Red Hat Ceph Storage 5

블록 장치 가이드

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

Last Updated: 2022-09-04

Red Hat Ceph Storage 5 블록 장치 가이드

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

Enter your first name here. Enter your surname here.
Enter your organisation's name here. Enter your organisational division here.
Enter your email address here.

법적 공지

초록

이 문서에서는 Red Hat Ceph Storage 블록 장치를 관리, 생성, 구성 및 사용하는 방법을 설명합니다. Red Hat은 코드, 문서, 웹 속성에서 문제가 있는 용어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지에서 참조하십시오.

차례

1장. CEPH 블록 장치 소개

블록은 일련의 바이트 길이입니다(예: 데이터의 512바이트 블록). 많은 블록을 단일 파일로 결합하면 에서 읽고 에 쓸 수 있는 스토리지 장치로 사용할 수 있습니다. 블록 기반 스토리지 인터페이스는 다음과 같은 회전 미디어를 사용하여 데이터를 저장하는 가장 일반적인 방법입니다.

- 하드 드라이브
- CD/DVD 디스크
- 플로피 디스크
- 기존의 9개 트랙 테이프

블록 장치 인터페이스를 쉽게 사용할 경우 가상 블록 장치는 Red Hat Ceph Storage와 같은 대용량 데이터 스토리지 시스템과 상호 작용하는 데 이상적입니다.

Ceph 블록 장치는 Ceph 스토리지 클러스터에서 씬 프로비저닝, 크기 조정 및 저장 데이터가 여러 OSD(오브젝트 스토리지 장치)에서 스트라이핑됩니다. Ceph 블록 장치는 RADOS(Reliable Autonomic Distributed Object Store) 블록 장치(RBD)라고 합니다. Ceph 블록 장치는 다음과 같은 RADOS 기능을 활용합니다.

- 스냅샷
- 복제
- 데이터 일관성

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

Ceph 블록 장치는 **libvirt** 및 QEMU 유틸리티를 사용하여 Ceph 블록 장치와 통합되는 OpenStack과 같은 클라우드 기반 컴퓨팅 시스템(QEMU)과 같은 KVM(커널 가상 시스템)에 무한 확장성과 함께 고성능을 제공합니다. 동일한 스토리지 클러스터를 사용하여 Ceph Object Gateway 및 Ceph 블록 장치를 동시에 작동할 수 있습니다.

중요

Ceph 블록 장치를 사용하려면 실행 중인 Ceph 스토리지 클러스터에 액세스할 수 있어야 합니다. Red Hat Ceph Storage 클러스터 설치에 대한 자세한 내용은 *Red Hat Ceph Storage 설치 가이드를 참조하십시오*.

2장. CEPH 블록 장치

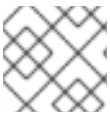
스토리지 관리자는 Ceph의 블록 장치 명령을 잘 알고 있으면 Red Hat Ceph Storage 클러스터를 효과적으로 관리할 수 있습니다. Ceph 블록 장치의 다양한 기능을 활성화 및 비활성화하는 것과 함께 블록 장치 풀과 이미지를 만들고 관리할 수 있습니다.

2.1. 사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.

2.2. 명령 도움말 표시

명령줄 인터페이스에서 명령 및 하위 명령 온라인 도움말을 표시합니다.



참고

h 옵션은 사용 가능한 모든 명령에 대한 도움말을 계속 표시합니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

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

   구문

   ```
   rbd help COMMAND SUBCOMMAND
   ```

2. **snap list** 명령에 대한 도움말을 표시하려면 다음을 수행합니다.

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

2.3. 블록 장치 풀 생성

블록 장치 클라이언트를 사용하기 전에 **rbd** 풀이 활성화되어 초기화되었는지 확인합니다.

참고

풀을 소스로 지정하기 전에 먼저 **생성해야** 합니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. **rbd** 풀을 생성하려면 다음을 실행합니다.

   구문

   ```
   ceph osd pool create POOL_NAME PG_NUM
   ceph osd pool application enable POOL_NAME rbd
   rbd pool init -p POOL_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# ceph osd pool create pool1
   [root@rbd-client ~]# ceph osd pool application enable pool1 rbd
   [root@rbd-client ~]# rbd pool init -p pool1
   ```

추가 리소스

- 자세한 내용은 *Red Hat Ceph Storage Storage Strategies Guide* 의 *Pools* 장을 참조하십시오.

2.4. 블록 장치 이미지 생성

블록 장치를 노드에 추가하기 전에 Ceph 스토리지 클러스터에 블록 장치를 위한 이미지를 만듭니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 블록 장치 이미지를 생성하려면 다음 명령을 실행합니다.

   구문

   ```
   rbd create IMAGE_NAME --size MEGABYTES --pool POOL_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd create image1 --size 1024 --pool pool1
   ```

   이 예제에서는 pool1 풀에 정보를 저장하는 **image1** 이라는 1GB 이미지를 생성합니다 **.**

   참고

   이미지를 생성하기 전에 풀이 있는지 확인합니다.

추가 리소스

- 자세한 내용은 *Red Hat Ceph Storage 블록 장치 가이드의 블록 장치 풀 생성* 섹션을 참조하십시오.

2.5. 블록 장치 이미지 나열

블록 장치 이미지를 나열합니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. **rbd** 풀의 블록 장치를 나열하려면 다음 명령을 실행합니다.

참고

rbd 는 기본 풀 이름입니다.

예제

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

2. 특정 풀의 블록 장치를 나열하려면 다음을 수행합니다.

구문

```
rbd ls POOL_NAME
```

예제

```
[root@rbd-client ~]# rbd ls pool1
```

2.6. 블록 장치 이미지 정보 검색

블록 장치 이미지에 대한 정보를 검색합니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 기본 **rbd** 풀의 특정 이미지에서 정보를 검색하려면 다음 명령을 실행합니다.

구문

```
rbd --image IMAGE_NAME info
```

예제

```
[root@rbd-client ~]# rbd --image image1 info
```

2. 풀 내에서 이미지 정보를 검색하려면 다음을 수행합니다.

구문

```
rbd --image IMAGE_NAME -p POOL_NAME info
```

예제

```
[root@rbd-client ~]# rbd --image image1 -p pool1 info
```

2.7. 블록 장치 이미지 크기 조정

Ceph 블록 장치 이미지는 씬 프로비저닝됩니다. 데이터 저장을 시작하기 전까지는 실제 스토리지를 사용하지 않습니다. 그러나 여기에는 **--size** 옵션으로 설정한 최대 용량이 있습니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 기본 **rbd** 풀의 Ceph 블록 장치 이미지의 최대 크기를 늘리거나 줄이려면 다음을 수행합니다.

구문

```
rbd resize --image IMAGE_NAME --size SIZE
```

예제

```
[root@rbd-client ~]# rbd resize --image image1 --size 1024
```

2. 특정 풀의 Ceph 블록 장치 이미지의 최대 크기를 늘리거나 줄이려면 다음을 수행합니다.

구문

```
rbd resize --image POOL_NAME/IMAGE_NAME --size SIZE
```

예제

```
[root@rbd-client ~]# rbd resize --image pool1/image1 --size 1024
```

2.8. 블록 장치 이미지 제거

블록 장치 이미지를 제거합니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 기본 **rbd** 풀에서 블록 장치를 제거하려면 다음을 수행합니다.

   구문

   ```
   rbd rm IMAGE_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd rm image1
   ```

2. 특정 풀에서 블록 장치를 제거하려면 다음을 수행합니다.

   구문

   ```
   rbd rm IMAGE_NAME -p POOL_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd rm image1 -p pool1
   ```

2.9. 블록 장치 이미지를 휴지통으로 이동

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

이미지가 휴지통으로 이동되면 나중에 휴지통에서 제거할 수 있습니다. 이는 실수로 삭제되는 것을 방지하는 데 도움이 됩니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 이미지를 휴지통으로 이동하려면 다음을 실행합니다.

   구문

   ```
   rbd trash mv [POOL_NAME/] IMAGE_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd trash mv pool1/image1
   ```

이미지가 휴지통에 있으면 고유 이미지 ID가 할당됩니다.

참고

휴지 옵션을 사용해야 하는 경우 나중에 이미지를 지정하려면 이 이미지 ID가 필요합니다.

2. **휴지통에 있는 이미지 ID 목록에 대해 rbd 휴지통 *POOL_NAME*** 을 실행합니다. 이 명령은 이미지의 사전 삭제 이름도 반환합니다. 또한 **rbd info 및 rbd snap** 명령과 함께 사용할 수 있는 선택적 **--image-id** 인수가 있습니다. **rbd info** 명령과 함께 **--image-id** 를 사용하여 휴지통의 이미지 속성을 확인하고 **rbd snap** 과 함께 사용하여 휴지통에서 이미지의 스냅샷을 제거합니다.

3. 휴지통에서 이미지를 제거하려면 다음을 실행합니다.

구문

```
rbd trash rm [POOL_NAME/] IMAGE_ID
```

예제

```
[root@rbd-client ~]# rbd trash rm pool1/d35ed01706a0
```

중요

휴지통에서 이미지를 제거한 후에는 복원할 수 없습니다.

4. 이미지를 **복원하려면 rbd 휴지통 복원** 명령을 실행합니다.

구문

```
rbd trash restore [POOL_NAME/] IMAGE_ID
```

예제

```
[root@rbd-client ~]# rbd trash restore pool1/d35ed01706a0
```

5. 휴지통에서 만료된 모든 이미지를 제거하려면 다음을 수행합니다.

구문

```
rbd trash purge POOL_NAME
```

예제

```
[root@rbd-client ~]# rbd trash purge pool1
Removing images: 100% complete...done.
```

2.10. 자동 삭제 일정 정의

풀에서 정기적인 휴지 삭제 작업을 예약할 수 있습니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 휴지 제거 일정을 추가하려면 다음을 실행합니다.

   구문

   ```
   rbd trash purge schedule add --pool POOL_NAME INTERVAL
   ```

   예제

   ```
   [ceph: root@host01 /]# rbd trash purge schedule add --pool pool1 10m
   ```

2. 휴지 제거 일정을 나열하려면 다음을 실행합니다.

   구문

   ```
   rbd trash purge schedule ls --pool POOL_NAME
   ```

   예제

   ```
   [ceph: root@host01 /]# rbd trash purge schedule ls --pool pool1
   every 10m
   ```

3. 휴지 제거 일정의 상태를 확인하려면 다음을 실행합니다.

   예제

   ```
   [ceph: root@host01 /]# rbd trash purge schedule status
   POOL   NAMESPACE   SCHEDULE  TIME
   pool1              2021-08-02 11:50:00
   ```

4. 휴지 제거 일정을 제거하려면 다음을 실행합니다.

   구문

   ```
   rbd trash purge schedule remove --pool POOL_NAME INTERVAL
   ```

   예제

   ```
   [ceph: root@host01 /]# rbd trash purge schedule remove --pool pool1 10m
   ```

2.11. 이미지 기능 활성화 및 비활성화

fast-diff,exclusive- lock,object- map 또는 deep- flatten 과 같은 블록 장치 이미지는 기본적으로 활성화되어 있습니다. 기존 이미지에서 이러한 이미지 기능을 활성화하거나 비활성화할 수 있습니다.

참고

심층 플랫 기능은 기존 이미지에서만 비활성화할 수 있지만 활성화되지는 않습니다. 심층 **플랫** 을 사용하려면 이미지를 만들 때 활성화합니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 풀의 특정 이미지에서 정보를 검색합니다.

   구문

   ```
   rbd --image POOL_NAME/IMAGE_NAME info
   ```

   예제

   ```
   [ceph: root@host01 /]# rbd --image pool1/image1 info
   ```

2. 기능을 활성화합니다.

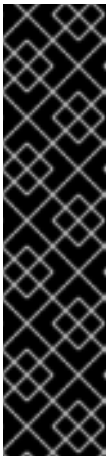
   구문

   ```
   rbd feature enable POOL_NAME/IMAGE_NAME FEATURE_NAME
   ```

   a. pool1 **풀의 image1 이미지에서 배타적 잠금** 기능을 활성화하려면 다음을 수행합니다.

      예제

      ```
      [ceph: root@host01 /]# rbd feature enable pool1/image1 exclusive-lock
      ```



중요

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

+ .Syntax

```
rbd object-map rebuild POOL_NAME/IMAGE_NAME
```

3. 기능을 비활성화합니다.

구문

```
rbd feature disable POOL_NAME/IMAGE_NAME FEATURE_NAME
```

a. **pool1** 풀의 **image1** 이미지에서 **fast-diff** 기능을 비활성화하려면 다음을 수행합니다.

예제

```
[ceph: root@host01 /]# rbd feature disable pool1/image1 fast-diff
```

2.12. 이미지 메타데이터 작업

Ceph에서는 사용자 지정 이미지 메타데이터를 키-값 쌍으로 추가할 수 있습니다. 쌍에는 엄격한 형식이 없습니다.

또한 메타데이터를 사용하면 특정 이미지에 대한 **RADOS** 블록 장치(**RBD**) 구성 매개변수를 설정할 수 있습니다.

rbd image-meta 명령을 사용하여 메타데이터로 작업합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 새 메타데이터 키-값 쌍을 설정하려면 다음을 수행합니다.

구문

```
rbd image-meta set POOL_NAME/IMAGE_NAME KEY VALUE
```

예제

```
[ceph: root@host01 /]# rbd image-meta set pool1/image1 last_update 2021-06-06
```

이 예제에서는 **pool1** 풀의 **image1** 이미지의 **2021-06-06** 값으로 **last_update** 키를 설정합니다.

2. 키 값을 보려면 다음을 수행합니다.

구문

```
rbd image-meta get POOL_NAME/IMAGE_NAME KEY
```

예제

```
[ceph: root@host01 /]# rbd image-meta get pool1/image1 last_update
```

이 예제에서는 last_update 키 값을 봅니다.

3. **이미지의 모든 메타데이터를 표시하려면 다음을 수행합니다.**

구문

```
rbd image-meta list POOL_NAME/IMAGE_NAME
```

예제

```
[ceph: root@host01 /]# rbd image-meta list pool1/image1
```

이 예에서는 pool1 풀의 image1 이미지에 설정된 메타데이터를 나열합니다.

4. **메타데이터 키-값 쌍을 제거하려면 다음을 수행합니다.**

구문

```
rbd image-meta remove POOL_NAME/IMAGE_NAME KEY
```

예제

```
[ceph: root@host01 /]# rbd image-meta remove pool1/image1 last_update
```

이 예제에서는 pool1 풀의 image1 이미지에서 last_update 키-값 쌍을 제거합니다.

5. **특정 이미지의 Ceph 구성 파일에 설정된 RBD 이미지 구성 설정을 재정의하려면 다음을 수행합니다.**

구문

```
rbd config image set POOL_NAME/IMAGE_NAME  PARAMETER VALUE
```

예제

```
[ceph: root@host01 /]# rbd config image set pool1/image1 rbd_cache false
```

이 예제에서는 pool1 풀에서 image1 이미지의 RBD 캐시를 비활성화합니다.

추가 리소스

- 가능한 구성 *옵션 목록은 **Red Hat Ceph Storage** 블록 장치 가이드의 블록 장치 일반* 옵션 섹션을 참조하십시오.

2.13. 풀 간에 이미지 이동

동일한 클러스터 내의 여러 풀 간에 **RADOS** 블록 장치(**RBD**) 이미지를 이동할 수 있습니다.

이 프로세스 중에 소스 이미지는 모든 스냅샷 기록이 포함된 대상 이미지에 복사되고, 선택적으로 스파스성 보존에 도움이 되도록 소스 이미지의 상위 항목에 대한 링크가 제공됩니다. 소스 이미지는 읽기 전용이며 대상 이미지는 쓸 수 있습니다. 마이그레이션이 진행되는 동안 대상 이미지는 소스 이미지에 연결됩니다.

새 대상 이미지가 사용 중인 동안 이 프로세스를 백그라운드로 안전하게 실행할 수 있습니다. 그러나 준비 단계 전에 대상 이미지를 사용하는 모든 클라이언트를 중지하여 새 대상 이미지를 가리키도록 이미지를 사용하는 클라이언트가 업데이트되었는지 확인합니다.

중요

krbd 커널 모듈은 현재 실시간 마이그레이션을 지원하지 않습니다.

사전 요구 사항

- 소스 이미지를 사용하는 모든 클라이언트를 중지합니다.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. 소스 및 대상 이미지를 교차 링크하는 새 대상 이미지를 생성하여 마이그레이션을 준비합니다.

   구문

```
rbd migration prepare SOURCE_IMAGE TARGET_IMAGE
```

교체:

- 이동할 이미지의 이름이 *SOURCE_IMAGE* 입니다. *POOL*/*IMAGE_NAME* 형식을 사용합니다.
- 새 이미지의 이름으로 *TARGET_IMAGE*. *POOL*/*IMAGE_NAME* 형식을 사용합니다.

예제

```
[root@rbd-client ~]# rbd migration prepare pool1/image1 pool2/image2
```

2. 준비해야 하는 새 대상 이미지의 상태를 확인합니다.

구문

```
rbd status TARGET_IMAGE
```

예제

```
[root@rbd-client ~]# rbd status pool2/image2
Watchers: none
Migration:
        source: pool1/image1 (5e2cba2f62e)
        destination: pool2/image2 (5e2ed95ed806)
        state: prepared
```

3. 선택적으로 새 대상 이미지 이름을 사용하여 클라이언트를 다시 시작합니다.

4. 소스 이미지를 대상 이미지에 복사합니다.

구문

```
rbd migration execute TARGET_IMAGE
```

예제

```
[root@rbd-client ~]# rbd migration execute pool2/image2
```

5. 마이그레이션이 완료되었는지 확인합니다.

예제

```
[root@rbd-client ~]# rbd status pool2/image2
Watchers:
   watcher=1.2.3.4:0/3695551461 client.123 cookie=123
Migration:
        source: pool1/image1 (5e2cba2f62e)
        destination: pool2/image2 (5e2ed95ed806)
        state: executed
```

6.

소스 이미지와 대상 이미지 간에 교차 링크를 제거하여 마이그레이션을 커밋하고 소스 이미지도 제거합니다.

구문

```
rbd migration commit TARGET_IMAGE
```

예제

```
[root@rbd-client ~]# rbd migration commit pool2/image2
```

소스 이미지가 하나 이상의 복제본의 상위인 경우 복제 이미지가 사용되지 않도록 한 후 **--force** 옵션을 사용합니다.

예제

```
[root@rbd-client ~]# rbd migration commit pool2/image2 --force
```

7. 준비 단계 후 클라이언트를 재시작하지 않은 경우 새 대상 이미지 이름을 사용하여 클라이언트를 다시 시작합니다.

2.14. RBDMAP 서비스

systemd 유닛 파일인 **rbdmap.service** 는 **ceph-common** 패키지에 포함되어 있습니다. **rbdmap.service** 장치는 **rbdmap** 쉘 스크립트를 실행합니다.

이 스크립트는 하나 이상의 **RBD** 이미지에 대한 **RBD(RADOS** 블록 장치) 매핑 및 매핑 해제를 자동화합니다. 스크립트는 언제든지 수동으로 실행할 수 있지만 일반적인 사용 사례는 부팅 시 **RBD** 이미지를 자동으로 마운트하고 종료 시 를 마운트 해제하는 것입니다. 이 스크립트는 **RBD** 이미지의 마운트를 해제하기 위해 매핑 하거나 매핑 해제 할 수 있는 단일 인수를 사용합니다. 스크립트는 구성 파일을 구문 분석합니다. 기본값은 **/etc/ceph/rbdmap** 이지만 **RBDMAPFILE** 이라는 환경 변수를 사용하여 재정의할 수 있습니다. 구성 파일의 각 행은 **RBD** 이미지에 해당합니다.

구성 파일 형식의 형식은 다음과 같습니다.

IMAGE_SPEC RBD_OPTS

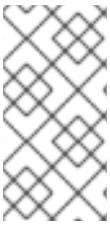
여기서 ***IMAGE_SPEC*** 는 ***POOL_NAME*** / ***IMAGE_NAME*** 또는 ***IMAGE_NAME*** 만 지정합니다. 이 경우 ***POOL_NAME*** 의 기본값은 **rbd** 입니다. ***RBD_OPTS*** 는 기본 **rbd map** 명령으로 전달할 옵션 목록입니다. 이러한 매개변수와 해당 값은 쉼표로 구분된 문자열로 지정해야 합니다.

OPT1=VAL1,OPT2=VAL2,...,OPT_N=VAL_N

이렇게 하면 스크립트에서 다음과 같이 **rbd map** 명령을 실행합니다.

구문

```
rbd map POOLNAME/IMAGE_NAME --OPT1 VAL1 --OPT2 VAL2
```



참고

쉼표 또는 등가 기호가 포함된 옵션과 값의 경우 간단한 **apostrophe**를 사용하여 교체하지 않도록 할 수 있습니다.

성공하면 **rbd** 맵 작업에서 이미지를 **/dev/rbdX** 장치에 매핑합니다. 이 장치에서 **udev** 규칙이 트리거되어 친숙한 장치 이름 **symlink(**예: **/dev/rbd/*POOL_NAME*/*IMAGE_NAME*)**가 실제 매핑된 장치를 가리킵니다. 마운트 또는 마운트 해제가 성공하려면 친숙한 장치 이름에 **/etc/fstab** 파일에 해당 항목이 있어야 합니다. **RBD** 이미지에 대한 **/etc/fstab** 항목을 작성할 때 **noauto** 또는 **no fail** 마운트 옵션을 지정하는 것이 좋습니다. 이렇게 하면 **init** 시스템이 장치가 존재하기 전에 장치를 너무 빨리 마운트하지 못하게 합니다.

추가 리소스

- 가능한 옵션의 전체 목록은 **rbd** 도움말 페이지를 참조하십시오.

2.15. RBDMAP 서비스 구성

부팅 시 또는 각각 종료 시 **RADOS** 블록 장치(**RBD**)를 자동으로 매핑 및 마운트 해제합니다.

사전 요구 사항

- 마운트를 수행하는 노드에 대한 루트 수준 액세스입니다.
- **ceph-common** 패키지 설치.

절차

1. **/etc/ceph/rbdmap** 구성 파일을 편집하기 위해 를 엽니다.
2. **RBD** 이미지 또는 이미지를 구성 파일에 추가합니다.

예제

```
foo/bar1    id=admin,keyring=/etc/ceph/ceph.client.admin.keyring
foo/bar2
id=admin,keyring=/etc/ceph/ceph.client.admin.keyring,options='lock_on_read,queue_depth=102
4'
```

3. 구성 파일에 변경 사항을 저장합니다.
4. **RBD** 매핑 서비스를 활성화합니다.

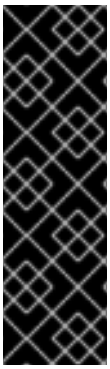
예제

```
[root@client ~]# systemctl enable rbdmap.service
```

추가 리소스

- RBD 시스템 *서비스에 대한 자세한 내용은 Red Hat Ceph Storage 블록 장치 가이드 의 rbdmap* 서비스 섹션을 참조하십시오.

2.16. 영구 쓰기 로그 캐시



중요

캐시 장치로 SSD를 사용하는 PWL(영구 쓰기 로그)은 기술 프리뷰 기능 전용입니다. 기술 프리뷰 기능은 Red Hat 프로덕션 서비스 수준 계약(SLA)에서 지원하지 않으며, 기능상 완전하지 않을 수 있어 프로덕션에 사용하지 않는 것이 좋습니다. 이러한 기능을 사용하면 향후 제품 기능을 조기에 이용할 수 있어 개발 과정에서 고객이 기능을 테스트하고 피드백을 제공할 수 있습니다. 자세한 내용은 Red Hat 기술 프리뷰 기능에 대한 지원 범위를 참조하십시오.

Red Hat Ceph Storage 클러스터에서 PDWL(Persistent Write Log) 캐시는 librbd 기반 RBD 클라이언트에 대해 영구적인 내결함성 쓰기 캐시를 제공합니다.

PWL 캐시는 로그 주문 쓰기백 설계를 사용하여 내부적으로 검사점을 유지 관리하여 클러스터에 플러시되는 쓰기가 항상 일관되게 충돌합니다. 클라이언트 캐시가 완전히 손실되면 디스크 이미지가 계속 일관되지만 데이터가 부실하게 표시됩니다. PWL 캐시는 PDM(영구 메모리) 또는 SSD(솔리드 스테이트 디스크)와 함께 캐시 장치로 사용할 수 있습니다.

PMEM의 경우 캐시 모드는 복제본 쓰기 로그(RWL)이고 SSD의 경우 캐시 모드는 (SSD)입니다. 현재 PWL 캐시는 RWL 및 SSD 모드를 지원하며 기본적으로 비활성화되어 있습니다.

PWL 캐시의 주요 장점은 다음과 같습니다.

- PWL 캐시는 캐시가 가득 차 있지 않을 때 높은 성능을 제공할 수 있습니다. 캐시가 클수록 성능이 길어집니다.

PWL 캐시는 지속성을 제공하며 RBD 캐시보다 훨씬 느리지 않습니다. RBD 캐시는 더 빠르지만 휘발성이며 데이터 순서 및 지속성을 보장할 수 없습니다.

- 캐시가 가득 차 있는 steady 상태에서 성능은 비행 중인 I/O 수의 영향을 받습니다. 예를 들어, PWL은 낮은 io_depth에서 더 높은 성능을 제공할 수 있지만 I/O 수가 32보다 클 때와 같이 높은 io_depth로 캐시가 없는 경우 성능이 저하되는 경우가 많습니다.

PMEM 캐싱 사용 사례는 다음과 같습니다.

- RBD 캐시와 다르면 PWL 캐시에는 비휘발성 특성이 있으며 데이터 손실을 원하지 않고 성능이 필요한 시나리오에서 사용됩니다.

- RWL 모드는 짧은 대기 시간을 제공합니다. 버스트 I/O에 대한 안정적인 짧은 대기 시간을 가지며 안정적인 짧은 대기 시간에 대한 높은 요구사항이 있는 시나리오에 적합합니다.

- RWL 모드는 또한 낮은 I/O 깊이가 있거나 너무 많은 inflight I/O가 없는 시나리오에서 높은 연속적이고 안정적인 성능 개선을 제공합니다.

SSD 캐싱 사용 사례는 다음과 같습니다.

- SSD 모드의 장점은 RWL 모드와 유사합니다. SSD 하드웨어는 상대적으로 저렴하고 인기가 있지만 성능은 PMEM보다 약간 낮습니다.

2.17. 영구 쓰기 로그 캐시 제한 사항

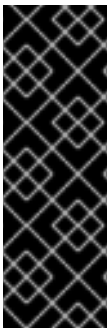
PWL(영구 쓰기 로그) 캐시를 사용하는 경우 고려해야 할 몇 가지 제한 사항이 있습니다.

- PMEM(영구 메모리) 및 SSD(Solid-state disks)의 기본 구현은 PMEM이 성능을 향상시킵니다. 현재 PMEM은 "쓰기 시"를 제공할 수 있으며 SSD는 "플러쉬 또는 검사점"입니다. 향후 릴리스에서는 이러한 두 가지 모드를 구성할 수 있습니다.

- 사용자가 자주적이고 열린 이미지를 반복해서 전환하면 Ceph에서 성능이 저하됩니다. PWL 캐시가 활성화된 경우 성능이 저하됩니다. flexible I/O(fio) 테스트에서 num_jobs 를 설정하지 않는 대신 다른 이미지를 작성하도록 여러 작업을 설정합니다.

2.18. 영구 쓰기 로그 캐시 활성화

Ceph RADOS 블록 장치(RBD) rbd_persistent_cache_mode 및 rbd_plugins 옵션을 설정하여 Red Hat Ceph Storage 클러스터에서 PDL(영구 쓰기 캐시)을 활성화할 수 있습니다.



중요

영구 쓰기 로그 캐시를 활성화하려면 **exclusive-lock** 기능을 활성화해야 합니다. 캐시는 배타적 잠금을 가져온 후에만 로드할 수 있습니다. **rbd_default_features** 구성 옵션 또는 **rbd create** 명령의 **--image-feature** 플래그로 재정의하지 않는 한 기본적으로 새로 생성된 이미지에서 **exclusive-locks**가 활성화됩니다. 독점 잠금 기능에 대한 자세한 내용은 *이미지 기능 활성화 및 비활성화 섹션을* 참조하십시오.

ceph config set 명령을 사용하여 호스트 수준에서 영구 쓰기 로그 캐시 옵션을 설정합니다. 풀 또는 이미지 수준에서 영구 쓰기 로그 캐시 옵션을 설정합니다. **rbd** 구성 풀 세트 또는 **rbd** 구성 이미지 세트 명령을 사용합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 모니터 노드에 대한 루트 수준 액세스입니다.
- **exclusive-lock** 기능을 사용할 수 있습니다.
- 클라이언트 측 디스크는 **PDM(영구 메모리)** 또는 **SSD(Solid-state disks)**입니다.
- **RBD** 캐시가 비활성화되어 있습니다.

절차

1. **PWL** 캐시를 활성화합니다.

   a. 호스트 수준에서 **ceph config set** 명령을 사용합니다.

   구문

```
ceph config set client rbd_persistent_cache_mode CACHE_MODE
ceph config set client rbd_plugins pwl_cache
```

CACHE_MODE **를 rwl 또는 ssd 로 바꿉니다.**

예제

```
[ceph: root@host01 /]# ceph config set client rbd_persistent_cache_mode ssd
[ceph: root@host01 /]# ceph config set client rbd_plugins pwl_cache
```

b.

풀 수준에서 rbd 구성 풀 세트 명령을 사용합니다.

구문

```
rbd config pool set POOL_NAME rbd_persistent_cache_mode CACHE_MODE
rbd config pool set POOL_NAME rbd_plugins pwl_cache
```

CACHE_MODE **를 rwl 또는 ssd 로 바꿉니다.**

예제

```
[ceph: root@host01 /]# rbd config pool set pool1 rbd_persistent_cache_mode ssd
[ceph: root@host01 /]# rbd config pool set pool1 rbd_plugins pwl_cache
```

c.

이미지 수준에서 rbd config image set 명령을 사용합니다.

구문

```
rbd config image set image set POOL_NAME/IMAGE_NAME
rbd_persistent_cache_mode CACHE_MODE
rbd config image set image set POOL_NAME/IMAGE_NAME rbd_plugins pwl_cache
```

CACHE_MODE **를 rwl 또는 ssd 로 바꿉니다.**

예제

```
[ceph: root@host01 /]# rbd config image set pool1/image1 rbd_persistent_cache_mode
ssd
[ceph: root@host01 /]# rbd config image set pool1/image1 rbd_plugins pwl_cache
```

2.

선택 사항: 호스트, 풀 또는 이미지 수준에서 RBD 옵션을 추가로 설정합니다.

구문

```
rbd_persistent_cache_mode CACHE_MODE
rbd_plugins pwl_cache
rbd_persistent_cache_path
/PATH_TO_DAX_ENABLED_FOLDER/WRITE_BACK_CACHE_FOLDER 1
rbd_persistent_cache_size PERSISTENT_CACHE_SIZE 2
```

1

2

rbd_persistent_cache_size - 최소 캐시 크기가 **1GB**인 이미지당 캐시 크기입니다. 캐시 크기가 클수록 성능이 향상됩니다.

예제

```
rbd_cache false
rbd_persistent_cache_mode rwl
rbd_plugins pwl_cache
rbd_persistent_cache_path /mnt/pmem/cache/
rbd_persistent_cache_size 1073741824
```

추가 리소스

- **DAX** 사용에 *대한 자세한 내용은 **kernel.org** 의 파일 문서에 대한 직접* 액세스를 참조하십시오.

2.19. 영구 쓰기 로그 캐시 상태 확인

영구 쓰기 로그(PWL) 캐시의 상태를 확인할 수 있습니다. 캐시는 배타적 잠금을 가져올 때 사용되며 배타적 잠금이 해제되면 영구 쓰기 로그 캐시가 닫힙니다. 캐시 상태는 캐시 크기, 위치, 유형 및 기타 캐시 관련 정보에 대한 정보를 표시합니다. 캐시 상태 업데이트는 캐시가 열리고 닫히면 수행됩니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 모니터 노드에 대한 루트 수준 액세스입니다.
- **PWL** 캐시가 활성화된 실행 중인 프로세스.

절차

- **PWL 캐시 상태 보기:**

구문

```
rbd status POOL_NAME/IMAGE_NAME
```

예제

```
[ceph: root@host01 /]# rbd status pool1/image1
Watchers:
        watcher=10.10.0.102:0/1061883624 client.25496 cookie=140338056493088
Persistent cache state:
        host: host02
        path: /mnt/nvme0/rbd-pwl.rbd.101e5824ad9a.pool
        size: 1 GiB
        mode: ssd
        stats_timestamp: Mon Apr 18 13:26:32 2022
        present: true    empty: false    clean: false
        allocated: 509 MiB
        cached: 501 MiB
        dirty: 338 MiB
        free: 515 MiB
        hits_full: 1450 / 61%
        hits_partial: 0 / 0%
        misses: 924
        hit_bytes: 192 MiB / 66%
        miss_bytes: 97 MiB
```

2.20. 영구 쓰기 로그 캐시 플러시

영구 쓰기 로그(PWL) 캐시를 삭제하기 전에 영구 캐시 플러시, 풀 이름 및 이미지 이름을 지정하여 캐시 파일을 **rbd** 명령으로 플러시할 수 있습니다. **flush** 명령은 캐시 파일을 **OSD**에 명시적으로 다시 쓸 수 있습니다. 캐시 중단이 있거나 애플리케이션이 예기치 않게 종료되면 캐시의 모든 항목이 **OSD**로 플러시되므로 데이터를 수동으로 플러시한 다음 캐시가 무효화 됩니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 모니터 노드에 대한 루트 수준 액세스입니다.
- **PWL** 캐시가 활성화되어 있습니다.

절차

- **PWL** 캐시를 플러시합니다.

  구문

```
rbd persistent-cache flush POOL_NAME/IMAGE_NAME
```

  예제

```
[ceph: root@host01 /]# rbd persistent-cache flush pool1/image1
```

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage Block Device Guide*** 의 *영구 쓰기 로그 캐시* 비활성화 섹션을 참조하십시오.

2.21. 영구 쓰기 로그 캐시 삭제

캐시의 데이터가 만료된 경우 영구 쓰기 로그(**PWL**) 캐시를 수동으로 삭제해야 할 수 있습니다. **rbd persistent-cache invalidate** 명령을 사용하여 이미지의 캐시 파일을 삭제할 수 있습니다. 명령은 지정된 이미지의 캐시 메타데이터를 제거하고 캐시 기능을 비활성화한 다음 로컬 캐시 파일이 있는 경우 삭제합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 모니터 노드에 대한 루트 수준 액세스입니다.
- **PWL** 캐시가 활성화되어 있습니다.

절차

- **PWL** 캐시 삭제:

구문

```
rbd persistent-cache invalidate POOL_NAME/IMAGE_NAME
```

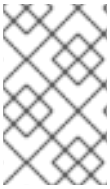
예제

```
[ceph: root@host01 /]# rbd persistent-cache invalidate pool1/image1
```

2.22. 명령줄 인터페이스를 사용하여 CEPH 블록 장치의 성능 모니터링

Red Hat Ceph Storage 4.1부터는 **Ceph OSD** 및 **Manager** 구성 요소 내에 성능 지표 수집 프레임워크가 통합됩니다. 이 프레임워크는 다른 **Ceph** 블록 장치 성능 모니터링 솔루션이 구축되는 성능 지표를 생성하고 처리할 수 있는 기본 방법을 제공합니다.

활성화된 경우 새 **Ceph Manager** 모듈인**rbd_support** 는 성능 지표를 집계합니다. **rbd** 명령에는 두 개의 새 작업(**iotop** 및 **iostat**)이 있습니다.



참고

이러한 작업을 처음 사용하는 데 데이터 필드를 채우는 데 약 30초가 걸릴 수 있습니다.

사전 요구 사항

- Ceph 모니터 노드에 대한 사용자 수준 액세스.

절차

1. rbd_support Ceph Manager 모듈이 활성화되어 있는지 확인합니다.

예제

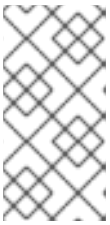
```
[ceph: root@host01 /]# ceph mgr module ls

{
       "always_on_modules": [
         "balancer",
          "crash",
          "devicehealth",
          "orchestrator",
          "pg_autoscaler",
          "progress",
          "rbd_support",   <--
          "status",
          "telemetry",
          "volumes"
}
```

2. "iotop" 이미지 스타일을 표시하려면 다음을 수행합니다.

예제

```
[user@mon ~]$ rbd perf image iotop
```



참고

오른쪽 및 왼쪽 화살표 키를 사용하여 쓰기 **ops, read-ops, write-bytes, read-latency** 및 **read-latency** 열을 동적으로 정렬할 수 있습니다.

3. **"iostat"** 이미지 스타일을 표시하려면 다음을 수행합니다.

예제

```
[user@mon ~]$ rbd perf image iostat
```

참고

이 명령의 출력은 **JSON** 또는 **XML** 형식일 수 있으며 다른 명령줄 툴을 사용하여 정렬할 수 있습니다.

2.23. 추가 리소스

- 매핑 및 매핑 해제에 대한 자세한 내용은 *8장. rbd 커널 모듈* 을 참조하십시오.

3장. 이미지 실시간 마이그레이션

스토리지 관리자는 스토리지 클러스터 내에서 여러 풀 또는 동일한 풀에서 **RBD** 이미지를 실시간 마이그레이션할 수 있습니다. 다양한 이미지 형식과 레이아웃 간에 마이그레이션할 수 있으며 외부 데이터 소스에서도 마이그레이션할 수 있습니다. 실시간 마이그레이션이 시작되면 소스 이미지가 대상 이미지에 깊이 복사되고 모든 스냅샷 기록을 가져와 가능한 데이터의 스파스 할당을 유지합니다.

중요

현재 **krbd** 커널 모듈은 실시간 마이그레이션을 지원하지 않습니다.

3.1. 사전 요구 사항

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

3.2. 실시간 마이그레이션 프로세스

기본적으로 동일한 스토리지 클러스터를 사용하는 **RBD** 이미지를 실시간 마이그레이션하는 동안 소스 이미지는 읽기 전용으로 표시됩니다. 모든 클라이언트는 **Input/Output(I/O)**을 새 대상 이미지로 리디렉션합니다. 또한 이 모드에서는 소스 이미지의 상위 항목에 대한 링크를 유지하여 스파스성을 보존하거나 마이그레이션 중에 이미지를 병합하여 소스 이미지의 상위 항목에 대한 종속성을 제거할 수 있습니다. 소스 이미지가 변경되지 않은 가져오기 전용 모드에서 실시간 마이그레이션 프로세스를 사용할 수 있습니다. 백업 파일, **HTTP** 파일 또는 **S3** 오브젝트와 같은 외부 데이터 소스에 대상 이미지를 연결할 수 있습니다. 새 대상 이미지를 사용하는 동안 실시간 마이그레이션 복사 프로세스를 백그라운드에서 안전하게 실행할 수 있습니다.

실시간 마이그레이션 프로세스는 다음 세 단계로 구성됩니다.

마이그레이션 준비: 첫 번째 단계는 새 대상 이미지를 생성하고 대상 이미지를 소스 이미지에 연결하는 것입니다. 가져오기 전용 모드가 구성되지 않은 경우 소스 이미지도 대상 이미지에 연결되고 읽기 전용으로 표시됩니다. 대상 이미지 내에서 초기화되지 않은 데이터 익스텐트를 읽으려는 시도는 내부적으로 소스 이미지로 읽기를 리디렉션하고, 대상 이미지 내의 초기화되지 않은 확장 영역에 쓰기는 내부적으로 복사되고, 중복되는 소스 이미지 확장 영역이 대상 이미지에 포함됩니다.

마이그레이션 실행: 이 작업은 소스 이미지에서 대상으로 초기화된 모든 블록을 심층적으로 복사하는 백그라운드 작업입니다. 클라이언트가 새 대상 이미지를 적극적으로 사용하는 경우 이 단계를 실행할 수 있습니다.

마이그레이션 완료: 백그라운드 마이그레이션 프로세스가 완료되면 마이그레이션을 커밋하거나 중단

할 수 있습니다. 마이그레이션을 커밋하면 소스 이미지와 대상 이미지 간의 교차 링크가 제거되고 가져오기 전용 모드에 구성되지 않은 경우 소스 이미지가 제거됩니다. 마이그레이션을 중단하면 교차 링크가 제거되고 대상 이미지가 제거됩니다.

3.3. 형식

기본 형식을 사용하여 **Red Hat Ceph Storage** 클러스터 내의 기본 **RBD** 이미지를 소스 이미지로 설명할 수 있습니다. **source-spec JSON** 문서는 다음과 같이 인코딩됩니다.

구문

```
{
    "type": "native",
    "pool_name": "POOL_NAME",
    ["pool_id": "POOL_ID",] (optional, alternative to "POOL_NAME" key)
    ["pool_namespace": "POOL_NAMESPACE",] (optional)
    "image_name": "IMAGE_NAME>",
    ["image_id": "IMAGE_ID",] (optional, useful if image is in trash)
    "snap_name": "SNAP_NAME",
    ["snap_id": "SNAP_ID",] (optional, alternative to "SNAP_NAME" key)
}
```

기본 형식에는 기본 **Ceph** 작업을 활용하므로 스트림 오브젝트가 포함되지 않습니다. 예를 들어 이미지 **rbd/ns1/image1@snap1** 에서 가져오려면 **source-spec** 을 다음과 같이 인코딩할 수 있습니다.

예제

```
{
"type": "native",
"pool_name": "rbd",
"pool_namespace": "ns1",
"image_name": "image1",
"snap_name": "snap1"
}
```

qcow 형식을 사용하여 **QEMU COW(QCOW)** 블록 장치를 설명할 수 있습니다. **QCOW v1** 및 **v2** 형식은 현재 압축, 암호화, 백업 파일, 외부 데이터 파일 등의 고급 기능을 제외하고 현재 지원됩니다. 지원되는 모든 스트림 소스에 **qcow** 형식 데이터를 연결할 수 있습니다.

예제

```
{
    "type": "qcow",
    "stream": {
      "type": "file",
   "file_path": "/mnt/image.qcow"
    }
}
```

원시 형식을 사용하여 **rbd** 내보내기 **-export-format 1** ***SNAP_SPEC***인 씩 프로비저닝된 원시 블록 장치 내보내기 를 설명할 수 있습니다. 지원되는 모든 스트림 소스에 원시 형식 데이터를 연결할 수 있습니다.

예제

```
{
    "type": "raw",
    "stream": {
      "type": "file",
      "file_path": "/mnt/image-head.raw"
    },
    "snapshots": [
        {
            "type": "raw",
            "name": "snap1",
            "stream": {
              "type": "file",
       "file_path": "/mnt/image-snap1.raw"
            }
        },
    ] (optional oldest to newest ordering of snapshots)
}
```

snapshot 배열 의 포함은 선택 사항이며 현재 씩 프로비저닝된 원시 스냅샷 내보내기만 지원합니다.

3.4. 스트림

파일 스트림

파일 스트림을 사용하여 로컬에서 액세스할 수 있는 **POSIX** 파일 소스에서 가져올 수 있습니다.

구문

```
{
    <format unique parameters>
    "stream": {
        "type": "file",
        "file_path": "FILE_PATH"
    }
}
```

예를 들어 **/mnt/image.raw**에 있는 파일에서 원시 형식 이미지를 가져오려면 **source-spec JSON** 파일은 다음과 같습니다.

예제

```
{
    "type": "raw",
    "stream": {
        "type": "file",
        "file_path": "/mnt/image.raw"
    }
}
```

HTTP 스트림

HTTP 스트림을 사용하여 원격 **HTTP** 또는 **HTTPS** 웹 서버에서 가져올 수 있습니다.

구문

```
{
    <format unique parameters>
    "stream": {
        "type": "http",
        "url": "URL_PATH"
    }
}
```

예를 들어, **http://download.ceph.com/image.raw** 에 있는 파일에서 원시 형식 이미지를 가져오려면 **source-spec JSON** 파일은 다음과 같습니다.

예제

```
{
    "type": "raw",
    "stream": {
        "type": "http",
        "url": "http://download.ceph.com/image.raw"
    }
}
```

S3 스트림

s3 스트림을 사용하여 원격 **S3** 버킷에서 가져올 수 있습니다.

구문

```
{
    <format unique parameters>
    "stream": {
```

```
        "type": "s3",
        "url": "URL_PATH",
        "access_key": "ACCESS_KEY",
        "secret_key": "SECRET_KEY"
    }
}
```

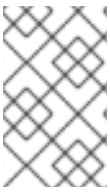
예를 들어 **http://s3.ceph.com/bucket/image.raw** 에 있는 파일에서 원시 형식 이미지를 가져오려면 **source-spec JSON**은 다음과 같이 인코딩됩니다.

예제

```
{
    "type": "raw",
    "stream": {
        "type": "s3",
        "url": "http://s3.ceph.com/bucket/image.raw",
        "access_key": "NX5QOQKC6BH2IDN8HC7A",
        "secret_key": "LnEsqNNqZIpkzauboDcLXLcYaWwLQ3Kop0zAnKIn"
    }
}
```

3.5. 실시간 마이그레이션 프로세스 준비

동일한 **Red Hat Ceph Storage** 클러스터 내에서 **RBD** 이미지에 대한 기본 실시간 마이그레이션 프로세스를 준비할 수 있습니다. **rbd migration prepare** 명령은 **rbd create** 명령과 동일한 레이아웃 옵션을 모두 허용합니다. **rbd create** 명령을 사용하면 변경 불가능한 이미지의 디스크 레이아웃을 변경할 수 있습니다. 디스크에 있는 레이아웃만 변경하고 원래 이미지 이름을 유지하려면 **migration_target** 인수를 건너뜁니다. 실시간 마이그레이션을 준비하기 전에 소스 이미지를 사용하는 모든 클라이언트를 중지해야 합니다. 읽기/쓰기 모드로 열린 이미지가 있는 실행 중인 클라이언트를 발견하면 **prepare** 단계가 실패합니다. 준비 단계가 완료되면 새 대상 이미지를 사용하여 클라이언트를 다시 시작할 수 있습니다.



참고

소스 이미지를 사용하여 클라이언트를 재시작할 수 없으므로 실패합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 블록 장치 풀 두 개.
- 블록 장치 이미지 **1**개.

절차

1. 스토리지 클러스터 내에서 실시간 마이그레이션을 준비합니다.

구문

```
rbd migration prepare SOURCE_POOL_NAME/SOURCE_IMAGE_NAME
TARGET_POOL_NAME/SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd migration prepare sourcepool1/sourceimage1
targetpool1/sourceimage1
```

OR

소스 이미지의 이름을 바꾸려면 다음을 수행합니다.

구문

```
rbd migration prepare SOURCE_POOL_NAME/SOURCE_IMAGE_NAME
TARGET_POOL_NAME/NEW_SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd migration prepare sourcepool1/sourceimage1
targetpool1/newsourceimage1
```

이 예제에서 newsourceimage1 은 이름이 변경된 소스 이미지입니다.

2. **다음 명령을 사용하여 실시간 마이그레이션 프로세스의 현재 상태를 확인할 수 있습니다.**

구문

```
rbd status TARGET_POOL_NAME/SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd status targetpool1/sourceimage1
Watchers: none
Migration:
source: sourcepool1/sourceimage1 (adb429cb769a)
destination: targetpool2/testimage1 (add299966c63)
state: prepared
```

중요

마이그레이션 프로세스 중에 잘못된 사용량을 방지하기 위해 소스 이미지가 **RBD** 휴지통으로 이동됩니다.

예제

```
[ceph: root@rbd-client /]# rbd info sourceimage1
rbd: error opening image sourceimage1: (2) No such file or directory
```

예제

```
[ceph: root@rbd-client /]# rbd trash ls --all sourcepool1
adb429cb769a sourceimage1
```

3.6. 가져오기 전용 마이그레이션 준비

--import -only 및 **--source-spec** 또는 **--source- spec- path** 옵션을 사용하여 **rbd** 마이그레이션 **prepare** 명령을 실행하여 가져오기 전용 실시간 마이그레이션 프로세스를 시작할 수 있으며 명령줄 또는 파일에서 직접 소스 이미지 데이터에 액세스하는 방법을 설명하는 **JSON** 문서를 전달할 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 버킷과 **S3** 오브젝트가 생성됩니다.

절차

1. **JSON** 파일을 생성합니다.

예제

예제

```
[ceph: root@rbd-client /]# cat testspec.json
 {
   "type": "raw",
    "stream": {
        "type": "s3",
        "url": "http:10.74.253.18:80/testbucket1/image.raw",
       "access_key": "RLJOCP6345BGB38YQXI5",
       "secret_key": "oahWRB2ote2rnLy4dojYjDrsvaBADriDDgtSfk6o"
 }
```

2. 가져오기 전용 실시간 마이그레이션 프로세스를 준비합니다.

구문

```
rbd migration prepare --import-only --source-spec-path "JSON_FILE"
TARGET_POOL_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd migration prepare --import-only --source-spec-path
"testspec.json" targetpool1
```

참고

rbd migration prepare 명령은 **rbd create** 명령과 동일한 이미지 옵션을 모두 허용합니다.

3. 가져오기 전용 실시간 마이그레이션의 상태를 확인할 수 있습니다.

예제

```
[ceph: root@rbd-client /]# rbd status targetpool1/sourceimage1
Watchers: none
Migration:
source: {"stream":
{"access_key":"RLJOCP6345BGB38YQXI5","secret_key":"oahWRB2ote2rnLy4dojYjDrsvaBADriDDgtSfk6o","type":"s3","url":"http://10.74.253.18:80/testbucket1/image.raw"},"type":"raw"}
destination: targetpool1/sourceimage1 (b13865345e66)
state: prepared
```

3.7. 실시간 마이그레이션 프로세스 실행

실시간 마이그레이션을 준비한 후 소스 이미지의 이미지 블록을 대상 이미지로 복사해야 합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 블록 장치 풀 두 개.
- 블록 장치 이미지 **1**개.

절차

1. 실시간 마이그레이션을 실행합니다.

구문

```
rbd migration execute TARGET_POOL_NAME/SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd migration execute targetpool1/sourceimage1
Image migration: 100% complete...done.
```

2. **마이그레이션 블록 심층 복사 프로세스의 진행 상태에 대한 피드백을 확인할 수 있습니다.**

구문

```
rbd status TARGET_POOL_NAME/SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd status targetpool1/sourceimage1
Watchers: none
Migration:
source: sourcepool1/testimage1 (adb429cb769a)
destination: targetpool1/testimage1 (add299966c63)
state: executed
```

3.8. 실시간 마이그레이션 프로세스 커밋

실시간 마이그레이션이 소스 이미지에서 대상 이미지로 모든 데이터 블록을 복사한 후 마이그레이션을 커밋할 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 블록 장치 풀 두 개.
- 블록 장치 이미지 **1**개.

절차

1. 마이그레이션을 커밋하면 심도 있는 복사가 완료됩니다.

구문

```
rbd migration commit TARGET_POOL_NAME/SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd migration commit targetpool1/sourceimage1
Commit image migration: 100% complete...done.
```

검증

실시간 마이그레이션을 커밋하면 소스 이미지와 대상 이미지 간에 교차 링크가 제거되고 소스 풀에서 소스 이미지도 제거됩니다.

예제

```
[ceph: root@rbd-client /]# rbd trash list --all sourcepool1
```

3.9. 실시간 마이그레이션 프로세스 중단

실시간 마이그레이션 프로세스를 되돌릴 수 있습니다. 실시간 마이그레이션을 중단하면 준비 및 실행 단계를 되돌립니다.

참고

실시간 마이그레이션을 커밋하지 않은 경우에만 중단할 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 블록 장치 풀 두 개.
- 블록 장치 이미지 **1**개.

절차

1. 실시간 마이그레이션 프로세스를 중단합니다.

구문

```
rbd migration abort TARGET_POOL_NAME/SOURCE_IMAGE_NAME
```

예제

```
[ceph: root@rbd-client /]# rbd migration abort targetpool1/sourceimage1
Abort image migration: 100% complete...done.
```

검증

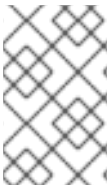
실시간 마이그레이션 프로세스가 중단되면 대상 이미지가 삭제되고 소스 풀에서 원본 소스 이미지에 대한 액세스가 복원됩니다.

예제

```
[ceph: root@rbd-client /]# rbd ls sourcepool1
sourceimage1
```

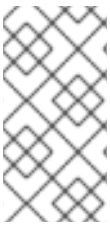
4장. 이미지 암호화

스토리지 관리자는 특정 **RBD** 이미지를 암호화하는 데 사용되는 비밀 키를 설정할 수 있습니다. 이미지 수준 암호화는 **RBD** 클라이언트가 내부적으로 처리합니다.



참고

krbd 모듈은 이미지 수준 암호화를 지원하지 않습니다.



참고

dm-crypt 또는 **QEMU** 와 같은 외부 툴을 사용하여 **RBD** 이미지를 암호화할 수 있습니다.

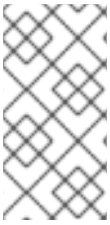
4.1. 사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage 5** 클러스터.
- 루트 수준 권한.

4.2. 암호화 형식

RBD 이미지는 기본적으로 암호화되지 않습니다. 지원되는 암호화 형식 중 하나로 포맷하여 **RBD** 이미지를 암호화할 수 있습니다. 형식 작업은 암호화 메타데이터를 **RBD** 이미지에 유지합니다. 암호화 메타데이터에는 암호화 형식 및 버전, 암호 알고리즘 및 모드 사양 등의 정보와 암호화 키를 보호하는 데 사용되는 정보가 포함됩니다.

암호화 키는 사용자가 암호인 비밀을 유지하며 **RBD** 이미지에 영구 데이터로 저장되지 않습니다. 암호화 형식 작업을 수행하려면 암호화 형식, 암호 알고리즘 및 모드 사양과 암호를 지정해야 합니다. 암호화 메타데이터는 현재 **RBD** 이미지에 저장되며, 현재 원시 이미지 시작 시 작성된 암호화 헤더로 저장됩니다. 즉, 암호화된 이미지의 유효 이미지 크기가 원시 이미지 크기보다 낮아집니다.



참고

현재는 플랫 **RBD** 이미지만 암호화할 수 있습니다. 암호화된 **RBD** 이미지의 복제본은 동일한 암호화 프로필과 암호를 사용하여 본질적으로 암호화됩니다.

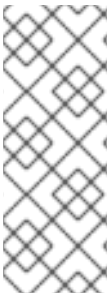
참고

스토리지 리소스를 차지할 수도 있지만 포맷하기 전에 **RBD** 이미지에 작성된 데이터는 읽을 수 없게 될 수 있습니다. 저널 기능이 활성화된 **RBD** 이미지는 암호화할 수 없습니다.

4.3. 암호화 로드

기본적으로 모든 **RBD API**는 암호화된 **RBD** 이미지를 암호화되지 않은 **RBD** 이미지와 동일한 방식으로 처리합니다. 이미지의 모든 위치에서 원시 데이터를 읽거나 쓸 수 있습니다. 원시 데이터를 이미지에 작성하면 암호화 형식의 무결성이 발생할 수 있습니다. 예를 들어 원시 데이터는 이미지 시작 부분에 있는 암호화 메타데이터를 재정의할 수 있습니다. 암호화된 **RBD** 이미지에서 암호화된 입/출력(**I/O**) 또는 유지 관리 작업을 안전하게 수행하려면 이미지를 연 직후 추가 암호화 로드 작업을 적용해야 합니다.

암호화 로드 작업을 수행하려면 암호화 형식과 암호를 지정해야 합니다. 열린 **RBD** 이미지의 모든 **I/O**는 암호화되거나 암호 해독됩니다. 복제된 **RBD** 이미지의 경우 상위 이미지의 **IO**가 포함됩니다. 암호화 키는 이미지가 닫힐 때까지 **RBD** 클라이언트에 의해 메모리에 저장됩니다.



참고

RBD 이미지에 암호화가 로드되면 다른 암호화 로드 또는 형식 작업을 적용할 수 없습니다. 또한 열린 이미지 컨텍스트를 사용하여 **RBD** 이미지 크기를 검색하는 **API** 호출에서 유효한 이미지 크기를 반환합니다. **rbd-nbd** 를 통해 **RBD** 이미지를 블록 장치로 매핑할 때 암호화가 자동으로 로드됩니다.

4.4. 지원되는 형식

Linux 통합 키 설정(**LUKS**) **1** 및 **2**가 모두 지원됩니다. 데이터 레이아웃은 **LUKS** 사양과 완벽하게 호환됩니다. **dm-crypt** 또는 **QEMU** 와 같은 외부 **LUKS** 호환 툴은 암호화된 **RBD** 이미지에서 암호화된 입/출력(**I/O**)을 안전하게 수행할 수 있습니다. 또한 원시 **LUKS** 데이터를 **RBD** 이미지로 복사하여 외부 툴에서 만든 기존 **LUKS** 이미지를 가져올 수 있습니다.

현재 **AES**(**Advanced Encryption Standards**) **128** 및 **256** 암호화 알고리즘만 지원됩니다. **xts-plain64**는 현재 유일하게 지원되는 암호화 모드입니다.

LUKS 형식을 사용하려면 다음 명령으로 **RBD** 이미지를 포맷합니다.

참고

passphrase.txt 라는 파일을 생성하고 암호를 입력해야 합니다. 임의로 **NULL** 문자를 포함할 수 있는 암호를 생성할 수 있습니다. 암호가 줄 바꿈 문자로 종료되면 이 문자가 제거됩니다.

구문

```
rbd encryption format POOL_NAME/LUKS_IMAGE luks1|luks2 passphrase.txt
```

예제

```
[ceph: root@host01 /]# rbd encryption format pool1/luksimage1 luks1 passphrase.txt
```

참고

luks1 또는 **luks** 암호화 형식을 선택할 수 있습니다.

암호화 형식 작업은 **LUKS** 헤더를 생성하고 **RBD** 이미지 시작 시 씁니다. 단일 키 슬롯이 헤더에 추가됩니다. **keylot**은 임의로 생성된 암호화 키를 보유하며 암호 파일에서 읽은 암호로 보호됩니다. 기본적으로 현재 권장 모드이며 다른 **LUKS** 도구의 기본값인 **xts-plain64** 모드의 **AES-256**이 사용됩니다. 현재 암호 추가 또는 제거는 기본적으로 지원되지 않지만 **cryptsetup** 과 같은 **LUKS** 도구를 사용하여 수행할 수 있습니다. **LUKS** 헤더 크기는 **LUKS**에서 최대 **136MiB**까지 다를 수 있지만, 일반적으로 설치된 **libcryptsetup** 버전에 따라 최대 **16MiB**입니다. 최적의 성능을 위해 암호화 형식은 이미지 오브젝트 크기와 정렬되도록 데이터 오프셋을 설정합니다. 예를 들어 **8MiB** 오브젝트 크기로 구성된 이미지를 사용하는 경우 최소 오버헤드가 **8MiB**여야 합니다.

LUKS1에서 최소 암호화 단위인 섹터는 **512**바이트로 고정됩니다. **LUKS2**는 대규모 섹터를 지원하며 더 나은 성능을 위해 기본 섹터 크기가 최대 **4KiB**로 설정됩니다. 섹터보다 작거나 섹터 시작과 일치하지 않는 쓰기는 클라이언트에서 보호된 읽기-수정-쓰기 체인을 트리거하여 대기 시간이 상당히 저하됩니다. 이러한 정렬되지 않은 쓰기 배치로 인해 **I/O** 경쟁이 발생하여 성능이 더욱 저하될 수 있습니다. **Red Hat**은 들어오는 쓰기가 **LUKS** 섹터가 정렬되도록 보장할 수 없는 경우 **RBD** 암호화를 사용하지 않는 것이 좋습니다.

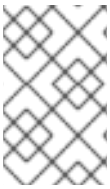
LUKS 암호화된 이미지를 매핑하려면 다음 명령을 실행합니다.

구문

```
rbd device map -t nbd -o encryption-format=luks1|luks2,encryption-passphrase-file=passphrase.txt
POOL_NAME/LUKS_IMAGE
```

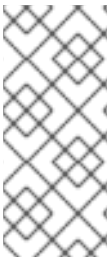
예제

```
[ceph: root@host01 /]# rbd device map -t nbd -o encryption-format=luks1,encryption-passphrase-
file=passphrase.txt pool1/luksimage1
```



참고

luks1 또는 **luks 2** 암호화 형식을 선택할 수 있습니다.



참고

보안상의 이유로 암호화 형식과 암호화 로드 작업이 모두 **CPU** 집약적이며 완료하는 데 몇 초가 걸릴 수 있습니다. 암호화된 **I/O**의 경우 **AES-NI**가 활성화되어 있다고 가정하면 상대적인 마이크로초 대기 시간이 추가될 수 있으며 **CPU** 사용률도 줄어듭니다.

5장. 스냅샷 관리

스토리지 관리자는 Ceph의 스냅샷 기능을 잘 알고 있으면 Red Hat Ceph Storage 클러스터에 저장된 이미지의 스냅샷과 복제본을 관리할 수 있습니다.

5.1. 사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.

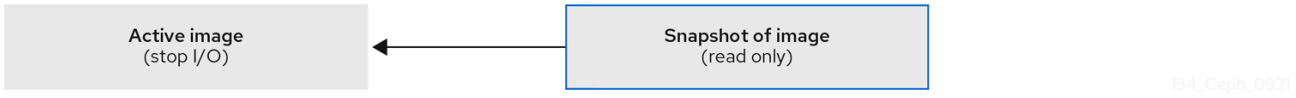
5.2. CEPH 블록 장치 스냅샷

스냅샷은 특정 시점에서의 이미지 상태에 대한 읽기 전용 사본입니다. Ceph 블록 장치의 고급 기능 중 하나는 이미지의 스냅샷을 만들어 이미지 상태 기록을 유지할 수 있다는 것입니다. Ceph는 또한 스냅샷 계층 지정을 지원하므로 이미지를 빠르고 쉽게 복제할 수 있습니다(예: 가상 시스템 이미지). Ceph는 rbd 명령 및 QEMU,libvirt, OpenStack, CloudStack을 비롯한 여러 상위 수준 인터페이스를 사용하는 블록 장치 스냅샷을 지원합니다.

참고

I/O 가 실행되는 동안 스냅샷을 생성하는 경우 스냅샷이 이미지의 정확한 또는 최신 데이터를 가져오지 못할 수 있으며, 마운트할 수 있도록 새 이미지로 스냅샷을 복제해야 할 수 있습니다. Red Hat은 이미지의 스냅샷을 생성하기 전에 I/O 를 중지할 것을 권장합니다. 이미지에 파일 시스템이 포함된 경우 스냅샷을 생성하기 전에 파일 시스템이 일관되게 유지되어야 합니다. I/O 를 중지하려면 fsfreeze 명령을 사용할 수 있습니다. 가상 시스템의 경우 qemu-guest-agent 를 사용하여 스냅샷을 만들 때 파일 시스템을 자동으로 중지할 수 있습니다.

그림 5.1. Ceph 블록 장치 스냅샷



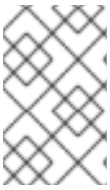
추가 리소스

- 자세한 내용은 fsfreeze(8) 도움말 페이지를 참조하십시오.

5.3. CEPH 사용자 및 인증 키

cephx 가 활성화되면 사용자 이름 또는 ID와 사용자의 해당 키가 포함된 인증 키의 경로를 지정해야 합

니다.



참고

cephx 는 기본적으로 활성화되어 있습니다.

다음 매개변수가 다시 입력되지 않도록 **CEPH_ARGS** 환경 변수를 추가할 수도 있습니다.

구문

```
rbd --id USER_ID --keyring=/path/to/secret [commands]
rbd --name USERNAME --keyring=/path/to/secret [commands]
```

예제

```
[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

매번 입력할 필요가 없도록 사용자 및 시크릿을 **CEPH_ARGS** 환경 변수에 추가합니다.

5.4. 블록 장치 스냅샷 생성

Ceph 블록 장치의 스냅샷을 만듭니다.

사전 요구 사항

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

- 노드에 대한 루트 수준 액세스.

절차

1. **snap create** 옵션, 풀 이름 및 이미지 이름을 지정합니다.

   - 방법 1:

   구문

   ```
   rbd --pool POOL_NAME snap create --snap SNAP_NAME IMAGE_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd --pool pool1 snap create --snap snap1 image1
   ```

   - 방법 2:

   구문

   ```
   rbd snap create POOL_NAME/IMAGE_NAME@SNAP_NAME
   ```

   예제

```
[root@rbd-client ~]# rbd snap create pool1/image1@snap1
```

5.5. 블록 장치 스냅샷 나열

블록 장치 스냅샷을 나열합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 풀 이름과 이미지 이름을 지정합니다.

구문

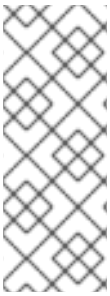
```
rbd --pool POOL_NAME --image IMAGE_NAME snap ls
rbd snap ls POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 --image image1 snap ls
[root@rbd-client ~]# rbd snap ls pool1/image1
```

5.6. 블록 장치 스냅샷 롤백

블록 장치 스냅샷을 롤백합니다.



참고

이미지를 스냅샷으로 롤백하면 현재 버전의 이미지를 스냅샷의 데이터로 덮어씁니다. 롤백을 실행하는 데 걸리는 시간은 이미지 크기와 함께 증가합니다. 스냅샷을 스냅샷으로 롤백하는 것보다 스냅샷에서 복제하는 것이 더 빠르며 기존 상태로 돌아가는 것이 좋습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. **snap rollback** 옵션, 풀 이름, 이미지 이름 및 **snap** 이름을 지정합니다.

구문

```
rbd --pool POOL_NAME snap rollback --snap SNAP_NAME IMAGE_NAME
rbd snap rollback POOL_NAME/IMAGE_NAME@SNAP_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 snap rollback --snap snap1 image1
[root@rbd-client ~]# rbd snap rollback pool1/image1@snap1
```

5.7. 블록 장치 스냅샷 삭제

Ceph 블록 장치의 스냅샷을 삭제합니다.

사전 요구 사항

- **실행 중인 Red Hat Ceph Storage 클러스터.**
- **노드에 대한 루트 수준 액세스.**

절차

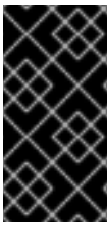
1. **블록 장치 스냅샷을 삭제하려면 snap rm 옵션, 풀 이름, 이미지 이름 및 스냅샷 이름을 지정합니다.**

구문

```
rbd --pool POOL_NAME snap rm --snap SNAP_NAME IMAGE_NAME
rbd snap rm POOL_NAME-/IMAGE_NAME@SNAP_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 snap rm --snap snap2 image1
[root@rbd-client ~]# rbd snap rm pool1/image1@snap1
```



중요

이미지에 복제본이 있는 경우 복제된 이미지는 상위 이미지 스냅샷에 대한 참조를 유지합니다. 상위 이미지 스냅샷을 삭제하려면 하위 이미지를 먼저 병합해야 합니다.

참고

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

추가 리소스

- 자세한 내용은 *Red Hat Ceph Storage 블록 장치 가이드에서 복제된 이미지* 플랫화를 참조하십시오.

5.8. 블록 장치 스냅샷 삭제

블록 장치 스냅샷 제거.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 특정 풀에 **snap purge** 옵션 및 이미지 이름을 지정합니다.

구문

```
rbd --pool POOL_NAME snap purge IMAGE_NAME
rbd snap purge POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 snap purge image1
[root@rbd-client ~]# rbd snap purge pool1/image1
```

5.9. 블록 장치 스냅샷 이름 변경

블록 장치 스냅샷의 이름을 바꿉니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 스냅샷 이름을 변경하려면 다음을 수행합니다.

구문

```
rbd snap rename POOL_NAME/IMAGE_NAME@ORIGINAL_SNAPSHOT_NAME
POOL_NAME/IMAGE_NAME@NEW_SNAPSHOT_NAME
```

예제

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

그러면 **data** 풀의 **dataset** 이미지의 **snap1** 스냅샷 이름을 **snap2** 로 변경합니다.

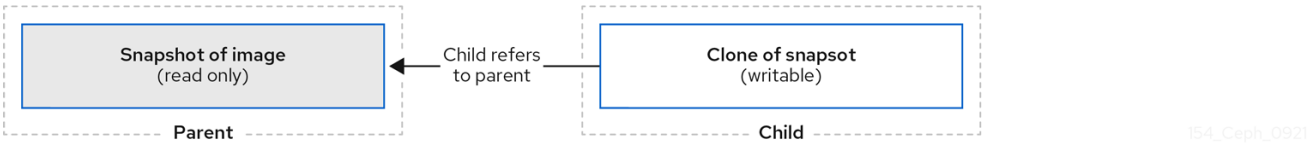
2. **rbd help snap rename** 명령을 실행하여 스냅샷 이름 변경에 대한 추가 세부 정보를 표시합

니다.

5.10. CEPH 블록 장치 계층 지정

Ceph에서는 블록 장치 스냅샷의 다양한 COW(Copy-On-Write) 또는 COW(Copy-On-read) 복제본을 만들 수 있습니다. 스냅샷 계층화를 사용하면 Ceph 블록 장치 클라이언트가 이미지를 매우 빠르게 만들 수 있습니다. 예를 들어 Linux VM이 작성된 블록 장치 이미지를 생성할 수 있습니다. 그런 다음 이미지 스냅숏을 만들고 스냅샷을 보호하고 원하는 만큼 복제본을 만듭니다. 스냅샷은 읽기 전용이므로 스냅샷을 복제하면 의미 체계가 간소화됩니다.신속하게 복제본을 만들 수 있도록 합니다.

그림 5.2. Ceph 블록 장치 계층 지정



참고

parent 및 child 용어는 Ceph 블록 장치 스냅샷, 상위 및 스냅샷에서 복제된 해당 이미지를 child를 의미합니다. 이러한 용어는 아래의 명령행 사용에 중요합니다.

복제된 각 이미지 하위는 상위 이미지에 참조를 저장하므로 복제된 이미지를 사용하여 상위 스냅샷을 열고 읽을 수 있습니다. 이 참조는 스냅샷의 정보가 복제본에 완전히 복사될 때 복제본이 병합 될 때 제거됩니다.

스냅샷 복제본은 다른 Ceph 블록 장치 이미지와 정확히 동일하게 작동합니다. 복제된 이미지의 읽기, 쓰기, 복제, 크기를 조정할 수 있습니다. 복제된 이미지는 특별한 제한 사항이 없습니다. 그러나 스냅샷 복제본은 스냅샷을 참조하므로 스냅샷을 복제하기 전에 보호해야 합니다.

스냅샷 복제본은 COW(Copy-On-Write) 또는 COW(Copy-On-read) 복제본이 될 수 있습니다. 복제본에는 COW(Copy-On-Write)가 항상 활성화되어 있으며 COW(Copy-on-read)는 명시적으로 활성화해야 합니다. COW(Copy-On-Write)는 복제본 내의 할당되지 않은 오브젝트에 쓰기 시 상위에서 복제본으로 데이터를 복사합니다. COOR(Copy-on-read)은 복제본 내에서 할당되지 않은 개체에서 읽을 때 상위에서 복제본으로 데이터를 복사합니다. 복제본에서 데이터를 읽는 작업은 오브젝트가 복제본에 아직 없는 경우에만 상위 데이터만 읽습니다. RADOS 블록 장치는 대규모 이미지를 여러 오브젝트로 나눕니다. 기본값은 4MB로 설정되고 모든 COW(Copy-On-Write) 및 COW(Copy-on-read) 작업이 전체 개체에서 발생합니다. 이 작업은 복제본에 1바이트를 기록하면 4MB의 개체를 상위로부터 읽고, 대상 개체가 이전 COW/COR 작업에서 복제에 아직 존재하지 않는 경우 복제본에 기록됩니다.

COW(Copy-on-read) 활성화 여부에 관계없이 복제본에서 기본 오브젝트를 읽고 충족할 수 없는 읽기 항목은 상위로 다시 라우팅됩니다. 상위 개수에는 거의 제한이 없기 때문에 복제본을 복제할 수 있다는 의

미이므로 이 재경로는 개체가 발견되거나 기본 상위 이미지에 도달할 때까지 계속됩니다. **COW(Copy-On-read)**가 활성화되면 복제본에서 직접 만족하지 못하는 읽기 결과 전체 개체가 상위에서 읽고 데이터를 복제본에 쓰고 데이터를 복제본에 쓰면 부모로부터 읽을 필요 없이 복제 자체에서 동일한 범위를 읽을 수 있습니다.

이는 기본적으로 온디맨드 개체별 플랫화된 작업입니다. 이 기능은 특히 다른 지리적 위치에서 다른 풀의 부모인 복제본이 대기 시간이 긴 상위 연결에 있을 때 유용합니다. **COW(Copy-On-read)**를 사용하면 읽기의 분할 대기 시간이 줄어듭니다. 처음 읽기에는 대기 시간이 길지만, 예를 들어 복제본에서 추가 데이터를 읽습니다. 예를 들어 복제본에서 **1**바이트를 읽지만, 이제 상위 항목에서 **4MB**를 읽고 복제본에 기록해야 하지만 모든 향후 읽기 항목은 복제본 자체에서 제공됩니다.

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

추가 리소스

- 플랫화에 대한 자세한 내용은 ***Red Hat Ceph Storage Block Device Gudie*** 의 *복제된 이미지* 플랫화를 참조하십시오.

5.11. 블록 장치 스냅샷 보호

복제본은 상위 스냅샷에 액세스합니다. 사용자가 실수로 상위 스냅샷을 삭제하면 모든 복제본이 손상됩니다.

set-require-min-compat-client 매개변수를 **Ceph**의 **mimic** 버전보다 크거나 같음으로 설정할 수 있습니다.

예제

```
ceph osd set-require-min-compat-client mimic
```

이렇게 하면 기본적으로 복제본 **v2**가 생성됩니다. 그러나 **mimic**보다 오래된 클라이언트는 해당 블록 장치 이미지에 액세스할 수 없습니다.

참고

복제본 **v2**에는 스냅샷을 보호할 필요가 없습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

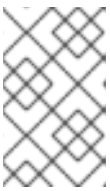
1. 다음 *명령에서* ***POOL_NAME***, ***IMAGE_NAME*** 및 ***SNAP_SHOT_NAME*** 을 지정합니다.

구문

```
rbd --pool POOL_NAME snap protect --image IMAGE_NAME --snap SNAPSHOT_NAME
rbd snap protect POOL_NAME/IMAGE_NAME@SNAPSHOT_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 snap protect --image image1 --snap snap1
[root@rbd-client ~]# rbd snap protect pool1/image1@snap1
```



참고

보호된 스냅샷을 삭제할 수 없습니다.

5.12. 블록 장치 스냅샷 복제

블록 장치 스냅샷을 복제하여 동일한 풀 또는 다른 풀 내에 스냅샷의 하위 이미지를 생성하거나 씁니다. 한 가지 사용 사례는 한 풀에서 읽기 전용 이미지와 스냅샷을 템플릿으로 유지하고 다른 풀에서 쓰기 가능한 복제본을 유지 관리하는 것입니다.

참고

복제본 **v2**에는 스냅샷을 보호할 필요가 없습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 스냅샷을 복제하려면 상위 풀, 스냅샷, 하위 풀 및 이미지 이름을 지정해야 합니다.

구문

```
rbd snap --pool POOL_NAME --image PARENT_IMAGE --snap SNAP_NAME --dest-pool
POOL_NAME --dest CHILD_IMAGE_NAME
rbd clone POOL_NAME/PARENT_IMAGE@SNAP_NAME
POOL_NAME/CHILD_IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd clone --pool pool1 --image image1 --snap snap2 --dest-pool pool2 --
dest childimage1
[root@rbd-client ~]# rbd clone pool1/image1@snap1 pool1/childimage1
```

5.13. 블록 장치 스냅샷 보호 해제

스냅샷을 삭제하려면 먼저 보호를 해제해야 합니다. 또한 복제본에서 참조가 있는 스냅샷을 삭제할 수 *없습니다*. 스냅샷을 삭제하려면 먼저 각 스냅샷 복제본을 병합해야 합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 다음 명령을 실행합니다.

구문

```
rbd --pool POOL_NAME snap unprotect --image IMAGE_NAME --snap SNAPSHOT_NAME
rbd snap unprotect POOL_NAME/IMAGE_NAME@SNAPSHOT_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 snap unprotect --image image1 --snap snap1

[root@rbd-client ~]# rbd snap unprotect pool1/image1@snap1
```

5.14. 스냅샷의 하위 항목 나열

스냅샷의 하위 항목을 나열합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 스냅샷의 하위 항목을 나열하려면 다음을 실행합니다.

구문

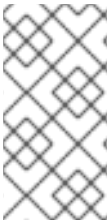
```
rbd --pool POOL_NAME children --image IMAGE_NAME --snap SNAP_NAME
rbd children POOL_NAME/IMAGE_NAME@SNAPSHOT_NAME
```

예제

```
[root@rbd-client ~]# rbd --pool pool1 children --image image1 --snap snap1
[root@rbd-client ~]# rbd children pool1/image1@snap1
```

5.15. 복제된 이미지 병합

복제된 이미지에는 상위 스냅샷에 대한 참조가 유지됩니다. 하위 복제본에서 상위 스냅샷에 대한 참조를 제거하면 스냅샷의 정보를 복제본에 복사하여 이미지를 실제로 "플랫"합니다. 복제본을 병합하는 데 걸리는 시간은 스냅샷 크기와 함께 증가합니다. 병합된 이미지에는 스냅샷의 모든 정보가 포함되어 있기 때문에 병합된 이미지에서 계층 복제보다 더 많은 스토리지 공간을 사용합니다.



참고

이미지에서 심층 플랫 기능이 활성화된 경우 이미지 복제본은 기본적으로 상위 항목과 분리됩니다.

사전 요구 사항

- **실행 중인 Red Hat Ceph Storage 클러스터.**
- **노드에 대한 루트 수준 액세스.**

절차

1. **하위 이미지와 연결된 상위 이미지 스냅샷을 삭제하려면 하위 이미지를 먼저 병합해야 합니다.**

   구문

   ```
   rbd --pool POOL_NAME flatten --image IMAGE_NAME
   rbd flatten POOL_NAME/IMAGE_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd --pool pool1 flatten --image childimage1
   [root@rbd-client ~]# rbd flatten pool1/childimage1
   ```

6장. CEPH 블록 장치 미러링

스토리지 관리자는 **Red Hat Ceph Storage** 클러스터 간에 데이터 이미지를 미러링하여 **Ceph** 블록 장치에 다른 중복 계층을 추가할 수 있습니다. **Ceph** 블록 장치 미러링을 이해하고 사용하면 사이트 장애와 같은 데이터 손실을 방지할 수 있습니다. **Ceph** 블록 장치를 미러링하는 데는 일방향 미러링 또는 양방향 미러링의 두 가지 구성이 있으며 풀과 개별 이미지에 미러링을 구성할 수 있습니다.

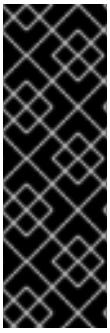
6.1. 사전 요구 사항

- **Red Hat Ceph Storage** 클러스터를 실행하는 최소 두 개의 정상 실행.
- 두 스토리지 클러스터 간의 네트워크 연결.
- 각 **Red Hat Ceph Storage** 클러스터의 **Ceph** 클라이언트 노드에 액세스합니다.
- 관리자 수준의 기능이 있는 **CephX** 사용자.

6.2. CEPH 블록 장치 미러링

RADOS 블록 장치(**RBD**) 미러링은 두 개 이상의 **Ceph** 스토리지 클러스터 간에 **Ceph** 블록 장치 이미지의 비동기 복제 프로세스입니다. 다른 지리적 위치에 **Ceph** 스토리지 클러스터를 배치하면 **RBD Mirroring**을 통해 사이트 재해에서 복구할 수 있습니다. 저널 기반 **Ceph** 블록 장치 미러링은 읽기 및 쓰기, 블록 장치 크기 조정, 스냅샷, 복제본 및 플랫화를 포함하여 모든 이미지에 대해 지점에서 일관된 복제를 보장합니다.

RBD 미러링은 배타적 잠금 및 저널링 기능을 사용하여 이미지에 대한 모든 수정 사항을 발생하는 순서대로 기록합니다. 따라서 이미지의 크래시 일관성 미러를 사용할 수 있습니다.



중요

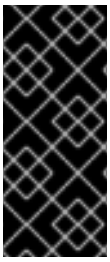
블록 장치 이미지를 미러링하는 기본 및 보조 풀을 지원하는 **CRUSH** 계층 구조에는 동일한 용량 및 성능 특성이 있어야 하며 초과 대기 시간 없이 미러링할 수 있는 적절한 대역폭이 있어야 합니다. 예를 들어 기본 스토리지 클러스터의 이미지에 대한 **X MB/s** 평균 쓰기 처리량이 있는 경우 네트워크에서 보조 사이트에 네트워크 연결에서 **N * X** 처리량을 지원하고 안전 계수 **Y%**를 사용하여 **N** 이미지를 미러링해야 합니다.

rbd-mirror 데몬은 원격 기본 이미지의 변경 사항을 가져와서 특정 **Ceph** 스토리지 클러스터에서 다른

Ceph 스토리지 클러스터로 이미지를 동기화하고 이러한 변경 사항을 로컬 비기본 이미지로 기록합니다. **rbd-mirror** 데몬은 단방향 미러링을 위해 단일 **Ceph** 스토리지 클러스터에서 실행하거나 미러링 관계에 참여하는 양방향 미러링을 위해 두 개의 **Ceph** 스토리지 클러스터에서 실행할 수 있습니다.

RBD 미러링이 작동하려면 단방향 또는 양방향 복제를 사용하는 경우 몇 가지 가정이 이루어집니다.

- 두 스토리지 클러스터에 모두 이름이 같은 풀이 있습니다.
- 풀에는 미러링하려는 저널 지원 이미지가 포함되어 있습니다.



중요

단방향 또는 양방향 복제에서는 **rbd-mirror** 의 각 인스턴스를 동시에 다른 **Ceph** 스토리지 클러스터에 연결할 수 있어야 합니다. 또한 네트워크에는 미러링을 처리하기 위해 두 데이터 센터 사이트 간에 충분한 대역폭이 있어야 합니다.

일방향 복제

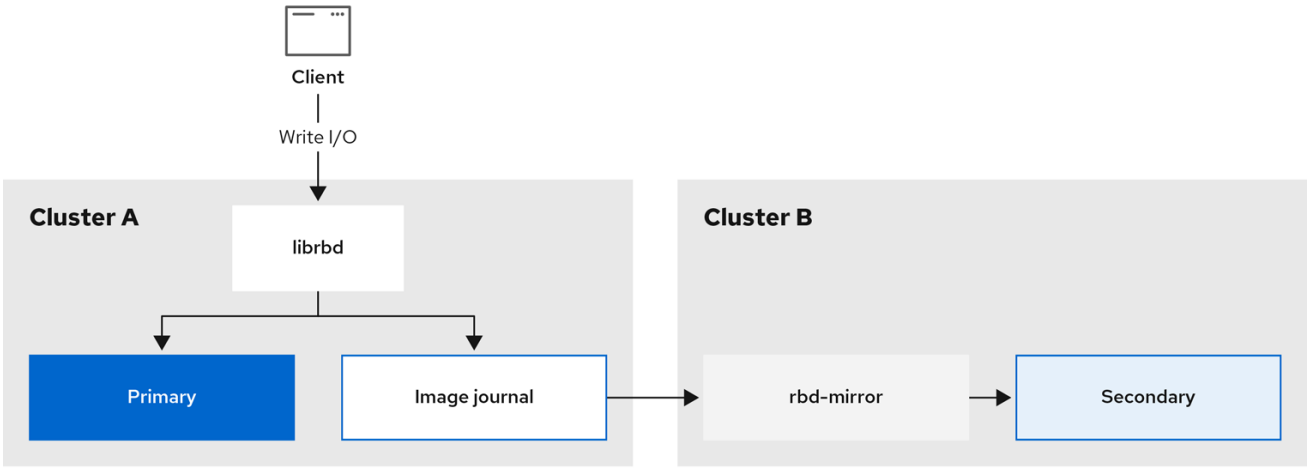
단방향 미러링은 한 스토리지 클러스터에 있는 기본 이미지 또는 이미지 풀이 보조 스토리지 클러스터에 복제됨을 의미합니다. 단방향 미러링은 여러 보조 스토리지 클러스터에 대한 복제도 지원합니다.

보조 스토리지 클러스터에서 이미지는 기본이 아닌 복제입니다. 즉, **Ceph** 클라이언트가 이미지에 쓸 수 없습니다. 데이터가 기본 스토리지 클러스터에서 보조 스토리지 클러스터로 미러링되면 **rbd-mirror** 는 보조 스토리지 클러스터에서만 실행됩니다.

단방향 미러링이 작동하려면 몇 가지 가정이 이루어집니다.

- **Ceph** 스토리지 클러스터가 두 개 있으며 기본 스토리지 클러스터에서 보조 스토리지 클러스터로 이미지를 복제해야 합니다.
- 보조 스토리지 클러스터에는 **rbd-mirror** 데몬을 실행하는 **Ceph** 클라이언트 노드가 연결되어 있습니다. **rbd-mirror** 데몬은 기본 스토리지 클러스터에 연결하여 이미지를 보조 스토리지 클러스터에 동기화합니다.

그림 6.1. 단방향 미러링



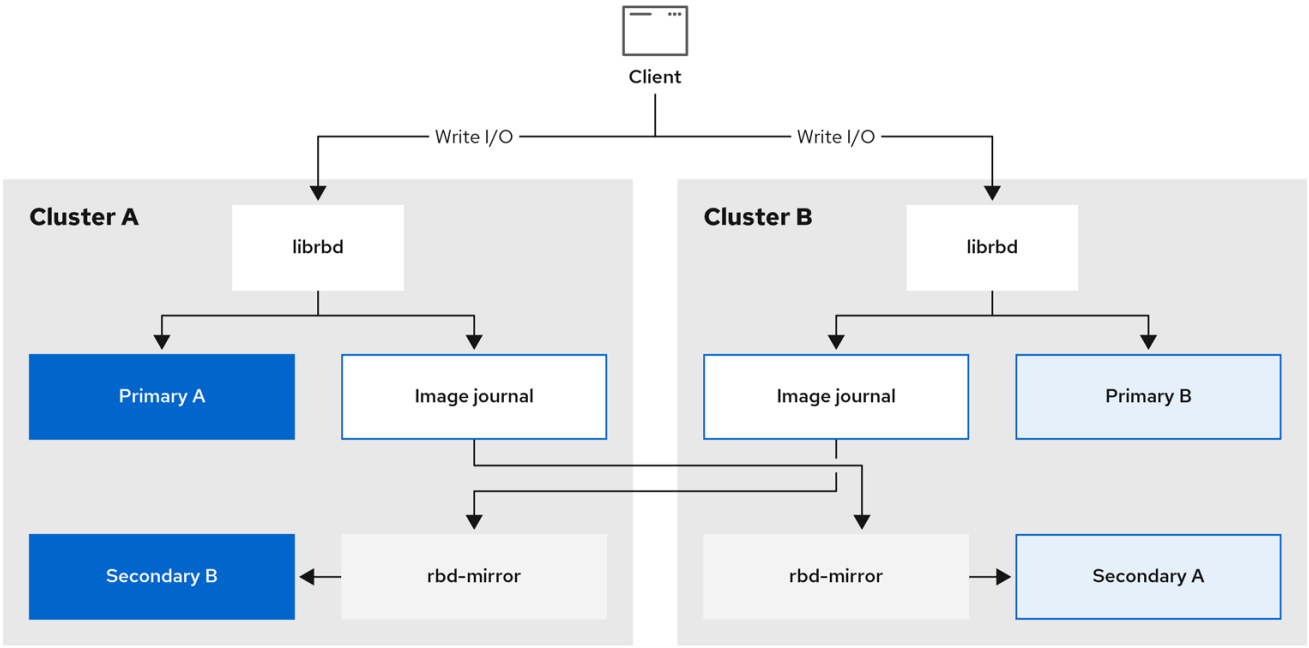
양방향 복제

양방향 복제는 기본 클러스터에 **rbd-mirror** 데몬을 추가하여 이미지를 강등하고 보조 클러스터에서 승격할 수 있도록 합니다. 그런 다음 보조 클러스터의 이미지를 변경할 수 있으며 **2**차에서 **1**차로 역방향으로 복제됩니다. 클러스터에서 이미지를 승격하고 강등할 수 있도록 두 클러스터 모두 **rbd-mirror** 를 실행해야 합니다. 현재는 양방향 복제는 두 사이트 사이에서만 지원됩니다.

양방향 미러링이 작동하려면 몇 가지 가정이 이루어집니다.

- 두 개의 스토리지 클러스터가 있으며 두 방향으로 이미지를 복제할 수 있습니다.
- 두 스토리지 클러스터에 모두 **rbd-mirror** 데몬을 실행하는 클라이언트 노드가 연결되어 있습니다. 보조 스토리지 클러스터에서 실행되는 **rbd-mirror** 데몬은 기본 스토리지 클러스터에 연결하여 이미지를 보조 스토리지에 동기화하고, 기본 스토리지 클러스터에서 실행되는 **rbd-mirror** 데몬은 보조 스토리지 클러스터에 연결하여 이미지를 기본 스토리지 클러스터에 동기화합니다.

그림 6.2. 양방향 미러링



참고

Red Hat Ceph Storage 4부터 단일 클러스터에서 여러 개의 활성 **rbd-mirror** 데몬을 실행할 수 있습니다.

미러링 모드

미러링은 미러 피어링 스토리지 클러스터와 함께 풀 단위로 구성됩니다. **Ceph**는 풀의 이미지 유형에 따라 두 가지 미러링 모드를 지원합니다.

풀 모드

저널링 기능이 활성화된 풀의 모든 이미지가 미러링됩니다.

이미지 모드

풀 내의 특정 이미지 하위 집합만 미러링됩니다. 각 이미지에 대해 별도로 미러링을 활성화해야 합니다.

이미지 상태

이미지를 수정할 수 있는지 여부는 상태에 따라 다릅니다.

- 기본 상태의 이미지를 수정할 수 있습니다.
- 기본이 아닌 상태의 이미지는 수정할 수 없습니다.

이미지에서 미러링을 처음 활성화하면 이미지가 자동으로 기본으로 승격됩니다. 프로모션이 발생할 수 있습니다.

- 풀 모드에서 미러링을 활성화하여 암시적으로.
- 특정 이미지의 미러링을 활성화하여 명시적으로 표시합니다.

기본 이미지를 강등하고 기본이 아닌 이미지를 승격할 수 있습니다.

추가 리소스

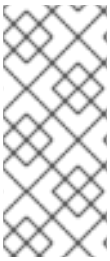
- 자세한 내용은 ***Red Hat Ceph Storage Block Device Guide*** 의 *이미지 승격 및 강등* 섹션을 참조하십시오.

6.3. 명령줄 인터페이스를 사용하여 일방향 미러링 구성

이 절차에서는 기본 스토리지 클러스터에서 보조 스토리지 클러스터로 풀의 일방향 복제를 구성합니다.

참고

단방향 복제를 사용하는 경우 여러 보조 스토리지 클러스터에 미러링할 수 있습니다.



참고

이 섹션의 예제에서는 기본 이미지를 **site-a**로 사용하여 기본 스토리지 클러스터를 참조하여 두 개의 스토리지 클러스터를 구분하고 이미지를 **site- b** 로 복제하는 보조 스토리지 클러스터를 구분합니다. 이러한 예제에서 사용된 풀 이름을 **data** 라고 합니다.

사전 요구 사항

- 최소 2개의 정상 및 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 각 스토리지 클러스터의 **Ceph** 클라이언트 노드에 대한 루트 수준 액세스.
- 관리자 수준의 기능이 있는 **CephX** 사용자.

절차

1. 두 사이트 모두에서 **cephadm** 쉘에 로그인합니다.

   예제

   ```
   [root@site-a ~]# cephadm shell
   [root@site-b ~]# cephadm shell
   ```

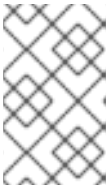
2. **site-b** 에서 보조 클러스터에서 **mirror** 데몬 배포를 예약합니다.

   구문

   ```
   ceph orch apply rbd-mirror --placement=NODENAME
   ```

   예제

   ```
   [ceph: root@site-b /]# ceph orch apply rbd-mirror --placement=host04
   ```



참고

nodename 은 보조 클러스터에서 미러링을 설정하려는 호스트입니다.

3. **site-a** 의 이미지에서 저널링 기능을 활성화합니다.

a. 새 이미지의 경우 **--image-feature** 옵션을 사용하십시오.

구문

```
rbd create IMAGE_NAME --size MEGABYTES --pool POOL_NAME --image-feature
FEATURE FEATURE
```

예제

```
[ceph: root@site-a /]# rbd create image1 --size 1024 --pool data --image-feature
exclusive-lock,journaling
```

참고

exclusive-lock 이 이미 활성화된 경우 **journaling** 을 유일한 인수로 사용하십시오. 그렇지 않으면 다음 오류를 반환합니다.

```
one or more requested features are already enabled
(22) Invalid argument
```

b. 기존 이미지의 경우 **rbd feature enable** 명령을 사용합니다.

구문

```
rbd feature enable POOL_NAME/IMAGE_NAME FEATURE, FEATURE
```

예제

```
[ceph: root@site-a /]# rbd feature enable data/image1 exclusive-lock, journaling
```

c. **기본적으로 모든 새 이미지에서 저널링을 활성화하려면 ceph config set 명령을 사용하여 구성 매개변수를 설정합니다.**

예제

```
[ceph: root@site-a /]# ceph config set global rbd_default_features 125
[ceph: root@site-a /]# ceph config show mon.host01 rbd_default_features
```

4. **두 스토리지 클러스터에서 미러링 모드(풀 또는 이미지 모드)를 선택합니다.**

a. **풀 모드 활성화 :**

구문

```
rbd mirror pool enable POOL_NAME MODE
```

예제

```
[ceph: root@site-a /]# rbd mirror pool enable data pool
[ceph: root@site-b /]# rbd mirror pool enable data pool
```

이 예에서는 data 라는 전체 풀을 미러링할 수 있습니다.

b.

이미지 모드 활성화 :

구문

```
rbd mirror pool enable POOL_NAME MODE
```

예제

```
[ceph: root@site-a /]# rbd mirror pool enable data image
[ceph: root@site-b /]# rbd mirror pool enable data image
```

이 예에서는 data 풀에서 이미지 모드 미러링을 활성화합니다.

참고

풀의 특정 이미지에 미러링을 사용하려면 *Red Hat Ceph Storage 블록 장치 가이드* 의 *이미지 미러링 활성화* 섹션을 참조하십시오.

c. **두 사이트 모두에서 미러링이 성공적으로 활성화되었는지 확인합니다.**

구문

```
rbd mirror pool info POOL_NAME
```

예제

```
[ceph: root@site-a /]# rbd mirror pool info data
Mode: pool
Site Name: c13d8065-b33d-4cb5-b35f-127a02768e7f

Peer Sites: none

[ceph: root@site-b /]# rbd mirror pool info data
Mode: pool
Site Name: a4c667e2-b635-47ad-b462-6faeeee78df7

Peer Sites: none
```

5. **Ceph 클라이언트 노드에서 스토리지 클러스터 피어를 부트스트랩합니다.**

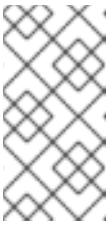
a. **Ceph 사용자 계정을 생성하고 스토리지 클러스터 피어를 풀에 등록합니다.**

구문

```
rbd mirror pool peer bootstrap create --site-name PRIMARY_LOCAL_SITE_NAME
POOL_NAME > PATH_TO_BOOTSTRAP_TOKEN
```

예제

```
[ceph: root@rbd-client-site-a /]# rbd mirror pool peer bootstrap create --site-name site-a
data > /root/bootstrap_token_site-a
```



참고

이 예제 부트스트랩 명령은 client.rbd-mirror.site-a 및 client.rbd-mirror-peer Ceph 사용자를 생성합니다.

b. **부트스트랩 토큰 파일을 site-b 스토리지 클러스터에 복사합니다.**

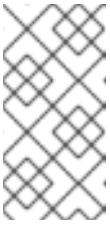
c. **site-b 스토리지 클러스터에서 부트스트랩 토큰을 가져옵니다.**

구문

```
rbd mirror pool peer bootstrap import --site-name SECONDARY_LOCAL_SITE_NAME --
direction rx-only POOL_NAME PATH_TO_BOOTSTRAP_TOKEN
```

예제

```
[ceph: root@rbd-client-site-b /]# rbd mirror pool peer bootstrap import --site-name site-b -
-direction rx-only data /root/bootstrap_token_site-a
```



참고

단방향 **RBD** 미러링의 경우 부트스트랩 피어를 부트 스트랩할 때 양방향 미러링이 기본값이므로 **--direction rx-only** 인수를 사용해야 합니다.

6.

미러링 상태를 확인하려면 기본 및 보조 사이트의 **Ceph** 모니터 노드에서 다음 명령을 실행합니다.

구문

```
rbd mirror image status POOL_NAME/IMAGE_NAME
```

예제

```
[ceph: root@mon-site-a /]# rbd mirror image status data/image1
image1:
  global_id:   c13d8065-b33d-4cb5-b35f-127a02768e7f
  state:       up+stopped
  description: remote image is non-primary
  service:     host03.yuoosv on host03
  last_update: 2021-10-06 09:13:58
```

여기서는 **rbd -mirror** 데몬이 실행 중임을 의미하며 중지됨 은 이 이미지가 다른 스토리지 클러스터에서 복제 대상이 아님을 의미합니다. 이는 이미지가 이 스토리지 클러스터에서 기본이기 때문입니다.

예제

```
[ceph: root@mon-site-b /]# rbd mirror image status data/image1
image1:
  global_id:   c13d8065-b33d-4cb5-b35f-127a02768e7f
```

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage Block Device Guide*의 *Ceph 블록 장치 미러링*** 섹션을 참조하십시오.
- **Ceph** 사용자에 대한 자세한 내용은 ***Red Hat Ceph Storage** 관리 가이드* 의 사용자 관리 *섹션* 을 참조하십시오.

6.4. 명령줄 인터페이스를 사용하여 양방향 미러링 구성

이 절차에서는 기본 스토리지 클러스터와 보조 스토리지 클러스터 간에 풀의 양방향 복제를 구성합니다.

참고

양방향 복제를 사용하는 경우 두 스토리지 클러스터만 미러링할 수 있습니다.

참고

이 섹션의 예제에서는 기본 이미지를 **site-a**로 사용하여 기본 스토리지 클러스터를 참조하여 두 개의 스토리지 클러스터를 구분하고 이미지를 **site- b** 로 복제하는 보조 스토리지 클러스터를 구분합니다. 이러한 예제에서 사용된 풀 이름을 **data** 라고 합니다.

사전 요구 사항

- 최소 **2**개의 정상 및 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 각 스토리지 클러스터의 **Ceph** 클라이언트 노드에 대한 루트 수준 액세스.
- 관리자 수준의 기능이 있는 **CephX** 사용자.

절차

1. 두 사이트 모두에서 **cephadm** 쉘에 로그인합니다.

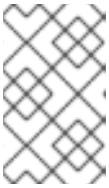
예제

```
[root@site-a ~]# cephadm shell
[root@site-b ~]# cephadm shell
```

2. **site-a** 기본 클러스터에서 다음 명령을 실행합니다.

예제

```
[ceph: root@site-a /]# ceph orch apply rbd-mirror --placement=host01
```



참고

nodename 은 미러링을 설정하려는 호스트입니다.

3. **site-b** 에서 보조 클러스터에서 **mirror** 데몬 배포를 예약합니다.

구문

```
ceph orch apply rbd-mirror --placement=NODENAME
```

예제

```
[ceph: root@site-b /]# ceph orch apply rbd-mirror --placement=host04
```

참고

nodename 은 보조 클러스터에서 미러링을 설정하려는 호스트입니다.

4. **site-a** 의 이미지에서 저널링 기능을 활성화합니다.

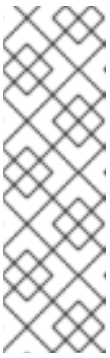
a. 새 이미지의 경우 **--image-feature** 옵션을 사용하십시오.

구문

```
rbd create IMAGE_NAME --size MEGABYTES --pool POOL_NAME --image-feature
FEATURE FEATURE
```

예제

```
[ceph: root@site-a /]# rbd create image1 --size 1024 --pool data --image-feature
exclusive-lock,journaling
```



참고

exclusive-lock 이 이미 활성화된 경우 **journaling** 을 유일한 인수로 사용하십시오. 그렇지 않으면 다음 오류를 반환합니다.

```
one or more requested features are already enabled
(22) Invalid argument
```

b.

기존 이미지의 경우 rbd feature enable 명령을 사용합니다.

구문

```
rbd feature enable POOL_NAME/IMAGE_NAME FEATURE, FEATURE
```

예제

```
[ceph: root@site-a /]# rbd feature enable data/image1 exclusive-lock, journaling
```

c.

기본적으로 모든 새 이미지에서 저널링을 활성화하려면 ceph config set 명령을 사용하여 구성 매개변수를 설정합니다.

예제

```
[ceph: root@site-a /]# ceph config set global rbd_default_features 125
[ceph: root@site-a /]# ceph config show mon.host01 rbd_default_features
```

5.

두 스토리지 클러스터에서 미러링 모드(풀 또는 이미지 모드)를 선택합니다.

a.

풀 모드 활성화 :

구문

```
rbd mirror pool enable POOL_NAME MODE
```

예제

```
[ceph: root@site-a /]# rbd mirror pool enable data pool
[ceph: root@site-b /]# rbd mirror pool enable data pool
```

이 예에서는 data 라는 전체 풀을 미러링할 수 있습니다.

b.

이미지 모드 활성화 :

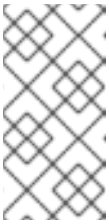
구문

```
rbd mirror pool enable POOL_NAME MODE
```

예제

```
[ceph: root@site-a /]# rbd mirror pool enable data image
[ceph: root@site-b /]# rbd mirror pool enable data image
```

이 예에서는 data 풀에서 이미지 모드 미러링을 활성화합니다.



참고

풀의 특정 이미지에 미러링을 사용하려면 ***Red Hat Ceph Storage*** *블록 장치 가이드* 의 *이미지 미러링 활성화* 섹션을 참조하십시오.

c. 두 사이트 모두에서 미러링이 성공적으로 활성화되었는지 확인합니다.

구문

```
rbd mirror pool info POOL_NAME
```

예제

```
[ceph: root@site-a /]# rbd mirror pool info data
Mode: pool
Site Name: c13d8065-b33d-4cb5-b35f-127a02768e7f

Peer Sites: none

[ceph: root@site-b /]# rbd mirror pool info data
Mode: pool
Site Name: a4c667e2-b635-47ad-b462-6faeeee78df7

Peer Sites: none
```

6. **Ceph** 클라이언트 노드에서 스토리지 클러스터 피어를 부트스트랩합니다.

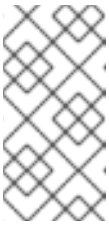
a. **Ceph** 사용자 계정을 생성하고 스토리지 클러스터 피어를 풀에 등록합니다.

구문

```
rbd mirror pool peer bootstrap create --site-name PRIMARY_LOCAL_SITE_NAME
POOL_NAME > PATH_TO_BOOTSTRAP_TOKEN
```

예제

```
[ceph: root@rbd-client-site-a /]# rbd mirror pool peer bootstrap create --site-name site-a
data > /root/bootstrap_token_site-a
```



참고

이 예제 부트스트랩 명령은 client.rbd-mirror.site-a 및 client.rbd-mirror-peer Ceph 사용자를 생성합니다.

b. **부트스트랩 토큰 파일을 site-b 스토리지 클러스터에 복사합니다.**

c. **site-b 스토리지 클러스터에서 부트스트랩 토큰을 가져옵니다.**

구문

```
rbd mirror pool peer bootstrap import --site-name SECONDARY_LOCAL_SITE_NAME --
direction rx-tx POOL_NAME PATH_TO_BOOTSTRAP_TOKEN
```

예제

```
[ceph: root@rbd-client-site-b /]# rbd mirror pool peer bootstrap import --site-name site-b -
-direction rx-tx data /root/bootstrap_token_site-a
```

참고

피어를 부트스트랩할 때 양방향 미러링이 기본값이므로 **--direction** 인수는 선택 사항입니다.

7. 미러링 상태를 확인하려면 기본 및 보조 사이트의 **Ceph** 모니터 노드에서 다음 명령을 실행합니다.

구문

```
rbd mirror image status POOL_NAME/IMAGE_NAME
```

예제

```
[ceph: root@mon-site-a /]# rbd mirror image status data/image1
image1:
  global_id:   a4c667e2-b635-47ad-b462-6faeeee78df7
  state:       up+stopped
  description: local image is primary
  service:     host03.glsdbv on host03.ceph.redhat.com
  last_update: 2021-09-16 10:55:58
  peer_sites:
    name: a
    state: up+stopped
    description: replaying,
{"bytes_per_second":0.0,"entries_behind_primary":0,"entries_per_second":0.0,"non_primary_p
osition":{"entry_tid":3,"object_number":3,"tag_tid":1},"primary_position":
{"entry_tid":3,"object_number":3,"tag_tid":1}}
    last_update: 2021-09-16 10:55:50
```

여기서는 **rbd -mirror** 데몬이 실행 중임을 의미하며 중지됨 은 이 이미지가 다른 스토리지 클러스터에서 복제 대상이 아님을 의미합니다. 이는 이미지가 이 스토리지 클러스터에서 기본이기

때문입니다.

예제

```
[ceph: root@mon-site-b /]# rbd mirror image status data/image1
image1:
  global_id:   a4c667e2-b635-47ad-b462-6faeeee78df7
  state:       up+replaying
  description: replaying,
{"bytes_per_second":0.0,"entries_behind_primary":0,"entries_per_second":0.0,"non_primary_p
osition":{"entry_tid":3,"object_number":3,"tag_tid":1},"primary_position":
{"entry_tid":3,"object_number":3,"tag_tid":1}}
  service:     host05.dtisty on host05
  last_update: 2021-09-16 10:57:20
  peer_sites:
    name: b
    state: up+stopped
    description: local image is primary
    last_update: 2021-09-16 10:57:28
```

이미지가 **up+replaying** 상태인 경우 미러링이 제대로 작동합니다. 여기서는 **rbd -mirror** 데몬이 실행 중 임을 의미하며 재생 은 이 이미지가 다른 스토리지 클러스터에서 복제 대상임을 의미합니다.

참고

사이트 간 연결에 따라 미러링은 이미지를 동기화하는 데 시간이 오래 걸릴 수 있습니다.

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage Block Device Guide의 Ceph블록 장치 미러링*** 섹션을 참조하십시오.
- **Ceph** 사용자에 대한 자세한 내용은 ***Red Hat Ceph Storage 관리 가이드*** 의 사용자 관리 *섹션* 을 참조하십시오.

6.5. CEPH 블록 장치 미러링 관리

스토리지 관리자는 **Ceph** 블록 장치 미러링 환경을 관리하는 데 도움이 되는 다양한 작업을 수행할 수 있습니다. 다음 작업을 수행할 수 있습니다.

- 스토리지 클러스터 피어에 대한 정보 보기.
- 스토리지 클러스터 피어 추가 또는 제거.
- 풀 또는 이미지의 미러링 상태 가져오기.
- 풀 또는 이미지에서 미러링 활성화.
- 풀 또는 이미지에서 미러링 비활성화.
- 블록 장치 복제 지연.
- 이미지 승격 및 데모.

6.5.1. 사전 요구 사항

- 최소 **2**개의 정상 실행 **Red Hat Ceph Storage** 클러스터.
- **Ceph** 클라이언트 노드에 대한 루트 수준 액세스.
- 단방향 또는 양방향 **Ceph** 블록 장치 미러링 관계.
- 관리자 수준의 기능이 있는 **CephX** 사용자.

6.5.2. 피어에 대한 정보 보기

스토리지 클러스터 피어에 대한 정보를 확인합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 피어에 대한 정보를 보려면 다음을 수행합니다.

구문

```
rbd mirror pool info POOL_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror pool info data
Mode: pool
Site Name: a

Peer Sites:

UUID: 950ddadf-f995-47b7-9416-b9bb233f66e3
Name: b
Mirror UUID: 4696cd9d-1466-4f98-a97a-3748b6b722b3
Direction: rx-tx
Client: client.rbd-mirror-peer
```

6.5.3. 풀에서 미러링 활성화

두 피어 클러스터에서 다음 명령을 실행하여 풀에서 미러링을 활성화합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 풀에서 미러링을 활성화하려면 다음을 수행합니다.

구문

```
rbd mirror pool enable POOL_NAME MODE
```

예제

```
[root@rbd-client ~]# rbd mirror pool enable data pool
```

이 예에서는 **data** 라는 전체 풀을 미러링할 수 있습니다.

예제

```
[root@rbd-client ~]# rbd mirror pool enable data image
```

이 예에서는 **data** 풀에서 이미지 모드 미러링을 활성화합니다.

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage** 블록 장치 가이드의 **Ceph** 블록 장치미러링* 섹션을 참조하십시오.

6.5.4. 풀에서 미러링 비활성화

미러링을 비활성화하기 전에 피어 클러스터를 제거합니다.

참고

풀에서 미러링을 비활성화하면 풀 내의 모든 이미지에서도 이미지 모드에서 별도로 미러링을 사용하도록 설정합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 풀에서 미러링을 비활성화하려면 다음을 수행합니다.

구문

```
rbd mirror pool disable POOL_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror pool disable data
```

이 예제에서는 **data** 풀의 미러링을 비활성화합니다.

6.5.5. 이미지 미러링 활성화

두 피어 스토리지 클러스터에서 이미지 모드에서 전체 풀에서 미러링을 활성화합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 풀 내에서 특정 이미지에 대한 미러링을 활성화합니다.

구문

```
rbd mirror image enable POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image enable data/image2
```

이 예에서는 데이터 풀에서 **image2** 이미지에 대한 미러링을 활성화합니다.

추가 리소스

-

자세한 내용은 ***Red Hat Ceph Storage*** *블록 장치 가이드* 의 *풀에서 미러링 활성화* 섹션을 참조하십시오.

6.5.6. 이미지 미러링 비활성화

이미지에서 **Ceph** 블록 장치 미러링을 비활성화할 수 있습니다.

사전 요구 사항

- 스냅샷 기반 미러링이 구성된 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 특정 이미지에 대한 미러링을 비활성화하려면 다음을 수행합니다.

구문

```
rbd mirror image disable POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image disable data/image2
```

이 예제에서는 데이터 풀에서 **image2** 이미지의 미러링을 비활성화합니다.

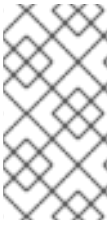
추가 리소스

-

cephadm-ansible *인벤토리에 클라이언트를 추가하는 방법에 대한 자세한 내용은* ***Red Hat Ceph Storage*** *설치 가이드 의* ***Ansible*** *인벤토리 위치 구성 섹션을* 참조하십시오.

6.5.7. 이미지 승격 및 강등

풀에서 이미지를 승격하거나 강등할 수 있습니다.



참고

승격 후에도 이미지가 유효하지 않으므로 기본이 아닌 이미지를 강제로 승격하지 마십시오.

사전 요구 사항

- 스냅샷 기반 미러링이 구성된 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 기본이 아닌 이미지를 강등하려면 다음을 수행합니다.

구문

```
rbd mirror image demote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image demote data/image2
```

이 예제에서는 데이터 풀에서 **image2** 이미지를 시연합니다.

2. 이미지를 **1**차로 승격하려면 다음을 수행합니다.

구문

```
rbd mirror image promote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image promote data/image2
```

이 예제에서는 데이터 풀의 **image2** 를 승격합니다.

사용 중인 미러링 유형에 따라 자세한 내용은 *양방향 미러링을 사용하여 재해에서 복구* 또는 *복구를 참조하십시오*.

구문

```
rbd mirror image promote --force POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image promote --force data/image2
```

강등이 피어 **Ceph** 스토리지 클러스터에 전파될 수 없는 경우 강제 승격을 사용합니다. 예를 들어 클러스터 오류 또는 통신 중단으로 인해 발생합니다.

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage*** *블록 장치 가이드에서* *주문되지 않은 종료 후* ***Failover*** 섹션을 참조하십시오.

6.5.8. 이미지 재동기화

이미지를 다시 동기화할 수 있습니다. 두 피어 클러스터 간의 상태가 일치하지 않는 경우 **rbd-mirror** 데몬은 불일치의 원인이 되는 이미지를 미러링하지 않습니다.

사전 요구 사항

- 스냅샷 기반 미러링이 구성된 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 기본 이미지에 다시 동기화를 요청하려면 다음을 수행합니다.

구문

```
rbd mirror image resync POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image resync data/image2
```

이 예제에서는 데이터 풀에서 **image2** 를 다시 동기화하도록 요청합니다.

추가 리소스

- 재해로 인해 일관성 없는 상태에서 복구하려면 자세한 내용은 양방향 미러링 *으로 재해* 에서 복구 또는 복구를 참조하십시오.

6.5.9. 풀의 미러링 상태 가져오기

스토리지 클러스터에서 풀의 미러 상태를 가져올 수 있습니다.

사전 요구 사항

- 스냅샷 기반 미러링이 구성된 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 미러링 풀 요약을 가져오려면 다음을 수행합니다.

구문

```
rbd mirror pool status POOL_NAME
```

예제

```
[root@site-a ~]# rbd mirror pool status data
health: OK
daemon health: OK
image health: OK
images: 1 total
    1 replaying
```

작은 정보

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

6.5.10. 단일 이미지의 미러링 상태 가져오기

mirror image status 명령을 실행하여 이미지의 미러 상태를 가져올 수 있습니다.

사전 요구 사항

- 스냅샷 기반 미러링이 구성된 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 미러링된 이미지의 상태를 가져오려면 다음을 수행합니다.

구문

```
rbd mirror image status POOL_NAME/IMAGE_NAME
```

예제

```
[root@site-a ~]# rbd mirror image status data/image2
image2:
  global_id:   1e3422a2-433e-4316-9e43-1827f8dbe0ef
  state:       up+unknown
  description: remote image is non-primary
  service:     pluto008.yuoosv on pluto008
  last_update: 2021-10-06 09:37:58
```

이 예제는 **data** 풀에서 **image2** 이미지의 상태를 가져옵니다.

6.5.11. 블록 장치 복제 지연

일방향 또는 양방향 복제를 사용하든 **RADOS** 블록 장치(**RBD**) 미러링 이미지 간 복제를 지연할 수 있습니다. 보조 이미지로 복제하기 전에 기본 이미지에 대한 원치 않는 변경을 취소해야 하는 경우, 시험 시간이 필요할 경우 지연된 복제를 구현할 수 있습니다.

지연된 복제를 구현하려면 대상 스토리지 클러스터 내의 **rbd-mirror** 데몬에서 **rbd_mirroring_replay_delay = *MINIMUM_DELAY_IN_SECONDS*** 구성 옵션을 설정해야 합니다. 이 설정은 **rbd-mirror** 데몬에서 사용하는 **ceph.conf** 파일 내에서 또는 개별 이미지 기반으로 전역적으로 적용할 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 특정 이미지에 대해 지연된 복제를 활용하려면 기본 이미지에서 다음 **rbd CLI** 명령을 실행합니다.

   구문

```
rbd image-meta set POOL_NAME/IMAGE_NAME conf_rbd_mirroring_replay_delay
MINIMUM_DELAY_IN_SECONDS
```

예제

```
[root@rbd-client ~]# rbd image-meta set vms/vm-1 conf_rbd_mirroring_replay_delay 600
```

이 예제에서는 **vms** 풀에서 이미지 **vm-1** 에 대해 **10**분 최소 복제 지연을 설정합니다.

6.5.12. 저널 기반 및 스냅샷 기반 미러링 개요

RBD 이미지는 두 가지 모드를 통해 두 개의 **Red Hat Ceph Storage** 클러스터 간에 비동기식으로 미러링할 수 있습니다.

저널 기반 미러링

이 모드에서는 **RBD** 저널링 이미지 기능을 사용하여 두 개의 **Red Hat Ceph Storage** 클러스터 간에 지점 간 일관된 복제를 보장합니다. **RBD** 이미지에 대한 모든 쓰기가 관련 저널에 먼저 기록되는 경우 실제 이미지는 수정되지 않습니다. 원격 클러스터는 이 저널에서 읽고 업데이트를 이미지의 로컬 복사본에 재생합니다. **RBD** 이미지를 쓸 때마다 **Ceph** 클러스터에 두 개의 쓰기 작업이 생성되므로 쓰기 대기 시간이 **RBD** 저널링 이미지 기능을 사용하여 거의 두 배가 됩니다.

스냅샷 기반 미러링

이 모드는 정기적으로 예약되거나 수동으로 생성된 **RBD** 이미지 미러 스냅샷을 사용하여 두 개의 **Red Hat Ceph Storage** 클러스터 간에 충돌하는 일관된 **RBD** 이미지를 복제합니다. 원격 클러스터는 두 개의 미러 스냅샷 간 데이터 또는 메타데이터 업데이트를 확인하고 **deltas**를 이미지의 로컬 사본에 복사합니다. **RBD fast-diff** 이미지 기능을 사용하면 전체 **RBD** 이미지를 스캔하지 않고도 업데이트된 데이터 블록을 신속하게 확인할 수 있습니다. 장애 조치**(failover)** 시나리오에서 사용하기 전에 두 스냅샷 간의 전체 **delta**를 동기화해야 합니다. 부분적으로 적용된 모든 **delta** 세트는 페일오버 시 롤백됩니다.

미러링을 비활성화하고 스냅샷을 활성화하여 저널 기반 미러링을 스냅샷 기반 미러링으로 변환할 수 있습니다.

예제

```
[ceph: root@site-a /]# rbd mirror image disable mirror_pool/mirror_image
Mirroring disabled
```

예제

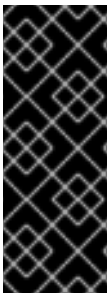
```
[ceph: root@rbd-client /]# rbd mirror image enable mirror_pool/mirror_image snapshot
Mirroring enabled
```

6.5.13. mirror-snapshot 이미지 생성

스냅샷 기반 미러링을 사용할 때 **RBD** 이미지의 변경된 내용을 미러링해야 하는 경우 이미지 **mirror-snapshot**을 만듭니다.

사전 요구 사항

- **Red Hat Ceph Storage** 클러스터를 실행하는 최소 두 개의 정상 실행.
- **Red Hat Ceph Storage** 클러스터의 **Ceph** 클라이언트 노드에 대한 루트 수준의 액세스.
- 관리자 수준의 기능이 있는 **CephX** 사용자.
- 스냅샷 미러가 생성되는 **Red Hat Ceph Storage** 클러스터에 액세스합니다.



중요

기본적으로 이미지당 **3**개의 이미지 **mirror-snapshots**만 만들 수 있습니다. 제한에 도달하면 최신 이미지 **mirror-snapshot**이 자동으로 제거됩니다. 필요한 경우 제한을 **rbd_mirroring_max_mirroring_snapshots** 구성을 통해 재정의할 수 있습니다. 이미지 **mirror-snapshots**는 이미지가 제거되거나 미러링을 비활성화할 때 자동으로 삭제됩니다.

절차

- 이미지 미러링자 스냅샷을 생성하려면 다음을 수행합니다.

구문

```
rbd --cluster CLUSTER_NAME mirror image snapshot POOL_NAME/IMAGE_NAME
```

예제

```
[root@site-a ~]# rbd mirror image snapshot data/image1
```

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage** 블록 장치 가이드의 **Ceph** 블록 장치미러링* 섹션을 참조하십시오.

6.5.14. mirror-snapshots 예약

mirror-snapshot 일정이 정의되면 **mirror-snapshots**를 자동으로 생성할 수 있습니다. **mirror-snapshot**은 전역적으로, 풀당 또는 이미지당 수준을 예약할 수 있습니다. 여러 **mirror-snapshot** 일정은 모든 수준에서 정의할 수 있지만 개별 미러링된 이미지와 일치하는 가장 구체적인 스냅샷 일정만 실행됩니다.

6.5.14.1. mirror-snapshot 일정 생성

snapshot schedule 명령을 사용하여 **mirror-snapshot** 일정을 생성할 수 있습니다.

사전 요구 사항

- **Red Hat Ceph Storage** 클러스터를 실행하는 최소 두 개의 정상 실행.

- **Red Hat Ceph Storage** 클러스터의 **Ceph** 클라이언트 노드에 대한 루트 수준의 액세스.
- 관리자 수준의 기능이 있는 **CephX** 사용자.
- 스냅샷 미러가 생성되는 **Red Hat Ceph Storage** 클러스터에 액세스합니다.

절차

1. **mirror-snapshot** 일정을 생성하려면 다음을 수행합니다.

구문

```
rbd --cluster CLUSTER_NAME mirror snapshot schedule add --pool POOL_NAME --image IMAGE_NAME INTERVAL [START_TIME]
```

The *CLUSTER_NAME* 은 클러스터 이름이 기본 이름 **ceph** 와 다른 경우에만 사용해야 합니다. 간격은 각각 **d, h** 또는 **m** 접미사를 사용하여 일, 시간 또는 분 단위로 지정할 수 있습니다. 선택적 **START_TIME**은 **ISO 8601** 시간 형식을 사용하여 지정할 수 있습니다.

예제

```
[root@site-a ~]# rbd mirror snapshot schedule add --pool data --image image1 6h
```

예제

```
[root@site-a ~]# rbd mirror snapshot schedule add --pool data --image image1 24h 14:00:00-05:00
```

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage*** *블록 장치 가이드의* ***Ceph*** *블록 장치미러링* 섹션을 참조하십시오.

6.5.14.2. 특정 수준에서 모든 스냅샷 일정 나열

특정 수준에서 모든 스냅샷 일정을 나열할 수 있습니다.

사전 요구 사항

- **Red Hat Ceph Storage** 클러스터를 실행하는 최소 두 개의 정상 실행.
- **Red Hat Ceph Storage** 클러스터의 **Ceph** 클라이언트 노드에 대한 루트 수준의 액세스.
- 관리자 수준의 기능이 있는 **CephX** 사용자.
- 스냅샷 미러가 생성되는 **Red Hat Ceph Storage** 클러스터에 액세스합니다.

절차

1. 선택적 풀 또는 이미지 이름을 사용하여 특정 글로벌, 풀 또는 이미지 수준에 대한 모든 스냅샷 일정을 나열하려면 다음을 수행합니다.

   구문

   ```
   rbd --cluster site-a mirror snapshot schedule ls --pool POOL_NAME --recursive
   ```

   또한 **--recursive** 옵션을 지정하여 다음과 같이 지정된 수준에서 모든 일정을 나열할 수 있습니다.

예제

```
[root@rbd-client ~]# rbd mirror snapshot schedule ls --pool data --recursive
POOL       NAMESPACE IMAGE  SCHEDULE
data       -         -      every 1d starting at 14:00:00-05:00
data       -         image1   every 6h
```

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage** 블록 장치 가이드의 **Ceph** 블록 장치미러링* 섹션을 참조하십시오.

6.5.14.3. mirror-snapshot 일정 제거

스냅샷 **schedule remove** 명령을 사용하여 **mirror-snapshot** 일정을 제거할 수 있습니다.

사전 요구 사항

- **Red Hat Ceph Storage** 클러스터를 실행하는 최소 두 개의 정상 실행.
- **Red Hat Ceph Storage** 클러스터의 **Ceph** 클라이언트 노드에 대한 루트 수준의 액세스.
- 관리자 수준의 기능이 있는 **CephX** 사용자.
- 스냅샷 미러가 생성되는 **Red Hat Ceph Storage** 클러스터에 액세스합니다.

절차

1. **mirror-snapshot** 일정을 제거하려면 다음을 수행합니다.

구문

```
rbd --cluster CLUSTER_NAME mirror snapshot schedule remove --pool POOL_NAME --
image IMAGE_NAME INTERVAL START_TIME
```

간격은 각각 **d, h, m** 접미사를 사용하여 일, 시간 또는 분 단위로 지정할 수 있습니다. 선택적 **START_TIME**은 **ISO 8601** 시간 형식을 사용하여 지정할 수 있습니다.

예제

```
[root@site-a ~]# rbd mirror snapshot schedule remove --pool data --image image1 6h
```

예제

```
[root@site-a ~]# rbd mirror snapshot schedule remove --pool data --image image1 24h
14:00:00-05:00
```

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage 블록 장치 가이드의 Ceph 블록 장치미러링*** 섹션을 참조하십시오.

6.5.14.4. 생성할 다음 스냅샷의 상태 보기

스냅샷 기반 미러링 **RBD** 이미지에 대해 생성할 다음 스냅샷의 상태를 볼 수 있습니다.

사전 요구 사항

- Red Hat Ceph Storage 클러스터를 실행하는 최소 두 개의 정상 실행.
- Red Hat Ceph Storage 클러스터의 Ceph 클라이언트 노드에 대한 루트 수준의 액세스.
- 관리자 수준의 기능이 있는 CephX 사용자.
- 스냅샷 미러가 생성되는 Red Hat Ceph Storage 클러스터에 액세스합니다.

절차

1. 생성할 다음 스냅샷의 상태를 보려면 다음을 수행합니다.

구문

```
rbd --cluster site-a mirror snapshot schedule status [--pool POOL_NAME] [--image
IMAGE_NAME]
```

예제

```
[root@rbd-client ~]# rbd mirror snapshot schedule status
SCHEDULE    TIME       IMAGE
2021-09-21 18:00:00 data/image1
```

추가 리소스

- 자세한 내용은 *Red Hat Ceph Storage 블록 장치 가이드의 Ceph 블록 장치미러링* 섹션을 참조하십시오.

6.6. 재해에서 복구

스토리지 관리자는 미러링이 구성된 다른 스토리지 클러스터에서 데이터를 복구하는 방법을 확인하여 최종 하드웨어 오류를 준비할 수 있습니다.

이 예제에서 기본 스토리지 클러스터를 **site-a** 라고 하며 보조 스토리지 클러스터를 **site-b** 라고 합니다. 또한 스토리지 클러스터에는 두 개의 이미지 **image 1** 및 **image2** 가 있는 데이터 풀이 있습니다.

6.6.1. 사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 단방향 또는 양방향 미러링이 구성되었습니다.

6.6.2. 재해 복구

두 개 이상의 **Red Hat Ceph Storage** 클러스터 간의 블록 데이터의 비동기 복제는 다운타임을 줄이고 중요한 데이터 센터 장애 발생 시 데이터 손실을 방지합니다. 이러한 오류는 대규모 인력이라고도 하는 광범위한 영향을 미치며, 전력 그리드와 자연 재해에 영향을 미치기 때문에 발생할 수 있습니다.

이러한 시나리오에서는 고객 데이터를 보호해야 합니다. 볼륨은 일관성 및 효율성과 함께 복제되어야 하며 **RPO(Re구re Point Objective)** 및 **RTO(Re구re Time Objective)** 대상 내에서도 복제해야 합니다. 이 솔루션은 **WAN-DR(Wide Area Network- Disaster Recovery)**이라고 합니다.

이러한 시나리오에서는 기본 시스템과 데이터 센터를 복원하기가 어렵습니다. 복구하는 가장 빠른 방법은 애플리케이션을 대체 **Red Hat Ceph Storage** 클러스터(재지정 복구 사이트)로 페일오버하고 사용 가능한 데이터의 최신 사본으로 클러스터가 작동하도록 하는 것입니다. 이러한 오류 시나리오에서 복구하는 데 사용되는 솔루션은 애플리케이션을 통해 안내합니다.

- 복구 지점 목표 **(RPO)**: 데이터 손실의 양으로, 애플리케이션이 최악의 경우 허용합니다.
- 복구 시간 목표 **(RTO)**: 사용 가능한 데이터의 최신 복사본에 따라 애플리케이션을 다시 가져오는 데 걸린 시간입니다.

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage 블록 장치 가이드의 Ceph 블록 장치미러링*** 장을 참조하십시오.

- 암호화된 상태에서 유선 데이터 전송에 대한 자세한 내용은 ***Red Hat Ceph Storage Data Security and Hardening Guide*** 의 전송 시 암호화 *섹션* 을 참조하십시오.

6.6.3. 단방향 미러링으로 재해 복구

단방향 미러링을 사용할 때 재해에서 복구하려면 다음 절차를 사용합니다. 기본 클러스터가 종료된 후 보조 클러스터로 장애 조치하는 방법과 다시 실패하는 방법을 보여줍니다. 종료는 순서에 따라 순서에 따라 정렬되지 않을 수 있습니다.

중요

단방향 미러링은 여러 보조 사이트를 지원합니다. 추가 보조 클러스터를 사용하는 경우 보조 클러스터 중 하나를 선택하여 실패합니다. 다시 실패하는 동안 동일한 클러스터에서 동기화합니다.

6.6.4. 양방향 미러링으로 재해 복구

양방향 미러링을 사용할 때 재해에서 복구하려면 다음 절차를 사용합니다. 기본 클러스터가 종료된 후 보조 클러스터에서 미러링된 데이터로 장애 조치하는 방법과 장애 복구 방법을 보여줍니다. 종료는 순서에 따라 순서에 따라 정렬되지 않을 수 있습니다.

6.6.5. 순서가 종료된 후 장애 조치

순서가 종료된 후 보조 스토리지 클러스터로 장애 조치(**failover**)합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터 **2**개 이상.
- 노드에 대한 루트 수준 액세스.
- 단방향 미러링으로 구성된 풀 미러링 또는 이미지 미러링.

절차

1.

기본 이미지를 사용하는 모든 클라이언트를 중지합니다. 이 단계는 이미지를 사용하는 클라이언트에 따라 다릅니다. 예를 들어 이미지를 사용하는 **OpenStack** 인스턴스에서 볼륨을 분리합니다.

2. **site-a** 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site-a** 클러스터에 있는 기본 이미지를 강등합니다.

구문

```
rbd mirror image demote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image demote data/image1
[root@rbd-client ~]# rbd mirror image demote data/image2
```

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

구문

```
rbd mirror image promote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image promote data/image1
[root@rbd-client ~]# rbd mirror image promote data/image2
```

4. 잠시 후에 **site-b** 클러스터의 모니터 노드에서 이미지의 상태를 확인합니다. **up+stopped** 상태로 표시되고 **primary**로 나열되어야 합니다.

```
[root@rbd-client ~]# 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
[root@rbd-client ~]# 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 재개합니다. 이 단계는 이미지를 사용하는 클라이언트에 따라 다릅니다.

추가 리소스

- ***Red Hat OpenStack Platform** 스토리지 가이드의 블록 스토리지 및 볼륨 장을 참조하십시오.*

6.6.6. 순서가 아닌 종료 후 장애 조치

주문이 아닌 종료 후 보조 스토리지 클러스터로 장애 조치(failover)합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터 2개 이상.
- 노드에 대한 루트 수준 액세스.
- 단방향 미러링으로 구성된 풀 미러링 또는 이미지 미러링.

절차

1. 기본 스토리지 클러스터가 다운되었는지 확인합니다.

2. 기본 이미지를 사용하는 모든 클라이언트를 중지합니다. 이 단계는 이미지를 사용하는 클라이언트에 따라 다릅니다. 예를 들어 이미지를 사용하는 **OpenStack** 인스턴스에서 볼륨을 분리합니다.

3. **site-b** 스토리지 클러스터의 **Ceph Monitor** 노드에서 기본이 아닌 이미지를 승격합니다. 강등이 **site-a** 스토리지 클러스터로 전파될 수 없기 때문에 **--force** 옵션을 사용합니다.

   구문

   ```
   rbd mirror image promote --force POOL_NAME/IMAGE_NAME
   ```

   예제

   ```
   [root@rbd-client ~]# rbd mirror image promote --force data/image1
   [root@rbd-client ~]# rbd mirror image promote --force data/image2
   ```

4. **site-b** 스토리지 클러스터의 **Ceph Monitor** 노드에서 이미지의 상태를 확인합니다. **up+stopping_replay** 의 상태를 표시해야 하며 설명은 **force promoted**로 표시되어야 합니다.

   예제

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

추가 리소스

- ***Red Hat OpenStack Platform** 스토리지 가이드의 블록 스토리지 및 볼륨 장을 참조하십시오.*

6.6.7. 실패를 위한 준비

원래 두 스토리지 클러스터가 단방향 미러링용으로만 구성된 경우 이미지를 반대로 복제하기 위해 미러링을 위해 기본 스토리지 클러스터를 구성합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 클라이언트 노드에 대한 루트 수준 액세스.

절차

1. **Cephadm** 쉘에 로그인합니다.

   예제

   ```
   [root@rbd-client ~]# cephadm shell
   ```

2. **site-a** 스토리지 클러스터에서 다음 명령을 실행합니다.

   예제

```
[ceph: root@rbd-client /]# ceph orch apply rbd-mirror --placement=host01
```

3. **Ceph 클라이언트 노드에서 스토리지 클러스터 피어를 부트스트랩합니다.**

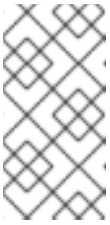
a. **Ceph 사용자 계정을 생성하고 스토리지 클러스터 피어를 풀에 등록합니다.**

구문

```
rbd mirror pool peer bootstrap create --site-name LOCAL_SITE_NAME POOL_NAME >
PATH_TO_BOOTSTRAP_TOKEN
```

예제

```
[ceph: root@rbd-client-site-a /]# rbd mirror pool peer bootstrap create --site-name site-a
data > /root/bootstrap_token_site-a
```



참고

이 예제 부트스트랩 명령은 client.rbd-mirror.site-a 및 client.rbd-mirror-peer Ceph 사용자를 생성합니다.

b. **부트스트랩 토큰 파일을 site-b 스토리지 클러스터에 복사합니다.**

c. **site-b 스토리지 클러스터에서 부트스트랩 토큰을 가져옵니다.**

구문

```
rbd mirror pool peer bootstrap import --site-name LOCAL_SITE_NAME --direction rx-only POOL_NAME PATH_TO_BOOTSTRAP_TOKEN
```

예제

```
[ceph: root@rbd-client-site-b /]# rbd mirror pool peer bootstrap import --site-name site-b --direction rx-only data /root/bootstrap_token_site-a
```

참고

단방향 **RBD** 미러링의 경우 부트스트랩 피어를 부트 스트랩할 때 양방향 미러링이 기본값이므로 **--direction rx-only** 인수를 사용해야 합니다.

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

예제

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

추가 리소스

- 자세한 내용은 ***Red Hat Ceph Storage** 관리 가이드의 사용자 관리 장을 참조하십시오.*

6.6.7.1. 기본 스토리지 클러스터로 다시 실패

이전의 기본 스토리지 클러스터가 복구되면 기본 스토리지 클러스터로 다시 실패합니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터 **2**개 이상.
- 노드에 대한 루트 수준 액세스.
- 단방향 미러링으로 구성된 풀 미러링 또는 이미지 미러링.

절차

1. **site-b** 클러스터의 모니터 노드에서 이미지의 상태를 다시 확인합니다. **up-stopped** 상태를 표시해야 하며 설명에 로컬 이미지가 **primary** 라고 표시되어야 합니다.

   예제

```
[root@rbd-client ~]# 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
[root@rbd-client ~]# 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** 스토리지 클러스터의 **Ceph Monitor** 노드에서 이미지가 여전히 기본인지 확인합니다.

구문

```
rbd mirror pool info POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd info data/image1
[root@rbd-client ~]# rbd info data/image2
```

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

3. **site-a** 스토리지 클러스터의 **Ceph Monitor** 노드에서 다음과 같은 명령을 실행하여 주로 나열된 이미지를 모두 강등합니다.

구문

```
rbd mirror image demote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image demote data/image1
```

4. 순서가 지정되지 않은 종료인 경우만 이미지를 다시 동기화합니다. **site-a** 스토리지 클러스터의 모니터 노드에서 다음 명령을 실행하여 **site- b**에서 **site- a** 로 이미지를 다시 동기화합니다.

구문

```
rbd mirror image resync POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image resync data/image1
Flagged image for resync from primary
[root@rbd-client ~]# rbd mirror image resync data/image2
Flagged image for resync from primary
```

5. 잠시 후에 **up+replaying** 상태인지 확인하여 이미지의 재동기화가 완료되었는지 확인합니다. **site-a** 스토리지 클러스터의 모니터 노드에서 다음 명령을 실행하여 해당 상태를 확인합니다.

구문

```
rbd mirror image status POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image status data/image1
[root@rbd-client ~]# rbd mirror image status data/image2
```

6. **site-b 스토리지 클러스터의 Ceph Monitor 노드에서 다음 명령을 실행하여 site-b 스토리지 클러스터에서 이미지를 강등합니다.**

구문

```
rbd mirror image demote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image demote data/image1
[root@rbd-client ~]# rbd mirror image demote data/image2
```

참고

보조 스토리지 클러스터가 여러 개 있는 경우 승격된 보조 스토리지 클러스터에서만 이 작업을 수행해야 합니다.

7. **site-a 스토리지 클러스터의 Ceph Monitor 노드에서 다음 명령을 실행하여 site-a 스토리지 클러스터에 있는 이전의 기본 이미지를 승격합니다.**

구문

```
rbd mirror image promote POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd mirror image promote data/image1
[root@rbd-client ~]# rbd mirror image promote data/image2
```

8. **site-a 스토리지 클러스터의 Ceph Monitor 노드에서 이미지의 상태를 확인합니다. up+stopped 상태가 표시되고 설명에 로컬 이미지가 primary 라고 표시되어야 합니다.**

구문

```
rbd mirror image status POOL_NAME/IMAGE_NAME
```

예제

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

6.6.8. 양방향 미러링 삭제

실패가 다시 완료되면 양방향 미러링을 제거하고 Ceph 블록 장치 미러링 서비스를 비활성화할 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

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

   예제

   ```
   [root@rbd-client ~]# rbd mirror pool peer remove data client.remote@remote --cluster local
   [root@rbd-client ~]# 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.

   구문

   ```
   systemctl stop ceph-rbd-mirror@CLIENT_ID
   systemctl disable ceph-rbd-mirror@CLIENT_ID
   systemctl disable ceph-rbd-mirror.target
   ```

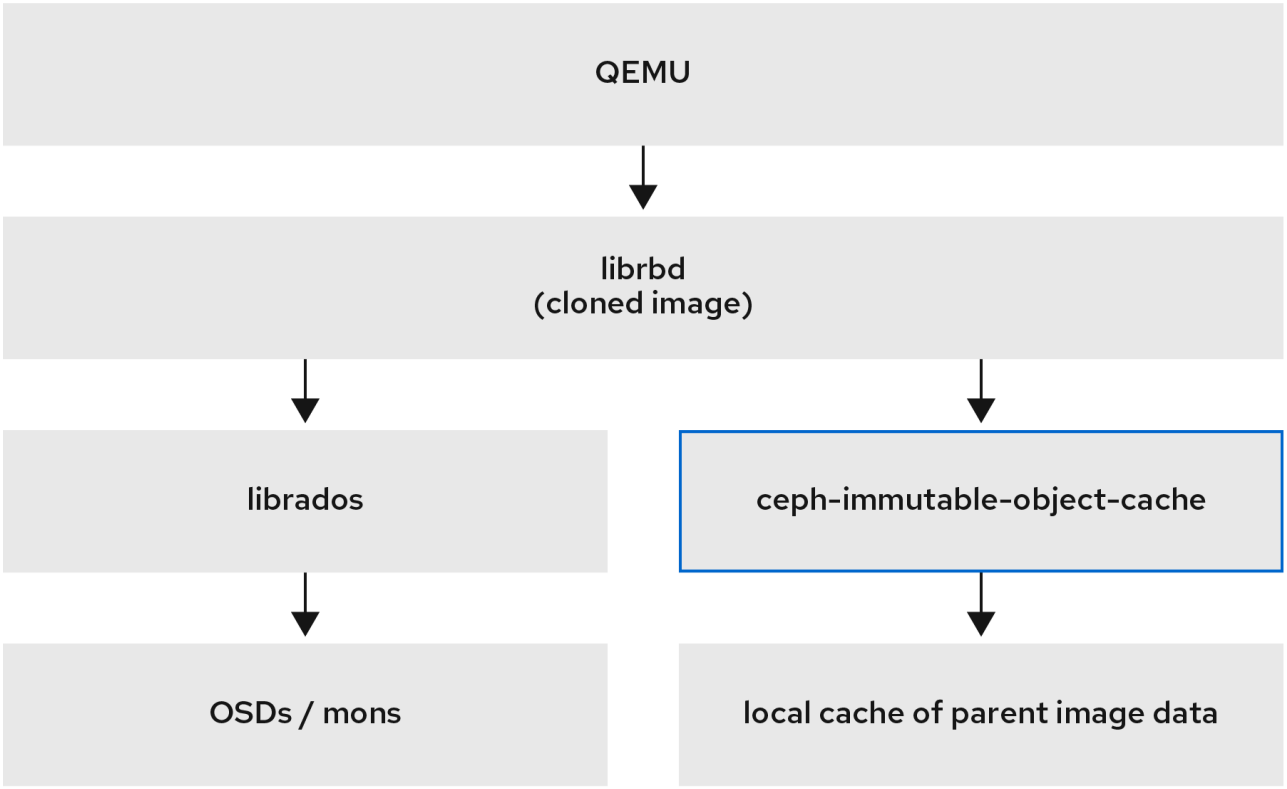
   예제

   ```
   [root@rbd-client ~]# systemctl stop ceph-rbd-mirror@site-a
   [root@rbd-client ~]# systemctl disable ceph-rbd-mirror@site-a
   [root@rbd-client ~]# systemctl disable ceph-rbd-mirror.target
   ```

7장. CEPH-IMMUTABLE-OBJECT-CACHE 데몬 관리

스토리지 관리자로 **ceph-immutable-object-cache** 데몬을 사용하여 로컬 디스크에 상위 이미지 콘텐츠를 캐시합니다. 이 캐시는 로컬 캐싱 디렉터리에 있습니다. 향후 해당 데이터를 읽고 로컬 캐시를 사용합니다.

그림 **7.1. Ceph** 불변 캐시 데몬



7.1. CEPH-IMMUTABLE-OBJECT-CACHE 데몬 설명

복제된 블록 장치 이미지는 일반적으로 상위 이미지의 작은 부분만 수정합니다. 예를 들어 **VDI**(가상 데스크탑 인터페이스)에서 가상 시스템은 동일한 기본 이미지에서 복제되며 처음에 호스트 이름과 **IP** 주소에만 따라 다릅니다. 부팅 중에 상위 이미지의 로컬 캐시를 사용하는 경우 캐싱 호스트에서 읽기 속도가 빨라집니다. 이 변경으로 인해 클라이언트가 클러스터 네트워크 트래픽으로 줄어듭니다.

ceph-immutable-object-cache 데몬을 사용해야 하는 이유

ceph-immutable-object-cache 데몬은 **Red Hat Ceph Storage**의 일부입니다. 확장 가능한 오픈 소스 및 분산 스토리지 시스템입니다. 기본 검색 경로를 사용하여 **ceph.conf** 파일을 찾고 주소 및 인증 정보(예: **/etc/ceph/CLUSTER.conf**, **/etc/ceph/CLUSTER.keyring** , **/etc/ceph/*CLUSTER*. keyring**) 및**/etc/ceph/*CLUSTER.NAME .keyring*** *과 같은* ***RADOS*** *프로토콜을 사용하여 로컬 클러스터에 연결합니다. 여기서* ***CLUSTER*** *는 사용자에게 친숙한 클러스터 이름이고* ***NAME*** *은 예제,* ***client.ceph-immutable-object-cache*** *와 같이 연결할* ***RADOS*** *사용자입니다.*

데몬의 주요 구성 요소

ceph-immutable-object-cache 데몬에 는 다음과 같은 부분이 있습니다.

- *도메인 소켓 기반 IPC(프로세스 간 통신): 데몬은 시작 시 로컬 도메인 소켓에서 수신 대기하고 librbd 클라이언트와의 연결을 기다립니다.*
- *최근 LRU(최근 프로모션) 기반 프로모션 또는 강등 정책: 데몬은 각 캐시 파일에서 cache-hits의 메모리 내 통계를 유지합니다. 용량이 구성된 임계값에 도달하면 콜드 캐시를 강등합니다.*
- *파일 기반 캐싱 저장소: 데몬은 간단한 파일 기반 캐시 저장소를 유지 관리합니다. 승격 시 RADOS 오브젝트가 RADOS 클러스터에서 가져와 로컬 캐싱 디렉터리에 저장됩니다.*

복제된 각 RBD 이미지를 열면 librbd 는 Unix 도메인 소켓을 통해 캐시 데몬에 연결을 시도합니다. 일단 성공적으로 연결되면 librbd 는 후속 읽기 시 데몬과 조정합니다. 캐시되지 않은 읽기가 있는 경우 데몬은 RADOS 오브젝트를 로컬 캐싱 디렉터리로 승격하므로 해당 오브젝트의 다음 읽기는 캐시에서 제공됩니다. 또한 데몬은 필요에 따라 콜드 캐시 파일을 제거하도록 용량 압박을 받을 수 있도록 간단한 LRU 통계도 유지합니다.

참고

성능 향상을 위해 SSD를 기본 스토리지로 사용합니다.

7.2. CEPH-IMMUTABLE-OBJECT-CACHE 데몬 구성

ceph-immutable-object-cache 는 Ceph 클러스터 간에 RADOS 오브젝트의 오브젝트 캐시를 위한 데몬입니다.

중요

ceph-immutable-object-cache 데몬을 사용하려면 RADOS 클러스터를 연결할 수 있어야 합니다.

데몬은 오브젝트를 로컬 디렉터리로 승격합니다. 이러한 캐시 오브젝트는 향후 읽기를 처리합니다. ceph-immutable-object-cache 패키지를 설치하여 데몬을 구성할 수 있습니다.

사전 요구 사항

- *실행 중인 **Red Hat Ceph Storage** 클러스터.*
- *캐시에 대해 하나 이상의 **SSD**.*

절차

1. ***RBD** 공유 읽기 전용 상위 이미지 캐시를 활성화합니다. **/etc/ceph/ceph.conf** 파일의 **[client]** 아래에 다음 매개변수를 추가합니다.*

   예제

   ```
   [root@ceph-host01 ~]# vi /etc/ceph/ceph.conf

   [client]
   rbd parent cache enabled = true
   rbd plugins = parent_cache
   ```

   클러스터를 다시 시작합니다.

2. ***ceph-immutable-object-cache** 패키지를 설치합니다.*

   예제

   ```
   [root@ceph-host1 ~]# dnf install ceph-immutable-object-cache
   ```

3. *고유한 **Ceph** 사용자 **ID**인 인증 키를 만듭니다.*

   구문

```
ceph auth get-or-create client.ceph-immutable-object-cache.USER_NAME mon 'profile rbd'
osd 'profile rbd-read-only'
```

예제

```
[root@ceph-host1 ~]# ceph auth get-or-create client.ceph-immutable-object-cache.user mon
'profile rbd' osd 'profile rbd-read-only'

[client.ceph-immutable-object-cache.user]
 key = AQCVPH1gFgHRAhAAp8ExRIsoxQK4QSYSRoVJLw==
```

이 인증 키를 복사합니다.

4. */etc/ceph* *디렉토리에서 파일을 생성하고 인증 키를 붙여넣습니다.*

예제

```
[root@ceph-host1 ]# vi /etc/ceph/ceph.client.ceph-immutable-object-cache.user.keyring

[client.ceph-immutable-object-cache.user]
 key = AQCVPH1gFgHRAhAAp8ExRIsoxQK4QSYSRoVJLw
```

5. *데몬을 활성화합니다.*

구문

```
systemctl enable ceph-immutable-object-cache@ceph-immutable-object-cache.USER_NAME
```

***USER_NAME** 을 데몬 인스턴스로 지정합니다.*

예제

```
[root@ceph-host1 ~]# systemctl enable ceph-immutable-object-cache@ceph-immutable-object-cache.user

Created symlink /etc/systemd/system/ceph-immutable-object-cache.target.wants/ceph-immutable-object-cache@ceph-immutable-object-cache.user.service → /usr/lib/systemd/system/ceph-immutable-object-cache@.service.
```

6. ***ceph-immutable-object-cache** 데몬을 시작합니다.*

구문

```
systemctl start ceph-immutable-object-cache@ceph-immutable-object-cache.USER_NAME
```

예제

```
[root@ceph-host1 ~]# systemctl start ceph-immutable-object-cache@ceph-immutable-object-cache.user
```

검증

- *구성 상태를 확인합니다.*

구문

```
systemctl status ceph-immutable-object-cache@ceph-immutable-object-cache.USER_NAME
```

예제

```
[root@ceph-host1 ~]# systemctl status ceph-immutable-object-cache@ceph-immutable-object-cache.user

● ceph-immutable-object-cache@ceph-immutable-object-cache.user>
  Loaded: loaded (/usr/lib/systemd/system/ceph-immutable-objec>
  Active: active (running) since Mon 2021-04-19 13:49:06 IST; >
Main PID: 85020 (ceph-immutable-)
   Tasks: 15 (limit: 49451)
  Memory: 8.3M
  CGroup: /system.slice/system-ceph\x2dimmutable\x2dobject\x2d>
          └─85020 /usr/bin/ceph-immutable-object-cache -f --cl>
```

7.3. CEPH-IMMUTABLE-OBJECT-CACHE 데몬의 일반 설정

***ceph-immutable-object-cache** 데몬의 몇 가지 중요한 일반 설정이 나열됩니다.*

immutable_object_cache_sock

설명

***librbd** 클라이언트와 **ceph-immutable-object-cache** 데몬 간의 통신에 사용되는 도메인 소켓 경로입니다.*

유형

문자열

Default

/var/run/ceph/immutable_object_cache_sock

immutable_object_cache_path

설명

변경할 수 없는 오브젝트 캐시 데이터 디렉터리입니다.

유형

문자열

Default

/tmp/ceph_immutable_object_cache

immutable_object_cache_max_size

설명

변경 불가능한 캐시의 최대 크기입니다.

유형

크기

Default

1G

immutable_object_cache_watermark

설명

*캐시의 수중 표시입니다. 값은 **0**에서 **1** 사이입니다. 캐시 크기가 이 임계값에 도달하면 **LRU** 통계를 기반으로 데몬이 콜드 캐시를 삭제하기 시작합니다.*

유형

교체

Default

0.9

7.4. CEPH-IMMUTABLE-OBJECT-CACHE 데몬의 QOS 설정

ceph-immutable-object-cache 데몬은 설명된 설정을 지원하는 제한 기능을 지원합니다.

immutable_object_cache_qos_schedule_tick_min

설명

불변 오브젝트 캐시의 최소 예약 틱.

유형

밀리초

Default

50

immutable_object_cache_qos_iops_limit

설명

사용자 정의 불변 오브젝트 캐시 초당 IO 작업 제한.

유형

정수

Default

0

immutable_object_cache_qos_iops_burst

설명

사용자 정의 불변 오브젝트 캐시 IO 작업의 버스트 제한.

유형

정수

Default

0

immutable_object_cache_qos_iops_burst_seconds

설명

사용자 정의 버스트 기간(초)의 불변 오브젝트 캐시 IO 작업입니다.

유형

초

Default

1

immutable_object_cache_qos_bps_limit

설명

사용자 정의 불변 오브젝트 캐시 초당 IO 바이트 제한입니다.

유형

정수

Default

0

immutable_object_cache_qos_bps_burst

설명

불변 오브젝트 캐시 IO 바이트의 사용자 정의 버스트 제한.

유형

정수

Default

0

immutable_object_cache_qos_bps_burst_seconds

설명

원하는 읽기 작업의 버스트 제한.

유형

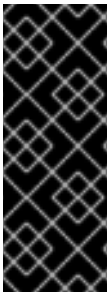
초

Default

1

8장. RBD 커널 모듈

스토리지 관리자는 **rbd** 커널 모듈을 통해 **Ceph** 블록 장치에 액세스할 수 있습니다. 블록 장치를 매핑 및 매핑 해제하고 해당 매핑을 표시할 수 있습니다. 또한 **rbd** 커널 모듈을 통해 이미지 목록을 가져올 수도 있습니다.



중요

RHEL(Red Hat Enterprise Linux) 이외의 **Linux** 배포판의 커널 클라이언트는 허용되지만 지원되지 않습니다. 이러한 커널 클라이언트를 사용할 때 스토리지 클러스터에서 문제가 발견되면 **Red Hat**은 문제를 해결하지만 근본 원인이 커널 클라이언트 측에 있는 경우 소프트웨어 벤더가 문제를 해결해야 합니다.

8.1. 사전 요구 사항

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

8.2. CEPH 블록 장치를 만들고 LINUX 커널 모듈 클라이언트에서 사용하십시오.

스토리지 관리자는 **Red Hat Ceph Storage** 대시보드에서 **Linux** 커널 모듈 클라이언트에 대한 **Ceph** 블록 장치를 생성할 수 있습니다. 시스템 관리자는 명령줄을 사용하여 해당 블록 장치를 **Linux** 클라이언트에 매핑하고 분할, 포맷 및 마운트할 수 있습니다. 그런 다음 파일을 읽고 쓸 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- **Red Hat Enterprise Linux** 클라이언트.

8.2.1. 대시보드를 사용하여 Linux 커널 모듈 클라이언트의 Ceph 블록 장치 만들기

지원하는 기능만 활성화하여 대시보드 웹 인터페이스를 사용하여 **Linux** 커널 모듈 클라이언트 전용 **Ceph** 블록 장치를 만들 수 있습니다.

커널 모듈 클라이언트는 **glance flatten, Layering, Exclusive lock, Object map, Fast diff** 등의 기능을 지원합니다.

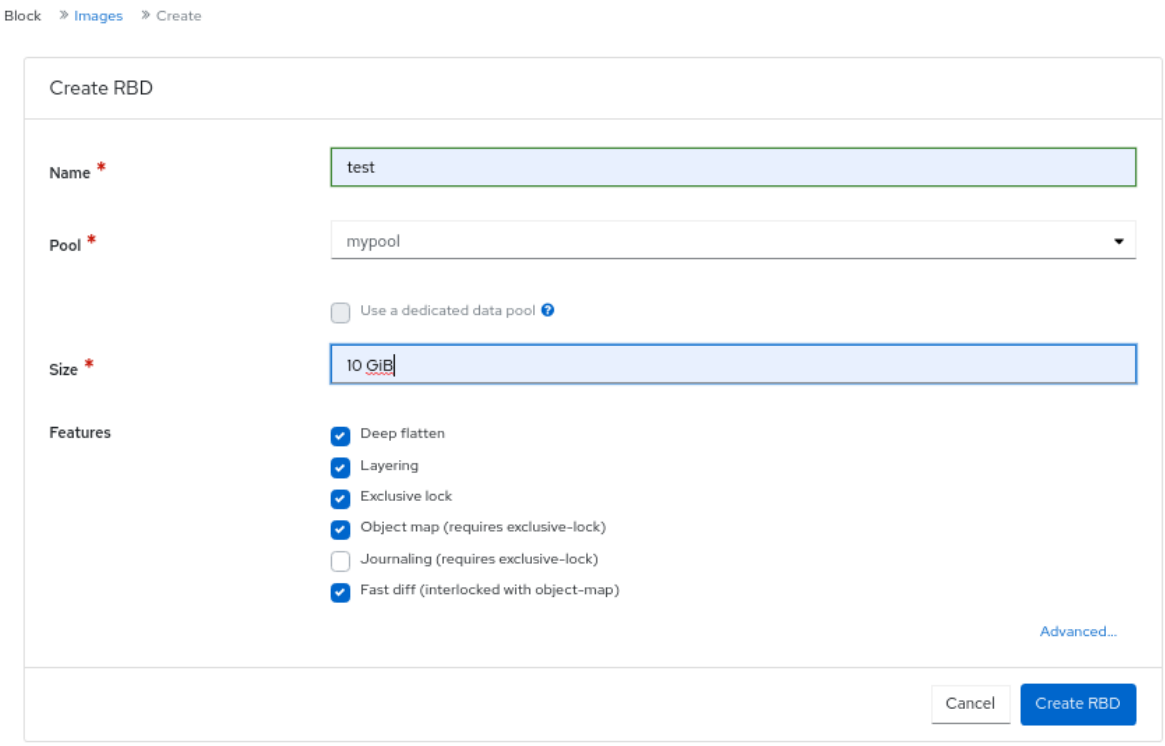
개체 맵, Fast diff 및 glance flatten 기능에는 Red Hat Enterprise Linux 8.2 이상이 필요합니다.

사전 요구 사항

- *실행 중인 Red Hat Ceph Storage 클러스터.*
- *복제된 RBD 풀이 생성 및 활성화되었습니다.*

절차

1. *Block(블록) 드롭다운 메뉴에서 Images(이미지)를 선택합니다.*
2. *생성을 클릭합니다.*
3. *RBD 생성 창에서 이미지 이름을 입력하고 RBD가 활성화된 풀을 선택한 다음 지원되는 기능을 선택합니다.*



4. *Create RBD(RBD 만들기)를 클릭합니다.*

검증

- 이미지가 성공적으로 생성되었음을 알리는 알림이 표시됩니다.

추가 리소스

- 자세한 내용은 **Red Hat Ceph Storage** 블록 장치 가이드의 명령줄을 사용하여 **Linux**의 맵 및 **Ceph** 블록 장치 마운트 를 참조하십시오.

- 자세한 내용은 **Red Hat Ceph Storage** 대시보드 가이드를 참조하십시오.

8.2.2. 명령줄을 사용하여 Linux에 Ceph 블록 장치 매핑 및 마운트

Linux rbd 커널 모듈을 사용하여 **Red Hat Enterprise Linux** 클라이언트에서 **Ceph** 블록 장치를 매핑할 수 있습니다. 매핑 후 파티션을 분할, 포맷 및 마운트할 수 있으므로 파일을 쓸 수 있습니다.

사전 요구 사항

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

- 대시보드를 사용하여 **Linux** 커널 모듈 클라이언트의 **Ceph** 블록 장치가 생성됩니다.

- **Red Hat Enterprise Linux** 클라이언트.

절차

1. **Red Hat Enterprise Linux** 클라이언트 노드에서 **Red Hat Ceph Storage 5 Tools** 리포지토리를 활성화합니다.

   Red Hat Enterprise Linux 8

   ```
   [root@rbd-client ~]# subscription-manager repos --enable=rhceph-5-tools-for-rhel-8-x86_64-rpms
   ```

2. *ceph-common RPM 패키지를 설치합니다.*

Red Hat Enterprise Linux 8

```
[root@rbd-client ~]# dnf install ceph-common
```

3. *모니터 노드에서 클라이언트 노드로 **Ceph** 구성 파일을 복사합니다.*

구문

```
scp root@MONITOR_NODE:/etc/ceph/ceph.conf /etc/ceph/ceph.conf
```

예제

```
[root@rbd-client ~]# scp root@cluster1-node2:/etc/ceph/ceph.conf /etc/ceph/ceph.conf
root@192.168.0.32's password:
ceph.conf                                     100%  497   724.9KB/s   00:00
[root@client1 ~]#
```

4. *모니터 노드에서 클라이언트 노드로 키 파일을 복사합니다.*

구문

```
scp root@MONITOR_NODE:/etc/ceph/ceph.client.admin.keyring
/etc/ceph/ceph.client.admin.keyring
```

예제

```
[root@rbd-client ~]# scp root@cluster1-node2:/etc/ceph/ceph.client.admin.keyring
/etc/ceph/ceph.client.admin.keyring
root@192.168.0.32's password:
ceph.client.admin.keyring                                    100%  151   265.0KB/s   00:00
[root@client1 ~]#
```

5. *이미지를 매핑합니다.*

구문

```
rbd map --pool POOL_NAME IMAGE_NAME --id admin
```

예제

```
[root@rbd-client ~]# rbd map --pool block-device-pool image1 --id admin
/dev/rbd0
[root@client1 ~]#
```

6. *블록 장치에 파티션 테이블을 생성합니다.*

구문

```
parted /dev/MAPPED_BLOCK_DEVICE mklabel msdos
```

예제

```
[root@rbd-client ~]# parted /dev/rbd0 mklabel msdos
Information: You may need to update /etc/fstab.
```

7. ***XFS* 파일 시스템의 파티션을 생성합니다.**

구문

```
parted /dev/MAPPED_BLOCK_DEVICE mkpart primary xfs 0% 100%
```

예제

```
[root@rbd-client ~]# parted /dev/rbd0 mkpart primary xfs 0% 100%
Information: You may need to update /etc/fstab.
```

8. **파티션을 포맷합니다.**

구문

```
mkfs.xfs /dev/MAPPED_BLOCK_DEVICE_WITH_PARTITION_NUMBER
```

예제

```
[root@rbd-client ~]# mkfs.xfs /dev/rbd0p1
meta-data=/dev/rbd0p1            isize=512    agcount=16, agsize=163824 blks
         =                       sectsz=512   attr=2, projid32bit=1
         =                       crc=1        finobt=1, sparse=1, rmapbt=0
         =                       reflink=1
data     =                       bsize=4096   blocks=2621184, imaxpct=25
         =                       sunit=16     swidth=16 blks
naming   =version 2              bsize=4096   ascii-ci=0, ftype=1
log      =internal log           bsize=4096   blocks=2560, version=2
         =                       sectsz=512   sunit=16 blks, lazy-count=1
realtime =none                   extsz=4096   blocks=0, rtextents=0
```

9. *새 파일 시스템을 마운트할 디렉토리를 만듭니다.*

구문

```
mkdir PATH_TO_DIRECTORY
```

예제

```
[root@rbd-client ~]# mkdir /mnt/ceph
```

10. *파일 시스템을 마운트합니다.*

구문

```
mount /dev/MAPPED_BLOCK_DEVICE_WITH_PARTITION_NUMBER
PATH_TO_DIRECTORY
```

예제

```
[root@rbd-client ~]# mount /dev/rbd0p1 /mnt/ceph/
```

11. *파일 시스템이 마운트되어 올바른 크기를 표시하는지 확인합니다.*

구문

```
df -h PATH_TO_DIRECTORY
```

예제

```
[root@rbd-client ~]# df -h /mnt/ceph/
Filesystem      Size  Used Avail Use% Mounted on
/dev/rbd0p1      10G  105M  9.9G   2% /mnt/ceph
```

추가 리소스

- *자세한 내용은 대시보드를 사용하여 **Linux** 커널 모듈 클라이언트의 **Ceph** 블록 장치 생성을 참조하십시오.*

- 자세한 내용은 **Red Hat Enterprise Linux 8**용 파일 시스템 관리를 참조하십시오.
- 자세한 내용은 **Red Hat Enterprise Linux 7**용 스토리지 관리 가이드를 참조하십시오.

8.3. 이미지 목록 가져오기

Ceph 블록 장치 이미지 목록을 가져옵니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 블록 장치 이미지를 마운트하려면 먼저 이미지 목록을 반환합니다.

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

8.4. 블록 장치 매핑

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

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 이미지 이름을 커널 모듈에 매핑합니다.

구문

```
rbd device map POOL_NAME/IMAGE_NAME --id USER_NAME
```

예제

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

2. *인증 키 또는 보안이 포함된 파일에 의해 **cephx** 인증을 사용할 때 보안을 지정합니다.*

구문

```
[root@rbd-client ~]# rbd device map POOL_NAME/IMAGE_NAME --id USER_NAME --
keyring PATH_TO_KEYRING
```

또는

```
[root@rbd-client ~]# rbd device map POOL_NAME/IMAGE_NAME --id USER_NAME --
keyfile PATH_TO_FILE
```

8.5. 매핑된 블록 장치 표시

***rbd** 명령을 사용하여 커널 모듈에 매핑되는 블록 장치 이미지를 표시할 수 있습니다.*

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 매핑된 블록 장치를 표시합니다.

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

8.6. 블록 장치 매핑 해제

unmap 옵션을 사용하고 장치 이름을 제공하여 **rbd** 명령으로 블록 장치 이미지를 매핑 해제할 수 있습니다.

사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. 블록 장치 이미지를 매핑 해제합니다.

구문

```
rbd device unmap /dev/rbd/POOL_NAME/IMAGE_NAME
```

예제

```
[root@rbd-client ~]# rbd device unmap /dev/rbd/pool1/image1
```

9장. CEPH 블록 장치 PYTHON 모듈 사용

rbd python 모듈은 Ceph 블록 장치 이미지에 대해 파일과 유사한 액세스를 제공합니다. 기본 제공 툴을 사용하려면 rbd 및 rados Python 모듈을 가져옵니다.

사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.
- 노드에 대한 루트 수준 액세스.

절차

1. RADOS에 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/O를 수행하려면 :class:rbd.Image 오브젝트를 인스턴스화합니다.

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

이렇게 하면 이미지의 처음 600바이트에 'foo'가 기록됩니다. 데이터는 :type:unicode 일 수 없습니다. librbd 는 :c:type:char 보다 넓은 문자를 처리하는 방법을 알지 못합니다.

4. 이미지를 닫고 IO 컨텍스트 및 RADOS에 대한 연결을 종료합니다.

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

*안전하게 하려면 이러한 호출 각각이 별도의 **:finally** 블록에 있어야 합니다.*

```
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()
```

*이 작업은 번거로울 수 있으므로 **Rados,Ioctx** 및 **Image** 클래스를 자동으로 닫거나 종료하는 컨텍스트 관리자로 사용할 수 있습니다. 이를 컨텍스트 관리자로 사용하면 위의 예가 됩니다.*

```
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)
```

10장. CEPH ISCSI 게이트웨이(LIMITED AVAILABILITY)

스토리지 관리자는 **Red Hat Ceph Storage** 클러스터의 **iSCSI** 게이트웨이를 설치하고 구성할 수 있습니다. **Ceph**의 **iSCSI** 게이트웨이를 사용하면 기존 **SAN(Storage Area Network)**의 모든 기능과 이점을 갖춘 완전히 통합된 블록 스토리지 인프라를 효과적으로 실행할 수 있습니다.

참고

이 기술은 제한적인 가용성입니다. 자세한 내용은 사용 중단된 기능 장을 참조하십시오.

주의

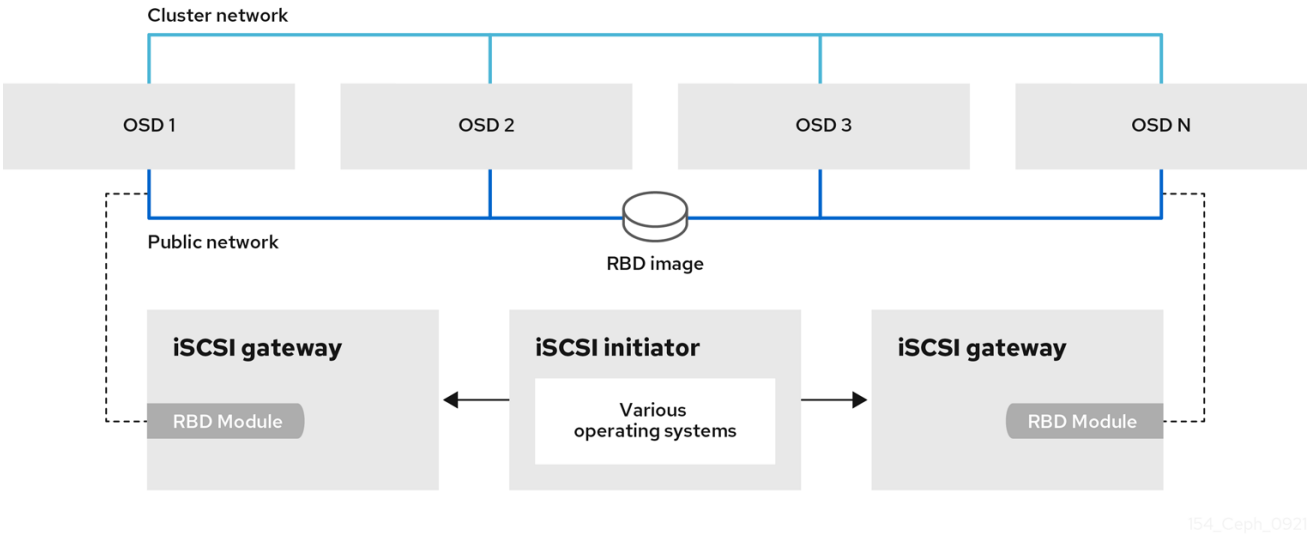
SCSI 영구 예약은 지원되지 않습니다. **SCSI** 영구 예약을 사용하지 않는 클러스터 인식 파일 시스템 또는 클러스터링 소프트웨어를 사용하는 경우 여러 **iSCSI** 이니시에이터를 **RBD** 이미지에 매핑하는 기능이 지원됩니다. 예를 들어 **ATS**를 사용하는 **VMware vSphere** 환경은 지원되지만 **Microsoft**의 클러스터링 서버(**MSCS**) 사용은 지원되지 않습니다.

10.1. CEPH ISCSI 게이트웨이 소개

일반적으로 **Ceph** 스토리지 클러스터에 대한 블록 수준 액세스는 **QEMU** 및 **librbd** 로 제한되어 있으며, 이는 **OpenStack** 환경 내에서 채택하기 위한 핵심 요소입니다. **Ceph** 스토리지 클러스터에 대한 블록 수준의 액세스는 이제 **iSCSI** 표준을 활용하여 데이터 스토리지를 제공할 수 있습니다.

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

그림 10.1. Ceph iSCSI 게이트웨이



10.2. iSCSI 대상의 요구 사항

Red Hat Ceph Storage HA(고가용성) iSCSI 게이트웨이 솔루션에는 OSD를 감지하기 위한 게이트웨이 노드 수, 메모리 용량 및 타이머 설정에 대한 요구 사항이 있습니다.

필요한 노드 수

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

메모리 요구 사항

RBD 이미지의 메모리 공간이 큰 크기로 증가할 수 있습니다. iSCSI 게이트웨이 호스트에 매핑된 각 RBD 이미지는 약 90MB의 메모리를 사용합니다. iSCSI 게이트웨이 호스트에 매핑된 각 RBD 이미지를 지원할 수 있는 충분한 메모리가 있는지 확인합니다.

다운 OSD 감지

Ceph 모니터 또는 OSD에 대한 특정 iSCSI 게이트웨이 옵션은 없지만 이니시에이터 시간 초과 가능성을 줄이기 위해 OSD를 감지하기 위한 기본 타이머를 낮추는 것이 중요합니다.

추가 리소스

- 자세한 내용은 Red Hat Ceph Storage Hardware Selection Guide 를 참조하십시오.

10.3. iSCSI 게이트웨이 설치

스토리지 관리자는 Ceph iSCSI 게이트웨이의 이점을 활용하기 전에 필수 소프트웨어 패키지를 설치해야 합니다. 명령줄 인터페이스를 사용하여 Ceph iSCSI 게이트웨이를 설치할 수 있습니다.

각 iSCSI 게이트웨이는 iSCSI 프로토콜 지원을 제공하기 위해 Linux I/O(대상 커널 하위 시스템)를 실행합니다. LIO는 사용자 공간 통과(TCMU)를 활용하여 Ceph librbd 라이브러리와 상호 작용하여 RBD 이미지를 iSCSI 클라이언트에 노출합니다. Ceph iSCSI 게이트웨이를 사용하면 기존 SAN(Storage Area Network)의 모든 기능과 이점을 갖춘 완전히 통합된 블록 스토리지 인프라를 효과적으로 실행할 수 있습니다.

10.3.1. 사전 요구 사항

- Red Hat Enterprise Linux 8.4 이상.
- 실행 중인 Red Hat Ceph Storage 5 이상 클러스터.

10.3.2. 명령줄 인터페이스를 사용하여 Ceph iSCSI 게이트웨이 설치

Ceph iSCSI 게이트웨이는 iSCSI 대상 노드이자 Ceph 클라이언트 노드이기도 합니다. Ceph iSCSI 게이트웨이는 독립 실행형 노드이거나 Ceph OSD(오브젝트 저장소 디스크) 노드에 함께 배치될 수 있습니다. Ceph iSCSI 게이트웨이를 설치하려면 다음 단계를 완료합니다.

사전 요구 사항

- Red Hat Enterprise Linux 8.4 이상
- Red Hat Ceph Storage 5 클러스터 이상
- Ceph iSCSI 게이트웨이가 OSD 노드에 배치되지 않은 경우 스토리지 클러스터에서 실행 중인 Ceph 노드에서 모든 iSCSI 게이트웨이 호스트로 /etc/ceph/ 디렉터리에 있는 Ceph 구성 파일을 복사합니다. Ceph 구성 파일은 /etc/ceph/ 의 iSCSI 게이트웨이 호스트에 있어야 합니다.
- 모든 Ceph iSCSI 게이트웨이 호스트에서 Ceph 툴 리포지토리를 활성화합니다.
- 모든 Ceph iSCSI 게이트웨이 호스트에서 Ceph 명령줄 인터페이스를 설치 및 구성합니다.

- *필요한 경우 모든 **Ceph iSCSI** 노드의 방화벽에서 **TCP** 포트 **3260** 및 **5000**을 엽니다.*
- *새 **RADOS** 블록 장치(**RBD**)를 새로 만들거나 기존 **RADOS** 블록 장치(**RBD**)를 사용합니다.*

절차

1. *호스트에서 **iSCSI** 컨테이너의 정보를 검색합니다.*

   예제

   ```
   [root@iscsigw ~]# podman ps
   ```

2. ***Cephadm** 쉘에 로그인합니다.*

   예제

   ```
   [root@iscsigw ~]# cephadm shell
   ```

3. *선택 사항: 필요한 경우 모든 **Ceph iSCSI** 게이트웨이 호스트에서 **OpenSSL** 유틸리티를 설치 및 구성합니다.*

   a. ***openssl** 패키지를 설치합니다.*

      예제

      ```
      [ceph: root@iscsigw /]# yum install openssl
      ```

b.

*기본 **iSCSI** 게이트웨이 노드에서 **SSL** 키를 보유할 디렉터리를 생성합니다.*

예제

```
[ceph: root@iscsigw /]# mkdir ~/ssl-keys
[ceph: root@iscsigw /]# cd ~/ssl-keys
```

c.

*기본 **iSCSI** 게이트웨이 노드에서 인증서와 키 파일을 만듭니다. 메시지가 표시되면 환경 정보를 입력합니다.*

예제

```
[ceph: root@iscsigw /]# openssl req -newkey rsa:2048 -nodes -keyout iscsi-gateway.key
-x509 -days 365 -out iscsi-gateway.crt
```

d.

*기본 **iSCSI** 게이트웨이 노드에서 **PEM** 파일을 생성합니다.*

예제

```
[ceph: root@iscsigw /]# cat iscsi-gateway.crt iscsi-gateway.key > iscsi-gateway.pem
```

e.

*기본 **iSCSI** 게이트웨이 노드에서 공개 키를 생성합니다.*

예제

```
[ceph: root@iscsigw /]# openssl x509 -inform pem -in iscsi-gateway.pem -pubkey -noout
> iscsi-gateway-pub.key
```

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

4. ***Ceph iSCSI** 게이트웨이 노드에 구성 파일을 만듭니다.*

a. */**etc**/**ceph**/* 디렉터리에 ***iscsi-gateway.yaml*** 이라는 파일을 생성합니다.

예제

```
[ceph: root@iscsigw /]# touch /etc/ceph/iscsi-gateway.yaml
```

b. ***iscsi-gateway.yaml*** *파일을 편집하고 다음 행을 추가합니다.*

구문

```
service_type: iscsi
service_id: iscsi
placement:
  hosts:
    - HOST_NAME
    - HOST_NAME
spec:
  pool: POOL_NAME  # RADOS pool where ceph-iscsi config data is stored.
  trusted_ip_list: "IP_ADDRESS_1,IP_ADDRESS_2"
```

예제

```
service_type: iscsi
service_id: iscsi
placement:
  hosts:
  - host01
  - host02
spec:
  pool: iscsipool1
  trusted_ip_list: "10.80.100.100,10.8.100.113"
```

5. */etc/ceph/* 로 경로를 변경하고 다음 명령을 사용하여 사양을 적용합니다.

예제

```
[ceph: root@iscsigw /]# ceph orch apply -i iscsi-gateway.yaml
```

6. 다음으로 타겟, LUN 및 클라이언트를 구성합니다. 자세한 내용은 *명령줄 인터페이스를 사용하여 iSCSI 대상 구성* 섹션을 참조하십시오.

추가 리소스

- 옵션에 대한 자세한 내용은 *iSCSI 게이트웨이 변수* 섹션을 참조하십시오.
- *Ceph 대시보드에서 iSCSI 대상 만들기*

10.4. ISCSI 대상 구성

스토리지 관리자는 **gwcli** 명령줄 유틸리티를 사용하여 타겟, **LUN** 및 클라이언트를 구성할 수 있습니다. **iSCSI** 타겟의 성능을 최적화하고 **gwcli reconfigure** 하위 명령을 사용할 수도 있습니다.

주의

Red Hat은 **gwcli** 와 같은 **Ceph iSCSI** 게이트웨이 툴에서 내보낸 **Ceph** 블록 장치 이미지 관리를 지원하지 않습니다. 또한 **rbd** 명령을 사용하여 **Ceph iSCSI** 게이트웨이에서 내보낸 **RBD** 이미지의 이름을 바꾸거나 제거하면 스토리지 클러스터가 불안정해질 수 있습니다.

주의

iSCSI 게이트웨이 구성에서 **RBD** 이미지를 제거하기 전에 운영 체제에서 스토리지 장치를 제거하기 위한 표준 절차를 따르십시오. 자세한 내용은 **Red Hat Enterprise Linux 7**용 스토리지 관리 가이드 의 스토리지 장치 제거 장 또는 **Red Hat Enterprise Linux 8**용 시스템 설계 가이드 를 참조하십시오.

10.4.1. 사전 요구 사항

- **Ceph iSCSI** 게이트웨이 소프트웨어 설치.

10.4.2. 명령줄 인터페이스를 사용하여 iSCSI 대상 구성

Ceph iSCSI 게이트웨이는 **iSCSI** 대상 노드이자 **Ceph** 클라이언트 노드이기도 합니다. 독립 실행형 노드에서 **Ceph iSCSI** 게이트웨이를 구성하거나 **Ceph OSD**(오브젝트 스토리지 장치) 노드와 함께 배치합니다.

주의

*이 문서 또는 **Red Hat** 지원 센터에 명시되어 있지 않는 한 **gwcli reconfigure** 하위 명령을 사용하여 다른 옵션을 조정하지 마십시오.*

사전 요구 사항

- ***Ceph iSCSI** 게이트웨이 소프트웨어 설치.*

절차

1. *호스트에서 실행 중인 **iSCSI** 컨테이너의 정보를 검색합니다.*

   예제

```
[root@iscsigw ~]# podman ps
[root@iscsigw ~]# podman exec -it 4b5ffb814409 /bin/bash
```

2. ***iSCSI** 게이트웨이 명령줄 인터페이스를 시작합니다.*

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

3. ***iscsi-targets** 디렉터리로 이동합니다.*

   예제

```
/>cd /iscsi-targets
```

4. ***IPv4** 또는 **IPv6** 주소를 사용하여 **iSCSI** 게이트웨이를 생성합니다.*

구문

```
/>iscsi-targets 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-targets 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
```

5. ***Ceph** 블록 장치를 추가합니다.*

구문

```
> cd /disks
/>disks/ create POOL_NAME image=IMAGE_NAME size=IMAGE_SIZE_m|g|t
```

예제

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

참고

풀 또는 이미지 이름에 마침표(.)를 사용하지 마십시오.

6. *클라이언트를 생성합니다.*

구문

```
> goto hosts
> create iqn.1994-05.com.redhat:_client_name_
> auth username=USER_NAME password=PASSWORD
```

예제

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

중요

***Red Hat**은 클라이언트 혼합을 지원하지 않으며 일부 **CHAP**가 비활성화되어 있습니다. 모든 클라이언트는 **CHAP**를 활성화하거나 **CHAP**를 비활성화해야 합니다. 기본 동작은 이니시에이터 이름으로만 이니시에이터를 인증하는 것입니다.*

*이니시에이터가 대상에 로그인하지 못하면 일부 이니시에이터에 대해 **CHAP** 인증이 올바르게 구성되지 않을 수 있습니다. 예를 들면 다음과 같습니다.*

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

***hosts** 수준에서 다음 명령을 사용하여 모든 **CHAP** 인증을 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)]
```

7. *클라이언트에 디스크를 추가합니다.*

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
```

8.

***Ceph ISCSI** 게이트웨이가 작동하는지 확인합니다.*

```
/> 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)]
```

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

9.

*선택적으로 **max_data_area_mb** 옵션을 재구성하십시오.*

구문

```
/>disks/ reconfigure POOL_NAME/IMAGE_NAME max_data_area_mb NEW_BUFFER_SIZE
```

예제

```
/>disks/ reconfigure rbd/disk_1 max_data_area_mb 64
```

참고

max_data_area_mb 옵션은 각 이미지가 **iSCSI** 대상과 **Ceph** 클러스터 간에 **SCSI** 명령 데이터를 전달하는 데 사용할 수 있는 메가바이트 단위의 메모리 양을 제어합니다. 이 값이 너무 작으면 과도한 대기열을 완전히 재시도하여 성능에 영향을 줄 수 있습니다. 값이 너무 크면 시스템 메모리를 너무 많이 사용하는 하나의 디스크가 생성되어 다른 하위 시스템에 할당 오류가 발생할 수 있습니다. **max_data_area_mb** 옵션의 기본값은 **8** 입니다.

10. **iSCSI** 이니시에이터 구성.

추가 리소스

- 자세한 내용은 **iSCSI** 게이트웨이 설치를 참조하십시오.
- 자세한 내용은 **iSCSI** 이니시에이터 구성 섹션을 참조하십시오.

10.4.3. iSCSI 대상의 성능 최적화

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

주의

Red Hat 지원 부서에 지시하거나 이 문서에 명시된 대로만 설정을 변경합니다.

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

iSCSI 대상의 성능에 영향을 주는 설정

max_data_area_mb

설명

커널 데이터 링 버퍼 크기(MB)입니다.

유형

정수

Default

8

cmdsn_depth

설명

최대 I/O를 제어하는 큐의 깊이를 나타냅니다.

유형

정수

Default

128

immediate_data

설명

*이니시에이터에서 새 세션을 설정할 때마다 즉시 데이터를 전송하도록 타겟에서 권한을 요청하는지 나타냅니다. 이 값이 **Yes** 이면 이니시에이터에서 새 세션을 설정할 때마다 즉각적인 데이터를 전송하도록 타겟의 권한을 요청합니다.*

유형

부울

Default

있음

initial_r2t

설명

*새 세션을 설정할 때마다 **HBA(호스트 버스 어댑터)** 이니시에이터가 타겟에서 권한을 요청하여 원하지 않는 **SCSI** 데이터를 전송할지 여부를 나타냅니다. 이 멤버가 **Yes** 이면 **HBA** 이니시에

*이터가 새 세션을 설정할 때마다 의도하지 않은 **SCSI** 데이터를 전송하도록 타겟에서 권한을 요청합니다.*

유형

부울

Default

있음

max_outstanding_r2t

설명

*작업을 시작하는 첫 번째 **R2T**를 제외하고 각 작업에 대한 **R2T**(전송 준비) 요청의 최대 수입니다.*

유형

정수

Default

1

first_burst_length

설명

*단일 **SCSI** 명령을 실행하는 동안 **iSCSI** 이니시에이터에서 타겟으로 보낼 수 있는 원하지 않는 최대 데이터 양입니다.*

유형

바이트 단위의 정수

Default

262144

max_burst_length

설명

*입력 **PDU** 시퀀스 또는 요청된 출력 **PDU** 시퀀스의 최대 **SCSI** 데이터 페이로드입니다.*

유형

바이트 단위의 정수

Default

524288

max_recv_data_segment_length

설명

이니시에이터가 타겟의 **iSCSI PDU**에서 수신할 수 있는 최대 데이터 바이트 수입니다.

유형

바이트 단위의 정수

Default

262144

max_xmit_data_segment_length

설명

이니시에이터가 **iSCSI PDU**에서 대상에 전송하는 최대 데이터 바이트 수입니다.

유형

바이트 단위의 정수

Default

0

추가 리소스

- **gwcli** 재구성 을 사용하여 조정하는 방법을 보여주는 예제를 포함하여 **max_data_area_mb** 에 대한 정보는 명령줄 인터페이스를 사용하여 **iSCSI** 대상 구성 섹션에 있습니다.

10.4.4. 명령줄 인터페이스를 사용하여 iSCSI 호스트 그룹 구성

Ceph iSCSI 게이트웨이는 동일한 디스크 구성을 공유하는 여러 서버를 관리하도록 호스트 그룹을 구성할 수 있습니다. **iSCSI** 호스트 그룹은 그룹의 각 호스트에서 액세스할 수 있는 호스트와 디스크의 논리

적 그룹을 생성합니다.

중요

여러 호스트에 디스크 장치를 공유하려면 클러스터를 인식하는 파일 시스템을 사용해야 합니다.

사전 요구 사항

- **Ceph iSCSI** 게이트웨이 소프트웨어 설치.
- **Ceph iSCSI** 게이트웨이 노드에 대한 루트 수준 액세스.

절차

1. 호스트에서 실행 중인 **iSCSI** 컨테이너의 정보를 검색합니다.

예제

```
[root@iscsigw ~] podman ps
CONTAINER ID  IMAGE                                    COMMAND   CREATED    STATUS
PORTS  NAMES
4b5ffb814409  registry.redhat.io/rhceph-alpha/rhceph-5-rhel8:latest    2 hours ago         Up 2
hours ago ceph-f838eb7a-597c-11eb-b0a9-525400e2439c-iscsi.iscsi.cephLab2-node-
01.anaahg
```

2. **iSCSI** 컨테이너 **ID**를 사용하여 컨테이너에 들어갑니다.

예제

```
[root@iscsigw ~]# podman exec -it 4b5ffb814409 /bin/bash
```

3. ***gwcli** 명령을 실행합니다.*

```
[ceph: root@iscsigw /]# gwcli
```

4. *새 호스트 그룹을 생성합니다.*

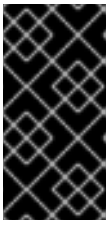
구문

```
cd iscsi-targets/
cd IQN/host-groups
create group_name=GROUP_NAME
```

예제

```
/> cd iscsi-targets/
/iscsi-targets> cd iqn.2003-01.com.redhat.iscsi-gw:ceph-igw/host-groups/
/iscsi-target.../host-groups> create group_name=igw_grp01
```

5. *호스트 그룹에 호스트를 추가합니다.*



중요

*호스트 그룹 **otheriwse**에 호스트를 추가하기 전에 호스트에 추가된 모든 디스크를 제거했는지 확인합니다. 호스트를 호스트 그룹에 추가할 수 없습니다.*

구문

```
cd GROUP_NAME
host add client_iqn=CLIENT_IQN
```

예제

```
> cd igw_grp01
/iscsi-target.../host-groups/igw_grp01> host add client_iqn=iqn.1994-05.com.redhat:rh8-client
```

이 단계를 반복하여 그룹에 호스트를 추가합니다.

6. *호스트 그룹에 디스크를 추가합니다.*

구문

```
cd /disks/
/disks> create pool=POOL image=IMAGE_NAME size=SIZE
cd /IQN/host-groups/GROUP_NAME
disk add POOL/IMAGE_NAME
```

예제

```
> cd /disks/
/disks> create pool=rbd image=rbdimage size=1G
/> cd iscsi-targets/iqn.2003-01.com.redhat.iscsi-gw:ceph-igw/host-groups/igw_grp01/
/iscsi-target...s/igw_grp01> disk add rbd/rbdimage
```

이 단계를 반복하여 그룹에 디스크를 추가합니다.

10.4.5. 추가 리소스

- Red Hat Ceph Storage 대시보드를 사용하여 iSCSI 대상을 구성하는 방법에 대한 자세한 내용은 Red Hat Ceph Storage 대시보드 가이드의 iSCSI 대상 만들기 섹션을 참조하십시오.

10.5. ISCSI 이니시에이터 구성

다음 플랫폼에서 Ceph iSCSI 게이트웨이에 연결하도록 iSCSI 이니시에이터를 구성할 수 있습니다.

- Red Hat Enterprise Linux
- Red Hat Virtualization
- Microsoft Windows Server 2016
- VMware ESXi

10.5.1. Red Hat Enterprise Linux용 iSCSI 이니시에이터 구성

사전 요구 사항

- Red Hat Enterprise Linux 7.7 이상.
- 패키지 iscsi-initiator-utils-6.2.0.873-35 이상을 설치해야 합니다.
- device-mapper-multipath-0.4.9-99 이상을 설치해야 합니다.

절차

1. iSCSI 이니시에이터 및 다중 경로 툴을 설치합니다.

```
[root@rhel ~]# yum install iscsi-initiator-utils
[root@rhel ~]# yum install device-mapper-multipath
```

2. ***/etc/iscsi/initiatorname.iscsi** 파일을 편집하여 이니시에이터 이름을 설정합니다. 이니시에이터 이름은 **gwcli** 명령을 사용하여 초기 설정 중에 사용된 이니시에이터 이름과 일치해야 합니다.*

3. *다중 경로 **I/O** 구성.*

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

   ```
   [root@rhel ~]# mpathconf --enable --with_multipathd y
   ```

   b. *다음과 같이 **/etc/multipath.conf** 파일을 업데이트합니다.*

   ```
   devices {
         device {
               vendor                "LIO-ORG"
               product               "TCMU device"
               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
         }
   }
   ```

   c. ***multipathd** 서비스를 다시 시작합니다.*

   ```
   [root@rhel ~]# systemctl reload multipathd
   ```

4. ***CHAP** 및 **iSCSI** 검색 및 로그인 설정.*

   a. */etc/iscsi/iscsid.conf* 파일을 적절하게 업데이트하여 **CHAP** 사용자 이름과 암호를 제공합니다. 예를 들면 다음과 같습니다.

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

b. *대상 포털을 검색합니다.*

구문

```
iscsiadm -m discovery -t st -p IP_ADDR
```

c. *대상에 로그인합니다.*

구문

```
iscsiadm -m node -T TARGET -l
```

5. *다중 경로 **I/O** 구성을 봅니다. **multipathd** 데몬은 **multipath.conf** 파일의 설정을 기반으로 장치를 자동으로 설정합니다.*

a. ***multipath** 명령을 사용하여 각 경로의 우선 순위 그룹이 있는 장애 조치 구성에서 장치 설정을 표시합니다. 예를 들면 다음과 같습니다.*

예제

```
[root@rhel ~]# multipath -ll
mpathbt (360014059ca317516a69465c883a29603) dm-1 LIO-ORG,TCMU device
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 상태의 iSCSI 게이트웨이 경로임을 나타내고 prio=10 은 Active-non-Optimized 경로임을 나타냅니다. status 필드는 사용 중인 경로를 나타냅니다. 여기서 active 는 현재 사용된 경로를 나타내며, enabled 는 활성 이 실패하는 경우 페일오버 경로를 나타냅니다.

b.

장치 이름(예: multipath -ll 출력의 sde)을 iSCSI 게이트웨이에 맞추려면 다음을 수행합니다.

예제

```
[root@rhel ~]# iscsiadm -m session -P 3
```

영구 포털 값은 gwcli 유틸리티에 나열된 iSCSI 게이트웨이에 할당된 IP 주소입니다.

10.5.2. Red Hat Virtualization의 iSCSI 이니시에이터 구성

사전 요구 사항

- *Red Hat Virtualization 4.1*
- *모든 Red Hat Virtualization 노드에 구성된 MPIO 장치*
- *iscsi-initiator-utils-6.2.0.873-35 패키지 이상*
- *device-mapper-multipath-0.4.9-99 패키지 이상*

절차

1. *다중 경로 **I/O** 구성.*

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

   ```
   [root@rhv ~]# mpathconf --enable --with_multipathd y
   ```

   b. *다음과 같이 **/etc/multipath.conf** 파일을 업데이트합니다.*

   ```
   devices {
         device {
               vendor                "LIO-ORG"
               product               "TCMU device"
               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
         }
   }
   ```

   c. ***multipathd** 서비스를 다시 시작합니다.*

   ```
   [root@rhv ~]# systemctl reload multipathd
   ```

2. ***Storage** (스토리지) 리소스 탭을 클릭하여 기존 스토리지 도메인을 나열합니다.*

3. ***New Domain**(새 도메인) 버튼을 클릭하여 **New Domain**(새 도메인) 창을 엽니다.*

4. *새 스토리지 도메인의 **Name** (이름)을 입력합니다.*

5. ***Data Center** (데이터 센터) 드롭다운 메뉴를 사용하여 데이터 센터를 선택합니다.*

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

니다.

7.

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

8.

***iSCSI**를 스토리지 유형으로 선택하면 **New Domain** (새 도메인) 창에서 사용되지 않는 **LUN**을 사용하여 알려진 타겟을 자동으로 표시합니다. 에서 스토리지를 추가하는 대상이 나열되지 않은 경우 대상 검색을 사용하여 찾을 수 있습니다. 그렇지 않으면 다음 단계로 진행합니다.*

a.

***Discover Targets** (대상 검색)를 클릭하여 대상 검색 옵션을 활성화합니다. 대상이 검색되어 에 로그인되면 **New Domain**(새 도메인) 창에 환경에서 사용하지 않는 **LUN**이 있는 대상이 자동으로 표시됩니다. 환경 외부의 **LUN**도 표시됩니다. **Discover Targets**(대상 검색) 옵션을 사용하여 여러 대상에 **LUN**을 추가하거나 동일한 **LUN**에 여러 경로를 추가할 수 있습니다.*

b.

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

c.

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

d.

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

e.

***Discover**(검색) 버튼을 클릭합니다.*

f.

*검색 결과에서 사용할 대상을 선택하고 **Login**(로그인) 버튼을 클릭합니다. 또는 **Login All** (로그인 모두)을 클릭하여 검색된 모든 대상에 로그인합니다.*

중요

두 개 이상의 경로 액세스가 필요한 경우 필요한 모든 경로를 통해 대상을 검색하고 로그인해야 합니다. 스토리지 도메인을 변경하여 추가 경로를 추가하는 것은 현재 지원되지 않습니다.

9.

원하는 대상 옆에 있는 **+** 버튼을 클릭합니다. 그러면 항목이 확장되고 대상에 연결된 사용하지 않는 모든 **LUN**이 표시됩니다.

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

11. 선택적으로 고급 매개 변수를 구성할 수 있습니다.

    a. **Advanced Parameters (**고급 매개 변수)를 클릭합니다.

    b. **Warninglow Space Indicator(**낮은 공간 표시기) 필드에 백분율 값을 입력합니다. 스토리지 도메인에서 사용 가능한 여유 공간이 이 백분율 미만인 경우 경고 메시지가 사용자에게 표시되고 기록됩니다.

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

    d. **Wipe after Delete(**삭제 후 **Wipe after Delete)** 확인란을 선택하여 삭제 후 지우기 옵션을 활성화합니다. 도메인을 생성한 후 이 옵션을 편집할 수 있지만, 이렇게 하면 이미 존재하는 디스크의 삭제 속성 후에는 **wipe** 가 변경되지 않습니다.

    e. 삭제 후 삭제 옵션을 활성화하려면 **Discard after Delete(**삭제 후 카드 비활성화) 확인란을 선택합니다. 도메인을 생성한 후 이 옵션을 편집할 수 있습니다. 이 옵션은 블록 스토리지 도메인에서만 사용할 수 있습니다.

12. **OK(**확인)를 클릭하여 스토리지 도메인을 생성하고 창을 닫습니다.

10.5.3. Microsoft Windows용 iSCSI 이니시에이터 구성

사전 요구 사항

- **Microsoft Windows Server 2016**

절차

1.

iSCSI 이니시에이터를 설치하고 검색 및 설정을 구성합니다.

a.

iSCSI 이니시에이터 드라이버 및 MPIO 툴을 설치합니다.

b.

MPIO 프로그램을 시작하고 Discover Multi-Paths(다중 경로 검색) 탭을 클릭하고 Add (추가)를 클릭합니다.

c.

MPIO 프로그램을 다시 부팅합니다.

d.

iSCSI Initiator 속성 창의 검색 탭에 대상 포털을 추가합니다. Ceph iSCSI 게이트웨이의 IP 주소 또는 DNS 이름 및 포트를 입력합니다.

e.

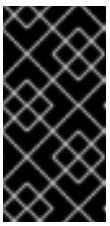
대상 탭에서 대상을 선택하고 연결을 클릭합니다.

f.

연결 대상 창에서 다중 경로 사용 옵션을 선택하고 고급 버튼을 클릭합니다.

g.

연결 using 섹션에서 Microsoft ISCSI Intiator 를 드롭다운 상자에서 로컬 어댑터 로 선택합니다. 드롭다운 상자에서 Windows 클라이언트 IP 주소를 Initiator IP 로 선택합니다. 대상 포털 IP 주소를 선택합니다. Enable CHAP login on 을 선택하고 Ceph iSCSI 클라이언트 자격 증명 섹션에서 Name and Target secret 값을 입력하고 OK 를 클릭합니다.



중요

Windows Server 2016은 12바이트 미만의 CHAP 시크릿을 허용하지 않습니다.

h.

연결 탭을 클릭하기 전에 iSCSI 게이트웨이를 설정할 때 정의된 각 대상 포털에 대해 이전 두 단계를 반복합니다.

i.

이니시에이터 이름이 초기 설정 중에 사용된 이니시에이터 이름과 다른 경우 이니시에이터 이름의 이름을 변경합니다. iSCSI Initiator Properties 창에서 Configuration (구성) 탭에서 Change (변경) 버튼을 클릭하여 이니시에이터 이름의 이름을 바꿉니다.

2.

다중 경로 I/O 설정. PowerShell에서 PDORemovePeriod 명령을 사용하여 MPIO 로드 밸런싱 정책 및 mpclaim 명령을 설정하여 부하 분산 정책을 설정합니다. iSCSI 이니시에이터 도구는 나머지 옵션을 구성합니다.

참고

Red Hat은 PDORemovePeriod 옵션을 PowerShell에서 120초로 늘릴 것을 권장합니다. 애플리케이션을 기반으로 이 값을 조정해야 할 수도 있습니다. 모든 경로가 다운되고 120초가 만료되면 운영 체제가 I/O 요청 실패를 시작합니다.

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

a. *페일오버 정책 설정*

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

b. *페일오버 정책 확인*

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

c. *iSCSI Initiator 툴을 사용하여 대상 탭에서 Devices...을 클릭합니다. 버튼:*

d. *장치 창에서 디스크를 선택하고 MPIO...을 클릭합니다. 버튼:*

e. *Device Details (장치 세부 정보) 창에는 각 대상 포털에 대한 경로가 표시됩니다. 부하 분산 Policy Fail Over only(로드 밸런싱 정책 실패)만 선택해야 합니다.*

f. *PowerShell에서 다중 경로 구성을 확인합니다.*

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

MPIO_DISK_ID 를 적절한 디스크 식별자로 바꿉니다.

참고

LUN을 소유하고 있는 **iSCSI** 게이트웨이 노드의 경로인 **Active/Optimized** 경로가 하나 있으며 다른 **iSCSI** 게이트웨이 노드마다 **Active/Unoptimized** 경로가 있습니다.

3. 선택적으로 설정을 조정합니다. 다음 레지스트리 설정을 사용하는 것이 좋습니다.

- **Windows** 디스크 시간 제한

키

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

현재의

```
TimeOutValue = 65
```

- **Microsoft iSCSI** 개시자 드라이버

키

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

값

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

10.5.4. VMware ESXi용 iSCSI 이니시에이터 구성

사전 요구 사항

- *지원되는 **VMware ESXi** 버전은 고객 포털 지식 베이스 문서 의 **IGW (iSCSI 게이트웨이)** 섹션을 참조하십시오.*
- ***VMware ESXi** 웹 인터페이스에 대한 액세스.*
- ***esxcli** 명령을 실행하기 위해 **VMware ESXi** 호스트 콘솔에 대한 루트 액세스.*

절차

1. ***VMware ESXi** 웹 인터페이스에 로그인합니다.*
2. *작업 → 서비스 강조 표시 → **SSH** 사용을 클릭합니다.*
3. ***VMware ESXi** 호스트 콘솔에 로그인하고 **HardwareAcceleratedMove (XCOPY)**를 비활성화합니다.*

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

4. ***VMware ESXi** 웹 인터페이스의 **Navigator** (네트워크) 창의 **Storage**(스토리지)를 클릭합니다. **Adapters** 탭을 클릭합니다. 어댑터를 강조 표시하고 **Configure iSCSI** (**iSCSI** 구성)를 클릭합니다.*
5. ***Name & alias**(이름 및 별칭) 필드에서 이니시에이터 이름을 확인합니다.*

6.

*이니시에이터 이름이 **gwcli** 유틸리티를 사용하여 초기 설정 중에 클라이언트를 생성할 때 사용되는 이니시에이터 이름과 다른 경우 이니시에이터 이름을 변경합니다. **VMware ESXi** 호스트 콘솔에서 다음 단계를 수행합니다.*

a.

***iSCSI** 소프트웨어의 어댑터 이름을 가져옵니다.*

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

b.

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:rh8-client
```

c.

***VMware ESXi** 웹 인터페이스에서 새 이니시에이터 이름을 확인합니다. 탐색기 창에서 **Storage**(스토리지)를 클릭합니다. **Software iSCSI**(소프트웨어 **iSCSI**)를 클릭합니다. 새 이니시에이터 이름은 **Ceph Object Gateway** 노드 이름과 함께 **Name & alias** (이름 및 별칭) 필드에 있습니다.*

7.

***CHAP** 인증 섹션을 확장합니다. 드롭다운 목록에서 대상에 필요한 경우가 아니면 **Do not use CHAP**(**CHAP**를 사용하지 않음)를 선택합니다. 초기 설정에 사용된 **CHAP Name** 및 **Secret** 자격 증명을 입력합니다. 상호 **CHAP** 인증 섹션이 **CHAP**를 사용하지 않는지 확인합니다.*

주의

***VMware** 호스트 클라이언트의 버그로 인해 **CHAP** 설정이 초기에 사용되지 않습니다. **Ceph iSCSI** 게이트웨이 노드에서 커널 로그에는 이 버그의 표시로 다음과 같은 오류가 포함됩니다.*

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

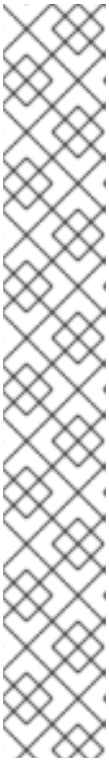
*이 버그를 해결하려면 **esxcli** 명령을 사용하여 **CHAP** 설정을 구성합니다. **authname** 인수는 **CHAP** 인증 섹션의 **Name** 입니다.*

구문

```
esxcli iscsi adapter auth chap set --direction=uni --
authname=ISCSI_USER_NAME --secret=ISCSI_PASSWORD --
level=discouraged -A ADAPTOR_NAME
```

8. ***Advanced settings**(고급 설정) 섹션을 확장합니다. **RecoveryTimeout** 값을 **25**로 설정합니다.*

9. ***Dynamic Target**(동적 타겟) 섹션에서 **Add dynamic target** (동적 대상 추가)을 클릭합니다. **Address** (주소) 필드에서 클릭하여 **Ceph iSCSI** 게이트웨이 중 하나에 대한 **IP** 주소를 추가합니다. 하나의 **IP** 주소만 추가해야 합니다. 마지막으로 **Save configuration** (구성 저장) 버튼을 클릭합니다. **Devices** (장치) 탭을 클릭하여 **RBD** 이미지를 확인합니다.*



참고

LUN은 **ALUA TFTP** 및 **MRU PSP**를 사용하여 자동으로 구성됩니다. 다른 **SatP** 및 **PSP**를 사용하지 마십시오. **esxcli** 명령으로 확인할 수 있습니다.

구문

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

DEVICE_ID 를 적절한 장치 식별자로 바꿉니다.

10. **VMware ESXi** 호스트 콘솔에서 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 부분이 있는 **iSCSI** 또는 **SCSI** 이름이 있습니다.*

*이니시에이터 이름 = **iqn.2005-03.com.ceph:esx1 ISID = 00023d000002** 목표 이름 = **iqn.2003-01.com.redhat.iscsi-gw:iscsi-igw Target port group = 2** 장치 **ID = naa.6001405f8d087846e7b4f0e9e3acd44b***

***active** 의 **Group State** 값은 이 값이 **iSCSI** 게이트웨이의 **Active-Optimized** 경로임을 나타냅니다. **gwcli** 명령은 **iSCSI** 게이트웨이 소유자로 활성 을 나열합니다. 나머지 경로에는 **Group State** 값이 최적화되지 않고 활성 경로가 **dead** 상태가 되면 페일오버 경로가 됩니다.*

c.

*각 **iSCSI** 게이트웨이의 모든 경로를 일치시키려면 다음을 수행합니다.*

예제

```
> esxcli iscsi session connection list
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** 값을 일치시키고 **RemoteAddress** 값은 소유하는 **iSCSI** 게이트웨이의 **IP** 주소입니다.*

11. ***VMware ESXi** 웹 인터페이스에서 장치 탭을 클릭하여 **iSCSI** 디스크를 확인합니다.*

12. ***New datastore** (새 데이터 저장소)를 클릭하여 마법사를 시작합니다.*

a.
*새 데이터 저장소의 이름을 입력하고 **Next (다음)**를 클릭합니다.*

b.
***Use full disk**를 선택하고 **Next (다음)**를 클릭합니다.*

c.
*완료를 클릭합니다. 디스크 삭제에 대한 경고 메시지가 표시됩니다. **Yes (예)**를 클릭하여 계속하고 새 데이터 저장소를 만듭니다.*

d.
*새 데이터 저장소가 **Datastores(데이터 저장소)** 탭에 표시됩니다.*

13.
*데이터 저장소 이름을 선택하여 디스크 사용량을 확인할 수 있습니다. 다음 명령을 실행하여 **Ceph**에서 디스크 사용량을 확인할 수도 있습니다.*

구문

```
rbd du --pool POOL_NAME
```

예제

```
[root@rbd-client ~]# rbd du --pool rbdpool
```

10.6. 더 많은 ISCSI 게이트웨이 추가

*스토리지 관리자는 **gwcli** 명령줄 도구 또는 **Red Hat Ceph Storage** 대시보드를 사용하여 초기 두 개의 **iSCSI** 게이트웨이를 **4**개의 **iSCSI** 게이트웨이로 확장할 수 있습니다. 더 많은 **iSCSI** 게이트웨이를 추가하면 부하 분산 및 페일오버 옵션을 사용할 때 중복성을 높일 수 있습니다.*

10.6.1. 사전 요구 사항

- 실행 중인 **Red Hat Ceph Storage 5** 클러스터
- 예비 노드 또는 기존 **OSD** 노드
- 루트 권한

10.6.2. gwcli 를 사용하여 iSCSI 게이트웨이 추가

gwcli 명령줄 도구를 사용하여 더 많은 **iSCSI** 게이트웨이를 추가할 수 있습니다. 이 절차에서는 기본값 **2**개의 **iSCSI** 게이트웨이를 **4**개의 **iSCSI** 게이트웨이로 확장합니다.

사전 요구 사항

- **Red Hat Enterprise Linux 8.0** 이상.
- 실행 중인 **Red Hat Ceph Storage** 클러스터.
- **iSCSI** 게이트웨이 소프트웨어 설치.
- 새 노드 또는 **OSD** 노드에 **root** 사용자가 액세스할 수 있어야 합니다.

절차

1. **Ceph iSCSI** 게이트웨이를 **OSD** 노드에 배치하지 않은 경우 스토리지 클러스터에서 실행 중인 **Ceph** 노드에서 새 **iSCSI** 게이트웨이 노드로 **/etc/ceph/** 디렉터리에 있는 **Ceph** 구성 파일을 복사합니다. **Ceph** 구성 파일은 **/etc/ceph/** 디렉터리의 **iSCSI** 게이트웨이 노드에 있어야 합니다.
2. **Ceph** 명령줄 인터페이스를 설치하고 구성합니다.
3. 새로운 **iSCSI** 게이트웨이 호스트에서 **Red Hat Ceph Storage Tools** 리포지토리를 활성화합니다.

예제

```
[root@iscsigw ~]# subscription-manager repos --enable=rhceph-5-tools-for-rhel-8-x86_64-rpms
```

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

예제

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

a. 필요한 경우 **openssl** 패키지를 설치합니다.

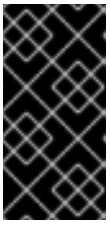
예제

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

5. 기존 **iSCSI** 게이트웨이 호스트 중 하나에서 **/etc/ceph/iscsi-gateway.cfg** 파일을 편집하고 새 **iSCSI** 게이트웨이 호스트의 새 **IP** 주소에 **trusted_ip_list** 옵션을 추가합니다. 예를 들면 다음과 같습니다.

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

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



중요

***iscsi-gateway.cfg** 파일은 모든 **iSCSI** 게이트웨이 호스트에서 동일해야 합니다.*

7. ***SSL**을 사용하는 경우 기존 **iSCSI** 게이트웨이 호스트 중 하나에서 기존 **iSCSI** 게이트웨이 호스트 중 하나에서~/**ssl-keys/iscsi- keys/iscsi-gateway.pem, ~/ssl-keys / iscsi-gateway.key** 파일도 또한 복사합니다.*

8. *새 **iSCSI** 게이트웨이 호스트에서 **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
```

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

추가 리소스

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

10.7. 이니시에이터가 ISCSI 대상에 연결되어 있는지 확인

iSCSI 게이트웨이를 설치하고 iSCSI 대상 및 이니시에이터를 구성한 후 이니시에이터가 iSCSI 대상에 올바르게 연결되어 있는지 확인합니다.

사전 요구 사항

- *Ceph iSCSI 게이트웨이 소프트웨어 설치.*
- *iSCSI 타겟을 구성합니다.*
- *iSCSI 이니시에이터 구성.*

절차

1. *iSCSI 게이트웨이 명령줄 인터페이스를 시작합니다.*

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

2. *이니시에이터가 iSCSI 타겟에 연결되어 있는지 확인합니다.*

```
/> 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)]
```

연결된 경우 이니시에이터 상태는 LOGGED-IN 입니다.

3.

LUN이 iSCSI 게이트웨이에서 분산되는지 확인합니다.

```
/> 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 만들 때 매핑된 LUN 수가 가장 낮은 게이트웨이에 따라 디스크에 iSCSI 게이트웨이가 소유자로 할당됩니다. 이 번호가 분산되면 게이트웨이가 라운드 로빈 할당을 기반으로 할당됩니다. 현재 LUN의 분산은 동적이 아니며 사용자가 선택할 수 없습니다.

이니시에이터가 타겟에 로그인하고 다중 경로 계층이 최적화된 상태에 있으면 이니시에이터의 운영 체제 다중 경로 유틸리티에서 소유자 게이트웨이의 경로를 ALUA Active-Optimized(ALUA Active-Optimized) 상태로 보고합니다. 다중 경로 유틸리티는 ALUA Active-non-Optimized(ANO) 상태인 다른 경로를 보고합니다.

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

10.8. ISCSI 게이트웨이 모니터링

Red Hat Ceph Storage 클러스터는 OSD 및 MGR 내에 일반 메트릭 수집 프레임워크를 통합하여 기본 제공 모니터링을 제공합니다. 지표는 Red Hat Ceph Storage 클러스터 내에서 생성되며 메트릭을 스크랩하기 위해 클라이언트 노드에 액세스할 필요가 없습니다.

RBD 이미지의 성능을 모니터링하기 위해 Ceph에는 개별 RADOS 개체 지표를 초당 I/O(입력/출력(I) 작업에 대한 집계된 RBD 이미지 지표로 변환하는 기본 제공 MGR Prometheus 내보내기 모듈이 있습니다. Ceph iSCSI 게이트웨이는 또한 Grafana와 같은 모니터링 및 시각화 툴을 지원하는 Linux-IO(Linux-IO) 수준 성능 지표에 대한 Prometheus 내보내기를 제공합니다. 이러한 지표에는 LUN에 대해 정의된 TPG(대상 포털 그룹) 및 매핑된 논리 단위 번호(LUN) 및 LUN(초당 입출력 작업 수), 클라이언트당 LUN 읽기 바이트 및 쓰기 바이트가 포함됩니다. 기본적으로 Prometheus 내보내기가 활성화됩니다.

iscsi-gateway.cfg에서 다음 옵션을 사용하여 기본 설정을 변경할 수 있습니다.

예제

-

```
[config]

prometheus_exporter = True
prometheus_port = 9287
prometheus_host = xx.xx.xx.xxx
```

참고

내보낸 **Ceph** 블록 장치(**RBD**) 이미지의 성능을 모니터링하기 위한 **Ceph iSCSI** 게이트웨이 환경에 사용되는 **gwtop** 툴은 더 이상 사용되지 않습니다.

추가 리소스

- 자세한 내용은 **Red Hat Ceph Storage Dashboard Guide**의 **Ceph** 대시보드를 사용하여 **iSCSI** 기능 관리 섹션을 참조하십시오.

10.9. ISCSI 구성 제거

iSCSI 구성을 제거하려면 **gwcli** 유틸리티를 사용하여 호스트와 디스크를 제거합니다.

사전 요구 사항

- 모든 **iSCSI** 이니시에이터의 연결을 끊습니다.
  - **Red Hat Enterprise Linux** 이니시에이터:

구문

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

TARGET_NAME 을 구성된 **iSCSI** 대상 이름으로 교체합니다. 예를 들면 다음과 같습니다.

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.
```

- ***Windows 이니시에이터:***

  자세한 내용은 Microsoft 설명서 를 참조하십시오.

- ***VMware ESXi 이니시에이터:***

  자세한 내용은 VMware 설명서 를 참조하십시오.

절차

1. ***iSCSI 게이트웨이 명령줄 유틸리티를 실행합니다.***

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

2. *호스트를 제거합니다.*

   구문

   ```
   /> cd /iscsi-target/iqn.2003-01.com.redhat.iscsi-gw:$TARGET_NAME/hosts
   /> /iscsi-target...TARGET_NAME/hosts> delete CLIENT_NAME
   ```

***TARGET_NAME** 을 구성된 **iSCSI** 대상 이름으로 바꾸고 **CLIENT_NAME** 을 **iSCSI** 이니시에이터 이름으로 교체합니다. 예를 들면 다음과 같습니다.*

예제

```
/> cd /iscsi-target/iqn.2003-01.com.redhat.iscsi-gw:ceph-igw/hosts
/> /iscsi-target...eph-igw/hosts> delete iqn.1994-05.com.redhat:rh7-client
```

3. *디스크를 제거합니다.*

구문

```
/> cd /disks/
/disks> delete POOL_NAME.IMAGE_NAME
```

***POOL_NAME** 을 풀 이름으로 바꾸고 **IMAGE_NAME**을 이미지 이름으로 교체합니다. 예를 들면 다음과 같습니다.*

예제

```
/> cd /disks/
/disks> delete rbd.disk_1
```

10.10. 추가 리소스

- ***Red Hat Ceph Storage** 대시보드를 사용하여 **iSCSI** 게이트웨이 관리에 대한 자세한 내용은 **Red Hat Ceph Storage 5**용 대시보드 가이드 의 **iSCSI** 기능 섹션을 참조하십시오.*

부록 A. CEPH 블록 장치 구성 참조

스토리지 관리자는 사용 가능한 다양한 옵션을 통해 Ceph 블록 장치의 동작을 미세 조정할 수 있습니다. 이 참조를 사용하여 기본 Ceph 블록 장치 옵션 및 Ceph 블록 장치 캐싱 옵션 등을 볼 수 있습니다.

A.1. 사전 요구 사항

- 실행 중인 Red Hat Ceph Storage 클러스터.

A.2. 블록 장치 기본 옵션

이미지를 생성하기 위한 기본 설정을 재정의할 수 있습니다. Ceph는 형식 2가 포함되고 스트라이핑 이 없는 이미지를 만듭니다.

rbd_default_format

설명

다른 형식이 지정되지않은 경우기본 형식(2)입니다. format 1 은 모든 버전의 librbd 및 커널 모듈과 호환되지만 복제와 같은 최신 기능을 지원하지 않는 새 이미지의 원본 형식입니다. format 2 는 librbd 및 버전 3.11 이후의 커널 모듈에서 지원합니다(제거를 제외하고). 형식 2 에서는 복제 기능이 추가되어 향후 더 많은 기능을 사용할 수 있도록 보다 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 수가 지정되지 않은 경우 기본 스트라이프 수입니다. 기본값을 변경하려면 제거 **v2** 기능이 필요합니다.*

유형

***64**비트 서명되지 않은 정수*

Default

0

rbd_default_stripe_unit

설명

*다른 스트라이프 유닛이 지정되지 않은 경우 기본 스트라이프 단위입니다. 장치를 **0** (즉, 오브젝트 크기)에서 변경하려면 스트라이핑 **v2** 기능이 필요합니다.*

유형

***64**비트 서명되지 않은 정수*

Default

0

rbd_default_features

설명

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

***1:** 계층 지정 지원. 계층 구성을 사용하면 복제를 사용할 수 있습니다.*

***2:** 스트라이핑 **v2** 지원. 스트라이핑은 여러 오브젝트에 데이터를 분산합니다. 스트라이핑은 순차 읽기/쓰기 워크로드의 병렬 처리에 도움이 됩니다.*

***4:** 독점 잠금 지원. 활성화하면 클라이언트가 쓰기 전에 개체에 대한 잠금을 가져와야 합니다.*

8: 개체 맵 지원. 블록 장치는 씬 프로비저닝된 것입니다.즉, 실제로 존재하는 데이터만 저장합니다. 개체 맵 지원은 실제로 존재하는 개체를 추적하는 데 도움이 됩니다(드라이브에 데이터가 저장되어 있어야 함). 오브젝트 맵을 사용하면 복제를 위한 I/O 작업 속도가 빨라지거나 채워진 이미지 가져오기 및 내보내기가 빨라집니다.

16: Fast-diff 지원. Fast-diff 지원은 객체 맵 지원 및 독점 잠금 지원에 따라 다릅니다. 개체 맵에 다른 속성을 추가하면 이미지의 스냅샷과 실제 스냅샷의 실제 데이터 사용량이 훨씬 더 빠르게 생성될 수 있습니다.

32: 심층 지원. deep-flatten은 이미지 자체 외에도 rbd가 이미지의 모든 스냅샷에 대해 플랫화되도록 합니다. 이 기능이 없으면 이미지의 스냅샷은 상위 항목을 계속 사용하므로 스냅샷이 삭제될 때까지 상위 항목을 삭제할 수 없습니다. 심층적인 플랫(deep-flatten)은 스냅샷이 있더라도 부모가 복제본과 독립적입니다.

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

활성화된 기능은 숫자 설정의 합계입니다.

유형

정수

Default

***61 -** 계층화, 독점 잠금, 객체 맵, **fast-diff**, 심층 플랫폼 사용 가능*

중요

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

rbd_default_map_options

설명

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

유형

문자열

Default

""

A.3. 블록 장치 일반 옵션

rbd_op_threads

설명

블록 장치 작업 스레드 수입니다.

유형

정수

Default

1

주의

1 보다 큰 숫자로 설정하면 데이터 손상이 발생할 수 있으므로 rbd_op_threads 의 기본값을 변경하지 마십시오.

rbd_op_thread_timeout

설명

블록 장치 작업 스레드의 시간 제한(초)입니다.

유형

정수

Default

60

rbd_non_blocking_aio

설명

true인 경우 Ceph는 차단을 방지하기 위해 블록 장치 비동기 I/O 작업을 작업자 스레드에서 처리합니다.

유형

부울

Default

true

rbd_concurrent_management_ops

설명

진행 중인 최대 동시 관리 작업 수(예: 이미지 삭제 또는 크기 조정).

유형

정수

Default

10

rbd_request_timed_out_seconds

설명

유지 관리 요청 시간이 초과되기 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 활성화되어 블록 장치는 *OSD* 백엔드에 힌트를 실행하여 예상되는 크기 오브젝트를 나타냅니다.

유형

부울

Default

true

rbd_skip_partial_discard

설명

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

유형

부울

Default

false

rbd_tracing

설명

Linux Trace Toolkit Next Generation User Space Tracer(LTTng-UST) 추적 포인트를 사용하려면 이 옵션을 *true* 로 설정합니다. 자세한 내용은 *RADOS* 블록 장치(*RBD*) 워크로드에서

RBD 재생 기능을 사용한 워크로드 를 참조하십시오.

유형

부울

Default

false

rbd_validate_pool

설명

RBD 호환성을 위해 비어 있는 풀의 유효성을 검사하려면 이 옵션을 **true** 로 설정합니다.

유형

부울

Default

true

rbd_validate_names

설명

이미지 사양의 유효성을 검사하려면 이 옵션을 **true** 로 설정합니다.

유형

부울

Default

true

A.4. 블록 장치 캐싱 옵션

Ceph 블록 장치의 사용자 공간 구현(예: **librbd**)은 **Linux** 페이지 캐시를 활용할 수 없으므로 **RBD** 캐싱이라는 자체 메모리 내 캐싱을 포함합니다. **Ceph** 블록 장치 캐싱은 필요한 하드 디스크 캐싱처럼 작동합니다. 운영 체제에서 장벽 또는 플러시 요청을 보내면 모든 더티 데이터가 **Ceph OSD**에 작성됩니다. 즉, 나중 쓰기 캐싱은 **Linux** 커널 버전 **2.6.32** 이상인 플러시를 올바르게 전송하는 가상 시스템에서 잘 필

요한 물리적 하드 디스크를 사용하는 것만큼 안전합니다. 캐시는 **Least Recently Used (LRU)** 알고리즘을 사용하며 나중 쓰기 모드에서는 연속된 요청을 병합하여 처리량을 높일 수 있습니다.

Ceph 블록 장치는 나중 쓰기 캐싱을 지원합니다. 나중 쓰기 캐싱을 활성화하려면 **rbd_cache = true** 를 **Ceph** 구성 파일의 **[client]** 섹션으로 설정합니다. 기본적으로 **librbd** 는 캐싱을 수행하지 않습니다. 쓰기 및 읽기 작업은 스토리지 클러스터로 직접 전송되고 쓰기 항목은 데이터가 모든 복제본에서 디스크에 있을 때만 반환됩니다. 캐싱을 활성화하면 **rbd_cache_max_dirty** 보다 많은 플러시되지 않은 바이트가 없으면 쓰기 항목은 즉시 반환됩니다. 이 경우 쓰기는 나중 쓰기를 트리거하고 충분한 바이트가 플러시될 때까지 차단합니다.

Ceph 블록 장치는 동시 쓰기 캐싱을 지원합니다. 캐시 크기를 설정할 수 있으며 대상 및 제한을 설정하여 나중 쓰기 캐싱에서 동시 쓰기 캐싱으로 전환할 수 있습니다. 동시 쓰기 모드를 활성화하려면 **rbd_cache_max_dirty** 를 **0**으로 설정합니다. 즉, 쓰기 항목은 데이터가 모든 복제본에서 디스크에 있을 때만 반환되지만 읽기 항목은 캐시에서 가져올 수 있습니다. 캐시는 클라이언트의 메모리에 있으며 각 **Ceph** 블록 장치 이미지에는 자체 캐시가 있습니다. 캐시는 클라이언트에 로컬이므로 다른 사용자가 이미지에 액세스하는 경우 일관성이 유지되지 않습니다. **Ceph** 블록 장치의 **GFS** 또는 **OCFS**와 같은 다른 파일 시스템을 실행하면 활성화된 캐싱에서 작동하지 않습니다.

Ceph 블록 장치의 **Ceph** 구성 설정은 기본적으로 **/etc/ceph/ceph.conf** 의 **Ceph** 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_cache_size

설명

RBD 캐시 크기(바이트)입니다.

유형

64비트 정수

필수 항목

없음

Default

32 MiB

rbd_cache_max_dirty

설명

캐시가 나중 쓰기를 트리거하는 더 티 바이트(바이트)입니다. 0 인 경우 동시 쓰기 캐싱을 사용합니다.

유형

64비트 정수

필수 항목

없음

제약 조건

rbd 캐시 크기 보다 작아야 합니다.

Default

24 MiB

rbd_cache_target_dirty

설명

캐시가 데이터 스토리지에 데이터 쓰기를 시작하기 전 더 티 대상입니다. 는 캐시에 대한 쓰기를 차단하지 않습니다.

유형

64비트 정수

필수 항목

없음

제약 조건

rbd 캐시 최대 더티 보다 작아야 합니다.

Default

16 MiB

rbd_cache_max_dirty_age

설명

writeback을 시작하기 전에 더티 데이터가 캐시에 있는 시간(초)입니다.

유형

교체

필수 항목

없음

Default

1.0

rbd_cache_max_dirty_object

설명

자동 계산의 경우 오브젝트의 더티 제한 - 0 으로 설정된 rbd_cache_size 에서 계산합니다.

유형

정수

Default

0

rbd_cache_block_writes_upfront

설명

true인 경우 aio_write 호출이 완료되기 전에 캐시에 쓰기가 차단됩니다. false인 경우 aio_completion 을 호출하기 전에 차단됩니다.

유형

부울

Default

false

rbd_cache_writethrough_until_flush

설명

동시 쓰기 모드로 시작하고 첫 번째 플러시 요청이 수신된 후 나중 쓰기로 전환합니다. 이 설정은 2.6.32 이전 Linux의 virtio 드라이버와 같이 플러시를 보낼 때 rbd에서 실행되는 VM이 너무 오래되었을 때의 보수적이지만 안전한 설정입니다.

유형

부울

필수 항목

없음

Default

true

A.5. 블록 장치 상위 및 하위 읽기 옵션

rbd_balance_snap_reads

설명

일반적으로 Ceph는 기본 OSD에서 오브젝트를 읽습니다. 읽기는 변경할 수 없으므로 이 기능을 활성화하여 기본 OSD와 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 임의로 지정합니다. **rbd_localize_snap_reads** 를 활성화하면 블록 장치에서 **CRUSH** 맵을 조회하여 스냅숏을 읽기 위한 가장 유사한 로컬 **OSD**를 찾습니다.*

유형

부울

Default

false

rbd_balance_parent_reads

설명

*일반적으로 **Ceph**는 기본 **OSD**에서 오브젝트를 읽습니다. 읽기는 변경할 수 없으므로 이 기능을 통해 기본 **OSD**와 복제본 간에 상위 읽기의 균형을 조정할 수 있습니다.*

유형

부울

Default

false

rbd_localize_parent_reads

설명

***rbd_balance_parent_reads** 는 상위 항목을 읽기 위한 복제본을 무작위로 지정합니다. **rbd_localize_parent_reads** 를 활성화하면 블록 장치에서 **CRUSH** 맵을 조회하여 상위 항목을 읽는 데 가장 가까운 로컬 **OSD**를 찾습니다.*

유형

부울

Default

true

A.6. 사전 옵션 읽기 블록 장치

RBD는 읽기-ahead/prefetching을 지원하여 작은 순차 읽기를 최적화합니다. 일반적으로 VM의 경우 게스트 OS에서 처리해야 하지만 부트 로더는 효율적으로 읽지 못할 수 있습니다. 캐싱이 비활성화된 경우 읽기-ahead가 자동으로 비활성화됩니다.

rbd_readahead_trigger_requests

설명

읽기-헤드를 트리거하는 데 필요한 순차적 읽기 요청 수입니다.

유형

정수

필수 항목

없음

Default

10

rbd_readahead_max_bytes

설명

읽기-헤드 요청의 최대 크기. 0인 경우 읽기-ahead가 비활성화됩니다.

유형

64비트 정수

필수 항목

없음

Default

512 KiB

rbd_readahead_disable_after_bytes

설명

RBD 이미지에서 이 여러 바이트를 읽은 후에는 종료될 때까지 해당 이미지에 대해 읽기-ahead가 비활성화됩니다. 이렇게 하면 게스트 **OS**가 부팅되면 읽기-세임을 대신할 수 있습니다. **0** 인 경우 읽기**-ahead**가 활성화된 상태로 유지됩니다.

유형

64비트 정수

필수 항목

없음

Default

50 MiB

A.7. 블록 장치 블록 목록 옵션

rbd_blocklist_on_break_lock

설명

잠금이 손상된 클라이언트를 차단할 것인지 여부입니다.

유형

부울

Default

true

rbd_blocklist_expire_seconds

설명

OSD 기본값으로 **blocklist - 0**으로 설정할 시간(초)입니다.

유형

정수

Default

0

A.8. 블록 장치 저널 옵션

rbd_journal_order

설명

저널 오브젝트 최대 크기 계산으로 이동할 비트 수입니다. 값은 12 에서 64 사이입니다.

유형

32 비트 서명되지 않은 정수

Default

24

rbd_journal_splay_width

설명

활성 저널 오브젝트 수입니다.

유형

32 비트 서명되지 않은 정수

Default

4

rbd_journal_commit_age

설명

커밋 시간 간격(초)입니다.

유형

이중유점 번호

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

저널 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

이중유점 번호

Default

0

rbd_journal_pool

설명

저널 오브젝트의 풀을 지정합니다.

유형

문자열

Default

""

A.9. 블록 장치 구성 덮어쓰기 옵션

전역 및 풀 수준에 대한 블록 장치 구성 덮어쓰기 옵션.

글로벌 레벨

사용 가능한 키

rbd_qos_bps_burst

설명

원하는 IO 바이트 제한입니다.

유형

정수

Default

0

rbd_qos_bps_limit

설명

초당 IO 바이트 제한입니다.

유형

정수

Default

0

rbd_qos_iops_burst

설명

원하는 IO 작업 한도입니다.

유형

정수

Default

0

rbd_qos_iops_limit

설명

원하는 초당 IO 작업 제한입니다.

유형

정수

Default

0

rbd_qos_read_bps_burst

설명

원하는 읽기 바이트의 버스트 제한입니다.

유형

정수

Default

0

rbd_qos_read_bps_limit

설명

원하는 초당 읽기 바이트 제한입니다.

유형

정수

Default

0

rbd_qos_read_iops_burst

설명

원하는 읽기 작업의 버스트 제한.

유형

정수

Default

0

rbd_qos_read_iops_limit

설명

초당 읽기 작업의 원하는 한도입니다.

유형

정수

Default

0

rbd_qos_write_bps_burst

설명

원하는 쓰기 바이트의 버스트 제한입니다.

유형

정수

Default

0

rbd_qos_write_bps_limit

설명

원하는 초당 쓰기 바이트 제한.

유형

정수

Default

0

rbd_qos_write_iops_burst

설명

원하는 쓰기 작업의 버스트 제한.

유형

정수

Default

0

rbd_qos_write_iops_limit

설명

원하는 쓰기 작업 수(초당 쓰기 작업 수).

유형

정수

Default

0

위의 키는 다음과 같이 사용할 수 있습니다.

RBD 구성 글로벌 설정 CONFIG_ENTITY KEY VALUE

설명

글로벌 수준 구성 덮어쓰기를 설정합니다.

RBD 구성 전역 get CONFIG_ENTITY KEY

설명

글로벌 수준 구성 덮어쓰기 가져오기.

RBD 구성 글로벌 목록 CONFIG_ENTITY

설명

글로벌 수준 구성 덮어쓰기를 나열합니다.

RBD config global remove CONFIG_ENTITY KEY

설명

전역 수준 구성 덮어쓰기를 제거합니다.

풀 수준

RBD 구성 풀 설정 POOL_NAME KEY VALUE

설명

풀 수준 구성 덮어쓰기를 설정합니다.

RBD 구성 풀 get POOL_NAME KEY

설명

풀 수준 구성을 덮어씁니다.

RBD 구성 풀 목록 POOL_NAME

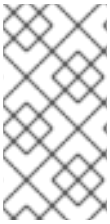
설명

풀 수준 구성 덮어쓰기를 나열합니다.

RBD 구성 풀이 POOL_NAME KEY제거

설명

풀 수준 구성 덮어쓰기를 제거합니다.



참고

***CONFIG_ENTITY** 는 글로벌, 클라이언트 또는 클라이언트 **ID**입니다. **KEY** 는 구성 키입니다. **VALUE** 는 **config** 값입니다. **POOL_NAME** 은 풀의 이름입니다.*

A.10. 블록 장치 입력 및 출력 옵션

***Red Hat Ceph Storage**의 일반적인 입력 및 출력 옵션.*

rbd_compression_hint

설명

*쓰기 작업에 대한 **OSD**에 보낼 힌트입니다. 압축 가능으로 설정되고 **OSD bluestore_compression_mode** 설정이 패시브 인 경우 **OSD**는 데이터를 압축하려고 시도합니다. 압축할 수 없는 것으로 설정되고 **OSD bluestore_compression_mode** 설정이 공격적 이면 **OSD** 에서 데이터를 압축하지 않습니다.*

유형

enum

필수 항목

없음

Default

none

값

없음,압축할 수 없음, 압축되지 않음

rbd_read_from_replica_policy

설명

*읽기 작업을 수신하는 **OSD**를 결정하는 정책입니다. **default** 로 설정하면 각 **PG**의 기본 **OSD**가 항상 읽기 작업에 사용됩니다. **balance** 으로 설정되면 읽기 작업이 복제본 세트 내에서 임의로 선택한 **OSD**로 전송됩니다. **localize**로 설정되면 **CRUSH** 맵과 **metering _location** 구성 옵션의 결*

*정에 따라 읽기 작업이 가장 가까운 **OSD**로 전송됩니다. 여기서 **keys =value** 를 사용하여 **keys _location** 이 표시됩니다. 키는 **CRUSH** 맵 키와 일치합니다.*

참고

*이 기능을 사용하려면 최신 **Red Hat Ceph Storage** 버전의 최소 호환 **OSD** 릴리스로 스토리지 클러스터를 구성해야 합니다.*

유형

enum

필수 항목

없음

Default

default

값

*기본,균형 , **localize***

부록 B. ISCSI 게이트웨이 변수

iSCSI 게이트웨이 일반 변수

seed_monitor

목적

각 iSCSI 게이트웨이는 RADOS 및 RBD 호출의 Ceph 스토리지 클러스터에 액세스해야 합니다. 즉, iSCSI 게이트웨이에 적절한 /etc/ceph/ 디렉토리가 정의되어 있어야 합니다. seed_monitor 호스트는 iSCSI 게이트웨이의 /etc/ceph/ 디렉터리를 채우는 데 사용됩니다.

gateway_keyring

목적

사용자 지정 인증 키 이름을 정의합니다.

perform_system_checks

목적

이는 각 iSCSI 게이트웨이에서 다중 경로 및 LVM 구성 설정을 확인하는 부울 값입니다. multipathd 데몬과 LVM이 올바르게 구성되었는지 확인하려면 최소한 첫 번째 실행에 대해 true 로 설정해야 합니다.

iSCSI 게이트웨이 RBD-TARGET-API 변수

api_user

목적

API의 사용자 이름입니다. 기본값은 admin 입니다.

api_password

목적

API를 사용하는 암호입니다. 기본값은 admin 입니다.

api_port

목적

API를 사용하기 위한 TCP 포트 번호입니다. 기본값은 5000 입니다.

api_secure

목적

*값은 **true** 또는 **false** 일 수 있습니다. 기본값은 **false**입니다.*

loop_delay

목적

*__iSCSI__ 관리 오브젝트를 폴링하기 위해 대기 간격(초)을 제어합니다. 기본값은 **1** 입니다.*

trusted_ip_list

목적

__API__에 액세스할 수 있는 __IPv4__ 또는 __IPv6__ 주소 목록입니다. 기본적으로 __iSCSI__ 게이트웨이 호스트만 액세스할 수 있습니다.

부록 C. 샘플 ISCSIGWS.YML 파일

예제

```
service_type: iscsi
service_id: iscsi
placement:
  hosts:
  - host11
  - magna12
spec:
  pool: iscsi_pool
  trusted_ip_list:
"10.80.100.100,10.8.100.113,2620:52:0:880:225:90ff:fefc:1bf2,2620:52:0:880:225:90ff:fefc:1a8c"
  api_user: user1
  api_password: password1
```