Supplement to Administrator Guide and End User Guide

SUSE OpenStack Cloud 7

Supplement to Administrator Guide and End User Guide

SUSE OpenStack Cloud 7

by Frank Sundermeyer and Tanja Roth

Publication Date: September 02, 2019

SUSE LLC
10 Canal Park Drive
Suite 200
Cambridge MA 02141
USA
https://www.suse.com/documentation

Contents

About This Guide

SUSE® OpenStack Cloud is an open source software solution that provides the fundamental capabilities to deploy and manage a cloud infrastructure based on SUSE Linux Enterprise. SUSE OpenStack Cloud is powered by OpenStack, the leading community-driven, open source cloud infrastructure project. It seamlessly manages and provisions workloads across a heterogeneous cloud environment in a secure, compliant, and fully-supported manner. The product tightly integrates with other SUSE technologies and with the SUSE maintenance and support infrastructure.

This guide is a supplement to the SUSE OpenStack Cloud *Administrator Guide* and SUSE OpenStack Cloud *End User Guide*. It contains additional information for admins and end users that is specific to SUSE OpenStack Cloud.

Many chapters in this manual contain links to additional documentation resources. These include additional documentation that is available on the system and documentation available on the Internet.

For an overview of the documentation available for your product and the latest documentation updates, refer to http://www.suse.com/documentation.

1 Available Documentation

Note: Online Documentation and Latest Updates

Documentation for our products is available at http://www.suse.com/documentation/, where you can also find the latest updates, and browse or download the documentation in various formats.

In addition, the product documentation is usually available in your installed system under `/usr/share/doc/manual`. You can also access the product-specific manuals and the upstream documentation from the *Help* links in the graphical Web interfaces.

The following documentation is available for this product:

Deployment Guide

Gives an introduction to the SUSE® OpenStack Cloud architecture, lists the requirements, and describes how to set up, deploy, and maintain the individual components. Also contains information about troubleshooting, support, and a glossary listing the most important terms and concepts for SUSE OpenStack Cloud.

Administrator Guide

Introduces the OpenStack services and their components.

Also guides you through tasks like managing images, roles, instances, flavors, volumes, shares, quotas, host aggregates, and viewing cloud resources. To complete these tasks, use either the graphical Web interface (OpenStack Dashboard, code name `Horizon`) or the OpenStack command line clients.

End User Guide

Describes how to manage images, instances, networks, object containers, volumes, shares, stacks, and databases. To complete these tasks, use either the graphical Web interface (OpenStack Dashboard, code name `Horizon`) or the OpenStack command line clients.

Supplement to Administrator Guide and End User Guide

A supplement to the SUSE OpenStack Cloud *Administrator Guide* and SUSE OpenStack Cloud *End User Guide*. It contains additional information for admins and end users that is specific to SUSE OpenStack Cloud.

Overview

A manual introducing SUSE OpenStack Cloud Monitoring. It is written for everybody interested in SUSE OpenStack Cloud Monitoring.

OpenStack Operator's Guide

A manual for SUSE OpenStack Cloud operators describing how to prepare their OpenStack platform for SUSE OpenStack Cloud Monitoring. The manual also describes how the operators use SUSE OpenStack Cloud Monitoring for monitoring their OpenStack services.

Monitoring Service Operator's Guide

A manual for system operators describing how to operate SUSE OpenStack Cloud Monitoring. The manual also describes how the operators can use SUSE OpenStack Cloud Monitoring for monitoring their environment.

2 Feedback

Several feedback channels are available:

Bugs and Enhancement Requests

To report bugs for a product component, or to submit enhancement requests, use https://bugzilla.suse.com/. For documentation bugs, submit a bug report for the component *Documentation* of the respective product.

If you are new to Bugzilla, you might find the following articles helpful:

- http://en.opensuse.org/openSUSE:Submitting_bug_reports
- http://en.opensuse.org/openSUSE:Bug_reporting_FAQ

Services and Support Options

For services and support options available for your product, refer to http://www.suse.com/support/.

User Comments/Bug Reports

We want to hear your comments about and suggestions for this manual and the other documentation included with this product. If you are reading the HTML version of this guide, use the Comments feature at the bottom of each page in the online documentation at http://www.suse.com/documentation/.

If you are reading the single-page HTML version of this guide, you can use the *Report Bug* link next to each section to open a bug report at https://bugzilla.suse.com/. A user account is needed for this.

Mail

For feedback on the documentation of this product, you can also send a mail to `doc-team@suse.com`. Make sure to include the document title, the product version, and the publication date of the documentation. To report errors or suggest enhancements, provide a concise description of the problem and refer to the respective section number and page (or URL).

3 Documentation Conventions

The following notices and typographical conventions are used in this documentation:

- **Warning**

  Vital information you must be aware of before proceeding. Warns you about security issues, potential loss of data, damage to hardware, or physical hazards.

Important

Important information you should be aware of before proceeding.

Note

Additional information, for example about differences in software versions.

Tip

Helpful information, like a guideline or a piece of practical advice.

- `tux > command`

  Commands that can be run by any user, including the `root` user.

- `root # command`

  Commands that must be run with `root` privileges. Often you can also prefix these commands with the **`sudo`** command to run them.

- `/etc/passwd`: directory names and file names
- *`PLACEHOLDER`*: replace *`PLACEHOLDER`* with the actual value
- `PATH`: the environment variable PATH
- **`ls`**, `--help`: commands, options, and parameters
- `user`: users or groups
- `Alt`, `Alt`–`F1`: a key to press or a key combination; keys are shown in uppercase as on a keyboard
- *File*, *File* › *Save As*: menu items, buttons
- `x86_64` This paragraph is only relevant for the AMD64/Intel 64 architecture. The arrows mark the beginning and the end of the text block.

  `System z, POWER` This paragraph is only relevant for the architectures `z Systems` and `POWER`. The arrows mark the beginning and the end of the text block.

- *Dancing Penguins* (Chapter *Penguins*, ↑Another Manual): This is a reference to a chapter in another manual.

4 About the Making of This Manual

This documentation is written in SUSEDoc, a subset of DocBook 5 (http://www.docbook.org). The XML source files were validated by **jing**, processed by **xsltproc**, and converted into XSL-FO using a customized version of Norman Walsh's stylesheets. The final PDF is formatted through FOP from Apache Software Foundation. The open source tools and the environment used to build this documentation are provided by the DocBook Authoring and Publishing Suite (DAPS). The project's home page can be found at https://github.com/openSUSE/daps.

The XML source code of this documentation can be found at https://github.com/SUSE-Cloud/doc-cloud.

1 Changing the SUSE OpenStack Cloud Dashboard Theme

The SUSE OpenStack Cloud Dashboard theme can now be customized. The default SUSE OpenStack Cloud theme is available in the `openstack-dashboard-theme-SUSE` package. If you want to replace it with a custom theme, you can explore the package contents as an example. When using a custom theme, make sure the resulting package name starts with `openstack-dashboard-theme-`. Apart from that, go to the Horizon barclamp in Crowbar, switch to the *Raw* view and adjust the *site_theme* attribute accordingly. For details about the barclamps in Crowbar, see the *Deployment Guide*.

2 Managing Images

In the SUSE OpenStack Cloud context, images are virtual disk images that represent the contents and structure of a storage medium or device, such as a hard disk, in a single file. Images are used as a template from which a virtual machine can be started. For starting a virtual machine, SUSE OpenStack Cloud always uses a copy of the image.

Images have both content and metadata. The latter are also called image properties.

Permissions to manage images are defined by the cloud operator during setup of SUSE OpenStack Cloud. Image upload and management may be restricted to cloud administrators or cloud operators only.

Managing images for SUSE OpenStack Cloud requires the following basic steps:

1. *Building Images with SUSE Studio*.
   For general and hypervisor-specific requirements, refer to *Section 2.1, "Image Requirements"*.
2. *Uploading Disk Images to SUSE OpenStack Cloud*.
   Images can either be uploaded to SUSE OpenStack Cloud with the unified **`python-openstackclient`** from command line or with the SUSE OpenStack Cloud Dashboard. As the Dashboard has some limitations regarding image upload and modification of properties, it is recommended to use the unified **`python-openstackclient`** for comprehensive image management.
3. Specifying Image Properties. You can do so during image upload (using **`openstack image create`**) or with **`openstack image set`** after the image has already been uploaded. Refer to *Procedure 2.2, "Uploading Disk Images to SUSE OpenStack Cloud"* and *Procedure 2.3, "Modifying Image Properties"*.

Important: Properties for Architecture, Hypervisor and VM Mode

OpenStack Image does *not* check nor automatically detect any image properties. Therefore you need to specify the image's properties manually.

This is especially important when using mixed virtualization environments to make sure that an image is only launched on appropriate hypervisors. The properties can specify a certain architecture, hypervisor type, or application binary interface (ABI) that the image requires.

2.1 Image Requirements

To build the images to use within the cloud, use SUSE Studio or SUSE Studio Onsite as they provide automatic insertion of management scripts and agents. Make sure any images that you build for SUSE OpenStack Cloud fulfill the following requirements.

2.1.1 General Image Requirements

- The network is set to DHCP.
- The image does not include YaST2 Firstboot.
- The image does not include any end-user license agreement (EULA) dialogs.
- The image contains the `cloud-init` package. The package will be automatically added to the image if the following check box in SUSE Studio or SUSE Studio Onsite is enabled: *Integrate with SUSE OpenStack Cloud/OpenStack*.
  The `cloud-init` package contains tools used for communication with the instance metadata API, which is provided by Compute. The metadata API is only accessible from inside the VM. The package is needed to pull keypairs into the virtual machine that will run the image.

If you intend to manage the image by the Orchestration module, you also need to include the following package: `openstack-heat-cfntools` (part of the SUSE OpenStack Cloud ISO).

2.1.2 Image Requirements Depending on Hypervisor

For a list of supported VM guests, refer to the SUSE® Linux Enterprise Server *Virtualization Guide*, section *Supported VM Guests*. It is available at https://www.suse.com/documentation/sles-12/book_virt/data/virt_support_guests.html.

Depending on the virtualization platform on which you want to use the image, make sure the image also fulfills the following requirements.

KVM

Appliance format: If you are using SUSE Studio or SUSE Studio Onsite 1.3 to build images, use the `SUSE OpenStack Cloud/OpenStack/KVM (.qcow2)` format.

Xen

Appliance format: If you are using SUSE Studio or SUSE Studio Onsite 1.3 to build images, use the `SUSE OpenStack Cloud/OpenStack/KVM (.qcow2)` format.

VMware

Appliance format: If you are using SUSE Studio or SUSE Studio Onsite 1.3 to build images, use the `VMware/VirtualBox/KVM (.vmdk)` format.

If you are using SUSE Studio or SUSE Studio Onsite to build images, the resulting image will be a monolithic `sparse` file.

Note: Image Conversion

Sparse images can be uploaded to OpenStack Image. However, it is recommended to convert sparse images into a different format before uploading them to OpenStack Image (because starting VMs from sparse images may take longer).

For a list of supported image types, refer to http://docs.openstack.org/liberty/config-reference/content/vmware.html, section *Supported image types*.

For details on how to convert a sparse image into different formats, refer to http://docs.openstack.org/liberty/config-reference/content/vmware.html, section *Optimize images*.

2.1.3 Images for Use With Multiple Hypervisors

If you build the images for SUSE OpenStack Cloud in SUSE Studio or SUSE Studio Onsite, they are compatible with multiple hypervisors by default—even if you may need to convert the resulting image formats before uploading them to OpenStack Image. See *Procedure 2.1, "Converting Disk Images to Different Formats"* for details.

If your image is not made in SUSE Studio or SUSE Studio Onsite, configure the image as follows to create an image that can be booted on KVM and Xen, for example:

/etc/sysconfig/kernel

```
INITRD_MODULES="virtio_blk virtio_pci ata_piix ata_generic hv_storvsc"
```

/boot/grub/menu.lst

To name the partition that should be booted, use:

```
root=UUID=...
```

To find the respective UUID value to use, execute the following command:

```
tune2fs -l /dev/sda2|grep UUID
```

/etc/fstab

Do not use device names (`/dev/...`), but `UUID=...` or `LABEL=root` entries. For the latter, add the label `root` to the root file system of your image (in this case, `/dev/sda2`):

```
tune2fs -L root /dev/sda2
```

Disk Format

Use `*.qcow2` as disk format for your image.

Image Properties in OpenStack Image

To upload the image to SUSE OpenStack Cloud only once and to use the same image for KVM and Xen, specify the following image options during or after upload:

```
--public --container-format bare \
--property architecture=x86_64 \
--property vm_mode=hvm \
--disk-format qcow2
```

2.2 Building Images with SUSE Studio

When creating an appliance for SUSE OpenStack Cloud the following steps are essential:

1. In SUSE Studio or SUSE Studio Onsite, switch to the *Configuration* › *Appliance* tab.
2. Enable the *Integrate with SUSE OpenStack Cloud/OpenStack* check box.
3. On the *Build* tab, choose the respective appliance format. It mainly depends on the hypervisor on which you want to use the image—see *Section 2.1.2, "Image Requirements Depending on Hypervisor"*.

For more detailed information on how to build appliance images, refer to the SUSE Studio Onsite documentation, available at http://www.suse.com/documentation/suse_studio/.

2.3 Image Properties

Images have both contents and metadata. The latter are also called properties. The following properties can be attached to an image in SUSE OpenStack Cloud. Set them from the command line when uploading or modifying images. For a list of image properties, see http://docs.openstack.org/developer/python-openstackclient/command-objects/image.html.

2.4 Uploading Images

If you have created an image for `SUSE OpenStack Cloud/OpenStack/KVM` with SUSE Studio or with SUSE Studio Onsite 1.3, you can upload the image directly as described in *Procedure 2.2, "Uploading Disk Images to SUSE OpenStack Cloud"*.

PROCEDURE 2.1: CONVERTING DISK IMAGES TO DIFFERENT FORMATS

1. Make sure the `virt-utils` package is installed on the machine used for conversion.
2. Download the image from SUSE Studio.
3. To convert `qcow2` to `vhd` images, use the following command:

   ```
   qemu-img convert -O vpc CURRENT_IMAGE_FILE FINAL_IMAGE_FILE.vhd
   ```

PROCEDURE 2.2: UPLOADING DISK IMAGES TO SUSE OPENSTACK CLOUD

Upload a disk image using the `python-openstackclient` client.

Images are owned by projects and can be `private` (accessible to members of the particular project only) or `public` (accessible to members of all projects). Private images can also be explicitly shared with other projects, so that members of those projects can access the images, too. Any image uploaded to OpenStack Image will get an `owner` attribute. By default, ownership is set to the primary project of the user that uploads the image.

Set or modify hypervisor-specific properties with the `--property key=value` option. This can be done directly during image upload (as shown in the examples below). To change the properties after image upload, refer to *Procedure 2.3, "Modifying Image Properties"*.

1. In a shell, source the OpenStack RC file for the project that you want to upload an image to. For details, refer to http://docs.openstack.org/user-guide/common/cli_set_environment_variables_using_openstack_rc.html.

2. Upload the image using **`openstack image create`**. Find some example commands for different hypervisors below:

   - For KVM:

     ```
     openstack image create IMAGE_NAME \
       --public --container-format bare \
       --property architecture=x86_64 \
       --property hypervisor_type=kvm \
       --disk-format qcow2 \
       --file PATH_TO_IMAGE_FILE.qcow2
     ```

   - For Xen:

     ```
     openstack image create IMAGE_NAME \
       --public --container-format bare \
       --property architecture=x86_64 \
       --property hypervisor_type=xen \
       --property vm_mode=xen \
       --disk-format qcow2 \
       --file PATH_TO_IMAGE_FILE.qcow2
     ```

     

     ### Note: Value of `vm_mode`

     For Xen PV image import, use `vm_mode=xen`, for Xen HVM image import use `vm_mode=hvm`.

   - For VMware:

     ```
     openstack image create IMAGE_NAME \
       --public --container-format bare \
       --property vmware_adaptertype="lsiLogic" \
       --property vmware_disktype="preallocated" \
       --property hypervisor_type=vmware \
       --disk-format vmdk --file PATH_TO_IMAGE_FILE.vmdk
     ```

Note: Value of `vmware_disktype`

Depending on which disk type you use, adjust the value of `vmware_disktype` accordingly. For an overview of which values to use, refer to http://docs.openstack.org/liberty/config-reference/content/vmware.html, *Table 2.8. OpenStack Image Service disk type settings*.

- For Docker:

  Find an image in the Docker registry you want to use and save it locally with **`docker pull`** *`IMAGE_NAME`*, where *`IMAGE_NAME`* is the name of the image in the Docker registry. The same name needs to be used when uploading the image with the following command:

```
docker save IMAGE_NAME | openstack image create \
--public --property hypervisor_type=docker \
--container-format=docker --disk-format=raw IMAGE_NAME
```

Important: Docker Images Need to Run a Long-Living Process

Docker instances will only be able to spawn successfully, when running a long-living process, for example `sshd`. Such a process can be configured with CMD (https://docs.docker.com/engine/reference/builder/#cmd) or ENTRYPOINT (https://docs.docker.com/engine/reference/builder/#entrypoint) in the `Docker`.

Alternatively, such a process can be specified on the command line with the image property `os_command_line`.

```
openstack image set --property os_command_line='/usr/sbin/sshd -D' \
       IMAGE_ID
```

If the image upload has been successful, a message appears, displaying the ID that has been assigned to the image.

Note: Updating Images

After having uploaded an image to SUSE OpenStack Cloud, the image contents cannot be modified—only the image's metadata, see *Procedure 2.3*. To update image contents, you need to delete the current image and upload a modified version of the image. You can also launch an instance from the respective image, change it, create a snapshot of the instance and use the snapshot as a new image.

2.5 Modifying Image Properties

Set or modify hypervisor-specific properties with the `--property key=value` option. This can be done directly during image upload (see *Procedure 2.2*) or after the image has been uploaded (as described below).

PROCEDURE 2.3: MODIFYING IMAGE PROPERTIES

1. In a shell, source the OpenStack RC file for the project that you want to upload an image to. For details, refer to http://docs.openstack.org/user-guide/common/cli_set_environment_variables_using_openstack_rc.html.

2. If you do not know the ID or the exact name of the image whose properties you want to modify, look it up with:

```
openstack image list
```

3. Use the **`openstack image set`** command to set the properties for architecture, hypervisor type, and virtual machine mode. In the following, find some examples with properties for different hypervisors:

   - For KVM:

```
openstacke image set IMAGE_ID_OR_NAME \
  --property architecture=x86_64 \
  --property hypervisor_type=kvm
```

   - For Xen:

```
openstack image set IMAGE_ID_OR_NAME \
  --property architecture=x86_64 \
  --property hypervisor_type=xen \
```

```
  --property vm_mode=xen
```

Note: Value of `vm_mode`

For Xen PV image import, use `vm_mode=xen`, for Xen HVM image import use `vm_mode=hvm`.

- For VMware:

```
openstack image update IMAGE_ID_OR_NAME \
  --property vmware_adaptertype="lsiLogic" \
  --property vmware_disktype="preallocated" \
  --property hypervisor_type=vmware
```

Note: Value of `vmware_disktype`

Depending on which disk type you use, adjust the value of `vmware_disktype` accordingly. For an overview of which values to use, refer to http://docs.openstack.org/liberty/config-reference/content/vmware.html, *Table 2.8. OpenStack Image Service disk type settings*.

For more information about the `architecture`, `hypervisor_type`, and `vm_mode` properties, refer to http://docs.openstack.org/image-guide/content/image-metadata.html.

2.6 Using a Custom Kernel to Boot Btrfs Images Under Xen

On Xen hosts, starting instances from an image that uses Btrfs as root file system may fail with SUSE OpenStack Cloud 7. As a work-around, boot the Btrfs image with a custom kernel to start the instances. Prepare the Btrfs image as described in *Procedure 2.4, "Booting Btrfs Images with a Custom Kernel"*.

PROCEDURE 2.4: BOOTING BTRFS IMAGES WITH A CUSTOM KERNEL

REQUIREMENTS

- The `python-openstackclient` is installed. After you have sourced an OpenStack RC file, use the command line client to upload images from a machine outside of the cloud.
- To run the `python-openstackclient`: An OpenStack RC file containing the credentials for the OpenStack project to which you want to upload the images.

1. Install the `grub2-xen` package. It provides the `grub.xen` file required to boot para-virtualized VMs that use the Btrfs file system:

```
root # zypper in grub2-x86_64-xen
```

2. Create a Glance image with the kernel from this package. For example:

```
openstack image create img-grub-xen-x64 \
 --file /usr/lib/grub2/x86_64-xen/grub.xen --public \
 --container-format bare --disk-format raw
```

3. Create a second image which uses Btrfs as root file system. For example:

```
openstack image create img-btrfs \
  --file openSUSE-Leap-42.1-JeOS-for-XEN.x86_64.qcow2 \
  --container-format bare --disk-format qcow2
```

4. Update the image named `img-btrfs` by adding a `kernel_id` property to it:

```
openstack image set 376c245d-24fe-41e2-8abd-655d4ed8da95 \
--property kernel_id=72ad3069-6003-4653-86f2-b5914ce33f66
```

   where `376c245d-24fe-41e2-8abd-655d4ed8da95` is the ID of the image named `img-btrfs` and `72ad3069-6003-4653-86f2-b5914ce33f66` is the ID of the image named `img-grub-xen-x64`

5. Boot the image to start the instance:

```
nova boot --flavor FLAVOR --image 376c245d-24fe-41e2-8abd-655d4ed8da95
```

   This results in a domain XML which contains the kernel you need:

```
<domain type='xen' id="2">
```

```
 <name>instance-00000003</name>
 <uuid>12b2ce2b-ba1d-4c14-847f-9476dbae7199</uuid>
 <memory unit='KiB'>524288</memory>
 <currentMemory unit='KiB'>524288</currentMemory>
 <vcpu placement='static'>1</vcpu>
 <bootloader></bootloader>
 <os>
   <type>linux</type>
   <kernel>/var/lib/nova/instances/12b2ce2b-ba1d-4c14-847f-9476dbae7199/kernel</
kernel>
   <cmdline>ro root=/dev/xvda</cmdline>
 </os>
```

2.7 Viewing Images and Image Properties, Deleting Images

In the following, find some examples on how to view images or image properties or how to remove images from OpenStack Image.

Listing Images

```
openstack image list
```

Lists ID, name, and status for all images in Image that the current user can access.

Showing Metadata for a Particular Image

```
openstack image show IMAGE_ID_OR_NAME
```

Shows metadata of the specified image.

Removing Image Properties

```
openstack image unset --property PROPERTY IMAGE_ID_OR_NAME
```

Deleting an Image

```
openstack image delete IMAGE_ID_OR_NAME
```

Removes the specified image from OpenStack Image.

2.8 Viewing and Modifying Membership of Private Images

In the following, find some examples on how to view or modify membership of private images:

Listing Members a Private Image is Shared With

```
glance member-list --image-id IMAGE_ID
```

Lists the IDs of the projects whose members have access to the private image.

Listing Private Images Shared With a Member

```
glance member-list --tenant-id PROJECT_ID
```

Lists the IDs of private images that members of the specified project can access.

Granting Members Access to a Private Image

```
glance member-create [--can-share] IMAGE_ID_OR_NAME PROJECT_ID_OR_NAME
```

Grants the specified member access to the specified private image.
By adding the `--can-share` option, you can allow the members to further share the image.

Revoking Member Access to a Private Image

```
glance member-delete IMAGE_ID_OR_NAME PROJECT_ID_OR_NAME
```

Revokes the access of the specified member to the specified private image.

3 Launching Instances from the SUSE OpenStack Cloud Dashboard

Instances are virtual machines that run inside the cloud. To start an instance, a virtual machine image must exist that contains the following information: which operating system to use, a user name and password with which to log in to the instance, file storage, etc. The cloud contains a pool of such images that have been uploaded to OpenStack Image and are accessible to members of different projects.

3.1 Key Parameters

When starting an instance, specify the following key parameters:

Flavor

In OpenStack, flavors define the compute, memory, and storage capacity of `nova` computing instances. To put it simply, a flavor is an available hardware configuration for a server. It defines the "size" of a virtual server that can be launched.

For more details and a list of default flavors available, refer to the *Administrator Guide*, chapter *Dashboard* or chapter *OpenStack command-line clients*, section *Manage Flavors*. The guide is available from http://www.suse.com/documentation.

Key Pair (optional, but recommended)

Key Pairs are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the `cloud-init` package.

It is recommended to create at least one key pair per project. If you already have generated a key pair with an external tool, you can import it into OpenStack. The key pair can be used for multiple instances belonging to that project.

For details, refer to the *End User Guide*, chapter *OpenStack dashboard* or chapter *OpenStack command-line clients*, section *Configure access and security for instances*. The guide is available from http://www.suse.com/documentation.

Security Group

In SUSE OpenStack Cloud, security groups are used to define which incoming network traffic should be forwarded to instances. Security groups hold a set of firewall policies (security group rules).

For details, refer to the *End User Guide*, chapter *OpenStack dashboard* or chapter *OpenStack command-line clients*, section *Configure access and security for instances*. The guide is available from http://www.suse.com/documentation.

Network

Instances can belong to one or multiple networks. By default, each instance is given a fixed IP address, belonging to the internal network.

Boot Source of the Instance

You can launch instances from the following sources. For details, see the *End User Guide*, chapter *OpenStack dashboard* or chapter *OpenStack command-line clients*, section *Launch and manage instances*. The guide is available from http://www.suse.com/documentation.

- Images that have been uploaded to SUSE OpenStack Cloud.
- Volumes that contain images.
- Instance snapshots.
- Volume snapshots.

3.2 Launching Instances from Images, Snaphots, or Volumes

For instructions on how to launch instances from images or snapshots, see *Launch an Instance* (http://docs.openstack.org/user-guide/dashboard-launch-instances.html).

4 Configuring Access to the Instances

Access to an instance is mainly influenced by the following parameters:

Security Groups and Rules

In SUSE OpenStack Cloud, security groups are used to define which incoming network traffic should be forwarded to instances. Security groups hold a set of firewall policies (security group rules).

For instructions on how to configure security groups and security group rules, see *Configure access and security for instances* (http://docs.openstack.org/user-guide/configure-access-and-security-for-instances.html).

Key Pairs

Key Pairs are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the `cloud-init` package.

It is recommended to create at least one key pair per project. If you already have generated a key pair with an external tool, you can import it into OpenStack. The key pair can be used for multiple instances belonging to that project.

For details on how to create or import keypairs, see *Configure access and security for instances* (http://docs.openstack.org/user-guide/configure-access-and-security-for-instances.html).

IP Addresses

Each instance can have two types of IP addresses: private (fixed) IP addresses and public (floating) ones. Private IP addresses are used for communication between instances, and public ones are used for communication with the outside world. When an instance is launched, it is automatically assigned private IP addresses in the networks to which it is assigned. The private IP stays the same until the instance is explicitly terminated. (Rebooting the instance does not have an effect on the private IP addresses.)

A floating IP is an IP address that can be dynamically added to a virtual instance. In OpenStack Networking, cloud administrators can configure pools of floating IP addresses. These pools are represented as external networks. Floating IPs are allocated from a subnet that is associated with the external network. You can allocate a certain number of floating IPs to a project—the maximum number of floating IP addresses per project is defined by the quota. From this set, you can then add a floating IP address to an instance of the project.

For information on how to assign floating IP addresses to instances, see *Configure access and security for instances* (http://docs.openstack.org/user-guide/configure-access-and-security-for-instances.html).

4.1 Security Group Rules

You can adjust rules of the default security group and rules of any other security group that has been created. When the rules for a group are modified, the new rules are automatically applied to all running instances belonging to that security group.

Adjust the rules in a security group to allow access to instances via different ports and protocols. This is necessary to be able to access instances via SSH, to ping them, or to allow UDP traffic (for example, for a DNS server running on an instance).

Rules in security groups are specified by the following parameters:

IP Protocol

Protocol to which the rule will apply. Choose between TCP (for SSH), ICMP (for pings), and UDP.

Port/Port Range

For TCP or UDP, define a single port or a port range to open on the virtual machine. ICMP does not support ports. In that case, enter values that define the codes and types of ICMP traffic to be allowed.

Source of traffic (*Remote* in the SUSE OpenStack Cloud Dashboard)

Decide whether to allow traffic to instances only from IP addresses inside the cloud (from other group members) or from *all* IP addresses. Specify either an IP address block (in CIDR notation) or a security group as source. Using a security group as source will allow any instance in that security group to access any other instance.

If no further security groups have been created, any instances are automatically assigned to the default security group (if not specified otherwise). Unless you change the rules for the default group, those instances cannot be accessed from any IP addresses outside the cloud.

PROCEDURE 4.1: CONFIGURING SECURITY GROUP RULES

For quicker configuration, the Dashboard provides templates for often-used rules that, including rules for well-known protocols on top of TCP (such as HTTP or SSH), or rules to allow all ICMP traffic (for pings).

1. Log in to SUSE OpenStack Cloud Dashboard and select a project from the drop-down box at the top-level row.
2. Click *Project* › *Compute* › *Access & Security*. The view shows the following tabs: *Security Groups*, *Key Pairs*, *Floating IPs*, and *API Access*.

3. On the *Security Group* tab, click *Manage Rules* for the security group you want to modify. This opens the *Security Group Rules* screen that shows the existing rules for the group and lets you add or delete rules.
4. Click *Add Rule* to open a new dialog.
   From the *Rule* drop-down box, you can select templates for often-used rules, including rules for well-known protocols on top of TCP (such as HTTP or SSH), or rules to allow all ICMP traffic (for pings). In the following steps, we will focus on the most commonly-used rules only:
5. To enable SSH access to the instances:
   a. Set *Rule* to `SSH`.
   b. Decide whether to allow traffic to instances only from IP addresses inside the cloud (from other group members) or from *all* IP addresses.
      - To enable access from *all* IP addresses (specified as IP subnet in CIDR notation as `0.0.0.0/0`), leave the *Remote* and *CIDR* fields unchanged.
      - Alternatively, allow only IP addresses from other security groups to access the specified port. In that case, set *Remote* to `Security Group`. Select the desired *Security Group* and *Ether Type* (`IPv4` or `IPv6`).
6. To enable pinging the instances:
   a. Set *Rule* to `ALL ICMP`.
   b. Decide whether to allow traffic to instances only from IP addresses inside the cloud (from other group members) or from *all* IP addresses.
      - To enable access from *all* IP addresses (specified as IP subnet in CIDR notation as `0.0.0.0/0`), leave the *Remote* and *CIDR* fields unchanged.
      - Alternatively, allow only IP addresses from other security groups to access the specified port. In that case, set *Remote* to `Security Group`. Select the desired *Security Group* and *Ether Type* (`IPv4` or `IPv6`).
7. To enable access via a UDP port (for example, for syslog):
   a. Set *Rule* to `Custom UDP`.
   b. Leave the *Direction* and *Open Port* values untouched.

c. In the *Port* text box, enter the value *514*.

d. Decide whether to allow traffic to instances only from IP addresses inside the cloud (from other group members) or from *all* IP addresses.

   - To enable access from *all* IP addresses (specified as IP subnet in CIDR notation as `0.0.0.0/0`), leave the *Remote* and *CIDR* fields unchanged.
   - Alternatively, allow only IP addresses from other security groups to access the specified port. In that case, set *Remote* to `Security Group`. Select the desired *Security Group* and *Ether Type* (`IPv4` or `IPv6`).

5 Deploying Kubernetes

By combining OpenStack, Docker, Kubernetes, and Flannel, you get a containers solution which works like other OpenStack services. With Magnum, Docker and Kubernetes are made available as first class resources in OpenStack.

5.1 Deploying a Kubernetes Cluster from Command Line

A cluster (formerly `bay`) is the construct in which Magnum launches container orchestration engines.

REQUIREMENTS

- The `python-openstackclient` is installed. After you have sourced an OpenStack RC file, use the command line client to upload images from a machine outside of the cloud.
- To run the `python-openstackclient`: An OpenStack RC file containing the credentials for the OpenStack project to which you want to upload the images.
- The `python-magnumclient` is installed.
- Install the `openstack-magnum-k8s-image-x86_64` package. This package provides a virtual machine image with Kubernetes pre-installed, `openstack-magnum-k8s-image.x86_64.qcow2`. OpenStack Magnum uses this image when creating clusters with its `k8s_opensuse_v1` driver.

1. In a shell, source the OpenStack RC file for the project that you want to upload an image to. For details, refer to http://docs.openstack.org/user-guide/common/cli_set_environment_variables_using_openstack_rc.html.
2. List the Magnum image uploaded to Glance using the **`openstack image list | grep openstack-magnum-k8s-image`**. If no image found, you can create an image for cluster setup as shown below:

```
openstack image create openstack-magnum-k8s-image \
  --public --disk-format qcow2 \
  --property os_distro='opensuse' \
  --container-format bare \
  --file /srv/tftpboot/files/openstack-magnum-k8s-image/openstack-magnum-k8s-
image.x86_64.qcow2
```

3. Create a Magnum flavor. For example:

```
openstack flavor create --public m1.magnum --id 9 --ram 1024 \
   --disk 10 --vcpus 1
```

   If you do not have enough resources and RAM on your compute nodes for a flavor of this size, create a smaller flavor instead.

4. Create a cluster template for Kubernetes. For example:

```
magnum cluster-template-create --name k8s_template \
  --image-id openstack-magnum-k8s-image \
  --keypair-id default \
  --external-network-id floating \
  --dns-nameserver 8.8.8.8 \
  --flavor-id m1.magnum \
  --master-flavor-id m1.magnum \
  --docker-volume-size 5 \
  --network-driver flannel \
  --coe kubernetes \
  --master-lb-enabled
```

5. Create a Kubernetes cluster using the cluster template you have created in the step above. For example:

```
magnum cluster-create --name k8s_cluster --cluster-template k8s_template \
  --master-count 1 --node-count 2
```

   The resulting cluster will have one master Kubernetes node and two minion nodes.

5.2 Deploying a Kubernetes Cluster from the Dashboard

Alternatively, you can deploy a Kubernetes cluster in the SUSE OpenStack Cloud Dashboard by creating a cluster template and creating a Kubernetes cluster afterward.

REQUIREMENTS

- You have created an image for cluster setup as described in *Section 5.1, Step 2*.
- You have created a Magnum flavor as described in *Section 5.1, Step 3*.

PROCEDURE 5.1: CREATING A CLUSTER TEMPLATE IN SUSE OPENSTACK CLOUD DASHBOARD

1. Log in to SUSE OpenStack Cloud Dashboard and select a project from the drop-down box at the top-level row.
2. Click *Container Infra* › *Cluster Templates* › *Create Cluster Template*.
   The *Create Cluster Template* dialog opens, showing the following sections: *Info*, *Node Spec*, *Network*, and *Labels*.
3. In the *Info* section:
   a. Enter a name for the cluster template to create.
   b. As *Container Orchestration Engine*, choose `Kubernetes`.
   c. If wanted, activate the following options:
      - *Public*: The cluster template will be visible for all users in OpenStack.
      - *Enable Registry*: The cluster can be built with Insecure Docker Registry service.
      - *Disable TLS*: Switch off the SSL protocol for the cluster.
4. In the *Node Spec* section:
   a. Choose the *Image* you have created in *Section 5.1*, *Step 2*.
   b. Choose a *Keypair*.
   c. Choose the *Flavor* you have created in *Section 5.1*, *Step 3*. It will be used for the minion nodes.
   d. Choose the same flavor as *Master Flavor*. It will be used for the master node.
   e. As *Volume Driver*, choose `Cinder`.
   f. As *Docker Storage Driver*, choose `Device Mapper`.
   g. Specify the *Docker Volume Size (GB)*. For example: `5`
5. In the *Network* section:
   a. As *Network Driver*, choose `Flannel`.
   b. Leave the *HTTP Proxy*, *HTTPS Proxy*, and *No Proxy* boxes empty or enter the respective addresses to use.

c. As *External Network ID*, enter `floating`. The network `floating` will be used to connect to the cluster template you are creating.

d. Leave the *Fixed Network* and *Fixed Subnet* boxes empty.

e. Enter the *DNS* server to use for this cluster template. For example: `8.8.8.8.`

f. To deploy the cluster with a load balancer service in front for the cluster services, activate *Master LB*.

g. To assign floating IP addresses to the nodes in the cluster, activate *Floating IP*.

6. Confirm your changes to create the cluster template.

PROCEDURE 5.2: CREATING A CLUSTER IN SUSE OPENSTACK CLOUD DASHBOARD

Based on the cluster template you have created in *Procedure 5.1, "Creating a Cluster Template in SUSE OpenStack Cloud Dashboard"*, you can now create a Kubernetes cluster.

1. Log in to SUSE OpenStack Cloud Dashboard and select a project from the drop-down box at the top-level row.

2. Click *Container Infra* › *Clusters* › *Create Cluster*.
   The *Create Cluster* dialog opens, showing the following sections: *Info*, *Size*, and *Misc*.

3. In the *Info* section:

   a. Enter a *Cluster Name*.

   b. From the *Cluster Template* list, select the template you have created in *Procedure 5.1, "Creating a Cluster Template in SUSE OpenStack Cloud Dashboard"*.

4. In the *Size* section, enter the number of master nodes and minion nodes you want the cluster to have. For example: `1` and `2`.

5. In the *Misc* section, you can optionally specify a custom URL for node discovery and a *Timeout* for cluster creation, if wanted. The default is no timeout.

6. Confirm your changes to create the cluster.

5.3 Deploying a Kubernetes Cluster Without Internet Access

In specific scenarios, you may need to deploy a Kubernetes cluster without access to Internet. For those cases, you need to set up a custom Insecure Docker Registry and use no discovery URL. You can do this either from command line (as described below) or from the SUSE OpenStack Cloud Dashboard.

1. In a shell, source the OpenStack RC file for the project that you want to upload an image to. For details, refer to http://docs.openstack.org/user-guide/common/cli_set_environment_variables_using_openstack_rc.html.

2. Create a cluster template as shown in , *Step 4*, but add the options `--registry-enabled` and `--labels`. The `registry_url` must include the protocol, e.g. http://*URL*. For example:

```
magnum cluster-template-create --name k8s_template_reg_enabled \
 [...]
 --registry-enabled
 --labels registry_url=http://192.168.255.10/srv/files
```

3. Create a cluster as shown in , *Step 5*, but with static IP configuration and setting the option `--discovery-url` to `none`. For example:

```
magnum cluster-create --name k8s_cluster_without \
  --cluster-template k8s_template_reg_enabled \
  [...]
  --discovery-url none
```

A Documentation Updates

This chapter lists content changes for this document.

This manual was updated on the following dates:

- *Section A.1, "July 2017 (Maintenance Release SUSE OpenStack Cloud 7)"*
- *Section A.2, "February 2017 (Initial Release SUSE OpenStack Cloud 7)"*
- *Section A.3, "January 2017 (Maintenance Release SUSE OpenStack Cloud 6)"*
- *Section A.4, "February 2015 (Initial Release SUSE OpenStack Cloud 5)"*

A.1 July 2017 (Maintenance Release SUSE OpenStack Cloud 7)

Bugfixes

- Corrected registry_url example *Section 5.3, "Deploying a Kubernetes Cluster Without Internet Access"* (https://bugzilla.suse.com/show_bug.cgi?id=1024683).
- Corrected `openstack image create` example *Procedure 2.2, "Uploading Disk Images to SUSE OpenStack Cloud"* (https://bugzilla.suse.com/show_bug.cgi?id=1047502).

A.2 February 2017 (Initial Release SUSE OpenStack Cloud 7)

Bugfixes

- Added *Chapter 5, Deploying Kubernetes* (https://bugzilla.suse.com/show_bug.cgi?id=999859).

A.3 January 2017 (Maintenance Release SUSE OpenStack Cloud 6)

Bugfixes

- Added *Section 2.6, "Using a Custom Kernel to Boot Btrfs Images Under Xen"* (https://bugzilla.suse.com/show_bug.cgi?id=964618).

A.4 February 2015 (Initial Release SUSE OpenStack Cloud 5)

General

- Moved the SUSE-specific additions for the *Administrator Guide* and *End User Guide* to the *Supplement to Administrator Guide and End User Guide*.