.5-1	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.11-insta ller-24.11.5-1.tar.gz	593efe4d66bef2f3e4 6d5a382fb5a32f7a3c a2510bcf1b3c85739f 4f951810d5
Slurm 24.05.8-2	https://aws-pcs-re po- *us-east-1* .s3.*us-* *east-1* .amazonaws.com/ aws-pcs-slurm/aws-pcs -slurm-24.05-insta ller-24.05.8-2.tar.gz	c494b0b55c319a4c2f 3faf668c759d46c32c 4c7aa94ae97d941283 28fe95364b

설치 관리자	다운로드 URL	SHA256 체크섬
Slurm 24.05.8-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-24.05-installer-24.05.8-1.tar.gz`	`210a43b376af082bbad640b2032655885790c5dab0e6489cc327c7310a375849`
Slurm 24.05.7-1	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-24.05-installer-24.05.7-1.tar.gz`	`0b5ed7c81195de2628c78f37c79e63fc4ae99132ca6b019b53a0d68792ee82c5`
Slurm 24.05.5-2	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-24.05-installer-24.05.5-2.tar.gz`	`7cc8d8294f2fbff95fe0602cf9e21e02003b5d96c0730e0a18c6aa04c7a4967b`
Slurm 23.11.10-4(사용되지 않음)	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-4.tar.gz`	`bb2d8c919c69dba38d14358f49c7f0427564c5dd4af85a1c9eca2c57ceeae29a`
Slurm 23.11.10-3(사용되지 않음)	`https://aws-pcs-repo-us-east-1.s3.us-east-1.amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-3.tar.gz`	`488a10ee0fbd57ec0e0ff7ea708a9e3038fafdc025c6bb391c75c2e2a7852a00`

설치 관리자	다운로드 URL	SHA256 체크섬
Slurm 23.11.10-2(사용되지 않음)	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-2.tar.gz`	`0bbe85423305c05987931168caf98da08a34c25f9eec0690e8e74de0b7bc8752`
Slurm 23.11.10-1(사용되지 않음)	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.10-1.tar.gz`	`27e8faa9980e92cdfd8cfdc71f937777f0934552ce61e33dac4ecf5a20321e44`
Slurm 23.11.9-1(사용되지 않음)	`https://aws-pcs-repo- us-east-1 .s3.us-east-1 .amazonaws.com/aws-pcs-slurm/aws-pcs-slurm-23.11-installer-23.11.9-1.tar.gz`	`1de7d919c8632fe8e2806611bed4fde1005a4fadc795412456e935c7bba2a9b8`

AWS PCS 샘플 AMIs 릴리스 정보

스케줄러의 지원되는 최신 메이저 버전에 대한 AMIs는 보안 업데이트 및 중요한 버그 수정을 수신합니다. 이러한 증분 보안 패치는 공식 릴리스 정보에 포함되지 않습니다.

> **Important**
>
> 이전 스케줄러 버전과 관련된 샘플 AMIs는 지원되지 않으며 업데이트를 수신하지 않습니다.

> **Important**
>
> 샘플 AMIs는 데모용이며 프로덕션 워크로드에는 권장되지 않습니다.

목차

AWS x86_64용 PCS 샘플 AMIs(Amazon Linux 2)

Slurm 25.05

AMI 이름

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05`

지원되는 EC2 인스턴스

- 64비트 x86 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음 Architectures=x86_64를 검색합니다.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: x86_64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.43.1
- GDRCopy: 2.5.1
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

Slurm 24.11

> **Note**
>
> AWS PCS는 Slurm 24.11 이상에 대한 회계를 지원합니다. 자세한 내용은 AWS PCS의 Slurm 회계 단원을 참조하십시오.

AMI 이름

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-24.11`

지원되는 EC2 인스턴스

- 64비트 x86 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음를 검색합니다`Architectures=x86_64`.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: x86_64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.33.0
- GDRCopy : 2.4
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

Slurm 24.05

AMI 이름

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-24.05`

지원되는 EC2 인스턴스

- 64비트 x86 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음를 검색합니다`Architectures=x86_64`.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: x86_64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.33.0
- GDRCopy : 2.4
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

Slurm 23.11

AMI 이름

- `aws-pcs-sample_ami-amzn2-x86_64-slurm-23.11`

지원되는 EC2 인스턴스

- 64비트 x86 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음를 검색합니다`Architectures=x86_64`.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: x86_64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.33.0

- GDRCopy : 2.4
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

AWS Arm64용 PCS 샘플 AMIs(Amazon Linux 2)

Slurm 25.05

AMI 이름

- `aws-pcs-sample_ami-amzn2-arm64-slurm-25.05`

지원되는 EC2 인스턴스

- 64비트 Arm 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음 Architectures=arm64를 검색합니다.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: arm64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.43.1
- GDRCopy: 2.5.1
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

Slurm 24.11

> **Note**
>
> AWS PCS는 Slurm 24.11 이상에 대한 회계를 지원합니다. 자세한 내용은 AWS PCS의 Slurm 회계 단원을 참조하십시오.

AMI 이름

- `aws-pcs-sample_ami-amzn2-arm64-slurm-24.11`

지원되는 EC2 인스턴스

- 64비트 Arm 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음를 검색합니다`Architectures=arm64`.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: arm64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.33.0
- GDRCopy : 2.4
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

Slurm 24.05

AMI 이름

- `aws-pcs-sample_ami-amzn2-arm64-slurm-24.05`

지원되는 EC2 인스턴스

- 64비트 Arm 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음를 검색합니다`Architectures=arm64`.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2

- 컴퓨팅 아키텍처: arm64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.33.0
- GDRCopy : 2.4
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

Slurm 23.11

AMI 이름

- `aws-pcs-sample_ami-amzn2-arm64-slurm-23.11`

지원되는 EC2 인스턴스

- 64비트 Arm 프로세서가 있는 모든 인스턴스. 호환되는 인스턴스를 찾으려면 Amazon EC2 콘솔로 이동합니다. 인스턴스 유형을 선택한 다음를 검색합니다`Architectures=arm64`.

AMI 콘텐츠

- 지원되는 AWS 서비스: AWS PCS
- 운영 체제: Amazon Linux 2
- 컴퓨팅 아키텍처: arm64
- EBS 볼륨 유형: gp2
- EFA 설치 관리자: 1.33.0
- GDRCopy : 2.4
- NVIDIA 드라이버: 550.127.08
- NVIDIA CUDA: 12.4.1_550.54.15

AWS PCS에서 지원되는 운영 체제

AWS PCS는 컴퓨팅 노드 그룹에 대해 구성된 Amazon Machine Image(AMI)를 사용하여 해당 컴퓨팅 노드 그룹에서 EC2 인스턴스를 시작합니다. AMI는 EC2 인스턴스가 사용하는 운영 체제를 결정합니다. AWS PCS 샘플 AMIs에서는 운영 체제를 변경할 수 없습니다. 다른 운영 체제를 사용하려면 사용자 지정 AMI를 생성해야 합니다. 자세한 내용은 AWS PCS용 Amazon Machine Image(AMIs) 단원을 참조하십시오.

지원되는 운영 체제

- Amazon Linux 2

  이는 AWS PCS 샘플 AMIs.

  > **Important**
  >
  > 샘플 AMIs 데모용이며 프로덕션 워크로드에는 권장되지 않습니다. Amazon Linux 2를 사용하려는 경우에도 프로덕션 워크로드에 사용자 지정 AMI를 생성하고 사용해야 합니다.

- Amazon Linux 2023
- RedHat Enterprise Linux 9(RHEL 9)

  RHEL 인스턴스 유형의 온디맨드 비용은 지원되는 다른 운영 체제보다 높습니다. 요금에 대한 자세한 내용은 온디맨드 요금 및 Amazon Elastic Compute Cloud용 Red Hat Enterprise Linux는 어떻게 제공되고 가격이 책정되나요?를 참조하세요.

- RedHat Enterprise Linux 8(RHEL 8)
- Rocky Linux 9

  공식 Rocky Linux 9 AMIs 사용자 지정 AMI의 기본으로 사용할 수 있습니다. 기본 AMI에 최신 커널이 없는 경우 사용자 지정 AMI 빌드가 실패할 수 있습니다.

  커널을 업그레이드하려면

  1. 여기에서 rocky9 AMI ID를 사용하여 인스턴스 시작: https://rockylinux.org/cloud-images/
  2. 인스턴스에 ssh하여 다음 명령을 실행:

     ```
     sudo yum -y update
     ```

3. 인스턴스에서 이미지를 생성합니다. 이 이미지를 사용자 지정 AMI`ParentImage`의 로 지정합니다.

- Rocky Linux 8
- Ubuntu 22.04

  Ubuntu 22.04는 SSH에 더 안전한 키가 필요하며 기본적으로 RSA 키를 지원하지 않습니다. 대신 ED25519 키를 생성하고 사용하는 것이 좋습니다.
- Ubuntu 24.04

AWS PCS 에이전트 버전

AWS PCS 에이전트 소프트웨어는 Slurm과 함께 사용하도록 EC2 인스턴스 AWS PCS 시작을 구성합니다. 클러스터에 대한 컴퓨팅 노드 그룹을 생성할 때 지정하는 Amazon 머신 이미지(AMI)에 에이전트를 포함합니다. 이러한 컴퓨팅 노드 그룹에서 시작된 EC2 인스턴스는 지정된 AMI와 포함된 AWS PCS 에이전트 소프트웨어를 사용합니다. AWS PCS 에이전트를 사용하면 EC2 인스턴스가 클러스터의 일부로 자신을 등록할 수 있습니다. 최신 AWS PCS 에이전트 소프트웨어를 사용하려면 사용자 지정 AMIs 업데이트해야 합니다. 자세한 설명은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs)에서 2단계 - AWS PCS 에이전트 설치 섹션을 참조하세요.

AWS PCS 에이전트 버전	릴리스 날짜	릴리스 노트
v1.3.2-1	2026년 3월 10일	• 해당 운영 체제에서 결함이 있는 `curl` SigV4 백포트로 인해 RHEL 8.10 또는 Rocky Linux 8.10을 실행하는 컴퓨팅 노드가 부트스트랩되지 않는 문제를 수정했습니다.
v1.3.1-1	2025년 11월 7일	• 사용 가능한 경우 `smt/control` sysfs 파라미터를 사용하여 하이퍼스레딩 비활성화를 개선했습니다. • PCS 에이전트가 하이퍼스레딩을 비활성화하려고 시도하는 동안 부팅 중에 CPU가 잠길 때 발생할 수 있는 레이스 조건을 수정했습니다. • Slurm 컴퓨팅 노드의 `InstanceId` 및 `InstanceType` 필드가 각각 타임스탬프와 하이픈으로 채워지는 문제를 수정했습니다.

AWS PCS 에이전트 버전	릴리스 날짜	릴리스 노트
v1.3.0-1	2025년 11월 3일	• 새로운 운영 체제에 대한 지원 추가: Amazon Linux 2023, Ubuntu 24, RHEL 8, Rocky 8.
v1.2.2-1	2025년 10월 16일	• IPv4 엔드포인트를 사용할 수 없는 경우 IPv6 엔드포인트에 허용된 인스턴스 메타데이터 쿼리입니다. IPv4 • 커널이 형제 스레드를 CPU ID 범위로 반환한 경우 하이퍼스레딩이 비활성화되지 않는 문제를 수정했습니다. • 하이퍼스레딩이 성공적으로 비활성화되었을 때 로그에 잘못된 실패 메시지가 발생하는 문제를 수정했습니다.
v1.2.1-1	2025년 6월 19일	• 이제 컨트롤러를 사용할 수 없는 경우 AWS PCS 에이전트가 최대 30분 동안 슬러밍을 시작하려고 합니다. • RegisterComputeNodeGroupInstance에 대한 응답에 SLURMDBD 엔드포인트가 포함된 경우 잘못된 슬러밍 구성이 발생하는 문제를 수정했습니다.
v1.2.0-1	2025년 3월 7일	• 에서 IPv6에 대한 지원을 활성화했습니다`slurmd.conf` .

AWS PCS 에이전트 버전	릴리스 날짜	릴리스 노트
v1.1.1-1	2024년 12월 13일	• RegisterComputeNodeGroupInstance 호출에서 잘못된 Slurm 버전이 보고된 문제를 수정했습니다. • 의 사용자 지정 스크립트가 실행된 경우 인스턴스 메타데이터를 올바르게 가져오지 못하는 문제를 수정/opt/aws/pcs/etc/bootstrap_hooks/ 했습니다.
v1.1.0-1	2024년 12월 6일	• 부트스트랩 단계 전에에서 사용자 지정 스크립트를 실행할 /opt/aws/pcs/etc/bootstrap_hooks/ 수 있습니다.
v1.0.1-1	2024년 10월 22일	• GPU 지원 인스턴스에서 slurmd 시작할 때 NVIDIA 디바이스가 작동하지 않는 문제를 수정했습니다.
v1.0.0-1	2024년 8월 28일	• 최초 릴리스입니다.

AWS PCS의 Slurm 스케줄러

Slurm은 HPC 워크로드에 대한 작업 예약, 리소스 할당 및 작업 모니터링 기능을 제공하는 Linux 클러스터용으로 설계된 오픈 소스 워크로드 관리자입니다. AWS PCS는 클러스터 워크로드를 관리하기 위해 Slurm 스케줄러를 지원합니다.

주제

AWS PCS의 Slurm 버전

SchedMD는 새로운 기능, 최적화 및 보안 패치로 Slurm을 지속적으로 개선합니다. SchedMD는 정기적으로 새 메이저 버전을 릴리스하고 언제든지 최대 3개의 버전을 지원할 계획입니다. AWS PCS는 Slurm 컨트롤러를 패치 버전으로 자동으로 업데이트하도록 설계되었습니다.

SchedMD가 특정 메이저 버전에 대한 지원을 종료하면 AWS PCS는 해당 버전을 수명 종료(EOL)로 지정합니다. EOL 후에는 기존 클러스터가 지원 보장 없이 최대 12개월 동안 계속 실행될 수 있지만 해당 버전으로 새 클러스터를 생성할 수 없습니다. Slurm 메이저 버전이 EOL에 가까운 경우 고객이 클러스터를 최신 지원 버전으로 업그레이드해야 하는 시기를 알 수 있도록 AWS PCS는 사전 알림을 보냅니다.

지원되는 최신 Slurm 버전을 사용하여 클러스터를 배포하고 최신 개선 및 개선 사항에 액세스하는 것이 좋습니다.

AWS PCS에서 지원되는 Slurm 버전

다음 표에는 지원되는 Slurm 버전과 각 버전에 대한 중요한 날짜 및 정보가 나와 있습니다.

Slurm 버전	SchedMD 릴리스 날짜	AWS PCS 릴리스 날짜	AWS PCS EOL 날짜	호환되는 최소 AWS PCS 에이전트 버전	지원되는 AWS PCS 샘플 AMIs
25.05	5/29/2025	10/16/2025	11/30/2026	1.0.0-1	• aws-pcs-sample_ami-amzn2-x86_64-slurm-25.05 • aws-pcs-sample_ami-amzn2-arm64-slurm-25.05
24.11	11/29/2024	5/14/2025	5/31/2026	1.0.0-1	• aws-pcs-sample_ami-amzn2-x86_64-slurm-24.11 • aws-pcs-sample_ami-amzn2-arm64-slurm-24.11

AWS PCS에서 지원되지 않는 Slurm 버전

다음 표에는 AWS PCS에서 지원되지 않는 Slurm 버전이 나와 있습니다.

Slurm 버전	SchedMD 릴리스 날짜	AWS PCS 릴리스 날짜	AWS PCS EOL 날짜		
24.05	5/30/2024	12/18/2024	11/30/2025		
23.11	11/21/2023	8/28/2024	5/31/2025		

AWS PCS의 Slurm 버전 릴리스 정보

이 주제에서는 현재 AWS PCS에서 지원되는 각 Slurm 버전의 중요한 변경 사항에 대해 설명합니다. 클러스터를 업그레이드할 때 이전 버전과 새 버전 간의 변경 사항을 검토하는 것이 좋습니다.

Slurm 25.05

AWS PCS에 구현된 변경 사항

- 이제 Slurm requeue_on_resume_failure SchedulerParameter가 기본적으로 활성화됩니다.
- "stderr"는 Slurm 25.05에서 비활성화되었으므로 LogTimeFormat의 옵션으로 제거되었습니다.
- AWS PCS는 다중 클러스터 sackd 구성을 지원합니다. 로그인 노드는 여러 클러스터에 액세스할 수 있습니다.

Slurm 25.05에 대한 자세한 내용은 다음 간행물을 참조하세요.

- SchedMD 릴리스 발표: https://www.schedmd.com/slurm-version-25-05-0-is-now-available/
- SchedMD 릴리스 정보: https://github.com/SchedMD/slurm/blob/slurm-25-05-0-1/RELEASE_NOTES.md

Slurm 24.11

AWS PCS에 구현된 변경 사항

- AWS PCS는 Slurm 회계를 지원합니다. 자세한 내용은 AWS PCS의 Slurm 회계 단원을 참조하십시오.

Slurm 24.11에 대한 자세한 내용은 다음 간행물을 참조하세요.

- SchedMD 릴리스 발표
- SchedMD 릴리스 정보

Slurm 24.05

AWS PCS에 구현된 변경 사항

- 이제 새 Slurm Step Manager 모듈이 AWS PCS에서 기본적으로 활성화됩니다. 이 모듈은 단계 관리를 중앙 컨트롤러에서 컴퓨팅 노드로 오프로드하여 단계 사용량이 많은 환경에서 시스템 동시성을 크게 개선함으로써 상당한 이점을 제공합니다. 이 구성을 지원하고 더 나은 격리 `Prolog` 및 `Epilog` 프로세스 실행을 위해 새 prolog 플래그(`Contain`, `Alloc`)가 활성화됩니다.
- 컨트롤러에서 컴퓨팅 노드로의 계층적 통신은 Slurm 노드 내 통신을 최적화하여 확장성과 성능을 개선합니다. 또한 라우팅 구성은 이제 플러그인의 기본 라우팅 알고리즘 대신 컨트롤러와의 통신에 파티션 노드 목록을 사용하여 시스템 복원력을 개선합니다.
- 새 해시 플러그인은 이전를 `HashPlugin=hash/sha3` 대체합니다`hash/k12 plugin`. 이제 AWS PCS 클러스터에서이 기능이 기본적으로 활성화됩니다.
- 이제 Slurm 컨트롤러 로그에에 대한 모든 인바운드 원격 프로시저 호출(RPC)에 대한 향상된 감사 기능이 포함됩니다`slurmctld`. 로그에는 연결 처리 전 소스 주소, 인증된 사용자 및 RPC 유형이 포함됩니다.

Slurm 24.05에 대한 자세한 내용은 다음 간행물을 참조하세요.

- SchedMD 릴리스 발표
- SchedMD 릴리스 정보

Slurm 23.11

AWS PCS에서 변경할 수 있는 Slurm 설정

- 의 `SuspendTime` 기본값은 입니다`60`. AWS PCS `scaleDownIdleTimeInSeconds` 구성 파라미터를 사용하여 설정합니다. 자세한 내용은 AWS PCS API 참조에서 `ClusterSlurmConfiguration` 데이터 형식의 `scaleDownIdleTimeInSeconds` 파라미터를 참조하세요.

- `MaxJobCount` 및 `MaxArraySize`는 클러스터에 대해 선택한 크기를 기반으로 합니다. 자세한 내용은 AWS PCS `CreateCluster` API 참조에서 API 작업의 `size` 파라미터를 참조하세요.
- `SelectTypeParameters` Slurm 설정은 기본적으로 입니다`CR_CPU`. 클러스터를 생성할 때 `slurmCustomSettings`에서 설정할 값으로 제공할 수 있습니다. 자세한 내용은 API 작업의 `slurmCustomSettings` 파라미터 및 PCS `CreateCluster` API 참조의 SlurmCustomSetting을 참조하세요. AWS
- 클러스터 `Epilog` 수준에서 `Prolog` 및를 설정할 수 있습니다. 클러스터를 생성할 때 `slurmCustomSettings`에서 설정할 값으로 제공할 수 있습니다. 자세한 내용은 `CreateCluster` 및 PCS API 참조의 SlurmCustomSetting을 참조하세요. AWS
- 컴퓨팅 노드 그룹 수준에서 `Weight` 및 `RealMemory`를 설정할 수 있습니다. 컴퓨팅 노드 그룹을 생성할 때에서 `slurmCustomSettings` 설정할 값으로 제공할 수 있습니다. 자세한 내용은 `CreateComputeNodeGroup` 및 PCS API 참조의 SlurmCustomSetting을 참조하세요. AWS

AWS PCS의 Slurm 버전에 대해 자주 묻는 질문

AWS PCS는 여러 Slurm 버전에 대한 지원을 유지합니다. 새 Slurm 버전이 도입되면 AWS PCS는 해당 버전이 SchedMD에서 지원 종료(EOS)에 도달할 때까지 기술 지원 및 보안 패치를 제공합니다. AWS PCS는 AWS 용어와 일치하도록 Slurm 버전의 EOS 날짜를 수명 종료(EOL)로 지칭합니다.

AWS PCS는 Slurm 버전을 얼마나 오래 지원하나요?

AWS Slurm 버전에 대한 PCS 지원은 SchedMD의 메이저 버전 지원 주기와 일치합니다. AWS PCS는 현재 버전과 2개의 최신 이전 메이저 버전을 지원합니다. SchedMD가 새 메이저 버전을 릴리스하면 AWS PCS는 지원되는 가장 오래된 버전에 대한 지원을 종료합니다. AWS PCS는 가능한 한 빨리 Slurm의 새 메이저 버전을 릴리스하지만 SchedMD의 릴리스와 AWS PCS에서의 가용성 사이에 지연이 있을 수 있습니다.

클러스터가 새 Slurm 패치 버전 릴리스를 받으려면 어떻게 해야 하나요?

버그 및 보안 수정을 해결하기 위해 AWS PCS는 내부 서비스 소유 계정에서 실행되는 클러스터 컨트롤러에 패치를 자동으로 적용하도록 설계되었습니다. 의 EC2 인스턴스에 패치를 설치하려면 컴퓨팅 노드 그룹의 AWS 계정 Amazon Machine Image(AMI)를 업데이트하고 업데이트된 AMI를 사용하도록 컴퓨팅 노드 그룹을 업데이트합니다. 자세한 내용은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs) 단원을 참조하십시오.

> **Note**
>
> Slurm 컨트롤러는 업데이트하는 동안 사용할 수 없습니다. 실행 중인 작업은 영향을 받지 않습니다. 클러스터의 컨트롤러를 사용할 수 없게 되기 전에 제출된 작업은 컨트롤러를 사용할 수 있을 때까지 보류됩니다.

예정된 Slurm 버전 EOL 이벤트에 대한 정보는 어떻게 받나요?

EOL 날짜 6개월 전에 이메일 메시지가 전송됩니다. 매월 EOL 날짜 1주일 전에 최종 이메일 메시지와 함께 EOL 날짜 1주일 전에 이메일 메시지가 전송됩니다. EOL 날짜 이후에는 EOL Slurm 버전으로 AWS PCS 클러스터를 실행하는 고객에게 12개월 동안 월별 이메일 메시지를 보냅니다. 해당 버전에 대한 보안 취약성이 식별되는 경우 EOL Slurm 버전이 있는 클러스터를 일시 중지할 수 있습니다.

클러스터에서 사용하는 Slurm 버전이 EOL Slurm 버전을 실행 중인지 확인하려면 어떻게 해야 합니까?

EOL Slurm 버전의 클러스터가 실행 중임을 알리는 이메일 메시지가 전송됩니다. EOL Slurm 버전이 있는 클러스터의 세부 정보가 포함된 알림에 AWS Health Dashboard 알림을 게시합니다. AWS PCS 콘솔을 사용하여 EOL Slurm 버전이 있는 클러스터를 식별할 수도 있습니다.

Slurm 버전이 EOL에 가깝거나 그 이상인 경우 어떻게 해야 하나요?

지원되는 최신 버전의 Slurm으로 새 클러스터를 생성하고 컴퓨팅 노드 그룹 AMIs. AMIs 및 실행 중인 EC2 인스턴스의 Slurm 버전은 클러스터의 Slurm 버전보다 2개를 초과할 수 없습니다. 자세한 내용은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs) 단원을 참조하십시오.

EOL 날짜까지 최신 버전의 Slurm으로 전환하지 않으면 어떻게 되나요?

EOL Slurm 버전으로 새 클러스터를 생성할 수 없습니다. 기존 클러스터는 AWS 지원 없이 최대 12개월 동안 작동할 수 있으며 작업을 유지 관리하는 데 즉각적인 조치가 필요하지 않습니다. EOL 날짜 이후에는 지원, 보안 업데이트 및 가용성이 보장되지 않습니다. 보안상의 이유로 클러스터를 일시 중지할 수 있습니다. 지원되는 Slurm 버전을 사용하여 AWS PCS 클러스터에 대한 보안 및 지원을 유지하는 것이 좋습니다.

EOL Slurm 버전으로 클러스터를 운영할 경우 어떤 위험이 있나요?

EOL Slurm 버전이 있는 클러스터는 상당한 보안 및 운영 위험을 초래합니다. SchedMD의 활성 모니터링이 없으면 보안 취약성이 감지되지 않거나 해결되지 않을 수 있습니다. 중요한 취약성이 발견되면 클러스터를 즉시 일시 중지할 수 있습니다.

클러스터가 일시 중지되면 작업, 클러스터 컴퓨팅, 스토리지 및 네트워킹 리소스는 어떻게 되나요?

AWS PCS에서 관리하는 모든 리소스가 종료됩니다. 여기에는 Slurm 컨트롤러, 컴퓨팅 노드 그룹 및 EC2 인스턴스가 포함됩니다. 컴퓨팅 인스턴스에서 실행되는 모든 작업은 즉시 종료되고 클러스터는 일시 중지 상태로 전환됩니다. 외부 파일 시스템과 같은 고객 관리형 리소스는 그대로 유지됩니다. AWS PCS 콘솔 및 API 작업을 사용하여 클러스터의 구성에 액세스할 수 있습니다.

일시 중지된 클러스터를 다시 시작하여 나머지 작업을 재개할 수 있습니까?

아니요, 일시 중지된 클러스터는 다시 시작할 수 없습니다. 일시 중지된 클러스터의 구성을 사용하여 지원되는 Slurm 버전으로 새 클러스터를 생성할 수 있습니다. 나머지 작업은 외부 파일 시스템에 저장한 경우 실행할 수 있습니다.

12개월 유예 기간 이후에 연장을 요청할 수 있나요?

아니요. 12개월 유예 기간 이후에는 클러스터를 실행하도록 확장을 요청할 수 없습니다. 지원되는 Slurm 버전으로 전환하는 데 도움이 되도록 연장된 시간을 제공합니다. 클러스터 작업이 중단되지 않도록 Slurm 버전이 EOL에 도달하기 전에 전환하는 것이 좋습니다.

AWS PCS의 Slurm 회계

새 AWS PCS 클러스터에 대한 회계를 활성화하여 클러스터 사용량을 모니터링하고, 리소스 제한을 적용하고, 특정 대기열 또는 컴퓨팅 노드 그룹에 대한 세분화된 액세스 제어를 관리할 수 있습니다. AWS PCS는 클러스터에 대한 회계 데이터베이스를 생성하고 관리하므로 별도의 회계 데이터베이스를 생성하고 관리할 필요가 없습니다. AWS PCS는 Slurm의 회계 기능을 사용합니다. Slurm의 회계 기능에 대한 자세한 내용은 SchedMD의 Slurm 설명서를 참조하세요.

회계를 사용하려면 새 클러스터를 생성할 때 활성화하고 선택적으로 회계 파라미터를 설정합니다. 클러스터 상태가 `Active` 이고 컴퓨팅 노드 그룹이 있는 경우 로그인 노드의 Linux 셸에 연결하여 Slurm `sacct` 명령을 사용하여 작업 데이터를 보는 등의 회계 기능을 수행할 수 있습니다.

> **Note**
>
> 회계는 Slurm 24.11 이상에서 지원됩니다.

AWS PCS console

클러스터 생성 페이지에서 유효한 버전의 Slurm(버전 24.11 이상)을 선택해야 합니다. 스케줄러 설정에서 회계를 활성화합니다.

AWS PCS API

`CreateCluster` API 작업에 대한 호출에 `accounting` 구성을 제공합니다. `accounting` 객체에서를 `mode`로 설정합니다`STANDARD`. 자세한 내용은 PCS API 참조의 CreateCluster and Accounting을 참조하세요. AWS

다음 예제에서는 AWS CLI 를 사용하여 `CreateCluster` API 작업을 호출합니다. 파라미터 값 하위 문자열은 회계를 `accounting='{mode=STANDARD}'` 활성화합니다.

```
aws pcs create-cluster --cluster-name cluster-name \
                       --scheduler type=SLURM,version=24.11 \
                       --size SMALL \
                       --networking subnetIds=cluster-subnet-
id,securityGroupIds=cluster-security-group-id \
                       --slurm-configuration
 scaleDownIdleTimeInSeconds=180,accounting='{mode=STANDARD}',slurmCustomSettings='[{paramete
```

Important

회계를 활성화하면 추가 청구 요금이 발생합니다. 자세한 내용은 AWS PCS 요금 페이지를 참조하세요.

회계 설정 수정

인프라를 다시 빌드하지 않고도 기존 클러스터에서 회계를 활성화하거나 비활성화할 수 있습니다. 자세한 내용은 AWS PCS에서 클러스터 업데이트 단원을 참조하십시오.

회계를 비활성화하면 클러스터가 `UPDATING` 상태가 되는 즉시 회계 기능에 대한 청구가 중지됩니다. 회계를 활성화하면 클러스터가 성공적으로 `ACTIVE` 상태로 돌아가면 결제가 시작됩니다.

AWS PCS에서 Slurm 회계의 주요 개념

다음 개념은 AWS PCS에 고유하며 AWS PCS가 Slurm 회계를 구현하는 방법을 제어합니다.

회계 데이터베이스

AWS PCS는 회계 데이터를 AWS 계정 AWS 소유한에서 생성된 데이터베이스에 저장합니다. 에 액세스할 수 없습니다. `slurmdbd.conf`.

기본 제거 시간

이 AWS PCS 설정은 모든 회계 레코드 유형(작업, 이벤트, 예약, 단계, 일시 중지, 트랜잭션, 사용 데이터)의 보존 기간(일)을 지정합니다. 예를 들어 값이 30인 경우 AWS PCS는 30일 동안 회계 레코드를 보관합니다. 클러스터를 생성할 때이 값을 제공합니다. 값을 제공하지 않으면 AWS PCS는 데이터베이스에 회계 레코드를 무기한 보관합니다.

AWS PCS console

클러스터를 생성하는 단계의 일부로 기본 제거 시간을 지정합니다. 클러스터 생성 페이지에서 유효한 버전의 Slurm(버전 24.11 이상)을 선택하고 회계를 활성화해야 합니다. 스케줄러 설정에서 기본 제거 시간(일)의 정수 값을 입력합니다.

AWS PCS API

`CreateCluster` API 작업을 호출할 때 제공하는 `accounting` 정보의 `defaultPurgeTimeInDays` 일부로를 지정합니다. 자세한 내용은 PCS API 참조의 CreateCluster and Accounting을 참조하세요. AWS

> **Note**
>
> AWS PCS API를 사용하여 클러스터를 생성하는 경우의 기본 값`defaultPurgeTimeInDays`은 `-1` 이며 유효한 값이 `0` 아닙니다.

회계 정책 적용

이 설정은 Slurm이 클러스터에 대한 작업 제출 규칙, 리소스 제한 및 회계 정책을 얼마나 엄격하게 적용하는지 결정합니다. 이 설정은 클러스터 `slurm.conf` 파일의 `AccountingStorageEnforce` 파라미터에 해당합니다. 적용 옵션의 조합을 선택할 수 있습니다. 옵션을 선택하지 않으면 클러스터의 작업에 적용되는 회계 제약 조건이 없습니다. AWS PCS는 다음 옵션을 지원합니다.

- 연결 - job-to-account 매핑
- 제한 - 리소스 제약 조건
- QoS - 서비스 품질 요구 사항
- 안전 모드 - 한도 내에서 완료 보장
- nosteps - 단계 회계 비활성화
- nojobs - 작업 회계 비활성화

이러한 옵션에 대한 자세한 내용은 SchedMD의 Slurm 설명서를 참조하세요.

AWS PCS console

클러스터를 생성하는 단계의 일부로 옵션을 설정합니다. 클러스터 생성 페이지에서 유효한 버전의 Slurm(버전 24.11 이상)을 선택하고 회계를 활성화해야 합니다. 스케줄러 설정의 회계 정책 적용 드롭다운 목록에서 원하는 옵션을 선택합니다.

AWS PCS API

Slurm에서 이러한 옵션은 클러스터의 `slurm.conf` 파일에 설정됩니다. AWS PCS 클러스터의 `slurm.conf`에 직접 액세스할 수 없습니다. 대신 클러스터`SlurmCustomSettings`를 생성할 때 `CreateCluster` API 작업에를 제공합니다. 자세한 내용은 PCS API 참조의 CreateCluster를 참조하세요. AWS

기존 AWS PCS 클러스터에 대한 회계 구성 가져오기

Slurm 회계 구성은 클러스터의 Slurm 구성에 포함됩니다.

AWS PCS console

1. 탐색 창에서 클러스터를 선택합니다.
2. 목록에서 클러스터 이름을 선택합니다.
3. 구성 탭의 Slurm 구성에서 회계 구성을 찾습니다.

AWS PCS API

`GetCluster` API 작업을 사용하여 클러스터 구성을 가져옵니다. 회계 구성은에서 찾을 수 있습니다`slurmConfiguration`. 의 설정`mode`과의 값은 아래에 `defaultPurgeTimeInDays` 있습니다`accounting`. 선택한 회계 정책 적용 옵션은 아래에 있습니다`slurmCustomSettings`. 자세한 내용은 PCS API 참조의 GetCluster를 참조하세요. AWS

AWS PCS의 Slurm REST API

AWS PCS는를 통해 Slurm의 기본 REST API에 대한 관리형 지원을 제공하여 프로그래밍 방식의 클러스터 상호 작용을 위한 HTTP 인터페이스를 `slurmrestd`제공합니다. 클러스터에 대한 직접 쉘 액세스 없이 표준 HTTP 요청을 통해 작업을 제출하고 클러스터 상태를 모니터링하며 리소스를 관리할 수 있습니다.

일반 사용 사례

Slurm REST API는 다양한 통합 시나리오를 지원합니다.

- 웹 애플리케이션 통합: 작업을 직접 제출하고 관리하는 사용자 지정 프런트엔드와 웹 애플리케이션을 구축합니다.
- Jupyter Notebook Integration: 연구원이 개발 워크플로를 벗어나지 않고 노트북 환경에서 작업을 제출할 수 있습니다.
- 파트너 솔루션 통합: 타사 HPC 도구 및 워크플로 관리자를 AWS PCS 클러스터에 연결합니다.
- 프로그래밍 방식 클러스터 관리: 작업 제출, 모니터링 및 리소스 관리 워크플로를 자동화합니다.
- 연구 컴퓨팅 워크플로: API 기반 작업 관리가 필요한 학술 및 엔터프라이즈 연구 환경을 지원합니다.

요구 사항 및 제한 사항

Slurm REST API를 사용하기 전에 다음 세부 정보를 검토합니다.

- 클러스터는 Slurm 버전 25.05 이상을 사용해야 합니다.
- API 엔드포인트는 클러스터의 VPC 내의 프라이빗 IP 주소를 통해서만 액세스할 수 있습니다.
- 클러스터 보안 그룹은 포트 6820에서 HTTP 트래픽을 허용해야 합니다.
- 인증에는 특정 사용자 자격 증명 클레임이 있는 JWT 토큰이 필요합니다.

현재 제한 사항은 다음과 같습니다.

- 에서 생성된 토큰`scontrol token`은 지원되지 않습니다.
- `X-SLURM-USER-NAME` 헤더 가장은 사용할 수 없습니다.
- 일부 기능을 사용하려면 Slurm 회계를 활성화해야 합니다.
- Slurm CLI 필터 플러그인 메커니즘과 호환되지 않습니다.
- REST API 엔드포인트에 대한 연결은 TLS로 암호화되지 않습니다.

주제

- AWS PCS에서 Slurm REST API 활성화
- AWS PCS에서 Slurm REST API로 인증
- AWS PCS에서 작업 관리에 Slurm REST API 사용

- AWS PCS의 Slurm REST API FAQ

AWS PCS에서 Slurm REST API 활성화

Slurm REST API를 활성화하여 프로그래밍 방식 작업 관리 및 모니터링을 위해 클러스터의 HTTP 인터페이스에 액세스합니다. 클러스터 생성 중에이 기능을 활성화하거나 요구 사항을 충족하는 기존 클러스터를 업데이트할 수 있습니다.

사전 조건

Slurm REST API를 활성화하기 전에 다음을 확인해야 합니다.

- 클러스터 버전: Slurm 버전 25.05 이상.
- 보안 그룹: 원하는 소스의 포트 6820에서 HTTP 트래픽을 허용하는 규칙입니다.

절차

새 클러스터에서 Slurm REST API를 활성화하려면

AWS Management Console

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 클러스터 생성을 선택합니다.
3. 클러스터 세부 정보에서 Slurm 버전 25.05 이상을 선택합니다.
4. 필요에 따라 다른 클러스터 설정을 구성합니다.
5. 스케줄러 구성 섹션에서 REST API를 활성화됨으로 설정합니다.
6. 원하는 소스에서 포트 6820의 HTTP 트래픽을 허용하도록 클러스터 보안 그룹을 구성합니다.
7. 클러스터 생성 프로세스를 완료합니다.

AWS CLI

1. 클러스터를 생성할 때 Slurm REST 구성을 추가합니다.

```
aws pcs create-cluster --region region \
    --cluster-name my-cluster \
    --scheduler type=SLURM, version=25.05 \
```

```
    --size SMALL \
    --networking subnetIds=subnet-ExampleId1,securityGroupIds=sg-ExampleId1 \
    --slurm-configuration slurmRest='{mode=STANDARD}'
```

2. 원하는 소스에서 포트 6820의 HTTP 트래픽을 허용하도록 클러스터 보안 그룹을 구성합니다.

기존 클러스터에서 Slurm REST API를 활성화하려면

AWS Management Console

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 목록에서 클러스터를 선택합니다.
3. 클러스터 세부 정보에서 클러스터가 Slurm 버전 25.05 이상을 사용하는지 확인합니다.
4. 클러스터 편집을 선택합니다.
5. 스케줄러 구성 섹션에서 REST API를 활성화됨으로 설정합니다.
6. 클러스터 업데이트를 선택하여 변경 사항을 적용합니다.
7. 원하는 소스에서 포트 6820의 HTTP 트래픽을 허용하도록 클러스터 보안 그룹을 구성합니다.

AWS CLI

1. 이 예제와 같이 클러스터를 Slurm REST 구성으로 업데이트합니다.

```
aws pcs update-cluster --cluster-identifier my-cluster \
    --slurm-configuration 'slurmRest={mode=STANDARD}'
```

2. 원하는 소스에서 포트 6820의 HTTP 트래픽을 허용하도록 클러스터 보안 그룹을 구성합니다.

활성화 후 발생하는 일

REST API를 활성화하면 AWS PCS가 자동으로 다음을 수행합니다.

- JWT 서명 키를 생성하여 AWS Secrets Manager에 저장합니다.
- VPC `https://<clusterPrivateIpAddress>:6820` 내의에서 API 엔드포인트를 노출합니다.
- REST API 엔드포인트 세부 정보를 표시하도록 클러스터 구성을 업데이트합니다.

이제 작업 관리 및 클러스터 작업에 REST API를 인증하고 사용할 수 있습니다.

AWS PCS에서 Slurm REST API로 인증

AWS PCS의 Slurm REST API는 JSON 웹 토큰(JWT) 인증을 사용하여 클러스터 리소스에 대한 보안 액세스를 보장합니다. AWS PCS는 AWS Secrets Manager에 저장된 관리형 서명 키를 제공하며, 이를 사용하여 특정 사용자 자격 증명 클레임이 포함된 JWT 토큰을 생성합니다.

사전 조건

Slurm REST API로 인증하기 전에 다음을 갖추어야 합니다.

- 클러스터 구성: Slurm 25.05 이상 및 REST API가 활성화된 AWS PCS 클러스터.
- AWS 권한: JWT 서명 키에 대한 AWS Secrets Manager에 대한 액세스입니다.
- 사용자 정보: 클러스터 계정의 사용자 이름, POSIX 사용자 ID 및 하나 이상의 POSIX 그룹 IDs.
- 네트워크 액세스: 포트 6820을 허용하는 보안 그룹을 사용한 클러스터의 VPC 내 연결.

절차

Slurm REST API 엔드포인트 주소를 검색하려면

AWS Management Console

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 목록에서 클러스터를 선택합니다.
3. 클러스터 구성 세부 정보에서 엔드포인트 섹션을 찾습니다.
4. Slurm REST API(slurmrestd)의 프라이빗 IP 주소 및 포트를 기록해 둡니다.
5. 올바른 형식의 HTTP 요청을이 주소로 전송하여 API를 호출할 수 있습니다.

AWS CLI

1. 를 사용하여 클러스터 상태를 쿼리합니다`aws pcs get-cluster`. 응답의 `endpoints` 필드에서 `SLURMRESTD` 엔드포인트를 찾습니다. 다음 예를 참고하세요

```
"endpoints": [
      {
          "type": "SLURMCTLD",
          "privateIpAddress": "192.0.2.1",
          "port": "6817"
      },
```

```
    {
        "type": "SLURMRESTD",
        "privateIpAddress": "192.0.2.1",
        "port": "6820"
    }
]
```

2. 올바른 형식의 HTTP 요청을 로 전송하여 API 호출을 수행할 수 있습니다. `http://<privateIpAddress>:<port>/`

JWT 서명 키를 검색하려면

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 목록에서 클러스터를 선택합니다.
3. 클러스터 구성 세부 정보에서 스케줄러 인증 섹션을 찾습니다.
4. JSON 웹 토큰(JWT) 키 ARN 및 버전을 기록해 둡니다.
5. AWS CLI 를 사용하여 Secrets Manager에서 서명 키를 검색합니다.

```
aws secretsmanager get-secret-value --secret-
id arn:aws:secretsmanager:region:account:secret:name --version-id version
```

JWT 토큰을 생성하려면

1. 다음과 같은 필수 클레임을 사용하여 JWT를 생성합니다.
   - `exp` - JWT의 1970년 이후 초 단위 만료 시간
   - `iat` - 1970년 이후 현재 초 단위 시간
   - `sun` - 인증을 위한 사용자 이름
   - `uid` - POSIX 사용자 ID
   - `gid` - POSIX 그룹 ID
   - `id` - 추가 POSIX 자격 증명 속성
     - `gecos` - 사람이 읽을 수 있는 이름을 저장하는 데 자주 사용되는 사용자 설명 필드
     - `dir` - 사용자의 홈 디렉터리
     - `shell` - 사용자의 기본 셸
     - `gids` - 사용자가 속한 추가 POSIX 그룹 IDs 목록

2. Secrets Manager에서 검색한 서명 키를 사용하여 JWT에 서명합니다.
3. 토큰의 적절한 만료 시간을 설정합니다.

> **Note**
>
> sun 클레임의 대안으로 다음 중 하나를 제공할 수 있습니다.
>
> - `username`
> - `userclaimfield`의를 통해 정의하는 사용자 지정 필드 이름 `AuthAltParameters Slurm custom settings`
> - `id` 클레임 내의 `name` 필드

API 요청을 인증하려면

1. 다음 방법 중 하나를 사용하여 HTTP 요청에 JWT 토큰을 포함합니다.
   - 베어러 토큰 - `Authorization: Bearer <jwt>` 헤더 추가
   - Slurm 헤더 - `X-SLURM-USER-TOKEN: <jwt>` 헤더 추가
2. REST API 엔드포인트에 HTTP 요청을 수행합니다.

   다음은 curl과 `Authorized: Bearer` 헤더를 사용하여 `/ping` API에 액세스하는 예제입니다.

```
curl -X GET -H "Authorization: Bearer <jwt>" \
      http://<privateIpAddress>:6820/slurm/v0.0.43/ping
```

JWT 생성 예

AWS PCS 클러스터 JWT 서명 키를 가져와 로컬 파일로 저장합니다. aws-region, secret-arn 및 secret 버전의 값을 클러스터에 적합한 값으로 바꿉니다.

```
#!/bin/bash
SECRET_KEY=$(aws secretsmanager get-secret-value \
  --region aws-region \
  --secret-id secret-arn \
  --version-stage secret-version \
  --query 'SecretString' \
```

```
  --output text)
echo "$SECRET_KEY" | base64 --decode > jwt.key
```

이 Python 예제에서는 서명 키를 사용하여 JWT 토큰을 생성하는 방법을 보여줍니다.

```
#!/usr/bin/env python3

import sys
import os
import pprint
import json
import time
from datetime import datetime, timedelta, timezone
from jwt import JWT
from jwt.jwa import HS256
from jwt.jwk import jwk_from_dict
from jwt.utils import b64decode,b64encode
if len(sys.argv) != 3:
    sys.exit("Usage: gen_jwt.py [jwt_key_file] [expiration_time_seconds]")
SIGNING_KEY = sys.argv[1]
EXPIRATION_TIME = int(sys.argv[2])
with open(SIGNING_KEY, "rb") as f:
    priv_key = f.read()
signing_key = jwk_from_dict({
    'kty': 'oct',
    'k': b64encode(priv_key)
})
message = {
    "exp": int(time.time() + EXPIRATION_TIME),
    "iat": int(time.time()),
    "sun": "ec2-user",
    "uid": 1000,
    "gid": 1000,
    "id": {
        "gecos": "EC2 User",
        "dir": "/home/ec2-user",
        "gids": [1000],
        "shell": "/bin/bash"
    }
}
a = JWT()
compact_jws = a.encode(message, signing_key, alg='HS256')
print(compact_jws)
```

스크립트는 화면에 JWT를 인쇄합니다.

```
abcdefgtjwttoken...
```

AWS PCS에서 작업 관리에 Slurm REST API 사용

Slurm REST API 개요

Slurm REST API는 HTTP 요청을 통해 클러스터 관리 함수에 프로그래밍 방식으로 액세스할 수 있습니다. 이러한 주요 특성을 이해하면 API를 AWS PCS와 함께 효과적으로 사용하는 데 도움이 됩니다.

- 액세스 프로토콜: API는 클러스터의 프라이빗 네트워크 내에서 통신하기 위해 HTTP(HTTPS 아님)를 사용합니다.
- 연결 세부 정보: 클러스터의 프라이빗 IP 주소와 `slurmrestd` 포트(일반적으로 6820)를 사용하여 API에 액세스합니다. 전체 기본 URL 형식은 입니다`http://<privateIpAddress>:6820`.
- API 버전 관리: API 버전은 Slurm 설치에 해당합니다. Slurm 25.05의 경우 버전 v0.0.43을 사용합니다. 버전 번호는 각 Slurm 릴리스에 따라 변경됩니다. Slurm 릴리스 정보에서 현재 지원되는 API 버전을 찾을 수 있습니다.
- URL 구조: Slurm REST API의 URL 구조는 입니다`http://<privateIpAddress>:<port>/<api-version>/<endpoint>`. REST API 엔드포인트에 대한 자세한 사용 정보는 Slurm 설명서에서 확인할 수 있습니다.

Slurm REST API 작업에 대한 자세한 내용은 REST 클라이언트 Slurm 설명서를 참조하세요.

사전 조건

Slurm REST API를 사용하기 전에 다음을 확인해야 합니다.

- 클러스터 구성: Slurm 25.05 이상 및 REST API가 활성화된 AWS PCS 클러스터.
- 인증: 적절한 사용자 자격 증명 클레임이 있는 유효한 JWT 토큰입니다.
- 네트워크 액세스: 포트 6820을 허용하는 보안 그룹을 사용하여 클러스터의 VPC 내에서 연결합니다.

절차

REST API를 사용하여 작업을 제출하려면

1. 필수 파라미터를 사용하여 작업 제출 요청을 생성합니다.

```
{
  "job": {
    "name": "my-job",
    "partition": "compute",
    "nodes": 1,
    "tasks": 1,
    "script": "#!/bin/bash\necho 'Hello from Slurm REST API'",
    "environment": ["PATH=/usr/local/bin:/usr/bin:/bin"]
  }
}
```

2. HTTP POST 요청을 사용하여 작업을 제출합니다.

```
curl -X POST \
  -H "Authorization: Bearer <jwt>" \
  -H "Content-Type: application/json" \
  -d '<job-json>' \
  https://<privateIpAddress>:6820/slurm/v0.0.43/job/submit
```

3. 모니터링 목적으로 응답에 반환된 작업 ID를 기록해 둡니다.

작업 상태를 모니터링하려면

1. 특정 작업에 대한 정보를 가져옵니다.

```
curl -X GET -H "Authorization: Bearer <jwt>" \
    https://<privateIpAddress>:6820/slurm/v0.0.43/job/<job-id>
```

2. 인증된 사용자의 모든 작업을 나열합니다.

```
curl -X GET -H "Authorization: Bearer <jwt>" \
    https://<privateIpAddress>:6820/slurm/v0.0.43/jobs
```

작업 취소

- DELETE 요청을 보내 특정 작업을 취소합니다.

```
curl -X DELETE -H "Authorization: Bearer <jwt>" \
    https://<privateIpAddress>:6820/slurm/v0.0.43/job/<job-id>
```

AWS PCS의 Slurm REST API FAQ

이 섹션에서는 AWS PCS의 Slurm REST API에 대해 자주 묻는 질문에 답변합니다.

Slurm REST API란 무엇입니까?

Slurm REST API는 Slurm 워크로드 관리자와 프로그래밍 방식으로 상호 작용할 수 있는 HTTP 인터페이스입니다. GET, POST, DELETE와 같은 표준 HTTP 메서드를 사용하여 클러스터에 대한 명령줄 액세스 없이 작업을 제출하고 클러스터 상태를 모니터링하며 리소스를 관리할 수 있습니다.

에서 생성한 토큰을 사용할 수 있습니까**`scontrol token`**?

아니요, 표준 `scontrol token` 출력은 AWS PCS와 호환되지 않습니다. PCS Slurm REST API에는 username(`sun`), POSIX 사용자 ID() 및 group ID(`uid`)를 포함하는 특정 자격 증명 클레임이 포함된 보강된 JWT 토큰이 필요합니다`gids`. IDs 표준 Slurm 토큰에는 이러한 필수 클레임이 없으며 API에서 거부됩니다.

VPC 외부에서 API에 액세스할 수 있나요?

아니요. REST API 엔드포인트는 Slurm 컨트롤러의 프라이빗 IP 주소를 사용하여 VPC 내에서만 액세스할 수 있습니다. 외부 액세스를 활성화하려면 VPC Link, API Gateway를 사용하여 Application Load Balancer와 같은 AWS 서비스를 구현하거나 보안 연결을 위해 VPC 피어링 또는 VPN 연결을 설정합니다.

API가 HTTPS 대신 HTTP를 사용하는 이유는 무엇입니까?

Slurm REST API는 클러스터의 프라이빗 네트워크 내에서 내부 엔드포인트가 되도록 설계되었습니다. 암호화가 필요한 프로덕션 배포의 경우 API 게이트웨이, 로드 밸런서 또는 역방향 프록시를 통해 아키텍처에서 더 높은 수준에서 SSL/TLS 종료를 구현할 수 있습니다.

REST API에 대한 액세스를 제어하려면 어떻게 해야 합니까?

클러스터의 보안 그룹 규칙을 구성하여 Slurm 컨트롤러의 포트 6820에 대한 액세스를 제한합니다. 신뢰할 수 있는 IP 범위 또는 VPC 내의 특정 소스로부터의 연결만 허용하도록 인바운드 규칙을 설정하여 API 엔드포인트에 대한 무단 액세스를 차단합니다.

JWT 서명 키를 교체하려면 어떻게 해야 합니까?

활성 인스턴스 없이 클러스터를 유지 관리 모드로 전환한 다음 AWS Secrets Manager를 통해 키 교체를 시작합니다. 교체가 완료되면 대기열을 다시 활성화합니다. 기존 JWT 토큰은 모두 유효하지 않게 되며 Secrets Manager의 새 서명 키를 사용하여 다시 생성해야 합니다.

REST API를 사용하려면 Slurm 회계를 활성화해야 하나요?

아니요, Slurm 회계는 작업 제출 및 모니터링과 같은 기본 REST API 작업에 필요하지 않습니다. 그러나 전체 `/slurmdb` 엔드포인트를 활성화하려면 회계가 필요합니다.

AWS PCS REST API에서 작동하는 타사 도구는 무엇입니까?

Prometheus용 Slurm Exporter, SlurmWeb 및 표준 Slurm REST API 형식을 따르는 사용자 지정 애플리케이션을 포함하여 많은 기존 Slurm REST API 클라이언트가 AWS PCS와 함께 작동해야 합니다. 그러나 인증`scontrol token`에를 사용하는 도구는 AWS PCS JWT 요구 사항으로 작동하려면 수정해야 합니다.

REST API를 사용하는 데 추가 비용이 발생하나요?

아니요. Slurm REST API 기능을 활성화하거나 사용하는 데 따르는 추가 요금은 없습니다. 기본 클러스터 리소스에 대한 비용만 평소와 같이 지불하면 됩니다.

REST API 문제를 해결하려면 어떻게 해야 합니까?

- 네트워크 연결 문제

  API 엔드포인트에 연결할 수 없는 경우 클러스터 컨트롤러에 HTTP 요청을 할 때 연결 제한 시간 또는 "연결 거부" 오류가 표시됩니다.

  조치: 클라이언트가 동일한 VPC에 있거나 적절한 네트워크 라우팅이 있는지 확인하고 보안 그룹이 소스 IP 또는 서브넷에서 포트 6820의 HTTP 트래픽을 허용하는지 확인합니다.

- Slurm REST 인증 문제

  JWT 토큰이 유효하지 않거나 만료되었거나 잘못 서명된 경우 API 요청은 응답의 오류 필드에 "프로토콜 인증 오류"를 반환합니다.

  오류 메시지 예:

```
{
"errors": [
    {
    "description": "Batch job submission failed",
    "error_number": 1007,
    "error": "Protocol authentication error",
    "source": "slurm_submit_batch_job()"
    }
  ]
}
```

조치: JWT 토큰의 형식이 올바른지, 만료되지 않았는지, Secrets Manager의 올바른 키로 서명되었는지 확인합니다. 토큰이 올바르게 구성되었고 필요한 클레임이 포함되어 있으며 올바른 인증 헤더 형식을 사용하고 있는지 확인합니다.

- 제출 후 작업이 실행되지 않음

  JWT 토큰이 유효하지만 내부 구조나 콘텐츠가 잘못된 경우 작업이 사유 코드와 함께 일시 중지(PD) 상태로 전환되었을 수 있습니다JobAdminHead. `scontrol show job <job-id>`를 사용하여 작업을 검사합니다. `JobState=PENDING, Reason=JobHeldAdmin`, 및가 표시됩니다`SystemComment=slurm_cred_create failure, holding job`.

  해야 할 일: 근본 원인은 JWT의 잘못된 값일 수 있습니다. 토큰이 적절하게 구성되어 있고 PCS 설명서에 따라 필요한 클레임이 포함되어 있는지 확인합니다.

- 작업 디렉터리 권한 문제

  JWT에 지정된 사용자 자격 증명에 작업의 작업 디렉터리에 대한 쓰기 권한이 없는 경우 액세스할 수 없는 디렉터리와 함께를 사용하는 것과 마찬가지로 권한 오류와 `sbatch --chdir` 함께 작업이 실패합니다.

  할 일: JWT 토큰에 지정된 사용자에게 작업의 작업 디렉터리에 대한 적절한 권한이 있는지 확인합니다.

- 여전히 문제가 발생하나요?

  1. REST API 사양에 대한 SchedMD의 설명서를 확인하세요.
  2. 오류에 대한 자세한 내용은 Slurm 컨트롤러 로그를 확인하세요(AWS PCS의 스케줄러 로그자세한 내용은 참조).

AWS PCS에서 Slurm을 사용하여 컴퓨팅 노드 재부팅

AWS PCS는 Slurm의 기본 `scontrol reboot` 명령을 지원합니다. EC2 인스턴스 교체 없이 컴퓨팅 노드를 재부팅하려면이 명령을 사용합니다. 다른 재부팅 방법(Amazon EC2 콘솔, , AWS CLI자동 패치 또는 시스템 유지 관리)으로 인해 AWS PCS는 EC2 인스턴스를 비정상으로 간주하고 교체합니다.

Slurm 재부팅의 이점

Slurm 재부팅은 클러스터 유지 관리에 몇 가지 이점을 제공합니다.

- 용량 보존 - 용량이 제한된 EC2 인스턴스를 다른 고객에게 잃지 않도록 합니다.

- 비용 절감 - 불필요한 인스턴스 교체 주기와 유휴 노드에 대한 지속적인 청구를 제거합니다.
- 더 빠른 복구 - 인스턴스 교체에 비해 프로비저닝 지연이 없습니다.
- 운영 유연성 - 메모리 누수를 지우고, 임시 파일을 제거하고, 성능이 저하된 상태에서 노드를 복구합니다.

Slurm 재부팅을 사용하는 경우

일반적인 운영 유지 관리 시나리오에는 Slurm 재부팅을 사용합니다.

- 문제 해결 - 특히 GPU 노드의 경우 성능 문제 또는 응답하지 않는 프로세스를 해결합니다.
- 리소스 정리 - 작업 성능에 영향을 미치는 메모리 누수/tmp,의 임시 파일 또는 중단된 프로세스를 지웁니다.
- 복구 - 전체 노드 교체가 필요하기 전에 중단되거나 성능이 저하된 상태에서 노드를 복구합니다.

제한 사항

- Slurm 관리자 사용자(루트 사용자)만 재부팅 명령을 실행할 수 있습니다.
- 재부팅 지원은 로`scontrol reboot`만 제한됩니다.
- RebootProgram 구성은 지원되지 않습니다.
- 콘솔 인터페이스 없음 - 명령줄 전용.

주제

- AWS PCS에서 Slurm을 사용하여 컴퓨팅 노드 재부팅
- AWS PCS에서 보류 중인 재부팅 취소
- AWS PCS에서 Slurm 재부팅 자주 묻는 질문
- AWS PCS에서 Slurm 재부팅 문제 해결

AWS PCS에서 Slurm을 사용하여 컴퓨팅 노드 재부팅

Slurm의 기본 재부팅 명령을 사용하여 성능 문제를 해결하거나, 리소스 문제를 해결하거나, EC2 인스턴스 용량 손실 없이 성능이 저하된 상태에서 복구할 수 있습니다.

사전 조건

- Slurm 관리자 권한(루트 사용자 액세스)
- AWS PCS 클러스터의 로그인 노드에 대한 액세스

절차

1. EC2 콘솔을 통해 로그인 노드에 연결합니다.
   a. EC2 콘솔에서 Instances(인스턴스)를 선택합니다.
   b. 로그인 노드 인스턴스를 선택합니다.
   c. 연결을 선택합니다.
2. `sinfo` 또는를 사용하여 대상 컴퓨팅 노드 이름을 식별합니다`scontrol show node`.

```
sinfo
# or
scontrol show node
```

3. 다음 옵션 중 하나를 사용하여 재부팅 명령을 실행합니다.

> **Warning**
>
> `scontrol reboot` 명령과 `nextstate=DOWN` 함께를 사용하지 마십시오. 이 파라미터는 노드를 비정상으로 표시하고 인스턴스 교체를 트리거합니다.

- 기본 재부팅(노드가 유휴 상태가 될 때까지 대기):

```
scontrol reboot nodename
```

- 즉시 재부팅(노드를 해제하고 작업이 완료되면 재부팅):

```
scontrol reboot ASAP nodename
```

- 이유와 함께 재부팅:

```
scontrol reboot ASAP reason="troubleshooting" nodename
```

- 재개 상태로 재부팅:

```
scontrol reboot ASAP nextstate=RESUME nodename
```

4. 를 사용하여 재부팅 진행 상황을 모니터링합니다`scontrol show node`.

```
scontrol show node nodename
```

5. 재부팅 완료 후 노드가 서비스로 돌아가는지 확인합니다.

AWS PCS에서 보류 중인 재부팅 취소

문제가 해결되거나 재부팅이 더 이상 필요하지 않을 때 불필요한 가동 중지를 방지하려면 보류 중인 재부팅을 취소합니다.

사전 조건

- Slurm 관리자 권한
- 노드에 보류 중인 재부팅이 있어야 합니다("재부팅이 실행됨" 상태 표시).
- 명령 실행을 위한 로그인 노드 액세스

절차

1. 로그인 노드에 연결합니다.
2. 를 사용하여 노드에 보류 중인 재부팅이 있는지 확인합니다`scontrol show node`.

```
scontrol show node nodename
```

노드 상태에서 "재부팅 실행"을 찾습니다.

3. cancel 명령을 실행합니다.

```
scontrol cancel_reboot nodename
```

4. 재부팅 취소를 확인하고 노드 상태가 정상으로 돌아가는지 확인합니다.

```
scontrol show node nodename
```

AWS PCS에서 Slurm 재부팅 자주 묻는 질문

AWS PCS에서 Slurm 재부팅 사용에 대한 일반적인 질문에 대한 답변을 찾습니다.

Slurm 재부팅 지원이란 무엇입니까?

네이티브 Slurm `scontrol reboot` 명령을 지원합니다. 이 명령을 사용하면 자동 인스턴스 교체 없이 컴퓨팅 노드를 재부팅하여 EC2 인스턴스 용량을 보존하고 운영 비용을 절감할 수 있습니다.

Slurm 재부팅 명령은 누가 사용할 수 있나요?

Slurm 관리자 사용자(루트 사용자)만 재부팅 명령을 실행할 수 있습니다. 를 사용하려는 일반 사용자는 노드에 영향을 주지 않고 Slurm으로부터 권한 거부 오류를 받게 `scontrol reboot` 됩니다.

재부팅 중에 작업을 실행하면 어떻게 되나요?

기본적으로 재부팅이 발생하기 전에 작업이 정상적으로 완료됩니다. ASAP 옵션을 사용하면 노드가 드레이닝되어 새 작업을 방지하고 현재 작업이 완료된 후 재부팅됩니다. 즉각적인 재부팅을 위해 작업을 취소하거나 다시 대기열에 추가할 수 있습니다.

EC2 콘솔 재부팅과 어떻게 다릅니까?

Slurm 재부팅은 EC2 인스턴스를 보존하고 교체를 방지하는 반면, EC2 콘솔 재부팅은 재부팅 프로세스 중 상태 확인 실패로 인해 인스턴스를 교체하기 위해 PCS를 트리거합니다.

사용자 지정 재부팅 스크립트를 구성할 수 있습니까?

아니요. 초기 릴리스에서는 RebootProgram 구성이 지원되지 않습니다. 이 기능은 사용자 지정 스크립트 지원 없이 표준 Slurm 재부팅 동작을 사용합니다.

Slurm 재부팅에는 얼마나 걸리나요?

재부팅 시간은 인스턴스 유형, 고객 부팅 프로세스, AMI 구성 및 작업을 먼저 완료해야 하는지 여부에 따라 달라집니다. 이 프로세스에는 작업이 완료될 때까지 대기, 물리적 재부팅, 상태 확인 및 슬러먼 데몬 등록이 포함됩니다.

재부팅 기록을 볼 수 있나요?

재부팅 이벤트는 CloudWatch를 통해 모니터링할 수 있는 Slurm 로그(slurmctld 및 slurmd)에 기록됩니다. 노드 상태의 이유 필드에는 프로세스 중 재부팅 이유가 표시됩니다.

재부팅 중에 노드가 멈추면 어떻게 되나요?

노드가 ResumeTimeout 내에서 재부팅 프로세스를 완료하지 않으면 DOWN으로 표시됩니다. CloudWatch 로그에서 오류를 확인하고, 네트워크 연결을 확인하고, 지연 로그를 검사합니다. 문제가 지속되면 AWS Support에 문의하세요.

여러 노드를 한 번에 재부팅할 수 있나요?

예, 재부팅 명령에서 여러 노드를 지정할 수 있습니다.

```
scontrol reboot ASAP node1,node2,node3
```

작업이 완료될 때까지 기다리지 않고 노드를 재부팅하려면 어떻게 해야 하나요?

다중 노드 작업에 영향을 미치는 문제가 있는 노드, 상당한 성능 저하 또는 불안정한 GPU 동작과 같은 문제가 발생할 때 즉시 노드 재부팅하는 경우 다음 두 가지 옵션이 있습니다.

- 취소 및 재부팅 - 먼저를 사용하여 영향을 받는 작업을 취소`scancel <job_id>`한 다음를 사용하여 즉시 재부팅을 시작합니다`scontrol reboot ASAP <nodename>`. 실행 중인 작업은 종료되며 노드가 복구된 후 다시 제출해야 합니다.
- 드레이닝 및 다시 대기열에 추가(영향력 낮음) - 드레이닝을 시작하고 로 재부팅한 다음 `scontrol reboot ASAP <nodename>`를 사용하여 영향을 받는 작업을 다시 대기열에 추가합니다`scontrol requeue <job_id>`. 그러면 작업을 취소하는 대신 작업이 다시 보류 중 상태로 전환됩니다.

nextstate=DOWN을 지정하면 어떻게 되나요?

`nextstate=DOWN`를 지정하면 재부팅 후 노드가 비정상으로 표시되고 인스턴스 교체가 트리거됩니다. 인스턴스 교체를 방지하려면 다음 상태를 지정하거나를 사용하지 마십시오`nextstate=RESUME`.

추가 리소스

- 기본 재부팅 절차는 섹션을 참조하세요AWS PCS에서 Slurm을 사용하여 컴퓨팅 노드 재부팅.
- 재부팅 문제 해결은 섹션을 참조하세요AWS PCS에서 Slurm 재부팅 문제 해결.
- Slurm 재부팅 설명서는 Slurm scontrol 설명서를 참조하세요.

AWS PCS에서 Slurm 재부팅 문제 해결

노드 재부팅 문제가 발생하면 먼저를 사용하여 노드 상태를 확인합니다`scontrol show node` *`nodename`*. 그런 다음 CloudWatch 로그에서 Slurm(slurmctld 및 slurmd) 및 시스템 로그를 모두 검사하여 잠재적 오류를 식별합니다.

기본 문제 해결을 위해 네트워크 연결을 확인하고, 보안 그룹 설정을 확인하고, 재부팅 후 필요한 모든 서비스가 실행 중인지 확인합니다. 기본 문제 해결 단계 후에도 문제가 지속되면 AWS Support에 문의

하십시오. 지원 팀에 문의할 때 관련 로그 발췌문, 노드 상태 정보 및 재부팅 시도 타임라인을 제공하여 해결 프로세스를 가속화합니다.

추가 리소스

- CloudWatch를 사용하여 AWS PCS 인스턴스를 모니터링하려면 Amazon CloudWatch를 사용하여 AWS PCS 인스턴스 모니터링을 참조하세요.
- 일반적인 문제 해결은 섹션을 참조하세요AWS 병렬 컴퓨팅 서비스의 문제 해결.
- Slurm 설명서는 Slurm 문제 해결 안내서를 참조하세요.

AWS PCS에서 사용자 지정 Slurm 설정 구성

사용자 지정 Slurm 설정을 사용하여 클러스터, 대기열 및 컴퓨팅 노드 그룹 리소스에서 추가 Slurm 파라미터를 구성합니다. 이 릴리스에는 대기열 리소스의 Slurm 설정에 대한 지원이 추가되어 파티션별 동작을 세부적으로 제어할 수 있습니다.

사용자 지정 Slurm 설정의 이점

사용자 지정 Slurm 설정은 AWS PCS 기반 HPC 환경을 세밀하게 제어할 수 있습니다. quality-of-service 구성 및 선점 정책을 통해 상세한 회계를 구현하고, 액세스 제어를 적용하고, 워크로드 실행을 최적화할 수 있습니다. 이러한 기능을 사용하면 중요한 작업이 효율적인 클러스터 사용률을 유지하면서 필요한 리소스를 받을 수 있습니다. GPU 가속 워크로드를 관리하든, 공정 공유 일정을 구현하든, 작업 수명 주기를 제어하든 관계없이 사용자 지정 설정은 HPC 인프라를 운영 요구 사항 및 연구 목표에 맞게 조정하는 데 도움이 됩니다.

사용자 지정 설정 구성

사용자 지정 Slurm 설정은 리소스 생성 중에 AWS 콘솔, CLI 또는 SDKs를 통해 구성하거나 나중에 업데이트 작업을 통해 수정할 수 있습니다.

AWS Management Console

모든 리소스 유형(클러스터, 대기열 또는 컴퓨팅 노드 그룹)에 대한 생성 또는 편집 페이지의 추가 스케줄러 설정으로 이동합니다.

새 설정을 추가하려면

1. 새 설정 추가를 선택합니다.
2. 드롭다운에서 파라미터 이름을 선택합니다(간단한 파라미터 설명 포함).

3. 해당 값을 입력합니다.

사용자 지정 설정을 해제하려면

1. 관련 파라미터/값 페어 옆에 있는 제거를 선택합니다.
2. 리소스를 생성하거나 업데이트합니다.

AWS CLI

사용자 지정 설정을 프로그래밍 방식으로 관리하려면 생성 또는 업데이트 작업에서 `SlurmCustomSettings` 필드를 사용합니다.

Example- 클러스터의 Prolog 파라미터 업데이트

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'SlurmCustomSettings=[{parameterName=Prolog,parameterValue="/path/to/prolog.sh"}]'
```

Example- 대기열을 클러스터Default의 로 설정

```
aws pcs update-queue \
    --cluster-identifier my-cluster \
    --queue-identifier my-queue \
    --slurm-configuration
 'SlurmCustomSettings=[{parameterName=Default,parameterValue=YES}]'
```

Example- Features 컴퓨팅 노드 그룹에서 사용자 지정 설정

```
aws pcs update-compute-node-group \
    --cluster-identifier my-cluster \
    --compute-node-group-identifier my-cng-1 \
    --slurm-configuration \
    'SlurmCustomSettings=[{parameterName=Features,parameterValue="gpu,nvme"}]'
```

검증 및 오류 처리

AWS PCS는 사용자 지정 Slurm 설정을 위한 다중 계층 검증 프로세스를 구현합니다. 생성 및 업데이트 작업 모두에서 다음을 포함하는 동기 검증을 수행합니다.

- 필드 수준 검사: 개별 설정에서 올바른 데이터 유형, 허용되는 값 및 형식 요구 사항을 검증합니다. 예를 들어 시간 값이 올바른 Slurm 형식이고 부울 값이 허용되는 Slurm 부울 표현을 사용하는지 확인합니다.
- 컨텍스트 인식 검증: 일부 설정은 더 광범위한 구성 컨텍스트와 비교하여 확인됩니다. 예를 들어 특정 파라미터는 Slurm 회계가 활성화된 경우에만 유효합니다.
- 설정 간 일관성: 상호 배타적인 옵션이 함께 설정되지 않고 상호 종속적인 설정이 올바르게 구성되었는지 확인합니다.

검증에 실패하면 특정 오류 코드(예: InvalidInput), 문제를 설명하는 명확한 오류 메시지, 잘못된 필드 목록 및 해당 오류 세부 정보가 `ValidationException` 포함된를 받게 됩니다.

이 초기 검증 중에 많은 문제가 발견되지만 구성을 적용할 때만 설정 간의 일부 복잡한 상호 작용이 명백해질 수 있습니다. 이러한 경우 유용한 오류 메시지와 함께 작업이 실패하고 부분적인 변경 사항이 롤백됩니다.

제한 사항

AWS PCS는 허용 목록 접근 방식을 구현하여 서비스 보안 및 운영 안정성을 보호합니다. 서비스 계정 보안을 손상시키거나 관리형 서비스 기능을 방해할 수 있는 설정은 제한됩니다. 그러나 고객 요구 사항을 지속적으로 평가하며 고객 피드백을 기반으로 추가 설정에 대한 지원을 추가할 수 있습니다.

주제

- AWS PCS 클러스터에 대한 사용자 지정 Slurm 설정
- AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정
- AWS PCS 대기열에 대한 사용자 지정 Slurm 설정
- AWS PCS에서 사용자 지정 Slurm 설정 문제 해결

AWS PCS 클러스터에 대한 사용자 지정 Slurm 설정

클러스터 수준에서 지원되는 사용자 지정 Slurm 설정은 다음과 같습니다.

- AccountingStorageEnforce
- AccountingStorageTRES
- AccountingStoreFlags
- DefMemPerCPU

- Epilog
- EnforcePartLimits
- FairShareDampeningFactor
- FirstJobId
- HealthCheckInterval
- HealthCheckNodeState
- HealthCheckProgram
- JobRequeue
- LaunchParameters
- Licenses
- MinJobAge

> **Note**
>
> AWS PCS 는에 대해 5초의 최소값을 지원합니다`MinJobAge`.

- OverTimeLimit
- PreemptExemptTime
- PreemptMode
- PreemptParameters
- PreemptType
- PriorityCalcPeriod
- PriorityDecayHalfLife
- PriorityFavorSmall
- PriorityFlags
- PriorityMaxAge
- PriorityUsageResetPeriod
- PriorityWeightAge
- PriorityWeightAssoc
- PriorityWeightFairshare

- PriorityWeightJobSize
- PriorityWeightPartition
- PriorityWeightQOS
- PriorityWeightTRES
- Prolog
- PrologFlags
- PropagatePrioProcess
- PropagateResourceLimits
- PropagateResourceLimitsExcept
- RequeueExit
- RequeueExitHold
- SchedulerParameters
- SelectTypeParameters
- SrunPortRange
- TaskEpilog
- TaskPluginParam
- TaskProlog
- TrackWCKey
- UnkillableStepProgram
- UnkillableStepTimeout

AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정

컴퓨팅 노드 그룹 수준에서 지원되는 사용자 지정 Slurm 설정은 다음과 같습니다.

- CpuSpecList
- Features
- MemSpecLimit
- RealMemory
- Weight

AWS PCS 대기열에 대한 사용자 지정 Slurm 설정

대기열 수준에서 지원되는 사용자 지정 Slurm 설정은 다음과 같습니다.

- AllowAccounts
- AllowQoS
- Default
- DefaultTime
- DenyAccounts
- DenyQoS
- ExclusiveUser
- GraceTime
- MaxTime
- OverSubscribe
- OverTimeLimit
- PreemptMode
- PriorityJobFactor
- PriorityTier
- QOS
- TRESBillingWeights

AWS PCS에서 사용자 지정 Slurm 설정 문제 해결

Slurm 사용자 지정 설정으로 AWS PCS 리소스를 생성하거나 업데이트할 때 오류가 발생하면 로깅을 사용하여 문제를 진단하고 해결할 수 있습니다.

호환되지 않는 Slurm 사용자 지정 설정 문제 해결

문제: 클러스터, 컴퓨팅 노드 그룹 또는 대기열 작업을 수행할 때 다음과 비슷한 오류 메시지가 표시됩니다.

```
{OPERATION} failed. The Slurm custom settings of the cluster might be incompatible.
 Check the settings and try again.
```

이 오류는 다음 작업에서 발생할 수 있습니다.

- CreateCluster
- CreateComputeNodeGroup
- UpdateComputeNodeGroup
- CreateQueue
- UpdateQueue

해결 방법: 로깅을 활성화하여 특정 문제를 이해하고 호환되지 않는 설정을 해결합니다.

호환되지 않는 Slurm 사용자 지정 설정 문제를 해결하려면

1. 클러스터가 아직 존재하지 않는 경우 클러스터를 생성하거나 기존 클러스터가 로깅을 활성화할 수 있는 상태인지 확인합니다.
2. 클러스터에 대한 로깅을 활성화합니다. 자세한 지침은 AWS PCS에 대한 로깅 및 모니터링 섹션을 참조하세요.

   > **Note**
   >
   > 클러스터가 생성되면 로깅을 활성화할 수 있습니다.

3. 로그를 검토하여 비호환성을 유발하는 특정 Slurm 구성 문제를 식별합니다.
4. 로그 정보를 기반으로 호환되지 않는 사용자 지정 설정을 수정하고 작업을 다시 시도합니다.

지원되는 Slurm 사용자 지정 설정에 대한 자세한 내용은 다음을 참조하세요.

- AWS PCS 클러스터에 대한 사용자 지정 Slurm 설정
- AWS PCS 컴퓨팅 노드 그룹에 대한 사용자 지정 Slurm 설정
- AWS PCS 대기열에 대한 사용자 지정 Slurm 설정

AWS PCS에서 사용자 지정 cgroup 설정 구성

Slurm은 Linux cgroup 하위 시스템을 사용하여 메모리, CPU 코어, 디바이스 및 스왑 공간을 포함한 작업에 대한 리소스를 관리하고 제한합니다. AWS PCS를 사용하면 클러스터 생성 또는 업데이트 `SlurmConfiguration` 중에의 `CgroupCustomSettings` 속성을 통해 클러스터 수준에서 `cgroup.conf` 설정을 사용자 지정할 수 있습니다.

cgroup 설정 구성

Cgroup 사용자 지정 설정은 클러스터 생성 중에 AWS 콘솔, CLI 또는 SDKs를 통해 구성하거나 나중에 업데이트 작업을 통해 수정할 수 있습니다.

AWS Management Console

클러스터 리소스의 생성 또는 편집 페이지에서 추가 스케줄러 설정으로 이동합니다.

새 설정을 추가하려면

1. 새 설정 추가를 선택합니다.
2. 드롭다운에서 파라미터 이름을 선택합니다(간단한 파라미터 설명 포함).
3. 해당 값을 입력합니다.

사용자 지정 설정을 해제하려면

1. 관련 파라미터/값 페어 옆에 있는 제거를 선택합니다.
2. 리소스를 생성하거나 업데이트합니다.

AWS CLI

cgroup 설정을 프로그래밍 방식으로 관리하려면 클러스터 생성 또는 업데이트 작업에서 `CgroupCustomSettings` 필드를 사용합니다.

Example- 클러스터ConstrainRAMSpace에서 설정

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'CgroupCustomSettings=[{parameterName=ConstrainRAMSpace,parameterValue="yes"}]'
```

클러스터에 지원되는 cgroup 설정

클러스터 수준에서는 다음과 같은 사용자 지정 cgroup 설정이 지원됩니다.

- AllowedRAMSpace
- AllowedSwapSpace

- ConstrainCores
- ConstrainDevices
- ConstrainRAMSpace
- ConstrainSwapSpace
- IgnoreSystemd
- MaxRAMPercent
- MaxSwapPercent
- MinRAMSpace
- SignalChildrenProcesses

AWS PCS에서 사용자 지정 SlurmDBD 설정 구성

Slurm의 데이터베이스 데몬(slurmdbd)은 회계 데이터, 데이터 보존 정책 및 개인 정보 제어를 관리합니다. AWS PCS를 사용하면 클러스터 생성 또는 업데이트 `SlurmConfiguration` 중에 `SlurmdbdCustomSettings` 속성을 통해 클러스터 수준에서 `slurmdbd.conf` 설정을 사용자 지정할 수 있습니다.

slurmdbd 설정 구성

Slurmdbd 사용자 지정 설정은 클러스터 생성 중에 AWS 콘솔, CLI 또는 SDKs를 통해 구성하거나 나중에 업데이트 작업을 통해 수정할 수 있습니다.

AWS Management Console

클러스터 리소스의 생성 또는 편집 페이지에서 추가 스케줄러 설정으로 이동합니다.

새 설정을 추가하려면

1. 새 설정 추가를 선택합니다.
2. 드롭다운에서 파라미터 이름을 선택합니다(간단한 파라미터 설명 포함).
3. 해당 값을 입력합니다.

사용자 지정 설정을 해제하려면

1. 관련 파라미터/값 페어 옆에 있는 제거를 선택합니다.

2. 리소스를 생성하거나 업데이트합니다.

AWS CLI

slurmdbd 설정을 프로그래밍 방식으로 관리하려면 클러스터 생성 또는 업데이트 작업에서 `SlurmdbdCustomSettings` 필드를 사용합니다.

Example- 클러스터TrackWCKey에서 설정

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'SlurmdbdCustomSettings=[{parameterName=TrackWCKey,parameterValue="yes"}]'
```

클러스터에 지원되는 slurmdbd 설정

클러스터 수준에서 지원되는 사용자 지정 slurmdbd 설정은 다음과 같습니다.

- AllowNoDefAcct
- AllResourcesAbsolute
- CommitDelay
- DefaultQOS
- MaxQueryTimeRange
- Parameters
- PrivateData
- PurgeEventAfter
- PurgeJobAfter
- PurgeResvAfter
- PurgeStepAfter
- PurgeSuspendAfter
- PurgeTXNAfter
- PurgeUsageAfter
- TrackWCKey

SPANK 플러그인을 사용하여 AWS PCS에서 Slurm 기능 확장

SPANK(Slurm Plug-in Architecture for Node and job Kontrol) 플러그인을 사용하여 AWS PCS 클러스터에서 작업을 시작하고 실행하는 동안 Slurm의 동작을 확장하고 수정할 수 있습니다. SPANK 플러그인은 작업 시작 단계를 가로채고 수정할 수 있는 일반 인터페이스를 제공합니다.

컴퓨팅 노드 AMI에 SPANK 플러그인을 설치하고 워크로드 요구 사항에 맞게 Slurm 클러스터의 동작을 사용자 지정하도록 구성합니다. SPANK에 대한 자세한 내용은 SchedMD 웹 사이트의 SPANK 설명서를 참조하세요.

목차

AWS PCS에 SPANK 플러그인 설치

플러그인 설명서에 따라 AMI에 SPANK 플러그인을 설치합니다.

클러스터의 특정 Slurm 버전에 대한 SPANK 플러그인을 컴파일합니다. AWS PCS에서 제공하는 Slurm 설치 관리자는 Slurm을에 저장합니다`/opt/aws/pcs/scheduler/slurm-`*`version`*. 플러그인을 컴파일할 때 Slurm 버전을 지정합니다.

다음 예제에서는 일부 플러그인에 대해 Slurm 버전을 지정하는 방법을 보여줍니다.

```
export CFLAGS="-I/opt/aws/pcs/scheduler/slurm-version/include"
```

AMI에 여러 Slurm 버전이 있는 경우 각 버전에 대한 플러그인을 컴파일합니다. 컴파일된 플러그인을 버전이 지정된 폴더에 저장합니다.

다음 예제에서는 일부 플러그인의 대상 폴더를 지정하는 방법을 보여줍니다.

```
export DESTDIR="your-preferred-versioned-path"
```

> **Important**
>
> 플러그인에는 다양한 변수가 필요할 수 있습니다. 설치 중인 플러그인에 대한 공식 설명서를 참조하세요.

AWS PCS에서 SPANK 플러그인 구성

기본적으로 구성 파일은에 저장합니다`/etc/aws/pcs/scheduler/slurm-version/plugstack.conf.d/`.

SPANK 구성을 다른 위치에 저장하려면 기본 디렉터리의 구성 파일에 위치를 추가합니다.

다음 예제에서는 다른 디렉터리의 구성 파일을 포함하는 방법을 보여줍니다.

```
# content of /etc/aws/pcs/scheduler/slurm-version/any-filename.conf
include path-to-your-configuration-folder/*.conf
include path-to-a-second-configuration-folder/*.conf
```

각 구성을 전용 파일 또는 공통 파일에 저장합니다. 여러 구성 파일을 사용할 수 있습니다.

다음 예제에서는 샘플 구성 파일을 보여줍니다.

```
# content of path-to-your-or-default-config-folder/filename-1.conf
required path-to-plugin-1 arguments
optional path-to-plugin-2 arguments
```

```
# content of path-to-your-or-default-config-folder/filename-2.conf
required path-to-plugin-3 arguments
```

플러그인 구성 방법에 대한 자세한 내용은 SchedMD 웹 사이트의 SPANK 구성 설명서를 참조하세요.

Important

플러그인 구성에 대한 무단 변경을 방지하려면 폴더 권한을 설정합니다.

Note

AWS PCS는 SPANK 플러그인을 관리하지 않습니다. 플러그인과 관련된 오류가 발생하는 경우 컴퓨팅 노드의 오류 로그를 확인합니다.

Note

Slurm은 SPANK 구성을 로드할 때 다음과 유사한 오류를 잘못 기록합니다.

```
error: "Include" failed in file /etc/slurm/plugstack.conf line 3
```

이 오류는 무시할 수 있습니다. SPANK 플러그인의 작동 방식에는 영향을 주지 않습니다.

AWS PCS의 SPANK 플러그인에 대해 자주 묻는 질문

이 섹션에서는 AWS PCS 클러스터에 SPANK 플러그인을 설치하고 구성하는 방법에 대한 일반적인 질문을 다룹니다.

로그인 노드와 컴퓨팅 노드 모두에 SPANK 플러그인을 설치해야 합니까?

일부 SPANK 플러그인은 모든 노드에 설치할 필요가 없지만 호환성을 높이려면 모든 노드에 모든 SPANK 플러그인을 설치하는 것이 좋습니다.

SPANK 플러그인의 프로덕션 사용에 필요한 추가 구성은 무엇입니까?

예제에 표시된 기본 설치 및 구성 외에도 프로덕션 배포에는 일반적으로 추가 설정이 필요합니다. Pyxis와 같은 컨테이너 기반 플러그인을 사용하려면 Enroot에 대한 환경 변수를 설정하고, PMI(프로세스 관리 인터페이스)를 활성화하고, 컨테이너 런타임에 대한 권한을 구성해야 할 수 있습니다. 자세한 프로덕션 배포 요구 사항은 특정 플러그인의 설명서를 참조하세요.

SPANK 플러그인 문제를 해결하려면 어떻게 해야 합니까?

AWS PCS는 SPANK 플러그인을 관리하지 않습니다. 컴퓨팅 노드의 오류 로그를 검사하여 문제를 해결합니다.

Slurm CLI 필터 플러그인을 사용하여 AWS PCS에서 작업 제출 사용자 지정

AWS PCS는 로그인 및 컴퓨팅 노드에서 작업 제출 파라미터를 검증하고 수정하는 사용자 지정 Lua 스크립트를 실행하기 위해 Slurm CLI 필터 플러그인을 지원합니다. CLI 필터 플러그인에 대한 자세한 내용은 SchedMD 웹 사이트의 cli_filter Plugin API 설명서를 참조하세요.

요구 사항

CLI 필터 플러그인에는 Slurm 버전 24.11 이상과 모든 로그인 및 컴퓨팅 노드에 배포된 Lua 스크립트가 필요합니다.

> **Important**
>
> Slurm 버전 24.11 및 25.05의 경우 CLI 필터 플러그인은 AWS PCS Slurm 설치 관리자(버전 24.11.6-2+ 또는 25.05.4-1+)를 사용하여 Slurm을 설치해야 합니다. Slurm 설치에 대한 자세한 내용은 섹션을 참조하세요3단계 - Slurm 설치.

제한 사항 및 보안 고려 사항

- 보안 적용 - CLI 필터 플러그인은 사용자가 쉽게 우회할 수 있으며 보안에 중요한 정책에 사용해서는 안 됩니다. 사용자는 작업을 제출하는 동안 비활성화된 사용자 지정 구성을 제공하여 CLI 필터 플러그인을 `CLIFilterPlugins` 비활성화할 수 있습니다.
- Lua 구현 전용 - Lua 스크립트 구현이 지원됩니다. C 구현은 지원되지 않습니다.

주제

- AWS PCS 클러스터에서 Slurm CLI 필터 플러그인 구성
- Amazon S3를 사용하여 AWS PCS에 CLI 필터 플러그인 스크립트 배포
- Slurm 작업 제출 플러그인 스크립트를 번역하여 AWS PCS에서 CLI 필터 플러그인 사용
- AWS PCS의 Slurm CLI 필터 플러그인에 대해 자주 묻는 질문
- AWS PCS의 Slurm CLI 필터 플러그인 문제 해결

AWS PCS 클러스터에서 Slurm CLI 필터 플러그인 구성

새 AWS PCS 클러스터를 생성할 때 CLI 필터 플러그인을 구성합니다. 클러스터를 다시 생성하지 않고 업데이트 API 또는 콘솔을 사용하여 기존 클러스터에서 CLI 필터 플러그인을 활성화하거나 비활성화할 수 있습니다.

사전 조건

CLI 필터 플러그인을 구성하기 전에 다음 작업을 완료합니다.

- CLI 필터 플러그인 API를 구현하는 Lua 스크립트 작성 및 테스트
- Lua 스크립트의 이름을 정확히 지정합니다. `cli_filter.lua`
- 스크립트를 모든 클러스터 인스턴스(AMI, S3 또는 파일 시스템)에 배포하는 방법을 선택합니다.
- Slurm 버전 24.11 이상을 사용하고 있는지 확인

새 클러스터에서 CLI 필터 플러그인 활성화

AWS PCS console

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 클러스터 생성을 선택합니다.
4. 유효한 버전의 Slurm(버전 24.11 이상)을 선택합니다.
5. 스케줄러 설정에서 추가 스케줄러 설정을 확장합니다.
6. 파라미터 이름이 로 설정`CliFilterPlugins`되고 파라미터 값이 로 설정된 새 Slurm 사용자 지정 설정을 추가합니다`cli_filter/lua`.
7. 나머지 클러스터 구성을 완료하고 클러스터를 생성합니다.

AWS PCS API

`CreateCluster` API 작업에 대한 호출에 `slurmCustomSettings` 구성을 제공합니다. 를 `parameterName` 로 `CliFilterPlugins` 설정하고를 `parameterValue`로 설정합니다`cli_filter/lua`. 자세한 내용은 PCS API 참조의 CreateCluster를 참조하세요. AWS

다음 예제에서는 AWS CLI 를 사용하여 `CreateCluster` API 작업을 호출합니다. 사용자 지정 설정은 CLI 필터 플러그인을 `CliFilterPlugins=cli_filter/lua` 활성화합니다.

```
aws pcs create-cluster --cluster-name cluster-name \
--scheduler type=SLURM,version=24.11 \
--size SMALL \
--networking subnetIds=cluster-subnet-id,securityGroupIds=cluster-security-group-id
 \
--slurm-configuration \
'slurmCustomSettings=[{parameterName=CliFilterPlugins,parameterValue="cli_filter/
lua"}]'
```

CLI 필터 플러그인 스크립트 배포

CLI 필터 플러그인 스크립트를 클러스터에 배포하려면

1. 컴퓨팅 노드 그룹에 사용되는 모든 AMIs에 Slurm이 설치되어 있는지 AWS PCS Slurm 설치 관리자를 통해 확인합니다.

> **Note**
>
> 모든 컴퓨팅 노드 그룹에 대해 AWS PCS 샘플 AMI를 사용하는 경우이 단계를 건너뜁니다. Slurm이 이미 설치되어 있습니다.

2. 클러스터의 모든 인스턴스/etc/aws/pcs/scheduler/slurm-<version>/cli_filter.lua에 cli_filter.lua 스크립트를 배포합니다.

   예를 들어 Slurm 버전 24.11의 경우:

   ```
   /etc/aws/pcs/scheduler/slurm-24.11/cli_filter.lua
   ```

3. 준비된 AMIs를 사용하여 모든 로그인 및 컴퓨팅 노드를 시작합니다.
4. 작업 제출을 테스트하여 CLI 필터 플러그인이 올바르게 실행되는지 확인합니다.

기존 클러스터에서 CLI 필터 플러그인 활성화 또는 비활성화

인프라를 다시 빌드하지 않고도 기존 클러스터에서 CLI 필터 플러그인을 활성화하거나 비활성화할 수 있습니다. 자세한 내용은 AWS PCS에서 클러스터 업데이트 단원을 참조하십시오.

AWS PCS console

1. https://console.aws.amazon.com/pcs/ AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 업데이트할 클러스터를 선택합니다.
4. 작업 편집을 선택합니다.
5. 클러스터 편집 페이지의 추가 스케줄러 설정에서 다음을 수행합니다.
   - CLI 필터 플러그인 활성화: 파라미터 이름이 로 설정CliFilterPlugins되고 파라미터 값이 로 설정된 새 Slurm 사용자 지정 설정을 추가합니다cli_filter/lua.
   - CLI 필터 플러그인을 비활성화하려면: 기존 CliFilterPlugins 설정을 제거합니다.
6. 클러스터 업데이트를 선택하여 변경 사항을 제출합니다.
7. 프로세스 중에 "업데이트"로 표시되고 업데이트가 완료되면 "활성"으로 표시되는 클러스터 상태를 모니터링합니다.

AWS PCS API

`UpdateCluster` API 작업을 사용하여 CLI 필터 플러그인을 활성화하거나 비활성화합니다. 자세한 내용은 PCS API 참조의 UpdateCluster를 참조하세요. AWS

기존 클러스터에서 CLI 필터 플러그인을 활성화하려면:

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'slurmCustomSettings=[{parameterName=CliFilterPlugins,parameterValue="cli_filter/
lua"}]'
```

기존 클러스터에서 CLI 필터 플러그인을 비활성화하려면:

```
aws pcs update-cluster --cluster-identifier my-cluster \
--slurm-configuration \
'slurmCustomSettings=[]'
```

예상 결과

구성을 완료한 후:

- CLI 필터 플러그인이 켜진 상태로 클러스터가 생성됩니다.
- 작업 제출은 Slurm 컨트롤러에 도달하기 전에 사용자 지정 검증 로직을 트리거합니다.
- 규정 미준수 작업은 사용자 지정 오류 메시지와 함께 거부됩니다.
- 규정 준수 작업은 Slurm 스케줄러를 통해 정상적으로 진행됩니다.

문제 해결

모든 노드에서 CLI 필터 플러그인 스크립트 누락

증상: 플러그인 로드 오류와 함께 작업 제출이 즉시 실패합니다.

가능한 원인: 스크립트가 모든 인스턴스에 배포되지 않았거나 파일 경로 또는 이름이 잘못되었습니다.

해결 방법: 스크립트가 정확한 파일 이름을 가진 모든 로그인 및 컴퓨팅 노드에 올바른 경로에 있는지 확인합니다`cli_filter.lua`.

잘못된 CLI 필터 플러그인 구성

증상: 클러스터 생성이 실패하고 검증 오류가 발생합니다.

가능한 원인: `CliFilterPlugins` 파라미터가 `cli_filter/lua` 형식으로 설정되지 않았습니다.

해결 방법: `cli_filter/lua`에서 정확한 파라미터 값을 사용합니다`slurmCustomSettings`.

Amazon S3를 사용하여 AWS PCS에 CLI 필터 플러그인 스크립트 배포

AMI를 재구축하지 않고 라이브 클러스터에서 작업 제출 로직을 업데이트하려는 경우 S3를 사용하여 CLI 필터 플러그인 스크립트를 배포합니다. AMIs 이 접근 방식은 사용자 데이터를 사용하여 인스턴스 시작 중에 S3에서 스크립트를 다운로드합니다.

사전 조건

S3를 사용하여 스크립트를 배포하기 전에 다음 작업을 완료합니다.

- CLI 필터 플러그인 Lua 스크립트를 사용하여 S3 버킷 생성
- S3 버킷에 대한 읽기 액세스 권한으로 IAM 인스턴스 프로파일 구성
- 인터넷 없이 직접 액세스할 수 있도록 S3 VPC Gateway 엔드포인트 설정
- S3에서 다운로드할 사용자 데이터 스크립트 준비

S3를 사용하여 CLI 필터 플러그인 스크립트를 배포하려면

1. `cli_filter.lua` 스크립트를 S3 버킷에 업로드합니다.
2. 버킷에 대한 S3 읽기 권한으로 IAM 인스턴스 프로파일을 구성합니다.
3. 시작 템플릿 사용자 데이터에 셸 코드를 추가하여 스크립트를 다운로드합니다.

```
aws s3 cp s3://my-bucket/cli_filter.lua /etc/aws/pcs/scheduler/slurm-24.11/
cli_filter.lua
chmod 644 /etc/aws/pcs/scheduler/slurm-24.11/cli_filter.lua
```

4. 업데이트된 시작 템플릿을 사용하여 컴퓨팅 노드 그룹을 배포합니다.
5. 작업 제출을 테스트하여 스크립트 기능을 확인합니다.

예상 결과

S3 배포를 완료한 후:

- CLI 필터 플러그인 스크립트는 시작 중에 모든 인스턴스에 자동으로 다운로드됩니다.
- S3의 스크립트 업데이트는 새로 시작된 인스턴스에 반영됩니다.
- 작업 제출 정책은 클러스터 전체에 일관되게 적용됩니다.

문제 해결

S3 액세스 거부됨

증상: 인스턴스 시작이 실패하거나 스크립트가 다운로드되지 않았습니다.

가능한 원인: IAM 권한 또는 S3 VPC 엔드포인트 누락.

해결 방법: IAM 인스턴스 프로파일에 `s3:GetObject` 권한이 있고 S3 VPC 엔드포인트가 구성되어 있는지 확인합니다.

Slurm 작업 제출 플러그인 스크립트를 번역하여 AWS PCS에서 CLI 필터 플러그인 사용

다른 Slurm 환경에서 마이그레이션할 때 기존 작업 제출 플러그인 Lua 스크립트를 CLI 필터 플러그인으로 변환합니다. 변환 프로세스에는 CLI 필터 플러그인 API와 함께 작동하도록 함수 이름 및 필드 액세스 패턴을 업데이트하는 작업이 포함됩니다.

사전 조건

스크립트를 번역하기 전에 다음 작업을 완료합니다.

- 기존 작업 제출 플러그인 Lua 스크립트 검토
- 작업 제출과 CLI 필터 플러그인 APIs의 차이점 이해
- Slurm CLI 필터 플러그인 설명서 액세스

작업 제출 플러그인 스크립트를 CLI 필터 플러그인으로 변환하려면

1. 기존 작업 제출 플러그인 스크립트 함수(`slurm_job_submit`, `slurm_job_modify`)를 검토합니다.

2. 동등한 CLI 필터 플러그인 함수를 식별합니다.
   - `slurm_job_submit`는 `slurm_cli_pre_submit`가 됩니다.
   - 기본 파라미터 설정에 `slurm_cli_setup_defaults` 추가
   - 제출 후 작업에 `slurm_cli_post_submit` 추가
3. 작업 검증 로직을 `job_desc` 필드에서 `options` 배열 액세스로 변환합니다.
   - `job_desc.account`는 `options["account"]`가 됩니다.
   - `job_desc.partition`는 `options["partition"]`가 됩니다.
   - `job_desc.features`는 `options["constraint"]`가 됩니다.
4. 로깅 호출을에서 `slurm.log_user()`로 업데이트합니다`slurm.log_error()`.
5. 개발 클러스터에서 번역된 스크립트를 테스트합니다.
6. 표준 CLI 필터 플러그인 배포 프로세스에 따라 프로덕션 클러스터에 배포합니다.

예상 결과

번역을 완료한 후:

- 번역된 스크립트는 동등한 작업 제출 검증을 제공합니다.
- 사용자는 원래 작업 제출 플러그인과 유사한 오류 메시지와 프롬프트를 볼 수 있습니다.
- 작업 제출 정책은 AWS PCS로 마이그레이션하는 동안 유지됩니다.

문제 해결

스크립트 번역 오류

증상: Lua 실행 오류와 함께 작업 제출이 실패합니다.

가능한 원인: 변환된 스크립트의 필드 액세스 또는 함수 호출이 잘못되었습니다.

해결 방법: CLI 필터 플러그인 API 설명서를 검토하고 작업 제출 인터페이스와 CLI 필터 인터페이스 간의 필드 매핑을 비교합니다.

AWS PCS의 Slurm CLI 필터 플러그인에 대해 자주 묻는 질문

CLI 필터 플러그인에 대해 자주 묻는 질문을 검토합니다.

CLI 필터 플러그인과 작업 제출 플러그인의 차이점은 무엇인가요?

CLI 필터 플러그인은 작업 제출이 컨트롤러에 도달하기 전에 로그인 및 컴퓨팅 노드에서 클라이언트 측을 실행하는 반면, 작업 제출 플러그인은 작업 제출 후 컨트롤러에서 서버 측을 실행합니다. CLI 필터 플러그인은 사용자가 우회할 수 있지만 컨트롤러 잠금은 유지하지 않지만 작업 제출은 안전하지만 실행 중에 클러스터 성능에 영향을 미칠 수 있습니다.

AWS PCS는 Slurm 작업 제출 플러그인을 지원하나요?

아니요, 작업 제출 플러그인은 AWS PCS에서 지원되지 않습니다. 대신 CLI 필터 플러그인을 작업 제출 검증 및 수정에 사용합니다.

CLI 필터 플러그인을 보안 적용에 사용할 수 있나요?

아니요. CLI 필터 플러그인은 확인된 사용자가 우회할 수 있으며 보안 적용에 의존해서는 안 됩니다. 보안에 중요한 정책이 아닌 사용자 경험 개선, 기본 파라미터 설정 및 정책 지침에 사용합니다.

로그인 노드뿐만 아니라 모든 컴퓨팅 노드에 스크립트가 있어야 하는 이유는 무엇입니까?

와 같은 Slurm 명령은 컴퓨팅 노드의 작업 스크립트 내에서 실행할 `srun` 수 있으며, 이는 CLI 필터 플러그인 실행도 트리거합니다. Slurm 명령이 실행되는 모든 곳에서 스크립트를 사용할 수 있어야 합니다.

라이브 클러스터에서 CLI 필터 플러그인 스크립트를 수정할 수 있나요?

예, S3 또는 파일 시스템 배포 접근 방식을 사용하는 경우 가능합니다. 새 인스턴스는 업데이트된 스크립트를 가져오지만 기존 인스턴스는 스크립트를 수동으로 또는 선택한 배포 방법을 통해 업데이트해야 합니다.

다른 컴퓨팅 노드 그룹에서 다른 CLI 필터 플러그인 스크립트를 사용할 수 있나요?

예, 하지만 권장되지 않습니다. 서로 다른 컴퓨팅 노드 그룹에 서로 다른 로직을 가진 스크립트를 제공할 수 있지만 상호 종속성을 관리하고 중복 로직을 방지하는 것은 사용자의 책임입니다. 대부분의 고객은 전체 클러스터에서 하나의 로직 세트를 제공합니다.

Lua 대신 C 구현과 함께 CLI 필터 플러그인을 사용할 수 있나요?

C 구현은 지원되지 않습니다. Lua 스크립트 구현만 AWS PCS에서 지원됩니다. SchedMD는 CLI 필터 플러그인을 구현할 때 고객이 쉽게 사용할 수 있도록 C 대신 Lua를 사용할 것을 권장합니다.

기존 클러스터에서 CLI 필터 플러그인을 켜거나 끌 수 있습니까?

예, 클러스터를 다시 생성하지 않고 업데이트 API를 사용하여 기존 클러스터에서 CLI 필터 플러그인을 활성화하거나 비활성화할 수 있습니다.

AWS PCS의 Slurm CLI 필터 플러그인 문제 해결

이 문제 해결 정보를 사용하여 일반적인 CLI 필터 플러그인 문제를 해결합니다.

플러그인 로드 오류와 함께 작업 제출이 즉시 실패함

증상: 사용자가 작업을 제출할 때 CLI 필터 플러그인 누락 또는 실패에 대한 오류 메시지를 수신합니다.

가능한 원인:

- 하나 이상의 노드에서 CLI 필터 플러그인 스크립트 누락
- 잘못된 스크립트 파일 이름(정확히 여야 함`cli_filter.lua`)
- 잘못된 디렉터리 경로에 배포된 스크립트
- 스크립트에 잘못된 파일 권한이 있음

해결 방법:

- 스크립트가 모든 로그인 및 컴퓨팅 노드의 `/etc/aws/pcs/scheduler/slurm-<version>/cli_filter.lua`에 존재하는지 확인
- 스크립트 파일 이름이 정확한지 확인 `cli_filter.lua`
- 스크립트에 읽기 가능한 권한(644 이상)이 있는지 확인합니다.
- 전체 클러스터에 배포하기 전에 단일 로그인 노드에서 스크립트 배포 테스트

CLI 필터 플러그인 검증 오류와 함께 클러스터 생성 실패

증상: 잘못된 `CliFilterPlugins` 파라미터에 대한 오류로 클러스터 생성이 실패합니다.

가능한 원인:

- 의 잘못된 파라미터 값 형식 `slurmCustomSettings`
- 파라미터 이름 또는 값의 오타

해결 방법:

- 정확한 파라미터 이름 사용: `CliFilterPlugins`
- 정확한 파라미터 값을 사용합니다. `cli_filter/lua`
- `slurmCustomSettings` 배열에서 JSON 구문 확인

CLI 필터 플러그인 스크립트가 실행되지만 작업 검증이 예상대로 작동하지 않음

증상: 작업이 성공적으로 제출되었지만 사용자 지정 검증 로직이 트리거되지 않거나 예기치 않은 결과를 생성합니다.

가능한 원인:

- Lua 스크립트 구문 오류
- 잘못된 필드 액세스 패턴(CLI 필터 플러그인 대신 작업 제출 플러그인 구문 사용)
- 검증 조건의 로직 오류

해결 방법:

- Lua 스크립트에서 구문 오류 검토
- 필드 액세스가 대신 `options["field_name"]` 형식을 사용하는지 확인 `job_desc.field_name`
- 로깅 문을 추가하여 스크립트 실행 흐름 디버그
- 간단한 검증 사례를 먼저 사용하여 스크립트 로직 테스트

S3 스크립트 배포 실패

증상: 인스턴스가 시작되지만 CLI 필터 플러그인 스크립트가 S3에서 다운로드되지 않습니다.

가능한 원인:

- IAM 인스턴스 프로파일에 S3 읽기 권한이 없음
- S3 VPC 엔드포인트가 구성되지 않음
- 사용자 데이터의 잘못된 S3 버킷 또는 객체 경로

해결 방법:

- IAM 인스턴스 프로파일에 버킷에 대한 `s3:GetObject` 권한이 있는지 확인
- 직접 액세스를 위한 S3 VPC Gateway 엔드포인트 구성
- 사용자 데이터 스크립트에서 S3 버킷 이름 및 객체 경로 확인
- 인스턴스 사용자 데이터 로그에서 S3 다운로드 오류 검토

AWS 병렬 컴퓨팅 서비스의 보안

의 클라우드 보안 AWS 이 최우선 순위입니다. AWS 고객은 가장 보안에 민감한 조직의 요구 사항을 충족하도록 구축된 데이터 센터 및 네트워크 아키텍처의 이점을 누릴 수 있습니다.

보안은 AWS 와 사용자 간의 공동 책임입니다. 공동 책임 모델은 이 사항을 클라우드의 보안 및 클라우드 내 보안으로 설명합니다.

- 클라우드 보안 - AWS 는에서 AWS 서비스를 실행하는 인프라를 보호할 책임이 있습니다 AWS 클라우드. AWS 또한는 안전하게 사용할 수 있는 서비스를 제공합니다. 타사 감사자는 AWS 규정 준수 프로그램 일환으로 보안의 효과를 정기적으로 테스트하고 확인합니다. AWS 병렬 컴퓨팅 서비스에 적용되는 규정 준수 프로그램에 대한 자세한 내용은 규정 준수 프로그램 AWS 제공 범위 내 서비스규정 준수 프로그램 .
- 클라우드의 보안 - 사용자의 책임은 사용하는 AWS 서비스에 따라 결정됩니다. 또한 귀하는 귀사의 데이터 민감도, 귀사의 요구 사항, 관련 법률 및 규정을 비롯한 기타 요소에 대해서도 책임이 있습니다.

이 설명서는 AWS PCS를 사용할 때 공동 책임 모델을 적용하는 방법을 이해하는 데 도움이 됩니다. 다음 주제에서는 보안 및 규정 준수 목적에 맞게 AWS PCS를 구성하는 방법을 보여줍니다. 또한 AWS PCS 리소스를 모니터링하고 보호하는 데 도움이 되는 다른 AWS 서비스를 사용하는 방법을 알아봅니다.

주제

AWS 병렬 컴퓨팅 서비스의 데이터 보호

AWS 공동 책임 모델 AWS 병렬 컴퓨팅 서비스의 데이터 보호에 적용됩니다. 이 모델에 설명된 대로 AWS 는 모든를 실행하는 글로벌 인프라를 보호할 책임이 있습니다 AWS 클라우드. 사용자는 이 인프라에 호스팅되는 콘텐츠에 대한 통제 권한을 유지할 책임이 있습니다. 사용하는 AWS 서비스 의 보안 구성과 관리 태스크에 대한 책임도 사용자에게 있습니다. 데이터 프라이버시에 관한 자세한 내용은 데이터 프라이버시 FAQ를 참조하세요. 유럽의 데이터 보호에 대한 자세한 내용은 AWS 보안 블로그의 AWS 공동 책임 모델 및 GDPR 블로그 게시물을 참조하세요.

데이터 보호를 위해 자격 증명을 보호하고 AWS 계정 AWS IAM Identity Center 또는 AWS Identity and Access Management (IAM)를 사용하여 개별 사용자를 설정하는 것이 좋습니다. 이렇게 하면 개별 사용자에게 자신의 직무를 충실히 이행하는 데 필요한 권한만 부여됩니다. 또한 다음과 같은 방법으로 데이터를 보호하는 것이 좋습니다.

- 각 계정에 다중 인증(MFA)을 사용합니다.
- SSL/TLS를 사용하여 AWS 리소스와 통신합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- 를 사용하여 API 및 사용자 활동 로깅을 설정합니다 AWS CloudTrail. CloudTrail 추적을 사용하여 AWS 활동을 캡처하는 방법에 대한 자세한 내용은 AWS CloudTrail 사용 설명서의 CloudTrail 추적 작업을 참조하세요.
- 내부의 모든 기본 보안 제어와 함께 AWS 암호화 솔루션을 사용합니다 AWS 서비스.
- Amazon S3에 저장된 민감한 데이터를 검색하고 보호하는 데 도움이 되는 Amazon Macie와 같은 고급 관리형 보안 서비스를 사용합니다.
- 명령줄 인터페이스 또는 API를 AWS 통해에 액세스할 때 FIPS 140-3 검증 암호화 모듈이 필요한 경우 FIPS 엔드포인트를 사용합니다. 사용 가능한 FIPS 엔드포인트에 대한 자세한 내용은 연방 정보 처리 표준(FIPS) 140-3을 참조하세요.

고객의 이메일 주소와 같은 기밀 정보나 중요한 정보는 태그나 이름 필드와 같은 자유 형식 텍스트 필드에 입력하지 않는 것이 좋습니다. 여기에는 콘솔 AWS CLI, API 또는 AWS SDKs를 사용하여 AWS PCS 또는 기타 AWS 서비스 로 작업하는 경우가 포함됩니다. 이름에 사용되는 태그 또는 자유 형식 텍스트 필드에 입력하는 모든 데이터는 청구 또는 진단 로그에 사용될 수 있습니다. 외부 서버에 URL을 제공할 때 해당 서버에 대한 요청을 검증하기 위해 보안 인증 정보를 URL에 포함시켜서는 안 됩니다.

저장된 데이터 암호화

암호화는,AWS ,PCS API 또는 AWS SDKs를 사용하여 AWS 병렬 컴퓨팅 서비스(AWS Management Console AWS CLI AWS PCS) 클러스터를 생성할 때 기본적으로 저장 데이터에 대해 활성화됩니다.

AWS PCS는 AWS 소유 KMS 키를 사용하여 저장 데이터를 암호화합니다. 자세한 내용은 AWS KMS 개발자 안내서의 고객 키 및 AWS 키를 참조하세요. 고객 관리형 키를 사용할 수도 있습니다. 자세한 내용은 AWS PCS에서 암호화된 EBS 볼륨과 함께 사용하는 데 필요한 KMS 키 정책 단원을 참조하십시오.

클러스터 보안 암호는에 저장 AWS Secrets Manager 되고 Secrets Manager 관리형 KMS 키로 암호화됩니다. 자세한 내용은 AWS PCS에서 클러스터 보안 암호 작업 단원을 참조하십시오.

AWS PCS 클러스터에서 다음 데이터는 유휴 상태입니다.

- 스케줄러 상태 - 클러스터에서 실행 중인 작업 및 프로비저닝된 노드에 대한 데이터가 포함됩니다. 이는 Slurm이에 `StateSaveLocation` 정의된에 유지하는 데이터입니다`slurm.conf`. 자세한 내용은 Slurm 설명서의 StateSaveLocation에 대한 설명을 참조하세요. AWS PCS는 작업이 완료된 후 작업 데이터를 삭제합니다.
- 스케줄러 인증 보안 암호 - AWS PCS는 이를 사용하여 클러스터의 모든 스케줄러 통신을 인증합니다.

스케줄러 상태 정보의 경우 AWS PCS는 데이터 및 메타데이터를 파일 시스템에 쓰기 전에 자동으로 암호화합니다. 암호화된 파일 시스템은 저장 데이터에 업계 표준 AES-256 암호화 알고리즘을 사용합니다.

전송 중 암호화

AWS PCS API에 대한 연결은 AWS Command Line Interface (AWS CLI) 또는 AWS SDKs. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 AWS API 요청 서명을 참조하세요. 연결에 사용하는 보안 자격 증명에 대한 IAM 정책을 사용하여 API를 통한 액세스 제어를 AWS 관리합니다.

AWS PCS는 TLS를 사용하여 다른 AWS 서비스에 연결합니다.

Slurm 클러스터 내에서 스케줄러는 모든 스케줄러 통신에 대한 `auth/slurm` 인증을 제공하는 인증 플러그인으로 구성됩니다. Slurm은 통신을 위해 애플리케이션 수준에서 암호화를 제공하지 않습니다. 클러스터 인스턴스 간에 흐르는 모든 데이터는 EC2 VPC에 로컬로 유지되므로 해당 인스턴스가 전송 중 암호화를 지원하는 경우 VPC 암호화가 적용됩니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 전송 중 암호화를 참조하세요. 통신은 계정의 클러스터 노드에 있는 컨트롤러(서비스 계정에 프로비저닝됨) 간에 암호화됩니다.

키 관리

AWS PCS는 AWS 소유 KMS 키를 사용하여 데이터를 암호화합니다. 자세한 내용은 AWS KMS 개발자 안내서의 고객 키 및 AWS 키를 참조하세요. 고객 관리형 키를 사용할 수도 있습니다. 자세한 내용은 AWS PCS에서 암호화된 EBS 볼륨과 함께 사용하는 데 필요한 KMS 키 정책 단원을 참조하십시오.

클러스터 보안 암호는에 저장 AWS Secrets Manager 되고 Secrets Manager 관리형 KMS 키로 암호화됩니다. 자세한 내용은 AWS PCS에서 클러스터 보안 암호 작업 단원을 참조하십시오.

인터네트워크 트래픽 개인 정보 보호

AWS 클러스터의 PCS 컴퓨팅 리소스는 고객 계정의 VPC 1개 내에 있습니다. 따라서 클러스터 내의 모든 내부 AWS PCS 서비스 트래픽은 AWS 네트워크 내에 유지되며 인터넷을 통과하지 않습니다. 사용자와 AWS PCS 노드 간의 통신은 인터넷을 통해 이동할 수 있으므로 SSH 또는 Systems Manager를 사용하여 노드에 연결하는 것이 좋습니다. 자세한 내용은 란 무엇입니까 AWS Systems Manager?를 참조하세요. AWS Systems Manager 사용 설명서의 .

다음 서비스를 사용하여 온프레미스 네트워크를에 연결할 수도 있습니다 AWS.

- AWS Site-to-Site VPN. 자세한 내용은 란 무엇입니까 AWS Site-to-Site VPN?를 참조하세요. AWS Site-to-Site VPN 사용 설명서의 .
- AWS Direct Connect. 자세한 내용은 란 무엇입니까 AWS Direct Connect?를 참조하세요. AWS Direct Connect 사용 설명서의 .

AWS PCS API에 액세스하여 서비스에 대한 관리 작업을 수행합니다. 사용자와 사용자는 Slurm 엔드포인트 포트에 액세스하여 스케줄러와 직접 상호 작용합니다.

API 트래픽 암호화

AWS PCS API에 액세스하려면 클라이언트가 TLS(전송 계층 보안) 1.2 이상을 지원해야 합니다. TLS 1.2는 필수이며 TLS 1.3을 권장합니다. 클라이언트는 DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Diffie-Hellman Ephemeral)와 같은 PFS(전달 완전 보안)가 포함된 암호 제품군도 지원해야 합니다. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다. 또한 요청은 액세스 키 ID 및 IAM 위탁자와 관련된 시크릿 액세스 키를 사용하여 서명해야 합니다. AWS Security Token Service (AWS STS)를 사용하여 요청에 서명하기 위한 임시 보안 자격 증명을 생성할 수도 있습니다.

데이터 트래픽 암호화

전송 중 데이터 암호화는 스케줄러 엔드포인트에 액세스하는 지원되는 EC2 인스턴스에서 활성화되고 내에서 ComputeNodeGroup 인스턴스 간에 활성화됩니다 AWS 클라우드. 자세한 내용은 전송 중 암호화 단원을 참조하십시오.

AWS PCS에서 암호화된 EBS 볼륨과 함께 사용하는 데 필요한 KMS 키 정책

AWS PCS는 서비스 연결 역할을 사용하여 다른 사용자에게 권한을 위임합니다 AWS 서비스. AWS PCS 서비스 연결 역할은 미리 정의되어 있으며 AWS PCS가 AWS 서비스 사용자를 대신하여 다른를 호출하는 데 필요한 권한을 포함합니다. 사전 정의된 권한에는 고객 관리형 키가 AWS 관리형 키 아닌에 대한 액세스도 포함됩니다.

이 주제에서는 Amazon EBS 암호화를 위한 고객 관리형 키를 지정할 때 인스턴스를 시작하는 데 필요한 키 정책을 설정하는 방법을 설명합니다.

> **Note**
>
> AWS PCS는 계정에서 암호화된 볼륨을 보호하기 위해 기본값을 사용하는 AWS 관리형 키 데 추가 권한이 필요하지 않습니다.

내용

개요

AWS PCS가 인스턴스를 시작할 때 Amazon EBS 암호화 AWS KMS keys 에 다음을 사용할 수 있습니다.

- AWS 관리형 키 - Amazon EBS에서 생성, 소유 및 관리하는 계정의 암호화 키입니다. 이 키는 새 계정을 위한 기본 암호화 키입니다. Amazon EBS는 고객 관리형 키를 지정하지 않는 한 암호화 AWS 관리형 키 에를 사용합니다.

- 고객 관리형 키 - 사용자가 생성, 소유 및 관리하는 사용자 지정 암호화 키입니다. 자세한 내용은 AWS Key Management Service 개발자 안내서의 KMS 키 생성을 참조하세요.

> **Note**
>
> 키는 대칭이어야 합니다. Amazon EBS는 비대칭 고객 관리형 키를 지원하지 않습니다.

암호화된 스냅샷 또는 암호화된 볼륨을 지정하는 시작 템플릿을 생성하거나 기본적으로 암호화를 활성화하도록 선택할 때 고객 관리형 키를 구성합니다.

키 정책 구성

KMS 키에는 AWS PCS가 고객 관리형 키로 암호화된 Amazon EBS 볼륨으로 인스턴스를 시작할 수 있도록 허용하는 키 정책이 있어야 합니다.

이 페이지의 예제를 사용하여 고객 관리형 키에 대한 액세스 권한을 AWS PCS에 부여하도록 키 정책을 구성합니다. 키를 생성할 때 또는 나중에 고객 관리형 키의 키 정책을 수정할 수 있습니다.

키 정책에는 다음 문이 있어야 합니다.

- `Principal` 요소에 지정된 IAM 자격 증명이 고객 관리형 키를 직접 사용하도록 허용하는 문입니다. 여기에는 키에 대해 AWS KMS `Encrypt`, `Decrypt`, `ReEncrypt*GenerateDataKey*`, 및 `DescribeKey` 작업을 수행할 수 있는 권한이 포함됩니다.
- `Principal` 요소에 지정된 IAM 자격 증명이 `CreateGrant` 작업을 사용하여 AWS KMS 또는 다른 보안 주체와 통합된에 자체 권한의 하위 집합을 위임 AWS 서비스 하는 권한 부여를 생성하도록 허용하는 문입니다. 이를 통해 사용자는 키를 사용하여 사용자 대신 암호화된 리소스를 생성할 수 있습니다.

키 정책에 새 정책 설명을 추가할 때 정책의 기존 문을 변경하지 마십시오.

자세한 내용은 다음을 참조하세요.

- AWS CLI 명령 참조의 create-key
- AWS CLI 명령 참조의 put-key-policy
- AWS Key Management Service 개발자 안내서에서 키 ID 및 키 ARN 찾기
- AWS PCS에 대한 서비스 연결 역할
- Amazon EBS 사용 설명서의 Amazon EBS 암호화

- AWS Key Management Service 개발자 안내서의 AWS Key Management Service

예 1: 고객 관리형 키에 대한 액세스를 허용하는 키 정책 섹션

고객 관리형 키의 키 정책에 다음 정책 설명을 추가합니다. 예제 ARN을 AWSServiceRoleForPCS 서비스 연결 역할의 ARN으로 바꿉니다. 이 예제 정책은 AWS PCS 서비스 연결 역할(AWSServiceRoleForPCS)에 고객 관리형 키를 사용할 수 있는 권한을 부여합니다.

```
{
   "Sid": "Allow service-linked role use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::account-id:role/aws-service-role/pcs.amazonaws.com/
AWSServiceRoleForPCS"
       ]
   },
   "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
       "kms:DescribeKey"
   ],
   "Resource": "*"
}
```

```
{
   "Sid": "Allow attachment of persistent resources",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::account-id:role/aws-service-role/pcs.amazonaws.com/
AWSServiceRoleForPCS"
       ]
   },
   "Action": [
       "kms:CreateGrant"
   ],
   "Resource": "*",
   "Condition": {
       "Bool": {
```

```
            "kms:GrantIsForAWSResource": true
        }
    }
}
```

예 2: 고객 관리형 키에 대한 교차 계정 액세스를 허용하는 키 정책 섹션

AWS PCS 클러스터와 다른 계정에 고객 관리형 키를 생성하는 경우 키 정책과 함께 권한 부여를 사용하여 키에 대한 교차 계정 액세스를 허용해야 합니다.

키에 대한 액세스 권한을 부여하려면

1. 고객 관리형 키의 키 정책에 다음 정책 설명을 추가합니다. 예제 ARN을 다른 계정의 ARN으로 바꿉니다. *111122223333*을 AWS PCS 클러스터를 생성 AWS 계정 하려는의 실제 계정 ID로 바꿉니다. 이렇게 하면 지정된 계정의 IAM 사용자 또는 역할에게 다음 CLI 명령을 사용하여 키에 대한 권한 부여를 생성할 수 있는 권한을 부여할 수 있습니다. 기본적으로 사용자는 키에 액세스할 수 없습니다.

```
{.
   "Sid": "Allow external account 111122223333 use of the customer managed key",
   "Effect": "Allow",
   "Principal": {
       "AWS": [
           "arn:aws:iam::111122223333:root"
       ]
   },
   "Action": [
       "kms:Encrypt",
       "kms:Decrypt",
       "kms:ReEncrypt*",
       "kms:GenerateDataKey*",
       "kms:DescribeKey"
   ],
   "Resource": "*"
}
```

```
{
   "Sid": "Allow attachment of persistent resources in external
 account 111122223333",
   "Effect": "Allow",
   "Principal": {
```

```
        "AWS": [
            "arn:aws:iam::111122223333:root"
        ]
    },
    "Action": [
        "kms:CreateGrant"
    ],
    "Resource": "*"
}
```

2. AWS PCS 클러스터를 생성하려는 계정에서 관련 권한을 AWS PCS 서비스 연결 역할에 위임하는 권한 부여를 생성합니다. 의 값은 서비스 연결 역할의 ARN`grantee-principal`입니다. 의 값은 키의 ARN`key-id`입니다.

   다음 예제 create-grant CLI 명령은 계정 *111122223333*`AWSServiceRoleForPCS`의 라는 서비스 연결 역할에 계정 *444455556666*의 고객 관리형 키를 사용할 수 있는 권한을 부여합니다.

```
aws kms create-grant \
  --region us-west-2 \
  --key-id arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d \
  --grantee-principal arn:aws:iam::111122223333:role/aws-service-role/
pcs.amazonaws.com/AWSServiceRoleForPCS \
  --operations "Encrypt" "Decrypt" "ReEncryptFrom" "ReEncryptTo" "GenerateDataKey"
 "GenerateDataKeyWithoutPlaintext" "DescribeKey" "CreateGrant"
```

   > **Note**
   >
   > 요청을 하는 사용자에게 `kms:CreateGrant` 작업을 사용할 수 있는 권한이 있어야 합니다.

   다음 예제 IAM 정책에서는 계정 *111122223333*의 IAM 자격 증명(사용자 또는 역할)이 계정 *444455556666*의 고객 관리형 키에 대한 권한 부여를 생성하도록 허용합니다.

   JSON

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

```
    {
      "Sid": "AllowCreationOfGrantForTheKMSKeyinExternalAccount444455556666",
      "Effect": "Allow",
      "Action": "kms:CreateGrant",
      "Resource": "arn:aws:kms:us-
west-2:444455556666:key/1a2b3c4d-5e6f-1a2b-3c4d-5e6f1a2b3c4d"
    }
  ]
}
```

다른 AWS 계정에서 KMS 키에 대한 권한 부여 생성에 대한 자세한 정보는 AWS Key Management Service 개발자 안내서의 AWS KMS의 권한 부여를 참조하세요.

> **Important**
>
> 피부여자 보안 주체로 지정된 서비스 연결 역할 이름은 기존 역할의 이름이어야 합니다. 권한 부여를 생성한 후 권한 부여가 AWS PCS가 지정된 KMS 키를 사용하도록 허용하려면 서비스 연결 역할을 삭제하고 다시 생성하지 마십시오.

AWS KMS 콘솔에서 키 정책 편집

이전 섹션의 예에서는 키 정책을 변경하는 유일한 방법인 키 정책에 명령문을 추가하는 방법을 보여줍니다. 키 정책을 변경하는 가장 쉬운 방법은 키 정책에 콘솔 AWS KMS 의 기본 보기를 사용하고 IAM 자격 증명(사용자 또는 역할)을 적절한 키 정책에 대한 키 사용자 중 하나로 만드는 것입니다. 자세한 내용은 AWS Key Management Service 개발자 안내서의 AWS Management Console 기본 보기 사용을 참조하세요.

> **Warning**
>
> 콘솔의 기본 보기 정책 설명에는 고객 관리형 키에 대한 작업을 수행할 AWS KMS Revoke 수 있는 권한이 포함됩니다. 계정의 고객 관리형 키에 대한 AWS 계정 액세스 권한을 부여하는 권한 부여를 취소하면의 사용자는 암호화된 데이터와 키에 대한 액세스 권한을 AWS 계정 잃게 됩니다.

인터페이스 엔드포인트를 AWS Parallel Computing Service 사용한 액세스(AWS PrivateLink)

AWS PrivateLink 를 사용하여 VPC와 AWS Parallel Computing Service () 간에 프라이빗 연결을 생성할 수 있습니다AWS PCS. 인터넷 게이트웨이, NAT 디바이스, VPN 연결 또는 Direct Connect 연결을 사용하지 않고 VPC에 있는 AWS PCS 것처럼에 액세스할 수 있습니다. VPC의 인스턴스에서 AWS PCS API에 액세스하는 데는 퍼블릭 IP 주소가 필요하지 않습니다.

AWS PrivateLink에서 제공되는 인터페이스 엔드포인트를 생성하여 이 프라이빗 연결을 설정합니다. 인터페이스 엔드포인트에 대해 사용 설정하는 각 서브넷에서 엔드포인트 네트워크 인터페이스를 생성합니다. 이는 AWS PCS로 향하는 트래픽의 진입점 역할을 하는 요청자 관리형 네트워크 인터페이스입니다.

자세한 내용은 AWS PrivateLink 가이드의를 AWS 서비스 통한 액세스를 AWS PrivateLink 참조하세요.

에 대한 고려 사항 AWS PCS

에 대한 인터페이스 엔드포인트를 설정하기 전에 AWS PrivateLink 가이드의 인터페이스 VPC 엔드포인트를 사용하여 AWS 서비스 액세스를 AWS PCS검토합니다.

AWS PCS 는 인터페이스 엔드포인트를 통해 모든 API 작업을 호출할 수 있도록 지원합니다.

VPC에 직접 인터넷 액세스 권한이 없는 경우 컴퓨팅 노드 그룹 인스턴스가 `RegisterComputeNodeGroupInstance` API 작업을 호출 AWS PCS 할 수 있도록 VPC 엔드포인트를 구성해야 합니다.

에 대한 인터페이스 엔드포인트 생성 AWS PCS

Amazon VPC 콘솔 또는 AWS Command Line Interface ()를 AWS PCS 사용하여 용 인터페이스 엔드포인트를 생성할 수 있습니다AWS CLI. 자세한 내용은 AWS PrivateLink 안내서의 인터페이스 엔드포인트 생성을 참조하세요.

다음 서비스 이름을 AWS PCS 사용하여 용 인터페이스 엔드포인트를 생성합니다.

```
com.amazonaws.region.pcs
```

##을와 같이 엔드포인트 AWS 리전 를 생성할의 ID로 바꿉니다us-east-1.

인터페이스 엔드포인트에 프라이빗 DNS를 사용하도록 설정하는 경우, 리전에 대한 기본 DNS 이름(예: AWS PCS)을 사용하여 에 API 요청을 할 수 있습니다. 예를 들어 `pcs.us-east-1.amazonaws.com`입니다.

엔드포인트의 엔드포인트 정책 생성

엔드포인트 정책은 인터페이스 엔드포인트에 연결할 수 있는 IAM 리소스입니다. 기본 엔드포인트 정책은 인터페이스 엔드포인트를 AWS PCS 통해에 대한 전체 액세스를 허용합니다. VPC AWS PCS 에서에 허용되는 액세스를 제어하려면 인터페이스 엔드포인트에 사용자 지정 엔드포인트 정책을 연결합니다.

엔드포인트 정책은 다음 정보를 지정합니다.

- 작업을 수행할 수 있는 위탁자(AWS 계정, IAM 사용자, IAM 역할)
- 수행할 수 있는 작업
- 작업을 수행할 수 있는 리소스.

자세한 내용은 AWS PrivateLink 안내서의 엔드포인트 정책을 사용하여 서비스에 대한 액세스 제어를 참조하세요.

예: AWS PCS 작업에 대한 VPC 엔드포인트 정책

다음은 사용자 지정 엔드포인트 정책의 예입니다. 이 정책을 인터페이스 엔드포인트에 연결하면 지정된 cluster*-id*를 사용하여 모든 보안 주체에 대해 나열된 AWS PCS 작업에 대한 액세스 권한을 클러스터에 부여합니다. *region*을와 같은 클러스터의 AWS 리전 ID로 바꿉니다`us-east-1`. *account-id*를 클러스터 AWS 계정 번호로 바꿉니다.

```
{
    "Statement": [
            {
                "Action": [
                "pcs:CreateCluster",
                "pcs:ListClusters",
                "pcs:DeleteCluster",
                "pcs:GetCluster",
                ],
                "Effect": "Allow",
                "Principal": "*",
                "Resource": [
                    "arn:aws:pcs:region:account-id:cluster/cluster-id*"
```

```
                ]
            }
        ]
}
```

AWS 병렬 컴퓨팅 서비스를 위한 ID 및 액세스 관리

AWS Identity and Access Management (IAM)는 관리자가 AWS 리소스에 대한 액세스를 안전하게 제어하는 데 도움이 AWS 서비스 되는 입니다. IAM 관리자는 누가 AWS PCS 리소스를 사용할 수 있는 인증(로그인) 및 권한(권한 있음)을 받을 수 있는지 제어합니다. IAM은 추가 비용 없이 사용할 수 AWS 서비스 있는 입니다.

주제

대상

AWS Identity and Access Management (IAM)를 사용하는 방법은 역할에 따라 다릅니다.

- 서비스 사용자 - 기능에 액세스할 수 없는 경우 관리자에게 권한 요청(참조AWS 병렬 컴퓨팅 서비스 자격 증명 및 액세스 문제 해결)
- 서비스 관리자 - 사용자 액세스 결정 및 권한 요청 제출(AWS 병렬 컴퓨팅 서비스가 IAM과 작동하는 방식 참조)
- IAM 관리자 - 액세스를 관리하기 위한 정책 작성(AWS 병렬 컴퓨팅 서비스에 대한 자격 증명 기반 정책 예제 참조)

ID를 통한 인증

인증은 자격 증명 자격 증명을 AWS 사용하여에 로그인하는 방법입니다. AWS 계정 루트 사용자, IAM 사용자 또는 IAM 역할을 수임하여 인증되어야 합니다.

AWS IAM Identity Center (IAM Identity Center), Single Sign-On 인증 또는 Google/Facebook 자격 증명과 같은 자격 증명 소스의 자격 증명을 사용하여 페더레이션 자격 증명으로 로그인할 수 있습니다. 로그인하는 방법에 대한 자세한 내용은 AWS 로그인 사용 설명서의 AWS 계정에 로그인하는 방법 섹션을 참조하세요.

프로그래밍 방식 액세스를 위해는 요청에 암호화 방식으로 서명할 수 있는 SDK 및 CLI를 AWS 제공합니다. 자세한 내용은 IAM 사용 설명서의 API 요청용AWS Signature Version 4 섹션을 참조하세요.

AWS 계정 루트 사용자

를 생성할 때 모든 AWS 서비스 및 리소스에 대한 완전한 액세스 권한이 있는 AWS 계정 theroot 사용자라는 하나의 로그인 자격 증명으로 AWS 계정시작합니다. 일상적인 태스크에 루트 사용자를 사용하지 않을 것을 강력히 권장합니다. 루트 사용자 자격 증명이 필요한 작업은 IAM 사용 설명서의 루트 사용자 자격 증명이 필요한 작업 섹션을 참조하세요.

페더레이션 ID

가장 좋은 방법은 인간 사용자에게 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 서비스 사용하여에 액세스하도록 요구하는 것입니다.

페더레이션 자격 증명은 엔터프라이즈 디렉터리, 웹 자격 증명 공급자 또는 자격 증명 소스의 자격 증명을 AWS 서비스 사용하여 Directory Service 에 액세스하는 사용자입니다. 페더레이션 ID는 임시 자격 증명을 제공하는 역할을 수임합니다.

중앙 집중식 액세스 관리를 위해 AWS IAM Identity Center를 추천합니다. 자세한 정보는 AWS IAM Identity Center 사용 설명서의 What is IAM Identity Center?를 참조하세요.

IAM 사용자 및 그룹

IAM 사용자는 단일 개인 또는 애플리케이션에 대한 특정 권한을 가진 ID입니다. 장기 자격 증명이 있는 IAM 사용자 대신 임시 자격 증명을 사용하는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 자격 증명 공급자와의 페더레이션을 사용하여 임시 자격 증명을 AWS 사용하여에 액세스하도록 인간 사용자에게 요구하기를 참조하세요.

IAM 그룹은 IAM 사용자 모음을 지정하고 대규모 사용자 집합에 대한 관리 권한을 더 쉽게 만듭니다. 자세한 내용은 IAM 사용 설명서의 IAM 사용자 사용 사례 섹션을 참조하세요.

IAM 역할

IAM 역할은 임시 자격 증명을 제공하는 특정 권한이 있는 자격 증명입니다. 사용자에서 IAM 역할(콘솔)로 전환하거나 또는 API 작업을 호출하여 역할을 수임할 수 있습니다. AWS CLI AWS 자세한 내용은 IAM 사용 설명서의 역할 수임 방법을 참조하세요.

IAM 역할은 페더레이션 사용자 액세스, 임시 IAM 사용자 권한, 교차 계정 액세스, 교차 서비스 액세스 및 Amazon EC2에서 실행되는 애플리케이션에 유용합니다. 자세한 내용은 IAM 사용 설명서의 교차 계정 리소스 액세스를 참조하세요.

정책을 사용하여 액세스 관리

정책을 AWS 생성하고 자격 증명 또는 리소스에 연결하여 AWS 에서 액세스를 제어합니다. 정책은 자격 증명 또는 리소스와 연결될 때 권한을 정의합니다.는 보안 주체가 요청할 때 이러한 정책을 AWS 평가합니다. 대부분의 정책은에 JSON 문서 AWS 로 저장됩니다. JSON 정책 문서에 대한 자세한 내용은 IAM 사용 설명서의 JSON 정책 개요 섹션을 참조하세요.

정책을 사용하여 관리자는 어떤 보안 주체가 어떤 리소스에 대해 어떤 조건에서 작업을 수행할 수 있는지 정의하여 누가 무엇을 액세스할 수 있는지 지정합니다.

기본적으로 사용자 및 역할에는 어떠한 권한도 없습니다. IAM 관리자는 IAM 정책을 생성하고 사용자가 수임할 수 있는 역할에 추가합니다. IAM 정책은 작업을 수행하기 위해 사용하는 방법과 관계없이 작업에 대한 권한을 정의합니다.

ID 기반 정책

ID 기반 정책은 ID(사용자, 사용자 그룹 또는 역할)에 연결하는 JSON 권한 정책 문서입니다. 이러한 정책은 자격 증명이 수행할 수 있는 작업, 대상 리소스 및 이에 관한 조건을 제어합니다. ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

ID 기반 정책은 인라인 정책(단일 ID에 직접 포함) 또는 관리형 정책(여러 ID에 연결된 독립 실행형 정책)일 수 있습니다. 관리형 정책 또는 인라인 정책을 선택하는 방법을 알아보려면 IAM 사용 설명서의 관리형 정책 및 인라인 정책 중에서 선택 섹션을 참조하세요.

리소스 기반 정책

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 예를 들어 IAM 역할 신뢰 정책 및 Amazon S3 버킷 정책이 있습니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한

정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다.

리소스 기반 정책은 해당 서비스에 있는 인라인 정책입니다. 리소스 기반 정책에서는 IAM의 AWS 관리형 정책을 사용할 수 없습니다.

기타 정책 유형

AWS 는 보다 일반적인 정책 유형에서 부여한 최대 권한을 설정할 수 있는 추가 정책 유형을 지원합니다.

- 권한 경계 - ID 기반 정책에서 IAM 엔터티에 부여할 수 있는 최대 권한을 설정합니다. 자세한 정보는 IAM 사용 설명서의 IAM 엔터티의 권한 범위를 참조하세요.
- 서비스 제어 정책(SCP) - AWS Organizations내 조직 또는 조직 단위에 대한 최대 권한을 지정합니다. 자세한 내용은AWS Organizations 사용 설명서의 서비스 제어 정책을 참조하세요.
- 리소스 제어 정책(RCP) – 계정의 리소스에 사용할 수 있는 최대 권한을 설정합니다. 자세한 내용은 AWS Organizations 사용 설명서의 리소스 제어 정책(RCP)을 참조하세요.
- 세션 정책 – 역할 또는 페더레이션 사용자에 대해 임시 세션을 프로그래밍 방식으로 생성할 때 파라미터로 전달하는 고급 정책입니다. 자세한 내용은 IAM 사용 설명서의 세션 정책을 참조하세요.

여러 정책 유형

여러 정책 유형이 요청에 적용되는 경우, 결과 권한은 이해하기가 더 복잡합니다. 에서 여러 정책 유형이 관련될 때 요청을 허용할지 여부를 AWS 결정하는 방법을 알아보려면 IAM 사용 설명서의 정책 평가 로직을 참조하세요.

AWS 병렬 컴퓨팅 서비스가 IAM과 작동하는 방식

IAM을 사용하여 AWS PCS에 대한 액세스를 관리하기 전에 AWS PCS에서 사용할 수 있는 IAM 기능에 대해 알아봅니다.

AWS 병렬 컴퓨팅 서비스와 함께 사용할 수 있는 IAM 기능

IAM 특성	AWS PCS 지원
자격 증명 기반 정책	예
리소스 기반 정책	아니요

IAM 특성	AWS PCS 지원
정책 작업	예
정책 리소스	예
정책 조건 키(서비스별)	예
ACL	아니요
ABAC(정책의 태그)	예
임시 보안 인증	예
엔터티 권한	예
서비스 역할	아니요
서비스 연결 역할	예

AWS PCS 및 기타 AWS 서비스가 대부분의 IAM 기능과 작동하는 방식을 개괄적으로 알아보려면 IAM 사용 설명서의 AWS IAM으로 작업하는 서비스를 참조하세요.

AWS PCS에 대한 자격 증명 기반 정책

ID 기반 정책 지원: 예

ID 기반 정책은 IAM 사용자, 사용자 그룹 또는 역할과 같은 ID에 연결할 수 있는 JSON 권한 정책 문서입니다. 이러한 정책은 사용자 및 역할이 어떤 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 제어합니다. 자격 증명 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서에서 고객 관리형 정책으로 사용자 지정 IAM 권한 정의를 참조하세요.

IAM ID 기반 정책을 사용하면 허용되거나 거부되는 작업과 리소스뿐 아니라 작업이 허용되거나 거부되는 조건을 지정할 수 있습니다. JSON 정책에서 사용할 수 있는 모든 요소에 대해 알아보려면 IAM 사용 설명서의 IAM JSON 정책 요소 참조를 참조하세요.

AWS PCS에 대한 자격 증명 기반 정책 예제

AWS PCS 자격 증명 기반 정책의 예를 보려면 섹션을 참조하세요AWS 병렬 컴퓨팅 서비스에 대한 자격 증명 기반 정책 예제.

AWS PCS 내 리소스 기반 정책

리소스 기반 정책 지원: 아니요

리소스 기반 정책은 리소스에 연결하는 JSON 정책 설명서입니다. 리소스 기반 정책의 예제는 IAM 역할 신뢰 정책과 Amazon S3 버킷 정책입니다. 리소스 기반 정책을 지원하는 서비스에서 서비스 관리자는 이러한 정책을 사용하여 특정 리소스에 대한 액세스를 통제할 수 있습니다. 정책이 연결된 리소스의 경우 정책은 지정된 보안 주체가 해당 리소스와 어떤 조건에서 어떤 작업을 수행할 수 있는지를 정의합니다. 리소스 기반 정책에서 보안 주체를 지정해야 합니다. 보안 주체에는 계정, 사용자, 역할, 페더레이션 사용자 또는가 포함될 수 있습니다 AWS 서비스.

교차 계정 액세스를 활성화하려는 경우, 전체 계정이나 다른 계정의 IAM 개체를 리소스 기반 정책의 보안 주체로 지정할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 IAM에서 교차 계정 리소스 액세스를 참조하세요.

AWS PCS에 대한 정책 작업

정책 작업 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

JSON 정책의 `Action`요소는 정책에서 액세스를 허용하거나 거부하는 데 사용할 수 있는 작업을 설명합니다. 연결된 작업을 수행할 수 있는 권한을 부여하기 위한 정책에 작업을 포함하세요.

AWS PCS 작업 목록을 보려면 서비스 승인 참조의 AWS 병렬 컴퓨팅 서비스에서 정의한 작업을 참조하세요.

AWS PCS의 정책 작업은 작업 앞에 다음 접두사를 사용합니다.

```
pcs
```

단일 문에서 여러 작업을 지정하려면 쉼표로 구분합니다.

```
"Action": [
      "pcs:action1",
      "pcs:action2"
         ]
```

AWS PCS에 대한 정책 리소스

정책 리소스 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Resource` JSON 정책 요소는 작업이 적용되는 하나 이상의 객체를 지정합니다. 모범 사례에 따라 Amazon 리소스 이름(ARN)을 사용하여 리소스를 지정합니다. 리소스 수준 권한을 지원하지 않는 작업의 경우, 와일드카드(*)를 사용하여 해당 문이 모든 리소스에 적용됨을 나타냅니다.

```
"Resource": "*"
```

AWS PCS 리소스 유형 및 해당 ARNs 목록을 보려면 서비스 승인 참조의 AWS 병렬 컴퓨팅 서비스에서 정의한 리소스를 참조하세요. 각 리소스의 ARN을 지정할 수 있는 작업을 알아보려면 AWS 병렬 컴퓨팅 서비스에서 정의한 작업을 참조하세요.

AWS PCS 자격 증명 기반 정책의 예를 보려면 섹션을 참조하세요AWS 병렬 컴퓨팅 서비스에 대한 자격 증명 기반 정책 예제.

AWS PCS에 사용되는 정책 조건 키

서비스별 정책 조건 키 지원: 예

관리자는 AWS JSON 정책을 사용하여 누가 무엇에 액세스할 수 있는지 지정할 수 있습니다. 즉, 어떤 보안 주체가 어떤 리소스와 어떤 조건에서 작업을 수행할 수 있는지를 지정할 수 있습니다.

`Condition` 요소는 정의된 기준에 따라 문이 실행되는 시기를 지정합니다. 같음(equals) 또는 미만(less than)과 같은 조건 연산자를 사용하여 정책의 조건을 요청의 값과 일치시키는 조건식을 생성할 수 있습니다. 모든 AWS 전역 조건 키를 보려면 IAM 사용 설명서의 AWS 전역 조건 컨텍스트 키를 참조하세요.

AWS PCS 조건 키 목록을 보려면 서비스 승인 참조의 AWS 병렬 컴퓨팅 서비스에 사용되는 조건 키를 참조하세요. 조건 키를 사용할 수 있는 작업과 리소스를 알아보려면 AWS 병렬 컴퓨팅 서비스에서 정의한 작업을 참조하세요.

AWS PCS 자격 증명 기반 정책의 예를 보려면 섹션을 참조하세요AWS 병렬 컴퓨팅 서비스에 대한 자격 증명 기반 정책 예제.

AWS PCSACLs

ACL 지원: 아니요

액세스 제어 목록(ACL)은 어떤 보안 주체(계정 멤버, 사용자 또는 역할)가 리소스에 액세스할 수 있는 권한을 가지고 있는지를 제어합니다. ACL은 JSON 정책 문서 형식을 사용하지 않지만 리소스 기반 정책과 유사합니다.

AWS PCS를 사용한 ABAC

ABAC 지원(정책의 태그): 예

속성 기반 액세스 제어(ABAC)는 태그라고 불리는 속성을 기반으로 권한을 정의하는 권한 부여 전략입니다. IAM 엔터티 및 AWS 리소스에 태그를 연결한 다음 보안 주체의 태그가 리소스의 태그와 일치할 때 작업을 허용하는 ABAC 정책을 설계할 수 있습니다.

태그에 근거하여 액세스를 제어하려면 `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`또는 `aws:TagKeys`조건 키를 사용하여 정책의 조건 요소에 태그 정보를 제공합니다.

서비스가 모든 리소스 유형에 대해 세 가지 조건 키를 모두 지원하는 경우, 값은 서비스에 대해 예입니다. 서비스가 일부 리소스 유형에 대해서만 세 가지 조건 키를 모두 지원하는 경우, 값은 부분적입니다.

ABAC에 대한 자세한 내용은 IAM 사용 설명서의 ABAC 권한 부여를 통한 권한 정의를 참조하세요. ABAC 설정 단계가 포함된 자습서를 보려면 IAM 사용 설명서의 속성 기반 액세스 제어(ABAC) 사용을 참조하세요.

AWS PCS에서 임시 자격 증명 사용

임시 자격 증명 지원: 예

임시 자격 증명은 AWS 리소스에 대한 단기 액세스를 제공하며 페더레이션을 사용하거나 역할을 전환할 때 자동으로 생성됩니다. 장기 액세스 키를 사용하는 대신 임시 자격 증명을 동적으로 생성하는 것이 AWS 좋습니다. 자세한 내용은 IAM 사용 설명서의 IAM의 임시 보안 자격 증명 및 IAM으로 작업하는AWS 서비스 섹션을 참조하세요.

AWS PCS에 대한 교차 서비스 보안 주체 권한

전달 액세스 세션(FAS) 지원: 예

전달 액세스 세션(FAS)은를 호출하는 보안 주체의 권한을 다운스트림 서비스에 AWS 서비스 대한 요청과 AWS 서비스함께 사용합니다. FAS 요청 시 정책 세부 정보는 전달 액세스 세션을 참조하세요.

AWS PCS에 대한 서비스 역할

서비스 역할 지원: 아니요

서비스 역할은 서비스가 사용자를 대신하여 작업을 수행하는 것으로 가정하는 IAM 역할입니다. IAM 관리자는 IAM 내에서 서비스 역할을 생성, 수정 및 삭제할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 AWS 서비스 AWS에 권한을 위임할 역할 생성을 참조하세요.

> **Warning**
>
> 서비스 역할에 대한 권한을 변경하면 AWS PCS 기능이 중단될 수 있습니다. AWS PCS가 관련 지침을 제공하는 경우에만 서비스 역할을 편집합니다.

AWS PCS에 대한 서비스 연결 역할

서비스 연결 역할 지원: 예

서비스 연결 역할은에 연결된 서비스 역할의 한 유형입니다 AWS 서비스. 서비스는 사용자를 대신하여 작업을 수행하기 위해 역할을 수임할 수 있습니다. 서비스 연결 역할은에 나타나 AWS 계정 며 서비스가 소유합니다. IAM 관리자는 서비스 연결 역할의 권한을 볼 수 있지만 편집은 할 수 없습니다.

AWS PCS 서비스 연결 역할 생성 또는 관리에 대한 자세한 내용은 섹션을 참조하세요AWS PCS에 대한 서비스 연결 역할.

AWS 병렬 컴퓨팅 서비스에 대한 자격 증명 기반 정책 예제

기본적으로 사용자 및 역할에는 AWS PCS 리소스를 생성하거나 수정할 수 있는 권한이 없습니다. 사용자에게 사용자가 필요한 리소스에서 작업을 수행할 권한을 부여하려면 IAM 관리자가 IAM 정책을 생성하면 됩니다.

이러한 예제 JSON 정책 문서를 사용하여 IAM ID 기반 정책을 생성하는 방법을 알아보려면 IAM 사용 설명서의 IAM 정책 생성(콘솔)을 참조하세요.

각 리소스 유형에 대한 ARNs 형식을 포함하여 AWS PCS에서 정의한 작업 및 리소스 유형에 대한 자세한 내용은 서비스 승인 참조의 AWS 병렬 컴퓨팅 서비스에 사용되는 작업, 리소스 및 조건 키를 참조하세요.

주제

- 정책 모범 사례
- AWS PCS 콘솔 사용
- 사용자가 자신의 고유한 권한을 볼 수 있도록 허용

정책 모범 사례

자격 증명 기반 정책에 따라 계정에서 사용자가 AWS PCS 리소스를 생성, 액세스 또는 삭제할 수 있는지 여부가 결정됩니다. 이 작업으로 인해 AWS 계정에 비용이 발생할 수 있습니다. ID 기반 정책을 생성하거나 편집할 때는 다음 지침과 권장 사항을 따르세요.

- AWS 관리형 정책을 시작하고 최소 권한으로 전환 - 사용자 및 워크로드에 권한 부여를 시작하려면 많은 일반적인 사용 사례에 대한 권한을 부여하는 AWS 관리형 정책을 사용합니다. 에서 사용할 수 있습니다 AWS 계정. 사용 사례에 맞는 AWS 고객 관리형 정책을 정의하여 권한을 추가로 줄이는 것이 좋습니다. 자세한 내용은 IAM 사용 설명서의 AWS 관리형 정책 또는 AWS 직무에 대한 관리형 정책을 참조하세요.
- 최소 권한 적용 – IAM 정책을 사용하여 권한을 설정하는 경우, 작업을 수행하는 데 필요한 권한만 부여합니다. 이렇게 하려면 최소 권한으로 알려진 특정 조건에서 특정 리소스에 대해 수행할 수 있는 작업을 정의합니다. IAM을 사용하여 권한을 적용하는 방법에 대한 자세한 정보는 IAM 사용 설명서에 있는 IAM의 정책 및 권한을 참조하세요.
- IAM 정책의 조건을 사용하여 액세스 추가 제한 – 정책에 조건을 추가하여 작업 및 리소스에 대한 액세스를 제한할 수 있습니다. 예를 들어, SSL을 사용하여 모든 요청을 전송해야 한다고 지정하는 정책 조건을 작성할 수 있습니다. AWS 서비스와 같은 특정를 통해 사용되는 경우 조건을 사용하여 서비스 작업에 대한 액세스 권한을 부여할 수도 있습니다 CloudFormation. 자세한 내용은 IAM 사용 설명서의 IAM JSON 정책 요소: 조건을 참조하세요.
- IAM Access Analyzer를 통해 IAM 정책을 확인하여 안전하고 기능적인 권한 보장 - IAM Access Analyzer에서는 IAM 정책 언어(JSON)와 모범 사례가 정책에서 준수되도록 새로운 및 기존 정책을 확인합니다. IAM Access Analyzer는 100개 이상의 정책 확인 항목과 실행 가능한 추천을 제공하여 안전하고 기능적인 정책을 작성하도록 돕습니다. 자세한 내용은 IAM 사용 설명서의 IAM Access Analyzer에서 정책 검증을 참조하세요.
- 다중 인증(MFA) 필요 -에서 IAM 사용자 또는 루트 사용자가 필요한 시나리오가 있는 경우 추가 보안을 위해 MFA를 AWS 계정켭니다. API 작업을 직접적으로 호출할 때 MFA가 필요하면 정책에 MFA 조건을 추가합니다. 자세한 내용은 IAM 사용 설명서의 MFA를 통한 보안 API 액세스를 참조하세요.

IAM의 모범 사례에 대한 자세한 내용은 IAM 사용 설명서의 IAM의 보안 모범 사례를 참조하세요.

AWS PCS 콘솔 사용

AWS 병렬 컴퓨팅 서비스 콘솔에 액세스하려면 최소 권한 집합이 있어야 합니다. 이러한 권한은에서 AWS PCS 리소스에 대한 세부 정보를 나열하고 볼 수 있도록 허용해야 합니다 AWS 계정. 최소 필수 권한보다 더 제한적인 ID 기반 정책을 생성하는 경우, 콘솔이 해당 정책에 연결된 엔티티(사용자 또는 역할)에 대해 의도대로 작동하지 않습니다.

AWS CLI 또는 AWS API만 호출하는 사용자에게는 최소 콘솔 권한을 허용할 필요가 없습니다. 대신, 수행하려는 API 작업과 일치하는 작업에만 액세스할 수 있도록 합니다.

AWS PCS 콘솔을 사용하는 데 필요한 최소 권한에 대한 자세한 내용은 섹션을 참조하세요AWS PCS에 대한 최소 권한.

사용자가 자신의 고유한 권한을 볼 수 있도록 허용

이 예제는 IAM 사용자가 자신의 사용자 ID에 연결된 인라인 및 관리형 정책을 볼 수 있도록 허용하는 정책을 생성하는 방법을 보여줍니다. 이 정책에는 콘솔에서 또는 AWS CLI 또는 AWS API를 사용하여 프로그래밍 방식으로이 작업을 완료할 수 있는 권한이 포함됩니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
```

```
}
```

AWS AWS 병렬 컴퓨팅 서비스에 대한 관리형 정책

AWS 관리형 정책은에서 생성하고 관리하는 독립 실행형 정책입니다 AWS. AWS 관리형 정책은 사용자, 그룹 및 역할에 권한 할당을 시작할 수 있도록 많은 일반적인 사용 사례에 대한 권한을 제공하도록 설계되었습니다.

AWS 관리형 정책은 모든 AWS 고객이 사용할 수 있으므로 특정 사용 사례에 대해 최소 권한을 부여하지 않을 수 있습니다. 사용 사례에 고유한 고객 관리형 정책을 정의하여 권한을 줄이는 것이 좋습니다.

AWS 관리형 정책에 정의된 권한은 변경할 수 없습니다. 가 관리형 정책에 정의된 권한을 AWS 업데이트하는 AWS 경우 업데이트는 정책이 연결된 모든 보안 주체 자격 증명(사용자, 그룹 및 역할)에 영향을 줍니다. AWS AWS 서비스 는 새가 시작되거나 기존 서비스에 새 API 작업을 사용할 수 있게 될 때 AWS 관리형 정책을 업데이트할 가능성이 높습니다.

자세한 내용은 IAM 사용자 가이드의 AWS 관리형 정책을 참조하세요.

AWS 관리형 정책: AWSPCSComputeNodePolicy

AWSPCSComputeNodePolicy를 IAM 엔터티에 연결할 수 있습니다. 이 정책을 지정된 AWS PCS 컴퓨팅 노드 IAM 역할에 연결하여 해당 역할을 사용하는 노드가 AWS PCS 클러스터에 연결하도록 허용할 수 있습니다.

AWS 콘솔을 사용하여 컴퓨팅 노드 그룹을 생성할 때 PCS는이 정책을 컴퓨팅 노드 그룹 역할에 연결합니다.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `pcs:RegisterComputeNodeGroupInstance` - AWS PCS 컴퓨팅 노드(EC2 인스턴스)가 AWS PCS 클러스터에 등록할 수 있도록 허용합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조의 AWSPCSComputeNodePolicy를 참조하세요.

AWS 관리형 정책: AWSPCSServiceRolePolicy

AWSPCSServiceRolePolicy를 IAM 엔터티에 연결할 수 없습니다. 이 정책은 AWS PCS가 사용자를 대신하여 작업을 수행할 수 있도록 서비스 연결 역할에 연결됩니다. 자세한 내용은 AWS PCS에 대한 서비스 연결 역할 단원을 참조하십시오.

권한 세부 정보

이 정책에는 다음 권한이 포함되어 있습니다.

- `ec2` - AWS PCS가 Amazon EC2 리소스를 생성하고 관리할 수 있도록 허용합니다.
- `iam` - AWS PCS가 Amazon EC2 플릿에 대한 서비스 연결 역할을 생성하고 해당 역할을 Amazon EC2에 전달할 수 있도록 허용합니다.
- `cloudwatch` - AWS PCS가 Amazon CloudWatch에 서비스 지표를 게시할 수 있도록 허용합니다.
- `secretsmanager` - AWS PCS가 AWS PCS 클러스터 리소스의 보안 암호를 관리할 수 있도록 허용합니다.

이 정책의 권한을 보려면 AWS 관리형 정책 참조의 AWSPCSServiceRolePolicy를 참조하세요.

AWS AWS 관리형 정책에 대한 PCS 업데이트

이 서비스가 이러한 변경 사항을 추적하기 시작한 이후 AWS 부터 PCS의 AWS 관리형 정책 업데이트에 대한 세부 정보를 봅니다. 이 페이지의 변경 사항에 대한 자동 알림을 받으려면 AWS PCS 문서 기록 페이지에서 RSS 피드를 구독하세요.

변경	설명	Date
AWSPCSServiceRolePolicy – 기존 정책에 대한 업데이트	AWS PCS는 예측 가능한 컴퓨팅 용량을 위해 용량 블록을 지원하는 새로운 권한을 추가했습니다. AWS PCS가 컴퓨팅 노드 그룹에 대한 용량 블록 예약을 검색하고 사용할 수 있는 `ec2:DescribeCapacityReservations` 권한을 추가했습니다.	2025년 9월 11일
AWSPCSComputeNodePolicy – 새 정책	AWS PCS는 AWS PCS 클러스터에 연결할 수 있는 권한을 AWS PCS 컴퓨팅 노드에 부여하는 새 정책을 추가했습니다. AWS PCS 콘솔에서 컴퓨팅 노드 그룹을 생성할 때 AWS	2025년 6월 23일

변경	설명	Date
	PCS는이 정책을 IAM 역할에 연결합니다.	
이 문서의 JSON 업데이트	를 포함하도록이 문서의 JSON을 수정했습니다`"arn:aws:ec2:*:*:spot-instances-request/*"` .	2024년 9월 5일
AWS PCS에서 변경 사항 추적 시작	AWS PCS가 AWS 관리형 정책에 대한 변경 사항 추적을 시작했습니다.	2024년 8월 28일

AWS PCS에 대한 서비스 연결 역할

AWS 병렬 컴퓨팅 서비스는 AWS Identity and Access Management (IAM) 서비스 연결 역할을 사용합니다. 서비스 연결 역할은 AWS PCS에 직접 연결된 고유한 유형의 IAM 역할입니다. 서비스 연결 역할은 AWS PCS에서 사전 정의하며 서비스가 사용자를 대신하여 다른 AWS 서비스를 호출하는 데 필요한 모든 권한을 포함합니다.

서비스 연결 역할을 사용하면 필요한 권한을 수동으로 추가할 필요가 없으므로 AWS PCS를 더 쉽게 설정할 수 있습니다. AWS PCS는 서비스 연결 역할의 권한을 정의하며, 달리 정의되지 않은 한 AWS PCS만 해당 역할을 수임할 수 있습니다. 정의된 권한에는 신뢰 정책과 권한 정책이 포함되며 이 권한 정책은 다른 IAM 엔티티에 연결할 수 없습니다.

먼저 관련 리소스를 삭제해야만 서비스 연결 역할을 삭제할 수 있습니다. 이렇게 하면 리소스에 대한 액세스 권한을 실수로 제거할 수 없기 때문에 AWS PCS 리소스가 보호됩니다.

서비스 연결 역할을 지원하는 다른 서비스에 대한 자세한 내용은 AWS IAM으로 작업하는 서비스를 참조하고 서비스 연결 역할 열에서 예인 서비스를 찾습니다. 해당 서비스에 대한 서비스 연결 역할 설명서를 보려면 예 링크를 선택합니다.

AWS PCS에 대한 서비스 연결 역할 권한

AWS PCS는 AWSServiceRoleForPCS라는 서비스 연결 역할을 사용합니다. - AWS PCS에 Amazon EC2 리소스를 관리할 수 있는 권한을 부여합니다.

AWSServiceRoleForPCS 서비스 연결 역할은 역할을 수임하기 위해 다음 서비스를 신뢰합니다.

- pcs.amazonaws.com

AWSPCSServiceRolePolicy라는 역할 권한 정책은 AWS PCS가 특정 리소스에 대한 작업을 완료하도록 허용합니다.

사용자, 그룹 또는 역할이 서비스 연결 역할을 생성, 편집 또는 삭제할 수 있도록 사용 권한을 구성해야 합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 권한을 참조하세요.

AWS PCS에 대한 서비스 연결 역할 생성

서비스 연결 역할을 수동으로 생성할 필요가 없습니다. 클러스터를 생성할 때 AWS PCS가 서비스 연결 역할을 생성합니다.

AWS PCS에 대한 서비스 연결 역할 편집

AWS PCS에서는 AWSServiceRoleForPCS 서비스 연결 역할을 편집할 수 없습니다. 서비스 연결 역할을 생성한 후에는 다양한 개체가 역할을 참조할 수 있기 때문에 역할 이름을 변경할 수 없습니다. 하지만 IAM을 사용하여 역할의 설명을 편집할 수 있습니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 편집을 참조하세요.

AWS PCS에 대한 서비스 연결 역할 삭제

서비스 연결 역할이 필요한 기능 또는 서비스가 더 이상 필요 없는 경우에는 해당 역할을 삭제하는 것이 좋습니다. 따라서 적극적으로 모니터링하거나 유지하지 않는 미사용 엔터티가 없도록 합니다. 단, 서비스 연결 역할에 대한 리소스를 먼저 정리해야 수동으로 삭제할 수 있습니다.

> **Note**
>
> 리소스를 삭제하려고 할 때 AWS PCS 서비스가 역할을 사용하는 경우 삭제에 실패할 수 있습니다. 이 문제가 발생하면 몇 분 기다렸다가 작업을 다시 시도하세요.

AWSServiceRoleForPCS에서 사용하는 AWS PCS 리소스를 제거하려면

AWSServiceRoleForPCS 서비스 연결 역할을 삭제하려면 모든 클러스터를 삭제해야 합니다. 자세한 내용은 클러스터 삭제를 참조하세요.

IAM을 사용하여 수동으로 서비스 연결 역할을 삭제하려면 다음을 수행하세요.

IAM 콘솔 AWS CLI, 또는 AWS API를 사용하여 AWSServiceRoleForPCS 서비스 연결 역할을 삭제합니다. 자세한 내용은 IAM 사용 설명서의 서비스 연결 역할 삭제를 참조하십시오.

AWS PCS 서비스 연결 역할에 지원되는 리전

AWS PCS는 서비스를 사용할 수 있는 모든 리전에서 서비스 연결 역할 사용을 지원합니다. 자세한 내용은 AWS 리전 및 엔드포인트 섹션을 참조하세요.

AWS PCS에 대한 Amazon EC2 스팟 역할

스팟을 구매 옵션으로 사용하는 AWS PCS 컴퓨팅 노드 그룹을 생성하려면에 AWSServiceRoleForEC2Spot 서비스 연결 역할도 있어야 합니다 AWS 계정. 다음 AWS CLI 명령을 사용하여 역할을 생성할 수 있습니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 서비스 연결 역할 생성 및 AWS 서비스에 권한을 위임할 역할 생성을 참조하세요.

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

> **Note**
>
> 에 AWS 계정 이미 AWSServiceRoleForEC2Spot IAM 역할이 있는 경우 다음 오류가 발생합니다.
>
> ```
> An error occurred (InvalidInput) when calling the CreateServiceLinkedRole
> operation: Service role name AWSServiceRoleForEC2Spot has been taken in this
> account, please try a different suffix.
> ```

AWS PCS에 대한 최소 권한

이 섹션에서는 IAM 자격 증명(사용자, 그룹 또는 역할)이 서비스를 사용하는 데 필요한 최소 IAM 권한을 설명합니다.

목차

- 서비스 관리자의 최소 권한

API 작업을 사용할 수 있는 최소 권한

API 작업	최소 권한	콘솔에 대한 추가 권한
CreateCluster	`ec2:CreateNetworkInterface, ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:GetSecurityGroupsForVpc, iam:CreateServiceLinkedRole, secretsmanager:CreateSecret, secretsmanager:TagResource, secretsmanager:RotateSecret, pcs:CreateCluster`	
ListClusters	`pcs:ListClusters`	
GetCluster	`pcs:GetCluster`	`ec2:DescribeSubnets`
DeleteCluster	`pcs:DeleteCluster`	
CreateComputeNodeGroup	`ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:DescribeLaunchTemplates, ec2:DescribeLaunchTemplateVersions,`	`iam:ListInstanceProfiles, ec2:DescribeImages, pcs:GetCluster`

API 작업	최소 권한	콘솔에 대한 추가 권한
	`ec2:DescribeInstanceTypes`, `ec2:DescribeInstanceTypeOfferings`, `ec2:RunInstances`, `ec2:CreateFleet`, `ec2:CreateTags`, `iam:PassRole`, `iam:GetInstanceProfile`, `pcs:CreateComputeNodeGroup`	
ListComputerNodeGroups	`pcs:ListComputeNodeGroups`	`pcs:GetCluster`
GetComputeNodeGroup	`pcs:GetComputeNodeGroup`	`ec2:DescribeSubnets`

API 작업	최소 권한	콘솔에 대한 추가 권한
UpdateComputeNodeGroup	`ec2:DescribeVpcs, ec2:DescribeSubnets, ec2:DescribeSecurityGroups, ec2:DescribeLaunchTemplates, ec2:DescribeLaunchTemplateVersions, ec2:DescribeInstanceTypes, ec2:DescribeInstanceTypeOfferings, ec2:RunInstances, ec2:CreateFleet, ec2:CreateTags, iam:PassRole, iam:GetInstanceProfile, pcs:UpdateComputeNodeGroup`	`pcs:GetComputeNodeGroup, iam:ListInstanceProfiles, ec2:DescribeImages, pcs:GetCluster`
DeleteComputeNodeGroup	`pcs:DeleteComputeNodeGroup`	
CreateQueue	`pcs:CreateQueue`	`pcs:ListComputeNodeGroups, pcs:GetCluster`
ListQueues	`pcs:ListQueues`	`pcs:GetCluster`
GetQueue	`pcs:GetQueue`	
UpdateQueue	`pcs:UpdateQueue`	`pcs:ListComputeNodeGroups, pcs:GetQueue`

API 작업	최소 권한	콘솔에 대한 추가 권한
DeleteQueue	`pcs:DeleteQueue`	

태그를 사용할 수 있는 최소 권한

AWS PCS의 리소스에 태그를 사용하려면 다음 권한이 필요합니다.

```
pcs:ListTagsForResource,
pcs:TagResource,
pcs:UntagResource
```

로그를 지원하는 최소 권한

AWS PCS는 Amazon CloudWatch Logs(CloudWatch Logs)로 로그 데이터를 전송합니다. 자격 증명에 CloudWatch Logs를 사용할 수 있는 최소 권한이 있는지 확인해야 합니다. 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 CloudWatch Logs 리소스에 대한 액세스 권한 관리 개요를 참조하세요. Amazon CloudWatch

서비스가 CloudWatch Logs로 로그를 전송하는 데 필요한 권한에 대한 자세한 내용은 Amazon CloudWatch Logs 사용 설명서의 AWS 서비스에서 로깅 활성화를 참조하세요. Amazon CloudWatch

용량 블록을 사용할 수 있는 최소 권한

ML용 Amazon EC2 용량 블록은 특정 날짜 및 시간 범위 내에 GPU 기반 가속 컴퓨팅 인스턴스를 미리 예약하여 단기 워크로드를 지원할 수 있는 Amazon EC2 구매 옵션입니다. 자세한 내용은 AWS PCS에서 ML에 Amazon EC2 용량 블록 사용 단원을 참조하십시오.

컴퓨팅 노드 그룹을 생성하거나 업데이트할 때 용량 블록을 사용하도록 선택합니다. 컴퓨팅 노드 그룹을 생성하거나 업데이트하는 데 사용하는 IAM 자격 증명에는 다음 권한이 있어야 합니다.

```
ec2:DescribeCapacityReservations
```

서비스 관리자의 최소 권한

다음 IAM 정책은 IAM 자격 증명(사용자, 그룹 또는 역할)이 AWS PCS 서비스를 구성하고 관리하는 데 필요한 최소 권한을 지정합니다.

> **Note**
>
> 서비스를 구성하고 관리하지 않는 사용자는 이러한 권한이 필요하지 않습니다. 작업을 실행하는 사용자는 SSL(Secure shell)을 사용하여 클러스터에 연결합니다. AWS Identity and Access Management (IAM)은 SSH에 대한 인증 또는 권한 부여를 처리하지 않습니다.

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "PCSAccess",
      "Effect": "Allow",
      "Action": [
        "pcs:*"
      ],
      "Resource": "*"
    },
    {
      "Sid": "EC2Access",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateNetworkInterface",
        "ec2:DescribeImages",
        "ec2:GetSecurityGroupsForVpc",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeVpcs",
        "ec2:DescribeLaunchTemplates",
        "ec2:DescribeLaunchTemplateVersions",
        "ec2:DescribeInstanceTypes",
        "ec2:DescribeInstanceTypeOfferings",
        "ec2:RunInstances",
        "ec2:CreateFleet",
        "ec2:CreateTags",
        "ec2:DescribeCapacityReservations"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamInstanceProfile",
      "Effect": "Allow",
```

```
      "Action": [
        "iam:GetInstanceProfile"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IamPassRole",
      "Effect": "Allow",
      "Action": [
        "iam:PassRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/*/AWSPCS*",
        "arn:aws:iam::*:role/AWSPCS*",
        "arn:aws:iam::*:role/aws-pcs/*",
        "arn:aws:iam::*:role/*/aws-pcs/*"
      ],
      "Condition": {
        "StringEquals": {
           "iam:PassedToService": [
             "ec2.amazonaws.com"
           ]
        }
      }
    },
    {
      "Sid": "SLRAccess",
      "Effect": "Allow",
      "Action": [
        "iam:CreateServiceLinkedRole"
      ],
      "Resource": [
        "arn:aws:iam::*:role/aws-service-role/pcs.amazonaws.com/AWSServiceRoleFor*",
        "arn:aws:iam::*:role/aws-service-role/spot.amazonaws.com/AWSServiceRoleFor*"
      ],
      "Condition": {
        "StringLike": {
          "iam:AWSServiceName": [
            "pcs.amazonaws.com",
            "spot.amazonaws.com"
          ]
        }
      }
    },
```

```
    {
      "Sid": "AccessKMSKey",
      "Effect": "Allow",
      "Action": [
        "kms:Decrypt",
        "kms:Encrypt",
        "kms:GenerateDataKey",
        "kms:CreateGrant",
        "kms:DescribeKey"
      ],
      "Resource": "*"
    },
    {
      "Sid": "SecretManagementAccess",
      "Effect": "Allow",
      "Action": [
        "secretsmanager:CreateSecret",
        "secretsmanager:TagResource",
        "secretsmanager:UpdateSecret",
        "secretsmanager:RotateSecret"
      ],
      "Resource": "*"
    },
    {
       "Sid": "ServiceLogsDelivery",
       "Effect": "Allow",
       "Action": [
         "pcs:AllowVendedLogDeliveryForResource",
         "logs:PutDeliverySource",
         "logs:PutDeliveryDestination",
         "logs:CreateDelivery"
       ],
       "Resource": "*"
    }
  ]
}
```

AWS 병렬 컴퓨팅 서비스를 위한 IAM 인스턴스 프로파일

EC2 인스턴스에서 실행되는 애플리케이션은 모든 AWS API 요청에 AWS 자격 증명을 포함해야 합니다. IAM 역할을 사용하여 EC2 인스턴스에서 임시 자격 증명을 관리하는 것이 좋습니다. 이를 위해 인스턴스 프로파일을 정의하고 인스턴스에 연결할 수 있습니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2에 대한 IAM 역할을 참조하세요.

> **Note**
>
> AWS Management Console 를 사용하여 Amazon EC2에 대한 IAM 역할을 생성하면 콘솔이 인스턴스 프로파일을 자동으로 생성하고 IAM 역할과 동일한 이름을 부여합니다. , AWS CLI AWS API 작업 또는 AWS SDK를 사용하여 IAM 역할을 생성하는 경우 인스턴스 프로파일을 별도의 작업으로 생성합니다. 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 인스턴스 프로파일을 참조하세요.

컴퓨팅 노드 그룹을 생성할 때 인스턴스 프로파일의 Amazon 리소스 이름(ARN)을 지정해야 합니다. 일부 또는 모든 컴퓨팅 노드 그룹에 대해 다른 인스턴스 프로파일을 선택할 수 있습니다.

요구 사항

인스턴스 프로파일의 IAM 역할

인스턴스 프로파일과 연결된 IAM 역할은 경로/aws-pcs/에가 있거나 이름이 로 시작해야 합니다AWSPCS.

IAM 역할 ARNs

- arn:aws:iam::*:role/AWSPCS-example-role-1
- arn:aws:iam::*:role/aws-pcs/example-role-2

권한

AWS PCS의 인스턴스 프로파일과 연결된 IAM 역할에는 다음 정책이 포함되어야 합니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "pcs:RegisterComputeNodeGroupInstance"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
```

```
    ]
}
```

추가 정책

인스턴스 프로파일에 관리형 정책을 추가하는 것이 좋습니다. 예제:

- AmazonS3ReadOnlyAccess는 모든 S3 버킷에 대한 읽기 전용 액세스를 제공합니다.
- AmazonSSMManagedInstanceCore는 Amazon Management Console에서 직접 원격 액세스와 같은 AWS Systems Manager 서비스 핵심 기능을 활성화합니다.
- CloudWatchAgentServerPolicy에는 서버에서 AmazonCloudWatchAgent를 사용하는 데 필요한 권한이 포함되어 있습니다.

특정 사용 사례를 지원하는 자체 IAM 정책을 포함할 수도 있습니다.

AWS PCS용 인스턴스 프로파일 생성

AWS PCS console

컴퓨팅 노드 그룹을 생성할 때 기본 프로필 생성을 선택하여 필요한 최소 정책으로 AWS PCS에서 생성하도록 합니다.

Amazon EC2 console

Amazon EC2 콘솔에서 직접 인스턴스 프로파일을 생성할 수 있습니다. 자세한 내용은 AWS Identity and Access Management 사용 설명서의 인스턴스 프로파일 사용을 참조하세요.

> **Important**
>
> IAM 역할 이름AWSPCS에 필수 접두사를 사용해야 합니다.

AWS CLI

AWS CLI를 사용하여 기본 인스턴스 프로파일 설정

> **Note**
>
> 다음 예제의 *example-role*을 IAM 역할의 이름으로 바꿉니다.

1. 를 경로 속성 /aws-pcs/ 또는 로 시작하는 이름으로 사용하여 IAM 역할을 생성합니다AWSPCS.

   a. 다음 콘텐츠를 복사하여 라는 새 텍스트 파일에 붙여 넣습니다trust_policy.json.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
             "Principal": {
                "Service": [
                    "ec2.amazonaws.com"
                ]
            },
            "Action": [
                "sts:AssumeRole"
            ]
        }
    ]
}
```

   b. 다음 명령 중 하나를 사용하여 IAM 역할을 생성합니다.

```
aws iam create-role --path /aws-pcs/ --role-name example-role --assume-role-policy-document file://trust_policy.json
```

   또는

```
aws iam create-role --role-name AWSPCS-example-role --assume-role-policy-document file://trust_policy.json
```

2. 권한을 연결합니다.

   a. 다음 콘텐츠를 복사하여 라는 새 텍스트 파일에 붙여 넣습니다policy_document.json.

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

```
            "Action": [
                "pcs:RegisterComputeNodeGroupInstance"
            ],
            "Resource": "*",
            "Effect": "Allow"
        }
    ]
}
```

b. 정책 문서를 역할에 연결합니다. 이 명령은 정책을 인라인 정책으로 연결합니다.

```
aws iam put-role-policy \
    --role-name example-role \
    --policy-name pcsRegisterInstancePolicy \
    --policy-document file://policy_document.json
```

3. 인스턴스 프로파일을 생성합니다. ***example-profile***을 인스턴스 프로파일 이름으로 바꿉니다.

```
aws iam create-instance-profile --instance-profile-name example-profile
```

4. IAM 역할을 인스턴스 프로파일과 연결합니다.

```
aws iam add-role-to-instance-profile \
   --instance-profile-name example-profile \
   --role-name example-role
```

AWS PCS에 사용되는 인스턴스 프로파일 찾기

1. AWS PCS에 대한 IAM 역할의 정확한 이름을 모르는 경우 다음 AWS CLI 명령을 사용하여 AWS PCS 이름 요구 사항을 충족하는 IAM 역할을 나열합니다.

```
aws iam list-roles --query "Roles[?starts_with(RoleName, 'AWSPCS') ||
 contains(Path, '/aws-pcs/')].[RoleName]" --output text
```

2. 다음 AWS CLI 명령을 사용하여 특정 IAM 역할과 연결된 인스턴스 프로파일을 나열합니다. *role-name*을 AWS PCS 이름 요구 사항을 충족하는 IAM 역할의 이름으로 바꿉니다.

```
aws iam list-instance-profiles-for-role --role-name role-name
```

AWS 병렬 컴퓨팅 서비스 자격 증명 및 액세스 문제 해결

다음 정보를 사용하여 AWS PCS 및 IAM 작업 시 발생할 수 있는 일반적인 문제를 진단하고 수정할 수 있습니다.

주제

- AWS PCS에서 작업을 수행할 권한이 없음
- iam:PassRole을 수행하도록 인증되지 않음
- 내 외부의 사람이 내 AWS PCS 리소스에 액세스 AWS 계정 하도록 허용하고 싶습니다.

AWS PCS에서 작업을 수행할 권한이 없음

작업을 수행할 권한이 없다는 오류가 표시되면 작업을 수행할 수 있도록 정책을 업데이트해야 합니다.

다음의 예제 오류는 `mateojackson` IAM 사용자가 콘솔을 사용하여 가상 *`my-example-widget`* 리소스에 대한 세부 정보를 보려고 하지만 가상 `pcs:`*`GetWidget`* 권한이 없을 때 발생합니다.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 pcs:GetWidget on resource: my-example-widget
```

이 경우, `pcs:`*`GetWidget`* 작업을 사용하여 *`my-example-widget`* 리소스에 액세스할 수 있도록 `mateojackson` 사용자 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

iam:PassRole을 수행하도록 인증되지 않음

`iam:PassRole` 작업을 수행할 권한이 없다는 오류가 수신되면 AWS PCS에 역할을 전달할 수 있도록 정책을 업데이트해야 합니다.

일부 AWS 서비스 에서는 새 서비스 역할 또는 서비스 연결 역할을 생성하는 대신 기존 역할을 해당 서비스에 전달할 수 있습니다. 이렇게 하려면 역할을 서비스에 전달할 권한이 있어야 합니다.

다음 예제 오류는 라는 IAM 사용자가 콘솔을 사용하여 AWS PCS에서 작업을 수행하려고 `marymajor` 할 때 발생합니다. 하지만 작업을 수행하려면 서비스 역할이 부여한 권한이 서비스에 있어야 합니다. Mary는 서비스에 역할을 전달할 권한이 없습니다.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

이 경우, Mary가 `iam:PassRole`작업을 수행할 수 있도록 Mary의 정책을 업데이트해야 합니다.

도움이 필요한 경우 AWS 관리자에게 문의하세요. 관리자는 로그인 자격 증명을 제공한 사람입니다.

내 외부의 사람이 내 AWS PCS 리소스에 액세스 AWS 계정 하도록 허용하고 싶습니다.

다른 계정의 사용자 또는 조직 외부의 사람이 리소스에 액세스할 때 사용할 수 있는 역할을 생성할 수 있습니다. 역할을 수임할 신뢰할 수 있는 사람을 지정할 수 있습니다. 리소스 기반 정책 또는 액세스 제어 목록(ACL)을 지원하는 서비스의 경우, 이러한 정책을 사용하여 다른 사람에게 리소스에 대한 액세스 권한을 부여할 수 있습니다.

자세한 내용은 다음을 참조하세요.

- AWS PCS가 이러한 기능을 지원하는지 여부를 알아보려면 섹션을 참조하세요AWS 병렬 컴퓨팅 서비스가 IAM과 작동하는 방식.
- 소유 AWS 계정 한의 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 소유한 다른의 IAM 사용자에게 액세스 권한 제공을 참조 AWS 계정 하세요.
- 타사에 리소스에 대한 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 타사가 AWS 계정 소유한에 대한 액세스 권한 제공을 AWS 계정참조하세요.
- ID 페더레이션을 통해 액세스 권한을 제공하는 방법을 알아보려면 IAM 사용 설명서의 외부에서 인증된 사용자에게 액세스 권한 제공(ID 페더레이션)을 참조하세요.
- 크로스 계정 액세스에 대한 역할과 리소스 기반 정책 사용의 차이점을 알아보려면 IAM 사용 설명서의 IAM의 크로스 계정 리소스 액세스를 참조하세요.

AWS 병렬 컴퓨팅 서비스에 대한 규정 준수 검증

AWS 서비스 가 특정 규정 준수 프로그램의 범위 내에 있는지 알아보려면 AWS 서비스 규정 준수 프로그램 제공 범위 내를 참조하고 관심 있는 규정 준수 프로그램을 선택합니다. 일반 정보는 AWS 규정 준수 프로그램.

를 사용하여 타사 감사 보고서를 다운로드할 수 있습니다 AWS Artifact. 자세한 내용은 Downloading Reports inDownloading AWS Artifact을 참조하세요.

사용 시 규정 준수 책임은 데이터의 민감도, 회사의 규정 준수 목표 및 관련 법률과 규정에 따라 AWS 서비스 결정됩니다. 사용 시 규정 준수 책임에 대한 자세한 내용은 AWS 보안 설명서를 AWS 서비스참조하세요.

AWS 병렬 컴퓨팅 서비스의 복원성

AWS 글로벌 인프라는 AWS 리전 및 가용 영역을 중심으로 구축됩니다.는 지연 시간이 짧고 처리량이 많으며 중복성이 높은 네트워킹과 연결된 물리적으로 분리되고 격리된 여러 가용 영역을 AWS 리전 제공합니다. 가용 영역을 사용하면 중단 없이 영역 간에 자동으로 장애 극복 조치가 이루어지는 애플리케이션 및 데이터베이스를 설계하고 운영할 수 있습니다. 가용 영역은 기존의 단일 또는 다중 데이터 센터 인프라보다 가용성, 내결함성, 확장성이 뛰어납니다.

AWS 리전 및 가용 영역에 대한 자세한 내용은 AWS 글로벌 인프라를 참조하세요.

AWS 병렬 컴퓨팅 서비스의 인프라 보안

관리형 서비스인 AWS Parallel Computing Service는 AWS 글로벌 네트워크 보안으로 보호됩니다. AWS 보안 서비스 및가 인프라를 AWS 보호하는 방법에 대한 자세한 내용은 AWS 클라우드 보안을 참조하세요. 인프라 보안 모범 사례를 사용하여 환경을 설계하려면 보안 원칙 AWS Well-Architected Framework의 인프라 보호를 참조하세요 AWS .

AWS 에서 게시한 API 호출을 사용하여 네트워크를 통해 AWS PCS에 액세스합니다. 클라이언트는 다음을 지원해야 합니다.

- Transport Layer Security(TLS). TLS 1.2는 필수이며 TLS 1.3을 권장합니다.
- DHE(Ephemeral Diffie-Hellman) 또는 ECDHE(Elliptic Curve Ephemeral Diffie-Hellman)와 같은 완전 전송 보안(PFS)이 포함된 암호 제품군. Java 7 이상의 최신 시스템은 대부분 이러한 모드를 지원합니다.

AWS PCS가 클러스터를 생성하면 서비스는 계정의 컴퓨팅 노드와 별도로 서비스 소유 계정에서 Slurm 컨트롤러를 시작합니다. 컨트롤러와 컴퓨팅 노드 간의 통신을 연결하기 위해 AWS PCS는 VPC에 교차 계정 탄력적 네트워크 인터페이스(ENI)를 생성합니다. Slurm 컨트롤러는 ENI를 사용하여 다양한 컴퓨팅 노드를 관리하고 통신 AWS 계정하여 리소스의 보안 및 격리를 유지하면서 효율적인 HPC 및 AI/ML 작업을 용이하게 합니다.

AWS 병렬 컴퓨팅 서비스의 취약성 분석 및 관리

구성 및 IT 제어는 AWS 와 사용자 간의 공동 책임입니다. 자세한 내용은 AWS 공동 책임 모델을 참조하세요.는 컨트롤러 인스턴스에서 운영 체제 패치 적용, 방화벽 구성 및 인프라 재해 복구와 같은 서비스 계정의 기본 AWS 인프라에 대한 기본 보안 작업을 AWS 처리합니다. 적합한 제3자가 이 절차를 검토하고 인증하였습니다. 자세한 내용은 보안, 자격 증명 및 규정 준수를 위한 모범 사례를 참조하세요.

> **Note**
>
> Slurm 컨트롤러는 업데이트하는 동안 사용할 수 없습니다. 실행 중인 작업은 영향을 받지 않습니다. 클러스터의 컨트롤러를 사용할 수 없을 때 제출된 작업은 컨트롤러를 사용할 수 있을 때까지 보류됩니다.

AWS 계정다음과 같은 기본 인프라의 보안에 대한 책임은 사용자에게 있습니다.

- 업데이트 및 보안 패치를 포함하여 코드를 유지 관리합니다.
- 컴퓨팅 노드 그룹의 Amazon Machine Image(AMI)에서 운영 체제를 패치 및 업데이트하고 업데이트된 AMI를 사용하도록 컴퓨팅 노드 그룹을 업데이트합니다.
- 스케줄러를 업데이트하여 지원되는 버전 내에 유지합니다. 컴퓨팅 노드 그룹의 AMI를 업데이트하고 업데이트된 AMI를 사용하도록 컴퓨팅 노드 그룹을 업데이트합니다.
- 사용자 클라이언트와 사용자 클라이언트가 연결하는 노드 간의 통신을 인증하고 암호화합니다.

컴퓨팅 노드 그룹의 AMI 업데이트에 대한 자세한 내용은 섹션을 참조하세요AWS PCS용 Amazon Machine Image(AMIs).

교차 서비스 혼동된 대리자 방지

혼동된 대리자 문제는 작업을 수행할 권한이 없는 엔터티가 권한이 더 많은 엔터티에 작업을 수행하도록 강요할 수 있는 보안 문제입니다. 에서 AWS교차 서비스 가장은 혼동된 대리자 문제를 초래할 수 있습니다. 교차 서비스 가장은 한 서비스(직접 호출하는 서비스)가 다른 서비스(직접 호출되는 서비스)를 직접 호출할 때 발생할 수 있습니다. 직접 호출하는 서비스는 다른 고객의 리소스에 대해 액세스 권한이 없는 방식으로 작동하게 권한을 사용하도록 조작될 수 있습니다. 이를 방지하기 위해 AWS 에서는 계정의 리소스에 대한 액세스 권한이 부여된 서비스 위탁자를 사용하여 모든 서비스에 대한 데이터를 보호하는 데 도움이 되는 도구를 제공합니다.

리소스 정책에서 `aws:SourceArn` 및 `aws:SourceAccount` 전역 조건 컨텍스트 키를 사용하여 AWS 병렬 컴퓨팅 서비스(AWS PCS)가 리소스에 다른 서비스를 제공하는 권한을 제한하는 것이 좋습니다. 하나의 리소스만 교차 서비스 액세스와 연결되도록 허용하려는 경우 `aws:SourceArn`을 사용하세요. 해당 계정의 모든 리소스가 교차 서비스 사용과 연결되도록 허용하려는 경우 `aws:SourceAccount`을(를) 사용합니다.

혼동된 대리자 문제로부터 보호하는 가장 효과적인 방법은 리소스의 전체 ARN이 포함된 `aws:SourceArn` 전역 조건 컨텍스트 키를 사용하는 것입니다. 리소스의 전체 ARN

을 모르거나 여러 리소스를 지정하는 경우, ARN의 알 수 없는 부분에 대해 와일드카드 문자(*)를 포함한 aws:SourceArn 글로벌 조건 컨텍스트 키를 사용합니다. 예를 들어 arn:aws:*servicename*:*:*123456789012*:*입니다.

만약 aws:SourceArn 값에 Amazon S3 버킷 ARN과 같은 계정 ID가 포함되어 있지 않은 경우, 권한을 제한하려면 두 글로벌 조건 컨텍스트 키를 모두 사용해야 합니다.

의 값은 클러스터 ARN이어야 aws:SourceArn 합니다.

다음 예제에서는 AWS PCS에서 aws:SourceArn 및 aws:SourceAccount 전역 조건 컨텍스트 키를 사용하여 혼동된 대리자 문제를 방지하는 방법을 보여줍니다.

```
{
"Version": "2012-10-17",
  "Statement": {
"Sid": "ConfusedDeputyPreventionExamplePolicy",
    "Effect": "Allow",
    "Principal": {
      "Service": "pcs.amazonaws.com"
    },
    "Action": "sts:AssumeRole",
    "Condition": {
      "ArnLike": {
        "aws:SourceArn": [
          "arn:aws:pcs:us-east-1:123456789012:cluster/*"
        ]
      },
      "StringEquals": {
        "aws:SourceAccount": "123456789012"
      }
    }
  }
}
```

컴퓨팅 노드 그룹의 일부로 프로비저닝된 Amazon EC2 인스턴스에 대한 IAM 역할

AWS PCS는 클러스터에 구성된 각 컴퓨팅 노드 그룹에 대해 Amazon EC2 용량을 자동으로 오케스트레이션합니다. 컴퓨팅 노드 그룹을 생성할 때 사용자는 iamInstanceProfileArn 필드를 통해 IAM 인스턴스 프로파일을 제공해야 합니다. 인스턴스 프로파일은 프로비저닝된 EC2 인스턴스와 연결된 권한을 지정합니다. AWS PCS는 AWSPCS 역할 이름 접두사 또는 역할 경로의 /aws-pcs/ 일부로가

있는 모든 역할을 허용합니다. 컴퓨팅 노드 그룹을 생성하거나 업데이트하는 IAM 자격 증명(사용자 또는 역할)에 대한 `iam:PassRole` 권한이 필요합니다. 사용자가 `CreateComputeNodeGroup` 또는 `UpdateComputeNodeGroup` API 작업을 호출하면 AWS PCS는 사용자가 `iam:PassRole` 작업을 수행할 수 있는지 확인합니다.

다음 예제 정책에서는 이름이 `AWSPCS`로 시작하는 IAM 역할만 전달할 수 있는 권한을 부여합니다.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/AWSPCS*",
            "Condition": {
                "StringEquals": {
                    "iam:PassedToService": [
                        "ec2.amazonaws.com"
                    ]
                }
            }
        }
    ]
}
```

AWS 병렬 컴퓨팅 서비스의 보안 모범 사례

이 섹션에서는 AWS 병렬 컴퓨팅 서비스(AWS PCS)와 관련된 보안 모범 사례를 설명합니다. 의 보안 모범 사례에 대한 자세한 내용은 보안, 자격 증명 및 규정 준수 모범 사례를 AWS참조하세요. https://aws.amazon.com/architecture/security-identity-compliance

AMI 관련 보안

- 프로덕션 워크로드에 AWS PCS 샘플 AMIs 사용하지 마세요. 샘플 AMIs입니다.
- 컴퓨팅 노드 그룹의 AMI에서 운영 체제와 소프트웨어를 정기적으로 업데이트하여 취약성을 완화합니다.
- 공식 AWS 소스에서 다운로드한 인증된 공식 AWS PCS 패키지만 사용합니다.

- 컴퓨팅 노드 그룹의 AMI에서 AWS PCS 패키지를 정기적으로 업데이트하고 업데이트된 AMI를 사용하도록 컴퓨팅 노드를 업데이트합니다. 취약성을 최소화하려면이 프로세스를 자동화하는 것이 좋습니다.

자세한 내용은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs) 단원을 참조하십시오.

Slurm Workload Manager 보안

- 액세스 제어 및 네트워크 제한을 구현하여 Slurm 제어 및 컴퓨팅 노드를 보호합니다. 신뢰할 수 있는 사용자 및 시스템만 작업을 제출하고 Slurm 관리 명령에 액세스하도록 허용합니다.
- Slurm 인증과 같은 Slurm의 기본 제공 보안 기능을 사용하여 작업 제출 및 통신이 인증되도록 합니다.
- 원활한 운영 및 클러스터 지원을 유지하기 위해 Slurm 버전을 업데이트합니다.

> **Important**
>
> 지원 수명 종료(EOSL)에 도달한 Slurm 버전을 사용하는 클러스터는 즉시 중지됩니다. 사용 설명서 페이지 상단의 링크를 사용하여 AWS PCS 설명서 RSS 피드를 구독하면 Slurm 버전이 EOSL에 가까워지면 알림을 받을 수 있습니다.

  자세한 내용은 AWS PCS의 Slurm 버전 단원을 참조하십시오.
- 클러스터 보안 암호를 정기적으로 교체하여 보안 규정 준수를 유지하고 잠재적 보안 침해를 해결합니다. 이는 HIPAA 및 FedRAMP 규정 준수에 필요합니다.

  자세한 내용은 AWS PCS에서 클러스터 보안 암호 교체 단원을 참조하십시오.

모니터링 및 로깅

- Amazon CloudWatch Logs 및 AWS CloudTrail 를 사용하여 클러스터 및의 작업을 모니터링하고 기록합니다 AWS 계정. 문제 해결 및 감사에 데이터를 사용합니다.

네트워크 보안

- 별도의 VPC에 AWS PCS 클러스터를 배포하여 HPC 환경을 다른 네트워크 트래픽과 격리합니다.

- 보안 그룹 및 네트워크 액세스 제어 목록(ACLs)을 사용하여 AWS PCS 인스턴스 및 서브넷에 대한 인바운드 및 아웃바운드 트래픽을 제어합니다.
- AWS PrivateLink 또는 VPC 엔드포인트를 사용하여 클러스터와 네트워크 내 다른 AWS 서비스 간의 AWS 네트워크 트래픽을 유지합니다. 자세한 내용은 인터페이스 엔드포인트를 AWS Parallel Computing Service 사용한 액세스(AWS PrivateLink) 단원을 참조하십시오.

AWS PCS에 대한 로깅 및 모니터링

모니터링은 AWS PCS 및 기타 AWS 리소스의 안정성, 가용성 및 성능을 유지하는 데 중요한 부분입니다. AWS는 다음과 같은 모니터링 도구를 제공하여 AWS PCS를 관찰하고, 이상이 있을 때 보고하고, 적절한 경우 자동 조치를 취합니다.

- Amazon CloudWatch는 AWS 리소스와 AWS 에서 실행하는 애플리케이션을 실시간으로 모니터링합니다. 지표를 수집 및 추적하고, 사용자 지정 대시보드를 생성할 수 있으며, 지정된 지표가 지정한 임곗값에 도달하면 사용자에게 알리거나 조치를 취하도록 경보를 설정할 수 있습니다. 예를 들어 CloudWatch에서 Amazon EC2 인스턴스의 CPU 사용량 또는 기타 지표를 추적하고 필요할 때 자동으로 새 인스턴스를 시작할 수 있습니다. 자세한 내용은 Amazon CloudWatch 사용 설명서를 참조하세요.
- Amazon CloudWatch Logs로 Amazon EC2 인스턴스, CloudTrail, 기타 소스의 로그 파일을 모니터링, 저장 및 액세스할 수 있습니다. CloudWatch Logs는 로그 파일의 정보를 모니터링하고 특정 임계값에 도달하면 사용자에게 알릴 수 있습니다. 또한 매우 내구성이 뛰어난 스토리지에 로그 데이터를 저장할 수 있습니다. 자세한 내용은 Amazon CloudWatch Logs 사용자 안내서를 참조하세요.
- AWS CloudTrail는 AWS 계정에 의해 또는 계정을 대신하여 수행된 API 호출 및 관련 이벤트를 캡처하고 사용자가 지정한 Amazon S3 버킷에 로그 파일을 전송합니다. 호출한 사용자 및 계정 AWS, 호출이 수행된 소스 IP 주소, 호출이 발생한 시기를 식별할 수 있습니다. 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하십시오.

AWS PCS의 작업 완료 로그

작업 완료 로그는 추가 비용 없이 완료 시 AWS 병렬 컴퓨팅 서비스(AWS PCS) 작업에 대한 주요 세부 정보를 제공합니다. 다른 AWS 서비스를 사용하여 Amazon CloudWatch Logs, Amazon Simple Storage Service(Amazon S3) 및 Amazon Data Firehose와 같은 로그 데이터에 액세스하고 처리할 수 있습니다. AWS PCS는 다음과 같이 작업에 대한 메타데이터를 기록합니다.

- 작업 ID 및 이름
- 사용자 및 그룹 정보
- 작업 상태(예: `COMPLETED`, `FAILED`, `CANCELLED`)
- 사용된 파티션
- 시간 제한
- 시작, 종료, 제출 및 적격 시간

- 노드 목록 및 개수
- 프로세서 수
- 작업 디렉터리
- 리소스 사용량(CPU, 메모리)
- 종료 코드
- 노드 세부 정보(이름, 인스턴스 IDs, 인스턴스 유형)

목차

사전 조건

AWS PCS 클러스터를 관리하는 IAM 보안 주체는 `pcs:AllowVendedLogDeliveryForResource` 작업을 허용해야 합니다.

다음 예제 IAM 정책은 필요한 권한을 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "PcsAllowVendedLogsDelivery",
         "Effect": "Allow",
         "Action": ["pcs:AllowVendedLogDeliveryForResource"],
         "Resource": [
            "arn:aws:pcs:*::cluster/*"
         ]
      }
```

```
    ]
}
```

작업 완료 로그 설정

AWS Management Console 또는를 사용하여 AWS PCS 클러스터에 대한 작업 완료 로그를 설정할 수 있습니다 AWS CLI.

AWS Management Console

콘솔을 사용하여 작업 완료 로그를 설정하려면

1. AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 작업 완료 로그를 추가할 클러스터를 선택합니다.
4. 클러스터 세부 정보 페이지에서 로그 탭을 선택합니다.
5. 작업 완료 로그에서 추가를 선택하여 CloudWatch Logs, Amazon S3 및 Firehose 중에서 최대 3개의 로그 전송 대상을 추가합니다. CloudWatch
6. 로그 전송 업데이트를 선택합니다.

AWS CLI

를 사용하여 작업 완료 로그를 설정하려면 AWS CLI

1. 로그 전송 대상 생성:

```
aws logs put-delivery-destination --region region \
  --name pcs-logs-destination \
  --delivery-destination-configuration \
  destinationResourceArn=resource-arn
```

다음과 같이 바꿉니다.

- *region* -와 같이 대상을 생성하려는 AWS 리전 입니다. `us-east-1`
- *pcs-logs-destination* - 대상의 이름
- *resource-arn* - CloudWatch Logs 로그 그룹, S3 버킷 또는 Firehose 전송 스트림의 Amazon 리소스 이름(ARN)입니다.

자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 PutDeliveryDestination을 참조하시기 바랍니다.

2. PCS 클러스터를 로그 전송 소스로 설정합니다.

```
aws logs put-delivery-source --region region \
  --name cluster-logs-source-name \
  --resource-arn cluster-arn \
  --log-type PCS_JOBCOMP_LOGS
```

다음과 같이 바꿉니다.

- *region* -와 같은 클러스터 AWS 리전 의 `us-east-1`
- *cluster-logs-source-name* - 소스의 이름입니다.
- *cluster-arn* - AWS PCS 클러스터의 ARN

자세한 내용은 Amazon CloudWatch Logs API 참조의 PutDeliverySource를 참조하세요. Amazon CloudWatch

3. 전송 소스를 전송 대상에 연결합니다.

```
aws logs create-delivery --region region \
  --delivery-source-name cluster-logs-source \
  --delivery-destination-arn destination-arn
```

다음과 같이 바꿉니다.

- *region* - AWS 리전와 같은 `us-east-1`
- *cluster-logs-source* - 전송 소스의 이름입니다.
- *destination-arn* - 전송 대상의 ARN입니다.

자세한 내용은 Amazon CloudWatch Logs API 참조의 CreateDelivery를 참조하세요. Amazon CloudWatch

작업 완료 로그를 찾는 방법

CloudWatch Logs에서 로그 대상을 구성할 수 있으며 Amazon S3. AWS PCS는 다음과 같은 구조화된 경로 이름과 파일 이름을 사용합니다.

CloudWatch Logs

AWS PCS는 CloudWatch Logs 스트림에 다음 이름 형식을 사용합니다.

```
AWSLogs/PCS/cluster-id/jobcomp.log
```

예: `AWSLogs/PCS/pcs_abc123de45/jobcomp.log`

Amazon S3

AWS PCS는 S3 경로에 다음 이름 형식을 사용합니다.

```
AWSLogs/account-id/PCS/region/cluster-id/jobcomp/year/month/day/hour/
```

예: `AWSLogs/111122223333/PCS/us-east-1/pcs_abc123de45/jobcomp/2025/06/19/11/`

AWS PCS는 로그 파일에 다음 이름 형식을 사용합니다.

```
PCS_jobcomp_year-month-day-hour_cluster-id_random-id.log.gz
```

예: `PCS_jobcomp_2025-06-19-11_pcs_abc123de45_04be080b.log.gz`

작업 완료 로그 필드

AWS PCS는 작업 완료 로그 데이터를 JSON 객체로 기록합니다. JSON 컨테이너에는 작업 세부 정보가 `jobcomp` 들어 있습니다. 다음 표에서는 `jobcomp` 컨테이너 내부의 필드를 설명합니다. 일부 필드는 배열 작업 또는 이기종 작업과 같은 특정 상황에서만 존재합니다.

작업 완료 로그 필드

이름	예시 값	필수	참고
`job_id`	`11`	yes	항상 값과 함께 표시
`user`	`"root"`	yes	항상 값과 함께 표시

이름	예시 값	필수	참고
user_id	0	yes	항상 값과 함께 표시
group	"root"	yes	항상 값과 함께 표시
group_id	0	yes	항상 값과 함께 표시
name	"wrap"	yes	항상 값과 함께 표시
job_state	"COMPLETED"	yes	항상 값과 함께 표시
partition	"Hydra-MpiQueue-abcdef01-7"	yes	항상 값과 함께 표시
time_limit	"UNLIMITED"	yes	항상 존재하지만 존재할 수 있음 "UNLIMITED"
start_time	"2025-06-19T10:58:57"	yes	항상 존재하지만 존재할 수 있음 "Unknown"
end_time	"2025-06-19T10:58:57"	yes	항상 존재하지만 존재할 수 있음 "Unknown"
node_list	"Hydra-MpiNG-abcdef01-2345-1"	yes	항상 값과 함께 표시
node_cnt	1	yes	항상 값과 함께 표시
proc_cnt	1	yes	항상 값과 함께 표시
work_dir	"/root"	yes	항상 존재하지만 존재할 수 있음 "Unknown"

이름	예시 값	필수	참고
reservation_name	"weekly_maintenance"	yes	항상 존재하지만 빈 문자열일 수 있음 ""
tres.cpu	1	yes	항상 값과 함께 표시
tres.mem.val	600	yes	항상 값과 함께 표시
tres.mem.unit	"M"	yes	"M" 또는 일 수 있음 "bb"
tres.node	1	yes	항상 값과 함께 표시
tres.billing	1	yes	항상 값과 함께 표시
account	"finance"	yes	항상 존재하지만 빈 문자열일 수 있음 ""
qos	"normal"	yes	항상 존재하지만 빈 문자열일 수 있음 ""
wc_key	"project_1"	yes	항상 존재하지만 빈 문자열일 수 있음 ""
cluster	"unknown"	yes	항상 존재하지만 존재할 수 있음 "unknown"
submit_time	"2025-06-19T10:55:46"	yes	항상 존재하지만 존재할 수 있음 "Unknown"
eligible_time	"2025-06-19T10:55:46"	yes	항상 존재하지만 존재할 수 있음 "Unknown"

이름	예시 값	필수	참고
array_job_id	12	아니요	작업이 배열 작업인 경우에만 표시됩니다.
array_task_id	1	아니요	작업이 배열 작업인 경우에만 표시됩니다.
het_job_id	10	아니요	작업이 이기종 작업인 경우에만 표시됩니다.
het_job_offset	0	아니요	작업이 이기종 작업인 경우에만 표시됩니다.
derived_exit_code_status	0	yes	항상 값과 함께 표시
derived_exit_code_signal	0	yes	항상 값과 함께 표시
exit_code_status	0	yes	항상 값과 함께 표시
exit_code_signal	0	yes	항상 값과 함께 표시
node_details[0].name	"Hydra-MpiNG-abcdef01-2345-1"	아니요	항상 존재하지만 존재할 node_details 수 있음 "[]"
node_details[0].instance_id	"i-0abcdef01234567a"	아니요	항상 존재하지만 존재할 node_details 수 있음 "[]"

이름	예시 값	필수	참고
node_details[0].instance_type	"t4g.micro"	아니요	항상 존재하지만 존재할 node_details 수 있음 "[]"

작업 완료 로그 예

다음 예제에서는 다양한 작업 유형 및 상태에 대한 작업 완료 로그를 보여줍니다.


```
{ "jobcomp": { "job_id": 1, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T16:32:57",
 "end_time": "2025-06-19T16:33:03", "node_list": "Hydra-MpiNG-abcdef01-2345-
[1-2]", "node_cnt": 2, "proc_cnt": 2, "work_dir": "/usr/bin", "reservation_name":
 "", "tres": { "cpu": 2, "mem": { "val": 1944, "unit": "M" }, "node": 2,
 "billing": 2 }, "account": "", "qos": "", "wc_key": "", "cluster": "unknown",
 "submit_time": "2025-06-19T16:29:40", "eligible_time": "2025-06-19T16:29:41",
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc123def45678", "instance_type": "t4g.micro" }, { "name":
 "Hydra-MpiNG-abcdef01-2345-2", "instance_id": "i-0def456abc78901", "instance_type":
 "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 2, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T16:33:13",
 "end_time": "2025-06-19T16:33:14", "node_list": "Hydra-MpiNG-abcdef01-2345-
[1-2]", "node_cnt": 2, "proc_cnt": 2, "work_dir": "/usr/bin", "reservation_name":
 "", "tres": { "cpu": 2, "mem": { "val": 1944, "unit": "M" }, "node": 2,
 "billing": 2 }, "account": "", "qos": "", "wc_key": "", "cluster": "unknown",
 "submit_time": "2025-06-19T16:33:13", "eligible_time": "2025-06-19T16:33:13",
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc123def45678", "instance_type": "t4g.micro" }, { "name":
 "Hydra-MpiNG-abcdef01-2345-2", "instance_id": "i-0def456abc78901", "instance_type":
 "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 3, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T22:58:57", "end_time":
 "2025-06-19T22:58:57", "node_list": "Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
```

```
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 972, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T22:55:46",
 "eligible_time": "2025-06-19T22:55:46", "derived_exit_code_status": 0,
 "derived_exit_code_signal": 0, "exit_code_status": 0, "exit_code_signal":
 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1", "instance_id":
 "i-0abc234def56789", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 4, "user": "root", "user_id": 0, "group": "root",
 "group_id": 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-
MpiQueue-abcdef01-7", "time_limit": "525600", "start_time": "2025-06-19T23:04:27",
 "end_time": "2025-06-19T23:04:27", "node_list": "Hydra-MpiNG-abcdef01-2345-
[1-2]", "node_cnt": 2, "proc_cnt": 2, "work_dir": "/root", "reservation_name":
 "", "tres": { "cpu": 2, "mem": { "val": 1944, "unit": "M" }, "node": 2,
 "billing": 2 }, "account": "", "qos": "", "wc_key": "", "cluster": "unknown",
 "submit_time": "2025-06-19T23:01:38", "eligible_time": "2025-06-19T23:01:38",
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc234def56789", "instance_type": "t4g.micro" }, { "name":
 "Hydra-MpiNG-abcdef01-2345-2", "instance_id": "i-0def345abc67890", "instance_type":
 "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 5, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "FAILED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:09:00", "end_time":
 "2025-06-19T23:09:00", "node_list": "(null)", "node_cnt": 0, "proc_cnt": 0,
 "work_dir": "/root", "reservation_name": "", "tres": { "cpu": 1, "mem": { "val":
 1, "unit": "G" }, "node": 1, "billing": 1 }, "account": "", "qos": "", "wc_key":
 "", "cluster": "unknown", "submit_time": "2025-06-19T23:09:00", "eligible_time":
 "2025-06-19T23:09:00", "derived_exit_code_status": 0, "derived_exit_code_signal": 0,
 "exit_code_status": 0, "exit_code_signal": 1, "node_details": [] } }
{ "jobcomp": { "job_id": 6, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "CANCELLED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:09:36",
 "end_time": "2025-06-19T23:09:36", "node_list": "(null)", "node_cnt": 0, "proc_cnt":
 0, "work_dir": "/root", "reservation_name": "", "tres": { "cpu": 1, "mem":
 { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "", "qos":
 "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:09:35",
 "eligible_time": "2025-06-19T23:09:36", "het_job_id": 6, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status": 0,
 "exit_code_signal": 1, "node_details": [] } }
{ "jobcomp": { "job_id": 7, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "CANCELLED", "partition": "Hydra-MpiQueue-
abcdef01-7", "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:10:03",
 "end_time": "2025-06-19T23:10:03", "node_list": "(null)", "node_cnt": 0, "proc_cnt":
 0, "work_dir": "/root", "reservation_name": "", "tres": { "cpu": 1, "mem":
```

```
 { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "", "qos":
 "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:10:03",
 "eligible_time": "2025-06-19T23:10:03", "het_job_id": 7, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status": 0,
 "exit_code_signal": 1, "node_details": [] } }
{ "jobcomp": { "job_id": 8, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:11:24", "end_time":
 "2025-06-19T23:11:24", "node_list": "Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:11:23",
 "eligible_time": "2025-06-19T23:11:23", "het_job_id": 8, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc234def56789", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 9, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:11:24", "end_time":
 "2025-06-19T23:11:24", "node_list": "Hydra-MpiNG-abcdef01-2345-2", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:11:23",
 "eligible_time": "2025-06-19T23:11:23", "het_job_id": 8, "het_job_offset": 1,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-2",
 "instance_id": "i-0def345abc67890", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 10, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:12:24", "end_time":
 "2025-06-19T23:12:24", "node_list":"Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 400, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:12:14",
 "eligible_time": "2025-06-19T23:12:14", "het_job_id": 10, "het_job_offset": 0,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc234def56789", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 11, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:12:24", "end_time":
 "2025-06-19T23:12:24", "node_list":"Hydra-MpiNG-abcdef01-2345-2", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 600, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
```

```
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:12:14",
 "eligible_time": "2025-06-19T23:12:14", "het_job_id": 10, "het_job_offset": 1,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-2",
 "instance_id": "i-0def345abc67890", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 13, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:47:57", "end_time":
 "2025-06-19T23:47:58", "node_list":"Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 972, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:43:56",
 "eligible_time": "2025-06-19T23:43:56" , "array_job_id": 12, "array_task_id": 1,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc345def67890", "instance_type": "t4g.micro" } ] } }
{ "jobcomp": { "job_id": 12, "user": "root", "user_id": 0, "group": "root", "group_id":
 0, "name": "wrap", "job_state": "COMPLETED", "partition": "Hydra-MpiQueue-abcdef01-7",
 "time_limit": "UNLIMITED", "start_time": "2025-06-19T23:47:58", "end_time":
 "2025-06-19T23:47:58", "node_list":"Hydra-MpiNG-abcdef01-2345-1", "node_cnt":
 1, "proc_cnt": 1, "work_dir": "/root", "reservation_name": "", "tres": { "cpu":
 1, "mem": { "val": 972, "unit": "M" }, "node": 1, "billing": 1 }, "account": "",
 "qos": "", "wc_key": "", "cluster": "unknown", "submit_time": "2025-06-19T23:43:56",
 "eligible_time": "2025-06-19T23:43:56" , "array_job_id": 12, "array_task_id": 2,
 "derived_exit_code_status": 0, "derived_exit_code_signal": 0, "exit_code_status":
 0, "exit_code_signal": 0, "node_details": [ { "name": "Hydra-MpiNG-abcdef01-2345-1",
 "instance_id": "i-0abc345def67890", "instance_type": "t4g.micro" } ] } }
```


AWS PCS의 스케줄러 로그

클러스터 스케줄러에서 Amazon CloudWatch Logs, Amazon Simple Storage Service(Amazon S3) 및 Amazon Data Firehose로 세부 로깅 데이터를 보내도록 AWS PCS를 구성할 수 있습니다. 이는 모니터링 및 문제 해결에 도움이 될 수 있습니다.

목차

사전 조건

AWS PCS 클러스터를 관리하는 IAM 보안 주체는 `pcs:AllowVendedLogDeliveryForResource` 작업을 허용해야 합니다.

다음 예제 IAM 정책은 필요한 권한을 부여합니다.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Sid": "PcsAllowVendedLogsDelivery",
         "Effect": "Allow",
         "Action": ["pcs:AllowVendedLogDeliveryForResource"],
         "Resource": [
            "arn:aws:pcs:*::cluster/*"
         ]
      }
   ]
}
```

스케줄러 로그 설정

AWS Management Console 또는를 사용하여 AWS PCS 클러스터에 대한 스케줄러 로그를 설정할 수 있습니다 AWS CLI.

AWS Management Console

콘솔을 사용하여 스케줄러 로그를 설정하려면

1. AWS PCS 콘솔을 엽니다.
2. 탐색 창에서 클러스터를 선택합니다.
3. 스케줄러 로그를 추가할 클러스터를 선택합니다.
4. 클러스터 세부 정보 페이지에서 로그 탭을 선택합니다.
5. 스케줄러 로그에서 추가를 선택하여 CloudWatch Logs, Amazon S3 및 Firehose 중에서 최대 3개의 로그 전송 대상을 추가합니다. CloudWatch

6. 로그 전송 업데이트를 선택합니다.

AWS CLI

를 사용하여 스케줄러 로그를 설정하려면 AWS CLI

1. 로그 전송 대상 생성:

```
aws logs put-delivery-destination --region region \
  --name pcs-logs-destination \
  --delivery-destination-configuration \
  destinationResourceArn=resource-arn
```

다음과 같이 바꿉니다.

- *region* -와 같이 대상을 생성하려는 AWS 리전 입니다. `us-east-1`
- *pcs-logs-destination* - 대상의 이름
- *resource-arn* - CloudWatch Logs 로그 그룹, S3 버킷 또는 Firehose 전송 스트림의 Amazon 리소스 이름(ARN)입니다.

자세한 내용은 Amazon CloudWatch Logs API 참조에 나와 있는 PutDeliveryDestination을 참조하시기 바랍니다.

2. PCS 클러스터를 로그 전송 소스로 설정합니다.

```
aws logs put-delivery-source --region region \
  --name cluster-logs-source-name \
  --resource-arn cluster-arn \
  --log-type PCS_SCHEDULER_LOGS
```

다음과 같이 바꿉니다.

- *region* -와 같은 클러스터 AWS 리전 의 `us-east-1`
- *cluster-logs-source-name* - 소스의 이름입니다.
- *cluster-arn* - AWS PCS 클러스터의 ARN

자세한 내용은 Amazon CloudWatch Logs API 참조의 PutDeliverySource를 참조하세요. Amazon CloudWatch

3. 전송 소스를 전송 대상에 연결합니다.

```
aws logs create-delivery --region region \
  --delivery-source-name cluster-logs-source \
  --delivery-destination-arn destination-arn
```

다음과 같이 바꿉니다.

- *region* - AWS 리전와 같은 us-east-1
- *cluster-logs-source* - 전송 소스의 이름입니다.
- *destination-arn* - 전송 대상의 ARN입니다.

자세한 내용은 Amazon CloudWatch Logs API 참조의 CreateDelivery를 참조하세요. Amazon CloudWatch

스케줄러 로그 스트림 경로 및 이름

AWS PCS 스케줄러 로그의 경로와 이름은 대상 유형에 따라 다릅니다.

- CloudWatch Logs
  - CloudWatch Logs 스트림은 다음 명명 규칙을 따릅니다.

```
AWSLogs/PCS/${cluster_id}/${log_name}_${scheduler_major_version}.log
```

Example

```
AWSLogs/PCS/abcdef0123/slurmctld_24.05.log
```

- S3 버킷
  - S3 버킷 출력 경로는 다음 명명 규칙을 따릅니다.

```
AWSLogs/${account-id}/PCS/${region}/${cluster_id}/${log_name}/
${scheduler_major_version}/yyyy/MM/dd/HH/
```

Example

```
AWSLogs/111111111111/PCS/us-east-2/abcdef0123/slurmctld/24.05/2024/09/01/00.
```

- S3 객체 이름은 다음 규칙을 따릅니다.

```
PCS_${log_name}_${scheduler_major_version}_#{expr date 'event_timestamp', format:
 "yyyy-MM-dd-HH"}_${cluster_id}_${hash}.log
```

Example

```
PCS_slurmctld_24.05_2024-09-01-00_abcdef0123_0123abcdef.log
```

스케줄러 로그 레코드 예

AWS PCS 스케줄러 로그는 구조화되어 있습니다. 여기에는 Slurm 컨트롤러 프로세스에서 내보낸 로그 메시지 외에도 클러스터 식별자, 스케줄러 유형, 메이저 및 패치 버전과 같은 필드가 포함됩니다. 다음 예를 참고하세요

```
{
    "resource_id": "s3431v9rx2",
    "resource_type": "PCS_CLUSTER",
    "event_timestamp": 1721230979,
    "log_level": "info",
    "log_name": "slurmctld",
    "scheduler_type": "slurm",
    "scheduler_major_version": "25.05",
    "scheduler_patch_version": "3",
    "node_type": "controller_primary",
    "message": "[2024-07-17T15:42:58.614+00:00] Running as primary controller\n"
}
```

Amazon CloudWatch를 사용한 AWS 병렬 컴퓨팅 서비스 모니터링

Amazon CloudWatch는 클러스터에서 지표를 정기적으로 수집하여 AWS 병렬 컴퓨팅 서비스(AWS PCS) 클러스터 상태 및 성능을 모니터링합니다. 이러한 지표는 보존되므로 기록 데이터에 액세스하고 시간이 지남에 따라 클러스터의 성능에 대한 인사이트를 얻을 수 있습니다.

또한 CloudWatch를 사용하면 조정 요구 사항을 충족하기 위해 AWS PCS에서 시작한 EC2 인스턴스를 모니터링할 수 있습니다. 실행 중인 인스턴스의 로그를 검사할 수 있지만 일반적으로 인스턴스가 종료되면 CloudWatch 지표 및 로깅 데이터가 삭제됩니다. 그러나 EC2 시작 템플릿을 사용하여 인스턴스

에서 CloudWatch 에이전트를 구성하여 인스턴스 종료 후에도 지표와 로그를 유지할 수 있으므로 장기 모니터링 및 분석이 가능합니다.

CloudWatch를 사용한 AWS PCS 모니터링에 대해 자세히 알아보려면이 섹션의 주제를 살펴보세요.

주제

- CloudWatch를 사용하여 AWS PCS 지표 모니터링
- Amazon CloudWatch를 사용하여 AWS PCS 인스턴스 모니터링

CloudWatch를 사용하여 AWS PCS 지표 모니터링

클러스터에서 데이터를 수집하여 실시간에 가까운 지표로 변환하는 Amazon CloudWatch를 사용하여 AWS PCS 클러스터 상태를 모니터링할 수 있습니다. 이러한 통계는 15개월 동안 보관되므로 기록 정보에 액세스하고 클러스터의 성능을 더 잘 파악할 수 있습니다. 클러스터 지표는 1분 간격으로 CloudWatch로 전송됩니다. CloudWatch에 대한 자세한 내용은 Amazon CloudWatch 사용 설명서의 Amazon CloudWatch란 무엇인가요?를 참조하세요.

AWS PCS는 CloudWatch의 AWS/PCS 네임스페이스에 다음 지표를 게시합니다. 단일 차원가 있습니다`ClusterId`.

이름	설명	단위
ActualCapacity	IdleCapacity + UtilizedCapacity	개수
CapacityUtilization	UtilizedCapacity/ActualCapacity	개수
DesiredCapacity	ActualCapacity + PendingCapacity	개수
IdleCapacity	실행 중이지만 작업에 할당되지 않은 인스턴스 수	개수
UtilizedCapacity	실행 중이고 작업에 할당된 인스턴스 수	개수

Amazon CloudWatch를 사용하여 AWS PCS 인스턴스 모니터링

AWS PCS는 PCS 컴퓨팅 노드 그룹에 정의된 조정 요구 사항을 충족하기 위해 필요에 따라 Amazon EC2 인스턴스를 시작합니다. Amazon CloudWatch를 사용하여 실행 중인 이러한 인스턴스를 모니터링할 수 있습니다. 인스턴스에 로그인하고 대화형 명령줄 도구를 사용하여 실행 중인 인스턴스의 로그를 검사할 수 있습니다. 그러나 기본적으로 CloudWatch 지표 데이터는 인스턴스가 종료된 후 제한된 기간 동안만 유지되며, 인스턴스 로그는 일반적으로 인스턴스를 지원하는 EBS 볼륨과 함께 삭제됩니다. 종료된 후 PCS에서 시작한 인스턴스의 지표 또는 로깅 데이터를 유지하려면 EC2 시작 템플릿을 사용하여 인스턴스에서 CloudWatch 에이전트를 구성할 수 있습니다. 이 주제에서는 실행 중인 인스턴스 모니터링에 대한 개요를 제공하고 영구 인스턴스 지표 및 로그를 구성하는 방법의 예를 제공합니다.

실행 중인 인스턴스 모니터링

AWS PCS 인스턴스 찾기

PCS에서 시작한 인스턴스를 모니터링하려면 클러스터 또는 컴퓨팅 노드 그룹과 연결된 실행 중인 인스턴스를 찾습니다. 그런 다음 지정된 인스턴스에 대한 EC2 콘솔에서 상태, 경보 및 모니터링 섹션을 검사합니다. 이러한 인스턴스에 대해 로그인 액세스가 구성된 경우 인스턴스에 연결하고 인스턴스의 다양한 로그 파일을 검사할 수 있습니다. PCS에서 관리하는 인스턴스를 식별하는 방법에 대한 자세한 내용은 섹션을 참조하세요AWS PCS에서 컴퓨팅 노드 그룹 인스턴스 찾기.

세부 지표 활성화

기본적으로 인스턴스 지표는 5분 간격으로 수집됩니다. 1분 간격으로 지표를 수집하려면 컴퓨팅 노드 그룹 시작 템플릿에서 자세한 CloudWatch 모니터링을 활성화합니다. 자세한 내용은 자세한 CloudWatch 모니터링 켜기 단원을 참조하십시오.

영구 인스턴스 지표 및 로그 구성

인스턴스에 Amazon CloudWatch 에이전트를 설치하고 구성하여 인스턴스의 지표와 로그를 유지할 수 있습니다. 이는 세 가지 주요 단계로 구성됩니다.

1. CloudWatch 에이전트 구성을 생성합니다.
2. 구성을 PCS 인스턴스에서 검색할 수 있는 위치에 저장합니다.
3. CloudWatch 에이전트 소프트웨어를 설치하고, 구성을 가져오고, 구성을 사용하여 CloudWatch 에이전트를 시작하는 EC2 시작 템플릿을 작성합니다.

자세한 내용은 Amazon CloudWatch 사용 설명서의 CloudWatch 에이전트를 사용하여 지표, 로그 및 추적 수집 및 섹션을 참조하세요AWS PCS에서 Amazon EC2 시작 템플릿 사용. Amazon CloudWatch

CloudWatch 에이전트 구성 생성

인스턴스에 CloudWatch 에이전트를 배포하기 전에 수집할 지표, 로그 및 추적을 지정하는 JSON 구성 파일을 생성해야 합니다. 구성 파일은 마법사를 사용하거나 텍스트 편집기를 사용하여 수동으로 생성할 수 있습니다. 구성 파일은이 데모를 위해 수동으로 생성됩니다.

AWS CLI가 설치된 컴퓨터에서 다음 내용이 포함된 config.json이라는 CloudWatch 구성 파일을 생성합니다. 다음 URL을 사용하여 파일 사본을 다운로드할 수도 있습니다.

```
https://aws-hpc-recipes.s3.amazonaws.com/main/recipes/pcs/cloudwatch/assets/config.json
```

참고

- 샘플 파일의 로그 경로는 Amazon Linux 2용입니다. 인스턴스가 다른 기본 운영 체제를 사용하는 경우 경로를 적절하게 변경합니다.
- 다른 로그를 캡처하려면 아래에 항목을 추가합니다`collect_list`.
- 의 값은 템플릿 형식 변수`{brackets}`입니다. 지원되는 변수의 전체 목록은 Amazon CloudWatch 사용 설명서의 수동으로 CloudWatch 에이전트 구성 파일 생성 또는 편집을 참조하세요. Amazon CloudWatch
- 이러한 정보 유형을 수집`metrics`하지 않으려면 `logs` 또는를 생략하도록 선택할 수 있습니다.

```
{
    "agent": {
        "metrics_collection_interval": 60
    },
    "logs": {
        "logs_collected": {
            "files": {
                "collect_list": [
                    {
                        "file_path": "/var/log/cloud-init.log",
                        "log_group_class": "STANDARD",
                        "log_group_name": "/PCSLogs/instances",
                        "log_stream_name": "{instance_id}.cloud-init.log",
                        "retention_in_days": 30
                    },
                    {
                        "file_path": "/var/log/cloud-init-output.log",
                        "log_group_class": "STANDARD",
                        "log_stream_name": "{instance_id}.cloud-init-output.log",
```

```
                    "log_group_name": "/PCSLogs/instances",
                    "retention_in_days": 30
                },
                {
                    "file_path": "/var/log/amazon/pcs/bootstrap.log",
                    "log_group_class": "STANDARD",
                    "log_stream_name": "{instance_id}.bootstrap.log",
                    "log_group_name": "/PCSLogs/instances",
                    "retention_in_days": 30
                },
                {
                    "file_path": "/var/log/slurmd.log",
                    "log_group_class": "STANDARD",
                    "log_stream_name": "{instance_id}.slurmd.log",
                    "log_group_name": "/PCSLogs/instances",
                    "retention_in_days": 30
                },
                {
                    "file_path": "/var/log/messages",
                    "log_group_class": "STANDARD",
                    "log_stream_name": "{instance_id}.messages",
                    "log_group_name": "/PCSLogs/instances",
                    "retention_in_days": 30
                },
                {
                    "file_path": "/var/log/secure",
                    "log_group_class": "STANDARD",
                    "log_stream_name": "{instance_id}.secure",
                    "log_group_name": "/PCSLogs/instances",
                    "retention_in_days": 30
                }
            ]
        }
    }
},
"metrics": {
    "aggregation_dimensions": [
        [
            "InstanceId"
        ]
    ],
    "append_dimensions": {
        "AutoScalingGroupName": "${aws:AutoScalingGroupName}",
        "ImageId": "${aws:ImageId}",
```

```
            "InstanceId": "${aws:InstanceId}",
            "InstanceType": "${aws:InstanceType}"
        },
        "metrics_collected": {
            "cpu": {
                "measurement": [
                    "cpu_usage_idle",
                    "cpu_usage_iowait",
                    "cpu_usage_user",
                    "cpu_usage_system"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ],
                "totalcpu": false
            },
            "disk": {
                "measurement": [
                    "used_percent",
                    "inodes_free"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ]
            },
            "diskio": {
                "measurement": [
                    "io_time"
                ],
                "metrics_collection_interval": 60,
                "resources": [
                    "*"
                ]
            },
            "mem": {
                "measurement": [
                    "mem_used_percent"
                ],
                "metrics_collection_interval": 60
            },
            "swap": {
                "measurement": [
```

```
                    "swap_used_percent"
                ],
                "metrics_collection_interval": 60
            }
        }
    }
}
```

이 파일은 CloudWatch 에이전트에 인스턴스 부트스트래핑, 인증 및 로그인, 기타 문제 해결 도메인의 오류를 진단하는 데 도움이 될 수 있는 여러 파일을 모니터링하도록 지시합니다. 다음이 포함됩니다.

- `/var/log/cloud-init.log` - 인스턴스 구성의 초기 단계에서 출력
- `/var/log/cloud-init-output.log` - 인스턴스 구성 중에 실행되는 명령의 출력
- `/var/log/amazon/pcs/bootstrap.log` - 인스턴스 구성 중에 실행되는 PCS별 작업의 출력
- `/var/log/slurmd.log` - Slurm 워크로드 관리자의 데몬 slurmd에서 출력
- `/var/log/messages` - 커널, 시스템 서비스 및 애플리케이션의 시스템 메시지
- `/var/log/secure` - SSH, sudo 및 기타 보안 이벤트와 같은 인증 시도와 관련된 로그

로그 파일은 라는 CloudWatch 로그 그룹으로 전송됩니다`/PCSLogs/instances`. 로그 스트림은 인스턴스 ID와 로그 파일의 기본 이름의 조합입니다. 로그 그룹의 보존 기간은 30일입니다.

또한이 파일은 CloudWatch 에이전트에 여러 가지 공통 지표를 수집하여 인스턴스 ID별로 집계하도록 지시합니다.

구성 저장

CloudWatch 에이전트 구성 파일은 PCS 컴퓨팅 노드 인스턴스에서 액세스할 수 있는 위치에 저장해야 합니다. 이를 수행하는 두 가지 일반적인 방법이 있습니다. 컴퓨팅 노드 그룹 인스턴스가 인스턴스 프로파일을 통해 액세스할 수 있는 Amazon S3 버킷에 업로드할 수 있습니다. 또는 Amazon Systems Manager Parameter Store에 SSM 파라미터로 저장할 수 있습니다.

S3 버킷에 업로드

파일을 S3에 저장하려면 다음 AWS CLI 명령을 사용합니다. 명령을 실행하기 전에 다음을 대체합니다.

- *`amzn-s3-demo-bucket`*을 자체 S3 버킷 이름으로 바꿉니다.

먼저 (기존 버킷이 있는 경우 선택 사항) 구성 파일을 보관할 버킷을 생성합니다.

```
aws s3 mb s3://amzn-s3-demo-bucket
```

그런 다음 파일을 버킷에 업로드합니다.

```
aws s3 cp ./config.json s3://amzn-s3-demo-bucket/
```

SSM 파라미터로 저장

파일을 SSM 파라미터로 저장하려면 다음 명령을 사용합니다. 명령을 실행하기 전에 다음을 대체합니다.

- *region-code*를 AWS PCS로 작업하는 AWS 리전으로 바꿉니다.
- (선택 사항) *AmazonCloudWatch-PCS*를 파라미터의 고유한 이름으로 바꿉니다. 이름의 접두사를에서 변경하는 경우 노드 그룹 인스턴스 프로파일의 SSM 파라미터에 읽기 액세스를 특별히 추가AmazonCloudWatch-해야 합니다.

```
aws ssm put-parameter \
   --region region-code \
   --name "AmazonCloudWatch-PCS" \
   --type String \
   --value file://config.json
```

EC2 시작 템플릿 작성

시작 템플릿의 특정 세부 정보는 구성 파일이 S3에 저장되는지 아니면 SSM에 저장되는지에 따라 달라집니다.

S3에 저장된 구성 사용

이 스크립트는 CloudWatch 에이전트를 설치하고, S3 버킷에서 구성 파일을 가져오고, 이를 사용하여 CloudWatch 에이전트를 시작합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- *amzn-s3-demo-bucket* – 계정이 읽을 수 있는 S3 버킷의 이름입니다.
- */config.json* - 구성이 저장되는 S3 버킷 루트 관련 경로

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="
```

```
--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-cloudwatch-agent

runcmd:
- aws s3 cp s3://amzn-s3-demo-bucket/config.json /etc/s3-cw-config.json
- /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m
 ec2 -s -c file://etc/s3-cw-config.json

--==MYBOUNDARY==--
```

노드 그룹의 IAM 인스턴스 프로파일에는 버킷에 대한 액세스 권한이 있어야 합니다. 다음은 위의 사용자 데이터 스크립트에 있는 버킷에 대한 IAM 정책의 예입니다.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject",
                "s3:ListBucket"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/*"
            ]
        }
    ]
}
```

또한 인스턴스는 S3 및 CloudWatch 엔드포인트로의 아웃바운드 트래픽을 허용해야 합니다. 이는 클러스터 아키텍처에 따라 보안 그룹 또는 VPC 엔드포인트를 사용하여 수행할 수 있습니다.

SSM에 저장된 구성 사용

이 스크립트는 CloudWatch 에이전트를 설치하고, SSM 파라미터에서 구성 파일을 가져오고, 이를 사용하여 CloudWatch 에이전트를 시작합니다. 이 스크립트의 다음 값을 사용자의 세부 정보로 바꿉니다.

- (선택 사항) *AmazonCloudWatch-PCS*를 파라미터의 고유한 이름으로 바꿉니다.

```
MIME-Version: 1.0
Content-Type: multipart/mixed; boundary="==MYBOUNDARY=="

--==MYBOUNDARY==
Content-Type: text/cloud-config; charset="us-ascii"

packages:
- amazon-cloudwatch-agent

runcmd:
- /opt/aws/amazon-cloudwatch-agent/bin/amazon-cloudwatch-agent-ctl -a fetch-config -m
 ec2 -s -c ssm:AmazonCloudWatch-PCS

--==MYBOUNDARY==--
```

노드 그룹의 IAM 인스턴스 정책에는 CloudWatchAgentServerPolicy가 연결되어 있어야 합니다.

파라미터 이름이 로 시작되지 않는 경우 노드 그룹 인스턴스 프로파일의 SSM 파라미터에 읽기 액세스를 특별히 추가AmazonCloudWatch-해야 합니다. 다음은 접두사 *DOC-EXAMPLE-PREFIX*에 대해 이를 설명하는 IAM 정책의 예입니다.

JSON

```
{
  "Version":"2012-10-17",
  "Statement" : [
    {
      "Sid" : "CustomCwSsmMParamReadOnly",
      "Effect" : "Allow",
      "Action" : [
        "ssm:GetParameter"
      ],
      "Resource" : "arn:aws:ssm:*:*:parameter/DOC-EXAMPLE-PREFIX*"
```

```
    }
  ]
}
```

또한 인스턴스는 SSM 및 CloudWatch 엔드포인트로의 아웃바운드 트래픽을 허용해야 합니다. 이는 클러스터 아키텍처에 따라 보안 그룹 또는 VPC 엔드포인트를 사용하여 수행할 수 있습니다.

를 사용하여 AWS 병렬 컴퓨팅 서비스 API 호출 로깅 AWS CloudTrail

AWS PCS는 사용자 AWS CloudTrail, 역할 또는 서비스가 AWS PCS에서 수행한 작업에 대한 레코드를 제공하는 AWS 서비스와 통합됩니다. CloudTrail은 AWS PCS에 대한 모든 API 호출을 이벤트로 캡처합니다. 캡처된 호출에는 AWS PCS 콘솔의 호출과 AWS PCS API 작업에 대한 코드 호출이 포함됩니다. 추적을 생성하면 AWS PCS 이벤트를 포함하여 CloudTrail 이벤트를 Amazon S3 버킷으로 지속적으로 전송할 수 있습니다. 트레일을 구성하지 않은 경우에도 CloudTrail 콘솔의 이벤트 기록에서 최신 이벤트를 볼 수 있습니다. CloudTrail에서 수집한 정보를 사용하여 AWS PCS에 수행된 요청, 요청이 수행된 IP 주소, 요청을 수행한 사람, 요청이 수행된 시간 및 추가 세부 정보를 확인할 수 있습니다.

CloudTrail에 대한 자세한 내용은 AWS CloudTrail 사용 설명서를 참조하세요.

AWS CloudTrail의 PCS 정보

CloudTrail은 계정을 생성할 AWS 계정 때에서 활성화됩니다. AWS PCS에서 활동이 발생하면 해당 활동이 이벤트 기록의 다른 AWS 서비스 이벤트와 함께 CloudTrail 이벤트에 기록됩니다. 에서 최근 이벤트를 보고 검색하고 다운로드할 수 있습니다 AWS 계정. 자세한 내용은 CloudTrail 이벤트 기록을 사용하여 이벤트 보기를 참조하세요.

AWS PCS 이벤트를 AWS 계정포함하여에서 이벤트를 지속적으로 기록하려면 추적을 생성합니다. CloudTrail은 추적을 사용하여 Amazon S3 버킷으로 로그 파일을 전송할 수 있습니다. 콘솔에서 추적을 생성하면 기본적으로 모든 AWS 리전에 추적이 적용됩니다. 추적은 AWS 파티션의 모든 리전에서 이벤트를 로깅하고 지정한 Amazon S3 버킷으로 로그 파일을 전송합니다. 또한 CloudTrail 로그에서 수집된 이벤트 데이터를 추가로 분석하고 조치를 취하도록 다른 AWS 서비스를 구성할 수 있습니다. 자세한 내용은 다음 자료를 참조하세요.

- 추적 생성 개요
- CloudTrail 지원 서비스 및 통합
- CloudTrail에 대한 Amazon SNS 알림 구성

- 여러 리전에서 CloudTrail 로그 파일 받기 및 여러 계정에서 CloudTrail 로그 파일 받기

모든 AWS PCS 작업은 CloudTrail에서 로깅되며 AWS 병렬 컴퓨팅 서비스 API 참조에 문서화됩니다. 예를 들어 `CreateComputeNodeGroup`, `UpdateQueue` 및 `DeleteCluster` 작업을 직접적으로 호출하면 CloudTrail 로그 파일에 항목이 생성됩니다.

모든 이벤트 또는 로그 항목에는 요청을 생성했던 사용자에 대한 정보가 포함됩니다. 자격 증명을 이용하면 다음을 쉽게 판단할 수 있습니다.

- 요청이 루트 또는 AWS Identity and Access Management (IAM) 사용자 자격 증명으로 이루어졌는지 여부입니다.
- 역할 또는 페더레이션 사용자에 대한 임시 보안 인증을 사용하여 요청이 생성되었는지 여부.
- 요청이 다른 AWS 서비스에서 이루어졌는지 여부.

자세한 내용은 CloudTrail userIdentity 요소를 참조하세요.

AWS PCS의 CloudTrail 로그 파일 항목 이해

추적이란 지정한 S3 버킷에 이벤트를 로그 파일로 입력할 수 있도록 하는 구성입니다. CloudTrail 로그 파일에는 하나 이상의 로그 항목이 포함될 수 있습니다. 이벤트는 모든 소스로부터의 단일 요청을 나타내며 요청 작업, 작업 날짜와 시간, 요청 파라미터 등에 대한 정보가 들어 있습니다. CloudTrail 로그 파일은 퍼블릭 API 직접 호출의 주문 스택 트레이스가 아니므로 특정 순서로 표시되지 않습니다.

다음 예제는 `CreateQueue` 작업에 대한 CloudTrail 로그 항목을 보여줍니다.

```
{
    "eventVersion": "1.09",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:admin",
        "arn": "arn:aws:sts::012345678910:assumed-role/Admin/admin",
        "accountId": "012345678910",
        "accessKeyId": "ASIAY36PTPIEXAMPLE",
        "sessionContext": {
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AROAY36PTPIEEXAMPLE",
                "arn": "arn:aws:iam::012345678910:role/Admin",
                "accountId": "012345678910",
                "userName": "Admin"
```

```
            },
            "attributes": {
                "creationDate": "2024-07-16T17:05:51Z",
                "mfaAuthenticated": "false"
            }
        }
    },
    "eventTime": "2024-07-16T17:13:09Z",
    "eventSource": "pcs.amazonaws.com",
    "eventName": "CreateQueue",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "127.0.0.1",
    "userAgent": "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36
 (KHTML, like Gecko) Chrome/126.0.0.0 Safari/537.36",
    "requestParameters": {
        "clientToken": "c13b7baf-2894-42e8-acec-example",
        "clusterIdentifier": "abcdef0123",
        "computeNodeGroupConfigurations": [
            {
                "computeNodeGroupId": "abcdef0123"
            }
        ],
        "queueName": "all"
    },
    "responseElements": {
        "queue": {
            "arn": "arn:aws:pcs:us-east-1:609783872011:cluster/abcdef0123/queue/
abcdef0123",
            "clusterId": "abcdef0123",
            "computeNodeGroupConfigurations": [
                {
                    "computeNodeGroupId": "abcdef0123"
                }
            ],
            "createdAt": "2024-07-16T17:13:09.276069393Z",
            "id": "abcdef0123",
            "modifiedAt": "2024-07-16T17:13:09.276069393Z",
            "name": "all",
            "status": "CREATING"
        }
    },
    "requestID": "a9df46d7-3f6d-43a0-9e3f-example",
    "eventID": "7ab18f88-0040-47f5-8388-example",
    "readOnly": false,
```

```
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "012345678910",
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.3",
        "cipherSuite": "TLS_AES_128_GCM_SHA256",
        "clientProvidedHostHeader": "pcs.us-east-1.amazonaws.com"
    },
    "sessionCredentialFromConsole": "true"
}
```

AWS PCS의 엔드포인트 및 서비스 할당량

다음 섹션에서는 AWS 병렬 컴퓨팅 서비스(AWS PCS)의 엔드포인트 및 서비스 할당량에 대해 설명합니다. 이전에는 제한이라고 했던 서비스 할당량은의 최대 서비스 리소스 또는 작업 수입니다 AWS 계정.

AWS 계정 에는 각 AWS 서비스에 대한 기본 할당량이 있습니다. 다르게 표시되지 않는 한 리전별로 각 할당량이 적용됩니다. 일부 할당량에 대한 증가를 요청할 수 있으며 다른 할당량은 늘릴 수 없습니다.

자세한 내용을 알아보려면 AWS 일반 참조의 AWS 서비스 할당량을 참조하세요.

목차

Service endpoints

리전 이름	리전	엔드포인트	프로토콜
미국 동부(오하이오)	us-east-2	pcs.us-east-2.amazonaws.com pcs-fips.us-east-2.amazonaws.com pcs-fips.us-east-2.api.aws pcs.us-east-2.api.aws	HTTPS
미국 동부(버지니아 북부)	us-east-1	pcs.us-east-1.amazonaws.com pcs-fips.us-east-1.amazonaws.com	HTTPS

리전 이름	리전	엔드포인트	프로토콜
		pcs-fips.us-east-1.api.aws pcs.us-east-1.api.aws	
미국 서부(오레곤)	us-west-2	pcs.us-west-2.amazonaws.com pcs-fips.us-west-2.amazonaws.com pcs-fips.us-west-2.api.aws pcs.us-west-2.api.aws	HTTPS
아시아 태평양(뭄바이)	ap-south-1	pcs.ap-south-1.amazonaws.com pcs.ap-south-1.api.aws	HTTPS
아시아 태평양(싱가포르)	ap-southeast-1	pcs.ap-southeast-1.amazonaws.com pcs.ap-southeast-1.api.aws	HTTPS
아시아 태평양(시드니)	ap-southeast-2	pcs.ap-southeast-2.amazonaws.com pcs.ap-southeast-2.api.aws	HTTPS
아시아 태평양(도쿄)	ap-northeast-1	pcs.ap-northeast-1.amazonaws.com pcs.ap-northeast-1.api.aws	HTTPS

리전 이름	리전	엔드포인트	프로토콜
유럽(프랑크푸르트)	eu-central-1	pcs.eu-central-1.amazonaws.com pcs.eu-central-1.api.aws	HTTPS
유럽(아일랜드)	eu-west-1	pcs.eu-west-1.amazonaws.com pcs.eu-west-1.api.aws	HTTPS
유럽(런던)	eu-west-2	pcs.eu-west-2.amazonaws.com pcs.eu-west-2.api.aws	HTTPS
유럽(파리)	eu-west-3	pcs.eu-west-3.amazonaws.com pcs-eu-west-3.api.aws	HTTPS
유럽(밀라노)	eu-south-1	pcs.eu-south-1.amazonaws.com pcs-eu-south-1.api.aws	HTTPS
유럽(스톡홀름)	eu-north-1	pcs.eu-north-1.amazonaws.com pcs.eu-north-1.api.aws	HTTPS

리전 이름	리전	엔드포인트	프로토콜
AWS GovCloud(미국 동부)	us-gov-east-1	pcs.us-gov-east-1.amazonaws.com pcs-fips.us-gov-east-1.amazonaws.com pcs-fips.us-gov-east-1.api.aws pcs.us-gov-east-1.api.aws	HTTPS
AWS GovCloud(미국 서부)	us-gov-west-1	pcs.us-gov-west-1.amazonaws.com pcs-fips.us-gov-west-1.amazonaws.com pcs-fips.us-gov-west-1.api.aws pcs.us-gov-west-1.api.aws	HTTPS

서비스 할당량

이름	기본값	조정 가능	설명
클러스터	5	예	당 최대 클러스터 수입니다 AWS 리전.

> **Note**
>
> 기본값은에서 설정한 초기 할당량입니다 AWS. 이러한 기본값은 실제 적용된 할당량 값 및 가능한 최대 서비스 할당량과 별개입니다. 자세한 내용은 Service Quotas 사용 설명서의 Service Quotas 용어를 참조하세요.

이러한 서비스 할당량은의 AWS 병렬 컴퓨팅 서비스(PCS) 아래에 나열되어 있습니다AWS Management Console. 조정 가능한 것으로 표시되는 값에 대한 할당량 증가를 요청하려면 Service Quotas 사용 설명서의 할당량 증가 요청을 참조하세요.

> **Important**
>
> 에서 현재 AWS 리전 설정을 확인해야 합니다 AWS Management Console.

내부 할당량

다음 할당량은 내부용이며 조정할 수 없습니다.

이름	기본값	조정 가능	설명
동시 클러스터 생성	1	아니요	AWS 리전당 `Creating` 상태인 클러스터의 최대 수입니다.
클러스터당 컴퓨팅 노드 그룹	10	아니요	클러스터당 최대 컴퓨팅 노드 그룹 수입니다.
클러스터당 대기열	10	아니요	클러스터당 최대 대기열 수입니다.

다른 AWS 서비스에 대한 관련 할당량

AWS PCS는 다른 AWS 서비스를 사용합니다. 이러한 서비스에 대한 서비스 할당량은 AWS PCS 사용에 영향을 미칩니다.

AWS PCS에 영향을 미치는 Amazon EC2 서비스 할당량

- 스팟 인스턴스 요청
- 실행되고 있는 온디맨드 인스턴스
- 시작 템플릿
- 시작 템플릿 버전
- Amazon EC2 API 요청

자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2 서비스 할당량을 참조하세요.

AWS 병렬 컴퓨팅 서비스의 문제 해결

다음 주제에서는 AWS PCS에서 발생할 수 있는 몇 가지 문제를 해결하기 위한 지침을 제공합니다.

재부팅 후 AWS PCS의 EC2 인스턴스가 종료되고 교체됨

문제 개요

컴퓨팅 노드 그룹의 EC2 인스턴스가 재부팅되면 AWS PCS는 인스턴스를 자동으로 종료하고 교체합니다.

이 문제가 발생하는 이유

AWS PCS는 인스턴스 재부팅을 지원하지 않습니다. EC2 인스턴스가 재부팅되면 AWS PCS는 인스턴스를 비정상으로 간주하고 교체합니다. AWS PCS가 지속적으로 인스턴스를 종료하고 교체하는 경우 시작 후 인스턴스를 재부팅하기 때문일 수 있습니다. 예를 들어 EC2 인스턴스에서 자동화를 통한 재부팅(패치 후 자동 재부팅 등), EC2 인스턴스 외부의 자동화(네트워크 관리 애플리케이션 등), 다른 AWS 서비스(예: AWS Systems Manager) 또는 사람의 수동 재부팅이 있습니다.

수행할 작업

`slurmctld` 또는 `slurmd` 로그를 확인하여 인스턴스가 재부팅되었는지 확인할 수 있습니다. 자세한 내용은 AWS PCS의 스케줄러 로그 및 Amazon CloudWatch를 사용하여 AWS PCS 인스턴스 모니터링 섹션을 참조하세요. 다음 예제 `slurmctld` 로그 항목은 인스턴스가 재부팅되었음을 나타냅니다.

Example

```
[2024-09-12T06:42:50.393+00:00] validate_node_specs: Node Login-1 unexpectedly rebooted
 boot_time=1726123354 last response=1726123285
```

패치 적용으로 인한 재부팅

패치를 적용한 후 재부팅이 필요한 경우가 많습니다. AWS PCS 컴퓨팅 노드 그룹의 일부인 EC2 인스턴스에는 패치를 직접 적용하지 마십시오. EC2 인스턴스를 패치해야 하는 경우 업데이트된 Amazon Machine Image(AMI)에 패치를 적용하고 업데이트된 AMI를 사용하도록 컴퓨팅 노드 그룹을 업데이트해야 합니다. 해당 컴퓨팅 노드 그룹에 대해 AWS PCS가 시작하는 새 EC2 인스턴스는 업데이트된(패치된) AMI를 사용합니다. 자세한 내용은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs) 단원을 참조하십시오.

AWS PCS의 컴퓨팅 노드 부트스트랩 및 등록 문제 해결

컴퓨팅 노드가 부트스트랩에 실패하거나 AWS PCS 클러스터에 제대로 등록되지 않으면 다음과 같은 증상이 발생할 수 있습니다.

- 작업이 시작되지 않음
- 에서 인스턴스에 연결할 수 없습니다. AWS Systems Manager
- 인스턴스가 예기치 않게 종료됨
- 인스턴스가 지속적으로 교체됨

이러한 장애는 EC2 인스턴스 시작 중 또는 AWS PCS 컴퓨팅 노드 부트스트랩 프로세스 중에 발생하는 문제로 인해 발생할 수 있습니다. 이 주제에서는 AWS PCS 노드 부트스트랩 프로세스 중에 문제를 해결하는 데 도움이 되는 절차를 설명합니다. EC2 인스턴스 시작 문제 해결에 대한 자세한 내용은 Amazon Elastic Compute Cloud 사용 설명서의 Amazon EC2 인스턴스 시작 문제 해결을 참조하세요.

부트스트랩 실패는 EC2 인스턴스가 성공적으로 시작되지만 AWS PCS 클러스터에 조인하는 과정에서 실패할 때 발생합니다. 부트스트랩 프로세스에는 두 가지 주요 단계가 포함됩니다.

- 노드 등록 - EC2 인스턴스가 RegisterComputeNodeGroupInstance AWS PCS API 작업을 호출하여 AWS PCS 서비스에 등록합니다. 다음과 같은 문제로 인해 장애가 발생할 수 있습니다.
  - 권한
    - 잘못된 인스턴스 프로파일
  - 네트워킹
    - AWS PCS 엔드포인트에 연결할 수 없음
    - 잘못 구성된 AWS PCS 엔드포인트
    - 퍼블릭 IP가 없는 퍼블릭 서브넷의 인스턴스

    - 퍼블릭 서브넷의 다중 NIC 인스턴스
  - 클러스터 보안 암호
    - 클러스터 보안 암호가 삭제되었거나 삭제 대상으로 표시되었습니다.
- Slurm 통합 - 인스턴스가 Slurm 클러스터를 실행`slurmd`하고 조인합니다. 다음과 같은 문제로 인해 장애가 발생할 수 있습니다.
  - 권한
    - 보안 그룹 구성
    - Slurmctld가 컴퓨팅 노드를 ping할 수 없음
  - 사용자 지정 AMI 설정
    - NVIDIA 드라이버 누락
    - ResumeTimeout에 도달함

Slurm이 AWS PCS에서 작동하는 방식

이는 Slurm이 작동하는 표준 방식과 Slurm이 AWS PCS에서 작동하는 방식을 비교하는 데 도움이 될 수 있습니다.

표준 Slurm 작업 처리

다음 단계는 표준 Slurm 작업 처리에서 수행됩니다.

1. 작업을 제출하면가 작업을 `slurmctld` 검증하고 대기열에 넣습니다.
2. 리소스를 사용할 수 있게 되면가 기존 노드를 `slurmctld` 할당합니다.
3. `slurmd` 데몬은 할당된 노드에서 작업을 실행합니다.

AWS PCS에서 Slurm 작업 처리

AWS PCS 작업 처리에서 다음 단계가 수행됩니다.

1. 작업을 제출하면가 작업을 `slurmctld` 검증하고 대기열에 넣습니다.
2. 추가 용량이 필요한 경우 AWS PCS는 컴퓨팅 노드 그룹에 대한 시작 템플릿을 사용하여 새 EC2 인스턴스를 시작합니다.
3. 새 인스턴스가 클러스터로 부트스트랩됩니다.
   a. 인스턴스는 AWS PCS에 등록됩니다.

b. 인스턴스는 Slurm 클러스터에 조인합니다.

4. 리소스가 준비되면는 노드(새로 부트스트랩된 노드 포함)를 `slurmctld` 할당합니다.

5. `slurmd` 데몬은 할당된 노드에서 작업을 실행합니다.

인스턴스 로그 검색

컴퓨팅 노드 부트스트랩 문제를 해결하는 첫 번째 단계는 인스턴스 로그를 검색하는 것입니다. 다음 방법 중 하나를 사용할 수 있습니다.

AWS CLI

다음 명령을 사용하여 컴퓨팅 노드에서 콘솔 출력을 검색합니다.

```
aws ec2 get-console-output --region us-east-1 --instance-id i-1234567890abcdef0 --
output text
```

*us-east-1*을 AWS 리전으로 바꾸고 *i-1234567890abcdef0*을 인스턴스 ID로 바꿉니다.

AWS Systems Manager

Systems Manager를 사용하여 인스턴스에 연결할 수 있는 경우 부트스트랩 로그 파일을 직접 볼 수 있습니다.

1. Systems Manager를 사용하여 인스턴스에 연결합니다. 자세한 내용은 Systems Manager 사용 설명서의 세션 시작을 참조하세요.
2. 부트스트랩 로그 파일 보기:

```
sudo cat /var/log/amazon/pcs/bootstrap.log
```

> **Note**
>
> 초기화 단계에서 문제가 있는 경우 인스턴스에 연결하기 전에 약 20분을 기다려야 할 수 있습니다. Systems Manager 및 SSH 서비스는 초기화가 완료된 후 또는 실패 시 부트스트랩 실행이 제한 시간에 도달한 경우에만 시작됩니다.

인스턴스 ID에서 VPC/Subnet/Security 그룹 검색

컴퓨팅 노드 문제를 해결하려면 인스턴스와 연결된 VPC, 서브넷 및 보안 그룹에 대한 정보를 검색해야 할 수 있습니다. 인스턴스 IDs 모르는 경우 섹션을 참조하세요AWS PCS에서 컴퓨팅 노드 그룹 인스턴스 찾기.

AWS Management Console

VPC, 서브넷 및 보안 그룹을 가져오는 방법

1. Amazon EC2 콘솔을 엽니다.
2. 인스턴스를 선택합니다.
3. 인스턴스 테이블에서 인스턴스 ID를 선택합니다.
4. 인스턴스에 대해 표시된 인스턴스 요약에서 VPC ID 및 서브넷 ID를 찾습니다.
5. 인스턴스 요약에서 보안 탭을 선택합니다.
6. 보안 탭에서 보안 그룹을 찾습니다.

AWS CLI

다음 명령을 사용하여 인스턴스에 대한 VPC, 서브넷 및 보안 그룹 정보를 검색합니다.

```
aws ec2 describe-instances --instance-ids i-1234567890abcdef0 --query
 'Reservations[*].Instances[*].
{InstanceId:InstanceId,VpcId:VpcId,SubnetId:SubnetId,SecurityGroups:SecurityGroups[*].GroupI
 --output table
```

노드 등록 문제

노드 등록은 부트스트랩 중에 컴퓨팅 노드가 실행하는 첫 번째 작업입니다. 노드는 AWS PCS API 엔드포인트를 호출하여 자체적으로 AWS PCS에 등록합니다. 등록 실패는 일반적으로 다음과 유사한 오류 메시지를 표시합니다.

```
<13>Nov 13 16:23:50 user-data: [2025-11-13T16:23:50.510+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Registering node to cluster <clusterId>
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.192+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Retriable exception detected.
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.193+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Response is [specific error message]
```

```
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.194+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Retrying in 31 seconds...
<13>Nov 13 16:24:18 user-data: [2025-11-13T16:24:18.192+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Retriable exception detected.
...
<13>Nov 13 16:25:18 user-data: [2025-11-13T16:25:18.195+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Registration timeout (600 seconds) reached. Exiting.
<13>Nov 13 16:25:18 user-data: [2025-11-13T16:25:18.200+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: ERROR: Error: (2) occurred on line 1 when running /opt/aws/pcs/
bin/pcs_bootstrap_init.sh. Shutting down instance.
```

잘못된 인스턴스 프로파일

잘못된 인스턴스 프로파일로 인해 노드를 등록할 수 없는 경우 다음 오류가 표시됩니다.

```
<13>Nov 13 18:43:08 user-data: [2025-11-13T18:43:08.268+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_init.sh: INFO: Response is {
<13>Nov 13 18:43:08 user-data:   "__type":
 "com.amazon.coral.service#AccessDeniedException",
<13>Nov 13 18:43:08 user-data:   "Message": "User: arn:aws:sts::<accountId>:assumed-
role/<roleName>/<instanceId> is not authorized to perform:
 pcs:RegisterComputeNodeGroupInstance on resource:
 arn:aws:pcs:<regionCode>:<accountId>:cluster/<clusterId> as either the resource does
 not exist, some policy explicitly denies access, or no policy grants access",
<13>Nov 13 18:43:08 user-data:   "nodeID": null
<13>Nov 13 18:43:08 user-data: }
```

컴퓨팅 노드와 연결된 인스턴스 프로파일에 `pcs:RegisterComputeNodeGroupInstance` 권한이 있는지 확인합니다. 유효한 인스턴스 프로파일을 생성하는 방법에 대한 자세한 내용은 섹션을 참조하세요AWS PCS용 인스턴스 프로파일 생성.

AWS PCS 엔드포인트에 연결할 수 없음

컴퓨팅 노드가 프라이빗 서브넷에 있는 경우 AWS PCS에 대한 VPC 엔드포인트를 구성했는지 또는 서브넷에 인터넷 액세스를 위한 NAT 게이트웨이에 대한 경로가 있는지 확인합니다. 자세한 내용은 다음을 참조하세요.

- Amazon Virtual Private Cloud AWS PrivateLink 가이드의 인터페이스 VPC 엔드포인트를 사용하여 AWS 서비스에 액세스합니다.
- AWS PCS의 엔드포인트 및 서비스 할당량.
- Amazon Virtual Private Cloud 사용 설명서의 다른 네트워크에 VPC 연결

- AWS PCS 네트워킹

잘못 구성된 AWS PCS 엔드포인트

다음과 유사한 오류 메시지가 표시되면 AWS PCS VPC 엔드포인트와 연결된 정책을 확인합니다.

```
com.amazon.coral.security.AccessDeniedException: User: arn:aws:sts::xxx:assumed-
role/<roleName>/<instanceId> is not authorized to perform:
 pcs:RegisterComputeNodeGroupInstance on resource:
 arn:aws:pcs:<regionCode>:<accountId>:cluster/<clusterId> as either the resource does
 not exist, some policy explicitly denies access, or no policy grants access
```

AWS PCS용 VPC 인터페이스 엔드포인트를 구성하는 방법에 대한 자세한 내용은 섹션을 참조하세요인터페이스 엔드포인트를 AWS Parallel Computing Service 사용한 액세스(AWS PrivateLink).

퍼블릭 IP가 없는 퍼블릭 서브넷의 인스턴스

서브넷에 퍼블릭 IP 자동 할당이 활성화되어 있지 않고 라우팅 구성이 인터넷 게이트웨이를 사용하는 경우 인스턴스가 AWS PCS API와 통신할 수 없습니다.

인터넷 게이트웨이가 있는 서브넷의 인스턴스에는 퍼블릭 IP 주소가 있어야 합니다. 이 문제를 해결하려면 다음 옵션 중 하나를 선택합니다.

- 클러스터 VPC에 AWS PCS용 VPC 엔드포인트를 추가합니다. 이렇게 하면 퍼블릭 IP 주소가 인터넷 게이트웨이를 통과할 필요 없이 인스턴스가 AWS PCS와 통신할 수 있습니다.
- 퍼블릭 IP 주소가 필요하지 않도록 NAT 게이트웨이와 함께 프라이빗 서브넷을 사용합니다.
- 인스턴스가 인터넷 게이트웨이를 통해 API에 연결할 수 있도록 서브넷 또는 시작 템플릿을 통해 자동 퍼블릭 IP 주소 할당을 활성화합니다. 이 옵션은 다중 네트워크 인터페이스 인스턴스에는 유효하지 않습니다.

퍼블릭 서브넷의 다중 NIC 인스턴스

여러 네트워크 인터페이스(NICs)가 있는 인스턴스 유형을 사용하는 경우 프라이빗 서브넷을 사용해야 합니다.

AWS 퍼블릭 IP 주소는 단일 네트워크 인터페이스로 시작된 인스턴스에만 할당할 수 있습니다. IP 주소에 대한 자세한 내용은 Linux 인스턴스용 Amazon EC2 사용 설명서의 인스턴스 시작 중 퍼블릭 IPv4 주소 할당을 참조하세요. Amazon EC2

다중 NIC 인스턴스 유형은 AWS PCS 엔드포인트에 액세스하려면 서브넷의 NAT 게이트웨이 또는 내부 프록시가 필요합니다. 또는 클러스터 VPC에 AWS PCS용 VPC 엔드포인트를 추가할 수 있습니다.

클러스터 보안 암호가 삭제되었거나 삭제 대상으로 표시되었습니다.

AWS Secrets Manager의 Slurm 공유 보안 암호가 삭제되었거나 삭제 대상으로 표시된 경우 컴퓨팅 노드가 등록되지 않고 클러스터가 손상됩니다.

AWS 클러스터를 생성할 때 PCS는 AWS Secrets Manager에서 Slurm 공유 보안 암호를 자동으로 생성합니다(이름 형식: `pcs!slurm-secret-<cluster-id>`). 이 보안 암호는 클러스터의 보안 통신에 필요합니다. 자세한 내용은 AWS PCS에서 클러스터 보안 암호 작업 단원을 참조하십시오.

이 보안 암호가 삭제되거나 삭제 표시되면 새 노드가 클러스터에 조인할 수 없으며 다시 시작하면 컨트롤러 또는 다른 클러스터 데몬(예: `slurmd` 및 `slurmdbd`)이 클러스터에 다시 조인할 수 없습니다.

이 문제를 해결하려면 삭제된 보안 암호가 복구 기간 내에 있는 경우 복원할 수 있습니다. 자세한 지침은 AWS Secrets Manager 보안 암호 복원을 참조하세요.

복구 기간이 만료되면 보안 암호를 복원할 수 없으며 영향을 받는 AWS PCS 클러스터를 복원할 수 없습니다. 동일한 구성으로 새 클러스터를 생성해야 합니다. AWS PCS는 자동으로 새 스케줄러 보안 암호를 생성합니다.

Slurm 클러스터 조인 문제

노드 등록에 성공하면 컴퓨팅 노드가 Slurm 클러스터에 조인을 시도합니다. 노드의 `slurmd` 데몬은 클러스터에 등록하기 위해 Slurm 컨트롤러에 접촉합니다. Slurm 조인 실패는 일반적으로 다음과 유사한 오류 메시지를 표시합니다.

```
<13>Nov  5 17:20:29 user-data: [2024-11-05T17:20:28+00:00] FATAL:
 Mixlib::ShellOut::ShellCommandFailed: service[slurmd] (aws-pcs-slurm::finalize_slurm
 line 18) had an error: Mixlib::ShellOut::ShellCommandFailed: Expected process to exit
 with [0], but received '1'
<13>Nov  5 17:20:29 user-data: ---- Begin output of ["/usr/bin/systemctl", "--system",
 "start", "slurmd"] ----
<13>Nov  5 17:20:29 user-data: STDOUT:
<13>Nov  5 17:20:29 user-data: STDERR: Job for slurmd.service failed because the
 control process exited with error code. See "systemctl status slurmd.service" and
 "journalctl -xe" for details.
<13>Nov  5 17:20:29 user-data: ---- End output of ["/usr/bin/systemctl", "--system",
 "start", "slurmd"] ----
```

보안 그룹 구성

컴퓨팅 노드와 Slurm 컨트롤러 간의 통신을 허용하도록 보안 그룹이 올바르게 구성되어 있는지 확인합니다. 보안 그룹은 다음 트래픽을 허용해야 합니다.

- 가 통신`slurmd`할 포트 6817 `slurmctld`
- 용 포트 6818에서 ping`slurmctld`으로 `slurmd`

보안 그룹 요구 사항에 대한 자세한 내용은 다음 주제를 참조하세요.

- AWS PCS에 대한 보안 그룹 생성
- AWS PCS용 시작 템플릿 생성
- 보안 그룹 요구 사항 및 고려 사항

> **Important**
>
> 클러스터 생성 중에 클러스터와 연결한 클러스터 보안 그룹도 컴퓨팅 노드가 컨트롤러와 통신할 수 있도록 컴퓨팅 노드 그룹 보안 그룹에 구성해야 합니다.

NVIDIA 드라이버 누락

인스턴스가 올바르게 부트스트랩되지만 작업이 시작되지 않고 인스턴스 로그에 다음과 유사한 오류 메시지가 표시되면 NVIDIA 드라이버가 누락되었을 수 있습니다.

```
<13>Dec  2 13:52:00 user-data: [2024-12-02T13:52:00.094+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_config_always.sh: INFO: nvidia-smi not found!
...
<13>Dec  2 13:54:10 user-data: Job for slurmd.service failed because the control
 process exited with error code. See "systemctl status slurmd.service" and "journalctl
 -xe" for details.
<13>Dec  2 13:54:12 user-data: [2024-12-02T13:54:12.718+00:00] - /opt/aws/pcs/bin/
pcs_bootstrap_finalize.sh: INFO: systemctl could not start slurmd!
```

인스턴스에 연결하고 `slurmd` 데몬 상태를 확인하는 경우 다음과 유사한 오류가 표시될 수 있습니다.

```
$ systemctl status slurmd
...
```

```
fatal: can't stat gres.conf file /dev/nvidia0: No such file or directory
```

이 문제를 해결하려면 사용자 지정 AMI에 NVIDIA 드라이버를 설치합니다. 자세한 내용은 4단계 - (선택 사항) 추가 드라이버, 라이브러리 및 애플리케이션 소프트웨어 설치 단원을 참조하십시오.

ResumeTimeout에 도달함

노드가 비정상으로 인해 컴퓨팅 노드와 해당 EC2 인스턴스가 종료되면 AWS PCS가 AMI를 지원하지 않거나 네트워크 문제가 있을 수 있습니다. EC2 인스턴스는 Slurm의 ResumeTimeout에 도달할 때까지 약 30분 동안 실행되고 노드를 로 표시합니다DOWN.

인스턴스가 올바르게 부트스트랩되지 않고 AWS PCS에 등록되지 않은 경우(EC2 인스턴스에 대한 RegisterComputeNodeGroupInstance 호출 없음) 인스턴스 로그에 다음과 유사한 오류 메시지가 있는지 확인합니다.

```
/opt/aws/pcs/bin/pcs_bootstrap_init.sh: No such file or directory
```

이 오류는 AWS PCS 부트스트랩 소프트웨어가 AMI의 일부가 아님을 나타냅니다. 이 문제를 해결하려면 사용자 지정 AMI에 AWS PCS 부트스트랩 소프트웨어가 포함되어 있는지 확인합니다. 자세한 내용은 AWS PCS용 사용자 지정 Amazon Machine Image(AMIs) 단원을 참조하십시오.

Slurmctld가 컴퓨팅 노드를 ping할 수 없음

인스턴스가 부트스트랩 절차를 올바르게 실행하고 AWS PCS에 등록되었지만 인스턴스를 보고 작업을 제출할 slurmctld 수 없는 경우 인스턴스는 잠시 DOWN 후에 로 설정된 다음 종료됩니다.

이는 잘못 구성된 보안 그룹으로 인해 발생할 수 있습니다. 예를 들어 포트 6817이와 slurmd 통신할 수 있도록 활성화되어 slurmctld있지만 포트 6818이 누락되어에 pingslurmctld을 허용합니다slurmd.

에 설명된 대로 보안 그룹에 필요한 모든 규칙이 포함되어 있는지 확인합니다보안 그룹 요구 사항 및 고려 사항.

MaxJobCount 제한으로 인한 작업 제출 실패 문제 해결

문제: 다음 오류 메시지와 함께 작업 제출이 실패합니다.

```
sbatch: error: Slurm temporarily unable to accept job, sleeping and retrying
```

이 오류는 실행 중인 작업과 보류 중인 작업 수가 클러스터의 작업 한도보다 훨씬 낮은 것으로 보이는 경우에도 발생합니다.

원인: `MaxJobCount` 제한에는 실행 중이거나 보류 중인 작업뿐만 아니라 Slurm에서 추적하는 모든 작업이 포함됩니다. 완료된 작업은 제거되기 전에 일정 기간(기본적으로 5분) 동안 Slurm의 메모리에 남아 있습니다. 작업 처리량이 많은 기간에는 활성 작업과 최근에 완료된 작업의 총 수가 한도를 초과할 수 있습니다.

클러스터 노드에서 다음 명령을 실행하여 총 작업 수를 확인할 수 있습니다.

```
scontrol show jobs | grep -c JobId
```

여기에는 제거 대기 중인 완료된 작업을 포함하여 Slurm이 추적 중인 총 작업 수가 표시됩니다.

해결 방법: 다음 방법 중 하나를 고려합니다.

- 더 큰 클러스터 생성 - 워크로드에 지속적으로 더 많은 동시 작업이 필요한 경우 크기가 더 큰 새 클러스터를 생성합니다. 클러스터 크기 및 제한에 대한 자세한 내용은 섹션을 참조하세요AWS PCS의 클러스터 크기.
- 작업 제출률 감소 - 작업 제출 스크립트를 조정하여 더 느린 속도로 작업을 제출하므로 완료된 작업 시간을 Slurm의 추적에서 제거할 수 있습니다.

AWS PCS 사용 설명서의 문서 기록

다음 표에서는 AWS PCS 설명서의 중요한 변경 사항에 대해 설명합니다.

Date	변경	설명서 업데이트	API 버전 업데이트됨
2026년 4월 16일	AWS 유럽(밀라노)에서 릴리스된 PCS	AWS 이제 유럽(밀라노)(eu-south-1)에서 PCS를 사용할 수 있습니다. CloudFormation 템플릿은 유럽(밀라노)에서 시작할 수 있습니다 AWS 리전. 자세한 내용은 CloudFormation 를 사용하여 샘플 AWS PCS 클러스터 생성 및 CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿 섹션을 참조하세요.	해당 사항 없음
2026년 3월 10일	업데이트된 PCS 에이전트	AWS PCS 에이전트 1.3.2-1에 대한 AMI 주제를 업데이트했습니다. RHEL 8.10 및 Rocky Linux 8.10 컴퓨팅 노드 부트스트랩에 영향을 미치는 문제를 수정했습니다. 자세한 내용은 AWS PCS 용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 AWS PCS 에이전트 버전 섹션을 참조하세요.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2026년 2월 11일	AWS 아시아 태평양(뭄바이) 및 유럽(파리)에서 릴리스된 PCS	AWS 이제 아시아 태평양(뭄바이)(ap-south-1) 및 유럽(파리)(eu-west-3)에서 PCS를 사용할 수 있습니다. CloudFormation 템플릿은 아시아 태평양(뭄바이) AWS 리전 및 유럽(파리)에서 시작할 수 있습니다 AWS 리전. 자세한 내용은 CloudFormation 를 사용하여 샘플 AWS PCS 클러스터 생성 및 CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿 섹션을 참조하세요.	해당 사항 없음
2025년 11월 18일	새로운 기능: Slurm REST API	이제 Slurm REST API가 Slurm 25.05 이상에서 지원됩니다. 자세한 내용은 AWS PCS의 Slurm REST API 단원을 참조하십시오.	AWS SDK: 2025-11-18
2025년 11월 17일	Slurm 설치 관리자 업데이트	AWS PCS Slurm 설치 관리자 24.11.7-1 및 25.05.5-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 AWS PCS 에이전트 버전 섹션을 참조하세요.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 11월 10일	새로운 기능: Slurm CLI 필터 플러그인 지원	AWS 이제 PCS는 Slurm CLI 필터 플러그인을 지원하여 Slurm 컨트롤러에 도달하기 전에 작업 제출 파라미터를 검증하고 수정하는 사용자 지정 Lua 스크립트를 실행합니다. CLI 필터를 사용하여 사용자 지정 정책을 적용하고, 기본 파라미터를 설정하고, 작업 제출 중에 사용자 지침을 제공합니다. 이 기능을 사용하려면 Slurm 버전 25.05 이상이 필요합니다. 자세한 내용은 Slurm CLI 필터 플러그인을 사용하여 AWS PCS에서 작업 제출 사용자 지정 단원을 참조하십시오.	해당 사항 없음
2025년 11월 7일	업데이트된 PCS 에이전트	AWS PCS 에이전트 1.3.1-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 AWS PCS 에이전트 버전 섹션을 참조하세요.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 11월 3일	업데이트된 PCS 에이전트 및 Slurm 설치 관리자	AWS PCS 에이전트 1.3.0-1 및 Slurm 설치 관리자 24.11.6-2, 24.05.8-2 및 23.11.10-4에 대한 AMI 주제를 업데이트했습니다. 지원되는 운영 체제 목록을 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 AWS PCS 에이전트 버전 섹션을 참조하세요.	해당 사항 없음
2025년 10월 23일	업데이트된 콘텐츠: pcs-multi-cluster-login-configure.sh	다중 클러스터 로그인 노드 구성 스크립트의 일부 오류를 수정했습니다. 자세한 내용은 AWS PCS 다중 클러스터 로그인 노드 구성 스크립트 코드 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 10월 21일	새로운 기능: 클러스터 보안 암호 교체	AWS 이제 PCS가 클러스터 보안 암호 교체를 지원하여 보안을 강화합니다. 자세한 내용은 AWS PCS에서 클러스터 보안 암호 교체 단원을 참조하십시오. 클러스터 보안 암호 교체를 지원하도록 최소 관리자 권한을 업데이트했습니다. 자세한 내용은 AWS PCS에 대한 최소 권한 단원을 참조하십시오.	해당 사항 없음
2025년 10월 17일	새 주제: 다중 클러스터 로그인 노드 구성 스크립트	여러 AWS PCS 클러스터에 연결하도록 독립 실행형 로그인 노드를 구성하는 스크립트를 제공하는 새 주제가 추가되었습니다. 스크립트는 여러 Slurm sackd 데몬의 구성을 자동화하고 클러스터 상호 작용을 위한 활성화 스크립트를 생성합니다. 자세한 내용은 독립 실행형 로그인 노드를 AWS PCS의 여러 클러스터에 연결 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 10월 16일	Slurm 25.05 업데이트	Slurm 25.05 지원을 위한 사용 설명서가 업데이트되었습니다. 이제 Slurm 25.05가 기본 버전입니다. 자세한 내용은 다음을 참조하세요. • AWS PCS의 Slurm 버전 • AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 • AWS PCS 샘플 AMIs 릴리스 정보	해당 사항 없음
2025년 10월 16일	업데이트된 PCS 에이전트	AWS PCS 에이전트 1.2.2-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 AWS PCS 에이전트 버전 섹션을 참조하세요.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 10월 2일	새로운 기능: Slurm 노드 재부팅, 클러스터 업데이트 및 사용자 지정 Slurm 설정	AWS PCS는 여러 가지 새로운 기능에 대한 지원을 추가합니다. • Slurm 노드 재부팅 - Slurm의 기본 `scontrol reboot` 명령을 사용하여 인스턴스 교체 없이 컴퓨팅 노드를 재부팅합니다. 자세한 내용은 AWS PCS에서 Slurm을 사용하여 컴퓨팅 노드 재부팅 단원을 참조하십시오. • 클러스터 업데이트 - 재구축 없이 생성 후 클러스터 구성을 수정합니다. 자세한 내용은 AWS PCS에서 클러스터 업데이트 단원을 참조하십시오. • Slurm 사용자 지정 설정 - 클러스터, 대기열 및 컴퓨팅 노드 그룹 리소스에서 고급 Slurm 파라미터를 구성합니다. 자세한 내용은 AWS PCS에서 사용자 지정 Slurm 설정 구성 단원을 참조하십시오.	2025-10-01

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 9월 23일	새로운 문제 해결 주제: 컴퓨팅 노드 부트스트랩 문제	컴퓨팅 노드 부트스트랩 문제를 진단하고 해결하기 위한 문제 해결 지침이 추가되었습니다. 자세한 내용은 AWS PCS의 컴퓨팅 노드 부트스트랩 및 등록 문제 해결 단원을 참조하십시오.	해당 사항 없음
2025년 9월 17일	새로운 기능: ML용 용량 블록	AWS 이제 PCS는 클러스터에 대한 GPU 기반 가속 컴퓨팅 인스턴스를 예약할 수 있는 ML용 Amazon EC2 용량 블록을 지원합니다. 자세한 내용은 AWS PCS에서 ML에 Amazon EC2 용량 블록 사용 단원을 참조하십시오. 용량 블록을 지원하는 최소 권한은 이제 서비스 관리자의 최소 권한에 포함됩니다. 자세한 내용은 AWS PCS에 대한 최소 권한 단원을 참조하십시오.	2025-09-17
2025년 9월 11일	AWS 관리형 정책 업데이트	AWS PCS는 용량 블록을 지원하도록 AWSPCSServiceRolePolicy를 업데이트했습니다. 자세한 내용은 AWS AWS 병렬 컴퓨팅 서비스에 대한 관리형 정책 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 8월 14일	업데이트된 인스턴스 프로파일 설명서	IAM 역할 및 인스턴스 프로파일을 생성하기 위한 포괄적인 CLI 지침으로 인스턴스 프로파일 설명서를 개선했습니다. 를 사용하여 AWS CLI 인스턴스 프로파일을 설정하는 step-by-step 절차를 추가하고 AWS PCS에 사용되는 인스턴스 프로파일을 찾기 위한 지침을 개선했습니다. 자세한 내용은 AWS 병렬 컴퓨팅 서비스를 위한 IAM 인스턴스 프로파일 단원을 참조하십시오.	2025-08-14
2025년 8월 1일	새 주제: SPANK 플러그인	AWS PCS 클러스터에서 작업 시작 및 실행 중에 Slurm의 동작을 확장하고 수정하는 데 사용할 수 있는 SPANK(Slurm Plug-in Architecture for Node and job Kontrol) 플러그인에 대한 설명서가 추가되었습니다. 자세한 정보는 SPANK 플러그인을 사용하여 AWS PCS에서 Slurm 기능 확장을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 8월 1일	IPv6 네트워킹 지원	AWS PCS 클러스터를 생성할 때 IPv6 네트워킹에 대한 지원이 추가되었습니다. 이제 VPC 요구 사항, 서브넷 구성, 보안 그룹 설정 및 클러스터 생성 절차에 대한 해당 업데이트와 함께 IPv6를 클러스터의 네트워크 유형으로 선택할 수 있습니다. 자세한 내용은 AWS PCS VPC 및 서브넷 요구 사항 및 고려 사항 및 AWS PCS에서 클러스터 생성 섹션을 참조하세요.	2025-08-01
2025년 7월 3일	AWS 유럽(런던)에서 릴리스된 PCS	AWS 이제 유럽(런던)(eu-west-2)에서 PCS를 사용할 수 있습니다. CloudFormation 템플릿은 유럽(런던)에서 시작할 수 있습니다 AWS 리전. 자세한 내용은 CloudFormation 를 사용하여 샘플 AWS PCS 클러스터 생성 및 CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿 섹션을 참조하세요.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 7월 1일	콘솔 지침 업데이트	이제 콘솔에서 클러스터 및 컴퓨팅 노드 그룹을 생성할 때 AWS PCS가 기본 인스턴스 프로파일 및 보안 그룹을 생성하도록 할 수 있습니다. 자세한 내용은 다음을 참조하세요. • AWS PCS에서 클러스터 생성 • AWS PCS에서 컴퓨팅 노드 그룹 생성 • AWS 병렬 컴퓨팅 서비스를 위한 IAM 인스턴스 프로파일	해당 사항 없음
2025년 6월 23일	새로운 관리형 정책: AWSPCSComputeNodePolicy	AWS PCS 클러스터에 연결할 수 있는 권한을 AWS PCS 컴퓨팅 노드에 부여하는 새로운 관리형 정책이 추가되었습니다. 자세한 정보는 AWS 관리형 정책: AWSPCSComputeNodePolicy을 참조하십시오.	해당 사항 없음
2025년 6월 19일	새 주제: 작업 완료 로그	작업 완료 로그를 사용하여 작업이 완료되면 추가 비용 없이 작업에 대한 세부 정보를 기록합니다. 자세한 내용은 AWS PCS의 작업 완료 로그 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 6월 18일	AWS 의 PCS 릴리스 AWS GovCloud (US)	AWS 이제 PCS를 AWS GovCloud(미국 동부) (us-gov-east-1) 및 AWS GovCloud(미국 서부)(us-gov-west-1)에서 사용할 수 있습니다. CloudFormation 템플릿은에서 시작할 수 있습니다 AWS GovCloud (US) Regions. 자세한 내용은 CloudFormation 를 사용하여 샘플 AWS PCS 클러스터 생성 및 CloudFormation 샘플 AWS PCS 클러스터를 생성하기 위한 템플릿 섹션을 참조하세요. 의 AWS PCS 서비스 엔드포인트에 대한 자세한 내용은 섹션을 AWS GovCloud (US) Regions 참조하세요AWS PCS의 엔드포인트 및 서비스 할당량. 의 차이점에 대한 자세한 내용은 AWS GovCloud (US) 사용 설명서의 AWS 의 PCS AWS GovCloud (US)를 AWS GovCloud (US) Regions참조하세요.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 6월 18일	업데이트된 PCS 에이전트	AWS PCS 에이전트 1.2.1-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 단원을 참조하십시오.	해당 사항 없음
2025년 5월 15일	새로운 기능: 회계	이제 Slurm 24.11 이상에서 Slurm 회계가 지원됩니다. 자세한 내용은 AWS PCS의 Slurm 회계 단원을 참조하십시오.	AWS SDK: 2025-05-15
2025년 5월 15일	Slurm 24.11 업데이트	Slurm 24.11.5 지원에 대한 사용 설명서가 업데이트되었습니다. 자세한 내용은 다음을 참조하세요. • AWS PCS의 Slurm 버전 • AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 • AWS PCS 샘플 AMIs 릴리스 정보	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 5월 5일	업데이트된 Slurm 버전 FAQ	수명 종료(EOL) 전후의 Slurm 버전에 대한 Slurm 버전 FAQ(자주 묻는 질문)를 업데이트했습니다. 자세한 내용은 AWS PCS의 Slurm 버전에 대해 자주 묻는 질문 단원을 참조하십시오.	해당 사항 없음
2025년 4월 17일	새 주제: 컴퓨팅 노드 그룹 세부 정보를 가져오는 방법	ID, ARN 및 AMI ID와 같은 AWS PCS 컴퓨팅 노드 그룹에 대한 세부 정보를 가져오는 방법을 알아봅니다. 자세한 내용은 AWS PCS에서 컴퓨팅 노드 그룹 세부 정보 가져오기 단원을 참조하십시오.	해당 사항 없음
2025년 4월 2일	Slurm 설치 관리자 업데이트	Slurm 설치 관리자 24.05.7-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 단원을 참조하십시오.	해당 사항 없음
2025년 3월 28일	최대 컴퓨팅 노드 그룹 및 대기열 수에 대한 할당량 추가	클러스터당 최대 컴퓨팅 노드 그룹 수와 클러스터당 최대 대기열 수에 대한 조정 불가능한 내부 할당량이 추가되었습니다. 자세한 내용은 내부 할당량 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 3월 14일	CloudFormation 템플릿에서 속성 키 변경	`Id`는 이제 CloudFormation 템플릿의 `CustomLaunchTemplate` 속성에 `TemplateId` 대한 것입니다. 자세한 설명은 AWS PCS용 CloudFormation 템플릿의 일부에서 리소스 섹션을 참조하세요.	해당 사항 없음
2025년 3월 13일	AWS PCS 에이전트 및 Slurm에 대한 버전 정보 추가	각 버전의 AWS PCS 에이전트에 대한 변경 사항을 설명하는 새 주제가 추가되었습니다. 자세한 내용은 AWS PCS 에이전트 버전 단원을 참조하십시오. Slurm 버전 주제에 Slurm용 AWS PCS 지원에 대한 중요한 지원 날짜와 자세한 릴리스 정보를 설명하는 추가 정보가 추가되었습니다. 자세한 내용은 AWS PCS의 Slurm 버전 단원을 참조하십시오.	해당 사항 없음
2025년 3월 7일	업데이트된 PCS 에이전트	AWS PCS 에이전트 1.2.0-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2025년 2월 3일	AWS CloudFormation AWS PCS에서 사용에 대한 주제 추가	AWS PCS와 CloudFormation 함께를 사용하는 방법의 예를 제공하는 주제를 사용 설명서에 추가했습니다. 이 주제에서는 샘플 CloudFormation 템플릿을 사용하여 샘플 AWS PCS 클러스터를 생성하는 절차를 제공하고 해당 템플릿의 섹션을 간략하게 설명합니다. 자세한 내용은 CloudFormation 및 AWS PCS 시작하기 단원을 참조하십시오.	해당 사항 없음
2024년 12월 18일	Slurm 24.05 업데이트	Slurm 24.05 지원을 위한 사용 설명서가 업데이트되었습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 AWS PCS 샘플 AMIs 릴리스 정보 섹션을 참조하세요.	해당 사항 없음
2024년 12월 18일	Slurm 23.11 샘플 AMIs용 NVIDIA 버전 업데이트	Slurm 23.11 샘플 AMIs. 자세한 내용은 AWS PCS 샘플 AMIs 릴리스 정보 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2024년 12월 17일	Slurm 설치 관리자 업데이트	Slurm 설치 관리자 23.11.10-3에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 단원을 참조하십시오.	해당 사항 없음
2024년 12월 13일	업데이트된 PCS 에이전트	AWS PCS 에이전트 1.1.1-1에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 단원을 참조하십시오.	해당 사항 없음
2024년 12월 6일	업데이트된 PCS 에이전트 및 Slurm 설치 관리자	AWS PCS 에이전트 1.1.0-1 및 Slurm 설치 관리자 23.11.10-2에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 단원을 참조하십시오.	해당 사항 없음
2024년 12월 6일	OS 지원에 대한 주제 추가	자세한 내용은 AWS PCS에서 지원되는 운영 체제 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2024년 11월 8일	재구성된 사용 설명서	주제를 최상위 수준으로 가져오고, 일부 주제를 자체 페이지로 이동하고, 유사한 주제를 그룹화하도록 사용 설명서를 재구성했습니다.	해당 사항 없음
2024년 11월 7일	업데이트된 AMI 주제	Slurm 23.11.10 및 libjwt 17.0에 대한 AMI 주제를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 3단계 - Slurm 설치 섹션을 참조하세요. AMIs. 자세한 내용은 AWS PCS 샘플 AMIs 릴리스 정보 단원을 참조하십시오.	해당 사항 없음
2024년 11월 7일	AWS PCS에서 암호화된 EBS 볼륨 사용에 대한 새 주제 추가	AWS PCS에서 암호화된 EBS 볼륨에 필요한 KMS 키 정책을 설명하는 주제가 추가되었습니다. 자세한 내용은 AWS PCS에서 암호화된 EBS 볼륨과 함께 사용하는 데 필요한 KMS 키 정책 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2024년 10월 18일	AWS PCS 에이전트 1.0.1-1 릴리스	AWS PCS 에이전트 버전 1.0.1-1을 참조하도록 AMI 관련 설명서를 업데이트했습니다. 자세한 내용은 AWS PCS용 사용자 지정 AMIs를 빌드하기 위한 소프트웨어 설치 관리자 및 2단계 - AWS PCS 에이전트 설치 섹션을 참조하세요.	해당 사항 없음
2024년 10월 10일	문제 해결 장 추가	재부팅 후 자동으로 교체되는 EC2 인스턴스에 대한 주제가 포함된 문제 해결 장이 추가되었습니다. 자세한 내용은 AWS 병렬 컴퓨팅 서비스의 문제 해결 단원을 참조하십시오.	해당 사항 없음
2024년 9월 23일	API 작업을 사용하고 서비스 관리자를 위해 최소 권한을 업데이트했습니다.	`ec2:DescribeInstanceTypeOfferings` 이제 `CreateComputeNodeGroup` 및 `UpdateComputeNodeGroup` API 작업에 권한이 필요합니다. 자세한 내용은 AWS PCS에 대한 최소 권한 단원을 참조하십시오.	해당 사항 없음
2024년 9월 5일	서비스 관리자의 최소 권한에 대한 예제 IAM 정책 업데이트	자세한 내용은 서비스 관리자의 최소 권한 단원을 참조하십시오.	해당 사항 없음

Date	변경	설명서 업데이트	API 버전 업데이트됨
2024년 9월 5일	관리형 정책 페이지에서 JSON에 누락된 권한 추가	이는 설명서만 수정한 것입니다. 실제 관리형 정책은 변경되지 않았습니다. 자세한 내용은 AWS AWS 병렬 컴퓨팅 서비스에 대한 관리형 정책 단원을 참조하십시오.	해당 사항 없음
2024년 8월 28일	관리형 정책 페이지 추가	자세한 내용은 AWS AWS 병렬 컴퓨팅 서비스에 대한 관리형 정책 단원을 참조하십시오.	해당 사항 없음
2024년 8월 28일	AWS PCS 릴리스	AWS PCS 사용 설명서의 최초 릴리스입니다.	AWS SDK: 2024-08-28

AWS 용어집

최신 AWS 용어는 AWS 용어집 참조의 AWS 용어집을 참조하세요.

기계 번역으로 제공되는 번역입니다. 제공된 번역과 원본 영어의 내용이 상충하는 경우에는 영어 버전이 우선합니다.