Red Hat Ceph Storage 3

블록 장치 가이드

Red Hat Ceph Storage 블록 장치 관리, 생성, 구성 및 사용

Last Updated: 2023-08-18

Red Hat Ceph Storage 3 블록 장치 가이드

Red Hat Ceph Storage 블록 장치 관리, 생성, 구성 및 사용

법적 공지

초록

이 문서에서는 Red Hat Ceph Storage 블록 장치를 관리, 생성, 구성 및 사용하는 방법을 설명합니다.

차례

1장. 개요

블록은 일련의 바이트(예: 512바이트 블록)입니다. 블록 기반 스토리지 인터페이스는 다음과 같은 회전 미디어를 사용하여 데이터를 저장하는 가장 일반적인 방법입니다.

- 하드 디스크,
- CDs,
- 플로피 디스크,
- 또한 기존 9 트랙 드라이크도 있습니다.

블록 장치 인터페이스의 유비쿼리티를 통해 가상 블록 장치는 Red Hat Ceph Storage와 같은 대용량 데이터 스토리지 시스템과 상호 작용하는 데 이상적인 후보가 됩니다.

Ceph 블록 장치(RADOS) RBD(Reliable Autonomic Distributed Object Store) 블록 장치(RBD)는 Ceph 스토리지 클러스터에서 여러 개의 OSD(Object Storage Devices)를 통해 스트립된 데이터를 씬 프로비저닝하고 다시 저장할 수 있습니다. Ceph 블록 장치는 다음과 같은 RADOS 기능을 활용합니다.

- 스냅샷 생성
- 복제,
- 및 일관성을 제공합니다.

Ceph 블록 장치는 **librbd** 라이브러리를 사용하여 OSD와 상호 작용합니다.

Ceph 블록 장치는 Ceph 블록 장치와 통합하기 위해 **libvirt** 및 QEMU 유틸리티를 사용하는 OpenStack 및 CloudStack과 같은 KVM(커널 가상 시스템)에 무한 확장성과 함께 고성능을 제공합니다. 동일한 클러스터를 사용하여 Ceph Object Gateway 및 Ceph Block Devices를 동시에 작동할 수 있습니다.

중요

Ceph 블록 장치를 사용하려면 실행 중인 Ceph Storage 클러스터에 액세스할 수 있어야 합니다. Red Hat Ceph Storage 설치에 대한 자세한 내용은 *Red Hat Enterprise Linux의 설치 가이드* 또는 *Ubuntu 용 설치 가이드를 참조하십시오*.

2장. 블록 장치 명령

rbd 명령을 사용하면 블록 장치 이미지를 생성, 나열, introspect 및 제거할 수 있습니다. 또한 이 파일을 사용하여 이미지를 복제하고, 스냅샷을 생성하며, 이미지를 스냅샷으로 롤백하고, 스냅샷을 확인하는 등의 작업을 수행할 수 있습니다.

2.1. 사전 요구 사항

Ceph 블록 장치와 **rbd** 명령을 사용하려면 다음 두 가지 사전 요구 사항을 충족해야 합니다.

- 실행 중인 Ceph Storage 클러스터에 액세스할 수 있어야 합니다. 자세한 내용은 Red Hat *Enterprise Linux용 Red Hat Ceph Storage 3 설치 가이드* 또는 *Ubuntu 용 설치 가이드를 참조하십시오*.
- Ceph Block Device 클라이언트를 설치해야 합니다. 자세한 내용은 Red Hat *Enterprise Linux용 Red Hat Ceph Storage 3 설치 가이드* 또는 *Ubuntu 용 설치 가이드를 참조하십시오*.

중요

수동으로 Ceph 블록 장치 설치 장은 클라이언트 노드에서 Ceph 블록 장치 마운트 및 사용에 대한 정보도 제공합니다. Ceph Storage 클러스터에서 블록 장치에 대한 이미지를 생성한 후에만 클라이언트 노드에서 이 단계를 실행합니다. 자세한 내용은 2.4절. "블록 장치 이미지 생성" 을 참조하십시오.

2.2. 도움말 표시

rbd help 명령을 사용하여 특정 **rbd** 명령 및 하위 명령에 대한 도움말을 표시합니다.

```
[root@rbd-client ~]# rbd help <command> <subcommand>
```

예제

snap list 명령에 대한 도움말을 표시하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd help snap list
```

참고

h 옵션은 사용 가능한 모든 명령에 대한 도움말을 계속 표시합니다.

2.3. 블록 장치 풀 생성

블록 장치 클라이언트를 사용하기 전에 **rbd** 의 풀이 존재하고 활성화 및 초기화되었는지 확인합니다. **rbd** 풀을 만들려면 다음을 실행합니다.

```
[root@rbd-client ~]# ceph osd pool create {pool-name} {pg-num} {pgp-num}
[root@rbd-client ~]# ceph osd pool application enable {pool-name} rbd
[root@rbd-client ~]# rbd pool init -p {pool-name}
```

참고

풀은 원본으로 지정하기 전에 먼저 만듭니다. 자세한 내용은 Red Hat Ceph *Storage 3의 스토리지 전략* 가이드의 풀 장을 참조하십시오.

2.4. 블록 장치 이미지 생성

블록 장치를 노드에 추가하기 전에 Ceph 스토리지 클러스터에 블록 장치를 생성할 이미지를 생성합니다. 블록 장치 이미지를 생성하려면 다음 명령을 실행합니다.

```
[root@rbd-client ~]# rbd create <image-name> --size <megabytes> --pool <pool-name>
```

예를 들어 스택 이라는 풀에 정보를 저장하는**data** 라는 1GB 이미지를 생성하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd create data --size 1024 --pool stack
```

참고

이미지를 생성하기 전에 **rbd** 풀이 있는지 확인합니다. 자세한 내용은블록 장치 풀 생성을 참조하십시오.

2.5. 블록 장치 이미지 나열

rbd 풀의 블록 장치를 나열하려면 다음 명령을 실행합니다(**rbd** 는 기본 풀 이름임).

```
[root@rbd-client ~]# rbd ls
```

특정 풀의 블록 장치를 나열하려면 다음을 실행하고 **{poolname}** 을 풀 이름으로 교체합니다.

```
[root@rbd-client ~]# rbd ls {poolname}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd ls swimmingpool
```

2.6. 이미지 정보 검색

특정 이미지에서 정보를 검색하려면 다음을 실행하고 **{image-name}** 을 이미지 이름으로 교체합니다.

```
[root@rbd-client ~]# rbd --image {image-name} info
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --image foo info
```

풀 내의 이미지에서 정보를 검색하려면 다음을 실행하고 **{image-name}** 을 이미지 이름으로 바꾸고**{pool-name}** 을 풀 이름으로 교체합니다.

```
[root@rbd-client ~]# rbd --image {image-name} -p {pool-name} info
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --image bar -p swimmingpool info
```

2.7. 블록 장치 이미지 크기 조정

Ceph 블록 장치 이미지는 씬 프로비저닝됩니다. 데이터 저장을 시작할 때까지 실제 스토리지는 사용하지 않습니다. 그러나 최대 용량은 **--size** 옵션으로 설정할 수 있습니다.

Ceph 블록 장치 이미지의 최대 크기를 늘리거나 줄이려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd resize --image <image-name> --size <size>
```

2.8. 블록 장치 이미지 제거

블록 장치를 제거하려면 다음을 실행하되 **{image-name}** 을 삭제하려는 이미지 이름으로 교체합니다.

```
[root@rbd-client ~]# rbd rm {image-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd rm foo
```

풀에서 블록 장치를 제거하려면 다음을 실행하고 **{image-name}** 을 이미지 이름으로 교체하여**{pool-name}** 을 풀 이름으로 삭제하고 교체합니다.

```
[root@rbd-client ~]# rbd rm {image-name} -p {pool-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd rm bar -p swimmingpool
```

2.9. 블록 장치 이미지를 TRASH로 이동

RADOS 블록 장치(RBD) 이미지는 **rbd trash** 명령을 사용하여 트리시로 이동할 수 있습니다. 이 명령은**rbd rm** 명령보다 더 많은 옵션을 제공합니다.

이미지가 트라쉬로 이동되면 나중에 트라쉬에서 제거할 수 있습니다. 이는 실수로 삭제되는 것을 방지하는 데 도움이 됩니다.

이미지를 트래시로 이동하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd trash move {image-spec}
```

이미지가 trash에 있으면 고유 이미지 ID가 할당됩니다. trash 옵션을 사용해야 하는 경우 나중에 이미지를 지정하려면 이 이미지 ID가 필요합니다. **trash**에 있는 이미지 **ID** 목록에 대해 **rbd** trash 목록을 실행합니다. 이 명령은 이미지의 사전 조건 이름도 반환합니다.

또한 **rbd info** 및 **rbd snap** 명령과 함께 사용할 수 있는 선택적**--image-id** 인수도 있습니다. **rbd info** 명령과 함께 **--image-id** 를 사용하여 Trash의 이미지 속성을 확인하고**rbd snap** 을 사용하여 Trash에서 이미지의 스냅샷을 제거합니다.

Trash에서 이미지 제거

trash에서 이미지를 제거하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd trash remove [{pool-name}/] {image-id}
```

중요

이미지가 트래시에서 제거되면 복원할 수 없습니다.

지연 테라시 제거

--delay 옵션을 사용하여 trash에서 이미지를 제거할 수 있을 때까지의 시간을 설정합니다. 이미지를 제거할 수 있을 때까지 대기하는 시간 (기본값: 0)으로 바꿉니다.

```
[root@rbd-client ~]# rbd trash move [--delay {time}] {image-spec}
```

delay 옵션이 활성화되면 강제 적용되지 않는 한 지정된 기간 내에 이미지를 트래시에서 제거할 수 없습니다.

Trash에서 이미지 복원

이미지가 trash에서 제거되지 않은 한 **rbd trash restore** 명령을 사용하여 복원할 수 있습니다.

rbd trash restore 명령을 실행하여 이미지를 복원합니다.

```
[root@rbd-client ~]# rbd trash restore [{pool-name}/] {image-id}
```

2.10. 이미지 기능 활성화 및 비활성화

이미 존재하는 이미지에서 **fast-diff**,**exclusive-lock**,**object-map** 또는 **journaling** 과 같은 이미지 기능을 활성화하거나 비활성화할 수 있습니다.

기능을 활성화하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd feature enable <pool-name>/<image-name> <feature-name>
```

기능을 비활성화하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd feature disable <pool-name>/<image-name> <feature-name>
```

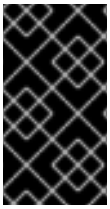
예

- 데이터 풀의 **image1** 이미지에서 배타적 잠금 기능을 활성화하려면 다음을 수행합니다.

  ```
  [root@rbd-client ~]# rbd feature enable data/image1 exclusive-lock
  ```

- 데이터 풀의 **image2** 이미지에서 **fast-diff** 기능을 비활성화하려면 다음을 수행합니다.

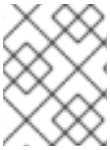
  ```
  [root@rbd-client ~]# rbd feature disable data/image2 fast-diff
  ```



중요

fast-diff 및 **object-map** 기능을 활성화한 후 오브젝트 맵을 다시 빌드합니다.

```
[root@rbd-client ~]# rbd object-map rebuild <pool-name>/<image-name>
```



참고

깊은 **flatten** 기능은 이미 존재하는 이미지에서만 비활성화할 수 있지만 활성화할 수는 없습니다. 깊은 **flatten** 을 사용하려면 이미지를 생성할 때 활성화합니다.

2.11. 이미지 메타데이터 작업

Ceph는 사용자 지정 이미지 메타데이터를 키-값 쌍으로 추가할 수 있도록 지원합니다. 쌍에는 엄격한 형식이 없습니다.

또한 메타데이터를 사용하면 특정 이미지에 대한 RBD 구성 매개변수를 설정할 수 있습니다. 자세한 내용은 파트에 대한 기본 구성 덮어쓰기를 참조하십시오.

rbd image-meta 명령을 사용하여 메타데이터로 작업합니다.

이미지 메타데이터 설정

새 메타데이터 키-값 쌍을 설정하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta set <pool-name>/<image-name> <key> <value>
```

예제

- **last_update** 키를 데이터 풀의 데이터 세트 이미지에 있는**2016-06-06** 값으로 설정하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta set data/dataset last_update 2016-06-06
```

이미지 메타데이터 제거

메타데이터 키-값 쌍을 제거하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta remove <pool-name>/<image-name> <key>
```

예제

- 데이터 풀의 데이터 세트 이미지에서 **last_update** 키-값 쌍을 제거하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta remove data/dataset last_update
```

키의 값 가져오기

키 값을 보려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta get <pool-name>/<image-name> <key>
```

예제

- **last_update** 키의 값을 보려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta get data/dataset last_update
```

이미지 메타데이터 나열

이미지에 모든 메타데이터를 표시하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta list <pool-name>/<image-name>
```

예제

- 데이터 풀의 데이터 세트 이미지에 메타데이터 세트를 나열하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd data/dataset image-meta list
```

부분 이미지의 기본 구성 덮어쓰기

특정 이미지의 **Ceph** 구성 파일에 설정된 **RBD** 이미지 구성 설정을 재정의하려면 **conf_** 접두사를 이미지 메타데이터로 사용하여 구성 매개변수를 설정합니다.

```
[root@rbd-client ~]# rbd image-meta set <pool-name>/<image-name> conf_<parameter> <value>
```

예제

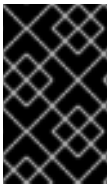
- 데이터 풀에서 데이터 세트 이미지에 대한 **RBD** 캐시를 비활성화하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd image-meta set data/dataset conf_rbd_cache false
```

가능한 구성 옵션 목록은 블록 장치 구성 참조 를 참조하십시오.

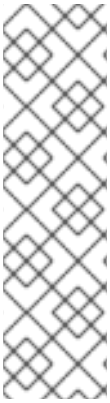
3장. 스냅샷

스냅샷은 특정 시점에 이미지 상태의 읽기 전용 사본입니다. Ceph 블록 장치의 고급 기능 중 하나는 이미지 상태 기록을 유지하기 위해 이미지 스냅샷을 생성할 수 있다는 것입니다. 또한 Ceph는 스냅샷 계층 지정을 지원하므로 이미지(예: VM 이미지)를 빠르고 쉽게 복제할 수 있습니다. Ceph는 rbd 명령과 QEMU,libvirt,OpenStack, CloudStack 을 비롯한 여러 고급 인터페이스를 사용하여 블록 장치 스냅샷을 지원합니다.



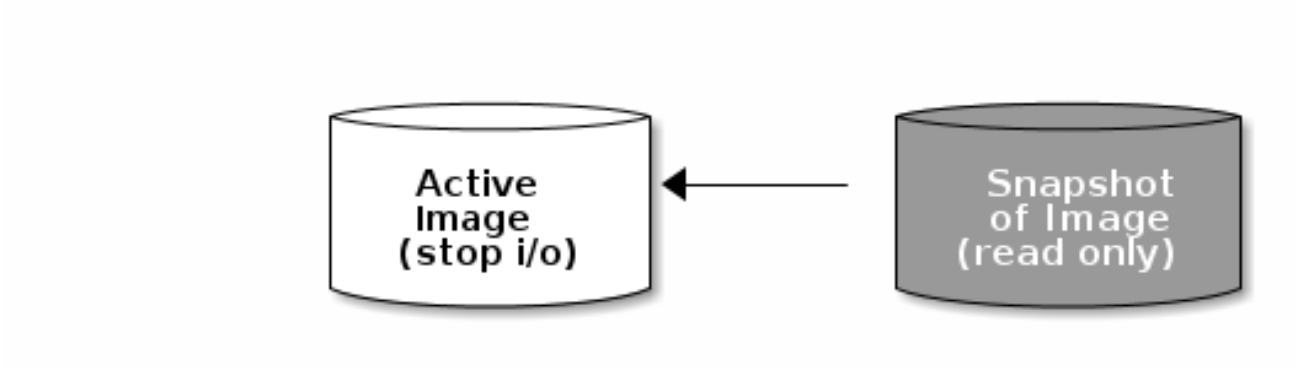
중요

RBD 스냅샷을 사용하려면 실행 중인 Ceph 클러스터가 있어야 합니다.



참고

I/O 가 이미지 진행 중인 동안 스냅샷을 찍으면 스냅샷에 이미지의 정확한 또는 최신 데이터가 표시되지 않을 수 있으며 스냅샷을 마운트하려면 새 이미지에 복제해야 할 수 있습니다. 따라서 이미지 스냅샷을 생성하기 전에 I/O 를 중지하는 것이 좋습니다. 이미지에 파일 시스템이 포함된 경우 스냅샷을 생성하기 전에 파일 시스템이 일관된 상태에 있어야 합니다. I/O 를 중지하려면 fsfreeze 명령을 사용할 수 있습니다. 자세한 내용은 fsfreeze(8) 매뉴얼 페이지를 참조하십시오. 가상 머신의 경우 qemu-guest-agent 를 사용하여 스냅샷을 생성할 때 파일 시스템을 자동으로 정지할 수 있습니다.



3.1. CEPHX NOTES

cephx 가 활성화되면 (기본적으로 해당) 사용자 이름 또는 ID와 사용자의 해당 키가 포함된 인증 키의 경로를 지정해야 합니다. 다음 매개변수를 다시 입력하지 않도록 CEPH_ARGS 환경 변수를 추가할 수도 있습니다.

```
[root@rbd-client ~]# rbd --id {user-ID} --keyring=/path/to/secret [commands]
[root@rbd-client ~]# rbd --name {username} --keyring=/path/to/secret [commands]
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --id admin --keyring=/etc/ceph/ceph.keyring [commands]
[root@rbd-client ~]# rbd --name client.admin --keyring=/etc/ceph/ceph.keyring [commands]
```

작은 정보

CEPH_ARGS 환경 변수에 사용자와 시크릿을 추가하여 매번 입력할 필요가 없습니다.

3.2. 스냅샷 기본 사항

다음 절차에서는 명령줄에서 **rbd** 명령을 사용하여 스냅샷을 생성, 나열 및 제거하는 방법을 보여줍니다.

3.2.1. 스냅샷 생성

rbd 로 스냅샷을 만들려면 **snap create** 옵션, 풀 이름 및 이미지 이름을 지정합니다.

```
[root@rbd-client ~]# rbd --pool {pool-name} snap create --snap {snap-name} {image-name}
[root@rbd-client ~]# rbd snap create {pool-name}/{image-name}@{snap-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd snap create --snap snapname foo
[root@rbd-client ~]# rbd snap create rbd/foo@snapname
```

3.2.2. 스냅샷 나열

이미지 스냅샷을 나열하려면 풀 이름과 이미지 이름을 지정합니다.

```
[root@rbd-client ~]# rbd --pool {pool-name} snap ls {image-name}
[root@rbd-client ~]# rbd snap ls {pool-name}/{image-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd snap ls foo
[root@rbd-client ~]# rbd snap ls rbd/foo
```

3.2.3. 스냅샷 롤백

rbd 를 사용하여 스냅샷으로 롤백하려면 **snap rollback** 옵션, 풀 이름, 이미지 이름 및 스냅 샷 이름을 지정합니다.

```
rbd --pool {pool-name} snap rollback --snap {snap-name} {image-name}
rbd snap rollback {pool-name}/{image-name}@{snap-name}
```

예를 들면 다음과 같습니다.

```
rbd --pool rbd snap rollback --snap snapname foo
rbd snap rollback rbd/foo@snapname
```

참고

스냅샷으로 이미지를 롤백한다는 것은 현재 버전의 이미지를 스냅샷의 데이터로 덮어쓸 수 있음을 의미합니다. 롤백을 실행하는 데 걸리는 시간은 이미지 크기와 함께 증가합니다. 스냅샷에서 이미지를 롤백하는 것보다 스냅샷을 복제하는 것이 더 빠르 기 때문에 기존 상태로 되돌리는 것이 좋습니다.

3.2.4. 스냅샷 삭제

rbd 를 사용하여 스냅샷을 삭제하려면 **snap rm** 옵션, 풀 이름, 이미지 이름 및 스냅샷 이름을 지정합니다.

```
[root@rbd-client ~]# rbd --pool <pool-name> snap rm --snap <snap-name> <image-name>
[root@rbd-client ~]# rbd snap rm <pool-name-/<image-name>@<snap-name>
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd snap rm --snap snapname foo
[root@rbd-client ~]# rbd snap rm rbd/foo@snapname
```

중요

이미지에 복제본이 있는 경우 복제된 이미지에는 상위 이미지 스냅샷에 대한 참조가 유지됩니다. 상위 이미지 스냅샷을 삭제하려면 하위 이미지를 먼저 병합해야 합니다. 자세한 내용은 복제된 이미지 고정 을 참조하십시오.

참고

Ceph OSD 데몬은 데이터를 비동기적으로 삭제하므로 스냅샷을 삭제해도 디스크 공간이 즉시 해제되지 않습니다.

3.2.5. 스냅샷 삭제

rbd 를 사용하여 이미지의 모든 스냅샷을 삭제하려면 **snap purge** 옵션과 이미지 이름을 지정합니다.

```
[root@rbd-client ~]# rbd --pool {pool-name} snap purge {image-name}
[root@rbd-client ~]# rbd snap purge {pool-name}/{image-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd snap purge foo
[root@rbd-client ~]# rbd snap purge rbd/foo
```

3.2.6. 스냅샷 이름 변경

스냅샷 이름을 변경하려면 다음을 수행합니다.

```
[root@rbd-client ~]# rbd snap rename <pool-name>/<image-name>@<original-snapshot-name>
<pool-name>/<image-name>@<new-snapshot-name>
```

예제

데이터 풀에서 **snap1** 스냅샷 의 이름을 **snap2** 로 변경하려면 다음을 수행합니다.

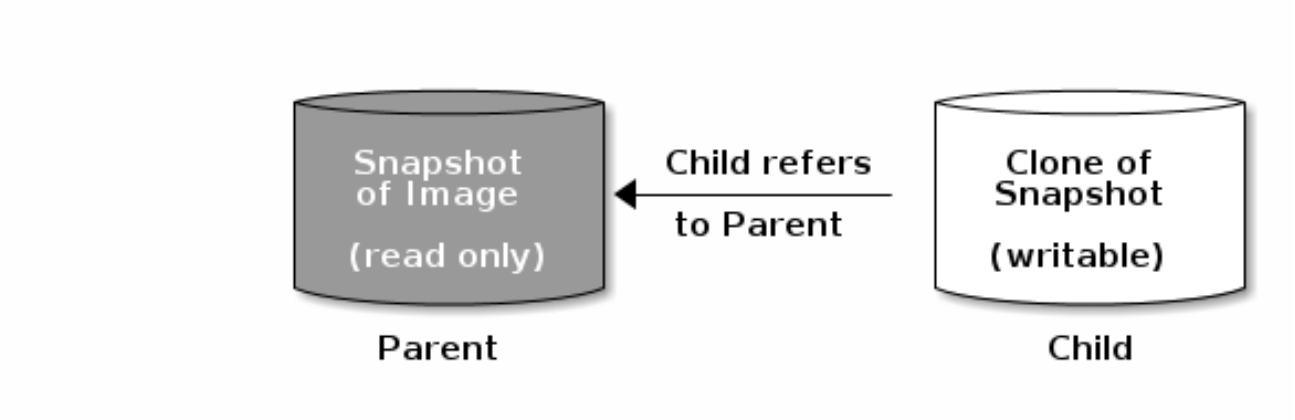
```
[root@rbd-client ~]# rbd snap rename data/dataset@snap1 data/dataset@snap2
```

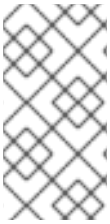
snapshot 이름 변경에 대한 추가 세부 정보를 표시하려면 **rbd help snap rename** 명령을 실행합니다.

3.3. 계층화

Ceph는 블록 장치 스냅샷의 많은 **COW(Copy-On-Write)** 또는 **COR(Copy-On-read)** 복제본을 생성하는 기능을 지원합니다. 스냅샷 계층을 사용하면 **Ceph** 블록 장치 클라이언트가 이미지를 매우 빠르게 생성할 수 있습니다. 예를 들어 **Linux VM**을 사용하여 블록 장치 이미지를 만들 수 있습니다. 그런 다음 이미

지를 스냅샷하고 스냅샷을 보호하며 원하는 만큼 복제를 생성할 수 있습니다. 스냅샷은 읽기 전용이므로 스냅샷 복제를 사용하면 신속하게 복제할 수 있습니다.





참고

상위 및 하위 용어는 **Ceph** 블록 장치 스냅샷(**parent**)과 스냅샷(하위)에서 복제된 해당 이미지를 의미합니다. 이러한 용어는 아래의 명령줄 사용에 중요합니다.

복제된 각 이미지(하위)는 해당 상위 이미지에 대한 참조를 저장하므로 복제된 이미지가 상위 스냅샷을 열고 읽을 수 있습니다. 이 참조는 스냅샷의 정보가 복제본에 완전히 복사될 때 복제본이 병합 되면 제거됩니다. 플랫화에 대한 자세한 내용은 3.3.6절. “복제된 이미지 병합” 을 참조하십시오.

스냅샷 복제본은 다른 **Ceph** 블록 장치 이미지와 동일하게 작동합니다. 복제된 이미지에 읽고, 복제하고, 크기를 조정할 수 있습니다. 복제된 이미지에는 특별한 제한이 없습니다. 그러나 스냅샷 복제본은 스냅샷을 참조하므로 복제하기 전에 스냅샷을 보호합니다.

스냅샷 복제본은 **COW(Copy-On-Write)** 또는 **COR(Copy-On-read)** 복제본일 수 있습니다. **COW(Copy-On-Write)**는 클론에 대해 항상 활성화되어 있고 **COR(Copy-on-read)**를 명시적으로 활성화해야 합니다. **COW(Copy-On-Write)**는 복제본 내의 할당되지 않은 오브젝트에 쓸 때 상위에서 복제본으로 데이터를 복사합니다. 복제본 내의 할당되지 않은 오브젝트에서 읽을 때 **COR(Copy-on-read)**는 상위에서 복제본으로 데이터를 복사합니다. 복제본에서 데이터를 읽으면 오브젝트가 복제본에 아직 없는 경우에만 상위 데이터만 읽습니다. **RADOS** 블록 장치는 큰 이미지를 여러 개체(기본값: **4MB**)로 분할하고 모든 **COW(Copy-On-Write)** 및 **COW(Copy-On-read)** 작업이 전체 오브젝트에서 발생하며(즉, **1**바이트를 복제에 쓰는 경우 대상 오브젝트가 이전 **COW/COR** 작업의 복제본에 아직 존재하지 않는 경우 복제에 **1MB** 오브젝트를 읽고 복제에 기록됨).

COR(Copy-On-read)가 활성화되어 있는지 여부에 관계없이 복제에서 기본 오브젝트를 읽으면 충족할 수 없는 모든 읽기는 상위로 다시 라우팅됩니다. 부모의 수에 대한 제한이 없기 때문에 (복제할 수 있습니다. 복제할 수 있음)이 **reroute**는 오브젝트가 발견되거나 기본 상위 이미지를 도달할 때까지 계속됩니다. **COR(Copy-On-read)**를 사용하도록 설정한 경우 복제에서 직접 만족하지 못하는 모든 읽기를 상위에서

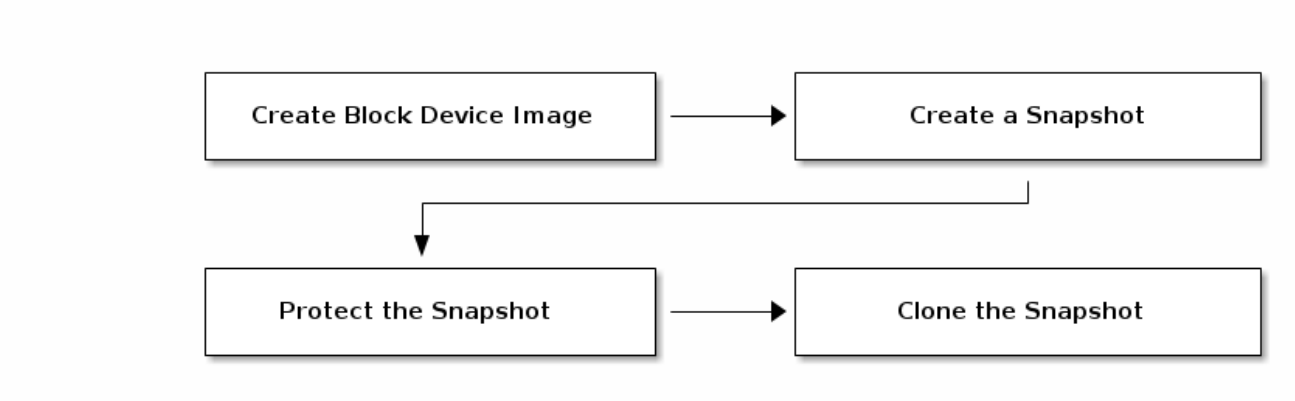
읽고 해당 데이터를 복제본에 작성하여 상위로부터 읽기 없이도 복제 자체에서 동일한 크기를 충족할 수 있습니다.

기본적으로 이 작업은 온 디맨드 오브젝트별 플랫 작업입니다. 이는 복제가 부모(다른 지리적 위치의 다른 풀의 상위)에서 대기 시간이 높은 연결에 있는 경우 특히 유용합니다. **COR(Copy-On-read)**는 읽기 지연 시간을 줄입니다. 처음 몇 개의 읽기는 상위에서 추가 데이터를 읽기 때문에 대기 시간이 길어지고(예: 복제본에서 **1**바이트를 읽으지만 현재는 **4MB**를 상위에서 읽고 복제에 기록해야 함) 모든 향후 읽기가 복제 자체에서 제공됩니다.

스냅샷에서 **COR(Copy-on-read)** 복제본을 생성하려면 **ceph.conf** 파일에 **rbd_clone_copy_on_read = true** 아래에 **rbd_clone_copy_on_read = true**를 추가하여 명시적으로 이 기능을 활성화해야 합니다.

3.3.1. 계층 지정 시작하기

Ceph 블록 장치 계층화는 간단한 프로세스입니다. 이미지가 있어야 합니다. 이미지의 스냅샷을 생성해야 합니다. 스냅샷을 보호해야 합니다. 이러한 단계를 수행한 후 스냅샷 복제를 시작할 수 있습니다.



복제된 이미지에는 상위 스냅샷에 대한 참조가 있으며 풀 **ID**, 이미지 **ID** 및 스냅샷 **ID**가 포함됩니다. 풀 **ID**가 포함되어 있으면 한 풀에서 다른 풀의 이미지로 스냅샷을 복제할 수 있습니다.

1. 이미지 템플릿: 블록 장치 계층화의 일반적인 사용 사례는 마스터 이미지와 복제본의 템플릿 역할을 하는 스냅샷을 생성하는 것입니다. 예를 들어 사용자는 **RHEL7** 배포용 이미지를 생성하고 해당 스냅샷을 생성할 수 있습니다. 사용자는 주기적으로 이미지를 업데이트하고 새 스냅샷을 만들 수 있습니다(예: **yum update**,**yum upgrade**, **rbd snap create**). 이미지가 성숙해짐에 따라 사용자는 스냅샷 중 하나를 복제할 수 있습니다.

2. 연장된 템플릿: 고급 사용 사례에는 기본 이미지보다 더 많은 정보를 제공하는 템플릿 이미지 확장이 포함됩니다. 예를 들어, 사용자는 이미지(예: **VM** 템플릿)를 복제하고 다른 소프트웨어(예: 데이터베이스, 컨텐츠 관리 시스템, 분석 시스템 등)를 설치한 다음, 기본 이미지와 마찬가지로 확장 이미지를 스냅샷할 수 있습니다.

3.

템플릿 풀: 블록 장치 계층 지정을 사용하는 한 가지 방법은 템플릿 역할을 하는 마스터 이미지와 해당 템플릿의 스냅샷이 포함된 풀을 만드는 것입니다. 그런 다음 사용자가 풀 내에서 쓰거나 실행하는 기능 없이 스냅샷을 복제할 수 있도록 읽기 전용 권한을 사용자에게 확장할 수 있습니다.

4.

이미지 마이그레이션/복구: 블록 장치 계층 지정을 사용하는 한 가지 방법은 한 풀에서 다른 풀로 데이터를 마이그레이션하거나 복구하는 것입니다.

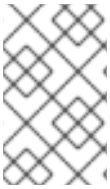
3.3.2. 스냅샷 보호

복제본은 상위 스냅샷에 액세스합니다. 사용자가 실수로 상위 스냅샷을 삭제한 경우 모든 복제본이 중단됩니다. 데이터 손실을 방지하기 위해 복제하기 전에 스냅샷을 보호합니다. 이렇게 하려면 다음 명령을 실행합니다.

```
[root@rbd-client ~]# rbd --pool {pool-name} snap protect --image {image-name} --snap {snapshot-name}
[root@rbd-client ~]# rbd snap protect {pool-name}/{image-name}@{snapshot-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd snap protect --image my-image --snap my-snapshot
[root@rbd-client ~]# rbd snap protect rbd/my-image@my-snapshot
```



참고

보호된 스냅샷을 삭제할 수 없습니다.

3.3.3. 스냅샷 복제

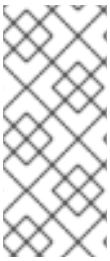
스냅샷을 복제하려면 상위 풀, 이미지, 스냅샷, 하위 풀 및 이미지 이름을 지정해야 합니다. 복제하려면 먼저 스냅샷을 보호해야 합니다. 이렇게 하려면 다음 명령을 실행합니다.

```
[root@rbd-client ~]# rbd --pool {pool-name} --image {parent-image} --snap {snap-name} --dest-pool {pool-name} --dest {child-image}

[root@rbd-client ~]# rbd clone {pool-name}/{parent-image}@{snap-name}  {pool-name}/{child-image-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd clone rbd/my-image@my-snapshot rbd/new-image
```



참고

한 풀에서 다른 풀의 이미지로 스냅샷을 복제할 수 있습니다. 예를 들어 하나의 풀에서 읽기 전용 이미지 및 스냅샷을 템플릿으로 유지 관리하고 다른 풀에서 쓰기 가능한 복제본을 유지할 수 있습니다.

3.3.4. 스냅샷 보호 해제

스냅샷을 삭제하려면 먼저 보호를 해제해야 합니다. 또한 복제본에서 참조가 있는 스냅샷을 삭제할 수 *없습니다*. 스냅샷을 삭제하기 전에 각 스냅샷 복제본을 평면화해야 합니다. 이렇게 하려면 다음 명령을 실행합니다.

```
[root@rbd-client ~]#rbd --pool {pool-name} snap unprotect --image {image-name} --snap {snapshot-name}

[root@rbd-client ~]# rbd snap unprotect {pool-name}/{image-name}@{snapshot-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd snap unprotect --image my-image --snap my-snapshot

[root@rbd-client ~]# rbd snap unprotect rbd/my-image@my-snapshot
```

3.3.5. 스냅샷 목록

스냅샷의 하위 항목을 나열하려면 다음을 실행합니다.

```
rbd --pool {pool-name} children --image {image-name} --snap {snap-name}
rbd children {pool-name}/{image-name}@{snapshot-name}
```

예를 들면 다음과 같습니다.

```
rbd --pool rbd children --image my-image --snap my-snapshot
rbd children rbd/my-image@my-snapshot
```

3.3.6. 복제된 이미지 병합

복제된 이미지에는 상위 스냅샷에 대한 참조가 유지됩니다. 하위 복제본에서 상위 스냅샷에 대한 참조

를 제거하면 스냅샷의 정보를 복제본으로 복사하여 이미지를 효과적으로 "플라이프"합니다. 복제를 병합하는 데 걸리는 시간은 스냅샷 크기에 따라 늘어납니다.

하위 이미지와 관련된 상위 이미지 스냅샷을 삭제하려면 먼저 하위 이미지를 병합해야 합니다.

```
[root@rbd-client ~]# rbd --pool <pool-name> flatten --image <image-name>
[root@rbd-client ~]# rbd flatten <pool-name>/<image-name>
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd --pool rbd flatten --image my-image
[root@rbd-client ~]# rbd flatten rbd/my-image
```

병합된 이미지에는 스냅샷의 모든 정보가 포함되어 있으므로 병합된 이미지는 계층화된 복제본보다 더 많은 스토리지 공간을 사용합니다.

참고

이미지에 깊은 **flatten** 기능이 활성화되면 기본적으로 이미지 복제가 상위에서 달랐습니다.

4장. 블록 장치 미러링

RADOS 블록 장치(RBD) 미러링은 둘 이상의 Ceph 클러스터 간에 Ceph 블록 장치 이미지를 비동기 복제하는 프로세스입니다. 미러링은 읽기 및 쓰기, 블록 장치 크기 조정, 스냅샷, 복제 및 플랫을 포함하여 이미지에 대한 모든 변경 사항에 대한 시점의 일관된 복제본을 보장합니다.

미러링은 필수 전용 잠금과 RBD 저널링 기능을 사용하여 이미지에 대한 모든 수정 사항을 발생하는 순서대로 기록합니다. 이렇게 하면 이미지의 충돌 일치 미러를 사용할 수 있습니다. 이미지를 피어 클러스터에 미러링하려면 먼저 저널링을 활성화해야 합니다. 자세한 내용은 4.1절. “journalctling 활성화” 을 참조하십시오.

미러링된 블록 장치와 연결된 기본 및 보조 풀에 저장된 이미지이므로 기본 및 보조 풀의 CloudEvent 계층 구조는 동일한 스토리지 용량 및 성능 특성을 보유해야 합니다. 또한 기본 사이트와 보조 사이트 간의 네트워크 연결에는 너무 많은 대기 시간 없이 미러링이 수행되도록 하기에 충분한 대역폭 이 있어야 합니다.

중요

블록 장치 이미지를 미러링하는 기본 및 보조 풀을 지원하는 CloudEvent 계층 구조에는 동일한 용량 및 성능 특성이 있어야 하며 초과 대기 시간 없이 미러링을 보장하기 위한 적절한 대역폭이 있어야 합니다. 예를 들어 기본 클러스터의 이미지에 대한 X MiB/s 평균 쓰기 처리량이 있는 경우 네트워크에서 네트워크 연결에서 N * X 처리량을 보조 사이트에 대한 N * X 처리량과 N 이미지를 미러링하려면 Y%의 안전 요인을 지원해야 합니다.

미러링은 주로 재해 복구를 위해 사용됩니다. 사용하는 미러링 유형에 따라 단방향 미러링을 사용하여 재해 복구 또는 양방향 미러링 이 있는 재해 복구를 참조하십시오.

rbd-mirror 데몬

rbd-mirror 데몬은 한 Ceph 클러스터의 이미지를 다른 Ceph 클러스터로 동기화합니다.

복제 유형에 따라 rbd-mirror 는 단일 클러스터 또는 미러링에 참여하는 모든 클러스터에서 실행됩니다.

- 단방향 복제
  - 기본 클러스터에서 백업 역할을 하는 보조 클러스터로 데이터가 미러링되면 rbd-mirror 는 보조 클러스터에서만 실행됩니다. RBD 미러링에는 여러 개의 보조 사이트가 있을 수 있습니다.

- 양방향 복제

  - 양방향 복제는 기본 클러스터에 **rbd-mirror** 데몬을 추가하므로 이미지가 시연되고 보조 클러스터에서 승격될 수 있습니다. 그런 다음 보조 클러스터의 이미지를 변경할 수 있으며 보조 클러스터에서 보조 클러스터로 역방향 방향으로 복제됩니다. 두 클러스터 모두 두 클러스터에서 이미지를 승격하고 해독할 수 있도록 **rbd-mirror** 가 실행되고 있어야 합니다. 현재 양방향 복제는 두 사이트 간에만 지원됩니다.

rbd-mirror 패키지는 **rbd-mirror** 를 제공합니다.

중요

양방향 복제에서 **rbd-mirror** 의 각 인스턴스는 다른 **Ceph** 클러스터에 동시에 연결할 수 있어야 합니다. 또한 미러링을 처리하기 위해 두 데이터 센터 사이트 간에 충분한 대역폭이 네트워크에 있어야 합니다.

주의

Ceph 클러스터당 단일 **rbd-mirror** 데몬만 실행합니다.

미러링 모드

미러링은 피어 클러스터 내에서 풀별로 구성됩니다. **Ceph**는 풀의 이미지가 미러링되는지에 따라 두 가지 모드를 지원합니다.

풀 모드

저널링 기능이 활성화된 풀의 모든 이미지가 미러링됩니다. 자세한 내용은 풀 미러링 구성 을 참조하십시오.

이미지 모드

풀 내의 특정 이미지 하위 집합만 미러링되며 각 이미지에 대해 미러링을 활성화해야 합니다. 자세한 내용은 이미지 미러링 구성 을 참조하십시오.

이미지 상태

이미지를 수정할 수 있는지 여부는 해당 상태에 따라 다릅니다.

- 기본 상태의 이미지를 수정할 수 있습니다.
- 비기본 상태의 이미지는 수정할 수 없습니다.

이미지에 미러링이 먼저 활성화되면 이미지가 자동으로 기본으로 승격됩니다. 승격이 발생할 수 있습니다.

- 풀 모드에서 미러링을 활성화하여 암시적으로 (4.2절. “풀 구성”참조)
- 특정 이미지 미러링을 활성화하여 명시적으로 지정합니다(4.3절. “이미지 설정”참조).

기본 이미지를 포맷하고 비기본 이미지를 승격할 수 있습니다. 자세한 내용은 4.3절. “이미지 설정” 을 참조하십시오.

4.1. JOURNALCTLING 활성화

RBD 저널링 기능을 활성화할 수 있습니다.

- 이미지가 생성된 경우
- 기존 이미지의 동적

중요

저널링은 또한 활성화해야 하는 배타적 잠금 기능에 따라 다릅니다. 다음 단계를 참조하십시오.

이미지를 생성할 때 저널링을 활성화하려면 **--image-feature** 옵션을 사용합니다.

```
rbd create <image-name> --size <megabytes> --pool <pool-name> --image-feature <feature>
```

예를 들면 다음과 같습니다.

```
# rbd create image1 --size 1024 --pool data --image-feature exclusive-lock,journaling
```

이전에 생성된 이미지에서 저널링을 활성화하려면 **rbd feature enable** 명령을 사용합니다.

```
rbd feature enable <pool-name>/<image-name> <feature-name>
```

예를 들면 다음과 같습니다.

```
# rbd feature enable data/image1 exclusive-lock
# rbd feature enable data/image1 journaling
```

기본적으로 모든 새 이미지에서 저널링을 활성화하려면 **Ceph** 구성 파일에 다음 설정을 추가합니다.

```
rbd default features = 125
```

4.2. 풀 구성

이 장에서는 다음 작업을 수행하는 방법을 설명합니다.

- 풀에서 미러링 활성화
- 풀에서 미러링 비활성화
- 클러스터 피어 추가
- 피어에 대한 정보 보기
- 클러스터 피어 제거

- 풀의 미러링 상태 가져오기

두 피어 클러스터에서 다음 명령을 실행합니다.

풀에서 미러링 활성화

풀에서 미러링을 활성화하려면 다음을 수행합니다.

```
rbd mirror pool enable <pool-name> <mode>
```

예

data 라는 전체 풀 미러링을 활성화하려면 다음을 수행합니다.

```
# rbd mirror pool enable data pool
```

data 라는 풀에서 이미지 모드 미러링을 활성화하려면 다음을 수행합니다.

```
# rbd mirror pool enable data image
```

자세한 내용은 미러링 모드를 참조하십시오.

풀에서 미러링 비활성화

풀에서 미러링을 비활성화하려면 다음을 수행합니다.

```
rbd mirror pool disable <pool-name>
```

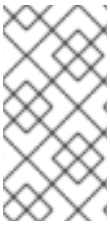
예제

data 라는 풀 미러링을 비활성화하려면 다음을 수행합니다.

```
# rbd mirror pool disable data
```

미러링을 비활성화하기 전에 피어 클러스터를 제거합니다. 자세한 내용은 4.2절. “풀 구성” 을 참조하십

시오.



참고

풀에서 미러링을 비활성화하면 미러링이 이미지 모드에서 별도로 활성화된 풀 내의 이미지에서도 해당 이미지를 비활성화합니다. 자세한 내용은 이미지 구성을 참조하십시오.

클러스터 피어 추가

rbd-mirror 데몬이 피어 클러스터를 검색하려면 풀에 피어를 등록해야 합니다.

```
rbd --cluster <cluster-name> mirror pool peer add <pool-name> <peer-client-name>@<peer-cluster-name> -n <client-name>
```

예제

site-a 클러스터를 **site-b** 클러스터에 피어로 추가하려면 **site-b** 클러스터의 클라이언트 노드에서 다음 명령을 실행합니다.

```
# rbd --cluster site-b mirror pool peer add data client.site-a@site-a -n client.site-b
```

피어에 대한 정보 보기

피어에 대한 정보를 보려면 다음을 수행합니다.

```
rbd mirror pool info <pool-name>
```

예제

```
# rbd mirror pool info data
Mode: pool
Peers:
  UUID                                 NAME   CLIENT
  7e90b4ce-e36d-4f07-8cbc-42050896825d site-a client.site-a
```

클러스터 피어 제거

미러링 피어 클러스터를 제거하려면 다음을 수행합니다.

```
rbd mirror pool peer remove <pool-name> <peer-uuid>
```

풀 이름과 피어 **UUID(Universally Unique Identifier)**를 지정합니다. 피어 **UUID**를 보려면 **rbd mirror pool info** 명령을 사용합니다.

예제

```
# rbd mirror pool peer remove data 7e90b4ce-e36d-4f07-8cbc-42050896825d
```

풀의 미러링 상태 가져오기

미러링 풀 요약을 가져오려면 다음을 수행합니다.

```
rbd mirror pool status <pool-name>
```

예제

데이터 풀 상태를 가져오려면 다음을 수행합니다.

```
# rbd mirror pool status data
health: OK
images: 1 total
```

풀의 모든 미러링 이미지에 대한 상태 세부 정보를 출력하려면 **--verbose** 옵션을 사용합니다.

4.3. 이미지 설정

이 장에서는 다음 작업을 수행하는 방법을 설명합니다.

- 이미지 미러링 활성화
- 이미지 미러링 비활성화
- 이미지 승격 및 데모

- 이미지 재동기화
- 단일 이미지의 미러링 상태 가져오기

단일 클러스터에서만 다음 명령을 실행합니다.

이미지 미러링 활성화

특정 이미지 미러링을 활성화하려면 다음을 수행합니다.

1. 두 피어 클러스터에서 이미지 모드에서 전체 풀의 미러링을 활성화합니다. 자세한 내용은 4.2 절. “풀 구성” 을 참조하십시오.
2. 그런 다음 풀 내에서 특정 이미지에 대한 미러링을 명시적으로 활성화합니다.

```
rbd mirror image enable <pool-name>/<image-name>
```

예제

데이터 풀에서 **image2** 이미지의 미러링을 활성화하려면 다음을 수행합니다.

```
# rbd mirror image enable data/image2
```

이미지 미러링 비활성화

특정 이미지의 미러링을 비활성화하려면 다음을 수행합니다.

```
rbd mirror image disable <pool-name>/<image-name>
```

예제

데이터 풀에서 **image2** 이미지 미러링을 비활성화하려면 다음을 수행합니다.

```
# rbd mirror image disable data/image2
```

이미지 승격 및 데모

이미지를 비기본(non-primary)으로 시연하려면 다음을 수행합니다.

```
rbd mirror image demote <pool-name>/<image-name>
```

예제

데이터 풀에서 **image2** 이미지를 시연하려면 다음을 수행합니다.

```
# rbd mirror image demote data/image2
```

이미지를 기본으로 승격하려면 다음을 수행합니다.

```
rbd mirror image promote <pool-name>/<image-name>
```

예제

데이터 풀에서 **image2** 이미지를 승격하려면 다음을 수행합니다.

```
# rbd mirror image promote data/image2
```

사용하는 미러링 유형에 따라 단방향 미러링을 사용하여 재해 복구 또는 양방향 미러링 이 있는 재해 복구를 참조하십시오.

--force 옵션을 사용하여 비기본 이미지를 강제로 승격합니다.

```
# rbd mirror image promote --force data/image2
```

클러스터 장애 또는 통신 중단으로 인해 **demotion**을 피어 **Ceph** 클러스터에 전파할 수 없는 경우 강제 승격을 사용합니다. 자세한 내용은 **Non-Orderlydown** 이후 페일오버 을 참조하십시오.

참고

승격 후에도 이미지가 유효하지 않기 때문에 동기화 중인 비기본 이미지를 강제로 승격하지 마십시오.

이미지 재동기화

기본 이미지에 재동기화를 요청하려면 다음을 수행합니다.

```
rbd mirror image resync <pool-name>/<image-name>
```

예제

데이터 풀에서 **image2** 이미지의 재동기화를 요청하려면 다음을 수행합니다.

```
# rbd mirror image resync data/image2
```

두 피어 클러스터 간에 일관되지 않은 상태의 경우 **rbd-mirror** 데몬은 불일치를 유발하는 이미지를 미러링하지 않습니다. 이 문제를 해결하는 방법에 대한 자세한 내용은 재해에서 복구하는 섹션을 참조하십시오. 사용하는 미러링 유형에 따라 단방향 미러링을 사용하여 재해 복구 또는 양방향 미러링 이 있는 재해 복구를 참조하십시오.

단일 이미지의 미러링 상태 가져오기

미러링된 이미지의 상태를 가져오려면 다음을 수행합니다.

```
rbd mirror image status <pool-name>/<image-name>
```

예제

데이터 풀에서 **image2** 이미지의 상태를 가져오려면 다음을 수행합니다.

```
# rbd mirror image status data/image2
image2:
  global_id:   703c4082-100d-44be-a54a-52e6052435a5
  state:       up+replaying
  description: replaying, master_position=[object_number=0, tag_tid=3, entry_tid=0], mirror_position=
[object_number=0, tag_tid=3, entry_tid=0], entries_behind_master=0
  last_update: 2019-04-23 13:39:15
```

4.4. 1-WAY MIRRORING 구성

단방향 미러링은 하나의 클러스터의 기본 이미지가 보조 클러스터에서 복제됨을 의미합니다. 보조 클러스터에서 복제된 이미지는 비기본입니다. 즉 블록 장치 클라이언트는 이미지에 쓸 수 없습니다.

참고

단방향 미러링은 여러 보조 사이트를 지원합니다. 여러 보조 사이트에서 단방향 미러링을 구성하려면 각 보조 클러스터에서 다음 절차를 반복합니다.

참고

단방향 미러링은 이미지의 충돌성 사본을 유지 관리하는 데 적합합니다. 단방향 미러링은 단방향 미러링을 사용할 때 클러스터가 실패할 수 없기 때문에 **OpenStack**에 자동 장애 조치 및 장애 복구를 위해 보조 이미지를 사용하는 등 모든 상황에 적합하지 않을 수 있습니다. 이러한 시나리오에서는 양방향 미러링을 사용합니다. 자세한 내용은 4.5절. “2-Way Mirroring 구성” 을 참조하십시오.

다음 절차에서는 다음을 가정합니다.

- 두 개의 클러스터가 있고 기본 클러스터에서 보조 클러스터로 이미지를 복제하려고 합니다. 이 절차의 목적으로 기본 이미지가 있는 클러스터를 **site-a** 클러스터로 참조하고 이미지를 **site-b** 클러스터로 복제하려는 클러스터를 참조하여 두 클러스터를 구분합니다. **Ceph Storage** 클러스터 설치에 대한 자세한 내용은 ***Red Hat Enterprise Linux용 설치 가이드*** 또는 ***Ubuntu용 설치 가이드를 참조하십시오***.

- **site-b** 클러스터에는 **rbd-mirror** 데몬이 실행되는 클라이언트 노드가 연결되어 있습니다. 이 데몬은 이미지를 **site-b** 클러스터에 동기화하기 위해 **site-a** 클러스터에 연결합니다. **Ceph** 클라이언트 설치에 대한 자세한 내용은 ***Red Hat Enterprise Linux용 설치 가이드*** 또는 ***Ubuntu용 설치 가이드를 참조하십시오***.

- 두 클러스터에 동일한 이름의 풀이 생성됩니다. 아래 예제에서 풀은 데이터 이름이 지정됩니다. 자세한 내용은 *스토리지 전략 가이드* 또는 **Red Hat Ceph Storage 3**의 풀 장을 참조하십시오.

- 풀에는 미러링하려는 이미지가 포함되어 있으며 저널링이 활성화되어 있습니다. 아래 예제에서 이미지의 이름은 **image1** 및 **image2** 입니다. 자세한 내용은 **MeteringConfig** 활성화를 참조하십시오.

블록 장치 미러링을 구성하는 방법은 다음 두 가지가 있습니다.

- 풀 미러링: 풀 내의 모든 이미지를 미러링하려면 풀 미러링 구성 프로세스를 사용합니다.
- 이미지 미러링: 풀 내에서 선택한 이미지를 미러링하려면 이미지 미러링 구성 프로세스를 사용합니다.

풀 미러링 구성

1. 데이터 풀 내의 모든 이미지에 독점 잠금 및 저널링이 활성화되어 있는지 확인합니다. 자세한 내용은 4.1절. “journalctling 활성화” 을 참조하십시오.

2. site-b 클러스터의 클라이언트 노드에 rbd-mirror 패키지를 설치합니다. 패키지는 Red Hat Ceph Storage 3 Tools 리포지토리에서 제공합니다.

   Red Hat Enterprise Linux

   ```
   # yum install rbd-mirror
   ```

   Ubuntu

   ```
   $ sudo apt-get install rbd-mirror
   ```

3. site-b 클러스터의 클라이언트 노드에서 CLUSTER 옵션을 적절한 파일에 추가하여 클러스터 이름을 지정합니다. Red Hat Enterprise Linux에서 /etc/sysconfig/ceph 파일을 업데이트하고 Ubuntu에서 /etc/default/ceph 파일을 적절하게 업데이트합니다.

   ```
   CLUSTER=site-b
   ```

4. 두 클러스터에서 모두 데이터 풀에 액세스하고 해당 키 링을 < cluster-name>.client.<user-name>.keyring 파일에 출력할 권한이 있는 사용자를 생성합니다.

   a. site-a 클러스터의 모니터 호스트에서 client.site-a 사용자를 생성하고 keyring을 site-a.client.site-a.keyring 파일에 출력합니다.

```
# ceph auth get-or-create client.site-a mon 'profile rbd' osd 'profile rbd pool=data' -o
/etc/ceph/site-a.client.site-a.keyring
```

b. **site-b** 클러스터의 모니터 호스트에서 **client.site-b** 사용자를 생성하고 인증 키를 **site-b.client.site-b.keyring** 파일에 출력합니다.

```
# ceph auth get-or-create client.site-b mon 'profile rbd' osd 'profile rbd pool=data' -o
/etc/ceph/site-b.client.site-b.keyring
```

5. **Ceph** 구성 파일과 새로 생성된 **RBD** 키링 파일을 사이트에서 모니터링 노드에서 **site- b** 모니터 및 클라이언트 노드로 복사합니다.

```
# scp /etc/ceph/ceph.conf <user>@<site-b_mon-host-name>:/etc/ceph/site-a.conf
# scp /etc/ceph/site-a.client.site-a.keyring <user>@<site-b_mon-host-name>:/etc/ceph/

# scp /etc/ceph/ceph.conf <user>@<site-b_client-host-name>:/etc/ceph/site-a.conf
# scp /etc/ceph/site-a.client.site-a.keyring <user>@<site-b_client-host-name>:/etc/ceph/
```

참고

Ceph 구성 파일을 **site-a** 에서 **site-b** 모니터로 전송하는 **scp** 명령은 파일 이름을 **site-a.conf** 로 변경합니다. 키링 파일 이름은 동일하게 유지됩니다.

6. **site-b** 클러스터 클라이언트 노드에서 **ceph.conf** 를 가리키는 **site-b.conf** 라는 심볼릭 링크를 생성합니다.

```
# cd /etc/ceph
# ln -s ceph.conf site-b.conf
```

7. **site-b** 클라이언트 노드에서 **rbd-mirror** 데몬을 활성화하고 시작합니다.

```
systemctl enable ceph-rbd-mirror.target
systemctl enable ceph-rbd-mirror@<client-id>
systemctl start ceph-rbd-mirror@<client-id>
```

& lt;client-id >를 **rbd-mirror** 데몬이 사용할 **Ceph Storage** 클러스터 사용자로 변경합니다. 사용자에게 클러스터에 대한 적절한 **cephx** 액세스 권한이 있어야 합니다. 자세한 내용은 **Red Hat Ceph Storage 3** *관리 가이드의 사용자 관리* 장을 참조하십시오.

site-b 를 사용한 사전 수행 예제에 따라 다음 명령을 실행합니다.

```
# systemctl enable ceph-rbd-mirror.target
# systemctl enable ceph-rbd-mirror@site-b
# systemctl start ceph-rbd-mirror@site-b
```

8. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-a** 클러스터에 있는 데이터 풀 미러링을 활성화합니다.

```
# rbd mirror pool enable data pool
```

미러링이 성공적으로 활성화되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: pool
Peers: none
```

9. **site-a** 클러스터를 **site-b** 클러스터의 클라이언트 노드에서 다음 명령을 실행하여 **site-b** 클러스터의 피어로 추가합니다.

```
# rbd --cluster site-b mirror pool peer add data client.site-a@site-a -n client.site-b
```

피어가 성공적으로 추가되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: pool
Peers:
  UUID                                 NAME   CLIENT
  7e90b4ce-e36d-4f07-8cbc-42050896825d site-a client.site-a
```

10. 잠시 후 **image1** 및 **image2** 이미지의 상태를 확인합니다. 상태 **up+replaying** 에 있는 경우 미러링이 제대로 작동합니다. **site-b** 클러스터의 모니터 노드에서 다음 명령을 실행합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   7d486c3f-d5a1-4bee-ae53-6c4f1e0c8eac
  state:       up+replaying
  description: replaying, master_position=[object_number=3, tag_tid=1, entry_tid=3],
mirror_position=[object_number=3, tag_tid=1, entry_tid=3], entries_behind_master=0
  last_update: 2019-04-22 13:19:27
```

```
# rbd mirror image status data/image2
image2:
  global_id:   703c4082-100d-44be-a54a-52e6052435a5
  state:       up+replaying
  description: replaying, master_position=[object_number=3, tag_tid=1, entry_tid=3],
mirror_position=[], entries_behind_master=3
  last_update: 2019-04-22 13:19:19
```

이미지 미러링 구성

1. 데이터 풀 내에서 선택한 이미지를 미러링하려면 독점 잠금 및 저널링이 활성화되어 있는지 확인합니다. 자세한 내용은 4.1절. “journalctling 활성화” 을 참조하십시오.

2. 풀 미러링 구성 프로세스에서 2~7단계를 수행합니다.

3. **site-a** 클러스터의 모니터 노드에서 데이터 풀의 이미지 미러링을 활성화합니다.

```
# rbd mirror pool enable data image
```

미러링이 성공적으로 활성화되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: image
Peers: none
```

4. **site-b** 클러스터의 클라이언트 노드에서 **site-a** 클러스터를 피어로 추가합니다.

```
# rbd --cluster site-b mirror pool peer add data client.site-a@site-a -n client.site-b
```

피어가 성공적으로 추가되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: image
Peers:
  UUID                                 NAME   CLIENT
  9c1da891-b9f4-4644-adee-6268fe398bf1 site-a client.site-a
```

5. **site-a** 클러스터의 모니터 노드에서 **image1** 및 **image2** 이미지의 이미지 미러링을 명시적으로 활성화합니다.

```
# rbd mirror image enable data/image1
Mirroring enabled
# rbd mirror image enable data/image2
Mirroring enabled
```

6. 잠시 후 **image1** 및 **image2** 이미지의 상태를 확인합니다. 상태 **up+replaying** 에 있는 경우 미러링이 제대로 작동합니다. **site-b** 클러스터의 모니터 노드에서 다음 명령을 실행합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+replaying
  description: replaying, master_position=[object_number=3, tag_tid=1, entry_tid=3],
mirror_position=[object_number=3, tag_tid=1, entry_tid=3], entries_behind_master=0
  last_update: 2019-04-12 17:24:04
```

```
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+replaying
  description: replaying, master_position=[object_number=3, tag_tid=1, entry_tid=3],
mirror_position=[object_number=3, tag_tid=1, entry_tid=3], entries_behind_master=0
  last_update: 2019-04-12 17:23:51
```

4.5. 2-WAY MIRRORING 구성

양방향 미러링을 사용하면 두 클러스터 간의 어느 방향으로도 이미지를 복제할 수 있습니다. 두 클러스터에서 동일한 이미지에 대한 변경 사항을 쓸 수 없으며 변경 사항이 뒤로 전파되도록 할 수 없습니다. 클러스터에서 이미지가 승격되거나 거부되어 쓰기 가능한 위치와 동기화 위치를 변경합니다.

다음 절차에서는 다음을 가정합니다.

- 두 개의 클러스터가 있으며 두 클러스터 간에 이미지를 복제할 수 있어야 합니다. 아래 예제에서 클러스터는 **site-a** 및 **site-b** 클러스터라고 합니다. **Ceph Storage** 클러스터 설치에 대한 자세한 내용은 ***Red Hat Enterprise Linux용 설치 가이드*** 또는 ***Ubuntu용 설치 가이드를 참조하십시오*.**

- 두 클러스터에 모두 **RBD -mirror** 데몬이 실행되는 클라이언트 노드가 연결되어 있습니다. **site-b** 클러스터의 데몬이 **site-a** 클러스터에 연결하여 이미지를 **site-b** 에 동기화하고, **site-a** 클러스터의 데몬은 **site-b** 클러스터에 연결하여 이미지를 **site-a** 에 동기화합니다. **Ceph** 클라이언트 설치에 대한 자세한 내용은 ***Red Hat Enterprise Linux용 설치 가이드*** 또는 ***Ubuntu 용 설치 가이드를 참조하십시오*.**

-

두 클러스터에 동일한 이름의 풀이 생성됩니다. 아래 예제에서 풀은 데이터 이름이 지정됩니다. 자세한 내용은 *스토리지 전략 가이드* 또는 **Red Hat Ceph Storage 3**의 풀 장을 참조하십시오.

- 풀에는 미러링하려는 이미지가 포함되어 있으며 저널링이 활성화되어 있습니다. 아래 예제에서 이미지 이름은 **image1** 및 **image2** 입니다. 자세한 내용은 **MeteringConfig** 활성화를 참조하십시오.

블록 장치 미러링을 구성하는 방법은 다음 두 가지가 있습니다.

- 풀 미러링: 풀 내의 모든 이미지를 미러링하려면 바로 아래에 *풀 미러링* 구성을 따릅니다.
- 이미지 미러링: 풀 내에서 선택한 이미지를 미러링하려면 아래 *이미지 미러링 구성을* 따릅니다.

풀 미러링 구성

1. 데이터 풀 내의 모든 이미지에 독점 잠금 및 저널링이 활성화되어 있는지 확인합니다. 자세한 내용은 **4.1**절. **"journalctling** 활성화" 을 참조하십시오.

2. **One-Way *Mirroring* 구성의 동등한 풀 미러링 구성** 섹션에서 **2 - 7**단계를 수행하여 한 가지 미러링을설정합니다.

3. **site-a** 클러스터의 클라이언트 노드에 **rbd-mirror** 패키지를 설치합니다. 패키지는 **Red Hat Ceph Storage 3 Tools** 리포지토리에서 제공합니다.

   Red Hat Enterprise Linux

   ```
   # yum install rbd-mirror
   ```

   Ubuntu

   ```
   $ sudo apt-get install rbd-mirror
   ```

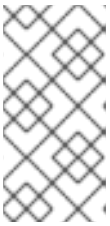
4. **site-a** 클러스터의 클라이언트 노드에서 **CLUSTER** 옵션을 적절한 파일에 추가하여 클러스터 이름을 지정합니다. **Red Hat Enterprise Linux**에서 **/etc/sysconfig/ceph** 파일을 업데이트하고

Ubuntu에서 **/etc/default/ceph** 파일을 적절하게 업데이트합니다.

```
CLUSTER=site-a
```

5. **site-b Ceph** 구성 파일 및 **RBD** 키링 파일을 **site-b** 모니터 및 클라이언트 노드로 복사합니다.

```
# scp /etc/ceph/ceph.conf <user>@<site-a_mon-host-name>:/etc/ceph/site-b.conf
# scp /etc/ceph/site-b.client.site-b.keyring root@<site-a_mon-host-name>:/etc/ceph/
# scp /etc/ceph/ceph.conf user@<site-a_client-host-name>:/etc/ceph/site-b.conf
# scp /etc/ceph/site-b.client.site-b.keyring user@<site-a_client-host-name>:/etc/ceph/
```



참고

site-b 모니터 노드에서 사이트**-b**로 **Ceph** 구성 파일을 전송하는 **scp** 명령은 파일의 이름을 **site-b.conf** 로 바꿉니다. 키링 파일 이름은 동일하게 유지됩니다.

6. **site-a RBD** 키링 파일을 **site-a monitor** 노드에서 **site-a** 클라이언트 노드로 복사합니다.

```
# scp /etc/ceph/site-a.client.site-a.keyring <user>@<site-a_client-host-name>:/etc/ceph/
```

7. **site-a** 클러스터 클라이언트 노드에서 **ceph.conf** 를 가리키는 **site-a.conf** 라는 심볼릭 링크를 생성합니다.

```
# cd /etc/ceph
# ln -s ceph.conf site-a.conf
```

8. **site-a** 클라이언트 노드에서 **rbd-mirror** 데몬을 활성화하고 시작합니다.

```
systemctl enable ceph-rbd-mirror.target
systemctl enable ceph-rbd-mirror@<client-id>
systemctl start ceph-rbd-mirror@<client-id>
```

여기서 **<client-id >**는 **rbd-mirror** 데몬이 사용할 **Ceph Storage** 클러스터 사용자입니다. 사용자에게 클러스터에 대한 적절한 **cephx** 액세스 권한이 있어야 합니다. 자세한 내용은 **Red Hat Ceph Storage 3** *관리 가이드의 사용자 관리* 장을 참조하십시오.

site-a 를 사용한 사전 수행 예제에 따라 다음 명령을 실행합니다.

```
# systemctl enable ceph-rbd-mirror.target
# systemctl enable ceph-rbd-mirror@site-a
# systemctl start ceph-rbd-mirror@site-a
```

9. **site-b 클러스터의 모니터 노드에서 다음 명령을 실행하여 site-b 클러스터에 있는 데이터 풀 미러링을 활성화합니다.**

```
# rbd mirror pool enable data pool
```

미러링이 성공적으로 활성화되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: pool
Peers: none
```

10. **site-b 클러스터의 클라이언트 노드에서 다음 명령을 실행하여 site-a 클러스터의 피어로 site-b 클러스터를 추가합니다.**

```
# rbd --cluster site-a mirror pool peer add data client.site-b@site-b -n client.site-a
```

피어가 성공적으로 추가되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: pool
Peers:
  UUID                                 NAME   CLIENT
  dc97bd3f-869f-48a5-9f21-ff31aafba733 site-b client.site-b
```

11. **site-a 클러스터의 클라이언트 노드에서 미러링 상태를 확인합니다.**

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-16 15:45:31
```

```
# rbd mirror image status data/image2
image1:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-16 15:55:33
```

이미지 상태가 **up+stopped** 여야 합니다. 여기에서 **up** 은 **rbd-mirror** 데몬이 실행 중이고 중지 되었음을 나타냅니다. 이는 이미지가 다른 클러스터에서 복제 대상이 아님을 의미합니다. 이미지가 이 클러스터에서 기본이기 때문입니다.

참고

이전 버전에서는 단방향 미러링을 설정할 때 이미지가 **site-b** 로 복제되도록 구성되었습니다. 이를 위해 **site-b** 클라이언트 노드에 **rbd-mirror** 를 설치하여 **site-a** 에서 **site-b** 까지 "가져오기" 업데이트할 수 있습니다. 이 시점에서 사이트-클러스터는 미러링할 준비가되었지만 필요한 상태에 있지 않습니다. **site-a** 의 이미지가 **demoted**되고 **site-b** 의 이미지가 승격된 경우 다른 방향으로 미러링이 시작됩니다. 이미지를 승격하고 시연하는 방법에 대한 자세한 내용은 이미지 설정을 참조하십시오.

이미지 미러링 구성

1. 아직 설정되지 않은 경우 하나의 방법으로 미러링을 설정합니다.

   a. **One-Way *Mirroring* 구성의 풀 미러링 구성** 섹션에서 **2 -7**단계를 수행합니다.

   b. **One-Way *Mirroring* 구성의 이미지 미러링 구성** 섹션에서 **3 -5**단계를 수행합니다.

2. **2- Way *Mirroring* 구성의 풀 미러링 구성** 섹션에서 **3 - 7** 단계를 수행합니다. 이 섹션은 즉시 위에 있습니다.

3. **site-b** 클러스터의 클라이언트 노드에서 다음 명령을 실행하여 **site-a** 클러스터의 피어로 **site-b** 클러스터를 추가합니다.

```
# rbd --cluster site-a mirror pool peer add data client.site-b@site-b -n client.site-a
```

피어가 성공적으로 추가되었는지 확인합니다.

```
# rbd mirror pool info data
Mode: pool
Peers:
  UUID                                 NAME   CLIENT
  dc97bd3f-869f-48a5-9f21-ff31aafba733 site-b client.site-b
```

4. **site-a** 클러스터의 클라이언트 노드에서 미러링 상태를 확인합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-16 15:45:31
```

```
# rbd mirror image status data/image2
image1:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-16 15:55:33
```

이미지 상태가 **up+stopped** 여야 합니다. 여기에서 **up** 은 **rbd-mirror** 데몬이 실행 중이고 중지 되었음을 나타냅니다. 이는 이미지가 다른 클러스터에서 복제 대상이 아님을 의미합니다. 이미지가 이 클러스터에서 기본이기 때문입니다.

참고

이전 버전에서는 단방향 미러링을 설정할 때 이미지가 **site-b** 로 복제되도록 구성되었습니다. 이를 위해 **site-b** 클라이언트 노드에 **rbd-mirror** 를 설치하여 **site-a** 에서 **site-b** 까지 "가져오기" 업데이트할 수 있습니다. 이 시점에서 사이트-클러스터는 미러링할 준비가되었지만 필요한 상태에 있지 않습니다. **site-a** 의 이미지가 **demoted**되고 **site-b** 의 이미지가 승격된 경우 다른 방향으로 미러링이 시작됩니다. 이미지를 승격하고 시연하는 방법에 대한 자세한 내용은 이미지 설정을 참조하십시오.

4.6. 지연된 복제

단방향 복제를 사용하든 양방향 복제를 사용하든 관계없이 **RADOS Block Device (RBD)** 미러링 이미지 간에 복제를 지연시킬 수 있습니다. 보조 이미지로 복제되기 전에 주 이미지를 원치 않는 변경을 해야 하는 경우 큐시온 시간 창을 원하는 경우 지연된 복제를 구현할 수 있습니다.

지연된 복제를 구현하려면 대상 클러스터 내의 **rbd-mirror** 데몬에서 **rbd** 미러링 재생 지연 **=** 초 단위로 **<minimum delay >** 설정을 설정해야 합니다. 이 설정은 **rbd-mirror** 데몬에서 사용하거나 개별 이미지 기반으로 사용하는 **ceph.conf** 파일 내에서 전역적으로 적용할 수 있습니다.

특정 이미지의 지연된 복제를 사용하려면 기본 이미지에서 다음 **rbd CLI** 명령을 실행합니다.

```
rbd image-meta set  <image-spec> conf_rbd_mirroring_replay_delay <minimum delay in seconds>
```

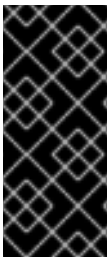
예를 들어 풀 **vms**의 이미지 **vm-1**에서 최소 **10**분 간의 최소 복제 지연을 설정하려면 다음을 실행합니다.

```
rbd image-meta set vms/vm-1 conf_rbd_mirroring_replay_delay 600
```

4.7. 단방향 미러링을 통한 재해 복구

단방향 미러링을 사용하는 경우 재해에서 복구하려면 다음 절차를 사용하십시오. 기본 클러스터가 종료된 후 보조 클러스터로 장애 조치(**failover**)하는 방법 및 장애 복구 방법을 보여줍니다. 종료는 순서 또는 순서가 아닌 순서일 수 있습니다.

다음 예제에서는 기본 클러스터를 **site-a** 클러스터라고 하며 보조 클러스터를 **site-b** 클러스터라고 합니다. 또한 클러스터에는 두 개의 이미지 **image1** 및 **image2** 가 있는 데이터 풀이 있습니다.



중요

단방향 미러링은 여러 보조 사이트를 지원합니다. 추가 보조 클러스터를 사용하는 경우 장애 조치할 보조 클러스터 중 하나를 선택합니다. 장애 복구 중에 동일한 클러스터에서 동기화합니다.

사전 요구 사항

- 실행 중인 클러스터 두 개 이상
- 풀 미러링 또는 이미지 미러링은 한 가지 방식으로 구성됩니다.

순서 종료 후 장애 조치(**failover**)

1. 기본 이미지를 사용하는 모든 클라이언트를 중지합니다. 이 단계는 이미지를 사용하는 고객에 따라 다릅니다. 예를 들어 이미지를 사용하는 모든 **OpenStack** 인스턴스에서 볼륨을 분리합니다. **Red Hat OpenStack Platform 13**의 *스토리지 가이드* 의 *블록 스토리지 및 볼륨* 장을 참조하십시오.
2. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-a** 클러스터에 있는 기본 이미지를 시연합니다.

```
# rbd mirror image demote data/image1
# rbd mirror image demote data/image2
```

3. **site-b** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-b** 클러스터에 있는 비기본 이미지를 승격합니다.

```
# rbd mirror image promote data/image1
# rbd mirror image promote data/image2
```

4. 잠시 후 **site-b** 클러스터의 모니터 노드에서 이미지의 상태를 확인합니다. **up+stopped** 상태를 표시해야 하며 설명은 **primary** 로 지정해야 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-17 13:18:36
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-17 13:18:36
```

Non-Orderlydown 이후 페일오버

1. 기본 클러스터가 다운되었는지 확인합니다.

2. 기본 이미지를 사용하는 모든 클라이언트를 중지합니다. 이 단계는 이미지를 사용하는 고객에 따라 다릅니다. 예를 들어 이미지를 사용하는 모든 **OpenStack** 인스턴스에서 볼륨을 분리합니다. **Red Hat OpenStack Platform 10**의 *스토리지 가이드* 의 *블록 스토리지 및 볼륨* 장을 참조하십시오.

3. **site-b** 클러스터의 모니터 노드에서 기본이 아닌 이미지를 승격합니다. **demotion**을 **site-a** 클러스터로 전파할 수 없기 때문에 **--force** 옵션을 사용합니다.

```
# rbd mirror image promote --force data/image1
# rbd mirror image promote --force data/image2
```

4. **site-b** 클러스터의 모니터 노드에서 이미지 상태를 확인합니다. **up+stopping_replay** 상태가 표시되고 설명은 **force promoted** 를 지정해야 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopping_replay
  description: force promoted
  last_update: 2019-04-17 13:25:06
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopping_replay
  description: force promoted
  last_update: 2019-04-17 13:25:06
```

failback 준비

이전 기본 클러스터가 복구되면 오류가 발생했습니다.

두 클러스터가 원래 단방향 미러링용으로만 구성된 경우 장애 조치를 위해 이미지를 반대 방향으로 복제하려면 미러링을 위해 기본 클러스터를 구성해야 합니다.

1. **site-a** 클러스터의 클라이언트 노드에 **rbd-mirror** 패키지를 설치합니다. 패키지는 **Red Hat Ceph Storage 3 Tools** 리포지토리에서 제공합니다.

   Red Hat Enterprise Linux

   ```
   # yum install rbd-mirror
   ```

   Ubuntu

   ```
   $ sudo apt-get install rbd-mirror
   ```

2. **site-a** 클러스터의 클라이언트 노드에서 **CLUSTER** 옵션을 적절한 파일에 추가하여 클러스터 이름을 지정합니다. **Red Hat Enterprise Linux**에서 **/etc/sysconfig/ceph** 파일을 업데이트하고 **Ubuntu**에서 **/etc/default/ceph** 파일을 적절하게 업데이트합니다.

   ```
   CLUSTER=site-b
   ```

3. **site-b Ceph** 구성 파일 및 **RBD** 키링 파일을 **site-b** 모니터 및 클라이언트 노드로 복사합니다.

   ```
   # scp /etc/ceph/ceph.conf <user>@<site-a_mon-host-name>:/etc/ceph/site-b.conf
   # scp /etc/ceph/site-b.client.site-b.keyring root@<site-a_mon-host-name>:/etc/ceph/
   ```

```
# scp /etc/ceph/ceph.conf user@<site-a_client-host-name>:/etc/ceph/site-b.conf
# scp /etc/ceph/site-b.client.site-b.keyring user@<site-a_client-host-name>:/etc/ceph/
```

참고

Ceph 구성 파일을 **site-b** 모니터 노드에서 **site-a.conf**로 전송하는 **scp** 명령은 파일의 이름을 **site-a.conf** 로 변경합니다. 키링 파일 이름은 동일하게 유지됩니다.

4. **site-a RBD** 키링 파일을 **site-a monitor** 노드에서 **site-a** 클라이언트 노드로 복사합니다.

```
# scp /etc/ceph/site-a.client.site-a.keyring <user>@<site-a_client-host-name>:/etc/ceph/
```

5. **site-a** 클라이언트 노드에서 **rbd-mirror** 데몬을 활성화하고 시작합니다.

```
systemctl enable ceph-rbd-mirror.target
systemctl enable ceph-rbd-mirror@<client-id>
systemctl start ceph-rbd-mirror@<client-id>
```

& lt;client-id >를 **rbd-mirror** 데몬이 사용할 **Ceph Storage** 클러스터 사용자로 변경합니다. 사용자에게 클러스터에 대한 적절한 **cephx** 액세스 권한이 있어야 합니다. 자세한 내용은 **Red Hat Ceph Storage 3** *관리 가이드의 사용자 관리* 장을 참조하십시오.

site-a 를 사용한 선행 예제에 따라 명령은 다음과 같습니다.

```
# systemctl enable ceph-rbd-mirror.target
# systemctl enable ceph-rbd-mirror@site-a
# systemctl start ceph-rbd-mirror@site-a
```

6. **site-a** 클러스터의 클라이언트 노드에서 **site-b** 클러스터를 피어로 추가합니다.

```
# rbd --cluster site-a mirror pool peer add data client.site-b@site-b -n client.site-a
```

여러 보조 클러스터를 사용하는 경우 보조 클러스터가 장애 조치(**Failover**)하도록 선택한 경우에만 장애 조치(**failback**)가 추가되어야 합니다.

7. **site-a** 클러스터의 모니터 노드에서 **site-b** 클러스터가 피어로 성공적으로 추가되었는지 확인합니다.

```
# rbd mirror pool info -p data
Mode: image
Peers:
 UUID                                 NAME   CLIENT
 d2ae0594-a43b-4c67-a167-a36c646e8643 site-b client.site-b
```

failback

이전 기본 클러스터가 복구되면 오류가 발생했습니다.

1. **site-a** 클러스터의 모니터 노드에서 이미지가 여전히 기본인지 확인합니다.

```
# rbd info data/image1
# rbd info data/image2
```

명령의 출력에서 **primary: true** 또는 **mirroring primary: false** 를 찾아 상태를 확인합니다.

2. **site-a** 클러스터의 모니터 노드에서 다음과 같이 명령을 실행하여 기본으로 나열된 이미지를 데모합니다.

```
# rbd mirror image demote data/image1
```

3. 순서가 아닌 종료가 있는 경우에만 이미지를 다시 동기화합니다. 사이트 **-** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-b** 에서 **site-a** 로 이미지를 다시 동기화합니다.

```
# rbd mirror image resync data/image1
Flagged image for resync from primary
# rbd mirror image resync data/image2
Flagged image for resync from primary
```

   a. 잠시 후 이미지가 **up+replaying** 상태인지 확인하여 이미지를 다시 동기화해야 합니다. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 상태를 확인합니다.

```
# rbd mirror image status data/image1
# rbd mirror image status data/image2
```

4. **site-b** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-b** 클러스터의 이미지를 시연합니다.

```
# rbd mirror image demote data/image1
# rbd mirror image demote data/image2
```

참고

보조 클러스터가 여러 개인 경우 승격된 보조 클러스터에서만 수행해야 합니다.

5. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-a** 클러스터에 있는 이전 기본 이미지를 승격합니다.

```
# rbd mirror image promote data/image1
# rbd mirror image promote data/image2
```

6. 클러스터의 모니터 노드에서 이미지 상태를 확인합니다. **up+stopped** 상태를 표시해야 하며 설명은 로컬 이미지가 기본 이미지라고 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-22 11:14:51
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-22 11:14:51
```

양방향 미러링 제거

위의 *장애 조치 준비* 섹션에서 양방향 미러링 기능에서는 **site-b** 클러스터에서 **site-a** 클러스터로 동기화할 수 있도록 구성되었습니다. **failback**이 완료되면 이러한 함수를 사용하지 않도록 설정할 수 있습니다.

1. **site-a** 클러스터에서 피어로 **site-b** 클러스터를 제거합니다.

```
$ rbd mirror pool peer remove data client.remote@remote --cluster local
# rbd --cluster site-a mirror pool peer remove data client.site-b@site-b -n client.site-a
```

2. **site-a** 클라이언트에서 **rbd-mirror** 데몬을 중지하고 비활성화합니다.

```
systemctl stop ceph-rbd-mirror@<client-id>
systemctl disable ceph-rbd-mirror@<client-id>
systemctl disable ceph-rbd-mirror.target
```

예를 들면 다음과 같습니다.

```
# systemctl stop ceph-rbd-mirror@site-a
# systemctl disable ceph-rbd-mirror@site-a
# systemctl disable ceph-rbd-mirror.target
```

추가 리소스

- 이미지 **demoting,** 승격 및 재동기화에 대한 자세한 내용은 블록 장치 가이드 의 이미지 구성을 참조하십시오.

4.8. 양방향 미러링을 통한 재해 복구

양방향 미러링을 사용하는 경우 재해에서 복구하려면 다음 절차를 사용하십시오. 기본 클러스터가 종료된 후 보조 클러스터의 미러링된 데이터로 장애 조치(failover)하는 방법 및 장애 복구 방법을 보여줍니다. 종료는 순서 또는 순서가 아닌 순서일 수 있습니다.

다음 예제에서는 기본 클러스터를 **site-a** 클러스터라고 하며 보조 클러스터를 **site-b** 클러스터라고 합니다. 또한 클러스터에는 두 개의 이미지 **image1** 및 **image2** 가 있는 데이터 풀이 있습니다.

사전 요구 사항

- 실행 중인 클러스터 두 개 이상
- 풀 미러링 또는 이미지 미러링은 한 가지 방식으로 구성됩니다.

순서 종료 후 장애 조치(failover)

1. 기본 이미지를 사용하는 모든 클라이언트를 중지합니다. 이 단계는 이미지를 사용하는 고객에 따라 다릅니다. 예를 들어 이미지를 사용하는 모든 **OpenStack** 인스턴스에서 볼륨을 분리합니다. **Red Hat OpenStack Platform 10**의 *스토리지 가이드* 의 *블록 스토리지 및 볼륨* 장을 참조하십시오.
2. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-a** 클러스터에 있는 기본 이미지를 시연합니다.

```
# rbd mirror image demote data/image1
# rbd mirror image demote data/image2
```

3. **site-b** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-b** 클러스터에 있는 비기본 이미지를 승격합니다.

```
# rbd mirror image promote data/image1
# rbd mirror image promote data/image2
```

4. 잠시 후 **site-b** 클러스터의 모니터 노드에서 이미지의 상태를 확인합니다. **up+stopped** 상태를 표시하고 기본으로 나열되어야 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-17 16:04:37
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-17 16:04:37
```

5. 이미지에 대한 액세스를 다시 시작합니다. 이 단계는 이미지를 사용하는 고객에 따라 다릅니다.

Non-Orderlydown 이후 페일오버

1. 기본 클러스터가 다운되었는지 확인합니다.

2. 기본 이미지를 사용하는 모든 클라이언트를 중지합니다. 이 단계는 이미지를 사용하는 고객에 따라 다릅니다. 예를 들어 이미지를 사용하는 모든 **OpenStack** 인스턴스에서 볼륨을 분리합니다. **Red Hat OpenStack Platform 10**의 *스토리지 가이드* 의 *블록 스토리지 및 볼륨* 장을 참조하십시오.

3. **site-b** 클러스터의 모니터 노드에서 기본이 아닌 이미지를 승격합니다. **demotion**을 **site-a** 클러스터로 전파할 수 없기 때문에 **--force** 옵션을 사용합니다.

```
# rbd mirror image promote --force data/image1
# rbd mirror image promote --force data/image2
```

4. **site-b** 클러스터의 모니터 노드에서 이미지 상태를 확인합니다. **up+stopping_replay** 상태가 표시되고 설명은 **force promoted** 를 지정해야 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopping_replay
  description: force promoted
  last_update: 2019-04-17 13:25:06
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopping_replay
  description: force promoted
  last_update: 2019-04-17 13:25:06
```

failback

이전 기본 클러스터가 복구되면 오류가 발생했습니다.

1. **site-b** 클러스터의 모니터 노드에서 이미지 상태를 다시 확인합니다. 이 명령은 **up-stopped** 상태를 표시해야 하며 설명은 로컬 이미지가 **primary** 인 것으로 표시되어야 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-22 17:37:48
# rbd mirror image status data/image2
image2:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-22 17:38:18
```

2. **site-a** 클러스터의 모니터 노드에서 이미지가 여전히 기본인지 확인합니다.

```
# rbd info data/image1
# rbd info data/image2
```

명령의 출력에서 **primary: true** 또는 **mirroring primary: false** 를 찾아 상태를 확인합니다.

3. **site-a** 클러스터의 모니터 노드에서 다음과 같이 명령을 실행하여 기본으로 나열된 이미지를 데모합니다.

```
# rbd mirror image demote data/image1
```

4. 순서가 아닌 종료가 있는 경우에만 이미지를 다시 동기화합니다. 사이트 - 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-b** 에서 **site-a** 로 이미지를 다시 동기화합니다.

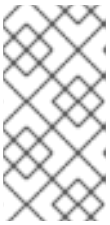
```
# rbd mirror image resync data/image1
Flagged image for resync from primary
# rbd mirror image resync data/image2
Flagged image for resync from primary
```

5. 잠시 후 이미지가 **up+replaying** 상태인지 확인하여 이미지를 다시 동기화해야 합니다. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 상태를 확인합니다.

```
# rbd mirror image status data/image1
# rbd mirror image status data/image2
```

6. **site-b** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-b** 클러스터의 이미지를 시연합니다.

```
# rbd mirror image demote data/image1
# rbd mirror image demote data/image2
```



참고

보조 클러스터가 여러 개인 경우 승격된 보조 클러스터에서만 수행해야 합니다.

7. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-a** 클러스터에 있는 이전 기본 이미지를 승격합니다.

```
# rbd mirror image promote data/image1
# rbd mirror image promote data/image2
```

8. 클러스터의 모니터 노드에서 이미지 상태를 확인합니다. **up+stopped** 상태를 표시해야 하며 설명은 로컬 이미지가 기본 이미지라고 합니다.

```
# rbd mirror image status data/image1
image1:
  global_id:   08027096-d267-47f8-b52e-59de1353a034
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-22 11:14:51
# rbd mirror image status data/image2
image2:
  global_id:   596f41bc-874b-4cd4-aefe-4929578cc834
  state:       up+stopped
  description: local image is primary
  last_update: 2019-04-22 11:14:51
```

추가 리소스

- 이미지 **demoting,** 승격 및 재동기화에 대한 자세한 내용은 블록 장치 가이드 의 이미지 구성 을 참조하십시오**.**

4.9. 미러링을 사용하여 인스턴스 업데이트

Ceph Block Device mirroring을 사용하여 비동기 업데이트로 클러스터를 업데이트할 때 업데이트의 설치 지침을 따릅니다**.** 그런 다음 **Ceph Block Device** 인스턴스를 다시 시작합니다**.**

참고

인스턴스를 다시 시작하는 데 필요한 순서가 없습니다**. Red Hat**은 기본 이미지를 사용한 다음 미러링된 풀을 가리키는 인스턴스가 있는 인스턴스를 다시 시작하는 것이 좋습니다**.**

5장. LIBRBD(PYTHON)

rbd python 모듈은 **RBD** 이미지에 파일와 유사한 액세스를 제공합니다. 이 기본 제공 툴을 사용하려면 **rbd** 및 **rados** 모듈을 가져와야 합니다.

생성 및 이미지에 쓰기

1. **RADOS**에 연결하고 **IO** 컨텍스트를 엽니다.

```
cluster = rados.Rados(conffile='my_ceph.conf')
cluster.connect()
ioctx = cluster.open_ioctx('mypool')
```

2. 이미지를 생성하는 데 사용하는 **:class:rbd.RBD** 오브젝트를 인스턴스화합니다.

```
rbd_inst = rbd.RBD()
size = 4 * 1024**3  # 4 GiB
rbd_inst.create(ioctx, 'myimage', size)
```

3. 이미지에서 **I/O**를 수행하려면 **:class:rbd.Image** 오브젝트를 인스턴스화합니다.

```
image = rbd.Image(ioctx, 'myimage')
data = 'foo' * 200
image.write(data, 0)
```

이 명령은 이미지의 처음 **600**바이트에 **'foo'**를 씁니다. 데이터는 **:type:unicode - librbd** 는 **:c:type:char** 보다 광범위한 문자를 처리하는 방법을 알 수 없습니다.

4. 이미지**, IO** 컨텍스트**, RADOS**에 대한 연결을 종료합니다.

```
image.close()
ioctx.close()
cluster.shutdown()
```

안전하려면 각 호출이 별도의 **:finally** 블록에 있어야합니다.

```
import rados
import rbd

cluster = rados.Rados(conffile='my_ceph_conf')
```

```
try:
    ioctx = cluster.open_ioctx('my_pool')
    try:
        rbd_inst = rbd.RBD()
        size = 4 * 1024**3  # 4 GiB
        rbd_inst.create(ioctx, 'myimage', size)
        image = rbd.Image(ioctx, 'myimage')
        try:
            data = 'foo' * 200
            image.write(data, 0)
        finally:
            image.close()
    finally:
        ioctx.close()
finally:
    cluster.shutdown()
```

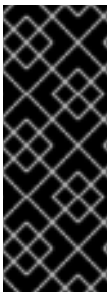
이는 번거로움일 수 있으므로 Rados,Ioctx, Image 클래스는 자동으로 닫히거나 종료되는 컨텍스트 관리자로 사용할 수 있습니다. 컨텍스트 관리자로 이를 사용하면 위의 예는 다음과 같습니다.

```
with rados.Rados(conffile='my_ceph.conf') as cluster:
    with cluster.open_ioctx('mypool') as ioctx:
        rbd_inst = rbd.RBD()
        size = 4 * 1024**3  # 4 GiB
        rbd_inst.create(ioctx, 'myimage', size)
        with rbd.Image(ioctx, 'myimage') as image:
            data = 'foo' * 200
            image.write(data, 0)
```

6장. 커널 모듈 작업

중요

커널 모듈 작업을 사용하려면 실행 중인 **Ceph** 클러스터가 있어야 합니다.



중요

RHEL(Red Hat Enterprise Linux)을 제외한 **Linux** 배포판의 클라이언트는 허용되지만 지원되지 않습니다. 이러한 클라이언트를 사용할 때 클러스터(예: **MDS)**에 문제가 있는 경우 **Red Hat**은 이러한 문제를 해결하지만 원인이 클라이언트 측에 있는 경우 커널 공급 업체에 의해 문제를 해결해야 합니다.

6.1. 이미지 목록 가져오기

블록 장치 이미지를 마운트하려면 먼저 이미지 목록을 반환합니다.

이렇게 하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd list
```

6.2. 블록 장치 매핑

rbd 를 사용하여 이미지 이름을 커널 모듈에 매핑합니다. 이미지 이름, 풀 이름 및 사용자 이름을 지정해야 합니다. **RBD** 가 아직 로드되지 않은 경우 **RBD** 커널 모듈을 로드합니다.

이렇게 하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd map {image-name} --pool {pool-name} --id {user-name}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd map --pool rbd myimage --id admin
```

cephx 인증을 사용하는 경우 시크릿도 지정해야 합니다. 인증 키 또는 시크릿을 포함하는 파일에서 가져올 수 있습니다.

이렇게 하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd map --pool rbd myimage --id admin --keyring /path/to/keyring
[root@rbd-client ~]# rbd map --pool rbd myimage --id admin --keyfile /path/to/file
```

6.3. 맵핑 블록 장치 표시

rbd 명령을 사용하여 커널 모듈에 매핑된 블록 장치 이미지를 표시하려면 **showmapped** 옵션을 지정합니다.

이렇게 하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd showmapped
```

6.4. 블록 장치 매핑 해제

rbd 명령을 사용하여 블록 장치 이미지를 매핑 해제하려면 **unmap** 옵션과 장치 이름(블록 장치 이미지 이름과 동일)을 지정합니다.

이렇게 하려면 다음을 실행합니다.

```
[root@rbd-client ~]# rbd unmap /dev/rbd/{poolname}/{imagename}
```

예를 들면 다음과 같습니다.

```
[root@rbd-client ~]# rbd unmap /dev/rbd/rbd/foo
```

7장. 블록 장치 구성 참조

7.1. 일반 설정

rbd_op_threads

설명

블록 장치 작업 스레드 수입니다.

유형

정수

Default

1

주의

1 보다 큰 숫자로 설정하면 데이터가 손상될 수 있으므로 **rbd_op_threads** 의 기본값을 변경하지 마십시오.

rbd_op_thread_timeout

설명

블록 장치 작업 스레드의 타임아웃(초)입니다.

유형

정수

Default

60

rbd_non_blocking_aio

설명

true 인 경우 **Ceph**는 차단을 방지하기 위해 작업자 스레드에서 블록 장치 비동기 **I/O** 작업을

처리합니다.

유형

부울

Default

true

rbd_concurrent_management_ops

설명

최대 동시 관리 작업 수(예: 이미지 삭제 또는 크기 조정)

유형

정수

Default

10

rbd_request_timed_out_seconds

설명

유지 관리 요청 시간 초과 전의 시간(초)입니다.

유형

정수

Default

30

rbd_clone_copy_on_read

설명

true 로 설정하면 **copy-on-read** 복제가 활성화됩니다.

유형

부울

Default

false

rbd_enable_alloc_hint

설명

true 인 경우 할당 힌트가 활성화되고 블록 장치는 예상되는 크기 오브젝트를 나타내기 위해 **OSD** 백엔드에 힌트를 발행합니다.

유형

부울

Default

true

rbd_skip_partial_discard

설명

true 인 경우 블록 장치는 오브젝트 내에서 범위를 삭제하려고 할 때 범위 **0**을 건너뜁니다.

유형

부울

Default

false

rbd_tracing

설명

Linux Trace Toolkit Next Generation User Space Tracer(LTTng-UST) 추적 지점을 활성화하려면 이 옵션을 **true** 로 설정합니다. 자세한 내용은 **RBD Replay** 기능을 사용하여 **RADOS Block Device (RBD)** 워크로드 추적을 참조하십시오.

유형

부울

Default

false

rbd_validate_pool

설명

RBD 호환성을 위해 빈 풀의 유효성을 검사하려면 이 옵션을 true 로 설정합니다.

유형

부울

Default

true

rbd_validate_names

설명

이미지 사양을 검증하려면 이 옵션을 true 로 설정합니다.

유형

부울

Default

true

7.2. 기본 설정

이미지를 생성하기 위한 기본 설정을 재정의할 수 있습니다. Ceph는 포맷 2 및 스트라이핑이 없는 이미지를 생성합니다.

rbd_default_format

설명

다른 형식이 지정되지 않은 경우 기본 형식(2)입니다. 형식 1 은 모든 버전의 librbd 및 kernel 모듈과 호환되는 새 이미지의 원본 형식이지만 복제와 같은 최신 기능은 지원하지 않습니다. 형식 2 는 librbd 에서 지원되며 버전 3.11 이후의 커널 모듈 (종종 제외)에서 지원됩니다. 형식 2 는 복제 지원을 추가하고 향후 더 많은 기능을 허용하도록 더 쉽게 확장할 수 있습니다.

유형

정수

Default

2

rbd_default_order

설명

다른 순서가 지정되지 않은 경우 기본 순서입니다.

유형

정수

Default

22

rbd_default_stripe_count

설명

다른 스트라이프 개수가 지정되지 않은 경우 기본 스트라이프 수가 계산됩니다. 기본값을 변경하려면 **v2** 기능을 스트라이핑해야 합니다.

유형

64비트 서명되지 않은 **Integer**

Default

0

rbd_default_stripe_unit

설명

다른 스트라이프 장치가 지정되지 않은 경우 기본 스트라이프 단위입니다. 단위를 **0** (즉, 오브젝트 크기)으로 변경하려면 스트라이핑 **v2** 기능이 필요합니다.

유형

64비트 서명되지 않은 **Integer**

Default

0

rbd_default_features

설명

블록 장치 이미지를 생성할 때 기본 기능이 활성화됩니다. 이 설정은 형식 2 이미지에만 적용됩니다. 설정은 다음과 같습니다.

1: 계층화 지원 계층화를 사용하면 복제를 사용할 수 있습니다.

2: 스트라이핑 v2 지원. 스트라이핑은 여러 오브젝트에 데이터를 분산합니다. 스트라이핑은 순차적 읽기/쓰기 워크로드에 대한 병렬 처리를 수행하는 데 도움이 됩니다.

4: 독점적인 잠금 지원 이 기능을 활성화하면 쓰기 전에 클라이언트가 개체에 대한 잠금을 받아야 합니다.

8: 오브젝트 맵 지원. 블록 장치는 씬 프로비저닝된(meaning)이며 실제로 존재하는 데이터만 저장합니다. 오브젝트 맵 지원은 어떤 오브젝트가 실제로 존재하는지 추적하는 데 도움이 됩니다(드라이브에 저장된 데이터). 오브젝트 맵 지원을 활성화하면 복제 또는 스파스로 채워진 이미지를 가져오고 내보낼 수 있는 I/O 작업의 속도가 빨라집니다.

16: 빠른 지원. 신속한 지원은 오브젝트 맵 지원 및 독점적인 잠금 지원에 따라 다릅니다. 개체 맵에 다른 속성을 추가하여 이미지의 스냅샷과 스냅샷의 실제 데이터 사용량 간에 diffs를 훨씬 빠르게 생성할 수 있습니다.

32: Deep-flatten 지원 Deep-flatten은 이미지 자체 외에 모든 이미지 스냅샷에서 rbd를 평면화 합니다. 이미지 스냅샷은 상위에 계속 의존하므로 스냅샷이 삭제될 때까지 상위 항목을 삭제할 수 없습니다. Deep-flatten은 스냅샷이 있는 경우에도 복제와는 별도로 상위 항목을 만듭니다.

64: 저널링 지원 저널링은 이미지에 대한 모든 수정 사항을 발생하는 순서대로 기록합니다. 이렇게 하면 원격 이미지의 충돌 일치 미러를 로컬에서 사용할 수 있습니다.

활성화된 기능은 숫자 설정의 합계입니다.

유형

정수

Default

61 - 계층화, 전용 잠금, 개체 맵, **fast-diff** 및 딥 플래그를 사용할 수 있습니다.

중요

현재 기본 설정은 **RBD** 커널 드라이버 또는 이전 **RBD** 클라이언트와 호환되지 않습니다.

rbd_default_map_options

설명

대부분의 옵션은 디버깅 및 벤치마킹에 주로 유용합니다. 자세한 내용은 맵 옵션 아래의 **man rbd** 를 참조하십시오.

유형

문자열

Default

""

7.3. 캐시 설정

Ceph 블록 장치(즉, **librbd**)의 사용자 공간 구현은 **Linux** 페이지 캐시를 활용할 수 없으므로 **RBD** 캐싱이라는 자체 메모리 내 캐싱을 포함합니다. **RBD** 캐싱은 하드 디스크 캐싱이 잘 적용된 것처럼 작동합니다. **OS**에서 장벽 또는 플러시 요청을 보내면 모든 더티 데이터가 **OSD**에 작성됩니다. 즉, 나중 쓰기 캐싱을 사용하는 것이 플러시(즉, **Linux** 커널 **>= 2.6.32**)를 올바르게 전송하는 **VM**과 함께 잘 작동되는 물리적 하드 디스크를 사용하는 것처럼 안전합니다. 캐시는 **Least Recently Used (LRU)** 알고리즘을 사용하고, 나중 쓰기 모드에서는 처리량 향상을 위해 연속된 요청을 병합할 수 있습니다.

Ceph는 **RBD**에 **write-back** 캐싱을 지원합니다. 활성화하려면 **ceph.conf** 파일의 **[client]** 섹션에 **rbd cache = true** 를 추가합니다. 기본적으로 **librbd** 는 캐싱을 수행하지 않습니다. 쓰기 및 읽기는 스토리지 클러스터에 직접 이동하고 쓰기는 데이터가 모든 복제본의 디스크에 있을 때만 반환됩니다. 캐싱이 활성화되면 **rbd cache max** 더 이상 플러시되지 않은 바이트가 없으면 쓰기가 즉시 반환됩니다. 이 경우 쓰기는 쓰기를 트리거하고 충분한 바이트가 플러시될 때까지 차단됩니다.

Ceph는 RBD에 write-through 캐싱을 지원합니다. 캐시 크기를 설정할 수 있으며, 나중 쓰기 캐싱에서 캐싱을 통해 쓰기로 전환하도록 대상 및 제한을 설정할 수 있습니다. 쓰기-스루 모드를 활성화하려면 rbd 캐시 최대 더티 를 0으로 설정합니다. 즉 쓰기는 데이터가 모든 복제본의 디스크에 있을 때만 반환되지만 읽기는 캐시에서 가져올 수 있습니다. 캐시는 클라이언트의 메모리에 있으며 각 RBD 이미지에는 자체 캐시가 있습니다. 캐시는 클라이언트에 로컬이므로 다른 사용자가 이미지에 액세스하는 경우 일관성이 없습니다. RBD에서 ScanSetting 또는 OCFS를 실행하면 캐싱이 활성화된 경우 작동하지 않습니다.

RBD에 대한 ceph.conf 파일 설정은 구성 파일의 [client] 섹션에 설정해야 합니다. 설정은 다음과 같습니다.

rbd cache

설명

RADOS 블록 장치(RBD)에 대한 캐싱을 활성화합니다.

유형

부울

필수 항목

없음

Default

true

RBD 캐시 크기

설명

RBD 캐시 크기(바이트)입니다.

유형

64비트 정수

필수 항목

없음

Default

32MiB

RBD 캐시 최대 더티

설명

캐시가 나중 쓰기를 트리거하는 바이트 단위의 더티 제한입니다. 0 인 경우 동시 쓰기 캐싱을 사용합니다.

유형

64비트 정수

필수 항목

없음

제약 조건

rbd 캐시 크기 보다 작아야 합니다.

Default

24MiB

RBD 캐시 대상 더티

설명

캐시가 데이터 스토리지에 데이터 쓰기를 시작하기 전에 더티 대상입니다. 캐시에 대한 쓰기를 차단하지 않습니다.

유형

64비트 정수

필수 항목

없음

제약 조건

rbd 캐시 최대 더티 보다 작아야 합니다.

Default

16MiB

RBD 캐시 최대 더티 기간

설명

writeback이 시작되기 전에 더티 데이터가 캐시에 있는 시간(초)입니다.

유형

float

필수 항목

없음

Default

1.0

rbd_cache_max_dirty_object

설명

objects에 대한 더티 제한 - rbd_cache_size 에서 자동 계산의 경우 0 으로 설정됩니다.

유형

정수

Default

0

rbd_cache_block_writes_upfront

설명

true 인 경우 aio_write 호출이 완료되기 전에 캐시에 대한 쓰기를 차단합니다. false 인 경우 aio_completion 이 호출되기 전에 차단됩니다.

유형

부울

Default

false

플러시할 때까지 RBD 캐시 쓰기를 수행합니다.

설명

동시 쓰기 모드로 시작하고 첫 번째 플러시 요청이 수신된 후 나중 쓰기로 전환합니다. 이를 활성화하는 것은 **rbd**에서 실행되는 **VM**이 **2.6.32** 이전의 **virtio** 드라이버와 같이 플러시를 전송하기에 너무 오래 된 경우 보수적이지만 안전한 설정입니다.

유형

부울

필수 항목

없음

Default

true

7.4. 상위/ECDHE 읽기 설정

rbd_balance_snap_reads

설명

Ceph는 일반적으로 기본 **OSD**에서 오브젝트를 읽습니다. 읽기는 변경할 수 없으므로 이 기능을 사용하여 기본 **OSD**와 복제본 간에 **snap** 읽기의 균형을 조정할 수 있습니다.

유형

부울

Default

false

rbd_localize_snap_reads

설명

rbd_balance_snap_reads 는 스냅샷을 읽기 위해 복제본을 무작위화하는 반면, **rbd_localize_snap_reads** 를 활성화하면 블록 장치는 **snapshot**을 읽기 위한 가장 가까운(로컬) **OSD**를 찾습니다.

유형

부울

Default

false

rbd_balance_parent_reads

설명

Ceph는 일반적으로 기본 OSD에서 오브젝트를 읽습니다. 읽기는 변경할 수 없으므로 이 기능을 사용하여 기본 OSD와 복제본 간에 상위 읽기의 균형을 조정할 수 있습니다.

유형

부울

Default

false

rbd_localize_parent_reads

설명

rbd_balance_parent_reads 는 상위 읽기를 위해 복제본을 무작위화하는 반면 rbd_localize_parent_reads 를 활성화하면 블록 장치는 parent를 읽기 위한 가장 가까운(로컬) OSD를 찾기 위해 CloudEvent 맵을 찾습니다.

유형

부울

Default

true

7.5. 미리 보기 설정

RBD는 읽기/사전 패치를 지원하여 작은 순차적 읽기를 최적화합니다. 일반적으로 VM의 경우 게스트 OS에서 이 작업을 처리해야 하지만 부트 로더에는 효율적인 읽기 문제가 발생하지 않을 수 있습니다. 캐싱이 비활성화되면 미리 읽기가 자동으로 비활성화됩니다.

RBD readahead 트리거 요청

설명

미리 읽기를 트리거하는 데 필요한 순차적 읽기 요청 수입니다.

유형

정수

필수 항목

없음

Default

10

RBD readahead 최대 바이트

설명

미리 읽기 요청의 최대 크기입니다. 0인 경우 미리 읽기가 비활성화됩니다.

유형

64비트 정수

필수 항목

없음

Default

512 KiB

RBD readahead가 바이트 후 비활성화

설명

RBD 이미지에서 많은 바이트를 읽은 후에는 이미지가 종료될 때까지 해당 이미지에 대해 읽기-헤드가 비활성화됩니다. 이렇게 하면 게스트 OS가 부팅되면 미리 읽기를 대신할 수 있습니다. 0인 경우 미리 읽기가 설정되어 있으면 됩니다.

유형

64비트 정수

필수 항목

없음

Default

50 MiB

7.6. 블랙리스트 설정

rbd_blacklist_on_break_lock

설명

잠금이 중단된 클라이언트를 블랙리스트로 지정할지 여부입니다.

유형

부울

Default

true

rbd_blacklist_expire_seconds

설명

OSD 기본값에 대해 블랙리스트에 추가할 시간(초)입니다.

유형

정수

Default

0

7.7. SCANSETTING 설정

rbd_journal_order

설명

저널 오브젝트 최대 크기를 계산하도록 전환할 비트 수입니다. 값은 12 에서 64 사이입니다.

유형

32비트 서명되지 않은 Integer

Default

24

rbd_journal_splay_width

설명

활성 저널 오브젝트 수입니다.

유형

32비트 서명되지 않은 Integer

Default

4

rbd_journal_commit_age

설명

커밋 시간 간격(초)입니다.

유형

부동 소수점 수를 두 배로 높입니다.

Default

5

rbd_journal_object_flush_interval

설명

저널 오브젝트당 최대 보류 중인 커밋 수입니다.

유형

정수

Default

0

rbd_journal_object_flush_bytes

설명

journal 오브젝트당 최대 보류 중인 바이트 수입니다.

유형

정수

Default

0

rbd_journal_object_flush_age

설명

보류 중인 커밋의 최대 시간 간격(초)입니다.

유형

부동 소수점 수를 두 배로 높입니다.

Default

0

rbd_journal_pool

설명

저널 오브젝트의 풀을 지정합니다.

유형

문자열

Default

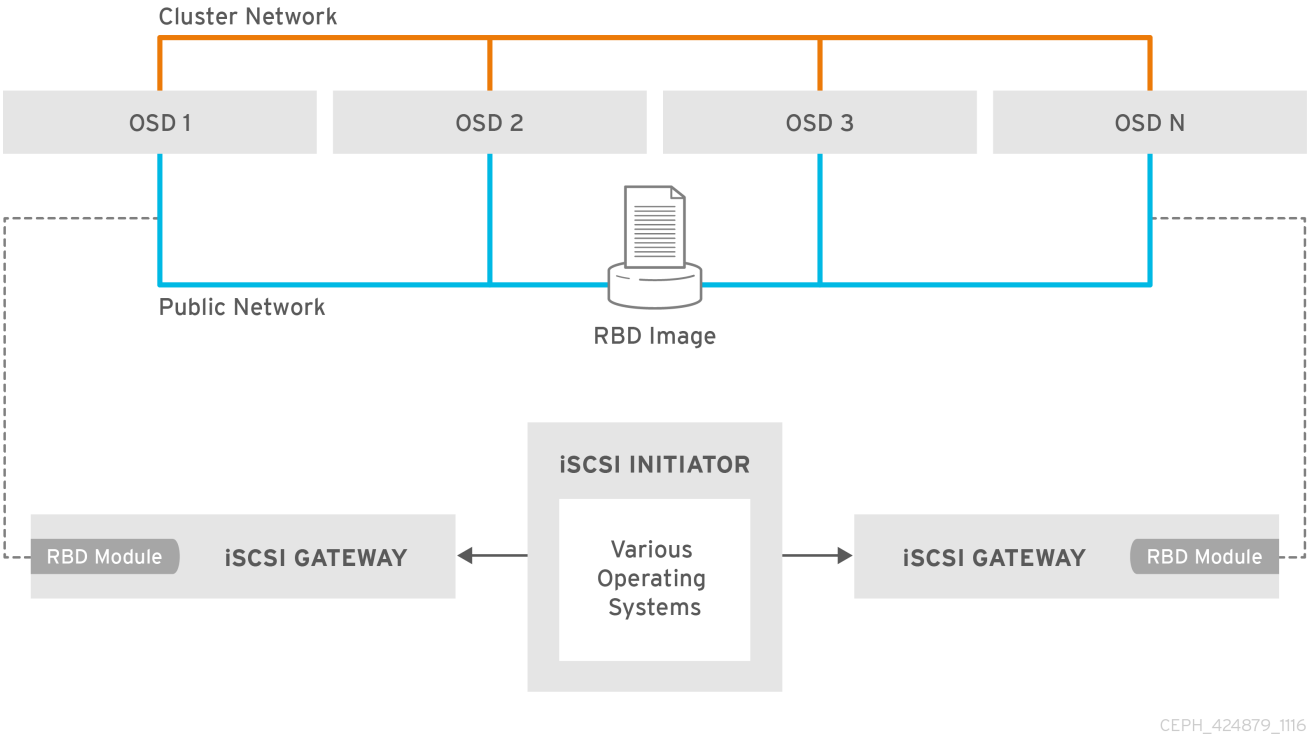
""

8장. iSCSI 게이트웨이 사용

iSCSI 게이트웨이는 **Red Hat Ceph Storage**를 **iSCSI** 표준과 통합하여 **RADOS Block Device (RBD)** 이미지를 **SCSI** 디스크로 내보내는 **HA**(고가용성) **iSCSI** 대상을 제공합니다. **iSCSI** 프로토콜을 사용하면 클라이언트(**initiator**)가 **TCP/IP** 네트워크를 통해 **SCSI** 명령(대상)에 **SCSI** 명령을 보낼 수 있습니다. 이를 통해 **Microsoft Windows**와 같은 이기종 클라이언트가 **Red Hat Ceph Storage** 클러스터에 액세스할 수 있습니다.

각 **iSCSI** 게이트웨이는 **Linux IO** 대상 커널 하위 시스템(**LIO**)을 실행하여 **iSCSI** 프로토콜 지원을 제공합니다. **LIO**는 **tIsMU**(사용자 공간 패스스루)를 사용하여 **Ceph**의 **librbd** 라이브러리와 상호 작용하여 **RBD** 이미지를 **iSCSI** 클라이언트에 노출합니다. **Ceph**의 **iSCSI** 게이트웨이를 사용하면 기존 **SAN**(**Storage Area Network**)의 모든 기능과 이점으로 완전히 통합된 블록 스토리지 인프라를 효과적으로 실행할 수 있습니다.

그림 8.1. Ceph iSCSI 게이트웨이 HA 설계



8.1. iSCSI 대상 요구사항

Red Hat Ceph Storage HA(고가용성) **iSCSI** 게이트웨이 솔루션에는 **OSD**를 감지하기 위해 게이트웨이 노드 수, 메모리 용량 및 타이머 설정이 필요합니다.

필요한 노드 수

최소 두 개의 **iSCSI** 게이트웨이 노드를 설치합니다. 복원력 및 **I/O** 처리를 늘리려면 최대 **4**개의 **iSCSI** 게이트웨이 노드를 설치합니다.

메모리 요구 사항

RBD 이미지의 메모리 풋프린트가 큰 크기로 증가할 수 있습니다. iSCSI 게이트웨이 노드에 매핑된 각 RBD 이미지는 약 90MB의 메모리를 사용합니다. iSCSI 게이트웨이 노드에 매핑된 각 RBD 이미지를 지원하기에 충분한 메모리가 있는지 확인합니다.

다운 OSD 탐지

Ceph Monitor 또는 OSD에 대한 특정 iSCSI 게이트웨이 옵션은 없지만, 이니시에이터 시간 초과 가능성을 줄이기 위해 OSD를 감지하기 위해 기본 타이머를 줄이는 것이 중요합니다. 이니시에이터 시간 초과 가능성 을 줄이기 위해 OSD를 감지하기 위한 타이머 감소 의 지침을 따르십시오.

추가 리소스

- 자세한 내용은 Red Hat Ceph Storage 하드웨어 선택 가이드를 참조하십시오.
- 자세한 내용은 블록 장치 가이드에서 OSD를 감지하기 위한 타이머 설정 감소를 참조하십시오.

8.2. OSD 중단을 위한 타이머 설정 감소

OSD 다운을 감지하기 위해 타이머 설정을 줄여야 하는 경우가 있습니다. 예를 들어 Red Hat Ceph Storage를 iSCSI 게이트웨이로 사용하는 경우 OSD를 감지하기 위해 타이머 설정을 줄임으로써 이니시에이터 시간 초과 가능성을 줄일 수 있습니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.

절차

1. 새 타이머 설정을 사용하도록 Ansible을 구성합니다.

   a. 다음과 같이 group_vars/all.yml 파일에 ceph_conf_overrides 섹션을 추가하거나 osd:로 시작하는 모든 행을 포함하도록 기존 ceph_conf_overrides 섹션을 편집합니다.

```
ceph_conf_overrides:
  osd:
    osd_client_watch_timeout: 15
```

```
    osd_heartbeat_grace: 20
    osd_heartbeat_interval: 5
```

site.yml Ansible 플레이북이 OSD 노드에 대해 실행되면 위의 설정이 ceph.conf 구성 파일에 추가됩니다.

b.

Ansible을 사용하여 ceph.conf 파일을 업데이트하고 모든 OSD 노드에서 OSD 데몬을 다시 시작합니다. Ansible 관리 노드에서 다음 명령을 실행합니다.

```
[user@admin ceph-ansible]$ ansible-playbook --limit osds site.yml
```

2.

ceph_conf_overrides 에 설정된 타이머 설정이 동일한지 확인합니다.

하나 이상의 OSD에서 ceph daemon 명령을 사용하여 설정을 확인합니다.

```
# ceph daemon osd.OSD_ID config get osd_client_watch_timeout
# ceph daemon osd.OSD_ID config get osd_heartbeat_grace
# ceph daemon osd.OSD_ID config get osd_heartbeat_interval
```

예제:

```
[root@osd1 ~]# ceph daemon osd.0 config get osd_client_watch_timeout
{
   "osd_client_watch_timeout": "15"
}
[root@osd1 ~]#  ceph daemon osd.0 config get osd_heartbeat_grace
{
   "osd_heartbeat_grace": "20"
}
[root@osd1 ~]# ceph daemon osd.0 config get osd_heartbeat_interval
{
   "osd_heartbeat_interval": "5"
}
```

3.

선택 사항: OSD 데몬을 즉시 다시 시작할 수 없는 경우 Ceph 모니터 노드에서 또는 모든 OSD 노드에서 직접 온라인 업데이트를 수행합니다. OSD 데몬을 다시 시작하면 위에 설명된 대로 Ansible을 사용하여 ceph.conf 에 새 타이머 설정을 추가하여 재부팅 시 설정이 유지됩니다.

- **모니터 노드에서 OSD 타이머 설정을 온라인 업데이트하려면 다음을 수행합니다.**

```
# ceph tell osd.OSD_ID injectargs '--osd_client_watch_timeout 15'
# ceph tell osd.OSD_ID injectargs '--osd_heartbeat_grace 20'
# ceph tell osd.OSD_ID injectargs '--osd_heartbeat_interval 5'
```

예제:

```
[root@mon ~]# ceph tell osd.0 injectargs '--osd_client_watch_timeout 15'
[root@mon ~]# ceph tell osd.0 injectargs '--osd_heartbeat_grace 20'
[root@mon ~]# ceph tell osd.0 injectargs '--osd_heartbeat_interval 5'
```

- **OSD** 노드에서 **OSD** 타이머 설정을 온라인 업데이트하려면 다음을 수행합니다.

```
# ceph daemon osd.OSD_ID config set osd_client_watch_timeout 15
# ceph daemon osd.OSD_ID config set osd_heartbeat_grace 20
# ceph daemon osd.OSD_ID config set osd_heartbeat_interval 5
```

예제:

```
[root@osd1 ~]# ceph daemon osd.0 config set osd_client_watch_timeout 15
[root@osd1 ~]# ceph daemon osd.0 config set osd_heartbeat_grace 20
[root@osd1 ~]# ceph daemon osd.0 config set osd_heartbeat_interval 5
```

추가 리소스

- **Red Hat Ceph Storage**를 **iSCSI** 게이트웨이로 사용하는 방법에 대한 자세한 내용은 블록 장치 가이드의 **Ceph iSCSI** 게이트웨이 소개 를 참조하십시오.

8.3. ISCSI 대상 구성

일반적으로 **Ceph** 스토리지 클러스터에 대한 블록 수준 액세스는 **QEMU** 및 **librbd** 로 제한되었습니다. 이는 **OpenStack** 환경 내에서 채택하는 주요 활성화자입니다. **Ceph** 스토리지 클러스터에 대한 블록 수준 액세스는 이제 **iSCSI** 표준을 활용하여 데이터 스토리지를 제공할 수 있습니다.

사전 요구 사항

- **Red Hat Enterprise Linux 7.5** 이상.
- 실행 중인 **Red Hat Ceph Storage** 클러스터, 버전 **3.1** 이상.

- OSD 노드 또는 전용 노드와 함께 배치할 수 있는 iSCSI 게이트웨이 노드.
- iSCSI 게이트웨이 노드의 유효한 Red Hat Enterprise Linux 7 및 Red Hat Ceph Storage 3.3 인타이틀먼트/서브스크립션.
- iSCSI 프런트 엔드 트래픽과 Ceph 백엔드 트래픽의 네트워크 서브넷을 분리합니다.

Ceph iSCSI 게이트웨이 배포는 Ansible 또는 명령줄 인터페이스를 사용하여 수행할 수 있습니다.

- Ansible
- 명령줄 인터페이스

8.3.1. Ansible을 사용하여 iSCSI 대상 구성

요구 사항:

- Red Hat Enterprise Linux 7.5 이상.
- 실행 중인 Red Hat Ceph Storage 3 이상.

설치:

1. iSCSI 게이트웨이 노드에서 Red Hat Ceph Storage 3 Tools 리포지토리를 활성화합니다. 자세한 내용은 Red Hat *Enterprise Linux 설치 가이드* 의 *Red Hat Ceph Storage 리포지토리 활성화 섹션*을 참조하십시오.

   a. ceph-iscsi-config 패키지를 설치합니다.

   ```
   # yum install ceph-iscsi-config
   ```

2. Ansible 관리 노드에서 root 사용자로 다음 단계를 수행합니다.

   a.

Red Hat Ceph Storage 3 Tools 리포지토리를 활성화합니다. 자세한 내용은 **Red Hat *Enterprise Linux* 설치 가이드** *의* ***Red Hat Ceph Storage*** *리포지토리 활성화* *섹션을* 참조하십시오.

b.

ceph-ansible 패키지를 설치합니다.

```
# yum install ceph-ansible
```

c.

게이트웨이 그룹의 **/etc/ansible/hosts** 파일에 항목을 추가합니다.

```
[iscsigws]
ceph-igw-1
ceph-igw-2
```

참고

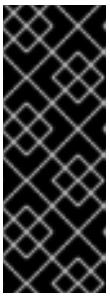
OSD 노드와 **iSCSI** 게이트웨이를 공동 배치하는 경우 **OSD** 노드를 **[iscsigws]** 섹션에 추가합니다.

구성:

ceph-ansible 패키지는 **iscsigws.yml.sample** 라는 **/usr/share/ceph-ansible/group_vars/** 디렉터리에 파일을 배치합니다.

1.

iscsigws.yml.sample 파일의 사본을 생성하고 이름을 **iscsigws.yml** 로 지정합니다.



중요

새 파일 이름(**iscsigws.yml**) 및 새 섹션 제목(**[iscsigws]**)은 **Red Hat Ceph Storage 3.1** 이상에만 적용할 수 있습니다. 이전 버전의 **Red Hat Ceph Storage** 에서 **3.1**로 업그레이드하면 이전 파일 이름(**iscsi-gws.yml**)과 이전 섹션 제목(**[iscsi-gws]**)을 계속 사용합니다.

2.

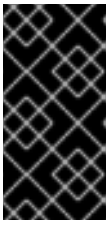
편집할 **iscsigws.yml** 파일을 엽니다.

3.

gateway_ip_list 옵션의 주석을 제거하고 **IPv4** 또는 **IPv6** 주소를 사용하여 값을 적절하게 업데이트합니다.

예를 들어 **10.172.19.21** 및 **10.172.19.22**의 **IPv4** 주소로 두 개의 게이트웨이를 추가하여 다음과 같이 **gateway_ip_list** 를 구성합니다.

```
gateway_ip_list: 10.172.19.21,10.172.19.22
```



중요

gateway_ip_list 옵션의 **IP** 주소를 제공해야 합니다. **IPv4** 및 **IPv6** 주소를 혼합하여 사용할 수 없습니다.

4. **rbd_devices** 변수의 주석을 제거하고 그에 따라 값을 업데이트합니다. 예를 들면 다음과 같습니다.

```
rbd_devices:
  - { pool: 'rbd', image: 'ansible1', size: '30G', host: 'ceph-1', state: 'present' }
  - { pool: 'rbd', image: 'ansible2', size: '15G', host: 'ceph-1', state: 'present' }
  - { pool: 'rbd', image: 'ansible3', size: '30G', host: 'ceph-1', state: 'present' }
  - { pool: 'rbd', image: 'ansible4', size: '50G', host: 'ceph-1', state: 'present' }
```

5. **client_connections** 변수의 주석을 해제하고 그에 따라 값을 업데이트합니다. 예를 들면 다음과 같습니다.

CHAP 인증 활성화 예

```
client_connections:
  - { client: 'iqn.1994-05.com.redhat:rh7-iscsi-client', image_list: 'rbd.ansible1,rbd.ansible2',
chap: 'rh7-iscsi-client/redhat', status: 'present' }
  - { client: 'iqn.1991-05.com.microsoft:w2k12r2', image_list: 'rbd.ansible4', chap:
'w2k12r2/microsoft_w2k12', status: 'absent' }
```

CHAP 인증 비활성화 예

```
client_connections:
  - { client: 'iqn.1991-05.com.microsoft:w2k12r2', image_list: 'rbd.ansible4', chap: '', status:
'present' }
```

```
  - { client: 'iqn.1991-05.com.microsoft:w2k16r2', image_list: 'rbd.ansible2', chap: '', status:
'present' }
```

중요

CHAP 비활성화는 Red Hat Ceph Storage 3.1 이상에서만 지원됩니다. Red Hat은 CHAP가 활성화되어 있고 일부 CHAP가 비활성화된 클라이언트 혼합을 지원하지 않습니다. 현재 표시된 모든 클라이언트는 CHAP이 활성화되어야 하며 CHAP이 비활성화되어 있어야 합니다.

6. 다음 **Ansible** 변수 및 설명을 검토하고 필요한 경우 적절하게 업데이트합니다.

표 8.1. iSCSI 게이트웨이 일반 변수

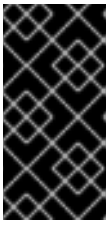
Variable	meaning/Purpose
seed_monitor	Each gateway needs access to the ceph cluster for rados and rbd calls. This means the iSCSI gateway must have an appropriate **/etc/ceph/** directory defined. The **seed_monitor** host is used to populate the iSCSI gateway's **/etc/ceph/** directory.
cluster_name	Define a custom storage cluster name.
gateway_keyring	Define a custom keyring name.
deploy_settings	If set to **true**, then deploy the settings when the playbook is ran.
perform_system_checks	This is a boolean value that checks for multipath and lvm configuration settings on each gateway. It must be set to true for at least the first run to ensure multipathd and lvm are configured properly.
gateway_iqn	This is the iSCSI IQN that all the gateways will expose to clients. This means each client will see the gateway group as a single subsystem.

Variable	meaning/Purpose
gateway_ip_list	The comma separated ip list defines the IPv4 or IPv6 addresses that will be used on the front end network for iSCSI traffic. This IP will be bound to the active target portal group on each node, and is the access point for iSCSI traffic. Each IP should correspond to an IP available on the hosts defined in the **iscsigws.yml** host group in **/etc/ansible/hosts**.
rbd_devices	This section defines the RBD images that will be controlled and managed within the iSCSI gateway configuration. Parameters like **pool** and **image** are self explanatory. Here are the other parameters: **size** = This defines the size of the RBD. You may increase the size later, by simply changing this value, but shrinking the size of an RBD is not supported and is ignored. **host** = This is the iSCSI gateway host name that will be responsible for the rbd allocation/resize. Every defined **rbd_device** entry must have a host assigned. **state** = This is typical Ansible syntax for whether the resource should be defined or removed. A request with a state of absent will first be checked to ensure the rbd is not mapped to any client. If the RBD is unallocated, it will be removed from the iSCSI gateway and deleted from the configuration.
client_connections	This section defines the iSCSI client connection details together with the LUN (RBD image) masking. Currently only CHAP is supported as an authentication mechanism. Each connection defines an **image_list** which is a comma separated list of the form **pool.rbd_image[,pool.rbd_image,...]**. RBD images can be added and removed from this list, to change the client masking. Note, that there are no checks done to limit RBD sharing across client connections.

표 8.2. iSCSI 게이트웨이 RBD-TARGET-API 변수

Variable	meaning/Purpose
api_user	API의 사용자 이름입니다. 기본값은 **admin** 입니다.
api_password	API를 사용하기 위한 암호입니다. 기본값은 **admin** 입니다.

Variable	meaning/Purpose
api_port	API 사용에 대한 TCP 포트 번호입니다. 기본값은 **5000** 입니다.
api_secure	value는 **true** 또는 **false** 일 수 있습니다. 기본값은 **false**입니다.
loop_delay	iSCSI 관리 오브젝트를 폴링하기 위한 유휴 상태 간격(초)을 제어합니다. 기본값은 **1** 입니다.
trusted_ip_list	API에 액세스할 수 있는 IPv4 또는 IPv6 주소 목록입니다. 기본적으로 iSCSI 게이트웨이 노드만 액세스할 수 있습니다.



중요

rbd_devices 의 경우 풀 이름 또는 이미지 이름에 마침표(.)가 있을 수 없습니다.

주의

게이트웨이 구성 변경은 한 번에 하나의 게이트웨이에서만 지원됩니다. 여러 게이트웨이를 통해 동시에 변경 사항을 실행하려고 하면 구성 불안정 및 불일치가 발생할 수 있습니다.

주의

Ansible은 **ceph-iscsi-cli** 패키지를 설치하고, **ansible-playbook** 명령이 실행될 때 **group_vars/iscsigws.yml** 파일의 설정에 따라 **/etc/ceph/iscsi-gateway.cfg** 파일을 생성한 다음 업데이트합니다. 명령줄 설치 절차를 사용하여 이전에 **ceph-iscsi-cli** 패키지를 설치한 경우 **iscsi-gateway.cfg** 파일의 기존 설정을 **group_vars/iscsigws.yml** 파일에 복사해야 합니다.

전체 **iscsigws.yml.sample** 파일을 보려면 부록 A. *샘플 iscsigws.yml 파일* 를 참조하십시오.

배포:

Ansible 관리 노드에서 **root** 사용자로 다음 단계를 수행합니다.

1. **Ansible** 플레이북을 실행합니다.

```
# cd /usr/share/ceph-ansible
# ansible-playbook site.yml
```

참고

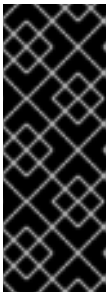
Ansible 플레이북은 **RPM** 종속 항목, **RBD** 생성 및 **Linux iSCSI** 대상 구성을 처리합니다.

주의

독립 실행형 **iSCSI** 게이트웨이 노드에서 올바른 **Red Hat Ceph Storage 3.3** 소프트웨어 리포지토리가 활성화되어 있는지 확인합니다. 사용할 수 없는 경우 잘못된 패키지가 설치됩니다.

2. 다음 명령을 실행하여 구성을 확인합니다.

```
# gwcli ls
```



중요

targetcli 유틸리티를 사용하여 구성을 변경하지 마십시오. 그러면 **ALUA** 잘못된 구성 및 경로 장애 조치 문제가 발생합니다. 데이터가 손상될 가능성이 있으며 **iSCSI** 게이트웨이에서 구성이 일치하지 않고 **WWN** 정보가 일치하지 않아 클라이언트 경로 문제가 발생할 수 있습니다.

서비스 관리:

ceph-iscsi-config 패키지에서는 구성 관리 논리와 **rbd-target-gw** 라는 **Systemd** 서비스를 설치합니다. **Systemd** 서비스가 활성화되면 부팅 시 **rbd-target-gw** 가 시작되고 **Linux iSCSI** 대상 상태가 복원됩니다. **Ansible** 플레이북을 사용하여 **iSCSI** 게이트웨이를 배포하면 대상 서비스가 비활성화됩니다.

```
# systemctl start rbd-target-gw
```

다음은 **rbd-target-gw Systemd** 서비스와 상호 작용하는 결과입니다.

```
# systemctl <start|stop|restart|reload> rbd-target-gw
```

reload

다시 로드 요청은 **rbd-target-gw** 가 구성을 다시 읽고 현재 실행 중인 환경에 적용하도록 강제 적용합니다. 변경 사항이 **Ansible**에서 모든 **iSCSI** 게이트웨이 노드로 병렬로 배포되므로 일반적으로 필요하지 않습니다.

중지

중지 요청은 게이트웨이의 포틸 인터페이스를 닫고 클라이언트에 대한 연결을 삭제하고 커널에서 현재 **Linux iSCSI** 대상 구성을 삭제합니다. 이렇게 하면 **iSCSI** 게이트웨이가 클린 상태로 반환됩니다. 클라이언트의 연결이 끊어지면 클라이언트 측 다중 경로 계층에 의해 활성 **I/O**가 다른 **iSCSI** 게이트웨이로 다시 예약됩니다.

관리:

/usr/share/ceph-ansible/group_vars/iscsigws.yml 파일에는 **Ansible** 플레이북이 지원하는 여러 운영 워크플로우가 있습니다.

주의

Red Hat은 **Ceph iSCSI** 게이트웨이 툴(예: **gwcli** 및 **ceph-ansible**)에서 내보낸 **RBD** 이미지 관리는 지원하지 않습니다. 또한 **rbd** 명령을 사용하여 **Ceph iSCSI** 게이트웨이에서 내보낸 **RBD** 이미지의 이름을 변경하거나 제거하면 불안정한 스토리지 클러스터가 발생할 수 있습니다.

주의

iSCSI 게이트웨이 구성에서 RBD 이미지를 제거하기 전에 운영 체제에서 스토리지 장치를 제거하는 표준 절차를 따르십시오.

Red Hat Enterprise Linux 7을 사용하는 클라이언트 및 시스템의 경우 장치 제거에 대한 자세한 내용은 Red Hat Enterprise Linux 7 *Storage 관리 가이드를* 참조하십시오.

표 8.3. 작업 워크플로우

I want to...	iscsigws.yml 파일을...으로 업데이트합니다.
RBD 이미지 추가	새 이미지와 함께 **rbd_devices** 섹션에 다른 항목을 추가합니다.
기존 RBD 이미지 크기 조정	**rbd_devices** 섹션에서 size 매개 변수를 업데이트합니다. 디스크의 새 크기를 선택하려면 클라이언트 측 작업이 필요합니다.
클라이언트 추가	**client_connections** 섹션에 항목을 추가합니다.
클라이언트에 RBD 추가	클라이언트의 **image_list** 변수에 관련 RBD **pool.image** 이름을 추가합니다.
클라이언트에서 RBD 제거	clients **image_list** 변수에서 RBD **pool.image** 이름을 제거합니다.
시스템에서 RBD 제거	RBD 항목 상태 변수를 **absent** 로 변경합니다. 먼저 RBD 이미지를 운영 체제에서 할당 해제해야 합니다.
클라이언트 CHAP 자격 증명 변경	**client_connections** 에서 관련 CHAP 세부 정보를 업데이트합니다. 이는 고객과 조정해야 합니다. 예를 들어 클라이언트는 iSCSI 로그아웃을 발행하고 자격 증명이 Ansible 플레이북에 의해 변경되고 자격 증명이 클라이언트에서 변경된 다음 클라이언트는 iSCSI 로그인을 수행합니다.
클라이언트 제거	관련 **client_connections** 항목을 **absent** 상태로 업데이트합니다. Ansible 플레이북이 실행되면 클라이언트가 시스템에서 제거되지만 디스크는 잠재적인 재사용을 위해 Linux iSCSI 대상에 남아 있습니다.

변경 사항이 완료되면 **Ansible** 플레이북을 다시 실행하여 **iSCSI** 게이트웨이 노드에 변경 사항을 적용합니다.

```
# ansible-playbook site.yml
```

설정 제거:

1. **iSCSI** 게이트웨이 구성을 제거하기 전에 모든 **iSCSI** 이니시에이터의 연결을 해제합니다. 적절한 운영 체제를 위해 아래 절차를 따르십시오.

   a. **Red Hat Enterprise Linux initiators:**

   구문

   ```
   iscsiadm -m node -T $TARGET_NAME --logout
   ```

   $TARGET_NAME 을 구성된 **iSCSI** 대상 이름으로 바꿉니다.

   예제

   ```
   # iscsiadm -m node -T iqn.2003-01.com.redhat.iscsi-gw:ceph-igw --logout
   Logging out of session [sid: 1, target: iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw, portal:
   10.172.19.21,3260]
   Logging out of session [sid: 2, target: iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw, portal:
   10.172.19.22,3260]
   Logout of [sid: 1, target: iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw, portal:
   10.172.19.21,3260] successful.
   Logout of [sid: 2, target: iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw, portal:
   10.172.19.22,3260] successful.
   ```

   b. **Windows** 이니시에이터:

자세한 내용은 Microsoft 설명서 를 참조하십시오.

c. **VMware ESXi 이니시에이터:**

자세한 내용은 VMware 설명서 를 참조하십시오.

2. **Ansible 관리 노드에서 Ansbile 사용자로 /usr/share/ceph-ansible/ 디렉터리로 변경합니다.**

```
[user@admin ~]$ cd /usr/share/ceph-ansible/
```

3. **Ansible 플레이북을 실행하여 iSCSI 게이트웨이 구성을 제거합니다.**

```
[user@admin ceph-ansible]$ ansible-playbook purge-cluster.yml --limit iscsigws
```

4. **Ceph Monitor 또는 Client 노드에서 root 사용자로 iSCSI 게이트웨이 구성 오브젝트 (gateway.conf)를 제거합니다.**

```
[root@mon ~]# rados rm -p pool gateway.conf
```

5. **선택 사항:**

내보낸 Ceph RADOS Block Device(RBD)가 더 이상 필요하지 않은 경우 RBD 이미지를 제거합니다. Ceph Monitor 또는 클라이언트 노드에서 root 사용자로 다음 명령을 실행합니다.

구문

```
rbd rm $IMAGE_NAME
```

$IMAGE_NAME 을 RBD 이미지 이름으로 바꿉니다.

예제

```
[root@mon ~]# rbd rm rbd01
```

8.3.2. 명령줄 인터페이스를 사용하여 iSCSI 대상 구성

Ceph iSCSI 게이트웨이는 **iSCSI** 대상 노드이며 **Ceph** 클라이언트 노드이기도 합니다. **Ceph iSCSI** 게이트웨이는 독립 실행형 노드이거나 **Ceph OSD(Object Store Disk)** 노드에 공동 배치될 수 있습니다. 다음 단계를 완료하면 기본 작업을 위해 **Ceph iSCSI** 게이트웨이를 설치하고 구성합니다.

요구 사항:

- **Red Hat Enterprise Linux 7.5** 이상
- 실행 중인 **Red Hat Ceph Storage 3.3** 클러스터 이상
- 다음 패키지를 설치해야 합니다.
  - **targetcli-2.1.fb47-0.1.20170815.git5bf3517.el7cp** 또는 최신 패키지
  - **python-rtslib-2.1.fb64-0.1.20170815.gitec364f3.el7cp** 또는 최신 패키지
  - **tcmu-runner-1.4.0-0.2.el7cp** 또는 최신 패키지
  - **OpenSSL-1.0.2k-8.el7** 또는 최신 패키지

중요

이러한 패키지의 이전 버전이 있는 경우 최신 버전을 설치하기 전에 먼저 패키지를 제거해야 합니다. 이러한 최신 버전은 **Red Hat Ceph Storage** 리포지토리에서 설치해야 합니다.

gwcli 유틸리티를 사용하기 전에 스토리지 클러스터의 모든 **Ceph** 모니터 노드에서 다음 단계를 수행합니다.

1. **root** 사용자로 **ceph-mon** 서비스를 다시 시작합니다.

```
# systemctl restart ceph-mon@$MONITOR_HOST_NAME
```

예를 들면 다음과 같습니다.

```
# systemctl restart ceph-mon@monitor1
```

Installing 섹션을 진행하기 전에 **Ceph iSCSI** 게이트웨이 노드에서 **root** 사용자로 다음 단계를 수행합니다.

1. **Ceph iSCSI** 게이트웨이가 **OSD** 노드에 배치되지 않은 경우 스토리지 클러스터의 실행 중인 **Ceph** 노드에서 **iSCSI** 게이트웨이 노드로 **Ceph** 구성 파일을 복사합니다. **Ceph** 구성 파일은 **/etc/ceph/** 의 **iSCSI** 게이트웨이 노드에 있어야 합니다.

2. **Ceph** 명령줄 인터페이스를 설치하고 구성하십시오. 자세한 내용은 **Red Hat** ***Enterprise Linux*****용** ***Red Hat Ceph Storage 3 설치 가이드의 Ceph 명령줄 인터페이스 설치*** 장을 참조하십시오.

3. **Ceph** 툴 리포지토리를 활성화합니다.

```
# subscription-manager repos --enable=rhel-7-server-rhceph-3-tools-rpms
```

4. 필요한 경우 방화벽에서 **TCP** 포트 **3260** 및 **5000**을 엽니다.

5. 새로 만들기 또는 기존 **RADOS** 블록 장치(**RBD**)를 사용합니다.

   a. 자세한 내용은 **2.1**절. “사전 요구 사항”를 참조하십시오.

주의

Ansible을 사용하여 **Ceph iSCSI** 게이트웨이를 이미 설치한 경우 이 절차를 사용하지 마십시오.

Ansible은 **ceph-iscsi-cli** 패키지를 설치하고, **ansible-playbook** 명령이 실행될 때 **group_vars/iscsigws.yml** 파일의 설정에 따라 **/etc/ceph/iscsi-gateway.cfg** 파일을 생성한 다음 업데이트합니다. 자세한 내용은 요구 사항:를 참조하십시오.

설치:

별도로 명시하지 않는 한 모든 iSCSI 게이트웨이 노드에서 root 사용자로 다음 단계를 수행합니다.

1. **ceph-iscsi-cli** 패키지를 설치합니다.

```
# yum install ceph-iscsi-cli
```

2. **tcmu-runner** 패키지를 설치합니다.

```
# yum install tcmu-runner
```

3. 필요한 경우 **openssl** 패키지를 설치합니다.

```
# yum install openssl
```

a. 기본 **iSCSI** 게이트웨이 노드에서 **SSL** 키를 저장할 디렉터리를 만듭니다.

```
# mkdir ~/ssl-keys
# cd ~/ssl-keys
```

b. 기본 **iSCSI** 게이트웨이 노드에서 인증서 및 키 파일을 생성합니다.

```
# openssl req -newkey rsa:2048 -nodes -keyout iscsi-gateway.key -x509 -days 365 -out
iscsi-gateway.crt
```

참고

환경 정보를 입력하라는 메시지가 표시됩니다.

c. 기본 iSCSI 게이트웨이 노드에서 PEM 파일을 생성합니다.

```
# cat iscsi-gateway.crt iscsi-gateway.key > iscsi-gateway.pem
```

d. 기본 iSCSI 게이트웨이 노드에서 공개 키를 생성합니다.

```
# openssl x509 -inform pem -in iscsi-gateway.pem -pubkey -noout > iscsi-gateway-pub.key
```

e. 기본 iSCSI 게이트웨이 노드에서 iscsi-gateway.crt,iscsi-gateway.pem,iscsi-gateway-pub.key, iscsi-gateway.key 파일을 다른 iSCSI 게이트웨이 노드의 /etc/ceph/ 디렉터리에 복사합니다.

4. /etc/ceph/ 디렉터리에 iscsi-gateway.cfg 라는 파일을 생성합니다.

```
# touch /etc/ceph/iscsi-gateway.cfg
```

a. iscsi-gateway.cfg 파일을 편집하고 다음 행을 추가합니다.

구문

```
[config]
cluster_name = <ceph_cluster_name>
gateway_keyring = <ceph_client_keyring>
api_secure = true
trusted_ip_list = <ip_addr>,<ip_addr>
```

예제

```
[config]
cluster_name = ceph
gateway_keyring = ceph.client.admin.keyring
api_secure = true
trusted_ip_list = 192.168.0.10,192.168.0.11
```

이러한 옵션에 대한 자세한 내용은 요구 사항: 의 표 **8.1** 및 **8.2**를 참조하십시오.

중요

모든 **iSCSI** 게이트웨이 노드에서 **iscsi-gateway.cfg** 파일이 동일해야 합니다.

b. **iscsi-gateway.cfg** 파일을 모든 **iSCSI** 게이트웨이 노드에 복사합니다.

5. **API** 서비스를 활성화하고 시작합니다.

```
# systemctl enable rbd-target-api
# systemctl start rbd-target-api
```

구성:

1. **iSCSI** 게이트웨이 명령줄 인터페이스를 시작합니다.

```
# gwcli
```

2. **IPv4** 또는 **IPv6** 주소를 사용하여 **iSCSI** 게이트웨이 생성:

구문

```
>/iscsi-target create iqn.2003-01.com.redhat.iscsi-gw:<target_name>
> goto gateways
> create <iscsi_gw_name> <IP_addr_of_gw>
> create <iscsi_gw_name> <IP_addr_of_gw>
```

예제

```
>/iscsi-target create iqn.2003-01.com.redhat.iscsi-gw:ceph-igw
> goto gateways
> create ceph-gw-1 10.172.19.21
> create ceph-gw-2 10.172.19.22
```

중요

IPv4 및 **IPv6** 주소를 혼합하여 사용할 수 없습니다.

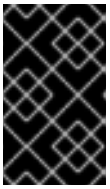
3. **RADOS** 블록 장치**(RBD)** 추가**:**

구문

```
> cd /disks
>/disks/ create <pool_name> image=<image_name> size=<image_size>m|g|t
max_data_area_mb=<buffer_size>
```

예제

```
> cd /disks
>/disks/ create rbd image=disk_1 size=50g max_data_area_mb=32
```



중요

풀 이름 또는 이미지 이름에 마침표(.)가 있을 수 없습니다.

주의

max_data_area_mb 옵션은 각 이미지가 **iSCSI** 대상과 **Ceph** 클러스터 간에 **SCSI** 명령 데이터를 전달하는 데 사용할 수 있는 메모리 크기(**MB**)를 제어합니다. 이 값이 너무 작으면 과도한 큐 전체 재시도를 발생시켜 성능에 영향을 미칠 수 있습니다. 값이 너무 크면 한 디스크에 시스템 메모리가 너무 많이 사용되어 다른 하위 시스템에 할당 오류가 발생할 수 있습니다. 기본값은 **8**입니다.

이 값은 **gwcli reconfigure** 하위 명령을 사용하여 변경할 수 있습니다. 이 명령을 적용하려면 **iSCSI** 이니시에이터에서 이미지를 사용해서는 안 됩니다. 이 문서에 지정하지 않거나 **Red Hat** 지원에서 지시하지 않은 경우 **gwcli reconfigure** 하위 명령을 사용하여 다른 옵션을 조정하지 마십시오.

구문

```
>/disks/ reconfigure max_data_area_mb <new_buffer_size>
```

예제

```
>/disks/ reconfigure max_data_area_mb 64
```

4. 클라이언트를 생성합니다.

구문

```
> goto hosts
> create iqn.1994-05.com.redhat:<client_name>
> auth chap=<user_name>/<password>
```

예제

```
> goto hosts
> create iqn.1994-05.com.redhat:rh7-client
> auth chap=iscsiuser1/temp12345678
```

중요

CHAP 비활성화는 Red Hat Ceph Storage 3.1 이상에서만 지원됩니다. Red Hat은 CHAP가 활성화되어 있고 일부 CHAP가 비활성화된 클라이언트 혼합을 지원하지 않습니다. 모든 클라이언트에는 CHAP이 활성화되어 있거나 CHAP이 비활성화되어 있어야 합니다. 기본 동작은 이니시에이터 이름으로만 이니시에이터를 인증하는 것입니다.

이니시에이터가 대상에 로그인하지 못하면 CHAP 인증이 일부 이니시에이터에 대해 잘못 구성되었을 수 있습니다.

예제

```
o- hosts ..................................... [Hosts: 2: Auth: MISCONFIG]
```

호스트 수준에서 다음 명령을 수행하여 CHAP 인증을 모두 재설정합니다.

```
/> goto hosts
/iscsi-target...csi-igw/hosts> auth nochap
ok
ok
/iscsi-target...csi-igw/hosts> ls
o- hosts ..................................... [Hosts: 2: Auth: None]
  o- iqn.2005-03.com.ceph:esx ..... [Auth: None, Disks: 4(310G)]
  o- iqn.1994-05.com.redhat:rh7-client .. [Auth: None, Disks: 0(0.00Y)]
```

5. **클라이언트에 디스크 추가:**

구문

```
>/iscsi-target..eph-igw/hosts> cd iqn.1994-05.com.redhat:<client_name>
> disk add <pool_name>.<image_name>
```

예제

```
>/iscsi-target..eph-igw/hosts> cd iqn.1994-05.com.redhat:rh7-client
> disk add rbd.disk_1
```

6. **API가 SSL을 올바르게 사용하고 있는지 확인하려면 /var/log/rbd-target-api.log 파일을 참조하십시오. 예를 들면 다음과 같습니다.**

```
Aug 01 17:27:42 test-node.example.com python[1879]:  * Running on https://0.0.0.0:5000/
```

7. **다음 단계는 iSCSI 이니시에이터를 구성하는 것입니다. iSCSI 이니시에이터 구성에 대한 자세한 내용은 8.4절. "iSCSI Initiator 구성" 을 참조하십시오.**

검증

1. **iSCSI 게이트웨이가 작동하는지 확인하려면 다음을 수행합니다.**

예제

```
/> goto gateways
/iscsi-target...-igw/gateways> ls
o- gateways ............................ [Up: 2/2, Portals: 2]
  o- ceph-gw-1  ........................ [ 10.172.19.21 (UP)]
  o- ceph-gw-2  ........................ [ 10.172.19.22 (UP)]
```

참고

상태가 UNKNOWN 이면 네트워크 문제 및 잘못된 설정을 확인합니다. 방화벽을 사용하는 경우 적절한 TCP 포트가 열려 있는지 확인합니다. iSCSI 게이트웨이가 trusted_ip_list 옵션에 나열되어 있는지 확인합니다. rbd-target-api 서비스가 iSCSI 게이트웨이 노드에서 실행 중인지 확인합니다.

2. 이니시에이터가 **iSCSI** 대상에 연결되어 있는지 확인하려면 이니시에이터 **LOGGED-IN** 이 표시됩니다.

예제

```
/> goto hosts
/iscsi-target...csi-igw/hosts> ls
o- hosts ..... [Hosts: 1: Auth: None]
  o- iqn.1994-05.com.redhat:rh7-client  [LOGGED-IN, Auth: None, Disks: 0(0.00Y)]
```

3. **LUN**이 **iSCSI** 게이트웨이 간에 분산되는지 확인하려면 다음을 수행하십시오.

```
/> goto hosts
/iscsi-target...csi-igw/hosts> ls
o- hosts ..... [Hosts: 2: Auth: None]
  o- iqn.2005-03.com.ceph:esx ..... [Auth: None, Disks: 4(310G)]
  | o- lun 0 ..... [rbd.disk_1(100G), Owner: ceph-gw-1]
  | o- lun 1 ..... [rbd.disk_2(10G), Owner: ceph-gw-2]
```

디스크를 생성할 때 이니시에이터의 다중 경로 계층에 따라 디스크에 **iSCSI** 게이트웨이가 소유자로 할당됩니다. 이니시에이터의 다중 경로 계층은 **ALUA AOptimized(AO)** 상태로 보고됩니다. 다른 경로는 **ALUA Active-non-Optimized (ANO)** 상태에 있는 것으로 보고됩니다.

AO 경로가 실패하면 다른 **iSCSI** 게이트웨이 중 하나가 사용됩니다. 장애 조치 게이트웨이의 순서는 일반적으로 먼저 검색된 경로를 기반으로 하는 이니시에이터의 다중 경로 계층에 따라 달라집니다.

현재 **LUN**의 밸런싱이 동적이지 않습니다. **iSCSI** 게이트웨이는 디스크 생성 시 선택되며 변경할 수 없습니다.

8.3.3. iSCSI 대상의 성능 최적화

iSCSI 대상에서 네트워크를 통해 데이터를 전송하는 방법을 제어하는 많은 설정이 있습니다. 이러한 설정을 사용하여 **iSCSI** 게이트웨이의 성능을 최적화할 수 있습니다.

주의

Red Hat 지원에서 지시하거나 이 문서에 명시된 대로만 해당 설정을 변경합니다.

gwcli reconfigure 하위 명령

gwcli reconfigure 하위 명령은 **iSCSI** 게이트웨이의 성능을 최적화하는 데 사용되는 설정을 제어합니다.

iSCSI 대상의 성능에 영향을 미치는 설정

- **max_data_area_mb**
- **cmdsn_depth**
- **immediate_data**
- **initial_r2t**
- **max_outstanding_r2t**
- **first_burst_length**
- **max_burst_length**
- **max_recv_data_segment_length**
- **max_xmit_data_segment_length**

추가 리소스

- gwcli reconfigure 를 사용하여 조정하는 방법을 보여주는 예제를 포함하여 max_data_area_mb 에 대한 정보는 블록 장치 가이드의 명령줄 인터페이스를 사용하여 iSCSI 대상 구성 섹션에 있으며 컨테이너 가이드 용 컨테이너에서 Ceph iSCSI 게이트웨이 구성 섹션에 있습니다.

8.3.4. 더 많은 iSCSI 게이트웨이 추가

스토리지 관리자는 Ansible 또는 gwcli 명령줄 도구를 사용하여 초기 두 개의 iSCSI 게이트웨이를 4개의 iSCSI 게이트웨이로 확장할 수 있습니다. 더 많은 iSCSI 게이트웨이를 추가하면 부하 분산 및 페일오버 옵션을 사용할 때 중복성을 더 많이 제공할 때 유연성이 향상됩니다.

8.3.4.1. 사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 3 클러스터.
- iSCSI 게이트웨이 소프트웨어 설치
- 노드 또는 기존 OSD 노드 예비

8.3.4.2. Ansible을 사용하여 iSCSI 게이트웨이 추가

Ansible 자동화 유틸리티를 사용하여 더 많은 iSCSI 게이트웨이를 추가할 수 있습니다. 이 절차에서는 두 개의 iSCSI 게이트웨이의 기본 설치를 4개의 iSCSI 게이트웨이로 확장합니다. 독립 실행형 노드에서 iSCSI 게이트웨이를 구성하거나 기존 OSD 노드와 함께 배치할 수 있습니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 3 클러스터.
- iSCSI 게이트웨이 소프트웨어 설치
- Ansible 관리 노드에서 root 사용자가 액세스할 수 있도록 합니다.
- 새 노드에서 root 사용자가 액세스할 수 있도록 합니다.

절차

1. *새로운 **iSCSI** 게이트웨이 노드에서 **Red Hat Ceph Storage 3 Tools** 리포지토리를 활성화합니다.*

```
[root@iscsigw ~]# subscription-manager repos --enable=rhel-7-server-rhceph-3-tools-els-rpms
```

*자세한 내용은 설치 가이드 의 **Red Hat Ceph Storage** 리포지토리 활성화 섹션 을 참조하십시오.*

2. ***ceph-iscsi-config** 패키지를 설치합니다.*

```
[root@iscsigw ~]# yum install ceph-iscsi-config
```

3. ***gateway** 그룹의 **/etc/ansible/hosts** 파일의 목록에 추가합니다.*

예제

```
[iscsigws]
...
ceph-igw-3
ceph-igw-4
```

참고

***OSD** 노드와 **iSCSI** 게이트웨이를 공동 배치하는 경우 **OSD** 노드를 **[iscsigws]** 섹션에 추가합니다.*

4. *편집을 위해 **/usr/share/ceph-ansible/group_vars/iscsigws.yml** 파일을 열고 **IPv4** 주소가 있는 두 개의 **iSCSI** 게이트웨이를 **gateway_ip_list** 옵션에 추가합니다.*

예제

```
gateway_ip_list: 10.172.19.21,10.172.19.22,10.172.19.23,10.172.19.24
```

중요

gateway_ip_list 옵션의 **IP** 주소를 제공해야 합니다. **IPv4** 및 **IPv6** 주소를 혼합하여 사용할 수 없습니다.

5. **Ansible** 관리 노드에서 **root** 사용자로 **Ansible** 플레이북을 실행합니다.

```
# cd /usr/share/ceph-ansible
# ansible-playbook site.yml
```

6. **iSCSI** 이니시에이터에서 새로 추가된 **iSCSI** 게이트웨이를 사용하도록 다시 로그인합니다.

추가 리소스

- **iSCSI Initiator** 사용에 대한 자세한 내용은 **iSCSI Initiator** 구성을 참조하십시오.

8.3.4.3. gwcli 를 사용하여 iSCSI 게이트웨이 추가

gwcli 명령줄 툴을 사용하여 **iSCSI** 게이트웨이를 추가할 수 있습니다. 이 절차에서는 두 개의 **iSCSI** 게이트웨이의 기본값을 **4**개의 **iSCSI** 게이트웨이로 확장합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage 3** 클러스터.
- **iSCSI** 게이트웨이 소프트웨어 설치
- **root** 사용자가 새 노드 또는 **OSD** 노드에 액세스하도록 합니다.

절차

1.

Ceph iSCSI 게이트웨이가 OSD 노드에 배치되지 않은 경우 스토리지 클러스터의 실행 중인 Ceph 노드에서 새 iSCSI 게이트웨이 노드로 Ceph 구성 파일을 복사합니다. Ceph 구성 파일은 /etc/ceph/ 디렉터리의 iSCSI 게이트웨이 노드에 있어야 합니다.

2. **Ceph 명령줄 인터페이스를 설치하고 구성합니다. 자세한 내용은 Red Hat Enterprise Linux용 Red Hat Ceph Storage 3 설치 가이드의 Ceph 명령줄 인터페이스 설치 장을 참조하십시오.**

3. **새로운 iSCSI 게이트웨이 노드에서 Red Hat Ceph Storage 3 Tools 리포지토리를 활성화합니다.**

```
[root@iscsigw ~]# subscription-manager repos --enable=rhel-7-server-rhceph-3-tools-els-rpms
```

자세한 내용은 설치 가이드 의 Red Hat Ceph Storage 리포지토리 활성화 섹션 을 참조하십시오.

4. **ceph-iscsi-cli 및 tcmu-runner 패키지를 설치합니다.**

```
[root@iscsigw ~]# yum install ceph-iscsi-cli tcmu-runner
```

a. **필요한 경우 openssl 패키지를 설치합니다.**

```
[root@iscsigw ~]# yum install openssl
```

5. **기존 iSCSI 게이트웨이 노드 중 하나에서 /etc/ceph/iscsi-gateway.cfg 파일을 편집하고 새 iSCSI 게이트웨이 노드의 새 IP 주소로 trusted_ip_list 옵션을 추가합니다.**

예제

```
[config]
...
trusted_ip_list = 10.172.19.21,10.172.19.22,10.172.19.23,10.172.19.24
```

이러한 옵션에 대한 자세한 내용은 Ansible 표를 사용하여 iSCSI 대상 구성을 참조하십시오.

6. *업데이트된 /etc/ceph/iscsi-gateway.cfg 파일을 모든 iSCSI 게이트웨이 노드에 복사합니다.*

중요

모든 iSCSI 게이트웨이 노드에서 iscsi-gateway.cfg 파일이 동일해야 합니다.

7. *필요한 경우 SSL을 사용하는 경우 기존 iSCSI 게이트웨이 노드 중 하나에서 ~/ssl-keys/iscsi-gateway. pem,~/ssl-keys/iscsi-gateway-pub.key 및 ~/ssl-keys/iscsi-gateway.key 파일을 새 iSCSI 게이트웨이 노드 중 하나에 있는 ~/ssl-keys / iscsi-gateway.key 파일을 새 iSCSI 게이트웨이 노드 중 하나에 복사합니다.*

8. *새 iSCSI 게이트웨이 노드에서 API 서비스를 활성화하고 시작합니다.*

```
[root@iscsigw ~]# systemctl enable rbd-target-api
[root@iscsigw ~]# systemctl start rbd-target-api
```

9. *iSCSI 게이트웨이 명령줄 인터페이스를 시작합니다.*

```
[root@iscsigw ~]# gwcli
```

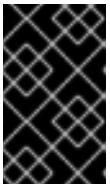
10. *IPv4 또는 IPv6 주소를 사용하여 iSCSI 게이트웨이 생성:*

구문

```
>/iscsi-target create iqn.2003-01.com.redhat.iscsi-gw:_TARGET_NAME_
> goto gateways
> create ISCSI_GW_NAME IP_ADDR_OF_GW
> create ISCSI_GW_NAME IP_ADDR_OF_GW
```

예제

```
>/iscsi-target create iqn.2003-01.com.redhat.iscsi-gw:ceph-igw
> goto gateways
> create ceph-gw-3 10.172.19.23
> create ceph-gw-4 10.172.19.24
```



중요

***IPv4** 및 **IPv6** 주소를 혼합하여 사용할 수 없습니다.*

11. ***iSCSI** 이니시에이터에서 새로 추가된 **iSCSI** 게이트웨이를 사용하도록 다시 로그인합니다.*

추가 리소스

- ***iSCSI Initiator** 사용에 대한 자세한 내용은 **iSCSI Initiator** 구성을 참조하십시오.*

8.4. ISCSI INITIATOR 구성

***Red Hat Ceph Storage**는 **Ceph iSCSI** 게이트웨이에 연결하기 위해 세 가지 운영 체제에서 **iSCSI** 이니시에이터를 지원합니다.*

- ***Red Hat Enterprise Linux***
- ***Red Hat Virtualization***
- ***Microsoft Windows Server 2016***
- ***VMware ESX vSphere Web Client***

8.4.1. Red Hat Enterprise Linux용 iSCSI Initiator

사전 요구 사항:

- ***iscsi-initiator-utils-6.2.0.873-35** 패키지를 설치해야 합니다.*
- ***device-mapper-multipath-0.4.9-99** 패키지 이상이 설치되어 있어야 합니다.*

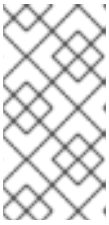
소프트웨어 설치:

1. ***iSCSI** 이니시에이터 및 다중 경로 툴을 설치합니다.*

```
# yum install iscsi-initiator-utils
# yum install device-mapper-multipath
```

***Initiator** 이름 설정*

1. ***/etc/iscsi/initiatorname.iscsi** 파일을 편집합니다.*



참고

*이니시에이터 이름은 **Ansible client_connections** 옵션에 사용된 이니시에이터 이름 또는 **gwcli** 를 사용하여 초기 설정 중에 사용된 항목과 일치해야 합니다.*

*다중 경로 **IO** 구성:*

1. *기본 **/etc/multipath.conf** 파일을 생성하고 **multipathd** 서비스를 활성화합니다.*

```
# mpathconf --enable --with_multipathd y
```

2. ***/etc/multipath.conf** 파일에 다음을 추가합니다.*

```
devices {
        device {
                vendor                "LIO-ORG"
```

```
        hardware_handler        "1 alua"
        path_grouping_policy    "failover"
        path_selector           "queue-length 0"
        failback                60
        path_checker            tur
        prio                    alua
        prio_args               exclusive_pref_bit
        fast_io_fail_tmo        25
        no_path_retry           queue
    }
}
```

3. ***multipathd** 서비스를 다시 시작합니다.*

```
# systemctl reload multipathd
```

CHAP** 설정 및 **iSCSI Discovery/Login:

1. *그에 따라 **/etc/iscsi/iscsid.conf** 파일을 업데이트하여 **CHAP** 사용자 이름과 암호를 입력합니다.*

   예제

```
node.session.auth.authmethod = CHAP
node.session.auth.username = user
node.session.auth.password = password
```

참고

*이러한 옵션을 업데이트하는 경우 **iscsiadm discovery** 명령을 재실행해야 합니다.*

2. *대상 포털을 검색합니다.*

```
# iscsiadm -m discovery -t st -p 192.168.56.101
192.168.56.101:3260,1 iqn.2003-01.org.linux-iscsi.rheln1
192.168.56.102:3260,2 iqn.2003-01.org.linux-iscsi.rheln1
```

3. *대상 로그인:*

```
# iscsiadm -m node -T iqn.2003-01.org.linux-iscsi.rheln1 -l
```

다중 경로 IO 구성 보기:

다중 경로 데몬(multipathd)은 multipath.conf 설정을 기반으로 장치를 자동으로 설정합니다. multipath 명령을 실행하면 각 경로에 대한 우선 순위 그룹이 있는 장애 조치 구성에서 장치 설정이 표시됩니다. 예를 들면 다음과 같습니다.

```
# multipath -ll
mpathbt (360014059ca317516a69465c883a29603) dm-1 LIO-ORG ,IBLOCK
size=1.0G features='0' hwhandler='1 alua' wp=rw
|-+- policy='queue-length 0' prio=50 status=active
| `- 28:0:0:1 sde  8:64  active ready running
`-+- policy='queue-length 0' prio=10 status=enabled
  `- 29:0:0:1 sdc  8:32  active ready running
```

multipath -ll output prio 값은 ALUA 상태를 나타냅니다. 여기서 prio=50 은 ALUA Active-Optimized 상태에서 iSCSI 게이트웨이의 경로입니다. prio=10 은 Active-non-Optmized 경로임을 나타냅니다. status 필드는 사용 중인 경로를 나타냅니다. 여기서 active 는 현재 사용된 경로를 나타내며, enabled 는 활성 상태가 실패하는 경우 장애 조치 경로를 나타냅니다. iSCSI 게이트웨이에서 장치 이름(예: multipath -ll 출력에서 sde)을 일치시키려면 iSCSI 게이트웨이에서 다음 명령을 실행합니다.

```
# iscsiadm -m session -P 3
```

영구 포털 값은 gwcli 에 나열된 iSCSI 게이트웨이 또는 Ansible이 사용된 경우 gateway_ip_list 에 나열된 iSCSI 게이트웨이 중 하나의 IP 주소입니다.

8.4.2. Red Hat Virtualization용 iSCSI Initiator

사전 요구 사항:

- *Red Hat Virtualization 4.1*
- *모든 Red Hat Virtualization 노드에 구성된 MPIO 장치*

- ***iscsi-initiator-utils-6.2.0.873-35*** *패키지를 설치해야 합니다.*

- ***device-mapper-multipath-0.4.9-99*** *패키지 이상이 설치되어 있어야 합니다.*

다중 경로 ***IO*** *구성:*

1. ***/etc/multipath/conf.d/DEVICE_NAME.conf*** *파일을 다음과 같이 업데이트합니다.*

```
devices {
        device {
                vendor                "LIO-ORG"
                hardware_handler      "1 alua"
                path_grouping_policy  "failover"
                path_selector         "queue-length 0"
                failback              60
                path_checker          tur
                prio                  alua
                prio_args             exclusive_pref_bit
                fast_io_fail_tmo      25
                no_path_retry         queue
        }
}
```

2. ***multipathd*** *서비스를 다시 시작합니다.*

```
# systemctl reload multipathd
```

iSCSI *스토리지 추가*

1. *스토리지 리소스 탭을 클릭하여 기존 스토리지 도메인을 나열합니다.*

2. ***New Domain*** *버튼을 클릭하여* ***New Domain*** *창을 엽니다.*

3. *새 스토리지 도메인의 이름을 입력합니다.*

4. *데이터 센터 드롭다운 메뉴를 사용하여 데이터 센터를 선택합니다.*

5.

*드롭다운 메뉴를 사용하여 **Domain Function** (도메인 기능) 및 **Storage Type** (스토리지 유형)을 선택합니다. 선택한 도메인 기능과 호환되지 않는 스토리지 도메인 유형은 사용할 수 없습니다.*

6.

***Use Host** (호스트 사용) 필드에서 활성 호스트를 선택합니다. 데이터 센터의 첫 번째 데이터 도메인이 아닌 경우 데이터 센터의 **10.0.0.1** 호스트를 선택해야 합니다.*

7.

***iSCSI**가 스토리지 유형으로 선택되면 **New Domain** 창은 사용되지 않는 **LUN**을 사용하여 알려진 대상을 자동으로 표시합니다. 스토리지를 추가하는 대상이 목록에 없는 경우 대상 검색을 사용하여 해당 항목을 찾을 수 있습니다. 그렇지 않으면 다음 단계를 진행합니다.*

a.

*대상 검색 옵션을 활성화하려면 대상 검색을 클릭합니다. 대상을 검색하고 로그인하면 **New Domain** (새 도메인) 창에서 환경에서 사용하지 않는 **LUN**을 사용하여 대상이 자동으로 표시됩니다.*

참고

*환경 외부의 **LUN**도 표시됩니다.*

***Discover Targets** 옵션을 사용하여 여러 대상에 **LUN**을 추가하거나 동일한 **LUN**에 여러 경로를 추가할 수 있습니다.*

b.

***Address** 필드에 **iSCSI** 호스트의 정규화된 도메인 이름 또는 **IP** 주소를 입력합니다.*

c.

***Port** (포트) 필드에서 대상을 검색할 때 호스트에 연결할 포트를 입력합니다. 기본값은 **3260** 입니다.*

d.

***CHAP(Challen Handshake Authentication Protocol)**를 사용하여 스토리지를 보호하는 경우 **User Authentication** (사용자 인증) 확인란을 선택합니다. **CHAP** 사용자 이름과 **CHAP** 암호를 입력합니다.*

e.

***Discover** (검색) 버튼을 클릭합니다.*

f.

검색 결과에서 사용할 대상을 선택하고 로그인 버튼을 클릭합니다. 또는 모두 로그인을 클릭하여 검색된 모든 대상에 로그인합니다.

중요

둘 이상의 경로 액세스 권한이 필요한 경우 필요한 모든 경로를 통해 대상을 검색하고 로그인해야 합니다. 현재 추가 경로를 추가하도록 스토리지 도메인을 수정하는 것은 현재 지원되지 않습니다.

8. *원하는 대상 옆에 있는 **+** 버튼을 클릭합니다. 이 명령은 항목을 확장하고 대상에 연결된 사용되지 않는 모든 **LUN**을 표시합니다.*

9. *스토리지 도메인을 생성하기 위해 사용 중인 각 **LUN**의 확인란을 선택합니다.*

10. *선택적으로 고급 매개변수를 구성할 수 있습니다.*

    a. ***Advanced Parameters** 를 클릭합니다.*

    b. ***Warning Low Space Indicator** 필드에 백분율 값을 입력합니다. 스토리지 도메인에서 사용 가능한 여유 공간이 이 백분율 미만인 경우 사용자에게 경고 메시지가 표시되고 기록됩니다.*

    c. ***Critical Space Action Blocker** 필드에 **GB** 값을 입력합니다. 스토리지 도메인에서 사용 가능한 여유 공간이 이 값보다 작으면 오류 메시지가 사용자에게 표시되고 기록되고 공간을 소비하는 새 작업이 일시적으로 차단됩니다.*

    d. *삭제 후 삭제 옵션을 활성화하려면 **Wipe After Delete** 확인란을 선택합니다. 이 옵션은 도메인을 생성한 후 편집할 수 있지만 이미 존재하는 디스크의 삭제 후 지웁니다.*

    e. *삭제 후 삭제 해제 옵션을 활성화하려면 **Discard After Delete** 확인란을 선택합니다. 이 옵션은 도메인을 생성한 후 편집할 수 있습니다. 이 옵션은 블록 스토리지 도메인에서만 사용할 수 있습니다.*

11. *확인을 클릭하여 스토리지 도메인을 생성하고 창을 닫습니다.*

8.4.3. Microsoft Windows용 iSCSI Initiator

사전 요구 사항:

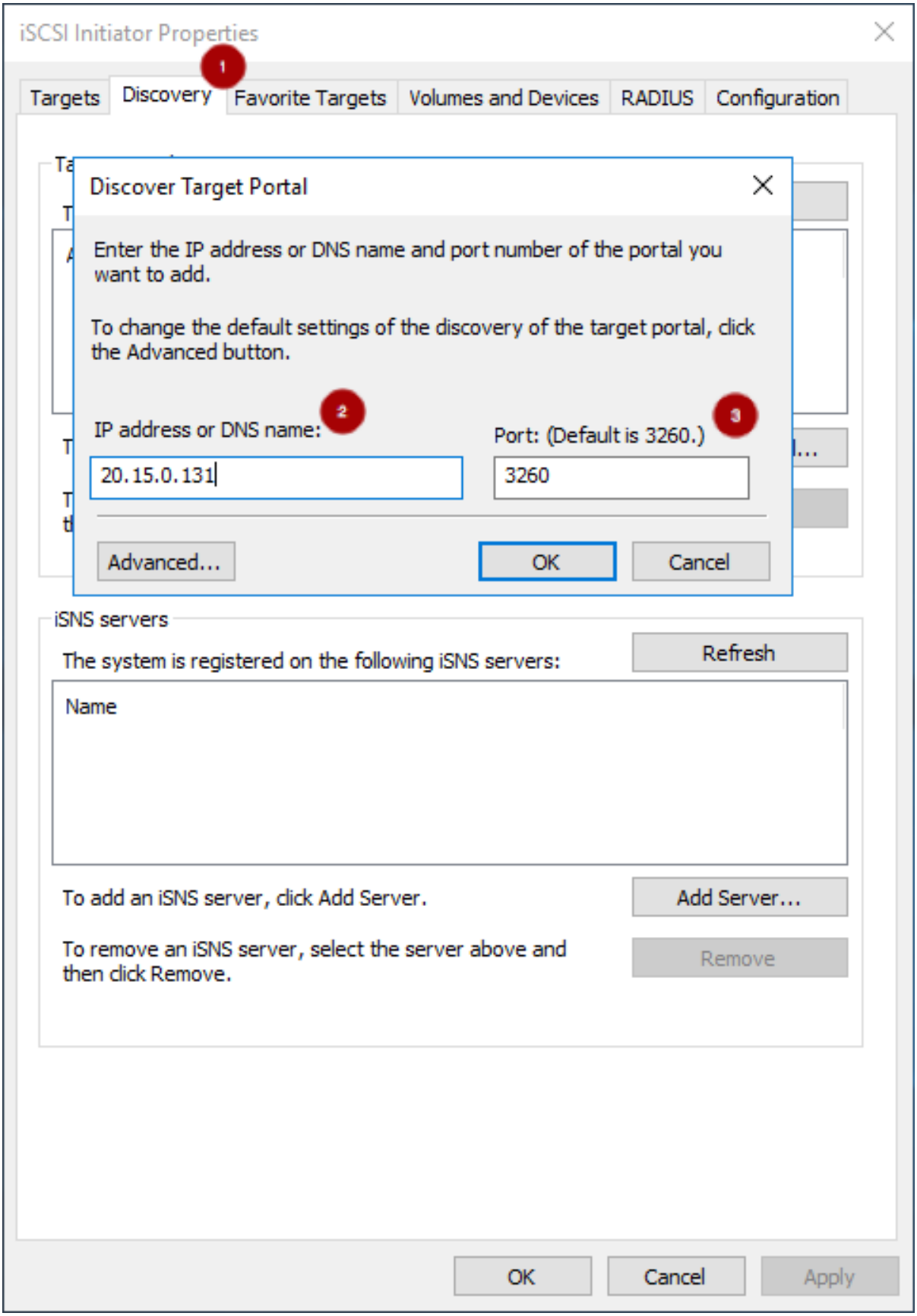
- ***Microsoft Windows Server 2016***

***iSCSI** 이니시에이터, 검색 및 설정:*

1. ***iSCSI** 이니시에이터 드라이버 및 **MPIO** 툴을 설치합니다.*

2. ***MPIO** 프로그램을 시작하고 **Discover Multi-Paths** 탭을 클릭하고 **iSCSI** 장치에 대한 지원 추가 확인란을 선택한 다음 추가 를 클릭합니다. 이 변경 사항은 재부팅이 필요합니다.*

3. ***iSCSI** 이니시에이터 속성 창의 검색 탭*
   1
   *에서 대상 포털을 추가합니다. **Ceph iSCSI** 게이트웨이의 **IP** 주소 또는 **DNS** 이름*
   2
   및 포트
   3
   를 입력합니다.

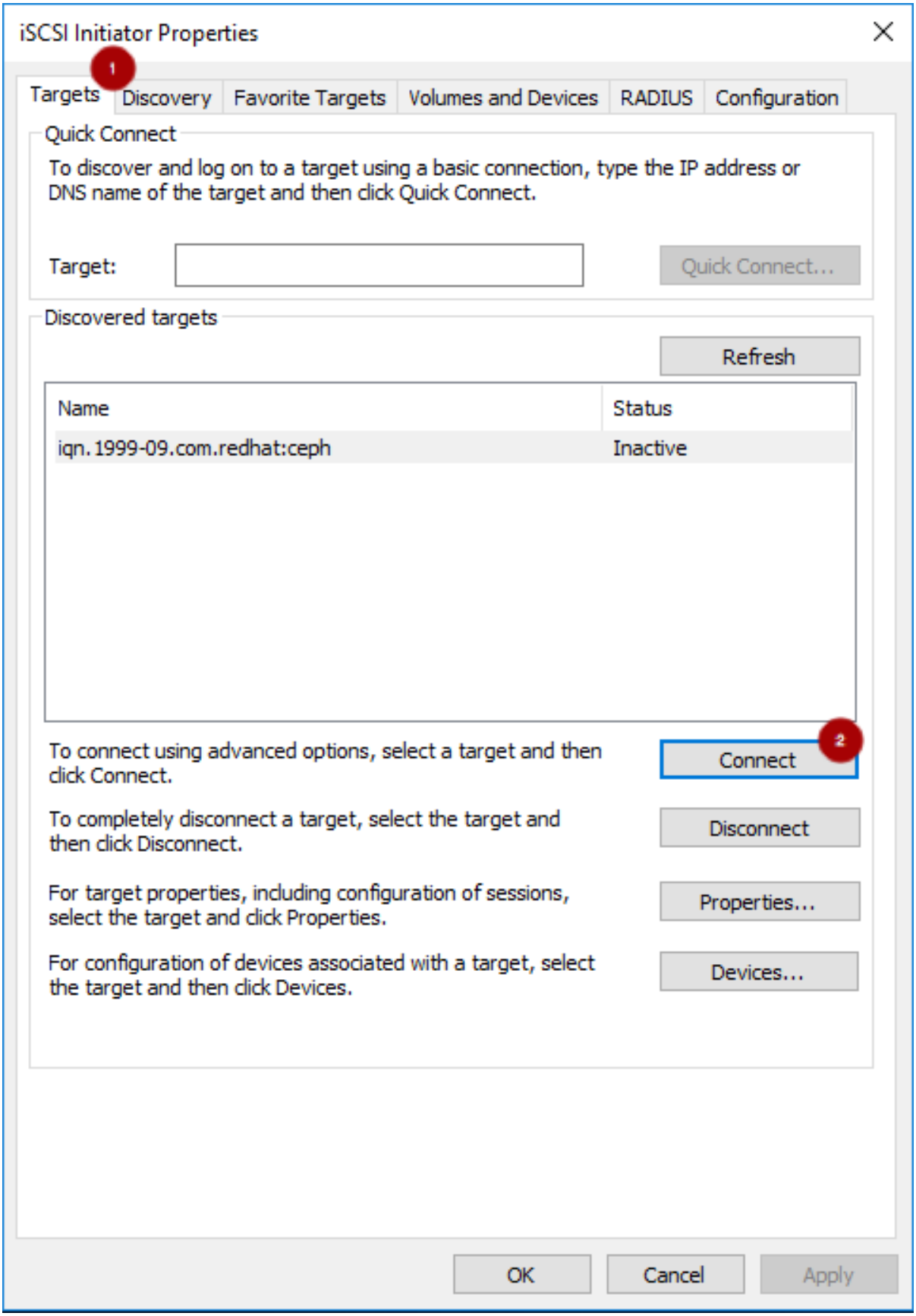

iSCSI Initiator Properties
1
Targets
Discovery
Favorite Targets
Volumes and Devices
RADIUS
Configuration
Discover Target Portal
Enter the IP address or DNS name and port number of the portal you want to add.
To change the default settings of the discovery of the target portal, click the Advanced button.
2
IP address or DNS name:
20.15.0.131
3
Port: (Default is 3260.)
3260
Advanced...
OK
Cancel
iSNS servers
The system is registered on the following iSNS servers:
Refresh
Name
To add an iSNS server, click Add Server.
Add Server...
To remove an iSNS server, select the server above and then click Remove.
Remove
OK
Cancel
Apply

4.

대상 탭

1

에서 대상을 선택하고 연결을 클릭합니다 2
:



5.

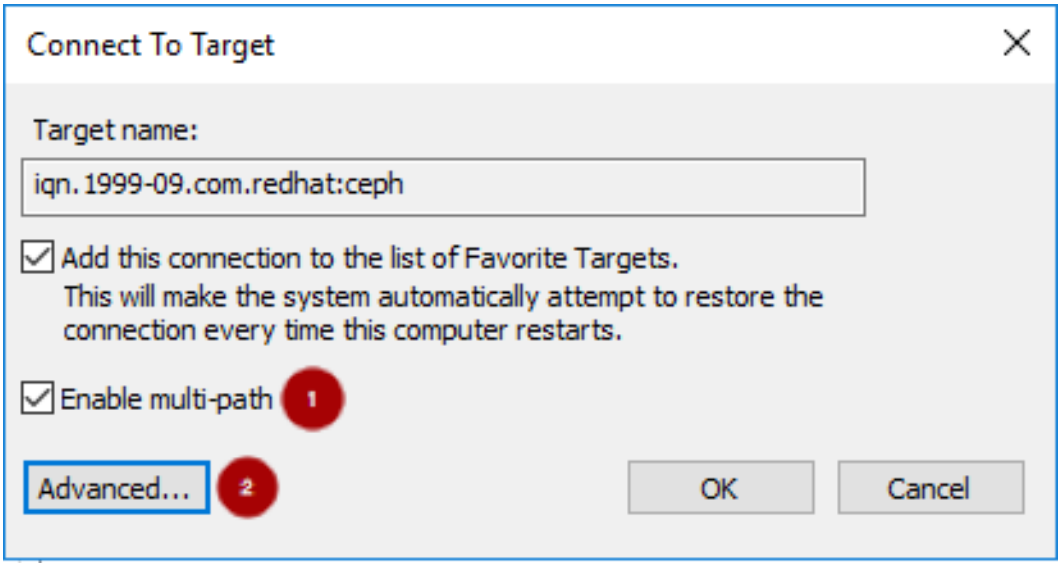
대상에 연결 창에서 다중 경로 활성화 옵션

1

을 선택하고 고급 버튼

2

를 클릭합니다.



6.

Connect using** 섹션에서 대상 포털 **IP

1

를 선택합니다.

2

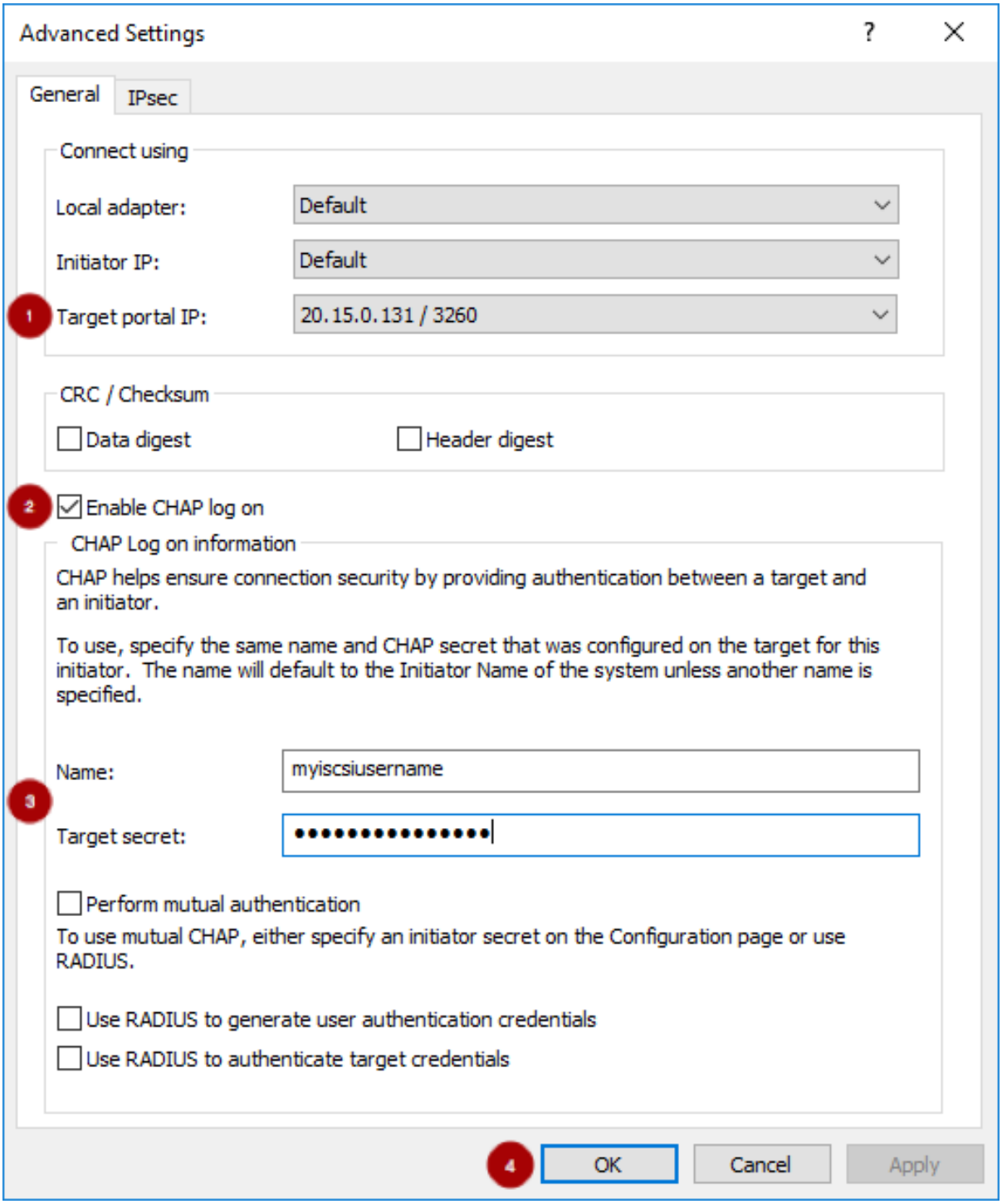
*에서 **CHAP** 로그인 활성화를 선택하고 **Ceph iSCSI Ansible** 클라이언트 인증 정보 섹션에서 이름 및 대상 시크릿 값*

3

를 입력하고 확인

4

을 클릭합니다.



중요

Windows Server 2016에서는 CHAP 시크릿을 12바이트 미만으로 허용하지 않습니다.

7. ***iSCSI 게이트웨이를 설정할 때 정의된 각 대상 포털에 대해 5단계와 6단계를 반복합니다.***

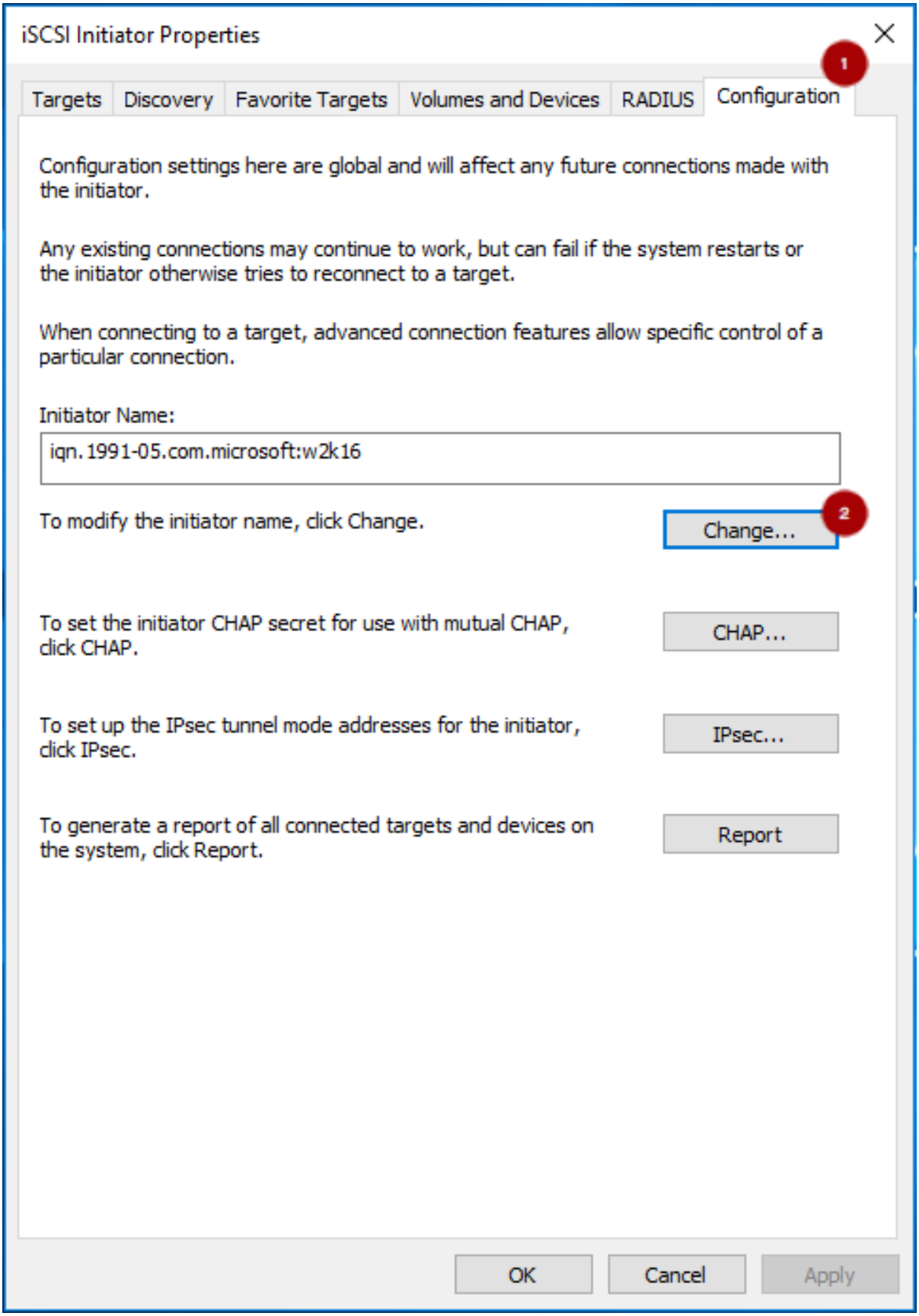
8. *이니시에이터 이름이 초기 설정 중에 사용된 이니시에이터 이름과 다른 경우 이니시에이터 이름의 이름을 변경합니다. **iSCSI** 이니시에이터 속성 창의 구성 탭*

1

에서 변경 버튼

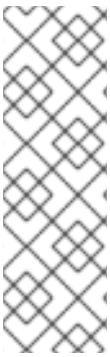
2

을 클릭하여 이니시에이터 이름의 이름을 바꿉니다.



*다중 경로 **IO** 설정:*

***MPIO** 로드 밸런싱 정책을 구성하고 시간 초과 및 재시도 옵션을 설정해도 **CHAP claim** 명령으로 **PowerShell** 을 사용합니다. **iSCSI Initiator** 툴에서는 나머지 옵션을 구성합니다.*



참고

***PowerShell**에서 **PDORemovePeriod** 옵션을 **120**초로 늘리는 것이 좋습니다. 애플리케이션에 따라 이 값을 조정해야 할 수 있습니다. 모든 경로가 다운되고 **120**초가 만료되면 운영 체제가 **IO** 요청 실패를 시작합니다.*

```
Set-MPIOSetting -NewPDORemovePeriod 120
```

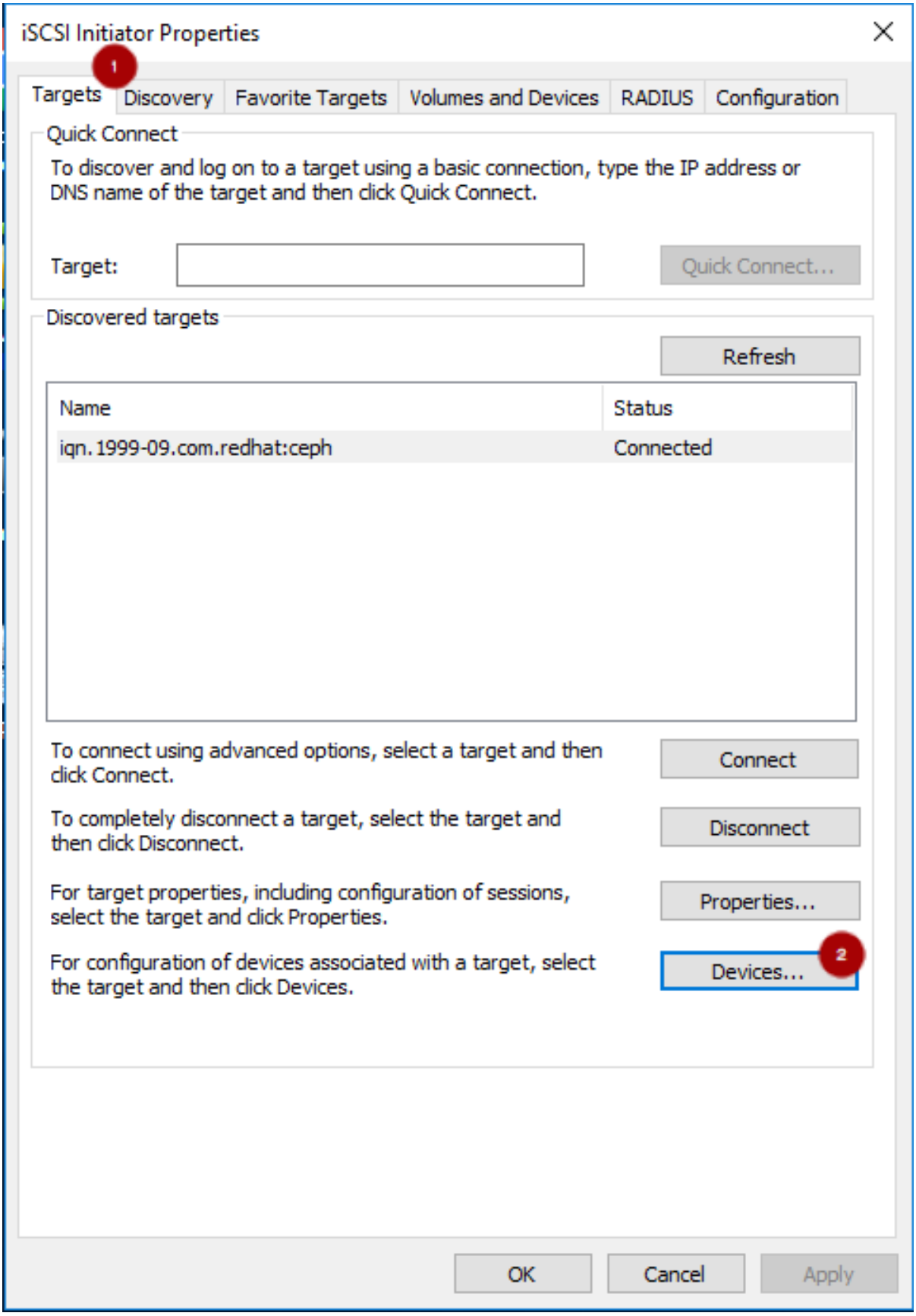
1. *장애 조치 정책 설정*

```
mpclaim.exe -l -m 1
```

1. *장애 조치 정책 확인*

```
mpclaim -s -m
MSDSM-wide Load Balance Policy: Fail Over Only
```

1. ***iSCSI Initiator** 툴을 사용하여 대상 탭*
   1
   *에서 **devices...** 버튼을 클릭합니다.*
   2
   :



2.

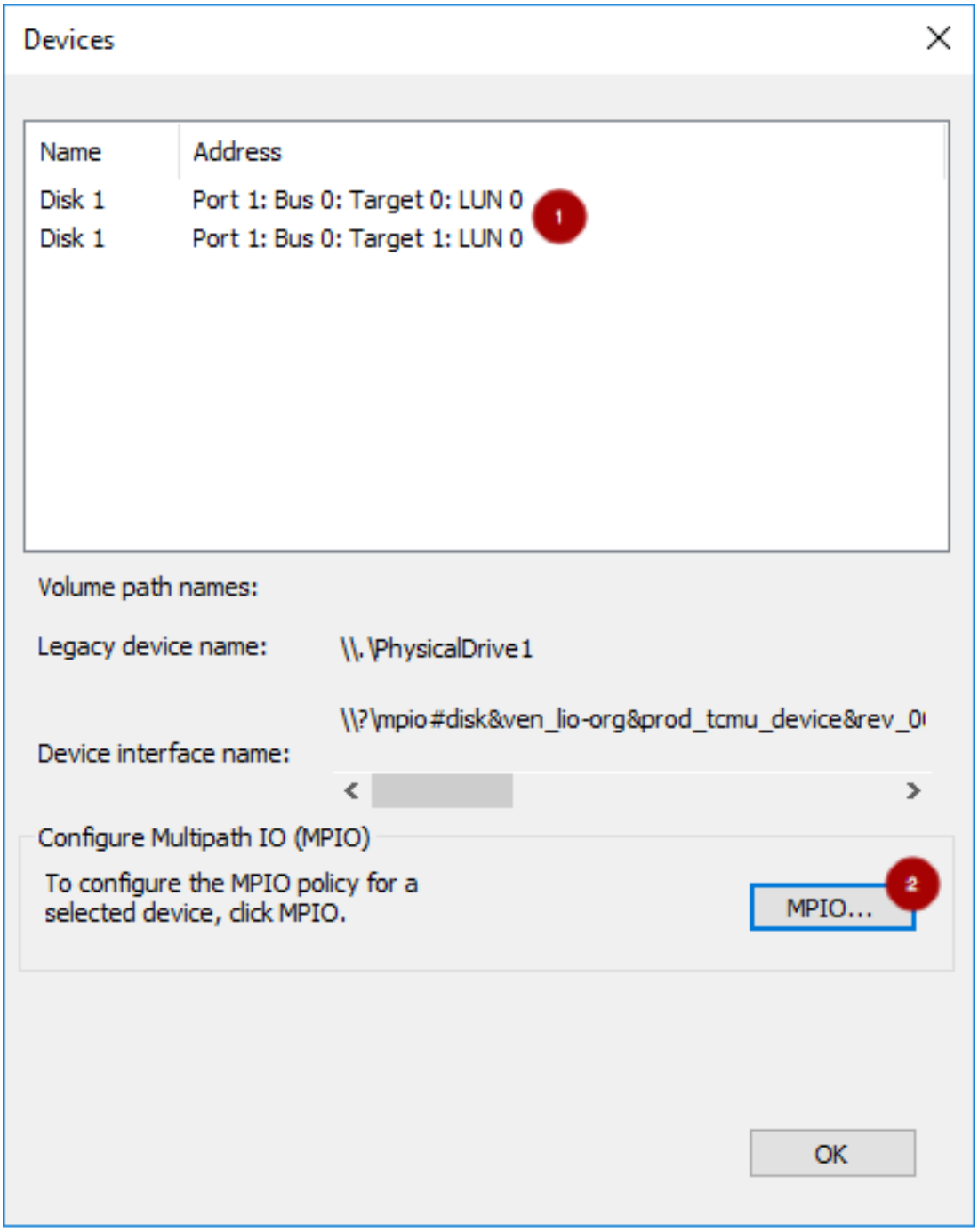
장치 창에서 디스크

1

를 선택하고 ***MPIO...*** *버튼*

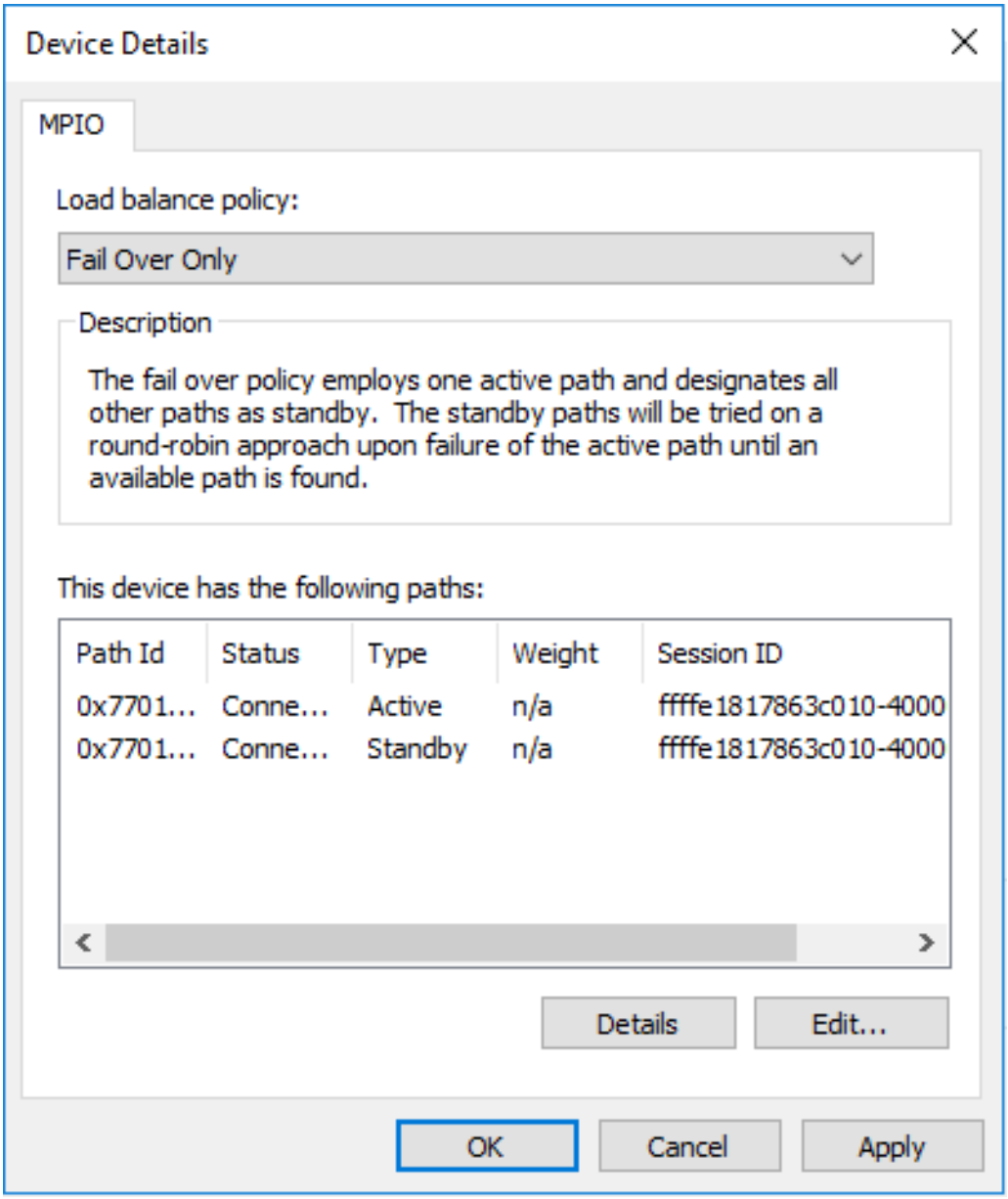
2

을 클릭합니다.



3.

장치 세부 정보 창에서 각 대상 포털의 경로가 표시됩니다. ***ceph-ansible*** *설정 방법을 사용하는 경우* ***iSCSI*** *게이트웨이는* ***ALUA****를 사용하여* ***iSCSI*** *이니시에이터에 기본 경로로 사용해야 하는 경로 및* ***iSCSI*** *게이트웨이를 알립니다.* ***Load Balancing Policy Fail Over****만 선택해야 합니다.*



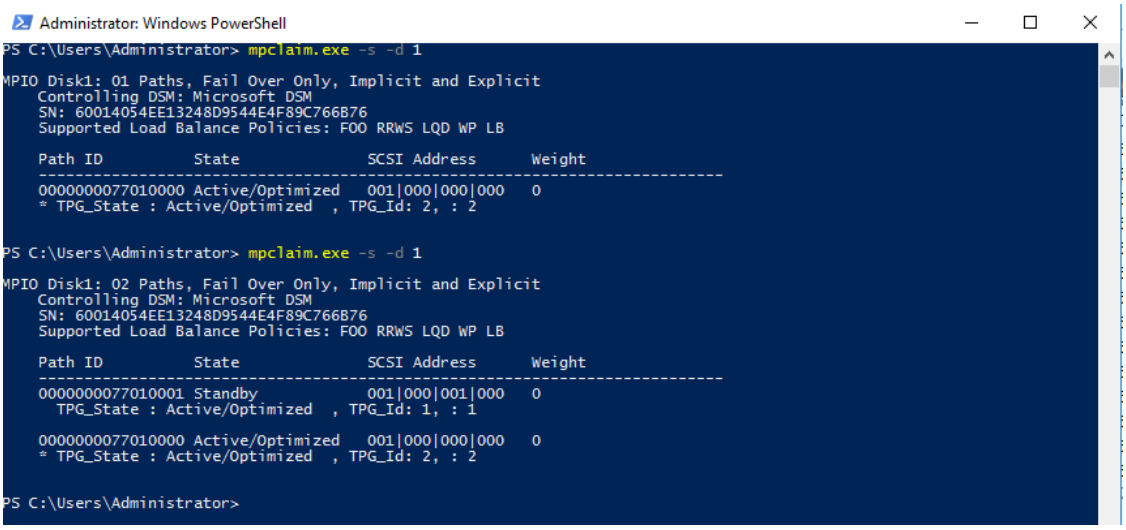
4. ***PowerShell에서 다중 경로 구성을 확인합니다.***

```
mpclaim -s -d $MPIO_DISK_ID
```

+ $MPIO_DISK_ID 를 적절한 디스크 식별자로 바꿉니다.

참고

***LUN**을 소유하고 있는 **iSCSI** 게이트웨이 노드의 경로인 활성/**Optimized** 경로가 하나 있으며 서로 다른 **iSCSI** 게이트웨이 노드에 대한 활성/정확하지 않은 경로가 있습니다.*



조정:

다음 레지스트리 설정을 사용하는 것이 좋습니다.

- ***Windows** 디스크 시간 제한*

  키

  ```
  HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Disk
  ```

  현재의

  ```
  TimeOutValue = 65
  ```

- ***Microsoft iSCSI** 이니시에이터 드라이버*

키

```
HKEY_LOCAL_MACHINE\\SYSTEM\CurrentControlSet\Control\Class\{4D36E97B-E325-11CE-BFC1-08002BE10318}\<Instance_Number>\Parameters
```

값

```
LinkDownTime = 25
SRBTimeoutDelta = 15
```

8.4.4. VMware ESX vSphere Web Client용 iSCSI Initiator

사전 요구 사항:

- ***VMware ESX 6.5** 이상에서 **VMFS 6**와 가상 머신 호환성*
- ***vSphere Web Client**에 액세스*
- ***VMware ESX** 호스트에 대한 루트 액세스로 **esxcli** 명령을 실행합니다.*

***iSCSI Discovery** 및 **Multipath** 장치 설정:*

1. ***HardwareAccelerated 10.0.0.1 (XCOPY)**을 비활성화합니다.*

```
# esxcli system settings advanced set --int-value 0 --option /DataMover/HardwareAcceleratedMove
```

-

2.

*iSCSI 소프트웨어를 활성화합니다. **Navigator** 창에서 스토리지*

1

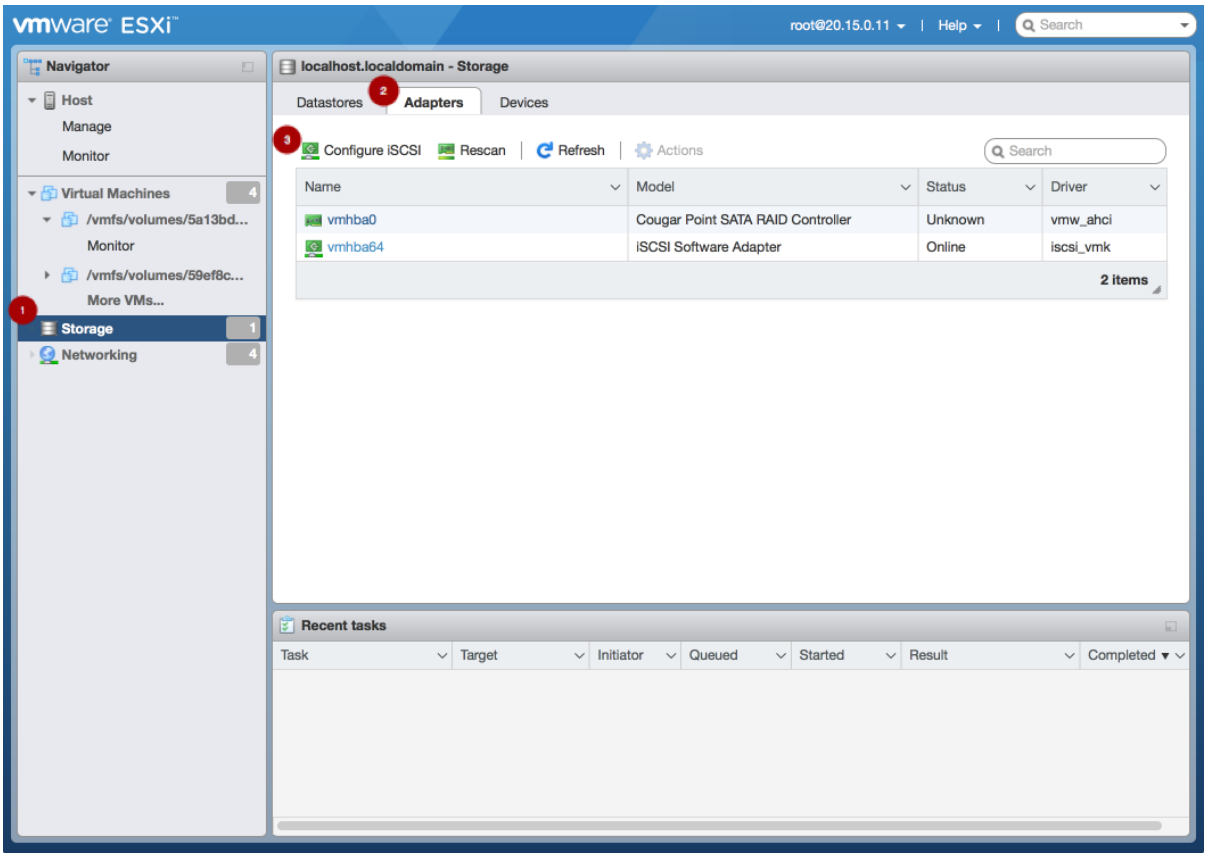
를 클릭합니다. 어댑터 탭

2

*을 선택합니다. **Confgure iSCSI***

3

를 클릭합니다.

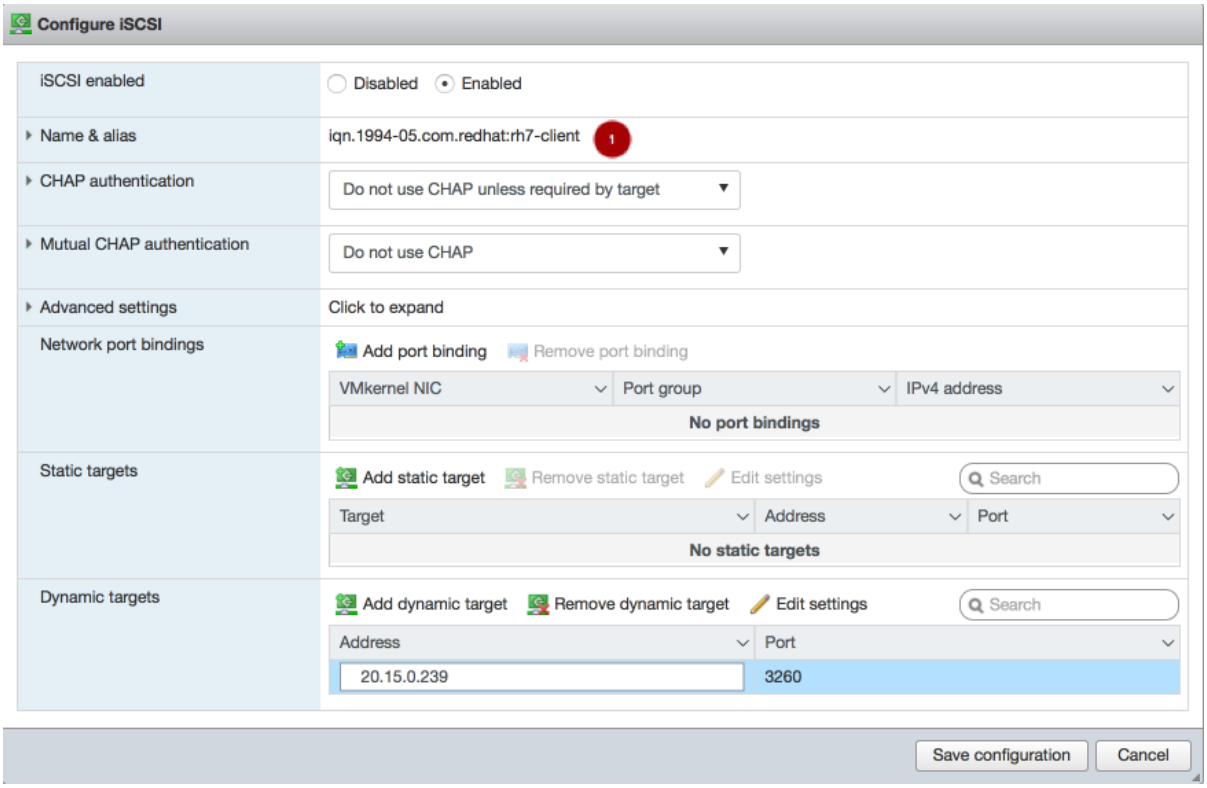


3.

이름 및 별칭 섹션에서 이니시에이터 이름을 확인합니다

1

.


Configure iSCSI
iSCSI enabled
Disabled
Enabled
Name & alias
iqn.1994-05.com.redhat:rh7-client
1
CHAP authentication
Do not use CHAP unless required by target
Mutual CHAP authentication
Do not use CHAP
Advanced settings
Click to expand
Network port bindings
Add port binding
Remove port binding
VMkernel NIC
Port group
IPv4 address
No port bindings
Static targets
Add static target
Remove static target
Edit settings
Search
Target
Address
Port
No static targets
Dynamic targets
Add dynamic target
Remove dynamic target
Edit settings
Search
Address
Port
20.15.0.239
3260
Save configuration
Cancel

참고

*이니시에이터 이름이 **gwcli** 를 사용하여 초기 설정 중에 클라이언트를 생성할 때 사용된 이니시에이터 이름과 다른 경우 또는 **Ansible client _connections:** 클라이언트 변수에 사용된 이니시에이터 이름이 다른 경우 다음 절차에 따라 이니시에이터 이름을 변경합니다. **VMware ESX** 호스트에서 다음 **esxcli** 명령을 실행합니다.*

1. ***iSCSI** 소프트웨어의 어댑터 이름을 가져옵니다.*

```
> esxcli iscsi adapter list
> Adapter  Driver     State   UID           Description
> -------  ---------  ------  ------------  ----------------------
> vmhba64  iscsi_vmk  online  iscsi.vmhba64  iSCSI Software Adapter
```

2. *이니시에이터 이름을 설정합니다.*

구문

```
> esxcli iscsi adapter set -A <adaptor_name> -n <initiator_name>
```

예제

```
> esxcli iscsi adapter set -A vmhba64 -n iqn.1994-05.com.redhat:rh7-client
```

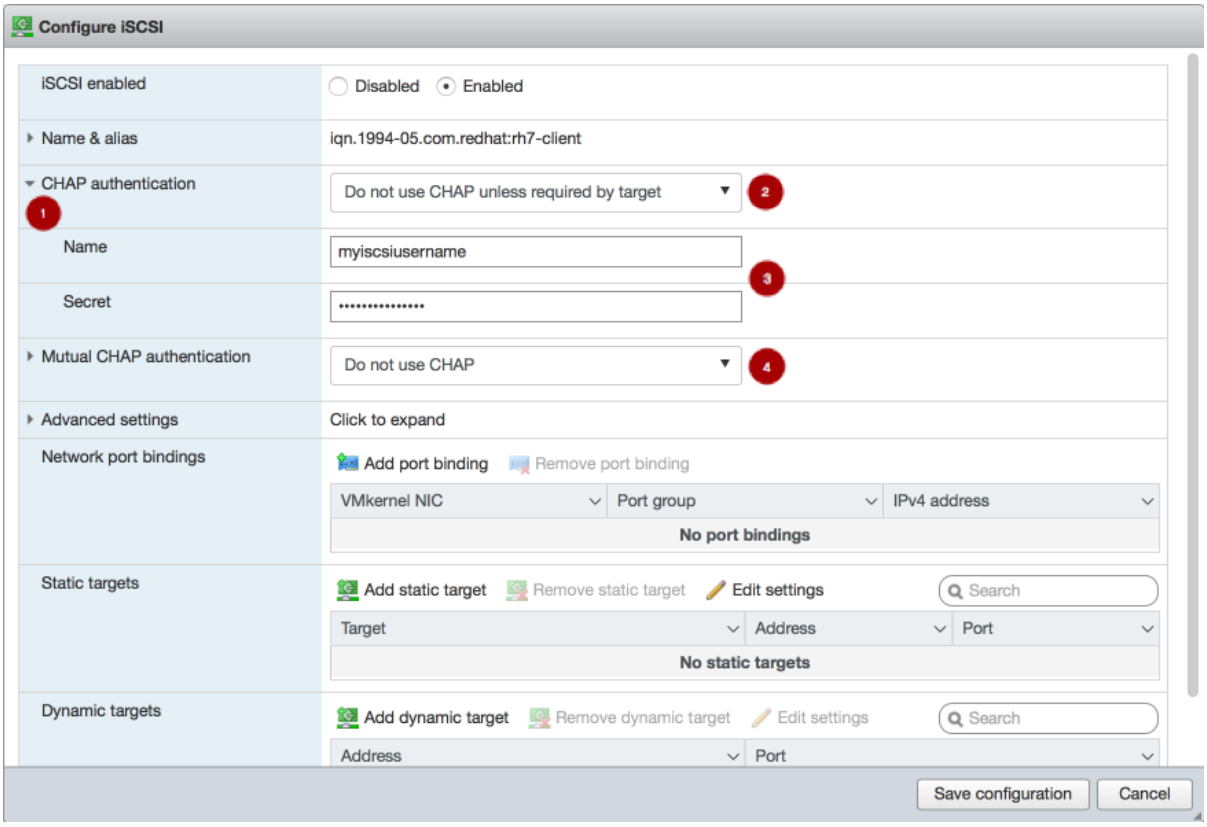
4. ***CHAP**을 구성합니다. **CHAP** 인증 섹션*

(1) *을 확장합니다. "대상에 의해 필요하지 않은 경우 **CHAP**을 사용하지 않음"을 선택합니다.*

(2) *. **gwcli auth** 명령 또는 **Ansible client_connections:** 자격 증명 변수를 사용하여 초기 설정에 사용된 **CHAP** 이름 및 **Secret** (3)*
*자격 증명을 입력합니다. 상호 **CHAP** 인증 섹션*

4

에 "CHAP을 사용하지 않음"이 선택되어 있는지 확인합니다.



주의

***vSphere Web Client**에는 **CHAP** 설정이 처음에 사용되지 않는 버그가 있습니다. **Ceph iSCSI** 게이트웨이 노드, 커널 로그에서 이 버그의 표시로 다음과 같은 오류가 표시됩니다.*

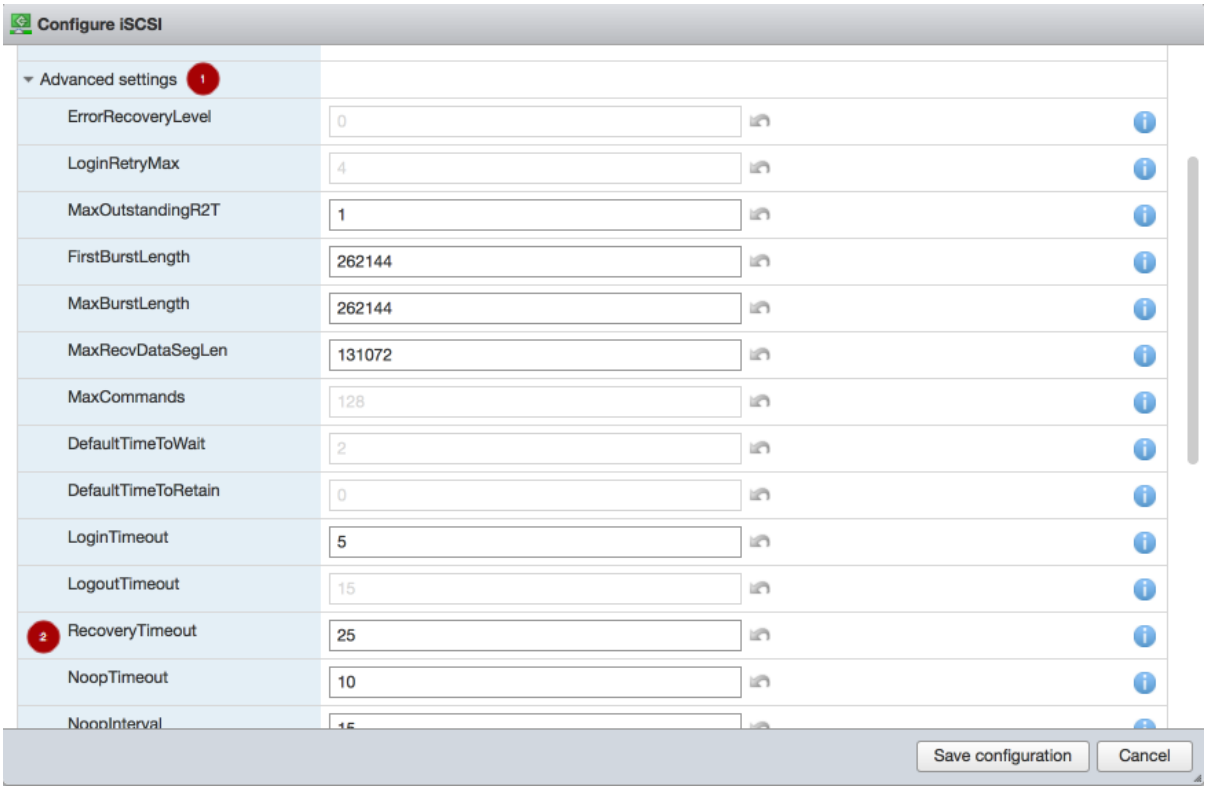
```
> kernel: CHAP user or password not set for Initiator ACL
> kernel: Security negotiation failed.
> kernel: iSCSI Login negotiation failed.
```

*이 버그를 해결하려면 **esxcli** 명령을 사용하여 **CHAP** 설정을 구성합니다. **authname** 인수는 **vSphere Web Client**의 **Name** 입니다.*

```
> esxcli iscsi adapter auth chap set --direction=uni --
authname=myiscsiusername --secret=myiscsipassword --
level=discouraged -A vmhba64
```

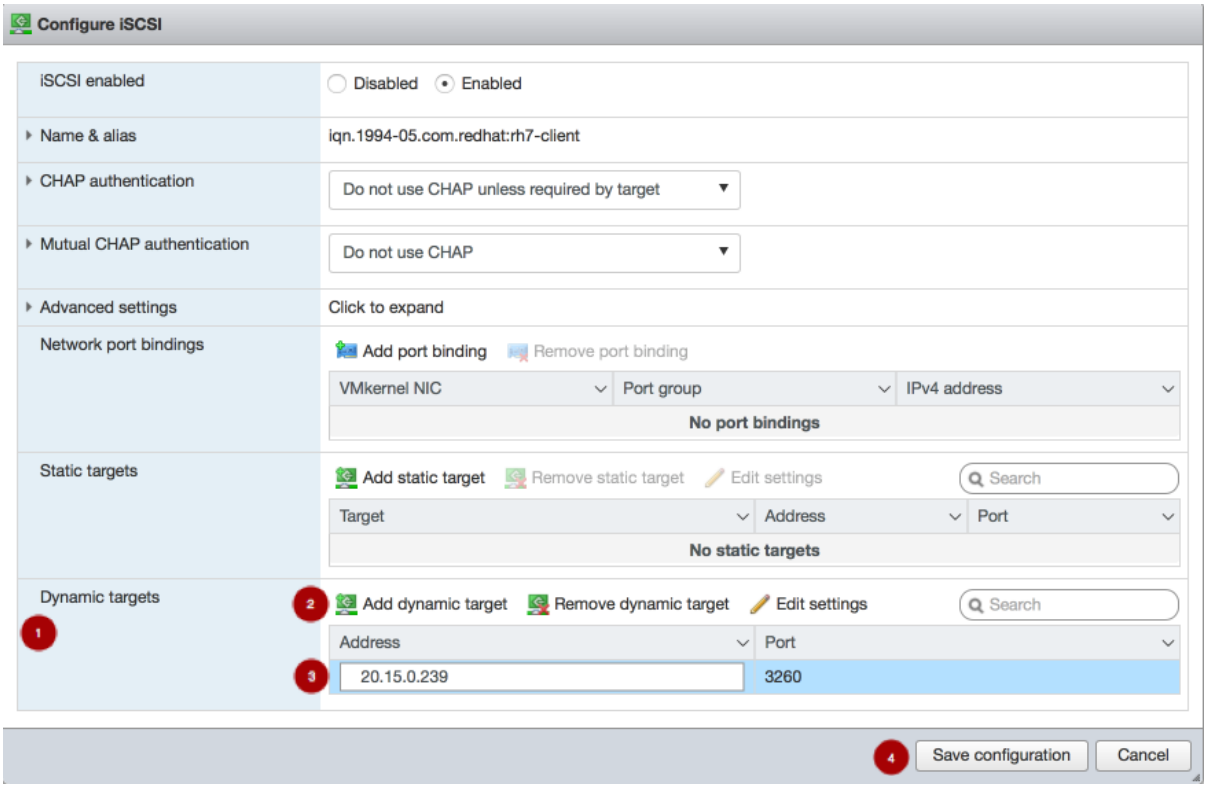
5.

iSCSI 설정을 구성합니다. 고급 설정 (1) *을 확장합니다. recoveryTimeout 값을 25* (2) *로 설정합니다.*



6.

검색 주소를 설정합니다. 동적 대상 섹션에서 동적 대상 (1) *추가 를 클릭합니다.* (2) *주소* (3) *에서 Ceph iSCSI 게이트웨이 중 하나의 IP 주소를 추가합니다. 하나의 IP 주소만 추가해야 합니다. 마지막으로 구성 저장 버튼* (4) *을 클릭합니다. 기본 인터페이스의 장치 탭에 RBD 이미지가 표시됩니다.*



참고

***LUN** 구성은 **ALUA SATP** 및 **MRU PSP**를 사용하여 자동으로 수행됩니다. 다른 **SATP** 및 **PSP**는 사용해서는 안 됩니다. 이는 **esxcli** 명령을 사용하여 확인할 수 있습니다.*

```
esxcli storage nmp path list -d eui.$DEVICE_ID
```

***$DEVICE_ID** 를 적절한 장치 식별자로 바꿉니다.*

7. *다중 경로가 올바르게 설정되었는지 확인합니다.*

   a. *장치를 나열합니다.*

   예제

```
# esxcli storage nmp device list | grep iSCSI
   Device Display Name: LIO-ORG iSCSI Disk
(naa.6001405f8d087846e7b4f0e9e3acd44b)
   Device Display Name: LIO-ORG iSCSI Disk
(naa.6001405057360ba9b4c434daa3c6770c)
```

b.

*이전 단계에서 **Ceph iSCSI** 디스크에 대한 다중 경로 정보를 가져옵니다.*

예제

```
# esxcli storage nmp path list -d naa.6001405f8d087846e7b4f0e9e3acd44b

iqn.2005-03.com.ceph:esx1-00023d000001,iqn.2003-01.com.redhat.iscsi-gw:iscsi-
igw,t,1-naa.6001405f8d087846e7b4f0e9e3acd44b
   Runtime Name: vmhba64:C0:T0:L0
   Device: naa.6001405f8d087846e7b4f0e9e3acd44b
   Device Display Name: LIO-ORG iSCSI Disk
(naa.6001405f8d087846e7b4f0e9e3acd44b)
   Group State: active
   Array Priority: 0
   Storage Array Type Path Config:
{TPG_id=1,TPG_state=AO,RTP_id=1,RTP_health=UP}
   Path Selection Policy Path Config: {current path; rank: 0}

iqn.2005-03.com.ceph:esx1-00023d000002,iqn.2003-01.com.redhat.iscsi-gw:iscsi-
igw,t,2-naa.6001405f8d087846e7b4f0e9e3acd44b
   Runtime Name: vmhba64:C1:T0:L0
   Device: naa.6001405f8d087846e7b4f0e9e3acd44b
   Device Display Name: LIO-ORG iSCSI Disk
(naa.6001405f8d087846e7b4f0e9e3acd44b)
   Group State: active unoptimized
   Array Priority: 0
   Storage Array Type Path Config:
{TPG_id=2,TPG_state=ANO,RTP_id=2,RTP_health=UP}
   Path Selection Policy Path Config: {non-current path; rank: 0}
```

*예제 출력에서 각 경로에는 다음 부분과 함께 **iSCSI/SCSI** 이름이 있습니다.*

*이니시에이터 이름 **=>-<n .ECDHE-03.com.ceph:esx1 ISID = 10.0.0.1 23d00000 2** 대상 이름 **= redhat.iscsi-gw:iscsi-igw Target** 포트 그룹 **= naa.6001405f8d087846e7b4f0e3e3acd44b***

***Group State** 값 **active** 는 **iSCSI** 게이트웨이의 활성 상태 경로임을 나타냅니다. **gwcli** 명령은 **iSCSI** 게이트웨이 소유자로 활성 상태를 나열합니다. 나머지 경로는 조정 되지 않은*

Group State 값을 가지며 활성 경로가 dead 상태가 되면 장애 조치 경로가 됩니다.

8. *해당 iSCSI 게이트웨이에 대한 모든 경로를 일치시키려면 다음 명령을 실행합니다.*

```
# esxcli iscsi session connection list
```

출력 예

```
vmhba64,iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw,00023d000001,0
   Adapter: vmhba64
   Target: iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw
   ISID: 00023d000001
   CID: 0
   DataDigest: NONE
   HeaderDigest: NONE
   IFMarker: false
   IFMarkerInterval: 0
   MaxRecvDataSegmentLength: 131072
   MaxTransmitDataSegmentLength: 262144
   OFMarker: false
   OFMarkerInterval: 0
   ConnectionAddress: 10.172.19.21
   RemoteAddress: 10.172.19.21
   LocalAddress: 10.172.19.11
   SessionCreateTime: 08/16/18 04:20:06
   ConnectionCreateTime: 08/16/18 04:20:06
   ConnectionStartTime: 08/16/18 04:30:45
   State: logged_in

vmhba64,iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw,00023d000002,0
   Adapter: vmhba64
   Target: iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw
   ISID: 00023d000002
   CID: 0
   DataDigest: NONE
   HeaderDigest: NONE
   IFMarker: false
   IFMarkerInterval: 0
   MaxRecvDataSegmentLength: 131072
   MaxTransmitDataSegmentLength: 262144
   OFMarker: false
   OFMarkerInterval: 0
   ConnectionAddress: 10.172.19.22
   RemoteAddress: 10.172.19.22
   LocalAddress: 10.172.19.12
   SessionCreateTime: 08/16/18 04:20:06
   ConnectionCreateTime: 08/16/18 04:20:06
   ConnectionStartTime: 08/16/18 04:30:41
   State: logged_in
```

경로 이름과 **ISID** 값과 일치하고 **RemoteAddress** 값은 소유 **iSCSI** 게이트웨이의 **IP** 주소입니다.

8.5. ANSIBLE을 사용하여 CEPH ISCSI 게이트웨이 업그레이드

롤링 업그레이드를 위해 설계된 **Ansible** 플레이북을 사용하여 **Red Hat Ceph Storage iSCSI** 게이트웨이 업그레이드를 수행할 수 있습니다.

사전 요구 사항

- 실행 중인 **Ceph iSCSI** 게이트웨이.
- 실행 중인 **Red Hat Ceph Storage** 클러스터.

절차

1. **Ansible** 인벤토리 파일(**/etc/ansible/hosts**)에 올바른 **iSCSI** 게이트웨이 노드가 나열되어 있는지 확인합니다.
2. 롤링 업그레이드 플레이북을 실행합니다.

```
[admin@ansible ~]$ ansible-playbook rolling_update.yml
```

3. 사이트 플레이북을 실행하여 업그레이드를 완료합니다.

```
[admin@ansible ~]$ ansible-playbook site.yml --limit iscsigws
```

8.6. 명령줄 인터페이스를 사용하여 CEPH ISCSI 게이트웨이 업그레이드

Red Hat Ceph Storage iSCSI 게이트웨이 게이트웨이 업그레이드는 한 번에 하나의 **iSCSI** 게이트웨이 노드를 업그레이드하여 롤링 방식으로 수행할 수 있습니다.

주의

***Ceph OSD**를 업그레이드 및 다시 시작하는 동안 **iSCSI** 게이트웨이를 업그레이드 하지 마십시오. **OSD** 업그레이드가 완료되고 스토리지 클러스터가 **active+clean** 상태가 될 때까지 기다립니다.*

사전 요구 사항

- *실행 중인 **Ceph iSCSI** 게이트웨이.*
- *실행 중인 **Red Hat Ceph Storage** 클러스터.*
- ***iSCSI** 게이트웨이 노드에 **root** 액세스 권한을 갖도록 합니다.*

절차

1. ***iSCSI** 게이트웨이 패키지를 업데이트합니다.*

```
[root@igw ~]# yum update ceph-iscsi-config ceph-iscsi-cli
```

2. ***iSCSI** 게이트웨이 데몬을 중지합니다.*

```
[root@igw ~]# systemctl stop rbd-target-api
[root@igw ~]# systemctl stop rbd-target-gw
```

3. ***iSCSI** 게이트웨이 데몬이 완전히 중지되었는지 확인합니다.*

```
[root@igw ~]# systemctl status rbd-target-gw
```

   a. ***rbd-target-gw** 서비스가 성공적으로 중지되면 **4**단계로 건너뜁니다.*

   b. ***rbd-target-gw** 서비스가 중지되지 않으면 다음 단계를 수행합니다.*

i.

targetcli 패키지가 설치되지 않은 경우 **targetcli** 패키지를 설치합니다.

```
[root@igw ~]# yum install targetcli
```

ii.

기존 대상 오브젝트를 확인합니다.

```
[root@igw ~]# targetlci ls
```

출력 예

```
o- / ..... [...]
o- backstores ..... [...]
| o- user:rbd ..... [Storage Objects: 0]
o- iscsi ..... [Targets: 0]
```

보조 저장소 및 스토리지 오브젝트 가 비어 있으면 **iSCSI** 대상이 완전히 종료되고 **4**단계로 건너뛸 수 있습니다.

iii.

대상 오브젝트가 아직 있는 경우 다음 명령을 실행하여 모든 대상 오브젝트를 강제로 제거합니다.

```
[root@igw ~]# targetlci clearconfig confirm=True
```

주의

여러 서비스가 **iSCSI** 대상을 사용하는 경우 대화형 모드에서 **targetcli** 를 실행하여 특정 오브젝트를 삭제합니다.

4.

tcmu-runner 패키지를 업데이트합니다.

```
[root@igw ~]# yum update tcmu-runner
```

5. **tcmu-runner** 서비스를 중지합니다.

```
[root@igw ~]# systemctl stop tcmu-runner
```

6. 다음 순서로 모든 **iSCSI** 게이트웨이 서비스를 다시 시작합니다.

```
[root@igw ~]# systemctl start tcmu-runner
[root@igw ~]# systemctl start rbd-target-gw
[root@igw ~]# systemctl start rbd-target-api
```

8.7. ISCSI 게이트웨이 모니터링

Red Hat은 내보낸 **RADOS Block Device(RBD)** 이미지의 성능을 모니터링하는 **Ceph iSCSI** 게이트웨이 환경에 추가 툴을 제공합니다.

gwtop 툴은 **iSCSI**를 통해 클라이언트에 내보낸 **RBD** 이미지의 집계 성능 지표를 표시하는 최상위툴입니다. 메트릭은 **PMDA(Performance Metrics Domain Agent)**에서 가져옵니다. **Linux-IO** 대상(**LIO**) **PMDA**의 정보는 연결된 클라이언트 및 관련 **I/O** 지표로 내보낸 각 **RBD** 이미지를 나열하는 데 사용됩니다.

요구 사항:

- 실행 중인 **Ceph iSCSI** 게이트웨이

설치:

root 사용자로 **iSCSI** 게이트웨이 노드에서 다음 단계를 수행합니다.

1. **Ceph** 툴 리포지토리를 활성화합니다.

```
# subscription-manager repos --enable=rhel-7-server-rhceph-3-tools-rpms
```

2. **ceph-iscsi-tools** 패키지를 설치합니다.

```
# yum install ceph-iscsi-tools
```

3. *성능 **co-pilot** 패키지를 설치합니다.*

```
# yum install pcp
```

참고

*성능 공동 컴파일에 대한 자세한 내용은 **Red Hat Enterprise Linux** 성능 튜닝 가이드를 참조하십시오.*

4. ***LIO PMDA** 패키지를 설치합니다.*

```
# yum install pcp-pmda-lio
```

5. ***performance co-pilot** 서비스를 활성화하고 시작합니다.*

```
# systemctl enable pmcd
# systemctl start pmcd
```

6. ***pcp-pmda-lio** 에이전트를 등록합니다.*

```
cd /var/lib/pcp/pmdas/lio
./Install
```

*기본적으로 **gwtop** 는 **iSCSI** 게이트웨이 구성 오브젝트가 **rbd** 풀의 **gateway.conf** 라는 **RADOS** 오브젝트에 저장된 것으로 가정합니다. 이 구성은 성능 통계를 수집하기 위해 연결할 **iSCSI** 게이트웨이를 정의합니다. **-g** 또는 **-c** 플래그를 사용하여 재정의할 수 있습니다. 자세한 내용은 **gwtop --help** 를 참조하십시오.*

***LIO** 구성은 성능 **co-pilot**에서 추출할 성능 통계 유형을 결정합니다. **gwtop** 가 시작되면 **LIO** 구성을 살펴보고 사용자 공간 디스크를 찾으면 **gwtop** 에서 **LIO** 수집기를 자동으로 선택합니다.*

***gwtop** 출력 예:*

*사용자 백업 스토리지(**iPXEMU**) 장치의 경우:*

```
gwtop  2/2 Gateways   CPU% MIN:  4 MAX:  5    Network Total In:    2M  Out:    3M   10:20:00
Capacity:   8G    Disks:   8   IOPS:  503   Clients:  1   Ceph: HEALTH_OK          OSDs:   3
Pool.Image       Src    Size     iops     rMB/s     wMB/s   Client
```

```
iscsi.t1703          500M     0     0.00     0.00
iscsi.testme1        500M     0     0.00     0.00
iscsi.testme2        500M     0     0.00     0.00
iscsi.testme3        500M     0     0.00     0.00
iscsi.testme5        500M     0     0.00     0.00
rbd.myhost_1     T     4G     504     1.95     0.00   rh460p(CON)
rbd.test_2           1G     0     0.00     0.00
rbd.testme           500M     0     0.00     0.00
```

*클라이언트 열에서 **(CON) iSCSI** 이니시에이터(클라이언트)가 현재 **iSCSI** 게이트웨이에 기록되었음을 나타냅니다. **-multi-** 가 표시되면 여러 클라이언트가 단일 **RBD** 이미지에 매핑됩니다.*

주의

***SCSI** 영구 예약은 지원되지 않습니다. **SCSI** 영구 예약을 사용하지 않는 클러스터 인식 파일 시스템 또는 클러스터링 소프트웨어를 사용하는 경우 여러 **iSCSI** 이니시에이터를 **RBD** 이미지에 매핑하는 기능이 지원됩니다. 예를 들어 **ATS**를 사용하는 **VMware vSphere** 환경은 지원되지만 **MSCS(Microsoft**의 클러스터링 서버) 사용은 지원되지 않습니다.*

부록 A. 샘플 ISCSIGWS.YML 파일

```
# Variables here are applicable to all host groups NOT roles

# This sample file generated by generate_group_vars_sample.sh

# Dummy variable to avoid error because ansible does not recognize the
# file as a good configuration file when no variable in it.
dummy:

# You can override vars by using host or group vars

############
# GENERAL #
############
# Specify the iqn for ALL gateways. This iqn is shared across the gateways, so an iscsi
# client sees the gateway group as a single storage subsystem.
#gateway_iqn: "iqn.2003-01.com.redhat.iscsi-gw:ceph-igw"

# gateway_ip_list provides a list of the IP Addrresses - one per gateway - that will be used
# as an iscsi target portal ip. The list must be comma separated - and the order determines
# the sequence of TPG's within the iscsi target across each gateway. Once set, additional
# gateways can be added, but the order must *not* be changed.
#gateway_ip_list: 0.0.0.0

# rbd_devices defines the images that should be created and exported from the iscsi gateways.
# If the rbd does not exist, it will be created for you. In addition you may increase the
# size of rbd's by changing the size parameter and rerunning the playbook. A size value lower
# than the current size of the rbd is ignored.
#
# the 'host' parameter defines which of the gateway nodes should handle the physical
# allocation/expansion or removal of the rbd
# to remove an image, simply use a state of 'absent'. This will first check the rbd is not allocated
# to any client, and the remove it from LIO and then delete the rbd image
#
# NB. this variable definition can be commented out to bypass LUN management
#
# Example:
#
#rbd_devices:
#  - { pool: 'rbd', image: 'ansible1', size: '30G', host: 'ceph-1', state: 'present' }
#  - { pool: 'rbd', image: 'ansible2', size: '15G', host: 'ceph-1', state: 'present' }
#  - { pool: 'rbd', image: 'ansible3', size: '30G', host: 'ceph-1', state: 'present' }
#  - { pool: 'rbd', image: 'ansible4', size: '50G', host: 'ceph-1', state: 'present' }
#rbd_devices: {}

# client_connections defines the client ACL's to restrict client access to specific LUNs
# The settings are as follows;
# - image_list is a comma separated list of rbd images of the form <pool name>.<rbd_image_name>
# - chap supplies the user and password the client will use for authentication of the
#   form <user>/<password>
# - status shows the intended state of this client definition - 'present' or 'absent'
#
# NB. this definition can be commented out to skip client (nodeACL) management
#
```

```
# Example:
#
#client_connections:
#  - { client: 'iqn.1994-05.com.redhat:rh7-iscsi-client', image_list: 'rbd.ansible1,rbd.ansible2', chap:
'rh7-iscsi-client/redhat', status: 'present' }
#  - { client: 'iqn.1991-05.com.microsoft:w2k12r2', image_list: 'rbd.ansible4', chap:
'w2k12r2/microsoft_w2k12', status: 'absent' }

#client_connections: {}


# Whether or not to generate secure certificate to iSCSI gateway nodes
#generate_crt: False


###################
# RBD-TARGET-API #
###################
# Optional settings related to the CLI/API service
#api_user: admin
#api_password: admin
#api_port: 5000
#api_secure: false
#loop_delay: 1
#trusted_ip_list: 192.168.122.1


##########
# DOCKER #
##########

# Resource limitation
# For the whole list of limits you can apply see: docs.docker.com/engine/admin/resource_constraints
# Default values are based from: https://access.redhat.com/documentation/en-
us/red_hat_ceph_storage/2/html/red_hat_ceph_storage_hardware_guide/minimum_recommendations

# These options can be passed using the 'ceph_mds_docker_extra_env' variable.

# TCMU_RUNNER resource limitation
#ceph_tcmu_runner_docker_memory_limit: 1g
#ceph_tcmu_runner_docker_cpu_limit: 1

# RBD_TARGET_GW resource limitation
#ceph_rbd_target_gw_docker_memory_limit: 1g
#ceph_rbd_target_gw_docker_cpu_limit: 1

# RBD_TARGET_API resource limitation
#ceph_rbd_target_api_docker_memory_limit: 1g
#ceph_rbd_target_api_docker_cpu_limit: 1
```