Cisco Edition OpenStack

User Guide

Contents

Figures

1.0 Introduction

1.1 Preface

This document is an OpenStack User Guide intended as a reference for both OpenStack Tenant User and Tenant Admin.

OpenStack is a global collaboration of developers and cloud computing technologists producing the ubiquitous open source cloud computing platform for public and private clouds. The project aims to deliver solutions for all types of clouds by being simple to implement, massively scalable, and feature rich. The technology consists of a series of interrelated projects delivering various components for a cloud infrastructure solution.[1]

OpenStack provides a tool to orchestrate a cloud, including running instances, managing networks, storing files, and controlling access to the cloud through users and projects. It provides the software that can control an Infrastructure as a Service (IaaS) cloud computing platform. It is similar in scope to Amazon EC2 Cloud Servers and Amazon S3. Major components include OpenStack Compute (also known as Nova), OpenStack Storage (also known as Swift), OpenStack Image Service (also known as Glance), OpenStack Dashboard (also known as Horizon), and OpenStack Identity Services (also known as Keystone). OpenStack Compute does not include any virtualization software; rather it defines drivers that interact with underlying virtualization mechanisms that run on your host operating system, and exposes functionality over a REST APIs. [2]

All components in OpenStack offer REST APIs that may be invoked by user-firendly web-based user interfaces, Command Line Interfaces (CLIs) or customized interfaces. In this guide we will explain how to use the web-based UI (Horizon) and python-based CLI Clients.

OpenStack can be used by many different projects (tenants) sharing resources in the same system. Earlier versions of OpenStack used the term "project" instead of "tenant". Because of this legacy terminology, the word projects and tenants are used interchangeably this document.

This User guide assumes the successful deployment of OpenStack based on the installation process indicated in the OpenStack Installation Guide [1].

[1] Description from http://OpenStack.org/, 06/26/2012.
[2] Adapted from http://nova.OpenStack.org/nova.concepts.html, 06/26/2012.

1.2 Audience

This document is intended for Cisco Advanced Services Data Center Practice Team and Customer Network Architecture Team.

1.3 Document Scope

The scope of this document will include all the steps necessary for a tenant user or tenant admin to administer all functions within a project such are creating key-pair, managing instances, security groups, etc. The document is divided into two major sections. First, Web-based OpenStack Tenant Operations and the second is a CLI-based OpenStack Tenant Operations.

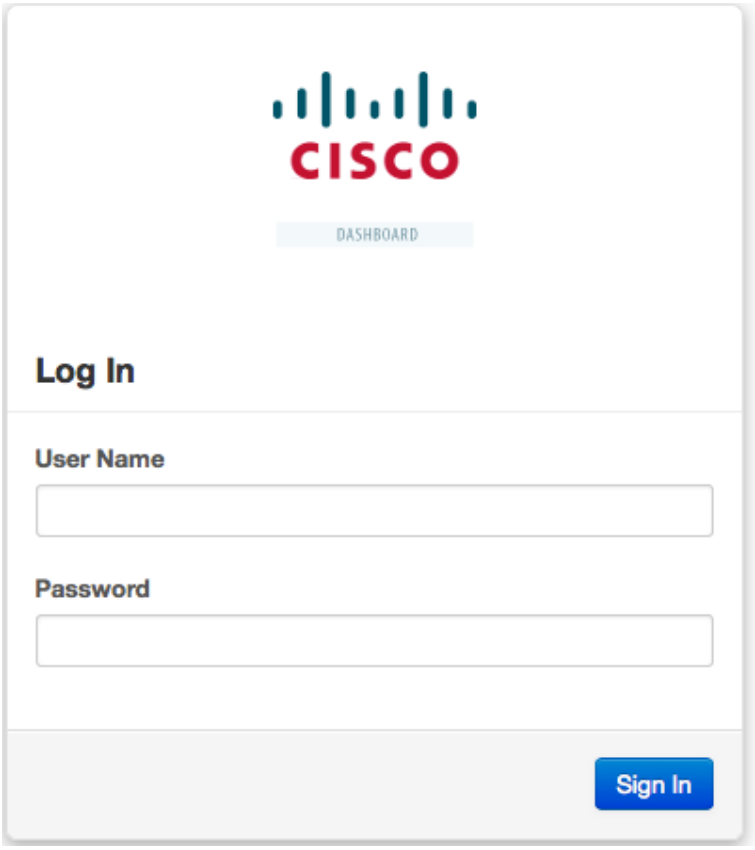
This document does not include anything that relates to OpenStack Administrative functions, such as Creating projects within OpenStack, creating new users, managing project quotas, etc. These items will be covered in a separate Cisco OpenStack Admin Guide document.

2.0 Web-Based OpenStack Tenant Operations

OpenStack offers a user-friendly graphical interface called the OpenStack Dashboard (also known by it's codename: Horizon). The dashboard offers two different views: the Admin System Panel and the Project Panel. The Admin System Panel is exclusively for cloud administration activities such as creating projects and users or registering images. Tenants use the Project Panel to control their projects and perform functions such as managing compute servers or creating new instances and volumes. In this guide we will focus on the Project Panel.

The OpenStack Dashboard is a web-based interface compatible with Firefox 13 (recommended), IE 7 and Crome 19. To access the Dashboard, open your browser and type the IP address or hostname of the Horizon host server. Figure 1 illustrates the OpenStack Dashboard Log In screen.

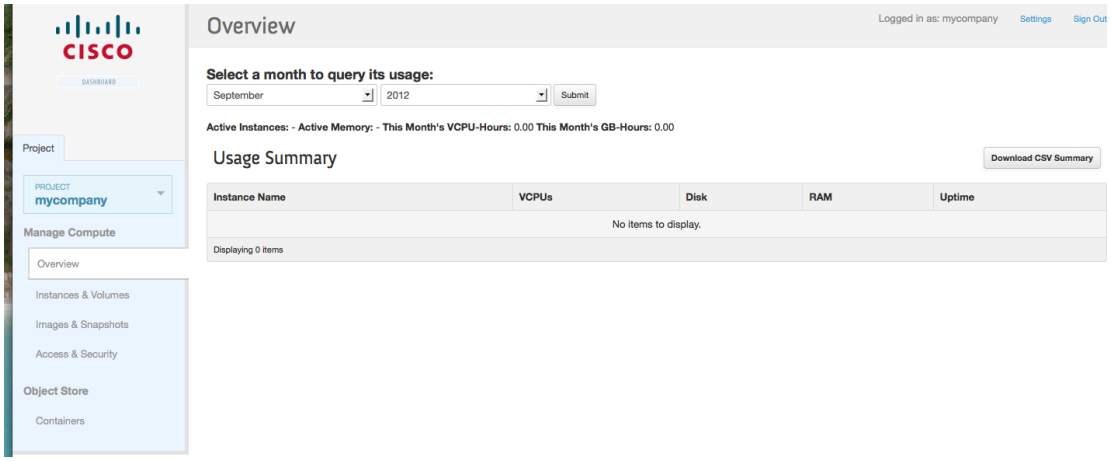
Figure 1: OpenStack Dashboard Login



2.1 Project Overview

Once project users have logged into the OpenStack Dashboard using the username and credentials provided by the OpenStack Administrator, they will be directed to the Overview section where they will find a project-based usage summary of the cloud infrastructure. The summary includes information such as the number of instances running, memory utilization, and disk space. Figure 2 shows the summary for the "mycompany" project (tenant). Since this is a new project, we have yet to create instances.

Figure 2: Project-based Overview of Usage Summary



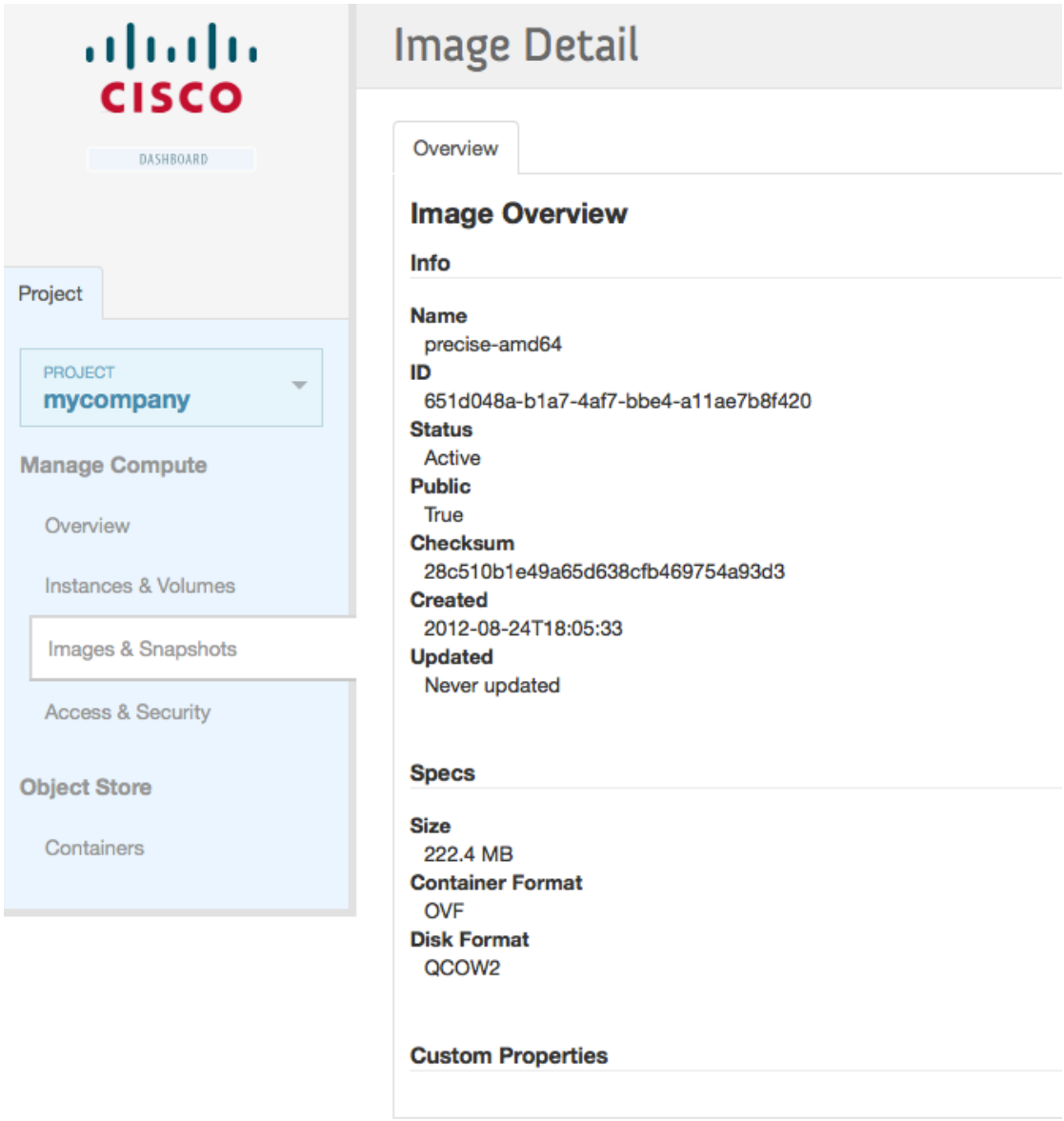
2.2 Image Overview

An image is a file containing information about a virtual disk that completely replicates all information about a working computer at a point in time including operating system information and file system information. OpenStack integrates Glance as the image service. Glance is in charge of fetching all available images on to the host machine. The OpenStack compute service then boots the images it finds on the host machine.
The OpenStack Dashboard has limited integration with the full set of Glance APIs [2]. Basically, it just provides an overview of the already uploaded images. There will be images owned by the Administrator that cannot be deleted or modified by users.

On the Project Panel select the section "Images & Snapshots". The complete list of available images will be displayed. Then, just select the name of the desired image

and the information will be displayed. Figure 3 shows detailed information from the Ubuntu image.

Figure 3: Glance Image Details

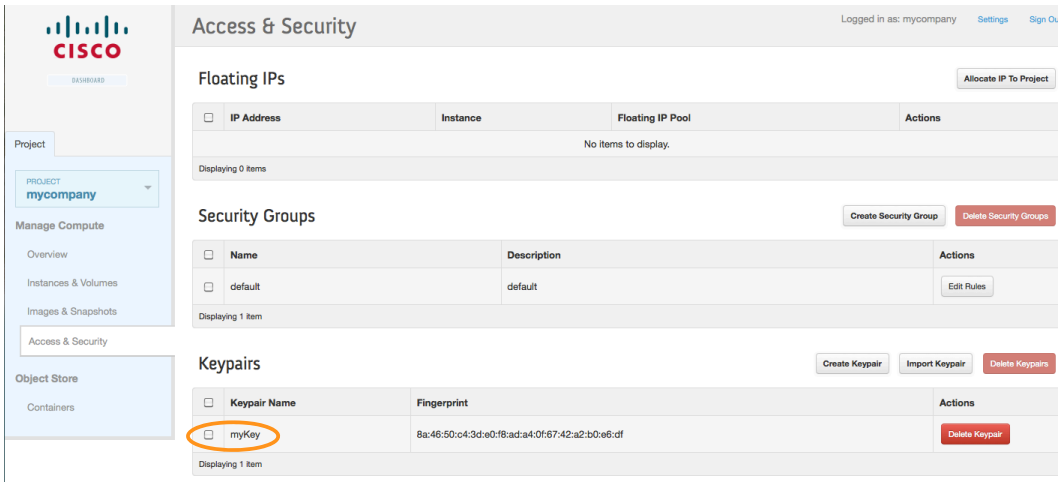


2.3 Creation of Key-Pair

Now that we've seen a summary of a project, let's consider how users will interact with the project. Public/private key pairs control programmatic access to the tenant-facing API's in an OpenStack cloud. Key pairs are created for each user within a project. To create a keypair, click on "Access & Security" in

the navigation menu on the left (Figure 4). Scroll to the "keypairs" section of the page and click on the "Create Keypair" button to generate a new key pair. You'll need to supply a name for each keypair—the name can be whatever you want, but in practice the name often reflects whom the keypair belongs to. A file with the key will be automatically downloaded to the computer used to interact with the OpenStack Dashboard.

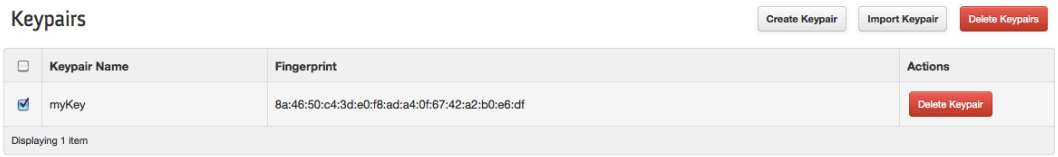
Figure 4: Access and Security



The file will have the extension .pem. Save this key to your laptop, as it will be needed to access the instances that will be created under this project.

OpenStack Dashboard will show the key created with the only available action of deleting it. Figure 5 shows the created keypair named "mykey".

Figure 5: Creating a new keypair



Section "Accessing Instances by Terminal Console" covers how to use keypair files to securely gain access to instances without the need for a password.

2.4 Creation of Security Groups

OpenStack provides ingress filtering for the instances based on the concept of security groups. A Security Group is a named set of rules that get applied to the incoming packets for the instances. You can specify a security group while

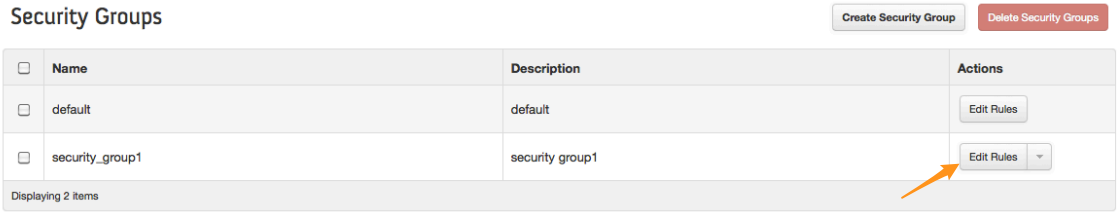
launching an instance. Each security group can have multiple rules associated with it. Each rule specifies the source IP/network, protocol type, destination ports etc. Any packet matching these parameters specified in a rule is allowed in. The rest of the packets are blocked.

To create a new Security Group, click on “Access & Security” in the navigation menu on the left. Scroll to the “Security Groups” section of the page and click on the “Create Security Group” button to generate a new Security Group.

Figure 6: Create Security Groups

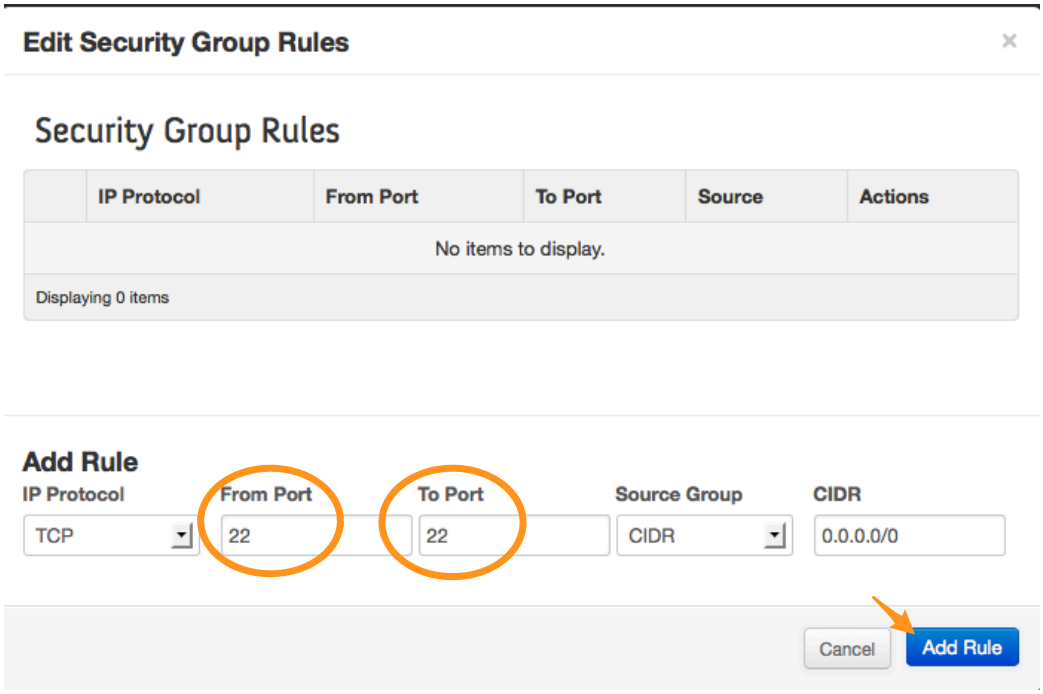
To add rules to the newly created security group, click on “Edit Rules” button.

Figure 7: Add Rules to Security Group



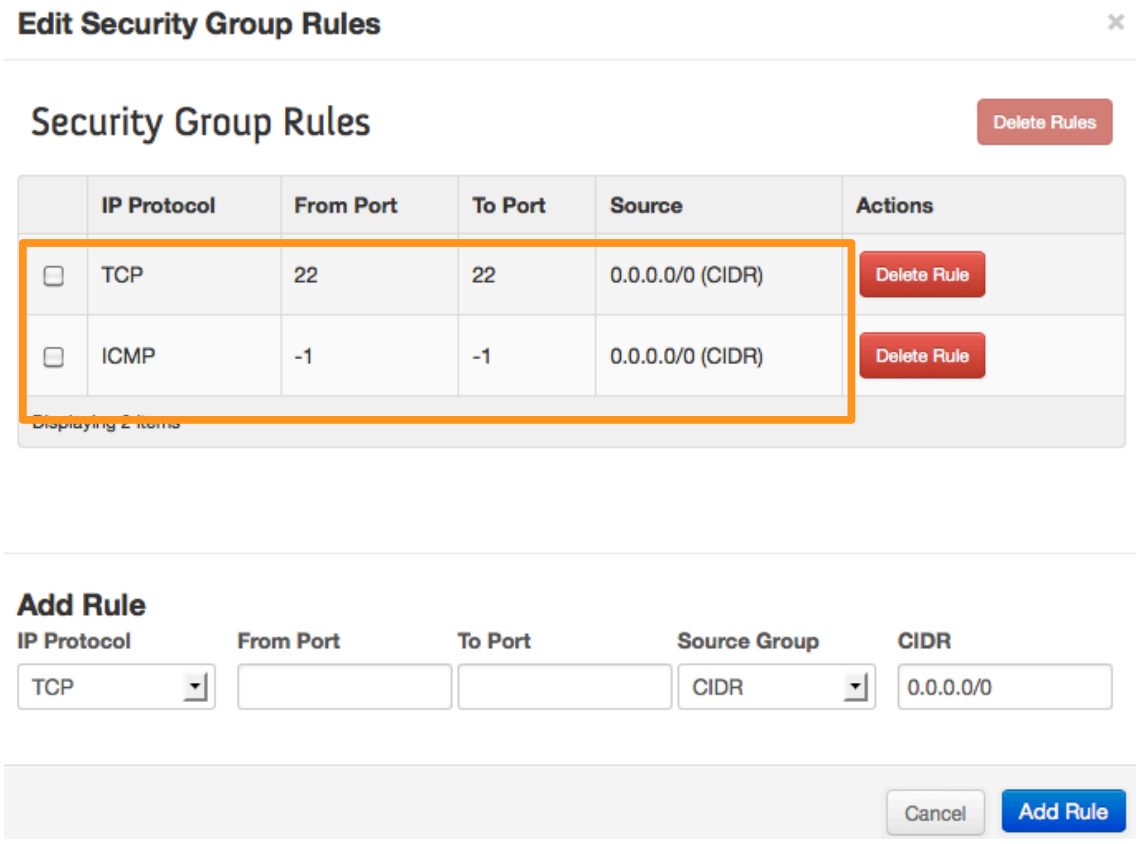
Add a rule to allow SSH traffic.

Figure 8: Edit Security Group



Also, repeat previous step to add another rule to allow ICMP packets.

Figure 9: Security Group Rules

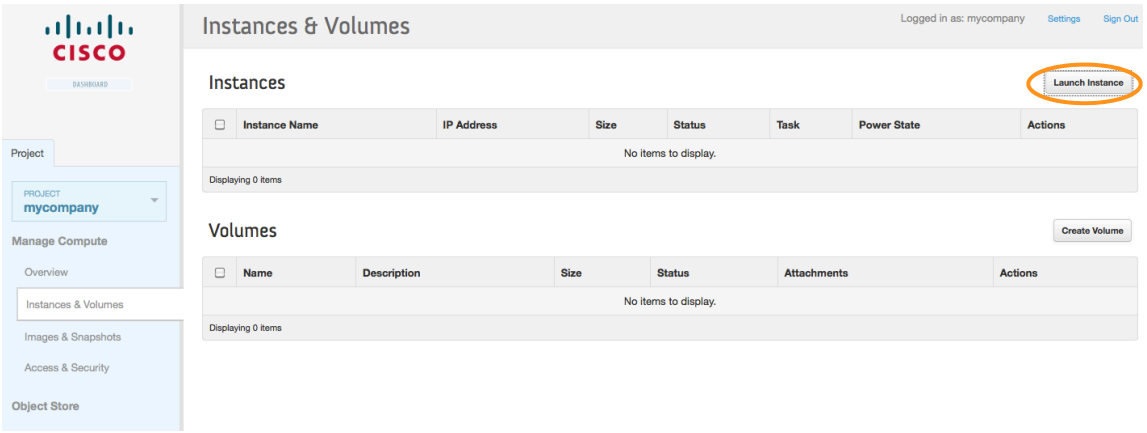


2.5 Instance Management

An instance is a running virtual machine within the OpenStack cloud. An instance has a life cycle that is controlled by OpenStack Compute. Compute creates the instances and it is responsible for building a disk image, launching it, reporting the state, attaching persistent storage, and terminating it.

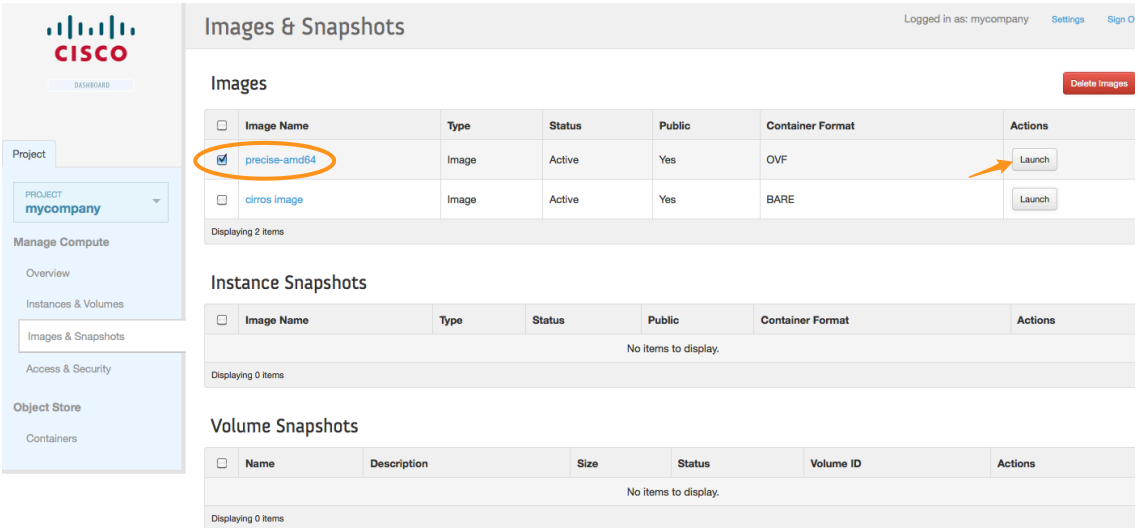
We are ready to start new instances. On the Project Panel select "Instances & Volumes" and then simply click over the "Launch Instance" bottom at the top of this screen (Figure 10).

Figure 10: List of Instances and Volumes



Select the desired VM image for the new instance and click over "Launch". Figure 11 shows the ubuntu image "precise-amd64" ready to be launched.

Figure 11: Images Available



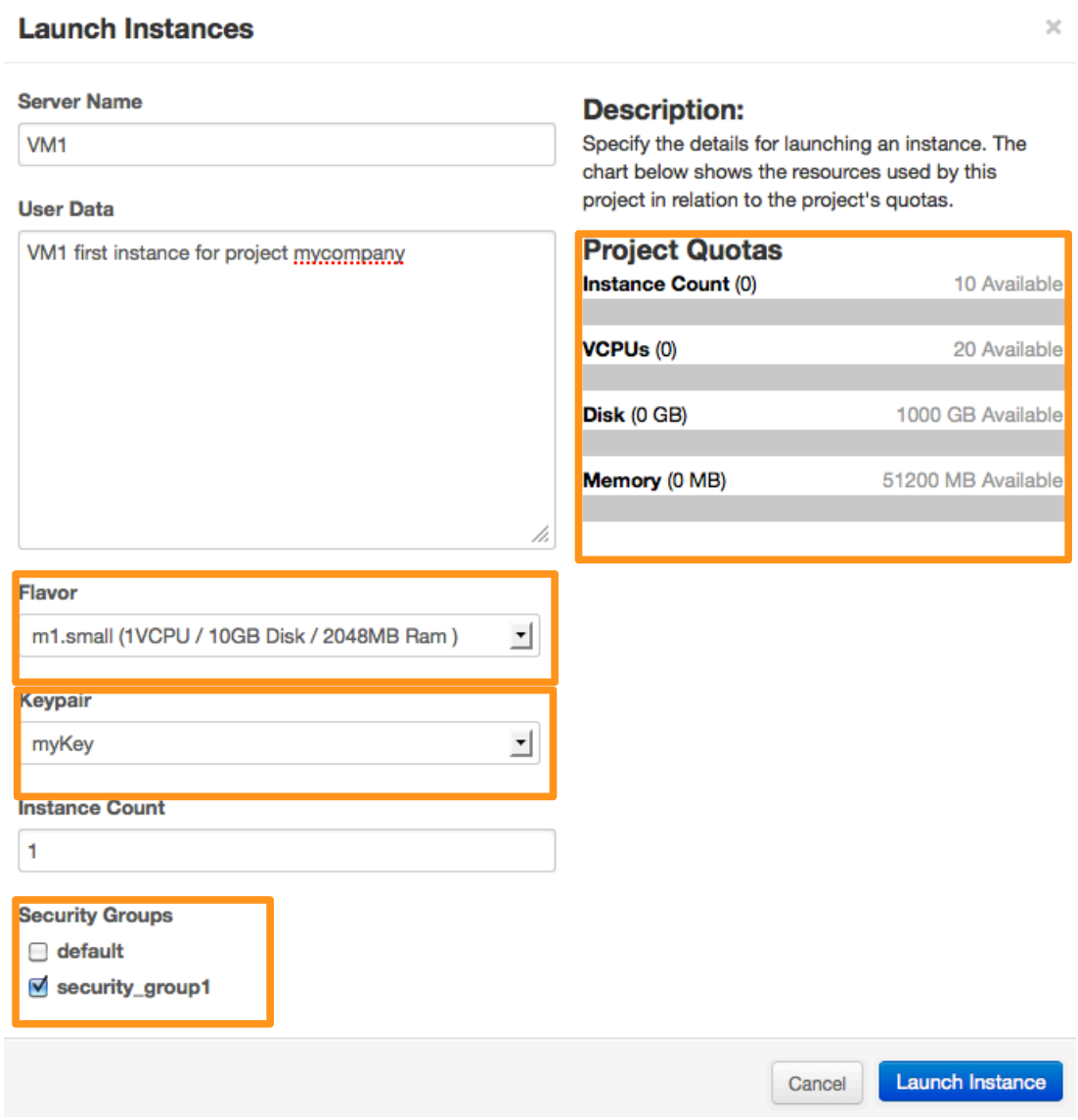
Next, we need to assign a name to the instance and select the parameters such as flavor (size of the VM based on CPU, HD and Memory), a keypair and a security group.

We can also assign also a brief description of the instance. If we want to create several instances with the same characteristics, we can also set the number of instances we wish to create. Ensure your image does not exceed your user quota (i.e. disk space); otherwise the instance will fail to spawn.

Figure 12 shows an example in which we've created a single instance called "vm1" using the "m1.small" flavor. Available keypair files may be selected in this section. OpenStack

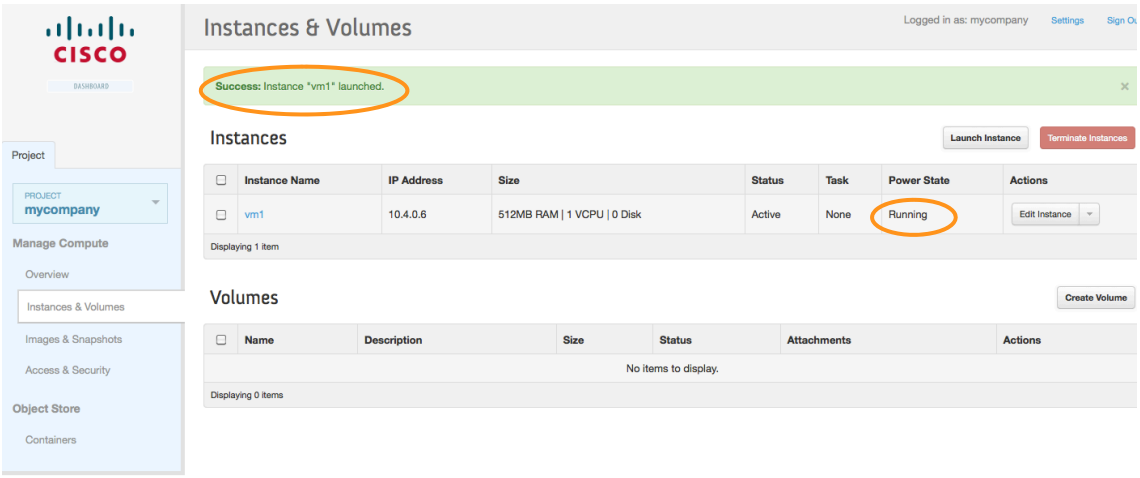
will inject the signed RSA private key into the instance after completing the instantiation process (image creating, booting and network assignment).

Figure 12: New Instances Required Parameters



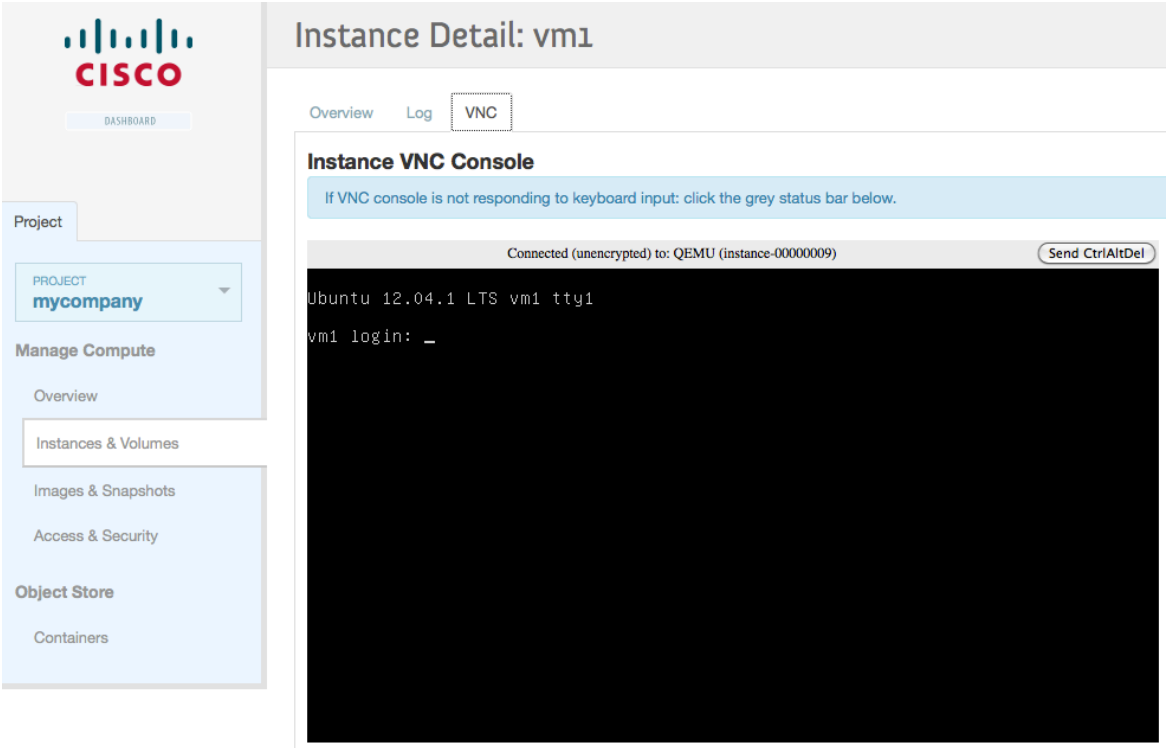
Once you've finished filling out the form, click the "Launch Instance" button to boot the instance. The instances section (Figure 13) will show the status of the newly created instance:

Figure 13: Status of the new Instance



The Action section allows Tenant Admins to see the Console Log of the recently created instance, by simply selecting the "View Log" option.
In the same section, the Project Admin can launch the VNC Console view (Figure 14). This view allows the Project Administrator to interact with the instance as it boots up. Note that you cannot use VNC Console from a Chrome browser. You need both Flash installed and a Firefox browser.
For VM created using Ubuntu image, you need to first create a user via ssh to the VM and then use that user to login to this console.

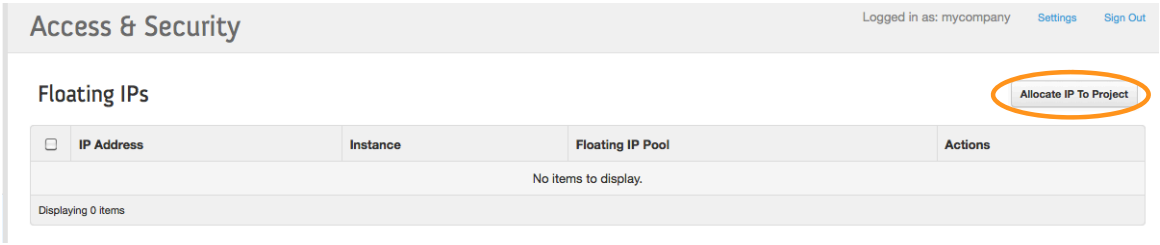
Figure 14: Console View per Instance



2.6 Floating IP Management

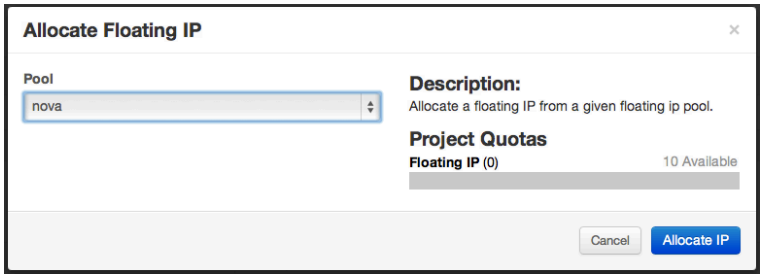
OpenStack automatically assigns one private IP address to each spawned instance. Users may assign a public IP that is commonly known as "Floating IP" because it is added to a running instance. Floating IP's let users connect to instances from a public network without having to first connect to the private IP.
Assigning a floating IP is done from the Access & Security section of the Project Panel. The first step is to allocate an IP for the project (tenant). Select Allocate IP To Project, as shown in the following figure:

Figure 15: Floating IP list in Dashboard



The OpenStack administrator will already have IP pools available (Figure 16). A user needs to select an available pool from the Pool drop-down list and click "Allocate IP".

Figure 16: Floating IP Allocation by Project



You should get a conformation message from the system about the allocation process. The next screen shows how to associate the floating IP to the instance.

Figure 17: Confirmation message after assigning Floating IPs

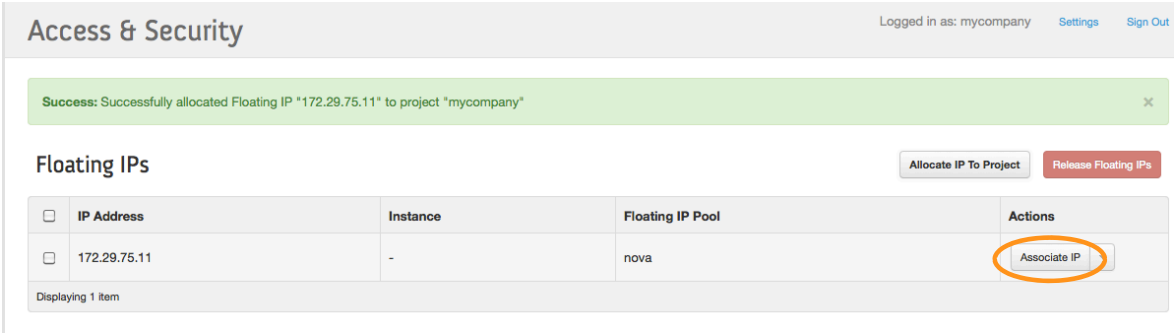


Figure 18: Associate Floating IP to Instance

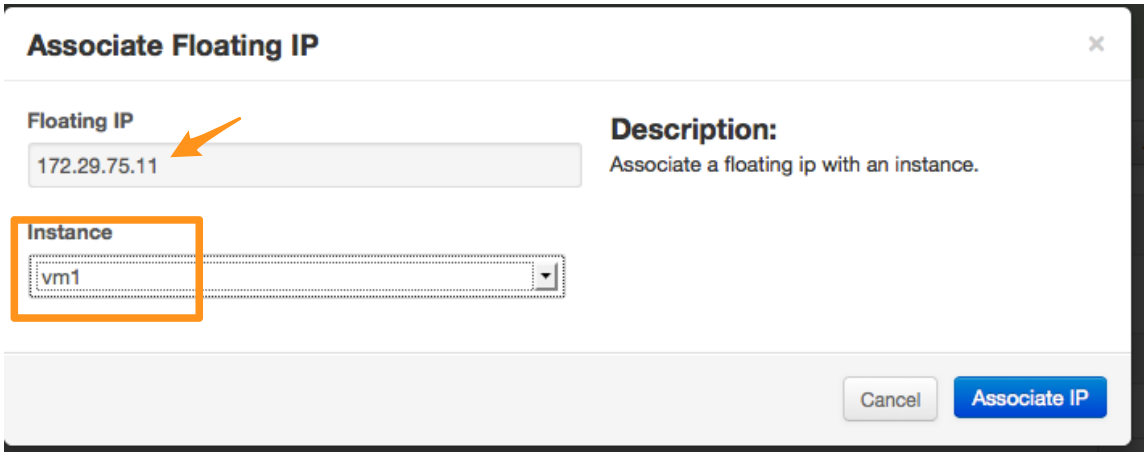


Figure 19: Confirmation message after associating Floating IP

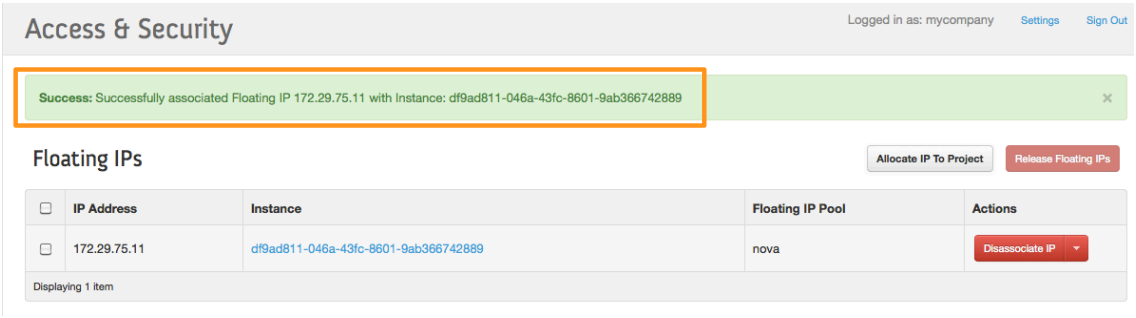
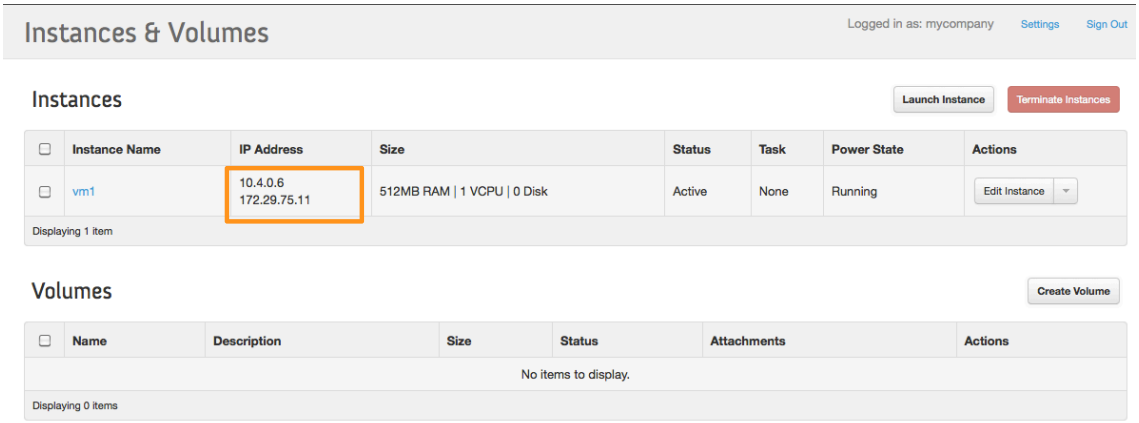


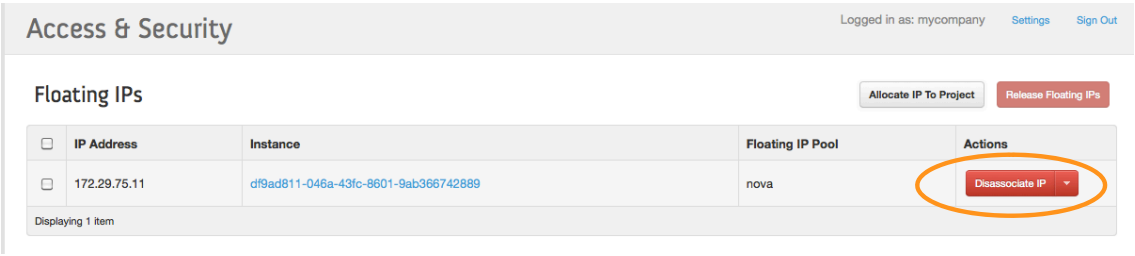
Figure below shows instance VM1 now has the floating IP assigned to it along with the private IP.

Figure 20: Verify Instance to Floating IP Association



The process of releasing floating IP address is known as Disassociate IP. This is done by clicking on the "Disassociate IP" button next to the respective instance.

Figure 21: Dis-associate Floating IP from Instance



Once a floating IP is disassociated from an instance, it can now be re-assigned to another instance as outlined in the previous steps.

2.7 Accessing Instance using Floating IP

An instance can be accessed using the floating IP assigned to the Virtual Machine. First, we must locate the key-pair downloaded to your laptop as documented in the "Creation of Key-Pair" section. Open a terminal on your laptop and locate the private key.

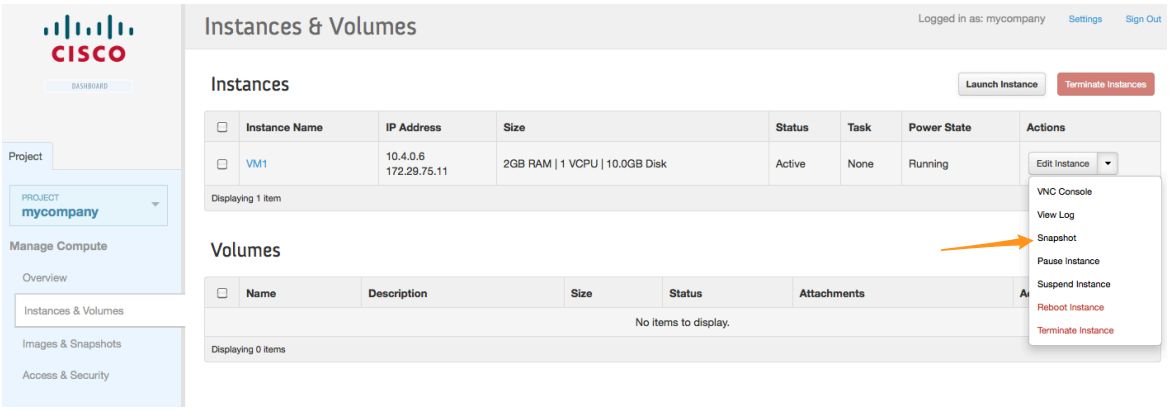
```
% pwd
/Users/username/Downloads
% ls -lrt | grep mykey
-rw-r--r--   1 username  staff        887 Sep  4 12:05 mykey.pem

Change private key permission to allow only read/write permission to the user.
% chmod 600 mykey.pem


You can now ssh to the VM1 instance.
 % ssh ubuntu@172.29.75.11 -i /<path>/mykey.pem
Welcome to Ubuntu 12.04.1 LTS (GNU/Linux 3.2.0-29-virtual x86_64)
….
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
ubuntu@vm1:~$
```

2.8 Instance Snapshots

A virtual machine (instance) is physically represented by one or more files that contain the operating system configuration and state. This is managed differently between various hypervisors but all of them are capable of saving the state of an instance at certain point. This is known as taking a snapshot. Snapshots are very useful is rolling back a system configuration to a state where everything was working as desired but they are also a great alternative to clone instances for creating customized version of images.

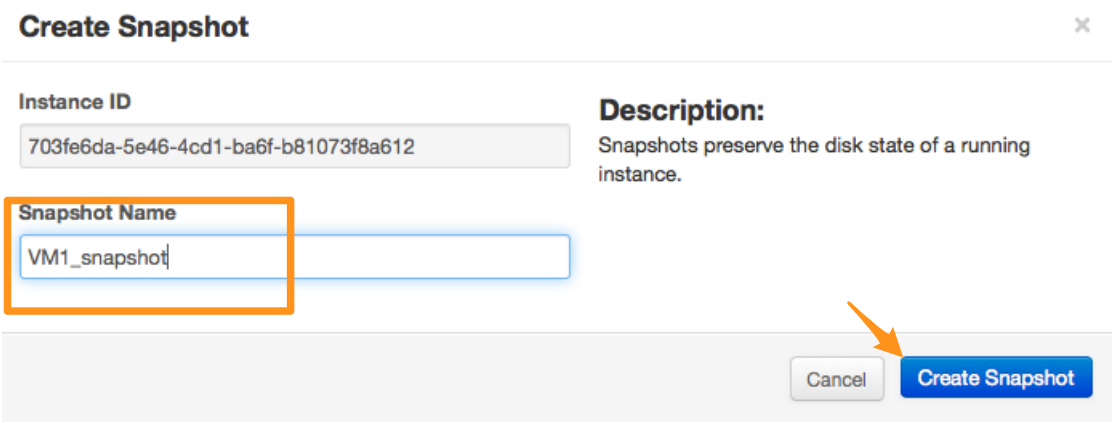
The system administrator will have uploaded some default images for use. Those images may not have all the software and configurations required to start building applications. Customizing a default image is vital to speed-up productivity and to provide a set of user-specific images with applications already configured. Snapshots are created from user instances, from the “Instances & Volumes” section select the instance to be snapshotted and select the option Snapshot from the “Edit Instance” menu as next Figure 22 shows.

Figure 22: Instance Snapshot



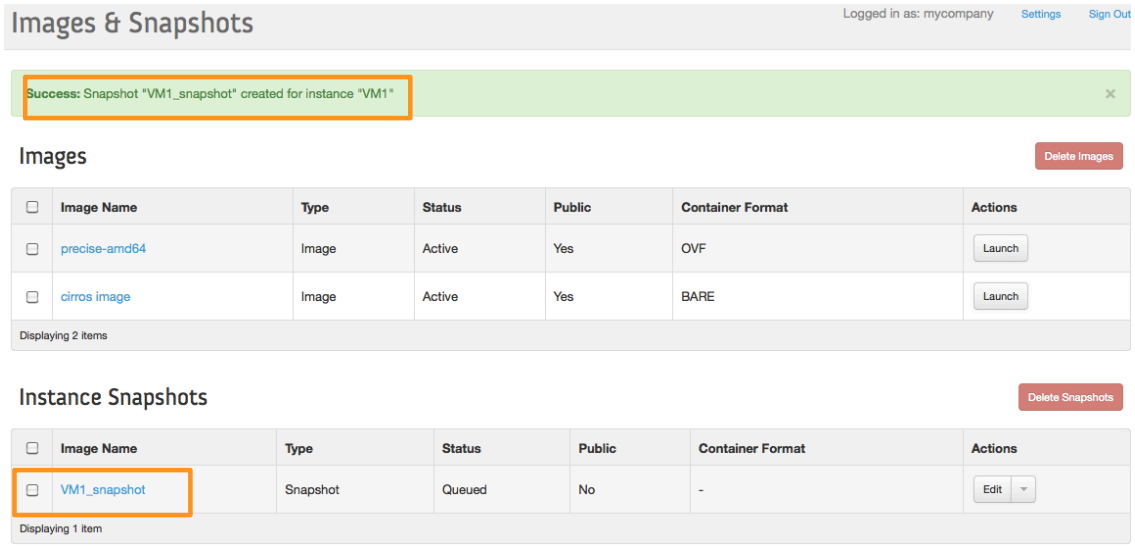
After assigning a name for the snapshot, simply click on Create Snapshot and the system will create it automatically.

Figure 23: Snapshot name assignment



The newly created snapshot will appear on the Images and Snapshots section of the Project Panel (Figure 24).

Figure 24: Instance Snapshot

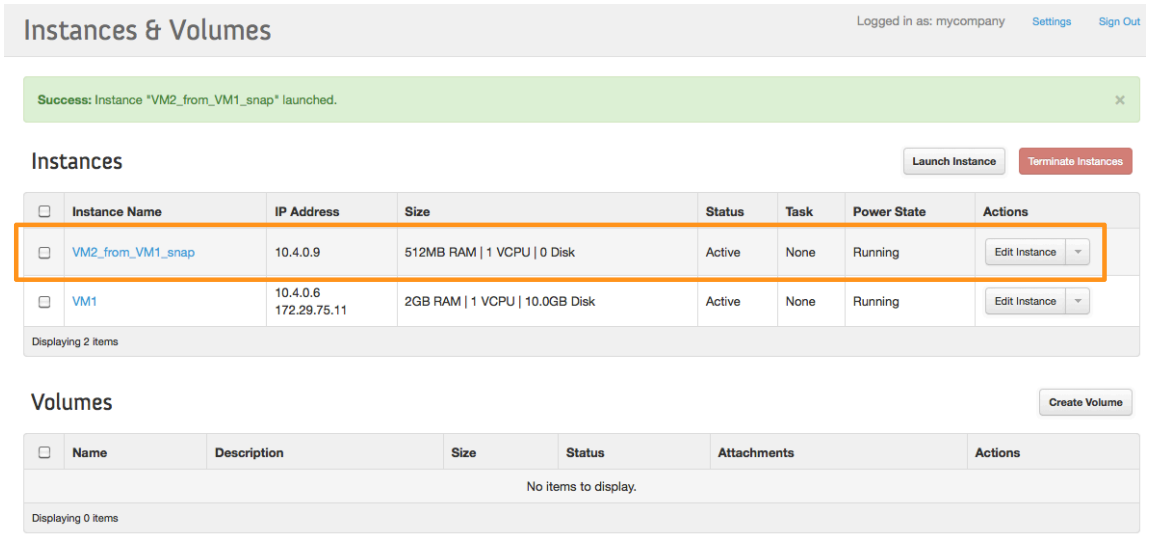


This is the place to launch a new Instance from that snapshot. In the Action column for the snapshot, there will be the launch button (figure 25). When clicked, OpenStack will request the same information that is requested when a new instance is created from a default image. This instance will be shown on the Instances and Volumes section.

Figure 25: Launch Instance from Snapshot

Figure 26, shows the vm-from-snapshot that we just created.

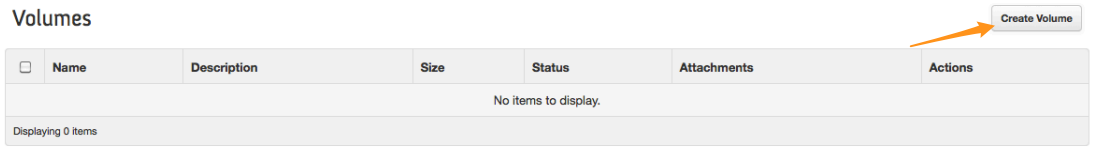
Figure 26: New Instance from VM1 Snapshot



2.9 Nova Volumes

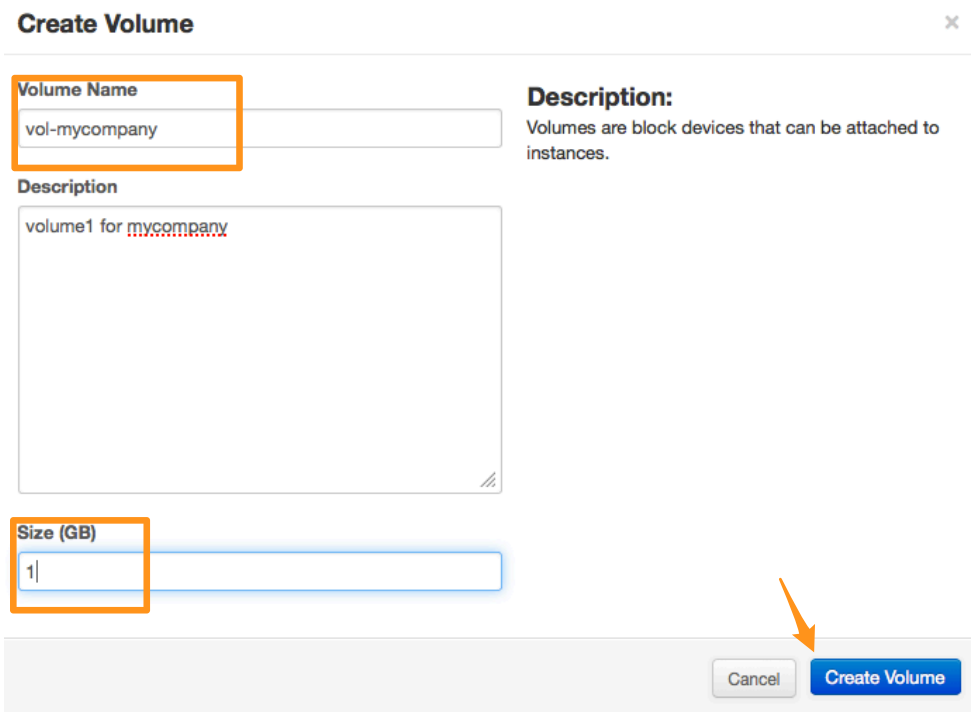
OpenStack (optionally) provides block-level persistent storage to instances through Volumes. This is an iSCSI solution that uses Logical Volume Manager (LVM) for Linux. Users may create volumes based on their respective quota limits. Volumes are managed from the Instances & Volumes section of the Project Panel. We create a volume by clicking over the "Create Volume" bottom as shown on Figure 27.

Figure 27: Create Volume



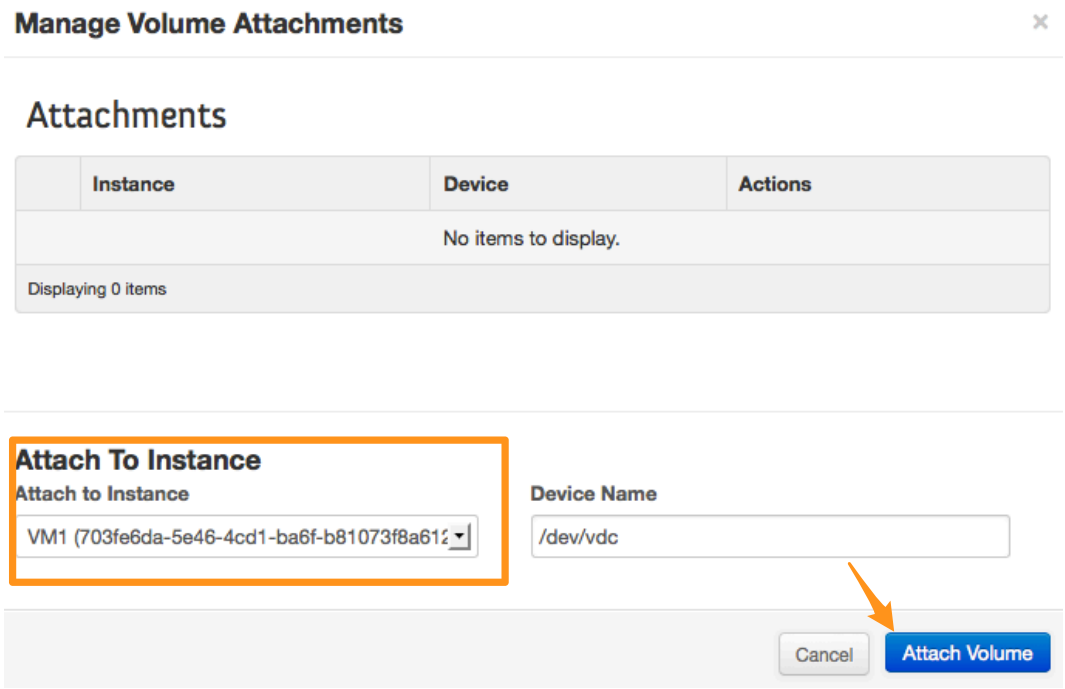
Users need to assign a reference name for the volume, a short description, and the size in GB. Then simply select "Create Volume" and the new volume will appear in the Instances & Volumes screen.

Figure 28: Fields required for creating a volume



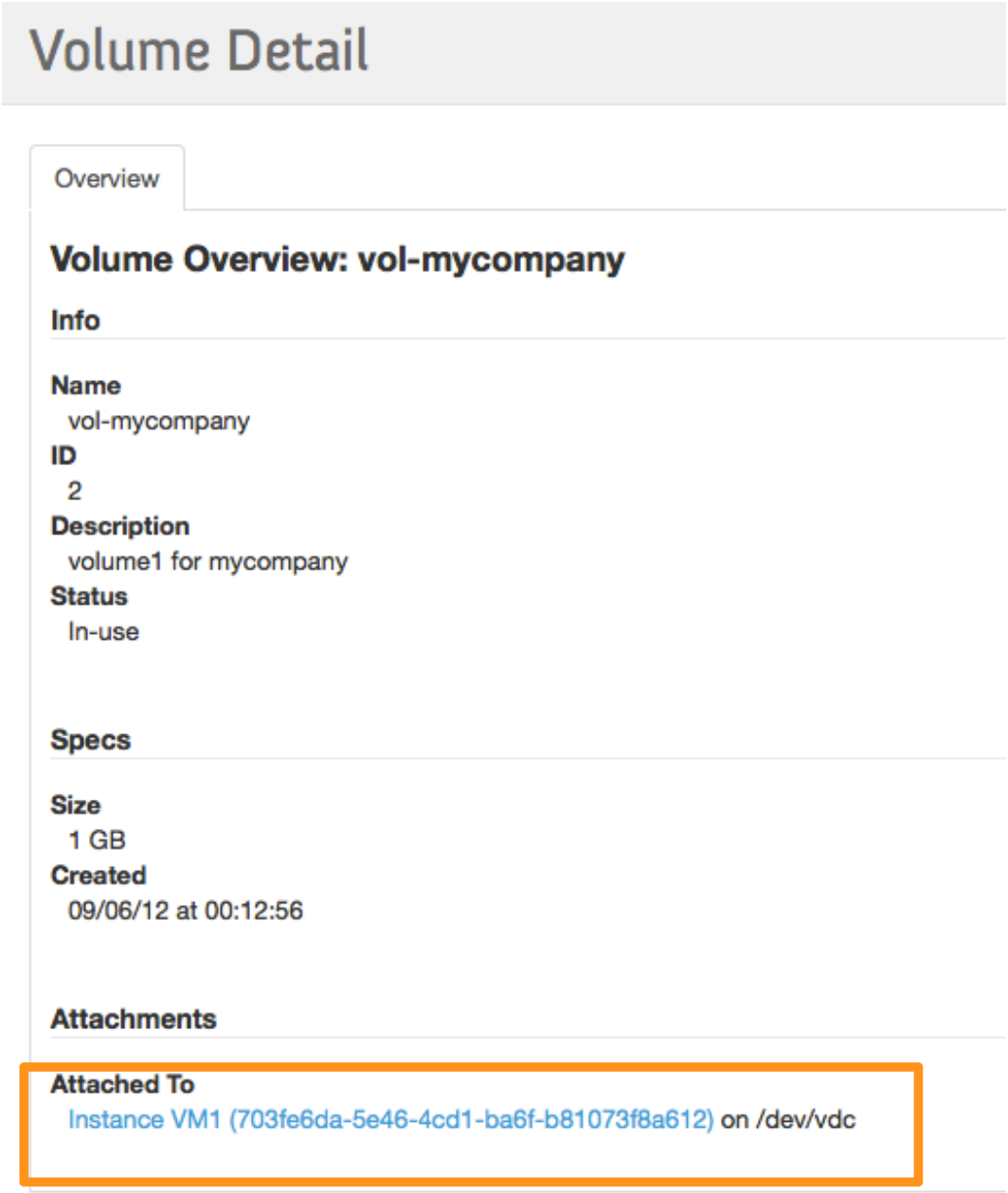
Once the volume has been created, it needs to be attached to an instance. Selecting "Edit Attachments" for the respective volume will open a new window where an instance needs to be selected. This is the attachment process shown in the Figure below.

Figure 29: Attaching a volume to an instance



The next screenshot shows the association between the volume created (vol-mycompany) and the instance selected.

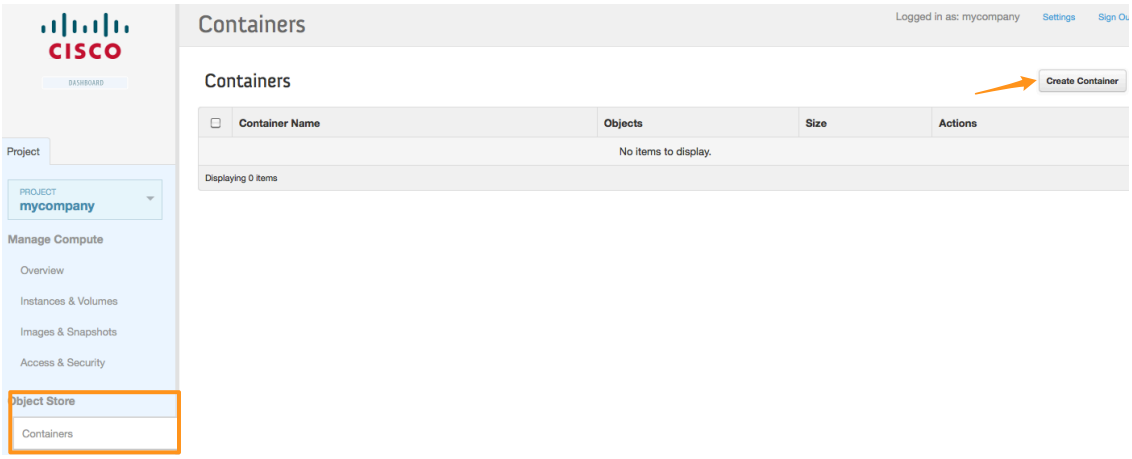
Figure 30: Volume Attached Overview



2.10 Swift Containers

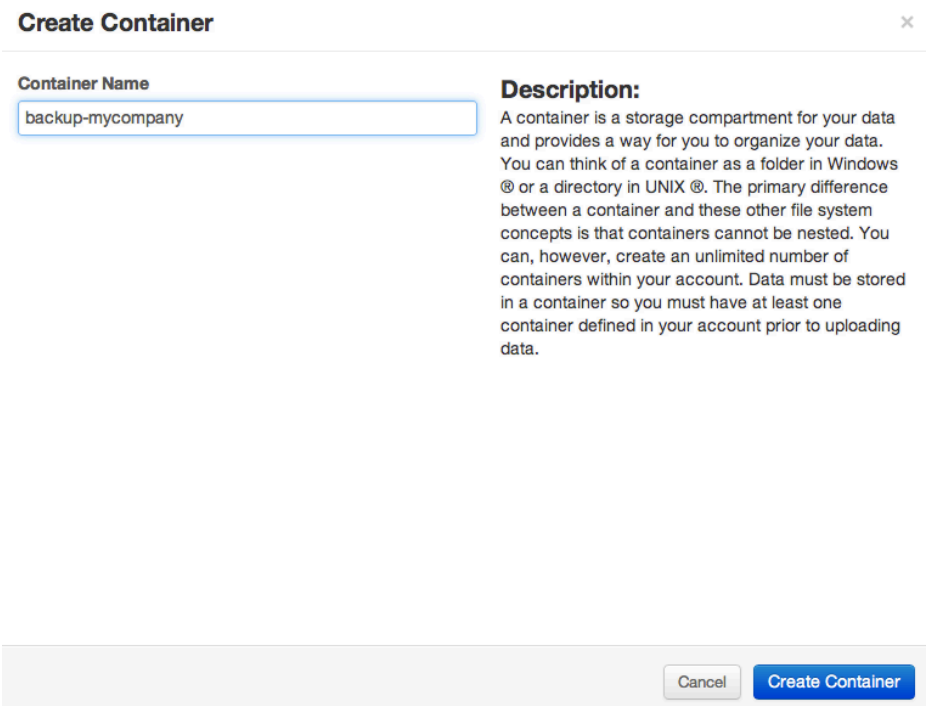
A container is part of the optional OpenStack Object Storage System. A container is basically a reserved storage location where users can allocate data. Swift containers offer high capacity and scalability, strong data consistency, REST APIs. Keep in mind that containers cannot be nested. The data allocating process in OpenStack is done in two steps. First, users need to create a container and second, upload all documents to be stored. Containers are created from the Project Panel: there is section called “Containers” just below “Object Store” shown in Figure 31.

Figure 31: Containers section in Project Panel



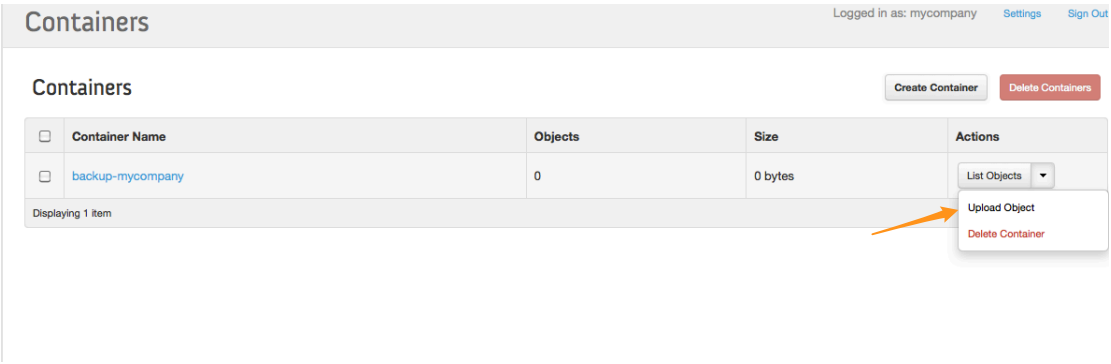
At the right side there two available actions: Create and Delete Containers. After selecting "Create Container", the user will be asked to assign a name for the container and confirmation (Figure 32) After the container is created, it will be listed on the containers section.

Figure 32: New Container Name Assignment



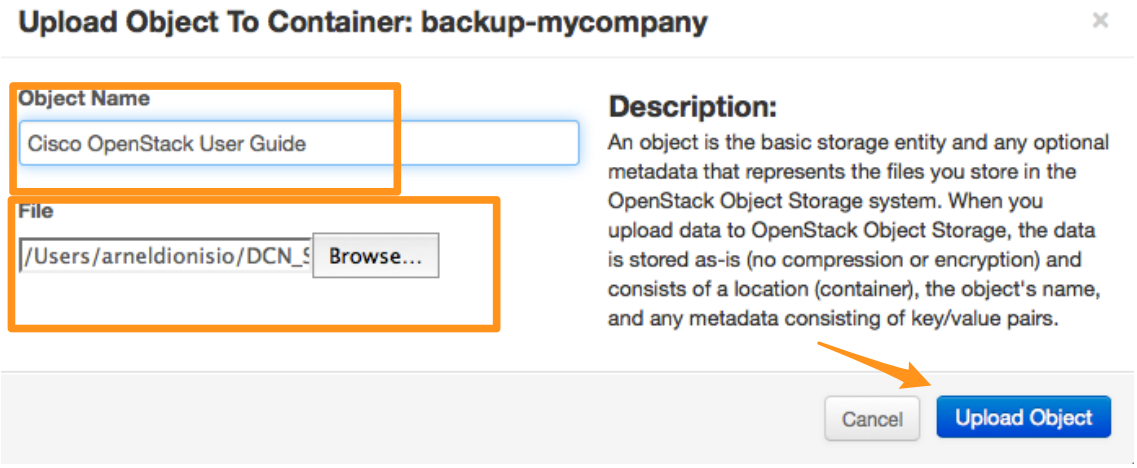
The second step is to upload objects (which is in fact the data from the users). In the Actions column from the containers list, the user may select Upload Objects. Figure 33 illustrates the location of this action.

Figure 33: List of Containers



The user will be requested to assign a name to the object and upload the file as the following figure shows.

Figure 34: Uploading Objects to Containers



Users may access the list of object from each container by selecting the container

Users may access the list of object from each container by selecting the container

3.0 CLI-Based OpenStack Tenant Operations

An alternative for invoking the OpenStack APIs is to use the python-based command line interface clients. These clients are available as individual packages that need to be installed in the Tenant user system. This section will include all equivalent CLI's needed to repeat the steps outlined in Section 2 (Web-Based OpenStack Tenant Operations) as well as CLI commands that are required for performing certain operations that are not included in the Dashboard.

3.1 OpenStack Python-Based Command Line Interface Clients

Pre-Requisite: All CLI-based commands will need to be executed from a Tenant system. For the purpose of this user guide, we used a Linux system with Ubuntu 12.04 LTS release.

In this Tenant Linux System, install the following OpenStack Service clients.

Installing Nova Client:
$ sudo apt-get install python-novaclient
Installing Glance Client:
$ sudo apt-get install glance-client
Installing Keystone Client:
$ sudo apt-get install python-keystoneclient

Installing Swift Client:
$ sudo add-apt-repository ppa:swift-core/release
$ sudo apt-get update
$ sudo easy_install pip
$ sudo pip install python-swiftclient

Clients include a help section that lists the available sub-commands and optional arguments. To verify the installation is successful, try the client help commands below.

$ nova help
$ glance –help

$ keystone help
$ swift help

3.2 User Credentials

Prior to using any of the OpenStack clients, users should set up few essential environment variables that are used by these clients to construct the proper API calls. These variables are:
OS_AUTH_URL
OS_TENANT_ID
OS_TENANT_NAME
OS_USERNAME
OS_PASSWORD

For descriptions of these variables, please refer to the Unified CLI wiki page.

These variables information can be retrieved from the Dashboard for a given tenant user. On the top right corner, select the "Settings" link. Then, select the section named "OpenStack Credentials" on the left panel as shown in Figure 35 and click over the Download RC File bottom.

Figure 35: Download OpenStack User Credentials

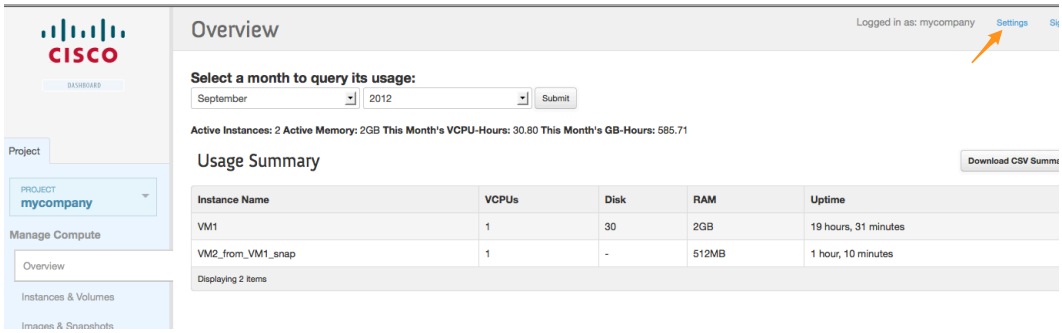
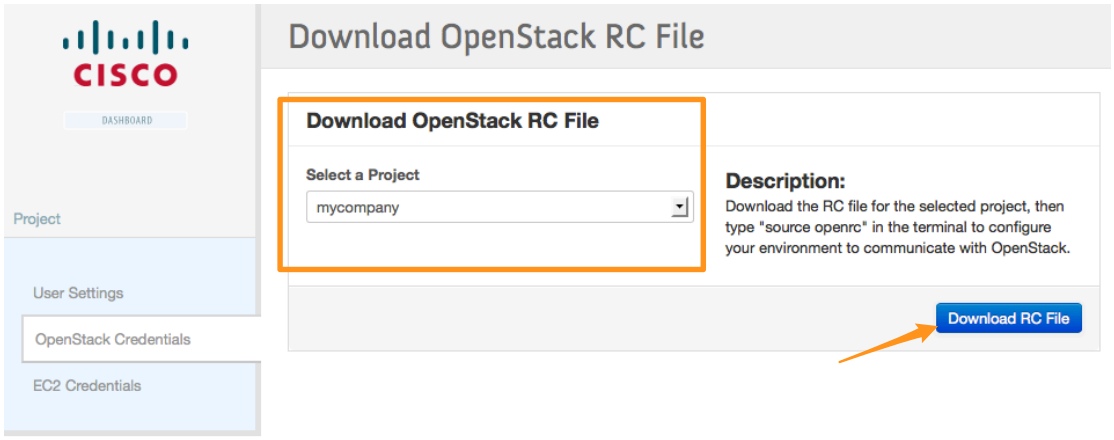


Figure 36: Download OpenStack RC File

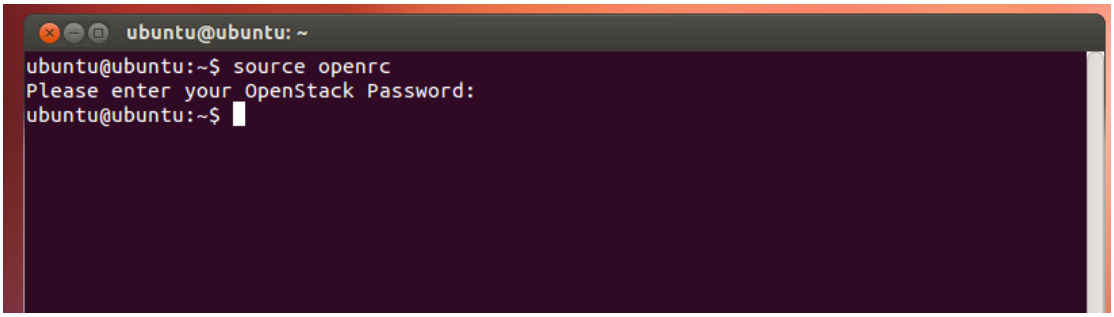


The file downloaded (openrc) will contain private information about the user account, it is important to not share it and keep in a private secure location.

```
#!/bin/bash

# With the addition of Keystone, to use an openstack cloud you should
# authenticate against keystone, which returns a **Token** and **Service
# Catalog**.  The catalog contains the endpoint for all services the
# user/tenant has access to - including nova, glance, keystone, swift.
#
# *NOTE*: Using the 2.0 *auth api* does not mean that compute api is 2.0.  We
# will use the 1.1 *compute api*
export OS_AUTH_URL=http://172.29.74.193:5000/v2.0

# With the addition of Keystone we have standardized on the term **tenant**
# as the entity that owns the resources.
export OS_TENANT_ID=6d35d471937e4862997372b1a1d3f7ff
export OS_TENANT_NAME=mycompany

# In addition to the owning entity (tenant), openstack stores the entity
# performing the action as the **user**.
export OS_USERNAME=mycompany

# With Keystone you pass the keystone password.
echo "Please enter your OpenStack Password: "
read -s OS_PASSWORD_INPUT
export OS_PASSWORD=$OS_PASSWORD_INPUT
```

Open a console terminal in your Tenant linux system and type "source openrc" to configure your environment to communicate with OpenStack. Note that you will need to do this in each shell from which you intend to use CLI clients. Note the openrc file will ask you to input the Tenant user password.

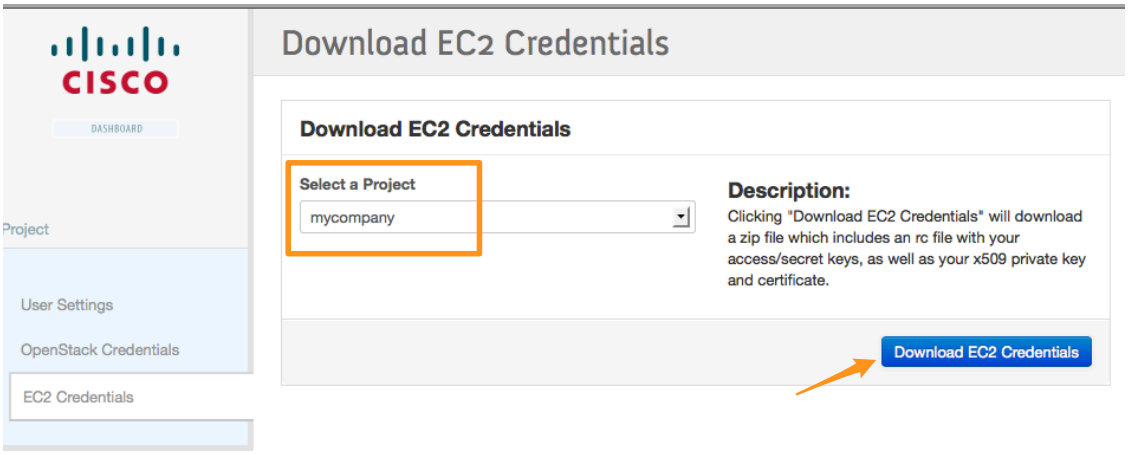
Figure 37: Source Openrc File



OpenStack offers a high level of compatibility for the EC2 APIs. The complete API compatibility matrix is included on section 4. When users use euca2ools as a client to construct EC2 APIs calls, different variables need to be set.
These are:
EC2_ACCESS_KEY
EC2_SECRET_KEY
EC2_URL
EC2_USER_ID
EC2_PRIVATE_KEY
EC2_CERT
EUCALYPTUS_CERT

For descriptions of these variables, please refer to the Eucalyptus documentation. The process to set up the EC2 credentials is very similar to the one for OpenStack credentials, but the settings will be provided in a zip file which includes the EC2 cert.pem keys as well as an rc file. This zip file can be downloaded from the same "settings" page.

Figure 38: EC2 Credentials



Sourcing User Credentials EC2 File:
 # source /PATH_TO_FILE/ec2rc.sh

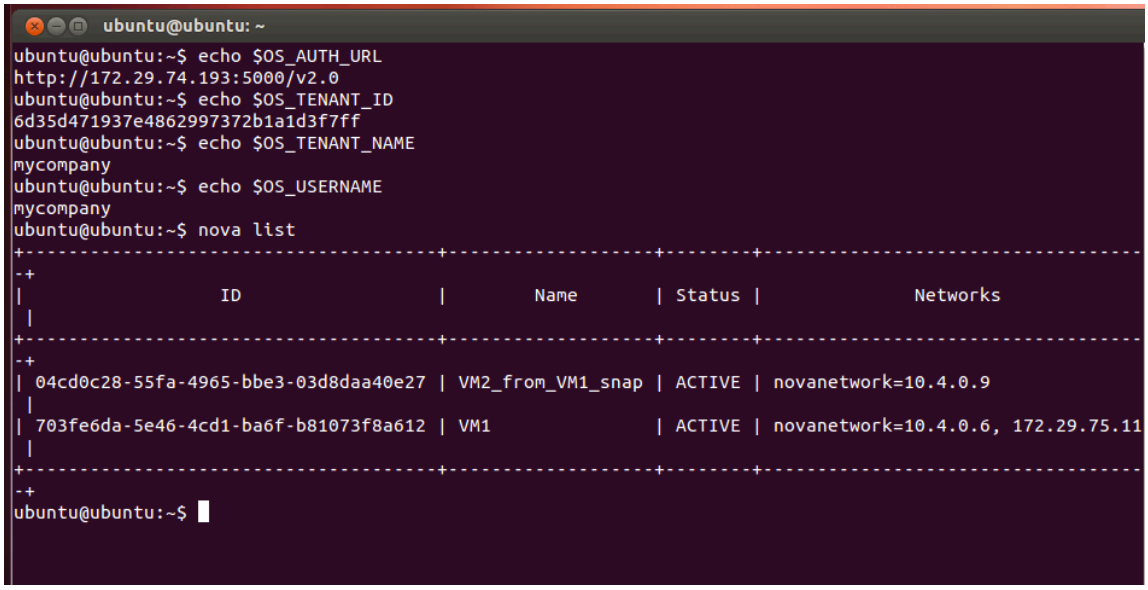
3.3 Tenant Operations

3.3.1 Project Overview

To list all existing instances in a project use the following nova command:
nova list
To list all command options for "nova list"
nova help list
To list an instance matching a particular name
nova list –name=<instance_name>

Figure 39; list all the existing instances in project "mycompany"

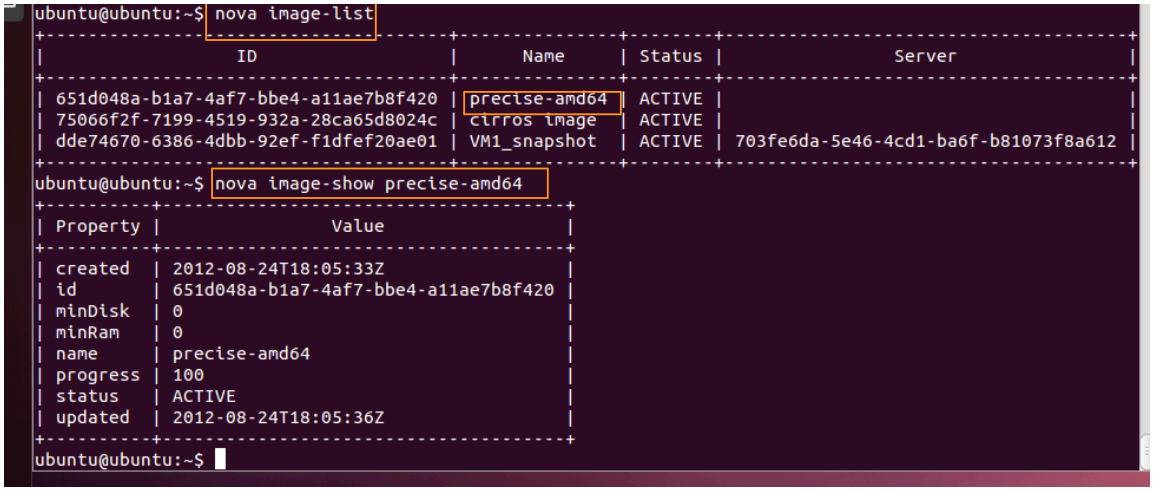
Figure 39: Tenant Instance List



3.3.2 Image Overview

To list all existing nova image related CLI's:
nova help | grep image
To list all existing image(s) in a project
nova image-list
To view details of a particular image
nova image-show <image_name>

Figure 40: Nova Image list for project "mycompany"

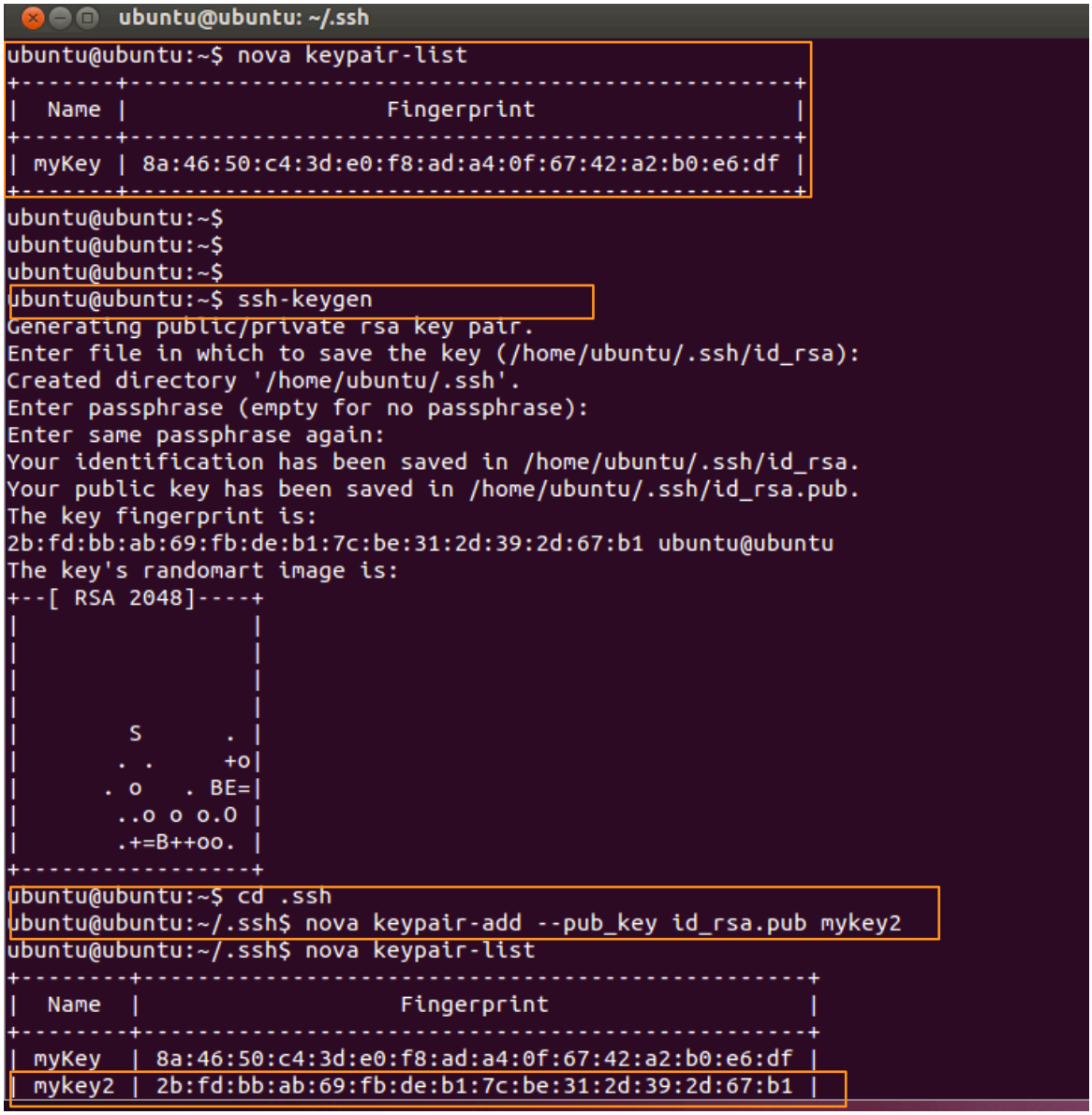


3.3.3 Creation of Key-Pair

To list all existing nova keypair related CLI's:
nova help | grep keypair

To list all existing keypair (s) in a project
nova keypair-list
To create a new keypair to be used for new instances
nova keypair-add –pub_key <id_rsa.pub> <key_name>

Figure 41: Create Key-Pair



3.3.4 Creation of Security Groups

To list all existing nova security group related CLI's:
nova help | grep secgroup
To list all existing security group (s) in a project
nova secgroup-list

To list all existing security group rules(s) in a specific security-group
nova secgroup-list-rules <*secgroup_name*>
To create a new security-group to be used for new instances
nova secgroup-create <*secgroup_name*> <*description*>
To add a new security-group-rule to an existing security group
nova secgroup-add-rule <*ip_protocol*> <*from_port*> <*to_port*> <*cidr*>

Figure 42: View Existing Security Groups and list rules

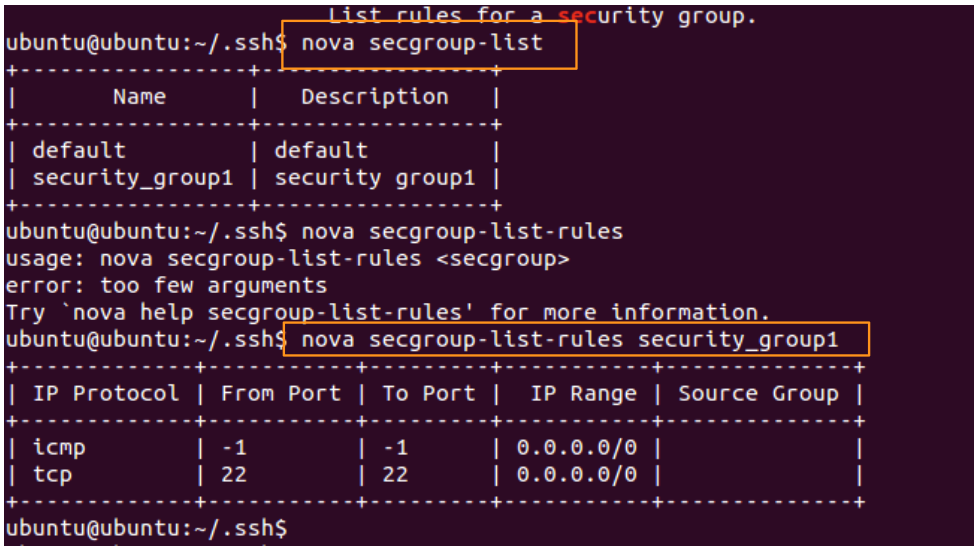
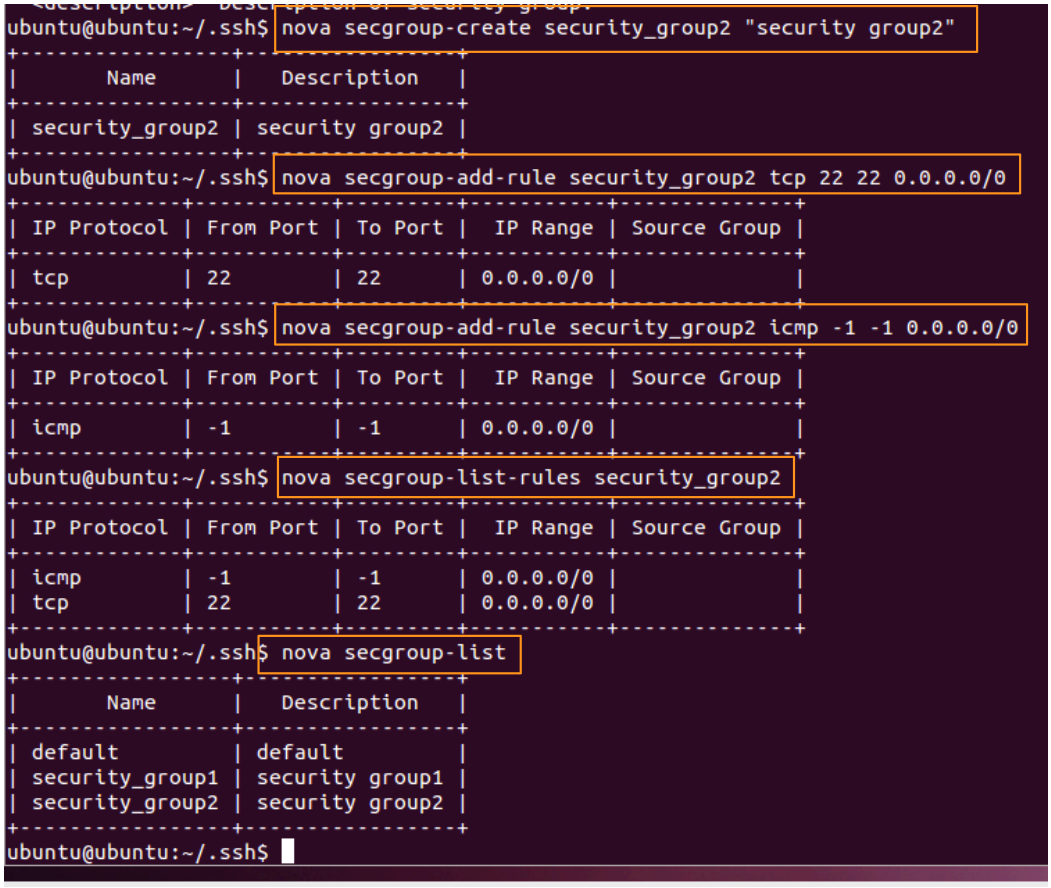


Figure 43: Create New Security Group and add Rules



3.3.5 Instance Management

To list all existing nova instance flavors
nova flavor-list
To list all existing image (s) in a project
nova image-list
To list all existing keypair (s) in a project
nova keypair-list
To list all existing security group(s) in a specific project
nova secgroup-list
To create a new instance within a project
nova boot –flavor <flavor_id> --image <image_id> --keyname <mykey_name> --security_groups <secgroup_name> <Instance_name>
To show details of an instance within a project
nova show <instance_name> or <instance_id>

Figure 44: Flavor list, image-list and secgroup-list

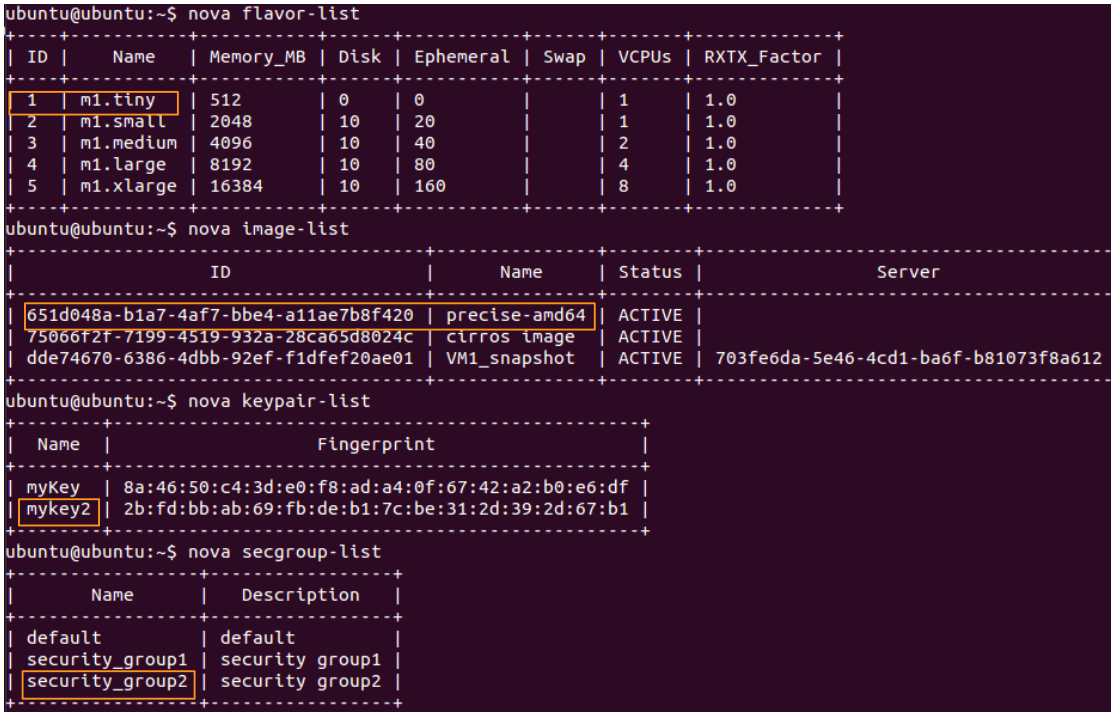


Figure 45: Adding an instance using nova CLI

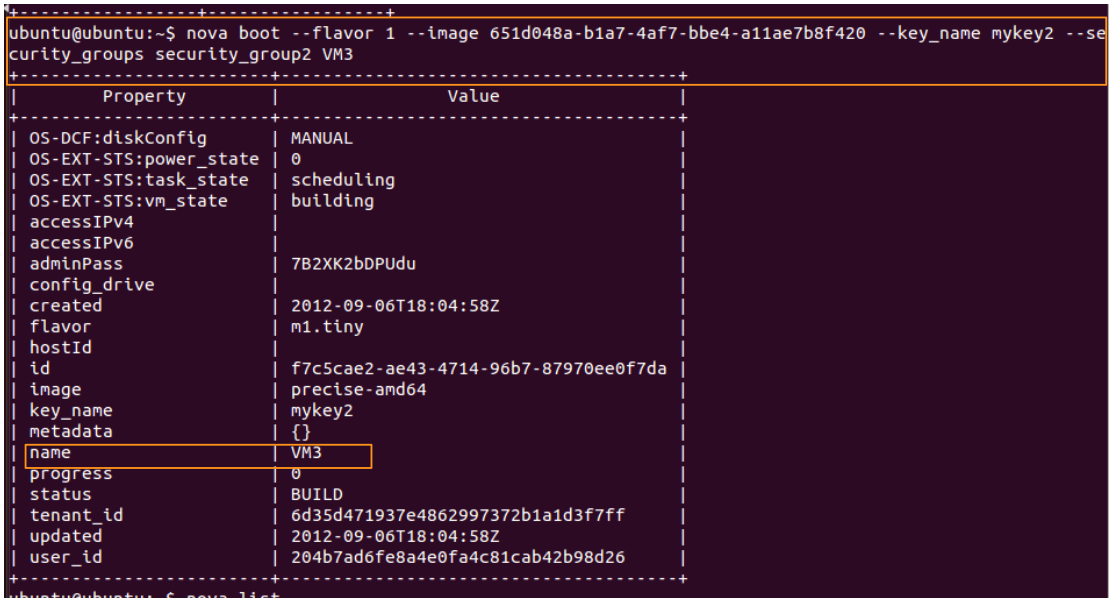
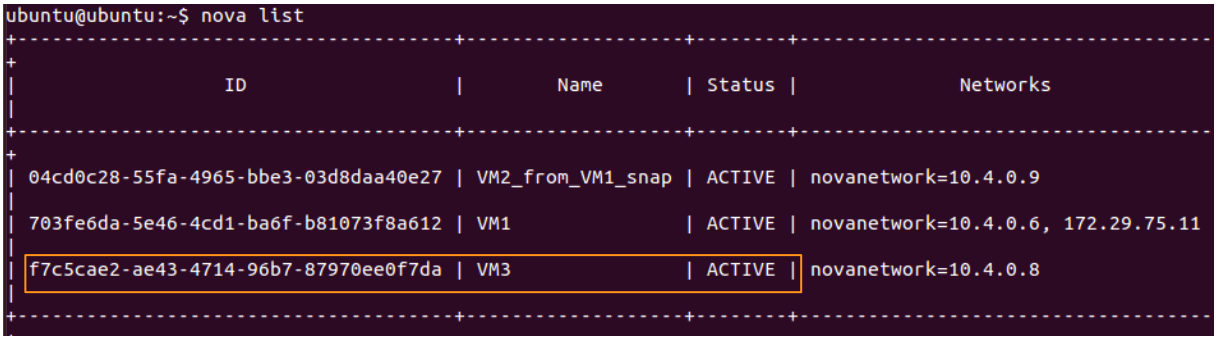


Figure 46: List of existing Instance within a project



3.3.6 Floating IP Management

To list all existing nova floating-ip commands
nova help | grep floating-ip
To list all existing ip pool list (s) in a project
nova floating-ip-pool-list
To list all existing ip pool in a project
nova floating-ip-list
To create a new floating-ip
nova floating-ip-create
To associate a new floating-ip to an instance
nova add-floating-ip <instance_name> <floating_ip_address>

Figure 47: Create Floating-IP

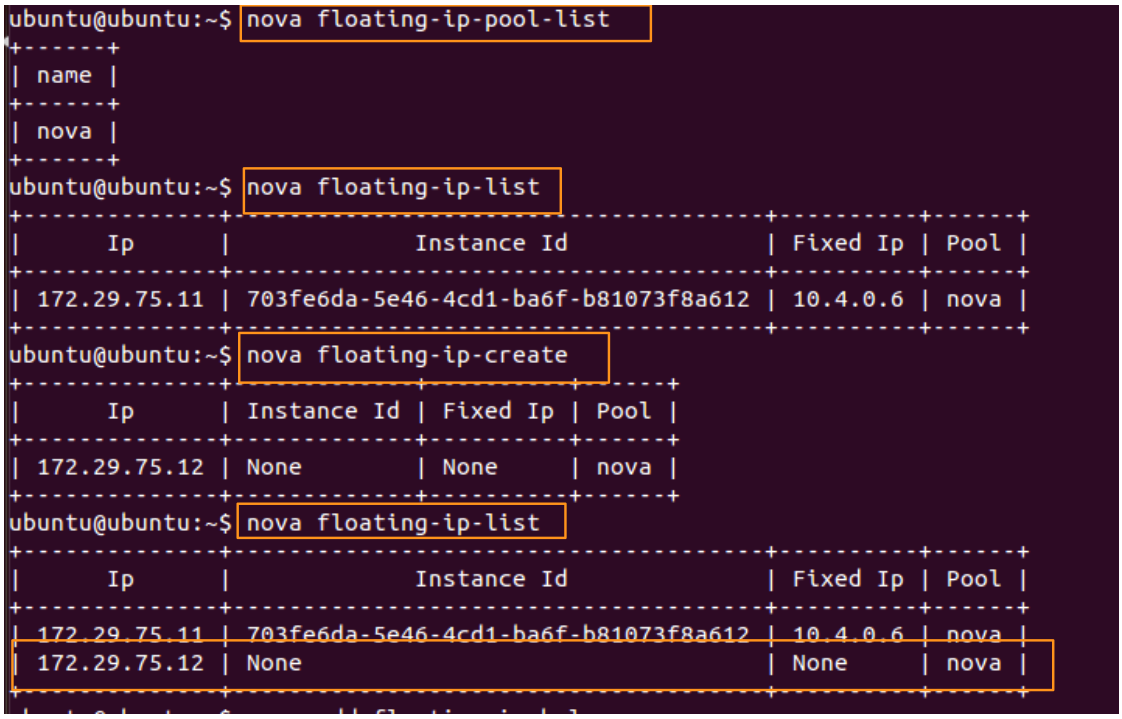
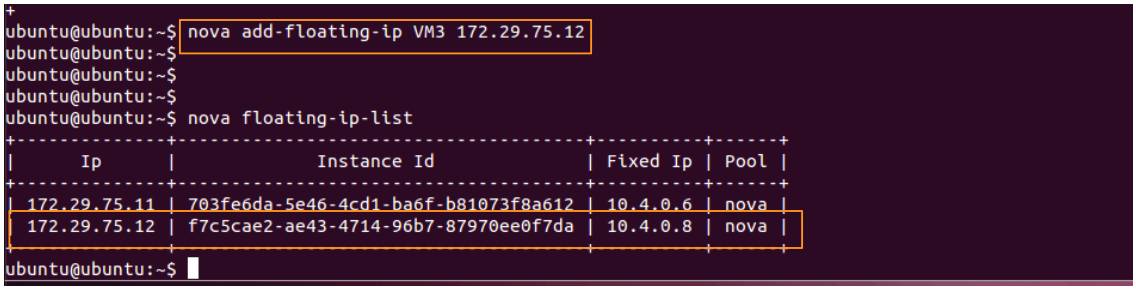


Figure 48: Associate Floating IP to an existing Instance



#To remove floating IP's, first dis-associate floating ip from the server
nova remove-floating-ip <server> <address>

#Then de-allocate the floating ip and return back to the pool
nova floating-ip-delete <address>

3.3.7 Accessing Instance using Floating IP

Verify the floating IP is associated with the correct Instance and the VM is currently active using "*nova show <instance_name>* "

Figure 49: Accessing Instance using Floating IP

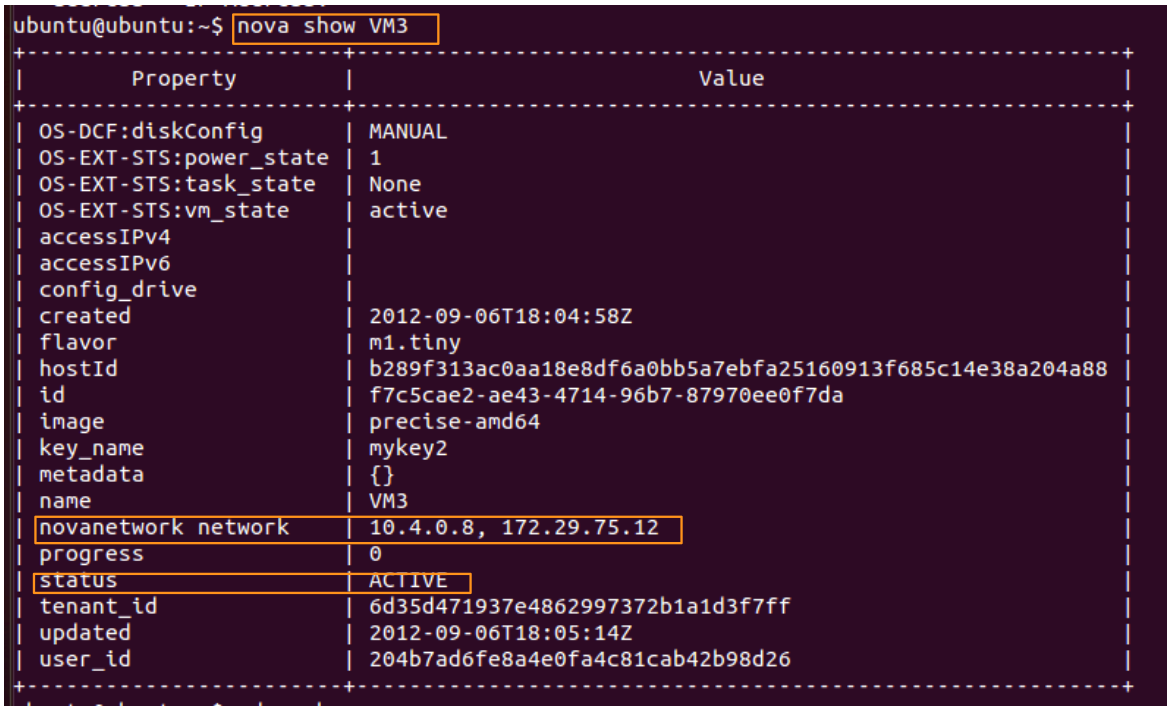
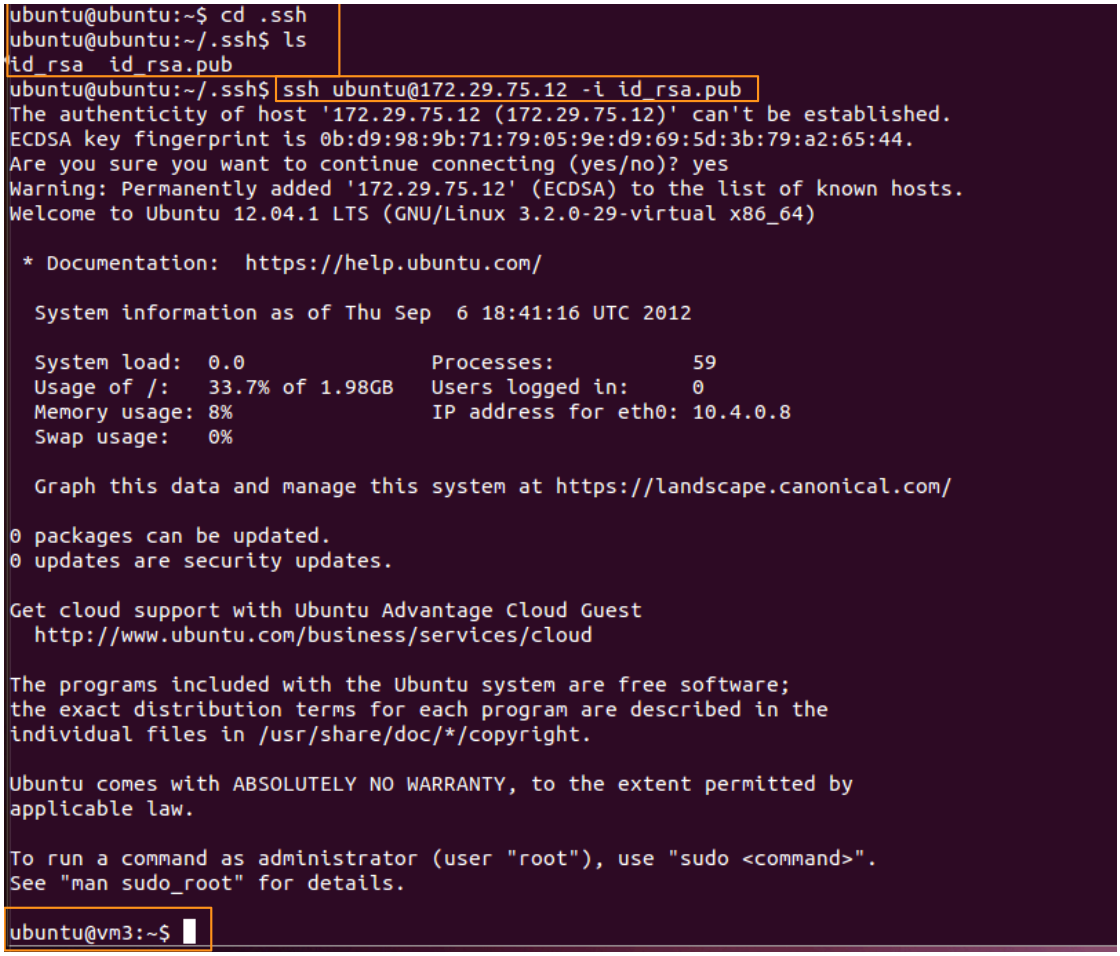


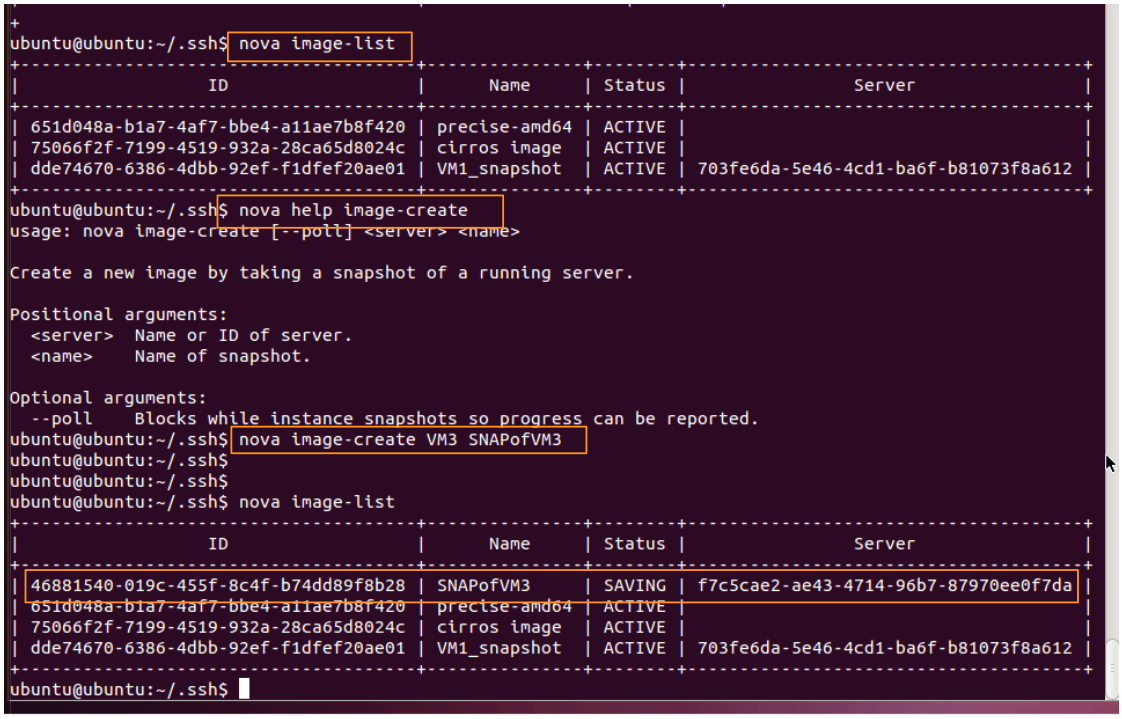
Figure 50: SSH to new VM using floating-ip



3.3.8 Instance Snapshots

To list all existing nova snapshot commands
nova help | grep snap
To list all existing snap shot list (s) in a project
nova image-list
To create a snap shot of an existing instance
nova image-create <instance_name> <snapshot_name>

Figure 51: Creating a Snapshot of an Instance

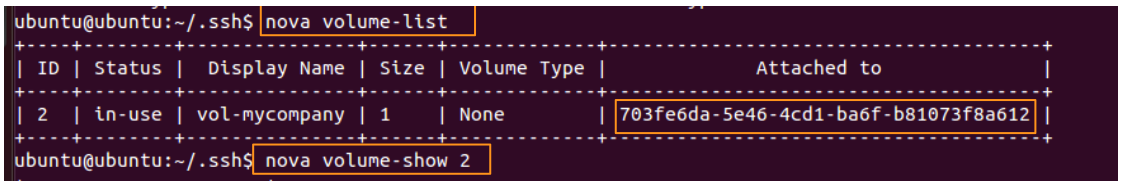


This snapshot image can be used to create a new instance within this project.

3.3.9 Nova Volumes

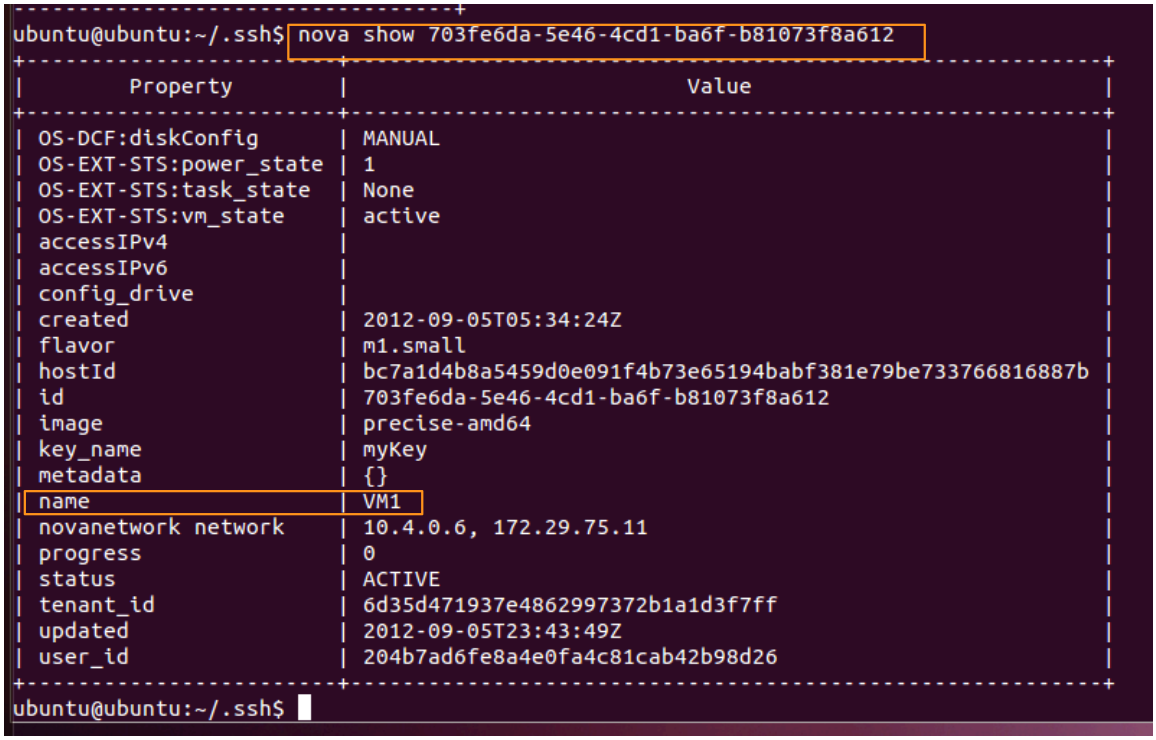
To list all existing nova volume commands
nova help | grep volume
To list all existing volume list (s) in a project
nova volume-list
To show volume details
nova volume-show <volume_id>

Figure 52: Nova Volume List



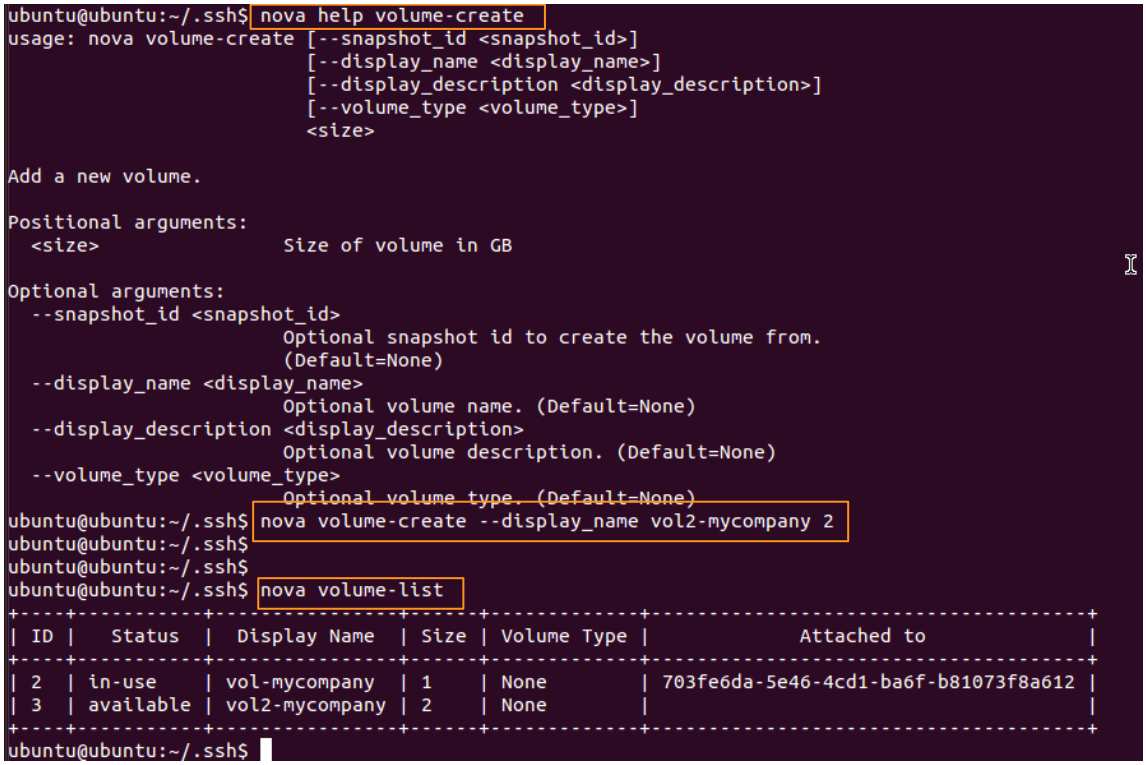
To find out which VM this volume is attached to
nova show <instance_id>

Figure 53: Nova Volume Attached to Instance



To create a new volume
nova volume-create –display_name <volume_name> <size>

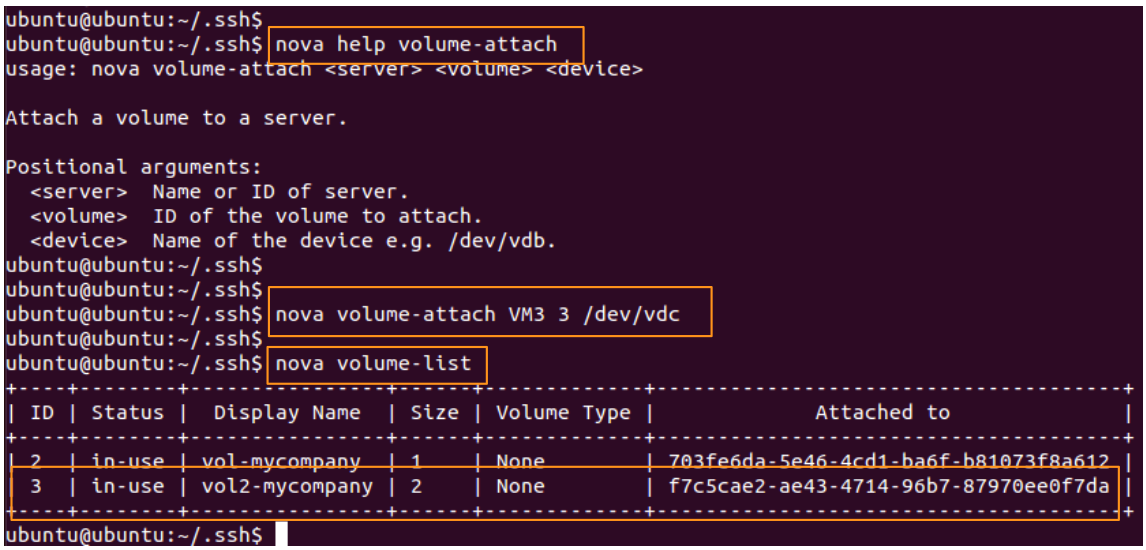
Figure 54: Create a new Nova Volume



To attach a new volume to a particular instance

nova volume-attach <*instance_name*> <*volume_id*> <*name_of_device*>

Figure 55: Attach New Nova Volume to Instance



3.3.10 Swift Containers

The

3.4 Image Management

This section will outlines the steps to add images, list images and delete images.
The last two steps, list and delete images, are possible to do in the OpenStack Horizon dashboard. However, the current version of the Horizon dashboard in OpenStack Essex release does not support adding an image directly. This section will demonstrate how to accomplish all three functions (add, list and delete images) using OpenStack Glance client.

3.4.1 Adding Images

Images are files containing information about a virtual disk previously formatted and configured. Uploading images to the Image Service Glance can only be done through the command line client. Glance client needs to be installed in the user system and the user credentials file (openrc) should be sourced in order to execute any of the command herein mentioned. These steps are covered in Sections 3.1 and 3.2.

General Glance import options are:

```
# glance add name=<A name for the created image> \
is_public= <Optional, remove this switch for private images) \
disk_format= <ami|aki|ari|qcow2|raw> \
container_format= <ami|aki|ari|qcow2|bare> \
< (Your downloaded image path)
```

Two formats are supported by Glance:

- AMI (Amazon Machine Image)
- QCOW (Qemu Copy On Write)
- VMDK (Virtual Machine Disk) only in the "Monolithic File Flat" format. http://www.vmware.com/pdf/VirtualDiskManager.pdf

Operating system images may be downloaded from a variety of free sources on the Internet or from your EC2 account.
Once you have your image, the first step is to identify the type of image you want to import. This information is usually included in the place you downloaded the image from. In case you have a raw iso image and want to convert it to a qcow image, you can take a look at the oz image builder (https://github.com/rackerjoe/oz-image-build). The repository also features links to various qcow2 images.
Once you have identified the image, follow the instructions below appropriate for your image type.

3.4.2 Adding AMI (Amazon Machine Image) Images

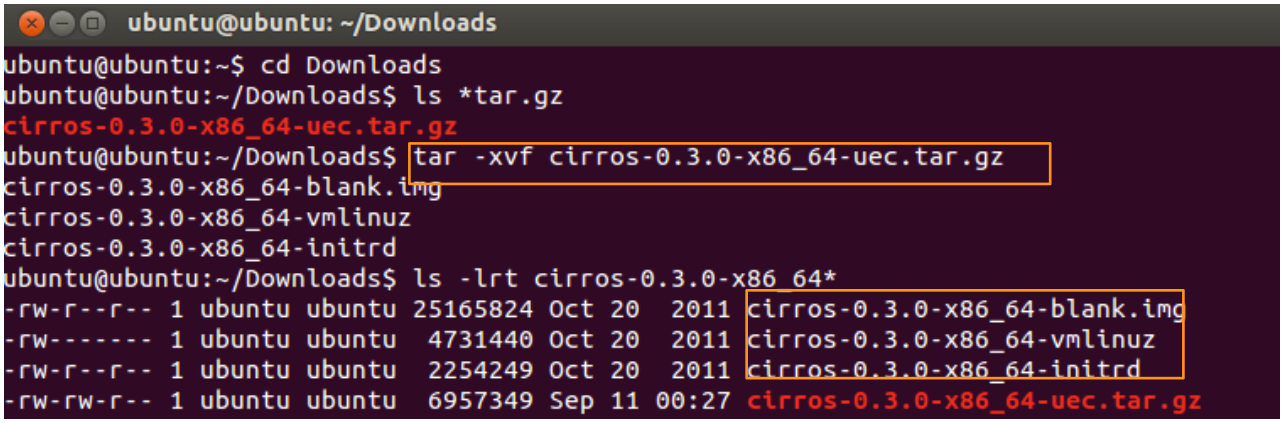
For the purpose of this install guide, test images are downloaded from this site;
https://launchpad.net/cirros/+download

STEP1: Download the AMI image from the link above and extract the contents:

%cd downloads

%tar -xvf **cirros-0.3.0-x86_64-uec.tar.gz**

Figure 56: AMI Image Download



Using The files required for the AMI format are the kernel (vmlinuz), ramdisk (initrd) and file disk (.img). The following steps will upload a test image known as Cirros. Cirros Image Files:

cirros-0.3.0-x86_64-blank.img
cirros-0.3.0-x86_64-initrd
cirros-0.3.0-x86_64-vmlinuz

From the Tenant Linux System, execute the following commands:

STEP2: Modify openrc and add the following entry (see Section 3.2). Make sure to source the openrc after the change.

export GLANCE_HOSTPORT=172.29.74.193

%source openrc

STEP3: Add AMI kernel image

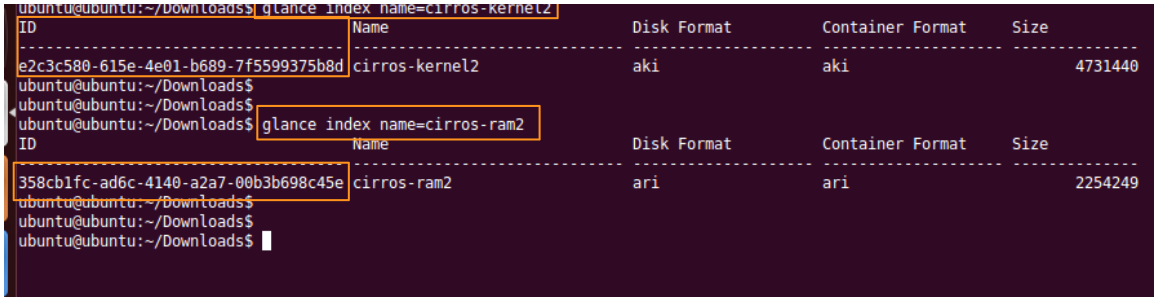
% glance --url http://$GLANCE_HOSTPORT add name="cirros-kernel2" is_public=true container_format=aki disk_format=aki < "cirros-0.3.0-x86_64-vmlinuz"

STEP4: Add AMI ram image

glance --url http://$GLANCE_HOSTPORT add name="cirros-ram2 is_public=true container_format=ari disk_format=ari < "cirros-0.3.0-x86_64-initrd"

STEP5: Verify AMI Kernel and RAM Image upload

Figure 57: Verify AMI Kernel and RAM Image upload



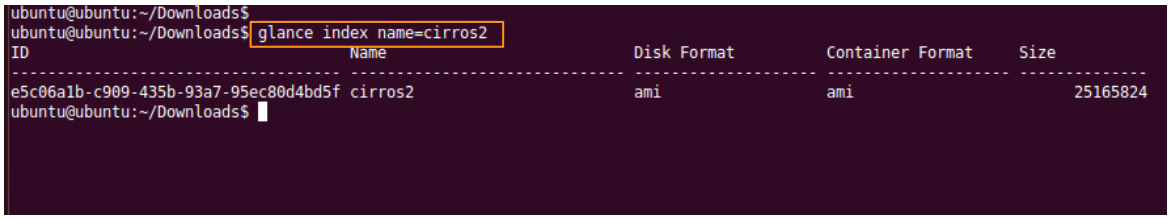
STEP6: Add AMI file disk

glance --url http://$GLANCE_HOSTPORT add name="cirros" is_public=true container_format=ami disk_format=ami kernel_id=<kernel_id> ramdisk_id=<ramdisk_id> < "cirros-0.3.0-x86_64-blank.img"

Use the AMI kernel_id and ramdisk_id in STEP5.

STEP6: Verify AMI image Upload

Figure 58: Verify AMI Image Upload



You can also verify from the Horizon dashboard that the image "cirros2" was added in the list of images available for MyCompany Tenant.

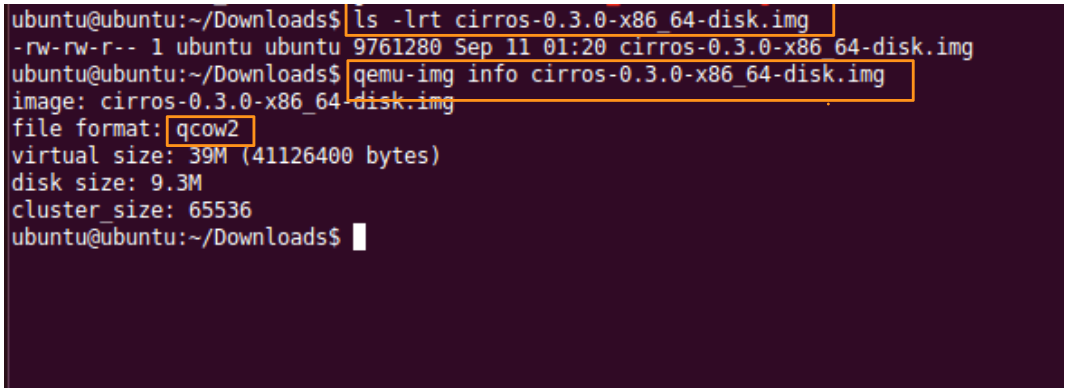
3.4.3 Adding QCOW (Qemo Copy on Write) Images

STEP1: Download the QCOW image from the link below:

https://launchpad.net/cirros/+download

%cd downloads
%qemu-img info cirros-0.3.0-x86_64-disk.img

Figure 59: QCOW Image download

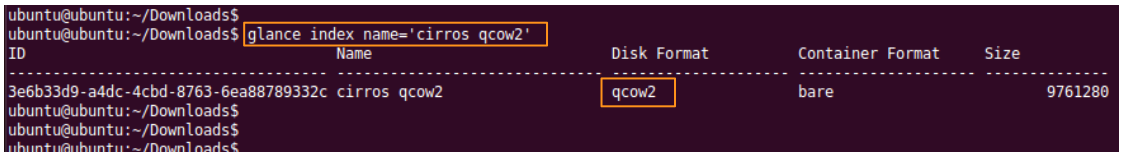


STEP2: Add QCOW2 image. These images do not require any kernel or ramdisk images, you can simply import them as:

glance --url http://$GLANCE_HOSTPORT add name="cirros qcow2" is_public=true container_format=bare disk_format=qcow2 < "cirros-0.3.0-x86_64-disk.img"

STEP3: Verify QCOW2 image upload:

Figure 60: QCOW2 Image upload



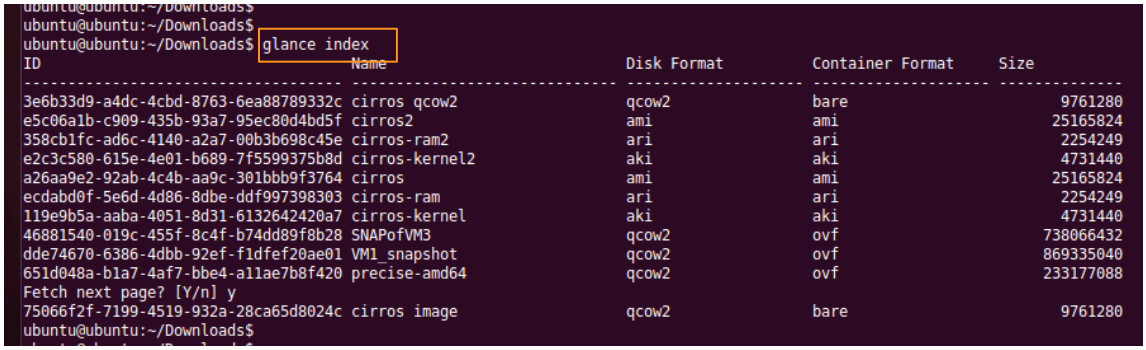
3.4.4 Listing Images

The list of all the available images that a tenant can use can be retrieved using the following methods;

1/ Horizon dashboard as explained in Section 2.5
2/ Using Glance client
3/ Using Nova Client

The following command will display the available images using glance client:

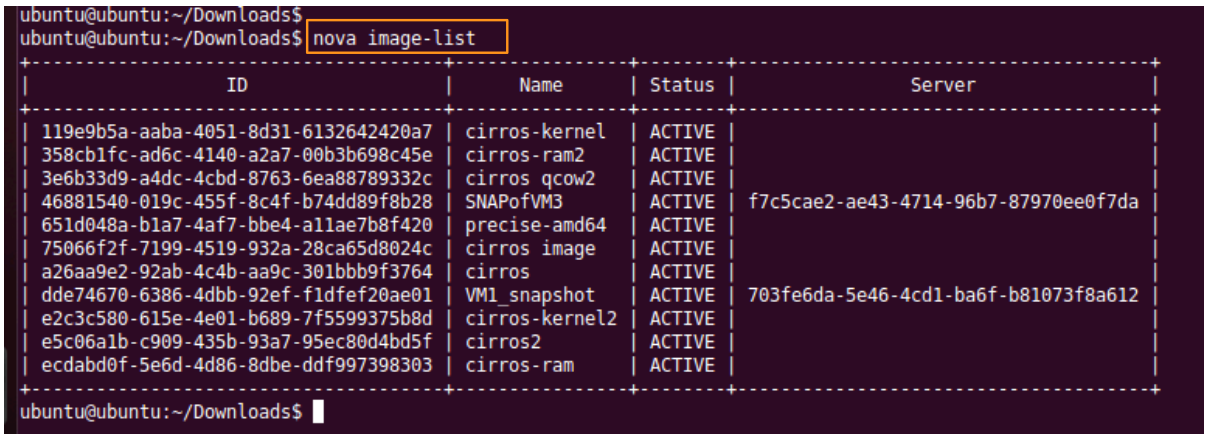
Figure 61: Glance Image List



Note, the list above includes all images that was created by the Cloud admin and initially assigned to the "mycompany" tenant (e.g. "precise-amd64") , the snapshot images created as part of Section 2.8 / Section 3.3.8 (e.g. "VM1_snapshot") and the AMI and QCOW2 image created in previous two sections 3.4.2 and 3.4.3.

Images can also be listed using nova client;

Figure 62: Nova Image List



3.4.5 Deleting Images

Images can be deleted using the same three methods used in viewing them; Horizon Dashboard, glance client and nova client.

Figure 63: Deleteing Images Using Dashboard

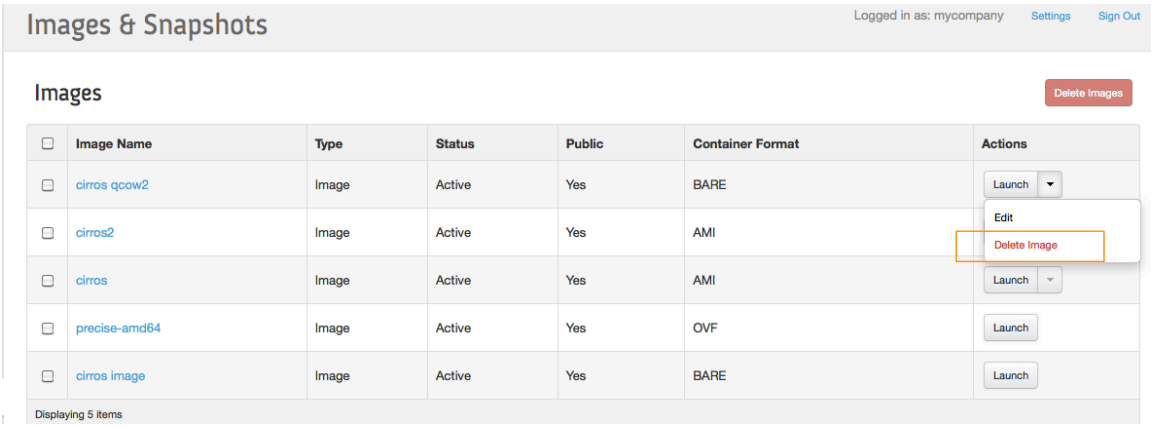


Figure 64: Delete an image using Glance CLI

```
ubuntu@ubuntu:~/Downloads$ glance help delete

glance delete [options] <ID>

Deletes an image from Glance
ubuntu@ubuntu:~/Downloads$ glance index name='cirros qcow2'
ID                                   Name                           Disk Format          Container Format     Size
------------------------------------ ------------------------------ -------------------- -------------------- -------------
3e6b33d9-a4dc-4cbd-8763-6ea88789332c cirros qcow2                   qcow2                bare                      9761280
ubuntu@ubuntu:~/Downloads$
ubuntu@ubuntu:~/Downloads$
ubuntu@ubuntu:~/Downloads$
ubuntu@ubuntu:~/Downloads$ glance delete 3e6b33d9-a4dc-4cbd-8763-6ea88789332c
Delete image 3e6b33d9-a4dc-4cbd-8763-6ea88789332c? [y/N] y
Deleted image 3e6b33d9-a4dc-4cbd-8763-6ea88789332c
ubuntu@ubuntu:~/Downloads$
ubuntu@ubuntu:~/Downloads$
ubuntu@ubuntu:~/Downloads$ glance index name='cirros qcow2'
ubuntu@ubuntu:~/Downloads$
```

Figure 65: Delete an image using nova CLI

```
ubuntu@ubuntu:~/Downloads$ nova image-list
+--------------------------------------+---------------+--------+--------------------------------------+
|                  ID                  |      Name     | Status |                Server                |
+--------------------------------------+---------------+--------+--------------------------------------+
| 119e9b5a-aaba-4051-8d31-6132642420a7 | cirros-kernel | ACTIVE |                                      |
| 358cb1fc-ad6c-4140-a2a7-00b3b698c45e | cirros-ram2   | ACTIVE |                                      |
| 46881540-019c-455f-8c4f-b74dd89f8b28 | SNAPofVM3     | ACTIVE | f7c5cae2-ae43-4714-96b7-87970ee0f7da |
| 651d048a-b1a7-4af7-bbe4-a11ae7b8f420 | precise-amd64 | ACTIVE |                                      |
| 75066f2f-7199-4519-932a-28ca65d8024c | cirros image  | ACTIVE |                                      |
| a26aa9e2-92ab-4c4b-aa9c-301bbb9f3764 | cirros        | ACTIVE |                                      |
| dde74670-6386-4dbb-92ef-f1dfef20ae01 | VM1_snapshot  | ACTIVE | 703fe6da-5e46-4cd1-ba6f-b81073f8a612 |
| e2c3c580-615e-4e01-b689-7f5599375b8d | cirros-kernel2| ACTIVE |                                      |
| e5c06a1b-c909-435b-93a7-95ec80d4bd5f | cirros2       | ACTIVE |                                      |
| ecdabd0f-5e6d-4d86-8dbe-ddf997398303 | cirros-ram    | ACTIVE |                                      |
+--------------------------------------+---------------+--------+--------------------------------------+
ubuntu@ubuntu:~/Downloads$ nova help image-delete
usage: nova image-delete <image>

Delete an image. It should go without saying, but you can only delete images
you created.

Positional arguments:
  <image>  Name or ID of image.
ubuntu@ubuntu:~/Downloads$ nova image-delete e5c06a1b-c909-435b-93a7-95ec80d4bd5f
ubuntu@ubuntu:~/Downloads$
ubuntu@ubuntu:~/Downloads$
```

4.0 Appendix

4.1 Glossary

CIMC	- Cisco Integrated Management Controller
RDP	- Remote Desktop Protocol
KVM(Console)	-Keyboard Video and Mouse
KVM(Hypervisor)	- Open Source KVM hypervisor
LTS(Ubuntu)	-Long Term Support
FCOE	-Fiber-channel Over Ethernet
LOM	-LAN on Motherboard
UCS B Series	- UCS Chassis Blade Servers
UCS C Series	- UCS Rack Based C Series Servers

4.2 Caveats

Caveats and Technical notes if needed

4.3 References

[1] Cisco-OpenStack Installation Guide
[2] http://glance.openstack.org/glanceapi.html
[3] OpenStack Essex Administration Guides (http://docs.openstack.org/)
[4] OpenStack Essex API Guides (http://docs.openstack.org/api/)
[5] OpenStack Essex Developer Documentation (http://docs.openstack.org/developer/)