Red Hat OpenStack Platform 17.1

이미지 생성 및 관리

Image 서비스(glance)를 사용하여 Red Hat OpenStack Platform에서 이미지 생성 및 관리

Last Updated: 2023-09-21

Red Hat OpenStack Platform 17.1 이미지 생성 및 관리

Image 서비스(glance)를 사용하여 Red Hat OpenStack Platform에서 이미지 생성 및 관리

OpenStack Team
rhos-docs@redhat.com

법적 공지

초록

이 가이드에서는 이미지를 생성하고 관리하는 절차와 Image 서비스(glance)를 구성하는 절차를 설명합니다.

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체

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 메시지 를 참조하십시오.

RED HAT 문서에 관한 피드백 제공

문서 개선을 위한 의견을 보내 주십시오. Red Hat이 어떻게 더 나은지 알려주십시오.

직접 문서 피드백(DDF) 기능 사용

피드백 추가 DDF 기능을 사용하여 특정 문장, 단락 또는 코드 블록에 대한 직접 의견을 제출할 수 있습니다.

1. *다중 페이지 HTML* 형식으로 문서를 봅니다.
2. 문서의 오른쪽 상단에 **피드백** 버튼이 표시되는지 확인합니다.
3. 주석 처리하려는 텍스트 부분을 강조 표시합니다.
4. **피드백 추가를** 클릭합니다.
5. 코멘트를 사용하여 **피드백 추가** 필드를 완료합니다.
6. 선택 사항: 문서 팀이 문제에 대한 자세한 설명을 위해 연락을 드릴 수 있도록 이메일 주소를 추가합니다.
7. **Submit** 을 클릭합니다.

1장. IMAGE 서비스(GLANCE)

RHOSP(Red Hat OpenStack Platform)에서 이미지 및 스토리지를 관리합니다.

1.1. VM(가상 머신) 이미지 형식

VM 이미지는 부팅 가능한 운영 체제가 설치된 가상 디스크가 포함된 파일입니다. VM 이미지는 다양한 형식으로 지원됩니다. 다음 형식은 RHOSP(Red Hat OpenStack Platform)에서 사용할 수 있습니다.

- **RAW** - 구조화되지 않은 디스크 이미지 형식입니다.
- **QCOW2** - QEMU 에뮬레이터에서 지원하는 디스크 형식입니다. 이 형식에는 QEMU 1.1 이상이 필요한 QCOW2v3(또는 QCOW3)이 포함됩니다.
- **ISO** - Sector-by-sector는 바이너리 파일에 저장된 디스크의 데이터 복사본입니다.
- **AKI** - Amazon 커널 이미지를 나타냅니다.
- **AMI** - Amazon 머신 이미지를 나타냅니다.
- **ARI** - Amazon RAMDisk 이미지를 나타냅니다.
- **VDI** - Cryostat VM 모니터 및 QEMU 에뮬레이터에서 지원하는 디스크 형식입니다.
- **VHD** - VMware, Cryostat 등의 VM 모니터에서 사용하는 일반 디스크 형식입니다.
- **PLECDHE** P - OS 컨테이너를 실행하기 위해 Virtu Cryostatzo에서 지원하고 사용하는 디스크 형식입니다.
- **OVA** - Image 서비스(glance)에 저장된 데이터가 OVA tar 아카이브 파일임을 나타냅니다.
- **DOCKER** - Image 서비스(glance)에 저장된 것이 컨테이너 파일 시스템의 Docker tar 아카이브임을 나타냅니다.

ISO 에는 일반적으로 VM 이미지 형식으로 간주되지 않지만 설치된 운영 체제가 있는 부팅 가능한 파일 시스템이 포함되어 있기 때문에 다른 VM 이미지 파일과 동일한 방식으로 사용합니다.

1.2. 지원되는 이미지 서비스 백엔드

다음과 같은 Image 서비스(glance) 백엔드 시나리오가 지원됩니다.

- RADOS 블록 장치(RBD)는 Ceph를 사용할 때 기본 백엔드입니다.
- RBD 다중 저장소.
- Object Storage(swift). 이미지 서비스는 Object Storage 유형 및 백엔드를 기본값으로 사용합니다.
- Block Storage(cinder).
- NFS

  중요

  NFS는 지원되는 이미지 서비스 배포 옵션이지만 더 강력한 옵션을 사용할 수 있습니다.

NFS는 이미지 서비스의 네이티브가 아닙니다. 이미지 서비스에 NFS 공유를 마운트하면 이미지 서비스에서 작업을 관리하지 않습니다. 이미지 서비스는 파일 시스템에 데이터를 작성하지만 백엔드가 NFS 공유임을 인식하지 못합니다.

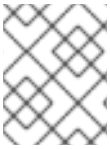
이러한 유형의 배포에서는 공유가 실패하면 이미지 서비스에서 요청을 다시 시도할 수 없습니다. 즉, 백엔드에서 오류가 발생하면 저장소가 읽기 전용 모드로 전환되거나 로컬 파일 시스템에 데이터를 계속 쓸 수 있으므로 데이터가 손실될 위험이 있습니다. 이 상황에서 복구하려면 공유가 마운트되고 동기화되었는지 확인한 다음 이미지 서비스를 다시 시작해야 합니다. 이러한 이유로 Red Hat은 NFS를 이미지 서비스 백엔드로 권장하지 않습니다.

그러나 NFS를 이미지 서비스 백엔드로 사용하도록 선택하는 경우 다음 모범 사례 중 일부는 위험을 완화하는 데 도움이 될 수 있습니다.

- 안정적인 프로덕션 수준의 NFS 백엔드를 사용합니다.
- 컨트롤러 노드와 NFS 백엔드 간에 강력하고 안정적인 연결이 있는지 확인합니다. 계층 2(L2) 네트워크 연결이 권장됩니다.
- 마운트된 공유에 대한 모니터링 및 경고를 포함합니다.
- 기본 파일 시스템 권한을 설정합니다. 저장소로 사용하는 공유 파일 시스템에 쓰기 권한이 있어야 합니다.
- glance-api 프로세스가 실행되는 사용자 및 그룹에 로컬 파일 시스템의 마운트 지점에 대한 쓰기 권한이 없는지 확인합니다. 즉, 프로세스에서 가능한 마운트 실패를 감지하고 쓰기 시도 중에 저장소를 읽기 전용 모드로 배치할 수 있습니다.

1.3. 이미지 서명 및 확인

이미지 서명 및 확인은 배포자가 이미지에 서명하고 서명 및 공개 키 인증서를 이미지 속성으로 저장할 수 있으므로 이미지 무결성 및 신뢰성을 보호합니다.



참고

Nova에서 RADOS Block Device(RBD)를 사용하여 가상 머신 디스크를 저장하는 경우 이미지 서명 및 확인이 지원되지 않습니다.

이미지 서명 및 확인에 대한 자세한 내용은 *키 관리자 서비스 가이드를 사용하여 시크릿 관리 가이드의 Image 서비스(glance) 이미지 유효성 검사를* 참조하십시오.

1.4. 이미지 변환

이미지 변환은 이미지를 가져오는 동안 작업 API를 호출하여 이미지를 변환합니다.

가져오기 워크플로의 일부로 플러그인은 이미지 변환을 제공합니다. 배포 구성에 따라 이 플러그인을 활성화하거나 비활성화할 수 있습니다. 배포자는 배포에 권장되는 이미지 형식을 지정해야 합니다.

내부적으로 Image 서비스(glance)는 이미지의 비트를 특정 형식으로 수신하고 비트를 임시 위치에 저장합니다. 이미지 서비스는 플러그인을 트리거하여 이미지를 대상 형식으로 변환하고 이미지를 최종 대상으로 이동합니다. 작업이 완료되면 이미지 서비스가 임시 위치를 삭제합니다. 이미지 서비스는 처음에 업로드된 형식을 유지하지 않습니다.

이미지를 가져올 때만 이미지 변환을 트리거할 수 있습니다. 이미지를 업로드할 때 실행되지 않습니다.

이미지 관리에 이미지 서비스 명령줄 클라이언트를 사용합니다.

예를 들면 다음과 같습니다.

```
$ glance image-create-via-import \
   --disk-format qcow2 \
   --container-format bare \
   --name NAME \
   --visibility public \
   --import-method web-download \
   --uri http://server/image.qcow2
```

1.5. 상호 운용 가능한 이미지 가져오기

상호 운용 가능한 이미지 가져오기 워크플로를 사용하면 다음 두 가지 방법으로 이미지를 가져올 수 있습니다.

- URI에서 이미지를 가져오려면 **web-download** (기본값) 방법을 사용합니다.
- **glance-direct** 방법을 사용하여 로컬 파일 시스템에서 이미지를 가져옵니다.
- **copy-image** 방법을 사용하여 기존 이미지를 배포에 있는 기타 Image 서비스(glance) 백엔드에 복사합니다. 배포에서 여러 이미지 서비스 백엔드가 활성화된 경우에만 이 가져오기 방법을 사용합니다.

1.6. 이미지 서비스 캐싱으로 확장성 개선

glance-api 캐싱 메커니즘을 사용하여 Image 서비스(glance) API 서버에 이미지 사본을 저장하고 자동으로 검색하여 확장성을 향상시킵니다. 이미지 서비스 캐싱을 사용하면 glance-api를 여러 호스트에서 실행할 수 있습니다. 즉, 백엔드 스토리지에서 동일한 이미지를 여러 번 검색할 필요가 없습니다. 이미지 서비스 캐싱은 이미지 서비스 작업에 영향을 미치지 않습니다.

Red Hat OpenStack Platform director(tripleo) heat 템플릿을 사용하여 이미지 서비스 캐싱을 설정합니다.

절차

1. 환경 파일에서 **GlanceCacheEnabled** 매개변수 값을 **true** 로 설정하여 **glance-api.conf** heat 템플릿에서 **플레이버** 값을 **keystone+cachemanagement** 로 자동으로 설정합니다.

   ```
   parameter_defaults:
     GlanceCacheEnabled: true
   ```

2. 오버클라우드를 재배포할 때 **openstack overcloud deploy** 명령에 환경 파일을 포함합니다.

3. 선택 사항: 오버클라우드를 재배포할 때 **glance_cache_pruner** 를 대체 빈도로 튜닝합니다. 다음 예제에서는 5 분의 빈도를 보여줍니다.

   ```
   parameter_defaults:
     ControllerExtraConfig:
       glance::cache::pruner::minute: '*/5'
   ```

   파일 시스템 전체 시나리오를 피하기 위해 필요에 따라 빈도를 조정합니다. 대체 빈도를 선택할 때 다음 요소를 포함합니다.

- 사용자 환경에서 캐시할 파일의 크기입니다.
- 사용 가능한 파일 시스템 공간의 양입니다.
- 환경에서 이미지를 캐시하는 빈도입니다.

1.7. 이미지 사전 캐싱

RHOSP(Red Hat OpenStack Platform) director는 **glance-api** 서비스의 일부로 이미지를 사전 캐시할 수 있습니다.

이미지 관리에 Image 서비스(glance) 명령줄 클라이언트를 사용합니다.

1.7.1. 주기적인 이미지 사전 캐싱의 기본 간격 구성

RHOSP(Red Hat OpenStack Platform) director는 **glance-api** 서비스의 일부로 이미지를 사전 캐시할 수 있습니다.

사전 캐싱 주기 작업은 **glance-api** 서비스가 실행 중인 각 컨트롤러 노드에서 300초(5분 기본 시간)마다 실행됩니다. 기본 시간을 변경하려면 glance-api.conf의 **Default** 섹션에서 **cache_prefetcher_interval** 매개변수를 설정할 수 있습니다.

프로세스

1. 요구 사항에 따라 언더클라우드의 환경 파일에 **ExtraConfig** 매개변수를 사용하여 새 간격을 추가합니다.

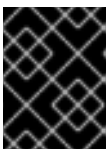
```
parameter_defaults:
  ControllerExtraConfig:
    glance::config::glance_api_config:
      DEFAULT/cache_prefetcher_interval:
        value: '<300>'
```

   <300>을 이미지를 사전 캐시하는 간격으로 원하는 시간(초)으로 바꿉니다.

2. **/home/stack/templates/** 의 환경 파일에서 간격을 조정한 후 **stack** 사용자로 로그인하고 구성을 배포합니다.

```
$ openstack overcloud deploy --templates \
-e /home/stack/templates/<env_file>.yaml
```

   <env_file>을 사용자가 추가한 **ExtraConfig** 설정이 포함된 환경 파일의 이름으로 바꿉니다.



중요

오버클라우드를 생성할 때 추가 환경 파일을 전달한 경우 오버클라우드를 원치 않는 변경하지 않도록 **-e** 옵션을 사용하여 여기에서 다시 전달합니다.

openstack overcloud deploy 명령에 대한 자세한 내용은 *director 가이드를 사용하여 Red Hat OpenStack Platform 설치 및 관리의* Deployment 명령을 참조하십시오.

1.7.2. 정기적인 작업을 사용하여 이미지를 사전 캐시

정기적인 작업을 사용하여 이미지를 사전 캐시합니다.

사전 요구 사항

주기적인 작업을 사용하여 이미지를 사전 캐시하려면 **glance_api** 서비스가 실행 중인 노드에 직접 연결된 **glance-cache-manage** 명령을 사용해야 합니다. 서비스 요청에 응답하는 노드를 숨기는 프록시를 사용하지 마십시오. 언더클라우드가 **glance_api** 서비스가 실행 중인 네트워크에 액세스할 수 없으므로 기본적으로 **controller-0** 이라는 첫 번째 오버클라우드 노드에서 명령을 실행합니다.

올바른 호스트에서 명령을 실행하고 필요한 인증 정보가 있고 **glance-api** 컨테이너 내부에서 **glance-cache-manage** 명령을 실행하고 있는지 확인하려면 다음 사전 요구 사항 절차를 완료합니다.

절차

1. stack 사용자로 언더클라우드에 로그인하고 **controller-0** 의 프로비저닝 IP 주소를 식별합니다.

```
(undercloud) [stack@site-undercloud-0 ~]$ openstack server list -f value -c Name -c Networks | grep controller
overcloud-controller-1 ctlplane=192.168.24.40
overcloud-controller-2 ctlplane=192.168.24.13
overcloud-controller-0 ctlplane=192.168.24.71
(undercloud) [stack@site-undercloud-0 ~]$
```

2. 오버클라우드에 인증하려면 기본적으로 **/home/stack/overcloudrc** 에 저장된 인증 정보를 **controller-0** 으로 복사합니다.

```
$ scp ~/overcloudrc tripleo-admin@192.168.24.71:/home/tripleo-admin/
```

3. **controller-0** 에 연결합니다.

```
$ ssh tripleo-admin@192.168.24.71
```

4. **controller-0** 에서 **tripleo-admin** 사용자로 **glance_api 서비스** 의 IP 주소를 식별합니다. 다음 예에서 IP 주소는 Cryostat **1.105** 입니다.

```
(overcloud) [root@controller-0 ~]# grep -A 10 '^listen glance_api' /var/lib/config-data/puppet-generated/haproxy/etc/haproxy/haproxy.cfg
listen glance_api
 server central-controller0-0.internalapi.redhat.local 172.25.1.105:9292 check fall 5 inter 2000 rise 2
```

5. **glance-cache-manage** 명령은 **glance_api** 컨테이너에서만 사용할 수 있으므로 오버클라우드에 인증할 환경 변수가 이미 설정되어 있는 해당 컨테이너에 exec 스크립트를 만듭니다. 다음 콘텐츠를 사용하여 **controller-0** 의 **/home/tripleo-admin** 에 **glance_pod.sh** 라는 스크립트를 생성합니다.

```
sudo podman exec -ti \
 -e NOVA_VERSION=$NOVA_VERSION \
 -e COMPUTE_API_VERSION=$COMPUTE_API_VERSION \
 -e OS_USERNAME=$OS_USERNAME \
 -e OS_PROJECT_NAME=$OS_PROJECT_NAME \
 -e OS_USER_DOMAIN_NAME=$OS_USER_DOMAIN_NAME \
 -e OS_PROJECT_DOMAIN_NAME=$OS_PROJECT_DOMAIN_NAME \
 -e OS_NO_CACHE=$OS_NO_CACHE \
 -e OS_CLOUDNAME=$OS_CLOUDNAME \
 -e no_proxy=$no_proxy \
```

```
 -e OS_AUTH_TYPE=$OS_AUTH_TYPE \
 -e OS_PASSWORD=$OS_PASSWORD \
 -e OS_AUTH_URL=$OS_AUTH_URL \
 -e OS_IDENTITY_API_VERSION=$OS_IDENTITY_API_VERSION \
 -e OS_COMPUTE_API_VERSION=$OS_COMPUTE_API_VERSION \
 -e OS_IMAGE_API_VERSION=$OS_IMAGE_API_VERSION \
 -e OS_VOLUME_API_VERSION=$OS_VOLUME_API_VERSION \
 -e OS_REGION_NAME=$OS_REGION_NAME \
glance_api /bin/bash
```

6. **overcloudrc** 파일을 가져오고 **glance_pod.sh** 스크립트를 실행하여 오버클라우드 컨트롤러 노드를 인증하는 데 필요한 환경 변수를 사용하여 **glance_api** 컨테이너를 실행합니다.

```
[tripleo-admin@controller-0 ~]$ source overcloudrc
(overcloudrc) [tripleo-admin@central-controller-0 ~]$ bash glance_pod.sh
()[glance@controller-0 /]$
```

7. **glance image-list** 와 같은 명령을 사용하여 컨테이너에서 오버클라우드에 대해 인증된 명령을 실행할 수 있는지 확인합니다.

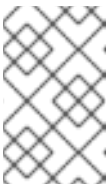
```
()[glance@controller-0 /]$ glance image-list
+--------------------------------------+---------------------------------+
| ID                                   | Name                            |
+--------------------------------------+---------------------------------+
| ad2f8daf-56f3-4e10-b5dc-d28d3a81f659 | cirros-0.4.0-x86_64-disk.img    |
+--------------------------------------+---------------------------------+
()[glance@controller-0 /]$
```

절차

1. admin 사용자로 캐시할 이미지를 대기열에 넣습니다.

```
$ glance-cache-manage --host=<host_ip> queue-image <image_id>
```

- <host_ip>를 **glance-api** 컨테이너가 실행 중인 컨트롤러 노드의 IP 주소로 바꿉니다.
- <image_id>를 큐하려는 이미지의 ID로 바꿉니다.
  사전 캐시하려는 이미지를 대기열에 추가하면 **cache_images** 정기 작업이 모든 대기 중인 이미지를 동시에 사전 가져옵니다.



참고

이미지 캐시는 각 노드의 로컬이므로 Red Hat OpenStack Platform이 HA(3, 5 또는 7 컨트롤러 포함)와 함께 배포된 경우 **glance-cache-manage** 명령을 실행할 때 **--host** 옵션으로 호스트 주소를 지정해야 합니다.

2. 다음 명령을 실행하여 이미지 캐시의 이미지를 확인합니다.

```
$ glance-cache-manage --host=<host_ip> list-cached
```

<host_ip>를 사용자 환경에 있는 호스트의 IP 주소로 바꿉니다.

관련 정보

다음과 같은 목적으로 추가 **glance-cache-manage** 명령을 사용할 수 있습니다.

- **나열-캐시** 되어 현재 캐시된 모든 이미지를 나열합니다.
- 캐싱을 위해 현재 **대기열에** 있는 모든 이미지를 나열하려면 목록을 큐에 넣습니다.
- 캐싱할 **이미지를** 대기열에 추가하는 대기열 이미지입니다.
- 캐시에서 이미지를 제거하기 위해 **delete-cached-image**
- cache에서 모든 이미지를 제거하기 위해 **delete-all-cached-images**
- 캐시 대기열에서 이미지를 삭제하려면 **delete-queued-image**
- 캐시 대기열에서 모든 이미지를 삭제하려면 **delete-all-queued-images**

1.8. 이미지 서비스 API를 사용하여 스파스 이미지 업로드 활성화

Image 서비스(glance) API를 사용하면 스파스 이미지 업로드를 사용하여 네트워크 트래픽을 줄이고 스토리지 공간을 절약할 수 있습니다. 이 기능은 DCN(Distributed Compute Node) 환경에서 특히 유용합니다. 스파스 이미지 파일을 사용하면 이미지 서비스에서 null 바이트 시퀀스를 쓰지 않습니다. 이미지 서비스는 지정된 오프셋을 사용하여 데이터를 작성합니다. 스토리지 백엔드는 이러한 오프셋을 실제로 스토리지 공간을 사용하지 않는 null 바이트로 해석합니다.

이미지 관리에 이미지 서비스 명령줄 클라이언트를 사용합니다.

제한

- 스파스 이미지 업로드는 Ceph RADOS Block Device(RBD)에서만 지원됩니다.
- 파일 시스템에서 스파스 이미지 업로드가 지원되지 않습니다.
- Sparseness는 클라이언트와 이미지 서비스 API 간의 전송 중에 유지 관리되지 않습니다. 이미지는 이미지 서비스 API 수준에서 스파스됩니다.

사전 요구 사항

- RHOSP(Red Hat OpenStack Platform) 배포에서는 이미지 서비스 백엔드에 RBD를 사용합니다.

절차

1. **stack** 사용자로 언더클라우드 노드에 로그인합니다.
2. **stackrc** 인증 정보 파일을 소싱합니다.

```
$ source stackrc
```

3. 다음 콘텐츠를 사용하여 환경 파일을 생성합니다.

```
parameter_defaults:
  GlanceSparseUploadEnabled: true
```

4. 다른 환경 파일과 함께 스택에 새 환경 파일을 추가하고 오버클라우드를 배포합니다.

```
$ openstack overcloud deploy \
```

```
--templates \
…
-e <existing_overcloud_environment_files> \
-e <new_environment_file>.yaml \
...
```

이미지 업로드에 대한 자세한 내용은 이미지 업로드를 참조하십시오.

검증

이미지를 가져오고 해당 크기를 확인하여 스파스 이미지 업로드를 확인할 수 있습니다.

다음 절차에서는 예제 명령을 사용합니다. 적절한 경우 해당 환경의 값으로 바꿉니다.

1. 이미지 파일을 로컬로 다운로드합니다.

```
$ wget <file_location>/<file_name>
```

   - & **lt;file_location** >을 파일 위치로 바꿉니다.
   - & **lt;file_name** >을 파일 이름으로 바꿉니다.
     예를 들면 다음과 같습니다.

```
$ wget https://cloud.centos.org/centos/6/images/CentOS-6-x86_64-GenericCloud-1508.qcow2
```

2. 디스크 크기 및 업로드할 이미지의 가상 크기를 확인합니다.

```
$ qemu-img info <file_name>
```

   예를 들면 다음과 같습니다.

```
$ qemu-img info CentOS-6-x86_64-GenericCloud-1508.qcow2

image: CentOS-6-x86_64-GenericCloud-1508.qcow2
file format: qcow2
virtual size: 8 GiB (8589934592 bytes)
disk size: 1.09 GiB
cluster_size: 65536
Format specific information:
compat: 0.10
refcount bits: 1
```

3. 이미지를 가져옵니다.

```
$ glance image-create-via-import --disk-format qcow2 --container-format bare --name centos_1 --file <file_name>
```

4. 이미지 ID를 기록합니다. 후속 단계에서 필요합니다.

5. 이미지를 가져오고 활성 상태인지 확인합니다.

```
$ glance image show <image_id>
```

6. Ceph Storage 노드에서 이미지 크기가 1단계의 출력에서 가상 크기보다 작은지 확인합니다.

```
$ sudo rbd -p images diff <image_id> | awk '{ SUM += $2 } END { print SUM/1024/1024/1024 " GB" }'

1.03906 GB
```

7. 선택 사항: **rbd_thin_provisioning** 이 컨트롤러 노드의 이미지 서비스 구성 파일에 구성되어 있는지 확인할 수 있습니다.

   a. SSH를 사용하여 컨트롤러 노드에 액세스합니다.

   ```
   $ ssh -A -t tripleo-admin@<controller_node_IP_address>
   ```

   b. 해당 컨트롤러 노드에서 **rbd_thin_provisioning** 이 **True** 와 같은지 확인합니다.

   ```
   $ sudo podman exec -it glance_api sh -c 'grep ^rbd_thin_provisioning /etc/glance/glance-api.conf'
   ```

1.9. 보안 METADEF API

RHOSP(Red Hat OpenStack Platform)에서 클라우드 관리자는 메타데이터 정의(metadef) API를 사용하여 키 값 쌍과 태그 메타데이터를 정의할 수 있습니다. 클라우드 관리자가 생성할 수 있는 metadef 네임스페이스, 오브젝트, 속성, 리소스 또는 태그 수에 제한이 없습니다.

이미지 서비스 정책은 metadef API를 제어합니다. 기본적으로 클라우드 관리자만 metadef API를 생성, 업데이트 또는 삭제할 수 있습니다. 이러한 제한은 metadef API가 무단 사용자에게 정보를 노출하지 못하도록 하며, 이미지 서비스(glance) 데이터베이스를 무제한 리소스로 채우는 악의적인 사용자의 위험을 완화하여 DoS(Denial of Service) 스타일 공격을 생성할 수 있습니다. 그러나 클라우드 관리자는 기본 정책을 재정의할 수 있습니다.

1.10. 클라우드 사용자를 위한 METADEF API 액세스 활성화

메타데이터 정의(metadef) API에 대한 쓰기 액세스 권한을 사용하는 클라우드 관리자는 기본 관리자 전용 정책을 재정의하여 모든 사용자가 해당 API에 액세스할 수 있도록 할 수 있습니다. 그러나 이러한 유형의 구성에서는 고객 이름 및 내부 프로젝트와 같은 실수로 민감한 리소스 이름을 누출시킬 가능성이 있습니다. 관리자는 모든 사용자에 대해 읽기 액세스만 활성화된 경우에도 이전에 생성된 리소스를 식별하기 위해 시스템을 감사해야 합니다.

프로세스

1. 클라우드 관리자로 언더클라우드에 로그인하여 정책 재정의를 위한 파일을 생성합니다. 예를 들면 다음과 같습니다.

```
$ cat open-up-glance-api-metadef.yaml
```

2. 모든 사용자에게 metadef API 읽기-쓰기 액세스를 허용하도록 정책 덮어쓰기 파일을 구성합니다.

```
GlanceApiPolicies: {
    glance-metadef_default: { key: 'metadef_default', value: '' },
    glance-get_metadef_namespace: { key: 'get_metadef_namespace', value: 'rule:metadef_default' },
    glance-get_metadef_namespaces: { key: 'get_metadef_namespaces', value:
```

```
'rule:metadef_default' },
    glance-modify_metadef_namespace: { key: 'modify_metadef_namespace', value:
'rule:metadef_default' },
    glance-add_metadef_namespace: { key: 'add_metadef_namespace', value:
'rule:metadef_default' },
    glance-delete_metadef_namespace: { key: 'delete_metadef_namespace', value:
'rule:metadef_default' },
    glance-get_metadef_object: { key: 'get_metadef_object', value: 'rule:metadef_default' },
    glance-get_metadef_objects: { key: 'get_metadef_objects', value: 'rule:metadef_default' },
    glance-modify_metadef_object: { key: 'modify_metadef_object', value:
'rule:metadef_default' },
    glance-add_metadef_object: { key: 'add_metadef_object', value: 'rule:metadef_default' },
    glance-delete_metadef_object: { key: 'delete_metadef_object', value: 'rule:metadef_default'
},
    glance-list_metadef_resource_types: { key: 'list_metadef_resource_types', value:
'rule:metadef_default' },
    glance-get_metadef_resource_type: { key: 'get_metadef_resource_type', value:
'rule:metadef_default' },
    glance-add_metadef_resource_type_association: { key:
'add_metadef_resource_type_association', value: 'rule:metadef_default' },
    glance-remove_metadef_resource_type_association: { key:
'remove_metadef_resource_type_association', value: 'rule:metadef_default' },
    glance-get_metadef_property: { key: 'get_metadef_property', value: 'rule:metadef_default'
},
    glance-get_metadef_properties: { key: 'get_metadef_properties', value:
'rule:metadef_default' },
    glance-modify_metadef_property: { key: 'modify_metadef_property', value:
'rule:metadef_default' },
    glance-add_metadef_property: { key: 'add_metadef_property', value: 'rule:metadef_default'
},
    glance-remove_metadef_property: { key: 'remove_metadef_property', value:
'rule:metadef_default' },
    glance-get_metadef_tag: { key: 'get_metadef_tag', value: 'rule:metadef_default' },
    glance-get_metadef_tags: { key: 'get_metadef_tags', value: 'rule:metadef_default' },
    glance-modify_metadef_tag: { key: 'modify_metadef_tag', value: 'rule:metadef_default' },
    glance-add_metadef_tag: { key: 'add_metadef_tag', value: 'rule:metadef_default' },
    glance-add_metadef_tags: { key: 'add_metadef_tags', value: 'rule:metadef_default' },
    glance-delete_metadef_tag: { key: 'delete_metadef_tag', value: 'rule:metadef_default' },
    glance-delete_metadef_tags: { key: 'delete_metadef_tags', value: 'rule:metadef_default' }
  }
```

참고

rule:metadeta_default 를 사용하도록 모든 metadef 정책을 구성해야 합니다.

3. 오버클라우드를 배포할 때 **-e** 옵션을 사용하여 배포 명령에 새 정책 파일을 추가합니다.

```
$ openstack overcloud deploy -e open-up-glance-api-metadef.yaml
```

2장. 이미지 관리

Image 서비스(glance)는 디스크 및 서버 이미지에 대한 검색, 등록 및 전달 서비스를 제공합니다. 서버 이미지를 복사하거나 스냅샷을 작성하고 저장할 수 있는 기능을 제공합니다. 저장된 이미지를 서버 운영 체제를 설치하고 개별적으로 서비스를 구성하는 것보다 새로운 서버를 빠르고 일관되게 위임하는 템플릿으로 사용할 수 있습니다.

2.1. 이미지 생성

RHEL(Red Hat Enterprise Linux) ISO 파일 또는 Windows ISO 파일을 사용하여 QCOW2 형식으로 RHOSP(Red Hat OpenStack Platform) 호환 이미지를 수동으로 생성합니다.

2.1.1. Red Hat OpenStack Platform에서 KVM 게스트 이미지 사용

준비된 RHEL KVM(커널 기반 가상 머신) 게스트 QCOW2 이미지를 사용할 수 있습니다.

- Red Hat Enterprise Linux 9 KVM 게스트 이미지
- Red Hat Enterprise Linux 8 KVM 게스트 이미지

이러한 이미지는 **cloud-init** 로 구성되며 SSH 키 프로비저닝이 올바르게 작동하려면 ec2-compatible 메타데이터 서비스를 활용해야 합니다.

준비된 Windows KVM 게스트 QCOW2 이미지를 사용할 수 없습니다.

참고

KVM 게스트 이미지의 경우:

- 이미지의 **root** 계정은 비활성화되어 있지만 **sudo** 액세스는 **cloud-user** 라는 특수 사용자에게 부여됩니다.
- 이 이미지에는 **루트** 암호가 설정되어 있지 않습니다.

루트 암호는 두 번째 필드에 **!!** 를 배치하여 **/etc/shadow** 에 잠겨 있습니다.

RHOSP 인스턴스의 경우 RHOSP 대시보드 또는 명령줄에서 ssh 키 쌍을 생성하고 해당 키 조합을 사용하여 인스턴스에 대한 SSH 공용 인증을 root로 수행합니다.

인스턴스가 시작되면 이 공개 키가 삽입됩니다. 그런 다음 키 쌍을 만들 때 다운로드한 개인 키를 사용하여 인증할 수 있습니다.

2.1.2. 사용자 지정 Red Hat Enterprise Linux 또는 Windows 이미지 만들기

사전 요구 사항

- 이미지를 생성할 Linux 호스트 머신입니다. 언더클라우드 또는 오버클라우드를 제외한 Linux 패키지를 설치하고 실행할 수 있는 머신일 수 있습니다.
- advanced-virt 리포지토리가 활성화되어 있습니다.

```
$ sudo subscription-manager repos --enable=advanced-virt-for-rhel-8-x86_64-rpms
```

- libvirt, virt-manager: 게스트 운영 체제를 생성하는 데 필요한 모든 패키지를 설치합니다.

```
$ sudo dnf module install -y virt
```

- libguestfs 툴을 설치하여 가상 머신 이미지에 액세스하고 수정합니다.

```
$ sudo dnf install -y libguestfs-tools-c
```

- RHEL 9 또는 8 ISO 파일 또는 Windows ISO 파일. RHEL ISO 파일에 대한 자세한 내용은 RHEL 9.0 Binary DVD 또는 RHEL 8.6 Binary DVD 를 참조하십시오. Windows ISO 파일이 없는 경우 Microsoft Evaluation Center 에서 평가판 이미지를 다운로드합니다.
- **Kickstart** 파일(RHEL만)을 변경하려면 텍스트 편집기입니다.

중요

언더클라우드에 **libguestfs-tools** 패키지를 설치하는 경우 언더클라우드에서 **tripleo_iscsid** 서비스와 포트 충돌을 방지하기 위해 **iscsid.socket** 을 비활성화합니다.

```
$ sudo systemctl disable --now iscsid.socket
```

2.1.3. Red Hat Enterprise Linux 9 이미지 생성

Red Hat Enterprise Linux 9 ISO 파일을 사용하여 QCOW2 형식으로 RHOSP(Red Hat OpenStack Platform) 호환 이미지를 수동으로 생성합니다.

참고

호스트 시스템에서 **[root@host]#** 을 사용하여 모든 명령을 실행해야 합니다.

절차

1. **virt-install** 을 사용하여 설치를 시작하십시오.

```
[root@host]# qemu-img create -f qcow2 rhel9.qcow2 10G.
[root@host]# virt-install --virt-type kvm --name rhel9 --ram 2048 \
--cdrom /var/lib/libvirt/images/rhel-9.0-x86_64-dvd.iso \
--disk rhel9.qcow2,format=qcow2 \
--network=bridge:virbr0 --graphics vnc,listen=127.0.0.1 \
--noautoconsole --os-type=linux --os-variant=rhel9.0
```

그러면 인스턴스가 시작되고 설치 프로세스가 시작됩니다.

참고

인스턴스가 자동으로 시작되지 않으면 **virt-viewer** 명령을 실행하여 콘솔을 확인합니다.

```
[root@host]# virt-viewer rhel9
```

2. 인스턴스를 구성합니다.

a. 초기 설치 관리자 부팅 메뉴에서 **Install Red Hat Enterprise Linux 9**를 선택합니다.

b. 적절한 **Language** 및 Cryostat **옵션을** 선택합니다.

c. 설치에서 사용하는 장치 유형에 대한 메시지가 표시되면 **자동 감지된 설치 미디어를** 선택합니다.

d. 설치 대상 유형에 대한 메시지가 표시되면 **로컬 표준 디스크** 를 선택합니다. 기타 스토리지 옵션의 경우 파티션 **자동 구성을 선택합니다**.

e. SSH **서버를 설치하는 기본** 서버 설치를 선택합니다.

f. 네트워크 및 호스트 이름의 경우 네트워크에 **eth0** 을 선택하고 장치의 호스트 이름을 선택합니다. 기본 호스트 이름은 **localhost.localdomain** 입니다.

g. **Root Password** 필드에 암호를 입력하고 **Confirm** 필드에 동일한 암호를 다시 입력합니다.

결과

설치 프로세스가 완료되고 **Complete!** 화면이 표시됩니다.

3. 설치가 완료되면 인스턴스를 재부팅하고 root 사용자로 로그인합니다.

4. 다음 값만 포함하도록 **/etc/sysconfig/network-scripts/ifcfg-eth0** 파일을 업데이트합니다.

```
TYPE=Ethernet
DEVICE=eth0
ONBOOT=yes
BOOTPROTO=dhcp
NM_CONTROLLED=no
```

5. 시스템을 재부팅합니다.

6. Content Delivery Network에 머신을 등록합니다.

```
# sudo subscription-manager register
# sudo subscription-manager attach --pool=Valid-Pool-Number-123456
# sudo subscription-manager repos --enable=rhel-9-server-rpms
```

7. 시스템을 업데이트합니다.

```
# dnf -y update
```

8. **cloud-init** 패키지를 설치합니다.

```
# dnf install -y cloud-utils-growpart cloud-init
```

9. **/etc/cloud/cloud.cfg** 구성 파일을 편집하고 **cloud_init_modules** 에서 다음을 추가합니다.

```
- resolv-conf
```

resolv-conf 옵션은 인스턴스가 처음 부팅될 때 **resolv.conf** 를 자동으로 구성합니다. 이 파일에는 **이름 서버,도메인** 및 기타 옵션과 같은 인스턴스와 관련된 정보가 포함되어 있습니다.

10. EC2 메타데이터 서비스에 액세스하는 문제를 방지하려면 **/etc/sysconfig/network** 에 다음 행을 추가합니다.

```
NOZEROCONF=yes
```

11. 콘솔 메시지가 대시보드 및 **nova console-log** 출력의 **로그** 탭에 표시되도록 하려면 **/etc/default/grub** 파일에 다음 부팅 옵션을 추가합니다.

```
GRUB_CMDLINE_LINUX_DEFAULT="console=tty0 console=ttyS0,115200n8"
```

12. **grub2-mkconfig** 명령을 실행합니다.

```
# grub2-mkconfig -o /boot/grub2/grub.cfg
```

출력은 다음과 같습니다.

```
Generating grub configuration file ...
Found linux image: /boot/vmlinuz-3.10.0-229.9.2.el9.x86_64
Found initrd image: /boot/initramfs-3.10.0-229.9.2.el9.x86_64.img
Found linux image: /boot/vmlinuz-3.10.0-121.el9.x86_64
Found initrd image: /boot/initramfs-3.10.0-121.el9.x86_64.img
Found linux image: /boot/vmlinuz-0-rescue-b82a3044fb384a3f9aeacf883474428b
Found initrd image: /boot/initramfs-0-rescue-b82a3044fb384a3f9aeacf883474428b.img
done
```

13. 생성된 이미지에 이 인스턴스의 서브스크립션 세부 정보가 포함되지 않도록 인스턴스를 등록합니다.

```
# subscription-manager repos --disable=*
# subscription-manager unregister
# dnf clean all
```

14. 인스턴스의 전원을 끕니다.

```
# poweroff
```

15. **virt-sysprep** 명령을 사용하여 문제 없이 인스턴스를 생성하는 데 사용할 수 있도록 이미지를 재설정하고 정리합니다.

```
[root@host]# virt-sysprep -d rhel9
```

16. 디스크 이미지 내의 사용 가능한 공간을 다시 호스트 내의 여유 공간으로 변환하여 이미지 크기를 줄입니다.

```
[root@host]# virt-sparsify --compress rhel9.qcow2 rhel9-cloud.qcow2
```

그러면 명령이 실행되는 위치에 새 **rhel9-cloud.qcow2** 파일이 생성됩니다. **rhel9-cloud.qcow2** 이미지 파일을 이미지 서비스에 업로드할 수 있습니다. 이 이미지를 RHOSP 배포에 업로드하는 방법에 대한 자세한 내용은 이미지 업로드를 참조하십시오.

2.1.4. Red Hat Enterprise Linux 8 이미지 생성

Red Hat Enterprise Linux 8 ISO 파일을 사용하여 QCOW2 형식으로 RHOSP(Red Hat OpenStack Platform) 호환 이미지를 수동으로 생성합니다.

참고

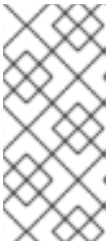
호스트 시스템에서 **[root@host]#** 을 사용하여 모든 명령을 실행해야 합니다.

절차

1. **virt-install** 을 사용하여 설치를 시작하십시오.

```
[root@host]# qemu-img create -f qcow2 rhel8.qcow2 10G
[root@host]# virt-install --virt-type kvm --name rhel8 --ram 2048 \
--cdrom /var/lib/libvirt/images/RHEL-8.0.0--x86_64-dvd1.iso \
--disk rhel8.qcow2,format=qcow2 \
--network=bridge:virbr0 --graphics vnc,listen=127.0.0.1 \
--noautoconsole --os-type=linux --os-variant=rhel8.0
```

그러면 인스턴스가 시작되고 설치 프로세스가 시작됩니다.



참고

인스턴스가 자동으로 시작되지 않으면 **virt-viewer** 명령을 실행하여 콘솔을 확인합니다.

```
[root@host]# virt-viewer rhel8
```

2. 인스턴스를 구성합니다.

   a. 초기 설치 프로그램 부팅 메뉴에서 **Install or upgrade an existing system**을 선택하고 설치 프롬프트를 따릅니다. 기본값을 수락합니다.
   디스크 설치 프로그램은 설치 전에 설치 미디어를 테스트하는 옵션을 제공합니다. **확인을** 선택하여 테스트 또는 **Skip** 을 실행하여 테스트 없이 진행합니다.

   b. 적절한 **Language** 및 Cryostat **옵션을** 선택합니다.

   c. 설치에서 사용하는 장치 유형에 대한 메시지가 표시되면 **Basic Storage Devices**를 선택합니다.

   d. 장치의 호스트 이름을 선택합니다. 기본 호스트 이름은 **localhost.localdomain** 입니다.

   e. **시간대** 및 **root** 암호를 설정합니다.

   f. 디스크 공간에 따라 설치하려는 옵션에서 **설치 유형을** 선택합니다.

   g. SSH **서버를 설치하는 기본** 서버 설치를 선택합니다.

   h. 설치 프로세스가 완료되고 **Congratulations가 완료되면 Red Hat Enterprise Linux 설치가 완전히** 표시됩니다.

3. 인스턴스를 재부팅하고 **root** 사용자로 로그인합니다.

4. 다음 값만 포함하도록 **/etc/sysconfig/network-scripts/ifcfg-eth0** 파일을 업데이트합니다.

```
TYPE=Ethernet
DEVICE=eth0
ONBOOT=yes
```

```
BOOTPROTO=dhcp
NM_CONTROLLED=no
```

5. 시스템을 재부팅합니다.
6. Content Delivery Network에 머신을 등록합니다.

```
# sudo subscription-manager register
# sudo subscription-manager attach --pool=Valid-Pool-Number-123456
# sudo subscription-manager repos --enable=rhel-8-server-rpms
```

7. 시스템을 업데이트합니다.

```
# dnf -y update
```

8. **cloud-init** 패키지를 설치합니다.

```
# dnf install -y cloud-utils-growpart cloud-init
```

9. **/etc/cloud/cloud.cfg** 구성 파일을 편집하고 **cloud_init_modules** 아래에 다음 콘텐츠를 추가합니다.

```
- resolv-conf
```

resolv-conf 옵션은 인스턴스가 처음 부팅될 때 **resolv.conf** 구성 파일을 자동으로 구성합니다. 이 파일에는 이름 서버 ,**도메인** 및 기타 옵션 **과** 같은 인스턴스 관련 정보가 포함되어 있습니다.

10. 네트워크 문제를 방지하려면 **/etc/udev/rules.d/75-persistent-net-generator.rules**:을 생성하십시오.

```
# echo "#" > /etc/udev/rules.d/75-persistent-net-generator.rules
```

이렇게 하면 **/etc/udev/rules.d/70-persistent-net.rules** 파일이 생성되지 않습니다. **/etc/udev/rules.d/70-persistent-net.rules** 가 생성되면 스냅샷에서 부팅할 때 네트워킹이 올바르게 작동하지 않을 수 있습니다. 네트워크 인터페이스는 **eth0** 이 아니라 **eth1** 로 생성되고 IP 주소가 할당되지 않습니다.

11. EC2 메타데이터 서비스에 액세스하는 문제를 방지하려면 **/etc/sysconfig/network** 에 다음 행을 추가합니다.

```
NOZEROCONF=yes
```

12. 콘솔 메시지가 대시보드 및 **nova console-log** 출력의 **로그** 탭에 표시되도록 하려면 **/etc/grub.conf** 파일에 다음 부팅 옵션을 추가합니다.

```
console=tty0 console=ttyS0,115200n8
```

13. 결과 이미지에 이 인스턴스에 대한 동일한 서브스크립션 세부 정보가 포함되지 않도록 가상 머신을 등록합니다.

```
# subscription-manager repos --disable=*
# subscription-manager unregister
# dnf clean all
```

14. 인스턴스의 전원을 끕니다.

```
# poweroff
```

15. **virt-sysprep** 명령을 사용하여 문제 없이 인스턴스를 생성하는 데 사용할 수 있도록 이미지를 재설정하고 정리합니다.

```
[root@host]# virt-sysprep -d rhel8
```

16. **virt-sparsify** 명령을 사용하여 이미지 크기를 줄입니다. 이 명령은 디스크 이미지 내의 사용 가능한 공간을 다시 변환하여 호스트 내의 사용 가능한 공간을 다시 변환합니다.

```
[root@host]# virt-sparsify --compress rhel8.qcow2 rhel8-cloud.qcow2
```

이렇게 하면 명령이 실행되는 위치에 새 **rhel8-cloud.qcow2** 파일이 생성됩니다.

참고

인스턴스에 적용되는 플레이버의 디스크 공간에 따라 이미지를 기반으로 인스턴스 파티션의 크기를 수동으로 조정해야 합니다.

rhel8-cloud.qcow2 이미지 파일을 이미지 서비스에 업로드할 준비가 되었습니다. 이 이미지를 RHOSP 배포에 업로드하는 방법에 대한 자세한 내용은 이미지 업로드를 참조하십시오.

2.1.5. Windows 이미지 생성

Windows ISO 파일을 사용하여 QCOW2 형식으로 RHOSP(Red Hat OpenStack Platform) 호환 이미지를 수동으로 생성합니다.

참고

호스트 시스템에서 **[root@host]#** 을 사용하여 모든 명령을 실행해야 합니다.

절차

1. **virt-install** 을 사용하여 설치를 시작하십시오.

```
[root@host]# virt-install --name=<name> \
--disk size=<size> \
--cdrom=<path> \
--os-type=windows \
--network=bridge:virbr0 \
--graphics spice \
--ram=<ram>
```

virt-install 매개변수의 다음 값을 바꿉니다.

- <name> true - Windows 인스턴스에 있는 이름입니다.
- <size> Cryostat- Cryostatdisk 크기(GB)입니다.
- Windows 설치 ISO 파일의 경로입니다.

- <RAM>유지 관리 - 요청된 RAM 양(MB)입니다.

참고

--os-type=windows 매개변수를 사용하면 시계가 Windows 게스트에 대해 올바르게 구성되어 있으며 Hyper-V 지원 기능을 활성화합니다. 이미지를 Image 서비스(glance)에 업로드하기 전에 이미지 메타데이터에서 **os_type=windows** 를 설정해야 합니다.

2. **virt-install** 은 게스트 이미지를 **/var/lib/libvirt/images/<name** > .**qcow2** 로 저장합니다. 게스트 이미지를 다른 곳에 유지하려면 **--disk** 옵션의 매개변수를 변경합니다.

```
--disk path=<filename>,size=<size>
```

<filename>을 인스턴스 이미지를 저장하는 파일 이름 및 선택적으로 해당 경로로 바꿉니다. 예를 들어 **path=win8.qcow2,size=8** 은 현재 작업 디렉터리에 **win8.qcow2** 라는 8GB 파일을 생성합니다.

작은 정보

게스트가 자동으로 시작되지 않으면 **virt-viewer** 명령을 실행하여 콘솔을 확인합니다.

```
[root@host]# virt-viewer <name>
```

Windows를 설치하는 방법에 대한 자세한 내용은 관련 Microsoft 설명서를 참조하십시오.

3. 새로 설치된 Windows 시스템이 가상화된 하드웨어를 사용하도록 허용하려면 VirtIO 드라이버를 설치해야 할 수 있습니다. 이렇게 하려면 이미지를 Windows 인스턴스에 CD-ROM 드라이브로 연결하여 설치합니다. **virtio-win** 패키지를 설치하려면 VirtIO ISO 이미지를 인스턴스에 추가하고 VirtIO 드라이버를 설치해야 합니다. 자세한 내용은 *가상화 구성 및 관리* 가이드에서 Windows 가상 머신용 KVM 반가상화 드라이버 설치를 참조하십시오.

4. 구성을 완료하려면 Windows 시스템에서 Cloudbase-Init 를 다운로드하고 실행합니다. Cloudbase-Init 설치가 끝나면 **Run Sysprep** 및 Cryostat 확인란 **을** 선택합니다. **Sysprep** 툴은 특정 Microsoft 서비스에서 사용하는 OS ID를 생성하여 게스트를 고유하게 만듭니다.

중요

Red Hat은 Cloudbase-Init에 대한 기술 지원을 제공하지 않습니다. 문제가 발생하면 Contact Cloudbase Solutions 를 참조하십시오.

Windows 시스템이 종료되면 < **name>.qcow2** 이미지 파일을 이미지 서비스에 업로드할 준비가 된 것입니다. 이 이미지를 RHOSP 배포에 업로드하는 방법에 대한 자세한 내용은 이미지 업로드를 참조하십시오.

2.1.5.1. 메타데이터 속성

Compute 서비스(nova)는 **libosinfo** 데이터를 사용하여 기본 장치 모델을 설정할 수 있도록 더 이상 지원되지 않습니다. 대신 다음 이미지 메타데이터 속성을 사용하여 인스턴스에 최적의 가상 하드웨어를 구성합니다.

- **os_distro**
- **os_version**

- **hw_cdrom_bus**
- **hw_disk_bus**
- **hw_scsi_model**
- **hw_vif_model**
- **hw_video_model**
- **hypervisor_type**

이러한 메타데이터 속성에 대한 자세한 내용은 이미지 구성 매개변수를 참조하십시오.

2.1.6. UEFI Secure Boot 용 이미지 생성

오버클라우드에 UEFI Secure Boot Compute 노드가 포함된 경우 클라우드 사용자가 Secure Boot 인스턴스를 시작하는 데 사용할 수 있는 Secure Boot 인스턴스 이미지를 생성할 수 있습니다.

프로세스

1. UEFI Secure Boot의 새 이미지를 생성합니다.

```
$ openstack image create --file <base_image_file> uefi_secure_boot_image
```

   - & **lt;base_image_file** >을 UEFI 및 GUID 파티션 테이블(GPT) 표준을 지원하고 EFI 시스템 파티션을 포함하는 이미지 파일로 바꿉니다.

2. 기본 머신 유형이 **q35** 가 아닌 경우 시스템 유형을 **q35** 로 설정합니다.

```
$ openstack image set --property hw_machine_type=q35 uefi_secure_boot_image
```

3. UEFI Secure Boot 호스트에서 인스턴스를 예약하도록 지정합니다.

```
$ openstack image set \
 --property hw_firmware_type=uefi \
 --property os_secure_boot=required \
 uefi_secure_boot_image
```

2.2. 이미지 업로드

이미지를 업로드합니다.

프로세스

- **glance image-create** 명령과 **속성** 옵션을 사용하여 이미지를 업로드합니다.
  예를 들면 다음과 같습니다.

```
$ glance image-create --name <NAME> \
    --is-public true --disk-format qcow2 \
    --container-format bare \
    --file <IMAGE_FILE> \
    --property <IMAGE_METADATA>
```

- **glance image-create** 명령 옵션 목록은 Image 서비스(glance) 명령 옵션을 참조하십시오.
- 속성 키 목록은 이미지 구성 매개변수를 참조하십시오.

2.3. 이미지 업데이트

이미지를 업데이트합니다.

프로세스

- **glance image-update** 명령을 **속성** 옵션과 함께 사용하여 이미지를 업데이트합니다.
  예를 들면 다음과 같습니다.

```
$ glance image-update IMG-UUID \
    --property architecture=x86_64
```

- **glance image-update** 명령 옵션 목록은 Image 서비스(glance) 명령 옵션을 참조하십시오.
- 속성 키 목록은 이미지 구성 매개변수를 참조하십시오.

2.4. 이미지 가져오기

다음 두 가지 방법 중 하나를 사용하여 Image 서비스(glance)로 이미지를 가져올 수 있습니다.

- **web-download** 를 사용하여 URI에서 이미지를 가져옵니다.
- **glance-direct** 를 사용하여 로컬 파일 시스템에서 이미지를 가져옵니다.

web-download 방법은 기본적으로 활성화되어 있습니다. 클라우드 관리자는 가져오기 방법을 구성합니다. **glance import-info** 명령을 실행하여 사용 가능한 가져오기 옵션을 나열할 수 있습니다.

2.4.1. 원격 URI에서 이미지 가져오기

web-download 방법을 사용하여 원격 URI에서 이미지를 복사할 수 있습니다.

1. 이미지를 생성하고 가져올 이미지의 URI를 지정합니다.

```
$ glance image-create-via-import \
    --container-format <CONTAINER FORMAT> \
    --disk-format <DISK-FORMAT> \
    --name <NAME> \
    --import-method web-download \
    --uri <URI>
```

- < **CONTAINER FORMAT** >를 이미지에 대해 설정된 컨테이너 형식(없음, ami, ari, aki, bare, ovf, ova, docker)으로 바꿉니다.
- < **DISK-FORMAT** >를 이미지에 대해 설정된 디스크 형식(없음, ami, ari, aki, vhd, vhdx, vmdk, raw, qcow2, vdi, iso, ploop)으로 바꿉니다.
- & **lt;NAME&** gt;을 이미지에 대한 설명이 포함된 이름으로 바꿉니다.
- &lt **;URI&** gt;를 이미지의 URI로 바꿉니다.

2. **glance image-show <IMAGE_ID> 명령을 사용하여 이미지** 의 가용성을 확인할 수 있습니다.
    - & **lt;IMAGE_ID&** gt;를 이미지 생성 중에 제공한 ID로 바꿉니다.

이미지 서비스 **웹 다운로드** 방법은 2단계 프로세스를 사용하여 가져오기를 수행합니다.

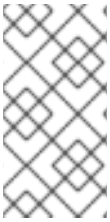
1. **웹 다운로드** 메서드는 이미지 레코드를 생성합니다.
2. **웹 다운로드** 방법은 지정된 URI에서 이미지를 검색합니다.

URI에는 선택적 거부 목록 및 허용 목록 필터링이 적용됩니다.

Image Property Cryostat 플러그인은 메타데이터 속성을 이미지에 삽입할 수 있습니다. 이러한 삽입된 속성은 이미지 인스턴스가 시작되는 컴퓨팅 노드를 결정합니다.

2.4.2. 로컬 볼륨에서 이미지 가져오기

glance-direct 메서드는 이미지 ID를 생성하는 이미지 레코드를 생성합니다. 이미지가 로컬 볼륨에서 이미지 서비스에 업로드되면 스테이징 영역에 저장되고 구성된 검사를 통과한 후 활성화됩니다. **glance-direct** 방법을 사용하려면 HA(고가용성) 구성에 사용할 때 공유 스테이징 영역이 필요합니다.



참고

공통 스테이징 영역이 없는 경우 HA 환경에서 **glance-direct** 방법을 사용하는 이미지 업로드가 실패할 수 있습니다. HA 활성-활성 환경에서는 API 호출이 이미지 서비스 컨트롤러에 배포됩니다. 다운로드 API 호출을 API 호출과 다른 컨트롤러로 보내 이미지를 업로드할 수 있습니다.

glance-direct 방법은 세 가지 호출을 사용하여 이미지를 가져옵니다.

- **glance image-create**
- **Glance 이미지 단계**
- **Glance image-import**

glance image-create-via-import 명령을 사용하여 하나의 명령으로 이러한 세 가지 호출을 모두 수행할 수 있습니다.

```
$ glance image-create-via-import \
   --container-format <CONTAINER FORMAT> \
   --disk-format <DISK-FORMAT> \
   --name <NAME> \
   --file </PATH/TO/IMAGE>
```

- < **CONTAINER FORMAT** >, < **DISK-FORMAT** >, < **NAME** > , < **/PATH/TO/IMAGE** >를 이미지와 관련된 값으로 바꿉니다.

이미지가 스테이징 영역에서 백엔드 위치로 이동하면 이미지가 나열됩니다. 그러나 이미지가 활성화되는 데 다소 시간이 걸릴 수 있습니다.

glance image-show <IMAGE_ID> 명령을 사용하여 이미지 의 가용성을 확인할 수 있습니다.

- & **lt;IMAGE_ID** 를 이미지 생성 중에 제공한 ID로 바꿉니다.

2.5. 이미지 삭제

프로세스

- **glance image-delete** 명령을 사용하여 하나 이상의 이미지를 삭제합니다.

```
$ glance image-delete <IMAGE_ID> [<IMAGE_ID> ...]
```

  - <IMAGE_ID>를 삭제하려는 이미지의 ID로 바꿉니다.

참고

glance image-delete 명령은 이미지 및 이미지의 모든 사본과 이미지 인스턴스 및 메타데이터를 영구적으로 삭제합니다.

2.6. 이미지 숨기거나 숨기기

사용자에게 제공되는 일반 목록에서 공용 이미지를 숨길 수 있습니다. 예를 들어 더 이상 사용되지 않는 CentOS 7 이미지를 숨기고 최신 버전만 표시하여 사용자 환경을 단순화할 수 있습니다. 사용자는 숨겨진 이미지를 검색하고 사용할 수 있습니다.

이미지를 숨기려면 다음을 수행합니다.

```
glance image-update <image_id> --hidden 'true'
```

숨겨진 이미지를 생성하려면 **glance image-create** 명령에 **--hidden** 인수를 추가합니다.

이미지를 분리하려면 다음을 수행합니다.

```
glance image-update <image_id> --hidden 'false'
```

숨겨진 이미지 표시
숨겨진 이미지를 나열하려면 다음을 수행합니다.

```
glance image-list --hidden 'true'
```

2.7. 이미지 변환 활성화

GlanceImageImportPlugins 매개변수를 활성화하여 QCOW2 이미지를 Image 서비스(glance)에 업로드할 수 있습니다. 그런 다음 QCOW2 이미지를 RAW 형식으로 변환할 수 있습니다.

참고

Red Hat Ceph Storage RADOS Block Device(RBD)를 사용하여 이미지를 저장하고 Nova 인스턴스를 부팅하면 이미지 변환이 자동으로 활성화됩니다.

이미지 변환을 활성화하려면 다음 매개변수 값이 포함된 환경 파일을 생성합니다. **openstack overcloud deploy** 명령에 **-e** 옵션을 사용하여 새 환경 파일을 포함합니다.

```
parameter_defaults:
  GlanceImageImportPlugins:'image_conversion'
```

이미지 관리에 이미지 서비스 명령줄 클라이언트를 사용합니다.

2.7.1. 이미지를 RAW 형식으로 변환

Red Hat Ceph Storage는 VM(가상 머신) 디스크를 호스팅하는 QCOW2 이미지를 저장할 수 있지만 사용을 지원하지 않습니다.

QCOW2 이미지를 업로드하고 VM을 생성할 때 컴퓨팅 노드는 이미지를 다운로드하고 이미지를 RAW로 변환한 후 다시 Ceph에 업로드할 수 있습니다. 이 프로세스는 특히 병렬 VM 생성 중에 VM을 생성하는 데 걸리는 시간에 영향을 미칩니다.

예를 들어 여러 VM을 동시에 생성할 때 변환된 이미지를 Ceph 클러스터에 업로드하면 이미 실행 중인 워크로드에 영향을 미칠 수 있습니다. 업로드 프로세스는 이러한 IOPS 워크로드와 스토리지 대응을 방해할 수 있습니다.

Ceph에서 VM을 보다 효율적으로 부팅하려면(임시 백엔드 또는 볼륨에서 부팅) Glance 이미지 형식은 RAW여야 합니다.

절차

1. 이미지를 RAW로 변환하면 원본 QCOW2 이미지 파일보다 크기가 큰 이미지가 발생할 수 있습니다. 변환 전에 다음 명령을 실행하여 최종 RAW 이미지 크기를 확인합니다.

```
qemu-img info <image>.qcow2
```

2. 이미지를 QCOW2에서 RAW 형식으로 변환합니다.

```
qemu-img convert -p -f qcow2 -O raw <original qcow2 image>.qcow2 <new raw image>.raw
```

2.7.1.1. Image 서비스(glance)에서 디스크 형식 구성

GlanceDiskFormats 매개변수를 사용하여 디스크 형식을 활성화하거나 거부하도록 Image 서비스(glance)를 구성할 수 있습니다.

절차

1. 언더클라우드 호스트에 **stack** 사용자로 로그인합니다.
2. 언더클라우드 인증 정보 파일을 가져옵니다.

```
$ source ~/stackrc
```

3. 환경 파일에 **GlanceDiskFormats** 매개변수를 포함합니다(예: **glance_disk_formats.yaml**):

```
parameter_defaults:
  GlanceDiskFormats:
    - <disk_format>
```

- 예를 들어 다음 구성을 사용하여 RAW 및 ISO 디스크 형식만 활성화합니다.

```
parameter_defaults:
  GlanceDiskFormats:
  - raw
  - iso
```

- 다음 예제 구성을 사용하여 QCOW2 디스크 이미지를 거부합니다.

```
parameter_defaults:
  GlanceDiskFormats:
  - raw
  - iso
  - aki
  - ari
  - ami
```

4. 해당 환경과 관련된 다른 환경 파일과 함께 **openstack overcloud deploy** 명령에 새 구성이 포함된 환경 파일을 포함합니다.

```
$ openstack overcloud deploy --templates \
  -e <overcloud_environment_files> \
  -e <new_environment_file> \
  …
```

- & **lt;overcloud_environment_files** >를 배포에 속하는 환경 파일 목록으로 바꿉니다.
- & **lt;new_environment_file** >을 새 구성이 포함된 환경 파일로 바꿉니다.

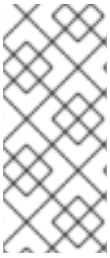
RHOSP에서 사용 가능한 디스크 형식에 대한 자세한 내용은 이미지 구성 매개변수를 참조하십시오.

2.7.2. RAW 형식으로 이미지 저장

GlanceImageImportPlugins 매개변수를 활성화하면 다음 명령을 실행하여 이전에 생성된 이미지를 RAW 형식으로 저장합니다.

```
$ glance image-create-via-import \
   --disk-format qcow2 \
   --container-format bare \
   --name NAME \
   --visibility public \
   --import-method web-download \
   --uri http://server/image.qcow2
```

- **--name**, **NAME** 을 이미지 이름으로 바꿉니다. **glance image-list** 에 표시될 이름입니다.
- **--uri** 의 경우 **http://server/image.qcow2** 를 QCOW2 이미지의 위치 및 파일 이름으로 교체합니다.



참고

이 명령 예제에서는 이미지 레코드를 생성하고 **web-download** 방법을 사용하여 가져옵니다. glance-api는 가져오기 프로세스 중에 **--uri** 위치에서 이미지를 다운로드합니다. **web-download** 를 사용할 수 없는 경우 **glanceclient** 가 이미지 데이터를 자동으로 다운로드할 수 없습니다. **glance import-info** 명령을 실행하여 사용 가능한 이미지 가져오기 방법을 나열합니다.

3장. 이미지 가져오기 및 공유 스테이징

OpenStack Image 서비스(glance)의 기본 설정은 RHOSP(Red Hat OpenStack Platform)를 설치할 때 사용하는 heat 템플릿에 따라 결정됩니다. 이미지 서비스 heat 템플릿은 **deployment/glance/glance-api-container-puppet.yaml** 입니다.

web-download 방법 또는 **glance-direct** 방법을 사용하여 이미지를 가져올 수 있습니다. 이러한 가져오기 방법에 대한 자세한 내용은 이미지 가져오기를 참조하십시오.

3.1. GLANCE-SETTINGS.YAML 파일 생성 및 배포

사용자 지정 환경 파일을 사용하여 가져오기 매개변수를 구성합니다. 이러한 매개변수는 코어 heat 템플릿 컬렉션에 있는 기본값을 재정의합니다. 예제 환경 콘텐츠에는 상호 운용 가능한 이미지 가져오기에 대한 매개변수가 포함되어 있습니다.

```
parameter_defaults:
  # Configure NFS backend
  GlanceBackend: file
  GlanceNfsEnabled: true
  GlanceNfsShare: 192.168.122.1:/export/glance

  # Enable glance-direct import method
  GlanceEnabledImportMethods: glance-direct,web-download

  # Configure NFS staging area (required for glance-direct import method)
  GlanceStagingNfsShare: 192.168.122.1:/export/glance-staging
```

GlanceBackend,**GlanceNfsEnabled** 및 **GlanceNfsShare** 매개 변수는 *Overcloud 매개변수* 가이드에 정의되어 있습니다.

상호 운용 가능한 이미지 가져오기를 위해 두 개의 새 매개 변수를 사용하여 가져오기 방법과 공유 NFS 스테이징 영역을 정의합니다.

GlanceEnabledImportMethods

사용 가능한 가져오기 메서드, web-download(기본값) 및 glance-direct를 정의합니다. 이 매개변수는 web-download 이외의 추가 메서드를 활성화하려면 경우에만 필요합니다.

GlanceStagingNfsShare

glance-direct 가져오기 방법에서 사용하는 NFS 스테이징 영역을 구성합니다. 이 공간은 고가용성 클러스터 구성의 노드 간에 공유할 수 있습니다. 이 매개변수를 사용하려면 **GlanceNfsEnabled** 매개변수를 **true** 로 설정해야 합니다.

절차

1. 새 파일(예: **glance-settings.yaml**)을 만듭니다. 예제의 구문을 사용하여 이 파일을 채웁니다.
2. **openstack overcloud deploy** 명령에 **glance-settings.yaml** 파일과 배포와 관련된 기타 환경 파일을 포함합니다.

```
$ openstack overcloud deploy --templates -e glance-settings.yaml
```

환경 파일 사용에 대한 자세한 내용은 *director를 사용하여 Red Hat OpenStack Platform 설치 및 관리를* 참조하십시오.

3.2. 이미지 웹-가져오기 소스 제어

URI 블록 목록을 추가하고 선택적 **glance-image-import.conf** 파일에 web-import 이미지 다운로드 소스를 제한할 수 있습니다.

세 가지 수준에서 이미지 소스 URI를 허용하거나 차단할 수 있습니다.

- 스키마(allowed_schemes, disallowed_schemes)
- host (allowed_hosts, disallowed_hosts)
- 포트(allowed_ports, disallowed_ports)

모든 수준에서 allowlist와 blocklist를 둘 다 지정하면 허용 목록이 지원되고 blocklist는 무시됩니다.

Image 서비스(glance)는 다음 결정 논리를 적용하여 이미지 소스 URI의 유효성을 검사합니다.

1. 스키마가 확인됩니다.
    a. 누락된 스키마: reject
    b. 허용 목록이 있고 스키마가 allowlist: reject에 존재하지 않는 경우입니다. 그렇지 않으면 C를 건너뛰고 2로 계속합니다.
    c. blocklist가 있고 스키마가 blocklist: reject에 있는 경우입니다.
2. 호스트 이름을 확인합니다.
    a. 누락된 호스트 이름: reject
    b. 허용 목록이 있고 allowlist: reject에 호스트 이름이 없습니다. 그렇지 않으면 C를 건너뛰고 3으로 계속 진행합니다.
    c. blocklist가 있고 호스트 이름이 blocklist: reject에 있습니다.
3. URI에 포트가 있으면 포트가 확인됩니다.
    a. 허용 목록이 있고 포트가 allowlist: reject에 존재하지 않는 경우입니다. 그렇지 않으면 B를 건너뛰고 4로 계속합니다.
    b. blocklist가 있고 포트가 blocklist: reject에 있습니다.
4. URI가 유효한 것으로 허용됩니다.

허용 목록에 추가하거나 블록 목록에 추가하지 않고 스키마를 허용하면 URI에 포트를 포함하지 않고 해당 구성 요소에 기본 포트를 사용하는 모든 URI가 허용됩니다. URI에 포트가 포함된 경우 기본 의사 결정 논리에 따라 URI의 유효성을 검사합니다.

3.3. 이미지 가져오기 예

예를 들어 FTP의 기본 포트는 21입니다. *ftp* 는 허용 목록에 있는 스키마이므로 이 URL은 ftp://example.org/some/resource. 그러나 21이 포트 허용 목록에 없으므로 동일한 리소스에 대한 이 URL은 거부됩니다. ftp://example.org:21/some/resource.

```
allowed_schemes = [http,https,ftp]
disallowed_schemes = []
allowed_hosts = []
```

```
disallowed_hosts = []
allowed_ports = [80,443]
disallowed_ports = []
```

3.4. 기본 이미지 가져오기 블록 목록 및 허용 목록 설정

glance-image-import.conf 파일은 다음 기본 옵션이 포함된 선택적 파일입니다.

- allowed_schemes - [*http*, *https*]
- disallowed_schemes - 빈 목록
- allowed_hosts - 빈 목록
- disallowed_hosts - 빈 목록
- allowed_ports - [80, 443]
- disallowed_ports - 빈 목록

기본값을 사용하는 경우 최종 사용자는 **http** 또는 **https** 스키마만 사용하여 URI에 액세스할 수 있습니다. 사용자가 지정할 수 있는 포트는 **80** 및 **443** 입니다. 사용자는 포트를 지정할 필요가 없지만 포트를 지정하는 경우 **80** 또는 **443** 이어야 합니다.

glance-image-import.conf 파일은 이미지 서비스 소스 코드 트리의 **etc/** 하위 디렉터리에서 찾을 수 있습니다. Red Hat OpenStack Platform 릴리스에 대한 올바른 분기를 찾고 있는지 확인하십시오.

3.5. 이미지 압축 해제

이미지 압축 플러그인은 상호 운용 가능한 이미지 가져오기를 위해 자동화된 이미지 압축을 구현합니다. **web-download** 가져오기 방법을 사용하려는 환경에서 이 플러그인을 사용할 수 있지만 이미지 공급자는 압축 이미지만 제공합니다. 이미지 압축 해제는 클라이언트와 이미지 서비스(glance) 간의 네트워크 사용량을 최적화합니다.

이미지 압축 해제에 대한 다음 아카이브 유형이 지원됩니다.

- ZIP
- LHA/LZH
- GZIP

LHA/LZH 형식은 Python 3 lhafile 종속성 라이브러리가 설치된 경우에만 지원됩니다. Python 3 lhafile 종속성 라이브러리가 설치되지 않은 경우 LHA/LZH 파일이 있는 가져오기 작업이 실패합니다. 현재 플러그인은 TAR.GZ와 같은 다중 계층 아카이브는 지원하지 않습니다.

3.5.1. 이미지 압축 해제 구현

이미지 압축 플러그인은 상호 운용 가능한 이미지 가져오기 워크플로우의 일부로만 사용할 수 있습니다 (**POST v2/images/{image_id}/import**). 플러그인은 이미지 데이터 업로드 호출(**PUT v2/images/{image_id}/file**)에 영향을 미치지 않습니다.

서비스 사용자가 상호 운용 가능한 이미지 가져오기 워크플로를 사용하도록 하려면 Image 서비스 (glance) **policy.json** 파일에서 **upload_image** 정책을 제한합니다. 기본적으로 이 정책은 제한되지 않으며 권한이 있는 모든 사용자가 이미지 업로드를 호출할 수 있습니다.

사전 요구 사항

Image Decompression 플러그인을 사용하려면 **image_decompression** 을 **glance-image-import.conf** 파일에 추가해야 합니다.

```
[image_import_opts]
image_import_plugins = ['image_decompression']
```

프로세스

- 이미지 서비스 **policy.json** 파일에서 **upload_image** 정책을 제한합니다. 이 예제에서는 클라우드 관리자 및 서비스 사용자만 이미지 업로드 호출을 수행할 수 있습니다.

```
"upload_image": "role:admin or (service_user_id:<uuid of nova user>) or (service_roles:
<service user role>)"
```

  - & **lt;uuid of nova user&** gt;를 service 사용자의 UUID로 바꿉니다.
  - & **lt;service user role** >을 서비스 사용자에 대해 생성되고 신뢰할 수 있는 서비스에 할당된 역할로 바꿉니다.

3.5.2. 이미지 가져오기 워크플로에 대한 여러 플러그인 활성화

glance-image-import.conf 파일에서 **image_import_plugins** 옵션을 구성하여 이미지 가져오기 워크플로에 대해 여러 플러그인을 활성화할 수 있습니다. 플러그인은 병렬로 실행되지 않습니다. **image_import_plugins** 목록에 표시되는 순서대로 실행됩니다.

프로세스

- **glance-image-import.conf** 파일에서 여러 플러그인을 구성합니다. 이 예제에서는 이미지를 변환하기 전에 압축을 풀어야 합니다. 목록에 있는 플러그인을 주문하여 이미지 압축을 먼저 확인합니다.

```
[image_import_opts]
image_import_plugins = ['image_decompression', 'image_conversion']
[image_conversion]
output_format = raw
```

3.6. 이미지 가져오기에 메타데이터 삽입하여 VM 시작 위치 제어

최종 사용자는 이미지 서비스에 이미지를 업로드하고 이러한 이미지를 사용하여 VM을 시작할 수 있습니다. 이러한 사용자 제공(관리자 이외의) 이미지는 특정 컴퓨팅 노드 세트에서 시작해야 합니다. 계산 노드에 인스턴스를 할당하면 이미지 메타데이터 속성으로 제어됩니다.

Image Property Cryostat 플러그인은 가져오는 동안 이미지에 메타데이터 속성을 삽입합니다. **glance-image-import.conf** 파일의 [image_import_opts] 및 [inject_metadata_properties] 섹션을 편집하여 속성을 지정합니다.

Image Property Cryostat 플러그인을 활성화하려면 **[image_import_opts]** 섹션에 다음 행을 추가합니다.

```
[image_import_opts]
image_import_plugins = [inject_image_metadata]
```

특정 사용자 집합에서 제공하는 이미지로 메타데이터 삽입을 제한하려면 **ignore_user_roles** 매개변수를 설정합니다. 예를 들어 다음 구성을 사용하여 **property1** 의 값을 하나의 값과admin이 아닌 사용자가 다운로드한 이미지에 **property2** 의 두 값을 삽입합니다.

```
[DEFAULT]
[image_conversion]
[image_import_opts]
image_import_plugins = [inject_image_metadata]
[import_filtering_opts]
[inject_metadata_properties]
ignore_user_roles = admin
inject = PROPERTY1:value,PROPERTY2:value;another value
```

ignore_user_roles 매개변수는 플러그인이 무시한 ID 서비스(keystone) 역할의 쉼표로 구분된 목록입니다. 즉, 이미지 가져오기 호출을 수행하는 사용자에게 이러한 역할이 있는 경우 플러그인은 이미지에 속성을 삽입하지 않습니다.

매개변수 **삽입** 은 가져온 이미지의 이미지 레코드에 삽입되는 쉼표로 구분된 속성 및 값 목록입니다. 각 속성 및 값은 콜론 **(':')** 으로 인용하고 구분해야 합니다.

glance-image-import.conf 파일은 이미지 서비스 소스 코드 트리의 **etc/** 하위 디렉터리에서 찾을 수 있습니다. Red Hat OpenStack Platform 릴리스에 대한 올바른 분기를 찾고 있는지 확인하십시오.

4장. 여러 저장소가 있는 이미지 서비스

RHOSP(Red Hat OpenStack Platform) 이미지 서비스(glance)는 분산 엣지 아키텍처가 있는 여러 저장소 사용을 지원하므로 모든 엣지 사이트에 이미지 풀을 사용할 수 있습니다.

4.1. 여러 저장소에 있는 이미지 사본

분산 엣지 아키텍처와 함께 여러 저장소를 사용하면 모든 에지 사이트에 이미지 풀을 사용할 수 있습니다. 허브 사이트라고도 하는 중앙 사이트와 에지 사이트 간에 이미지를 복사할 수 있습니다.

이미지 메타데이터에는 각 복사본의 위치가 포함되어 있습니다. 예를 들어 두 에지 사이트에 있는 이미지는 중앙 사이트와 두 개의 에지 사이트라는 세 위치가 있는 단일 UUID로 노출됩니다. 즉, 여러 저장소에 단일 UUID를 공유하는 이미지 데이터 사본이 있을 수 있습니다. 위치에 대한 자세한 내용은 이미지 위치 이해를 참조하십시오.

모든 에지 사이트의 RADOS 블록 장치(RBD) 이미지 풀을 사용하면 Ceph RBD COW(Copy-On-Write) 및 스냅샷 계층 기술을 사용하여 VM(가상 머신)을 빠르게 부팅할 수 있습니다. 즉, 볼륨에서 VM을 부팅하고 실시간 마이그레이션을 수행할 수 있습니다. Ceph RBD를 사용한 계층 지정에 대한 자세한 내용은 블록 장치 *가이드의 Ceph 블록 장치 계층 지정을* 참조하십시오.

에지 사이트에서 인스턴스를 시작하면 필요한 이미지가 로컬 Image 서비스(glance) 저장소에 자동으로 복사됩니다. 그러나 인스턴스 시작 중에 시간을 절약하기 위해 glance multistore를 사용하여 중앙 이미지 저장소에서 에지 사이트로 미리 이미지를 복사할 수 있습니다.

4.2. 스토리지 엣지 아키텍처 요구사항

에지 사이트에서 이미지를 사용하려면 다음 요구 사항을 참조하십시오.

- 각 이미지의 사본은 중앙 위치의 Image 서비스(glance)에 있어야 합니다.
- 이미지를 다른 에지 사이트에 복사하려면 먼저 에지 사이트의 이미지를 중앙 위치로 복사해야 합니다.
- Red Hat Ceph Storage를 사용하여 DCN(Distributed Compute Node) 아키텍처를 배포할 때 원시 이미지를 사용해야 합니다.
- RADOS 블록 장치(RBD)는 이미지, 컴퓨팅 및 블록 스토리지 서비스의 스토리지 드라이버여야 합니다.
- 각 사이트에 대해 **NovaComputeAvailabilityZone** 및 **CinderStorageAvailabilityZone** 매개변수에 동일한 값을 할당해야 합니다.

4.3. 여러 저장소로 이미지 가져오기

상호 운용 가능한 이미지 가져오기 워크플로를 사용하여 여러 Ceph Storage 클러스터로 이미지 데이터를 가져옵니다. 로컬 파일 시스템 또는 웹 서버를 통해 사용할 수 있는 Image 서비스(glance)로 이미지를 가져올 수 있습니다.

웹 서버에서 이미지를 가져오는 경우 이미지를 한 번에 여러 저장소로 가져올 수 있습니다. 웹 서버에서 이미지를 사용할 수 없는 경우 로컬 파일 시스템의 이미지를 중앙 저장소로 가져온 다음 추가 저장소에 복사할 수 있습니다. 자세한 내용은 기존 이미지 복사를 여러 저장소에 참조하십시오.

이미지 관리에 이미지 서비스 명령줄 클라이언트를 사용합니다.

중요

중앙 위치에서 이미지를 사용하는 인스턴스가 없는 경우에도 중앙 사이트에 이미지 복사본을 항상 저장합니다. 이미지 서비스로 이미지를 가져오는 방법에 대한 자세한 내용은 분산 컴퓨팅 노드 아키텍처 배포 가이드를 참조하십시오.

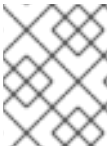
4.3.1. 이미지 가져오기 실패 관리

--allow-failure 매개변수를 사용하여 이미지 가져오기 작업의 오류를 관리할 수 있습니다.

- **--allow-failure** 매개변수의 값이 **true** 인 경우 첫 번째 저장소에서 데이터를 성공적으로 가져온 후 이미지 상태가 **활성화** 됩니다. 이 설정은 기본 설정입니다. **os_glance_failed_import** 이미지 속성을 사용하여 이미지 데이터를 가져오지 못한 저장소 목록을 볼 수 있습니다.
- **--allow-failure** 매개변수의 값을 **false** 로 설정하면 지정된 모든 저장소에서 데이터를 성공적으로 가져온 후에만 이미지 상태가 **활성화** 됩니다. 이미지 데이터를 가져오는 저장소 실패로 인해 이미지 상태가 **실패** 하게 됩니다. 이미지는 지정된 저장소 중 하나로 가져오지 않습니다.

4.3.2. 여러 저장소로 이미지 데이터 가져오기

--allow-failure 매개변수의 기본 설정은 **true** 이므로 일부 저장소가 이미지 데이터를 가져오지 못하는 경우 명령에 매개 변수를 포함할 필요가 없습니다.



참고

이 절차에서는 이미지 데이터를 성공적으로 가져오기 위해 모든 저장소가 필요하지는 않습니다.

절차

- 지정된 여러 저장소로 이미지 데이터를 가져옵니다.

```
$ glance image-create-via-import \
--container-format bare \
--name IMAGE-NAME \
--import-method web-download \
--uri URI \
--stores STORE1,STORE2,STORE3
```

  - *IMAGE-NAME* 을 가져올 이미지의 이름으로 교체합니다.
  - *URI* 를 이미지 URI로 바꿉니다.
  - *STORE1,STORE2, STORE3* 을 이미지 데이터를 가져오려는 저장소 이름으로 교체합니다.
  - 또는 모든 저장소에 이미지를 업로드하려면 **--stores** 를 **--all-stores true** 로 바꿉니다.

참고

QCOW2 이미지를 RAW 형식으로 자동 변환하는 **glance image-create-via-import** 명령은 **web-download** 방법에서만 작동합니다. **glance-direct** 방법을 사용할 수 있지만 구성된 공유 파일 시스템이 있는 배포에서만 작동합니다.

4.3.3. 실패 없이 여러 저장소로 이미지 데이터 가져오기

이 절차에서는 이미지 데이터를 성공적으로 가져오려면 모든 저장소가 필요합니다.

절차

1. 지정된 여러 저장소로 이미지 데이터를 가져옵니다.

```
$ glance image-create-via-import \
--container-format bare \
--name IMAGE-NAME \
--import-method web-download \
--uri URI \
--stores STORE1,STORE2
```

- *IMAGE-NAME* 을 가져올 이미지의 이름으로 교체합니다.
- *URI* 를 이미지 URI로 바꿉니다.
- *STORE1,STORE2, STORE3* 을 이미지 데이터를 복사할 저장소 이름으로 교체합니다.
- 또는 모든 저장소에 이미지를 업로드하려면 **--stores** 를 **--all-stores true** 로 바꿉니다.

참고

--allow-failure 매개변수를 **false** 로 설정하면 이미지 서비스에서 이미지 데이터를 가져오지 못하는 저장소를 무시하지 않습니다. 이미지 속성 **os_glance_failed_import** 를 사용하여 실패한 저장소 목록을 볼 수 있습니다. 자세한 내용은 이미지 가져오기 작업의 진행률 검사를 참조하십시오.

2. 이미지 데이터가 특정 저장소에 추가되었는지 확인합니다.

```
$ glance image-show IMAGE-ID | grep stores
```

IMAGE-ID 를 기존 이미지의 ID로 바꿉니다.

출력에는 쉼표로 구분된 저장소 목록이 표시됩니다.

4.3.4. 단일 저장소로 이미지 데이터 가져오기

이미지 데이터를 단일 저장소로 가져올 수 있습니다.

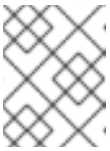
절차

1. 이미지 데이터를 단일 저장소로 가져옵니다.

```
$ glance image-create-via-import \
--container-format bare \
--name IMAGE-NAME \
--import-method web-download \
--uri URI \
--store STORE
```

- *IMAGE-NAME* 을 가져올 이미지의 이름으로 교체합니다.
- *URI* 를 이미지 URI로 바꿉니다.

- *STORE* 를 이미지 데이터를 복사할 저장소 이름으로 교체합니다.



참고

명령에 **--stores,--all-stores** 또는 **--store** 옵션을 포함하지 않으면 이미지 서비스에서 중앙 저장소에 이미지를 생성합니다.

2. 이미지 데이터가 특정 저장소에 추가되었는지 확인합니다.

```
$ glance image-show IMAGE-ID | grep stores
```

IMAGE-ID 를 기존 이미지의 ID로 바꿉니다.

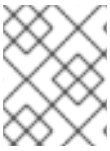
출력에는 쉼표로 구분된 저장소 목록이 표시됩니다.

4.3.5. 이미지 가져오기 작업의 진행률 확인

상호 운용 가능한 이미지 가져오기 워크플로우는 이미지 데이터를 저장소로 순차적으로 가져옵니다. 이미지 크기, 저장소 수, 중앙 사이트와 에지 사이트 간의 네트워크 속도는 이미지 가져오기 작업을 완료하는 데 걸리는 시간에 영향을 미칩니다.

이미지 가져오기 작업 중에 전송된 알림에 표시되는 두 개의 이미지 속성을 확인하여 이미지 가져오기 진행 상황을 확인할 수 있습니다.

- **os_glance_importing_to_stores** 속성에는 이미지 데이터를 가져오지 않은 저장소가 나열됩니다. 가져오기 시작 시 요청된 모든 저장소가 목록에 표시됩니다. 저장소가 이미지 데이터를 가져올 때마다 이미지 서비스는 목록에서 저장소를 제거합니다.
- **os_glance_failed_import** 속성은 이미지 데이터를 가져오지 못하는 저장소를 나열합니다. 이 목록은 이미지 가져오기 작업 시작 시 비어 있습니다.



참고

다음 절차에서 환경에는 3개의 Ceph Storage 클러스터(**중앙** 저장소와 **dcn0** 및 **dcn1** 에 있는 두 개의 저장소)가 있습니다.

절차

1. 이미지 데이터가 특정 저장소에 추가되었는지 확인합니다.

```
$ glance image-show IMAGE-ID
```

IMAGE-ID 를 기존 이미지의 ID로 바꿉니다.

출력에는 다음 예제 스니펫과 유사한 쉼표로 구분된 저장소 목록이 표시됩니다.

```
| os_glance_failed_import      |
| os_glance_importing_to_stores | central,dcn0,dcn1
| status                       | importing
```

2. 이미지 가져오기 작업의 상태를 모니터링합니다. **watch** 를 사용하여 명령 앞에 도달하면 명령 출력이 2초마다 새로 고쳐집니다.

```
$ watch glance image-show IMAGE-ID
```

IMAGE-ID 를 기존 이미지의 ID로 바꿉니다.

이미지 가져오기 작업이 진행됨에 따라 작업 상태가 변경됩니다.

```
| os_glance_failed_import      |
| os_glance_importing_to_stores | dcn0,dcn1
| status                       | importing
```

이미지를 가져오지 못했습니다. 이미지가 다음 예와 유사합니다.

```
| os_glance_failed_import      | dcn0
| os_glance_importing_to_stores | dcn1
| status                       | importing
```

작업이 완료되면 상태가 active로 변경됩니다.

```
| os_glance_failed_import      | dcn0
| os_glance_importing_to_stores |
| status                       | active
```

4.4. 기존 이미지를 여러 저장소에 복사

이 기능을 사용하면 상호 운용 가능한 이미지 가져오기 워크플로를 사용하여 에지의 여러 Ceph Storage 저장소로 Red Hat OpenStack Image 서비스(glance) 이미지 데이터를 사용하여 기존 이미지를 복사할 수 있습니다.

참고

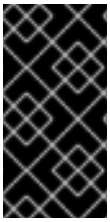
이미지를 에지 사이트에 복사하기 전에 중앙 사이트에 있어야 합니다. 이미지 소유자 또는 관리자만 새로 추가된 저장소에 기존 이미지를 복사할 수 있습니다.

이미지 데이터를 수신하도록 **--all-stores** 를 **true** 로 설정하거나 특정 저장소를 지정하여 기존 이미지 데이터를 복사할 수 있습니다.

- **--all-stores** 옵션의 기본 설정은 **false** 입니다. **--all-stores** 가 **false** 인 경우 **--stores STORE1,STORE2** 를 사용하여 이미지 데이터를 수신하는 저장소를 지정해야 합니다. 지정된 저장소에 이미지 데이터가 이미 있으면 요청이 실패합니다.
- **all-stores** 를 **true** 로 설정하고 이미지 데이터가 일부 저장소에 이미 있는 경우 해당 저장소가 목록에서 제외됩니다.

이미지 데이터를 수신하는 저장소를 지정하면 이미지 서비스는 중앙 사이트의 데이터를 스테이징 영역으로 복사합니다. 그런 다음 이미지 서비스는 상호 운용 가능한 이미지 가져오기 워크플로를 사용하여 이미지 데이터를 가져옵니다. 자세한 내용은 여러 저장소에 이미지 가져오기를 참조하십시오.

이미지 관리에 이미지 서비스 명령줄 클라이언트를 사용합니다.



중요

Red Hat은 관리자가 정해진 이미지 복사 요청을 신중하게 방지하는 것이 좋습니다. 동일한 이미지에 대한 두 개의 시간이 지정된 복사 이미지 작업으로 인해 경쟁 조건과 예기치 않은 결과가 발생합니다. 기존 이미지 데이터는 그대로 유지되지만 새 저장소에 데이터를 복사하지 못합니다.

4.4.1. 모든 저장소에 이미지 복사

다음 절차에 따라 사용 가능한 모든 저장소에 이미지 데이터를 복사합니다.

절차

1. 사용 가능한 모든 저장소에 이미지 데이터를 복사합니다.

```
$ glance image-import IMAGE-ID \
--all-stores true \
--import-method copy-image
```

   IMAGE-ID 를 복사할 이미지 이름으로 교체합니다.

2. 이미지 데이터가 사용 가능한 모든 저장소에 성공적으로 복제되었는지 확인합니다.

```
$ glance image-list --include-stores
```

   이미지 가져오기 작업의 상태를 확인하는 방법에 대한 자세한 내용은 이미지 가져오기 작업 의 진행 상황 확인을 참조하십시오.

4.4.2. 특정 저장소에 이미지 복사

다음 절차에 따라 이미지 데이터를 특정 저장소에 복사합니다.

절차

1. 이미지 데이터를 특정 저장소에 복사합니다.

```
$ glance image-import IMAGE-ID \
--stores STORE1,STORE2 \
--import-method copy-image
```

   - *IMAGE-ID* 를 복사할 이미지 이름으로 교체합니다.
   - *STORE1* 및 *STORE2* 를 이미지 데이터를 복사할 저장소 이름으로 교체합니다.

2. 이미지 데이터가 지정된 저장소에 성공적으로 복제되었는지 확인합니다.

```
$ glance image-list --include-stores
```

   이미지 가져오기 작업의 상태를 확인하는 방법에 대한 자세한 내용은 이미지 가져오기 작업 의 진행 상황 확인을 참조하십시오.

4.5. 특정 저장소에서 이미지 삭제

Red Hat OpenStack Image 서비스(glance)를 사용하여 특정 저장소의 기존 이미지 사본을 삭제합니다.

이미지 관리에 이미지 서비스 명령줄 클라이언트를 사용합니다.

프로세스

특정 저장소에서 이미지를 삭제합니다.

```
$ glance stores-delete --store _STORE_ID_ _IMAGE_ID_
```

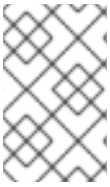
- _STORE_ID를 이미지 복사본을 삭제해야 하는 저장소 이름으로 바꿉니다.
- *IMAGE_ID* 를 삭제하려는 이미지의 ID로 바꿉니다.

주의

glance image-delete 를 사용하면 모든 사이트의 이미지가 영구적으로 삭제됩니다. 이미지 인스턴스 및 메타데이터뿐만 아니라 모든 이미지 사본이 삭제됩니다.

4.6. 이미지 위치 이해

이미지가 여러 사이트에 존재할 수 있지만 지정된 이미지에 대한 단일 UUID(Universal Unique Identifier)만 있습니다. 이미지 메타데이터에는 각 복사본의 위치가 포함되어 있습니다. 예를 들어 두 에지 사이트에 있는 이미지는 중앙 사이트와 두 개의 에지 사이트 등 3개의 위치가 있는 단일 UUID로 노출됩니다.



참고

이미지 관리를 위해 OpenStack 명령줄 클라이언트 대신 Image 서비스(glance) 명령줄 클라이언트를 사용합니다. 그러나 **openstack image show** 명령을 사용하여 이미지 위치 속성을 나열합니다. **glance image-show** 명령 출력에는 위치가 포함되어 있지 않습니다.

프로세스

1. 이미지 사본이 존재하는 사이트를 표시합니다.

```
$ glance image-show ID | grep "stores"

| stores |  default_backend,dcn1,dcn2
```

이 예제에서는 중앙 사이트인 **default_backend** 및 두 에지 사이트 **dcn1** 및 **dcn2** 에 이미지가 있습니다.

2. 또는 **--include-stores** 옵션과 함께 **glance image-list** 명령을 실행하여 이미지가 존재하는 사이트를 확인할 수 있습니다.

```
$ glance image-list --include-stores

| ID                                   | Name    | Stores

| 2bd882e7-1da0-4078-97fe-f1bb81f61b00 | cirros | default_backend,dcn1,dcn2
```

3. 각 위치의 세부 정보를 표시하려면 이미지 위치 속성을 나열합니다.

```
$ openstack image show ID -c properties

| properties |

(--- cut ---)
locations='[{'url': 'rbd://79b70c32-df46-4741-93c0-8118ae2ae284/images/2bd882e7-1da0-
4078-97fe-f1bb81f61b00/snap', 'metadata': {'store': 'default_backend'}}, {'url': 'rbd://63df2767-
8ddb-4e06-8186-8c155334f487/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap',
'metadata': {'store': 'dcn1'}}, {'url': 'rbd://1b324138-2ef9-4ef9-bd9e-
aa7e6d6ead78/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap', 'metadata': {'store':
'dcn2'}}]',
(--- cut --)
```

이미지 속성에는 각 이미지의 위치에 대한 다양한 Ceph RBD URI가 표시됩니다.

이 예에서 중앙 이미지 위치 URI는 다음과 같습니다.

```
rbd://79b70c32-df46-4741-93c0-8118ae2ae284/images/2bd882e7-1da0-4078-97fe-
f1bb81f61b00/snap', 'metadata': {'store': 'default_backend'}}
```

URI는 다음 데이터로 구성됩니다.

- **79b70c32-df46-4741-93c0-8118ae284** 는 중앙 Ceph FSID에 해당합니다. 각 Ceph 클러스터에는 고유한 FSID가 있습니다.
- 모든 사이트의 기본값은 **이미지가** 저장되는 Ceph 풀에 해당하는 images 입니다.
- **2bd882e7-1da0-4078-97fe-f1bb81f61b00** 은 이미지 UUID에 해당합니다. UUID는 위치에 관계없이 지정된 이미지에 대해 동일합니다.
- 메타데이터는 이 위치가 매핑되는 Glance 저장소를 보여줍니다. 이 예제에서는 중앙 허브 사이트인 **default_backend** 에 매핑됩니다.

부록 A. IMAGE 서비스(GLANCE) 명령 옵션

glance image-create 및 **glance image-update** 명령과 함께 다음 선택적 인수를 사용할 수 있습니다.

표 A.1. 명령 옵션

특정 대상	옵션	설명
All	**--architecture <ARCHITECTURE>**	https://docs.openstack.org/glance/latest/user/common-image-properties.html#architecture에 지정된 운영 체제 아키텍처
All	**--protected [True_False]**	true인 경우 이미지를 해제할 수 없습니다.
All	**--name <NAME>**	이미지에 대한 설명이 포함된 이름
All	**--instance-uuid <INSTANCE_UUID>**	이 이미지가 연결된 인스턴스를 기록하는 데 사용할 수 있는 메타데이터입니다. (정보만 있으면 인스턴스 스냅샷을 생성하지 않습니다.)
All	**--min-disk <MIN_DISK>**	이미지를 부팅하는 데 필요한 디스크 공간(GB)의 양입니다.
All	**--visibility <VISIBILITY>**	이미지 접근성 범위. 유효한 값: public, private, community, shared
All	**--kernel-id <KERNEL_ID>**	AMI 스타일 이미지를 부팅할 때 커널로 사용해야 하는 Image 서비스(glance)에 저장된 이미지 ID입니다.
All	**--os-version <OS_VERSION>**	배포자가 지정한 운영 체제 버전
All	**--disk-format <DISK_FORMAT>**	디스크 형식입니다. 유효한 값: None, ami, ari, aki, vhd, vhdx, vmdk, raw, qcow2, vdi, iso, ploop
All	**--os-distro <OS_DISTRO>**	https://docs.openstack.org/glance/latest/user/common-image-properties.html#os-distro에 지정된 대로 운영 체제 배포의 일반적인 이름
All	**--owner <OWNER>**	이미지의 소유자
All	**--ramdisk-id <RAMDISK_ID>**	AMI 스타일 이미지를 부팅할 때 램디스크로 사용해야 하는 이미지 서비스에 저장된 이미지 ID입니다.
All	**--min-ram <MIN_RAM>**	이미지를 부팅하는 데 필요한 RAM 용량(MB)입니다.
All	**--container-format <CONTAINER_FORMAT>**	컨테이너 형식. 유효한 값: None, ami, ari, aki, bare, ovf, ova, docker
All	**--property <key=value>**	이미지와 연결할 임의의 속성입니다. 여러 번 사용할 수 있습니다.

특정 대상	옵션	설명
glance image-create	**--Tags <TAGS> [<TAGS> ...]**	이미지와 관련된 문자열 목록
glance image-create	**--id <ID>**	이미지의 식별자
Glance image-update	**--remove-property**	이미지에서 제거할 임의의 속성의 키 이름입니다.

부록 B. 이미지 구성 매개변수

glance image-create 및 **glance image-update** 명령 모두에 **property** 옵션과 함께 다음 키를 사용할 수 있습니다.

표 B.1. 속성 키

특정 대상	키	설명	지원되는 값
All	**아키텍처**	하이퍼바이저에서 지원해야 하는 CPU 아키텍처입니다. 예를 들면 **x86_64**,**arm** 또는 **ppc64** 입니다. **uname -m** 을 실행하여 머신 아키텍처를 가져옵니다.	• **Aarch** - ARM 64비트 • **Alpha** - DEC 64비트 RISC • **armv7l** - ARM Cortex-A7 MPCore • **CRIS**- 이더넷, 토큰 링, AXis-Code Reduced Instruction Set • **i686** - Intel 6세대 x86 (P6 마이크로 아키텍처) • **ia64** - Itanium • **lm32** - Lattice Micro32 • **m68k** - engineola 68000 • **microblaze** - Xilinx 32bit Cryostat (Big Endian) • **microblazeel** - Xilinx 32bit Cryostat (Little Endian) • **ChePS** -MIPS 32비트 RISC (Big Endian) • **mipsel** -MIPS 32비트 RISC(Little Endian) • **mips64** -MIPS 64비트 RISC(Big Endian) • **mips64el** -MIPS 64비트 RISC(Little Endian) • **openrisc** - OpenCores RISC • **parisc** - HP Precision Architecture RISC • **parisc64** - HP Precision Architecture 64비트 RISC • **ppc** - PowerPC 32-bit • **ppc64** - PowerPC 64-bit • **ppcemb** - PowerPC (32비트 포함)

특정 대상	키	설명	지원되는 값
			• **s390** - IBM Enterprise Systems Architecture/390 • **s390x** - S/390 64비트 • **sh4** - SuperH SH-4 (Little Endian) • **sh4eb** - SuperH SH-4 (Big Endian) • **Cryostat** - Scalable Processor Architecture, 32-bit • **sparc64** - 확장 가능한 프로세서 아키텍처, 64비트 • **Unicore32** - 마이크로 프로세서 연구 및 개발 센터 RISC Unicore32 • **x86_64** - 64비트 확장 IA-32 • **xtensa** - Tensilica Xtensa 구성 가능한 마이크로프로세서 코어 • **xtensaeb** - Tensilica Xtensa 구성 가능한 마이크로프로세서 코어 (Big Endian)
All	**hypervisor_type**	하이퍼바이저 유형입니다.	**kvm**, **vmware**
All	**instance_uuid**	스냅샷 이미지의 경우 이 이미지를 생성하는 데 사용되는 서버의 UUID입니다.	유효한 서버 UUID
All	**kernel_id**	AMI 스타일 이미지를 부팅할 때 커널로 사용해야 하는 이미지 서비스에 저장된 이미지의 ID입니다.	유효한 이미지 ID
All	**os_distro**	소문자로 운영 체제 배포의 일반적인 이름입니다.	• **Arch** - Arch Linux. **archlinux** 또는 **org.archlinux** 를 사용하지 마십시오. • **CentOS** - 커뮤니티 엔터프라이즈 운영 체제. **org.centos** 또는 **CentOS** 를 사용하지 마십시오. • **Debian** - Debian. **Debian** 또는 **org.debian** 을 사용하지 마십시오. • **Fedora** - Fedora.

특정 대상	키	설명	지원되는 값
			Fedora,**org.fedora** 또는 **org.fedoraproject** 를 사용하지 마십시오. • **Cryostat** - Cryostat. **org.freebsd**,**freeBSD** 또는 Cryostat를 사용하지 **마십시오**. • **Gentoo** - Linux. **Gentoo** 또는 **org.gentoo** 를 사용하지 마십시오. • Mandrake **-** Mandrakelinux (MandrakeSoft) 배포. **mandrakelinux** 또는 **MandrakeLinux** 를 사용하지 마십시오. • Mandri **va** - Mandriva Linux. **mandrivalinux** 를 사용하지 마십시오. • **MES** - Mandriva Enterprise Server. **mandrivaent** 또는 **mandrivaES** 를 사용하지 마십시오. • **MSDOS -** Microsoft 디스크 운영 체제. **ms-dos** 를 사용하지 마십시오. • **netbsd** - NetBSD. **NetBSD** 또는 **org.netbsd** 를 사용하지 마십시오. • **NetWare -** Novell NetWare. 새로운 **NetWare** 또는 **NetWare** 를 사용하지 마십시오. • Open **BSD** - OpenBSD. **OpenBSD** 또는 **org.openbsd** 를 사용하지 마십시오. • OpenSolaris **-** OpenSolaris. **OpenSolaris** 또는 **org.opensolaris** 를 사용하지 마십시오. • **o** penSUSE - openSUSE. **suse**,**SuSE** 또는 **org.opensuse** 를 사용하지 마십시오. • **RHEL** - Red Hat Enterprise Linux. **redhat**,**RedHat** 또는 **com.redhat** 을 사용하지 마십시오. • **슬림** - SUSE Linux Enterprise Desktop. **com.suse** 를 사용하지 마십시오.

특정 대상	키	설명	지원되는 값
			• **Ubuntu** - Ubuntu. **ubuntu**,**com.ubuntu**,**org.ubuntu** 또는 **canonical** 을 사용하지 마십시오. • **Windows** - Microsoft Windows. **com.microsoft.server** 를 사용하지 마십시오.
All	**os_version**	배포자가 지정한 운영 체제 버전입니다.	버전 번호(예: "11.10")
All	**ramdisk_id**	AMI 스타일 이미지를 부팅할 때 램디스크로 사용해야 하는 이미지 서비스에 저장된 이미지의 ID입니다.	유효한 이미지 ID
All	**vm_mode**	가상 머신 모드입니다. 가상 머신에 사용되는 호스트/게스트 ABI(애플리케이션 바이너리 인터페이스)를 나타냅니다.	H **VM**- 완전 가상화. 이는 QEMU 및 KVM에서 사용하는 모드입니다.
libvirt API 드라이버	**hw_cdrom_bus**	CD-ROM 장치를 연결할 디스크 컨트롤러의 유형을 지정합니다.	**SCSI**,**virtio**,**ide** 또는 **usb**. **iscsi** 를 지정하는 경우 **hw_scsi_model** 매개변수를 **virtio-scsi** 로 설정해야 합니다.
libvirt API 드라이버	**hw_disk_bus**	디스크 장치를 연결할 디스크 컨트롤러의 유형을 지정합니다.	**SCSI**,**virtio**,**ide** 또는 **usb**. **iscsi** 를 사용하는 경우 **hw_scsi_model** 을 **virtio-scsi** 로 설정해야 합니다.
libvirt API 드라이버	**hw_firmware_type**	인스턴스를 부팅하는 데 사용할 펌웨어 유형을 지정합니다.	다음 유효한 값 중 하나로 설정합니다. • **BIOS** • **uefi**
libvirt API 드라이버	**hw_machine_type**	지정된 시스템 유형을 사용하여 ARM 시스템을 부팅할 수 있습니다. ARM 이미지가 사용되고 해당 시스템 유형이 명시적으로 지정되지 않은 경우 Compute는 virt 시스템 유형을 ARMv7 및 AArch64의 기본값으로 사용합니다.	유효한 유형은 **virsh capabilities** 명령을 사용하여 볼 수 있습니다. 시스템 유형이 시스템 태그에 표시됩니다.
libvirt API 드라이버	**hw_numa_nodes**	인스턴스에 노출할 NUMA 노드 수입니다(플레이버 정의를 재정의하지 않음).	정수.

특정 대상	키	설명	지원되는 값
libvirt API 드라이버	**hw_numa_cpus.0**	vCPUs N-M을 NUMA 노드 0에 매핑(플레이버 정의를 재정의하지 않음).	쉼표로 구분된 정수 목록입니다.
libvirt API 드라이버	**hw_numa_cpus.1**	vCPUs N-M을 NUMA 노드 1에 매핑(플레이버 정의를 재정의하지 않음).	쉼표로 구분된 정수 목록입니다.
libvirt API 드라이버	**hw_numa_mem.0**	NMB의 RAM을 NUMA 노드 0에 매핑합니다(플레이버 정의를 재정의하지 않음).	정수
libvirt API 드라이버	**hw_numa_mem.1**	NMB의 RAM을 NUMA 노드 1에 매핑합니다(플레이버 정의를 재정의하지 않음).	정수
libvirt API 드라이버	**hw_pci_numa_affinity_policy**	PCI 패스스루 장치 및 SR-IOV 인터페이스에 대한 NUMA 선호도 정책을 지정합니다.	다음 유효한 값 중 하나로 설정합니다. ● **필수**: Compute 서비스는 인스턴스의 NUMA 노드 중 하나 이상이 PCI 장치와 선호도가 있는 경우에만 PCI 장치를 요청하는 인스턴스를 생성합니다. 이 옵션은 최상의 성능을 제공합니다. ● **preferred**: Compute 서비스는 NUMA 선호도에 따라 최상의 PCI 장치를 선택합니다. 유사성이 불가능한 경우 Compute 서비스는 PCI 장치와 유사성이 없는 NUMA 노드에 인스턴스를 예약합니다. ● **legacy**: (기본값) Compute 서비스는 다음 경우 중 하나에서 PCI 장치를 요청하는 인스턴스를 생성합니다. o PCI 장치는 NUMA 노드 중 하나 이상과 선호도를 갖습니다. o PCI 장치는 NUMA 기능에 대한 정보를 제공하지 않습니다.
libvirt API 드라이버	**hw_qemu_guest_agent**	게스트 에이전트 지원. **yes** 로 설정하고 **qemu-ga** 도 설치된 경우 파일 시스템을 정지(fr#159en)하고 스냅샷이 자동으로 생성될 수 있습니다.	**제공됨 / 없음**

특정 대상	키	설명	지원되는 값
libvirt API 드라이버	**hw_rng_model**	이 이미지로 시작된 인스턴스에 난수 생성기(RNG) 장치를 추가합니다. 인스턴스 플레이버는 기본적으로 RNG 장치를 활성화합니다. RNG 장치를 비활성화하려면 클라우드 관리자가 플레이버에서 **hw_rng:allowed** 를 **False** 로 설정해야 합니다. 기본 엔트로피 소스는 **/dev/random** 입니다. 하드웨어 RNG 장치를 지정하려면 Compute 환경 파일에서 **rng_dev_path** 를 **/dev/hwrng** 로 설정합니다.	**virtio** 또는 기타 지원되는 장치.
libvirt API 드라이버	**hw_scsi_model**	VirtIO SCSI(virtio-scsi)를 사용하여 컴퓨팅 인스턴스에 대한 블록 장치 액세스를 제공할 수 있습니다. 기본적으로 인스턴스는 VirtIO Block(virtio-blk)을 사용합니다. virtio SCSI는 향상된 확장성 및 성능을 제공하고 고급 SCSI 하드웨어를 지원하는 반가상화 SCSI 컨트롤러 장치입니다.	**virtio-scsi**
libvirt API 드라이버	**hw_tpm_model**	사용할 TPM 장치의 모델로 설정합니다. **hw:tpm_version** 이 구성되지 않은 경우 무시됩니다.	• TPM **-tis**: (기본값) TPM 인터페이스 사양. • **TPM-crb**: 명령 응답 버퍼. TPM 버전 2.0과만 호환됩니다.
libvirt API 드라이버	**hw_tpm_version**	사용할 TPM 버전으로 설정합니다. TPM 버전 **2.0** 은 지원되는 유일한 버전입니다.	**2.0**

특정 대상	키	설명	지원되는 값
libvirt API 드라이버	**hw_video_model**	가상 머신 인스턴스에서 사용할 표시 장치의 비디오 장치 드라이버입니다.	다음 값 중 하나로 설정하여 사용할 지원되는 드라이버를 지정합니다. ● **virtio** - (기본값) 대부분의 아키텍처에서 지원하는 가상 머신 표시 장치의 **권장** 드라이버입니다. VirtIO GPU 드라이버는 RHEL-7 이상 및 Linux 커널 버전 4.4 이상에 포함되어 있습니다. 인스턴스 커널에 VirtIO GPU 드라이버가 있는 경우 인스턴스에서 모든 VirtIO GPU 기능을 사용할 수 있습니다. 인스턴스 커널에 VirtIO GPU 드라이버가 없는 경우 VirtIO GPU 장치는 정상적으로 VGA 호환성 모드로 전환되어 인스턴스에 대한 작업 표시를 제공합니다. ● **QXL** - **더 이상 유지 관리되지 않는 Spice 또는 noVNC 환경의 경우 더 이상 사용되지 않는** 드라이버입니다. ● **Cirrus -** 이전 버전과의 호환성을 위해서만 지원됩니다. 새 인스턴스에는 사용하지 마십시오. ● **VGA -** IBM Power 환경에 이 드라이버를 사용합니다. ● **GOP** - QEMU/KVM 환경에서는 지원되지 않습니다. ● **Cryostat** - KVM 환경에서 지원되지 않습니다. ● **vmvga** - 레거시 드라이버를 사용하지 마십시오. ● **none** - 이 값을 사용하여 드라이버가 별도로 구성된 가상 GPU(vGPU) 인스턴스에서 에뮬레이션된 그래픽 또는 그래픽을 비활성화합니다.
libvirt API 드라이버	**hw_video_ram**	비디오 이미지의 최대 RAM입니다. **hw_video:ram_max_mb** 값이 플레이버의 **extra_specs** 에 설정된 경우에만 사용되며 해당 값은 **hw_video_ram** 에 설정된 값보다 높습니다.	정수(MB)(예: *64*)

특정 대상	키	설명	지원되는 값
libvirt API 드라이버	**hw_watchdog_action**	서버가 중단된 경우 지정된 작업을 수행하는 가상 하드웨어 워치독 장치를 활성화합니다. 워치독은 i6300esb 장치(PCI Intel 6300 Cryostat를 추정)를 사용합니다. **hw_watchdog_action** 을 지정하지 않으면 워치독이 비활성화됩니다.	• disabled-장치가 연결되어 있지 않습니다. 이미지의 플레이버를 사용하여 활성화된 경우에도 사용자가 이미지에 대한 워치독을 비활성화할 수 있습니다. 이 매개변수의 기본값은 disabled입니다. • reset-forcefully reset the guest를 설정합니다. • poweroff-Forcefully power off 게스트의 전원을 끕니다. • 게스트를 일시 중지합니다. • none-Only는 워치독을 활성화합니다. 서버가 중단된 경우 아무 작업도 수행하지 않습니다.
libvirt API 드라이버	**os_command_line**	**libvirt** 드라이버가 기본값 대신 사용할 커널 명령줄입니다. Linux Containers(LXC)의 경우 값은 초기화를 위한 인수로 사용됩니다. 이 키는 Amazon 커널, 램디스크 또는 머신 이미지(aki, ari 또는 ami)에만 유효합니다.	

특정 대상	키	설명	지원되는 값
libvirt API 드라이버	**os_secure_boot**	을 사용하여 UEFI Secure Boot로 보호되는 인스턴스를 생성합니다.	다음 유효한 값 중 하나로 설정합니다. • **필수**: 이 이미지로 시작된 인스턴스에 대해 Secure Boot를 활성화합니다. 인스턴스는 Compute 서비스가 Secure Boot를 지원할 수 있는 호스트를 찾는 경우에만 시작됩니다. 호스트를 찾을 수 없는 경우 Compute 서비스에서 "No valid host" 오류를 반환합니다. • **disabled**: 이 이미지로 시작된 인스턴스의 Secure Boot를 비활성화합니다. 기본적으로 비활성되어 있습니다. • **선택 사항**: Compute 서비스에서 호스트가 Secure Boot를 지원할 수 있다고 결정하는 경우에만 이 이미지로 시작된 인스턴스에 대해 Secure Boot를 활성화합니다.
libvirt API 드라이버 및 VMware API 드라이버	**hw_vif_model**	사용할 가상 네트워크 인터페이스 장치의 모델을 지정합니다.	유효한 옵션은 구성된 하이퍼바이저에 따라 다릅니다. • KVM 및 QEMU: e1000, ne2k_pci, pcnet, rtl8139 및 virtio. • VMware: e1000, e1000e, VirtualE1000e, VirtualPCNet32, VirtualSriovEthernetCard 및 VirtualVmxnet. • Cryostat: e1000, netfront, ne2k_pci, pcnet 및 rtl8139.
VMware API 드라이버	**vmware_adaptertype**	하이퍼바이저에서 사용하는 가상 SCSI 또는 IDE 컨트롤러입니다.	**lsiLogic**, **busLogic**, or **ide**
VMware API 드라이버	**vmware_ostype**	이미지에 설치된 운영 체제를 설명하는 VMware GuestID입니다. 이 값은 가상 머신을 생성할 때 하이퍼바이저에 전달됩니다. 지정하지 않으면 기본값은 **otherGuest** 입니다.	자세한 내용은 VMware vSphere의 이미지를 참조하십시오.

특정 대상	키	설명	지원되는 값
VMware API 드라이버	**vmware_image_version**	현재 사용되지 않습니다.	**1**
CryostatAPI 드라이버	**auto_disk_config**	true인 경우 인스턴스가 부팅되기 전에 디스크의 루트 파티션의 크기를 자동으로 조정합니다. 이 값은 CryostatAPI 드라이버와 함께 Cryostat 기반 하이퍼바이저를 사용할 때 Compute 서비스에서만 고려합니다. Compute 서비스는 이미지에 단일 파티션이 있고 파티션이 **ext3** 또는 **ext4** 형식인 경우에만 크기 조정을 시도합니다.	**true** / **false**
libvirt API 드라이버 및 CryostatAPI 드라이버	**os_type**	이미지에 설치된 운영 체제입니다. CryostatAPI 드라이버에는 이미지의 **os_type** 매개변수 값에 따라 다양한 작업을 수행하는 논리가 포함되어 있습니다. 예를 들어 **os_type=windows** 이미지의 경우 Linux 스왑 파티션 대신 FAT32 기반 스왑 파티션을 생성하고 삽입된 호스트 이름을 16자 미만으로 제한합니다.	**Linux** 또는 **Windows**