Legal Notice

Copyright © 2022 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at http://creativecommons.org/licenses/by-sa/3.0/. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux® is the registered trademark of Linus Torvalds in the United States and other countries.

Java® is a registered trademark of Oracle and/or its affiliates.

XFS® is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL® is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js® is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack® Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation’s permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

Abstract

This guide provides procedures for creating and managing images, and procedures for configuring the Image service.
Table of Contents

MAKING OPEN SOURCE MORE INCLUSIVE ......................................................... 3

CHAPTER 1. THE IMAGE SERVICE (GLANCE) ...................................................... 4
  1.1. UNDERSTANDING THE IMAGE SERVICE ............................................. 4
     1.1.1. Supported Image service back ends ........................................... 4

CHAPTER 2. IMAGE IMPORT AND SHARED STAGING ........................................ 6
  2.1. CREATING AND DEPLOYING THE GLANCE-SETTINGS.YAML FILE ........... 6
  2.2. CONTROLLING IMAGE WEB-IMPORT SOURCES ...................................... 7
  2.3. IMAGE IMPORT EXAMPLE .................................................................. 7
  2.4. DEFAULT IMAGE IMPORT BLOCKLIST AND ALLOWLIST SETTINGS .......... 8
  2.5. INJECTING METADATA ON IMAGE IMPORT TO CONTROL WHERE VMS LAUNCH 8

CHAPTER 3. IMAGE SERVICE WITH MULTIPLE STORES .................................... 10
  3.1. REQUIREMENTS OF STORAGE EDGE ARCHITECTURE ............................ 10
  3.2. IMPORT AN IMAGE TO MULTIPLE STORES .......................................... 10
     3.2.1. Manage image import failures .................................................. 10
     3.2.2. Importing image data to multiple stores ...................................... 11
     3.2.3. Importing image data to multiple stores without failure ................. 11
     3.2.4. Importing image data to a single store ....................................... 12
     3.2.5. Checking the progress of the image import operation ..................... 13
  3.3. COPY AN EXISTING IMAGE TO MULTIPLE STORES ............................... 14
     3.3.1. Copying an image to all stores ................................................... 14
     3.3.2. Copying an image to specific stores ........................................... 15
  3.4. DELETING AN IMAGE FROM A SPECIFIC STORE ................................... 15
  3.5. UNDERSTANDING THE LOCATIONS OF IMAGES .................................... 16

APPENDIX A. IMAGE CONFIGURATION PARAMETERS ..................................... 18
MAKING OPEN SOURCE MORE INCLUSIVE

Red Hat is committed to replacing problematic language in our code, documentation, and web properties. We are beginning with these four terms: master, slave, blacklist, and whitelist. Because of the enormity of this endeavor, these changes will be implemented gradually over several upcoming releases. For more details, see our CTO Chris Wright’s message.
CHAPTER 1. THE IMAGE SERVICE (GLANCE)

Manage images and storage in Red Hat OpenStack Platform (RHOSP).

A virtual machine image is a file that contains a virtual disk with a bootable operating system installed. Virtual machine images are supported in different formats. The following formats are available in RHOSP:

- **RAW** - Unstructured disk image format.
- **QCOW2** - Disk format supported by QEMU emulator. This format includes QCOW2v3 (sometimes referred to as QCOW3), which requires QEMU 1.1 or higher.
- **ISO** - Sector-by-sector copy of the data on a disk, stored in a binary file.
- **AKI** - Indicates an Amazon Kernel Image.
- **AMI** - Indicates an Amazon Machine Image.
- **ARI** - Indicates an Amazon RAMDisk Image.
- **VDI** - Disk format supported by VirtualBox virtual machine monitor and the QEMU emulator.
- **VHD** - Common disk format used by virtual machine monitors from VMware, VirtualBox, and others.
- **PLOOP** - A disk format supported and used by Virtuozzo to run OS containers.
- **OVA** - Indicates that what is stored in the Image service (glance) is an OVA tar archive file.
- **DOCKER** - Indicates that what is stored in the Image service (glance) is a Docker tar archive of the container file system.

Although **ISO** is not normally considered a virtual machine image format, because ISOs contain bootable file systems with an installed operating system, you use them in the same way as other virtual machine image files.

To download the official Red Hat Enterprise Linux cloud images, your account must have a valid Red Hat Enterprise Linux subscription:

- **Red Hat Enterprise Linux 8 KVM Guest Image**
- **Red Hat Enterprise Linux 7 KVM Guest Image**
- **Red Hat Enterprise Linux 6 KVM Guest Image**

If you are not logged in to the Customer Portal, a prompt opens where you must enter your Red Hat account credentials.

1.1. UNDERSTANDING THE IMAGE SERVICE

Red Hat OpenStack Platform (RHOSP) Image service (glance) features.

1.1.1. Supported Image service back ends

The following Image service (glance) back-end scenarios are supported:
- RBD is the default back end when you use Ceph.
- RBD multi-store. For more information, see Section 3.1, "Requirements of storage edge architecture".
- Object Storage (swift)
- Block Storage (cinder). The Image service uses the Block Storage type and back end as the default.
- NFS
CHAPTER 2. IMAGE IMPORT AND SHARED STAGING

The default settings for the OpenStack Image service (glance) are determined by the heat templates that you use when you install Red Hat OpenStack Platform. The Image service heat template is `de...` file.

You can import images with the following methods:

web-download
   Use the `web-download` method to import an image from a URL.

glance-direct
   Use the `glance-direct` method to import an image from a local volume.

2.1. CREATING AND DEPLOYING THE GLANCE-SETTINGS.YAML FILE

Use a custom environment file to configure the import parameters. These parameters override the default values that are present in the core heat template collection. The example environment content contains parameters for the interoperable image import.

```yaml
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
```

The `GlanceBackend`, `GlanceNfsEnabled`, and `GlanceNfsShare` parameters are defined in the Storage Configuration section in the Advanced Overcloud Customization Guide.

Use two new parameters for interoperable image import to define the import method and a shared NFS staging area.

**GlanceEnabledImportMethods**

Defines the available import methods, web-download (default) and glance-direct. This parameter is necessary only if you want to enable additional methods besides web-download.

**GlanceStagingNfsShare**

Configures the NFS staging area that the glance-direct import method uses. This space can be shared among nodes in a high-availability cluster configuration. If you want to use this parameter, you must also set the `GlanceNfsEnabled` parameter to `true`.

Procedure

1. Create a new file, for example, `glance-settings.yaml`. Use the syntax from the example to populate this file.

2. Include the `glance-settings.yaml` file in the `openstack overcloud deploy` command, as well as any other environment files that are relevant to your deployment:
$ openstack overcloud deploy --templates -e glance-settings.yaml

For more information about using environment files, see the Including Environment Files in Overcloud Creation section in the Advanced Overcloud Customization Guide.

2.2. CONTROLLING IMAGE WEB-IMPORT SOURCES

You can limit the sources of web-import image downloads by adding URI blocklists and allowlists to the optional glance-image-import.conf file.

You can allow or block image source URIs at three levels:

- scheme (allowed_schemes, disallowed_schemes)
- host (allowed_hosts, disallowed_hosts)
- port (allowed_ports, disallowed_ports)

If you specify both allowlist and blocklist at any level, the allowlist is honored and the blocklist is ignored.

The Image service (glance) applies the following decision logic to validate image source URIs:

1. The scheme is checked.
   a. Missing scheme: reject
   b. If there is an allowlist, and the scheme is not present in the allowlist: reject. Otherwise, skip C and continue on to 2.
   c. If there is a blocklist, and the scheme is present in the blocklist: reject.

2. The host name is checked.
   a. Missing host name: reject
   b. If there is an allowlist, and the host name is not present in the allowlist: reject. Otherwise, skip C and continue on to 3.
   c. If there is a blocklist, and the host name is present in the blocklist: reject.

3. If there is a port in the URI, the port is checked.
   a. If there is a allowlist, and the port is not present in the allowlist: reject. Otherwise, skip B and continue on to 4.
   b. If there is a blocklist, and the port is present in the blocklist: reject.

4. The URI is accepted as valid.

If you allow a scheme, either by adding it to an allowlist or by not adding it to a blocklist, any URI that uses the default port for that scheme by not including a port in the URI is allowed. If it does include a port in the URI, the URI is validated according to the default decision logic.

2.3. IMAGE IMPORT EXAMPLE

For example, the default port for FTP is 21. Because ftp is an allowlisted scheme, this URL is allowed:
For example, the default port for FTP is 21. Because `ftp` is an allowlisted scheme, this URL is allowed: `ftp://example.org/some/resource` But because 21 is not in the port allowlist, this URL to the same resource is rejected: `ftp://example.org:21/some/resource`

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

### 2.4. DEFAULT IMAGE IMPORT BLOCKLIST AND ALLOWLIST SETTINGS

The `glance-image-import.conf` file is an optional file that contains the following default options:

- **allowed_schemes** - `[http, https]`
- **disallowed_schemes** - empty list
- **allowed_hosts** - empty list
- **disallowed_hosts** - empty list
- **allowed_ports** - `[80, 443]`
- **disallowed_ports** - empty list

If you use the defaults, end users can access URIs by using only the `http` or `https` scheme. The only ports that users can specify are **80** and **443**. Users do not have to specify a port, but if they do, it must be either **80** or **443**.

You can find the `glance-image-import.conf` file in the `etc/` subdirectory of the Image service source code tree. Ensure that you are looking in the correct branch for your release of Red Hat OpenStack Platform.

### 2.5. INJECTING METADATA ON IMAGE IMPORT TO CONTROL WHERE VMS LAUNCH

End users can upload images to the Image service and use these images to launch VMs. These user-provided (non-admin) images must be launched on a specific set of compute nodes. The assignment of an instance to a compute node is controlled by image metadata properties.

The Image Property Injection plugin injects metadata properties to images during import. Specify the properties by editing the `[image_import_opts]` and `[inject_metadata_properties]` sections of the `glance-image-import.conf` file.

To enable the Image Property Injection plugin, add the following line to the `[image_import_opts]` section:

```plaintext
[image_import_opts]
image_import_plugins = [inject_image_metadata]
```
To limit the metadata injection to images provided by a certain set of users, set the `ignore_user_roles` parameter. For example, use the following configuration to inject one value for `property1` and two values for `property2` into images downloaded by any non-admin user.

```
[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
```

The parameter `ignore_user_roles` is a comma-separated list of the Identity service (keystone) roles that the plugin ignores. This means that if the user that makes the image import call has any of these roles, the plugin does not inject any properties into the image.

The parameter `inject` is a comma-separated list of properties and values that are injected into the image record for the imported image. Each property and value must be quoted and separated by a colon (`:`).

You can find the `glance-image-import.conf` file in the `etc/` subdirectory of the Image service source code tree. Ensure that you are looking in the correct branch for your release of Red Hat OpenStack Platform.
CHAPTER 3. IMAGE SERVICE WITH MULTIPLE STORES

The Red Hat OpenStack Platform Image service (glance) supports using multiple stores with distributed edge architecture so that you can have an image pool at every edge site. You can copy images between the central site, which is also known as the hub site, and the edge sites.

The image metadata contains the location of each copy. For example, an image present on two edge sites is exposed as a single UUID with three locations: the central site plus the two edge sites. This means you can have copies of image data that share a single UUID on many stores. For more information about locations, see Understanding the location of images.

With an RBD image pool at every edge site, you can boot VMs quickly by using Ceph RBD copy-on-write (COW) and snapshot layering technology. This means that you can boot VMs from volumes and have live migration. For more information about layering with Ceph RBD, see Ceph block device layering in the Block Device Guide.

3.1. REQUIREMENTS OF STORAGE EDGE ARCHITECTURE

- A copy of each image must exist in the Image service at the central location.
- Prior to creating an instance at an edge site, you must have a local copy of the image at that edge site.
- Images uploaded to an edge site must be copied to the central location before they can be copied to other edge sites.
- You must use raw images when deploying a DCN architecture with Ceph storage.
- RBD must be the storage driver for the Image, Compute and Block Storage services.
- For each site, you must assign the same value to the NovaComputeAvailabilityZone and CinderStorageAvailabilityZone parameters.

3.2. IMPORT AN IMAGE TO MULTIPLE STORES

Use the interoperable image import workflow to import image data into multiple Ceph Storage clusters. You can import images into the Image service that are available on the local file system or through a web server.

If you import an image from a web server, the image can be imported into multiple stores at once. If the image is not available on a web server, you can import the image from a local file system into the central store and then copy it to additional stores. For more information, see Copy an existing image to multiple stores.

IMPORTANT

Always store an image copy on the central site, even if there are no instances using the image at the central location. For more information about importing images into the Image service, see the Distributed compute node and storage deployment guide.

3.2.1. Manage image import failures

You can manage failures of the image import operation by using the --allow-failure parameter:
• If the value of the --allow-failure parameter to true, the image status becomes active after the first store successfully imports the data. This is the default setting. You can view a list of stores that failed to import the image data by using the os_glance_failed_import image property.

• If you set the value of the --allow-failure parameter to false, the image status only becomes active after all specified stores successfully import the data. Failure of any store to import the image data results in an image status of failed. The image is not imported into any of the specified stores.

3.2.2. Importing image data to multiple stores

Because the default setting of the --allow-failure parameter is true, you do not need to include the parameter in the command if it is acceptable for some stores to fail to import the image data.

NOTE

This procedure does not require all stores to successfully import the image data.

Procedure

1. Import image data to multiple, specified stores:

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

• Replace IMAGE-NAME with the name of the image you want to import.

• Replace URI with the URI of the image.

• Replace STORE1, STORE2, and STORE3 with the names of the stores to which you want to import the image data.

• Alternatively, replace --stores with --all-stores true to upload the image to all the stores.

NOTE

The glance image-create-via-import command, which automatically converts the QCOW2 image to RAW format, works only with the web-download method. The glance-direct method is available, but it works only in deployments with a configured shared file system.

3.2.3. Importing image data to multiple stores without failure

This procedure requires all stores to successfully import the image data.

Procedure

1. Import image data to multiple, specified stores:

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

- Replace IMAGE-NAME with the name of the image you want to import.
- Replace URI with the URI of the image.
- Replace STORE1, STORE2, and STORE3 with the names of stores to which you want to copy the image data.
- Alternatively, replace --stores with --all-stores true to upload the image to all the stores.

NOTE

With the --allow-failure parameter set to false, the Image service does not ignore stores that fail to import the image data. You can view the list of failed stores with the image property os_glance_failed_import. For more information see Checking the progress of image import operation.

2. Verify that the image data was added to specific stores:

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

Replace IMAGE-ID with the ID of the original existing image.

The output displays a comma-delimited list of stores.

### 3.2.4. Importing image data to a single store

You can import image data to a single store.

**Procedure**

1. Import image data to a single store:

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

- Replace IMAGE-NAME with the name of the image you want to import.
- Replace URI with the URI of the image.
- Replace STORE with the name of the store to which you want to copy the image data.
If you do not include the options of --stores, --all-stores, or --store in the command, the Image service creates the image in the central store.

2. Verify that the image data was added to specific store:

   $ glance image-show IMAGE-ID | grep stores

   Replace IMAGE-ID with the ID of the original existing image.

   The output displays a comma-delimited list of stores.

3.2.5. Checking the progress of the image import operation

The interoperable image import workflow sequentially imports image data into stores. The size of the image, the number of stores, and the network speed between the central site and the edge sites impact how long it takes for the image import operation to complete.

You can follow the progress of the image import by looking at two image properties, which appear in notifications sent during the image import operation:

- The `os_glance_importing_to_stores` property lists the stores that have not imported the image data. At the beginning of the import, all requested stores show up in the list. Each time a store successfully imports the image data, the Image service removes the store from the list.

- The `os_glance_failed_import` property lists the stores that fail to import the image data. This list is empty at the beginning of the image import operation.

### Procedure

1. Verify that the image data was added to specific stores:

   $ glance image-show IMAGE-ID

   Replace IMAGE-ID with the ID of the original existing image.

   The output displays a comma-delimited list of stores similar to the following example snippet:

<table>
<thead>
<tr>
<th>os_glance_failed_import</th>
<th>os_glance_importing_to_stores</th>
<th>status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>central,dcn0,dcn1</td>
<td>importing</td>
</tr>
</tbody>
</table>

2. Monitor the status of the image import operation. When you precede a command with `watch`, the command output refreshes every two seconds.

   $ watch glance image-show IMAGE-ID

   Replace IMAGE-ID with the ID of the original existing image.

**NOTE**

In the following procedure, the environment has three Ceph Storage clusters: the central store and two stores at the edge, dcn0 and dcn1.
The status of the operation changes as the image import operation progresses:

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

Output that shows that an image failed to import resembles the following example:

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

After the operation completes, the status changes to active:

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

3.3. COPY AN EXISTING IMAGE TO MULTIPLE STORES

This feature enables you to copy existing images using Red Hat OpenStack Image service (glance) image data into multiple Ceph Storage stores at the edge by using the interoperable image import workflow.

NOTE

The image must be present at the central site before you copy it to any edge sites. Only the image owner or administrator can copy existing images to newly added stores.

You can copy existing image data either by setting `--all-stores` to `true` or by specifying specific stores to receive the image data.

- The default setting for the `--all-stores` option is `false`. If `--all-stores` is `false`, you must specify which stores receive the image data by using `--stores STORE1,STORE2`. If the image data is already present in any of the specified stores, the request fails.

- If you set `all-stores` to `true`, and the image data already exists in some of the stores, then those stores are excluded from the list.

After you specify which stores receive the image data, the Image service copies data from the central site to a staging area. Then the Image service imports the image data by using the interoperable image import workflow. For more information, see Importing an image to multiple stores.

IMPORTANT

Red Hat recommends that administrators carefully avoid closely timed image copy requests. Two closely timed copy-image operations for the same image causes race conditions and unexpected results. Existing image data remains as it is, but copying data to new stores fails.

3.3.1. Copying an image to all stores

Use the following procedure to copy image data to all available stores.
Procedure

1. Copy image data to all available stores:

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

   Replace IMAGE-ID with the name of the image you want to copy.

2. Confirm that the image data successfully replicated to all available stores:

   $ glance image-list --include-stores

   For information about how to check the status of the image import operation, see Checking the progress of the image import operation.

### 3.3.2. Copying an image to specific stores

Use the following procedure to copy image data to specific stores.

Procedure

1. Copy image data to specific stores:

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

   - Replace IMAGE-ID with the name of the image you want to copy.
   - Replace STORE1 and STORE2 with the names of the stores to which you want to copy the image data.

2. Confirm that the image data successfully replicated to the specified stores:

   $ glance image-list --include-stores

   For information about how to check the status of the image import operation, see Checking the progress of the image import operation.

### 3.4. DELETING AN IMAGE FROM A SPECIFIC STORE

This feature enables you to delete an existing image copy on a specific store using Red Hat OpenStack Image service (glance).

Procedure

Delete an image from a specific store:

$ glance stores-delete --store _STORE_ID_ _IMAGE_ID_

- Replace _STORE_ID_ with the name of the store on which the image copy should be deleted.
3.5. UNDERSTANDING THE LOCATIONS OF IMAGES

Although an image can be present on multiple sites, there is only a single UUID for a given image. The image metadata contains the locations of each copy. For example, an image present on two edge sites is exposed as a single UUID with three locations: the central site plus the two edge sites.

Procedure

1. Show the sites on which a copy of the image exists:

   $ glance image-show ID | grep "stores"

   | stores | default_backend,dcn1,dcn2

   In the example, the image is present on the central site, the default_backend, and on the two edge sites dcn1 and dcn2.

2. Alternatively, you can run the glance image-list command with the --include-stores option to see the sites where the images exist:

   $ glance image-list --include-stores

   | ID                                   | Name    | Stores               
   | 2bd882e7-1da0-4078-97fe-f1bb81f61b00 | cirros  | default_backend,dcn1,dcn2

3. List the image locations properties to show the details of each location:

   $ openstack image show ID -c properties

   | properties |
   | (--- cut ---) locations=['rbd://79b70c32-df46-4741-93c0-8118ae2ae284/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap', 'metadata': {'store': 'default_backend'}], ['rbd://63df2767-8dd6-4e06-818e-8c155334f487/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap', 'metadata': {'store': 'dcn1'}], ['rbd://1b324138-2ef9-4ef9-bd9e-aa7e6d6ead78/images/2bd882e7-1da0-4078-97fe-f1bb81f61b00/snap', 'metadata': {'store': 'dcn2'}]], (--- cut --)

   The image properties show the different Ceph RBD URIs for the location of each image.
In the example, the central image location URI is:

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

The URI is composed of the following data:

- **79b70c32-df46-4741-93c0-8118ae2ae284** corresponds to the central Ceph FSID. Each Ceph cluster has a unique FSID.

- The default value for all sites is **images**, which corresponds to the Ceph pool on which the images are stored.

- **2bd882e7-1da0-4078-97fe-f1bb81f61b00** corresponds to the image UUID. The UUID is the same for a given image regardless of its location.

- The metadata shows the glance store to which this location maps. In this example, it maps to the **default_backend**, which is the central hub site.
APPENDIX A. IMAGE CONFIGURATION PARAMETERS

The following keys can be used with the property option for both the glance image-update and glance image-create commands.

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

Table A.1. Property keys

<table>
<thead>
<tr>
<th>Specific to</th>
<th>Key</th>
<th>Description</th>
<th>Supported values</th>
</tr>
</thead>
</table>
| All        | architecture | The CPU architecture that must be supported by the hypervisor. For example, x86_64, arm, or ppc64. Run `uname -m` to get the architecture of a machine. | • alpha - DEC 64-bit RISC  
<pre><code>         |        |                                                                              | • armv7l - ARM Cortex-A7 MPCore |
</code></pre>
<p>|            |        |                                                                              | • cris - Ethernet, Token Ring, AXis-Code Reduced Instruction Set |
|            |        |                                                                              | • i686 - Intel sixth-generation x86 (P6 micro architecture)    |
|            |        |                                                                              | • ia64 - Itanium         |
|            |        |                                                                              | • lm32 - Lattice Micro32 |
|            |        |                                                                              | • m68k - Motorola 68000  |
|            |        |                                                                              | • microblaze - Xilinx 32-bit FPGA (Big Endian)     |
|            |        |                                                                              | • microblazeel - Xilinx 32-bit FPGA (Little Endian)         |
|            |        |                                                                              | • mips - MIPS 32-bit RISC (Big Endian)        |
|            |        |                                                                              | • mipsel - MIPS 32-bit RISC (Little Endian)         |
|            |        |                                                                              | • mips64 - MIPS 64-bit RISC (Big Endian)      |
|            |        |                                                                              | • mips64el - MIPS 64-bit RISC (Little Endian)        |
|            |        |                                                                              | • openrisc - OpenCores RISC         |
|            |        |                                                                              | • parisc - HP Precision Architecture RISC          |
|            |        |                                                                              | • parisc64 - HP Precision Architecture 64-bit RISC  |
|            |        |                                                                              | • ppc - PowerPC 32-bit        |
|            |        |                                                                              | • ppc64 - PowerPC 64-bit       |</p>
<table>
<thead>
<tr>
<th>Specific to</th>
<th>Key</th>
<th>Description</th>
<th>Supported values</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>hypervisor_type</td>
<td>The hypervisor type.</td>
<td>kvm, vmware</td>
</tr>
<tr>
<td>All</td>
<td>instance_uuid</td>
<td>For snapshot images, this is the UUID of the server used to create this image.</td>
<td>Valid server UUID</td>
</tr>
<tr>
<td>All</td>
<td>kernel_id</td>
<td>The ID of an image stored in the Image Service that should be used as the kernel when booting an AMI-style image.</td>
<td>Valid image ID</td>
</tr>
<tr>
<td>All</td>
<td>os_distro</td>
<td>The common name of the operating system distribution in lowercase.</td>
<td></td>
</tr>
</tbody>
</table>

### Supported values

- **ppcemb** - PowerPC (Embedded 32-bit)
- **s390** - IBM Enterprise Systems Architecture/390
- **s390x** - S/390 64-bit
- **sh4** - SuperH SH-4 (LittleEndian)
- **sh4eb** - SuperH SH-4 (BigEndian)
- **sparc** - Scalable Processor Architecture, 32-bit
- **sparc64** - Scalable Processor Architecture, 64-bit
- **unicore32** - Microprocessor Research and Development Center RISC Unicore32
- **x86_64** - 64-bit extension of IA-32
- **xtensa** - Tensilica Xtensa configurable microprocessor core
- **xtensaeb** - Tensilica Xtensa configurable microprocessor core (BigEndian)
<table>
<thead>
<tr>
<th>Specific to</th>
<th>Key</th>
<th>Description</th>
<th>Supported values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>debian</td>
<td>Debian. Do not use Debian, org.debian, or org.debian</td>
<td></td>
</tr>
<tr>
<td></td>
<td>fedora</td>
<td>Fedora. Do not use Fedora, org.fedora, or org.fedoraproject</td>
<td></td>
</tr>
<tr>
<td></td>
<td>freebsd</td>
<td>FreeBSD. Do not use org.freebsd, freeBSD, or FreeBSD</td>
<td></td>
</tr>
<tr>
<td></td>
<td>gentoo</td>
<td>Gentoo Linux. Do not use Gentoo or org.gentoo.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mandrake</td>
<td>Mandrakelinux (MandrakeSoft) distribution. Do not use mandrakelinux or MandrakeLinux.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mandriva</td>
<td>Mandriva Linux. Do not use mandrivalinux.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>mes</td>
<td>Mandriva Enterprise Server. Do not use mandrivaent or mandrivaES.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>msdos</td>
<td>Microsoft Disc Operating System. Do not use ms-dos.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>netbsd</td>
<td>NetBSD. Do not use NetBSD or org.netbsd.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>netware</td>
<td>Novell NetWare. Do not use novell or NetWare.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>opensbsd</td>
<td>OpenBSD. Do not use OpenBSD or org.openbsd.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>opensolaris</td>
<td>OpenSolaris. Do not use OpenSolaris or org.opensolaris.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>opensuse</td>
<td>openSUSE. Do not use suse, SuSE, or org.opensuse.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>rhel</td>
<td>Red Hat Enterprise Linux. Do not use redhat, RedHat, or com.redhat.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sled</td>
<td>SUSE Linux Enterprise Desktop. Do not use com.suse.</td>
<td></td>
</tr>
</tbody>
</table>
### APPENDIX A. IMAGE CONFIGURATION PARAMETERS

<table>
<thead>
<tr>
<th>Specific to</th>
<th>Key</th>
<th>Description</th>
<th>Supported values</th>
</tr>
</thead>
<tbody>
<tr>
<td>All</td>
<td>os_version</td>
<td>The operating system version as specified by the distributor.</td>
<td>Version number (for example, &quot;11.10&quot;)</td>
</tr>
<tr>
<td>All</td>
<td>ramdisk_id</td>
<td>The ID of image stored in the Image Service that should be used as the ramdisk when booting an AMI-style image.</td>
<td>Valid image ID</td>
</tr>
<tr>
<td>All</td>
<td>vm_mode</td>
<td>The virtual machine mode. This represents the host/guest ABI (application binary interface) used for the virtual machine.</td>
<td>hvm: Fully virtualized. This is the mode used by QEMU and KVM.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_disk_bus</td>
<td>Specifies the type of disk controller to attach disk devices to.</td>
<td>scsi, virtio, ide, or usb. Note that if using iscsi, the hw_scsi_model needs to be set to virtio-scsi.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_cdrom_bus</td>
<td>Specifies the type of disk controller to attach CD-ROM devices to.</td>
<td>scsi, virtio, ide, or usb. If you specify iscsi, you must set the hw_scsi_model parameter to virtio-scsi.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_numa_nodes</td>
<td>Number of NUMA nodes to expose to the instance (does not override flavor definition).</td>
<td>Integer.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_numa_cpus.0</td>
<td>Mapping of vCPUs N-M to NUMA node 0 (does not override flavor definition).</td>
<td>Comma-separated list of integers.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_numa_cpus.1</td>
<td>Mapping of vCPUs N-M to NUMA node 1 (does not override flavor definition).</td>
<td>Comma-separated list of integers.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_numa_mem.0</td>
<td>Mapping N MB of RAM to NUMA node 0 (does not override flavor definition).</td>
<td>Integer</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_numa_mem.1</td>
<td>Mapping N MB of RAM to NUMA node 1 (does not override flavor definition).</td>
<td>Integer</td>
</tr>
<tr>
<td>Specific to</td>
<td>Key</td>
<td>Description</td>
<td>Supported values</td>
</tr>
<tr>
<td>----------------</td>
<td>--------------------------</td>
<td>-----------------------------------------------------------------------------</td>
<td>------------------</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_pci_numa_affinity_policy</td>
<td>Specifies the NUMA affinity policy for PCI passthrough devices and SR-IOV interfaces.</td>
<td>Set to one of the following valid values:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- <strong>required</strong>: The Compute service creates an instance that requests a PCI device only when at least one of the NUMA nodes of the instance has affinity with the PCI device. This option provides the best performance.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- <strong>preferred</strong>: The Compute service attempts a best effort selection of PCI devices based on NUMA affinity. If affinity is not possible, then the Compute service schedules the instance on a NUMA node that has no affinity with the PCI device.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- <strong>legacy</strong>: (Default) The Compute service creates instances that request a PCI device in one of the following cases:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The PCI device has affinity with at least one of the NUMA nodes.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>- The PCI devices do not provide information about their NUMA affinities.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_qemu_guest_agent</td>
<td>Guest agent support. If set to <strong>yes</strong>, and if qemu-ga is also installed, file systems can be quiesced (frozen) and snapshots created automatically.</td>
<td><strong>yes / no</strong></td>
</tr>
<tr>
<td>Specific to</td>
<td>Key</td>
<td>Description</td>
<td>Supported values</td>
</tr>
<tr>
<td>---------------------</td>
<td>--------------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_rng_mode</td>
<td>Adds a random number generator (RNG) device to instances launched with this image. The instance flavor enables the RNG device by default. To disable the RNG device, the cloud administrator must set <code>hw_rng:allowed</code> to <code>False</code> on the flavor. The default entropy source is <code>/dev/random</code>. To specify a hardware RNG device, set <code>rng_dev_path</code> to <code>/dev/hwrng</code> in your Compute environment file.</td>
<td>virtio, or other supported device.</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>hw_scsi_mod</td>
<td>Enables the use of VirtIO SCSI (virtio-scsi) to provide block device access for compute instances; by default, instances use VirtIO Block (virtio-blk). VirtIO SCSI is a para-virtualized SCSI controller device that provides improved scalability and performance, and supports advanced SCSI hardware.</td>
<td>virtio-scsi</td>
</tr>
</tbody>
</table>
The video device driver for the display device to use in virtual machine instances. Set to one of the following values to specify the supported driver to use:

- **virtio** - Recommended Driver for the virtual machine display device, supported by most architectures. The VirtIO GPU driver is included in RHEL-7 and later, and Linux kernel versions 4.4 and later. If an instance kernel has the VirtIO GPU driver, then the instance can use all the VirtIO GPU features. If an instance kernel does not have the VirtIO GPU driver, the VirtIO GPU device gracefully falls back to VGA compatibility mode, which provides a working display for the instance.

- **qxl** - Deprecated Driver for Spice or noVNC environments that is no longer maintained.

- **cirrus** - Legacy driver.

- **vga** - Use this driver for IBM Power environments.

- **gop** - Not supported for QEMU/KVM environments.

- **xen** - Not supported for KVM environments.

- **vmvga** - Legacy driver, do not use.

- **none** - Use this value to disable emulated graphics or video in virtual GPU (vGPU) instances where the driver is configured separately.

Maximum RAM for the video image. Used only if a `hw_video:ram_max_m` value has been set in the flavor's `extra_specs` and that value is higher than the value set in `hw_video_ram`.

Integer in MB (for example, 64)
<table>
<thead>
<tr>
<th>Specific to</th>
<th>Key</th>
<th>Description</th>
<th>Supported values</th>
</tr>
</thead>
</table>
| libvirt API driver  | hw_watchdog_action          | Enables a virtual hardware watchdog device that carries out the specified action if the server hangs. The watchdog uses the i6300esb device (emulating a PCI Intel 6300ESB). If `hw_watchdog_action` is not specified, the watchdog is disabled. | - disabled—The device is not attached. Allows the user to disable the watchdog for the image, even if it has been enabled using the image’s flavor. The default value for this parameter is disabled.  
  - reset—Forcefully reset the guest.  
  - poweroff—Forcefully power off the guest.  
  - pause—Pause the guest.  
  - none—Only enable the watchdog; do nothing if the server hangs. |
| libvirt API driver  | os_command_line             | The kernel command line to be used by the libvirt driver, instead of the default. For Linux Containers (LXC), the value is used as arguments for initialization. This key is valid only for Amazon kernel, ramdisk, or machine images (aki, ari, or ami). |                                                                                                       |
| libvirt API driver  | hw_vif_model                | Specifies the model of virtual network interface device to use.                                                                             | The valid options depend on the configured hypervisor.  
  - KVM and QEMU: e1000, ne2k_pci, pcnet, rtl8139, and virtio.  
  - VMware: e1000, e1000e, VirtualE1000, VirtualE1000e, VirtualPCNet32, VirtualSriovEthernetCard, and VirtualVmxnet.  
  - Xen: e1000, netfront, ne2k_pci, pcnet, and rtl8139. |
<p>| VMware API driver   | vmware_adaptertype          | The virtual SCSI or IDE controller used by the hypervisor.                                                                                   | <strong>LiLogic, busLogic, or ide</strong>                                                                                         |</p>
<table>
<thead>
<tr>
<th>Specific to</th>
<th>Key</th>
<th>Description</th>
<th>Supported values</th>
</tr>
</thead>
<tbody>
<tr>
<td>VMware API driver</td>
<td>vmware_ostype</td>
<td>A VMware GuestID which describes the operating system installed in the image. This value is passed to the hypervisor when creating a virtual machine. If not specified, the key defaults to otherGuest.</td>
<td>For more information, see Images with VMware vSphere.</td>
</tr>
<tr>
<td>VMware API driver</td>
<td>vmware_image_version</td>
<td>Currently unused.</td>
<td>1</td>
</tr>
<tr>
<td>XenAPI driver</td>
<td>auto_disk_config</td>
<td>If true, the root partition on the disk is automatically resized before the instance boots. This value is only taken into account by the Compute service when using a Xen-based hypervisor with the XenAPI driver. The Compute service will only attempt to resize if there is a single partition on the image, and only if the partition is in ext3 or ext4 format.</td>
<td>true / false</td>
</tr>
<tr>
<td>libvirt API driver</td>
<td>os_type</td>
<td>The operating system installed on the image. The XenAPI driver contains logic that takes different actions depending on the value of the os_type parameter of the image. For example, for os_type=windows images, it creates a FAT32-based swap partition instead of a Linux swap partition, and it limits the injected host name to less than 16 characters.</td>
<td>linux or windows</td>
</tr>
</tbody>
</table>