Red Hat OpenStack Platform 17.1

イメージの作成および管理

Red Hat OpenStack Platform で Image サービス (glance) を使用してイメージを作成および管理する

Last Updated: 2024-08-16

Red Hat OpenStack Platform 17.1 イメージの作成および管理

Red Hat OpenStack Platform で Image サービス (glance) を使用してイメージを作成および管理する

OpenStack Team
rhos-docs@redhat.com

法律上の通知

概要

このガイドでは、イメージを作成および管理する手順、ならびに Image サービス (glance) を設定する手順について説明します。

目次

多様性を受け入れるオープンソースの強化

Red Hat では、コード、ドキュメント、Web プロパティーにおける配慮に欠ける用語の置き換えに取り組んでいます。まずは、マスター (master)、スレーブ (slave)、ブラックリスト (blacklist)、ホワイトリスト (whitelist) の 4 つの用語の置き換えから始めます。この取り組みは膨大な作業を要するため、用語の置き換えは、今後の複数のリリースにわたって段階的に実施されます。詳細は、Red Hat CTO である Chris Wright のメッセージ をご覧ください。

RED HAT ドキュメントへのフィードバック (英語のみ)

Red Hat ドキュメントに対するご意見をお聞かせください。ドキュメントの改善点があればお知らせください。

Jira でドキュメントのフィードバックを提供する

ドキュメントに関するフィードバックを提供するには、Create Issue フォームを使用します。Red Hat OpenStack Platform Jira プロジェクトで Jira Issue が作成され、フィードバックの進行状況を追跡できます。

1. Jira にログインしていることを確認してください。Jira アカウントをお持ちでない場合は、アカウントを作成してフィードバックを送信してください。
2. Create Issue をクリックして、**Create Issue** ページを開きます。
3. **Summary** フィールドと **Description** フィールドに入力します。**Description** フィールドに、ドキュメントの URL、章またはセクション番号、および問題の詳しい説明を入力します。フォーム内の他のフィールドは変更しないでください。
4. **Create** をクリックします。

第1章 IMAGE サービス (GLANCE)

Image サービス (glance) は、ディスクおよびサーバーイメージの検出、登録、および配信のサービスを提供します。サーバーイメージのコピーやスナップショットを作成して直ちに保管する機能を提供します。保管したイメージをテンプレートとして使用し、新規サーバーを迅速に稼働させることができます。これはサーバーのオペレーティングシステムをインストールしてサービスを個別に設定するよりも一貫性の高い方法です。

1.1. 仮想マシンイメージの形式

仮想マシン (VM) イメージは、起動可能な OS がインストールされた仮想ディスクを含むファイルです。Red Hat OpenStack Platform (RHOSP) は、さまざまな形式の仮想マシンイメージをサポートします。

仮想マシンイメージのディスク形式は、基になっているディスクイメージの形式です。コンテナー形式は、仮想マシンに関するメタデータも含んでいるファイル形式の仮想マシンイメージかどうかを示します。

Image サービス (glance) にイメージを追加するときに、**glance image-create**、**glance image-create-via-import**、および **glance image-update** コマンドで **--disk-format** および **--container-format** コマンドオプションを使用すると、イメージのディスク形式またはコンテナー形式を次の表のいずれかの値に設定できます。仮想マシンイメージのコンテナー形式が不明な場合は、**bare** に設定できます。

表1.1 ディスクイメージ形式

形式	説明
aki	Image サービスに保存されている Amazon カーネルイメージを示します。
ami	Image サービスに保存されている Amazon マシンイメージを示します。
ari	Image サービスに保存されている Amazon RAM ディスクイメージを示します。
iso	ディスク上のデータをセクター単位でコピーし、バイナリーファイルに格納した形式。通常、ISO ファイルは仮想マシンイメージの形式とはみなされませんが、ISO ファイルにはオペレーティングシステムがインストールされた起動可能なファイルシステムが含まれており、ISO ファイルは他の仮想マシンイメージファイルと同じように使用されます。
ploop	Virtuozzo が OS コンテナーを実行するためにサポートおよび使用するディスク形式。
qcow2	QEMU エミュレーターでサポートされています。この形式には、QEMU 1.1 以降が必要な QCOW2v3 (QCOW3 と呼ばれる場合があります) が含まれます。
raw	構造化されていないディスクイメージ形式。
vdi	VirtualBox 仮想マシンモニターと QEMU エミュレーターでサポートされています。

形式	説明
vhd	仮想ハードディスク。VMware、VirtualBox などの仮想マシンモニターによって使用されます。
vhdx	仮想ハードディスク v2。VHD よりも大きな記憶容量を持つディスクイメージ形式。
vmdk	仮想マシンディスク。前回のバックアップ時からのデータ変更の増分バックアップを可能にするディスクイメージ形式。

表1.2 コンテナーイメージ形式

形式	説明
aki	Image サービスに保存されている Amazon カーネルイメージを示します。
ami	Image サービスに保存されている Amazon マシンイメージを示します。
ari	Image サービスに保存されている Amazon RAM ディスクイメージを示します。
bare	コンテナーやメタデータエンベロープがイメージに存在しないことを示します。
docker	Image サービスに保存されている Docker コンテナーのファイルシステムの TAR アーカイブを示します。
ova	Image サービスに保存されている Open Virtual Appliance (OVA) TAR アーカイブファイルを示します。このファイルは、Open Virtualization Format (OVF) コンテナーファイルに保存されます。
ovf	OVF コンテナーファイル形式。仮想アプライアンスまたは仮想マシン上で実行されるソフトウェアをパッケージ化して配布するためのオープン標準。

1.2. サポート対象の IMAGE サービスバックエンド

以下に示す Image サービス (glance) バックエンドのシナリオがサポートされます。

- Ceph を使用する場合には、RBD がデフォルトのバックエンドです。
- RBD マルチストア。
- Object Storage (swift)。Image サービスは、Object Storage のタイプとバックエンドをデフォルトとして使用します。
- Block Storage (cinder)。
- NFS

  Important

NFS はサポート対象の Image サービス用デプロイメントオプションですが、より堅牢なオプションを利用することができます。
NFS は Image サービスネイティブではありません。NFS 共有を Image サービスにマウントした場合、Image サービスは操作を管理しません。Image サービスはファイルシステムにデータを書き込みますが、バックエンドが NFS 共有であることを認識しません。

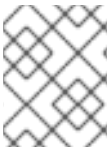
この種別のデプロイメントでは、ファイル共有に異常が発生しても、Image サービスは要求をリトライすることができません。つまり、バックエンドで障害が発生した場合、ストアは読み取り専用モードに移行するか、ローカルファイルシステムにデータの書き込みを続けます。この場合、データを損失する可能性があります。この状況から回復するには、ファイル共有がマウントされ同期されている状態にし、続いて Image サービスを再起動する必要があります。このような理由により、Red Hat では、Image サービスのバックエンドとして NFS を推奨しません。

ただし、Image サービスのバックエンドに NFS を使用することを選択した場合には、以下のベストプラクティスがリスクを軽減するのに役立ちます。

- 信頼性の高い実稼働環境グレードの NFS バックエンドを使用する。
- コントローラーノードと NFS バックエンドの間に強力で信頼性の高い接続があることを確認してください。レイヤー 2 (L2) ネットワーク接続が推奨されます。
- マウントされたファイル共有のモニタリングおよびアラート機能を追加する。
- 基になるファイルシステムのアクセス許可を設定します。書き込み権限は、ストアとして使用する共有ファイルシステムに設定する必要があります。
- glance-api プロセスが実行されるユーザーおよびグループが、ローカルファイルシステムのマウントポイントに対する書き込み権限を持たないようにしてください。これにより、プロセスはマウントの異常を検出して、書き込みを試みる際にストアを読み取り専用モードに移行することができます。

1.3. イメージの署名および検証

イメージの署名および検証により、デプロイ担当者がイメージに署名して、その署名と公開鍵の証明書をイメージの属性として保存できるようにすることで、イメージの整合性と信頼性を保護します。



注記

Nova が RADOS Block Device (RBD) を使用して仮想マシンディスクを格納している場合、イメージの署名と検証はサポートされません。

イメージの署名と検証の詳細は、**Key Manager サービスによるシークレットの管理** ガイドの Image サービス (グランス) イメージの検証 を参照してください。

1.4. イメージ形式の変換

イメージを Image サービス (glance) にインポートする場合、イメージ変換プラグインを有効にするとイメージを別の形式に変換できます。

Red Hat OpenStack Platform (RHOSP) デプロイメント設定に基づき、イメージ変換プラグインを有効化または無効化できます。デプロイ担当者は、デプロイメントで優先的に使用する形式を設定します。

内部的には、Image サービスはイメージのビットを特定の形式で受け取り、そのビットを一時的な場所

に保存します。Image サービスはプラグインをトリガーして、イメージをターゲット形式に変換し、イメージを最終的な宛先に移動します。タスクが完了すると、Image サービスは一時的な場所を削除します。Image サービスは、最初にアップロードされた形式を保持しません。

イメージのインポート時にのみ、イメージ変換をトリガーできます。イメージのアップロード時には実行されません。

イメージ管理には Image service コマンドラインクライアントを使用します。

以下に例を示します。

```
$ 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>** をイメージの名前に置き換えます。

1.5. IMAGE サービスのキャッシュ機能を使用したスケーラビリティーの向上

Image サービス (glance) キャッシュメカニズムを使用して、Image サービス API サーバーにイメージのコピーを保存し、それらを自動的に取得してスケーラビリティーを向上させます。Image サービスキャッシュを使用すると、複数のホストで glance-api を実行できます。つまり、同じイメージをバックエンドストレージから何度も取得する必要はありません。Image サービスのキャッシュ機能は、Image サービスの動作には一切影響を与えません。

Red Hat OpenStack Platform director (tripleo) heat テンプレートを使用して、Image サービスのキャッシュ機能を設定します。

手順

1. 環境ファイルの **GlanceCacheEnabled** パラメーターの値を **true** に設定します。これにより、**glance-api.conf** Heat テンプレートの **flavor** の値が自動的に **keystone+cachemanagement** に設定されます。

   ```
   parameter_defaults:
     GlanceCacheEnabled: true
   ```

2. オーバークラウドを再デプロイする際に、**openstack overcloud deploy** コマンドにその環境ファイルを追加します。

3. オプション: オーバークラウドを再デプロイする際に、**glance_cache_pruner** を異なる頻度に調節します。5 分間の頻度の例を以下に示します。

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

   ファイルシステムを使い果たす状況を回避するために、ご自分のニーズに合わせて頻度を調節します。異なる頻度を選択する場合は、以下の要素を考慮に入れます。

- 実際の環境でキャッシュするファイルのサイズ
- 利用可能なファイルシステムの容量
- 環境がイメージをキャッシュする頻度

1.6. イメージの事前キャッシュ

Red Hat OpenStack Platform (RHOSP) director を使用して、**glance-api** サービスの一部としてイメージを事前にキャッシュできます。

イメージ管理には Image service (glance) コマンドラインクライアントを使用します。

1.6.1. 定期的にイメージを事前キャッシュする際のデフォルト間隔の設定

Image サービス (glance) の事前キャッシュを行う定期ジョブは、**glance-api** サービスが実行されている各コントローラーノード上で 300 秒 (デフォルトで 5 分) ごとに実行されます。デフォルト時間を変更するには、glance-api.conf 環境ファイルの **Default** セクションで、**cache_prefetcher_interval** パラメーターを設定します。

手順

1. アンダークラウドの環境ファイルの **ExtraConfig** パラメーターを使用して、実際の要件に応じて新しい間隔を追加します。

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

   - **<300>** を、イメージを事前キャッシュする間隔 (秒数) に置き換えてください。

2. **/home/stack/templates/** の環境ファイルで間隔を修正したら、**stack** ユーザーとしてログインして設定をデプロイします。

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

   - <env_file> は、追加した **ExtraConfig** 設定が含まれる環境ファイルの名前に置き換えてください。

重要

オーバークラウドの作成時に追加の環境ファイルを渡した場合には、予定外の変更がオーバークラウドに加えられないように、ここで **-e** オプションを使用して環境ファイルを再度渡します。

関連情報

openstack overcloud deploy コマンドの詳細は、**director を使用した Red Hat OpenStack Platform のインストールと管理** ガイドの Deployment コマンド を参照してください。

1.6.2. イメージの事前キャッシュを行う定期ジョブを使用するための準備

定期的なジョブを使用してイメージを事前キャッシュするには、**glance_api** サービスを実行しているノードに直接接続された **glance-cache-manage** コマンドを使用する必要があります。サービスの要求に応答するノードを非表示にするプロキシーは使用しないでください。アンダークラウドは **glance_api** サービスを実行しているネットワークにアクセスできない可能性があるため、最初のオーバークラウドノード (デフォルトでは **controller-0** という名前です) でコマンドを実行します。

前提条件として以下の手順を実施して、正しいホストからコマンドが実行され、必要な認証情報が設定されるようにします。また、**glance-api** コンテナー内から **glance-cache-manage** コマンドが実行されるようにします。

手順

1. アンダークラウドに stack ユーザーとしてログインし、**controller-0** のプロビジョニング IP アドレスを特定します。

```
(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. オーバークラウドに対して認証するには、**/home/stack/overcloudrc** (デフォルト) に保存されている認証情報を **controller-0** にコピーします。

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

3. **controller-0** に接続します。

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

4. **controller-0** で **tripleo-admin** ユーザーとして、**glance_api サービス** の IP アドレスを特定します。以下の例では、IP アドレスは **172.25.1.105** です。

```
(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** コマンドは **glance_api** コンテナーでしか利用できないため、そのコンテナーに対して実行するスクリプトを作成します。このコンテナーには、オーバークラウドに対して認証するための環境変数がすでに設定されています。**controller-0** の **/home/tripleo-admin** に、以下の内容でスクリプト **glance_pod.sh** を作成します。

```
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. source コマンドで **overcloudrc** ファイルを読み込み、**glance_pod.sh** スクリプトを実行して、オーバークラウドのコントローラーノードに対して認証するのに必要な環境変数が設定されている **glance_api** コンテナーに対して実行します。

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

7. **glance image-list** 等のコマンドを使用して、コンテナーでオーバークラウドに対して認証されたコマンドを実行できることを確認します。

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

1.6.3. 定期的なジョブを使用したイメージの事前キャッシュ

「イメージの事前キャッシュを行う定期ジョブを使用するための準備」 で説明されている事前の手順が完了したら、定期ジョブを使用してイメージを事前キャッシュできます。

手順

1. 管理ユーザーとして、キャッシュするイメージをキューに追加します。

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

- <host_ip> を **glance-api** コンテナーが実行されているコントローラーノードの IP アドレスに置き換えます。
- <image_id> をキューに追加するイメージの ID に置き換えます。
  事前にキャッシュするイメージをキューに追加すると、**cache_images** 定期ジョブはキューに追加されたすべてのイメージを同時に事前取得します。

注記

イメージキャッシュは各ノードにローカルであるため、Red Hat OpenStack Platform (RHOSP) デプロイメントが 3、5、または 7 台のコントローラーを備えた HA の場合、**glance-cache-manage** コマンドを実行する際に **--host** オプションでホストのアドレスを指定する必要があります。

2. 以下のコマンドを実行して、イメージキャッシュ内のイメージを表示します。

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

- <host_ip> を環境内のホストの IP アドレスに置き換えてください。

1.6.4. イメージキャッシュのコマンドオプション

次の **glance-cache-manage** コマンドオプションを使用して、イメージをキャッシュ用のキューに追加し、キャッシュされたイメージを管理できます。

- **list-cached**: 現在キャッシュされているすべてのイメージをリスト表示する。
- **list-queued**: キャッシュするために現在キューに追加されているすべてのイメージをリスト表示する。
- **queue-image**: キャッシュするためにイメージをキューに追加する。
- **delete-cached-image**: キャッシュからイメージを削除する。
- **delete-all-cached-images**: キャッシュからすべてのイメージを削除する。
- **delete-queued-image**: キャッシュのキューからイメージを削除する。
- **delete-all-queued-images**: キャッシュのキューからすべてのイメージを削除する。

1.7. IMAGE サービス API を使用したスパースイメージのアップロードの有効化

Image サービス (glance) API を使用すると、スパースイメージのアップロードを使用して、ネットワークトラフィックを削減し、ストレージスペースを節約できます。この機能は、分散コンピュートノード (DCN) 環境で特に便利です。スパースイメージファイルの場合、Image サービスは null バイトシーケンスを書き込みません。Image サービスは、指定されたオフセットでデータを書き込みます。ストレージバックエンドは、これらのオフセットを、実際にはストレージスペースを消費しない null バイトとして解釈します。

イメージ管理には Image service コマンドラインクライアントを使用します。

制限事項

- スパースイメージのアップロードは、Ceph RADOS Block Device (RBD) でのみサポートされます。
- スパースイメージのアップロードは、ファイルシステムではサポートされません。
- スパース性は、クライアントと Image サービス API 間の転送中は維持されません。イメージは、Image サービス API レベルでスパース化されます。

前提条件

- Red Hat OpenStack Platform (RHOSP) デプロイメントで、Image サービスのバックエンドに RBD を使用している。

手順

1. アンダークラウドノードに **stack** ユーザーとしてログインします。
2. source コマンドで **stackrc** 認証情報ファイルを読み込みます。

```
$ source stackrc
```

3. 以下の内容で環境ファイルを作成します。

```
parameter_defaults:
  GlanceSparseUploadEnabled: true
```

4. その他の環境ファイルと共に新しい環境ファイルをスタックに追加して、オーバークラウドをデプロイします。

```
$ openstack overcloud deploy \
  --templates \
  …
  -e <existing_overcloud_environment_files> \
  -e <new_environment_file>.yaml \
  ...
```

イメージのアップロードの詳細は、Image サービスにイメージをアップロードする を参照してください。

検証

イメージをインポートしてそのサイズを確認し、スパースイメージのアップロードを検証することができます。

次の手順では、コマンド例を使用します。必要に応じて、値をご使用の環境の値に置き換えてください。

1. イメージファイルをローカルにダウンロードします。

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

- **<file_location>** をファイルの場所に置き換えます。
- **<file_name>** は、ファイルの名前に置き換えます。
  以下に例を示します。

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

2. アップロードするイメージのディスクサイズと仮想サイズを確認します。

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

以下に例を示します。

```
$ 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. イメージをインポートします。

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

4. イメージ ID を記録します。後続のステップで必要になります。

5. イメージがインポートされ、アクティブ状態にあることを確認します。

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

6. Ceph Storage ノードから、イメージのサイズが、ステップ 1 出力の仮想サイズよりも小さいことを確認します。

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

1.03906 GB
```

7. オプション：コントローラーノードの Image サービス設定ファイルで **rbd_thin_provisioning** が設定されていることを確認できます。

   a. コントローラーノードにアクセスするために SSH を使用します。

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

   b. そのコントローラーノードで **rbd_thin_provisioning** が **True** に等しいことを確認します。

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

1.8. METADEF API の保護

Red Hat OpenStack Platform (RHOSP) では、クラウド管理者はメタデータ定義 (metadef) API を使用してキー/値のペアおよびタグメタデータを定義することができます。クラウド管理者が作成できるメタデフ名前空間、オブジェクト、プロパティー、リソース、またはタグの数に制限はありません。

Image サービスのポリシーは metadef API を制御します。デフォルトでは、クラウド管理者のみがメタデフ API を作成、更新、または削除 (CUD) できます。この制限により、metadef API が権限のないユーザーに情報を公開することが防止され、悪意のあるユーザーが Image サービス (glance) データベースに無制限のリソースを埋め込み、サービス妨害 (DoS) 型の攻撃を引き起こすリスクが軽減されます。ただし、クラウド管理者はデフォルトのポリシーをオーバーライドできます。

1.9. クラウドユーザーの METADEF API アクセスの有効化

メタデータ定義 (metadef) API への書き込みアクセスに依存するユーザーを持つクラウド管理者は、デ

フォルトの管理者専用ポリシーをオーバーライドすることで、すべてのユーザーがそれらの API にアクセスできるようにできます。ただし、この種の設定では、顧客名や内部プロジェクト等の秘匿すべきリソース名が意図せず漏えいする可能性があります。すべてのユーザーに読み取りアクセスしか付与していない場合であっても、管理者はシステムを監査し、過去に作成したセキュリティー的に脆弱なリソースを識別する必要があります。

手順

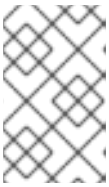
1. クラウド管理者としてアンダークラウドにログインし、ポリシーオーバーライド用のファイルを作成します。以下に例を示します。

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

2. すべてのユーザーが metadef API を読み取り/書き込みできるように、ポリシーオーバーライドファイルを設定します。

```
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' }
  }
```



注記

すべての metadef ポリシーを設定する際に、**rule:metadef_default** を使用する必要があります。ポリシーとポリシー構文の詳細は、この ポリシー の章を参照してください。

3. オーバークラウドのデプロイ時に **-e** オプションを使用して、デプロイメントコマンドに新しいポリシーファイルを追加します。

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

第2章 RHEL KVM または RHOSP 互換イメージの作成

Red Hat OpenStack Platform (RHOSP) Image サービス (glance) で管理できるイメージを作成するには、Red Hat Enterprise Linux (RHEL) Kernel-based Virtual Machine (KVM) のインスタンスイメージを使用するか、RHEL ISO ファイルまたは Windows ISO ファイルを使用して、QCOW2 形式の RHOSP 互換イメージを手動で作成できます。

2.1. RHEL KVM イメージの作成

Red Hat Enterprise Linux (RHEL) Kernel-based Virtual Machine (KVM) のインスタンスイメージを使用して、Red Hat OpenStack Platform (RHOSP) の Image サービス (glance) で管理できるイメージを作成します。

2.1.1. Red Hat OpenStack Platform で RHEL KVM インスタンスイメージを使用する

Red Hat OpenStack Platform (RHOSP) では、次に示す Red Hat Enterprise Linux (RHEL) Kernel-based Virtual Machine (KVM) のいずれかを使用できます。

- Red Hat Enterprise Linux 9 KVM Guest Image
- Red Hat Enterprise Linux 8 KVM Guest Image

これらの QCOW2 イメージは、**cloud-init** を使用して設定されています。これが適切に機能するには、セキュアシェル (SSH) キーをプロビジョニングするための EC2 互換メタデータサービスが必要です。

QCOW2 形式の Ready Windows KVM インスタンスイメージは利用できません。

注記

KVM インスタンスイメージの場合:

- イメージでは **root** アカウントが非アクティブ化されていますが、**cloud-user** という名前の特別なユーザーには **sudo** アクセスが許可されています。
- このイメージには **root** パスワードは設定されていません。

root パスワードは、**/etc/shadow** で 2 番目のフィールドに **!!** と記載することによりロックされます。

RHOSP インスタンスでは、RHOSP Dashboard またはコマンドラインから SSH キーペアを生成し、その鍵の組み合わせを使用して、インスタンスに対して root ユーザーとして SSH 公開認証を実行します

インスタンスを起動すると、この公開鍵がインスタンスに注入されます。続いて、キーペア作成時にダウンロードする秘密鍵を使用して認証を行うことができます。

2.1.2. ベアメタルインスタンス用の RHEL ベースのルートパーティションイメージを作成する

ベアメタルインスタンスのカスタムルートパーティションイメージを作成するには、ベースとなる Red Hat Enterprise Linux KVM インスタンスイメージをダウンロードし、そのイメージを Image サービス (glance) にアップロードします。

手順

1. カスタマーポータル から、ベースとなる Red Hat Enterprise Linux KVM インスタンスイメージをダウンロードします。
2. **DIB_LOCAL_IMAGE** をダウンロードしたイメージとして定義します。

```
$ export DIB_LOCAL_IMAGE=rhel-<ver>-x86_64-kvm.qcow2
```

   - **<ver>** を、そのイメージの RHEL バージョン番号に置き換えます。

3. 登録方法に応じて登録情報を設定します。
   - Red Hat カスタマーポータルの場合:

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_AUTO_ATTACH=true
$ export REG_METHOD=portal
$ export https_proxy='<IP_address:port>' (if applicable)
$ export http_proxy='<IP_address:port>' (if applicable)
```

   - Red Hat Satellite の場合:

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_SAT_URL='<satellite-url>'
$ export REG_ORG='<satellite-org>'
$ export REG_ENV='<satellite-env>'
$ export REG_METHOD=<method>
```

   - 山かっこ **<>** 内の値を、適切な Red Hat カスタマーポータルの登録値、または Red Hat Satellite の登録値に置き換えます。

4. オプション: オフラインのリポジトリーがある場合は、**DIB_YUM_REPO_CONF** をローカルリポジトリーの設定として定義できます。

```
$ export DIB_YUM_REPO_CONF=<file-path>
```

   - **<file-path>** を、ローカルリポジトリー設定ファイルへのパスに置き換えます。

5. **diskimage-builder** ツールを使用して、カーネルを **rhel-image.vmlinuz** として、初期 RAM ディスクを **rhel-image.initrd** として展開します。

```
$ export DIB_RELEASE=<ver>
$ disk-image-create rhel baremetal \
  -o rhel-image
```

6. イメージを Image サービスにアップロードします。

```
$ KERNEL_ID=$(openstack image create \
  --file rhel-image.vmlinuz --public \
  --container-format aki --disk-format aki \
  -f value -c id rhel-image.vmlinuz)
$ RAMDISK_ID=$(openstack image create \
  --file rhel-image.initrd --public \
  --container-format ari --disk-format ari \
```

```
  -f value -c id rhel-image.initrd)
$ openstack image create \
  --file rhel-image.qcow2   --public \
  --container-format bare \
  --disk-format qcow2 \
  --property kernel_id=$KERNEL_ID \
  --property ramdisk_id=$RAMDISK_ID \
  rhel-root-partition-bare-metal-image
```

2.1.3. ベアメタルインスタンス用に RHEL ベースのディスク全体のユーザーイメージを作成する

ベアメタルインスタンス用にディスク全体のユーザーイメージを作成するには、ベースとなる Red Hat Enterprise Linux KVM インスタンスイメージをダウンロードし、そのイメージを Image サービス (glance) にアップロードします。

手順

1. カスタマーポータル から、ベースとなる Red Hat Enterprise Linux KVM インスタンスイメージをダウンロードします。
2. **DIB_LOCAL_IMAGE** をダウンロードしたイメージとして定義します。

```
$ export DIB_LOCAL_IMAGE=rhel-<ver>-x86_64-kvm.qcow2
```

   - **<ver>** を、そのイメージの RHEL バージョン番号に置き換えます。

3. 登録方法に応じて登録情報を設定します。
   - Red Hat カスタマーポータルの場合:

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_AUTO_ATTACH=true
$ export REG_METHOD=portal
$ export https_proxy='<IP_address:port>' (if applicable)
$ export http_proxy='<IP_address:port>' (if applicable)
```

   - Red Hat Satellite の場合:

```
$ export REG_USER='<username>'
$ export REG_PASSWORD='<password>'
$ export REG_SAT_URL='<satellite-url>'
$ export REG_ORG='<satellite-org>'
$ export REG_ENV='<satellite-env>'
$ export REG_METHOD=<method>
```

   - 山かっこ **<>** 内の値を、適切な Red Hat カスタマーポータルの登録値、または Red Hat Satellite の登録値に置き換えます。

4. オプション: オフラインのリポジトリーがある場合は、**DIB_YUM_REPO_CONF** をローカルリポジトリーの設定として定義できます。

```
$ export DIB_YUM_REPO_CONF=<file-path>
```

- **<file-path>** を、ローカルリポジトリー設定ファイルへのパスに置き換えます。

5. イメージを Image サービスにアップロードします。

```
$ openstack image create \
  --file rhel-image.qcow2 --public \
  --container-format bare \
  --disk-format qcow2 \
  rhel-whole-disk-bare-metal-image
```

2.2. RHEL または WINDOWS ISO ファイルを使用してインスタンスイメージを作成する

ISO ファイルから、QCOW2 形式でカスタム Red Hat Enterprise Linux (RHEL) イメージまたは Windows イメージを作成し、それを Red Hat OpenStack Platform (RHOSP) Image サービス (glance) にアップロードしてインスタンス作成時に使用できます。

2.2.1. 前提条件

- イメージを作成する Linux ホストマシン。これは、アンダークラウドまたはオーバークラウドを除き、Linux パッケージをインストールして実行できる任意のマシンです。
- **advanced-virt** リポジトリーが有効になっている。

```
$ sudo subscription-manager repos --enable=advanced-virt-for-rhel-<ver>-x86_64-rpms
```

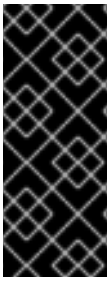
- ゲストオペレーティングシステムの作成に必要なすべてのパッケージが含まれる **virt-manager** アプリケーションがインストールされている。

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

- 仮想マシンイメージにアクセスして変更するためのツールセットが含まれる **libguestfs-tools** パッケージがインストールされている。

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

- RHEL 9 または 8 ISO ファイルまたは Windows ISO ファイル。RHEL ISO ファイルの詳細は、RHEL 9.0 Binary DVD または RHEL 8.6 Binary DVD を参照してください。Windows ISO ファイルがない場合は、Microsoft Evaluation Center にアクセスして評価イメージをダウンロードしてください。
- **kickstart** ファイルを編集する必要がある場合はテキストエディター (RHEL のみ)。



重要

アンダークラウドに **libguestfs-tools** パッケージをインストールする場合は、アンダークラウドの **tripleo_iscsid** サービスとのポートの競合を避けるために **iscsid.socket** を非アクティブ化します。

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

前提条件を満たすと、RHEL または Windows イメージの作成に進めます。

- Red Hat Enterprise Linux 9 イメージの作成
- Red Hat Enterprise Linux 8 イメージの作成
- Windows イメージの作成

2.2.2. Red Hat Enterprise Linux 9 イメージの作成

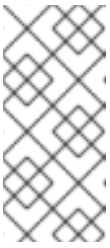
Red Hat Enterprise Linux (RHEL) 9 の ISO ファイルを使用して、QCOW2 形式の Red Hat OpenStack Platform (RHOSP) イメージを作成できます。

手順

1. **root** ユーザーとしてホストマシンにログオンします。
2. **virt-install** でインストールを開始します。

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

- 山かっこ **<>** 内の値を、使用している RHEL 9 イメージに応じた適切な値に置き換えます。このコマンドは、インスタンスを起動してインストールプロセスを開始します。



注記

インスタンスが自動的に起動しない場合には、**virt-viewer** のコマンドを実行して、コンソールを確認します。

```
[root@host]# virt-viewer <rhel9-cloud-image>
```

3. インスタンスを設定します。
   a. インストーラーの初期ブートメニューで、**Install Red Hat Enterprise Linux 9**を選択します。
   b. 適切な **言語** および **キーボード** オプションを選択します。
   c. インストールに使用するデバイス種別を尋ねるプロンプトが表示されたら、**自動検出したインストールメディア** を選択します。
   d. インストール先を尋ねるプロンプトが表示されたら、**ローカルの標準ディスク** を選択します。その他のストレージオプションについては、**Automatically configure partitioning** を選択します。
   e. **Which type of installation would you like?**ウィンドウで、SSH サーバーをインストールする **Basic Server** インストールを選択します。

f. ネットワークとホスト名の設定では、ネットワークに **eth0** を選択し、デバイスのホスト名を指定します。デフォルトのホスト名は **localhost.localdomain** です。

g. **root パスワード** フィールドにパスワードを入力し、**確認** フィールドに同じパスワードをもう一度入力します。

4. 画面上のメッセージでインストールの完了を確認できたら、インスタンスを再起動し、root ユーザーとしてログインします。

5. **/etc/sysconfig/network-scripts/ifcfg-eth0** ファイルを編集して、以下の値のみが記載されている状態にします。

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

6. マシンを再起動します。

7. コンテンツ配信ネットワークにマシンを登録します。

```
# sudo subscription-manager register
# sudo subscription-manager attach \
  --pool=<pool-id>
# sudo subscription-manager repos \
  --enable rhel-9-for-x86_64-baseos-rpms \
  --enable rhel-9-for-x86_64-appstream-rpms
```

- **pool-id** を、有効なプール ID に置き換えます。**subscription-manager list --available** コマンドを実行すると、使用可能なプール ID のリストが表示されます。

8. システムを更新します。

```
# dnf -y update
```

9. **cloud-init** パッケージをインストールします。

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

10. **/etc/cloud/cloud.cfg** 設定ファイルを編集し、**cloud_init_modules** の下に以下の内容を追加します。

```
- resolv-conf
```

resolv-conf オプションは、インスタンスの初回起動時に **resolv.conf** ファイルを自動的に設定します。このファイルには、**nameservers**、**domain**、その他のオプションなどのインスタンスに関連した情報が記載されています。

11. EC2 メタデータサービスにアクセスするときの問題を回避するには、次の行を **/etc/sysconfig/network** に追加します。

```
NOZEROCONF=yes
```

12. コンソールメッセージが Dashboard の **ログ** タブおよび **nova console-log** の出力に表示されるようにするには、以下のブートオプションを **/etc/default/grub** ファイルに追記します。

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

13. **grub2-mkconfig** コマンドを実行します。

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

以下のような出力が表示されます。

```
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
```

14. インスタンスの登録を解除して、作成されるイメージにこのインスタンスのサブスクリプション情報が含まれないようにします。

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

15. インスタンスの電源をオフにします。

```
# poweroff
```

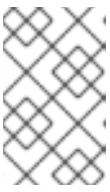
16. **virt-sysprep** コマンドでイメージのリセットおよびクリーニングをして、問題なくインスタンスの作成に使用できるようにします。

```
[root@host]# virt-sysprep -d <rhel9-cloud-image>
```

17. ディスクイメージ内の空き領域をホスト内の空き領域に変換して、イメージサイズを縮小します。

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

このコマンドは、コマンドが実行された場所に新しい **<rhel9-cloud.qcow2>** ファイルを作成します。



注記

インスタンスに適用されているフレーバーのディスクスペースに応じて、イメージをベースとするインスタンスのパーティションを手動でリサイズする必要があります。

<rhel9-cloud.qcow2> イメージファイルを Image サービスにアップロードする準備が整いました。このイメージを RHOSP デプロイメントにアップロードする方法の詳細は、イメージを Image サービスにアップロードする を参照してください。

2.2.3. Red Hat Enterprise Linux 8 イメージの作成

Red Hat Enterprise Linux (RHEL) 8 の ISO ファイルを使用して、QCOW2 形式の Red Hat OpenStack Platform (RHOSP) イメージを作成できます。

手順

1. **root** ユーザーとしてホストマシンにログオンします。
2. **virt-install** でインストールを開始します。

```
[root@host]# virt-install \
  --virt-type kvm \
  --name <rhel86-cloud-image> \
  --ram <2048> \
  --vcpus <2> \
  --disk <rhel86.qcow2>,format=qcow2,size=<10> \
  --location <rhel-8.6-x86_64-boot.iso> \
  --network=bridge:virbr0 \
  --graphics vnc,listen=127.0.0.1 \
  --noautoconsole \
  --os-variant <rhel8.6>
```

- 山かっこ **<>** 内の値を、使用している RHEL イメージに応じた適切な値に置き換えます。このコマンドは、インスタンスを起動してインストールプロセスを開始します。

注記

インスタンスが自動的に起動しない場合には、**virt-viewer** のコマンドを実行して、コンソールを確認します。

```
[root@host]# virt-viewer <rhel86-cloud-image>
```

3. インスタンスを設定します。
   a. インストーラーの初期ブートメニューで、**Install Red Hat Enterprise Linux 8**を選択します。
   b. 適切な **言語** および **キーボード** オプションを選択します。
   c. インストールに使用するデバイス種別を尋ねるプロンプトが表示されたら、**基本ストレージデバイス** を選択します。
   d. デバイスのホスト名を指定します。デフォルトのホスト名は **localhost.localdomain** です。
   e. **タイムゾーン** と **root** パスワードを設定します。
   f. **Which type of installation would you like?**ウィンドウで、SSH サーバーをインストールする **Basic Server**インストールを選択します。

4. 画面上のメッセージでインストールの完了を確認できたら、インスタンスを再起動し、root ユーザーとしてログインします。

5. **/etc/sysconfig/network-scripts/ifcfg-eth0** ファイルを編集して、以下の値のみが記載されている状態にします。

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

6. マシンを再起動します。

7. コンテンツ配信ネットワークにマシンを登録します。

```
# sudo subscription-manager register
# sudo subscription-manager attach \
  --pool=<pool-id>
# sudo subscription-manager repos \
  --enable rhel-8-for-x86_64-baseos-rpms \
  --enable rhel-8-for-x86_64-appstream-rpms
```

   - **pool-id** を、有効なプール ID に置き換えます。**subscription-manager list --available** コマンドを実行すると、使用可能なプール ID のリストが表示されます。

8. システムを更新します。

```
# dnf -y update
```

9. **cloud-init** パッケージをインストールします。

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

10. **/etc/cloud/cloud.cfg** 設定ファイルを編集し、**cloud_init_modules** の下に以下の内容を追加します。

```
- resolv-conf
```

    resolv-conf オプションは、インスタンスの初回起動時に **resolv.conf** ファイルを自動的に設定します。このファイルには、**nameservers**、**domain**、その他のオプションなどのインスタンスに関連した情報が記載されています。

11. ネットワークの問題が発生するのを防ぐために、**/etc/udev/rules.d/75-persistent-net-generator.rules** ファイルを作成します。

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

    これにより、**/etc/udev/rules.d/70-persistent-net.rules** ファイルが作成されるのを防ぎます。**/etc/udev/rules.d/70-persistent-net.rules** ファイルが作成されると、ネットワークインターフェイスが **eth0** ではなく **eth1** として作成されて IP アドレスが割り当てられないため、スナップショットから起動するときにネットワークが正しく機能しない可能性があります。

12. ISO イメージデータサービスに正常にアクセスするための問題を回避するには、次の行を

12. EC2 メタデータサービスにアクセスするときの問題を回避するには、次の行を **/etc/sysconfig/network** に追加します。

```
NOZEROCONF=yes
```

13. コンソールメッセージが Dashboard の **ログ** タブおよび **nova console-log** の出力に表示されるようにするには、以下のブートオプションを **/etc/grub.conf** ファイルに追記します。

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

14. インスタンスの登録を解除して、作成されるイメージにこのインスタンスの同じサブスクリプション情報が含まれないようにします。

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

15. インスタンスの電源をオフにします。

```
# poweroff
```

16. **virt-sysprep** コマンドでイメージのリセットおよびクリーニングをして、問題なくインスタンスの作成に使用できるようにします。

```
[root@host]# virt-sysprep -d <rhel86-cloud-image>
```

17. ディスクイメージ内の空き領域をホスト内の空き領域に変換して、イメージサイズを縮小します。

```
[root@host]# virt-sparsify \
  --compress <rhel86.qcow2> <rhel86-cloud.qcow2>
```

このコマンドは、コマンドが実行された場所に新しい **<rhel86-cloud.qcow2>** ファイルを作成します。

注記

インスタンスに適用されているフレーバーのディスクスペースに応じて、イメージをベースとするインスタンスのパーティションを手動でリサイズする必要があります。

<rhel86-cloud.qcow2> イメージファイルを Image サービスにアップロードする準備が整いました。このイメージを RHOSP デプロイメントにアップロードする方法の詳細は、イメージを Image サービスにアップロードする を参照してください。

2.2.4. Windows イメージの作成

Windows の ISO ファイルを使用して、QCOW2 形式で Red Hat OpenStack Platform (RHOSP) イメージを作成できます。

手順

1. **root** ユーザーとしてホストマシンにログオンします。

2. **virt-install** でインストールを開始します。

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

- 山かっこ **<>** 内の値を、使用している Windows イメージに応じた適切な値に置き換えます。

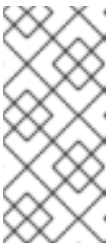
注記

--os-type=windows パラメーターにより、Windows インスタンスのクロックが正しく設定され、Hyper-V Enlightenment 機能が有効化されるようになります。Image サービス (glance) にイメージをアップロードする前に、イメージメタデータに **os_type=windows** を設定する必要もあります。

3. **virt-install** コマンドは、デフォルトでインスタンスイメージを **/var/lib/libvirt/images/<windows-image>.qcow2** として保存します。インスタンスイメージを別の場所に保存する場合は、**--disk** オプションのパラメーターを変更します。

```
--disk path=<file-name>,size=<size>
```

- **<file-name>** を、インスタンスイメージを保存するファイルの名前 (およびオプションでそのパス) に置き換えます。たとえば、**path=win8.qcow2,size=8** は現在の作業ディレクトリーに **win8.qcow2** という名前の 8 GB ファイルを作成します。



注記

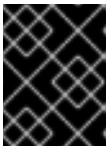
インスタンスが自動的に起動しない場合には、**virt-viewer** のコマンドを実行して、コンソールを確認します。

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

Windows のインストール方法に関する詳細は、Microsoft のドキュメントを参照してください。

4. 新規インストールした Windows システムで仮想化ハードウェアを使用できるようにするには、VirtIO ドライバーのインストールが必要な場合があります。詳細は、**仮想化の設定および管理** の Windows 仮想マシン用の KVM 準仮想化ドライバーのインストール を参照してください。

5. Windows システムで Cloudbase-Init をダウンロードして実行すると、設定が完了します。Cloudbase-Init のインストールの最後に、**Run Sysprep** と **Shutdown** チェックボックスを選択します。**Sysprep** ツールは、特定の Microsoft サービスで使用する OS ID を生成して、インスタンスを一意にします。



重要

Red Hat は Cloudbase-Init に関するテクニカルサポートは提供しません。問題が発生した場合は、Cloudbase Solutions に連絡する を参照してください。

Windows システムがシャットダウンすると、**<windows-image.qcow2>** イメージファイルを Image サービスにアップロードできます。このイメージを RHOSP デプロイメントにアップロードする方法の詳細は、イメージを Image サービスにアップロードする を参照してください。

2.3. UEFI セキュアブート用のイメージの作成

オーバークラウドに UEFI セキュアブートコンピュートノードが含まれている場合は、クラウドユーザーがセキュアブートインスタンスを起動するために使用できるセキュアブートインスタンスイメージを作成できます。

手順

1. UEFI セキュアブート用の新しいイメージを作成します。

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

   - **<base_image_file>** を UEFI および GUID パーティションテーブル (GPT) 標準をサポートし、EFI システムパーティションを含むイメージファイルに置き換えます。

2. デフォルトのマシンタイプが **q35** ではない場合は、マシンタイプを **q35** に設定します。

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

3. インスタンスを UEFI セキュアブートホストでスケジュールする必要があることを指定します。

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

2.4. 仮想ハードウェアのメタデータプロパティー

Compute サービス (nova) では、**libosinfo** データを使用してデフォルトのデバイスモデルを設定する操作が非推奨になりました。これに代わって、以下のイメージメタデータ属性を使用して、インスタンス用の最適な仮想ハードウェアを設定します。

- **os_distro**
- **os_version**
- **hw_cdrom_bus**
- **hw_disk_bus**
- **hw_scsi_model**
- **hw_vif_model**

- **hw_video_model**
- **hypervisor_type**

これらのメタデータプロパティーの詳細は、イメージ設定パラメーター を参照してください。

第3章 イメージ、イメージプロパティー、イメージ形式の管理

Red Hat OpenStack Platform (RHOSP) Image サービス (glance) でアップロード、インポート、または保存するイメージ、およびイメージのプロパティーと形式を管理します。

3.1. IMAGE サービスにイメージをアップロードする

--property オプションを指定した **glance image-create** コマンドを使用して、Red Hat OpenStack Platform (RHOSP) イメージサービス (glance) にイメージをアップロードできます。

glance image-create コマンドオプションのリストについては、Image サービスのコマンドオプションを参照してください。プロパティーキーの一覧は、イメージ設定パラメーター を参照してください。

手順

- **glance image-create** コマンドに **--property** のオプションを指定して実行する方法でイメージをアップロードします。
  以下に例を示します。

```
$ glance image-create --name <name> \
  --is-public true --disk-format <qcow2> \
  --container-format <bare> \
  --file </path/to/image> \
  --property <os_version>=<11.10>
```

  - **<name>** は、わかりやすいイメージ名に置き換えます。
  - **<disk-format>** を、None、ami、ari、aki、vhd、vhdx、vmdk、raw、qcow2、vdi、iso、ploop のいずれかのディスク形式に置き換えます。
  - **<container-format>** を、None、ami、ari、aki、bare、ovf、ova、docker のいずれかのコンテナー形式に置き換えます。
  - **</path/to/image>** をイメージファイルへのファイルパスに置き換えます。
  - **<os_version>** と **<11.10>** を、イメージに関連付けるプロパティーのキーと値のペアに置き換えます。イメージに関連付ける異なるキーと値のペアを使用して、**--property** オプションを複数回使用できます。

3.2. IMAGE サービスのイメージのインポートメソッド

以下の方法を使用して、Red Hat OpenStack Platform (RHOSP) イメージサービス (glance) にイメージをインポートできます。

- **web-download** (デフォルト) メソッドを使用して、URI からイメージをインポートする。
- **glance-direct** メソッドを使用して、ローカルファイルシステムからイメージをインポートする。
- **copy-image** メソッドを使用して、デプロイメント内の他の Image サービスバックエンドに既存のイメージをコピーします。このインポート方法は、デプロイで複数の Image サービスバックエンドが有効になっている場合にのみ使用してください。

web-download 方式はデフォルトで有効になっていますが、クラウド管理者が他のインポート方式を設定します。利用可能なインポートオプションを一覧表示するには、**glance import-info** コマンドを実行します。

3.2.1. リモート URI からイメージをインポートする

web-download イメージインポートメソッドを使用して、リモート URI から Red Hat OpenStack Platform (RHOSP) Image サービス (glance) にイメージをコピーできます。

Image サービスの **web-download** メソッドでは、2 段階のプロセスでインポートを実施します。

1. **web-download** 方式では、イメージレコードが作成されます。
2. **web-download** メソッドは、指定された URI からイメージを取得します。

URI は、オプションの **denylist** リストおよび **allowlist** リストのフィルタリングの対象となります。

イメージプロパティー注入プラグインにより、メタデータ属性をイメージに注入できます。注入されたプロパティーに応じて、イメージインスタンスを起動するコンピュートノードが決定します。

手順

- イメージを作成して、インポートするイメージの URI を指定します。

```
$ glance image-create-via-import \
   --container-format <container-format> \
   --disk-format <disk-format> \
   --name <name> \
   --import-method web-download \
   --uri <uri>
```

  - **<container-format>** を、None、ami、ari、aki、bare、ovf、ova、docker のいずれかのコンテナー形式に置き換えます。
  - **<disk-format>** を、None、ami、ari、aki、vhd、vhdx、vmdk、raw、qcow2、vdi、iso、ploop のいずれかのディスク形式に置き換えます。
  - **<name>** は、わかりやすいイメージ名に置き換えます。
  - **<uri>** は、イメージの URI に置き換えます。

検証

- イメージが利用可能か確認します。

```
$ glance image-show <image-id>
```

  - **<image-id>** は、イメージの作成時に指定したイメージ ID に置き換えます。

3.2.2. ローカルボリュームからイメージをインポートする

glance-direct イメージインポートメソッドは、イメージレコードを作成し、それによりイメージ ID が生成されます。ローカルボリュームから Image サービス (glance) にイメージをアップロードすると、イメージはステージングエリアに保存され、設定されたチェックに合格するとアクティブになります。

注記

高可用性 (HA) 設定で使用される場合、**glance-direct** メソッドには共通のステージングエリアが必要です。**glance-direct** インポート方法を使用してイメージをアップロードする場合、共有ステージングエリアが存在しない場合は HA 環境でアップロードが失敗する可能性があります。HA のアクティブ/アクティブ環境では、API コールは複数の Image サービスのコントローラーに分散されます。ダウンロード API コールは、イメージをアップロードする API コールとは別のコントローラーに送信することが可能です。

glance-direct イメージインポートメソッドでは、3 種類の呼び出しを使用してイメージをインポートします。

- **glance image-create**
- **glance image-stage**
- **glance image-import**

glance image-create-via-import コマンドを使用すると、3 つの **glance-direct** 呼び出しすべてを 1 つのコマンドで実行できます。

手順

1. Source コマンドで認証情報ファイルを読み込みます。
2. ローカルイメージをインポートするには、**glance image-create-via-import** コマンドを使用します。

```
$ glance image-create-via-import \
  --container-format <container-format> \
  --disk-format <disk-format> \
  --name <name> \
  --file </path/to/image>
```

   - **<container-format>** を、None、ami、ari、aki、bare、ovf、ova、docker のいずれかのコンテナー形式に置き換えます。
   - **<disk-format>** を、None、ami、ari、aki、vhd、vhdx、vmdk、raw、qcow2、vdi、iso、ploop のいずれかのディスク形式に置き換えます。
   - **<name>** は、わかりやすいイメージ名に置き換えます。
   - **</path/to/image>** をイメージファイルへのファイルパスに置き換えます。
   イメージがステージングエリアからバックエンドストレージの場所に移動すると、そのイメージはリストされます。ただし、イメージがアクティブになるには、多少時間がかかる場合があります。

検証

- イメージが利用可能か確認します。

```
$ glance image-show <image-id>
```

  - **<image-id>** は、イメージの作成時に指定したイメージ ID に置き換えます。

3.3. イメージプロパティーの更新

Red Hat OpenStack Platform (RHOSP) Image サービス (glance) に保存したイメージのプロパティーを更新します。

手順

- **glance image-update** コマンドに **--property** オプションを指定して実行する方法でイメージを更新します。
  以下に例を示します。

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

  - **glance image-update** コマンドオプションのリストについては、Image サービス (glance) コマンドオプション を参照してください。
  - プロパティーキーの一覧は、イメージ設定パラメーター を参照してください。

3.4. イメージ変換の有効化

GlanceImageImportPlugins パラメーターを有効にすることで、QCOW2 イメージを Image サービス (glance) にアップロードできます。その後、QCOW2 イメージを RAW 形式に変換できます。

注記

Red Hat Ceph Storage RADOS Block Device (RBD) を使用してイメージを保存し、Nova インスタンスを起動すると、イメージ変換が自動的に有効になります。

イメージ変換を有効にするには、次のパラメーター値を含む環境ファイルを作成します。**-e** オプションを使用して新しい環境ファイルを **openstack overcloud deploy** コマンドに含めます。

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

イメージ管理には Image service コマンドラインクライアントを使用します。

3.4.1. RAW 形式へのイメージの変換

Red Hat Ceph Storage は QCOW2 イメージを保管することはできますが、そのイメージを使用して仮想マシン (VM) のディスクをホストすることはできません。

アップロードした QCOW2 イメージから仮想マシンを作成する場合には、コンピュートノードはイメージをダウンロードして RAW に変換し、それを Ceph にアップロードし直してからでないと使用することができません。このプロセスは仮想マシンの作成時間に影響を及ぼします (特に、並行して仮想マシンを作成する場合)。

たとえば、複数の仮想マシンを同時に作成する場合には、Ceph クラスターへの変換済みイメージのアップロードが、すでに実行中の負荷に影響を及ぼす可能性があります。IOPS のこれらの負荷に対するリソースがアップロードプロセスにより枯渇し、ストレージの反応が遅くなる場合があります。

Ceph において仮想マシンをより効率的に起動するには (一時バックエンドまたはボリュームからの起動)、glance イメージの形式を RAW にする必要があります。

手順

1. イメージを RAW に変換すると、イメージサイズが元の QCOW2 イメージファイルより大きくなる場合があります。最終的な RAW イメージのサイズを確認するには、変換前に以下のコマンドを実行します。

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

2. イメージを QCOW2 から RAW 形式に変換します。

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

3.4.2. GlanceDiskFormats パラメーターを使用してディスクフォーマットを設定する

GlanceDiskFormats パラメーターを使用して、ディスクフォーマットを有効または拒否するように Image サービス (glance) を設定することができます。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. source コマンドでアンダークラウドの認証情報ファイルを読み込みます。

```
$ source ~/stackrc
```

3. 環境ファイルに **GlanceDiskFormats** パラメーターを追加します (例: **glance_disk_formats.yaml**)。

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

- たとえば、RAW および ISO ディスクフォーマットだけを有効にするには、以下の設定を使用します。

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

- QCOW2 ディスクイメージを拒否するには、以下の設定例を使用します。

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

4. ご自分の環境に該当するその他の環境ファイルと共に、新しい設定が含まれる環境ファイルを **openstack overcloud deploy** コマンドに追加します。

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

- **<overcloud_environment_files>** をデプロイメントに追加する環境ファイルのリストに置き換えます。
- **<new_environment_file>** を新しい設定が含まれる環境ファイルに置き換えます。

RHOSP で利用可能なディスクフォーマットの詳細は、イメージ設定パラメーター を参照してください。

3.4.3. RAW 形式でのイメージの保存

以前に作成したイメージを RAW 形式で保存するには、**GlanceImageImportPlugins** パラメーターを有効にして以下のコマンドを実行します。

```
$ 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>** は、イメージ名に置き換えます。この名前が **glance image-list** に表示されます。
- **<http://server/image.qcow2>** を、QCOW2 イメージの場所とファイル名に置き換えます。

注記

このコマンド例では、イメージレコードを作成し、**web-download** メソッドを使用してそのイメージレコードをインポートします。glance-api は、インポートプロセス中に **--uri** で定義した場所からイメージをダウンロードします。**web-download** が利用できない場合、**glanceclient** はイメージデータを自動的にダウンロードすることができません。利用可能なイメージのインポート方法をリスト表示するには、**glance import-info** コマンドを実行します。

3.5. イメージの非表示と再表示

ユーザーに表示される通常のリストからパブリックイメージを非表示にすることができます。たとえば、廃止された CentOS 7 イメージを非表示にし、最新バージョンだけを表示してユーザーエクスペリエンスを簡素化できます。ユーザーは、非表示のイメージを検出して使用することができます。

非表示のイメージを作成するには、**glance image-create** コマンドに **--hidden** 引数を追加します。

手順

- イメージを非表示にするには、以下を実行します。

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

- イメージを再表示するには、以下を実行します。

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

- 非表示にしたイメージをリスト表示するには、以下を実行します。

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

3.6. IMAGE サービスからイメージを削除する

Image サービス (glance) に保存する必要のない 1 つ以上のイメージを削除するには、**glance image-delete** コマンドを使用します。

手順

- 1 つ以上のイメージを削除します。

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

  - **<image-id>** を、削除するイメージの ID に置き換えます。

警告

glance image-delete コマンドは、イメージとイメージのすべてのコピー、およびイメージインスタンスとメタデータを完全に削除します。

第4章 イメージサービスのイメージインポート方法の設定

Image サービス (glance) のデフォルト設定は、Red Hat OpenStack Platform (RHOSP) のインストール時に使用する Orchestration サービス (heat) テンプレートで定義されます。イメージサービスの Orchestration サービステンプレートは、**deployment/glance/glance-api-container-puppet.yaml** です。

カスタム環境ファイルを使用して、イメージサービスの側面をカスタマイズできます。カスタム環境ファイルは、Orchestration サービステンプレートをカスタマイズするために使用できる特別なタイプのテンプレートです。Orchestration サービステンプレートと環境ファイルの詳細は、**director を使用した Red Hat OpenStack Platform のインストールと管理** の heat テンプレートの概要 を参照してください。

クラウド管理者は、**web-download** または **glance-direct** インポート方法を使用して、クラウドユーザーが独自のイメージをイメージサービスにアップロードできるように、イメージインポートワークフローを設定できます。アップロードされたイメージがストレージバックエンドでアクティブになる前にステージング領域で監視したり、メタデータのイメージプロパティーインジェクションプラグインやイメージ形式のイメージ変換プラグインなど、一連のプラグインを実行してユーザーイメージを検出可能にするようにインポートワークフローを設定したりできます。

web-download イメージのインポート方法はデフォルトで有効になっていますが、クラウド管理者は **glance-direct** 方式を設定できます。Red Hat OpenStack Platform (RHOSP) で利用可能なインポート方法の詳細は、イメージサービスのイメージのインポート方法 を参照してください。

4.1. GLANCE-DIRECT イメージインポート方法の設定

クラウド管理者が **glance-direct** イメージインポート方法を有効にすると、クラウドユーザーは、すべての Image サービス API ワーカーに共通する一時的な共有ストレージの場所である OpenStack Image サービス (Glance) の共有ステージングエリアにローカルイメージをアップロードできます。

手順

1. アンダークラウドホストに **stack** ユーザーとしてログインします。
2. **stackrc** アンダークラウド認証情報ファイルを入手します。

```
$ source ~/stackrc
```

3. インポートパラメーターを設定するには、YAML 環境ファイルを作成するか開きます。

   例

```
$ vi /home/stack/templates/<glance-import-settings>.yaml
```

   - **<glance-import-settings>** をファイル名に置き換えます。

4. 共有ステージングに必要な NFS バックエンドを設定します。

```
parameter_defaults:
  GlanceBackend: file
  GlanceNfsEnabled: true
  GlanceNfsShare: 192.168.122.1:/export/glance
```

5. **glance-direct** インポートメソッドを有効にするには、**GlanceEnabledImportMethods** パラメーターに **glance-direct** を追加します。

```
parameter_defaults:
  [...]
  GlanceEnabledImportMethods: glance-direct,web-download
```

6. **glance-direct** インポート方法に必要な NFS ステージングエリアを設定します。

```
parameter_defaults:
  [...]
  GlanceStagingNfsShare: 192.168.122.1:/export/glance-staging
```

web-download 以外の方法を有効にする場合は、**GlanceEnabledImportMethods** パラメーターが必要です。**GlanceBackend**、**GlanceNfsEnabled**、および **GlanceStagingNfsShare** パラメーターに関する詳細は、**オーバークラウドパラメーター** の イメージストレージ (glance) パラメーター を参照してください。

7. **<glance-import-settings>.yaml** ファイルを他の環境ファイルとともにスタックに追加し、オーバークラウドをデプロイします。

```
(undercloud)$ openstack overcloud deploy --templates \
-e [your environment files] \
-e /home/stack/templates/<glance-import-settings>.yaml
```

4.2. イメージの WEB インポートソースの制御

Web インポートによるイメージダウンロードのソースを制限することができます。そのためには、オプションの **glance-image-import.conf** ファイルに URI のブラックリストおよび許可リストを追加します。

3 段階のレベルで、イメージソースの URI を許可またはブロックすることができます。

- スキームレベル (allowed_schemes、disallowed_schemes)
- ホストレベル (allowed_hosts、disallowed_hosts)
- ポートレベル (allowed_ports、disallowed_ports)

レベルにかかわらず、許可リストとブロックリストの両方を指定した場合には、許可リストが優先されブロックリストは無視されます。

Image サービス (glance) は、以下の判断ロジックを使用してイメージソースの URI を検証します。

1. スキームを確認する。
   a. スキームが定義されていない場合: 拒否する。
   b. 許可リストがあり、そのスキームが許可リストに記載されていない場合: 拒否する。記載されている場合: c 項をスキップして 2 項に進む。
   c. ブロックリストがあり、そのスキームがブロックリストに記載されている場合: 拒否する。
2. ホスト名を確認する。
   a. ホスト名が定義されていない場合: 拒否する。

b. 許可リストがあり、そのホスト名が許可リストに記載されていない場合: 拒否する。記載されている場合: c 項をスキップして 3 項に進む。

c. ブロックリストがあり、そのホスト名がブロックリストに記載されている場合: 拒否する。

3. URI にポートが含まれていれば、ポートを確認する。

a. 許可リストがあり、そのポートが許可リストに記載されていない場合: 拒否する。記載されている場合: b 項をスキップして 4 項に進む。

b. ブロックリストがあり、そのポートがブロックリストに記載されている場合: 拒否する。

4. 有効な URI として受け入れる。

(許可リストに追加する、あるいはブロックリストに登録しないことにより) スキームを許可した場合には、URI にポートを含めないことでそのスキームのデフォルトポートを使用する URI は、すべて許可されます。URI にポートが含まれている場合には、URI はデフォルトの判断ロジックに従って検証されます。

4.2.1. イメージインポート許可リストの例

この例では、FTP のデフォルトポートは 21 です。

ftp は **allowed_schemes** のリストに含まれているため、イメージリソースへの URL ftp://example.org/some/resource が許可されます。

ただし、21 は **allowed_ports** のリストにないため、同じイメージリソースへのこの URL ftp://example.org:21/some/resource は拒否されます。

```
allowed_schemes = [http,https,ftp]
disallowed_schemes = []
allowed_hosts = []
disallowed_hosts = []
allowed_ports = [80,443]
disallowed_ports = []
```

4.2.2. イメージのインポートに関するブロックリストおよび許可リストのデフォルト設定

glance-image-import.conf ファイルは、以下のデフォルトオプションが含まれるオプションのファイルです。

- allowed_schemes: [**http**, **https**]
- disallowed_schemes: ブランク
- allowed_hosts: ブランク
- disallowed_hosts: ブランク
- allowed_ports: [80, 443]
- disallowed_ports: ブランク

デフォルトの設定を使用する場合、エンドユーザーは **http** または **https** スキームを使用することでしか URI にアクセスすることができません。ユーザーが指定することのできるポートは、**80** および **443**

だけです。(ユーザーはポートを指定する必要はありませんが、指定する場合には **80** または **443** のどちらかでなければなりません)。

glance-image-import.conf ファイルは、Image サービスのソースコードツリーの **etc/** サブディレクトリーにあります。お使いの Red Hat OpenStack Platform のリリースに対応する正しいブランチを使用してください。

4.3. イメージのインポートにメタデータを注入してインスタンスの起動場所を制御する

クラウドユーザーは、Image サービス (glance) にイメージをアップロードし、そのイメージを使用してインスタンスを起動できます。クラウドユーザーは、特定のコンピュートノードセット上でこれらのイメージを起動する必要があります。イメージメタデータプロパティーを使用して、コンピュートノードへのインスタンスの割り当てを制御できます。

イメージプロパティー注入プラグインにより、メタデータ属性がインポート時にイメージに注入されます。プロパティーは、**glance-image-import.conf** ファイルの **[image_import_opts]** セクションと **[inject_metadata_properties]** セクションを編集することで指定できます。**glance-image-import.conf** ファイルは、Image サービスのソースコードツリーの **etc/** サブディレクトリーにあります。使用している Red Hat OpenStack Platform (RHOSP) リリースに応じた適切なブランチを使用してください。

イメージプロパティー注入プラグインを有効にするには、**[image_import_opts]** セクションに以下の行を追加します。

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

メタデータの注入を特定ユーザーが提供したイメージに制限するには、**ignore_user_roles** パラメーターを設定します。たとえば、以下の設定では、**property1** に関する 1 つの値および **property2** に関する 2 つの値が、任意の非管理者ユーザーによってダウンロードされたイメージに注入されます。

```
[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
```

パラメーター **ignore_user_roles** は、プラグインが無視する Identity サービス (keystone) ロールのコンマ区切りリストです。つまり、イメージインポートの呼び出しを行うユーザーにこれらのロールが設定されている場合、プラグインはイメージに属性を注入しません。

パラメーター **inject** は、インポートされたイメージのイメージレコードに注入される属性と値のコンマ区切りリストです。それぞれの属性と値は、コロン **(':')** で区切る必要があります。

第5章 複数のストアに対応した IMAGE サービス

Red Hat OpenStack Platform (RHOSP) Image サービス (glance) では、分散エッジアーキテクチャーによる複数ストアの使用がサポートされます。そのため、すべてのエッジサイトにイメージプールを設定することができます。

5.1. 複数ストアでのイメージのコピー

分散エッジアーキテクチャーで複数のストアを使用する場合は、すべてのエッジサイトにイメージプールを設定できます。ハブサイトとも呼ばれる中央サイトとエッジサイトの間で、イメージをコピーすることができます。

イメージのメタデータには、各コピーの場所が含まれます。たとえば、2 つのエッジサイトに存在するイメージは、3 つの場所 (中央サイトおよび 2 つのエッジサイト) に単一の UUID として公開されます。つまり、多くのストアで単一の UUID を共有するイメージデータのコピーを持つことができます。場所についての詳細は、イメージの場所について を参照してください。

すべてのエッジサイトで RADOS ブロックデバイス (RBD) イメージプールを使用すると、Ceph RBD コピーオンライト (COW) とスナップショットレイヤリングテクノロジーを使用して仮想マシン (VM) をすばやく起動できます。これは、仮想マシンをボリュームからブートできるのと共に、ライブマイグレーションが可能であることを意味します。Ceph RBD を使用した階層化についての詳細は、**ブロックデバイスガイド**の Ceph ブロックデバイスの階層化 を参照してください。

エッジサイトでインスタンスを起動すると、必要なイメージがローカルの Image Service (glance) ストアに自動的にコピーされます。ただし、インスタンスの起動時に時間を節約するために、glance マルチストアを使用して central のイメージストアからエッジサイトにイメージを事前にコピーできます。

5.2. ストレージエッジアーキテクチャーの要件

エッジサイトでイメージを使用するには、次の要件を参照してください。

- 各イメージのコピーは、central サイトの Image サービス (glance) に存在している必要があります。
- イメージを他のエッジサイトにコピーする前に、エッジサイトから central の場所にイメージをコピーする必要があります。
- Red Hat Ceph Storage を使用して分散コンピュートノード (DCN) アーキテクチャーをデプロイする場合は、未加工のイメージを使用する必要があります。
- RADOS Block Device (RBD) は、Image、Compute、および Block Storage サービスのストレージドライバーである必要があります。
- それぞれのサイトで、**NovaComputeAvailabilityZone** および **CinderStorageAvailabilityZone** パラメーターに同じ値を割り当てる必要があります。

5.3. イメージを複数のストアにインポートする

相互運用可能なイメージのインポートワークフローを使用して、イメージデータを複数の Red Hat Ceph Storage クラスターにインポートします。ローカルファイルシステムで、または Web サーバーから利用可能なイメージを、Image サービス (glance) にインポートすることができます。

Web サーバーからイメージをインポートする場合、イメージを複数のストアに一度にインポートすることができます。イメージが Web サーバーで利用できない場合は、イメージをローカルファイルシステムから中央のストアにインポートし、それをさらに別のストアにコピーすることができます。詳細

は、複数ストアへの既存イメージのコピー を参照してください。

イメージ管理には Image service コマンドラインクライアントを使用します。

重要

中央サイトにイメージを使用するインスタンスがない場合でも、必ず中央サイトにイメージのコピーを保存してください。Image サービスへのイメージのインポートの詳細は、分散コンピュートノードのアーキテクチャー ガイドを参照してください。

5.3.1. イメージのインポート失敗時の対応

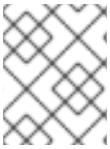
--allow-failure パラメーターを使用して、イメージインポート操作の失敗に対応することができます。

- **--allow-failure** パラメーターの値を **true** に設定した場合、最初のストアにデータが正常にインポートされると、イメージのステータスは **active** になります。これがデフォルトの設定です。**os_glance_failed_import** イメージ属性を使用して、イメージデータのインポートに失敗したストアのリストを表示することができます。

- **--allow-failure** パラメーターの値を **false** に設定すると、指定したすべてのストアにデータが正常にインポートされた場合に限り、イメージのステータスが **active** になります。いずれかのストアでイメージデータのインポートが失敗した場合、イメージのステータスは **failed** になります。イメージは指定されたどのストアにもインポートされません。

5.3.2. 複数ストアへのイメージデータのインポート

--allow-failure パラメーターのデフォルト設定は **true** なので、一部のストアがイメージデータのインポートに失敗するのを許容するのであれば、コマンドにパラメーターを追加する必要はありません。



注記

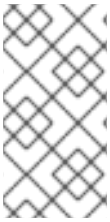
この手順では、すべてのストアがイメージデータを正常にインポートすることは求められません。

手順

- 指定した複数のストアにイメージデータをインポートします。

```
$ glance image-create-via-import \
--container-format bare \
--name <image-name> \
--import-method web-download \
--uri <uri> \
--stores <store-1>,<store-2>,<store-3>
```

  - **<image-name>** をインポートするイメージの名前に置き換えます。
  - **<uri>** をイメージの URI に置き換えます。
  - **<store-1>**、**<store-2>**、**<store-3>** を、イメージデータをインポートするストアの名前に置き換えます。
  - あるいは、**--stores** を **--all-stores true** に置き換え、すべてのストアにイメージをアップロードします。



注記

QCOW2 イメージを自動的に RAW 形式に変換する **glance image-create-via-import** コマンドは、**web-download** メソッドでのみ機能します。**glance-direct** メソッドを使用することはできますが、共有ファイルシステムが設定されたデプロイメントでのみ機能します。

5.3.3. 複数ストアへのイメージデータのインポート (失敗を許容しない)

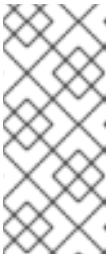
この手順では、すべてのストアがイメージデータを正常にインポートすることが求められます。

手順

1. 指定した複数のストアにイメージデータをインポートします。

```
$ glance image-create-via-import \
--container-format bare \
--name <image-name> \
--import-method web-download \
--uri <uri> \
--stores <store-1>,<store-2>,<store-3>
```

   - **<image-name>** をインポートするイメージの名前に置き換えます。
   - **<uri>** をイメージの URI に置き換えます。
   - **<store-1>**、**<store-2>**、**<store-3>** を、イメージデータをコピーするストアの名前に置き換えます。
   - あるいは、**--stores** を **--all-stores true** に置き換え、すべてのストアにイメージをアップロードします。



注記

--allow-failure パラメーターを **false** に設定すると、Image サービス (glance) はイメージデータのインポートに失敗したストアを無視しません。イメージ属性 **os_glance_failed_import** を使用して、失敗したストアのリストを表示することができます。詳細は、「イメージインポート操作の進捗の確認」 を参照してください。

2. イメージデータが特定のストアに追加されたことを確認します。

```
$ glance image-show <image-id> | grep stores
```

   <image-id> を元の既存イメージの ID に置き換えます。

   出力には、ストアのコンマ区切りリストが表示されます。

5.3.4. 1 つのストアへのイメージデータのインポート

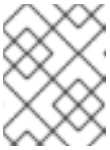
Image サービス (glance) を使用して、イメージデータを単一のストアにインポートできます。

手順

1. イメージデータを 1 つのストアにインポートします。

```
$ glance image-create-via-import \
--container-format bare \
--name <image-name> \
--import-method web-download \
--uri <uri> \
--store <store>
```

- **<image-name>** をインポートするイメージの名前に置き換えます。
- **<uri>** をイメージの URI に置き換えます。
- **<store>** をイメージデータをコピーするストアの名前に置き換えます。



注記

コマンドに **--stores**、**--all-stores**、または **--store** オプションを指定しないと、Image サービスは中央ストアにイメージを作成します。

2. イメージデータが特定のストアに追加されたことを確認します。

```
$ glance image-show <image-id> | grep stores
```

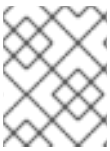
- **<image-id>** を元の既存イメージの ID に置き換えます。
出力には、ストアのコンマ区切りリストが表示されます。

5.4. イメージインポート操作の進捗の確認

相互運用可能なイメージのインポートワークフローでは、イメージデータが順次ストアにインポートされます。イメージのサイズ、ストア数、および中央サイトとエッジサイト間のネットワーク速度が、イメージのインポート操作が完了するのにかかる時間に影響を及ぼします。

イメージのインポート操作中に送付される通知に表示される 2 つのイメージ属性を見て、イメージインポートの進捗を把握することができます。

- **os_glance_importing_to_stores** 属性: イメージデータをインポートしていないストアがリスト表示されます。インポートの開始時点では、要求されたすべてのストアがリストに表示されます。ストアがイメージデータを正常にインポートするたびに、Image サービスはストアをリストから削除します。
- **os_glance_failed_import** 属性: イメージデータのインポートに失敗したストアがリスト表示されます。イメージインポート操作の開始時点では、このリストには何も表示されません。



注記

以下の手順の環境には、**central** ストアおよび 2 つのエッジストア (**dcn0** および **dcn1**) という 3 つの Red Hat Ceph Storage クラスターがあります。

手順

1. イメージデータが特定のストアに追加されたことを確認します。

```
$ glance image-show <image-id>
```

-

- **<image-id>** を元の既存イメージの ID に置き換えます。
  出力には、以下のスニペット例のようなストアのコンマ区切りリストが表示されます。

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

2. イメージインポート操作のステータスを監視します。このコマンドを **watch** コマンドの引数にすると、コマンドの出力は 2 秒ごとに更新されます。

```
$ watch glance image-show <image-id>
```

- **<image-id>** を元の既存イメージの ID に置き換えます。
  イメージのインポート操作が進むと、操作のステータスが変わります。

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

  イメージのインポートに失敗したことを示す出力は、以下の例のようになります。

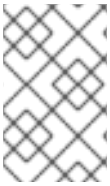
```
| os_glance_failed_import      | dcn0
| os_glance_importing_to_stores | dcn1
| status                       | importing
```

  操作が完了すると、ステータスが active に変わります。

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

5.5. 既存イメージを複数のストアにコピーする

この機能により、Red Hat OpenStack Image サービス (glance) のイメージデータを使用して、既存イメージを相互運用可能なイメージのインポートワークフローを使用してエッジにある複数の Red Hat Ceph Storage ストアにコピーできます。



注記

イメージをエッジサイトのいずれかにコピーするためには、そのイメージは中央サイトに存在していなければなりません。既存のイメージを新たに追加したストアにコピーできるのは、イメージの所有者または管理者だけです。

--all-stores を **true** に設定するか、イメージデータを受け取る特定のストアを指定して、既存のイメージデータをコピーすることができます。

- **--all-stores** オプションのデフォルト設定は **false** です。**--all-stores** が **false** の場合は、**--stores <store-1>,<store-2>** を使用してイメージデータを受け取るストアを指定する必要があります。指定されたストアにイメージデータがすでに存在する場合、要求は失敗します。

- **--all-stores** を **true** に設定した場合、一部のストアにイメージデータがすでに存在していたら、それらのストアはリストから除外されます。

イメージデータを受け取るストアを指定すると、Image サービス (glance) は中央サイトからステージングエリアにデータをコピーします。続いて、Image サービスは相互運用可能なイメージのインポートワークフローを使用してイメージデータをインポートします。詳細は、複数ストアへのイメージのインポート を参照してください。

イメージ管理には Image service コマンドラインクライアントを使用します。

重要

Red Hat では、短時間に連続してイメージのコピー要求を行わないことを推奨します。同じイメージに対して短時間に 2 回イメージのコピー操作を行うと、競合状態が発生し予期せぬ結果を招きます。既存のイメージデータに影響はありませんが、新規ストアへのデータコピーに失敗します。

5.5.1. 全ストアへのイメージのコピー

利用可能なすべてのストアにイメージデータをコピーするには、以下の手順を使用します。

手順

1. 利用可能なすべてのストアにイメージデータをコピーします。

```
$ glance image-import <image-id> \
--all-stores true \
--import-method copy-image
```

   - **<image-id>** をコピーするイメージの名前に置き換えます。

2. 利用可能なすべてのストアにイメージデータが正常に複製されたことを確認します。

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

   イメージインポート操作のステータスを確認する方法について、詳細は 「イメージインポート操作の進捗の確認」 を参照してください。

5.5.2. 特定ストアへのイメージのコピー

特定のストアにイメージデータをコピーするには、以下の手順を使用します。

手順

1. 特定のストアにイメージデータをコピーします。

```
$ glance image-import <image-id> \
--stores <store-1>,<store-2> \
--import-method copy-image
```

   - **<image-id>** をコピーするイメージの名前に置き換えます。
   - **<store-1>** と **<store-2>** を、イメージデータのコピー先のストア名に置き換えます。

2. 指定したストアにイメージデータが正常に複製されたことを確認します。

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

イメージインポート操作のステータスを確認する方法について、詳細は 「イメージインポート操作の進捗の確認」 を参照してください。

5.6. 特定ストアからのイメージの削除

Red Hat OpenStack Platform (RHOSP) Image サービス (glance) を使用して、特定のストアにある既存のイメージコピーを削除します。

イメージ管理には Image service コマンドラインクライアントを使用します。

手順

- 特定のストアからイメージを削除します。

```
$ glance stores-delete --store <store-id> <image-id>
```

- **<store-id>** を、イメージのコピーを削除するストアの名前に置き換えます。
- **<image-id>** を、削除するイメージの ID に置き換えます。

警告

lance image-delete コマンドを使用すると、すべてのサイトからイメージが完全に削除されます。イメージのすべてのコピーに加えて、イメージインスタンスおよびメタデータが削除されます。

5.7. イメージの場所と場所のプロパティーのリスト表示

イメージは複数のサイトに存在できますが、特定のイメージの Universal Unique Identifier (UUID) は 1 つだけです。イメージのメタデータには、各コピーの場所が含まれます。たとえば、2 つのエッジサイトに存在するイメージは、3 つの場所 (中央サイトおよび 2 つのエッジサイト) に単一の UUID として公開されます。

注記

イメージ管理には、OpenStack コマンドラインクライアントではなく、Image Service (glance) コマンドラインクライアントを使用します。ただし、**openstack image show** コマンドを使用して、イメージの場所のプロパティーを一覧表示します。**glance image-show** コマンドの出力には場所が含まれていません。

手順

1. イメージのコピーが存在するサイトを表示します。

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

| stores |  default_backend,dcn1,dcn2
```

この例では、イメージは中央サイト (**default_backend**) ならびに 2 つのエッジサイト (**dcn1** および **dcn2**) に存在します。

2. あるいは、**--include-stores** オプションを指定して **glance image-list** コマンドを実行し、イメージが存在するサイトを表示することができます。

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

| ID                                   | Name    | Stores

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

3. イメージの場所のプロパティーをリスト表示し、それぞれの場所の詳細を表示します。

```
$ 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 --)
```

イメージの属性には、各イメージの場所ごとに異なる Ceph RBD URI が表示されます。

この例では、中央のイメージの場所の URI は以下のとおりです。

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

URI は以下のデータで構成されます。

- **79b70c32-df46-4741-93c0-8118ae2ae284** は中央の Ceph FSID を表します。それぞれの Ceph クラスターは、固有の FSID を持ちます。
- すべてのサイトのデフォルト値は **images** です。これは、イメージが保存される Ceph プールを表します。
- **2bd882e7-1da0-4078-97fe-f1bb81f61b00** はイメージの UUID を表します。あるイメージの UUID は、場所に関係なく同一です。
- メタデータには、この場所がマッピングする glance ストアが表示されます。この例では、中央のハブサイトである **default_backend** にマッピングします。

付録A IMAGE サービスのコマンドオプション

glance image-create、**glance image-create-via-import**、および **glance image-update** コマンドで次のオプションの引数を使用できます。

表A.1 コマンドオプション

対象コンポーネント	オプション	説明
すべて	**--architecture <ARCHITECTURE>**	https://docs.openstack.org/glance/latest/user/common-image-properties.html#architecture で指定されているオペレーティングシステムアーキテクチャー
すべて	**--protected [True_False]**	true の場合、イメージは削除できません。
すべて	**--name <NAME>**	イメージのわかりやすい名前
すべて	**--instance-uuid <INSTANCE_UUID>**	このイメージが関連付けられているインスタンスを記録するために使用できるメタデータ。(情報のみ。インスタンスのスナップショットは作成されません。)
すべて	**--min-disk <MIN_DISK>**	イメージの起動に必要なディスク容量 (GB)。
すべて	**--visibility <VISIBILITY>**	イメージアクセシビリティの範囲。有効な値: public、private、community、shared
すべて	**--kernel-id <KERNEL_ID>**	AMI 形式のイメージをブートする際にカーネルとして使用する必要のある Image サービス (glance) に保管されているイメージの ID
すべて	**--os-version <OS_VERSION>**	ディストリビューターによって指定されるオペレーティングシステムのバージョン
すべて	**--disk-format <DISK_FORMAT>**	ディスクのフォーマット。有効な値: None、ami、ari、aki、vhd、vhdx、vmdk、raw、qcow2、vdi、iso、ploop
すべて	**--os-distro <OS_DISTRO>**	https://docs.openstack.org/glance/latest/user/common-image-properties.html#os-distro で指定されているオペレーティングシステムディストリビューションの一般名
すべて	**--owner <OWNER>**	イメージの所有者
すべて	**--ramdisk-id <RAMDISK_ID>**	AMI 形式のイメージをブートする際に ramdisk として使用する必要のある、Image サービスに保管されているイメージの ID

対象コンポーネント	オプション	説明
すべて	**--min-ram <MIN_RAM>**	イメージの起動に必要な RAM の量 (MB)。
すべて	**--container-format <CONTAINER_FORMAT>**	コンテナーのフォーマット。有効な値: None、ami、ari、aki、bare、ovf、ova、docker
すべて	**--property <key=value>**	イメージに関連付ける任意のプロパティー。複数回の使用が可能です。
glance image-create	**--tags <TAGS> [<TAGS> ...]**	イメージに関連する文字列のリスト
glance image-create	**--id <ID>**	イメージの識別子
glance image-update	**--remove-property**	イメージから削除する任意のプロパティーのキー名。

付録B イメージの設定パラメーター

次のキーは、**glance image-create**、**glance image-create-via-import**、**glance image-update** コマンドの **--property** オプションで使用できます。

表B.1 属性のキー

対象コンポーネント	キー	説明	サポートされている値
すべて	**architecture**	ハイパーバイザーがサポートする必要のある CPU アーキテクチャー。たとえば、**x86_64**、**arm**、**ppc64** 等。マシンのアーキテクチャーを確認するには、**uname -m** を実行します。	• **aarch** - ARM 64-bit • **alpha** - DEC 64 ビット RISC • **armv7l** - ARM Cortex-A7 MPCore • **cris** - Ethernet, Token Ring、AXis-Code Reduced Instruction Set • **i686** - Intel sixth-generation x86 (P6 マイクロアーキテクチャー) • **ia64** - Itanium • **lm32** - Lattice Micro32 • **m68k** - Motorola 68000 • **microblaze** - Xilinx 32 ビット FPGA (Big Endian) • **microblazeel** - Xilinx 32 ビット FPGA (Little Endian) • **mips** - MIPS 32 ビット RISC (Big Endian) • **mipsel** - MIPS 32 ビット RISC (Little Endian) • **mips64** - MIPS 64 ビット RISC (Big Endian) • **mips64el** - MIPS 64 ビット RISC (Little Endian) • **openrisc** - OpenCores RISC • **parisc** - HP Precision Architecture RISC • **parisc64** - HP Precision Architecture 64 ビット RISC • **ppc** - PowerPC 32 ビット

対象コンポーネント	キー	説明	サポートされている値
			• **ppc64** - PowerPC 64 ビット • **ppcemb** - PowerPC (Embedded 32 ビット) • **s390** - IBM Enterprise Systems Architecture/390 • **s390x** - S/390 64 ビット • **sh4** - SuperH SH-4 (Little Endian) • **sh4eb** - SuperH SH-4 (Big Endian) • **sparc** - Scalable Processor Architecture、32 ビット • **sparc64** - Scalable Processor Architecture、64 ビット • **unicore32** - Microprocessor Research and Development Center RISC Unicore32 • **x86_64** - IA-32 の 64 ビット拡張 • **xtensa** - Tensilica Xtensa 設定可能マイクロプロセッサーコア • **xtensaeb** - Tensilica Xtensa 設定可能マイクロプロセッサーコア (Big Endian)
すべて	**hypervisor_type**	ハイパーバイザーの種別	**kvm**、**vmware**
すべて	**instance_uuid**	スナップショットイメージの場合、このイメージを作成するのに使用したサーバーの UUID	有効なサーバーの UUID
すべて	**kernel_id**	AMI 形式のイメージをブートする際にカーネルとして使用する必要のある Image サービスに保管されているイメージの ID	有効なイメージ ID
すべて	**os_distro**	オペレーティングシステムのディストリビューションの小文字による一般名	• **arch** - Arch Linux。**archlinux** および **org.archlinux** は使用しないでください。

対象コンポーネント	キー	説明	サポートされている値
			• **centos** - Community ENTerprise Operating System。**org.centos** および **CentOS** は使用しないでください。 • **debian** - Debian。**Debian** および **org.debian** は使用しないでください。 • **fedora**: Fedora。**Fedora**、**org.fedora**、**org.fedoraproject** は使用しないでください。 • **freebsd**: FreeBSD。**org.freebsd**、**freeBSD**、**FreeBSD** は使用しないでください。 • **gentoo**: Gentoo Linux。**Gentoo** および **org.gentoo** は使用しないでください。 • **mandrake**: Mandrakelinux (MandrakeSoft) ディストリビューション。**mandrakelinux** および **MandrakeLinux** は使用しないでください。 • **mandriva**: Mandriva Linux。**mandrivalinux** は使用しないでください。 • **mes**: Mandriva Enterprise Server。**mandrivaent** および **mandrivaES** は使用しないでください。 • **msdos** Microsoft Disc Operating System。**ms-dos** は使用しないでください。 • **netbsd**: NetBSD。**NetBSD** および **org.netbsd** は使用しないでください。 • **netware**: Novell NetWare。**novell** および **NetWare** は使用しないでください。 • **openbsd**: OpenBSD。**OpenBSD** および **org.openbsd** は使用しないでください。 • **opensolaris**: OpenSolaris。**OpenSolaris** および **org.opensolaris** は使用しないでください。

対象コンポーネント	キー	説明	サポートされている値
			• **opensuse**: openSUSE。**suse**、**SuSE**、**org.opensuse** は使用しないでください。 • **rhel**: Red Hat Enterprise Linux。**redhat**、**RedHat**、**com.redhat** は使用しないでください。 • **sled**: SUSE Linux Enterprise Desktop。**com.suse** は使用しないでください。 • **ubuntu**: Ubuntu。**Ubuntu**、**com.ubuntu**、**org.ubuntu**、**canonical** は使用しないでください。 • **windows**: Microsoft Windows。**com.microsoft.server** は使用しないでください。
すべて	**os_version**	ディストリビューターによって指定されるオペレーティングシステムのバージョン	バージョン番号 (例: "11.10")
すべて	**ramdisk_id**	AMI 形式のイメージをブートする際に ramdisk として使用する必要のある、Image サービスに保管されているイメージの ID	有効なイメージ ID
すべて	**vm_mode**	仮想マシンのモード。仮想マシンに使用されるホスト/ゲストの ABI (アプリケーションバイナリーインターフェイス) を示します。	**hvm**: 完全仮想化。これは QEMU および KVM で使用されるモードです。
libvirt API ドライバー	**hw_cdrom_bus**	CD-ROM デバイスの接続先となるディスクコントローラーの種別を指定します。	**scsi**、**virtio**、**ide**、**usb** のいずれか。**iscsi** を指定する場合は、**hw_scsi_model** パラメーターを **virtio-scsi** に設定する必要があります。
libvirt API ドライバー	**hw_disk_bus**	ディスクデバイスの接続先となるディスクコントローラーのタイプを指定します。	**scsi**、**virtio**、**ide**、**usb** のいずれか。**iscsi** を使用している場合には、**hw_scsi_model** を **virtio-scsi** に設定する必要がある点に注意してください。

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**hw_firmware_type**	インスタンスの起動に使用するファームウェアのタイプを指定します。	以下の有効な値のいずれかに設定します。 • **bios** • **uefi**
libvirt API ドライバー	**hw_machine_type**	指定されたマシンタイプを使用して、ARM システムを起動できるようにします。ARM イメージが使用されており、そのマシンタイプが明示的に指定されていない場合、Compute は仮想マシンタイプを ARMv7 および AArch64 のデフォルトとして使用します。	有効なタイプは、**virsh capabilities** コマンドを使用すると、表示できます。マシンタイプはマシンタグに表示されます。
libvirt API ドライバー	**hw_numa_nodes**	インスタンスに公開する NUMA ノードの数 (フレーバーの定義はオーバーライドしません)	整数
libvirt API ドライバー	**hw_numa_cpus.0**	仮想 CPU N-M から NUMA ノード 0 へのマッピング (フレーバーの定義はオーバーライドしません)	整数のコンマ区切りリスト
libvirt API ドライバー	**hw_numa_cpus.1**	仮想 CPU N-M から NUMA ノード 1 へのマッピング (フレーバーの定義はオーバーライドしません)	整数のコンマ区切りリスト
libvirt API ドライバー	**hw_numa_mem.0**	N MB の RAM から NUMA ノード 0 へのマッピング (フレーバーの定義はオーバーライドしません)	整数
libvirt API ドライバー	**hw_numa_mem.1**	N MB の RAM から NUMA ノード 1 へのマッピング (フレーバーの定義はオーバーライドしません)	整数

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**hw_pci_numa_affinity_policy**	PCI パススルーデバイスおよび SR-IOV インターフェイスの NUMA アフィニティーポリシーを指定します。	以下の有効な値のいずれかに設定します。 ● **required**: インスタンスの NUMA ノードの少なくとも 1 つが PCI デバイスとのアフィニティーを持つ場合に限り、Compute サービスは PCI デバイスを要求するインスタンスを作成します。このオプションは、最高のパフォーマンスを提供します。 ● **preferred**: Compute サービスは、NUMA アフィニティーに基づきベストエフォートで PCI デバイスの選択を試みます。アフィニティーを使用できない場合には、Compute サービスは PCI デバイスとのアフィニティーを持たない NUMA ノード上でインスタンスをスケジュールします。 ● **legacy**: (デフォルト) 以下のどちらかのケースで、Compute サービスは PCI デバイスを要求するインスタンスを作成します。 ○ PCI デバイスが少なくとも 1 つの NUMA ノードとのアフィニティーを持つ。 ○ PCI デバイスが NUMA アフィニティーに関する情報を提供しない。
libvirt API ドライバー	**hw_qemu_guest_agent**	ゲストエージェントのサポート。**yes** に設定し、かつ **qemu-ga** もインストールされている場合には、ファイルシステムが休止 (フリーズ) し、スナップショットが自動的に作成されます。	**yes** / **no**

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**hw_rng_model**	このイメージを使用して起動したインスタンスに、乱数生成器 (RNG) デバイスを追加します。 インスタンスフレーバーにより、RNG デバイスがデフォルトで有効になります。RNG デバイスを無効にするには、クラウド管理者はフレーバーで **hw_rng:allowed** を **False** に設定する必要があります。 デフォルトのエントロピーソースは **/dev/random** です。ハードウェアの乱数生成器を指定するには、Compute 環境ファイルで **rng_dev_path** を **/dev/hwrng** に設定します。	**virtio** またはその他のサポートされているデバイス
libvirt API ドライバー	**hw_scsi_model**	VirtIO SCSI (virtio-scsi) の使用を有効にして、コンピュートインスタンスのブロックデバイスアクセスを提供します。デフォルトでは、インスタンスは VirtIO Block (virtio-blk) を使用します。VirtIO SCSI は準仮想化 SCSI コントローラーデバイスで、より高いスケーラビリティーとパフォーマンスを提供し、高度な SCSI ハードウェアに対応します。	**virtio-scsi**
libvirt API ドライバー	**hw_tpm_model**	使用する TPM デバイスのモデルに設定します。**hw:tpm_version** が設定されていない場合は無視されます。	- **tpm-tis**: (デフォルト) TPM インターフェイス仕様。 - **tpm-crb**: コマンド応答バッファー。TPM バージョン 2.0 とのみ互換性があります。

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**hw_tpm_version**	使用する TPM のバージョンを設定します。TPM バージョン **2.0** が唯一サポートされているバージョンです。	**2.0**

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**hw_video_model**	仮想マシンインスタンスで使用するディスプレイデバイス用のビデオデバイスドライバー。	次のいずれかの値に設定して、使用するサポートされているドライバーを指定します。 ● **virtio** - (デフォルト) ほとんどのアーキテクチャーでサポートされている仮想マシンディスプレイデバイスに **推奨** されるドライバー。VirtIO GPU ドライバーは、RHEL-7 以降、および Linux カーネルバージョン 4.4 以降に含まれています。インスタンスカーネルに VirtIO GPU ドライバーがある場合、インスタンスはすべての VirtIO GPU 機能を使用できます。インスタンスカーネルに VirtIO GPU ドライバーがない場合、VirtIO GPU デバイスは VGA 互換モードに正常にフォールバックし、インスタンスに機能する表示を提供します。 ● **qxl** - メンテナンスされなくなった Spice または noVNC 環境用の **非推奨** ドライバー。 ● **cirrus** - 下位互換性のためにのみサポートされているレガシードライバー。新しいインスタンスには使用しないでください。 ● **vga** - IBM Power 環境用にこのドライバーを使用します。 ● **gop** - QEMU/KVM 環境ではサポートされません。 ● **xen** - KVM 環境ではサポートされません。 ● **vmvga** - レガシードライバー。使用しないでください。 ● **none** - この値を使用して、仮想 GPU (vGPU) インスタンスのエミュレートされたグラフィックスまたはビデオを無効にします。この場合、ドライバーは別途設定されます。

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**hw_video_ram**	ビデオイメージの最大 RAM。フレーバーの **extra_specs** で **hw_video:ram_max_mb** の値が設定済みで、かつその値が **hw_video_ram** で設定されている値を上回る場合にのみ使用されます。	整数 (MB 単位。例: 64)
libvirt API ドライバー	**hw_watchdog_action**	サーバーがハングした場合に指定したアクションを実行する仮想ハードウェアウォッチドッグデバイスを有効にします。このウォッチドッグは、i6300esb デバイスを使用します (PCI Intel 6300ESB をエミュレート)。**hw_watchdog_action** が指定されていない場合には、ウォッチドッグは無効になります。	• disabled: デバイスは接続されていません。イメージのフレーバーを使用して有効化されている場合でも、ユーザーがイメージのウォッチドッグを無効にすることができます。このパラメーターのデフォルト値は disabled です。 • reset: ゲストを強制的にリセットします。 • poweroff: ゲストの電源を強制的に切断します。 • pause: ゲストを一時停止します。 • none: ウォッチドッグを有効化するのみで、サーバーがハングした場合には何もしません。
libvirt API ドライバー	**os_command_line**	デフォルトではなく、**libvirt** ドライバーで使用されるカーネルコマンドライン。Linux Containers (LXC) の場合は、この値が初期化の引数として使用されます。このキーは、Amazon カーネル、ramdisk、またはマシンイメージ (aki、ari、または ami) にのみ有効です。	

対象コンポーネント	キー	説明	サポートされている値
libvirt API ドライバー	**os_secure_boot**	UEFI セキュアブートで保護されたインスタンスを作成するために使用します。	以下の有効な値のいずれかに設定します。 ● **required**: このイメージで起動したインスタンスのセキュアブートを有効にします。インスタンスは、Compute サービスがセキュアブートをサポートできるホストを見つけた場合のみ、起動します。ホストが見つからない場合、Compute サービスはエラー "No valid host" を返します。 ● **disabled**: このイメージで起動したインスタンスのセキュアブートを無効にします。デフォルトでは無効になっています。 ● **optional**: ホストがセキュアブートをサポートできると Compute サービスが判断した場合のみ、このイメージで起動したインスタンスのセキュアブートを有効にします。
libvirt API ドライバーおよび VMware API ドライバー	**hw_vif_model**	使用する仮想ネットワークインターフェイスデバイスのモデルを指定します。	設定したハイパーバイザーによって有効なオプションは異なります。 ● KVM および QEMU: e1000、ne2k_pci、pcnet、rtl8139、virtio ● VMware: e1000、e1000e、VirtualE1000、VirtualE1000e、VirtualPCNet32、VirtualSriovEthernetCard、VirtualVmxnet ● Xen: e1000、netfront、ne2k_pci、pcnet、rtl8139
VMware API ドライバー	**vmware_adaptertype**	ハイパーバイザーが使用する仮想 SCSI または IDE コントローラー	**lsiLogic**、**busLogic**、または **ide**

対象コンポーネント	キー	説明	サポートされている値
VMware API ドライバー	**vmware_ostype**	イメージにインストールされているオペレーティングシステムを示す VMware GuestID。この値は、仮想マシンの作成時にハイパーバイザーに渡されます。指定しなかった場合には、このキーの値はデフォルトの **otherGuest** に設定されます。	詳細は、Images with VMware vSphere を参照してください。
VMware API ドライバー	**vmware_image_version**	現在は使用されていません。	**1**
XenAPI ドライバー	**auto_disk_config**	true に指定した場合には、ディスク上のルートパーティションは、インスタンスがブートする前に自動的にリサイズされます。この値は、Xen ベースのハイパーバイザーを XenAPI ドライバーと共に使用する場合にのみ Compute サービスによって考慮されます。Compute サービスは、イメージに単一のパーティションがあり、かつそのパーティションが **ext3** または **ext4** のフォーマットの場合にのみリサイズを試みます。	**true** / **false**
libvirt API ドライバーおよび XenAPI ドライバー	**os_type**	イメージ上にインストールされるオペレーティングシステム。XenAPI ドライバーには、イメージの **os_type** パラメーターの値によって異なるアクションを実行するロジックが組み込まれています。たとえば、**os_type=windows** イメージの場合には、Linux スワップパーティションの代わりに、FAT32 ベースのスワップパーティションを作成し、挿入されるホスト名を 16 文字未満に制限します。	**linux** または **windows**