

# OpenStack Training Guides

Copyright © 2013 OpenStack Foundation Some rights reserved.

Licensed under the Apache License, Version 2.0 (the "License"); you may not use this file except in compliance with the License. You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software distributed under the License is distributed on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the License for the specific language governing permissions and limitations under the License.



Except where otherwise noted, this document is licensed under  
**Creative Commons Attribution ShareAlike 3.0 License.**

<http://creativecommons.org/licenses/by-sa/3.0/legalcode>

2014-04-26

OpenStack™ Training Guides offer the open source community software training for cloud administration and management for any organization.



# Table of Contents

|                                                           |    |
|-----------------------------------------------------------|----|
| Start Here .....                                          | i  |
| Preface .....                                             | 7  |
| Document change history .....                             | 7  |
| A. OpenStack Training Guides Are Under Construction ..... | 1  |
| B. Building the Training Cluster .....                    | 5  |
| Important Terms .....                                     | 5  |
| Building the Training Cluster, Scripted .....             | 6  |
| Building the Training Cluster, Manually .....             | 7  |
| C. Community support .....                                | 47 |
| Documentation .....                                       | 47 |
| ask.openstack.org .....                                   | 49 |
| OpenStack mailing lists .....                             | 49 |
| The OpenStack wiki .....                                  | 50 |
| The Launchpad Bugs area .....                             | 50 |
| The OpenStack IRC channel .....                           | 51 |
| Documentation feedback .....                              | 52 |
| OpenStack distribution packages .....                     | 52 |
| Associate Training Guide .....                            | i  |
| 1. Getting Started .....                                  | 1  |
| Day 1, 09:00 to 11:00 .....                               | 1  |
| Overview .....                                            | 1  |
| Introduction Text .....                                   | 2  |
| Brief Overview .....                                      | 4  |
| Core Projects .....                                       | 7  |
| OpenStack Architecture .....                              | 21 |
| Virtual Machine Provisioning Walk-Through .....           | 33 |
| 2. Getting Started Quiz .....                             | 41 |
| Day 1, 10:40 to 11:00 .....                               | 41 |

|                                             |     |
|---------------------------------------------|-----|
| 3. Controller Node .....                    | 45  |
| Day 1, 11:15 to 12:30, 13:30 to 14:45 ..... | 45  |
| Overview Horizon and OpenStack CLI .....    | 45  |
| Keystone Architecture .....                 | 95  |
| OpenStack Messaging and Queues .....        | 100 |
| Administration Tasks .....                  | 111 |
| 4. Controller Node Quiz .....               | 149 |
| Day 1, 14:25 to 14:45 .....                 | 149 |
| 5. Compute Node .....                       | 155 |
| Day 1, 15:00 to 17:00 .....                 | 155 |
| VM Placement .....                          | 155 |
| VM provisioning in-depth .....              | 163 |
| OpenStack Block Storage .....               | 167 |
| Administration Tasks .....                  | 172 |
| 6. Compute Node Quiz .....                  | 317 |
| Day 1, 16:40 to 17:00 .....                 | 317 |
| 7. Network Node .....                       | 319 |
| Day 2, 09:00 to 11:00 .....                 | 319 |
| Networking in OpenStack .....               | 319 |
| OpenStack Networking Concepts .....         | 325 |
| Administration Tasks .....                  | 327 |
| 8. Network Node Quiz .....                  | 463 |
| Day 2, 10:40 to 11:00 .....                 | 463 |
| 9. Object Storage Node .....                | 465 |
| Day 2, 11:30 to 12:30, 13:30 to 14:45 ..... | 465 |
| Introduction to Object Storage .....        | 465 |
| Features and Benefits .....                 | 466 |
| Administration Tasks .....                  | 467 |
| 10. Object Storage Node Quiz .....          | 477 |
| Day 2, 14:25 to 14:45 .....                 | 477 |
| 11. Assessment .....                        | 479 |

|                                                                   |     |
|-------------------------------------------------------------------|-----|
| Day 2, 15:00 to 16:00 .....                                       | 479 |
| Questions .....                                                   | 479 |
| 12. Review of Concepts .....                                      | 481 |
| Day 2, 16:00 to 17:00 .....                                       | 481 |
| Operator Training Guide .....                                     | i   |
| 1. Getting Started .....                                          | 1   |
| Day 1, 09:00 to 11:00, 11:15 to 12:30 .....                       | 1   |
| Overview .....                                                    | 1   |
| Review Associate Introduction .....                               | 2   |
| Review Associate Brief Overview .....                             | 4   |
| Review Associate Core Projects .....                              | 7   |
| Review Associate OpenStack Architecture .....                     | 21  |
| Review Associate Virtual Machine Provisioning Walk-Through .....  | 33  |
| 2. Getting Started Lab .....                                      | 41  |
| Day 1, 13:30 to 14:45, 15:00 to 17:00 .....                       | 41  |
| Getting the Tools and Accounts for Committing Code .....          | 41  |
| Fix a Documentation Bug .....                                     | 45  |
| Submit a Documentation Bug .....                                  | 49  |
| Create a Branch .....                                             | 49  |
| Optional: Add to the Training Guide Documentation .....           | 51  |
| 3. Getting Started Quiz .....                                     | 53  |
| Day 1, 16:40 to 17:00 .....                                       | 53  |
| 4. Controller Node .....                                          | 55  |
| Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30 .....                  | 55  |
| Review Associate Overview Horizon and OpenStack CLI .....         | 55  |
| Review Associate Keystone Architecture .....                      | 105 |
| Review Associate OpenStack Messaging and Queues .....             | 110 |
| Review Associate Administration Tasks .....                       | 121 |
| 5. Controller Node Lab .....                                      | 123 |
| Days 2 to 4, 13:30 to 14:45, 15:00 to 16:30, 16:45 to 18:15 ..... | 123 |
| Control Node Lab .....                                            | 123 |

|                                                       |     |
|-------------------------------------------------------|-----|
| 6. Controller Node Quiz .....                         | 143 |
| Days 2 to 4, 16:40 to 17:00 .....                     | 143 |
| 7. Network Node .....                                 | 145 |
| Days 7 to 8, 09:00 to 11:00, 11:15 to 12:30 .....     | 145 |
| Review Associate Networking in OpenStack .....        | 145 |
| Review Associate OpenStack Networking Concepts .....  | 151 |
| Review Associate Administration Tasks .....           | 153 |
| Operator OpenStack Neutron Use Cases .....            | 153 |
| Operator OpenStack Neutron Security .....             | 163 |
| Operator OpenStack Neutron Floating IPs .....         | 165 |
| 8. Network Node Lab .....                             | 167 |
| Days 7 to 8, 13:30 to 14:45, 15:00 to 17:00 .....     | 167 |
| Network Node Lab .....                                | 167 |
| 9. Network Node Quiz .....                            | 175 |
| Days 7 to 8, 16:40 to 17:00 .....                     | 175 |
| 10. Compute Node .....                                | 177 |
| Days 5 to 6, 09:00 to 11:00, 11:15 to 12:30 .....     | 177 |
| Review Associate VM Placement .....                   | 177 |
| Review Associate VM Provisioning Indepth .....        | 185 |
| Review Associate OpenStack Block Storage .....        | 189 |
| Review Associate Administration Tasks .....           | 194 |
| 11. Compute Node Lab .....                            | 195 |
| Days 5 to 6, 13:30 to 14:45, 15:00 to 17:00 .....     | 195 |
| Compute Node Lab .....                                | 195 |
| 12. Compute Node Quiz .....                           | 205 |
| Days 5 to 6, 16:40 to 17:00 .....                     | 205 |
| 13. Object Storage Node .....                         | 207 |
| Day 9, 09:00 to 11:00, 11:15 to 12:30 .....           | 207 |
| Review Associate Introduction to Object Storage ..... | 207 |
| Review Associate Features and Benefits .....          | 208 |
| Review Associate Administration Tasks .....           | 209 |

|                                                                 |     |
|-----------------------------------------------------------------|-----|
| Object Storage Capabilities .....                               | 209 |
| Object Storage Building Blocks .....                            | 211 |
| Swift Ring Builder .....                                        | 222 |
| More Swift Concepts .....                                       | 225 |
| Swift Cluster Architecture .....                                | 229 |
| Swift Account Reaper .....                                      | 233 |
| Swift Replication .....                                         | 234 |
| 14. Object Storage Node Lab .....                               | 237 |
| Day 9, 13:30 to 14:45, 15:00 to 17:00 .....                     | 237 |
| Installing Object Node .....                                    | 237 |
| Configuring Object Node .....                                   | 239 |
| Configuring Object Proxy .....                                  | 242 |
| Start Object Node Services .....                                | 247 |
| Developer Training Guide .....                                  | i   |
| 1. Getting Started .....                                        | 1   |
| Day 1, 09:00 to 11:00, 11:15 to 12:30 .....                     | 1   |
| Overview .....                                                  | 1   |
| Review Operator Introduction .....                              | 2   |
| Review Operator Brief Overview .....                            | 4   |
| Review Operator Core Projects .....                             | 7   |
| Review Operator OpenStack Architecture .....                    | 21  |
| Review Operator Virtual Machine Provisioning Walk-Through ..... | 33  |
| 2. Getting Started Lab .....                                    | 41  |
| Day 1, 13:30 to 14:45, 15:00 to 17:00 .....                     | 41  |
| Getting the Tools and Accounts for Committing Code .....        | 41  |
| Fix a Documentation Bug .....                                   | 45  |
| Submit a Documentation Bug .....                                | 49  |
| Create a Branch .....                                           | 49  |
| Optional: Add to the Training Guide Documentation .....         | 51  |
| 3. Getting Started Quiz .....                                   | 53  |
| Day 1, 16:40 to 17:00 .....                                     | 53  |

|                                                       |    |
|-------------------------------------------------------|----|
| 4. Developer APIs in Depth .....                      | 55 |
| Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30 .....      | 55 |
| 5. Developer APIs in Depth Lab Day Two .....          | 57 |
| Day 2, 13:30 to 14:45, 15:00 to 16:30 .....           | 57 |
| 6. Developer APIs in Depth Day Two Quiz .....         | 59 |
| Day 2, 16:40 to 17:00 .....                           | 59 |
| 7. Developer APIs in Depth Lab Day Three .....        | 61 |
| Day 3, 13:30 to 14:45, 15:00 to 16:30 .....           | 61 |
| 8. Developer APIs in Depth Day Three Quiz .....       | 63 |
| Day 3, 16:40 to 17:00 .....                           | 63 |
| 9. Developer How To Participate Lab Day Four .....    | 65 |
| Day 4, 13:30 to 14:45, 15:00 to 16:30 .....           | 65 |
| 10. Developer APIs in Depth Day Four Quiz .....       | 67 |
| Day 4, 16:40 to 17:00 .....                           | 67 |
| 11. Developer How To Participate .....                | 69 |
| Day 5 to 9, 09:00 to 11:00, 11:15 to 12:30 .....      | 69 |
| 12. Developer How To Participate Lab Day Five .....   | 71 |
| Day 5, 13:30 to 14:45, 15:00 to 16:30 .....           | 71 |
| 13. Developer How To Participate Day Five Quiz .....  | 73 |
| Day 5, 16:40 to 17:00 .....                           | 73 |
| 14. Developer How To Participate Lab Day Six .....    | 75 |
| Day 6, 13:30 to 14:45, 15:00 to 16:30 .....           | 75 |
| 15. Developer How To Participate Day Six Quiz .....   | 77 |
| Day 6, 16:40 to 17:00 .....                           | 77 |
| 16. Developer How To Participate Lab Day Seven .....  | 79 |
| Day 7, 13:30 to 14:45, 15:00 to 16:30 .....           | 79 |
| 17. Developer How To Participate Day Seven Quiz ..... | 81 |
| Day 7, 16:40 to 17:00 .....                           | 81 |
| 18. Developer How To Participate Lab Day Eight .....  | 83 |
| Day 8, 13:30 to 14:45, 15:00 to 16:30 .....           | 83 |
| 19. Developer How To Participate Day Eight Quiz ..... | 85 |



|                                                                                          |    |
|------------------------------------------------------------------------------------------|----|
| Day 8, 16:40 to 17:00 .....                                                              | 85 |
| 20. Developer How To Participate Lab Day Nine .....                                      | 87 |
| Day 9, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 87 |
| 21. Developer How To Participate Day Nine Quiz .....                                     | 89 |
| Day 9, 16:40 to 17:00 .....                                                              | 89 |
| 22. Assessment .....                                                                     | 91 |
| Day 10, 9:00 to 11:00, 11:15 to 12:30, hands on lab 13:30 to 14:45, 15:00 to 17:00 ..... | 91 |
| Questions .....                                                                          | 91 |
| 23. Developer How To Participate Bootcamp .....                                          | 93 |
| One Day with Focus on Contribution .....                                                 | 93 |
| Overview .....                                                                           | 93 |
| Morning Classroom 10:00 to 11:15 .....                                                   | 94 |
| Morning Lab 11:30 to 12:30 .....                                                         | 95 |
| Morning Quiz 12:30 to 12:50 .....                                                        | 95 |
| Afternoon Classroom 13:30 to 14:45 .....                                                 | 95 |
| Afternoon Lab 15:00 to 17:00 .....                                                       | 96 |
| Afternoon Quiz 17:00 to 17:20 .....                                                      | 96 |
| Architect Training Guide .....                                                           | i  |
| 1. Architect Training Guide Coming Soon .....                                            | 1  |



## **Start Here**



## Table of Contents

|                                                           |    |
|-----------------------------------------------------------|----|
| Preface .....                                             | 7  |
| Document change history .....                             | 7  |
| A. OpenStack Training Guides Are Under Construction ..... | 1  |
| B. Building the Training Cluster .....                    | 5  |
| Important Terms .....                                     | 5  |
| Building the Training Cluster, Scripted .....             | 6  |
| Building the Training Cluster, Manually .....             | 7  |
| C. Community support .....                                | 47 |
| Documentation .....                                       | 47 |
| ask.openstack.org .....                                   | 49 |
| OpenStack mailing lists .....                             | 49 |
| The OpenStack wiki .....                                  | 50 |
| The Launchpad Bugs area .....                             | 50 |
| The OpenStack IRC channel .....                           | 51 |
| Documentation feedback .....                              | 52 |
| OpenStack distribution packages .....                     | 52 |



## List of Figures

|                                        |    |
|----------------------------------------|----|
| B.1. Network Diagram .....             | 11 |
| B.2. Create Host Only Networks .....   | 13 |
| B.3. Vboxnet0 .....                    | 15 |
| B.4. Vboxnet1 .....                    | 17 |
| B.5. Image: Vboxnet2 .....             | 19 |
| B.6. Create New Virtual Machine .....  | 21 |
| B.7. Adapter1 - Vboxnet0 .....         | 23 |
| B.8. Adapter2 - Vboxnet2 .....         | 25 |
| B.9. Adapter3 - NAT .....              | 27 |
| B.10. Create New Virtual Machine ..... | 29 |
| B.11. Adapter 1 - Vboxnet0 .....       | 31 |
| B.12. Adapter2 - Vboxnet1 .....        | 33 |
| B.13. Adapter3 - Vboxnet2 .....        | 35 |
| B.14. Adapter4 - NAT .....             | 37 |
| B.15. Create New Virtual Machine ..... | 39 |
| B.16. Adapter1 - Vboxnet0 .....        | 41 |
| B.17. Adapter2 - Vboxnet1 .....        | 43 |
| B.18. Adapter3 - NAT .....             | 45 |





# Preface

## Document change history

This version of the guide replaces and obsoletes all previous versions. The following table describes the most recent changes:

| Revision Date      | Summary of Changes                                                                  |
|--------------------|-------------------------------------------------------------------------------------|
| November 4, 2013   | <ul style="list-style-type: none"><li>• major restructure of guides</li></ul>       |
| September 11, 2013 | <ul style="list-style-type: none"><li>• first training guides sprint held</li></ul> |
| August 7, 2013     | <ul style="list-style-type: none"><li>• rough draft published to the web</li></ul>  |
| July 9, 2013       | <ul style="list-style-type: none"><li>• first draft released</li></ul>              |
| June 18, 2013      | <ul style="list-style-type: none"><li>• blueprint created</li></ul>                 |



# Appendix A. OpenStack Training Guides Are Under Construction

We need your help! This is a community driven project to provide the user group community access to OpenStack training materials. We cannot make this work without your help.

There are a few ways to get involved. The easiest way is to use the training guides. Look at the end of each section and you will see the Submit a Bug link. When you find something that can be improved or fixed, submit a bug by clicking on the link.

If you want to get involved with the effort around OpenStack community training, read on, here are the options:

- **Attending a user group using the training materials.** The OpenStack community training started at the SFBay OpenStack User Group. More information on this user group and others using the training guides on the [OpenStack User Groups page](#).
- **Teach / Lead a user group using the training materials.** Awesome! Your experience will not only give you more experience with OpenStack, but you will help some people find new jobs. We have put all the information about [How To Run An OpenStack Hackathon](#) here.
- **Help create the training pages.**
  - We are currently working on creating the Associate Training Guide. It is the first of four training guides. We are using the Install Guide, Administration Guides, Developer Documentation, and Aptira supplied content as the sources for most of the Associate Training Guide. The basic idea is that we use XML include statements to actually use the source content to create new pages. We aim to use as much of the material as possible from existing documentation. By doing this we reuse and improve the existing docs. The topics in the Associate Training Guide are in a bunch of KanBan story board cards. Each card in the

story board represents something that an Associate trainee needs to learn. But first things first, you need to get some basic tools and accounts installed and configured before you can really start.

- **Getting Accounts and Tools:** We can't do this without operators and developers using and creating the content. Anyone can contribute content. You will need the tools to get started. Go to the [Getting Tools and Accounts](#) page.
- **Pick a Card:** Once you have your tools ready to go, you can assign some work to yourself. Go to the [Training Trello/KanBan storyboard](#) and assign a card / user story from the Sprint Backlog to yourself. If you do not have a Trello account, no problem, just create one. Email [seanrob@yahoo-inc.com](mailto:seanrob@yahoo-inc.com) and you will have access.
- **Create the Content:** Each card / user story from the KanBan story board will be a separate chunk of content that you will add to the openstack-manuals repository openstack-training sub-project. [More details on creating training content here.](#)



### Note

Here are more details on committing changes to OpenStack [fixing a documentation bug](#) , [OpenStack Gerrit Workflow](#), [OpenStack Documentation HowTo](#) and , [Git Documentation](#)

More details on the OpenStack Training project.

1. [OpenStack Training Wiki](#) (describes the project in detail)
2. [OpenStack Training blueprint](#)(this is the key project page)
3. [Bi-Weekly SFBay Hackathon meetup page](#)(we discuss project details with all team members)
4. [Bi-Weekly SFBay Hackathon Etherpad](#)(meetup notes)
5. [Core Training Weekly Meeting Agenda](#)(we review project action items here)

6. [Training Trello/KanBan storyboard](#)(we develop high level project action items here)

Submit a bug. Enter the summary as "Training, " with a few words. Be descriptive as possible in the description field. Open the tag pull-down and enter training-manuals.



# Appendix B. Building the Training Cluster

## Table of Contents

|                                               |   |
|-----------------------------------------------|---|
| Important Terms .....                         | 5 |
| Building the Training Cluster, Scripted ..... | 6 |
| Building the Training Cluster, Manually ..... | 7 |

## Important Terms

**Host Operating System (Host).** The operating system that is installed on your laptop or desktop that hosts virtual machines. Commonly referred to as *host OS* or *host*. In short, the machine where your Virtual Box is installed.

**Guest Operating System (Guest).** The operating system that is installed on your Virtual Box Virtual Machine. This virtual instance is independent of the host OS. Commonly referred to as *guest OS* or *guest*.

**Node.** In this context, refers specifically to servers. Each OpenStack server is a node.

**Control Node.** Hosts the database, Keystone (Middleware), and the servers for the scope of the current OpenStack deployment. Acts as the brains behind OpenStack and drives services such as authentication, database, and so on.

**Compute Node.** Has the required Hypervisor (Qemu/KVM) and is your Virtual Machine host.

**Network Node.** Provides Network-as-a-Service and virtual networks for OpenStack.

**Using OpenSSH.** After you set up the network interfaces file, you can switch to an SSH session by using an OpenSSH client to log in remotely to the required server node (Control, Network, Compute). Open a terminal on your host machine. Run the following command:

```
$ ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/u/kim/.ssh/id_rsa): [RETURN]
Enter passphrase (empty for no passphrase): <can be left empty>
Enter same passphrase again: <can be left empty>
Your identification has been saved in /home/user/.ssh/id_rsa.
Your public key has been saved in /home/user/.ssh/id_rsa.pub.
The key fingerprint is:
b7:18:ad:3b:0b:50:5c:e1:da:2d:6f:5b:65:82:94:c5 xyz@example
```

## Building the Training Cluster, Scripted

Extract the scripts locally by downloading and running the scripts tar file.

Currently, only \*/Scripts/ folders content are being tested. Run the ~/Scripts/test\_scripts.sh file to test all scripts at once.

### To test scripts

#### 1. Set up the test environment

To use Virtual Box as test environment, you must attach the following network adapters:

- Host-Only/ Bridged – 10.10.10.51 (Guest) – 10.10.10.xx (Host IP for Host-Only)
- Host-Only/ Bridged – 192.168.100.51 (Guest) – 192.168.100.xx (Host IP for Host-Only)
- Bridged/NAT – DHCP – These Scripts should be run without internet connection after Pre-Install.sh. The Templates/\* should be changed to the required IP Addresses for custom networks.



## 2. Test scripts individually

Run the shell scripts in the `Scripts` folder to verify they run correctly. Do not install Virtual Box, although it is recommended because your host machine might fail.

To test the scripts, run them. Some scripts require input parameters. If you do not want to run them manually, run the `Scripts/test_scripts.sh` file. Virtual Box guest add-ons are not required to test the scripts as units.

## 3. Test the entire system

You must install Virtual Box, Ubuntu Server 12.04 or 13.04, and the Virtual Box guest add-ons.

To install Virtual Box guest add-ons, complete one of these steps:

- Install the Virtual Box guest add-ons through ISO:

```
# apt-get install linux-headers-generic
```

```
# mount /dev/cdrom0/ /tmp/cdrom
```

```
#cd /tmp/cdrom/
```

```
# ./virtualbox
```

- Install the Virtual Box guest add-ons through Ubuntu repositories:

```
# apt-get install linux-headers-generic
```

```
# apt-get --no-install-recommends install virtualbox-guest-additions
```

# Building the Training Cluster, Manually

## Getting Started

The following are the conventional methods of deploying OpenStack on Virtual Box for the sake of a test/sandbox or just to try out OpenStack on commodity hardware.

1. DevStack

2. Vagrant

But DevStack and Vagrant bring in some level of automated deployment as running the scripts will get your VirtualBox Instance configured as the required OpenStack deployment. We will be manually deploying OpenStack on VirtualBox Instance to get better view of how OpenStack works.

**Prerequisite:**

Well, its a daunting task to just cover all of OpenStack's concepts let alone Virtualization and Networking. So some basic idea/knowledge on Virtualization, Networking and Linux is required. Even though I will try to keep the level as low as possible for making it easy for Linux Newbies as well as experts.

These Virtual Machines and Virtual Networks will be given equal privilege as a physical machine on a physical network.

Just for those who would want to do a deeper research or study, for more information you may refer the following links

**OpenStack:**OpenStack Official Documentation ([docs.openstack.org](http://docs.openstack.org))

**Networking:**Computer Networks (5th Edition) by Andrew S. Tanenbaum

**VirtualBox:**Virtual Box Manual (<http://www.virtualbox.org/manual/UserManual.html>)

**Requirements :**

Operating Systems - I recommend Ubuntu Server 12.04 LTS, Ubuntu Server 13.10 or Debian Wheezy

**Note :**Ubuntu 12.10 is not supporting OpenStack Grizzly Packages. Ubuntu team has decided not to package Grizzly Packages for Ubuntu 12.10.

- Recommended Requirements.

|                |                          |
|----------------|--------------------------|
| VT Enabled PC: | Intel ix or AMD QuadCore |
| 4 GB RAM:      | DDR2/DDR3                |

- Minimum Requirements.

|              |                                   |
|--------------|-----------------------------------|
| Non-VT PC's: | Intel Core 2 Duo or Amd Dual Core |
| 2GB Ram:     | DDR2/DDR3                         |

If you don't know whether your processor is VT enabled, you could check it by installing cpu checker

```
# apt-get install cpu-checker
# kvm-ok
```

If your device does not support VT it will show

INFO:Your CPU does not support KVM extensions

KVM acceleration can NOT be used

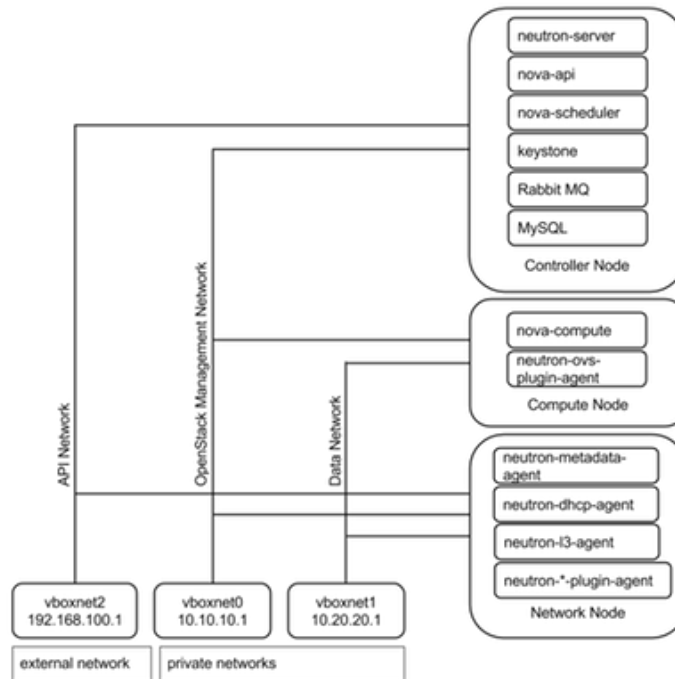
You will still be able to use Virtual Box but the instances will be very slow.

There are many ways to configure your OpenStack Setup, we will be deploying OpenStack Multi Node using OVS as the Network Plugin and QEMU/ KVM as the hypervisor.

**Host Only Connections:**

- Host only connections provide an Internal network between your host and the Virtual Machine instances up and running on your host machine. This network is not traceable by other networks.
- You may even use Bridged connection if you have a router/switch. I am assuming the worst case (one IP without any router), so that it is simple to get the required networks running without the hassle of IP tables.
- The following are the host only connections that you will be setting up later on :
  1. vboxnet0 - OpenStack Management Network - Host static IP 10.10.10.1
  2. vboxnet1 - VM Conf.Network - Host Static IP 10.20.20.1
  3. vboxnet2 - VM External Network Access (Host Machine) 192.168.100.1

**Network Diagram :**

**Figure B.1. Network Diagram**

Publicly editable image source at [https://docs.google.com/drawings/d/1GX3FXmkz3c\\_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing](https://docs.google.com/drawings/d/1GX3FXmkz3c_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing)

Vboxnet0, Vboxnet1, Vboxnet2 - are virtual networks setup up by virtual box with your host machine. This is the way your host can communicate with the virtual machines. These networks are in turn used by virtual box VM's for OpenStack networks, so that OpenStack's services can communicate with each other.

## Setup Your VM Environment

Before you can start configuring your Environment you need to download some of the following stuff:

### 1. [Oracle Virtual Box](#)

Note: You cannot set up a amd64 VM on a x86 machine.

### 1. [Ubuntu 12.04 Server or Ubuntu 13.04 Server](#)

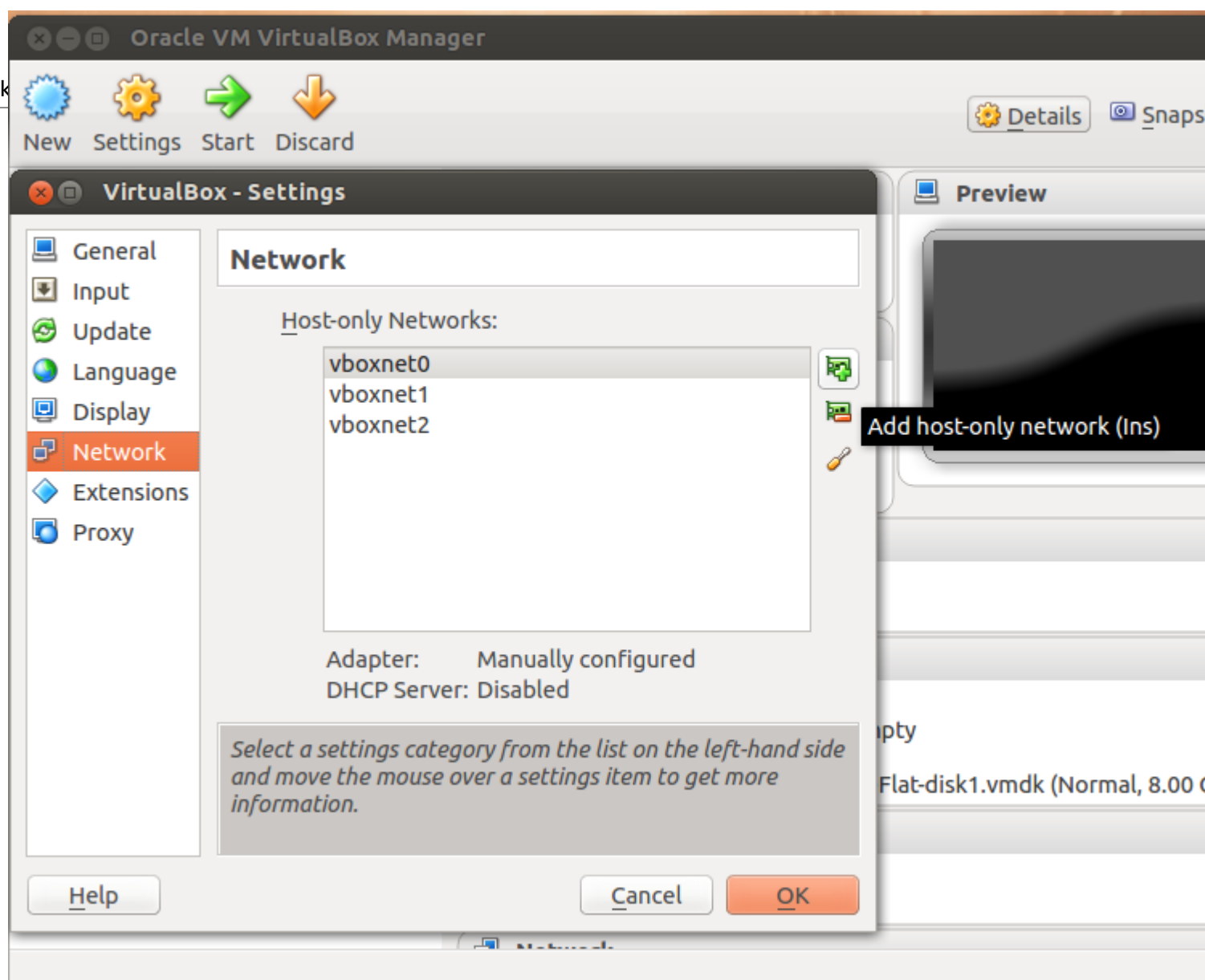
Note: You need a x86 image for VM's if kvm-ok fails, even though you are on amd64 machine.

Note: Even Though I'm using Ubuntu as Host, the same is applicable to Windows, Mac and other Linux Hosts.

- If you have i5 or i7 2nd gen processor you can have VT technology inside VM's provided by VmWare. This means that your OpenStack nodes (Which are in turn VM's) will give positive result on KVM-OK. (I call it - Nesting of type-2 Hypervisors). Rest of the configurations remain same except for the UI and few other trivial differences.

## Configure Virtual Networks

- This section of the guide will help you setup your networks for your Virtual Machine.
- Launch Virtual Box
- Click on **File>Preferences** present on the menu bar of Virtual Box.
- Select the **Network tab**.
- On the right side you will see an option to add Host-Only networks.

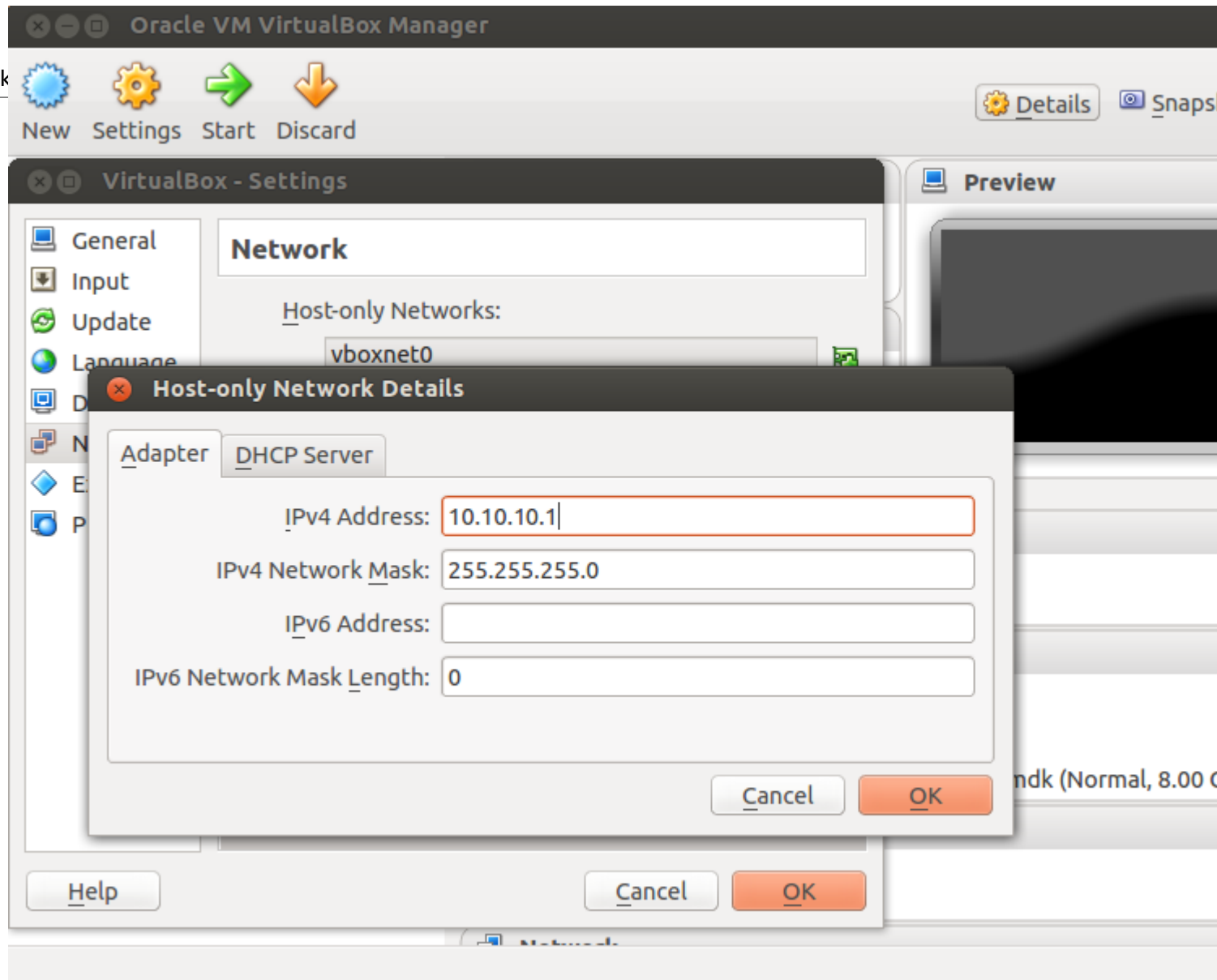


- Create three Host-Only Network Connections. As shown above.
- Edit the Host-Only Connections to have the following settings.

**Vboxnet0**

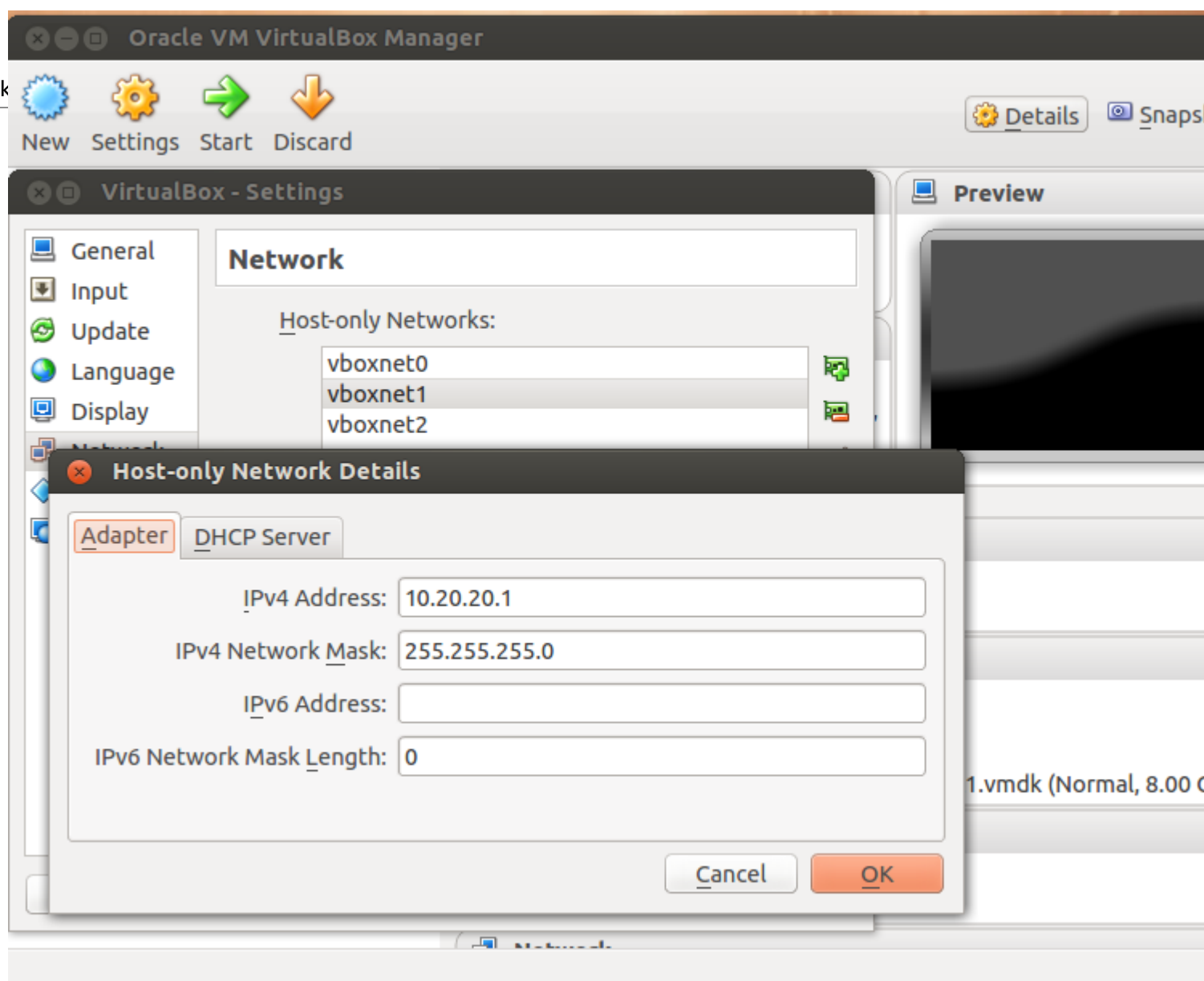
| Option                     | Value             |
|----------------------------|-------------------|
| IPv4 Address:              | 10.10.10.1        |
| IPv4 Network Mask:         | 255.255.255.0     |
| IPv6 Address:              | Can be Left Blank |
| IPv6 Network Mask Length : | Can be Left Blank |





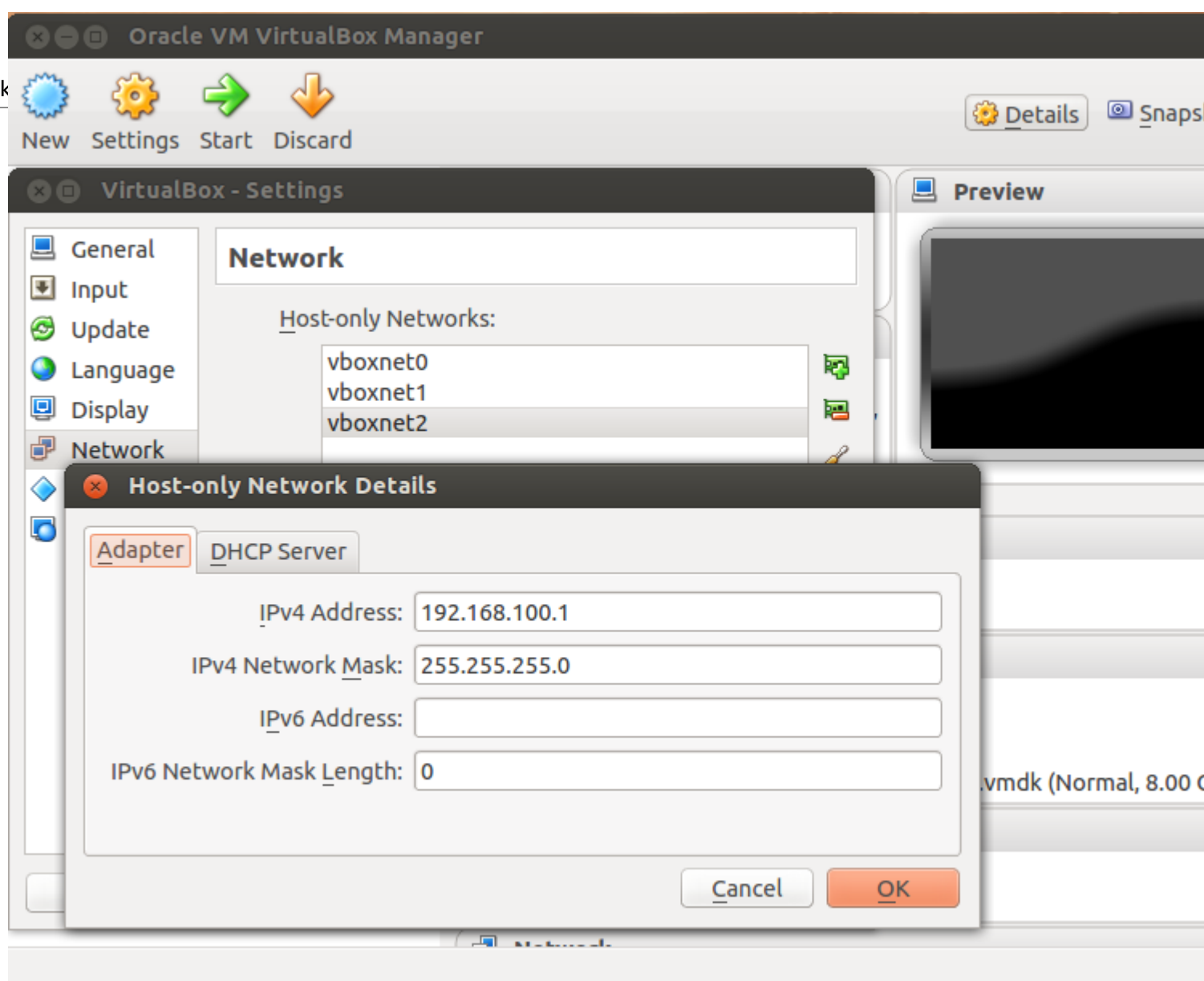
**Vboxnet1**

| Option                     | Value             |
|----------------------------|-------------------|
| IPv4 Address:              | 10.20.20.1        |
| IPv4 Network Mask:         | 255.255.255.0     |
| IPv6 Address:              | Can be Left Blank |
| IPv6 Network Mask Length : | Can be Left Blank |



**Vboxnet2**

| Option                     | Value             |
|----------------------------|-------------------|
| IPv4 Address:              | 192.168.100.1     |
| IPv4 Network Mask:         | 255.255.255.0     |
| IPv6 Address:              | Can be Left Blank |
| IPv6 Network Mask Length : | Can be Left Blank |



### Install SSH and FTP

- You may benefit by installing SSH and FTP so that you could use your remote shell to login into the machine and use your terminal which is more convenient than using the Virtual Machines tty through the Virtual Box's UI. You get a few added comforts like copy - paste commands into the remote terminal which is not possible directly on VM.
- FTP is for transferring files to and from ... you can also use SFTP or install FTPD on both HOST and VM's.
- Installation of SSH and FTP with its configuration is out of scope of this GUIDE and I may put it up but it depends upon my free time. If someone wants to contribute to this - please do so.

**Note:** Please set up the Networks from inside the VM before trying to SSH and FTP into the machines. I would suggest setting it up at once just after the installation of the Server on VM's is over.

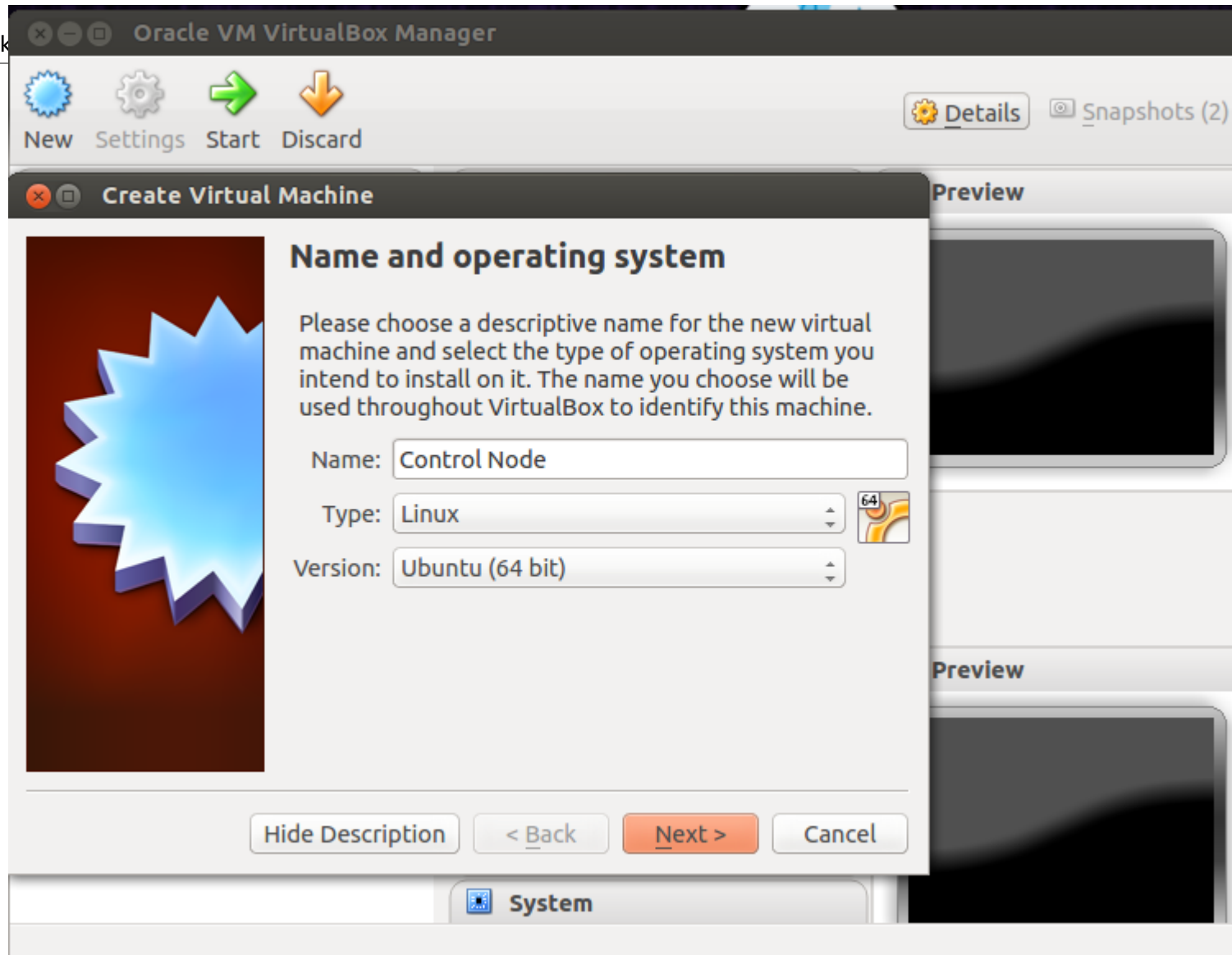
### Install Your VM's Instances

- During Installation of The Operating Systems you will be asked for Custom Software to Install, if you are confused or not sure about this, just skip this step by pressing Enter Key without selecting any of the given Options.

**Warning** - Please do not install any of the other packages except for which are mentioned below unless you know what you are doing. There is a good chance that you may end up getting unwanted errors, package conflicts ... due to the same.

### Control Node:

Create a new virtual machine. Select Ubuntu Server



Select the appropriate amount of RAM. For the control node, the minimum is 512 MB of RAM. For other settings, use the defaults. The hard disk size can be 8 GB as default.

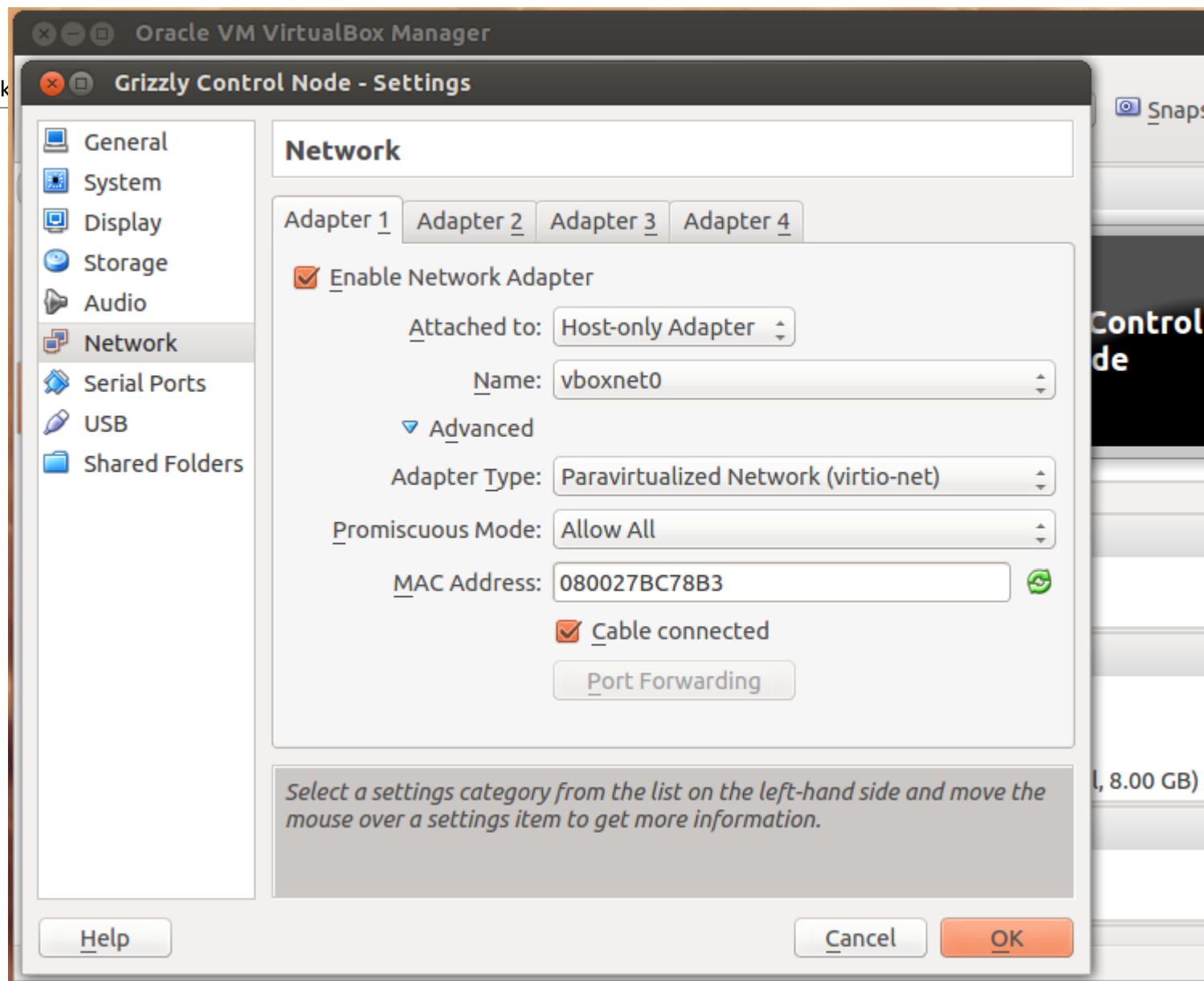
Configure the networks

(Ignore the IP Address for now, you will set it up from inside the VM)

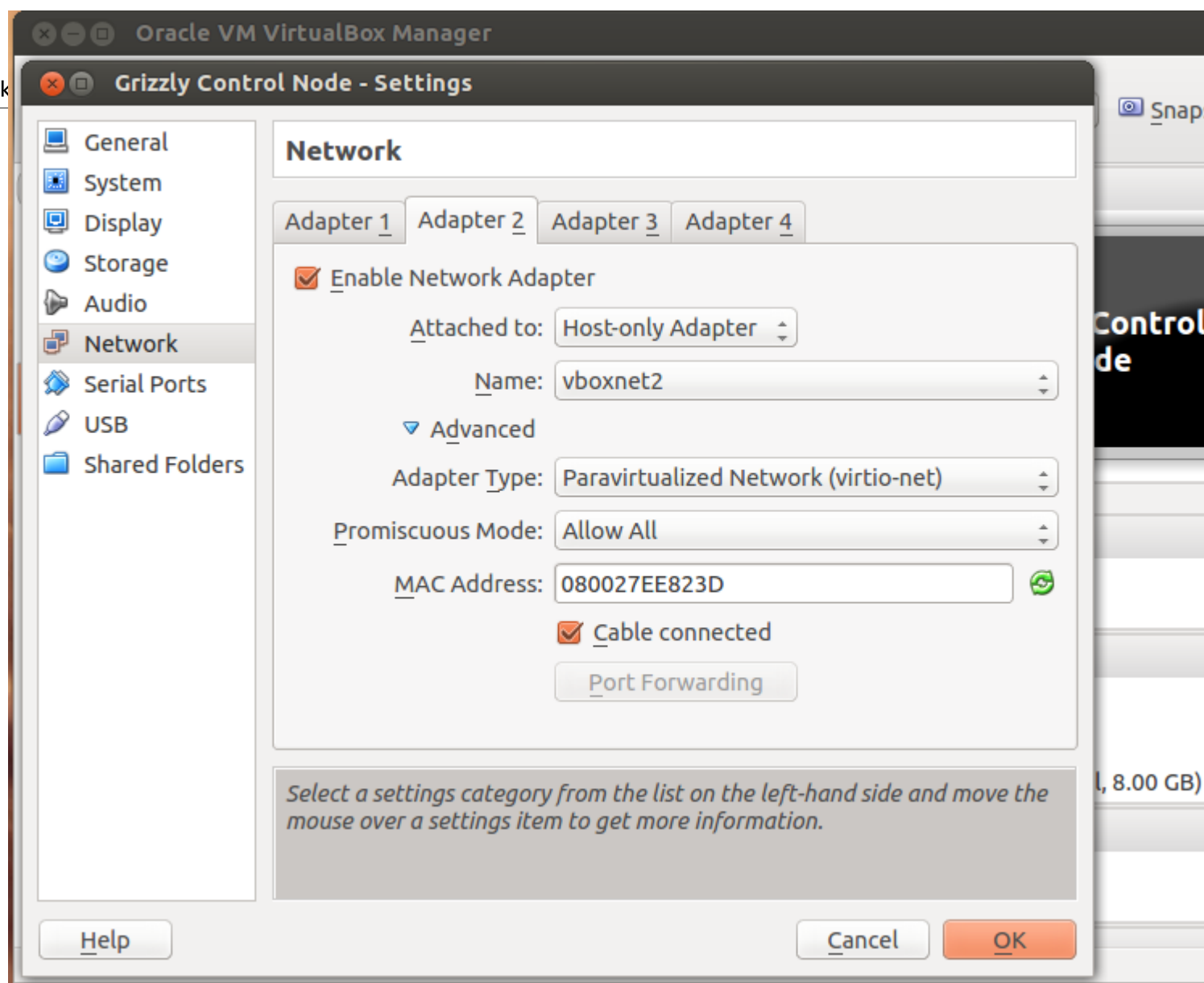
| Network Adapter | Host-Only Adapter Name | IP Address     |
|-----------------|------------------------|----------------|
| eth0            | Vboxnet0               | 10.10.10.51    |
| eth1            | Vboxnet2               | 192.168.100.51 |
| eth2            | NAT                    | DHCP           |

**Adapter 1 (Vboxnet0)**

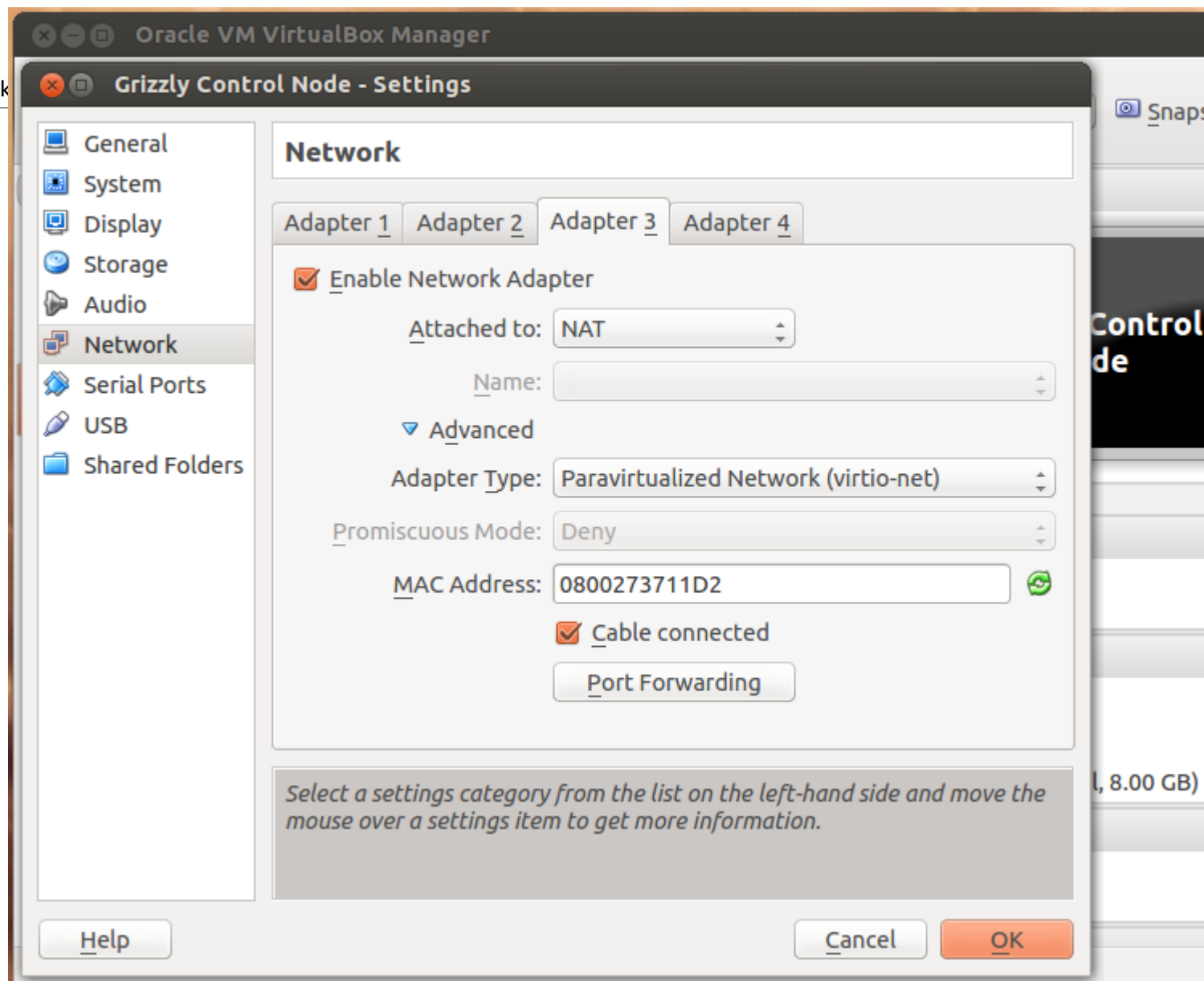




## **Adapter 2 (Vboxnet2)**



### **Adapter 3 (NAT)**



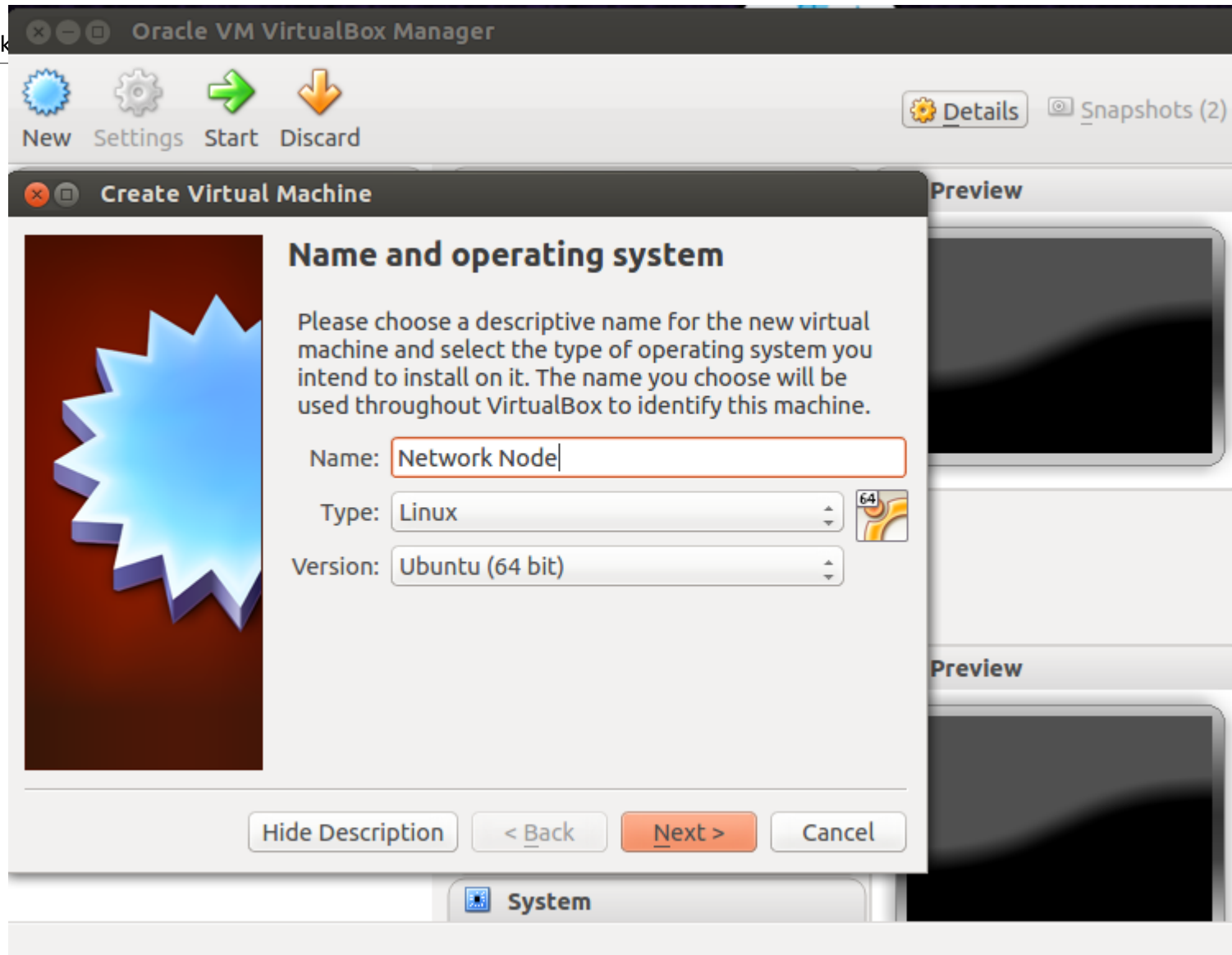
Now Install Ubuntu Server 12.04 or 13.04 on this machine.

**Note :**Install SSH server when asked for Custom Software to Install. Rest of the packages are not required and may come in the way of OpenStack packages - like DNS servers etc. (not necessary). Unless you know what you are doing.

**Network Node:**

Create a new Virtual Machine,

Minimum RAM is 512 MB. Rest all can be left default. Minimum HDD space 8 GB.



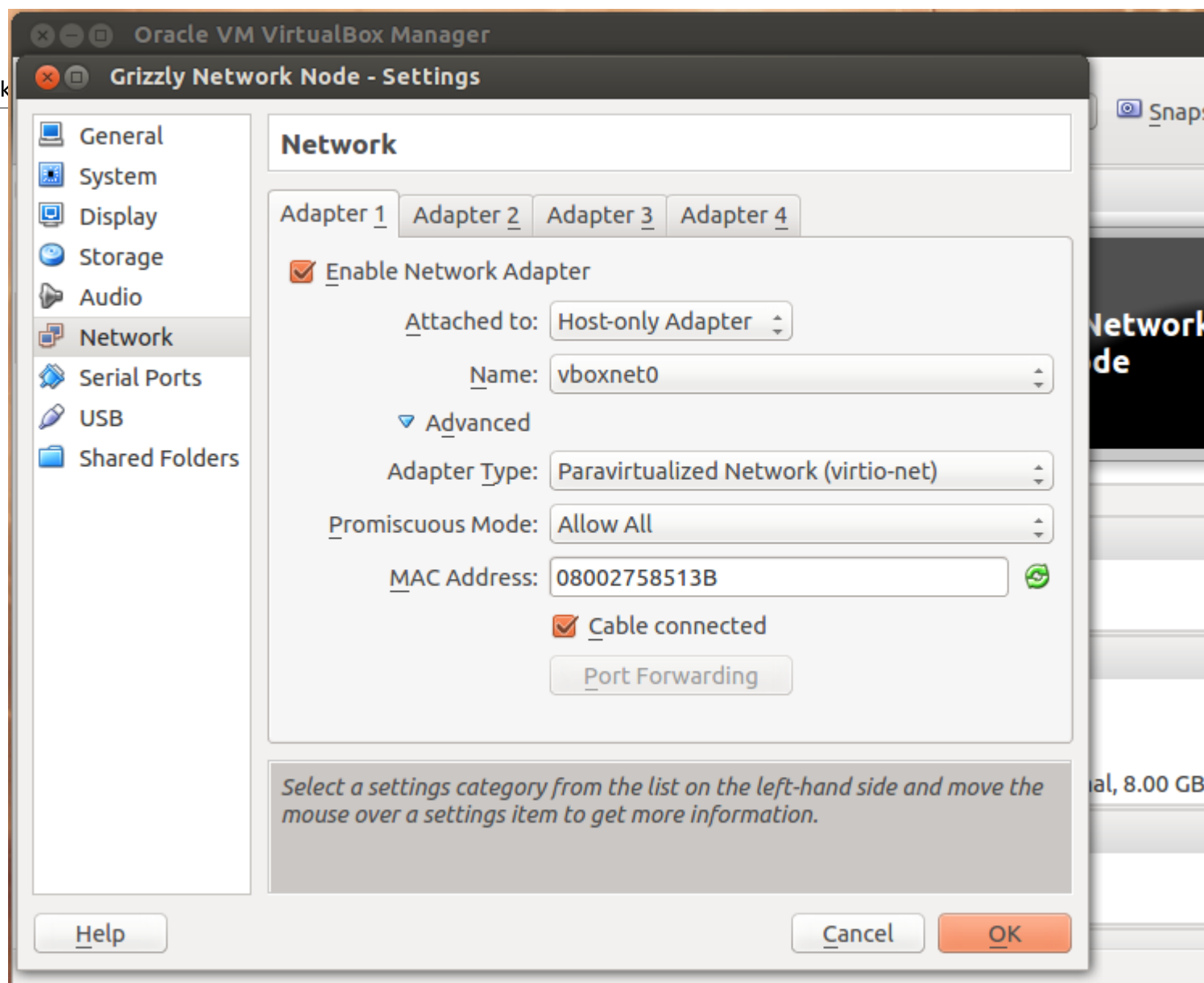
Configure the networks

(Ignore the IP Address for now, you will set it up from inside the VM)

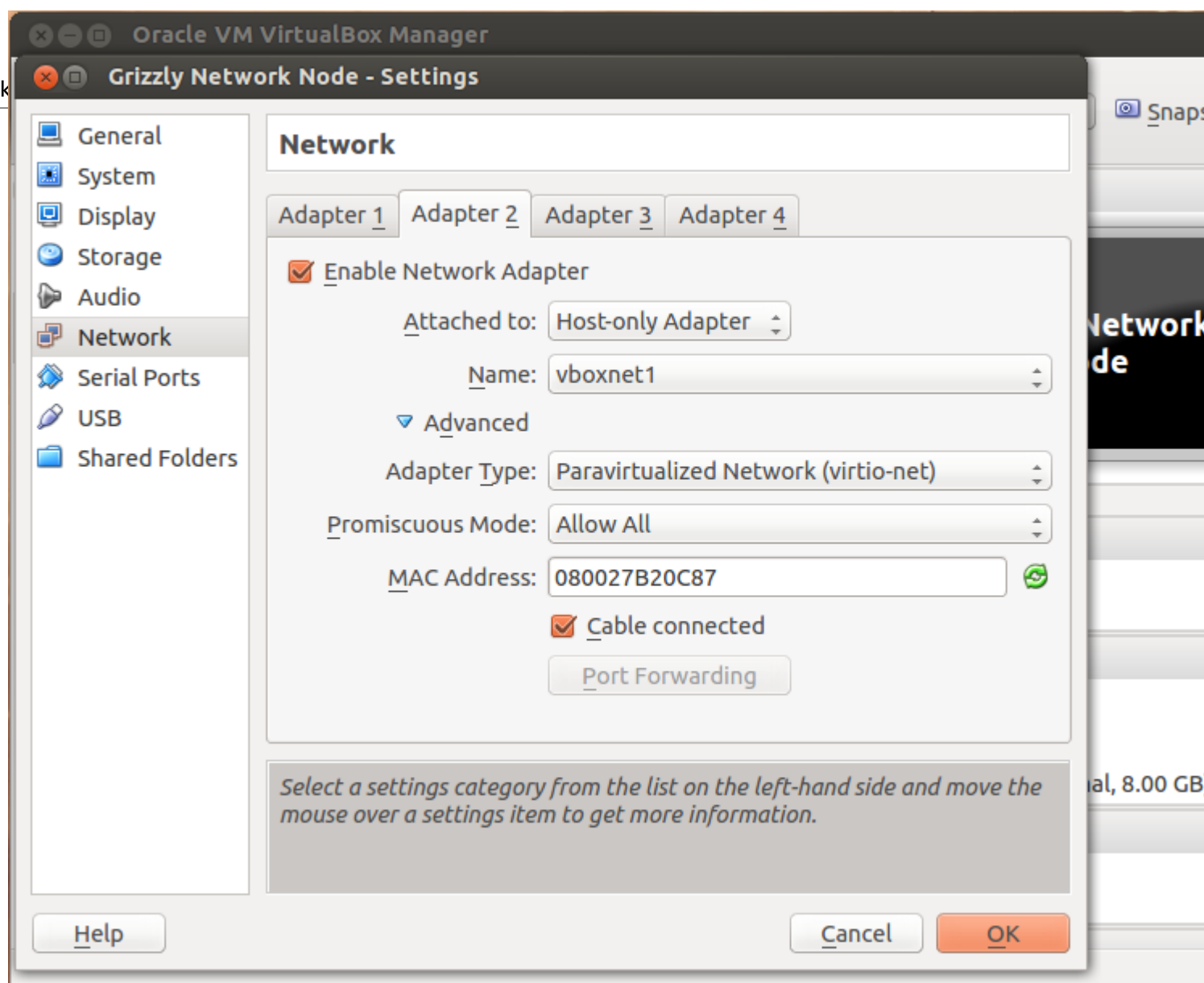
| Network Adapter | Host-Only Adapter Name | IP Address     |
|-----------------|------------------------|----------------|
| eth0            | Vboxnet0               | 10.10.10.52    |
| eth1            | Vboxnet1               | 10.20.20.52    |
| eth2            | Vboxnet2               | 192.168.100.51 |
| eth3            | NAT                    | DHCP           |

**Adapter 1 (Vboxnet0)**

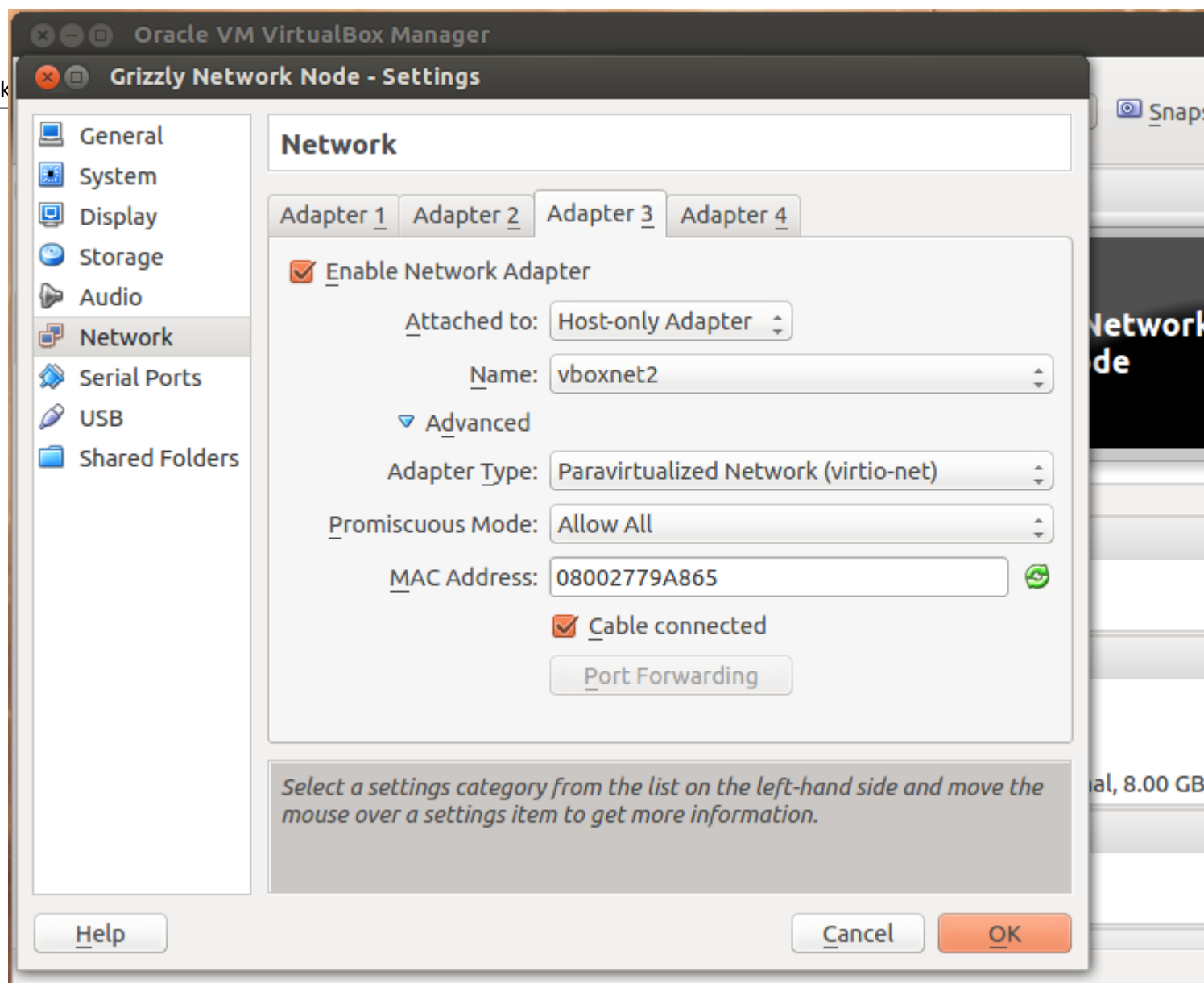




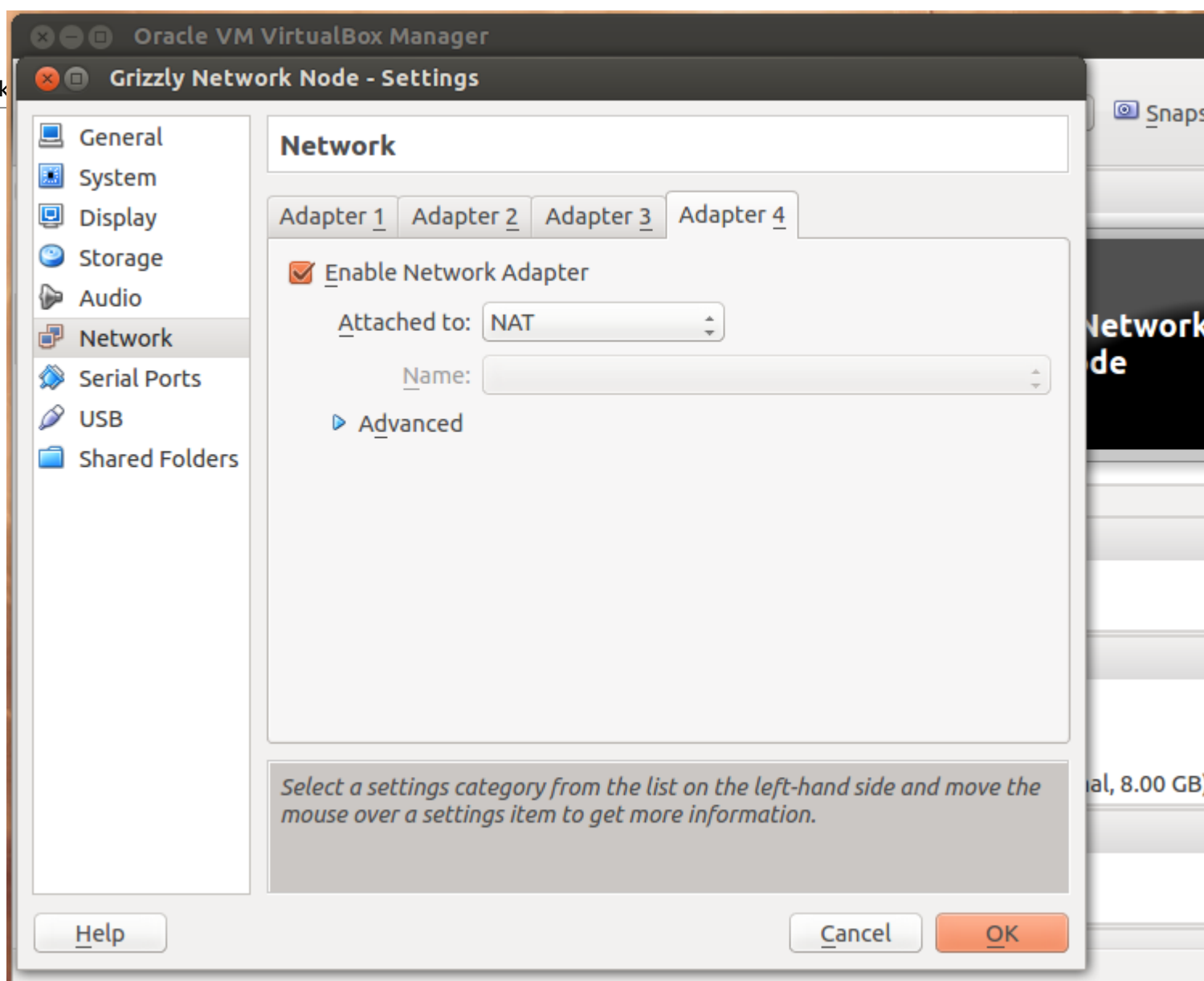
## **Adapter 2 (Vboxnet1)**



### **Adapter 3 (Vboxnet2)**



## **Adapter 4 (NAT)**



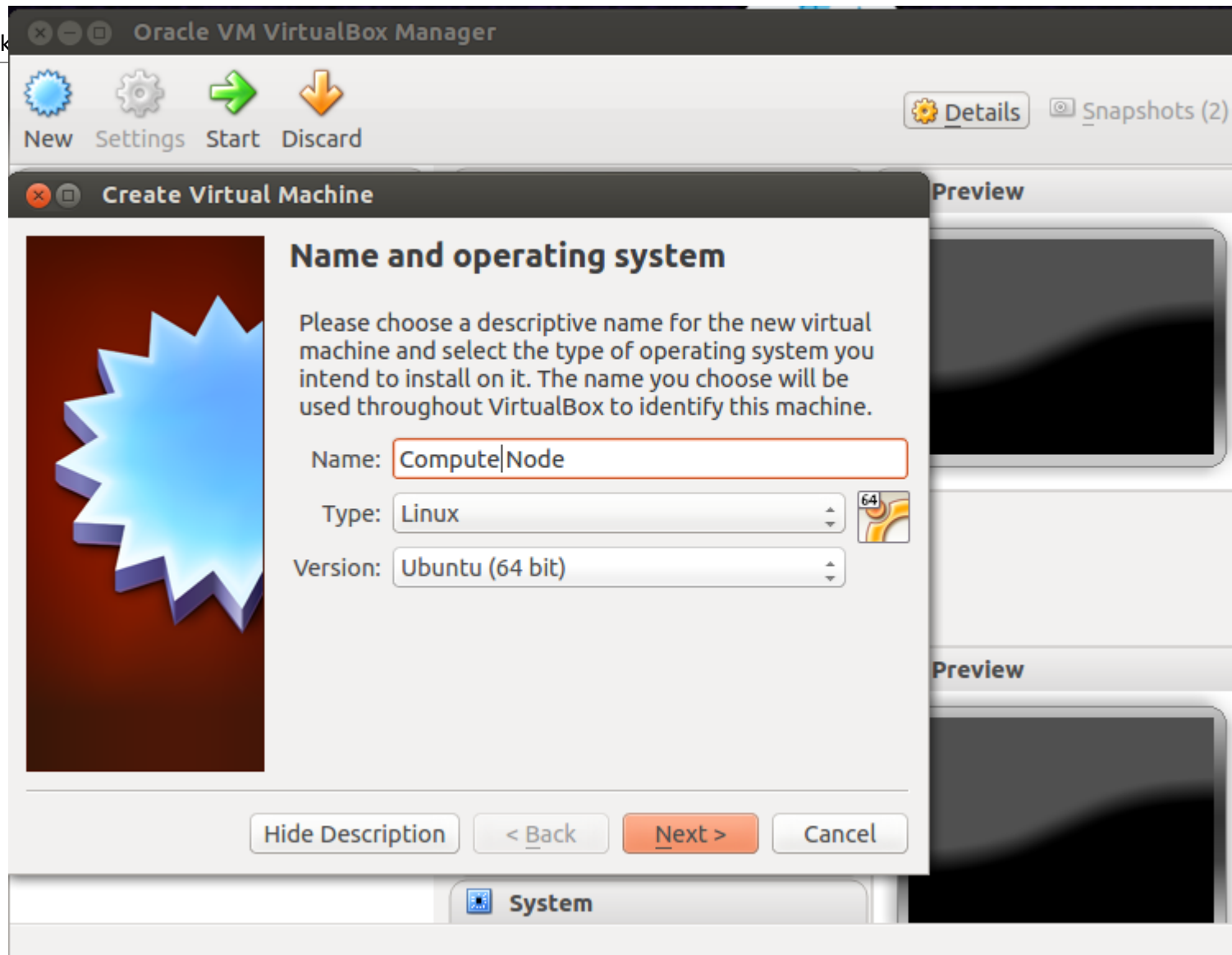
Now Install Ubuntu Server 12.04 or 13.04 on this machine.

**Note :**Install SSH server when asked for Custom Software to Install. Rest of the packages are not required and may come in the way of OpenStack packages - like DNS servers etc. (not necessary). Unless you know what you are doing.

**Compute Node:**

Create a virtual machine with at least 1,000 MB RAM and 8 GB HDD. For other settings, use the defaults.



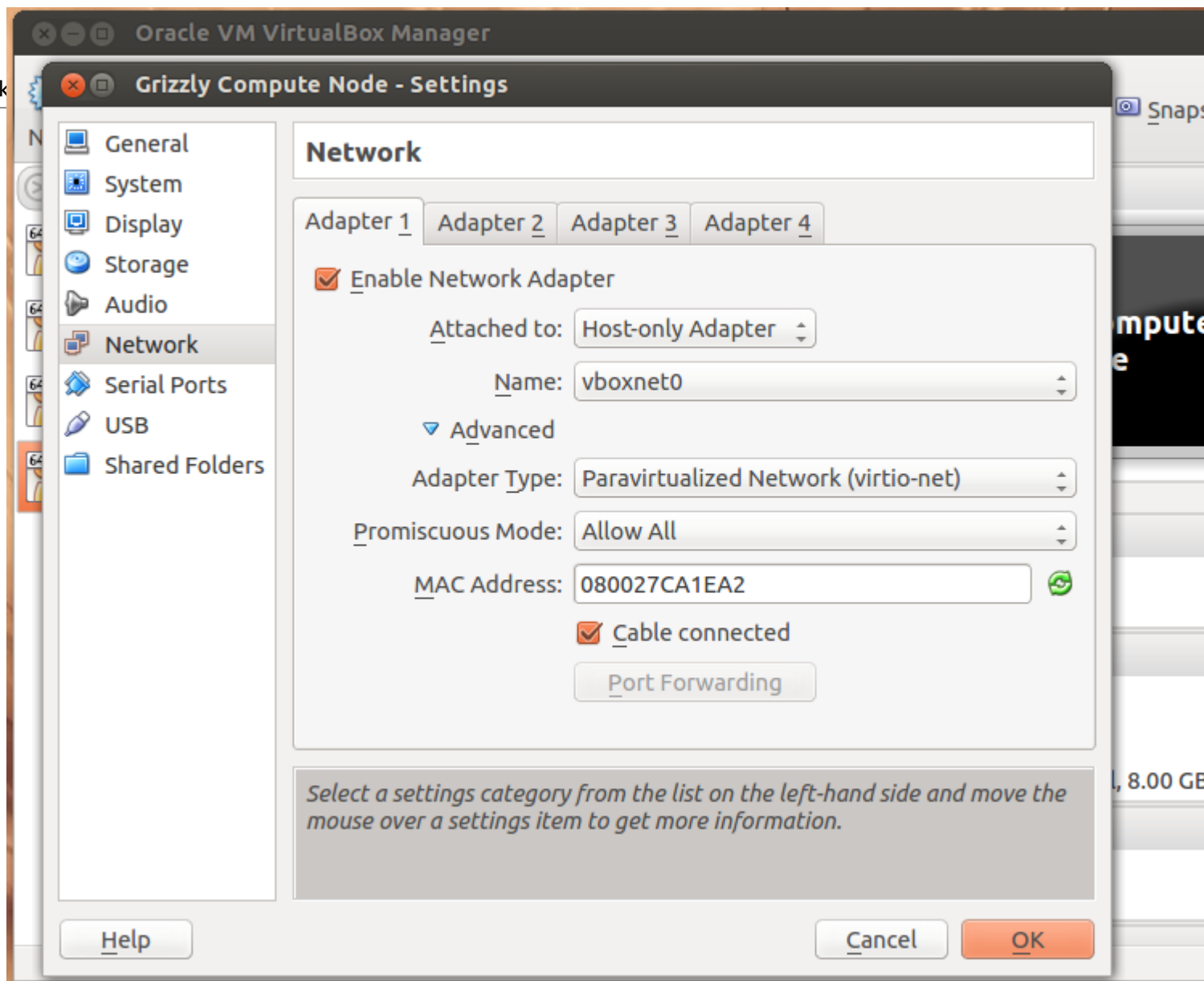


Configure the networks

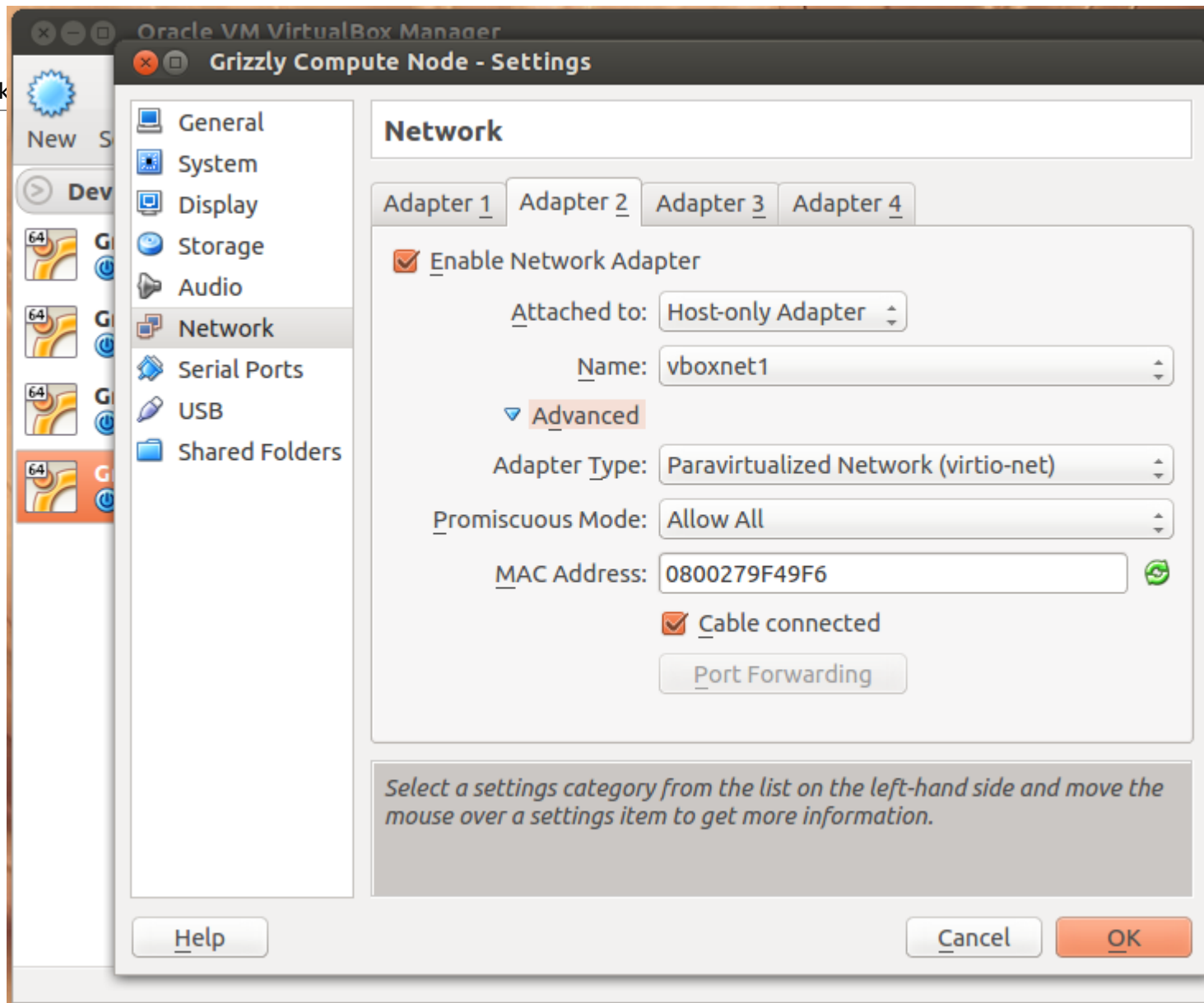
(Ignore the IP Address for now, you will set it up from inside the VM)

| Network Adapter | Host-Only Adapter Name | IP Address  |
|-----------------|------------------------|-------------|
| eth0            | Vboxnet0               | 10.10.10.53 |
| eth1            | Vboxnet1               | 10.20.20.53 |
| eth2            | NAT                    | DHCP        |

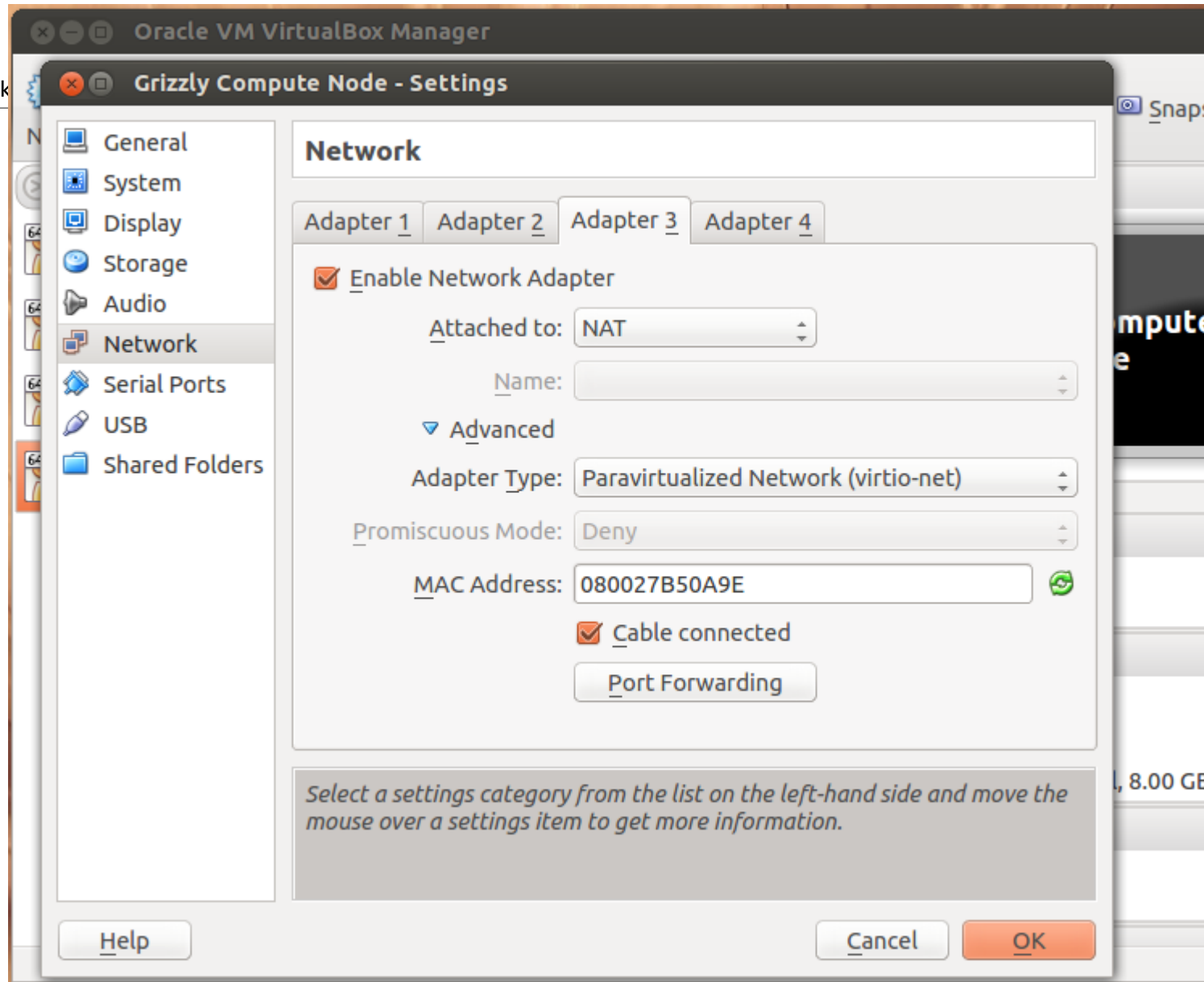
**Adapter 1 (Vboxnet0)**



## **Adapter 2 (Vboxnet1)**



### **Adapter 3 (NAT)**



Now Install Ubuntu Server 12.04 or 13.04 on this machine.

**Note :** Install SSH server when asked for Custom Software to Install. Rest of the packages are not required and may come in the way of OpenStack packages - like DNS servers etc. (not necessary). Unless you know what you are doing.

**Warnings/Advice :**

- Well there are a few warnings that I must give you out of experience due to common habits that most people may have :

Sometimes shutting down your Virtual Machine may lead to malfunctioning of OpenStack Services. Try not to direct shutdown your 3. In case your VM's don't get internet.

- From your VM Instance, use **ping** command to see whether Internet is on.

```
$ ping www.google.com
```

- If its not connected, restart networking service:

```
# service networking restart  
# ping www.google.com
```

- If this doesn't work, you need to check your network settings from Virtual Box, you may have left something or misconfigured it.
- This should reconnect your network about 99% of the times. If you are really unlucky you must be having some other problems or your Internet connection itself is not functioning.
- Note :There are known bugs with the ping under NAT. Although the latest versions of Virtual Box have better performance, sometimes ping may not work even if your Network is connected to internet.

Congrats, you are ready with the infrastructure for deploying OpenStack. Just make sure that you have installed Ubuntu Server on the above setup Virtual Box Instances. In the next section we will go through deploying OpenStack using the above created Virtual Box instances.



# Appendix C. Community support

## Table of Contents

|                                       |    |
|---------------------------------------|----|
| Documentation .....                   | 47 |
| ask.openstack.org .....               | 49 |
| OpenStack mailing lists .....         | 49 |
| The OpenStack wiki .....              | 50 |
| The Launchpad Bugs area .....         | 50 |
| The OpenStack IRC channel .....       | 51 |
| Documentation feedback .....          | 52 |
| OpenStack distribution packages ..... | 52 |

The following resources are available to help you run and use OpenStack. The OpenStack community constantly improves and adds to the main features of OpenStack, but if you have any questions, do not hesitate to ask. Use the following resources to get OpenStack support, and troubleshoot your installations.

## Documentation

For the available OpenStack documentation, see [docs.openstack.org](https://docs.openstack.org).

To provide feedback on documentation, join and use the <openstack-docs@lists.openstack.org> mailing list at [OpenStack Documentation Mailing List](#), or [report a bug](#).

The following books explain how to install an OpenStack cloud and its associated components:

- [Installation Guide for Debian 7.0](#)

- [\*Installation Guide for openSUSE and SUSE Linux Enterprise Server\*](#)
- [\*Installation Guide for Red Hat Enterprise Linux, CentOS, and Fedora\*](#)
- [\*Installation Guide for Ubuntu 12.04/14.04 \(LTS\)\*](#)

The following books explain how to configure and run an OpenStack cloud:

- [\*Cloud Administrator Guide\*](#)
- [\*Configuration Reference\*](#)
- [\*Operations Guide\*](#)
- [\*High Availability Guide\*](#)
- [\*Security Guide\*](#)
- [\*Virtual Machine Image Guide\*](#)

The following books explain how to use the OpenStack dashboard and command-line clients:

- [\*API Quick Start\*](#)
- [\*End User Guide\*](#)
- [\*Admin User Guide\*](#)
- [\*Command-Line Interface Reference\*](#)

The following documentation provides reference and guidance information for the OpenStack APIs:

- [\*OpenStack API Complete Reference \(HTML\)\*](#)
- [\*API Complete Reference \(PDF\)\*](#)

- [OpenStack Block Storage Service API v2 Reference](#)
- [OpenStack Compute API v2 and Extensions Reference](#)
- [OpenStack Identity Service API v2.0 Reference](#)
- [OpenStack Image Service API v2 Reference](#)
- [OpenStack Networking API v2.0 Reference](#)
- [OpenStack Object Storage API v1 Reference](#)

The [Training Guides](#) offer software training for cloud administration and management.

## ask.openstack.org

During the set up or testing of OpenStack, you might have questions about how a specific task is completed or be in a situation where a feature does not work correctly. Use the [ask.openstack.org](http://ask.openstack.org) site to ask questions and get answers. When you visit the <http://ask.openstack.org> site, scan the recently asked questions to see whether your question has already been answered. If not, ask a new question. Be sure to give a clear, concise summary in the title and provide as much detail as possible in the description. Paste in your command output or stack traces, links to screen shots, and any other information which might be useful.

## OpenStack mailing lists

A great way to get answers and insights is to post your question or problematic scenario to the OpenStack mailing list. You can learn from and help others who might have similar issues. To subscribe or view the archives, go to <http://lists.openstack.org/cgi-bin/mailman/listinfo/openstack>. You might be interested in the other mailing lists for specific projects or development, which you can find [on the wiki](#). A description of all mailing lists is available at <http://wiki.openstack.org/MailingLists>.

## The OpenStack wiki

The [OpenStack wiki](#) contains a broad range of topics but some of the information can be difficult to find or is a few pages deep. Fortunately, the wiki search feature enables you to search by title or content. If you search for specific information, such as about networking or nova, you can find a large amount of relevant material. More is being added all the time, so be sure to check back often. You can find the search box in the upper-right corner of any OpenStack wiki page.

## The Launchpad Bugs area

The OpenStack community values your set up and testing efforts and wants your feedback. To log a bug, you must sign up for a Launchpad account at <https://launchpad.net/+login>. You can view existing bugs and report bugs in the Launchpad Bugs area. Use the search feature to determine whether the bug has already been reported or already been fixed. If it still seems like your bug is unreported, fill out a bug report.

Some tips:

- Give a clear, concise summary.
- Provide as much detail as possible in the description. Paste in your command output or stack traces, links to screen shots, and any other information which might be useful.
- Be sure to include the software and package versions that you are using, especially if you are using a development branch, such as, "Juno release" vs `git commit bc79c3ecc55929bac585d04a03475b72e06a3208`.
- Any deployment-specific information is helpful, such as whether you are using Ubuntu 14.04 or are performing a multi-node installation.

The following Launchpad Bugs areas are available:

- [Bugs: OpenStack Block Storage \(cinder\)](#)
- [Bugs: OpenStack Compute \(nova\)](#)
- [Bugs: OpenStack Dashboard \(horizon\)](#)
- [Bugs: OpenStack Identity \(keystone\)](#)
- [Bugs: OpenStack Image Service \(glance\)](#)
- [Bugs: OpenStack Networking \(neutron\)](#)
- [Bugs: OpenStack Object Storage \(swift\)](#)
- [Bugs: Bare Metal \(ironic\)](#)
- [Bugs: Data Processing Service \(sahara\)](#)
- [Bugs: Database Service \(trove\)](#)
- [Bugs: Orchestration \(heat\)](#)
- [Bugs: Telemetry \(ceilometer\)](#)
- [Bugs: Queue Service \(marconi\)](#)
- [Bugs: OpenStack API Documentation \(api.openstack.org\)](#)
- [Bugs: OpenStack Documentation \(docs.openstack.org\)](#)

## The OpenStack IRC channel

The OpenStack community lives in the #openstack IRC channel on the Freenode network. You can hang out, ask questions, or get immediate feedback for urgent and pressing issues. To install an IRC client or

use a browser-based client, go to <http://webchat.freenode.net/>. You can also use Colloquy (Mac OS X, <http://colloquy.info/>), mIRC (Windows, <http://www.mirc.com/>), or XChat (Linux). When you are in the IRC channel and want to share code or command output, the generally accepted method is to use a Paste Bin. The OpenStack project has one at <http://paste.openstack.org>. Just paste your longer amounts of text or logs in the web form and you get a URL that you can paste into the channel. The OpenStack IRC channel is #openstack on `irc.freenode.net`. You can find a list of all OpenStack IRC channels at <https://wiki.openstack.org/wiki/IRC>.

## Documentation feedback

To provide feedback on documentation, join and use the <openstack-docs@lists.openstack.org> mailing list at [OpenStack Documentation Mailing List](#), or [report a bug](#).

## OpenStack distribution packages

The following Linux distributions provide community-supported packages for OpenStack:

- **Debian:** <http://wiki.debian.org/OpenStack>
- **CentOS, Fedora, and Red Hat Enterprise Linux:** <http://openstack.redhat.com/>
- **openSUSE and SUSE Linux Enterprise Server:** <http://en.opensuse.org/Portal:OpenStack>
- **Ubuntu:** <https://wiki.ubuntu.com/ServerTeam/CloudArchive>

## **Associate Training Guide**





## Table of Contents

|                                                 |     |
|-------------------------------------------------|-----|
| 1. Getting Started .....                        | 1   |
| Day 1, 09:00 to 11:00 .....                     | 1   |
| Overview .....                                  | 1   |
| Introduction Text .....                         | 2   |
| Brief Overview .....                            | 4   |
| Core Projects .....                             | 7   |
| OpenStack Architecture .....                    | 21  |
| Virtual Machine Provisioning Walk-Through ..... | 33  |
| 2. Getting Started Quiz .....                   | 41  |
| Day 1, 10:40 to 11:00 .....                     | 41  |
| 3. Controller Node .....                        | 45  |
| Day 1, 11:15 to 12:30, 13:30 to 14:45 .....     | 45  |
| Overview Horizon and OpenStack CLI .....        | 45  |
| Keystone Architecture .....                     | 95  |
| OpenStack Messaging and Queues .....            | 100 |
| Administration Tasks .....                      | 111 |
| 4. Controller Node Quiz .....                   | 149 |
| Day 1, 14:25 to 14:45 .....                     | 149 |
| 5. Compute Node .....                           | 155 |
| Day 1, 15:00 to 17:00 .....                     | 155 |
| VM Placement .....                              | 155 |
| VM provisioning in-depth .....                  | 163 |
| OpenStack Block Storage .....                   | 167 |
| Administration Tasks .....                      | 172 |
| 6. Compute Node Quiz .....                      | 317 |
| Day 1, 16:40 to 17:00 .....                     | 317 |
| 7. Network Node .....                           | 319 |
| Day 2, 09:00 to 11:00 .....                     | 319 |

|                                             |     |
|---------------------------------------------|-----|
| Networking in OpenStack .....               | 319 |
| OpenStack Networking Concepts .....         | 325 |
| Administration Tasks .....                  | 327 |
| 8. Network Node Quiz .....                  | 463 |
| Day 2, 10:40 to 11:00 .....                 | 463 |
| 9. Object Storage Node .....                | 465 |
| Day 2, 11:30 to 12:30, 13:30 to 14:45 ..... | 465 |
| Introduction to Object Storage .....        | 465 |
| Features and Benefits .....                 | 466 |
| Administration Tasks .....                  | 467 |
| 10. Object Storage Node Quiz .....          | 477 |
| Day 2, 14:25 to 14:45 .....                 | 477 |
| 11. Assessment .....                        | 479 |
| Day 2, 15:00 to 16:00 .....                 | 479 |
| Questions .....                             | 479 |
| 12. Review of Concepts .....                | 481 |
| Day 2, 16:00 to 17:00 .....                 | 481 |

## List of Figures

|                                                                                              |     |
|----------------------------------------------------------------------------------------------|-----|
| 1.1. Nebula (NASA) .....                                                                     | 5   |
| 1.2. Community Heartbeat .....                                                               | 9   |
| 1.3. Various Projects under OpenStack .....                                                  | 10  |
| 1.4. Programming Languages used to design OpenStack .....                                    | 12  |
| 1.5. OpenStack Compute: Provision and manage large networks of virtual machines .....        | 14  |
| 1.6. OpenStack Storage: Object and Block storage for use with servers and applications ..... | 15  |
| 1.7. OpenStack Networking: Pluggable, scalable, API-driven network and IP management .....   | 17  |
| 1.8. Conceptual Diagram .....                                                                | 23  |
| 1.9. Logical Diagram .....                                                                   | 25  |
| 1.10. Horizon Dashboard .....                                                                | 27  |
| 1.11. Initial State .....                                                                    | 36  |
| 1.12. Launch VM Instance .....                                                               | 38  |
| 1.13. End State .....                                                                        | 40  |
| 3.1. OpenStack Dashboard - Overview .....                                                    | 47  |
| 3.2. OpenStack Dashboard - Security Groups .....                                             | 50  |
| 3.3. OpenStack Dashboard - Security Group Rules .....                                        | 50  |
| 3.4. OpenStack Dashboard- Instances .....                                                    | 58  |
| 3.5. OpenStack Dashboard : Actions .....                                                     | 60  |
| 3.6. OpenStack Dashboard - Track Usage .....                                                 | 61  |
| 3.7. Keystone Authentication .....                                                           | 97  |
| 3.8. Messaging in OpenStack .....                                                            | 100 |
| 3.9. AMQP .....                                                                              | 102 |
| 3.10. RabbitMQ .....                                                                         | 105 |
| 3.11. RabbitMQ .....                                                                         | 106 |
| 3.12. RabbitMQ .....                                                                         | 107 |
| 5.1. Nova .....                                                                              | 156 |
| 5.2. Filtering .....                                                                         | 158 |
| 5.3. Weights .....                                                                           | 162 |

|                                 |     |
|---------------------------------|-----|
| 5.4. Nova VM provisioning ..... | 166 |
| 7.1. Network Diagram .....      | 324 |

## List of Tables

|                                                              |     |
|--------------------------------------------------------------|-----|
| 3.1. Disk and CD-ROM bus model values .....                  | 140 |
| 3.2. VIF model values .....                                  | 140 |
| 3.3. Description of configuration options for rabbitmq ..... | 144 |
| 3.4. Description of configuration options for kombu .....    | 144 |
| 3.5. Description of configuration options for qpid .....     | 146 |
| 3.6. Description of configuration options for zeromq .....   | 147 |
| 3.7. Description of configuration options for rpc .....      | 147 |
| 11.1. Assessment Question 1 .....                            | 479 |
| 11.2. Assessment Question 2 .....                            | 479 |



# 1. Getting Started

## Table of Contents

|                                                 |    |
|-------------------------------------------------|----|
| Day 1, 09:00 to 11:00 .....                     | 1  |
| Overview .....                                  | 1  |
| Introduction Text .....                         | 2  |
| Brief Overview .....                            | 4  |
| Core Projects .....                             | 7  |
| OpenStack Architecture .....                    | 21 |
| Virtual Machine Provisioning Walk-Through ..... | 33 |

## Day 1, 09:00 to 11:00

### Overview

Training will take 1 month self paced, (2) 2 week periods with a user group meeting, or 16 hours instructor led.

#### Prerequisites

1. Working knowledge of Linux CLI, basic Linux SysAdmin skills (directory structure, vi, ssh, installing software)
2. Basic networking knowledge (Ethernet, VLAN, IP addressing)
3. Laptop with VirtualBox installed (highly recommended)

## Introduction Text

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering users to provision resources through a web interface.

Cloud computing provides users with access to a shared collection of computing resources: networks for transfer, servers for storage, and applications or services for completing tasks.

The compelling features of a cloud are:

- On-demand self-service: Users can automatically provision needed computing capabilities, such as server time and network storage, without requiring human interaction with each service provider.
- Network access: Any computing capabilities are available over the network. Many different devices are allowed access through standardized mechanisms.
- Resource pooling: Multiple users can access clouds that serve other consumers according to demand.
- Elasticity: Provisioning is rapid and scales out or is based on need.
- Metered or measured service: Cloud systems can optimize and control resource use at the level that is appropriate for the service. Services include storage, processing, bandwidth, and active user accounts. Monitoring and reporting of resource usage provides transparency for both the provider and consumer of the utilized service.

Cloud computing offers different service models depending on the capabilities a consumer may require.

- SaaS: Software-as-a-Service. Provides the consumer the ability to use the software in a cloud environment, such as web-based email for example.



- PaaS: Platform-as-a-Service. Provides the consumer the ability to deploy applications through a programming language or tools supported by the cloud platform provider. An example of Platform-as-a-service is an Eclipse/Java programming platform provided with no downloads required.
- IaaS: Infrastructure-as-a-Service. Provides infrastructure such as computer instances, network connections, and storage so that people can run any software or operating system.

Terms such as public cloud or private cloud refer to the deployment model for the cloud. A private cloud operates for a single organization, but can be managed on-premise or off-premise. A public cloud has an infrastructure that is available to the general public or a large industry group and is likely owned by a cloud services company.

Clouds can also be described as hybrid. A hybrid cloud can be a deployment model, as a composition of both public and private clouds, or a hybrid model for cloud computing may involve both virtual and physical servers.

Cloud computing can help with large-scale computing needs or can lead consolidation efforts by virtualizing servers to make more use of existing hardware and potentially release old hardware from service. Cloud computing is also used for collaboration because of its high availability through networked computers. Productivity suites for word processing, number crunching, and email communications, and more are also available through cloud computing. Cloud computing also avails additional storage to the cloud user, avoiding the need for additional hard drives on each user's desktop and enabling access to huge data storage capacity online in the cloud.

When you explore OpenStack and see what it means technically, you can see its reach and impact on the entire world.

OpenStack is an open source software for building private and public clouds which delivers a massively scalable cloud operating system.

OpenStack is backed up by a global community of technologists, developers, researchers, corporations and cloud computing experts.



## Brief Overview

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter. It is all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.

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
- feature rich.

To check out more information on OpenStack visit <http://goo.gl/Ye9DFT>

### OpenStack Foundation:

The OpenStack Foundation, established September 2012, is an independent body providing shared resources to help achieve the OpenStack Mission by protecting, empowering, and promoting OpenStack software and the community around it. This includes users, developers and the entire ecosystem. For more information visit <http://goo.gl/3uvmNX>.

### Who's behind OpenStack?

Founded by Rackspace Hosting and NASA, OpenStack has grown to be a global software community of developers collaborating on a standard and massively scalable open source cloud operating system. The OpenStack Foundation promotes the development, distribution and adoption of the OpenStack cloud operating system. As the independent home for OpenStack, the Foundation has already attracted more than 7,000 individual members from 100 countries and 850 different organizations. It has also secured more than \$10 million in funding and is ready to fulfill the OpenStack mission of becoming the ubiquitous cloud computing platform. Checkout <http://goo.gl/BZHJKd> for more on the same.

**Figure 1.1. Nebula (NASA)**



The goal of the OpenStack Foundation is to serve developers, users, and the entire ecosystem by providing a set of shared resources to grow the footprint of public and private OpenStack clouds, enable technology vendors targeting the platform and assist developers in producing the best cloud software in the industry.

**Who uses OpenStack?**

Corporations, service providers, VARS, SMBs, researchers, and global data centers looking to deploy large-scale cloud deployments for private or public clouds leveraging the support and resulting technology of a global open source community. This is just three years into OpenStack, it's new, it's yet to mature and has immense possibilities. How do I say that? All these 'buzz words' will fall into a properly solved jigsaw puzzle as you go through this article.

**It's Open Source:**

All of the code for OpenStack is freely available under the Apache 2.0 license. Anyone can run it, build on it, or submit changes back to the project. This open development model is one of the best ways to foster badly-needed cloud standards, remove the fear of proprietary lock-in for cloud customers, and create a large ecosystem that spans cloud providers.

**Who it's for:**

Enterprises, service providers, government and academic institutions with physical hardware that would like to build a public or private cloud.

**How it's being used today:**

Organizations like CERN, Cisco WebEx, DreamHost, eBay, The Gap, HP, MercadoLibre, NASA, PayPal, Rackspace and University of Melbourne have deployed OpenStack clouds to achieve control, business agility and cost savings without the licensing fees and terms of proprietary software. For complete user stories visit <http://goo.gl/aF4lsL>, this should give you a good idea about the importance of OpenStack.

## Core Projects

### Project history and releases overview.

OpenStack is a cloud computing project that provides an Infrastructure-as-a-Service (IaaS). It is free open source software released under the terms of the Apache License. The project is managed by the OpenStack Foundation, a non-profit corporate entity established in September 2012 to promote OpenStack software and its community.

More than 200 companies joined the project, among which are AMD, Brocade Communications Systems, Canonical, Cisco, Dell, EMC, Ericsson, Groupe Bull, HP, IBM, Inktank, Intel, NEC, Rackspace Hosting, Red Hat, SUSE Linux, VMware, and Yahoo!

The technology consists of a series of interrelated projects that control pools of processing, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering its users to provision resources through a web interface.

The OpenStack community collaborates around a six-month, time-based release cycle with frequent development milestones. During the planning phase of each release, the community gathers for the OpenStack Design Summit to facilitate developer working sessions and assemble plans.

In July 2010 Rackspace Hosting and NASA jointly launched an open-source cloud-software initiative known as OpenStack. The OpenStack project intended to help organizations which offer cloud-computing services running on standard hardware. The first official release, code-named Austin, appeared four months later, with plans to release regular updates of the software every few months. The early code came from the NASA Nebula platform and from the Rackspace Cloud Files platform. In July 2011, Ubuntu Linux developers adopted OpenStack.

### OpenStack Releases

| Release Name | Release Date | Included Components |
|--------------|--------------|---------------------|
|--------------|--------------|---------------------|

|          |                   |                                                                             |
|----------|-------------------|-----------------------------------------------------------------------------|
| Austin   | 21 October 2010   | Nova, Swift                                                                 |
| Bexar    | 3 February 2011   | Nova, Glance, Swift                                                         |
| Cactus   | 15 April 2011     | Nova, Glance, Swift                                                         |
| Diablo   | 22 September 2011 | Nova, Glance, Swift                                                         |
| Essex    | 5 April 2012      | Nova, Glance, Swift, Horizon, Keystone                                      |
| Folsom   | 27 September 2012 | Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder                     |
| Grizzly  | 4 April 2013      | Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder                     |
| Havana   | 17 October 2013   | Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder                     |
| Icehouse | April 2014        | Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, (More to be added) |

Some OpenStack users include:

- PayPal / eBay
- NASA
- CERN
- Yahoo!
- Rackspace Cloud
- HP Public Cloud
- MercadoLibre.com
- AT&T

- KT (formerly Korea Telecom)
- Deutsche Telekom
- Wikimedia Labs
- Hostalia of Telefonica Group
- SUSE Cloud solution
- Red Hat OpenShift PaaS solution
- Zadara Storage
- Mint Services
- GridCentric

OpenStack is a true and innovative open standard. For more user stories, see <http://goo.gl/aF4IsL>.

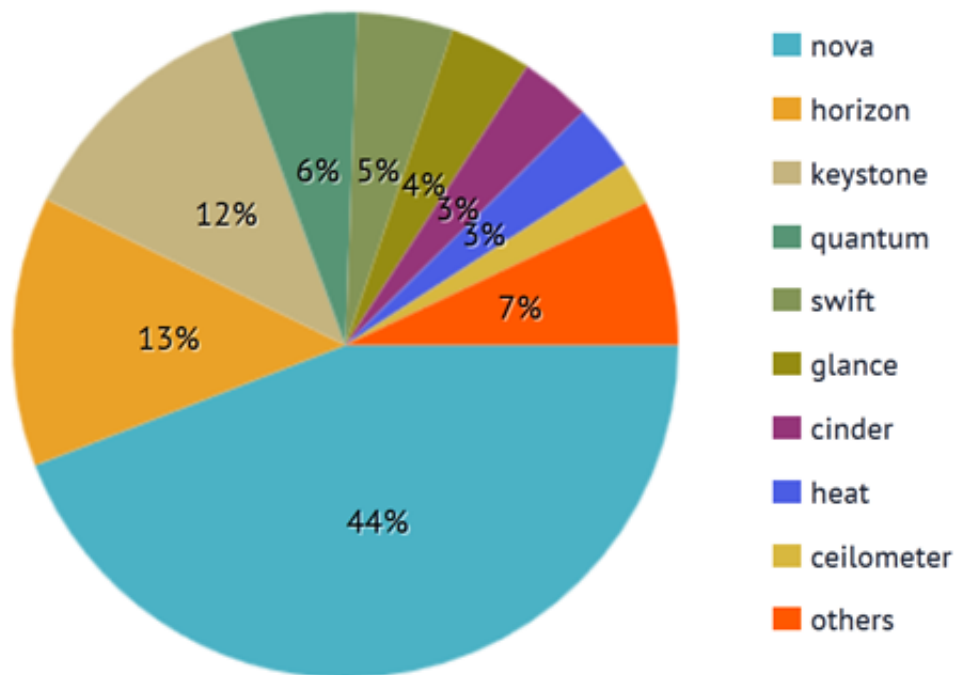
### Release Cycle

**Figure 1.2. Community Heartbeat**



OpenStack is based on a coordinated 6-month release cycle with frequent development milestones. You can find a link to the current development release schedule [here](#). The Release Cycle is made of four major stages:

**Figure 1.3. Various Projects under OpenStack**



The creation of OpenStack took an estimated 249 years of effort (COCOMO model).

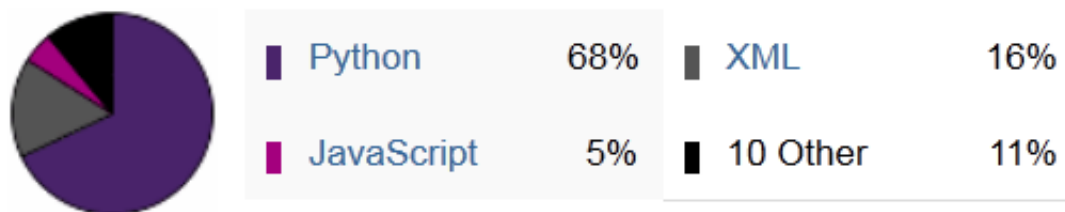
In a nutshell, OpenStack has:

- 64,396 commits made by 1,128 contributors, with its first commit made in May, 2010.

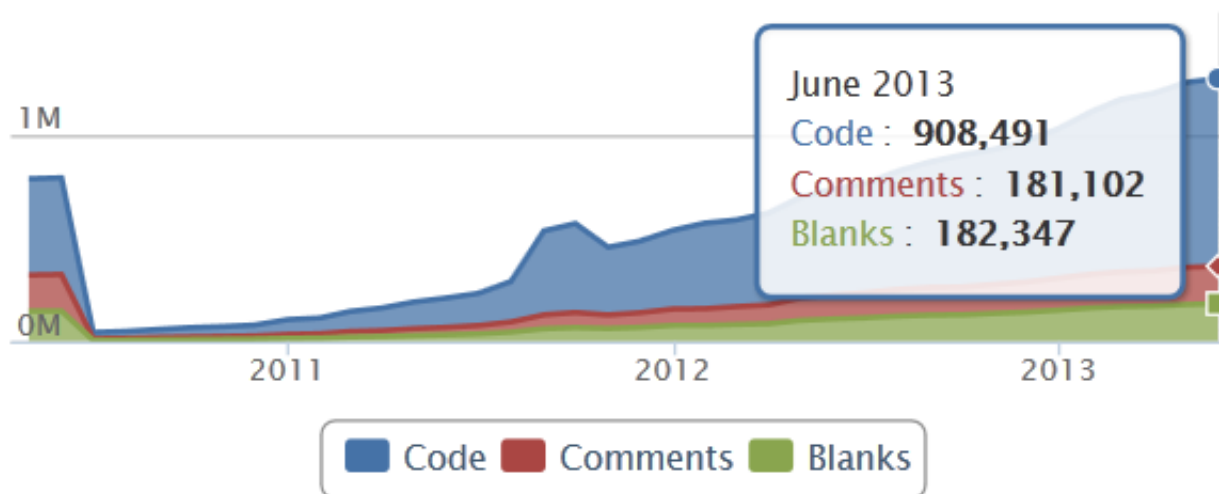


- 908,491 lines of code. OpenStack is written mostly in Python with an average number of source code comments.
- A code base with a long source history.
- Increasing Y-O-Y commits.
- A very large development team comprised of people from around the world.

**Figure 1.4. Programming Languages used to design OpenStack**  
**Languages**



## Lines of Code



For an overview of OpenStack refer to <http://www.openstack.org> or <http://goo.gl/4q7nVI>. Common questions and answers are also covered here.

### **Core Projects Overview**

Let's take a dive into some of the technical aspects of OpenStack. Its scalability and flexibility are just some of the awesome features that make it a rock-solid cloud computing platform. The OpenStack core projects serve the community and its demands.

Being a cloud computing platform, OpenStack consists of many core and incubated projects which makes it really good as an IaaS cloud computing platform/Operating System. The following points are the main components necessary to call it an OpenStack Cloud.

### **Components of OpenStack**

OpenStack has a modular architecture with various code names for its components. OpenStack has several shared services that span the three pillars of compute, storage and networking, making it easier to implement and operate your cloud. These services - including identity, image management and a web interface - integrate the OpenStack components with each other as well as external systems to provide a unified experience for users as they interact with different cloud resources.

### **Compute (Nova)**

The OpenStack cloud operating system enables enterprises and service providers to offer on-demand computing resources, by provisioning and managing large networks of virtual machines. Compute resources are accessible via APIs for developers building cloud applications and via web interfaces for administrators and users. The compute architecture is designed to scale horizontally on standard hardware.

**Figure 1.5. OpenStack Compute: Provision and manage large networks of virtual machines**



OpenStack Compute (Nova) is a cloud computing fabric controller (the main part of an IaaS system). It is written in Python and uses many external libraries such as Eventlet (for concurrent programming), Kombu (for AMQP communication), and SQLAlchemy (for database access). Nova's architecture is designed to scale horizontally on standard hardware with no proprietary hardware or software requirements and provide the ability to integrate with legacy systems and third party technologies. It is designed to manage and automate pools of computer resources and can work with widely available virtualization technologies, as well as bare metal and high-performance computing (HPC) configurations. KVM and XenServer are available choices for hypervisor technology, together with Hyper-V and Linux container technology such as LXC. In addition to different hypervisors, OpenStack runs on ARM.

**Popular Use Cases:**

- Service providers offering an IaaS compute platform or services higher up the stack
- IT departments acting as cloud service providers for business units and project teams
- Processing big data with tools like Hadoop
- Scaling compute up and down to meet demand for web resources and applications
- High-performance computing (HPC) environments processing diverse and intensive workloads

**Object Storage(Swift)**

In addition to traditional enterprise-class storage technology, many organizations now have a variety of storage needs with varying performance and price requirements. OpenStack has support for both Object Storage and Block Storage, with many deployment options for each depending on the use case.

**Figure 1.6. OpenStack Storage: Object and Block storage for use with servers and applications**



OpenStack Object Storage (Swift) is a scalable redundant storage system. Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used.

Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. Block Storage allows block devices to be exposed and connected to compute instances for expanded storage, better performance and integration with enterprise storage platforms, such as NetApp, Nexenta and SolidFire.

A few details on OpenStack's Object Storage

- OpenStack provides redundant, scalable object storage using clusters of standardized servers capable of storing petabytes of data

- Object Storage is not a traditional file system, but rather a distributed storage system for static data such as virtual machine images, photo storage, email storage, backups and archives. Having no central "brain" or master point of control provides greater scalability, redundancy and durability.
- Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster.
- Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.

### **Block Storage(Cinder)**

OpenStack Block Storage (Cinder) provides persistent block level storage devices for use with OpenStack compute instances. The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs. In addition to local Linux server storage, it can use storage platforms including Ceph, CloudByte, Coraid, EMC (VMAX and VNX), GlusterFS, IBM Storage (Storwize family, SAN Volume Controller, and XIV Storage System), Linux LIO, NetApp, Nexenta, Scalality, SolidFire and HP (Store Virtual and StoreServ 3Par families). Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage. Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

#### **A few points on OpenStack Block Storage:**

- OpenStack provides persistent block level storage devices for use with OpenStack compute instances.
- The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs.

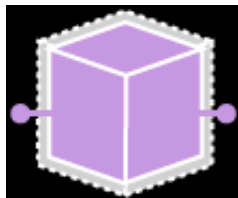
- In addition to using simple Linux server storage, it has unified storage support for numerous storage platforms including Ceph, NetApp, Nexenta, SolidFire, and Zadara.
- Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage.
- Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

### Networking(Neutron)

Today's data center networks contain more devices than ever before. From servers, network equipment, storage systems and security appliances, many of which are further divided into virtual machines and virtual networks. The number of IP addresses, routing configurations and security rules can quickly grow into the millions. Traditional network management techniques fall short of providing a truly scalable, automated approach to managing these next-generation networks. At the same time, users expect more control and flexibility with quicker provisioning.

OpenStack Networking is a pluggable, scalable and API-driven system for managing networks and IP addresses. Like other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing data center assets. OpenStack Networking ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self-service, even over their network configurations.

**Figure 1.7. OpenStack Networking: Pluggable, scalable, API-driven network and IP management**



OpenStack Networking (Neutron, formerly Quantum) is a system for managing networks and IP addresses. Like other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing data center assets. OpenStack Networking ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self-service, even over their network configurations.

OpenStack Neutron provides networking models for different applications or user groups. Standard models include flat networks or VLANs for separation of servers and traffic. OpenStack Networking manages IP addresses, allowing for dedicated static IPs or DHCP. Floating IPs allow traffic to be dynamically re routed to any of your compute resources, which allows you to redirect traffic during maintenance or in the case of failure. Users can create their own networks, control traffic and connect servers and devices to one or more networks. Administrators can take advantage of software-defined networking (SDN) technology like OpenFlow to allow for high levels of multi-tenancy and massive scale. OpenStack Networking has an extension framework allowing additional network services, such as intrusion detection systems (IDS), load balancing, firewalls and virtual private networks (VPN) to be deployed and managed.

#### Networking Capabilities

- OpenStack provides flexible networking models to suit the needs of different applications or user groups. Standard models include flat networks or VLANs for separation of servers and traffic.
- OpenStack Networking manages IP addresses, allowing for dedicated static IPs or DHCP. Floating IPs allow traffic to be dynamically re-routed to any of your compute resources, which allows you to redirect traffic during maintenance or in the case of failure.
- Users can create their own networks, control traffic and connect servers and devices to one or more networks.
- The pluggable backend architecture lets users take advantage of commodity gear or advanced networking services from supported vendors.
- Administrators can take advantage of software-defined networking (SDN) technology like OpenFlow to allow for high levels of multi-tenancy and massive scale.



- OpenStack Networking has an extension framework allowing additional network services, such as intrusion detection systems (IDS), load balancing, firewalls and virtual private networks (VPN) to be deployed and managed.

### **Dashboard(Horizon)**

OpenStack Dashboard (Horizon) provides administrators and users a graphical interface to access, provision and automate cloud-based resources. The design allows for third party products and services, such as billing, monitoring and additional management tools. Service providers and other commercial vendors can customize the dashboard with their own brand.

The dashboard is just one way to interact with OpenStack resources. Developers can automate access or build tools to manage their resources using the native OpenStack API or the EC2 compatibility API.

### **Identity Service(Keystone)**

OpenStack Identity (Keystone) provides a central directory of users mapped to the OpenStack services they can access. It acts as a common authentication system across the cloud operating system and can integrate with existing backend directory services like LDAP. It supports multiple forms of authentication including standard username and password credentials, token-based systems, and Amazon Web Services log in credentials such as those used for EC2.

Additionally, the catalog provides a query-able list of all of the services deployed in an OpenStack cloud in a single registry. Users and third-party tools can programmatically determine which resources they can access.

The OpenStack Identity Service enables administrators to:

- Configure centralized policies across users and systems
- Create users and tenants and define permissions for compute, storage, and networking resources by using role-based access control (RBAC) features
- Integrate with an existing directory, like LDAP, to provide a single source of authentication across the enterprise

The OpenStack Identity Service enables users to:

- List the services to which they have access
- Make API requests
- Log into the web dashboard to create resources owned by their account

### **Image Service(Glance)**

OpenStack Image Service (Glance) provides discovery, registration and delivery services for disk and server images. Stored images can be used as a template. They can also be used to store and catalog an unlimited number of backups. The Image Service can store disk and server images in a variety of back-ends, including OpenStack Object Storage. The Image Service API provides a standard REST interface for querying information about disk images and lets clients stream the images to new servers.

Capabilities of the Image Service include:

- Administrators can create base templates from which their users can start new compute instances
- Users can choose from available images, or create their own from existing servers
- Snapshots can also be stored in the Image Service so that virtual machines can be backed up quickly

A multi-format image registry, the image service allows uploads of private and public images in a variety of formats, including:

- Raw
- Machine (kernel/ramdisk outside of image, also known as AMI)
- VHD (Hyper-V)
- VDI (VirtualBox)

- qcow2 (Qemu/KVM)
- VMDK (VMWare)
- OVF (VMWare, others)

To checkout the complete list of Core and Incubated projects under OpenStack check out OpenStack's Launchpad Project Page here : <http://goo.gl/ka4SrV>

### **Amazon Web Services compatibility**

OpenStack APIs are compatible with Amazon EC2 and Amazon S3 and thus client applications written for Amazon Web Services can be used with OpenStack with minimal porting effort.

### **Governance**

OpenStack is governed by a non-profit foundation and its board of directors, a technical committee and a user committee.

The foundation's stated mission is by providing shared resources to help achieve the OpenStack Mission by Protecting, Empowering, and Promoting OpenStack software and the community around it, including users, developers and the entire ecosystem. Though, it has little to do with the development of the software, which is managed by the technical committee - an elected group that represents the contributors to the project, and has oversight on all technical matters.

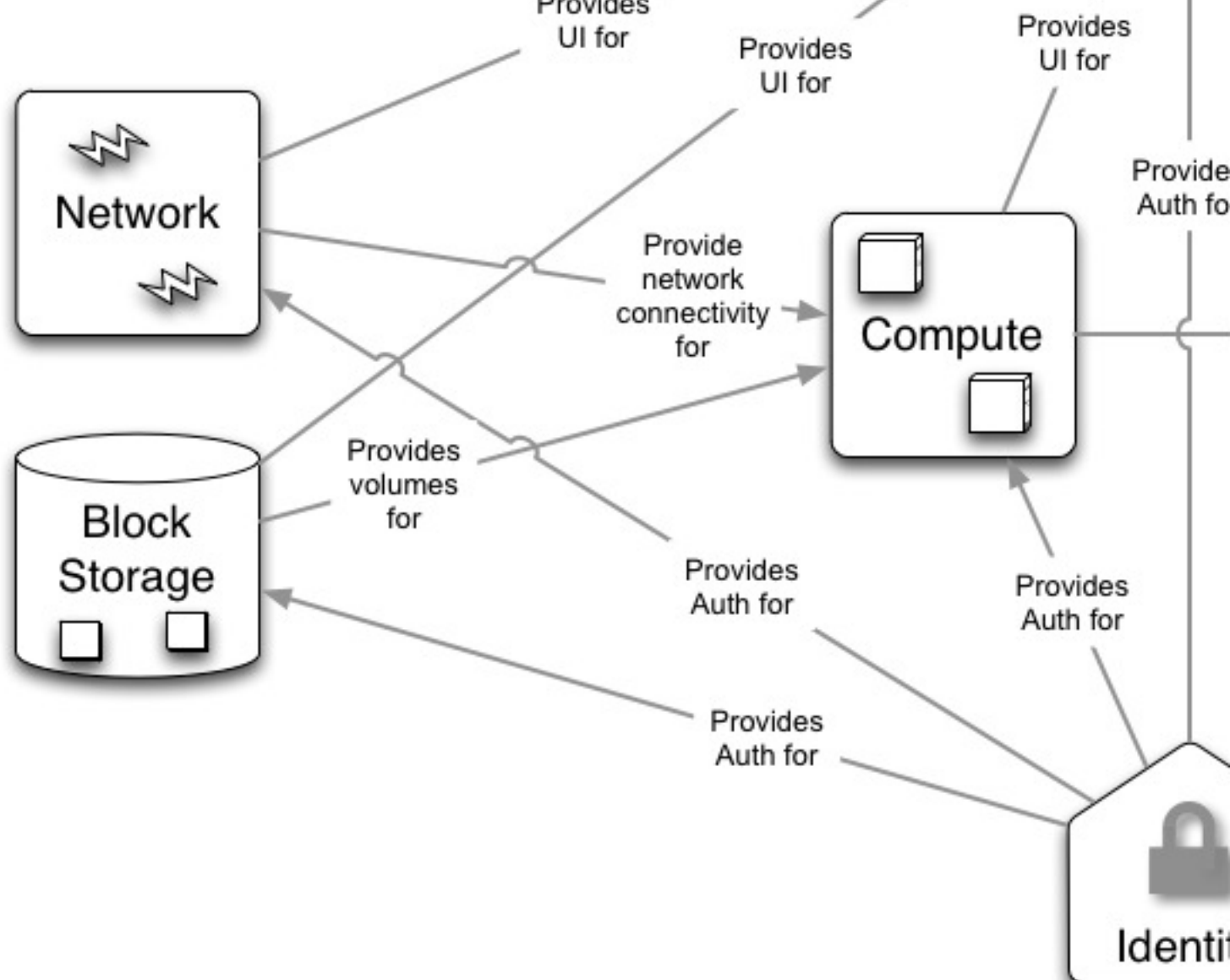
## **OpenStack Architecture**

### **Conceptual Architecture**

The OpenStack project as a whole is designed to deliver a massively scalable cloud operating system. To achieve this, each of the constituent services are designed to work together to provide a complete

Infrastructure-as-a-Service (IaaS). This integration is facilitated through public application programming interfaces (APIs) that each service offers (and in turn can consume). While these APIs allow each of the services to use another service, it also allows an implementer to switch out any service as long as they maintain the API. These are (mostly) the same APIs that are available to end users of the cloud.

Conceptually, you can picture the relationships between the services as so:



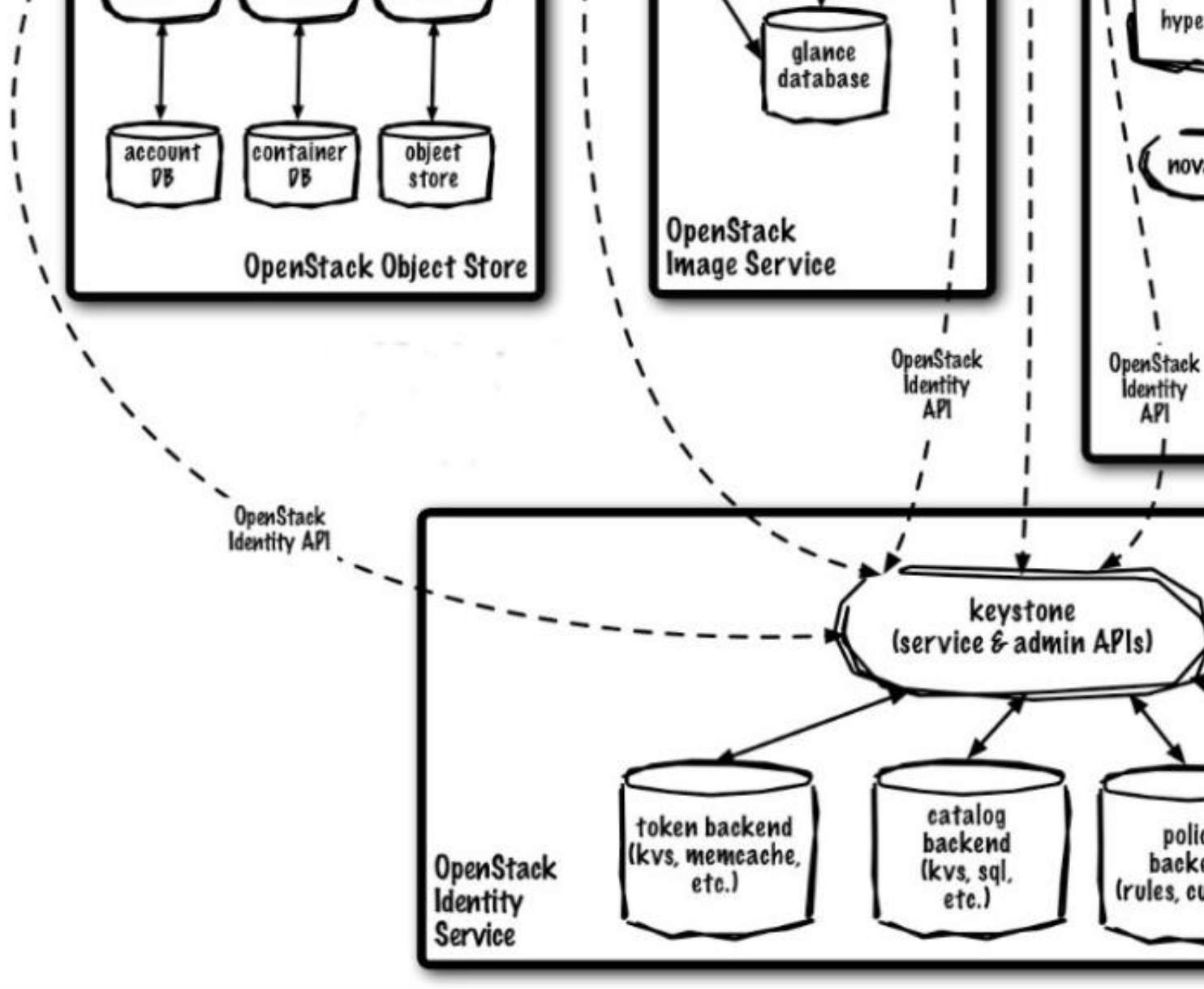
- Dashboard ("Horizon") provides a web front end to the other OpenStack services
- Compute ("Nova") stores and retrieves virtual disks ("images") and associated metadata in Image ("Glance")
- Network ("Neutron") provides virtual networking for Compute.
- Block Storage ("Cinder") provides storage volumes for Compute.
- Image ("Glance") can store the actual virtual disk files in the Object Store("Swift")
- All the services authenticate with Identity ("Keystone")

This is a stylized and simplified view of the architecture, assuming that the implementer is using all of the services together in the most common configuration. It also only shows the "operator" side of the cloud – it does not picture how consumers of the cloud may actually use it. For example, many users will access object storage heavily (and directly).

### **Logical Architecture**

This picture is consistent with the conceptual architecture above:

OpenStack



- End users can interact through a common web interface (Horizon) or directly to each service through their API
- All services authenticate through a common source (facilitated through keystone)
- Individual services interact with each other through their public APIs (except where privileged administrator commands are necessary)

In the sections below, we'll delve into the architecture for each of the services.

### **Dashboard**

Horizon is a modular Django web application that provides an end user and administrator interface to OpenStack services.



PROJECT  
**invisible\_to\_admin** ▼

## Manage Compute

Overview

Instances & Volumes

Access & Security

Images & Snapshots

## Object Store

Containers

|                          | Name | IP Address | Size      |
|--------------------------|------|------------|-----------|
| <input type="checkbox"/> | test |            | 512MB RAM |

Displaying 1 item

## Volumes

|                    | Name | Description |
|--------------------|------|-------------|
| Displaying 0 items |      |             |

As with most web applications, the architecture is fairly simple:

- Horizon is usually deployed via `mod_wsgi` in Apache. The code itself is separated into a reusable python module with most of the logic (interactions with various OpenStack APIs) and presentation (to make it easily customizable for different sites).
- A database (configurable as to which one) which relies mostly on the other services for data. It also stores very little data of its own.

From a network architecture point of view, this service will need to be customer accessible as well as be able to talk to each service's public APIs. If you wish to use the administrator functionality (i.e. for other services), it will also need connectivity to their Admin API endpoints (which should be non-customer accessible).

## Compute

Nova is the most complicated and distributed component of OpenStack. A large number of processes cooperate to turn end user API requests into running virtual machines. Below is a list of these processes and their functions:

- `nova-api` accepts and responds to end user compute API calls. It supports OpenStack Compute API, Amazon's EC2 API and a special Admin API (for privileged users to perform administrative actions). It also initiates most of the orchestration activities (such as running an instance) as well as enforces some policy (mostly quota checks).
- The `nova-compute` process is primarily a worker daemon that creates and terminates virtual machine instances via hypervisor's APIs (XenAPI for XenServer/XCP, `libvirt` for KVM or QEMU, `VMwareAPI` for VMware, etc.). The process by which it does so is fairly complex but the basics are simple: accept actions from the queue and then perform a series of system commands (like launching a KVM instance) to carry them out while updating state in the database.
- `nova-volume` manages the creation, attaching and detaching of z volumes to compute instances (similar functionality to Amazon's Elastic Block Storage). It can use volumes from a variety of providers such as iSCSI or Rados Block Device in Ceph. A new OpenStack project, Cinder, will eventually replace nova-

volume functionality. In the Folsom release, nova-volume and the Block Storage service will have similar functionality.

- The nova-network worker daemon is very similar to nova-compute and nova-volume. It accepts networking tasks from the queue and then performs tasks to manipulate the network (such as setting up bridging interfaces or changing iptables rules). This functionality is being migrated to Neutron, a separate OpenStack project. In the Folsom release, much of the functionality will be duplicated between nova-network and Neutron.
- The nova-schedule process is conceptually the simplest piece of code in OpenStack Nova: it takes a virtual machine instance request from the queue and determines where it should run (specifically, which compute server host it should run on).
- The queue provides a central hub for passing messages between daemons. This is usually implemented with RabbitMQ today, but could be any AMQP message queue (such as Apache Qpid). New to the Folsom release is support for Zero MQ.
- The SQL database stores most of the build-time and runtime state for a cloud infrastructure. This includes the instance types that are available for use, instances in use, networks available and projects. Theoretically, OpenStack Nova can support any database supported by SQL-Alchemy but the only databases currently being widely used are SQLite3 (only appropriate for test and development work), MySQL and PostgreSQL.
- Nova also provides console services to allow end users to access their virtual instance's console through a proxy. This involves several daemons (nova-console, nova-novncproxy and nova-consoleauth).

Nova interacts with many other OpenStack services: Keystone for authentication, Glance for images and Horizon for web interface. The Glance interactions are central. The API process can upload and query Glance while nova-compute will download images for use in launching images.

### **Object Store**

The swift architecture is very distributed to prevent any single point of failure as well as to scale horizontally. It includes the following components:

- Proxy server (swift-proxy-server) accepts incoming requests via the OpenStack Object API or just raw HTTP. It accepts files to upload, modifications to metadata or container creation. In addition, it will also serve files or container listing to web browsers. The proxy server may utilize an optional cache (usually deployed with memcache) to improve performance.
- Account servers manage accounts defined with the object storage service.
- Container servers manage a mapping of containers (i.e folders) within the object store service.
- Object servers manage actual objects (i.e. files) on the storage nodes.
- There are also a number of periodic processes which run to perform housekeeping tasks on the large data store. The most important of these is the replication services, which ensures consistency and availability through the cluster. Other periodic processes include auditors, updaters and reapers.

Authentication is handled through configurable WSGI middleware (which will usually be Keystone).

### Image Store

The Glance architecture has stayed relatively stable since the Cactus release. The biggest architectural change has been the addition of authentication, which was added in the Diablo release. Just as a quick reminder, Glance has four main parts to it:

- glance-api accepts Image API calls for image discovery, image retrieval and image storage.
- glance-registry stores, processes and retrieves metadata about images (size, type, etc.).
- A database to store the image metadata. Like Nova, you can choose your database depending on your preference (but most people use MySQL or SQLite).
- A storage repository for the actual image files. In the diagram above, Swift is shown as the image repository, but this is configurable. In addition to Swift, Glance supports normal filesystems, RADOS block devices, Amazon S3 and HTTP. Be aware that some of these choices are limited to read-only usage.

There are also a number of periodic processes which run on Glance to support caching. The most important of these is the replication services, which ensures consistency and availability through the cluster. Other periodic processes include auditors, updaters and reapers.

As you can see from the diagram in the Conceptual Architecture section, Glance serves a central role to the overall IaaS picture. It accepts API requests for images (or image metadata) from end users or Nova components and can store its disk files in the object storage service, Swift.

### **Identity**

Keystone provides a single point of integration for OpenStack policy, catalog, token and authentication.

- Keystone handles API requests as well as providing configurable catalog, policy, token and identity services.
- Each Keystone function has a pluggable backend which allows different ways to use the particular service. Most support standard backends like LDAP or SQL, as well as Key Value Stores (KVS).

Most people will use this as a point of customization for their current authentication services.

### **Network**

Neutron provides "network connectivity as a service" between interface devices managed by other OpenStack services (most likely Nova). The service works by allowing users to create their own networks and then attach interfaces to them. Like many of the OpenStack services, Neutron is highly configurable due to its plug-in architecture. These plug-ins accommodate different networking equipment and software. As such, the architecture and deployment can vary dramatically. In the above architecture, a simple Linux networking plug-in is shown.

- neutron-server accepts API requests and then routes them to the appropriate Neutron plug-in for action.
- Neutron plug-ins and agents perform the actual actions such as plugging and unplugging ports, creating networks or subnets and IP addressing. These plug-ins and agents differ depending on the vendor and

technologies used in the particular cloud. Neutron ships with plug-ins and agents for: Cisco virtual and physical switches, NEC OpenFlow products, Open vSwitch, Linux bridging, the Ryu Network Operating System, and VMware NSX.

- The common agents are L3 (layer 3), DHCP (dynamic host IP addressing) and the specific plug-in agent.
- Most Neutron installations will also make use of a messaging queue to route information between the neutron-server and various agents as well as a database to store networking state for particular plug-ins.

Neutron will interact mainly with Nova, where it will provide networks and connectivity for its instances.

### **Block Storage**

Cinder separates out the persistent block storage functionality that was previously part of OpenStack Compute (in the form of nova-volume) into its own service. The OpenStack Block Storage API allows for manipulation of volumes, volume types (similar to compute flavors) and volume snapshots.

- cinder-api accepts API requests and routes them to cinder-volume for action.
- cinder-volume acts upon the requests by reading or writing to the Cinder database to maintain state, interacting with other processes (like cinder-scheduler) through a message queue and directly upon block storage providing hardware or software. It can interact with a variety of storage providers through a driver architecture. Currently, there are drivers for IBM, SolidFire, NetApp, Nexenta, Zadara, linux iSCSI and other storage providers.
- Much like nova-scheduler, the cinder-scheduler daemon picks the optimal block storage provider node to create the volume on.
- Cinder deployments will also make use of a messaging queue to route information between the cinder processes as well as a database to store volume state.

Like Neutron, Cinder will mainly interact with Nova, providing volumes for its instances.

# Virtual Machine Provisioning Walk-Through

More Content To be Added ...

OpenStack Compute gives you a tool to orchestrate a cloud, including running instances, managing networks, and controlling access to the cloud through users and projects. The underlying open source project's name is Nova, and it provides the software that can control an Infrastructure-as-a-Service (IaaS) cloud computing platform. It is similar in scope to Amazon EC2 and Rackspace Cloud Servers. 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 web-based API.

## Hypervisors

OpenStack Compute requires a hypervisor and Compute controls the hypervisors through an API server. The process for selecting a hypervisor usually means prioritizing and making decisions based on budget and resource constraints as well as the inevitable list of supported features and required technical specifications. The majority of development is done with the KVM and Xen-based hypervisors. Refer to <http://wiki.openstack.org/HypervisorSupportMatrix> <http://goo.gl/n7AXnC> for a detailed list of features and support across the hypervisors.

With OpenStack Compute, you can orchestrate clouds using multiple hypervisors in different zones. The types of virtualization standards that may be used with Compute include:

- KVM- Kernel-based Virtual Machine (visit <http://goo.gl/70dvRb>)
- LXC- Linux Containers (through libvirt) (visit <http://goo.gl/Ous3ly>)
- QEMU- Quick EMUlator (visit <http://goo.gl/WWV9IL>)
- UML- User Mode Linux (visit <http://goo.gl/4HAKJj>)
- VMWare vSphere4.1 update 1 and newer (visit <http://goo.gl/0DBeo5>)

- Xen- Xen, Citrix XenServer and Xen Cloud Platform (XCP) (visit <http://goo.gl/yXP9t1>)
- Bare Metal- Provisions physical hardware via pluggable sub-drivers. (visit <http://goo.gl/exfeSg>)

### **Users and Tenants (Projects)**

The OpenStack Compute system is designed to be used by many different cloud computing consumers or customers, basically tenants on a shared system, using role-based access assignments. Roles control the actions that a user is allowed to perform. In the default configuration, most actions do not require a particular role, but this is configurable by the system administrator editing the appropriate `policy.json` file that maintains the rules. For example, a rule can be defined so that a user cannot allocate a public IP without the admin role. A user's access to particular images is limited by tenant, but the username and password are assigned per user. Key pairs granting access to an instance are enabled per user, but quotas to control resource consumption across available hardware resources are per tenant.

While the original EC2 API supports users, OpenStack Compute adds the concept of tenants. Tenants are isolated resource containers forming the principal organizational structure within the Compute service. They consist of a separate VLAN, volumes, instances, images, keys, and users. A user can specify which tenant he or she wishes to be known as by appending `:project_id` to his or her access key. If no tenant is specified in the API request, Compute attempts to use a tenant with the same ID as the user

For tenants, quota controls are available to limit the:

- Number of volumes which may be created
- Total size of all volumes within a project as measured in GB
- Number of instances which may be launched
- Number of processor cores which may be allocated
- Floating IP addresses (assigned to any instance when it launches so the instance has the same publicly accessible IP addresses)



- Fixed IP addresses (assigned to the same instance each time it boots, publicly or privately accessible, typically private for management purposes)

### **Images and Instances**

This introduction provides a high level overview of what images and instances are and description of the life-cycle of a typical virtual system within the cloud. There are many ways to configure the details of an OpenStack cloud and many ways to implement a virtual system within that cloud. These configuration details as well as the specific command-line utilities and API calls to perform the actions described are presented in the Image Management and Volume Management chapters.

Images are disk images which are templates for virtual machine file systems. The OpenStack Image Service is responsible for the storage and management of images within OpenStack.

Instances are the individual virtual machines running on physical compute nodes. The OpenStack Compute service manages instances. Any number of instances may be started from the same image. Each instance is run from a copy of the base image so runtime changes made by an instance do not change the image it is based on. Snapshots of running instances may be taken which create a new image based on the current disk state of a particular instance.

When starting an instance a set of virtual resources known as a flavor must be selected. Flavors define how many virtual CPUs an instance has and the amount of RAM and size of its ephemeral disks. OpenStack provides a number of predefined flavors which cloud administrators may edit or add to. Users must select from the set of available flavors defined on their cloud.

Additional resources such as persistent volume storage and public IP address may be added to and removed from running instances. The examples below show the cinder-volume service which provide persistent block storage as opposed to the ephemeral storage provided by the instance flavor.

Here is an example of the life cycle of a typical virtual system within an OpenStack cloud to illustrate these concepts.

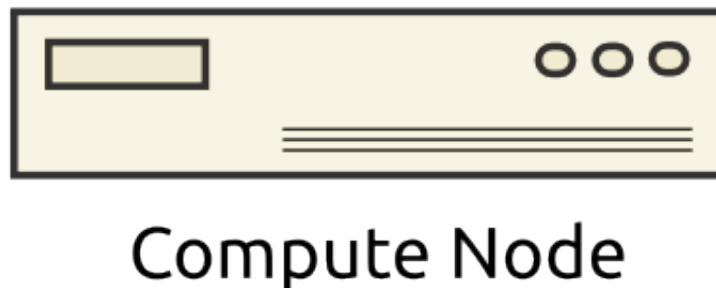
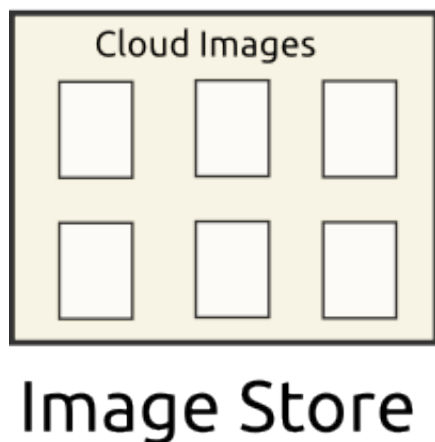
### **Initial State**

## Images and Instances

The following diagram shows the system state prior to launching an instance. The image store fronted by the Image Service has some number of predefined images. In the cloud, there is an available compute node with available vCPU, memory and local disk resources. Plus there are a number of predefined volumes in the `cinder-volume` service.

Figure 2.1. Base image state with no running instances

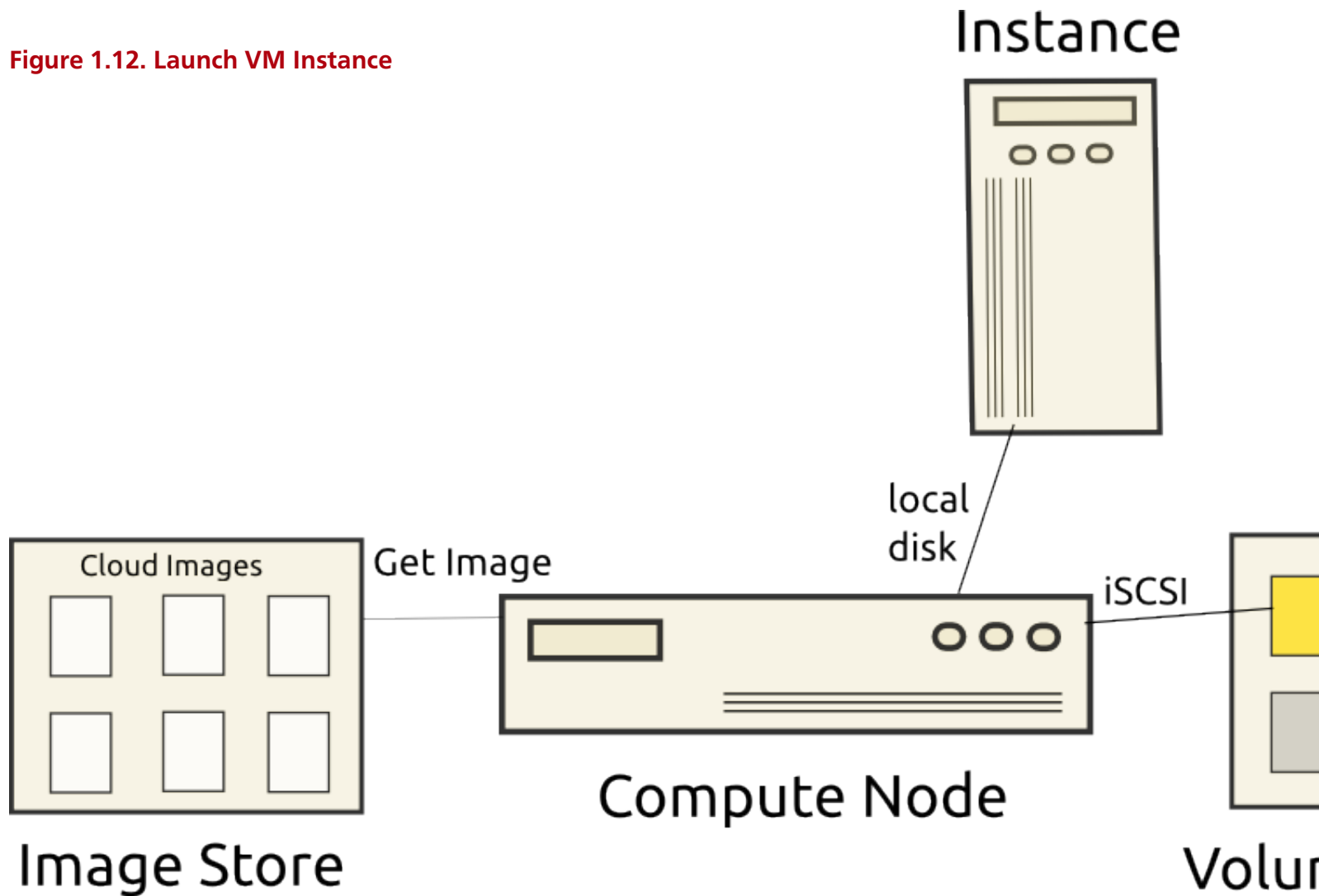
### Figure 1.11. Initial State



### Launching an instance

To launch an instance, the user selects an image, a flavor, and other optional attributes. In this case the selected flavor provides a root volume (as all flavors do) labeled `vda` in the diagram and additional ephemeral storage labeled `vdb` in the diagram. The user has also opted to map a volume from the `cinder-volume` store to the third virtual disk, `vdc`, on this instance.

Figure 2.2. Instance creation from image and run time state

**Figure 1.12. Launch VM Instance**

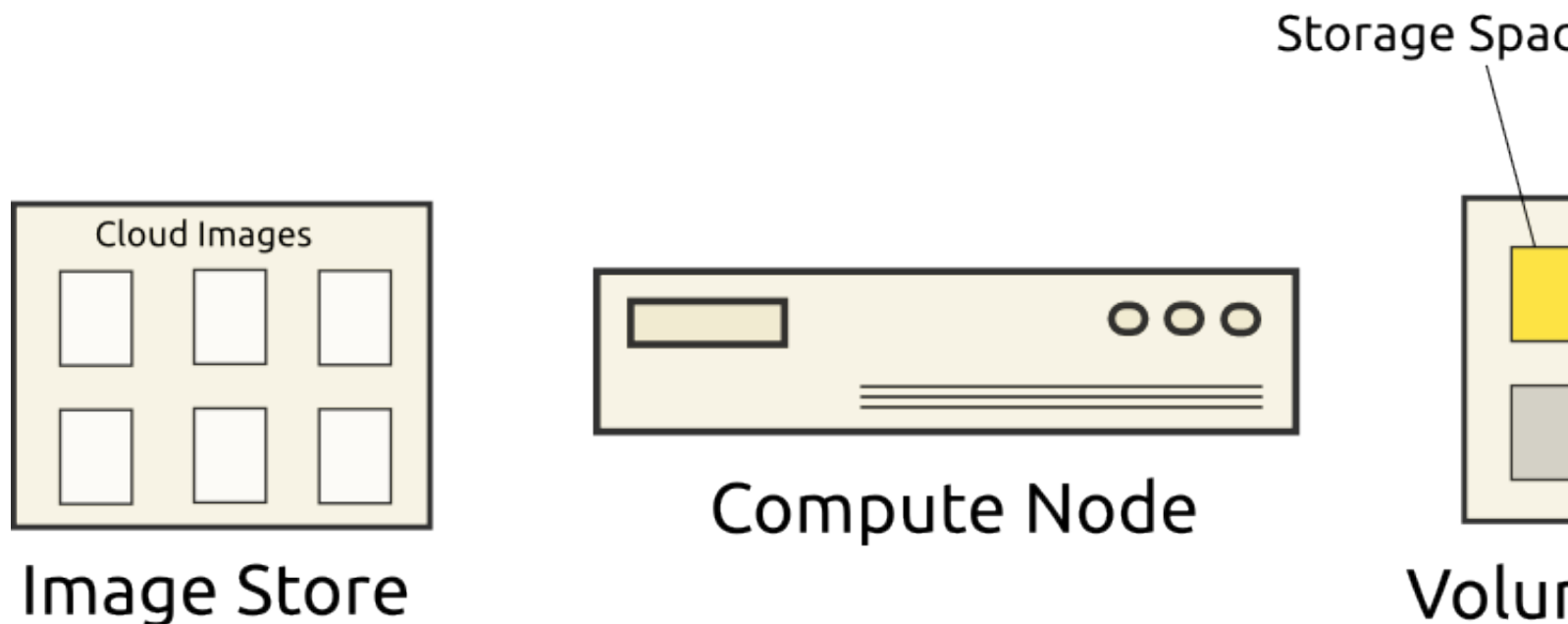
The OpenStack system copies the base image from the image store to local disk which is used as the first disk of the instance (vda). Having small images will result in faster start up of your instances as less data needs to be copied across the network. The system also creates a new empty disk image to present as the second disk (vdb). Be aware that the second disk is an empty disk with an ephemeral life as it is destroyed when you delete the instance. The compute node attaches to the requested cinder -volume using iSCSI and maps this to the third disk (vdc) as requested. The vCPU and memory resources are provisioned and the instance is booted from the first drive. The instance runs and changes data on the disks indicated in red in the diagram.

There are many possible variations in the details of the scenario, particularly in terms of what the backing storage is and the network protocols used to attach and move storage. One variant worth mentioning here is that the ephemeral storage used for volumes vda and vdb in this example may be backed by network storage rather than local disk. The details are left for later chapters.

### End State

Once the instance has served its purpose and is deleted all state is reclaimed, except the persistent volume. The ephemeral storage is purged. Memory and vCPU resources are released. And of course the image has remained unchanged throughout.

Figure 2.3. End state of image and volume after instance exits

**Figure 1.13. End State**

Once you launch a VM in OpenStack, there's something more going on in the background. To understand what's happening behind the dashboard, let's take a deeper dive into OpenStack's VM provisioning. For launching a VM, you can either use the command-line interfaces or the OpenStack dashboard.

## 2. Getting Started Quiz

### Table of Contents

|                             |    |
|-----------------------------|----|
| Day 1, 10:40 to 11:00 ..... | 41 |
|-----------------------------|----|

### Day 1, 10:40 to 11:00

**Associate Training Guide, Getting Started Quiz Questions.**

1. **What are some of the compelling features of a cloud? (choose all that apply).**
  - a. On-demand self-service
  - b. Resource pooling
  - c. Metered or measured service
  - d. Elasticity
  - e. Network access
2. **What three service models does cloud computing provide? (choose all that apply).**
  - a. Software-as-a-Service (SaaS)
  - b. Applications-as-a-Service (AaaS)

- c. Hardware-as-a-Service (HaaS)
  - d. Infrastructure-as-a-Service (IaaS)
  - e. Platform-as-a-Service (PaaS)
3. **What does the OpenStack project aim to deliver? (choose all that apply).**
- a. Simple to implement cloud solution
  - b. Massively scalable cloud solution
  - c. Feature rich cloud solution
  - d. Multi-vendor interoperability cloud solution
  - e. A new hypervisor cloud solution
4. **OpenStack code is freely available via the FreeBSD license. (True or False).**
- a. True
  - b. False
5. **OpenStack Swift is Object Storage. (True or False).**
- a. True
  - b. False
6. **OpenStack Networking is now called Quantum. (True or False).**
- a. True



b. False

**7. The Image Service (Glance) in OpenStack provides: (Choose all that apply).**

- a. Base Templates which users can start new compute instances
- b. Configuration of centralized policies across users and systems
- c. Available images for users to choose from or create their own from existing servers
- d. A central directory of users
- e. Ability to take store snapshots in the Image Service for backup

**8. OpenStack APIs are compatible with Amazon EC2 and Amazon S3. (True or False).**

- a. True
- b. False

**9. Horizon is the OpenStack name for Compute. (True or False).**

- a. True
- b. False

**10. Which Hypervisors can be supported in OpenStack? (Choose all that apply).**

- a. KVM
- b. VMware vSphere 4.1, update 1 or greater
- c. bhyve (BSD)

d. Xen

e. LXC

**Associate Training Guide, Getting Started Quiz Answers.**

1. A, B, C, D, E

2. A, D, E

3. A, B, C

4. B

5. A

6. B

7. A, C, E

8. A

9. B

10A, B, D, E

## 3. Controller Node

### Table of Contents

|                                             |     |
|---------------------------------------------|-----|
| Day 1, 11:15 to 12:30, 13:30 to 14:45 ..... | 45  |
| Overview Horizon and OpenStack CLI .....    | 45  |
| Keystone Architecture .....                 | 95  |
| OpenStack Messaging and Queues .....        | 100 |
| Administration Tasks .....                  | 111 |

### Day 1, 11:15 to 12:30, 13:30 to 14:45

### Overview Horizon and OpenStack CLI

#### How can I use an OpenStack cloud?

As an OpenStack cloud end user, you can provision your own resources within the limits set by administrators. The examples in this guide show you how to complete these tasks by using the OpenStack dashboard and command-line clients. The dashboard, also known as horizon, is a Web-based graphical interface. The command-line clients let you run simple commands to create and manage resources in a cloud and automate tasks by using scripts. Each of the core OpenStack projects has its own command-line client.

You can modify these examples for your specific use cases.

In addition to these ways of interacting with a cloud, you can access the OpenStack APIs indirectly through cURL commands or open SDKs, or directly through the APIs. You can automate access or build tools to manage resources and services by using the native OpenStack APIs or the EC2 compatibility API.

To use the OpenStack APIs, it helps to be familiar with HTTP/1.1, RESTful web services, the OpenStack services, and JSON or XML data serialization formats.

## OpenStack dashboard

As a cloud end user, the OpenStack dashboard lets you to provision your own resources within the limits set by administrators. You can modify these examples to create other types and sizes of server instances.

### Overview

The following requirements must be fulfilled to access the OpenStack dashboard:

- The cloud operator has set up an OpenStack cloud.
- You have a recent Web browser that supports HTML5. It must have cookies and JavaScript enabled. To use the VNC client for the dashboard, which is based on noVNC, your browser must support HTML5 Canvas and HTML5 WebSockets. For more details and a list of browsers that support noVNC, see <https://github.com/kanaka/noVNC/blob/master/README.md> and <https://github.com/kanaka/noVNC/wiki/Browser-support>, respectively.

Learn how to log in to the dashboard and get a short overview of the interface.

### Log in to the dashboard

To log in to the dashboard

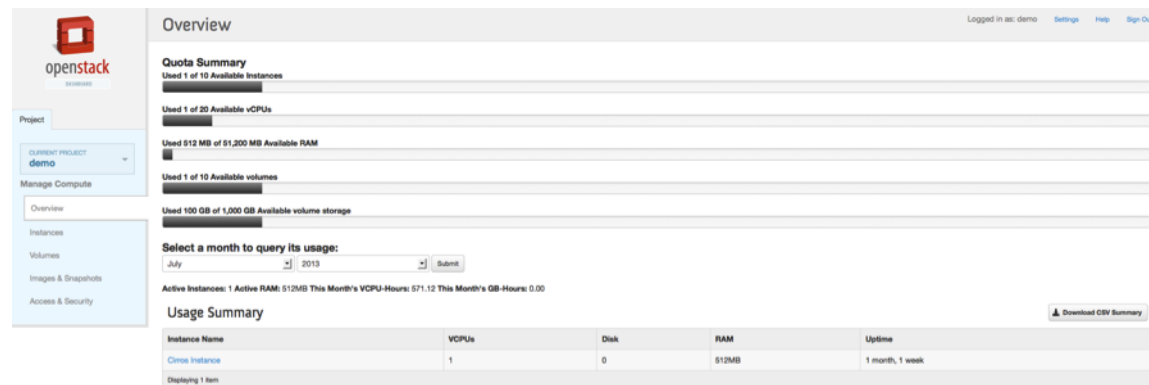
1. Ask your cloud operator for the following information:

- The hostname or public IP address from which you can access the dashboard.
- The dashboard is available on the node that has the nova-dashboard server role.

- The username and password with which you can log in to the dashboard.
1. Open a Web browser that supports HTML5. Make sure that JavaScript and cookies are enabled.
  2. As a URL, enter the host name or IP address that you got from the cloud operator.
  3. [https://IP\\_ADDRESS\\_OR\\_HOSTNAME/](https://IP_ADDRESS_OR_HOSTNAME/)
  4. On the dashboard log in page, enter your user name and password and click Sign In.

After you log in, the following page appears:

**Figure 3.1. OpenStack Dashboard - Overview**



The top-level row shows the username that you logged in with. You can also access Settings or Sign Out of the Web interface.

If you are logged in as an end user rather than an admin user, the main screen shows only the Project tab.

### OpenStack dashboard – Project tab

This tab shows details for the projects, or projects, of which you are a member.

Select a project from the drop-down list on the left-hand side to access the following categories:

### **Overview**

Shows basic reports on the project.

### **Instances**

Lists instances and volumes created by users of the project.

From here, you can stop, pause, or reboot any instances or connect to them through virtual network computing (VNC).

### **Volumes**

Lists volumes created by users of the project.

From here, you can create or delete volumes.

### **Images & Snapshots**

Lists images and snapshots created by users of the project, plus any images that are publicly available. Includes volume snapshots. From here, you can create and delete images and snapshots, and launch instances from images and snapshots.

### **Access & Security**

On the **Security Groupstab**, you can list, create, and delete security groups and edit rules for security groups.

On the **Keypairstab**, you can list, create, and import keypairs, and delete keypairs.

On the **Floating IPstab**, you can allocate an IP address to or release it from a project.

On the **API Accesstab**, you can list the API endpoints.

### **Manage images**

During setup of OpenStack cloud, the cloud operator sets user permissions to manage images. Image upload and management might be restricted to only cloud administrators or cloud operators. Though you can complete most tasks with the OpenStack dashboard, you can manage images through only the glance and nova clients or the Image Service and Compute APIs.

### **Set up access and security**

Before you launch a virtual machine, you can add security group rules to enable users to ping and SSH to the instances. To do so, you either add rules to the default security group or add a security group with rules. For information, see [the section called "Add security group rules"](#).

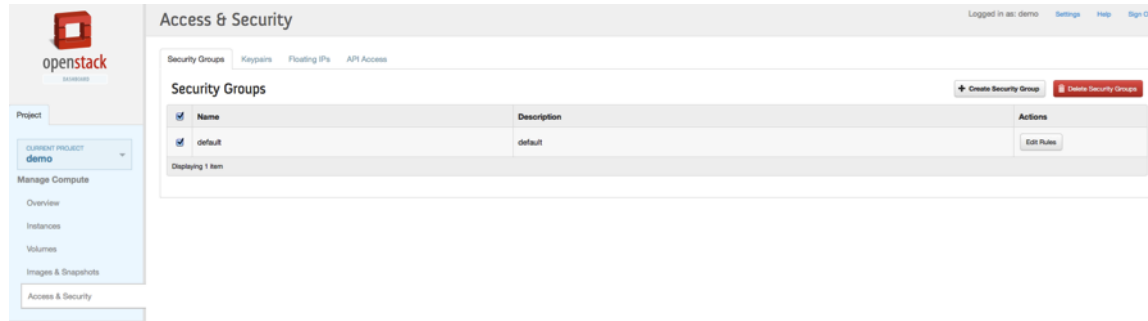
Keypairs are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the cloud-init package. For information, see [the section called "Add keypairs"](#).

### **Add security group rules**

The following procedure shows you how to add rules to the default security group.

To add rules to the default security group

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Projecttab.
3. Click the Access & Securitycategory.
4. The dashboard shows the security groups that are available for this project.

**Figure 3.2. OpenStack Dashboard - Security Groups**

1. Select the default security group and click Edit Rules.
2. The Security Group Rulespage appears:

**Figure 3.3. OpenStack Dashboard - Security Group Rules**

1. Add a TCP rule
2. Click Add Rule.



3. The Add Rulewindow appears.

1. In the IP Protocol list, select TCP.

2. In the Open list, select Port.

3. In the Port box, enter 22.

4. In the Source list, select CIDR.

5. In the CIDR box, enter 0.0.0.0/0.

6. Click Add.

7. Port 22 is now open for requests from any IP address.

8. If you want to accept requests from a particular range of IP addresses, specify the IP address block in the CIDR box.

1. Add an ICMP rule

2. Click Add Rule.

3. The Add Rulewindow appears.

1. In the IP Protocol list, select ICMP.

2. In the Type box, enter -1.

3. In the Code box, enter -1.

4. In the Source list, select CIDR.

5. In the CIDR box, enter 0.0.0.0/0.

6. Click Add.

### **Add keypairs**

Create at least one keypair for each project. If you have generated a keypair with an external tool, you can import it into OpenStack. The keypair can be used for multiple instances that belong to a project.

To add a keypair

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Projecttab.
3. Click the Access & Securitycategory.
4. Click the Keypairstab. The dashboard shows the keypairs that are available for this project.
5. To add a keypair
6. Click Create Keypair.
7. The Create Keypairwindow appears.
  1. In the Keypair Namebox, enter a name for your keypair.
  2. Click Create Keypair.
  3. Respond to the prompt to download the keypair.
    1. To import a keypair
    2. Click Import Keypair.
    3. The Import Keypairwindow appears.

1. In the Keypair Namebox, enter the name of your keypair.
2. In the Public Keybox, copy the public key.
3. Click Import Keypair.
1. Save the \*.pem file locally and change its permissions so that only you can read and write to the file:
2. `$ chmod 0600 MY_PRIV_KEY.pem`
3. Use the `ssh-add` command to make the keypair known to SSH:
4. `$ ssh-add MY_PRIV_KEY.pem`

The public key of the keypair is registered in the Nova database.

The dashboard lists the keypair in the Access & Security category.

### **Launch instances**

Instances are virtual machines that run inside the cloud. You can launch an instance directly from one of the available OpenStack images or from an image that you have copied to a persistent volume. The OpenStack Image Service provides a pool of images that are accessible to members of different projects.

#### **Launch an instance from an image**

When you launch an instance from an image, OpenStack creates a local copy of the image on the respective compute node where the instance is started.

To launch an instance from an image

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.

3. Click the Images & Snapshotcategory.
4. The dashboard shows the images that have been uploaded to OpenStack Image Service and are available for this project.
5. Select an image and click Launch.
6. In the Launch Imagewindow, specify the following:
  1. Enter an instance name to assign to the virtual machine.
  2. From the Flavordrop-down list, select the size of the virtual machine to launch.
  3. Select a keypair.
  4. In case an image uses a static root password or a static key set (neither is recommended), you do not need to provide a keypair to launch the instance.
  5. In Instance Count, enter the number of virtual machines to launch from this image.
  6. Activate the security groups that you want to assign to the instance.
  7. Security groups are a kind of cloud firewall that define which incoming network traffic should be forwarded to instances. For details, see[the section called "Add security group rules"](#).
  8. If you have not created any specific security groups, you can only assign the instance to the default security group.
  9. If you want to boot from volume, click the respective entry to expand its options.  
Set the options as described in[http://docs.openstack.org/user-guide/content/dashboard\\_launch\\_instances.html#dashboard\\_launch\\_instances\\_from\\_volumethe](http://docs.openstack.org/user-guide/content/dashboard_launch_instances.html#dashboard_launch_instances_from_volumethe) [section called "Launch an instance from a volume"](#).

1. Click Launch Instance. The instance is started on any of the compute nodes in the cloud.

After you have launched an instance, switch to the Instances category to view the instance name, its (private or public) IP address, size, status, task, and power state.

Figure 5. OpenStack dashboard – Instances

If you did not provide a keypair, security groups, or rules so far, by default the instance can only be accessed from inside the cloud through VNC at this point. Even pinging the instance is not possible. To access the instance through a VNC console, see [http://docs.openstack.org/user-guide/content/instance\\_console.html](http://docs.openstack.org/user-guide/content/instance_console.html) the section called “Get a console to an instance”.

### Launch an instance from a volume

You can launch an instance directly from an image that has been copied to a persistent volume.

In that case, the instance is booted from the volume, which is provided by nova-volume, through iSCSI.

For preparation details, see [http://docs.openstack.org/user-guide/content/dashboard\\_manage\\_volumes.html#create\\_or\\_delete\\_volumes](http://docs.openstack.org/user-guide/content/dashboard_manage_volumes.html#create_or_delete_volumes) the section called “Create or delete a volume”.

To boot an instance from the volume, especially note the following steps:

- To be able to select from which volume to boot, launch an instance from an arbitrary image. The image you select does not boot. It is replaced by the image on the volume that you choose in the next steps.
- In case you want to boot a Xen image from a volume, note the following requirement: The image you launch in must be the same type, fully virtualized or paravirtualized, as the one on the volume.
- Select the volume or volume snapshot to boot from.
- Enter a device name. Enter vda for KVM images or xvda for Xen images.

To launch an instance from a volume

You can launch an instance directly from one of the images available through the OpenStack Image Service or from an image that you have copied to a persistent volume. When you launch an instance from a volume, the procedure is basically the same as when launching an instance from an image in OpenStack Image Service, except for some additional steps.

1. Create a volume as described in [http://docs.openstack.org/user-guide/content/dashboard\\_manage\\_volumes.html#create\\_or\\_delete\\_volumesthe](http://docs.openstack.org/user-guide/content/dashboard_manage_volumes.html#create_or_delete_volumesthe) section called "Create or delete a volume".
2. It must be large enough to store an unzipped image.
3. Create an image.
4. For details, see [Creating images manually](#) in the *OpenStack Virtual Machine Image Guide*.
5. Launch an instance.
6. Attach the volume to the instance as described in [http://docs.openstack.org/user-guide/content/dashboard\\_manage\\_volumes.html#attach\\_volumes\\_to\\_instancethe](http://docs.openstack.org/user-guide/content/dashboard_manage_volumes.html#attach_volumes_to_instancethe) section called "Attach volumes to instances".
7. Assuming that the attached volume is mounted as `/dev/vdb`, use one of the following commands to copy the image to the attached volume:
  - For a raw image:
  - `$ cat IMAGE >/dev/null`
  - Alternatively, use `dd`.
  - For a non-raw image:

- `$ qemu-img convert -O raw IMAGE /dev/vdb`
  - For a \*.tar.bz2 image:
  - `$ tar xjfO IMAGE >/dev/null`
1. Only detached volumes are available for booting. Detach the volume.
  2. To launch an instance from the volume, continue with [http://docs.openstack.org/user-guide/content/dashboard\\_launch\\_instances.html#dashboard\\_launch\\_instances\\_from\\_image](http://docs.openstack.org/user-guide/content/dashboard_launch_instances.html#dashboard_launch_instances_from_image) the section called "Launch an instance from an image".
  3. You can launch an instance directly from one of the images available through the OpenStack Image Service. When you do that, OpenStack creates a local copy of the image on the respective compute node where the instance is started.
  4. SSH in to your instance

To SSH into your instance, you use the downloaded keypair file.

To SSH into your instance

1. Copy the IP address for your instance.
2. Use the SSH command to make a secure connection to the instance. For example:
3. `$ ssh -i MyKey.pem ubuntu@10.0.0.2`
4. A prompt asks, "Are you sure you want to continue connection (yes/no)?" Type yes and you have successfully connected.

## **Manage instances**

### **Create instance snapshots**

**Figure 3.4. OpenStack Dashboard- Instances**

The screenshot shows the OpenStack Dashboard's 'Instances' page. On the left is a sidebar with the OpenStack logo and navigation links: Project, Manage Compute, Overview, Instances (selected), Volumes, Images & Snapshots, and Access & Security. The main content area is titled 'Instances' and shows a table of two instances. The table has columns: Instance Name, IP Address, Size, Keypair, Status, Task, Power State, and Actions. The first instance is 'Instance from volume' with IP 10.0.0.3 and size m1.tiny | 512MB RAM | 1 VCPU | 0 Disk. The second is 'myserver' with IP 10.0.0.2 and size m1.small | 2GB RAM | 1 VCPU | 20GB Disk. Both are 'Active' and 'Running'. The 'Actions' column for each has a 'Create Snapshot' button. Above the table are buttons for 'Launch Instance' and 'Terminate Instances'. The top right shows 'Logged in as: demo' with 'Settings' and 'Sign Out' links.

| Instance Name        | IP Address | Size                                    | Keypair | Status | Task | Power State | Actions         |
|----------------------|------------|-----------------------------------------|---------|--------|------|-------------|-----------------|
| Instance from volume | 10.0.0.3   | m1.tiny   512MB RAM   1 VCPU   0 Disk   | -       | Active | None | Running     | Create Snapshot |
| myserver             | 10.0.0.2   | m1.small   2GB RAM   1 VCPU   20GB Disk | -       | Active | None | Running     | Create Snapshot |

To create instance snapshots

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.
3. Click the Instances category.
4. The dashboard lists the instances that are available for this project.
5. Select the instance of which to create a snapshot. From the Actions drop-down list, select Create Snapshot.
6. In the Create Snapshot window, enter a name for the snapshot. Click Create Snapshot. The dashboard shows the instance snapshot in the Images & Snapshots category.
7. To launch an instance from the snapshot, select the snapshot and click Launch. Proceed with <http://docs.openstack.org/user-guide/content/>



[dashboard\\_launch\\_instances.html#dashboard\\_launch\\_instances\\_from\\_image](#) the section called “Launch an instance from an image”.

### **Control the state of an instance**

To control the state of an instance

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.
3. Click the Instances category.
4. The dashboard lists the instances that are available for this project.
5. Select the instance for which you want to change the state.
6. In the More drop-down list in the Actions column, select the state.
7. Depending on the current state of the instance, you can choose to pause, un-pause, suspend, resume, soft or hard reboot, or terminate an instance.

**Figure 3.5. OpenStack Dashboard : Actions**

+

Launch Instance

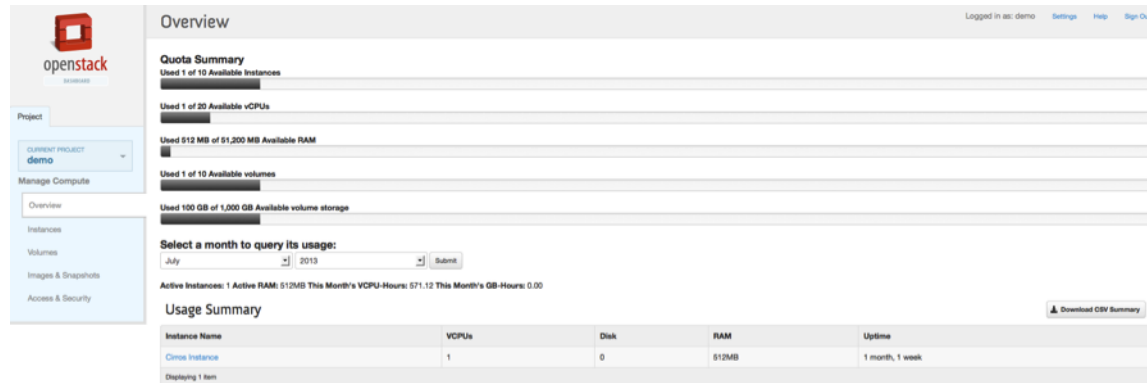
Terminate Instances

|  | Keypair    | Status | Task | Power State | Actions                                                                                                                                                                                                                                                                                                                                                                |
|--|------------|--------|------|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | OS-keypair | Active | None | Running     | <div>Create Snapshot</div> <div>More ▾</div> <div><div>Associate Floating IP</div><div>Disassociate Floating IP</div><div>Edit Instance</div><div>Edit Security Groups</div><div>Console</div><div>View Log</div><div>Pause Instance</div><div>Suspend Instance</div><div>Soft Reboot Instance</div><div>Hard Reboot Instance</div><div>Terminate Instance</div></div> |

## Track usage

Use the dashboard's Overviewcategory to track usage of instances for each project.

**Figure 3.6. OpenStack Dashboard - Track Usage**



You can track costs per month by showing metrics like number of VCPUs, disks, RAM, and uptime of all your instances.

To track usage

1. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.
2. Select a month and click Submit to query the instance usage for that month.
3. Click Download CSV Summary to download a CSV summary.

## Manage volumes

Volumes are block storage devices that you can attach to instances. They allow for persistent storage as they can be attached to a running instance, or detached and attached to another instance at any time.

In contrast to the instance's root disk, the data of volumes is not destroyed when the instance is deleted.

### **Create or delete a volume**

To create or delete a volume

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a Project from the drop-down list at the top of the tab.
3. Click the Volumes category.
4. To create a volume
  1. Click Create Volume.
  2. In the window that opens, enter a name to assign to a volume, a description (optional), and define the size in GBs.
  3. Confirm your changes.
  4. The dashboard shows the volume in the Volumes category.
1. To delete one or multiple volumes
  1. Activate the checkboxes in front of the volumes that you want to delete.
  2. Click Delete Volumes and confirm your choice in the pop-up that appears.
  3. A message indicates whether the action was successful.

After you create one or more volumes, you can attach them to instances.

You can attach a volume to one instance at a time.

View the status of a volume in the Instances & Volumes category of the dashboard: the volume is either available or In-Use.

### **Attach volumes to instances**

To attach volumes to instances

1. Log in to OpenStack dashboard.
2. If you are a member of multiple projects, select a Project from the drop-down list at the top of the tab.
3. Click the Volumes category.
4. Select the volume to add to an instance and click Edit Attachments.
5. In the Manage Volume Attachments window, select an instance.
6. Enter a device name under which the volume should be accessible on the virtual machine.
7. Click Attach Volume to confirm your changes. The dashboard shows the instance to which the volume has been attached and the volume's device name.
8. Now you can log in to the instance, mount the disk, format it, and use it.
9. To detach a volume from an instance
  1. Select the volume and click Edit Attachments.
  2. Click Detach Volume and confirm your changes.
  3. A message indicates whether the action was successful.

## OpenStack command-line clients

### Overview

You can use the OpenStack command-line clients to run simple commands that make API calls and automate tasks by using scripts. Internally, each client command runs cURL commands that embed API requests. The OpenStack APIs are RESTful APIs that use the HTTP protocol, including methods, URIs, media types, and response codes.

These open-source Python clients run on Linux or Mac OS X systems and are easy to learn and use. Each OpenStack service has its own command-line client. On some client commands, you can specify a `debug` parameter to show the underlying API request for the command. This is a good way to become familiar with the OpenStack API calls.

The following command-line clients are available for the respective services' APIs:

`cinder`(`python-cinderclient`)

Client for the Block Storage service API. Use to create and manage volumes.

`glance`(`python-glanceclient`)

Client for the Image Service API. Use to create and manage images.

`keystone`(`python-keystoneclient`)

Client for the Identity Service API. Use to create and manage users, tenants, roles, endpoints, and credentials.

`nova`(`python-novaclient`)

Client for the Compute API and its extensions. Use to create and manage images, instances, and flavors.

`neutron`(`python-neutronclient`)

Client for the Networking API. Use to configure networks for guest servers. This client was previously known as neutron.

swift(python-swiftclient)

Client for the Object Storage API. Use to gather statistics, list items, update metadata, upload, download and delete files stored by the object storage service. Provides access to a swift installation for ad hoc processing.

heat(python-heatclient)

Client for the Orchestration API. Use to launch stacks from templates, view details of running stacks including events and resources, and update and delete stacks.

### Install the OpenStack command-line clients

To install the clients, install the prerequisite software and the Python package for each OpenStack client.

#### Install the clients

Use pip to install the OpenStack clients on a Mac OS X or Linux system. It is easy and ensures that you get the latest version of the client from the <http://pypi.python.org/pypi> Python Package Index. Also, pip lets you update or remove a package. After you install the clients, you must source an openrc file to set required environment variables before you can request OpenStack services through the clients or the APIs.

To install the clients

1. You must install each client separately.
2. Run the following command to install or update a client package:

```
# pip install [--update] python-<project>client
```

Where <project> is the project name and has one of the following values:

- nova. Compute API and extensions.
- neutron. Networking API.
- keystone. Identity Service API.
- glance. Image Service API.
- swift. Object Storage API.
- cinder. Block Storage service API.
- heat. Orchestration API.

3. For example, to install the nova client, run the following command:

```
# pip install python-novaclient
```

4. To update the nova client, run the following command:

```
# pip install --upgrade python-novaclient
```

5. To remove the nova client, run the following command:

```
# pip uninstall python-novaclient
```

6. Before you can issue client commands, you must download and source the openrc file to set environment variables. Proceed to [the section called "OpenStack RC file"](#).

### Get the version for a client

After you install an OpenStack client, you can search for its version number, as follows:

```
$ pip freeze | grep python-
```



```
python-glanceclient==0.4.0python-keystoneclient==0.1.2-e git+https://github.com/openstack/python-novaclient.git@077cc0bf22e378c4c4b970f2331a695e440a939f#egg=python_novaclient-devpython-neutronclient==0.1.1python-swiftclient==1.1.1
```

You can also use the `yolk -l` command to see which version of the client is installed:

```
$ yolk -l | grep python-novaclient
```

```
python-novaclient - 2.6.10.27 - active development (/Users/your.name/src/cloud-servers/src/src/python-novaclient)python-novaclient - 2012.1 - non-active
```

### OpenStack RC file

To set the required environment variables for the OpenStack command-line clients, you must download and source an environment file, `openrc.sh`. It is project-specific and contains the credentials used by OpenStack Compute, Image, and Identity services.

When you source the file and enter the password, environment variables are set for that shell. They allow the commands to communicate to the OpenStack services that run in the cloud.

You can download the file from the OpenStack dashboard as an administrative user or any other user.

To download the OpenStack RC file

1. Log in to the OpenStack dashboard.
2. On the Projecttab, select the project for which you want to download the OpenStack RC file.
3. Click Access & Security. Then, click Download OpenStack RC Fileand save the file.
4. Copy the `openrc.sh` file to the machine from where you want to run OpenStack commands.
5. For example, copy the file to the machine from where you want to upload an image with a glance client command.

6. On any shell from where you want to run OpenStack commands, source the `openrc.sh` file for the respective project.
7. In this example, we source the `demo-openrc.sh` file for the demo project:
8. `$ source demo-openrc.sh`
9. When you are prompted for an OpenStack password, enter the OpenStack password for the user who downloaded the `openrc.sh` file.
10. When you run OpenStack client commands, you can override some environment variable settings by using the options that are listed at the end of the nova help output. For example, you can override the `OS_PASSWORD` setting in the `openrc.sh` file by specifying a password on a nova command, as follows:
11. `$ nova --password <password> image-list`
12. Where password is your password.

## Manage images

During setup of OpenStack cloud, the cloud operator sets user permissions to manage images.

Image upload and management might be restricted to only cloud administrators or cloud operators.

After you upload an image, it is considered golden and you cannot change it.

You can upload images through the glance client or the Image Service API. You can also use the nova client to list images, set and delete image metadata, delete images, and take a snapshot of a running instance to create an image.

## Manage images with the glance client

To list or get details for images

1. To list the available images:
2. `$ glance image-list`
3. You can use `grep` to filter the list, as follows:
4. `$ glance image-list | grep 'cirros'`
5. To get image details, by name or ID:
6. `$ glance image-show myCirrosImage`

To add an image

- The following example uploads a CentOS 6.3 image in qcow2 format and configures it for public access:
- `$glance image-create --name centos63-image --disk-format=qcow2 --container-format=bare --is-public=True ./centos63.qcow2`

To create an image

1. Write any buffered data to disk.
2. For more information, see the [Taking Snapshots](#) in the OpenStack Operations Guide.
3. To create the image, list instances to get the server ID:
4. `$ nova list`
5. In this example, the server is named `myCirrosServer`. Use this server to create a snapshot, as follows:
6. `$ nova image-create myCirrosServer myCirrosImage`
7. The command creates a qemu snapshot and automatically uploads the image to your repository. Only the tenant that creates the image has access to it.

8. Get details for your image to check its status:

9. `$ nova image-show IMAGE`

10. The image status changes from `SAVING` to `ACTIVE`. Only the tenant who creates the image has access to it.

To launch an instance from your image

- To launch an instance from your image, include the image ID and flavor ID, as follows:

- `$ nova boot newServer --image 7e5142af-1253-4634-bcc6-89482c5f2e8a --flavor 3`

### **Troubleshoot image creation**

- You cannot create a snapshot from an instance that has an attached volume. Detach the volume, create the image, and re-mount the volume.
- Make sure the version of qemu you are using is version 0.14 or greater. Older versions of qemu result in an "unknown option -s" error message in the nova-compute.log.
- Examine the `/var/log/nova-api.log` and `/var/log/nova-compute.log` log files for error messages.

### **Set up access and security for instances**

When you launch a virtual machine, you can inject a key pair, which provides SSH access to your instance. For this to work, the image must contain the cloud-init package. Create at least one key pair for each project. If you generate a keypair with an external tool, you can import it into OpenStack. You can use the key pair for multiple instances that belong to that project. In case an image uses a static root password or a static key set – neither is recommended – you must not provide a key pair when you launch the instance.

A security group is a named collection of network access rules that you use to limit the types of traffic that have access to instances. When you launch an instance, you can assign one or more security groups to it. If you do not create security groups, new instances are automatically assigned to the default security group,

unless you explicitly specify a different security group. The associated rules in each security group control the traffic to instances in the group. Any incoming traffic that is not matched by a rule is denied access by default. You can add rules to or remove rules from a security group. You can modify rules for the default and any other security group.

You must modify the rules for the default security group because users cannot access instances that use the default group from any IP address outside the cloud.

You can modify the rules in a security group to allow access to instances through different ports and protocols. For example, you can modify rules to allow access to instances through SSH, to ping them, or to allow UDP traffic – for example, for a DNS server running on an instance. You specify the following parameters for rules:

- Source of traffic. Enable traffic to instances from either IP addresses inside the cloud from other group members or from all IP addresses.
- Protocol. Choose TCP for SSH, ICMP for pings, or UDP.
- Destination port on virtual machine. Defines a port range. To open a single port only, enter the same value twice. ICMP does not support ports: Enter values to define the codes and types of ICMP traffic to be allowed.

Rules are automatically enforced as soon as you create or modify them.

You can also assign a floating IP address to a running instance to make it accessible from outside the cloud. You assign a floating IP address to an instance and attach a block storage device, or volume, for persistent storage.

### **Add or import keypairs**

To add a key

You can generate a keypair or upload an existing public key.

1. To generate a keypair, run the following command:
2. `$ nova keypair-add KEY_NAME > MY_KEY.pem`
3. The command generates a keypair named KEY\_NAME, writes the private key to the MY\_KEY.pem file, and registers the public key at the Nova database.
4. To set the permissions of the MY\_KEY.pem file, run the following command:
5. `$ chmod 600 MY_KEY.pem`
6. The command changes the permissions of the MY\_KEY.pem file so that only you can read and write to it.

To import a key

1. If you have already generated a keypair with the public key located at `~/.ssh/id_rsa.pub`, run the following command to upload the public key:
2. `$ nova keypair-add --pub_key ~/.ssh/id_rsa.pub KEY_NAME`
3. The command registers the public key at the Nova database and names the keypair KEY\_NAME.
4. List keypairs to make sure that the uploaded keypair appears in the list:
5. `$ nova keypair-list`

### **Configure security groups and rules**

To configure security groups

1. To list all security groups
2. To list security groups for the current project, including descriptions, enter the following command:

3. `$ nova secgroup-list`
4. To create a security group
5. To create a security group with a specified name and description, enter the following command:
6. `$ nova secgroup-create SEC_GROUP_NAME GROUP_DESCRIPTION`
7. To delete a security group
8. To delete a specified group, enter the following command:
9. `$ nova secgroup-delete SEC_GROUP_NAME`

To configure security group rules

Modify security group rules with the `nova secgroup-*-rule` commands.

1. On a shell, source the OpenStack RC file. For details, see [http://docs.openstack.org/user-guide/content/cli\\_openrc.html](http://docs.openstack.org/user-guide/content/cli_openrc.html) the section called "OpenStack RC file".
2. To list the rules for a security group
3. `$ nova secgroup-list-rules SEC_GROUP_NAME`
4. To allow SSH access to the instances
5. Choose one of the following sub-steps:
  1. Add rule for all IPs
  2. Either from all IP addresses (specified as IP subnet in CIDR notation as 0.0.0.0/0):
  3. `$ nova secgroup-add-rule SEC_GROUP_NAME tcp 22 22 0.0.0.0/0`

1. Add rule for security groups
2. Alternatively, you can allow only IP addresses from other security groups (source groups) to access the specified port:
3. `$ nova secgroup-add-group-rule --ip_proto tcp --from_port 22 \ --to_port 22 SEC_GROUP_NAME SOURCE_GROUP_NAME`
1. To allow ping to the instances
2. Choose one of the following sub-steps:
  1. To allow ping from IPs
  2. Specify all IP addresses as IP subnet in CIDR notation: 0.0.0.0/0. This command allows access to all codes and all types of ICMP traffic, respectively:
  3. `$ nova secgroup-add-rule SEC_GROUP_NAME icmp -1 -1 0.0.0.0/0`
  4. To allow ping from other security groups
  5. To allow only members of other security groups (source groups) to ping instances:
  6. `$ nova secgroup-add-group-rule --ip_proto icmp --from_port -1 \ --to_port -1 SEC_GROUP_NAME SOURCE_GROUP_NAME`
  1. To allow access through UDP port
  2. To allow access through a UDP port, such as allowing access to a DNS server that runs on a VM, complete one of the following sub-steps:
    1. To allow UDP access from IPs
    2. Specify all IP addresses as IP subnet in CIDR notation: 0.0.0.0/0.



3. `$ nova secgroup-add-rule SEC_GROUP_NAME udp 53 53 0.0.0.0/0`
4. To allow UDP access
5. To allow only IP addresses from other security groups (source groups) to access the specified port:
6. `$ nova secgroup-add-group-rule --ip_proto udp --from_port 53 \ --to_port 53 SEC_GROUP_NAME SOURCE_GROUP_NAME`
1. To delete a security group rule, specify the same arguments that you used to create the rule.
2. To delete the security rule that you created in [Step 3.a](#):
3. `$ nova secgroup-delete-rule SEC_GROUP_NAME tcp 22 22 0.0.0.0/0`
4. To delete the security rule that you created in [Step 3.b](#):
5. `$ nova secgroup-delete-group-rule --ip_proto tcp --from_port 22 \ --to_port 22 SEC_GROUP_NAME SOURCE_GROUP_NAME`

### Launch instances

Instances are virtual machines that run inside the cloud.

Before you can launch an instance, you must gather parameters such as the image and flavor from which you want to launch your instance.

You can launch an instance directly from one of the available OpenStack images or from an image that you have copied to a persistent volume. The OpenStack Image Service provides a pool of images that are accessible to members of different projects.

### Gather parameters to launch an instance

To launch an instance, you must specify the following parameters:

- The instance source, which is an image or snapshot. Alternatively, you can boot from a volume, which is block storage, to which you've copied an image or snapshot.
- The image or snapshot, which represents the operating system.
- A name for your instance.
- The flavor for your instance, which defines the compute, memory, and storage capacity of nova computing instances. 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, see Section 1.5, "Managing Flavors," (# User Guide for Administrators ).
- User Data is a special key in the metadata service which holds a file that cloud aware applications within the guest instance can access. For example the [cloudinit](#) system is an open source package from Ubuntu that handles early initialization of a cloud instance that makes use of this user data.
- Access and security credentials, which include one or both of the following credentials:
  - A key-pair for your instance, which are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the cloud-init package. Create at least one keypair for each project. If you already have generated a key-pair with an external tool, you can import it into OpenStack. You can use the keypair for multiple instances that belong to that project. For details, refer to Section 1.5.1, Creating or Importing Keys.
  - A security group, which defines which incoming network traffic is forwarded to instances. Security groups hold a set of firewall policies, known as security group rules. For details, see xx.
- If needed, you can assign a floating (public) IP address to a running instance and attach a block storage device, or volume, for persistent storage. For details, see Section 1.5.3, Managing IP Addresses and Section 1.7, Managing Volumes.

After you gather the parameters you need to launch an instance, you can launch it from an [image](#) or a [volume](#).

To gather the parameters to launch an instance

1. On a shell, source the OpenStack RC file.
2. List the available flavors:
3. `$ nova flavor-list`
4. Note the ID of the flavor that you want to use for your instance.
5. List the available images:
6. `$ nova image-list`
7. You can also filter the image list by using `grep` to find a specific image, like this:
8. `$ nova image-list | grep 'kernel'`
9. Note the ID of the image that you want to boot your instance from.
10. List the available security groups:  
`$ nova secgroup-list --all-tenants`
1. If you have not created any security groups, you can assign the instance to only the default security group.
2. You can also list rules for a specified security group:
3. `$ nova secgroup-list-rules default`
4. In this example, the default security group has been modified to allow HTTP traffic on the instance by permitting TCP traffic on Port 80.
5. List the available keypairs.

6. `$ nova keypair-list`
7. Note the name of the keypair that you use for SSH access.

### **Launch an instance from an image**

Use this procedure to launch an instance from an image.

To launch an instance from an image

1. Now you have all parameters required to launch an instance, run the following command and specify the server name, flavor ID, and image ID. Optionally, you can provide a key name for access control and security group for security. You can also include metadata key and value pairs. For example you can add a description for your server by providing the `--meta description="My Server"` parameter.
2. You can pass user data in a file on your local system and pass it at instance launch by using the flag `--user-data <user-data-file>`.
3. `$ nova boot --flavor FLAVOR_ID --image IMAGE_ID --key_name KEY_NAME --user-data mydata.file \ --security_group SEC_GROUP NAME_FOR_INSTANCE --meta KEY=VALUE --meta KEY=VALUE`
4. The command returns a list of server properties, depending on which parameters you provide.
5. A status of `BUILD` indicates that the instance has started, but is not yet online.
6. A status of `ACTIVE` indicates that your server is active.
7. Copy the server ID value from the `id` field in the output. You use this ID to get details for or delete your server.
8. Copy the administrative password value from the `adminPass` field. You use this value to log into your server.
9. Check if the instance is online:

10\$ nova list

11.This command lists all instances of the project you belong to, including their ID, their name, their status, and their private (and if assigned, their public) IP addresses.

12.If the status for the instance is ACTIVE, the instance is online.

13.To view the available options for the nova listcommand, run the following command:

14\$ nova help list

15.If you did not provide a keypair, security groups, or rules, you can only access the instance from inside the cloud through VNC. Even ping the instance is not possible.

### **Launch an instance from a volume**

After you[create a bootable volume](#), you[launch an instance from the volume](#).

To launch an instance from a volume

1. To create a bootable volume
2. To create a volume from an image, run the following command:
3. # cinder create --image-id 397e713c-b95b-4186-ad46-6126863ea0a9 --display-name my-bootable-vol 8
4. Optionally, to configure your volume, see the [Configuring Image Service and Storage for Compute](#) chapter in the OpenStack Configuration Reference.
5. To list volumes
6. Enter the following command:
7. \$ nova volume-list

8. Copy the value in the ID field for your volume.

1. To launch an instance

2. Enter the nova boot command with the `--block_device_mapping` parameter, as follows:

3. `$ nova boot --flavor <flavor> --block_device_mapping  
    <dev_name>=<id>:<type>:<size>:<delete_on_terminate> <name>`

4. The command arguments are:

5. `--flavor flavor`

6. The flavor ID.

7. `--block_device_mapping dev- name=id:type:size:delete-on-terminate`

- `dev-name`. A device name where the volume is attached in the system at `/dev/dev_name`. This value is typically `vda`.
- `id`. The ID of the volume to boot from, as shown in the output of `nova volume-list`.
- `type`. Either `snap` or any other value, including a blank string. `snap` means that the volume was created from a snapshot.
- `size`. The size of the volume, in GBs. It is safe to leave this blank and have the Compute service infer the size.
- `delete-on-terminate`. A boolean that indicates whether the volume should be deleted when the instance is terminated. You can specify
  - True or 1
  - False or 0

name

1. The name for the server.
2. For example, you might enter the following command to boot from a volume with ID bd7cf584-45de-44e3-bf7f-f7b50bf235e. The volume is not deleted when the instance is terminated:
3. 

```
$ nova boot --flavor 2 --image 397e713c-b95b-4186-ad46-6126863ea0a9 --block_device_mapping vda=bd7cf584-45de-44e3-bf7f-f7b50bf235e:::0 myInstanceFromVolume
```
4. Now when you list volumes, you can see that the volume is attached to a server:
5. 

```
$ nova volume-list
```
6. Additionally, when you list servers, you see the server that you booted from a volume:
7. 

```
$ nova list
```

### **Manage instances and hosts**

Instances are virtual machines that run inside the cloud.

### **Manage IP addresses**

Each instance can have a private, or fixed, IP address and a public, or floating, one.

Private IP addresses are used for communication between instances, and public ones are used for communication with the outside world.

When you launch an instance, it is automatically assigned a private IP address that stays the same until you explicitly terminate the instance. Rebooting an instance has no effect on the private IP address.

A pool of floating IPs, configured by the cloud operator, is available in OpenStack Compute.

You can allocate a certain number of these to a project: The maximum number of floating IP addresses per project is defined by the quota.

You can add a floating IP address from this set to an instance of the project. Floating IP addresses can be dynamically disassociated and associated with other instances of the same project at any time.

Before you can assign a floating IP address to an instance, you first must allocate floating IPs to a project. After floating IP addresses have been allocated to the current project, you can assign them to running instances.

One floating IP address can be assigned to only one instance at a time. Floating IP addresses can be managed with the nova `*floating-ip-*` commands, provided by the `python-novaclient` package.

To list pools with floating IP addresses

- To list all pools that provide floating IP addresses:
- `$ nova floating-ip-pool-list`

To allocate a floating IP address to the current project

- The output of the following command shows the freshly allocated IP address:
- `$ nova floating-ip-pool-list`
- If more than one pool of IP addresses is available, you can also specify the pool from which to allocate the IP address:
- `$ floating-ip-create POOL_NAME`

To list floating IP addresses allocated to the current project

- If an IP is already associated with an instance, the output also shows the IP for the instance, the fixed IP address for the instance, and the name of the pool that provides the floating IP address.



- `$ nova floating-ip-list`

To release a floating IP address from the current project

- The IP address is returned to the pool of IP addresses that are available for all projects. If an IP address is currently assigned to a running instance, it is automatically disassociated from the instance.
- `$ nova floating-ip-delete FLOATING_IP`

To assign a floating IP address to an instance

- To associate an IP address with an instance, one or multiple floating IP addresses must be allocated to the current project. Check this with:
- `$ nova floating-ip-list`
- In addition, you must know the instance's name (or ID). To look up the instances that belong to the current project, use the `nova list` command.
- `$ nova add-floating-ip INSTANCE_NAME_OR_ID FLOATING_IP`
- After you assign the IP with `nova add-floating-ip` and configure security group rules for the instance, the instance is publicly available at the floating IP address.

To remove a floating IP address from an instance

- To remove a floating IP address from an instance, you must specify the same arguments that you used to assign the IP.
- `$ nova remove-floating-ip INSTANCE_NAME_OR_ID FLOATING_IP`

### **Change the size of your server**

You change the size of a server by changing its flavor.

To change the size of your server

1. List the available flavors:
2. `$ nova flavor-list`
3. Show information about your server, including its size:
4. `$ nova show myCirrosServer`
5. The size of the server is m1.small (2).
6. To resize the server, pass the server ID and the desired flavor to the nova resizecommand. Include the `-poll` parameter to report the resize progress.
7. `$ nova resize myCirrosServer 4 -poll`
8. Instance resizing... 100% completeFinished
9. Show the status for your server:
10. `$ nova list`
11. When the resize completes, the status becomes VERIFY\_RESIZE. To confirm the resize:
12. `$ nova resize-confirm 6beefcf7-9de6-48b3-9ba9-e11b343189b3`
13. The server status becomes ACTIVE.
14. If the resize fails or does not work as expected, you can revert the resize:
15. `$ nova resize-revert 6beefcf7-9de6-48b3-9ba9-e11b343189b3`
16. The server status becomes ACTIVE.

**Stop and start an instance**

Use one of the following methods to stop and start an instance.

**Pause and un-pause an instance**

To pause and un-pause a server

- To pause a server, run the following command:
- `$ nova pause SERVER`
- This command stores the state of the VM in RAM. A paused instance continues to run in a frozen state.
- To un-pause the server, run the following command:
- `$ nova unpause SERVER`

**Suspend and resume an instance**

To suspend and resume a server

Administrative users might want to suspend an infrequently used instance or to perform system maintenance.

1. When you suspend an instance, its VM state is stored on disk, all memory is written to disk, and the virtual machine is stopped. Suspending an instance is similar to placing a device in hibernation; memory and vCPUs become available.
2. To initiate a hypervisor-level suspend operation, run the following command:
3. `$ nova suspend SERVER`
4. To resume a suspended server:
5. `$ nova resume SERVER`

## Reboot an instance

You can perform a soft or hard reboot of a running instance. A soft reboot attempts a graceful shutdown and restart of the instance. A hard reboot power cycles the instance.

To reboot a server

- By default, when you reboot a server, it is a soft reboot.
- `$ nova reboot SERVER`

To perform a hard reboot, pass the `–hard` parameter, as follows:

```
$ nova reboot –hard SERVER
```

## Evacuate instances

If a cloud compute node fails due to a hardware malfunction or another reason, you can evacuate instances to make them available again.

You can choose evacuation parameters for your use case.

To preserve user data on server disk, you must configure shared storage on the target host. Also, you must validate that the current VM host is down. Otherwise the evacuation fails with an error.

To evacuate your server

1. To find a different host for the evacuated instance, run the following command to lists hosts:
2. `$ nova host-list`
3. You can pass the instance password to the command by using the `–password <pwd>` option. If you do not specify a password, one is generated and printed after the command finishes successfully. The following command evacuates a server without shared storage:

4. `$ nova evacuate evacuated_server_name host_b`
5. The command evacuates an instance from a down host to a specified host. The instance is booted from a new disk, but preserves its configuration including its ID, name, uid, IP address, and so on. The command returns a password:
6. To preserve the user disk data on the evacuated server, deploy OpenStack Compute with shared filesystem.
7. `$ nova evacuate evacuated_server_name host_b --on-shared-storage`

### **Delete an instance**

When you no longer need an instance, you can delete it.

To delete an instance

1. List all instances:
2. `$ nova list`
3. Use the following command to delete the newServer instance, which is in ERROR state:
4. `$ nova delete newServer`
5. The command does not notify that your server was deleted.
6. Instead, run the nova list command:
7. `$ nova list`
8. The deleted instance does not appear in the list.

### **Get a console to an instance**

To get a console to an instance

To get a VNC console to an instance, run the following command:

```
$ nova get-vnc-console myCirrosServer xvpvnc
```

The command returns a URL from which you can access your instance:

### **Manage bare metal nodes**

If you use the bare metal driver, you must create a bare metal node and add a network interface to it. You then launch an instance from a bare metal image. You can list and delete bare metal nodes. When you delete a node, any associated network interfaces are removed. You can list and remove network interfaces that are associated with a bare metal node.

#### Commands

- `baremetal-interface-add`
- Adds a network interface to a bare metal node.
- `baremetal-interface-list`
- Lists network interfaces associated with a bare metal node.
- `baremetal-interface-remove`
- Removes a network interface from a bare metal node.
- `baremetal-node-create`
- Creates a bare metal node.
- `baremetal-node-delete`

- Removes a bare metal node and any associated interfaces.
- `baremetal-node-list`
- Lists available bare metal nodes.
- `baremetal-node-show`
- Shows information about a bare metal node.

To manage bare metal nodes

1. Create a bare metal node.
2. `$ nova baremetal-node-create --pm_address=1.2.3.4 --pm_user=ipmi --pm_password=ipmi $(hostname -f) 1 512 10 aa:bb:cc:dd:ee:ff`
3. Add network interface information to the node:
4. `$ nova baremetal-interface-add 1 aa:bb:cc:dd:ee:ff`
5. Launch an instance from a bare metal image:
6. `$ nova boot --image my-baremetal-image --flavor my-baremetal-flavor test`
7. | ... wait for instance to become active ...
8. You can list bare metal nodes and interfaces. When a node is in use, its status includes the UUID of the instance that runs on it:
9. `$ nova baremetal-node-list`
10. Show details about a bare metal node:

11\$ nova baremetal-node-show 1

### **Show usage statistics for hosts and instances**

You can show basic statistics on resource usage for hosts and instances.

To show host usage statistics

1. List the hosts and the nova-related services that run on them:
2. \$ nova host-list
3. Get a summary of resource usage of all of the instances running on the host.
4. \$ nova host-describe devstack-grizzly
5. The cpu column shows the sum of the virtual CPUs for instances running on the host.
6. The memory\_mb column shows the sum of the memory (in MB) allocated to the instances that run on the hosts.
7. The disk\_gb column shows the sum of the root and ephemeral disk sizes (in GB) of the instances that run on the hosts.

To show instance usage statistics

1. Get CPU, memory, I/O, and network statistics for an instance.
2. First, list instances:
3. \$ nova list
4. Then, get diagnostic statistics:
5. \$ nova diagnostics myCirrosServer



6. Get summary statistics for each tenant:

7. `$ nova usage-list`

8. Usage from 2013-06-25 to 2013-07-24:

### **Create and manage networks**

Before you run commands, set the following environment variables:

```
export OS_USERNAME=admin export OS_PASSWORD=password export OS_TENANT_NAME=admin export OS_AUTH_URL=http://localhost:5000/v2.0
```

To create and manage networks

1. List the extensions of the system:

2. `$ neutron ext-list -c alias -c name`

3. Create a network:

4. `$ neutron net-create net1`

5. Created a new network:

6. Create a network with specified provider network type:

7. `$ neutron net-create net2 --provider:network-type local`

8. Created a new network:

9. Just as shown previous, the unknown option `--provider:network-type` is used to create a local provider network.

10. Create a subnet:

11\$ neutron subnet-create net1 192.168.2.0/24 --name subnet1

12.Created a new subnet:

13.In the previous command, net1 is the network name, 192.168.2.0/24 is the subnet's CIDR. They are positional arguments. --name subnet1 is an unknown option, which specifies the subnet's name.

14.Create a port with specified IP address:

15\$ neutron port-create net1 --fixed-ip ip\_address=192.168.2.40

16.Created a new port:

17.In the previous command, net1 is the network name, which is a positional argument. --fixed-ip ip\_address=192.168.2.40 is an option, which specifies the port's fixed IP address we wanted.

18.Create a port without specified IP address:

19\$ neutron port-create net1

20.Created a new port:

21.We can see that the system will allocate one IP address if we don't specify the IP address in command line.

22.Query ports with specified fixed IP addresses:

23\$ neutron port-list --fixed-ips ip\_address=192.168.2.2 ip\_address=192.168.2.40

24.--fixed-ips ip\_address=192.168.2.2 ip\_address=192.168.2.40 is one unknown option.

25.How to find unknown options?The unknown options can be easily found by watching the output of create\_xxx or show\_xxx command. For example, in the port creation command, we see the fixed\_ips fields, which can be used as an unknown option.

## Create and manage stacks

To create a stack from an example template file

1. To create a stack, or template, from an [example template file](#), run following command:
2. `$ heat stack-create mystack --template-file=/path/to/heat/templates/WordPress_Single_Instance.template--parameters="InstanceType=m1.large;DBUsername=wp;DBPassword=verybadpassword;KeyName=heat_key;LinuxDistrib`
3. The `--parameters` values that you specify depend on which parameters are defined in the template. If the template file is hosted on a website, you can specify the URL with `--template-url` parameter instead of the `--template-file` parameter.
4. The command returns the following output:
5. You can also use the `stack-create` command to validate a template file without creating a stack from it.
6. To do so, run the following command:
7. `$ heat stack-create mystack --template-file=/path/to/heat/templates/WordPress_Single_Instance.template`
8. If validation fails, the response returns an error message.

To list stacks

- To see which stacks are visible to the current user, run the following command:
- `$ heat stack-list`

To view stack details

To explore the state and history of a particular stack, you can run a number of commands.

1. To show the details of a stack, run the following command:
2. `$ heat stack-show mystack`
3. A stack consists of a collection of resources. To list the resources, including their status, in a stack, run the following command:
4. `$ heat resource-list mystack`
5. To show the details for the specified resource in a stack, run the following command:
6. `$ heat resource-show mystack WikiDatabase`
7. Some resources have associated metadata which can change throughout the life-cycle of a resource:
8. `$ heat resource-metadata mystack WikiDatabase`
9. A series of events is generated during the life-cycle of a stack. This command will display those events.
10. `$ heat event-list mystack`
11. To show the details for a particular event, run the following command:
12. `$ heat event-show WikiDatabase 1`

To update a stack

- To update an existing stack from a modified template file, run a command like the following command:
- `$ heat stack-update mystack --template-file=/path/to/heat/templates/WordPress_Single_Instance_v2.template --parameters="InstanceType=m1.large;DBUsername=wp;DBPassword=verybadpassword;KeyName=heat_key;LinuxDistribution=F"`

- Some resources are updated in-place, while others are replaced with new resources.

## Keystone Architecture

The Identity service performs these functions:

- User management. Tracks users and their permissions.
- Service catalog. Provides a catalog of available services with their API endpoints.

To understand the Identity Service, you must understand these concepts:

### User

Digital representation of a person, system, or service who uses OpenStack cloud services. Identity authentication services will validate that incoming request are being made by the user who claims to be making the call. Users have a login and may be assigned tokens to access resources. Users may be directly assigned to a particular tenant and behave as if they are contained in that tenant.

### Credentials

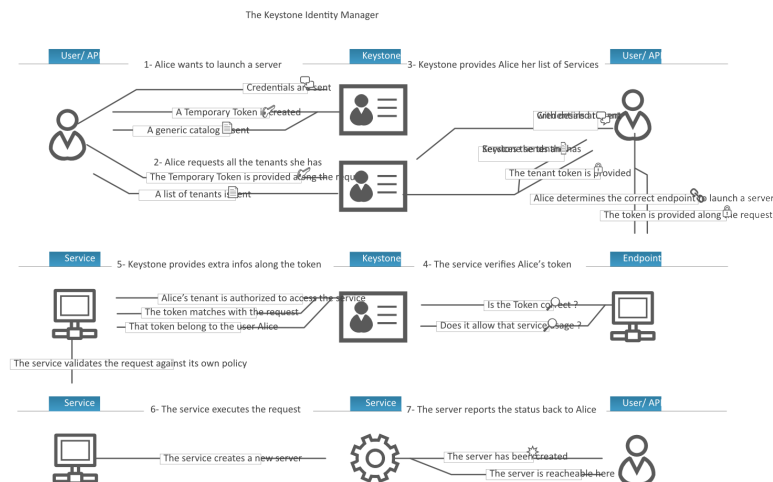
Data that is known only by a user that proves who they are. In the Identity Service, examples are:

- Username and password
- Username and API key
- An authentication token provided by the Identity Service

### Authentication

The act of confirming the identity of a user. The Identity Service confirms an incoming request by validating a set of credentials supplied by the user. These credentials are initially a username and password or a username and API key. In response to these credentials, the Identity Service issues

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                 | <p>the user an authentication token, which the user provides in subsequent requests.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Token</b>    | <p>An arbitrary bit of text that is used to access resources. Each token has a scope which describes which resources are accessible with it. A token may be revoked at anytime and is valid for a finite duration.</p> <p>While the Identity Service supports token-based authentication in this release, the intention is for it to support additional protocols in the future. The intent is for it to be an integration service foremost, and not aspire to be a full-fledged identity store and management solution.</p> |
| <b>Tenant</b>   | <p>A container used to group or isolate resources and/or identity objects. Depending on the service operator, a tenant may map to a customer, account, organization, or project.</p>                                                                                                                                                                                                                                                                                                                                         |
| <b>Service</b>  | <p>An OpenStack service, such as Compute (Nova), Object Storage (Swift), or Image Service (Glance). Provides one or more endpoints through which users can access resources and perform operations.</p>                                                                                                                                                                                                                                                                                                                      |
| <b>Endpoint</b> | <p>An network-accessible address, usually described by URL, from where you access a service. If using an extension for templates, you can create an endpoint template, which represents the templates of all the consumable services that are available across the regions.</p>                                                                                                                                                                                                                                              |
| <b>Role</b>     | <p>A personality that a user assumes that enables them to perform a specific set of operations. A role includes a set of rights and privileges. A user assuming that role inherits those rights and privileges.</p> <p>In the Identity Service, a token that is issued to a user includes the list of roles that user can assume. Services that are being called by that user determine how they interpret the set of roles a user has and which operations or resources each role grants access to.</p>                     |

**Figure 3.7. Keystone Authentication**

## User management

The main components of Identity user management are:

- Users
- Tenants
- Roles

A user represents a human user, and has associated information such as username, password and email. This example creates a user named "alice":

```
$ keystone user-create --name=alice --pass=mypassword123 --email=alice@example.com
```

A tenant can be a project, group, or organization. Whenever you make requests to OpenStack services, you must specify a tenant. For example, if you query the Compute service for a list of running instances, you get a list of all running instances for the specified tenant. This example creates a tenant named "acme":

```
$ keystone tenant-create --name=acme
```

A role captures what operations a user is permitted to perform in a given tenant. This example creates a role named "compute-user":

```
$ keystone role-create --name=compute-user
```

The Identity service associates a user with a tenant and a role. To continue with our previous examples, we may wish to assign the "alice" user the "compute-user" role in the "acme" tenant:

```
$ keystone user-list
```

```
$ keystone user-role-add --user=892585 --role=9a764e --tenant-id=6b8fd2
```

A user can be assigned different roles in different tenants. For example, Alice may also have the "admin" role in the "Cyberdyne" tenant. A user can also be assigned multiple roles in the same tenant.

The `/etc/[SERVICE_CODENAME]/policy.json` file controls what users are allowed to do for a given service. For example, `/etc/nova/policy.json` specifies the access policy for the Compute service, `/etc/glance/policy.json` specifies the access policy for the Image Service, and `/etc/keystone/policy.json` specifies the access policy for the Identity service.



The default `policy.json` files in the Compute, Identity, and Image Service recognize only the admin role: all operations that do not require the admin role will be accessible by any user that has any role in a tenant.

If you wish to restrict users from performing operations in, say, the Compute service, you need to create a role in the Identity service and then modify `/etc/nova/policy.json` so that this role is required for Compute operations.

For example, this line in `/etc/nova/policy.json` specifies that there are no restrictions on which users can create volumes: if the user has any role in a tenant, they will be able to create volumes in that tenant.

## Service Management

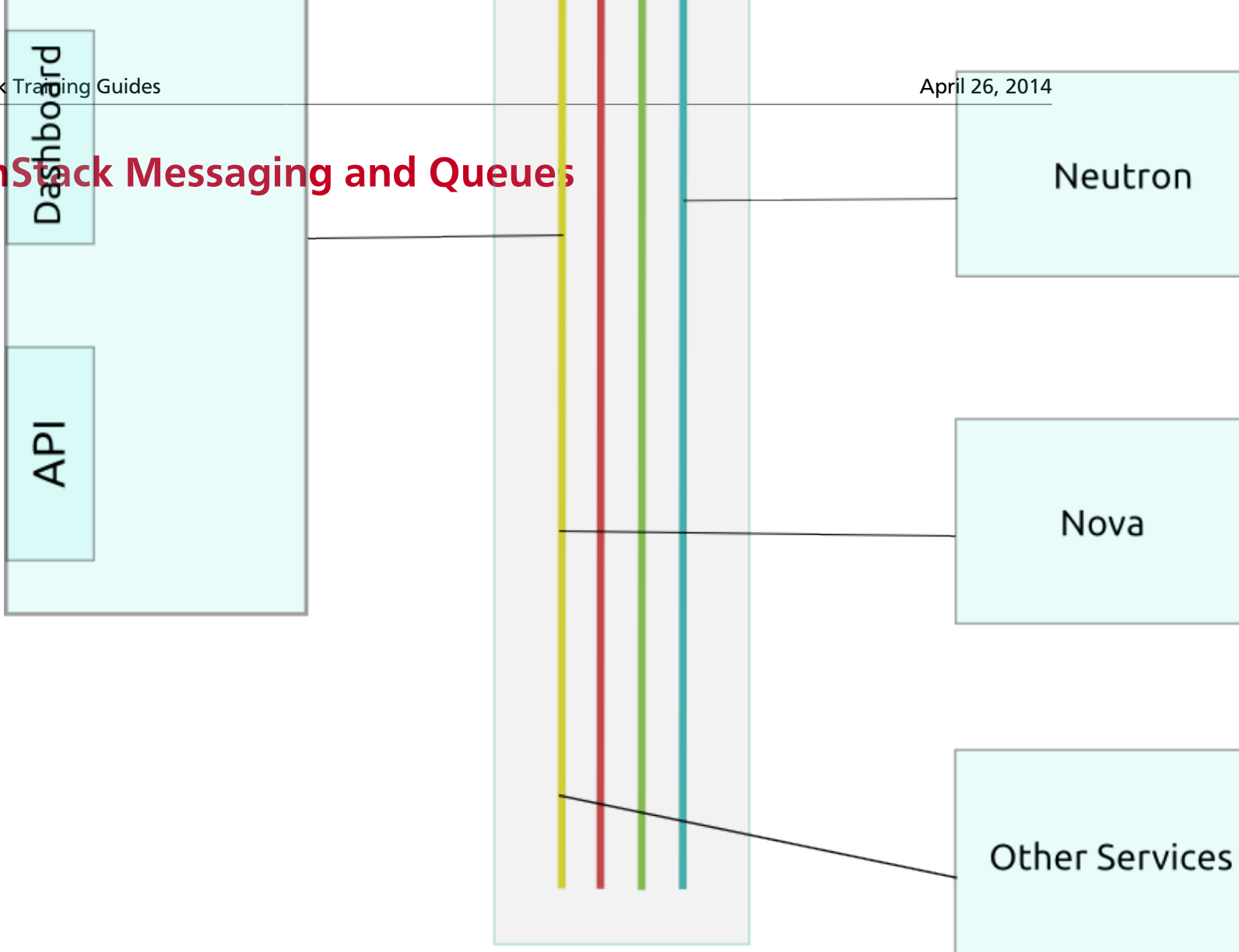
The Identity Service provides the following service management functions:

- Services
- Endpoints

The Identity Service also maintains a user that corresponds to each service, such as a user named `nova`, for the Compute service) and a special service tenant, which is called `service`.

The commands for creating services and endpoints are described in a later section.

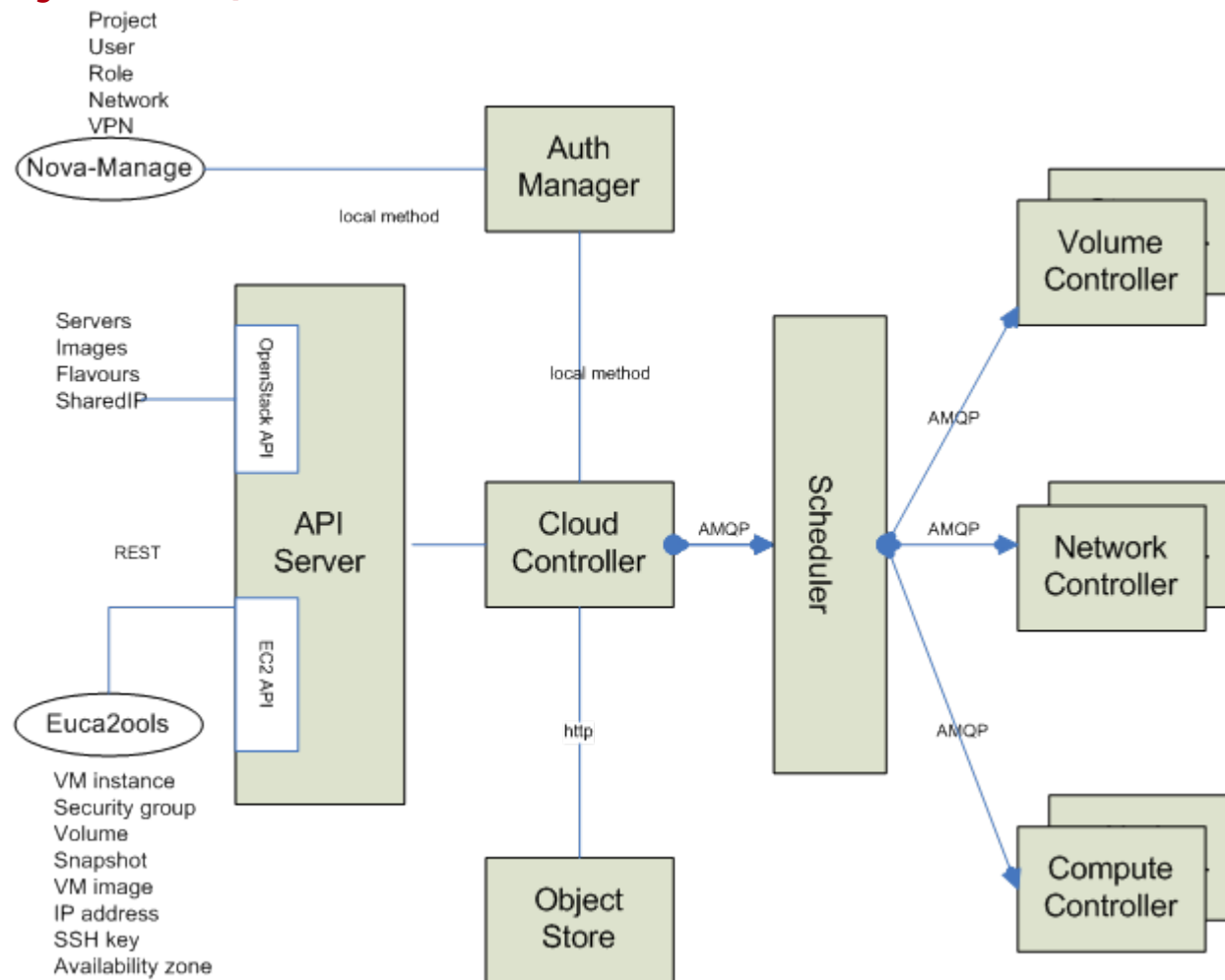
# OpenStack Messaging and Queues



AMQP is the messaging technology chosen by the OpenStack cloud. The AMQP broker, either RabbitMQ or Qpid, sits between any two Nova components and allows them to communicate in a loosely coupled fashion. More precisely, Nova components (the compute fabric of OpenStack) use Remote Procedure Calls (RPC hereinafter) to communicate to one another; however such a paradigm is built atop the publish/subscribe paradigm so that the following benefits can be achieved:

- Decoupling between client and servant (such as the client does not need to know where the servant reference is).
- Full a-synchronism between client and servant (such as the client does not need the servant to run at the same time of the remote call).
- Random balancing of remote calls (such as if more servants are up and running, one-way calls are transparently dispatched to the first available servant).

Nova uses direct, fanout, and topic-based exchanges. The architecture looks like the one depicted in the figure below:

**Figure 3.9. AMQP**

Nova implements RPC (both request+response, and one-way, respectively nicknamed 'rpc.call' and 'rpc.cast') over AMQP by providing an adapter class which take cares of marshaling and un-marshaling of messages into function calls. Each Nova service, such as Compute, Scheduler, and so on, creates two queues at the initialization time, one which accepts messages with routing keys 'NODE-TYPE.NODE-ID', for example, compute.hostname, and another, which accepts messages with routing keys as generic 'NODE-TYPE', for example compute. The former is used specifically when Nova-API needs to redirect commands to a specific node like 'euca-terminate instance'. In this case, only the compute node whose host's hypervisor is running the virtual machine can kill the instance. The API acts as a consumer when RPC calls are request/response, otherwise is acts as publisher only.

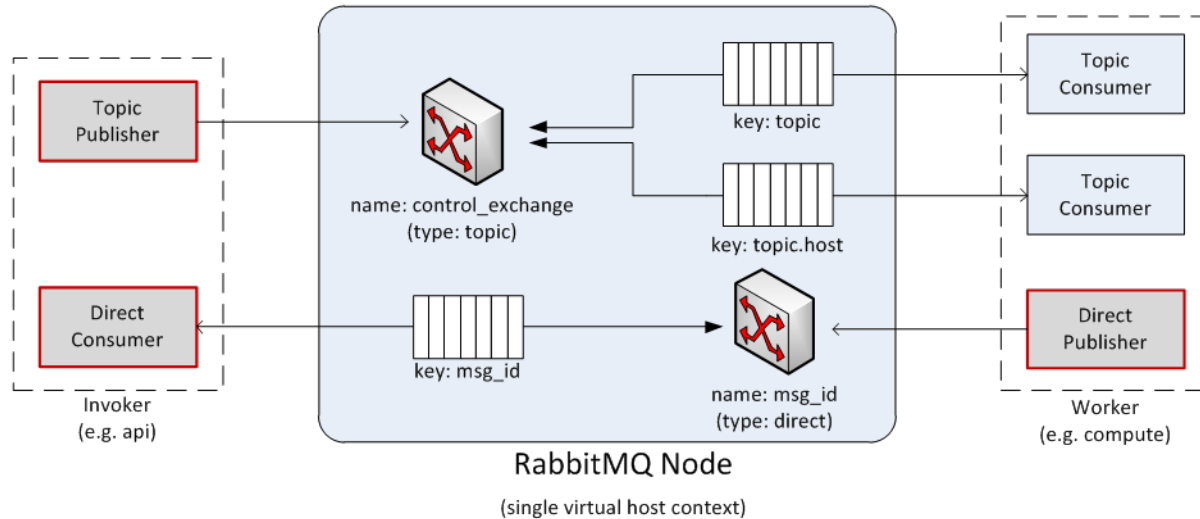
### Nova RPC Mappings

The figure below shows the internals of a message broker node (referred to as a RabbitMQ node in the diagrams) when a single instance is deployed and shared in an OpenStack cloud. Every component within Nova connects to the message broker and, depending on its personality, such as a compute node or a network node, may use the queue either as an Invoker (such as API or Scheduler) or a Worker (such as Compute or Network). Invokers and Workers do not actually exist in the Nova object model, but in this example they are used as an abstraction for the sake of clarity. An Invoker is a component that sends messages in the queuing system using **rpc.call** and **rpc.cast**. A worker is a component that receives messages from the queuing system and replies accordingly to rcp.call operations.

Figure 2 shows the following internal elements:

- **Topic Publisher:** A Topic Publisher comes to life when an rpc.call or an rpc.cast operation is executed; this object is instantiated and used to push a message to the queuing system. Every publisher connects always to the same topic-based exchange; its life-cycle is limited to the message delivery.
- **Direct Consumer:** A Direct Consumer comes to life if (an only if) a rpc.call operation is executed; this object is instantiated and used to receive a response message from the queuing system; Every consumer connects to a unique direct-based exchange via a unique exclusive queue; its life-cycle is limited to the message delivery; the exchange and queue identifiers are determined by a UUID generator, and are marshaled in the message sent by the Topic Publisher (only rpc.call operations).

- **Topic Consumer:** A Topic Consumer comes to life as soon as a Worker is instantiated and exists throughout its life-cycle; this object is used to receive messages from the queue and it invokes the appropriate action as defined by the Worker role. A Topic Consumer connects to the same topic-based exchange either via a shared queue or via a unique exclusive queue. Every Worker has two topic consumers, one that is addressed only during `rpc.cast` operations (and it connects to a shared queue whose exchange key is 'topic') and the other that is addressed only during `rpc.call` operations (and it connects to a unique queue whose exchange key is 'topic.host').
- **Direct Publisher:** A Direct Publisher comes to life only during `rpc.call` operations and it is instantiated to return the message required by the request/response operation. The object connects to a direct-based exchange whose identity is dictated by the incoming message.
- **Topic Exchange:** The Exchange is a routing table that exists in the context of a virtual host (the multi-tenancy mechanism provided by Qpid or RabbitMQ); its type (such as `topic` vs. `direct`) determines the routing policy; a message broker node will have only one topic-based exchange for every topic in Nova.
- **Direct Exchange:** This is a routing table that is created during `rpc.call` operations; there are many instances of this kind of exchange throughout the life-cycle of a message broker node, one for each `rpc.call` invoked.
- **Queue Element:** A Queue is a message bucket. Messages are kept in the queue until a Consumer (either Topic or Direct Consumer) connects to the queue and fetch it. Queues can be shared or can be exclusive. Queues whose routing key is 'topic' are shared amongst Workers of the same personality.

**Figure 3.10. RabbitMQ**

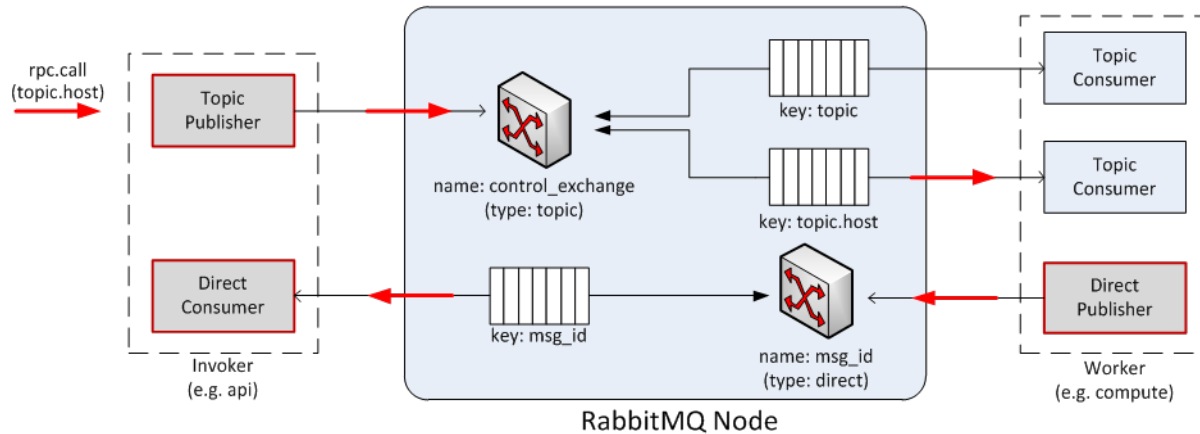
### RPC Calls

The diagram below shows the message flow during an rp.call operation:

1. A Topic Publisher is instantiated to send the message request to the queuing system; immediately before the publishing operation. A Direct Consumer is instantiated to wait for the response message.
2. Once the message is dispatched by the exchange, it is fetched by the Topic Consumer dictated by the routing key (such as 'topic.host') and passed to the Worker in charge of the task.
3. Once the task is completed, a Direct Publisher is allocated to send the response message to the queuing system.

4. Once the message is dispatched by the exchange, it is fetched by the Direct Consumer dictated by the routing key (such as 'msg\_id') and passed to the Invoker.

**Figure 3.11. RabbitMQ**

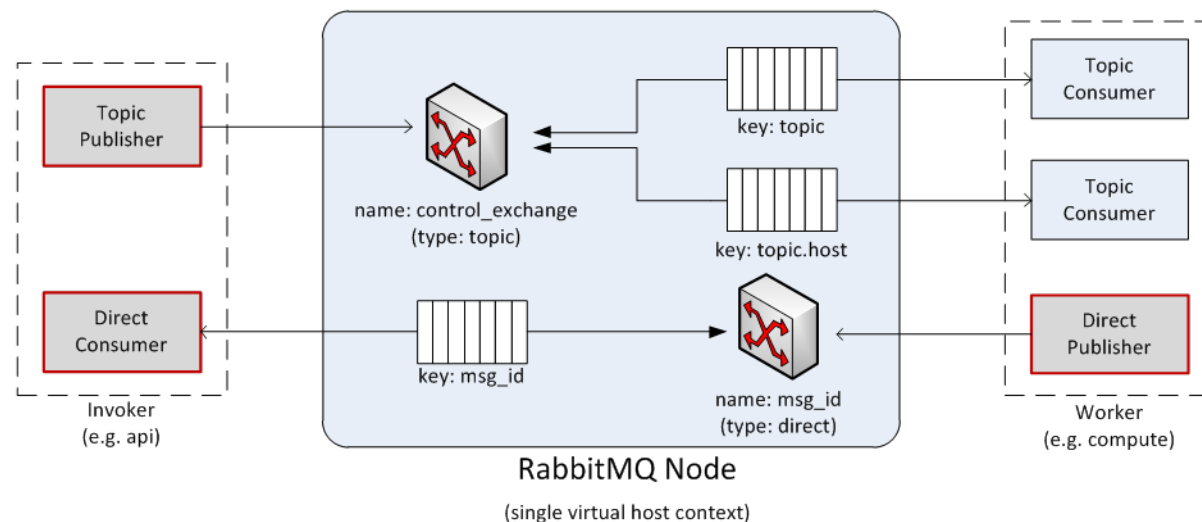


### RPC Casts

The diagram below the message flow during an `rp.cast` operation:

1. A Topic Publisher is instantiated to send the message request to the queuing system.
2. Once the message is dispatched by the exchange, it is fetched by the Topic Consumer dictated by the routing key (such as 'topic') and passed to the Worker in charge of the task.



**Figure 3.12. RabbitMQ**

### AMQP Broker Load

At any given time the load of a message broker node running either Qpid or RabbitMQ is a function of the following parameters:

- **Throughput of API calls:** the number of API calls (more precisely rpc.call ops) being served by the OpenStack cloud dictates the number of direct-based exchanges, related queues and direct consumers connected to them.
- **Number of Workers:** there is one queue shared amongst workers with the same personality; however there are as many exclusive queues as the number of workers; the number of workers dictates also the number of routing keys within the topic-based exchange, which is shared amongst all workers.

The figure below shows the status of a RabbitMQ node after Nova components' bootstrap in a test environment. Exchanges and queues being created by Nova components are:

- Exchanges

1. nova (topic exchange)

- Queues

1. compute.phantom (phantom is the hostname)
2. compute
3. network.phantom (phantom is the hostname)
4. network
5. scheduler.phantom (phantom is the hostname)
6. scheduler

### **RabbitMQ Gotchas**

Nova uses Kombu to connect to the RabbitMQ environment. Kombu is a Python library that in turn uses AMQPLib, a library that implements the standard AMQP 0.8 at the time of writing. When using Kombu, Invokers and Workers need the following parameters in order to instantiate a Connection object that connects to the RabbitMQ server (please note that most of the following material can be also found in the Kombu documentation; it has been summarized and revised here for the sake of clarity):

- **Hostname:** The hostname to the AMQP server.
- **UserId:** A valid username used to authenticate to the server.
- **Password:** The password used to authenticate to the server.

- **Virtual\_host:** The name of the virtual host to work with. This virtual host must exist on the server, and the user must have access to it. Default is `"/"`.
- **Port:** The port of the AMQP server. Default is 5672 (amqp).

The following parameters are default:

- **Insist:** Insist on connecting to a server. In a configuration with multiple load-sharing servers, the Insist option tells the server that the client is insisting on a connection to the specified server. Default is False.
- **Connect\_timeout:** The timeout in seconds before the client gives up connecting to the server. The default is no timeout.
- **SSL:** Use SSL to connect to the server. The default is False.

More precisely consumers need the following parameters:

- **Connection:** The above mentioned Connection object.
- **Queue:** Name of the queue.
- **Exchange:** Name of the exchange the queue binds to.
- **Routing\_key:** The interpretation of the routing key depends on the value of the `exchange_type` attribute.
- **Direct exchange:** If the routing key property of the message and the `routing_key` attribute of the queue are identical, then the message is forwarded to the queue.
- **Fanout exchange:** Messages are forwarded to the queues bound the exchange, even if the binding does not have a key.
- **Topic exchange:** If the routing key property of the message matches the routing key of the key according to a primitive pattern matching scheme, then the message is forwarded to the queue. The message routing

key then consists of words separated by dots (".", like domain names), and two special characters are available; star ("") and hash ("#"). The star matches any word, and the hash matches zero or more words. For example ".stock.#" matches the routing keys "usd.stock" and "eur.stock.db" but not "stock.nasdaq".

- **Durable:** This flag determines the durability of both exchanges and queues; durable exchanges and queues remain active when a RabbitMQ server restarts. Non-durable exchanges/queues (transient exchanges/queues) are purged when a server restarts. It is worth noting that AMQP specifies that durable queues cannot bind to transient exchanges. Default is True.
- **Auto\_delete:** If set, the exchange is deleted when all queues have finished using it. Default is False.
- **Exclusive:** Exclusive queues (such as non-shared) may only be consumed from by the current connection. When exclusive is on, this also implies auto\_delete. Default is False.
- **Exchange\_type:** AMQP defines several default exchange types (routing algorithms) that covers most of the common messaging use cases.
- **Auto\_ack:** Acknowledgement is handled automatically once messages are received. By default auto\_ack is set to False, and the receiver is required to manually handle acknowledgment.
- **No\_ack:** It disables acknowledgement on the server-side. This is different from auto\_ack in that acknowledgement is turned off altogether. This functionality increases performance but at the cost of reliability. Messages can get lost if a client dies before it can deliver them to the application.
- **Auto\_declare:** If this is True and the exchange name is set, the exchange will be automatically declared at instantiation. Auto declare is on by default. Publishers specify most the parameters of consumers (they do not specify a queue name), but they can also specify the following:
- **Delivery\_mode:** The default delivery mode used for messages. The value is an integer. The following delivery modes are supported by RabbitMQ:
- **1 or "transient":** The message is transient. Which means it is stored in memory only, and is lost if the server dies or restarts.

- **2 or “persistent”**: The message is persistent. Which means the message is stored both in-memory, and on disk, and therefore preserved if the server dies or restarts.

The default value is 2 (persistent). During a send operation, Publishers can override the delivery mode of messages so that, for example, transient messages can be sent over a durable queue.

## Administration Tasks

### Identity CLI Commands

Before you can use keystone client commands, you must download and source an OpenStack RC file. For information, see the *OpenStack Admin User Guide*.

The keystone command-line client uses the following syntax:

```
$ keystone PARAMETER COMMAND ARGUMENT
```

For example, you can run the **user-list** and **tenant-create** commands, as follows:

```
# Using OS_SERVICE_ENDPOINT and OS_SERVICE_TOKEN environment variables
$ export OS_SERVICE_ENDPOINT=http://127.0.0.1:5000/v2.0/
$ export OS_SERVICE_TOKEN=secrete_token
$ keystone user-list
$ keystone tenant-create --name demo
# Using --os-token and os-endpoint parameters
$ keystone --os-token token --os-endpoint endpoint user-list
$ keystone --os-token token --os-endpoint endpoint tenant-create --name demo
# Using OS_USERNAME, OS_PASSWORD, and OS_TENANT_NAME environment variables
$ export OS_USERNAME=admin
$ export OS_PASSWORD=secrete
$ export OS_TENANT_NAME=admin
$ keystone user-list
$ keystone tenant-create --name demo
```

```
# Using tenant_id parameter
$ keystone user-list --tenant_id id
# Using --name, --description, and --enabled parameters
$ keystone tenant-create --name demo --description "demo tenant" --enabled true
```

For information about using the keystone client commands to create and manage users, roles, and projects, see the *OpenStack Admin User Guide*.

## Identity User Management

The main components of Identity user management are:

- **User.** Represents a human user. Has associated information such as user name, password, and email. This example creates a user named *alice*:

```
$ keystone user-create --name=alice --pass=mypassword123 --email=alice@example.com
```

- **Tenant.** A project, group, or organization. When you make requests to OpenStack services, you must specify a tenant. For example, if you query the Compute service for a list of running instances, you get a list of all running instances in the tenant that you specified in your query. This example creates a tenant named *acme*:

```
$ keystone tenant-create --name=acme
```



### Note

Because the term *project* was used instead of *tenant* in earlier versions of OpenStack Compute, some command-line tools use `--project_id` instead of `--tenant-id` or `--os-tenant-id` to refer to a tenant ID.

- **Role.** Captures the operations that a user can perform in a given tenant.

This example creates a role named *compute-user*:

```
$ keystone role-create --name=compute-user
```



## Note

Individual services, such as Compute and the Image Service, assign meaning to roles. In the Identity Service, a role is simply a name.

The Identity Service assigns a tenant and a role to a user. You might assign the `compute-user` role to the `alice` user in the `acme` tenant:

```
$ keystone user-list
```

| id     | enabled | email             | name  |
|--------|---------|-------------------|-------|
| 892585 | True    | alice@example.com | alice |

```
$ keystone role-list
```

| id     | name         |
|--------|--------------|
| 9a764e | compute-user |

```
$ keystone tenant-list
```

| id     | name | enabled |
|--------|------|---------|
| 6b8fd2 | acme | True    |

```
$ keystone user-role-add --user=892585 --role=9a764e --tenant-id=6b8fd2
```

A user can have different roles in different tenants. For example, Alice might also have the `admin` role in the `Cyberdyne` tenant. A user can also have multiple roles in the same tenant.

The `/etc/[SERVICE_CODENAME]/policy.json` file controls the tasks that users can perform for a given service. For example, `/etc/nova/policy.json` specifies the access policy for the Compute service, `/etc/glance/policy.json` specifies the access policy for the Image Service, and `/etc/keystone/policy.json` specifies the access policy for the Identity Service.

The default `policy.json` files in the Compute, Identity, and Image Service recognize only the `admin` role: all operations that do not require the `admin` role are accessible by any user that has any role in a tenant.

If you wish to restrict users from performing operations in, say, the Compute service, you need to create a role in the Identity Service and then modify `/etc/nova/policy.json` so that this role is required for Compute operations.

For example, this line in `/etc/nova/policy.json` specifies that there are no restrictions on which users can create volumes: if the user has any role in a tenant, they can create volumes in that tenant.

```
"volume:create": [],
```

To restrict creation of volumes to users who had the `compute-user` role in a particular tenant, you would add `"role:compute-user"`, like so:

```
"volume:create": ["role:compute-user"],
```

To restrict all Compute service requests to require this role, the resulting file would look like:

```
{
  "admin_or_owner": [
    [
      "role:admin"
    ],
    [
      "project_id:%(project_id)s"
    ]
  ],
  "default": [
    [
      "rule:admin_or_owner"
    ]
  ]
}
```



```
    ],
    "compute:create": [
        "role:compute-user"
    ],
    "compute:create:attach_network": [
        "role:compute-user"
    ],
    "compute:create:attach_volume": [
        "role:compute-user"
    ],
    "compute:get_all": [
        "role:compute-user"
    ],
    "compute:unlock_override": [
        "rule:admin_api"
    ],
    "admin_api": [
        [
            "role:admin"
        ]
    ],
    "compute_extension:accounts": [
        [
            "rule:admin_api"
        ]
    ],
    "compute_extension:admin_actions": [
        [
            "rule:admin_api"
        ]
    ],
    "compute_extension:admin_actions:pause": [
        [
            "rule:admin_or_owner"
        ]
    ]
}
```

```
],
"compute_extension:admin_actions:unpause":[
  [
    "rule:admin_or_owner"
  ]
],
"compute_extension:admin_actions:suspend":[
  [
    "rule:admin_or_owner"
  ]
],
"compute_extension:admin_actions:resume":[
  [
    "rule:admin_or_owner"
  ]
],
"compute_extension:admin_actions:lock":[
  [
    "rule:admin_or_owner"
  ]
],
"compute_extension:admin_actions:unlock":[
  [
    "rule:admin_or_owner"
  ]
],
"compute_extension:admin_actions:resetNetwork":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:admin_actions:injectNetworkInfo":[
  [
    "rule:admin_api"
  ]
],
],
```

```
"compute_extension:admin_actions:createBackup":[
  [
    "rule:admin_or_owner"
  ]
],
"compute_extension:admin_actions:migrateLive":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:admin_actions:migrate":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:aggregates":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:certificates":[
  "role:compute-user"
],
"compute_extension:cloudpipe":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:console_output":[
  "role:compute-user"
],
"compute_extension:consoles":[
  "role:compute-user"
],
"compute_extension:createserverext":[
  "role:compute-user"
```

```
],
"compute_extension:deferred_delete":[
    "role:compute-user"
],
"compute_extension:disk_config":[
    "role:compute-user"
],
"compute_extension:evacuate":[
    [
        "rule:admin_api"
    ]
],
"compute_extension:extended_server_attributes":[
    [
        "rule:admin_api"
    ]
],
"compute_extension:extended_status":[
    "role:compute-user"
],
"compute_extension:flavorextradata":[
    "role:compute-user"
],
"compute_extension:flavorextraspecs":[
    "role:compute-user"
],
"compute_extension:flavormanage":[
    [
        "rule:admin_api"
    ]
],
"compute_extension:floating_ip_dns":[
    "role:compute-user"
],
"compute_extension:floating_ip_pools":[
    "role:compute-user"
```

```
],
"compute_extension:floating_ips":[
  "role:compute-user"
],
"compute_extension:hosts":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:keypairs":[
  "role:compute-user"
],
"compute_extension:multinic":[
  "role:compute-user"
],
"compute_extension:networks":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:quotas":[
  "role:compute-user"
],
"compute_extension:rescue":[
  "role:compute-user"
],
"compute_extension:security_groups":[
  "role:compute-user"
],
"compute_extension:server_action_list":[
  [
    "rule:admin_api"
  ]
],
"compute_extension:server_diagnostics":[
  [
```

```
        "rule:admin_api"
      ]
    ],
    "compute_extension:simple_tenant_usage:show": [
      [
        "rule:admin_or_owner"
      ]
    ],
    "compute_extension:simple_tenant_usage:list": [
      [
        "rule:admin_api"
      ]
    ],
    "compute_extension:users": [
      [
        "rule:admin_api"
      ]
    ],
    "compute_extension:virtual_interfaces": [
      "role:compute-user"
    ],
    "compute_extension:virtual_storage_arrays": [
      "role:compute-user"
    ],
    "compute_extension:volumes": [
      "role:compute-user"
    ],
    "compute_extension:volume_attachments:index": [
      "role:compute-user"
    ],
    "compute_extension:volume_attachments:show": [
      "role:compute-user"
    ],
    "compute_extension:volume_attachments:create": [
      "role:compute-user"
    ],
  ],
```

```
"compute_extension:volume_attachments:delete":[
    "role:compute-user"
],
"compute_extension:volumetypes":[
    "role:compute-user"
],
"volume:create":[
    "role:compute-user"
],
"volume:get_all":[
    "role:compute-user"
],
"volume:get_volume_metadata":[
    "role:compute-user"
],
"volume:get_snapshot":[
    "role:compute-user"
],
"volume:get_all_snapshots":[
    "role:compute-user"
],
"network:get_all_networks":[
    "role:compute-user"
],
"network:get_network":[
    "role:compute-user"
],
"network:delete_network":[
    "role:compute-user"
],
"network:disassociate_network":[
    "role:compute-user"
],
"network:get_vifs_by_instance":[
    "role:compute-user"
],
]
```

```
"network:allocate_for_instance":[
    "role:compute-user"
],
"network:deallocate_for_instance":[
    "role:compute-user"
],
"network:validate_networks":[
    "role:compute-user"
],
"network:get_instance_uuids_by_ip_filter":[
    "role:compute-user"
],
"network:get_floating_ip":[
    "role:compute-user"
],
"network:get_floating_ip_pools":[
    "role:compute-user"
],
"network:get_floating_ip_by_address":[
    "role:compute-user"
],
"network:get_floating_ips_by_project":[
    "role:compute-user"
],
"network:get_floating_ips_by_fixed_address":[
    "role:compute-user"
],
"network:allocate_floating_ip":[
    "role:compute-user"
],
"network:deallocate_floating_ip":[
    "role:compute-user"
],
"network:associate_floating_ip":[
    "role:compute-user"
],
]
```



```
"network:disassociate_floating_ip":[
  "role:compute-user"
],
"network:get_fixed_ip":[
  "role:compute-user"
],
"network:add_fixed_ip_to_instance":[
  "role:compute-user"
],
"network:remove_fixed_ip_from_instance":[
  "role:compute-user"
],
"network:add_network_to_project":[
  "role:compute-user"
],
"network:get_instance_nw_info":[
  "role:compute-user"
],
"network:get_dns_domains":[
  "role:compute-user"
],
"network:add_dns_entry":[
  "role:compute-user"
],
"network:modify_dns_entry":[
  "role:compute-user"
],
"network:delete_dns_entry":[
  "role:compute-user"
],
"network:get_dns_entries_by_address":[
  "role:compute-user"
],
"network:get_dns_entries_by_name":[
  "role:compute-user"
],
]
```

```
"network:create_private_dns_domain":[
    "role:compute-user"
],
"network:create_public_dns_domain":[
    "role:compute-user"
],
"network:delete_dns_domain":[
    "role:compute-user"
]
}
```

## Image CLI Commands

The **glance** client is the command-line interface (CLI) for the OpenStack Image Service API and its extensions. This chapter documents **glance** version 0.12.0.

For help on a specific **glance** command, enter:

```
$ glance help COMMAND
```

## glance usage

```
usage: glance [--version] [-d] [-v] [--get-schema] [-k]
             [--cert-file CERT_FILE] [--key-file KEY_FILE]
             [--os-cacert <ca-certificate-file>] [--ca-file OS_CACERT]
             [--timeout TIMEOUT] [--no-ssl-compression] [-f] [--dry-run]
             [--ssl] [-H ADDRESS] [-p PORT] [--os-username OS_USERNAME]
             [-I OS_USERNAME] [--os-password OS_PASSWORD] [-K OS_PASSWORD]
             [--os-tenant-id OS_TENANT_ID] [--os-tenant-name OS_TENANT_NAME]
             [-T OS_TENANT_NAME] [--os-auth-url OS_AUTH_URL] [-N OS_AUTH_URL]
             [--os-region-name OS_REGION_NAME] [-R OS_REGION_NAME]
             [--os-auth-token OS_AUTH_TOKEN] [-A OS_AUTH_TOKEN]
             [--os-image-url OS_IMAGE_URL] [-U OS_IMAGE_URL]
             [--os-image-api-version OS_IMAGE_API_VERSION]
             [--os-service-type OS_SERVICE_TYPE]
```

```
[--os-endpoint-type OS_ENDPOINT_TYPE] [-S OS_AUTH_STRATEGY]  
<subcommand> ...
```

## Subcommands

|                       |                                               |
|-----------------------|-----------------------------------------------|
| <b>add</b>            | <i>DEPRECATED!</i> Use image-create instead.  |
| <b>clear</b>          | <i>DEPRECATED!</i>                            |
| <b>delete</b>         | <i>DEPRECATED!</i> Use image-delete instead.  |
| <b>details</b>        | <i>DEPRECATED!</i> Use image-list instead.    |
| <b>image-create</b>   | Create a new image.                           |
| <b>image-delete</b>   | Delete specified image(s).                    |
| <b>image-download</b> | Download a specific image.                    |
| <b>image-list</b>     | List images you can access.                   |
| <b>image-members</b>  | <i>DEPRECATED!</i> Use member-list instead.   |
| <b>image-show</b>     | Describe a specific image.                    |
| <b>image-update</b>   | Update a specific image.                      |
| <b>index</b>          | <i>DEPRECATED!</i> Use image-list instead.    |
| <b>member-add</b>     | <i>DEPRECATED!</i> Use member-create instead. |
| <b>member-create</b>  | Share a specific image with a tenant.         |
| <b>member-delete</b>  | Remove a shared image from a tenant.          |
| <b>member-images</b>  | <i>DEPRECATED!</i> Use member-list instead.   |

|                        |                                                            |
|------------------------|------------------------------------------------------------|
| <b>member-list</b>     | Describe sharing permissions by image or tenant.           |
| <b>members-replace</b> | <i>DEPRECATED!</i>                                         |
| <b>show</b>            | <i>DEPRECATED!</i> Use image-show instead.                 |
| <b>update</b>          | <i>DEPRECATED!</i> Use image-update instead.               |
| <b>help</b>            | Display help about this program or one of its subcommands. |

## glance optional arguments

|                                               |                                                                                                                                                                                                       |
|-----------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-version</b>                               | show program's version number and exit                                                                                                                                                                |
| <b>-d, -debug</b>                             | Defaults to env[GLANCECLIENT_DEBUG]                                                                                                                                                                   |
| <b>-v, -verbose</b>                           | Print more verbose output                                                                                                                                                                             |
| <b>-get-schema</b>                            | Force retrieving the schema used to generate portions of the help text rather than using a cached copy. Ignored with api version 1                                                                    |
| <b>-k, -insecure</b>                          | Explicitly allow glanceclient to perform "insecure SSL" (https) requests. The server's certificate will not be verified against any certificate authorities. This option should be used with caution. |
| <b>-cert-file CERT_FILE</b>                   | Path of certificate file to use in SSL connection. This file can optionally be prepended with the private key.                                                                                        |
| <b>-key-file KEY_FILE</b>                     | Path of client key to use in SSL connection. This option is not necessary if your key is prepended to your cert file.                                                                                 |
| <b>-os-cacert &lt;ca-certificate-file&gt;</b> | Path of CA TLS certificate(s) used to verify the remote server's certificate. Without this option glance looks for the default system CA certificates.                                                |
| <b>-ca-file OS_CACERT</b>                     | <i>DEPRECATED!</i> Use -os-cacert.                                                                                                                                                                    |

|                                           |                                                                                      |
|-------------------------------------------|--------------------------------------------------------------------------------------|
| <b>-timeout TIMEOUT</b>                   | Number of seconds to wait for a response                                             |
| <b>-no-ssl-compression</b>                | Disable SSL compression when using https.                                            |
| <b>-f, -force</b>                         | Prevent select actions from requesting user confirmation.                            |
| <b>-dry-run</b>                           | <i>DEPRECATED!</i> Only used for deprecated legacy commands.                         |
| <b>-ssl</b>                               | <i>DEPRECATED!</i> Send a fully-formed endpoint using <b>-os- image-url</b> instead. |
| <b>-H ADDRESS, -host ADDRESS</b>          | <i>DEPRECATED!</i> Send a fully-formed endpoint using <b>-os- image-url</b> instead. |
| <b>-p PORT, -port PORT</b>                | <i>DEPRECATED!</i> Send a fully-formed endpoint using <b>-os- image-url</b> instead. |
| <b>-os-username OS_USERNAME</b>           | Defaults to env [OS_USERNAME]                                                        |
| <b>-I OS_USERNAME</b>                     | <i>DEPRECATED!</i> Use <b>-os-username</b> .                                         |
| <b>-os-password OS_PASSWORD</b>           | Defaults to env [OS_PASSWORD]                                                        |
| <b>-K OS_PASSWORD</b>                     | <i>DEPRECATED!</i> Use <b>-os-password</b> .                                         |
| <b>-os-tenant-id OS_TENANT_ID</b>         | Defaults to env [OS_TENANT_ID]                                                       |
| <b>-os-tenant-name<br/>OS_TENANT_NAME</b> | Defaults to env [OS_TENANT_NAME]                                                     |
| <b>-T OS_TENANT_NAME</b>                  | <i>DEPRECATED!</i> Use <b>-os-tenant-name</b> .                                      |
| <b>-os-auth-url OS_AUTH_URL</b>           | Defaults to env [OS_AUTH_URL]                                                        |
| <b>-N OS_AUTH_URL</b>                     | <i>DEPRECATED!</i> Use <b>-os-auth-url</b> .                                         |
| <b>-os-region-name<br/>OS_REGION_NAME</b> | Defaults to env [OS_REGION_NAME]                                                     |

|                                                                         |                                                         |
|-------------------------------------------------------------------------|---------------------------------------------------------|
| <b>-R OS_REGION_NAME</b>                                                | <i>DEPRECATED!</i> Use <code>--os-region-name</code> .  |
| <b>--os-auth-token<br/>OS_AUTH_TOKEN</b>                                | Defaults to <code>env[OS_AUTH_TOKEN]</code>             |
| <b>-A OS_AUTH_TOKEN, --<br/>auth_token OS_AUTH_TOKEN</b>                | <i>DEPRECATED!</i> Use <code>--os-auth-token</code> .   |
| <b>--os-image-url OS_IMAGE_URL</b>                                      | Defaults to <code>env[OS_IMAGE_URL]</code>              |
| <b>-U OS_IMAGE_URL, --url<br/>OS_IMAGE_URL</b>                          | <i>DEPRECATED!</i> Use <code>--os-image-url</code> .    |
| <b>--os-image-api-version<br/>OS_IMAGE_API_VERSION</b>                  | Defaults to <code>env[OS_IMAGE_API_VERSION]</code> or 1 |
| <b>--os-service-type<br/>OS_SERVICE_TYPE</b>                            | Defaults to <code>env[OS_SERVICE_TYPE]</code>           |
| <b>--os-endpoint-type<br/>OS_ENDPOINT_TYPE</b>                          | Defaults to <code>env[OS_ENDPOINT_TYPE]</code>          |
| <b>-S OS_AUTH_STRATEGY,<br/>--os_auth_strategy<br/>OS_AUTH_STRATEGY</b> | <i>DEPRECATED!</i> This option is completely ignored.   |

## glance image-create command

```
usage: glance image-create [--id <IMAGE_ID>] [--name <NAME>] [--store <STORE>]
                             [--disk-format <DISK_FORMAT>]
                             [--container-format <CONTAINER_FORMAT>]
                             [--owner <TENANT_ID>] [--size <SIZE>]
                             [--min-disk <DISK_GB>] [--min-ram <DISK_RAM>]
                             [--location <IMAGE_URL>] [--file <FILE>]
```

```
[--checksum <CHECKSUM>] [--copy-from <IMAGE_URL>]
[--is-public {True,False}]
[--is-protected {True,False}]
[--property <key=value>] [--human-readable]
[--progress]
```

Create a new image.

### Optional arguments

|                                                       |                                                                                                                                                                          |
|-------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-id &lt;IMAGE_ID&gt; ID</b>                        | of image to reserve.                                                                                                                                                     |
| <b>-name &lt;NAME&gt;</b>                             | Name of image.                                                                                                                                                           |
| <b>-store &lt;STORE&gt;</b>                           | Store to upload image to.                                                                                                                                                |
| <b>-disk-format &lt;DISK_FORMAT&gt;</b>               | Disk format of image. Acceptable formats: ami, ari, aki, vhd, vmdk, raw, qcow2, vdi, and iso.                                                                            |
| <b>-container-format<br/>&lt;CONTAINER_FORMAT&gt;</b> | Container format of image. Acceptable formats: ami, ari, aki, bare, and ovf.                                                                                             |
| <b>-owner &lt;TENANT_ID&gt;</b>                       | Tenant who should own image.                                                                                                                                             |
| <b>-size &lt;SIZE&gt;</b>                             | Size of image data (in bytes). Only used with '- location' and '-copy_from'.                                                                                             |
| <b>-min-disk &lt;DISK_GB&gt;</b>                      | Minimum size of disk needed to boot image (in gigabytes).                                                                                                                |
| <b>-min-ram &lt;DISK_RAM&gt;</b>                      | Minimum amount of ram needed to boot image (in megabytes).                                                                                                               |
| <b>-location &lt;IMAGE_URL&gt;</b>                    | URL where the data for this image already resides. For example, if the image data is stored in swift, you could specify 'swift://account:key@example.com/container/obj'. |

|                                     |                                                                                                                                                         |
|-------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-file &lt;FILE&gt;</b>           | Local file that contains disk image to be uploaded during creation. Alternatively, images can be passed to the client via stdin.                        |
| <b>-checksum &lt;CHECKSUM&gt;</b>   | Hash of image data used Glance can use for verification. Provide a md5 checksum here.                                                                   |
| <b>-copy-from &lt;IMAGE_URL&gt;</b> | Similar to '-location' in usage, but this indicates that the Glance server should immediately copy the data and store it in its configured image store. |
| <b>-is-public {True,False}</b>      | Make image accessible to the public.                                                                                                                    |
| <b>-is-protected {True,False}</b>   | Prevent image from being deleted.                                                                                                                       |
| <b>-property &lt;key=value&gt;</b>  | Arbitrary property to associate with image. May be used multiple times.                                                                                 |
| <b>-human-readable</b>              | Print image size in a human-friendly format.                                                                                                            |
| <b>-progress</b>                    | Show upload progress bar.                                                                                                                               |

## glance image-delete command

```
usage: glance image-delete <IMAGE> [<IMAGE> ...]
```

Delete specified image(s).

### Positional arguments

**<IMAGE>** Name or ID of image(s) to delete.

## glance image-list command

```
usage: glance image-list [--name <NAME>] [--status <STATUS>]
```



```

[--container-format <CONTAINER_FORMAT>]
[--disk-format <DISK_FORMAT>] [--size-min <SIZE>]
[--size-max <SIZE>] [--property-filter <KEY=VALUE>]
[--page-size <SIZE>] [--human-readable]
created_at,updated_at}}]
[--sort-key {name,status,container_format,disk_format,size,id,
[--sort-dir {asc,desc}}] [--is-public {True,False}}]
[--owner <TENANT_ID>] [--all-tenants]

```

List images you can access.

### Optional arguments

|                                                    |                                                                                                                            |
|----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| <b>--name &lt;NAME&gt;</b>                         | Filter images to those that have this name.                                                                                |
| <b>--status &lt;STATUS&gt;</b>                     | Filter images to those that have this status.                                                                              |
| <b>--container-format &lt;CONTAINER_FORMAT&gt;</b> | Filter images to those that have this container format. Acceptable formats: ami, ari, aki, bare, and ovf.                  |
| <b>--disk-format &lt;DISK_FORMAT&gt;</b>           | Filter images to those that have this disk format. Acceptable formats: ami, ari, aki, vhd, vmdk, raw, qcow2, vdi, and iso. |
| <b>--size-min &lt;SIZE&gt;</b>                     | Filter images to those with a size greater than this.                                                                      |
| <b>--size-max &lt;SIZE&gt;</b>                     | Filter images to those with a size less than this.                                                                         |
| <b>--property-filter &lt;KEY=VALUE&gt;</b>         | Filter images by a user-defined image property.                                                                            |
| <b>--page-size &lt;SIZE&gt;</b>                    | Number of images to request in each paginated request.                                                                     |
| <b>--human-readable</b>                            | Print image size in a human-friendly format.                                                                               |

|                                                                                                     |                                                                                                                                                                                                                                                                                                                                                                 |
|-----------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-sort-key</b><br><b>{name,status,container_format,disk_format,size,id,created_at,updated_at}</b> | Sort image list by specified field.                                                                                                                                                                                                                                                                                                                             |
| <b>-sort-dir {asc,desc}</b>                                                                         | Sort image list in specified direction.                                                                                                                                                                                                                                                                                                                         |
| <b>-is-public {True,False}</b>                                                                      | Allows the user to select a listing of public or non public images.                                                                                                                                                                                                                                                                                             |
| <b>-owner &lt;TENANT_ID&gt;</b>                                                                     | Display only images owned by this tenant id. Filtering occurs on the client side so may be inefficient. This option is mainly intended for admin use. Use an empty string ("") to list images with no owner. Note: This option overrides the <b>-is-public</b> argument if present. Note: the v2 API supports more efficient server-side owner based filtering. |
| <b>-all-tenants</b>                                                                                 | Allows the admin user to list all images irrespective of the image's owner or <b>is_public</b> value.                                                                                                                                                                                                                                                           |

## glance image-show command

```
usage: glance image-show [--human-readable] <IMAGE>
```

Describe a specific image.

### Positional arguments

**<IMAGE>** Name or ID of image to describe.

### Optional arguments

**-human-readable** Print image size in a human-friendly format.

## glance image-update command

```
usage: glance image-update [--name <NAME>] [--disk-format <DISK_FORMAT>]
```

```
[--container-format <CONTAINER_FORMAT>]
[--owner <TENANT_ID>] [--size <SIZE>]
[--min-disk <DISK_GB>] [--min-ram <DISK_RAM>]
[--location <IMAGE_URL>] [--file <FILE>]
[--checksum <CHECKSUM>] [--copy-from <IMAGE_URL>]
[--is-public {True,False}]
[--is-protected {True,False}]
[--property <key=value>] [--purge-props]
[--human-readable] [--progress]
<IMAGE>
```

Update a specific image.

### Positional arguments

**<IMAGE>** Name or ID of image to modify.

### Optional arguments

|                                                    |                                                                                               |
|----------------------------------------------------|-----------------------------------------------------------------------------------------------|
| <b>--name &lt;NAME&gt;</b>                         | Name of image.                                                                                |
| <b>--disk-format &lt;DISK_FORMAT&gt;</b>           | Disk format of image. Acceptable formats: ami, ari, aki, vhd, vmdk, raw, qcow2, vdi, and iso. |
| <b>--container-format &lt;CONTAINER_FORMAT&gt;</b> | Container format of image. Acceptable formats: ami, ari, aki, bare, and ovf.                  |
| <b>--owner &lt;TENANT_ID&gt;</b>                   | Tenant who should own image.                                                                  |
| <b>--size &lt;SIZE&gt;</b>                         | Size of image data (in bytes).                                                                |
| <b>--min-disk &lt;DISK_GB&gt;</b>                  | Minimum size of disk needed to boot image (in gigabytes).                                     |

|                                     |                                                                                                                                                                          |
|-------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-min-ram &lt;DISK_RAM&gt;</b>    | Minimum amount of ram needed to boot image (in megabytes).                                                                                                               |
| <b>-location &lt;IMAGE_URL&gt;</b>  | URL where the data for this image already resides. For example, if the image data is stored in swift, you could specify 'swift://account:key@example.com/container/obj'. |
| <b>-file &lt;FILE&gt;</b>           | Local file that contains disk image to be uploaded during update. Alternatively, images can be passed to the client via stdin.                                           |
| <b>-checksum &lt;CHECKSUM&gt;</b>   | Hash of image data used Glance can use for verification.                                                                                                                 |
| <b>-copy-from &lt;IMAGE_URL&gt;</b> | Similar to '-location' in usage, but this indicates that the Glance server should immediately copy the data and store it in its configured image store.                  |
| <b>-is-public {True,False}</b>      | Make image accessible to the public.                                                                                                                                     |
| <b>-is-protected {True,False}</b>   | Prevent image from being deleted.                                                                                                                                        |
| <b>-property &lt;key=value&gt;</b>  | Arbitrary property to associate with image. May be used multiple times.                                                                                                  |
| <b>-purge-props</b>                 | If this flag is present, delete all image properties not explicitly set in the update request. Otherwise, those properties not referenced are preserved.                 |
| <b>-human-readable</b>              | Print image size in a human-friendly format.                                                                                                                             |
| <b>-progress</b>                    | Show upload progress bar.                                                                                                                                                |

## glance member-create command

```
usage: glance member-create [--can-share] <IMAGE> <TENANT_ID>
```

Share a specific image with a tenant.

### Positional arguments

<IMAGE>      Image to add member to.

<TENANT\_ID>    Tenant to add as member

### Optional arguments

–can-share      Allow the specified tenant to share this image.

## glance member-delete command

```
usage: glance member-delete <IMAGE> <TENANT_ID>
```

Remove a shared image from a tenant.

### Positional arguments

<IMAGE>      Image from which to remove member

<TENANT\_ID>    Tenant to remove as member

## glance member-list command

```
usage: glance member-list [--image-id <IMAGE_ID>] [--tenant-id <TENANT_ID>]
```

Describe sharing permissions by image or tenant.

### Optional arguments

–image-id <IMAGE\_ID>      Filter results by an image ID.

–tenant-id <TENANT\_ID>      Filter results by a tenant ID.

## Image List Images

To get a list of images and to then get further details about a single image, use **glance image-list** and **glance image-show**.

```
$ glance image-list
```

| ID                                   |        |          |        | Name                            | Disk Format |     |
|--------------------------------------|--------|----------|--------|---------------------------------|-------------|-----|
| Container                            | Format | Size     | Status |                                 |             |     |
| 397e713c-b95b-4186-ad46-6126863ea0a9 | ami    | 25165824 | active | cirros-0.3.2-x86_64-uec         | ami         | ami |
| df430cc2-3406-4061-b635-a51c16e488ac | aki    | 4955792  | active | cirros-0.3.2-x86_64-uec-kernel  | aki         | aki |
| 3cf852bd-2332-48f4-9ae4-7d926d50945e | ari    | 3714968  | active | cirros-0.3.2-x86_64-uec-ramdisk | ari         | ari |
| 7e5142af-1253-4634-bcc6-89482c5f2e8a | ami    | 14221312 | active | myCirrosImage                   | ami         | ami |

```
$ glance image-show myCirrosImage
```

| Property                              | Value                                |
|---------------------------------------|--------------------------------------|
| Property 'base_image_ref'             | 397e713c-b95b-4186-ad46-6126863ea0a9 |
| Property 'image_location'             | snapshot                             |
| Property 'image_state'                | available                            |
| Property 'image_type'                 | snapshot                             |
| Property 'instance_type_ephemeral_gb' | 0                                    |
| Property 'instance_type_flavorid'     | 2                                    |
| Property 'instance_type_id'           | 5                                    |
| Property 'instance_type_memory_mb'    | 2048                                 |
| Property 'instance_type_name'         | m1.small                             |
| Property 'instance_type_root_gb'      | 20                                   |
| Property 'instance_type_rxtx_factor'  | 1                                    |
| Property 'instance_type_swap'         | 0                                    |
| Property 'instance_type_vcpu_weight'  | None                                 |
| Property 'instance_type_vcpus'        | 1                                    |
| Property 'instance_uuid'              | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |
| Property 'kernel_id'                  | df430cc2-3406-4061-b635-a51c16e488ac |
| Property 'owner_id'                   | 66265572db174a7aa66eba661f58eb9e     |
| Property 'ramdisk_id'                 | 3cf852bd-2332-48f4-9ae4-7d926d50945e |

```

| Property 'user_id'      | 376744b5910b4b4da7d0e6cb483b06a8 |
| checksum               | 8e4838effa1969ad591655d6485c7ba8 |
| container_format       | ami |
| created_at             | 2013-07-22T19:45:58 |
| deleted                | False |
| disk_format            | ami |
| id                     | 7e5142af-1253-4634-bcc6-89482c5f2e8a |
| is_public              | False |
| min_disk               | 0 |
| min_ram                | 0 |
| name                   | myCirrosImage |
| owner                  | 66265572db174a7aa66eba661f58eb9e |
| protected              | False |
| size                   | 14221312 |
| status                 | active |
| updated_at             | 2013-07-22T19:46:42 |
+-----+-----+

```

When viewing a list of images, you can also use **grep** to filter the list, as follows:

```

$ glance image-list | grep 'cirros'
| 397e713c-b95b-4186-ad46-6126863ea0a9 | cirros-0.3.2-x86_64-uec | ami | ami
| | 25165824 | active |
| df430cc2-3406-4061-b635-a51c16e488ac | cirros-0.3.2-x86_64-uec-kernel | aki | aki
| | 4955792 | active |
| 3cf852bd-2332-48f4-9ae4-7d926d50945e | cirros-0.3.2-x86_64-uec-ramdisk | ari | ari
| | 3714968 | active |

```



## Note

To store location metadata for images, which enables direct file access for a client, update the `/etc/glance/glance.conf` file with the following statements:

- `show_multiple_locations = True`
- `filesystem_store_metadata_file = filePath`, where *filePath* points to a JSON file that defines the mount point for OpenStack images on your system and a unique ID. For example:

```

[{"id": "2d9bb53f-70ea-4066-a68b-67960eaae673",
 "mountpoint": "/var/lib/glance/images/"
}]

```

After you restart the Image Service, you can use the following syntax to view the image's location information:

```
$ glance --os-image-api-version=2 image-show imageID
```

For example, using the image ID shown above, you would issue the command as follows:

```
$ glance --os-image-api-version=2 image-show 2d9bb53f-70ea-4066-a68b-67960eaae673
```

## Image Adding Images

To create an image, use **glance image-create**:

```
$ glance image-create imageName
```

To update an image by name or ID, use **glance image-update**:

```
$ glance image-update imageName
```

The following table lists the optional arguments that you can use with the **create** and **update** commands to modify image properties. For more information, refer to Image Service chapter in the [OpenStack Command-Line Interface Reference](#).

|                                                  |                                                                                                          |
|--------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <code>--name NAME</code>                         | The name of the image.                                                                                   |
| <code>--disk-format DISK_FORMAT</code>           | The disk format of the image. Acceptable formats are ami, ari, aki, vhd, vmdk, raw, qcow2, vdi, and iso. |
| <code>--container-format CONTAINER_FORMAT</code> | The container format of the image. Acceptable formats are ami, ari, aki, bare, and ovf.                  |
| <code>--owner TENANT_ID</code>                   | The tenant who should own the image.                                                                     |
| <code>--size SIZE</code>                         | The size of image data, in bytes.                                                                        |
| <code>--min-disk DISK_GB</code>                  | The minimum size of the disk needed to boot the image, in gigabytes.                                     |
| <code>--min-ram DISK_RAM</code>                  | The minimum amount of RAM needed to boot the image, in megabytes.                                        |



|                                          |                                                                                                                                                                                  |
|------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>--location IMAGE_URL</code>        | The URL where the data for this image resides. For example, if the image data is stored in swift, you could specify <code>swift://account:key@example.com/container/obj</code> . |
| <code>--file FILE</code>                 | Local file that contains the disk image to be uploaded during the update. Alternatively, you can pass images to the client through stdin.                                        |
| <code>--checksum CHECKSUM</code>         | Hash of image data to use for verification.                                                                                                                                      |
| <code>--copy-from IMAGE_URL</code>       | Similar to <code>--location</code> in usage, but indicates that the image server should immediately copy the data and store it in its configured image store.                    |
| <code>--is-public [True False]</code>    | Makes an image accessible for all the tenants.                                                                                                                                   |
| <code>--is-protected [True False]</code> | Prevents an image from being deleted.                                                                                                                                            |
| <code>--property KEY=VALUE</code>        | Arbitrary property to associate with image. This option can be used multiple times.                                                                                              |
| <code>--purge-props</code>               | Deletes all image properties that are not explicitly set in the update request. Otherwise, those properties not referenced are preserved.                                        |
| <code>--human-readable</code>            | Prints the image size in a human-friendly format.                                                                                                                                |

The following example shows the command that you would use to upload a CentOS 6.3 image in qcow2 format and configure it for public access:

```
$ glance image-create --name centos63-image --disk-format=qcow2 \
  --container-format=bare --is-public=True --file=./centos63.qcow2
```

The following example shows how to update an existing image with a properties that describe the disk bus, the CD-ROM bus, and the VIF model:

```
$ glance image-update \
  --property hw_disk_bus=scsi \
  --property hw_cdrom_bus=ide \
  --property hw_vif_model=e1000 \
  f16-x86_64-openstack-sda
```

Currently the libvirt virtualization tool determines the disk, CD-ROM, and VIF device models based on the configured hypervisor type (`libvirt_type` in `/etc/nova/nova.conf`). For the sake of optimal performance, libvirt defaults to using virtio for both disk and VIF (NIC) models. The disadvantage of this

approach is that it is not possible to run operating systems that lack virtio drivers, for example, BSD, Solaris, and older versions of Linux and Windows.

If you specify a disk or CD-ROM bus model that is not supported, see [Table 3.1, “Disk and CD-ROM bus model values” \[140\]](#). If you specify a VIF model that is not supported, the instance fails to launch. See [Table 3.2, “VIF model values” \[140\]](#).

The valid model values depend on the `libvirt_type` setting, as shown in the following tables.

**Table 3.1. Disk and CD-ROM bus model values**

| libvirt_type setting | Supported model values                                                                                 |
|----------------------|--------------------------------------------------------------------------------------------------------|
| qemu or kvm          | <ul style="list-style-type: none"><li>• virtio</li><li>• scsi</li><li>• ide</li><li>• virtio</li></ul> |
| xen                  | <ul style="list-style-type: none"><li>• xen</li><li>• ide</li></ul>                                    |

**Table 3.2. VIF model values**

| libvirt_type setting | Supported model values                                                                                                        |
|----------------------|-------------------------------------------------------------------------------------------------------------------------------|
| qemu or kvm          | <ul style="list-style-type: none"><li>• virtio</li><li>• ne2k_pci</li><li>• pcnet</li><li>• rtl8139</li><li>• e1000</li></ul> |
| xen                  | <ul style="list-style-type: none"><li>• netfront</li><li>• ne2k_pci</li></ul>                                                 |

| libvirt_type setting | Supported model values                                                                                          |
|----------------------|-----------------------------------------------------------------------------------------------------------------|
|                      | <ul style="list-style-type: none"><li>• pcnet</li><li>• rtl8139</li><li>• e1000</li></ul>                       |
| vmware               | <ul style="list-style-type: none"><li>• VirtualE1000</li><li>• VirtualPCNet32</li><li>• VirtualVmxnet</li></ul> |

## Image Manage Images

You can use the **nova** client to take a snapshot of a running instance to create an image.

To minimize the potential for data loss and ensure that you create an accurate image, you should shut down the instance before you take a snapshot.

You cannot create a snapshot from an instance that has an attached volume. Detach the volume, create the image, and remount the volume.

1. Write any buffered data to disk.

For more information, see [Taking Snapshots](#) in the *OpenStack Operations Guide*.

2. List instances to get the server name:

```
$ nova list
+-----+-----+-----+-----+-----+-----+
| ID | Name | Status | Task State | Power State | Networks |
+-----+-----+-----+-----+-----+-----+
| 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 | myCirrosServer | ACTIVE | None | Running | private=10.0.0.3 |
+-----+-----+-----+-----+-----+-----+
```

In this example, the server is named myCirrosServer.

3. Use this server to create a snapshot:

```
$ nova image-create myCirrosServer myCirrosImage
```

The command creates a qemu snapshot and automatically uploads the image to your repository. Only the tenant that creates the image has access to it.

#### 4. Get details for your image to check its status:

```
$ nova image-show myCirrosImage
```

| Property                            | Value                                |
|-------------------------------------|--------------------------------------|
| metadata owner_id                   | 66265572db174a7aa66eba661f58eb9e     |
| minDisk                             | 0                                    |
| metadata instance_type_name         | m1.small                             |
| metadata instance_type_id           | 5                                    |
| metadata instance_type_memory_mb    | 2048                                 |
| id                                  | 7e5142af-1253-4634-bcc6-89482c5f2e8a |
| metadata instance_type_root_gb      | 20                                   |
| metadata instance_type_rxtx_factor  | 1                                    |
| metadata ramdisk_id                 | 3cf852bd-2332-48f4-9ae4-7d926d50945e |
| metadata image_state                | available                            |
| metadata image_location             | snapshot                             |
| minRam                              | 0                                    |
| metadata instance_type_vcpus        | 1                                    |
| status                              | ACTIVE                               |
| updated                             | 2013-07-22T19:46:42Z                 |
| metadata instance_type_swap         | 0                                    |
| metadata instance_type_vcpu_weight  | None                                 |
| metadata base_image_ref             | 397e713c-b95b-4186-ad46-6126863ea0a9 |
| progress                            | 100                                  |
| metadata instance_type_flavorid     | 2                                    |
| OS-EXT-IMG-SIZE:size                | 14221312                             |
| metadata image_type                 | snapshot                             |
| metadata user_id                    | 376744b5910b4b4da7d8e6cb483b06a8     |
| name                                | myCirrosImage                        |
| created                             | 2013-07-22T19:45:58Z                 |
| metadata instance_uuid              | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |
| server                              | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |
| metadata kernel_id                  | df430cc2-3406-4061-b635-a51c16e488ac |
| metadata instance_type_ephemeral_gb | 0                                    |

The image status changes from SAVING to ACTIVE. Only the tenant who creates the image has access to it.

To launch an instance from your image, include the image ID and flavor ID, as in the following example:

```
$ nova boot newServer --image 7e5142af-1253-4634-bcc6-89482c5f2e8a \
--flavor 3
```

| Property | Value |
|----------|-------|
|----------|-------|

|                                     |                                      |
|-------------------------------------|--------------------------------------|
| OS-EXT-STS:task_state               | scheduling                           |
| image                               | myCirrosImage                        |
| OS-EXT-STS:vm_state                 | building                             |
| OS-EXT-SRV-ATTR:instance_name       | instance-00000007                    |
| flavor                              | m1.medium                            |
| id                                  | d7efd3e4-d375-46d1-9d57-372b6e4bdb7f |
| security_groups                     | [[{'name': 'u'default'}]]            |
| user_id                             | 376744b5910b4b4da7d8e6cb483b06a8     |
| OS-DCF:diskConfig                   | MANUAL                               |
| accessIPv4                          |                                      |
| accessIPv6                          |                                      |
| progress                            | 0                                    |
| OS-EXT-STS:power_state              | 0                                    |
| OS-EXT-AZ:availability_zone         | nova                                 |
| config_drive                        |                                      |
| status                              | BUILD                                |
| updated                             | 2013-07-22T19:58:33Z                 |
| hostId                              |                                      |
| OS-EXT-SRV-ATTR:host                | None                                 |
| key_name                            | None                                 |
| OS-EXT-SRV-ATTR:hypervisor_hostname | None                                 |
| name                                | newServer                            |
| adminPass                           | j1s88nN46RGP                         |
| tenant_id                           | 66265572db174a7aa66eba661f58eb9e     |
| created                             | 2013-07-22T19:58:33Z                 |
| metadata                            | {}                                   |

## Message Queue Configuration

OpenStack projects use AMQP, an open standard for messaging middleware. OpenStack services that run on multiple servers to talk to each other. OpenStack Oslo RPC supports three implementations of AMQP: RabbitMQ, Qpid, and ZeroMQ.

### Configure RabbitMQ

OpenStack Oslo RPC uses RabbitMQ by default. Use these options to configure the RabbitMQ message system. The `rpc_backend` option is not required as long as RabbitMQ is the default messaging system. However, if it is included the configuration, you must set it to `nova.openstack.common.rpc.impl_kombu`.

```
rpc_backend=nova.openstack.common.rpc.impl_kombu
```

You can use these additional options to configure the RabbitMQ messaging system. You can configure messaging communication for different installation scenarios, tune retries for RabbitMQ, and define

the size of the RPC thread pool. To monitor notifications through RabbitMQ, you must set the `notification_driver` option to `nova.notifier.rabbit_notifier` in the `nova.conf` file. The default for sending usage data is sixty seconds plus a random number of seconds from zero to sixty.

**Table 3.3. Description of configuration options for rabbitmq**

| Configuration option = Default value                    | Description                                                                                                             |
|---------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
| [DEFAULT]                                               |                                                                                                                         |
| <code>rabbit_ha_queues = False</code>                   | (BoolOpt) Use HA queues in RabbitMQ (x-ha-policy: all). If you change this option, you must wipe the RabbitMQ database. |
| <code>rabbit_host = localhost</code>                    | (StrOpt) The RabbitMQ broker address where a single node is used.                                                       |
| <code>rabbit_hosts = \$rabbit_host:\$rabbit_port</code> | (ListOpt) RabbitMQ HA cluster host:port pairs.                                                                          |
| <code>rabbit_login_method = AMQPLAIN</code>             | (StrOpt) the RabbitMQ login method                                                                                      |
| <code>rabbit_max_retries = 0</code>                     | (IntOpt) Maximum number of RabbitMQ connection retries. Default is 0 (infinite retry count).                            |
| <code>rabbit_password = guest</code>                    | (StrOpt) The RabbitMQ password.                                                                                         |
| <code>rabbit_port = 5672</code>                         | (IntOpt) The RabbitMQ broker port where a single node is used.                                                          |
| <code>rabbit_retry_backoff = 2</code>                   | (IntOpt) How long to backoff for between retries when connecting to RabbitMQ.                                           |
| <code>rabbit_retry_interval = 1</code>                  | (IntOpt) How frequently to retry connecting with RabbitMQ.                                                              |
| <code>rabbit_use_ssl = False</code>                     | (BoolOpt) Connect over SSL for RabbitMQ.                                                                                |
| <code>rabbit_userid = guest</code>                      | (StrOpt) The RabbitMQ userid.                                                                                           |
| <code>rabbit_virtual_host = /</code>                    | (StrOpt) The RabbitMQ virtual host.                                                                                     |

**Table 3.4. Description of configuration options for kombu**

| Configuration option = Default value     | Description                                                                                          |
|------------------------------------------|------------------------------------------------------------------------------------------------------|
| [DEFAULT]                                |                                                                                                      |
| <code>kombu_reconnect_delay = 1.0</code> | (FloatOpt) How long to wait before reconnecting in response to an AMQP consumer cancel notification. |
| <code>kombu_ssl_ca_certs =</code>        | (StrOpt) SSL certification authority file (valid only if SSL enabled).                               |

| Configuration option = Default value | Description                                                                                                                                      |
|--------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| kombu_ssl_certfile =                 | (StrOpt) SSL cert file (valid only if SSL enabled).                                                                                              |
| kombu_ssl_keyfile =                  | (StrOpt) SSL key file (valid only if SSL enabled).                                                                                               |
| kombu_ssl_version =                  | (StrOpt) SSL version to use (valid only if SSL enabled). valid values are TLSv1, SSLv23 and SSLv3. SSLv2 may be available on some distributions. |

## Configure Qpid

Use these options to configure the Qpid messaging system for OpenStack Oslo RPC. Qpid is not the default messaging system, so you must enable it by setting the `rpc_backend` option in the `nova.conf` file.

```
rpc_backend=nova.openstack.common.rpc.impl_qpid
```

This critical option points the compute nodes to the Qpid broker (server). Set `qpid_hostname` to the host name where the broker runs in the `nova.conf` file.



### Note

The `--qpid_hostname` option accepts a host name or IP address value.

```
qpid_hostname=hostname.example.com
```

If the Qpid broker listens on a port other than the AMQP default of 5672, you must set the `qpid_port` option to that value:

```
qpid_port=12345
```

If you configure the Qpid broker to require authentication, you must add a user name and password to the configuration:

```
qpid_username=username  
qpid_password=password
```

By default, TCP is used as the transport. To enable SSL, set the `qpid_protocol` option:

```
qpid_protocol=ssl
```

This table lists additional options that you use to configure the Qpid messaging driver for OpenStack Oslo RPC. These options are used infrequently.

**Table 3.5. Description of configuration options for qpid**

| Configuration option = Default value                  | Description                                                                                                                                                                                                                                                                                                          |
|-------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [DEFAULT]                                             |                                                                                                                                                                                                                                                                                                                      |
| <code>qpid_heartbeat = 60</code>                      | (IntOpt) Seconds between connection keepalive heartbeats.                                                                                                                                                                                                                                                            |
| <code>qpid_hostname = localhost</code>                | (StrOpt) Qpid broker hostname.                                                                                                                                                                                                                                                                                       |
| <code>qpid_hosts = \$qpid_hostname:\$qpid_port</code> | (ListOpt) Qpid HA cluster host:port pairs.                                                                                                                                                                                                                                                                           |
| <code>qpid_password =</code>                          | (StrOpt) Password for Qpid connection.                                                                                                                                                                                                                                                                               |
| <code>qpid_port = 5672</code>                         | (IntOpt) Qpid broker port.                                                                                                                                                                                                                                                                                           |
| <code>qpid_protocol = tcp</code>                      | (StrOpt) Transport to use, either 'tcp' or 'ssl'.                                                                                                                                                                                                                                                                    |
| <code>qpid_sasl_mechanisms =</code>                   | (StrOpt) Space separated list of SASL mechanisms to use for auth.                                                                                                                                                                                                                                                    |
| <code>qpid_tcp_nodelay = True</code>                  | (BoolOpt) Whether to disable the Nagle algorithm.                                                                                                                                                                                                                                                                    |
| <code>qpid_topology_version = 1</code>                | (IntOpt) The qpid topology version to use. Version 1 is what was originally used by <code>impl_qpid</code> . Version 2 includes some backwards-incompatible changes that allow broker federation to work. Users should update to version 2 when they are able to take everything down, as it requires a clean break. |
| <code>qpid_username =</code>                          | (StrOpt) Username for Qpid connection.                                                                                                                                                                                                                                                                               |

## Configure ZeroMQ

Use these options to configure the ZeroMQ messaging system for OpenStack Oslo RPC. ZeroMQ is not the default messaging system, so you must enable it by setting the `rpc_backend` option in the `nova.conf` file.



**Table 3.6. Description of configuration options for zeromq**

| Configuration option = Default value                                        | Description                                                                                                                                      |
|-----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------|
| [DEFAULT]                                                                   |                                                                                                                                                  |
| rpc_zmq_bind_address = *                                                    | (StrOpt) ZeroMQ bind address. Should be a wildcard (*), an ethernet interface, or IP. The "host" option should point or resolve to this address. |
| rpc_zmq_contexts = 1                                                        | (IntOpt) Number of ZeroMQ contexts, defaults to 1.                                                                                               |
| rpc_zmq_host = oslo                                                         | (StrOpt) Name of this node. Must be a valid hostname, FQDN, or IP address. Must match "host" option, if running Nova.                            |
| rpc_zmq_ipc_dir = /var/run/openstack                                        | (StrOpt) Directory for holding IPC sockets.                                                                                                      |
| rpc_zmq_matchmaker = oslo.messaging._drivers.matchmaker.MatchMakerLocalhost | (StrOpt) MatchMaker driver.                                                                                                                      |
| rpc_zmq_port = 9501                                                         | (IntOpt) ZeroMQ receiver listening port.                                                                                                         |
| rpc_zmq_topic_backlog = None                                                | (IntOpt) Maximum number of ingress messages to locally buffer per topic. Default is unlimited.                                                   |

## Configure messaging

Use these options to configure the RabbitMQ and Qpid messaging drivers.

**Table 3.7. Description of configuration options for rpc**

| Configuration option = Default value | Description                                                                                                                               |
|--------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------|
| [DEFAULT]                            |                                                                                                                                           |
| amqp_auto_delete = False             | (BoolOpt) Auto-delete queues in amqp.                                                                                                     |
| amqp_durable_queues = False          | (BoolOpt) Use durable queues in amqp.                                                                                                     |
| control_exchange = openstack         | (StrOpt) The default exchange under which topics are scoped. May be overridden by an exchange name specified in the transport_url option. |
| matchmaker_heartbeat_freq = 300      | (IntOpt) Heartbeat frequency.                                                                                                             |
| matchmaker_heartbeat_ttl = 600       | (IntOpt) Heartbeat time-to-live.                                                                                                          |

| Configuration option = Default value      | Description                                                                                                                |
|-------------------------------------------|----------------------------------------------------------------------------------------------------------------------------|
| rpc_backend = rabbit                      | (StrOpt) The messaging driver to use, defaults to rabbit. Other drivers include qpid and zmq.                              |
| rpc_cast_timeout = 30                     | (IntOpt) Seconds to wait before a cast expires (TTL). Only supported by impl_zmq.                                          |
| rpc_conn_pool_size = 30                   | (IntOpt) Size of RPC connection pool.                                                                                      |
| rpc_response_timeout = 60                 | (IntOpt) Seconds to wait for a response from a call.                                                                       |
| rpc_thread_pool_size = 64                 | (IntOpt) Size of RPC greenthread pool.                                                                                     |
| [cells]                                   |                                                                                                                            |
| rpc_driver_queue_base = cells.intercell   | (StrOpt) Base queue name to use when communicating between cells. Various topics by message type will be appended to this. |
| [matchmaker_ring]                         |                                                                                                                            |
| ringfile = /etc/oslo/matchmaker_ring.json | (StrOpt) Matchmaker ring file (JSON).                                                                                      |
| [upgrade_levels]                          |                                                                                                                            |
| baseapi = None                            | (StrOpt) Set a version cap for messages sent to the base api in any service                                                |

## 4. Controller Node Quiz

### Table of Contents

|                             |     |
|-----------------------------|-----|
| Day 1, 14:25 to 14:45 ..... | 149 |
|-----------------------------|-----|

### Day 1, 14:25 to 14:45

Associate Training Guide, Controller Node Quiz Questions.

1. When managing images for OpenStack you can complete all those tasks with the OpenStack dashboard. (True or False).
  - a. True
  - b. False
2. When setting up access and security, SSH credentials (keypairs) must be injected into images after they are launched with a script. (True or False).
  - a. True
  - b. False
3. You can track monthly costs with metrics like: (choose all that apply).
  - a. VCPU
  - b. QoS

- c. Uptime
  - d. Disks
  - e. RAM
4. The following OpenStack command-line clients are available. (choose all that apply).
- a. python-keystoneclient
  - b. python-hypervisorclient
  - c. python-imageclient
  - d. python-cinderclient
  - e. python-novaclient
5. To install a client package. Run this command:
- ```
# pip install [--update] python-project client (True or False)
```
- a. True
  - b. False
6. To list images. Run this command:
- ```
$ glance image-list
```
- a. True
  - b. False

7. **When troubleshooting image creation you will need to look at which of the following log files for errors? (choose all that apply).**
- a. Examine the /var/log/nova-api.log
  - b. Examine the /var/log/nova-compute.log
  - c. Examine the /var/log/nova-error.log
  - d. Examine the /var/log/nova-status.log
  - e. Examine the /var/log/nova-image.log
8. **To generate a keypair use the following command syntax: \$ nova keypair-add --pub\_key ~/.ssh/id\_rsa.pub KEY\_NAME.**
- a. True
  - b. False
9. **When you want to launch an instance you can only do that from an image. (True or False).**
- a. True
  - b. False
10. **An instance has a Private IP address which has the following properties? (choose all that apply).**
- a. Used for communication between instances
  - b. VMware vSphere 4.1, update 1 or greater
  - c. Stays the same, even after reboots

- d. Stays allocated, even if you terminate the instance
- e. To see the status of the Private IP addresses you use the following command: `$ nova floating-ip-pool-list`

**11.To start and stop and instance you can use the following options: (choose all that apply).**

- a. Pause/Un-pause
- b. Suspend/Resume
- c. Reboot
- d. Evacuate
- e. Shutdown/Restart

**12.To create a network in OpenStack use the following command: `$ neutron net-create net1` (True or False).**

- a. True
- b. False

**13.Identity Service provides the following functions: (choose all that apply).**

- a. Group policy objects
- b. Message queuing
- c. User management
- d. Publishing

- e. Service catalog

**14.The AMQP supports the following messaging bus options: (choose all that apply).**

- a. ZeroMQ
- b. RabbitMQ
- c. Tibco Rendezvous
- d. IBM WebSphere Message Broker
- e. Qpid

**15.OpenStack uses the term tenant but in earlier versions it used the term customer. (True or False).**

- a. True
- b. False

**Associate Training Guide, Controller Node Quiz Answers.**

1. B (False) - you can manage images through only the glance and nova clients or the Image Service and Compute APIs.
2. B (False) - Keypairs are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the cloud-init package
3. A, C, D, E - You can track costs per month by showing metrics like number of VCPUs, disks, RAM, and uptime of all your instances
4. A, D, E - The following command-line clients are available for the respective services' APIs: cinder(python-cinderclient) Client for the Block Storage service API. Use to create and manage volumes. glance(python-

glanceclient) Client for the Image Service API. Use to create and manage images. keystone(python-keystoneclient) Client for the Identity Service API. Use to create and manage users, tenants, roles, endpoints, and credentials. nova(python-novaclient) Client for the Compute API and its extensions. Use to create and manage images, instances, and flavors. neutron(python-neutronclient) Client for the Networking API. Use to configure networks for guest servers. This client was previously known as neutron. swift(python-swiftclient) Client for the Object Storage API. Use to gather statistics, list items, update metadata, upload, download and delete files stored by the object storage service. Provides access to a swift installation for ad hoc processing. heat(python-heatclient)

- 5. A (True)
- 6. A (True)
- 7. A, B
- 8. B (False) - \$ nova keypair-add KEY\_NAME > MY\_KEY.pem
- 9. B (False) - you can launch an instance from an image or a volume
- 10. A, B, C
- 11. A, B, C, D
- 12. A (True)
- 13. C, E
- 14. A, B, E
- 15. B (False) - Because the term project was used instead of tenant in earlier versions of OpenStack Compute, some command-line tools use --project-id instead of --tenant-id or --os-tenant-id to refer to a tenant ID.



## 5. Compute Node

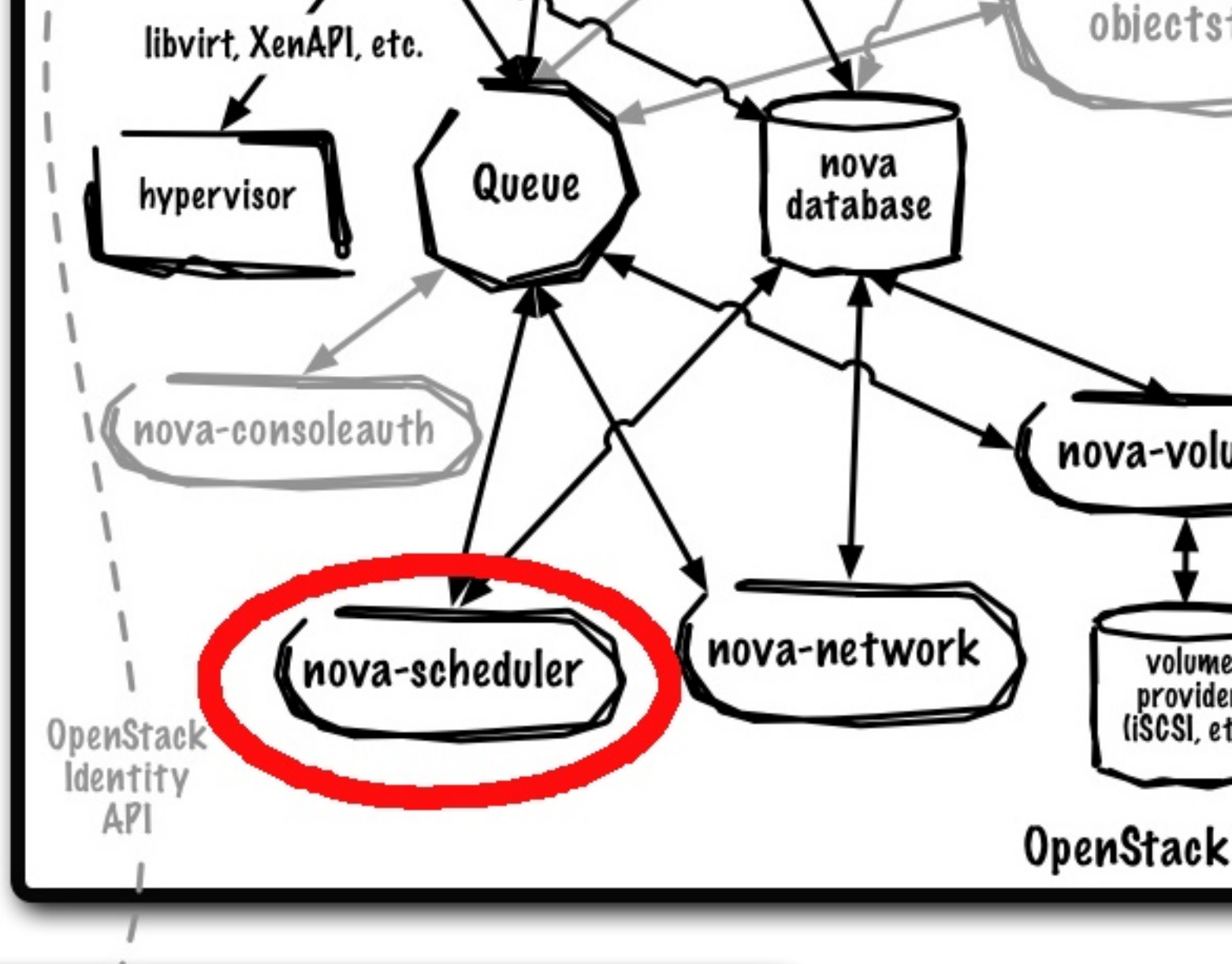
### Table of Contents

|                                |     |
|--------------------------------|-----|
| Day 1, 15:00 to 17:00 .....    | 155 |
| VM Placement .....             | 155 |
| VM provisioning in-depth ..... | 163 |
| OpenStack Block Storage .....  | 167 |
| Administration Tasks .....     | 172 |

### Day 1, 15:00 to 17:00

#### VM Placement

Compute uses the nova-scheduler service to determine how to dispatch compute and volume requests. For example, the nova-scheduler service determines which host a VM should launch on. The term host in the context of filters means a physical node that has a nova - compute service running on it. You can configure the scheduler through a variety of options.



Just as shown by above figure, nova-scheduler interacts with other components through queue and central database repo. For scheduling, queue is the essential communications hub.

All compute nodes (also known as hosts in terms of OpenStack) periodically publish their status, resources available and hardware capabilities to nova-scheduler through the queue. nova-scheduler then collects this data and uses it to make decisions when a request comes in.

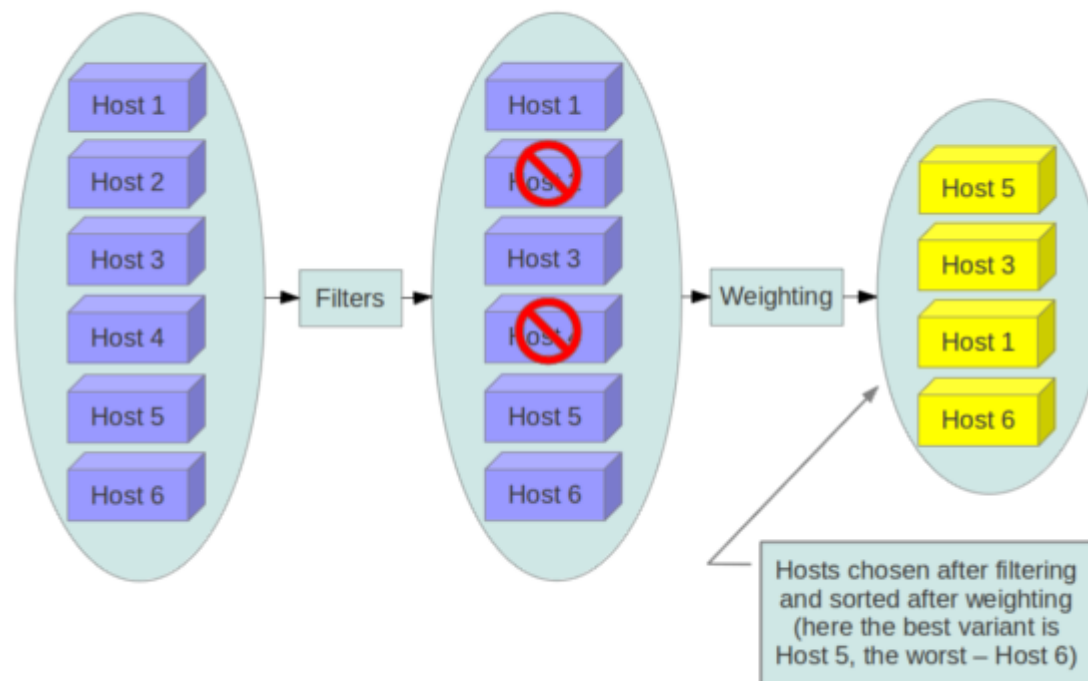
By default, the compute scheduler is configured as a filter scheduler, as described in the next section. In the default configuration, this scheduler considers hosts that meet all the following criteria:

- Are in the requested availability zone (AvailabilityZoneFilter).
- Have sufficient RAM available (RamFilter).
- Are capable of servicing the request (ComputeFilter).

### **Filter Scheduler**

The Filter Scheduler supports filtering and weighting to make informed decisions on where a new instance should be created. This Scheduler supports only working with Compute Nodes.

### **Filtering**

**Figure 5.2. Filtering**

During its work, Filter Scheduler first makes a dictionary of unfiltered hosts, then filters them using filter properties and finally chooses hosts for the requested number of instances (each time it chooses the most weighed host and appends it to the list of selected hosts).

If it turns up, that it can't find candidates for the next instance, it means that there are no more appropriate hosts where the instance could be scheduled.

If we speak about filtering and weighting, their work is quite flexible in the Filter Scheduler. There are a lot of filtering strategies for the Scheduler to support. Also you can even implement your own algorithm of filtering.

There are some standard filter classes to use (`nova.scheduler.filters`):

- `AllHostsFilter` - frankly speaking, this filter does no operation. It passes all the available hosts.
- `ImagePropertiesFilter` - filters hosts based on properties defined on the instance's image. It passes hosts that can support the specified image properties contained in the instance.
- `AvailabilityZoneFilter` - filters hosts by availability zone. It passes hosts matching the availability zone specified in the instance properties.
- `ComputeCapabilitiesFilter` - checks that the capabilities provided by the host Compute service satisfy any extra specifications associated with the instance type. It passes hosts that can create the specified instance type.
- The extra specifications can have a scope at the beginning of the key string of a key/value pair. The scope format is `scope:key` and can be nested, i.e. `key_string := scope:key_string`. Example like `capabilities:cpu_info: features` is valid scope format. A key string without any `:` is non-scope format. Each filter defines its valid scope, and not all filters accept non-scope format.
- The extra specifications can have an operator at the beginning of the value string of a key/value pair. If there is no operator specified, then a default operator of `s==` is used. Valid operators are:  
`*` = (equal to or greater than as a number; same as `vcpus` case) `* ==` (equal to as a number) `* !=` (not equal to as a number) `* >=` (greater than or equal to as a number) `* <=` (less than or equal to as a number) `* s==` (equal to as a string) `* s!=` (not equal to as a string) `* s>=` (greater than or equal to as a string) `* s>` (greater than as a string) `* s<=` (less than or equal to as a string) `* s<` (less than as a string) `* <in>` (substring) `* <or>` (find one of these) Examples are: `">= 5"`, `"s== 2.1.0"`, `"<in> gcc"`, and `"<or> fpu <or> gpu"`

```
class RamFilter(filters.BaseHostFilter):
    """Ram Filter with over subscription flag"""

    def host_passes(self, host_state, filter_properties):
        """Only return hosts with sufficient available RAM."""

        instance_type = filter_properties.get('instance_type')
        requested_ram = instance_type['memory_mb']
        free_ram_mb = host_state.free_ram_mb
        total_usable_ram_mb = host_state.total_usable_ram_mb
        used_ram_mb = total_usable_ram_mb - free_ram_mb
        return total_usable_ram_mb * FLAGS.ram_allocation_ratio - used_ram_mb >= requested_ram
```

Here `ram_allocation_ratio` means the virtual RAM to physical RAM allocation ratio (it is 1.5 by default). Really, nice and simple.

Next standard filter to describe is `AvailabilityZoneFilter` and it isn't difficult too. This filter just looks at the availability zone of compute node and availability zone from the properties of the request. Each Compute service has its own availability zone. So deployment engineers have an option to run scheduler with availability zones support and can configure availability zones on each compute host. This classes method `host_passes` returns True if availability zone mentioned in request is the same on the current compute host.

The `ImagePropertiesFilter` filters hosts based on the architecture, hypervisor type, and virtual machine mode specified in the instance. E.g., an instance might require a host that supports the arm architecture on a qemu compute host. The `ImagePropertiesFilter` will only pass hosts that can satisfy this request. These instance properties are populated from properties define on the instance's image. E.g. an image can be decorated with these properties using `glance image-update img-uuid --property architecture=arm --property hypervisor_type=qemu` Only hosts that satisfy these requirements will pass the `ImagePropertiesFilter`.

`ComputeCapabilitiesFilter` checks if the host satisfies any `extra_specs` specified on the instance type. The `extra_specs` can contain key/value pairs. The key for the filter is either non-scope format (i.e. no : contained), or scope format in capabilities scope (i.e. capabilities:xxx:yyy). One example of capabilities scope is `capabilities:cpu_info:features`, which will match host's cpu features capabilities. The `ComputeCapabilitiesFilter`

will only pass hosts whose capabilities satisfy the requested specifications. All hosts are passed if no `extra_specs` are specified.

ComputeFilter is quite simple and passes any host whose Compute service is enabled and operational.

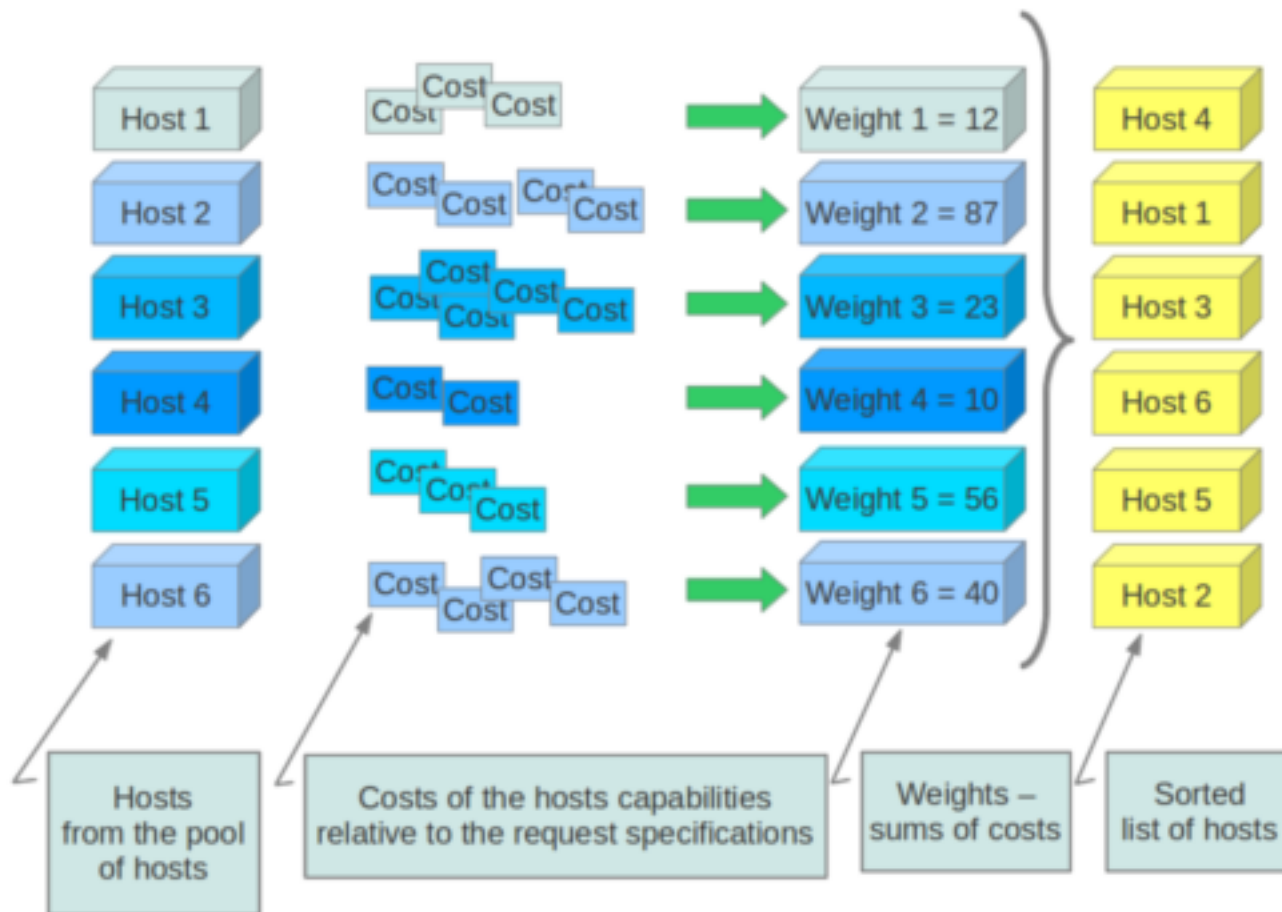
Now we are going to IsolatedHostsFilter. There can be some special hosts reserved for specific images. These hosts are called isolated. So the images to run on the isolated hosts are also called isolated. This Scheduler checks if `image_isolated` flag named in instance specifications is the same that the host has.

## Weights

Filter Scheduler uses so-called weights during its work.

The Filter Scheduler weights hosts based on the config option `scheduler_weight_classes`, this defaults to `nova.scheduler.weights.all_weighters`, which selects the only weigher available – the `RamWeigher`. Hosts are then weighted and sorted with the largest weight winning.

Filter Scheduler finds local list of acceptable hosts by repeated filtering and weighing. Each time it chooses a host, it virtually consumes resources on it, so subsequent selections can adjust accordingly. It is useful if the customer asks for the same large amount of instances, because weight is computed for each instance requested.

**Figure 5.3. Weights**



In the end Filter Scheduler sorts selected hosts by their weight and provisions instances on them.

## VM provisioning in-depth

The request flow for provisioning an instance goes like this:

1. The dashboard or CLI gets the user credentials and authenticates with the Identity Service via REST API.

The Identity Service authenticates the user with the user credentials, and then generates and sends back an auth-token which will be used for sending the request to other components through REST-call.

2. The dashboard or CLI converts the new instance request specified in **launch instance** or **nova-boot** form to a REST API request and sends it to nova-api.
3. nova-api receives the request and sends a request to the Identity Service for validation of the auth-token and access permission.

The Identity Service validates the token and sends updated authentication headers with roles and permissions.

4. nova-api checks for conflicts with nova-database.

nova-api creates initial database entry for a new instance.

5. nova-api sends the rpc.call request to nova-scheduler expecting to get updated instance entry with host ID specified.
6. nova-scheduler picks up the request from the queue.
7. nova-scheduler interacts with nova-database to find an appropriate host via filtering and weighing.  
nova-scheduler returns the updated instance entry with the appropriate host ID after filtering and weighing.

nova-scheduler sends the rpc.cast request to nova-compute for launching an instance on the appropriate host.

8. nova-compute picks up the request from the queue.

9. nova-compute sends the rpc.call request to nova-conductor to fetch the instance information such as host ID and flavor (RAM, CPU, Disk).

10. nova-conductor picks up the request from the queue.

11. nova-conductor interacts with nova-database.

nova-conductor returns the instance information.

nova-compute picks up the instance information from the queue.

12. nova-compute performs the REST call by passing the auth-token to glance-api. Then, nova-compute uses the Image ID to retrieve the Image URI from the Image Service, and loads the image from the image storage.

13. glance-api validates the auth-token with keystone.

nova-compute gets the image metadata.

14. nova-compute performs the REST-call by passing the auth-token to Network API to allocate and configure the network so that the instance gets the IP address.

15. neutron-server validates the auth-token with keystone.

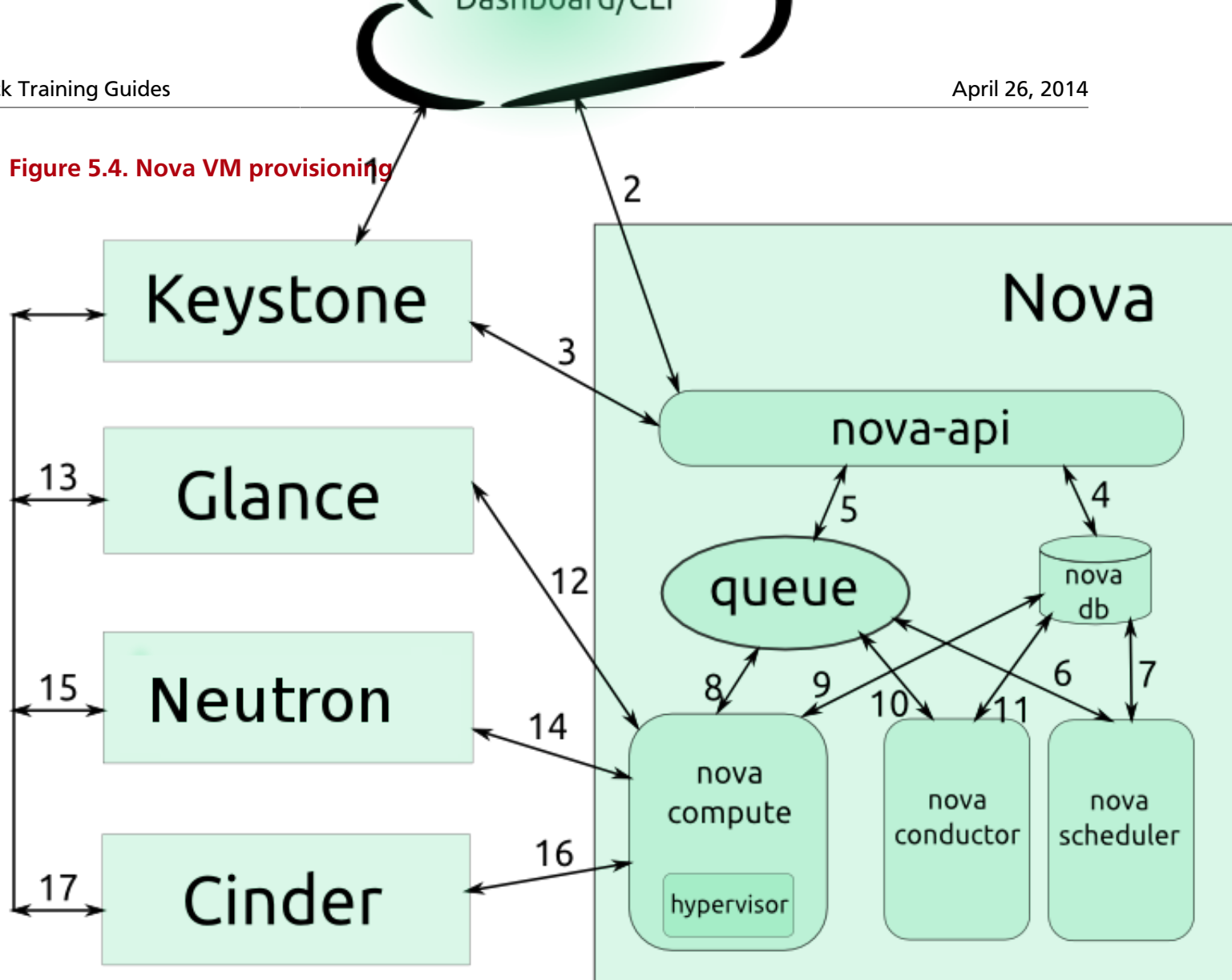
nova-compute retrieves the network info.

16. nova-compute performs the REST call by passing the auth-token to Volume API to attach volumes to the instance.

17.cinder - api validates the auth-token with keystone.

nova - compute retrieves the block storage info.

18.nova - compute generates data for the hypervisor driver and executes the request on the hypervisor (via libvirt or API).

**Figure 5.4. Nova VM provisioning**

# OpenStack Block Storage

## Block Storage and OpenStack Compute

OpenStack provides two classes of block storage, "ephemeral" storage and persistent "volumes". Ephemeral storage exists only for the life of an instance, it will persist across reboots of the guest operating system but when the instance is deleted so is the associated storage. All instances have some ephemeral storage. Volumes are persistent virtualized block devices independent of any particular instance. Volumes may be attached to a single instance at a time, but may be detached or reattached to a different instance while retaining all data, much like a USB drive.

## Ephemeral Storage

Ephemeral storage is associated with a single unique instance. Its size is defined by the flavor of the instance.

Data on ephemeral storage ceases to exist when the instance it is associated with is terminated. Rebooting the VM or restarting the host server, however, will not destroy ephemeral data. In the typical use case an instance's root filesystem is stored on ephemeral storage. This is often an unpleasant surprise for people unfamiliar with the cloud model of computing.

In addition to the ephemeral root volume all flavors except the smallest, m1.tiny, provide an additional ephemeral block device varying from 20G for the m1.small through 160G for the m1.xlarge by default - these sizes are configurable. This is presented as a raw block device with no partition table or filesystem. Cloud aware operating system images may discover, format, and mount this device. For example the cloud-init package included in Ubuntu's stock cloud images will format this space as an ext3 filesystem and mount it on /mnt. It is important to note this a feature of the guest operating system. OpenStack only provisions the raw storage.

## Volume Storage

Volume storage is independent of any particular instance and is persistent. Volumes are user created and within quota and availability limits may be of any arbitrary size.

When first created volumes are raw block devices with no partition table and no filesystem. They must be attached to an instance to be partitioned and/or formatted. Once this is done they may be used much like an external disk drive. Volumes may attached to only one instance at a time, but may be detached and reattached to either the same or different instances.

It is possible to configure a volume so that it is bootable and provides a persistent virtual instance similar to traditional non-cloud based virtualization systems. In this use case the resulting instance may still have ephemeral storage depending on the flavor selected, but the root filesystem (and possibly others) will be on the persistent volume and thus state will be maintained even if the instance is shutdown. Details of this configuration are discussed in the [OpenStack End User Guide](#).

Volumes do not provide concurrent access from multiple instances. For that you need either a traditional network filesystem like NFS or CIFS or a cluster filesystem such as GlusterFS. These may be built within an OpenStack cluster or provisioned outside of it, but are not features provided by the OpenStack software.

The OpenStack Block Storage service works via the interaction of a series of daemon processes named cinder-\* that reside persistently on the host machine or machines. The binaries can all be run from a single node, or spread across multiple nodes. They can also be run on the same node as other OpenStack services.

**The current services available in OpenStack Block Storage are:**

- **cinder-api** - The cinder-api service is a WSGI app that authenticates and routes requests throughout the Block Storage system. It supports the OpenStack API's only, although there is a translation that can be done via Nova's EC2 interface which calls in to the cinderclient.
- **cinder-scheduler** - The cinder-scheduler is responsible for scheduling/routing requests to the appropriate volume service. As of Grizzly; depending upon your configuration this may be simple round-robin scheduling to the running volume services, or it can be more sophisticated through the use of the Filter Scheduler. The Filter Scheduler is the default in Grizzly and enables filter on things like Capacity, Availability Zone, Volume Types and Capabilities as well as custom filters.
- **cinder-volume** - The cinder-volume service is responsible for managing Block Storage devices, specifically the back-end devices themselves.

- **cinder-backup** - The cinder-backup service provides a means to back up a Cinder Volume to OpenStack Object Store (SWIFT).

### **Introduction to OpenStack Block Storage**

OpenStack Block Storage provides persistent High Performance Block Storage resources that can be consumed by OpenStack Compute instances. This includes secondary attached storage similar to Amazon's Elastic Block Storage (EBS). In addition images can be written to a Block Storage device and specified for OpenStack Compute to use a bootable persistent instance.

There are some differences from Amazon's EBS that one should be aware of. OpenStack Block Storage is not a shared storage solution like NFS, but currently is designed so that the device is attached and in use by a single instance at a time.

### **Backend Storage Devices**

OpenStack Block Storage requires some form of back-end storage that the service is built on. The default implementation is to use LVM on a local Volume Group named "cinder-volumes". In addition to the base driver implementation, OpenStack Block Storage also provides the means to add support for other storage devices to be utilized such as external Raid Arrays or other Storage appliances.

### **Users and Tenants (Projects)**

The OpenStack Block Storage system is designed to be used by many different cloud computing consumers or customers, basically tenants on a shared system, using role-based access assignments. Roles control the actions that a user is allowed to perform. In the default configuration, most actions do not require a particular role, but this is configurable by the system administrator editing the appropriate policy.json file that maintains the rules. A user's access to particular volumes is limited by tenant, but the username and password are assigned per user. Key pairs granting access to a volume are enabled per user, but quotas to control resource consumption across available hardware resources are per tenant.

**For tenants, quota controls are available to limit the:**

- Number of volumes which may be created
- Number of snapshots which may be created
- Total number of Giga Bytes allowed per tenant (shared between snapshots and volumes)

### **Volumes Snapshots and Backups**

This introduction provides a high level overview of the two basic resources offered by the OpenStack Block Storage service. The first is Volumes and the second is Snapshots which are derived from Volumes.

#### **Volumes**

Volumes are allocated block storage resources that can be attached to instances as secondary storage or they can be used as the root store to boot instances. Volumes are persistent R/W Block Storage devices most commonly attached to the compute node via iSCSI.

#### **Snapshots**

A Snapshot in OpenStack Block Storage is a read-only point in time copy of a Volume. The Snapshot can be created from a Volume that is currently in use (via the use of '-force True') or in an available state. The Snapshot can then be used to create a new volume via create from snapshot.

#### **Backups**

A Backup is an archived copy of a Volume currently stored in Object Storage (Swift).

### **Managing Volumes**

Cinder is the OpenStack service that allows you to give extra block level storage to your OpenStack Compute instances. You may recognize this as a similar offering from Amazon EC2 known as Elastic Block Storage (EBS). The default Cinder implementation is an iSCSI solution that employs the use of Logical Volume Manager (LVM) for Linux. Note that a volume may only be attached to one instance at a time. This is not a 'shared storage' solution like a SAN or NFS on which multiple servers can attach to. It's also important to note that



Cinder also includes a number of drivers to allow you to use a number of other vendor's back-end storage devices in addition to or instead of the base LVM implementation.

Here is brief walk-through of a simple create/attach sequence, keep in mind this requires proper configuration of both OpenStack Compute via `cinder.conf` and OpenStack Block Storage via `cinder.conf`.

1. The volume is created via `cinder create`; which creates an LV into the volume group (VG) "cinder-volumes"
2. The volume is attached to an instance via `nova volume-attach`; which creates a unique iSCSI IQN that will be exposed to the compute node
3. The compute node which run the concerned instance has now an active iSCSI session; and a new local storage (usually a `/dev/sdX` disk)
4. `libvirt` uses that local storage as a storage for the instance; the instance get a new disk (usually a `/dev/vdX` disk)

### **Block Storage Capabilities**

- OpenStack provides persistent block level storage devices for use with OpenStack compute instances.
- The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs.
- In addition to using simple Linux server storage, it has unified storage support for numerous storage platforms including Ceph, NetApp, Nexenta, SolidFire, and Zadara.
- Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage.
- Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

# Administration Tasks

## Block Storage CLI Commands

The **cinder** client is the command-line interface (CLI) for the OpenStack Block Storage API and its extensions. This chapter documents **cinder** version 1.0.8.

For help on a specific **cinder** command, enter:

```
$ cinder help COMMAND
```

### cinder usage

```
usage: cinder [--version] [--debug] [--os-username <auth-user-name>]
             [--os-password <auth-password>]
             [--os-tenant-name <auth-tenant-name>]
             [--os-tenant-id <auth-tenant-id>] [--os-auth-url <auth-url>]
             [--os-region-name <region-name>] [--service-type <service-type>]
             [--service-name <service-name>]
             [--volume-service-name <volume-service-name>]
             [--endpoint-type <endpoint-type>]
             [--os-volume-api-version <volume-api-ver>]
             [--os-cacert <ca-certificate>] [--retries <retries>]
             <subcommand> ...
```

### Subcommands

|                               |                                            |
|-------------------------------|--------------------------------------------|
| <b>absolute-limits</b>        | Print a list of absolute limits for a user |
| <b>availability-zone-list</b> | List all the availability zones.           |
| <b>backup-create</b>          | Creates a backup.                          |

---

|                               |                                                                      |
|-------------------------------|----------------------------------------------------------------------|
| <b>backup-delete</b>          | Remove a backup.                                                     |
| <b>backup-list</b>            | List all the backups.                                                |
| <b>backup-restore</b>         | Restore a backup.                                                    |
| <b>backup-show</b>            | Show details about a backup.                                         |
| <b>create</b>                 | Add a new volume.                                                    |
| <b>credentials</b>            | Show user credentials returned from auth.                            |
| <b>delete</b>                 | Remove volume(s).                                                    |
| <b>encryption-type-create</b> | Create a new encryption type for a volume type (Admin Only).         |
| <b>encryption-type-delete</b> | Delete the encryption type for a volume type (Admin Only).           |
| <b>encryption-type-list</b>   | List encryption type information for all volume types (Admin Only).  |
| <b>encryption-type-show</b>   | Show the encryption type information for a volume type (Admin Only). |
| <b>endpoints</b>              | Discover endpoints that get returned from the authenticate services. |
| <b>extend</b>                 | Attempt to extend the size of an existing volume.                    |
| <b>extra-specs-list</b>       | Print a list of current 'volume types and extra specs' (Admin Only). |
| <b>force-delete</b>           | Attempt forced removal of volume(s), regardless of the state(s).     |
| <b>list</b>                   | List all the volumes.                                                |
| <b>metadata</b>               | Set or Delete metadata on a volume.                                  |

---

|                             |                                                      |
|-----------------------------|------------------------------------------------------|
| <b>metadata-show</b>        | Show metadata of given volume.                       |
| <b>metadata-update-all</b>  | Update all metadata of a volume.                     |
| <b>migrate</b>              | Migrate the volume to the new host.                  |
| <b>qos-associate</b>        | Associate qos specs with specific volume type.       |
| <b>qos-create</b>           | Create a new qos specs.                              |
| <b>qos-delete</b>           | Delete a specific qos specs.                         |
| <b>qos-disassociate</b>     | Disassociate qos specs from specific volume type.    |
| <b>qos-disassociate-all</b> | Disassociate qos specs from all of its associations. |
| <b>qos-get-association</b>  | Get all associations of specific qos specs.          |
| <b>qos-key</b>              | Set or unset specifications for a qos spec.          |
| <b>qos-list</b>             | Get full list of qos specs.                          |
| <b>qos-show</b>             | Get a specific qos specs.                            |
| <b>quota-class-show</b>     | List the quotas for a quota class.                   |
| <b>quota-class-update</b>   | Update the quotas for a quota class.                 |
| <b>quota-defaults</b>       | List the default quotas for a tenant.                |
| <b>quota-show</b>           | List the quotas for a tenant.                        |
| <b>quota-update</b>         | Update the quotas for a tenant.                      |
| <b>quota-usage</b>          | List the quota usage for a tenant.                   |

---

|                                     |                                                         |
|-------------------------------------|---------------------------------------------------------|
| <b>rate-limits</b>                  | Print a list of rate limits for a user                  |
| <b>readonly-mode-update</b>         | Update volume read-only access mode read_only.          |
| <b>rename</b>                       | Rename a volume.                                        |
| <b>reset-state</b>                  | Explicitly update the state of a volume.                |
| <b>service-disable</b>              | Disable the service.                                    |
| <b>service-enable</b>               | Enable the service.                                     |
| <b>service-list</b>                 | List all the services. Filter by host & service binary. |
| <b>show</b>                         | Show details about a volume.                            |
| <b>snapshot-create</b>              | Add a new snapshot.                                     |
| <b>snapshot-delete</b>              | Remove a snapshot.                                      |
| <b>snapshot-list</b>                | List all the snapshots.                                 |
| <b>snapshot-metadata</b>            | Set or Delete metadata of a snapshot.                   |
| <b>snapshot-metadata-show</b>       | Show metadata of given snapshot.                        |
| <b>snapshot-metadata-update-all</b> | Update all metadata of a snapshot.                      |
| <b>snapshot-rename</b>              | Rename a snapshot.                                      |
| <b>snapshot-reset-state</b>         | Explicitly update the state of a snapshot.              |
| <b>snapshot-show</b>                | Show details about a snapshot.                          |
| <b>transfer-accept</b>              | Accepts a volume transfer.                              |

---

|                        |                                                            |
|------------------------|------------------------------------------------------------|
| <b>transfer-create</b> | Creates a volume transfer.                                 |
| <b>transfer-delete</b> | Undo a transfer.                                           |
| <b>transfer-list</b>   | List all the transfers.                                    |
| <b>transfer-show</b>   | Show details about a transfer.                             |
| <b>type-create</b>     | Create a new volume type.                                  |
| <b>type-delete</b>     | Delete a specific volume type.                             |
| <b>type-key</b>        | Set or unset extra_spec for a volume type.                 |
| <b>type-list</b>       | Print a list of available 'volume types'.                  |
| <b>upload-to-image</b> | Upload volume to image service as image.                   |
| <b>bash-completion</b> | Print arguments for bash_completion.                       |
| <b>help</b>            | Display help about this program or one of its subcommands. |
| <b>list-extensions</b> | List all the os-api extensions that are available.         |

## cinder optional arguments

|                                            |                                        |
|--------------------------------------------|----------------------------------------|
| <b>-version</b>                            | show program's version number and exit |
| <b>-debug</b>                              | Print debugging output                 |
| <b>-os-username &lt;auth-user-name&gt;</b> | Defaults to env [OS_USERNAME].         |
| <b>-os-password &lt;auth-password&gt;</b>  | Defaults to env [OS_PASSWORD].         |

|                                                         |                                                                                                           |
|---------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| <b>-os-tenant-name &lt;auth-tenant-name&gt;</b>         | Defaults to env[OS_TENANT_NAME].                                                                          |
| <b>-os-tenant-id &lt;auth-tenant-id&gt;</b>             | Defaults to env[OS_TENANT_ID].                                                                            |
| <b>-os-auth-url &lt;auth-url&gt;</b>                    | Defaults to env[OS_AUTH_URL].                                                                             |
| <b>-os-region-name &lt;region-name&gt;</b>              | Defaults to env[OS_REGION_NAME].                                                                          |
| <b>-service-type &lt;service-type&gt;</b>               | Defaults to volume for most actions                                                                       |
| <b>-service-name &lt;service-name&gt;</b>               | Defaults to env[CINDER_SERVICE_NAME]                                                                      |
| <b>-volume-service-name &lt;volume-service-name&gt;</b> | Defaults to env[CINDER_VOLUME_SERVICE_NAME]                                                               |
| <b>-endpoint-type &lt;endpoint-type&gt;</b>             | Defaults to env[CINDER_ENDPOINT_TYPE] or publicURL.                                                       |
| <b>-os-volume-api-version &lt;volume-api-ver&gt;</b>    | Accepts 1 or 2, defaults to env[OS_VOLUME_API_VERSION].                                                   |
| <b>-os-cacert &lt;ca-certificate&gt;</b>                | Specify a CA bundle file to use in verifying a TLS (https) server certificate. Defaults to env[OS_CACERT] |
| <b>-retries &lt;retries&gt;</b>                         | Number of retries.                                                                                        |

## cinder absolute-limits command

```
usage: cinder absolute-limits
```

Print a list of absolute limits for a user

## cinder availability-zone-list command

```
usage: cinder availability-zone-list
```

List all the availability zones.

## cinder backup-create command

```
usage: cinder backup-create [--container <container>]
                             [--display-name <display-name>]
                             [--display-description <display-description>]
                             <volume>
```

Creates a backup.

### Positional arguments

**<volume>**     Name or ID of the volume to backup.

### Optional arguments

|                                                         |                                                |
|---------------------------------------------------------|------------------------------------------------|
| <b>-container &lt;container&gt;</b>                     | Optional Backup container name. (Default=None) |
| <b>-display-name &lt;display-name&gt;</b>               | Optional backup name. (Default=None)           |
| <b>-display-description &lt;display-description&gt;</b> | Optional backup description. (Default=None)    |

## cinder backup-delete command

```
usage: cinder backup-delete <backup>
```



Remove a backup.

### Positional arguments

**<backup>** Name or ID of the backup to delete.

## cinder backup-list command

```
usage: cinder backup-list
```

List all the backups.

## cinder backup-restore command

```
usage: cinder backup-restore [--volume-id <volume>] <backup>
```

Restore a backup.

### Positional arguments

**<backup>** ID of the backup to restore.

### Optional arguments

**-volume-id <volume>** Optional ID(or name) of the volume to restore to.

## cinder backup-show command

```
usage: cinder backup-show <backup>
```

Show details about a backup.

## Positional arguments

**<backup>**     Name or ID of the backup.

## cinder create command

```
usage: cinder create [--snapshot-id <snapshot-id>]
                   [--source-volid <source-volid>] [--image-id <image-id>]
                   [--display-name <display-name>]
                   [--display-description <display-description>]
                   [--volume-type <volume-type>]
                   [--availability-zone <availability-zone>]
                   [--metadata [<key=value> [<key=value> ...]]]
                   <size>
```

Add a new volume.

## Positional arguments

**<size>**     Size of volume in GB

## Optional arguments

|                                                          |                                                         |
|----------------------------------------------------------|---------------------------------------------------------|
| <b>--snapshot-id &lt;snapshot-id&gt;</b>                 | Create volume from snapshot id (Optional, Default=None) |
| <b>--source-volid &lt;source-volid&gt;</b>               | Create volume from volume id (Optional, Default=None)   |
| <b>--image-id &lt;image-id&gt;</b>                       | Create volume from image id (Optional, Default=None)    |
| <b>--display-name &lt;display-name&gt;</b>               | Volume name (Optional, Default=None)                    |
| <b>--display-description &lt;display-description&gt;</b> | Volume description (Optional, Default=None)             |

|                                                                  |                                                       |
|------------------------------------------------------------------|-------------------------------------------------------|
| <b>-volume-type &lt;volume-type&gt;</b>                          | Volume type (Optional, Default=None)                  |
| <b>-availability-zone &lt;availability-zone&gt;</b>              | Availability zone for volume (Optional, Default=None) |
| <b>-metadata [&lt;key=value&gt;<br/>[&lt;key=value&gt; ...]]</b> | Metadata key=value pairs (Optional, Default=None)     |

## cinder credentials command

```
usage: cinder credentials
```

Show user credentials returned from auth.

## cinder delete command

```
usage: cinder delete <volume> [<volume> ...]
```

Remove volume(s).

### Positional arguments

**<volume>** Name or ID of the volume(s) to delete.

## cinder encryption-type-create command

```
usage: cinder encryption-type-create [--cipher <cipher>]  
                                     [--key_size <key_size>]  
                                     [--control_location <control_location>]  
                                     <volume_type> <provider>
```

Create a new encryption type for a volume type (Admin Only).

### Positional arguments

|                            |                                                         |
|----------------------------|---------------------------------------------------------|
| <b>&lt;volume_type&gt;</b> | Name or ID of the volume type                           |
| <b>&lt;provider&gt;</b>    | Class providing encryption support (e.g. LuksEncryptor) |

### Optional arguments

|                                                       |                                                                                                                                    |
|-------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|
| <b>-cipher &lt;cipher&gt;</b>                         | Encryption algorithm/mode to use (e.g., aes-xts-plain64) (Optional, Default=None)                                                  |
| <b>-key_size &lt;key_size&gt;</b>                     | Size of the encryption key, in bits (e.g., 128, 256) (Optional, Default=None)                                                      |
| <b>-control_location<br/>&lt;control_location&gt;</b> | Notional service where encryption is performed (e.g., front-end=Nova).<br>Values: 'front-end', 'back-end' (Optional, Default=None) |

## cinder encryption-type-delete command

```
usage: cinder encryption-type-delete <volume_type>
```

Delete the encryption type for a volume type (Admin Only).

### Positional arguments

|                            |                               |
|----------------------------|-------------------------------|
| <b>&lt;volume_type&gt;</b> | Name or ID of the volume type |
|----------------------------|-------------------------------|

## cinder encryption-type-list command

```
usage: cinder encryption-type-list
```

List encryption type information for all volume types (Admin Only).

## cinder encryption-type-show command

```
usage: cinder encryption-type-show <volume_type>
```

Show the encryption type information for a volume type (Admin Only).

### Positional arguments

**<volume\_type>**     Name or ID of the volume type

## cinder endpoints command

```
usage: cinder endpoints
```

Discover endpoints that get returned from the authenticate services.

## cinder extend command

```
usage: cinder extend <volume> <new-size>
```

Attempt to extend the size of an existing volume.

### Positional arguments

**<volume>**     Name or ID of the volume to extend.

**<new-size>**      New size of volume in GB

## cinder extra-specs-list command

```
usage: cinder extra-specs-list
```

Print a list of current 'volume types and extra specs' (Admin Only).

## cinder force-delete command

```
usage: cinder force-delete <volume> [<volume> ...]
```

Attempt forced removal of volume(s), regardless of the state(s).

### Positional arguments

**<volume>**      Name or ID of the volume(s) to delete.

## cinder list command

```
usage: cinder list [--all-tenants [<0|1>]] [--display-name <display-name>]
                  [--status <status>]
                  [--metadata [<key=value> [<key=value> ...]]]
```

List all the volumes.

### Optional arguments

**--all-tenants [<0|1>]**      Display information from all tenants (Admin only).

|                                                                  |                                |
|------------------------------------------------------------------|--------------------------------|
| <b>-display-name &lt;display-name&gt;</b>                        | Filter results by display-name |
| <b>-status &lt;status&gt;</b>                                    | Filter results by status       |
| <b>-metadata [&lt;key=value&gt;<br/>[&lt;key=value&gt; ...]]</b> | Filter results by metadata     |

## cinder list-extensions command

```
usage: cinder list-extensions
```

List all the os-api extensions that are available.

## cinder metadata command

```
usage: cinder metadata <volume> <action> <key=value> [<key=value> ...]
```

Set or Delete metadata on a volume.

### Positional arguments

|                          |                                                        |
|--------------------------|--------------------------------------------------------|
| <b>&lt;volume&gt;</b>    | Name or ID of the volume to update metadata on.        |
| <b>&lt;action&gt;</b>    | Actions: 'set' or 'unset'                              |
| <b>&lt;key=value&gt;</b> | Metadata to set/unset (only key is necessary on unset) |

## cinder metadata-show command

```
usage: cinder metadata-show <volume>
```

Show metadata of given volume.

### Positional arguments

**<volume>** ID of volume

## cinder metadata-update-all command

```
usage: cinder metadata-update-all <volume> <key=value> [<key=value> ...]
```

Update all metadata of a volume.

### Positional arguments

**<volume>** ID of the volume to update metadata on.

**<key=value>** Metadata entry/entries to update.

## cinder migrate command

```
usage: cinder migrate [--force-host-copy <True|False>] <volume> <host>
```

Migrate the volume to the new host.

### Positional arguments

**<volume>** ID of the volume to migrate

**<host>** Destination host



### Optional arguments

**-force-host-copy <True|False>** Optional flag to force the use of the generic host- based migration mechanism, bypassing driver optimizations (Default=False).

## cinder qos-associate command

```
usage: cinder qos-associate <qos_specs> <volume_type_id>
```

Associate qos specs with specific volume type.

### Positional arguments

**<qos\_specs>** ID of qos\_specs.

**<volume\_type\_id>** ID of volume type to be associated with.

## cinder qos-create command

```
usage: cinder qos-create <name> <key=value> [<key=value> ...]
```

Create a new qos specs.

### Positional arguments

**<name>** Name of the new QoS specs

**<key=value>** Specifications for QoS

## cinder qos-delete command

```
usage: cinder qos-delete [--force <True|False>] <qos_specs>
```

Delete a specific qos specs.

### Positional arguments

**<qos\_specs>** ID of the qos\_specs to delete.

### Optional arguments

**-force <True|False>** Optional flag that indicates whether to delete specified qos specs even if it is in-use.

## cinder qos-disassociate command

```
usage: cinder qos-disassociate <qos_specs> <volume_type_id>
```

Disassociate qos specs from specific volume type.

### Positional arguments

**<qos\_specs>** ID of qos\_specs.

**<volume\_type\_id>** ID of volume type to be associated with.

## cinder qos-disassociate-all command

```
usage: cinder qos-disassociate-all <qos_specs>
```

Disassociate qos specs from all of its associations.

### Positional arguments

**<qos\_specs>** ID of qos\_specs to be operate on.

## cinder qos-get-association command

```
usage: cinder qos-get-association <qos_specs>
```

Get all associations of specific qos specs.

### Positional arguments

**<qos\_specs>** ID of the qos\_specs.

## cinder qos-key command

```
usage: cinder qos-key <qos_specs> <action> key=value [key=value ...]
```

Set or unset specifications for a qos spec.

### Positional arguments

**<qos\_specs>** ID of qos specs

**<action>** Actions: 'set' or 'unset'

**key=value** QoS specs to set/unset (only key is necessary on unset)

## cinder qos-list command

```
usage: cinder qos-list
```

Get full list of qos specs.

## cinder qos-show command

```
usage: cinder qos-show <qos_specs>
```

Get a specific qos specs.

### Positional arguments

**<qos\_specs>** ID of the qos\_specs to show.

## cinder quota-class-show command

```
usage: cinder quota-class-show <class>
```

List the quotas for a quota class.

### Positional arguments

**<class>** Name of quota class to list the quotas for.

## cinder quota-class-update command

```
usage: cinder quota-class-update [--volumes <volumes>]
                                [--snapshots <snapshots>]
                                [--gigabytes <gigabytes>]
                                [--volume-type <volume_type_name>]
                                <class>
```

Update the quotas for a quota class.

### Positional arguments

**<class>**      Name of quota class to set the quotas for.

### Optional arguments

|                                                  |                                      |
|--------------------------------------------------|--------------------------------------|
| <b>-volumes &lt;volumes&gt;</b>                  | New value for the "volumes" quota.   |
| <b>-snapshots &lt;snapshots&gt;</b>              | New value for the "snapshots" quota. |
| <b>-gigabytes &lt;gigabytes&gt;</b>              | New value for the "gigabytes" quota. |
| <b>-volume-type<br/>&lt;volume_type_name&gt;</b> | Volume type (Optional, Default=None) |

## cinder quota-defaults command

```
usage: cinder quota-defaults <tenant_id>
```

List the default quotas for a tenant.

### Positional arguments

**<tenant\_id>**      UUID of tenant to list the default quotas for.

## cinder quota-show command

```
usage: cinder quota-show <tenant_id>
```

List the quotas for a tenant.

### Positional arguments

**<tenant\_id>**      UUID of tenant to list the quotas for.

## cinder quota-update command

```
usage: cinder quota-update [--volumes <volumes>] [--snapshots <snapshots>]
                             [--gigabytes <gigabytes>]
                             [--volume-type <volume_type_name>]
                             <tenant_id>
```

Update the quotas for a tenant.

### Positional arguments

**<tenant\_id>**      UUID of tenant to set the quotas for.

### Optional arguments

|                                                  |                                      |
|--------------------------------------------------|--------------------------------------|
| <b>-volumes &lt;volumes&gt;</b>                  | New value for the "volumes" quota.   |
| <b>-snapshots &lt;snapshots&gt;</b>              | New value for the "snapshots" quota. |
| <b>-gigabytes &lt;gigabytes&gt;</b>              | New value for the "gigabytes" quota. |
| <b>-volume-type<br/>&lt;volume_type_name&gt;</b> | Volume type (Optional, Default=None) |

## cinder quota-usage command

```
usage: cinder quota-usage <tenant_id>
```

List the quota usage for a tenant.

### Positional arguments

**<tenant\_id>**      UUID of tenant to list the quota usage for.

## cinder rate-limits command

```
usage: cinder rate-limits
```

Print a list of rate limits for a user

## cinder readonly-mode-update command

```
usage: cinder readonly-mode-update <volume> <True|true|False|false>
```

Update volume read-only access mode read\_only.

### Positional arguments

**<volume>**                      ID of the volume to update.

**<True|true|False|false>**      Flag to indicate whether to update volume to read-only access mode.

## cinder rename command

```
usage: cinder rename [--display-description <display-description>]
                   <volume> [<display-name>]
```

Rename a volume.

### Positional arguments

**<volume>** Name or ID of the volume to rename.

**<display-name>** New display-name for the volume.

### Optional arguments

**-display-description <display-description>** Optional volume description. (Default=None)

## cinder reset-state command

```
usage: cinder reset-state [--state <state>] <volume> [<volume> ...]
```

Explicitly update the state of a volume.

### Positional arguments

**<volume>** Name or ID of the volume to modify.

### Optional arguments

**-state <state>** Indicate which state to assign the volume. Options include available, error, creating, deleting, error\_deleting. If no state is provided, available will be used.

## cinder service-disable command

```
usage: cinder service-disable <hostname> <binary>
```



Disable the service.

### Positional arguments

**<hostname>**     Name of host.

**<binary>**        Service binary.

## cinder service-enable command

```
usage: cinder service-enable <hostname> <binary>
```

Enable the service.

### Positional arguments

**<hostname>**     Name of host.

**<binary>**        Service binary.

## cinder service-list command

```
usage: cinder service-list [--host <hostname>] [--binary <binary>]
```

List all the services. Filter by host & service binary.

### Optional arguments

**-host <hostname>**     Name of host.

**-binary <binary>**      Service binary.

## cinder show command

```
usage: cinder show <volume>
```

Show details about a volume.

### Positional arguments

**<volume>**    Name or ID of the volume.

## cinder snapshot-create command

```
usage: cinder snapshot-create [--force <True|False>]
                               [--display-name <display-name>]
                               [--display-description <display-description>]
                               <volume>
```

Add a new snapshot.

### Positional arguments

**<volume>**    Name or ID of the volume to snapshot

### Optional arguments

|                                                         |                                                                                                              |
|---------------------------------------------------------|--------------------------------------------------------------------------------------------------------------|
| <b>-force &lt;True False&gt;</b>                        | Optional flag to indicate whether to snapshot a volume even if it's attached to an instance. (Default=False) |
| <b>-display-name &lt;display-name&gt;</b>               | Optional snapshot name. (Default=None)                                                                       |
| <b>-display-description &lt;display-description&gt;</b> | Optional snapshot description. (Default=None)                                                                |

## cinder snapshot-delete command

```
usage: cinder snapshot-delete <snapshot>
```

Remove a snapshot.

### Positional arguments

**<snapshot>** Name or ID of the snapshot to delete.

## cinder snapshot-list command

```
usage: cinder snapshot-list [--all-tenants [<0|1>]]  
                             [--display-name <display-name>]  
                             [--status <status>] [--volume-id <volume-id>]
```

List all the snapshots.

### Optional arguments

|                                            |                                                    |
|--------------------------------------------|----------------------------------------------------|
| <b>--all-tenants [&lt;0 1&gt;]</b>         | Display information from all tenants (Admin only). |
| <b>--display-name &lt;display-name&gt;</b> | Filter results by display-name                     |
| <b>--status &lt;status&gt;</b>             | Filter results by status                           |
| <b>--volume-id &lt;volume-id&gt;</b>       | Filter results by volume-id                        |

## cinder snapshot-metadata command

```
usage: cinder snapshot-metadata <snapshot> <action> <key=value>  
                                [<key=value> ...]
```

Set or Delete metadata of a snapshot.

### Positional arguments

- <snapshot>** ID of the snapshot to update metadata on.
- <action>** Actions: 'set' or 'unset'
- <key=value>** Metadata to set/unset (only key is necessary on unset)

## cinder snapshot-metadata-show command

```
usage: cinder snapshot-metadata-show <snapshot>
```

Show metadata of given snapshot.

### Positional arguments

- <snapshot>** ID of snapshot

## cinder snapshot-metadata-update-all command

```
usage: cinder snapshot-metadata-update-all <snapshot> <key=value>  
[<key=value> ...]
```

Update all metadata of a snapshot.

### Positional arguments

- <snapshot>** ID of the snapshot to update metadata on.

**<key=value>**      Metadata entry/entries to update.

## cinder snapshot-rename command

```
usage: cinder snapshot-rename [--display-description <display-description>]
                               <snapshot> [<display-name>]
```

Rename a snapshot.

### Positional arguments

**<snapshot>**              Name or ID of the snapshot.

**<display-name>**        New display-name for the snapshot.

### Optional arguments

**--display-description <display-description>**      Optional snapshot description. (Default=None)

## cinder snapshot-reset-state command

```
usage: cinder snapshot-reset-state [--state <state>]
                                     <snapshot> [<snapshot> ...]
```

Explicitly update the state of a snapshot.

### Positional arguments

**<snapshot>**              Name or ID of the snapshot to modify.

### Optional arguments

**-state <state>**      Indicate which state to assign the snapshot. Options include available, error, creating, deleting, error\_deleting. If no state is provided, available will be used.

## cinder snapshot-show command

```
usage: cinder snapshot-show <snapshot>
```

Show details about a snapshot.

### Positional arguments

**<snapshot>**      Name or ID of the snapshot.

## cinder transfer-accept command

```
usage: cinder transfer-accept <transfer> <auth_key>
```

Accepts a volume transfer.

### Positional arguments

**<transfer>**      ID of the transfer to accept.

**<auth\_key>**      Auth key of the transfer to accept.

## cinder transfer-create command

```
usage: cinder transfer-create [--display-name <display-name>] <volume>
```

Creates a volume transfer.

### Positional arguments

**<volume>**    Name or ID of the volume to transfer.

### Optional arguments

**-display-name <display-name>**    Optional transfer name. (Default=None)

## cinder transfer-delete command

```
usage: cinder transfer-delete <transfer>
```

Undo a transfer.

### Positional arguments

**<transfer>**    Name or ID of the transfer to delete.

## cinder transfer-list command

```
usage: cinder transfer-list
```

List all the transfers.

## cinder transfer-show command

```
usage: cinder transfer-show <transfer>
```

Show details about a transfer.

### Positional arguments

**<transfer>**      Name or ID of the transfer to accept.

## cinder type-create command

```
usage: cinder type-create <name>
```

Create a new volume type.

### Positional arguments

**<name>**      Name of the new volume type

## cinder type-delete command

```
usage: cinder type-delete <id>
```

Delete a specific volume type.

### Positional arguments

**<id>**      Unique ID of the volume type to delete

## cinder type-key command

```
usage: cinder type-key <vtype> <action> [<key=value> [<key=value> ...]]
```



Set or unset extra\_spec for a volume type.

### Positional arguments

- <vtype>**            Name or ID of the volume type
- <action>**           Actions: 'set' or 'unset'
- <key=value>**       Extra\_specs to set/unset (only key is necessary on unset)

## cinder type-list command

```
usage: cinder type-list
```

Print a list of available 'volume types'.

## cinder upload-to-image command

```
usage: cinder upload-to-image [--force <True|False>]
                               [--container-format <container-format>]
                               [--disk-format <disk-format>]
                               <volume> <image-name>
```

Upload volume to image service as image.

### Positional arguments

- <volume>**            Name or ID of the volume to upload to an image
- <image-name>**       Name for created image

### Optional arguments

|                                                   |                                                                                                            |
|---------------------------------------------------|------------------------------------------------------------------------------------------------------------|
| <b>-force &lt;True False&gt;</b>                  | Optional flag to indicate whether to upload a volume even if it's attached to an instance. (Default=False) |
| <b>-container-format &lt;container-format&gt;</b> | Optional type for container format (Default=bare)                                                          |
| <b>-disk-format &lt;disk-format&gt;</b>           | Optional type for disk format (Default=raw)                                                                |

## Block Storage Manage Volumes

A volume is a detachable block storage device, similar to a USB hard drive. You can attach a volume to only one instance. To create and manage volumes, you use a combination of nova and cinder client commands.

This example creates a my-new-volume volume based on an image.

### Migrate a volume

As an administrator, you can migrate a volume with its data from one location to another in a manner that is transparent to users and workloads. You can migrate only detached volumes with no snapshots.

Possible use cases for data migration:

- Bring down a physical storage device for maintenance without disrupting workloads.
- Modify the properties of a volume.
- Free up space in a thinly-provisioned back end.

Migrate a volume, as follows:

```
$ cinder migrate volumeID destinationHost --force-host-copy=True|False
```

Where `--force-host-copy=True` forces the generic host-based migration mechanism and bypasses any driver optimizations.



### Note

If the volume is in use or has snapshots, the specified host destination cannot accept the volume.

If the user is not an administrator, the migration fails.

## Create a volume

1. List images, and note the ID of the image to use for your volume:

```
$ nova image-list
```

| ID                                   | Name                            | Status | Server                               |
|--------------------------------------|---------------------------------|--------|--------------------------------------|
| 397e713c-b95b-4186-ad46-6126863ea0a9 | cirros-0.3.2-x86_64-uec         | ACTIVE |                                      |
| df430cc2-3406-4061-b635-a51c16e488ac | cirros-0.3.2-x86_64-uec-kernel  | ACTIVE |                                      |
| 3cf852bd-2332-48f4-9ae4-7d926d50945e | cirros-0.3.2-x86_64-uec-ramdisk | ACTIVE |                                      |
| 7e5142af-1253-4634-bcc6-89482c5f2e8a | myCirrosImage                   | ACTIVE | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |
| 89bcd424-9d15-4723-95ec-61540e8a1979 | mynsnapshot                     | ACTIVE | f51ebd07-c33d-4951-8722-1df6aa8afaa4 |

2. List the availability zones, and note the ID of the availability zone in which to create your volume:

```
$ nova availability-zone-list
```

```
+-----+-----+
```

| Name               | Status                                  |
|--------------------|-----------------------------------------|
| internal           | available                               |
| - devstack         |                                         |
| - nova-conductor   | enabled :- ) 2013-07-25T16:50:44.000000 |
| - nova-consoleauth | enabled :- ) 2013-07-25T16:50:44.000000 |
| - nova-scheduler   | enabled :- ) 2013-07-25T16:50:44.000000 |
| - nova-cert        | enabled :- ) 2013-07-25T16:50:44.000000 |
| - nova-network     | enabled :- ) 2013-07-25T16:50:44.000000 |
| nova               | available                               |
| - devstack         |                                         |
| - nova-compute     | enabled :- ) 2013-07-25T16:50:39.000000 |

3. Create a volume with 8 GB of space. Specify the availability zone and image:

```
$ cinder create 8 --display-name my-new-volume --image-id 397e713c-b95b-4186-ad46-6126863ea0a9 --availability-zone nova
```

| Property            | Value                                |
|---------------------|--------------------------------------|
| attachments         | []                                   |
| availability_zone   | nova                                 |
| bootable            | false                                |
| created_at          | 2013-07-25T17:02:12.472269           |
| display_description | None                                 |
| display_name        | my-new-volume                        |
| id                  | 573e024d-5235-49ce-8332-be1576d323f8 |
| image_id            | 397e713c-b95b-4186-ad46-6126863ea0a9 |
| metadata            | {}                                   |
| size                | 8                                    |
| snapshot_id         | None                                 |
| source_volid        | None                                 |
| status              | creating                             |
| volume_type         | None                                 |

4. To verify that your volume was created successfully, list the available volumes:

```
$ cinder list
```

| ID | Status | Display Name | Size | Volume Type | Bootable | Attached to |
|----|--------|--------------|------|-------------|----------|-------------|
|----|--------|--------------|------|-------------|----------|-------------|

|                                       |           |                 |   |      |      |  |
|---------------------------------------|-----------|-----------------|---|------|------|--|
| 573e024d-5235-49ce-8332-be1576d323f8  | available | my-new-volume   | 8 | None | true |  |
| bd7cf584-45de-44e3-bf7f-f7b50bf235e3  | available | my-bootable-vol | 8 | None | true |  |
| +-----+-----+-----+-----+-----+-----+ |           |                 |   |      |      |  |

If your volume was created successfully, its status is available. If its status is error, you might have exceeded your quota.

## Attach a volume to an instance

1. Attach your volume to a server:

```
$ nova volume-attach 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 573e024d-5235-49ce-8332-be1576d323f8 /dev/vdb
```

| Property | Value                                |
|----------|--------------------------------------|
| device   | /dev/vdb                             |
| serverId | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |
| id       | 573e024d-5235-49ce-8332-be1576d323f8 |
| volumeId | 573e024d-5235-49ce-8332-be1576d323f8 |

Note the ID of your volume.

2. Show information for your volume:

```
$ cinder show 573e024d-5235-49ce-8332-be1576d323f8
```

| Property    | Value                                                                                                                                                                                |
|-------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| attachments | [{'device': u'/dev/vdb', 'server_id': u'84c6e57d-a6b1-44b6-81eb-fcb36afd31b5', 'id': u'573e024d-5235-49ce-8332-be1576d323f8', 'volume_id': u'573e024d-5235-49ce-8332-be1576d323f8'}] |

```

|      availability_zone      |
|      nova                   |
|      |                       |
|      bootable               |
|      true                   |
|      |                       |
|      created_at             |
|      2013-07-25T17:02:12.000000 |
|      |                       |
|      display_description    |
|      None                   |
|      |                       |
|      display_name           |
|      my-new-volume          |
|      |                       |
|      id                     |
|      573e024d-5235-49ce-8332-be1576d323f8 |
|      |                       |
|      metadata               |
|      {}                     |
|      |                       |
|      os-vol-host-attr:host  |
|      devstack               |
|      |                       |
|      os-vol-tenant-attr:tenant_id |
|      66265572db174a7aa66eba661f58eb9e |
|      |                       |
|      size                   |
|      8                      |
|      |                       |
|      snapshot_id            |
|      None                   |
|      |                       |
|      source_volid           |
|      None                   |
|      |                       |
|      status                 |
|      in-use                 |
|      |                       |
|      volume_image_metadata  | {u'kernel_id': u'df430cc2-3406-4061-b635-a51c16e488ac', u'image_id': u'397e713c-
b95b-4186-ad46-6126863ea0a9', u'ramdisk_id': u'3cf852bd-2332-48f4-9ae4-7d926d50945e', u'image_name': u'cirros-0.3.2-
x86_64-uec'} |
|      volume_type            |
|      None                   |
|      |                       |

```

```
+-----+
+-----+
+
```

The output shows that the volume is attached to the server with ID 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5, is in the nova availability zone, and is bootable.

## Resize a volume

1. To resize your volume, you must first detach it from the server.

To detach the volume from your server, pass the server ID and volume ID to the command:

```
$ nova volume-detach 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 573e024d-5235-49ce-8332-be1576d323f8
```

The **volume-detach** command does not return any output.

2. List volumes:

```
$ cinder list
```

| ID                                   | Status    | Display Name    | Size | Volume Type | Bootable | Attached to |
|--------------------------------------|-----------|-----------------|------|-------------|----------|-------------|
| 573e024d-5235-49ce-8332-be1576d323f8 | available | my-new-volume   | 8    | None        | true     |             |
| bd7cf584-45de-44e3-bf7f-f7b50bf235e3 | available | my-bootable-vol | 8    | None        | true     |             |

Note that the volume is now available.

3. Resize the volume by passing the volume ID and the new size (a value greater than the old one) as parameters:

```
$ cinder extend 573e024d-5235-49ce-8332-be1576d323f8 10
```

The **extend** command does not return any output.

## Delete a volume

1. To delete your volume, you must first detach it from the server.

To detach the volume from your server and check for the list of existing volumes, see steps 1 and 2 mentioned in [the section called "Resize a volume" \[209\]](#).

2. Delete the volume:

```
$ cinder delete my-new-volume
```

The delete command does not return any output.

3. List the volumes again, and note that the status of your volume is deleting:

```
$ cinder list
```

| ID                                   | Status    | Display Name    | Size | Volume Type | Bootable | Attached to |
|--------------------------------------|-----------|-----------------|------|-------------|----------|-------------|
| 573e024d-5235-49ce-8332-be1576d323f8 | deleting  | my-new-volume   | 8    | None        | true     |             |
| bd7cf584-45de-44e3-bf7f-f7b50bf235e3 | available | my-bootable-vol | 8    | None        | true     |             |

When the volume is fully deleted, it disappears from the list of volumes:

```
$ cinder list
```

| ID                                   | Status    | Display Name    | Size | Volume Type | Bootable | Attached to |
|--------------------------------------|-----------|-----------------|------|-------------|----------|-------------|
| bd7cf584-45de-44e3-bf7f-f7b50bf235e3 | available | my-bootable-vol | 8    | None        | true     |             |



## Transfer a volume

You can transfer a volume from one owner to another by using the **cinder transfer\*** commands. The volume donor, or original owner, creates a transfer request and sends the created transfer ID and authorization key to the volume recipient. The volume recipient, or new owner, accepts the transfer by using the ID and key.



### Note

The procedure for volume transfer is intended for tenants (both the volume donor and recipient) within the same cloud.

Use cases include:

- Create a custom bootable volume or a volume with a large data set and transfer it to the end customer.
- For bulk import of data to the cloud, the data ingress system creates a new Block Storage volume, copies data from the physical device, and transfers device ownership to the end user.

## Create a volume transfer request

1. While logged in as the volume donor, list available volumes:

```
$ cinder list
```

| ID                                   |             | Status    | Display Name | Size | Volume Type |
|--------------------------------------|-------------|-----------|--------------|------|-------------|
| Bootable                             | Attached to |           |              |      |             |
| 72bfce9f-cacf-477a-a092-bf57a7712165 |             | error     | None         | 1    | None        |
| false                                |             |           |              |      |             |
| a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f |             | available | None         | 1    | None        |
| false                                |             |           |              |      |             |

```
+-----+-----+-----+-----+
+-----+-----+
```

2. As the volume donor, request a volume transfer authorization code for a specific volume:

```
$ cinder transfer-create volumeID
```

The volume must be in an 'available' state or the request will be denied. If the transfer request is valid in the database (that is, it has not expired or been deleted), the volume is placed in an awaiting transfer state. For example:

```
$ cinder transfer-create a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f
```

| Property   | Value                                |
|------------|--------------------------------------|
| auth_key   | b2c8e585cbc68a80                     |
| created_at | 2013-10-14T15:20:10.121458           |
| id         | 6e4e9aa4-bed5-4f94-8f76-df43232f44dc |
| name       | None                                 |
| volume_id  | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f |



### Note

Optionally, you can specify a name for the transfer by using the `--display-name displayName` parameter.

3. Send the volume transfer ID and authorization key to the new owner (for example, by email).
4. View pending transfers:

```
$ cinder transfer-list
```

```
+-----+-----+-----+-----+
+-----+-----+
```

| ID                                   | VolumeID                             | Name |
|--------------------------------------|--------------------------------------|------|
| 6e4e9aa4-bed5-4f94-8f76-df43232f44dc | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f | None |

- After the volume recipient, or new owner, accepts the transfer, you can see that the transfer is no longer available:

```
$ cinder transfer-list
```

| ID | Volume ID | Name |
|----|-----------|------|
|    |           |      |

## Accept a volume transfer request

- As the volume recipient, you must first obtain the transfer ID and authorization key from the original owner.
- Display the transfer request details using the ID:

```
$ cinder transfer-show transferID
```

For example:

```
$ cinder transfer-show 6e4e9aa4-bed5-4f94-8f76-df43232f44dc
```

| Property   | Value                                |
|------------|--------------------------------------|
| created_at | 2013-10-14T15:20:10.000000           |
| id         | 6e4e9aa4-bed5-4f94-8f76-df43232f44dc |
| name       | None                                 |
| volume_id  | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f |

- Accept the request:

```
$ cinder transfer-accept transferID authKey
```

For example:

```
$ cinder transfer-accept 6e4e9aa4-bed5-4f94-8f76-df43232f44dc b2c8e585cbc68a80
```

| Property  | Value                                |
|-----------|--------------------------------------|
| id        | 6e4e9aa4-bed5-4f94-8f76-df43232f44dc |
| name      | None                                 |
| volume_id | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f |



### Note

If you do not have a sufficient quota for the transfer, the transfer is refused.

## Delete a volume transfer

- List available volumes and their statuses:

```
$ cinder list
```

| Type                                 | Bootable | ID    | Attached to | Status            | Display Name | Size | Volume |
|--------------------------------------|----------|-------|-------------|-------------------|--------------|------|--------|
| 72bfce9f-cacf-477a-a092-bf57a7712165 | None     | false |             | error             | None         | 1    |        |
| a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f | None     | false |             | awaiting-transfer | None         | 1    |        |

```
+-----+-----+-----+-----+
+-----+-----+-----+

```

- Find the matching transfer ID:

```
$ cinder transfer-list
```

```
+-----+-----+-----+
|          ID          |          VolumeID          | Name |
+-----+-----+-----+
| a6da6888-7cdf-4291-9c08-8c1f22426b8a | a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f | None |
+-----+-----+-----+
```

- Delete the volume:

```
$ cinder transfer-delete transferID
```

For example:

```
$ cinder transfer-delete a6da6888-7cdf-4291-9c08-8c1f22426b8a
```

- The transfer list is now empty and the volume is again available for transfer:

```
$ cinder transfer-list
```

```
+---+-----+-----+
| ID | Volume ID | Name |
+---+-----+-----+
+---+-----+-----+
```

```
$ cinder list
```

```
+-----+-----+-----+-----+-----+
+-----+-----+
|          ID          | Status | Display Name | Size | Volume Type |
+-----+-----+
| Bootable | Attached to |
```

|                                      |           |  |      |  |      |  |
|--------------------------------------|-----------|--|------|--|------|--|
| +-----+-----+-----+-----+-----+      |           |  |      |  |      |  |
| +-----+-----+                        |           |  |      |  |      |  |
| 72bfce9f-cacf-477a-a092-bf57a7712165 | error     |  | None |  | 1    |  |
| false                                |           |  |      |  | None |  |
| a1cdace0-08e4-4dc7-b9dc-457e9bcfe25f | available |  | None |  | 1    |  |
| false                                |           |  |      |  | None |  |
| +-----+-----+-----+-----+-----+      |           |  |      |  |      |  |
| +-----+-----+                        |           |  |      |  |      |  |

## Set a volume to read-only access

To give multiple users shared, secure access to the same data, you can set a volume to read-only access.

Run this command to set a volume to read-only access:

```
$ cinder readonly-mode-update VOLUME BOOLEAN
```

Where *VOLUME* is the ID of the target volume and *BOOLEAN* is a flag that enables read-only or read/write access to the volume.

Valid values for *BOOLEAN* are:

- `true`. Sets the read-only flag in the volume. When you attach the volume to an instance, the instance checks for this flag to determine whether to restrict volume access to read-only.
- `false`. Sets the volume to read/write access.

## Compute CLI Commands

The **nova** client is the command-line interface (CLI) for the OpenStack Compute API and its extensions. This chapter documents **nova** version 2.17.0.

For help on a specific **nova** command, enter:

```
$ nova help COMMAND
```

## nova usage

```
usage: nova [--version] [--debug] [--os-cache] [--timings]
           [--timeout <seconds>] [--os-auth-token OS_AUTH_TOKEN]
           [--os-username <auth-user-name>] [--os-password <auth-password>]
           [--os-tenant-name <auth-tenant-name>]
           [--os-tenant-id <auth-tenant-id>] [--os-auth-url <auth-url>]
           [--os-region-name <region-name>] [--os-auth-system <auth-system>]
           [--service-type <service-type>] [--service-name <service-name>]
           [--volume-service-name <volume-service-name>]
           [--endpoint-type <endpoint-type>]
           [--os-compute-api-version <compute-api-ver>]
           [--os-cacert <ca-certificate>] [--insecure]
           [--bypass-url <bypass-url>]
           <subcommand> ...
```

## Subcommands

|                        |                                                        |
|------------------------|--------------------------------------------------------|
| <b>absolute-limits</b> | Print a list of absolute limits for a user             |
| <b>add-fixed-ip</b>    | Add new IP address on a network to server.             |
| <b>add-floating-ip</b> | <i>DEPRECATED</i> , use floating-ip-associate instead. |
| <b>add-secgroup</b>    | Add a Security Group to a server.                      |
| <b>agent-create</b>    | Create new agent build.                                |
| <b>agent-delete</b>    | Delete existing agent build.                           |
| <b>agent-list</b>      | List all builds.                                       |
| <b>agent-modify</b>    | Modify existing agent build.                           |

---

|                               |                                                               |
|-------------------------------|---------------------------------------------------------------|
| <b>aggregate-add-host</b>     | Add the host to the specified aggregate.                      |
| <b>aggregate-create</b>       | Create a new aggregate with the specified details.            |
| <b>aggregate-delete</b>       | Delete the aggregate.                                         |
| <b>aggregate-details</b>      | Show details of the specified aggregate.                      |
| <b>aggregate-list</b>         | Print a list of all aggregates.                               |
| <b>aggregate-remove-host</b>  | Remove the specified host from the specified aggregate.       |
| <b>aggregate-set-metadata</b> | Update the metadata associated with the aggregate.            |
| <b>aggregate-update</b>       | Update the aggregate's name and optionally availability zone. |
| <b>availability-zone-list</b> | List all the availability zones.                              |
| <b>backup</b>                 | Backup a server by creating a 'backup' type snapshot.         |
| <b>boot</b>                   | Boot a new server.                                            |
| <b>clear-password</b>         | Clear password for a server.                                  |
| <b>cloudpipe-configure</b>    | Update the VPN IP/port of a cloudpipe instance.               |
| <b>cloudpipe-create</b>       | Create a cloudpipe instance for the given project.            |
| <b>cloudpipe-list</b>         | Print a list of all cloudpipe instances.                      |
| <b>console-log</b>            | Get console log output of a server.                           |
| <b>credentials</b>            | Show user credentials returned from auth.                     |
| <b>delete</b>                 | Immediately shut down and delete specified server(s).         |



---

|                                  |                                                                      |
|----------------------------------|----------------------------------------------------------------------|
| <b>diagnostics</b>               | Retrieve server diagnostics.                                         |
| <b>dns-create</b>                | Create a DNS entry for domain, name and ip.                          |
| <b>dns-create-private-domain</b> | Create the specified DNS domain.                                     |
| <b>dns-create-public-domain</b>  | Create the specified DNS domain.                                     |
| <b>dns-delete</b>                | Delete the specified DNS entry.                                      |
| <b>dns-delete-domain</b>         | Delete the specified DNS domain.                                     |
| <b>dns-domains</b>               | Print a list of available dns domains.                               |
| <b>dns-list</b>                  | List current DNS entries for domain and ip or domain and name.       |
| <b>endpoints</b>                 | Discover endpoints that get returned from the authenticate services. |
| <b>evacuate</b>                  | Evacuate server from failed host to specified one.                   |
| <b>fixed-ip-get</b>              | Retrieve info on a fixed ip.                                         |
| <b>fixed-ip-reserve</b>          | Reserve a fixed IP.                                                  |
| <b>fixed-ip-unreserve</b>        | Unreserve a fixed IP.                                                |
| <b>flavor-access-add</b>         | Add flavor access for the given tenant.                              |
| <b>flavor-access-list</b>        | Print access information about the given flavor.                     |
| <b>flavor-access-remove</b>      | Remove flavor access for the given tenant.                           |
| <b>flavor-create</b>             | Create a new flavor                                                  |
| <b>flavor-delete</b>             | Delete a specific flavor                                             |

---

|                                 |                                                         |
|---------------------------------|---------------------------------------------------------|
| <b>flavor-key</b>               | Set or unset extra_spec for a flavor.                   |
| <b>flavor-list</b>              | Print a list of available 'flavors' (sizes of servers). |
| <b>flavor-show</b>              | Show details about the given flavor.                    |
| <b>floating-ip-associate</b>    | Associate a floating IP address to a server.            |
| <b>floating-ip-bulk-create</b>  | Bulk create floating ips by range.                      |
| <b>floating-ip-bulk-delete</b>  | Bulk delete floating ips by range.                      |
| <b>floating-ip-bulk-list</b>    | List all floating ips.                                  |
| <b>floating-ip-create</b>       | Allocate a floating IP for the current tenant.          |
| <b>floating-ip-delete</b>       | De-allocate a floating IP.                              |
| <b>floating-ip-disassociate</b> | Disassociate a floating IP address from a server.       |
| <b>floating-ip-list</b>         | List floating ips for this tenant.                      |
| <b>floating-ip-pool-list</b>    | List all floating ip pools.                             |
| <b>get-password</b>             | Get password for a server.                              |
| <b>get-rdp-console</b>          | Get a rdp console to a server.                          |
| <b>get-spice-console</b>        | Get a spice console to a server.                        |
| <b>get-vnc-console</b>          | Get a vnc console to a server.                          |
| <b>host-action</b>              | Perform a power action on a host.                       |
| <b>host-describe</b>            | Describe a specific host.                               |

---

|                           |                                                              |
|---------------------------|--------------------------------------------------------------|
| <b>host-list</b>          | List all hosts by service.                                   |
| <b>host-update</b>        | Update host settings.                                        |
| <b>hypervisor-list</b>    | List hypervisors.                                            |
| <b>hypervisor-servers</b> | List servers belonging to specific hypervisors.              |
| <b>hypervisor-show</b>    | Display the details of the specified hypervisor.             |
| <b>hypervisor-stats</b>   | Get hypervisor statistics over all compute nodes.            |
| <b>hypervisor-uptime</b>  | Display the uptime of the specified hypervisor.              |
| <b>image-create</b>       | Create a new image by taking a snapshot of a running server. |
| <b>image-delete</b>       | Delete specified image(s).                                   |
| <b>image-list</b>         | Print a list of available images to boot from.               |
| <b>image-meta</b>         | Set or Delete metadata on an image.                          |
| <b>image-show</b>         | Show details about the given image.                          |
| <b>interface-attach</b>   | Attach a network interface to a server.                      |
| <b>interface-detach</b>   | Detach a network interface from a server.                    |
| <b>interface-list</b>     | List interfaces attached to a server.                        |
| <b>keypair-add</b>        | Create a new key pair for use with servers.                  |
| <b>keypair-delete</b>     | Delete keypair given by its name.                            |
| <b>keypair-list</b>       | Print a list of keypairs for a user                          |

---

|                                  |                                                                            |
|----------------------------------|----------------------------------------------------------------------------|
| <b>keypair-show</b>              | Show details about the given keypair.                                      |
| <b>list</b>                      | List active servers.                                                       |
| <b>list-secgroup</b>             | List Security Group(s) of a server.                                        |
| <b>live-migration</b>            | Migrate running server to a new machine.                                   |
| <b>lock</b>                      | Lock a server.                                                             |
| <b>meta</b>                      | Set or Delete metadata on a server.                                        |
| <b>migrate</b>                   | Migrate a server. The new host will be selected by the scheduler.          |
| <b>network-associate-host</b>    | Associate host with network.                                               |
| <b>network-associate-project</b> | Associate project with network.                                            |
| <b>network-create</b>            | Create a network.                                                          |
| <b>network-disassociate</b>      | Disassociate host and/or project from the given network.                   |
| <b>network-list</b>              | Print a list of available networks.                                        |
| <b>network-show</b>              | Show details about the given network.                                      |
| <b>pause</b>                     | Pause a server.                                                            |
| <b>quota-class-show</b>          | List the quotas for a quota class.                                         |
| <b>quota-class-update</b>        | Update the quotas for a quota class.                                       |
| <b>quota-defaults</b>            | List the default quotas for a tenant.                                      |
| <b>quota-delete</b>              | Delete quota for a tenant/user so their quota will Revert back to default. |

---

|                           |                                                           |
|---------------------------|-----------------------------------------------------------|
| <b>quota-show</b>         | List the quotas for a tenant/user.                        |
| <b>quota-update</b>       | Update the quotas for a tenant/user.                      |
| <b>rate-limits</b>        | Print a list of rate limits for a user                    |
| <b>reboot</b>             | Reboot a server.                                          |
| <b>rebuild</b>            | Shutdown, re-image, and re-boot a server.                 |
| <b>refresh-network</b>    | Refresh server network information.                       |
| <b>remove-fixed-ip</b>    | Remove an IP address from a server.                       |
| <b>remove-floating-ip</b> | <i>DEPRECATED</i> , use floating-ip-disassociate instead. |
| <b>remove-secgroup</b>    | Remove a Security Group from a server.                    |
| <b>rename</b>             | Rename a server.                                          |
| <b>rescue</b>             | Rescue a server.                                          |
| <b>reset-network</b>      | Reset network of a server.                                |
| <b>reset-state</b>        | Reset the state of a server.                              |
| <b>resize</b>             | Resize a server.                                          |
| <b>resize-confirm</b>     | Confirm a previous resize.                                |
| <b>resize-revert</b>      | Revert a previous resize (and return to the previous VM). |
| <b>resume</b>             | Resume a server.                                          |
| <b>root-password</b>      | Change the root password for a server.                    |

---

|                                   |                                                               |
|-----------------------------------|---------------------------------------------------------------|
| <b>scrub</b>                      | Delete data associated with the project.                      |
| <b>secgroup-add-group-rule</b>    | Add a source group rule to a security group.                  |
| <b>secgroup-add-rule</b>          | Add a rule to a security group.                               |
| <b>secgroup-create</b>            | Create a security group.                                      |
| <b>secgroup-delete</b>            | Delete a security group.                                      |
| <b>secgroup-delete-group-rule</b> | Delete a source group rule from a security group.             |
| <b>secgroup-delete-rule</b>       | Delete a rule from a security group.                          |
| <b>secgroup-list</b>              | List security groups for the current tenant.                  |
| <b>secgroup-list-rules</b>        | List rules for a security group.                              |
| <b>secgroup-update</b>            | Update a security group.                                      |
| <b>service-disable</b>            | Disable the service.                                          |
| <b>service-enable</b>             | Enable the service.                                           |
| <b>service-list</b>               | Show a list of all running services. Filter by host & binary. |
| <b>shelve</b>                     | Shelve a server.                                              |
| <b>shelve-offload</b>             | Remove a shelved server from the compute node.                |
| <b>show</b>                       | Show details about the given server.                          |
| <b>ssh</b>                        | SSH into a server.                                            |
| <b>start</b>                      | Start a server.                                               |

---

|                               |                                      |
|-------------------------------|--------------------------------------|
| <b>stop</b>                   | Stop a server.                       |
| <b>suspend</b>                | Suspend a server.                    |
| <b>unlock</b>                 | Unlock a server.                     |
| <b>unpause</b>                | Unpause a server.                    |
| <b>unrescue</b>               | Unrescue a server.                   |
| <b>unshelve</b>               | Unshelve a server.                   |
| <b>usage</b>                  | Show usage data for a single tenant. |
| <b>usage-list</b>             | List usage data for all tenants.     |
| <b>volume-attach</b>          | Attach a volume to a server.         |
| <b>volume-create</b>          | Add a new volume.                    |
| <b>volume-delete</b>          | Remove volume(s).                    |
| <b>volume-detach</b>          | Detach a volume from a server.       |
| <b>volume-list</b>            | List all the volumes.                |
| <b>volume-show</b>            | Show details about a volume.         |
| <b>volume-snapshot-create</b> | Add a new snapshot.                  |
| <b>volume-snapshot-delete</b> | Remove a snapshot.                   |
| <b>volume-snapshot-list</b>   | List all the snapshots.              |
| <b>volume-snapshot-show</b>   | Show details about a snapshot.       |

---

|                                   |                                                                                                                          |
|-----------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>volume-type-create</b>         | Create a new volume type.                                                                                                |
| <b>volume-type-delete</b>         | Delete a specific flavor                                                                                                 |
| <b>volume-type-list</b>           | Print a list of available 'volume types'.                                                                                |
| <b>volume-update</b>              | Update volume attachment.                                                                                                |
| <b>x509-create-cert</b>           | Create x509 cert for a user in tenant.                                                                                   |
| <b>x509-get-root-cert</b>         | Fetch the x509 root cert.                                                                                                |
| <b>bash-completion</b>            | Prints all of the commands and options to stdout so that the nova.bash_completion script doesn't have to hard code them. |
| <b>help</b>                       | Display help about this program or one of its subcommands.                                                               |
| <b>force-delete</b>               | Force delete a server.                                                                                                   |
| <b>restore</b>                    | Restore a soft-deleted server.                                                                                           |
| <b>net</b>                        | Show a network                                                                                                           |
| <b>net-create</b>                 | Create a network                                                                                                         |
| <b>net-delete</b>                 | Delete a network                                                                                                         |
| <b>net-list</b>                   | List networks                                                                                                            |
| <b>baremetal-interface-add</b>    | Add a network interface to a baremetal node.                                                                             |
| <b>baremetal-interface-list</b>   | List network interfaces associated with a baremetal node.                                                                |
| <b>baremetal-interface-remove</b> | Remove a network interface from a baremetal node.                                                                        |



|                              |                                                                       |
|------------------------------|-----------------------------------------------------------------------|
| <b>baremetal-node-create</b> | Create a baremetal node.                                              |
| <b>baremetal-node-delete</b> | Remove a baremetal node and any associated interfaces.                |
| <b>baremetal-node-list</b>   | Print list of available baremetal nodes.                              |
| <b>baremetal-node-show</b>   | Show information about a baremetal node.                              |
| <b>host-evacuate</b>         | Evacuate all instances from failed host to specified one.             |
| <b>instance-action</b>       | Show an action.                                                       |
| <b>instance-action-list</b>  | List actions on a server.                                             |
| <b>migration-list</b>        | Print a list of migrations.                                           |
| <b>host-servers-migrate</b>  | Migrate all instances of the specified host to other available hosts. |
| <b>cell-capacities</b>       | Get cell capacities for all cells or a given cell.                    |
| <b>cell-show</b>             | Show details of a given cell.                                         |
| <b>host-meta</b>             | Set or Delete metadata on all instances of a host.                    |
| <b>list-extensions</b>       | List all the os-api extensions that are available.                    |

## **nova optional arguments**

|                  |                                                                          |
|------------------|--------------------------------------------------------------------------|
| <b>-version</b>  | show program's version number and exit                                   |
| <b>-debug</b>    | Print debugging output                                                   |
| <b>-os-cache</b> | Use the auth token cache. Defaults to False if env[OS_CACHE] is not set. |
| <b>-timings</b>  | Print call timing info                                                   |

---

|                                                            |                                                            |
|------------------------------------------------------------|------------------------------------------------------------|
| <b>-timeout &lt;seconds&gt;</b>                            | Set HTTP call timeout (in seconds)                         |
| <b>-os-auth-token<br/>OS_AUTH_TOKEN</b>                    | Defaults to env[OS_AUTH_TOKEN]                             |
| <b>-os-username &lt;auth-user-name&gt;</b>                 | Defaults to env[OS_USERNAME].                              |
| <b>-os-password &lt;auth-password&gt;</b>                  | Defaults to env[OS_PASSWORD].                              |
| <b>-os-tenant-name &lt;auth-tenant-name&gt;</b>            | Defaults to env[OS_TENANT_NAME].                           |
| <b>-os-tenant-id &lt;auth-tenant-id&gt;</b>                | Defaults to env[OS_TENANT_ID].                             |
| <b>-os-auth-url &lt;auth-url&gt;</b>                       | Defaults to env[OS_AUTH_URL].                              |
| <b>-os-region-name &lt;region-name&gt;</b>                 | Defaults to env[OS_REGION_NAME].                           |
| <b>-os-auth-system &lt;auth-system&gt;</b>                 | Defaults to env[OS_AUTH_SYSTEM].                           |
| <b>-service-type &lt;service-type&gt;</b>                  | Defaults to compute for most actions                       |
| <b>-service-name &lt;service-name&gt;</b>                  | Defaults to env[NOVA_SERVICE_NAME]                         |
| <b>-volume-service-name &lt;volume-service-name&gt;</b>    | Defaults to env[NOVA_VOLUME_SERVICE_NAME]                  |
| <b>-endpoint-type &lt;endpoint-type&gt;</b>                | Defaults to env[NOVA_ENDPOINT_TYPE] or publicURL.          |
| <b>-os-compute-api-version<br/>&lt;compute-api-ver&gt;</b> | Accepts 1.1 or 3, defaults to env[OS_COMPUTE_API_VERSION]. |

|                                          |                                                                                                                                                                                                     |
|------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-os-cacert &lt;ca-certificate&gt;</b> | Specify a CA bundle file to use in verifying a TLS (https) server certificate. Defaults to env[OS_CACERT]                                                                                           |
| <b>-insecure</b>                         | Explicitly allow novaclient to perform "insecure" SSL (https) requests. The server's certificate will not be verified against any certificate authorities. This option should be used with caution. |
| <b>-bypass-url &lt;bypass-url&gt;</b>    | Use this API endpoint instead of the Service Catalog                                                                                                                                                |

## nova absolute-limits command

```
usage: nova absolute-limits [--tenant [<tenant>]] [--reserved]
```

Print a list of absolute limits for a user

### Optional arguments

|                                 |                                                      |
|---------------------------------|------------------------------------------------------|
| <b>-tenant [&lt;tenant&gt;]</b> | Display information from single tenant (Admin only). |
| <b>-reserved</b>                | Include reservations count.                          |

## nova add-fixed-ip command

```
usage: nova add-fixed-ip <server> <network-id>
```

Add new IP address on a network to server.

### Positional arguments

|                           |                       |
|---------------------------|-----------------------|
| <b>&lt;server&gt;</b>     | Name or ID of server. |
| <b>&lt;network-id&gt;</b> | Network ID.           |

## nova add-secgroup command

```
usage: nova add-secgroup <server> <secgroup>
```

Add a Security Group to a server.

### Positional arguments

**<server>**            Name or ID of server.

**<secgroup>**        Name of Security Group.

## nova agent-create command

```
usage: nova agent-create <os> <architecture> <version> <url> <md5hash>
                               <hypervisor>
```

Create new agent build.

### Positional arguments

**<os>**                type of os.

**<architecture>**    type of architecture

**<version>**           version

**<url>**                url

**<md5hash>**          md5 hash

**<hypervisor>**        type of hypervisor.

## nova agent-delete command

```
usage: nova agent-delete <id>
```

Delete existing agent build.

### Positional arguments

**<id>** id of the agent-build

## nova agent-list command

```
usage: nova agent-list [--hypervisor <hypervisor>]
```

List all builds.

### Optional arguments

**-hypervisor <hypervisor>** type of hypervisor.

## nova agent-modify command

```
usage: nova agent-modify <id> <version> <url> <md5hash>
```

Modify existing agent build.

### Positional arguments

**<id>** id of the agent-build

**<version>** version

**<url>**            url  
**<md5hash>**    md5hash

## nova aggregate-add-host command

```
usage: nova aggregate-add-host <aggregate> <host>
```

Add the host to the specified aggregate.

### Positional arguments

**<aggregate>**    Name or ID of aggregate.  
**<host>**            The host to add to the aggregate.

## nova aggregate-create command

```
usage: nova aggregate-create <name> [<availability-zone>]
```

Create a new aggregate with the specified details.

### Positional arguments

**<name>**                    Name of aggregate.  
**<availability-zone>**    The availability zone of the aggregate (optional).

## nova aggregate-delete command

```
usage: nova aggregate-delete <aggregate>
```

Delete the aggregate.

### Positional arguments

**<aggregate>**      Name or ID of aggregate to delete.

## nova aggregate-details command

```
usage: nova aggregate-details <aggregate>
```

Show details of the specified aggregate.

### Positional arguments

**<aggregate>**      Name or ID of aggregate.

## nova aggregate-list command

```
usage: nova aggregate-list
```

Print a list of all aggregates.

## nova aggregate-remove-host command

```
usage: nova aggregate-remove-host <aggregate> <host>
```

Remove the specified host from the specified aggregate.

### Positional arguments

**<aggregate>**      Name or ID of aggregate.

**<host>**            The host to remove from the aggregate.

## nova aggregate-set-metadata command

```
usage: nova aggregate-set-metadata <aggregate> <key=value> [<key=value> ...]
```

Update the metadata associated with the aggregate.

### Positional arguments

**<aggregate>**        Name or ID of aggregate to update.

**<key=value>**        Metadata to add/update to aggregate

## nova aggregate-update command

```
usage: nova aggregate-update <aggregate> <name> [<availability-zone>]
```

Update the aggregate's name and optionally availability zone.

### Positional arguments

**<aggregate>**                Name or ID of aggregate to update.

**<name>**                      Name of aggregate.

**<availability-zone>**        The availability zone of the aggregate.

## nova availability-zone-list command

```
usage: nova availability-zone-list
```



List all the availability zones.

## nova backup command

```
usage: nova backup <server> <name> <backup-type> <rotation>
```

Backup a server by creating a 'backup' type snapshot.

### Positional arguments

|                            |                                                             |
|----------------------------|-------------------------------------------------------------|
| <b>&lt;server&gt;</b>      | Name or ID of server.                                       |
| <b>&lt;name&gt;</b>        | Name of the backup image.                                   |
| <b>&lt;backup-type&gt;</b> | The backup type, like "daily" or "weekly".                  |
| <b>&lt;rotation&gt;</b>    | Int parameter representing how many backups to keep around. |

## nova baremetal-interface-add command

```
usage: nova baremetal-interface-add [--datapath_id <datapath_id>]  
                                     [--port_no <port_no>]  
                                     <node> <address>
```

Add a network interface to a baremetal node.

### Positional arguments

|                        |                          |
|------------------------|--------------------------|
| <b>&lt;node&gt;</b>    | ID of node               |
| <b>&lt;address&gt;</b> | MAC address of interface |

### Optional arguments

- `-datapath_id <datapath_id>` OpenFlow Datapath ID of interface
- `-port_no <port_no>` OpenFlow port number of interface

## nova baremetal-interface-list command

```
usage: nova baremetal-interface-list <node>
```

List network interfaces associated with a baremetal node.

### Positional arguments

`<node>` ID of node

## nova baremetal-interface-remove command

```
usage: nova baremetal-interface-remove <node> <address>
```

Remove a network interface from a baremetal node.

### Positional arguments

`<node>` ID of node

`<address>` MAC address of interface

## nova baremetal-node-create command

```
usage: nova baremetal-node-create [--pm_address <pm_address>]  
                                   [--pm_user <pm_user>]
```

```
[--pm_password <pm_password>]  
[--terminal_port <terminal_port>]  
<service_host> <cpus> <memory_mb> <local_gb>  
<prov_mac_address>
```

Create a baremetal node.

### Positional arguments

|                    |                                                                  |
|--------------------|------------------------------------------------------------------|
| <service_host>     | Name of nova compute host which will control this baremetal node |
| <cpus>             | Number of CPUs in the node                                       |
| <memory_mb>        | Megabytes of RAM in the node                                     |
| <local_gb>         | Gigabytes of local storage in the node                           |
| <prov_mac_address> | MAC address to provision the node                                |

### Optional arguments

|                                |                                          |
|--------------------------------|------------------------------------------|
| -pm_address <pm_address>       | Power management IP for the node         |
| -pm_user <pm_user>             | Username for the node's power management |
| -pm_password<br><pm_password>  | Password for the node's power management |
| -terminal_port <terminal_port> | ShellInABox port?                        |

## nova baremetal-node-delete command

```
usage: nova baremetal-node-delete <node>
```

Remove a baremetal node and any associated interfaces.

### Positional arguments

**<node>** ID of the node to delete.

## nova baremetal-node-list command

```
usage: nova baremetal-node-list
```

Print list of available baremetal nodes.

## nova baremetal-node-show command

```
usage: nova baremetal-node-show <node>
```

Show information about a baremetal node.

### Positional arguments

**<node>** ID of node

## nova boot command

```
usage: nova boot [--flavor <flavor>] [--image <image>]
               [--image-with <key=value>] [--boot-volume <volume_id>]
               [--snapshot <snapshot_id>] [--num-instances <number>]
               [--meta <key=value>] [--file <dst-path=src-path>]
               [--key-name <key-name>] [--user-data <user-data>]
               [--availability-zone <availability-zone>]
               [--security-groups <security-groups>]
               [--block-device-mapping <dev-name=mapping>]
```

```
[--block-device key1=value1[,key2=value2...]]
[--swap <swap_size>]
[--ephemeral size=<size>[,format=<format>]]
[--hint <key=value>]
[--nic <net-id=net-uuid,v4-fixed-ip=ip-addr,port-id=port-uuid>]
[--config-drive <value>] [--poll]
<name>
```

Boot a new server.

### Positional arguments

**<name>** Name for the new server

### Optional arguments

|                                        |                                                                                                             |
|----------------------------------------|-------------------------------------------------------------------------------------------------------------|
| <b>-flavor &lt;flavor&gt;</b>          | Name or ID of flavor (see 'nova flavor-list').                                                              |
| <b>-image &lt;image&gt;</b>            | Name or ID of image (see 'nova image-list').                                                                |
| <b>-image-with &lt;key=value&gt;</b>   | Image metadata property (see 'nova image-show').                                                            |
| <b>-boot-volume &lt;volume_id&gt;</b>  | Volume ID to boot from.                                                                                     |
| <b>-snapshot &lt;snapshot_id&gt;</b>   | Snapshot ID to boot from (will create a volume).                                                            |
| <b>-num-instances &lt;number&gt;</b>   | boot multiple servers at a time (limited by quota).                                                         |
| <b>-meta &lt;key=value&gt;</b>         | Record arbitrary key/value metadata to /meta.js on the new server. Can be specified multiple times.         |
| <b>-file &lt;dst-path=src-path&gt;</b> | Store arbitrary files from <src-path> locally to <dst-path> on the new server. You may store up to 5 files. |

|                                                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
|---------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-key-name &lt;key-name&gt;</b>                                         | Key name of keypair that should be created earlier with the command <code>keypair-add</code>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>-user-data &lt;user-data&gt;</b>                                       | user data file to pass to be exposed by the metadata server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |
| <b>-availability-zone &lt;availability-zone&gt;</b>                       | The availability zone for server placement.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| <b>-security-groups &lt;security-groups&gt;</b>                           | Comma separated list of security group names.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
| <b>-block-device-mapping &lt;dev-name=mapping&gt;</b>                     | Block device mapping in the format <code>&lt;dev-name&gt;=&lt;id&gt;:&lt;type&gt;:&lt;size(GB)&gt;:&lt;delete-on-terminate&gt;</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
| <b>-block-device</b>                                                      | <code>key1=value1[,key2=value2...]</code> Block device mapping with the keys: <code>id=image_id, snapshot_id</code> or <code>volume_id</code> , <code>source=source</code> type (image, snapshot, volume or blank), <code>dest=destination</code> type of the block device (volume or local), <code>bus=device's bus</code> , <code>device=name</code> of the device (e.g. <code>vda</code> , <code>xda</code> , ...), <code>size=size</code> of the block device in GB, <code>format=device</code> will be formatted (e.g. <code>swap</code> , <code>ext3</code> , <code>ntfs</code> , ...), <code>bootindex=integer</code> used for ordering the boot disks, <code>type=device</code> type (e.g. <code>disk</code> , <code>cdrom</code> , ...) and <code>shutdown=shutdown</code> behaviour (either <code>preserve</code> or <code>remove</code> ). |
| <b>-swap &lt;swap_size&gt;</b>                                            | Create and attach a local swap block device of <code>&lt;swap_size&gt;</code> MB.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>-ephemeral</b>                                                         | <code>size=&lt;size&gt;[,format=&lt;format&gt;]</code> Create and attach a local ephemeral block device of <code>&lt;size&gt;</code> GB and format it to <code>&lt;format&gt;</code> .                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| <b>-hint &lt;key=value&gt;</b>                                            | Send arbitrary key/value pairs to the scheduler for custom use.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <b>-nic &lt;net-id=net-uuid,v4-fixed-ip=ip-addr,port-id=port-uuid&gt;</b> | Create a NIC on the server. Specify option multiple times to create multiple NICs. <code>net-id</code> : attach NIC to network with this UUID (required if no <code>port-id</code> ),                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |

|                                    |                                                                                                                         |
|------------------------------------|-------------------------------------------------------------------------------------------------------------------------|
|                                    | v4 -fixed-ip: IPv4 fixed address for NIC (optional), port-id: attach NIC to port with this UUID (required if no net-id) |
| <b>-config-drive &lt;value&gt;</b> | Enable config drive                                                                                                     |
| <b>-poll</b>                       | Blocks while server builds so progress can be reported.                                                                 |

## nova cell-capacities command

```
usage: nova cell-capacities [--cell <cell-name>]
```

Get cell capacities for all cells or a given cell.

### Optional arguments

**-cell <cell-name>**      Name of the cell to get the capacities.

## nova cell-show command

```
usage: nova cell-show <cell-name>
```

Show details of a given cell.

### Positional arguments

**<cell-name>**      Name of the cell.

## nova clear-password command

```
usage: nova clear-password <server>
```

Clear password for a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova cloudpipe-configure command

```
usage: nova cloudpipe-configure <ip address> <port>
```

Update the VPN IP/port of a cloudpipe instance.

### Positional arguments

**<ip address>**      New IP Address.

**<port>**            New Port.

## nova cloudpipe-create command

```
usage: nova cloudpipe-create <project_id>
```

Create a cloudpipe instance for the given project.

### Positional arguments

**<project\_id>**      UUID of the project to create the cloudpipe for.

## nova cloudpipe-list command

```
usage: nova cloudpipe-list
```



Print a list of all cloudpipe instances.

## nova console-log command

```
usage: nova console-log [--length <length>] <server>
```

Get console log output of a server.

### Positional arguments

**<server>**      Name or ID of server.

### Optional arguments

**-length <length>**      Length in lines to tail.

## nova credentials command

```
usage: nova credentials [--wrap <integer>]
```

Show user credentials returned from auth.

### Optional arguments

**-wrap <integer>**      wrap PKI tokens to a specified length, or 0 to disable

## nova delete command

```
usage: nova delete <server> [<server> ...]
```

Immediately shut down and delete specified server(s).

### Positional arguments

**<server>**      Name or ID of server(s).

## nova diagnostics command

```
usage: nova diagnostics <server>
```

Retrieve server diagnostics.

### Positional arguments

**<server>**      Name or ID of server.

## nova dns-create command

```
usage: nova dns-create [--type <type>] <ip> <name> <domain>
```

Create a DNS entry for domain, name and ip.

### Positional arguments

**<ip>**            ip address

**<name>**        DNS name

**<domain>**     DNS domain

### Optional arguments

**-type <type>**      dns type (e.g. "A")

## nova dns-create-private-domain command

```
usage: nova dns-create-private-domain  
                                [--availability-zone <availability-zone>]  
                                <domain>
```

Create the specified DNS domain.

### Positional arguments

**<domain>**     DNS domain

### Optional arguments

**--availability-zone <availability-zone>**     Limit access to this domain to servers in the specified availability zone.

## nova dns-create-public-domain command

```
usage: nova dns-create-public-domain [--project <project>] <domain>
```

Create the specified DNS domain.

### Positional arguments

**<domain>**     DNS domain

### Optional arguments

**--project <project>**     Limit access to this domain to users of the specified project.

## nova dns-delete command

```
usage: nova dns-delete <domain> <name>
```

Delete the specified DNS entry.

### Positional arguments

**<domain>**     DNS domain

**<name>**       DNS name

## nova dns-delete-domain command

```
usage: nova dns-delete-domain <domain>
```

Delete the specified DNS domain.

### Positional arguments

**<domain>**     DNS domain

## nova dns-domains command

```
usage: nova dns-domains
```

Print a list of available dns domains.

## nova dns-list command

```
usage: nova dns-list [--ip <ip>] [--name <name>] <domain>
```

List current DNS entries for domain and ip or domain and name.

### Positional arguments

**<domain>**    DNS domain

### Optional arguments

**-ip <ip>**                      ip address

**-name <name> DNS**        name

## nova endpoints command

```
usage: nova endpoints
```

Discover endpoints that get returned from the authenticate services.

## nova evacuate command

```
usage: nova evacuate [--password <password>] [--on-shared-storage]
                   <server> <host>
```

Evacuate server from failed host to specified one.

### Positional arguments

**<server>**        Name or ID of server.

**<host>**         Name or ID of target host.

### Optional arguments

- password <password>** Set the provided password on the evacuated server. Not applicable with on-shared-storage flag
- on-shared-storage** Specifies whether server files are located on shared storage

## nova fixed-ip-get command

```
usage: nova fixed-ip-get <fixed_ip>
```

Retrieve info on a fixed ip.

### Positional arguments

**<fixed\_ip>** Fixed IP Address.

## nova fixed-ip-reserve command

```
usage: nova fixed-ip-reserve <fixed_ip>
```

Reserve a fixed IP.

### Positional arguments

**<fixed\_ip>** Fixed IP Address.

## nova fixed-ip-unreserve command

```
usage: nova fixed-ip-unreserve <fixed_ip>
```

Unreserve a fixed IP.

### Positional arguments

**<fixed\_ip>**      Fixed IP Address.

## nova flavor-access-add command

```
usage: nova flavor-access-add <flavor> <tenant_id>
```

Add flavor access for the given tenant.

### Positional arguments

**<flavor>**            Flavor name or ID to add access for the given tenant.

**<tenant\_id>**        Tenant ID to add flavor access for.

## nova flavor-access-list command

```
usage: nova flavor-access-list [--flavor <flavor>] [--tenant <tenant_id>]
```

Print access information about the given flavor.

### Optional arguments

**-flavor <flavor>**            Filter results by flavor name or ID.

**-tenant <tenant\_id>**        Filter results by tenant ID.

## nova flavor-access-remove command

```
usage: nova flavor-access-remove <flavor> <tenant_id>
```

Remove flavor access for the given tenant.

### Positional arguments

**<flavor>** Flavor name or ID to remove access for the given tenant.

**<tenant\_id>** Tenant ID to remove flavor access for.

## nova flavor-create command

```
usage: nova flavor-create [--ephemeral <ephemeral>] [--swap <swap>]
                        [--rxtx-factor <factor>] [--is-public <is-public>]
                        <name> <id> <ram> <disk> <vcpus>
```

Create a new flavor

### Positional arguments

**<name>** Name of the new flavor

**<id>** Unique ID (integer or UUID) for the new flavor. If specifying 'auto', a UUID will be generated as id

**<ram>** Memory size in MB

**<disk>** Disk size in GB

**<vcpus>** Number of vcpus



### Optional arguments

|                               |                                                     |
|-------------------------------|-----------------------------------------------------|
| <b>-ephemeral</b> <ephemeral> | Ephemeral space size in GB (default 0)              |
| <b>-swap</b> <swap>           | Swap space size in MB (default 0)                   |
| <b>-rxtx-factor</b> <factor>  | RX/TX factor (default 1)                            |
| <b>-is-public</b> <is-public> | Make flavor accessible to the public (default true) |

## nova flavor-delete command

```
usage: nova flavor-delete <flavor>
```

Delete a specific flavor

### Positional arguments

**<flavor>**      Name or ID of the flavor to delete

## nova flavor-key command

```
usage: nova flavor-key <flavor> <action> <key=value> [<key=value> ...]
```

Set or unset extra\_spec for a flavor.

### Positional arguments

**<flavor>**      Name or ID of flavor

**<action>**      Actions: 'set' or 'unset'

**<key=value>**      Extra\_specs to set/unset (only key is necessary on unset)

## nova flavor-list command

```
usage: nova flavor-list [--extra-specs] [--all]
```

Print a list of available 'flavors' (sizes of servers).

### Optional arguments

**--extra-specs**      Get extra-specs of each flavor.  
**--all**                Display all flavors (Admin only).

## nova flavor-show command

```
usage: nova flavor-show <flavor>
```

Show details about the given flavor.

### Positional arguments

**<flavor>**      Name or ID of flavor

## nova floating-ip-associate command

```
usage: nova floating-ip-associate [--fixed-address <fixed_address>]  
                                <server> <address>
```

Associate a floating IP address to a server.

### Positional arguments

**<server>**      Name or ID of server.

**<address>**      IP Address.

### Optional arguments

**-fixed-address <fixed\_address>**      Fixed IP Address to associate with.

## nova floating-ip-bulk-create command

```
usage: nova floating-ip-bulk-create [--pool <pool>] [--interface <interface>]
                                     <range>
```

Bulk create floating ips by range.

### Positional arguments

**<range>**      Address range to create

### Optional arguments

**-pool <pool>**      Pool for new Floating IPs

**-interface <interface>**      Interface for new Floating IPs

## nova floating-ip-bulk-delete command

```
usage: nova floating-ip-bulk-delete <range>
```

Bulk delete floating ips by range.

### Positional arguments

**<range>**     Address range to delete

## nova floating-ip-bulk-list command

```
usage: nova floating-ip-bulk-list [--host <host>]
```

List all floating ips.

### Optional arguments

**-host <host>**     Filter by host

## nova floating-ip-create command

```
usage: nova floating-ip-create [<floating-ip-pool>]
```

Allocate a floating IP for the current tenant.

### Positional arguments

**<floating-ip-pool>**     Name of Floating IP Pool. (Optional)

## nova floating-ip-delete command

```
usage: nova floating-ip-delete <address>
```

De-allocate a floating IP.

**Positional arguments**

**<address>** IP of Floating Ip.

**nova floating-ip-disassociate command**

```
usage: nova floating-ip-disassociate <server> <address>
```

Disassociate a floating IP address from a server.

**Positional arguments**

**<server>** Name or ID of server.

**<address>** IP Address.

**nova floating-ip-list command**

```
usage: nova floating-ip-list
```

List floating ips for this tenant.

**nova floating-ip-pool-list command**

```
usage: nova floating-ip-pool-list
```

List all floating ip pools.

**nova force-delete command**

```
usage: nova force-delete <server>
```

Force delete a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova get-password command

```
usage: nova get-password <server> [<private-key>]
```

Get password for a server.

### Positional arguments

**<server>**              Name or ID of server.

**<private-key>**      Private key (used locally to decrypt password) (Optional). When specified, the command displays the clear (decrypted) VM password. When not specified, the ciphered VM password is displayed.

## nova get-rdp-console command

```
usage: nova get-rdp-console <server> <console-type>
```

Get a rdp console to a server.

### Positional arguments

**<server>**              Name or ID of server.

**<console-type>**      Type of rdp console ("rdp-html5").

## nova get-spice-console command

```
usage: nova get-spice-console <server> <console-type>
```

Get a spice console to a server.

### Positional arguments

**<server>**              Name or ID of server.

**<console-type>**      Type of spice console ("spice-html5").

## nova get-vnc-console command

```
usage: nova get-vnc-console <server> <console-type>
```

Get a vnc console to a server.

### Positional arguments

**<server>**              Name or ID of server.

**<console-type>**      Type of vnc console ("novnc" or "xvpvnc").

## nova host-action command

```
usage: nova host-action [--action <action>] <hostname>
```

Perform a power action on a host.

### Positional arguments

**<hostname>**    Name of host.

### Optional arguments

**-action <action> A**            power action: startup, reboot, or shutdown.

## nova host-describe command

```
usage: nova host-describe <hostname>
```

Describe a specific host.

### Positional arguments

**<hostname>**    Name of host.

## nova host-evacuate command

```
usage: nova host-evacuate [--target_host <target_host>] [--on-shared-storage]
                           <host>
```

Evacuate all instances from failed host to specified one.

### Positional arguments

**<host>**        Name of host.

### Optional arguments

**-target\_host <target\_host>**        Name of target host.



**-on-shared-storage**

Specifies whether all instances files are on shared storage

## nova host-list command

```
usage: nova host-list [--zone <zone>]
```

List all hosts by service.

### Optional arguments

**-zone <zone>** Filters the list, returning only those hosts in the availability zone <zone>.

## nova host-meta command

```
usage: nova host-meta <host> <action> <key=value> [<key=value> ...]
```

Set or Delete metadata on all instances of a host.

### Positional arguments

**<host>** Name of host.

**<action>** Actions: 'set' or 'delete'

**<key=value>** Metadata to set or delete (only key is necessary on delete)

## nova host-servers-migrate command

```
usage: nova host-servers-migrate <host>
```

Migrate all instances of the specified host to other available hosts.

### Positional arguments

**<host>**     Name of host.

## nova host-update command

```
usage: nova host-update [--status <enable|disable>]
                        [--maintenance <enable|disable>]
                        <hostname>
```

Update host settings.

### Positional arguments

**<hostname>**     Name of host.

### Optional arguments

**--status <enable|disable>**     Either enable or disable a host.

**--maintenance <enable|disable>**     Either put or resume host to/from maintenance.

## nova hypervisor-list command

```
usage: nova hypervisor-list [--matching <hostname>]
```

List hypervisors.

### Optional arguments

**--matching <hostname>**     List hypervisors matching the given <hostname>.

## nova hypervisor-servers command

```
usage: nova hypervisor-servers <hostname>
```

List servers belonging to specific hypervisors.

### Positional arguments

**<hostname>** The hypervisor hostname (or pattern) to search for.

## nova hypervisor-show command

```
usage: nova hypervisor-show <hypervisor>
```

Display the details of the specified hypervisor.

### Positional arguments

**<hypervisor>** Name or ID of the hypervisor to show the details of.

## nova hypervisor-stats command

```
usage: nova hypervisor-stats
```

Get hypervisor statistics over all compute nodes.

## nova hypervisor-uptime command

```
usage: nova hypervisor-uptime <hypervisor>
```

Display the uptime of the specified hypervisor.

### Positional arguments

**<hypervisor>**      Name or ID of the hypervisor to show the uptime of.

## nova image-create command

```
usage: nova image-create [--show] [--poll] <server> <name>
```

Create a new image by taking a snapshot of a running server.

### Positional arguments

**<server>**      Name or ID of server.

**<name>**      Name of snapshot.

### Optional arguments

**--show**      Print image info.

**--poll**      Blocks while server snapshots so progress can be reported.

## nova image-delete command

```
usage: nova image-delete <image> [<image> ...]
```

Delete specified image(s).

### Positional arguments

**<image>**     Name or ID of image(s).

## nova image-list command

```
usage: nova image-list [--limit <limit>]
```

Print a list of available images to boot from.

### Optional arguments

**--limit <limit>**     number of images to return per request

## nova image-meta command

```
usage: nova image-meta <image> <action> <key=value> [<key=value> ...]
```

Set or Delete metadata on an image.

### Positional arguments

**<image>**     Name or ID of image

**<action>**     Actions: 'set' or 'delete'

**<key=value>**     Metadata to add/update or delete (only key is necessary on delete)

## nova image-show command

```
usage: nova image-show <image>
```

Show details about the given image.

### Positional arguments

**<image>**     Name or ID of image

## nova instance-action command

```
usage: nova instance-action <server> <request_id>
```

Show an action.

### Positional arguments

**<server>**             Name or UUID of the server to show an action for.

**<request\_id>**        Request ID of the action to get.

## nova instance-action-list command

```
usage: nova instance-action-list <server>
```

List actions on a server.

### Positional arguments

**<server>**     Name or UUID of the server to list actions for.

## nova interface-attach command

```
usage: nova interface-attach [--port-id <port_id>] [--net-id <net_id>]
```

```
[--fixed-ip <fixed_ip>]  
<server>
```

Attach a network interface to a server.

### Positional arguments

**<server>**      Name or ID of server.

### Optional arguments

**-port-id <port\_id>**      Port ID.  
**-net-id <net\_id>**      Network ID  
**-fixed-ip <fixed\_ip>**      Requested fixed IP.

## nova interface-detach command

```
usage: nova interface-detach <server> <port_id>
```

Detach a network interface from a server.

### Positional arguments

**<server>**      Name or ID of server.  
**<port\_id>**      Port ID.

## nova interface-list command

```
usage: nova interface-list <server>
```

List interfaces attached to a server.

### Positional arguments

**<server>**     Name or ID of server.

## nova keypair-add command

```
usage: nova keypair-add [--pub-key <pub-key>] <name>
```

Create a new key pair for use with servers.

### Positional arguments

**<name>**     Name of key.

### Optional arguments

**--pub-key <pub-key>**     Path to a public ssh key.

## nova keypair-delete command

```
usage: nova keypair-delete <name>
```

Delete keypair given by its name.

### Positional arguments

**<name>**     Keypair name to delete.



## nova keypair-list command

```
usage: nova keypair-list
```

Print a list of keypairs for a user

## nova keypair-show command

```
usage: nova keypair-show <keypair>
```

Show details about the given keypair.

### Positional arguments

**<keypair>**      Name or ID of keypair

## nova list command

```
usage: nova list [--reservation-id <reservation-id>] [--ip <ip-regex>]
               [--ip6 <ip6-regex>] [--name <name-regex>]
               [--instance-name <name-regex>] [--status <status>]
               [--flavor <flavor>] [--image <image>] [--host <hostname>]
               [--all-tenants [<0|1>]] [--tenant [<tenant>]] [--deleted]
               [--fields <fields>] [--minimal]
```

List active servers.

### Optional arguments

**--reservation-id <reservation-id>**      Only return servers that match reservation-id.

|                                           |                                                                                                    |
|-------------------------------------------|----------------------------------------------------------------------------------------------------|
| <b>-ip &lt;ip-regexp&gt;</b>              | Search with regular expression match by IP address (Admin only).                                   |
| <b>-ip6 &lt;ip6-regexp&gt;</b>            | Search with regular expression match by IPv6 address (Admin only).                                 |
| <b>-name &lt;name-regexp&gt;</b>          | Search with regular expression match by name                                                       |
| <b>-instance-name &lt;name-regexp&gt;</b> | Search with regular expression match by server name (Admin only).                                  |
| <b>-status &lt;status&gt;</b>             | Search by server status                                                                            |
| <b>-flavor &lt;flavor&gt;</b>             | Search by flavor name or ID                                                                        |
| <b>-image &lt;image&gt;</b>               | Search by image name or ID                                                                         |
| <b>-host &lt;hostname&gt;</b>             | Search servers by hostname to which they are assigned (Admin only).                                |
| <b>-all-tenants [&lt;0 1&gt;]</b>         | Display information from all tenants (Admin only).                                                 |
| <b>-tenant [&lt;tenant&gt;]</b>           | Display information from single tenant (Admin only).                                               |
| <b>-deleted</b>                           | Only display deleted servers (Admin only).                                                         |
| <b>-fields &lt;fields&gt;</b>             | Comma-separated list of fields to display. Use the show command to see which fields are available. |
| <b>-minimal</b>                           | Get only uuid and name.                                                                            |

## nova list-extensions command

```
usage: nova list-extensions
```

List all the os-api extensions that are available.

## nova list-secgroup command

```
usage: nova list-secgroup <server>
```

List Security Group(s) of a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova live-migration command

```
usage: nova live-migration [--block-migrate] [--disk-over-commit]
                           <server> [<host>]
```

Migrate running server to a new machine.

### Positional arguments

**<server>**      Name or ID of server.

**<host>**        destination host name.

### Optional arguments

**--block-migrate**      True in case of block\_migration. (Default=False:live\_migration)

**--disk-over-commit**    Allow overcommit.(Default=False)

## nova lock command

```
usage: nova lock <server>
```

Lock a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova meta command

```
usage: nova meta <server> <action> <key=value> [<key=value> ...]
```

Set or Delete metadata on a server.

### Positional arguments

**<server>**      Name or ID of server

**<action>**      Actions: 'set' or 'delete'

**<key=value>**    Metadata to set or delete (only key is necessary on delete)

## nova migrate command

```
usage: nova migrate [--poll] <server>
```

Migrate a server. The new host will be selected by the scheduler.

### Positional arguments

**<server>**      Name or ID of server.

### Optional arguments

**-poll**      Blocks while server migrates so progress can be reported.

## nova migration-list command

```
usage: nova migration-list [--host <host>] [--status <status>]
                             [--cell_name <cell_name>]
```

Print a list of migrations.

### Optional arguments

**-host <host>**                      Fetch migrations for the given host.  
**-status <status>**                  Fetch migrations for the given status.  
**-cell\_name <cell\_name>**            Fetch migrations for the given cell\_name.

## nova net command

```
usage: nova net <network_id>
```

Show a network

### Positional arguments

**<network\_id>**      ID of network

## nova net-create command

```
usage: nova net-create <network_label> <cidr>
```

Create a network

### Positional arguments

**<network\_label>**      Network label (ex. my\_new\_network)

**<cidr>**                IP block to allocate from (ex. 172.16.0.0/24 or 2001:DB8::/64)

## nova net-delete command

```
usage: nova net-delete <network_id>
```

Delete a network

### Positional arguments

**<network\_id>**      ID of network

## nova net-list command

```
usage: nova net-list
```

List networks

## nova network-associate-host command

```
usage: nova network-associate-host <network> <host>
```

Associate host with network.

### Positional arguments

**<network>**     uuid of network

**<host>**         Name of host

## nova network-associate-project command

```
usage: nova network-associate-project <network>
```

Associate project with network.

### Positional arguments

**<network>**     uuid of network

## nova network-create command

```
usage: nova network-create [--fixed-range-v4 <x.x.x.x/yy>]
                           [--fixed-range-v6 CIDR_V6] [--vlan <vlan id>]
                           [--vpn <vpn start>] [--gateway GATEWAY]
                           [--gateway-v6 GATEWAY_V6] [--bridge <bridge>]
                           [--bridge-interface <bridge interface>]
                           [--multi-host <'T'/'F'>] [--dns1 <DNS Address>]
                           [--dns2 <DNS Address>] [--uuid <network uuid>]
                           [--fixed-cidr <x.x.x.x/yy>]
                           [--project-id <project id>] [--priority <number>]
                           <network_label>
```

Create a network.

## Positional arguments

**<network\_label>**      Label for network

## Optional arguments

|                                                   |                                                   |
|---------------------------------------------------|---------------------------------------------------|
| <b>-fixed-range-v4 &lt;x.x.x.x/yy&gt;</b>         | IPv4 subnet (ex: 10.0.0.0/8)                      |
| <b>-fixed-range-v6</b>                            | CIDR_V6 IPv6 subnet (ex: fe80::/64)               |
| <b>-vlan &lt;vlan id&gt;</b>                      | vlan id                                           |
| <b>-vpn &lt;vpn start&gt;</b>                     | vpn start                                         |
| <b>-gateway GATEWAY</b>                           | gateway                                           |
| <b>-gateway-v6</b>                                | GATEWAY_V6 ipv6 gateway                           |
| <b>-bridge &lt;bridge&gt;</b>                     | VIFs on this network are connected to this bridge |
| <b>-bridge-interface &lt;bridge interface&gt;</b> | the bridge is connected to this interface         |
| <b>-multi-host &lt;'T'   'F'&gt;</b>              | Multi host                                        |
| <b>-dns1 &lt;DNS Address&gt;</b>                  | First DNS                                         |
| <b>-dns2 &lt;DNS Address&gt;</b>                  | Second DNS                                        |
| <b>-uuid &lt;network uuid&gt;</b>                 | Network UUID                                      |
| <b>-fixed-cidr &lt;x.x.x.x/yy&gt;</b>             | IPv4 subnet for fixed IPS (ex: 10.20.0.0/16)      |



|                                 |                            |
|---------------------------------|----------------------------|
| <b>-project-id</b> <project id> | Project id                 |
| <b>-priority</b> <number>       | Network interface priority |

## nova network-disassociate command

```
usage: nova network-disassociate [--host-only [<0|1>]]  
                                [--project-only [<0|1>]]  
                                <network>
```

Disassociate host and/or project from the given network.

### Positional arguments

<network>    uuid of network

### Optional arguments

**-host-only** [<0|1>]

**-project-only** [<0|1>]

## nova network-list command

```
usage: nova network-list
```

Print a list of available networks.

## nova network-show command

```
usage: nova network-show <network>
```

Show details about the given network.

### Positional arguments

**<network>**    uuid or label of network

## nova pause command

```
usage: nova pause <server>
```

Pause a server.

### Positional arguments

**<server>**    Name or ID of server.

## nova quota-class-show command

```
usage: nova quota-class-show <class>
```

List the quotas for a quota class.

### Positional arguments

**<class>**    Name of quota class to list the quotas for.

## nova quota-class-update command

```
usage: nova quota-class-update [--instances <instances>] [--cores <cores>]
```

```
[--ram <ram>] [--floating-ips <floating-ips>]  
[--metadata-items <metadata-items>]  
[--injected-files <injected-files>]  
[--injected-file-content-bytes <injected-file-content-bytes>]  
[--injected-file-path-bytes <injected-file-path-bytes>]  
[--key-pairs <key-pairs>]  
[--security-groups <security-groups>]  
[--security-group-rules <security-group-rules>]  
<class>
```

Update the quotas for a quota class.

### Positional arguments

**<class>**      Name of quota class to set the quotas for.

### Optional arguments

|                                                                         |                                                        |
|-------------------------------------------------------------------------|--------------------------------------------------------|
| <b>-instances &lt;instances&gt;</b>                                     | New value for the "instances" quota.                   |
| <b>-cores &lt;cores&gt;</b>                                             | New value for the "cores" quota.                       |
| <b>-ram &lt;ram&gt;</b>                                                 | New value for the "ram" quota.                         |
| <b>-floating-ips &lt;floating-ips&gt;</b>                               | New value for the "floating-ips" quota.                |
| <b>-metadata-items &lt;metadata-items&gt;</b>                           | New value for the "metadata-items" quota.              |
| <b>-injected-files &lt;injected-files&gt;</b>                           | New value for the "injected-files" quota.              |
| <b>-injected-file-content-bytes &lt;injected-file-content-bytes&gt;</b> | New value for the "injected-file-content-bytes" quota. |

|                                                                             |                                                     |
|-----------------------------------------------------------------------------|-----------------------------------------------------|
| <b>-injected-file-path-bytes</b><br><b>&lt;injected-file-path-bytes&gt;</b> | New value for the "injected-file-path-bytes" quota. |
| <b>-key-pairs &lt;key-pairs&gt;</b>                                         | New value for the "key-pairs" quota.                |
| <b>-security-groups &lt;security-groups&gt;</b>                             | New value for the "security-groups" quota.          |
| <b>-security-group-rules &lt;security-group-rules&gt;</b>                   | New value for the "security-group-rules" quota.     |

## nova quota-defaults command

```
usage: nova quota-defaults [--tenant <tenant-id>]
```

List the default quotas for a tenant.

### Optional arguments

**-tenant <tenant-id> ID** of tenant to list the default quotas for.

## nova quota-delete command

```
usage: nova quota-delete [--tenant <tenant-id>] [--user <user-id>]
```

Delete quota for a tenant/user so their quota will Revert back to default.

### Optional arguments

**-tenant <tenant-id> ID** of tenant to delete quota for.

**-user <user-id> ID** of user to delete quota for.

## nova quota-show command

```
usage: nova quota-show [--tenant <tenant-id>] [--user <user-id>]
```

List the quotas for a tenant/user.

### Optional arguments

**--tenant <tenant-id> ID** of tenant to list the quotas for.

**--user <user-id> ID** of user to list the quotas for.

## nova quota-update command

```
usage: nova quota-update [--user <user-id>] [--instances <instances>]
                        [--cores <cores>] [--ram <ram>]
                        [--floating-ips <floating-ips>]
                        [--fixed-ips <fixed-ips>]
                        [--metadata-items <metadata-items>]
                        [--injected-files <injected-files>]
                        [--injected-file-content-bytes <injected-file-content-bytes>]
                        [--injected-file-path-bytes <injected-file-path-bytes>]
                        [--key-pairs <key-pairs>]
                        [--security-groups <security-groups>]
                        [--security-group-rules <security-group-rules>]
                        [--force]
                        <tenant-id>
```

Update the quotas for a tenant/user.

### Positional arguments

**<tenant-id>** ID of tenant to set the quotas for.

## Optional arguments

|                                                                         |                                                        |
|-------------------------------------------------------------------------|--------------------------------------------------------|
| <b>-user &lt;user-id&gt; ID</b>                                         | of user to set the quotas for.                         |
| <b>-instances &lt;instances&gt;</b>                                     | New value for the "instances" quota.                   |
| <b>-cores &lt;cores&gt;</b>                                             | New value for the "cores" quota.                       |
| <b>-ram &lt;ram&gt;</b>                                                 | New value for the "ram" quota.                         |
| <b>-floating-ips &lt;floating-ips&gt;</b>                               | New value for the "floating-ips" quota.                |
| <b>-fixed-ips &lt;fixed-ips&gt;</b>                                     | New value for the "fixed-ips" quota.                   |
| <b>-metadata-items &lt;metadata-items&gt;</b>                           | New value for the "metadata-items" quota.              |
| <b>-injected-files &lt;injected-files&gt;</b>                           | New value for the "injected-files" quota.              |
| <b>-injected-file-content-bytes &lt;injected-file-content-bytes&gt;</b> | New value for the "injected-file-content-bytes" quota. |
| <b>-injected-file-path-bytes &lt;injected-file-path-bytes&gt;</b>       | New value for the "injected-file-path-bytes" quota.    |
| <b>-key-pairs &lt;key-pairs&gt;</b>                                     | New value for the "key-pairs" quota.                   |
| <b>-security-groups &lt;security-groups&gt;</b>                         | New value for the "security-groups" quota.             |
| <b>-security-group-rules &lt;security-group-rules&gt;</b>               | New value for the "security-group-rules" quota.        |

**-force**

Whether force update the quota even if the already used and reserved exceeds the new quota

## nova rate-limits command

```
usage: nova rate-limits
```

Print a list of rate limits for a user

## nova reboot command

```
usage: nova reboot [--hard] [--poll] <server>
```

Reboot a server.

### Positional arguments

**<server>**     Name or ID of server.

### Optional arguments

**-hard**     Perform a hard reboot (instead of a soft one).

**-poll**     Blocks while server is rebooting.

## nova rebuild command

```
usage: nova rebuild [--rebuild-password <rebuild-password>] [--poll]
                  [--minimal] [--preserve-ephemeral]
                  <server> <image>
```

Shutdown, re-image, and re-boot a server.

### Positional arguments

**<server>**     Name or ID of server.

**<image>**     Name or ID of new image.

### Optional arguments

**-rebuild-password <rebuild-password>**     Set the provided password on the rebuild server.

**-poll**     Blocks while server rebuilds so progress can be reported.

**-minimal**     Skips flavor/image lookups when showing servers

**-preserve-ephemeral**     Preserve the default ephemeral storage partition on rebuild.

## nova refresh-network command

```
usage: nova refresh-network <server>
```

Refresh server network information.

### Positional arguments

**<server>**     Name or ID of a server for which the network cache should be refreshed from neutron (Admin only).



## nova remove-fixed-ip command

```
usage: nova remove-fixed-ip <server> <address>
```

Remove an IP address from a server.

### Positional arguments

**<server>**      Name or ID of server.

**<address>**    IP Address.

## nova remove-secgroup command

```
usage: nova remove-secgroup <server> <secgroup>
```

Remove a Security Group from a server.

### Positional arguments

**<server>**      Name or ID of server.

**<secgroup>**    Name of Security Group.

## nova rename command

```
usage: nova rename <server> <name>
```

Rename a server.

### Positional arguments

**<server>**     Name (old name) or ID of server.

**<name>**       New name for the server.

## nova rescue command

```
usage: nova rescue <server>
```

Rescue a server.

### Positional arguments

**<server>**     Name or ID of server.

## nova reset-network command

```
usage: nova reset-network <server>
```

Reset network of a server.

### Positional arguments

**<server>**     Name or ID of server.

## nova reset-state command

```
usage: nova reset-state [--active] <server>
```

Reset the state of a server.

### Positional arguments

**<server>**      Name or ID of server.

### Optional arguments

**-active**        Request the server be reset to "active" state instead of "error" state (the default).

## nova resize command

```
usage: nova resize [--poll] <server> <flavor>
```

Resize a server.

### Positional arguments

**<server>**      Name or ID of server.

**<flavor>**      Name or ID of new flavor.

### Optional arguments

**-poll**        Blocks while servers resizes so progress can be reported.

## nova resize-confirm command

```
usage: nova resize-confirm <server>
```

Confirm a previous resize.

### Positional arguments

**<server>**     Name or ID of server.

## nova resize-revert command

```
usage: nova resize-revert <server>
```

Revert a previous resize (and return to the previous VM).

### Positional arguments

**<server>**     Name or ID of server.

## nova restore command

```
usage: nova restore <server>
```

Restore a soft-deleted server.

### Positional arguments

**<server>**     Name or ID of server.

## nova resume command

```
usage: nova resume <server>
```

Resume a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova root-password command

```
usage: nova root-password <server>
```

Change the root password for a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova scrub command

```
usage: nova scrub <project_id>
```

Delete data associated with the project.

### Positional arguments

**<project\_id>**      The ID of the project.

## nova secgroup-add-group-rule command

```
usage: nova secgroup-add-group-rule <secgroup> <source-group> <ip-proto>  
                                     <from-port> <to-port>
```

Add a source group rule to a security group.

### Positional arguments

|                             |                               |
|-----------------------------|-------------------------------|
| <b>&lt;secgroup&gt;</b>     | ID or name of security group. |
| <b>&lt;source-group&gt;</b> | ID or name of source group.   |
| <b>&lt;ip-proto&gt;</b>     | IP protocol (icmp, tcp, udp). |
| <b>&lt;from-port&gt;</b>    | Port at start of range.       |
| <b>&lt;to-port&gt;</b>      | Port at end of range.         |

## nova secgroup-add-rule command

```
usage: nova secgroup-add-rule <secgroup> <ip-proto> <from-port> <to-port>
                                <cidr>
```

Add a rule to a security group.

### Positional arguments

|                          |                               |
|--------------------------|-------------------------------|
| <b>&lt;secgroup&gt;</b>  | ID or name of security group. |
| <b>&lt;ip-proto&gt;</b>  | IP protocol (icmp, tcp, udp). |
| <b>&lt;from-port&gt;</b> | Port at start of range.       |
| <b>&lt;to-port&gt;</b>   | Port at end of range.         |
| <b>&lt;cidr&gt;</b>      | CIDR for address range.       |

## nova secgroup-create command

```
usage: nova secgroup-create <name> <description>
```

Create a security group.

### Positional arguments

**<name>**                Name of security group.

**<description>**        Description of security group.

## nova secgroup-delete command

```
usage: nova secgroup-delete <secgroup>
```

Delete a security group.

### Positional arguments

**<secgroup>**        ID or name of security group.

## nova secgroup-delete-group-rule command

```
usage: nova secgroup-delete-group-rule <secgroup> <source-group> <ip-PROTO>  
                                     <from-port> <to-port>
```

Delete a source group rule from a security group.

### Positional arguments

**<secgroup>**        ID or name of security group.

|                             |                               |
|-----------------------------|-------------------------------|
| <b>&lt;source-group&gt;</b> | ID or name of source group.   |
| <b>&lt;ip-proto&gt;</b>     | IP protocol (icmp, tcp, udp). |
| <b>&lt;from-port&gt;</b>    | Port at start of range.       |
| <b>&lt;to-port&gt;</b>      | Port at end of range.         |

## nova secgroup-delete-rule command

```
usage: nova secgroup-delete-rule <secgroup> <ip-proto> <from-port> <to-port>
                                     <cidr>
```

Delete a rule from a security group.

### Positional arguments

|                          |                               |
|--------------------------|-------------------------------|
| <b>&lt;secgroup&gt;</b>  | ID or name of security group. |
| <b>&lt;ip-proto&gt;</b>  | IP protocol (icmp, tcp, udp). |
| <b>&lt;from-port&gt;</b> | Port at start of range.       |
| <b>&lt;to-port&gt;</b>   | Port at end of range.         |
| <b>&lt;cidr&gt;</b>      | CIDR for address range.       |

## nova secgroup-list command

```
usage: nova secgroup-list [--all-tenants [<0|1>]]
```

List security groups for the current tenant.



### Optional arguments

**-all-tenants** [**<0|1>**]      Display information from all tenants (Admin only).

## nova secgroup-list-rules command

```
usage: nova secgroup-list-rules <secgroup>
```

List rules for a security group.

### Positional arguments

**<secgroup>**      ID or name of security group.

## nova secgroup-update command

```
usage: nova secgroup-update <secgroup> <name> <description>
```

Update a security group.

### Positional arguments

**<secgroup>**      ID or name of security group.

**<name>**          Name of security group.

**<description>**      Description of security group.

## nova service-disable command

```
usage: nova service-disable [--reason <reason>] <hostname> <binary>
```

Disable the service.

### Positional arguments

**<hostname>**     Name of host.

**<binary>**       Service binary.

### Optional arguments

**-reason <reason>**     Reason for disabling service.

## nova service-enable command

```
usage: nova service-enable <hostname> <binary>
```

Enable the service.

### Positional arguments

**<hostname>**     Name of host.

**<binary>**       Service binary.

## nova service-list command

```
usage: nova service-list [--host <hostname>] [--binary <binary>]
```

Show a list of all running services. Filter by host & binary.

### Optional arguments

**-host <hostname>**      Name of host.

**-binary <binary>**      Service binary.

## nova shelve command

```
usage: nova shelve <server>
```

Shelve a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova shelve-offload command

```
usage: nova shelve-offload <server>
```

Remove a shelved server from the compute node.

### Positional arguments

**<server>**      Name or ID of server.

## nova show command

```
usage: nova show [--minimal] <server>
```

Show details about the given server.

### Positional arguments

**<server>**      Name or ID of server.

### Optional arguments

**-minimal**      Skips flavor/image lookups when showing servers

## nova ssh command

```
usage: nova ssh [--port PORT] [--private] [--ipv6] [--login <login>]
               [-i IDENTITY] [--extra-opts EXTRA]
               <server>
```

SSH into a server.

### Positional arguments

**<server>**      Name or ID of server.

### Optional arguments

|                   |                                                                                                                                                           |
|-------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-port PORT</b> | Optional flag to indicate which port to use for ssh. (Default=22)                                                                                         |
| <b>-private</b>   | Optional flag to indicate whether to only use private address attached to an instance. (Default=False). If no public address is found try private address |
| <b>-ipv6</b>      | Optional flag to indicate whether to use an IPv6 address attached to a server. (Defaults to IPv4 address)                                                 |

|                                        |                                                             |
|----------------------------------------|-------------------------------------------------------------|
| <b>-login &lt;login&gt;</b>            | Login to use.                                               |
| <b>-i IDENTITY, -identity IDENTITY</b> | Private key file, same as the -i option to the ssh command. |
| <b>-extra-opts EXTRA</b>               | Extra options to pass to ssh. see: man ssh                  |

## nova start command

```
usage: nova start <server>
```

Start a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova stop command

```
usage: nova stop <server>
```

Stop a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova suspend command

```
usage: nova suspend <server>
```

Suspend a server.

### Positional arguments

**<server>**     Name or ID of server.

## nova unlock command

```
usage: nova unlock <server>
```

Unlock a server.

### Positional arguments

**<server>**     Name or ID of server.

## nova unpause command

```
usage: nova unpause <server>
```

Unpause a server.

### Positional arguments

**<server>**     Name or ID of server.

## nova unrescue command

```
usage: nova unrescue <server>
```

Unrescue a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova unshelve command

```
usage: nova unshelve <server>
```

Unshelve a server.

### Positional arguments

**<server>**      Name or ID of server.

## nova usage command

```
usage: nova usage [--start <start>] [--end <end>] [--tenant <tenant-id>]
```

Show usage data for a single tenant.

### Optional arguments

|                                        |                                                             |
|----------------------------------------|-------------------------------------------------------------|
| <b>--start &lt;start&gt;</b>           | Usage range start date ex 2012-01-20 (default: 4 weeks ago) |
| <b>--end &lt;end&gt;</b>               | Usage range end date, ex 2012-01-20 (default: tomorrow)     |
| <b>--tenant &lt;tenant-id&gt; UUID</b> | or name of tenant to get usage for.                         |

## nova usage-list command

```
usage: nova usage-list [--start <start>] [--end <end>]
```

List usage data for all tenants.

### Optional arguments

- start <start>**            Usage range start date ex 2012-01-20 (default: 4 weeks ago)
- end <end>**                Usage range end date, ex 2012-01-20 (default: tomorrow)

## nova volume-attach command

```
usage: nova volume-attach <server> <volume> [<device>]
```

Attach a volume to a server.

### Positional arguments

- <server>**     Name or ID of server.
- <volume>**     ID of the volume to attach.
- <device>**     Name of the device e.g. /dev/vdb. Use "auto" for autoassign (if supported)

## nova volume-create command

```
usage: nova volume-create [--snapshot-id <snapshot-id>]
                           [--image-id <image-id>]
                           [--display-name <display-name>]
                           [--display-description <display-description>]
                           [--volume-type <volume-type>]
                           [--availability-zone <availability-zone>]
                           <size>
```



Add a new volume.

### Positional arguments

**<size>**      Size of volume in GB

### Optional arguments

|                                                         |                                                                |
|---------------------------------------------------------|----------------------------------------------------------------|
| <b>-snapshot-id &lt;snapshot-id&gt;</b>                 | Optional snapshot id to create the volume from. (Default=None) |
| <b>-image-id &lt;image-id&gt;</b>                       | Optional image id to create the volume from. (Default=None)    |
| <b>-display-name &lt;display-name&gt;</b>               | Optional volume name. (Default=None)                           |
| <b>-display-description &lt;display-description&gt;</b> | Optional volume description. (Default=None)                    |
| <b>-volume-type &lt;volume-type&gt;</b>                 | Optional volume type. (Default=None)                           |
| <b>-availability-zone &lt;availability-zone&gt;</b>     | Optional Availability Zone for volume. (Default=None)          |

## nova volume-delete command

```
usage: nova volume-delete <volume> [<volume> ...]
```

Remove volume(s).

### Positional arguments

**<volume>**      Name or ID of the volume(s) to delete.

## nova volume-detach command

```
usage: nova volume-detach <server> <volume>
```

Detach a volume from a server.

### Positional arguments

**<server>**     Name or ID of server.

**<volume>**    Attachment ID of the volume.

## nova volume-list command

```
usage: nova volume-list [--all-tenants [<0|1>]]
```

List all the volumes.

### Optional arguments

**--all-tenants [<0|1>]**     Display information from all tenants (Admin only).

## nova volume-show command

```
usage: nova volume-show <volume>
```

Show details about a volume.

### Positional arguments

**<volume>**    Name or ID of the volume.

## nova volume-snapshot-create command

```
usage: nova volume-snapshot-create [--force <True|False>]
                                     [--display-name <display-name>]
                                     [--display-description <display-description>]
                                     <volume-id>
```

Add a new snapshot.

### Positional arguments

**<volume-id>** ID of the volume to snapshot

### Optional arguments

|                                                          |                                                                                                          |
|----------------------------------------------------------|----------------------------------------------------------------------------------------------------------|
| <b>--force &lt;True False&gt;</b>                        | Optional flag to indicate whether to snapshot a volume even if its attached to a server. (Default=False) |
| <b>--display-name &lt;display-name&gt;</b>               | Optional snapshot name. (Default=None)                                                                   |
| <b>--display-description &lt;display-description&gt;</b> | Optional snapshot description. (Default=None)                                                            |

## nova volume-snapshot-delete command

```
usage: nova volume-snapshot-delete <snapshot>
```

Remove a snapshot.

### Positional arguments

**<snapshot>** Name or ID of the snapshot to delete.

## nova volume-snapshot-list command

```
usage: nova volume-snapshot-list
```

List all the snapshots.

## nova volume-snapshot-show command

```
usage: nova volume-snapshot-show <snapshot>
```

Show details about a snapshot.

### Positional arguments

**<snapshot>** Name or ID of the snapshot.

## nova volume-type-create command

```
usage: nova volume-type-create <name>
```

Create a new volume type.

### Positional arguments

**<name>** Name of the new flavor

## nova volume-type-delete command

```
usage: nova volume-type-delete <id>
```

Delete a specific flavor

### Positional arguments

**<id>** Unique ID of the volume type to delete

## nova volume-type-list command

```
usage: nova volume-type-list
```

Print a list of available 'volume types'.

## nova volume-update command

```
usage: nova volume-update <server> <volume> <volume>
```

Update volume attachment.

### Positional arguments

**<server>** Name or ID of server.

**<volume>** Attachment ID of the volume.

**<volume>** ID of the volume to attach.

## nova x509-create-cert command

```
usage: nova x509-create-cert [<private-key-filename>] [<x509-cert-filename>]
```

Create x509 cert for a user in tenant.

### Positional arguments

|                        |                                                        |
|------------------------|--------------------------------------------------------|
| <private-key-filename> | Filename for the private key [Default: pk.pem]         |
| <x509-cert-filename>   | Filename for the X.509 certificate [Default: cert.pem] |

## nova x509-get-root-cert command

```
usage: nova x509-get-root-cert [<filename>]
```

Fetch the x509 root cert.

### Positional arguments

<filename>      Filename to write the x509 root cert.

## Compute Image creation

You can use the nova client to list images, set and delete image metadata, delete images, and take a snapshot of a running instance to create an image.

The safest approach is to shut down the instance before you take a snapshot.

You cannot create a snapshot from an instance that has an attached volume. Detach the volume, create the image, and re-mount the volume.

### To create an image

1. Write any buffered data to disk.

For more information, see the [Taking Snapshots](#) in the *OpenStack Operations Guide*.

2. To create the image, list instances to get the server ID:

```
$ nova list
```

```
+-----+-----+-----+-----+
| ID                               | Name                | Status | Task State |
| Power State | Networks            |        |            |
+-----+-----+-----+-----+
| 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 | myCirrosServer      | ACTIVE | None       |
| Running      | private=10.0.0.3    |        |            |
+-----+-----+-----+-----+
```

In this example, the server is named `myCirrosServer`. Use this server to create a snapshot, as follows:

```
$ nova image-create myCirrosServer myCirrosImage
```

The command creates a qemu snapshot and automatically uploads the image to your repository. Only the tenant that creates the image has access to it.

3. Get details for your image to check its status:

```
$ nova image-show IMAGE
```

```
+-----+-----+
| Property                               | Value                |
+-----+-----+
| metadata owner_id                     | 66265572db174a7aa66eba661f58eb9e |
| minDisk                               | 0                    |
| metadata instance_type_name           | m1.small             |
| metadata instance_type_id             | 5                    |
| metadata instance_type_memory_mb      | 2048                 |
| id                                     | 7e5142af-1253-4634-bcc6-89482c5f2e8a |
| metadata instance_type_root_gb        | 20                   |
| metadata instance_type_rxtx_factor    | 1                    |
| metadata ramdisk_id                   | 3cf852bd-2332-48f4-9ae4-7d926d50945e |
+-----+-----+
```

|                                     |                                      |  |
|-------------------------------------|--------------------------------------|--|
| metadata image_state                | available                            |  |
| metadata image_location             | snapshot                             |  |
| minRam                              | 0                                    |  |
| metadata instance_type_vcpus        | 1                                    |  |
| status                              | ACTIVE                               |  |
| updated                             | 2013-07-22T19:46:42Z                 |  |
| metadata instance_type_swap         | 0                                    |  |
| metadata instance_type_vcpu_weight  | None                                 |  |
| metadata base_image_ref             | 397e713c-b95b-4186-ad46-6126863ea0a9 |  |
| progress                            | 100                                  |  |
| metadata instance_type_flavorid     | 2                                    |  |
| OS-EXT-IMG-SIZE:size                | 14221312                             |  |
| metadata image_type                 | snapshot                             |  |
| metadata user_id                    | 376744b5910b4b4da7d8e6cb483b06a8     |  |
| name                                | myCirrosImage                        |  |
| created                             | 2013-07-22T19:45:58Z                 |  |
| metadata instance_uuid              | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |  |
| server                              | 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 |  |
| metadata kernel_id                  | df430cc2-3406-4061-b635-a51c16e488ac |  |
| metadata instance_type_ephemeral_gb | 0                                    |  |
| +-----+-----+                       |                                      |  |

The image status changes from SAVING to ACTIVE. Only the tenant who creates the image has access to it.

### To launch an instance from your image

- To launch an instance from your image, include the image ID and flavor ID, as follows:

```
$ nova boot newServer --image 7e5142af-1253-4634-bcc6-89482c5f2e8a --flavor 3
```

|                       |               |  |
|-----------------------|---------------|--|
| Property              | Value         |  |
| OS-EXT-STS:task_state | scheduling    |  |
| image                 | myCirrosImage |  |



|                                     |                                      |
|-------------------------------------|--------------------------------------|
| OS-EXT-STS:vm_state                 | building                             |
| OS-EXT-SRV-ATTR:instance_name       | instance-00000007                    |
| flavor                              | m1.medium                            |
| id                                  | d7efd3e4-d375-46d1-9d57-372b6e4bdb7f |
| security_groups                     | [{'u'name': u'default'}]             |
| user_id                             | 376744b5910b4b4da7d8e6cb483b06a8     |
| OS-DCF:diskConfig                   | MANUAL                               |
| accessIPv4                          |                                      |
| accessIPv6                          |                                      |
| progress                            | 0                                    |
| OS-EXT-STS:power_state              | 0                                    |
| OS-EXT-AZ:availability_zone         | nova                                 |
| config_drive                        |                                      |
| status                              | BUILD                                |
| updated                             | 2013-07-22T19:58:33Z                 |
| hostId                              |                                      |
| OS-EXT-SRV-ATTR:host                | None                                 |
| key_name                            | None                                 |
| OS-EXT-SRV-ATTR:hypervisor_hostname | None                                 |
| name                                | newServer                            |
| adminPass                           | jis88nN46RGP                         |
| tenant_id                           | 66265572db174a7aa66eba661f58eb9e     |
| created                             | 2013-07-22T19:58:33Z                 |
| metadata                            | {}                                   |

## Troubleshoot image creation

- You cannot create a snapshot from an instance that has an attached volume. Detach the volume, create the image, and re-mount the volume.
- Make sure the version of qemu you are using is version 0.14 or greater. Older versions of qemu result in an "unknown option -s" error message in the nova-compute.log.

- Examine the `/var/log/nova-api.log` and `/var/log/nova-compute.log` log files for error messages.

## Compute Boot Instance

You can boot instances from a volume instead of an image. Use the **nova boot** `--block-device` parameter to define how volumes are attached to an instance when you create it. You can use the `--block-device` parameter with existing or new volumes that you create from a source image, volume, or snapshot.



### Note

To attach a volume to a running instance, see [Manage volumes](#).

## Create volume from image and boot instance

Use this procedure to create a volume from an image, and use it to boot an instance.

1. You can create a volume from an existing image, volume, or snapshot.

List available images:

```
$ nova image-list
+-----+-----+-----+-----+
+-----+
| ID                               | Name                               | Status | Server |
+-----+-----+-----+-----+
+-----+
| e0b7734d-2331-42a3-b19e-067adc0da17d | cirros-0.3.2-x86_64-uec          | ACTIVE |        |
+-----+-----+-----+-----+
| 75bf193b-237b-435e-8712-896c51484de9 | cirros-0.3.2-x86_64-uec-kernel  | ACTIVE |        |
+-----+-----+-----+-----+
```

```
| 19eee81c-f972-44e1-a952-1dceee148c47 | cirros-0.3.2-x86_64-uec-ramdisk | ACTIVE |
|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+
+-----+
```

2. To create a bootable volume from an image and launch an instance from this volume, use the `--block-device` parameter.

For example:

```
$ nova boot --flavor FLAVOR --block-device source=SOURCE,id=ID,dest=DEST,size=SIZE,
shutdown=PRESERVE,bootindex=INDEX NAME
```

The parameters are:

| Parameter                                                                                                                                                       | Description                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>--flavor <i>FLAVOR</i></code>                                                                                                                             | The flavor ID or name.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <code>--block-device</code><br><code>source=<i>SOURCE</i>,id=<i>ID</i>,dest=<i>DEST</i>,size=<i>SIZE</i>,shutdown=<i>PRESERVE</i>,bootindex=<i>INDEX</i></code> | <ul style="list-style-type: none"> <li>• <i>SOURCE</i>: The type of object used to create the block device. Valid values are image, volume, snapshot, image and blank.</li> <li>• <i>ID</i>: The ID of the source object.</li> <li>• <i>DEST</i>: The type of the target virtual device. Valid values are volume and local.</li> <li>• <i>SIZE</i>: The size of the volume that will be created.</li> <li>• <i>PRESERVE</i>: What to do with the volume when the instance is terminated. preserve will not delete the volume, remove will.</li> <li>• <i>INDEX</i>: Used to order the boot disks. Use 0 to boot from this volume.</li> </ul> |
| <i>NAME</i>                                                                                                                                                     | The name for the server.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     |

3. Create a bootable volume from an image, before the instance boots. The volume is not deleted when the instance is terminated:

```
$ nova boot --flavor 2 \
  --block-device source=image,id=e0b7734d-2331-42a3-b19e-067adc0da17d,dest=volume,size=
10,shutdown=preserve,bootindex=0 \
  myInstanceFromVolume
```

| Property                             | Value                                           |
|--------------------------------------|-------------------------------------------------|
| OS-EXT-STS:task_state                | scheduling                                      |
| image                                | Attempt to boot from volume - no image supplied |
| OS-EXT-STS:vm_state                  | building                                        |
| OS-EXT-SRV-ATTR:instance_name        | instance-00000003                               |
| OS-SRV-USG:launched_at               | None                                            |
| flavor                               | m1.small                                        |
| id                                   | 2e65c854-dba9-4f68-8f08-fe332e546ecc            |
| security_groups                      | [{u'name': u'default'}]                         |
| user_id                              | 352b37f5c89144d4ad0534139266d51f                |
| OS-DCF:diskConfig                    | MANUAL                                          |
| accessIPv4                           |                                                 |
| accessIPv6                           |                                                 |
| progress                             | 0                                               |
| OS-EXT-STS:power_state               | 0                                               |
| OS-EXT-AZ:availability_zone          | nova                                            |
| config_drive                         |                                                 |
| status                               | BUILD                                           |
| updated                              | 2014-02-02T13:29:54Z                            |
| hostId                               |                                                 |
| OS-EXT-SRV-ATTR:host                 | None                                            |
| OS-SRV-USG:terminated_at             | None                                            |
| key_name                             | None                                            |
| OS-EXT-SRV-ATTR:hypervisor_hostname  | None                                            |
| name                                 | myInstanceFromVolume                            |
| adminPass                            | TzjqyGsRcJo9                                    |
| tenant_id                            | f7ac731cc11f40efbc03a9f9e1d1d21f                |
| created                              | 2014-02-02T13:29:53Z                            |
| os-extended-volumes:volumes_attached | []                                              |

```
| metadata | {} |
+-----+-----+
```

4. List volumes to see the bootable volume and its attached myInstanceFromVolume instance:

```
$ cinder list
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
|          ID          | Status | Display Name | Size | Volume Type |
| Bootable | Attached to |          |      |             |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
| 2fff50ab-1a9c-4d45-ae60-1d054d6bc868 | in-use |          | 10 | None |
| true | 2e65c854-dba9-4f68-8f08-fe332e546ecc |          |      |             |
+-----+-----+-----+-----+-----+
+-----+-----+-----+-----+-----+
```

## Attach non-bootable volume to an instance

Use the `--block-device` parameter to attach an existing, non-bootable volume to a new instance.

1. Create a volume:

```
$ cinder create --display-name my-volume 8
+-----+-----+
| Property | Value |
+-----+-----+
| attachments | [] |
| availability_zone | nova |
| bootable | false |
| created_at | 2014-02-04T21:25:18.730961 |
| display_description | None |
| display_name | my-volume |
| id | 3195a5a7-fd0d-4ac3-b919-7ba6cbe11d46 |
| metadata | {} |
| size | 8 |
+-----+-----+
```

|              |          |
|--------------|----------|
| snapshot_id  | None     |
| source_volid | None     |
| status       | creating |
| volume_type  | None     |

## 2. List volumes:

```
$ cinder list
```

| ID                                   | Status    | Display Name | Size | Volume Type |
|--------------------------------------|-----------|--------------|------|-------------|
| Bootable   Attached to               |           |              |      |             |
| 3195a5a7-fd0d-4ac3-b919-7ba6cbe11d46 | available | my-volume    | 8    | None        |
| false                                |           |              |      |             |



### Note

The volume is not bootable because it was not created from an image.

The volume is also entirely empty: It has no partition table and no file system.

## 3. Run this command to create an instance and boot it with the volume that is attached to this instance. An image is used as boot source:

```
$ nova boot --flavor 2 --image e0b7734d-2331-42a3-b19e-067adc0da17d \
  --block-device source=volume,id=3195a5a7-fd0d-4ac3-b919-7ba6cbe11d46,dest=volume,
shutdown=preserve \
  myInstanceWithVolume
```

| Property                      | Value                                |
|-------------------------------|--------------------------------------|
| +-----+                       |                                      |
| OS-EXT-STS:task_state         | scheduling                           |
| image                         | e0b7734d-2331-42a3-b19e-067adc0da17d |
| OS-EXT-STS:vm_state           | building                             |
| OS-EXT-SRV-ATTR:instance_name | instance-00000003                    |
| flavor                        | m1.small                             |
| id                            | 8ed8b0f9-70de-4662-a16c-0b51ce7b17b4 |
| security_groups               | [{'name': 'default'}]                |
| user_id                       | 352b37f5c89144d4ad0534139266d51f     |
| OS-DCF:diskConfig             | MANUAL                               |
| accessIPv4                    |                                      |
| accessIPv6                    |                                      |
| progress                      | 0                                    |
| OS-EXT-STS:power_state        | 0                                    |
| OS-EXT-AZ:availability_zone   | nova                                 |
| config_drive                  |                                      |
| status                        | BUILD                                |

```

| updated | 2013-10-16T01:43:26Z
|
| hostId |
|
| OS-EXT-SRV-ATTR:host | None
|
| OS-SRV-USG:terminated_at | None
|
| key_name | None
|
| OS-EXT-SRV-ATTR:hypervisor_hostname | None
|
| name | myInstanceWithVolume
|
| adminPass | BULD33uzYwhq
|
| tenant_id | f7ac731cc11f40efbc03a9f9e1d1d21f
|
| created | 2013-10-16T01:43:25Z
|
| os-extended-volumes:volumes_attached | [{u'id': u'3195a5a7-fd0d-4ac3-
b919-7ba6cbe11d46'}] |
| metadata | {}
|
+-----+
+-----+

```

#### 4. List volumes:

```
$ nova volume-list
```

Note that the volume is attached to a server:

```

+-----+-----+-----+-----+-----+
+-----+

```



| ID                                   | Status | Display Name | Size | Volume Type |
|--------------------------------------|--------|--------------|------|-------------|
| Attached to                          |        |              |      |             |
| 3195a5a7-fd0d-4ac3-b919-7ba6cbe11d46 | in-use | my-volume    | 8    | None        |
| 8ed8b0f9-70de-4662-a16c-0b51ce7b17b4 |        |              |      |             |

## Attach swap or ephemeral disk to an instance

Use the **nova boot** `--swap` parameter to attach a swap disk on boot or the **nova boot** `--ephemeral` parameter to attach an ephemeral disk on boot. When you terminate the instance, both disks are deleted.

Boot an instance with a 512 MB swap disk and 2 GB ephemeral disk:

```
$ nova boot --flavor FLAVOR --image IMAGE_ID --swap 512 --ephemeral size=2 NAME
```



### Note

The flavor defines the maximum swap and ephemeral disk size. You cannot exceed these maximum values.

## Compute Terminate Instance

When you no longer need an instance, you can delete it.

1. List all instances:

```
$ nova list
```

| ID                     | Name | Status | Task State |
|------------------------|------|--------|------------|
| Power State   Networks |      |        |            |

```
+-----+-----+-----+-----+
+-----+-----+
| 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 | myCirrosServer      | ACTIVE | None      |
Running      | private=10.0.0.3 |
| 8a99547e-7385-4ad1-ae50-4ecfaaad5f42 | myInstanceFromVolume | ACTIVE | None      |
Running      | private=10.0.0.4 |
| d7efd3e4-d375-46d1-9d57-372b6e4bdb7f | newServer           | ERROR  | None      |
NOSTATE      |                   |
+-----+-----+-----+-----+
+-----+-----+
```

2. Run the **nova delete** command to delete the instance. The following example shows deletion of the `newServer` instance, which is in `ERROR` state:

```
$ nova delete newServer
```

The command does not notify that your server was deleted.

3. To verify that the server was deleted, run the **nova list** command:

```
$ nova list
+-----+-----+-----+-----+
+-----+-----+
| ID                                     | Name                  | Status | Task State |
Power State | Networks              |
+-----+-----+-----+-----+
+-----+-----+
| 84c6e57d-a6b1-44b6-81eb-fcb36afd31b5 | myCirrosServer        | ACTIVE | None      |
Running      | private=10.0.0.3 |
| 8a99547e-7385-4ad1-ae50-4ecfaaad5f42 | myInstanceFromVolume | ACTIVE | None      |
Running      | private=10.0.0.4 |
+-----+-----+-----+-----+
+-----+-----+
```

The deleted instance does not appear in the list.

## 6. Compute Node Quiz

### Table of Contents

|                             |     |
|-----------------------------|-----|
| Day 1, 16:40 to 17:00 ..... | 317 |
|-----------------------------|-----|

### Day 1, 16:40 to 17:00



## 7. Network Node

### Table of Contents

|                                     |     |
|-------------------------------------|-----|
| Day 2, 09:00 to 11:00 .....         | 319 |
| Networking in OpenStack .....       | 319 |
| OpenStack Networking Concepts ..... | 325 |
| Administration Tasks .....          | 327 |

### Day 2, 09:00 to 11:00

## Networking in OpenStack

### Networking in OpenStack

OpenStack Networking provides a rich tenant-facing API for defining network connectivity and addressing in the cloud. The OpenStack Networking project gives operators the ability to leverage different networking technologies to power their cloud networking. It is a virtual network service that provides a powerful API to define the network connectivity and addressing used by devices from other services, such as OpenStack Compute. It has a rich API which consists of the following components.

- **Network:** An isolated L2 segment, analogous to VLAN in the physical networking world.
- **Subnet:** A block of v4 or v6 IP addresses and associated configuration state.

- **Port:** A connection point for attaching a single device, such as the NIC of a virtual server, to a virtual network. Also describes the associated network configuration, such as the MAC and IP addresses to be used on that port.

You can configure rich network topologies by creating and configuring networks and subnets, and then instructing other OpenStack services like OpenStack Compute to attach virtual devices to ports on these networks. In particular, OpenStack Networking supports each tenant having multiple private networks, and allows tenants to choose their own IP addressing scheme, even if those IP addresses overlap with those used by other tenants. This enables very advanced cloud networking use cases, such as building multi-tiered web applications and allowing applications to be migrated to the cloud without changing IP addresses.

### Plugin Architecture: Flexibility to Choose Different Network Technologies

Enhancing traditional networking solutions to provide rich cloud networking is challenging. Traditional networking is not designed to scale to cloud proportions or to configure automatically.

The original OpenStack Compute network implementation assumed a very basic model of performing all isolation through Linux VLANs and IP tables. OpenStack Networking introduces the concept of a plug-in, which is a pluggable back-end implementation of the OpenStack Networking API. A plug-in can use a variety of technologies to implement the logical API requests. Some OpenStack Networking plug-ins might use basic Linux VLANs and IP tables, while others might use more advanced technologies, such as L2-in-L3 tunneling or OpenFlow, to provide similar benefits.

The current set of plug-ins include:

- **Big Switch, Floodlight REST Proxy:** <http://www.openflowhub.org/display/floodlightcontroller/Quantum+REST+Proxy+Plugin>
- **Brocade Plugin**
- **Cisco:** Documented externally at: <http://wiki.openstack.org/cisco-quantum>
- **Hyper-V Plugin**

- **Linux Bridge:** Documentation included in this guide and <http://wiki.openstack.org/Quantum-Linux-Bridge-Plugin>
- **Midonet Plugin**
- **NEC OpenFlow:** <http://wiki.openstack.org/Quantum-NEC-OpenFlow-Plugin>
- **Open vSwitch:** Documentation included in this guide.
- **PLUMgrid:** <https://wiki.openstack.org/wiki/Plumgrid-quantum>
- **Ryu:** <https://github.com/osrg/ryu/wiki/OpenStack>
- **VMware NSX:** Documentation include in this guide, [NSX Product Overview](#) , and [NSX Product Support](#).

Plugins can have different properties in terms of hardware requirements, features, performance, scale, operator tools, etc. Supporting many plug-ins enables the cloud administrator to weigh different options and decide which networking technology is right for the deployment.

### Components of OpenStack Networking

To deploy OpenStack Networking, it is useful to understand the different components that make up the solution and how those components interact with each other and with other OpenStack services.

OpenStack Networking is a standalone service, just like other OpenStack services such as OpenStack Compute, OpenStack Image Service, OpenStack Identity service, and the OpenStack Dashboard. Like those services, a deployment of OpenStack Networking often involves deploying several processes on a variety of hosts.

The main process of the OpenStack Networking server is quantum-server, which is a Python daemon that exposes the OpenStack Networking API and passes user requests to the configured OpenStack Networking plug-in for additional processing. Typically, the plug-in requires access to a database for persistent storage, similar to other OpenStack services.

If your deployment uses a controller host to run centralized OpenStack Compute components, you can deploy the OpenStack Networking server on that same host. However, OpenStack Networking is entirely standalone and can be deployed on its own server as well. OpenStack Networking also includes additional agents that might be required depending on your deployment:

- **plugin agent (quantum-\*agent):**Runs on each hypervisor to perform local vswitch configuration. Agent to be run depends on which plug-in you are using, as some plug-ins do not require an agent.
- **dhcp agent (quantum-dhcp-agent):**Provides DHCP services to tenant networks. This agent is the same across all plug-ins.
- **l3 agent (quantum-l3-agent):**Provides L3/NAT forwarding to provide external network access for VMs on tenant networks. This agent is the same across all plug-ins.

These agents interact with the main quantum-server process in the following ways:

- Through RPC. For example, rabbitmq or qpid.
- Through the standard OpenStack Networking API.

OpenStack Networking relies on the OpenStack Identity Project (Keystone) for authentication and authorization of all API request.

OpenStack Compute interacts with OpenStack Networking through calls to its standard API. As part of creating a VM, nova-compute communicates with the OpenStack Networking API to plug each virtual NIC on the VM into a particular network.

The OpenStack Dashboard (Horizon) has integration with the OpenStack Networking API, allowing administrators and tenant users, to create and manage network services through the Horizon GUI.

### Place Services on Physical Hosts

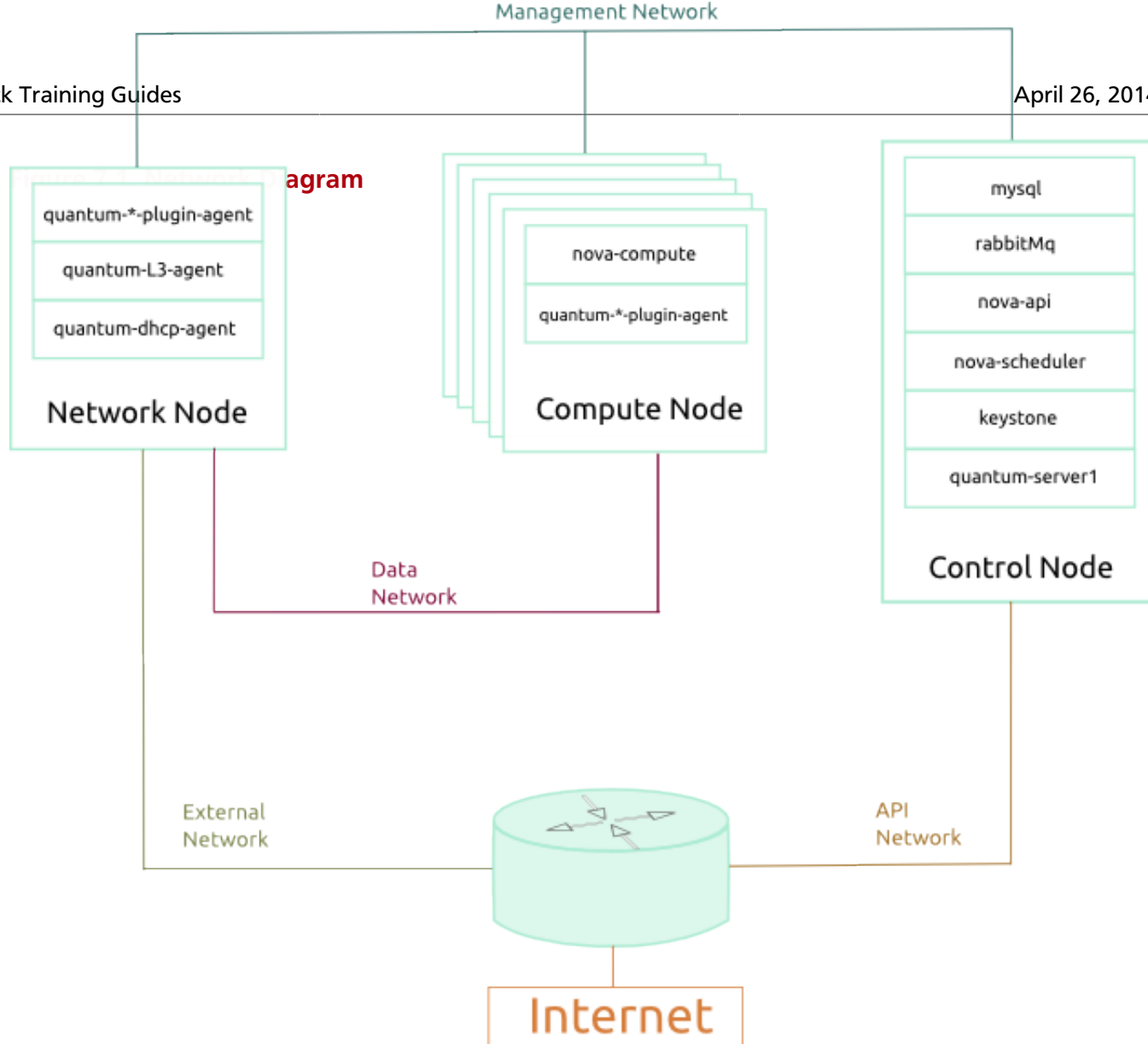
Like other OpenStack services, OpenStack Networking provides cloud administrators with significant flexibility in deciding which individual services should run on which physical devices. On one extreme, all service



daemons can be run on a single physical host for evaluation purposes. On the other, each service could have its own physical hosts, and some cases be replicated across multiple hosts for redundancy.

In this guide, we focus primarily on a standard architecture that includes a “cloud controller” host, a “network gateway” host, and a set of hypervisors for running VMs. The “cloud controller” and “network gateway” can be combined in simple deployments, though if you expect VMs to send significant amounts of traffic to or from the Internet, a dedicated network gateway host is suggested to avoid potential CPU contention between packet forwarding performed by the quantum-l3-agent and other OpenStack services.

### **Network Connectivity for Physical Hosts**



A standard OpenStack Networking setup has up to four distinct physical data center networks:

- **Management network:**Used for internal communication between OpenStack Components. The IP addresses on this network should be reachable only within the data center.
- **Data network:**Used for VM data communication within the cloud deployment. The IP addressing requirements of this network depend on the OpenStack Networking plug-in in use.
- **External network:**Used to provide VMs with Internet access in some deployment scenarios. The IP addresses on this network should be reachable by anyone on the Internet.
- **API network:**Exposes all OpenStack APIs, including the OpenStack Networking API, to tenants. The IP addresses on this network should be reachable by anyone on the Internet. This may be the same network as the external network, as it is possible to create a subnet for the external network that uses IP allocation ranges to use only less than the full range of IP addresses in an IP block.

## OpenStack Networking Concepts

### Network Types

The OpenStack Networking configuration provided by the Rackspace Private Cloud cookbooks allows you to choose between VLAN or GRE isolated networks, both provider- and tenant-specific. From the provider side, an administrator can also create a flat network.

The type of network that is used for private tenant networks is determined by the `network_type` attribute, which can be edited in the Chef `override_attributes`. This attribute sets both the default provider network type and the only type of network that tenants are able to create. Administrators can always create flat and VLAN networks. GRE networks of any type require the `network_type` to be set to `gre`.

### Namespaces

For each network you create, the Network node (or Controller node, if combined) will have a unique network namespace (`netns`) created by the DHCP and Metadata agents. The `netns` hosts an interface and IP addresses

for dnsmasq and the quantum-ns-metadata-proxy. You can view the namespaces with the `ip netns [list]`, and can interact with the namespaces with the `ip netns exec <namespace> <command>` command.

## Metadata

Not all networks or VMs need metadata access. Rackspace recommends that you use metadata if you are using a single network. If you need metadata, you may also need a default route. (If you don't need a default route, no-gateway will do.)

To communicate with the metadata IP address inside the namespace, instances need a route for the metadata network that points to the dnsmasq IP address on the same namespaced interface. OpenStack Networking only injects a route when you do not specify a gateway-ip in the subnet.

If you need to use a default route and provide instances with access to the metadata route, create the subnet without specifying a gateway IP and with a static route from 0.0.0.0/0 to your gateway IP address. Adjust the DHCP allocation pool so that it will not assign the gateway IP. With this configuration, dnsmasq will pass both routes to instances. This way, metadata will be routed correctly without any changes on the external gateway.

## OVS Bridges

An OVS bridge for provider traffic is created and configured on the nodes where single-network-node and single-compute are applied. Bridges are created, but physical interfaces are not added. An OVS bridge is not created on a Controller-only node.

When creating networks, you can specify the type and properties, such as Flat vs. VLAN, Shared vs. Tenant, or Provider vs. Overlay. These properties identify and determine the behavior and resources of instances attached to the network. The cookbooks will create bridges for the configuration that you specify, although they do not add physical interfaces to provider bridges. For example, if you specify a network type of GRE, a br-tun tunnel bridge will be created to handle overlay traffic.

# Administration Tasks

## Network CLI Commands

The **neutron** client is the command-line interface (CLI) for the OpenStack Networking API and its extensions. This chapter documents **neutron** version 2.3.4.

For help on a specific **neutron** command, enter:

```
$ neutron help COMMAND
```

### neutron usage

```
usage: neutron [--version] [-v] [-q] [-h] [--os-auth-strategy <auth-strategy>]
               [--os-auth-url <auth-url>]
               [--os-tenant-name <auth-tenant-name>]
               [--os-tenant-id <auth-tenant-id>]
               [--os-username <auth-username>] [--os-password <auth-password>]
               [--os-region-name <auth-region-name>] [--os-token <token>]
               [--endpoint-type <endpoint-type>] [--os-url <url>]
               [--os-cacert <ca-certificate>] [--insecure]
```

### neutron optional arguments

|                               |                                                                              |
|-------------------------------|------------------------------------------------------------------------------|
| <b>-version</b>               | show program's version number and exit                                       |
| <b>-v, --verbose, --debug</b> | Increase verbosity of output and show tracebacks on errors. Can be repeated. |
| <b>-q, --quiet</b>            | Suppress output except warnings and errors                                   |
| <b>-h, --help</b>             | Show this help message and exit                                              |

|                                                 |                                                                                                                                                                                                        |
|-------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-os-auth-strategy &lt;auth-strategy&gt;</b>  | Authentication strategy (Env: OS_AUTH_STRATEGY, default keystone). For now, any other value will disable the authentication                                                                            |
| <b>-os-auth-url &lt;auth-url&gt;</b>            | Authentication URL (Env: OS_AUTH_URL)                                                                                                                                                                  |
| <b>-os-tenant-name &lt;auth-tenant-name&gt;</b> | Authentication tenant name (Env: OS_TENANT_NAME)                                                                                                                                                       |
| <b>-os-tenant-id &lt;auth-tenant-id&gt;</b>     | Authentication tenant name (Env: OS_TENANT_ID)                                                                                                                                                         |
| <b>-os-username &lt;auth-username&gt;</b>       | Authentication username (Env: OS_USERNAME)                                                                                                                                                             |
| <b>-os-password &lt;auth-password&gt;</b>       | Authentication password (Env: OS_PASSWORD)                                                                                                                                                             |
| <b>-os-region-name &lt;auth-region-name&gt;</b> | Authentication region name (Env: OS_REGION_NAME)                                                                                                                                                       |
| <b>-os-token &lt;token&gt;</b>                  | Defaults to env[OS_TOKEN]                                                                                                                                                                              |
| <b>-endpoint-type &lt;endpoint-type&gt;</b>     | Defaults to env[OS_ENDPOINT_TYPE] or publicURL.                                                                                                                                                        |
| <b>-os-url &lt;url&gt;</b>                      | Defaults to env[OS_URL]                                                                                                                                                                                |
| <b>-os-cacert &lt;ca-certificate&gt;</b>        | Specify a CA bundle file to use in verifying a TLS (https) server certificate. Defaults to env[OS_CACERT]                                                                                              |
| <b>-insecure</b>                                | Explicitly allow neutronclient to perform "insecure" SSL (https) requests. The server's certificate will not be verified against any certificate authorities. This option should be used with caution. |

## neutron API v2.0 commands

|                     |                       |
|---------------------|-----------------------|
| <b>agent-delete</b> | Delete a given agent. |
|---------------------|-----------------------|

---

|                                     |                                                      |
|-------------------------------------|------------------------------------------------------|
| <b>agent-list</b>                   | List agents.                                         |
| <b>agent-show</b>                   | Show information of a given agent.                   |
| <b>agent-update</b>                 | Update a given agent.                                |
| <b>cisco-credential-create</b>      | Creates a credential.                                |
| <b>cisco-credential-delete</b>      | Delete a given credential.                           |
| <b>cisco-credential-list</b>        | List credentials that belong to a given tenant.      |
| <b>cisco-credential-show</b>        | Show information of a given credential.              |
| <b>cisco-network-profile-create</b> | Creates a network profile.                           |
| <b>cisco-network-profile-delete</b> | Delete a given network profile.                      |
| <b>cisco-network-profile-list</b>   | List network profiles that belong to a given tenant. |
| <b>cisco-network-profile-show</b>   | Show information of a given network profile.         |
| <b>cisco-network-profile-update</b> | Update network profile's information.                |
| <b>cisco-policy-profile-list</b>    | List policy profiles that belong to a given tenant.  |
| <b>cisco-policy-profile-show</b>    | Show information of a given policy profile.          |
| <b>cisco-policy-profile-update</b>  | Update policy profile's information.                 |
| <b>complete</b>                     | print bash completion command                        |
| <b>dhcp-agent-list-hosting-net</b>  | List DHCP agents hosting a network.                  |

---

|                                    |                                                       |
|------------------------------------|-------------------------------------------------------|
| <b>dhcp-agent-network-add</b>      | Add a network to a DHCP agent.                        |
| <b>dhcp-agent-network-remove</b>   | Remove a network from a DHCP agent.                   |
| <b>ext-list</b>                    | List all extensions.                                  |
| <b>ext-show</b>                    | Show information of a given resource.                 |
| <b>firewall-create</b>             | Create a firewall.                                    |
| <b>firewall-delete</b>             | Delete a given firewall.                              |
| <b>firewall-list</b>               | List firewalls that belong to a given tenant.         |
| <b>firewall-policy-create</b>      | Create a firewall policy.                             |
| <b>firewall-policy-delete</b>      | Delete a given firewall policy.                       |
| <b>firewall-policy-insert-rule</b> | Insert a rule into a given firewall policy.           |
| <b>firewall-policy-list</b>        | List firewall policies that belong to a given tenant. |
| <b>firewall-policy-remove-rule</b> | Remove a rule from a given firewall policy.           |
| <b>firewall-policy-show</b>        | Show information of a given firewall policy.          |
| <b>firewall-policy-update</b>      | Update a given firewall policy.                       |
| <b>firewall-rule-create</b>        | Create a firewall rule.                               |
| <b>firewall-rule-delete</b>        | Delete a given firewall rule.                         |
| <b>firewall-rule-list</b>          | List firewall rules that belong to a given tenant.    |



---

|                                     |                                                          |
|-------------------------------------|----------------------------------------------------------|
| <b>firewall-rule-show</b>           | Show information of a given firewall rule.               |
| <b>firewall-rule-update</b>         | Update a given firewall rule.                            |
| <b>firewall-show</b>                | Show information of a given firewall.                    |
| <b>firewall-update</b>              | Update a given firewall.                                 |
| <b>floatingip-associate</b>         | Create a mapping between a floating ip and a fixed ip.   |
| <b>floatingip-create</b>            | Create a floating ip for a given tenant.                 |
| <b>floatingip-delete</b>            | Delete a given floating ip.                              |
| <b>floatingip-disassociate</b>      | Remove a mapping from a floating ip to a fixed ip.       |
| <b>floatingip-list</b>              | List floating ips that belong to a given tenant.         |
| <b>floatingip-show</b>              | Show information of a given floating ip.                 |
| <b>help</b>                         | print detailed help for another command                  |
| <b>ipsec-site-connection-create</b> | Create an IPsecSiteConnection.                           |
| <b>ipsec-site-connection-delete</b> | Delete a given IPsecSiteConnection.                      |
| <b>ipsec-site-connection-list</b>   | List IPsecSiteConnections that belong to a given tenant. |
| <b>ipsec-site-connection-show</b>   | Show information of a given IPsecSiteConnection.         |
| <b>ipsec-site-connection-update</b> | Update a given IPsecSiteConnection.                      |
| <b>l3-agent-list-hosting-router</b> | List L3 agents hosting a router.                         |

---

|                                      |                                                       |
|--------------------------------------|-------------------------------------------------------|
| <b>l3-agent-router-add</b>           | Add a router to a L3 agent.                           |
| <b>l3-agent-router-remove</b>        | Remove a router from a L3 agent.                      |
| <b>lb-agent-hosting-pool</b>         | Get loadbalancer agent hosting a pool.                |
| <b>lb-healthmonitor-associate</b>    | Create a mapping between a health monitor and a pool. |
| <b>lb-healthmonitor-create</b>       | Create a healthmonitor.                               |
| <b>lb-healthmonitor-delete</b>       | Delete a given healthmonitor.                         |
| <b>lb-healthmonitor-disassociate</b> | Remove a mapping from a health monitor to a pool.     |
| <b>lb-healthmonitor-list</b>         | List healthmonitors that belong to a given tenant.    |
| <b>lb-healthmonitor-show</b>         | Show information of a given healthmonitor.            |
| <b>lb-healthmonitor-update</b>       | Update a given healthmonitor.                         |
| <b>lb-member-create</b>              | Create a member.                                      |
| <b>lb-member-delete</b>              | Delete a given member.                                |
| <b>lb-member-list</b>                | List members that belong to a given tenant.           |
| <b>lb-member-show</b>                | Show information of a given member.                   |
| <b>lb-member-update</b>              | Update a given member.                                |
| <b>lb-pool-create</b>                | Create a pool.                                        |
| <b>lb-pool-delete</b>                | Delete a given pool.                                  |

---

|                                |                                                     |
|--------------------------------|-----------------------------------------------------|
| <b>lb-pool-list</b>            | List pools that belong to a given tenant.           |
| <b>lb-pool-list-on-agent</b>   | List the pools on a loadbalancer agent.             |
| <b>lb-pool-show</b>            | Show information of a given pool.                   |
| <b>lb-pool-stats</b>           | Retrieve stats for a given pool.                    |
| <b>lb-pool-update</b>          | Update a given pool.                                |
| <b>lb-vip-create</b>           | Create a vip.                                       |
| <b>lb-vip-delete</b>           | Delete a given vip.                                 |
| <b>lb-vip-list</b>             | List vips that belong to a given tenant.            |
| <b>lb-vip-show</b>             | Show information of a given vip.                    |
| <b>lb-vip-update</b>           | Update a given vip.                                 |
| <b>meter-label-create</b>      | Create a metering label for a given tenant.         |
| <b>meter-label-delete</b>      | Delete a given metering label.                      |
| <b>meter-label-list</b>        | List metering labels that belong to a given tenant. |
| <b>meter-label-rule-create</b> | Create a metering label rule for a given label.     |
| <b>meter-label-rule-delete</b> | Delete a given metering label.                      |
| <b>meter-label-rule-list</b>   | List metering labels that belong to a given label.  |
| <b>meter-label-rule-show</b>   | Show information of a given metering label rule.    |

---

|                               |                                                       |
|-------------------------------|-------------------------------------------------------|
| <b>meter-label-show</b>       | Show information of a given metering label.           |
| <b>net-create</b>             | Create a network for a given tenant.                  |
| <b>net-delete</b>             | Delete a given network.                               |
| <b>net-external-list</b>      | List external networks that belong to a given tenant. |
| <b>net-gateway-connect</b>    | Add an internal network interface to a router.        |
| <b>net-gateway-create</b>     | Create a network gateway.                             |
| <b>net-gateway-delete</b>     | Delete a given network gateway.                       |
| <b>net-gateway-disconnect</b> | Remove a network from a network gateway.              |
| <b>net-gateway-list</b>       | List network gateways for a given tenant.             |
| <b>net-gateway-show</b>       | Show information of a given network gateway.          |
| <b>net-gateway-update</b>     | Update the name for a network gateway.                |
| <b>net-list</b>               | List networks that belong to a given tenant.          |
| <b>net-list-on-dhcp-agent</b> | List the networks on a DHCP agent.                    |
| <b>net-show</b>               | Show information of a given network.                  |
| <b>net-update</b>             | Update network's information.                         |
| <b>port-create</b>            | Create a port for a given tenant.                     |
| <b>port-delete</b>            | Delete a given port.                                  |

---

|                                |                                                               |
|--------------------------------|---------------------------------------------------------------|
| <b>port-list</b>               | List ports that belong to a given tenant.                     |
| <b>port-show</b>               | Show information of a given port.                             |
| <b>port-update</b>             | Update port's information.                                    |
| <b>queue-create</b>            | Create a queue.                                               |
| <b>queue-delete</b>            | Delete a given queue.                                         |
| <b>queue-list</b>              | List queues that belong to a given tenant.                    |
| <b>queue-show</b>              | Show information of a given queue.                            |
| <b>quota-delete</b>            | Delete defined quotas of a given tenant.                      |
| <b>quota-list</b>              | List quotas of all tenants who have non-default quota values. |
| <b>quota-show</b>              | Show quotas of a given tenant                                 |
| <b>quota-update</b>            | Define tenant's quotas not to use defaults.                   |
| <b>router-create</b>           | Create a router for a given tenant.                           |
| <b>router-delete</b>           | Delete a given router.                                        |
| <b>router-gateway-clear</b>    | Remove an external network gateway from a router.             |
| <b>router-gateway-set</b>      | Set the external network gateway for a router.                |
| <b>router-interface-add</b>    | Add an internal network interface to a router.                |
| <b>router-interface-delete</b> | Remove an internal network interface from a router.           |

---

|                                   |                                                                  |
|-----------------------------------|------------------------------------------------------------------|
| <b>router-list</b>                | List routers that belong to a given tenant.                      |
| <b>router-list-on-l3-agent</b>    | List the routers on a L3 agent.                                  |
| <b>router-port-list</b>           | List ports that belong to a given tenant, with specified router. |
| <b>router-show</b>                | Show information of a given router.                              |
| <b>router-update</b>              | Update router's information.                                     |
| <b>security-group-create</b>      | Create a security group.                                         |
| <b>security-group-delete</b>      | Delete a given security group.                                   |
| <b>security-group-list</b>        | List security groups that belong to a given tenant.              |
| <b>security-group-rule-create</b> | Create a security group rule.                                    |
| <b>security-group-rule-delete</b> | Delete a given security group rule.                              |
| <b>security-group-rule-list</b>   | List security group rules that belong to a given tenant.         |
| <b>security-group-rule-show</b>   | Show information of a given security group rule.                 |
| <b>security-group-show</b>        | Show information of a given security group.                      |
| <b>security-group-update</b>      | Update a given security group.                                   |
| <b>service-provider-list</b>      | List service providers.                                          |
| <b>subnet-create</b>              | Create a subnet for a given tenant.                              |
| <b>subnet-delete</b>              | Delete a given subnet.                                           |

---

|                               |                                                               |
|-------------------------------|---------------------------------------------------------------|
| <b>subnet-list</b>            | List subnets that belong to a given tenant.                   |
| <b>subnet-show</b>            | Show information of a given subnet.                           |
| <b>subnet-update</b>          | Update subnet's information.                                  |
| <b>vpn-ikepolicy-create</b>   | Create an IKEPolicy.                                          |
| <b>vpn-ikepolicy-delete</b>   | Delete a given IKE Policy.                                    |
| <b>vpn-ikepolicy-list</b>     | List IKEPolicies that belong to a tenant.                     |
| <b>vpn-ikepolicy-show</b>     | Show information of a given IKEPolicy.                        |
| <b>vpn-ikepolicy-update</b>   | Update a given IKE Policy.                                    |
| <b>vpn-ipsecpolicy-create</b> | Create an ipsecpolicy.                                        |
| <b>vpn-ipsecpolicy-delete</b> | Delete a given ipsecpolicy.                                   |
| <b>vpn-ipsecpolicy-list</b>   | List ipsecpolicies that belongs to a given tenant connection. |
| <b>vpn-ipsecpolicy-show</b>   | Show information of a given ipsecpolicy.                      |
| <b>vpn-ipsecpolicy-update</b> | Update a given ipsec policy.                                  |
| <b>vpn-service-create</b>     | Create a VPNService.                                          |
| <b>vpn-service-delete</b>     | Delete a given VPNService.                                    |
| <b>vpn-service-list</b>       | List VPNService configurations that belong to a given tenant. |
| <b>vpn-service-show</b>       | Show information of a given VPNService.                       |

**vpn-service-update**

Update a given VPNService.

**neutron agent-delete command**

```
usage: neutron agent-delete [-h] [--request-format {json,xml}] AGENT
```

Delete a given agent.

**Positional arguments****AGENT** ID of agent to delete**Optional arguments****-h, --help** show this help message and exit**--request-format {json,xml}** The xml or json request format**neutron agent-list command**

```
usage: neutron agent-list [-h] [-f {csv,table}] [-c COLUMN]
                        [--quote {all,minimal,none,nonnumeric}]
                        [--request-format {json,xml}] [-D] [-F FIELD]
```

List agents.

**Optional arguments****-h, --help** show this help message and exit**--request-format {json,xml}** The xml or json request format



|                                |                                                                |
|--------------------------------|----------------------------------------------------------------|
| <b>-D, --show-details</b>      | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b> | Specify the field(s) to be returned by server, can be repeated |

## neutron agent-show command

```
usage: neutron agent-show [-h] [-f {shell,table}] [-c COLUMN]
                          [--variable VARIABLE] [--prefix PREFIX]
                          [--request-format {json,xml}] [-D] [-F FIELD]
                          AGENT
```

Show information of a given agent.

### Positional arguments

**AGENT** ID of agent to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron agent-update command

```
usage: neutron agent-update [-h] [--request-format {json,xml}] AGENT
```

Update a given agent.

### Positional arguments

**AGENT** ID or name of agent to update

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron cisco-credential-create command

```
usage: neutron cisco-credential-create [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}]
                                     [--tenant-id TENANT_ID]
                                     [--username USERNAME]
                                     [--password PASSWORD]
                                     credential_name credential_type
```

Creates a credential.

### Positional arguments

**credential\_name** Name/Ip address for Credential

**credential\_type** Type of the Credential

### Optional arguments

**-h, --help** show this help message and exit

|                                   |                                |
|-----------------------------------|--------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format |
| <b>-tenant-id TENANT_ID</b>       | The owner tenant ID            |
| <b>-username USERNAME</b>         | Username for the credential    |
| <b>-password PASSWORD</b>         | Password for the credential    |

## neutron cisco-credential-delete command

```
usage: neutron cisco-credential-delete [-h] [--request-format {json,xml}]
                                         CREDENTIAL
```

Delete a given credential.

### Positional arguments

**CREDENTIAL** ID of credential to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron cisco-credential-list command

```
usage: neutron cisco-credential-list [-h] [-f {csv,table}] [-c COLUMN]
                                       [--quote {all,minimal,none,nonnumeric}]
                                       [--request-format {json,xml}] [-D]
                                       [-F FIELD]
```

List credentials that belong to a given tenant.

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron cisco-credential-show command

```
usage: neutron cisco-credential-show [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
                                     CREDENTIAL
```

Show information of a given credential.

### Positional arguments

**CREDENTIAL** ID of credential to look up

### Optional arguments

|                   |                                 |
|-------------------|---------------------------------|
| <b>-h, --help</b> | show this help message and exit |
|-------------------|---------------------------------|

|                                   |                                                                |
|-----------------------------------|----------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>         | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated |

## neutron cisco-network-profile-create command

```
usage: neutron cisco-network-profile-create [-h] [-f {shell,table}]
                                           [-c COLUMN] [--variable VARIABLE]
                                           [--prefix PREFIX]
                                           [--request-format {json,xml}]
                                           [--tenant-id TENANT_ID]
                                           [--sub_type SUB_TYPE]
                                           [--segment_range SEGMENT_RANGE]
                                           [--physical_network PHYSICAL_NETWORK]
                                           [--multicast_ip_range MULTICAST_IP_RANGE]
                                           [--add-tenant ADD_TENANT]
                                           name
                                           {vlan,overlay,multi-segment,trunk}
```

Creates a network profile.

### Positional arguments

|                                           |                          |
|-------------------------------------------|--------------------------|
| <b>name</b>                               | Name for Network Profile |
| <b>{vlan,overlay,multi-segment,trunk}</b> | Segment type             |

### Optional arguments

|                   |                                 |
|-------------------|---------------------------------|
| <b>-h, --help</b> | show this help message and exit |
|-------------------|---------------------------------|

|                                                   |                                                                                                                          |
|---------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|
| <b>-request-format {json,xml}</b>                 | The xml or json request format                                                                                           |
| <b>-tenant-id TENANT_ID</b>                       | The owner tenant ID                                                                                                      |
| <b>-sub_type SUB_TYPE</b>                         | Sub-type for the segment. Available sub-types for overlay segments: native, enhanced; For trunk segments: vlan, overlay. |
| <b>-segment_range<br/>SEGMENT_RANGE</b>           | Range for the Segment                                                                                                    |
| <b>-physical_network<br/>PHYSICAL_NETWORK</b>     | Name for the Physical Network                                                                                            |
| <b>-multicast_ip_range<br/>MULTICAST_IP_RANGE</b> | Multicast IPv4 Range                                                                                                     |
| <b>-add-tenant ADD_TENANT</b>                     | Add tenant to the network profile                                                                                        |

## neutron cisco-network-profile-delete command

```
usage: neutron cisco-network-profile-delete [-h] [--request-format {json,xml}]  
                                             NETWORK_PROFILE
```

Delete a given network profile.

### Positional arguments

**NETWORK\_PROFILE** ID or name of network\_profile to delete

### Optional arguments

**-h, -help** show this help message and exit

**-request-format {json,xml}**      The xml or json request format

## neutron cisco-network-profile-list command

```
usage: neutron cisco-network-profile-list [-h] [-f {csv,table}] [-c COLUMN]
                                          [--quote {all,minimal,none,nonnumeric}]
                                          [--request-format {json,xml}] [-D]
                                          [-F FIELD]
```

List network profiles that belong to a given tenant.

### Optional arguments

**-h, -help**      show this help message and exit

**-request-format {json,xml}**      The xml or json request format

**-D, --show-details**      Show detailed info

**-F FIELD, --field FIELD**      Specify the field(s) to be returned by server, can be repeated

## neutron cisco-network-profile-show command

```
usage: neutron cisco-network-profile-show [-h] [-f {shell,table}] [-c COLUMN]
                                          [--variable VARIABLE]
                                          [--prefix PREFIX]
                                          [--request-format {json,xml}] [-D]
                                          [-F FIELD]
                                          NETWORK_PROFILE
```

Show information of a given network profile.

### Positional arguments

**NETWORK\_PROFILE** ID or name of network\_profile to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron cisco-network-profile-update command

```
usage: neutron cisco-network-profile-update [-h] [--request-format {json,xml}]
                                           NETWORK_PROFILE
```

Update network profile's information.

### Positional arguments

**NETWORK\_PROFILE** ID or name of network\_profile to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |



## neutron cisco-policy-profile-list command

```
usage: neutron cisco-policy-profile-list [-h] [-f {csv,table}] [-c COLUMN]
                                         [--quote {all,minimal,none,nonnumeric}]
                                         [--request-format {json,xml}] [-D]
                                         [-F FIELD]
```

List policy profiles that belong to a given tenant.

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron cisco-policy-profile-show command

```
usage: neutron cisco-policy-profile-show [-h] [-f {shell,table}] [-c COLUMN]
                                         [--variable VARIABLE]
                                         [--prefix PREFIX]
                                         [--request-format {json,xml}] [-D]
                                         [-F FIELD]
                                         POLICY_PROFILE
```

Show information of a given policy profile.

### Positional arguments

**POLICY\_PROFILE** ID or name of policy\_profile to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron cisco-policy-profile-update command

```
usage: neutron cisco-policy-profile-update [-h] [--request-format {json,xml}]
                                           POLICY_PROFILE
```

Update policy profile's information.

### Positional arguments

**POLICY\_PROFILE** ID or name of policy\_profile to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron dhcp-agent-list-hosting-net command

```
usage: neutron dhcp-agent-list-hosting-net [-h] [-f {csv,table}] [-c COLUMN]
                                           [--quote {all,minimal,none,nonnumeric}]
```

```
[--request-format {json,xml}] [-D]  
[-F FIELD]  
network
```

List DHCP agents hosting a network.

### Positional arguments

**network**    Network to query

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron dhcp-agent-network-add command

```
usage: neutron dhcp-agent-network-add [-h] [--request-format {json,xml}]  
dhcp_agent network
```

Add a network to a DHCP agent.

### Positional arguments

**dhcp\_agent**    ID of the DHCP agent

**network**      Network to add

### Optional arguments

**-h, --help**      show this help message and exit

**--request-format {json,xml}**      The xml or json request format

## neutron dhcp-agent-network-remove command

```
usage: neutron dhcp-agent-network-remove [-h] [--request-format {json,xml}]
                                          dhcp_agent network
```

Remove a network from a DHCP agent.

### Positional arguments

**dhcp\_agent**      ID of the DHCP agent

**network**      Network to remove

### Optional arguments

**-h, --help**      show this help message and exit

**--request-format {json,xml}**      The xml or json request format

## neutron ext-list command

```
usage: neutron ext-list [-h] [-f {csv,table}] [-c COLUMN]
                        [--quote {all,minimal,none,nonnumeric}]
                        [--request-format {json,xml}] [-D] [-F FIELD]
```

List all extensions.

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron ext-show command

```
usage: neutron ext-show [-h] [-f {shell,table}] [-c COLUMN]
                        [--variable VARIABLE] [--prefix PREFIX]
                        [--request-format {json,xml}] [-D] [-F FIELD]
                        EXT-ALIAS
```

Show information of a given resource.

### Positional arguments

**EXT-ALIAS**    The extension alias

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

|                                |                                                                |
|--------------------------------|----------------------------------------------------------------|
| <b>-D, --show-details</b>      | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b> | Specify the field(s) to be returned by server, can be repeated |

## neutron firewall-create command

```
usage: neutron firewall-create [-h] [-f {shell,table}] [-c COLUMN]
                               [--variable VARIABLE] [--prefix PREFIX]
                               [--request-format {json,xml}]
                               [--tenant-id TENANT_ID] [--name NAME]
                               [--description DESCRIPTION] [--shared]
                               [--admin-state-down]
                               POLICY
```

Create a firewall.

### Positional arguments

**POLICY** Firewall policy id

### Optional arguments

|                                    |                                    |
|------------------------------------|------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit    |
| <b>--request-format {json,xml}</b> | The xml or json request format     |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID                |
| <b>--name NAME</b>                 | Name for the firewall              |
| <b>--description DESCRIPTION</b>   | Description for the firewall rule  |
| <b>--shared</b>                    | Set shared to True (default False) |

**-admin-state-down**                      Set admin state up to false

## neutron firewall-delete command

```
usage: neutron firewall-delete [-h] [--request-format {json,xml}] FIREWALL
```

Delete a given firewall.

### Positional arguments

**FIREWALL**    ID or name of firewall to delete

### Optional arguments

**-h, -help**                              show this help message and exit

**-request-format {json,xml}**            The xml or json request format

## neutron firewall-list command

```
usage: neutron firewall-list [-h] [-f {csv,table}] [-c COLUMN]
                             [--quote {all,minimal,none,nonnumeric}]
                             [--request-format {json,xml}] [-D] [-F FIELD]
                             [-P SIZE] [--sort-key FIELD]
                             [--sort-dir {asc,desc}]
```

List firewalls that belong to a given tenant.

### Optional arguments

**-h, -help**                              show this help message and exit

|                                   |                                                                                                                                                                                                                      |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>         | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>  | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>           | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>      | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron firewall-policy-create command

```
usage: neutron firewall-policy-create [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}]
                                     [--tenant-id TENANT_ID]
                                     [--description DESCRIPTION] [--shared]
                                     [--firewall-rules FIREWALL_RULES]
                                     [--audited]
                                     NAME
```

Create a firewall policy.

### Positional arguments

**NAME** Name for the firewall policy



### Optional arguments

|                                            |                                                                                                           |
|--------------------------------------------|-----------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                          | show this help message and exit                                                                           |
| <b>--request-format {json,xml}</b>         | The xml or json request format                                                                            |
| <b>--tenant-id TENANT_ID</b>               | The owner tenant ID                                                                                       |
| <b>--description DESCRIPTION</b>           | Description for the firewall policy                                                                       |
| <b>--shared</b>                            | To create a shared policy                                                                                 |
| <b>--firewall-rules<br/>FIREWALL_RULES</b> | Ordered list of whitespace-delimited firewall rule names or IDs; e.g., --<br>firewall-rules "rule1 rule2" |
| <b>--audited</b>                           | To set audited to True                                                                                    |

### neutron firewall-policy-delete command

```
usage: neutron firewall-policy-delete [-h] [--request-format {json,xml}]  
                                     FIREWALL_POLICY
```

Delete a given firewall policy.

### Positional arguments

**FIREWALL\_POLICY** ID or name of firewall\_policy to delete

### Optional arguments

|                   |                                 |
|-------------------|---------------------------------|
| <b>-h, --help</b> | show this help message and exit |
|-------------------|---------------------------------|

**-request-format {json,xml}**      The xml or json request format

## neutron firewall-policy-insert-rule command

```
usage: neutron firewall-policy-insert-rule [-h] [--request-format {json,xml}]
                                           [--insert-before FIREWALL_RULE]
                                           [--insert-after FIREWALL_RULE]
                                           FIREWALL_POLICY FIREWALL_RULE
```

Insert a rule into a given firewall policy.

### Positional arguments

**FIREWALL\_POLICY**      ID or name of firewall\_policy to update

**FIREWALL\_RULE**      New rule to insert

### Optional arguments

**-h, -help**      show this help message and exit

**-request-format {json,xml}**      The xml or json request format

**-insert-before FIREWALL\_RULE**      Insert before this rule

**-insert-after FIREWALL\_RULE**      Insert after this rule

## neutron firewall-policy-list command

```
usage: neutron firewall-policy-list [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
```

```
[--request-format {json,xml}] [-D]  
[-F FIELD] [-P SIZE] [--sort-key FIELD]  
[--sort-dir {asc,desc}]
```

List firewall policies that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron firewall-policy-remove-rule command

```
usage: neutron firewall-policy-remove-rule [-h] [--request-format {json,xml}]  
FIREWALL_POLICY FIREWALL_RULE
```

Remove a rule from a given firewall policy.

### Positional arguments

**FIREWALL\_POLICY** ID or name of firewall\_policy to update

**FIREWALL\_RULE** Firewall rule to remove from policy

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron firewall-policy-show command

```
usage: neutron firewall-policy-show [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
                                     FIREWALL_POLICY
```

Show information of a given firewall policy.

### Positional arguments

**FIREWALL\_POLICY** ID or name of firewall\_policy to look up

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

|                                |                                                                |
|--------------------------------|----------------------------------------------------------------|
| <b>-D, --show-details</b>      | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b> | Specify the field(s) to be returned by server, can be repeated |

## neutron firewall-policy-update command

```
usage: neutron firewall-policy-update [-h] [--request-format {json,xml}]
                                     FIREWALL_POLICY
```

Update a given firewall policy.

### Positional arguments

**FIREWALL\_POLICY** ID or name of firewall\_policy to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron firewall-rule-create command

```
usage: neutron firewall-rule-create [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}]
                                     [--tenant-id TENANT_ID] [--name NAME]
                                     [--description DESCRIPTION] [--shared]
                                     [--source-ip-address SOURCE_IP_ADDRESS]
                                     [--destination-ip-address DESTINATION_IP_ADDRESS]
                                     [--source-port SOURCE_PORT]
                                     [--destination-port DESTINATION_PORT]
```

```
[--disabled] --protocol {tcp,udp,icmp,any}  
--action {allow,deny}
```

Create a firewall rule.

### Optional arguments

|                                                           |                                                          |
|-----------------------------------------------------------|----------------------------------------------------------|
| <b>-h, -help</b>                                          | show this help message and exit                          |
| <b>-request-format {json,xml}</b>                         | The xml or json request format                           |
| <b>-tenant-id TENANT_ID</b>                               | The owner tenant ID                                      |
| <b>-name NAME</b>                                         | Name for the firewall rule                               |
| <b>-description DESCRIPTION</b>                           | Description for the firewall rule                        |
| <b>-shared</b>                                            | Set shared to True (default False)                       |
| <b>-source-ip-address<br/>SOURCE_IP_ADDRESS</b>           | Source ip address or subnet                              |
| <b>-destination-ip-address<br/>DESTINATION_IP_ADDRESS</b> | Destination ip address or subnet                         |
| <b>-source-port SOURCE_PORT</b>                           | Source port (integer in [1, 65535] or range in a:b)      |
| <b>-destination-port<br/>DESTINATION_PORT</b>             | Destination port (integer in [1, 65535] or range in a:b) |
| <b>-disabled</b>                                          | To disable this rule                                     |

**-protocol {tcp,udp,icmp,any}** Protocol for the firewall rule

**-action {allow,deny}** Action for the firewall rule

## neutron firewall-rule-delete command

```
usage: neutron firewall-rule-delete [-h] [--request-format {json,xml}]
                                     FIREWALL_RULE
```

Delete a given firewall rule.

### Positional arguments

**FIREWALL\_RULE** ID or name of firewall\_rule to delete

### Optional arguments

**-h, -help** show this help message and exit

**-request-format {json,xml}** The xml or json request format

## neutron firewall-rule-list command

```
usage: neutron firewall-rule-list [-h] [-f {csv,table}] [-c COLUMN]
                                   [--quote {all,minimal,none,nonnumeric}]
                                   [--request-format {json,xml}] [-D]
                                   [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                   [--sort-dir {asc,desc}]
```

List firewall rules that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron firewall-rule-show command

```
usage: neutron firewall-rule-show [-h] [-f {shell,table}] [-c COLUMN]
                                  [--variable VARIABLE] [--prefix PREFIX]
                                  [--request-format {json,xml}] [-D]
                                  [-F FIELD]
                                  FIREWALL_RULE
```

Show information of a given firewall rule.

### Positional arguments

**FIREWALL\_RULE** ID or name of firewall\_rule to look up



### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron firewall-rule-update command

```
usage: neutron firewall-rule-update [-h] [--request-format {json,xml}]
                                     [--protocol {tcp,udp,icmp,any}]
                                     FIREWALL_RULE
```

Update a given firewall rule.

### Positional arguments

**FIREWALL\_RULE** ID or name of firewall\_rule to update

### Optional arguments

|                                      |                                 |
|--------------------------------------|---------------------------------|
| <b>-h, --help</b>                    | show this help message and exit |
| <b>--request-format {json,xml}</b>   | The xml or json request format  |
| <b>--protocol {tcp,udp,icmp,any}</b> | Protocol for the firewall rule  |

## neutron firewall-show command

```
usage: neutron firewall-show [-h] [-f {shell,table}] [-c COLUMN]
```

```
[--variable VARIABLE] [--prefix PREFIX]
[--request-format {json,xml}] [-D] [-F FIELD]
FIREWALL
```

Show information of a given firewall.

### Positional arguments

**FIREWALL** ID or name of firewall to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron firewall-update command

```
usage: neutron firewall-update [-h] [--request-format {json,xml}] FIREWALL
```

Update a given firewall.

### Positional arguments

**FIREWALL** ID or name of firewall to update

### Optional arguments

- h, --help** show this help message and exit
- request-format {json,xml}** The xml or json request format

## neutron floatingip-associate command

```
usage: neutron floatingip-associate [-h] [--request-format {json,xml}]
                                     [--fixed-ip-address FIXED_IP_ADDRESS]
                                     FLOATINGIP_ID PORT
```

Create a mapping between a floating ip and a fixed ip.

### Positional arguments

- FLOATINGIP\_ID** ID of the floating IP to associate
- PORT** ID or name of the port to be associated with the floatingip

### Optional arguments

- h, --help** show this help message and exit
- request-format {json,xml}** The xml or json request format
- fixed-ip-address  
FIXED\_IP\_ADDRESS** IP address on the port (only required if port has multiple IPs)

## neutron floatingip-create command

```
usage: neutron floatingip-create [-h] [-f {shell,table}] [-c COLUMN]
```

```
[--variable VARIABLE] [--prefix PREFIX]
[--request-format {json,xml}]
[--tenant-id TENANT_ID] [--port-id PORT_ID]
[--fixed-ip-address FIXED_IP_ADDRESS]
FLOATING_NETWORK
```

Create a floating ip for a given tenant.

### Positional arguments

**FLOATING\_NETWORK** Network name or id to allocate floating IP from

### Optional arguments

|                                                |                                                                |
|------------------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                              | show this help message and exit                                |
| <b>--request-format {json,xml}</b>             | The xml or json request format                                 |
| <b>--tenant-id TENANT_ID</b>                   | The owner tenant ID                                            |
| <b>--port-id PORT_ID ID</b>                    | of the port to be associated with the floatingip               |
| <b>--fixed-ip-address<br/>FIXED_IP_ADDRESS</b> | IP address on the port (only required if port has multipleIPs) |

## neutron floatingip-delete command

```
usage: neutron floatingip-delete [-h] [--request-format {json,xml}] FLOATINGIP
```

Delete a given floating ip.

### Positional arguments

**FLOATINGIP** ID of floatingip to delete

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron floatingip-disassociate command

```
usage: neutron floatingip-disassociate [-h] [--request-format {json,xml}]
                                         FLOATINGIP_ID
```

Remove a mapping from a floating ip to a fixed ip.

### Positional arguments

**FLOATINGIP\_ID** ID of the floating IP to associate

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron floatingip-list command

```
usage: neutron floatingip-list [-h] [-f {csv,table}] [-c COLUMN]
                                [--quote {all,minimal,none,nonnumeric}]
```

```
[--request-format {json,xml}] [-D] [-F FIELD]
[-P SIZE] [--sort-key FIELD]
[--sort-dir {asc,desc}]
```

List floating ips that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron floatingip-show command

```
usage: neutron floatingip-show [-h] [-f {shell,table}] [-c COLUMN]
                                [--variable VARIABLE] [--prefix PREFIX]
                                [--request-format {json,xml}] [-D] [-F FIELD]
                                FLOATINGIP
```

Show information of a given floating ip.

### Positional arguments

**FLOATINGIP** ID of floatingip to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron ipsec-site-connection-create command

```
usage: neutron ipsec-site-connection-create [-h] [-f {shell,table}]
                                             [-c COLUMN] [--variable VARIABLE]
                                             [--prefix PREFIX]
                                             [--request-format {json,xml}]
                                             [--tenant-id TENANT_ID]
                                             [--admin-state-down] [--name NAME]
                                             [--description DESCRIPTION]
                                             [--mtu MTU]
                                             [--initiator {bi-directional,response-only}]
                                             [--dpd action=ACTION,interval=INTERVAL,timeout=
TIMEOUT]
                                             --vpnservice-id VPNSERVICE
                                             --ikepolicy-id IKEPOLICY
                                             --ipsecpolicy-id IPSECPOLICY
                                             --peer-address PEER_ADDRESS
                                             --peer-id PEER_ID --peer-cidr
```

```
PEER_CIDRS --psk PSK
```

Create an IPsecSiteConnection.

### Optional arguments

|                                                   |                                                                                                                                                                                                                                                                                                                                   |
|---------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                                 | show this help message and exit                                                                                                                                                                                                                                                                                                   |
| <b>--request-format {json,xml}</b>                | The xml or json request format                                                                                                                                                                                                                                                                                                    |
| <b>--tenant-id TENANT_ID</b>                      | The owner tenant ID                                                                                                                                                                                                                                                                                                               |
| <b>--admin-state-down</b>                         | Set admin state up to false                                                                                                                                                                                                                                                                                                       |
| <b>--name NAME</b>                                | Set friendly name for the connection                                                                                                                                                                                                                                                                                              |
| <b>--description DESCRIPTION</b>                  | Set a description for the connection                                                                                                                                                                                                                                                                                              |
| <b>--mtu MTU MTU</b>                              | size for the connection, default:1500                                                                                                                                                                                                                                                                                             |
| <b>--initiator {bi-directional,response-only}</b> | Initiator state in lowercase, default:bi-directional                                                                                                                                                                                                                                                                              |
| <b>--dpd</b>                                      | action=ACTION,interval=INTERVAL,timeout=TIMEOUT Ipsec connection Dead Peer Detection Attributes. 'action'-hold,clear,disabled,restart,restart-by-peer. 'interval' and 'timeout' are non negative integers. 'interval' should be less than 'timeout' value. 'action', default:hold 'interval', default:30, 'timeout', default:120. |
| <b>--vpnservice-id VPNSERVICE</b>                 | VPNService instance id associated with this connection                                                                                                                                                                                                                                                                            |
| <b>--ikepolicy-id IKEPOLICY</b>                   | IKEPolicy id associated with this connection                                                                                                                                                                                                                                                                                      |



|                                    |                                                                                                     |
|------------------------------------|-----------------------------------------------------------------------------------------------------|
| <b>-ipsecpolicy-id IPSECPOLICY</b> | IPsecPolicy id associated with this connection                                                      |
| <b>-peer-address PEER_ADDRESS</b>  | Peer gateway public IPv4/IPv6 address or FQDN.                                                      |
| <b>-peer-id PEER_ID</b>            | Peer router identity for authentication. Can be IPv4/IPv6 address, e-mail address, key id, or FQDN. |
| <b>-peer-cidr PEER_CIDRS</b>       | Remote subnet(s) in CIDR format                                                                     |
| <b>-psk PSK</b>                    | Pre-Shared Key string                                                                               |

## neutron ipsec-site-connection-delete command

```
usage: neutron ipsec-site-connection-delete [-h] [--request-format {json,xml}]
                                             IPSEC_SITE_CONNECTION
```

Delete a given IPsecSiteConnection.

### Positional arguments

**IPSEC\_SITE\_CONNECTION** ID or name of ipsec\_site\_connection to delete

### Optional arguments

**-h, -help** show this help message and exit

**-request-format {json,xml}** The xml or json request format

## neutron ipsec-site-connection-list command

```
usage: neutron ipsec-site-connection-list [-h] [-f {csv,table}] [-c COLUMN]
                                           [--quote {all,minimal,none,nonnumeric}]
```

```
[--request-format {json,xml}] [-D]
[-F FIELD] [-P SIZE]
[--sort-key FIELD]
[--sort-dir {asc,desc}]
```

List IPsecSiteConnections that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron ipsec-site-connection-show command

```
usage: neutron ipsec-site-connection-show [-h] [-f {shell,table}] [-c COLUMN]
                                           [--variable VARIABLE]
                                           [--prefix PREFIX]
                                           [--request-format {json,xml}] [-D]
```

```
[ -F FIELD]  
IPSEC_SITE_CONNECTION
```

Show information of a given IPsecSiteConnection.

### Positional arguments

**IPSEC\_SITE\_CONNECTION** ID or name of ipsec\_site\_connection to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron ipsec-site-connection-update command

```
usage: neutron ipsec-site-connection-update [-h] [--request-format {json,xml}]  
                                           [--dpd action=ACTION,interval=INTERVAL,timeout=  
TIMEOUT]  
                                           IPSEC_SITE_CONNECTION
```

Update a given IPsecSiteConnection.

### Positional arguments

**IPSEC\_SITE\_CONNECTION** ID or name of ipsec\_site\_connection to update

### Optional arguments

|                                    |                                                                                                                                                                                                                                                                                                                                  |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                                                                                                                                  |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                                                                                                                                   |
| <b>-dpd</b>                        | action=ACTION,interval=INTERVAL,timeout=TIMEOUT Isec connection Dead Peer Detection Attributes. 'action'-hold,clear,disabled,restart,restart-by-peer. 'interval' and 'timeout' are non negative integers. 'interval' should be less than 'timeout' value. 'action', default:hold 'interval', default:30, 'timeout', default:120. |

### neutron l3-agent-list-hosting-router command

```
usage: neutron l3-agent-list-hosting-router [-h] [-f {csv,table}] [-c COLUMN]
                                           [--quote {all,minimal,none,nonnumeric}]
                                           [--request-format {json,xml}] [-D]
                                           [-F FIELD]
                                           router
```

List L3 agents hosting a router.

### Positional arguments

**router** Router to query

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

|                                |                                                                |
|--------------------------------|----------------------------------------------------------------|
| <b>-D, --show-details</b>      | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b> | Specify the field(s) to be returned by server, can be repeated |

## neutron l3-agent-router-add command

```
usage: neutron l3-agent-router-add [-h] [--request-format {json,xml}]
                                   l3_agent router
```

Add a router to a L3 agent.

### Positional arguments

**l3\_agent**     ID of the L3 agent

**router**       Router to add

### Optional arguments

**-h, --help**                    show this help message and exit

**--request-format {json,xml}**    The xml or json request format

## neutron l3-agent-router-remove command

```
usage: neutron l3-agent-router-remove [-h] [--request-format {json,xml}]
                                       l3_agent router
```

Remove a router from a L3 agent.

### Positional arguments

**l3\_agent** ID of the L3 agent

**router** Router to remove

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron lb-agent-hosting-pool command

```
usage: neutron lb-agent-hosting-pool [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
                                     pool
```

Get loadbalancer agent hosting a pool. Deriving from ListCommand though server will return only one agent to keep common output format for all agent schedulers

### Positional arguments

**pool** Pool to query

### Optional arguments

**-h, --help** show this help message and exit

|                                   |                                                                |
|-----------------------------------|----------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>         | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated |

## neutron lb-healthmonitor-associate command

```
usage: neutron lb-healthmonitor-associate [-h] [--request-format {json,xml}]
                                          HEALTH_MONITOR_ID POOL
```

Create a mapping between a health monitor and a pool.

### Positional arguments

**HEALTH\_MONITOR\_ID** Health monitor to associate

**POOL** ID of the pool to be associated with the health monitor

### Optional arguments

**-h, --help** show this help message and exit

**-request-format {json,xml}** The xml or json request format

## neutron lb-healthmonitor-create command

```
usage: neutron lb-healthmonitor-create [-h] [-f {shell,table}] [-c COLUMN]
                                         [--variable VARIABLE] [--prefix PREFIX]
                                         [--request-format {json,xml}]
```

```
[--tenant-id TENANT_ID]
[--admin-state-down]
[--expected-codes EXPECTED_CODES]
[--http-method HTTP_METHOD]
[--url-path URL_PATH] --delay DELAY
--max-retries MAX_RETRIES --timeout
TIMEOUT --type {PING,TCP,HTTP,HTTPS}
```

Create a healthmonitor.

### Optional arguments

|                                            |                                                                                                                                                                                                                                                                          |
|--------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                          | show this help message and exit                                                                                                                                                                                                                                          |
| <b>--request-format {json,xml}</b>         | The xml or json request format                                                                                                                                                                                                                                           |
| <b>--tenant-id TENANT_ID</b>               | The owner tenant ID                                                                                                                                                                                                                                                      |
| <b>--admin-state-down</b>                  | Set admin state up to false                                                                                                                                                                                                                                              |
| <b>--expected-codes<br/>EXPECTED_CODES</b> | The list of HTTP status codes expected in response from the member to declare it healthy. This attribute can contain one value, or a list of values separated by comma, or a range of values (e.g. "200-299"). If this attribute is not specified, it defaults to "200". |
| <b>--http-method HTTP_METHOD</b>           | The HTTP method used for requests by the monitor of type HTTP.                                                                                                                                                                                                           |
| <b>--url-path URL_PATH</b>                 | The HTTP path used in the HTTP request used by the monitor to test a member health. This must be a string beginning with a / (forward slash)                                                                                                                             |
| <b>--delay DELAY</b>                       | The time in seconds between sending probes to members.                                                                                                                                                                                                                   |



|                                    |                                                                                                                                                      |
|------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-max-retries MAX_RETRIES</b>    | Number of permissible connection failures before changing the member status to INACTIVE. [1..10]                                                     |
| <b>-timeout TIMEOUT</b>            | Maximum number of seconds for a monitor to wait for a connection to be established before it times out. The value must be less than the delay value. |
| <b>-type {PING,TCP,HTTP,HTTPS}</b> | One of predefined health monitor types                                                                                                               |

## neutron lb-healthmonitor-delete command

```
usage: neutron lb-healthmonitor-delete [-h] [--request-format {json,xml}]
                                         HEALTH_MONITOR
```

Delete a given healthmonitor.

### Positional arguments

**HEALTH\_MONITOR** ID or name of health\_monitor to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron lb-healthmonitor-disassociate command

```
usage: neutron lb-healthmonitor-disassociate [-h]
                                              [--request-format {json,xml}]
                                              HEALTH_MONITOR_ID POOL
```

Remove a mapping from a health monitor to a pool.

### Positional arguments

**HEALTH\_MONITOR\_ID** Health monitor to associate

**POOL** ID of the pool to be associated with the health monitor

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron lb-healthmonitor-list command

```
usage: neutron lb-healthmonitor-list [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                     [--sort-dir {asc,desc}]
```

List healthmonitors that belong to a given tenant.

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

|                                  |                                                                                                                                                                                                                      |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-D, --show-details</b>        | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>   | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b> | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>          | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>     | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron lb-healthmonitor-show command

```
usage: neutron lb-healthmonitor-show [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
                                     HEALTH_MONITOR
```

Show information of a given healthmonitor.

### Positional arguments

**HEALTH\_MONITOR** ID or name of health\_monitor to look up

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

- |                                |                                                                |
|--------------------------------|----------------------------------------------------------------|
| <b>-D, --show-details</b>      | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b> | Specify the field(s) to be returned by server, can be repeated |

## neutron lb-healthmonitor-update command

```
usage: neutron lb-healthmonitor-update [-h] [--request-format {json,xml}]
                                         HEALTH_MONITOR
```

Update a given healthmonitor.

### Positional arguments

**HEALTH\_MONITOR** ID or name of health\_monitor to update

### Optional arguments

- |                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron lb-member-create command

```
usage: neutron lb-member-create [-h] [-f {shell,table}] [-c COLUMN]
                                [--variable VARIABLE] [--prefix PREFIX]
                                [--request-format {json,xml}]
                                [--tenant-id TENANT_ID] [--admin-state-down]
                                [--weight WEIGHT] --address ADDRESS
                                --protocol-port PROTOCOL_PORT
                                POOL
```

Create a member.

### Positional arguments

**POOL** Pool id or name this vip belongs to

### Optional arguments

|                                          |                                                                    |
|------------------------------------------|--------------------------------------------------------------------|
| <b>-h, --help</b>                        | show this help message and exit                                    |
| <b>--request-format {json,xml}</b>       | The xml or json request format                                     |
| <b>--tenant-id TENANT_ID</b>             | The owner tenant ID                                                |
| <b>--admin-state-down</b>                | Set admin state up to false                                        |
| <b>--weight WEIGHT</b>                   | Weight of pool member in the pool (default:1, [0..256])            |
| <b>--address ADDRESS IP</b>              | address of the pool member on the pool network.                    |
| <b>--protocol-port<br/>PROTOCOL_PORT</b> | Port on which the pool member listens for requests or connections. |

## neutron lb-member-delete command

```
usage: neutron lb-member-delete [-h] [--request-format {json,xml}] MEMBER
```

Delete a given member.

### Positional arguments

**MEMBER** ID or name of member to delete

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron lb-member-list command

```
usage: neutron lb-member-list [-h] [-f {csv,table}] [-c COLUMN]
                             [--quote {all,minimal,none,nonnumeric}]
                             [--request-format {json,xml}] [-D] [-F FIELD]
                             [-P SIZE] [--sort-key FIELD]
                             [--sort-dir {asc,desc}]
```

List members that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |

**-sort-dir {asc,desc}**

Sort list in specified directions (This option can be repeated)

## neutron lb-member-show command

```
usage: neutron lb-member-show [-h] [-f {shell,table}] [-c COLUMN]
                               [--variable VARIABLE] [--prefix PREFIX]
                               [--request-format {json,xml}] [-D] [-F FIELD]
                               MEMBER
```

Show information of a given member.

### Positional arguments

**MEMBER** ID or name of member to look up

### Optional arguments

**-h, --help**

show this help message and exit

**--request-format {json,xml}**

The xml or json request format

**-D, --show-details**

Show detailed info

**-F FIELD, --field FIELD**

Specify the field(s) to be returned by server, can be repeated

## neutron lb-member-update command

```
usage: neutron lb-member-update [-h] [--request-format {json,xml}] MEMBER
```

Update a given member.

## Positional arguments

**MEMBER** ID or name of member to update

## Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron lb-pool-create command

```
usage: neutron lb-pool-create [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}]
                             [--tenant-id TENANT_ID] [--admin-state-down]
                             [--description DESCRIPTION] --lb-method
                             {ROUND_ROBIN,LEAST_CONNECTIONS,SOURCE_IP} --name
                             NAME --protocol {HTTP,HTTPS,TCP} --subnet-id
                             SUBNET [--provider PROVIDER]
```

Create a pool.

## Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

**--tenant-id TENANT\_ID** The owner tenant ID

**--admin-state-down** Set admin state up to false



|                                                                       |                                                                       |
|-----------------------------------------------------------------------|-----------------------------------------------------------------------|
| <b>-description DESCRIPTION</b>                                       | Description of the pool                                               |
| <b>-lb-method</b><br><b>{ROUND_ROBIN,LEAST_CONNECTIONS,SOURCE_IP}</b> | The algorithm used to distribute load between the members of the pool |
| <b>-name NAME</b>                                                     | The name of the pool                                                  |
| <b>-protocol {HTTP,HTTPS,TCP}</b>                                     | Protocol for balancing                                                |
| <b>-subnet-id SUBNET</b>                                              | The subnet on which the members of the pool will be located           |
| <b>-provider PROVIDER</b>                                             | Provider name of loadbalancer service                                 |

## neutron lb-pool-delete command

```
usage: neutron lb-pool-delete [-h] [--request-format {json,xml}] POOL
```

Delete a given pool.

### Positional arguments

**POOL** ID or name of pool to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron lb-pool-list command

```
usage: neutron lb-pool-list [-h] [-f {csv,table}] [-c COLUMN]
```

```
[--quote {all,minimal,none,nonnumeric}]
[--request-format {json,xml}] [-D] [-F FIELD]
[-P SIZE] [--sort-key FIELD]
[--sort-dir {asc,desc}]
```

List pools that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron lb-pool-list-on-agent command

```
usage: neutron lb-pool-list-on-agent [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
```

```
lbaas_agent
```

List the pools on a loadbalancer agent.

### Positional arguments

**lbaas\_agent** ID of the loadbalancer agent to query

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron lb-pool-show command

```
usage: neutron lb-pool-show [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}] [-D] [-F FIELD]
                             POOL
```

Show information of a given pool.

### Positional arguments

**POOL** ID or name of pool to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron lb-pool-stats command

```
usage: neutron lb-pool-stats [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}] [-D] [-F FIELD]
                             POOL
```

Retrieve stats for a given pool.

### Positional arguments

**POOL** ID or name of pool to look up

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |
| <b>-D, --show-details</b>          | Show detailed info              |

**-F FIELD, --field FIELD**

Specify the field(s) to be returned by server, can be repeated

## neutron lb-pool-update command

```
usage: neutron lb-pool-update [-h] [--request-format {json,xml}] POOL
```

Update a given pool.

### Positional arguments

**POOL** ID or name of pool to update

### Optional arguments

**-h, --help**

show this help message and exit

**--request-format {json,xml}**

The xml or json request format

## neutron lb-vip-create command

```
usage: neutron lb-vip-create [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}]
                             [--tenant-id TENANT_ID] [--address ADDRESS]
                             [--admin-state-down]
                             [--connection-limit CONNECTION_LIMIT]
                             [--description DESCRIPTION] --name NAME
                             --protocol-port PROTOCOL_PORT --protocol
                             {TCP,HTTP,HTTPS} --subnet-id SUBNET
                             POOL
```

Create a vip.

## Positional arguments

**POOL** Pool id or name this vip belongs to

## Optional arguments

|                                                |                                                                                                                     |
|------------------------------------------------|---------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                              | show this help message and exit                                                                                     |
| <b>--request-format {json,xml}</b>             | The xml or json request format                                                                                      |
| <b>--tenant-id TENANT_ID</b>                   | The owner tenant ID                                                                                                 |
| <b>--address ADDRESS IP</b>                    | address of the vip                                                                                                  |
| <b>--admin-state-down</b>                      | Set admin state up to false                                                                                         |
| <b>--connection-limit<br/>CONNECTION_LIMIT</b> | The maximum number of connections per second allowed for the vip.<br>Positive integer or -1 for unlimited (default) |
| <b>--description DESCRIPTION</b>               | Description of the vip                                                                                              |
| <b>--name NAME</b>                             | Name of the vip                                                                                                     |
| <b>--protocol-port<br/>PROTOCOL_PORT</b>       | TCP port on which to listen for client traffic that is associated with the vip<br>address                           |
| <b>--protocol {TCP,HTTP,HTTPS}</b>             | Protocol for balancing                                                                                              |
| <b>--subnet-id SUBNET</b>                      | The subnet on which to allocate the vip address                                                                     |

## neutron lb-vip-delete command

```
usage: neutron lb-vip-delete [-h] [--request-format {json,xml}] VIP
```

Delete a given vip.

### Positional arguments

**VIP** ID or name of vip to delete

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron lb-vip-list command

```
usage: neutron lb-vip-list [-h] [-f {csv,table}] [-c COLUMN]
                           [--quote {all,minimal,none,nonnumeric}]
                           [--request-format {json,xml}] [-D] [-F FIELD]
                           [-P SIZE] [--sort-key FIELD]
                           [--sort-dir {asc,desc}]
```

List vips that belong to a given tenant.

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

**-D, --show-details** Show detailed info

**-F FIELD, --field FIELD** Specify the field(s) to be returned by server, can be repeated

|                                  |                                                                                                                                                                                                                      |
|----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-P SIZE, --page-size SIZE</b> | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>          | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>     | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron lb-vip-show command

```
usage: neutron lb-vip-show [-h] [-f {shell,table}] [-c COLUMN]
                          [--variable VARIABLE] [--prefix PREFIX]
                          [--request-format {json,xml}] [-D] [-F FIELD]
                          VIP
```

Show information of a given vip.

### Positional arguments

**VIP** ID or name of vip to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |



## neutron lb-vip-update command

```
usage: neutron lb-vip-update [-h] [--request-format {json,xml}] VIP
```

Update a given vip.

### Positional arguments

**VIP** ID or name of vip to update

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron meter-label-create command

```
usage: neutron meter-label-create [-h] [-f {shell,table}] [-c COLUMN]
                                   [--variable VARIABLE] [--prefix PREFIX]
                                   [--request-format {json,xml}]
                                   [--tenant-id TENANT_ID]
                                   [--description DESCRIPTION]
                                   NAME
```

Create a metering label for a given tenant.

### Positional arguments

**NAME** Name of metering label to create

### Optional arguments

|                                    |                                         |
|------------------------------------|-----------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit         |
| <b>--request-format {json,xml}</b> | The xml or json request format          |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID                     |
| <b>--description DESCRIPTION</b>   | Description of metering label to create |

## neutron meter-label-delete command

```
usage: neutron meter-label-delete [-h] [--request-format {json,xml}]
                                   METERING_LABEL
```

Delete a given metering label.

### Positional arguments

**METERING\_LABEL** ID or name of metering\_label to delete

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron meter-label-list command

```
usage: neutron meter-label-list [-h] [-f {csv,table}] [-c COLUMN]
                                [--quote {all,minimal,none,nonnumeric}]
                                [--request-format {json,xml}] [-D] [-F FIELD]
```

```
[-P SIZE] [--sort-key FIELD]
[--sort-dir {asc,desc}]
```

List metering labels that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron meter-label-rule-create command

```
usage: neutron meter-label-rule-create [-h] [-f {shell,table}] [-c COLUMN]
[--variable VARIABLE] [--prefix PREFIX]
[--request-format {json,xml}]
[--tenant-id TENANT_ID]
[--direction {ingress,egress}]
[--excluded]
LABEL REMOTE_IP_PREFIX
```

Create a metering label rule for a given label.

### Positional arguments

**LABEL** Id or Name of the label

**REMOTE\_IP\_PREFIX** CIDR to match on

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

**--tenant-id TENANT\_ID** The owner tenant ID

**--direction {ingress,egress}** Direction of traffic, default:ingress

**--excluded** Exclude this cidr from the label, default:not excluded

## neutron meter-label-rule-delete command

```
usage: neutron meter-label-rule-delete [-h] [--request-format {json,xml}]  
                                         METERING_LABEL_RULE
```

Delete a given metering label.

### Positional arguments

**METERING\_LABEL\_RULE** ID or name of metering\_label\_rule to delete

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron meter-label-rule-list command

```
usage: neutron meter-label-rule-list [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                     [--sort-dir {asc,desc}]
```

List metering labels that belong to a given label.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |

**-sort-dir {asc,desc}**

Sort list in specified directions (This option can be repeated)

## neutron meter-label-rule-show command

```
usage: neutron meter-label-rule-show [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
                                     METERING_LABEL_RULE
```

Show information of a given metering label rule.

### Positional arguments

**METERING\_LABEL\_RULE** ID or name of metering\_label\_rule to look up

### Optional arguments

**-h, --help**

show this help message and exit

**--request-format {json,xml}**

The xml or json request format

**-D, --show-details**

Show detailed info

**-F FIELD, --field FIELD**

Specify the field(s) to be returned by server, can be repeated

## neutron meter-label-show command

```
usage: neutron meter-label-show [-h] [-f {shell,table}] [-c COLUMN]
                                [--variable VARIABLE] [--prefix PREFIX]
                                [--request-format {json,xml}] [-D] [-F FIELD]
                                METERING_LABEL
```

Show information of a given metering label.

### Positional arguments

**METERING\_LABEL** ID or name of metering\_label to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron net-create command

```
usage: neutron net-create [-h] [-f {shell,table}] [-c COLUMN]
                          [--variable VARIABLE] [--prefix PREFIX]
                          [--request-format {json,xml}]
                          [--tenant-id TENANT_ID] [--admin-state-down]
                          [--shared]
                          NAME
```

Create a network for a given tenant.

### Positional arguments

**NAME** Name of network to create

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID             |
| <b>--admin-state-down</b>          | Set Admin State Up to false     |
| <b>--shared</b>                    | Set the network as shared       |

## neutron net-delete command

```
usage: neutron net-delete [-h] [--request-format {json,xml}] NETWORK
```

Delete a given network.

### Positional arguments

**NETWORK** ID or name of network to delete

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron net-external-list command

```
usage: neutron net-external-list [-h] [-f {csv,table}] [-c COLUMN]
```



```
[--quote {all,minimal,none,nonnumeric}]
[--request-format {json,xml}] [-D] [-F FIELD]
[-P SIZE] [--sort-key FIELD]
[--sort-dir {asc,desc}]
```

List external networks that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron net-gateway-connect command

```
usage: neutron net-gateway-connect [-h] [--request-format {json,xml}]
                                   [--segmentation-type SEGMENTATION_TYPE]
                                   [--segmentation-id SEGMENTATION_ID]
                                   NET-GATEWAY-ID NETWORK-ID
```

Add an internal network interface to a router.

### Positional arguments

**NET-GATEWAY-ID** ID of the network gateway

**NETWORK-ID** ID of the internal network to connect on the gateway

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

**--segmentation-type** L2 segmentation strategy on the external side of the gateway (e.g.: VLAN,  
**SEGMENTATION\_TYPE** FLAT)

**--segmentation-id** Identifier for the L2 segment on the external side of the gateway  
**SEGMENTATION\_ID**

## neutron net-gateway-create command

```
usage: neutron net-gateway-create [-h] [-f {shell,table}] [-c COLUMN]
                                   [--variable VARIABLE] [--prefix PREFIX]
                                   [--request-format {json,xml}]
                                   [--tenant-id TENANT_ID] [--device DEVICE]
                                   NAME
```

Create a network gateway.

### Positional arguments

**NAME** Name of network gateway to create

### Optional arguments

|                                    |                                                                                                                                                        |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                        |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                         |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID                                                                                                                                    |
| <b>--device DEVICE</b>             | Device info for this gateway device_id=<device identifier>,interface_name=<name_or_identifier> It can be repeated for multiple devices for HA gateways |

## neutron net-gateway-delete command

```
usage: neutron net-gateway-delete [-h] [--request-format {json,xml}]  
                                NETWORK_GATEWAY
```

Delete a given network gateway.

### Positional arguments

**NETWORK\_GATEWAYID** or name of network\_gateway to delete

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron net-gateway-disconnect command

```
usage: neutron net-gateway-disconnect [-h] [--request-format {json,xml}]
                                       [--segmentation-type SEGMENTATION_TYPE]
                                       [--segmentation-id SEGMENTATION_ID]
                                       NET-GATEWAY-ID NETWORK-ID
```

Remove a network from a network gateway.

### Positional arguments

**NET-GATEWAY-ID** ID of the network gateway

**NETWORK-ID** ID of the internal network to connect on the gateway

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

**--segmentation-type SEGMENTATION\_TYPE** L2 segmentation strategy on the external side of the gateway (e.g.: VLAN, FLAT)

**--segmentation-id SEGMENTATION\_ID** Identifier for the L2 segment on the external side of the gateway

## neutron net-gateway-list command

```
usage: neutron net-gateway-list [-h] [-f {csv,table}] [-c COLUMN]
                                [--quote {all,minimal,none,nonnumeric}]
```

```
[--request-format {json,xml}] [-D] [-F FIELD]
```

List network gateways for a given tenant.

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron net-gateway-show command

```
usage: neutron net-gateway-show [-h] [-f {shell,table}] [-c COLUMN]
                                [--variable VARIABLE] [--prefix PREFIX]
                                [--request-format {json,xml}] [-D] [-F FIELD]
                                NETWORK_GATEWAY
```

Show information of a given network gateway.

### Positional arguments

**NETWORK\_GATEWAYID** or name of network\_gateway to look up

### Optional arguments

|                   |                                 |
|-------------------|---------------------------------|
| <b>-h, --help</b> | show this help message and exit |
|-------------------|---------------------------------|

|                                   |                                                                |
|-----------------------------------|----------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>         | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated |

## neutron net-gateway-update command

```
usage: neutron net-gateway-update [-h] [--request-format {json,xml}]  
                                NETWORK_GATEWAY
```

Update the name for a network gateway.

### Positional arguments

**NETWORK\_GATEWAYID** or name of network\_gateway to update

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, --help</b>                 | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron net-list command

```
usage: neutron net-list [-h] [-f {csv,table}] [-c COLUMN]  
                        [--quote {all,minimal,none,nonnumeric}]  
                        [--request-format {json,xml}] [-D] [-F FIELD]  
                        [-P SIZE] [--sort-key FIELD] [--sort-dir {asc,desc}]
```

List networks that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron net-list-on-dhcp-agent command

```
usage: neutron net-list-on-dhcp-agent [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                     [--sort-dir {asc,desc}]
                                     dhcp_agent
```

List the networks on a DHCP agent.

### Positional arguments

**dhcp\_agent** ID of the DHCP agent

## Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron net-show command

```
usage: neutron net-show [-h] [-f {shell,table}] [-c COLUMN]
                        [--variable VARIABLE] [--prefix PREFIX]
                        [--request-format {json,xml}] [-D] [-F FIELD]
                        NETWORK
```

Show information of a given network.

## Positional arguments

**NETWORK** ID or name of network to look up



### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron net-update command

```
usage: neutron net-update [-h] [--request-format {json,xml}] NETWORK
```

Update network's information.

### Positional arguments

**NETWORK** ID or name of network to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron port-create command

```
usage: neutron port-create [-h] [-f {shell,table}] [-c COLUMN]
                           [--variable VARIABLE] [--prefix PREFIX]
                           [--request-format {json,xml}]
                           [--tenant-id TENANT_ID] [--name NAME]
```

```
[--admin-state-down] [--mac-address MAC_ADDRESS]
[--device-id DEVICE_ID]
[--fixed-ip subnet_id=SUBNET,ip_address=IP_ADDR]
[--security-group SECURITY_GROUP | --no-security-groups]
[--extra-dhcp-opt EXTRA_DHCP_OPTS]
NETWORK
```

Create a port for a given tenant.

### Positional arguments

**NETWORK** Network id or name this port belongs to

### Optional arguments

|                                    |                                                                                                                                                    |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                    |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                     |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID                                                                                                                                |
| <b>--name NAME</b>                 | Name of this port                                                                                                                                  |
| <b>--admin-state-down</b>          | Set admin state up to false                                                                                                                        |
| <b>--mac-address MAC_ADDRESS</b>   | MAC address of this port                                                                                                                           |
| <b>--device-id DEVICE_ID</b>       | Device id of this port                                                                                                                             |
| <b>--fixed-ip</b>                  | subnet_id=SUBNET,ip_address=IP_ADDR Desired IP and/or subnet for this port: subnet_id=<name_or_id>,ip_address=<ip>, (This option can be repeated.) |

|                                            |                                                                                                                                  |
|--------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <b>-security-group<br/>SECURITY_GROUP</b>  | Security group associated with the port (This option can be repeated)                                                            |
| <b>-no-security-groups</b>                 | Associate no security groups with the port                                                                                       |
| <b>-extra-dhcp-opt<br/>EXTRA_DHCP_OPTS</b> | Extra dhcp options to be assigned to this port:<br>opt_name=<dhcp_option_name>,opt_value=<value>, (This option can be repeated.) |

## neutron port-delete command

```
usage: neutron port-delete [-h] [--request-format {json,xml}] PORT
```

Delete a given port.

### Positional arguments

**PORT** ID or name of port to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron port-list command

```
usage: neutron port-list [-h] [-f {csv,table}] [-c COLUMN]
                        [--quote {all,minimal,none,nonnumeric}]
                        [--request-format {json,xml}] [-D] [-F FIELD]
                        [-P SIZE] [--sort-key FIELD] [--sort-dir {asc,desc}]
```

List ports that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron port-show command

```
usage: neutron port-show [-h] [-f {shell,table}] [-c COLUMN]
                        [--variable VARIABLE] [--prefix PREFIX]
                        [--request-format {json,xml}] [-D] [-F FIELD]
                        PORT
```

Show information of a given port.

### Positional arguments

**PORT** ID or name of port to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron port-update command

```
usage: neutron port-update [-h] [--request-format {json,xml}]
                           [--security-group SECURITY_GROUP | --no-security-groups]
                           [--extra-dhcp-opt EXTRA_DHCP_OPTS]
                           PORT
```

Update port's information.

### Positional arguments

**PORT** ID or name of port to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

|                                            |                                                                                                                                  |
|--------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|
| <b>-security-group<br/>SECURITY_GROUP</b>  | Security group associated with the port (This option can be repeated)                                                            |
| <b>-no-security-groups</b>                 | Associate no security groups with the port                                                                                       |
| <b>-extra-dhcp-opt<br/>EXTRA_DHCP_OPTS</b> | Extra dhcp options to be assigned to this port:<br>opt_name=<dhcp_option_name>,opt_value=<value>, (This option can be repeated.) |

## neutron queue-create command

```
usage: neutron queue-create [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}]
                             [--tenant-id TENANT_ID] [--min MIN] [--max MAX]
                             [--qos-marking QOS_MARKING] [--default DEFAULT]
                             [--dscp DSCP]
                             NAME
```

Create a queue.

### Positional arguments

**NAME** Name of queue

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |
| <b>-tenant-id TENANT_ID</b>       | The owner tenant ID             |

|                                 |                                                                                    |
|---------------------------------|------------------------------------------------------------------------------------|
| <b>-min MIN</b>                 | min-rate                                                                           |
| <b>-max MAX</b>                 | max-rate                                                                           |
| <b>-qos-marking QOS_MARKING</b> | QOS marking untrusted/trusted                                                      |
| <b>-default DEFAULT</b>         | If true all ports created with be the size of this queue if queue is not specified |
| <b>-dscp DSCP</b>               | Differentiated Services Code Point                                                 |

## neutron queue-delete command

```
usage: neutron queue-delete [-h] [--request-format {json,xml}] QOS_QUEUE
```

Delete a given queue.

### Positional arguments

**QOS\_QUEUE** ID or name of qos\_queue to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron queue-list command

```
usage: neutron queue-list [-h] [-f {csv,table}] [-c COLUMN]
                        [--quote {all,minimal,none,nonnumeric}]
```

```
[--request-format {json,xml}] [-D] [-F FIELD]
```

List queues that belong to a given tenant.

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron queue-show command

```
usage: neutron queue-show [-h] [-f {shell,table}] [-c COLUMN]
                          [--variable VARIABLE] [--prefix PREFIX]
                          [--request-format {json,xml}] [-D] [-F FIELD]
                          QOS_QUEUE
```

Show information of a given queue.

### Positional arguments

**QOS\_QUEUE** ID or name of qos\_queue to look up

### Optional arguments

|                   |                                 |
|-------------------|---------------------------------|
| <b>-h, --help</b> | show this help message and exit |
|-------------------|---------------------------------|



|                                   |                                                                |
|-----------------------------------|----------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>         | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated |

## neutron quota-delete command

```
usage: neutron quota-delete [-h] [--request-format {json,xml}]
                             [--tenant-id tenant-id]
```

Delete defined quotas of a given tenant.

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, --help</b>                 | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |
| <b>-tenant-id</b>                 | tenant-id The owner tenant ID   |

## neutron quota-list command

```
usage: neutron quota-list [-h] [-f {csv,table}] [-c COLUMN]
                           [--quote {all,minimal,none,nonnumeric}]
                           [--request-format {json,xml}]
```

List quotas of all tenants who have non-default quota values.

### Optional arguments

|                   |                                 |
|-------------------|---------------------------------|
| <b>-h, --help</b> | show this help message and exit |
|-------------------|---------------------------------|

**-request-format {json,xml}**      The xml or json request format

## neutron quota-show command

```
usage: neutron quota-show [-h] [-f {shell,table}] [-c COLUMN]
                          [--variable VARIABLE] [--prefix PREFIX]
                          [--request-format {json,xml}]
                          [--tenant-id tenant-id]
```

Show quotas of a given tenant

### Optional arguments

**-h, -help**      show this help message and exit

**-request-format {json,xml}**      The xml or json request format

**-tenant-id**      tenant-id The owner tenant ID

## neutron quota-update command

```
usage: neutron quota-update [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}]
                             [--tenant-id tenant-id] [--network networks]
                             [--subnet subnets] [--port ports]
                             [--router routers] [--floatingip floatingips]
                             [--security-group security_groups]
                             [--security-group-rule security_group_rules]
```

Define tenant's quotas not to use defaults.

## Optional arguments

|                                    |                                                         |
|------------------------------------|---------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                         |
| <b>--request-format {json,xml}</b> | The xml or json request format                          |
| <b>--tenant-id</b>                 | tenant-id The owner tenant ID                           |
| <b>--network</b>                   | networks The limit of networks                          |
| <b>--subnet</b>                    | subnets The limit of subnets                            |
| <b>--port</b>                      | ports The limit of ports                                |
| <b>--router</b>                    | routers The limit of routers                            |
| <b>--floatingip</b>                | floatingips The limit of floating IPs                   |
| <b>--security-group</b>            | security_groups The limit of security groups            |
| <b>--security-group-rule</b>       | security_group_rules The limit of security groups rules |

## neutron router-create command

```
usage: neutron router-create [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}]
                             [--tenant-id TENANT_ID] [--admin-state-down]
                             NAME
```

Create a router for a given tenant.

### Positional arguments

|                    |                                                      |
|--------------------|------------------------------------------------------|
| <b>NAME</b>        | Name of router to create                             |
| <b>distributed</b> | Create a distributed router (VMware NSX plugin only) |

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID             |
| <b>--admin-state-down</b>          | Set Admin State Up to false     |

## neutron router-delete command

```
usage: neutron router-delete [-h] [--request-format {json,xml}] ROUTER
```

Delete a given router.

### Positional arguments

**ROUTER** ID or name of router to delete

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron router-gateway-clear command

```
usage: neutron router-gateway-clear [-h] [--request-format {json,xml}]
                                     router-id
```

Remove an external network gateway from a router.

### Positional arguments

**router-id**      ID of the router

### Optional arguments

**-h, --help**                              show this help message and exit

**--request-format {json,xml}**            The xml or json request format

## neutron router-gateway-set command

```
usage: neutron router-gateway-set [-h] [--request-format {json,xml}]
                                   [--disable-snat]
                                   router-id external-network-id
```

Set the external network gateway for a router.

### Positional arguments

**router-id**                              ID of the router

**external-network-id**                  ID of the external network for the gateway

### Optional arguments

|                                    |                                          |
|------------------------------------|------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit          |
| <b>--request-format {json,xml}</b> | The xml or json request format           |
| <b>--disable-snat</b>              | Disable Source NAT on the router gateway |

## neutron router-interface-add command

```
usage: neutron router-interface-add [-h] [--request-format {json,xml}]
                                     router-id INTERFACE
```

Add an internal network interface to a router.

### Positional arguments

|                  |                                                                                                                                                                                                        |
|------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>router-id</b> | ID of the router                                                                                                                                                                                       |
| <b>INTERFACE</b> | The format is "SUBNET subnet=SUBNET port=PORT". Either a subnet or port must be specified. Both ID and name are accepted as SUBNET or PORT. Note that "subnet=" can be omitted when specifying subnet. |

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron router-interface-delete command

```
usage: neutron router-interface-delete [-h] [--request-format {json,xml}]
```

```
router-id INTERFACE
```

Remove an internal network interface from a router.

### Positional arguments

**router-id** ID of the router

**INTERFACE** The format is "SUBNET|subnet=SUBNET|port=PORT". Either a subnet or port must be specified. Both ID and name are accepted as SUBNET or PORT. Note that "subnet=" can be omitted when specifying subnet.

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron router-list command

```
usage: neutron router-list [-h] [-f {csv,table}] [-c COLUMN]
                           [--quote {all,minimal,none,nonnumeric}]
                           [--request-format {json,xml}] [-D] [-F FIELD]
                           [-P SIZE] [--sort-key FIELD]
                           [--sort-dir {asc,desc}]
```

List routers that belong to a given tenant.

### Optional arguments

**-h, --help** show this help message and exit

|                                   |                                                                                                                                                                                                                      |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>         | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>  | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>           | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>      | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron router-list-on-l3-agent command

```
usage: neutron router-list-on-l3-agent [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD]
                                     l3_agent
```

List the routers on a L3 agent.

### Positional arguments

**l3\_agent**     ID of the L3 agent to query

### Optional arguments

**-h, --help**                    show this help message and exit



|                                   |                                                                |
|-----------------------------------|----------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>         | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated |

## neutron router-port-list command

```
usage: neutron router-port-list [-h] [-f {csv,table}] [-c COLUMN]
                                [--quote {all,minimal,none,nonnumeric}]
                                [--request-format {json,xml}] [-D] [-F FIELD]
                                [-P SIZE] [--sort-key FIELD]
                                [--sort-dir {asc,desc}]
                                router
```

List ports that belong to a given tenant, with specified router.

### Positional arguments

**router** ID or name of router to look up

### Optional arguments

|                                   |                                                                                   |
|-----------------------------------|-----------------------------------------------------------------------------------|
| <b>-h, --help</b>                 | show this help message and exit                                                   |
| <b>-request-format {json,xml}</b> | The xml or json request format                                                    |
| <b>-D, --show-details</b>         | Show detailed info                                                                |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated                    |
| <b>-P SIZE, --page-size SIZE</b>  | Specify retrieve unit of each request, then split one request to several requests |

|                             |                                                                                                                                                                                                                      |
|-----------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-sort-key FIELD</b>      | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>-sort-dir {asc,desc}</b> | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron router-show command

```
usage: neutron router-show [-h] [-f {shell,table}] [-c COLUMN]
                           [--variable VARIABLE] [--prefix PREFIX]
                           [--request-format {json,xml}] [-D] [-F FIELD]
                           ROUTER
```

Show information of a given router.

### Positional arguments

**ROUTER** ID or name of router to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron router-update command

```
usage: neutron router-update [-h] [--request-format {json,xml}] ROUTER
```

Update router's information.

### Positional arguments

**ROUTER** ID or name of router to update

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron security-group-create command

```
usage: neutron security-group-create [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}]
                                     [--tenant-id TENANT_ID]
                                     [--description DESCRIPTION]
                                     NAME
```

Create a security group.

### Positional arguments

**NAME** Name of security group

### Optional arguments

**-h, --help** show this help message and exit

|                                   |                                |
|-----------------------------------|--------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format |
| <b>-tenant-id TENANT_ID</b>       | The owner tenant ID            |
| <b>-description DESCRIPTION</b>   | Description of security group  |

## neutron security-group-delete command

```
usage: neutron security-group-delete [-h] [--request-format {json,xml}]
                                     SECURITY_GROUP
```

Delete a given security group.

### Positional arguments

**SECURITY\_GROUP** ID or name of security\_group to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron security-group-list command

```
usage: neutron security-group-list [-h] [-f {csv,table}] [-c COLUMN]
                                   [--quote {all,minimal,none,nonnumeric}]
                                   [--request-format {json,xml}] [-D]
                                   [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                   [--sort-dir {asc,desc}]
```

List security groups that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron security-group-rule-create command

```
usage: neutron security-group-rule-create [-h] [-f {shell,table}] [-c COLUMN]
                                         [--variable VARIABLE]
                                         [--prefix PREFIX]
                                         [--request-format {json,xml}]
                                         [--tenant-id TENANT_ID]
                                         [--direction {ingress,egress}]
                                         [--ethertype ETHERTYPE]
                                         [--protocol PROTOCOL]
                                         [--port-range-min PORT_RANGE_MIN]
                                         [--port-range-max PORT_RANGE_MAX]
                                         [--remote-ip-prefix REMOTE_IP_PREFIX]
                                         [--remote-group-id REMOTE_GROUP]
                                         SECURITY_GROUP
```

---

Create a security group rule.

### Positional arguments

**SECURITY\_GROUP** Security group name or id to add rule.

### Optional arguments

|                                                |                                                |
|------------------------------------------------|------------------------------------------------|
| <b>-h, --help</b>                              | show this help message and exit                |
| <b>--request-format {json,xml}</b>             | The xml or json request format                 |
| <b>--tenant-id TENANT_ID</b>                   | The owner tenant ID                            |
| <b>--direction {ingress,egress}</b>            | Direction of traffic: ingress/egress           |
| <b>--ethertype ETHERTYPE</b>                   | IPv4/IPv6                                      |
| <b>--protocol PROTOCOL</b>                     | Protocol of packet                             |
| <b>--port-range-min<br/>PORT_RANGE_MIN</b>     | Starting port range                            |
| <b>--port-range-max<br/>PORT_RANGE_MAX</b>     | Ending port range                              |
| <b>--remote-ip-prefix<br/>REMOTE_IP_PREFIX</b> | CIDR to match on                               |
| <b>--remote-group-id<br/>REMOTE_GROUP</b>      | Remote security group name or id to apply rule |

## neutron security-group-rule-delete command

```
usage: neutron security-group-rule-delete [-h] [--request-format {json,xml}]
                                          SECURITY_GROUP_RULE
```

Delete a given security group rule.

### Positional arguments

**SECURITY\_GROUP\_RULE** ID of security\_group\_rule to delete

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

## neutron security-group-rule-list command

```
usage: neutron security-group-rule-list [-h] [-f {csv,table}] [-c COLUMN]
                                         [--quote {all,minimal,none,nonnumeric}]
                                         [--request-format {json,xml}] [-D]
                                         [-F FIELD] [-P SIZE]
                                         [--sort-key FIELD]
                                         [--sort-dir {asc,desc}]
                                         [--no-nameconv]
```

List security group rules that belong to a given tenant.

### Optional arguments

**-h, --help** show this help message and exit

|                                   |                                                                                                                                                                                                                      |
|-----------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>         | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>    | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>  | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>           | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>      | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |
| <b>--no-nameconv</b>              | Do not convert security group ID to its name                                                                                                                                                                         |

## neutron security-group-rule-show command

```
usage: neutron security-group-rule-show [-h] [-f {shell,table}] [-c COLUMN]
                                         [--variable VARIABLE]
                                         [--prefix PREFIX]
                                         [--request-format {json,xml}] [-D]
                                         [-F FIELD]
                                         SECURITY_GROUP_RULE
```

Show information of a given security group rule.

### Positional arguments

**SECURITY\_GROUP\_RULE** ID of security\_group\_rule to look up



### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

### neutron security-group-show command

```
usage: neutron security-group-show [-h] [-f {shell,table}] [-c COLUMN]
                                   [--variable VARIABLE] [--prefix PREFIX]
                                   [--request-format {json,xml}] [-D]
                                   [-F FIELD]
                                   SECURITY_GROUP
```

Show information of a given security group.

### Positional arguments

**SECURITY\_GROUP** ID or name of security\_group to look up

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |
| <b>-D, --show-details</b>          | Show detailed info              |

**-F FIELD, --field FIELD**

Specify the field(s) to be returned by server, can be repeated

## neutron security-group-update command

```
usage: neutron security-group-update [-h] [--request-format {json,xml}]
                                     [--name NAME] [--description DESCRIPTION]
                                     SECURITY_GROUP
```

Update a given security group.

### Positional arguments

**SECURITY\_GROUP** ID or name of security\_group to update

### Optional arguments

**-h, --help** show this help message and exit

**--request-format {json,xml}** The xml or json request format

**--name NAME** Name of security group

**--description DESCRIPTION** Description of security group

## neutron service-provider-list command

```
usage: neutron service-provider-list [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                     [--sort-dir {asc,desc}]
```

List service providers.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron subnet-create command

```
usage: neutron subnet-create [-h] [-f {shell,table}] [-c COLUMN]
                             [--variable VARIABLE] [--prefix PREFIX]
                             [--request-format {json,xml}]
                             [--tenant-id TENANT_ID] [--name NAME]
                             [--ip-version {4,6}] [--gateway GATEWAY_IP]
                             [--no-gateway]
                             [--allocation-pool start=IP_ADDR,end=IP_ADDR]
                             [--host-route destination=CIDR,nexthop=IP_ADDR]
                             [--dns-nameserver DNS_NAMESERVER]
                             [--disable-dhcp]
```

---

**NETWORK CIDR**

Create a subnet for a given tenant.

**Positional arguments**

**NETWORK** Network id or name this subnet belongs to

**CIDR** CIDR of subnet to create

**Optional arguments**

|                                    |                                                                                                         |
|------------------------------------|---------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                         |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                          |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID                                                                                     |
| <b>--name NAME</b>                 | Name of this subnet                                                                                     |
| <b>--ip-version {4,6} IP</b>       | version with default 4                                                                                  |
| <b>--gateway GATEWAY_IP</b>        | Gateway ip of this subnet                                                                               |
| <b>--no-gateway</b>                | No distribution of gateway                                                                              |
| <b>--allocation-pool</b>           | start=IP_ADDR,end=IP_ADDR Allocation pool IP addresses for this subnet<br>(This option can be repeated) |
| <b>--host-route</b>                | destination=CIDR,nexthop=IP_ADDR Additional route (This option can be<br>repeated)                      |

|                                                 |                                                               |
|-------------------------------------------------|---------------------------------------------------------------|
| <b>-dns-nameserver</b><br><b>DNS_NAMESERVER</b> | DNS name server for this subnet (This option can be repeated) |
| <b>-disable-dhcp</b>                            | Disable DHCP for this subnet                                  |

## neutron subnet-delete command

```
usage: neutron subnet-delete [-h] [--request-format {json,xml}] SUBNET
```

Delete a given subnet.

### Positional arguments

**SUBNET** ID or name of subnet to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron subnet-list command

```
usage: neutron subnet-list [-h] [-f {csv,table}] [-c COLUMN]
                           [--quote {all,minimal,none,nonnumeric}]
                           [--request-format {json,xml}] [-D] [-F FIELD]
                           [-P SIZE] [--sort-key FIELD]
                           [--sort-dir {asc,desc}]
```

List subnets that belong to a given tenant.

## Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron subnet-show command

```
usage: neutron subnet-show [-h] [-f {shell,table}] [-c COLUMN]
                           [--variable VARIABLE] [--prefix PREFIX]
                           [--request-format {json,xml}] [-D] [-F FIELD]
                           SUBNET
```

Show information of a given subnet.

## Positional arguments

**SUBNET** ID or name of subnet to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron subnet-update command

```
usage: neutron subnet-update [-h] [--request-format {json,xml}] SUBNET
```

Update subnet's information.

### Positional arguments

**SUBNET** ID or name of subnet to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## neutron vpn-ikepolicy-create command

```
usage: neutron vpn-ikepolicy-create [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}]
                                     [--tenant-id TENANT_ID]
```

```
[--description DESCRIPTION]
[--auth-algorithm {sha1}]
[--encryption-algorithm {3des,aes-128,aes-192,aes-256}]
[--phase1-negotiation-mode {main}]
[--ike-version {v1,v2}]
[--pfs {group2,group5,group14}]
[--lifetime units=UNITS,value=VALUE]
NAME
```

Create an IKEPolicy.

## Positional arguments

**NAME** Name of the IKE Policy

## Optional arguments

|                                                              |                                                        |
|--------------------------------------------------------------|--------------------------------------------------------|
| <b>-h, --help</b>                                            | show this help message and exit                        |
| <b>--request-format {json,xml}</b>                           | The xml or json request format                         |
| <b>--tenant-id TENANT_ID</b>                                 | The owner tenant ID                                    |
| <b>--description DESCRIPTION</b>                             | Description of the IKE policy                          |
| <b>--auth-algorithm {sha1}</b>                               | Authentication algorithm in lowercase. default:sha1    |
| <b>--encryption-algorithm {3des,aes-128,aes-192,aes-256}</b> | Encryption Algorithm in lowercase, default:aes-128     |
| <b>--phase1-negotiation-mode {main}</b>                      | IKE Phase1 negotiation mode in lowercase, default:main |



|                                     |                                                                                                                              |
|-------------------------------------|------------------------------------------------------------------------------------------------------------------------------|
| <b>-ike-version {v1,v2}</b>         | IKE version in lowercase, default:v1                                                                                         |
| <b>-pfs {group2,group5,group14}</b> | Perfect Forward Secrecy in lowercase, default:group5                                                                         |
| <b>-lifetime</b>                    | units=UNITS,value=VALUE IKE Lifetime Attributes.'units'-seconds,default:seconds. 'value'-non negative integer, default:3600. |

## neutron vpn-ikepolicy-delete command

```
usage: neutron vpn-ikepolicy-delete [-h] [--request-format {json,xml}]
                                     IKEPOLICY
```

Delete a given IKE Policy.

### Positional arguments

**IKEPOLICY** ID or name of ikepolicy to delete

### Optional arguments

**-h, -help** show this help message and exit

**-request-format {json,xml}** The xml or json request format

## neutron vpn-ikepolicy-list command

```
usage: neutron vpn-ikepolicy-list [-h] [-f {csv,table}] [-c COLUMN]
                                   [--quote {all,minimal,none,nonnumeric}]
                                   [--request-format {json,xml}] [-D]
                                   [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                   [--sort-dir {asc,desc}]
```

List IKEPolicies that belong to a tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron vpn-ikepolicy-show command

```
usage: neutron vpn-ikepolicy-show [-h] [-f {shell,table}] [-c COLUMN]
                                   [--variable VARIABLE] [--prefix PREFIX]
                                   [--request-format {json,xml}] [-D]
                                   [-F FIELD]
                                   IKEPOLICY
```

Show information of a given IKEPolicy.

### Positional arguments

**IKEPOLICY** ID or name of ikepolicy to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron vpn-ikepolicy-update command

```
usage: neutron vpn-ikepolicy-update [-h] [--request-format {json,xml}]
                                     [--lifetime units=UNITS,value=VALUE]
                                     IKEPOLICY
```

Update a given IKE Policy.

### Positional arguments

**IKEPOLICY** ID or name of ikepolicy to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

**-lifetime**

units=UNITS,value=VALUE IKE Lifetime Attributes.'units'-seconds,default:seconds. 'value'-non negative integer, default:3600.

## neutron vpn-ipsecpolicy-create command

```
usage: neutron vpn-ipsecpolicy-create [-h] [-f {shell,table}] [-c COLUMN]
                                     [--variable VARIABLE] [--prefix PREFIX]
                                     [--request-format {json,xml}]
                                     [--tenant-id TENANT_ID]
                                     [--description DESCRIPTION]
                                     [--transform-protocol {esp,ah,ah-esp}]
                                     [--auth-algorithm {sha1}]
                                     [--encryption-algorithm {3des,aes-128,aes-192,aes-256}]
                                     [--encapsulation-mode {tunnel,transport}]
                                     [--pfs {group2,group5,group14}]
                                     [--lifetime units=UNITS,value=VALUE]
                                     NAME
```

Create an ipsecpolicy.

### Positional arguments

**NAME** Name of the IPsecPolicy

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |
| <b>--tenant-id TENANT_ID</b>       | The owner tenant ID             |
| <b>--description DESCRIPTION</b>   | Description of the IPsecPolicy  |

|                                                             |                                                                                                                                |
|-------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>-transform-protocol {esp,ah,ah-esp}</b>                  | Transform Protocol in lowercase, default:esp                                                                                   |
| <b>-auth-algorithm {sha1}</b>                               | Authentication algorithm in lowercase, default:sha1                                                                            |
| <b>-encryption-algorithm {3des,aes-128,aes-192,aes-256}</b> | Encryption Algorithm in lowercase, default:aes-128                                                                             |
| <b>-encapsulation-mode {tunnel,transport}</b>               | Encapsulation Mode in lowercase, default:tunnel                                                                                |
| <b>-pfs {group2,group5,group14}</b>                         | Perfect Forward Secrecy in lowercase, default:group5                                                                           |
| <b>-lifetime</b>                                            | units=UNITS,value=VALUE IPsec Lifetime Attributes.'units'-seconds,default:seconds. 'value'-non negative integer, default:3600. |

## neutron vpn-ipsecpolicy-delete command

```
usage: neutron vpn-ipsecpolicy-delete [-h] [--request-format {json,xml}]
                                         IPSECPOLICY
```

Delete a given ipsecpolicy.

### Positional arguments

**IPSECPOLICY** ID or name of ipsecpolicy to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron vpn-ipsecpolicy-list command

```
usage: neutron vpn-ipsecpolicy-list [-h] [-f {csv,table}] [-c COLUMN]
                                     [--quote {all,minimal,none,nonnumeric}]
                                     [--request-format {json,xml}] [-D]
                                     [-F FIELD] [-P SIZE] [--sort-key FIELD]
                                     [--sort-dir {asc,desc}]
```

List ipsecpolicies that belongs to a given tenant connection.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

## neutron vpn-ipsecpolicy-show command

```
usage: neutron vpn-ipsecpolicy-show [-h] [-f {shell,table}] [-c COLUMN]
```

```
[--variable VARIABLE] [--prefix PREFIX]
[--request-format {json,xml}] [-D]
[-F FIELD]
IPSECPOLICY
```

Show information of a given ipsecpolicy.

### Positional arguments

**IPSECPOLICY** ID or name of ipsecpolicy to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron vpn-ipsecpolicy-update command

```
usage: neutron vpn-ipsecpolicy-update [-h] [--request-format {json,xml}]
                                       [--lifetime units=UNITS,value=VALUE]
                                       IPSECPOLICY
```

Update a given ipsec policy.

### Positional arguments

**IPSECPOLICY** ID or name of ipsecpolicy to update

### Optional arguments

|                                    |                                                                                                                                |
|------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                 |
| <b>--lifetime</b>                  | units=UNITS,value=VALUE IPsec Lifetime Attributes.'units'-seconds,default:seconds. 'value'-non negative integer, default:3600. |

## neutron vpn-service-create command

```
usage: neutron vpn-service-create [-h] [-f {shell,table}] [-c COLUMN]
                                  [--variable VARIABLE] [--prefix PREFIX]
                                  [--request-format {json,xml}]
                                  [--tenant-id TENANT_ID] [--admin-state-down]
                                  [--name NAME] [--description DESCRIPTION]
                                  ROUTER SUBNET
```

Create a VPNService.

### Positional arguments

**ROUTER** Router unique identifier for the vpnservice

**SUBNET** Subnet unique identifier for the vpnservice deployment

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |



|                                 |                                      |
|---------------------------------|--------------------------------------|
| <b>-tenant-id TENANT_ID</b>     | The owner tenant ID                  |
| <b>-admin-state-down</b>        | Set admin state up to false          |
| <b>-name NAME</b>               | Set a name for the vpnservice        |
| <b>-description DESCRIPTION</b> | Set a description for the vpnservice |

## neutron vpn-service-delete command

```
usage: neutron vpn-service-delete [-h] [--request-format {json,xml}]
                                   VPNSERVICE
```

Delete a given VPNService.

### Positional arguments

**VPNSERVICE** ID or name of vpnservice to delete

### Optional arguments

|                                   |                                 |
|-----------------------------------|---------------------------------|
| <b>-h, -help</b>                  | show this help message and exit |
| <b>-request-format {json,xml}</b> | The xml or json request format  |

## neutron vpn-service-list command

```
usage: neutron vpn-service-list [-h] [-f {csv,table}] [-c COLUMN]
                                [--quote {all,minimal,none,nonnumeric}]
                                [--request-format {json,xml}] [-D] [-F FIELD]
                                [-P SIZE] [--sort-key FIELD]
```

```
[--sort-dir {asc,desc}]
```

List VPNService configurations that belong to a given tenant.

### Optional arguments

|                                    |                                                                                                                                                                                                                      |
|------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                                                                                                                                                                      |
| <b>--request-format {json,xml}</b> | The xml or json request format                                                                                                                                                                                       |
| <b>-D, --show-details</b>          | Show detailed info                                                                                                                                                                                                   |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated                                                                                                                                                       |
| <b>-P SIZE, --page-size SIZE</b>   | Specify retrieve unit of each request, then split one request to several requests                                                                                                                                    |
| <b>--sort-key FIELD</b>            | Sort list by specified fields (This option can be repeated), The number of sort_dir and sort_key should match each other, more sort_dir specified will be omitted, less will be filled with asc as default direction |
| <b>--sort-dir {asc,desc}</b>       | Sort list in specified directions (This option can be repeated)                                                                                                                                                      |

### neutron vpn-service-show command

```
usage: neutron vpn-service-show [-h] [-f {shell,table}] [-c COLUMN]
                                [--variable VARIABLE] [--prefix PREFIX]
                                [--request-format {json,xml}] [-D] [-F FIELD]
                                VPNSERVICE
```

Show information of a given VPNService.

### Positional arguments

**VPNSERVICE** ID or name of vpnservice to look up

### Optional arguments

|                                    |                                                                |
|------------------------------------|----------------------------------------------------------------|
| <b>-h, --help</b>                  | show this help message and exit                                |
| <b>--request-format {json,xml}</b> | The xml or json request format                                 |
| <b>-D, --show-details</b>          | Show detailed info                                             |
| <b>-F FIELD, --field FIELD</b>     | Specify the field(s) to be returned by server, can be repeated |

## neutron vpn-service-update command

```
usage: neutron vpn-service-update [-h] [--request-format {json,xml}]
                                   VPNSERVICE
```

Update a given VPNService.

### Positional arguments

**VPNSERVICE** ID or name of vpnservice to update

### Optional arguments

|                                    |                                 |
|------------------------------------|---------------------------------|
| <b>-h, --help</b>                  | show this help message and exit |
| <b>--request-format {json,xml}</b> | The xml or json request format  |

## Manage Networks

Before you run commands, set the following environment variables:

```
export OS_USERNAME=admin
export OS_PASSWORD=password
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://localhost:5000/v2.0
```

### Create networks

1. List the extensions of the system:

```
$ neutron ext-list -c alias -c name
```

| alias           | name                     |
|-----------------|--------------------------|
| agent_scheduler | Agent Schedulers         |
| binding         | Port Binding             |
| quotas          | Quota management support |
| agent           | agent                    |
| provider        | Provider Network         |
| router          | Neutron L3 Router        |
| lbaas           | LoadBalancing service    |
| extraroute      | Neutron Extra Route      |

2. Create a network:

```
$ neutron net-create net1
```

Created a new network:

| Field | Value |
|-------|-------|
|       |       |

|                           |                                      |
|---------------------------|--------------------------------------|
| admin_state_up            | True                                 |
| id                        | 2d627131-c841-4e3a-ace6-f2dd75773b6d |
| name                      | net1                                 |
| provider:network_type     | vlan                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  | 1001                                 |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 3671f46ec35e4bbca6ef92ab7975e463     |



### Note

Some fields of the created network are invisible to non-admin users.

3. Create a network with specified provider network type:

```
$ neutron net-create net2 --provider:network-type local
```

Created a new network:

| Field                     | Value                                |
|---------------------------|--------------------------------------|
| admin_state_up            | True                                 |
| id                        | 524e26ea-fad4-4bb0-b504-1ad0dc770e7a |
| name                      | net2                                 |
| provider:network_type     | local                                |
| provider:physical_network |                                      |
| provider:segmentation_id  |                                      |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | 3671f46ec35e4bbca6ef92ab7975e463     |

```
+-----+-----+
```

Just as shown previously, the unknown option `--provider:network-type` is used to create a local provider network.

## Create subnets

- Create a subnet:

```
$ neutron subnet-create net1 192.168.2.0/24 --name subnet1
```

Created a new subnet:

```
+-----+-----+
| Field          | Value                                                                 |
+-----+-----+
| allocation_pools | {"start": "192.168.2.2", "end": "192.168.2.254"} |
| cidr            | 192.168.2.0/24 |
| dns_nameservers  | |
| enable_dhcp      | True |
| gateway_ip       | 192.168.2.1 |
| host_routes      | |
| id              | 15a09f6c-87a5-4d14-b2cf-03d97cd4b456 |
| ip_version       | 4 |
| name            | subnet1 |
| network_id       | 2d627131-c841-4e3a-ace6-f2dd75773b6d |
| tenant_id       | 3671f46ec35e4bbca6ef92ab7975e463 |
+-----+-----+
```

The **subnet-create** command has the following positional and optional parameters:

- The name or ID of the network to which the subnet belongs.

In this example, `net1` is a positional argument that specifies the network name.

- The CIDR of the subnet.

In this example, `192.168.2.0/24` is a positional argument that specifies the CIDR.

- The subnet name, which is optional.

In this example, `--name subnet1` specifies the name of the subnet.

## Create routers

1. Create a router:

```
$ neutron router-create router1
```

Created a new router:

| Field                 | Value                                |
|-----------------------|--------------------------------------|
| admin_state_up        | True                                 |
| external_gateway_info |                                      |
| id                    | 6e1f11ed-014b-4c16-8664-f4f615a3137a |
| name                  | router1                              |
| status                | ACTIVE                               |
| tenant_id             | 7b5970fbe7724bf9b74c245e66b92abf     |

Take note of the unique router identifier returned, this will be required in subsequent steps.

2. Link the router to the external provider network:

```
$ neutron router-gateway-set ROUTER NETWORK
```

Replace *ROUTER* with the unique identifier of the router, replace *NETWORK* with the unique identifier of the external provider network.

3. Link the router to the subnet:

```
$ neutron router-interface-add ROUTER SUBNET
```

Replace *ROUTER* with the unique identifier of the router, replace *SUBNET* with the unique identifier of the subnet.

## Create ports

1. Create a port with specified IP address:

```
$ neutron port-create net1 --fixed-ip ip_address=192.168.2.40
```

Created a new port:

```
+-----+
+-----+
| Field                | Value |
+-----+
+-----+
| admin_state_up       | True  |
| binding:capabilities | {"port_filter": false} |
| binding:vif_type     | ovs   |
| device_id            |       |
| device_owner         |       |
| fixed_ips             | {"subnet_id": "15a09f6c-87a5-4d14-b2cf-03d97cd4b456", |
| "ip_address": "192.168.2.40"} |
| id                   | f7a08fe4-e79e-4b67-bbb8-a5002455a493 |
| mac_address          | fa:16:3e:97:e0:fc |
+-----+
```



```

| name |
| network_id | 2d627131-c841-4e3a-ace6-f2dd75773b6d
| status | DOWN
| tenant_id | 3671f46ec35e4bbca6ef92ab7975e463
+-----+
+-----+

```

In the previous command, `net1` is the network name, which is a positional argument. `--fixed-ip ip_address=192.168.2.40` is an option, which specifies the port's fixed IP address we wanted.



### Note

When creating a port, you can specify any unallocated IP in the subnet even if the address is not in a pre-defined pool of allocated IP addresses (set by your cloud provider).

2. Create a port without specified IP address:

```
$ neutron port-create net1
```

```
Created a new port:
```

```

+-----+
+-----+
| Field| Value
|      |
+-----+
+-----+
| admin_state_up | True
|                |
| binding:capabilities | {"port_filter": false}
|                  |

```

```

| binding:vif_type      | ovs
| device_id             |
| device_owner          |
| fixed_ips             | {"subnet_id": "15a09f6c-87a5-4d14-b2cf-03d97cd4b456",
"ip_address": "192.168.2.2"} |
| id                    | baf13412-2641-4183-9533-de8f5b91444c
| mac_address           | fa:16:3e:f6:ec:c7
| name                  |
| network_id            | 2d627131-c841-4e3a-ace6-f2dd75773b6d
| status                | DOWN
| tenant_id             | 3671f46ec35e4bbca6ef92ab7975e463
+-----+
+-----+

```



### Note

Note that the system allocates one IP address if you do not specify an IP address in the **neutron port-create** command.

3. Query ports with specified fixed IP addresses:

```
$ neutron port-list --fixed-ips ip_address=192.168.2.2 ip_address=192.168.2.40
```

```

+-----+-----+-----+-----+
+-----+-----+-----+-----+
| id                    | name | mac_address | fixed_ips |
|

```

```
+-----+-----+-----+
+-----+
| baf13412-2641-4183-9533-de8f5b91444c |   | fa:16:3e:f6:ec:c7 | {"subnet_id":
| "15a09f6c-87a5-4d14-b2cf-03d97cd4b456", "ip_address": "192.168.2.2"} |
| f7a08fe4-e79e-4b67-bbb8-a5002455a493 |   | fa:16:3e:97:e0:fc | {"subnet_id":
| "15a09f6c-87a5-4d14-b2cf-03d97cd4b456", "ip_address": "192.168.2.40"} |
+-----+-----+-----+
+-----+
```

--fixed-ips ip\_address=192.168.2.2 ip\_address=192.168.2.40 is one unknown option.

**How to find unknown options?** The unknown options can be easily found by watching the output of create\_xxx or show\_xxx command. For example, in the port creation command, we see the fixed\_ips fields, which can be used as an unknown option.



## 8. Network Node Quiz

### Table of Contents

|                             |     |
|-----------------------------|-----|
| Day 2, 10:40 to 11:00 ..... | 463 |
|-----------------------------|-----|

### Day 2, 10:40 to 11:00



## 9. Object Storage Node

### Table of Contents

|                                             |     |
|---------------------------------------------|-----|
| Day 2, 11:30 to 12:30, 13:30 to 14:45 ..... | 465 |
| Introduction to Object Storage .....        | 465 |
| Features and Benefits .....                 | 466 |
| Administration Tasks .....                  | 467 |

### Day 2, 11:30 to 12:30, 13:30 to 14:45

### Introduction to Object Storage

OpenStack Object Storage (code-named Swift) is open source software for creating redundant, scalable data storage using clusters of standardized servers to store petabytes of accessible data. It is a long-term storage system for large amounts of static data that can be retrieved, leveraged, and updated. Object Storage uses a distributed architecture with no central point of control, providing greater scalability, redundancy and permanence. Objects are written to multiple hardware devices, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally by adding new nodes. Should a node fail, OpenStack works to replicate its content from other active nodes. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.

Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data

retention. Block Storage allows block devices to be exposed and connected to compute instances for expanded storage, better performance and integration with enterprise storage platforms, such as NetApp, Nexenta and SolidFire.

## Features and Benefits

| Features                                                                                                           | Benefits                                                                                                                                                             |
|--------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Leverages commodity hardware</b>                                                                                | No lock-in, lower price/GB                                                                                                                                           |
| <b>HDD/node failure agnostic</b>                                                                                   | Self healingReliability, data redundancy protecting from failures                                                                                                    |
| <b>Unlimited storage</b>                                                                                           | Huge & flat namespace, highly scalable read/write accessAbility to serve content directly from storage system                                                        |
| <b>Multi-dimensional scalability</b> (scale out architecture)Scale vertically and horizontally-distributed storage | Backup and archive large amounts of data with linear performance                                                                                                     |
| <b>Account/Container/Object structure</b> No nesting, not a traditional file system                                | Optimized for scaleScales to multiple petabytes, billions of objects                                                                                                 |
| <b>Built-in replication</b> 3x+ data redundancy compared to 2x on RAID                                             | Configurable number of accounts, container and object copies for high availability                                                                                   |
| <b>Easily add capacity</b> unlike RAID resize                                                                      | Elastic data scaling with ease                                                                                                                                       |
| <b>No central database</b>                                                                                         | Higher performance, no bottlenecks                                                                                                                                   |
| <b>RAID not required</b>                                                                                           | Handle lots of small, random reads and writes efficiently                                                                                                            |
| <b>Built-in management utilities</b>                                                                               | Account Management: Create, add, verify, delete usersContainer Management: Upload, download, verifyMonitoring: Capacity, host, network, log trawling, cluster health |
| <b>Drive auditing</b>                                                                                              | Detect drive failures preempting data corruption                                                                                                                     |
| <b>Expiring objects</b>                                                                                            | Users can set an expiration time or a TTL on an object to control access                                                                                             |
| <b>Direct object access</b>                                                                                        | Enable direct browser access to content, such as for a control panel                                                                                                 |
| <b>Realtime visibility into client requests</b>                                                                    | Know what users are requesting                                                                                                                                       |
| <b>Supports S3 API</b>                                                                                             | Utilize tools that were designed for the popular S3 API                                                                                                              |
| <b>Restrict containers per account</b>                                                                             | Limit access to control usage by user                                                                                                                                |
| <b>Support for NetApp, Nexenta, SolidFire</b>                                                                      | Unified support for block volumes using a variety of storage systems                                                                                                 |



|                                           |                                                                                |
|-------------------------------------------|--------------------------------------------------------------------------------|
| Snapshot and backup API for block volumes | Data protection and recovery for VM data                                       |
| Standalone volume API available           | Separate endpoint and API for integration with other compute systems           |
| Integration with Compute                  | Fully integrated to Compute for attaching block volumes and reporting on usage |

## Administration Tasks

### Object Storage CLI Commands

The **swift** client is the command-line interface (CLI) for the OpenStack Object Storage API and its extensions. This chapter documents **swift** version 2.0.3.

For help on a specific **swift** command, enter:

```
$ swift help COMMAND
```

#### swift usage

```
[--debug] [--info] [--quiet] [--auth <auth_url>]
[--auth-version <auth_version>] [--user <username>]
[--key <api_key>] [--retries <num_retries>]
[--os-username <auth-user-name>] [--os-password <auth-password>]
[--os-tenant-id <auth-tenant-id>]
[--os-tenant-name <auth-tenant-name>]
[--os-auth-url <auth-url>] [--os-auth-token <auth-token>]
[--os-storage-url <storage-url>] [--os-region-name <region-name>]
[--os-service-type <service-type>]
[--os-endpoint-type <endpoint-type>]
[--os-cacert <ca-certificate>] [--insecure]
[--no-ssl-compression]
<subcommand> ...
```

## Subcommands

|                     |                                                                                                   |
|---------------------|---------------------------------------------------------------------------------------------------|
| <b>delete</b>       | Delete a container or objects within a container                                                  |
| <b>download</b>     | Download objects from containers                                                                  |
| <b>list</b>         | Lists the containers for the account or the objects for a container                               |
| <b>post</b>         | Updates meta information for the account, container, or object; creates containers if not present |
| <b>stat</b>         | Displays information for the account, container, or object                                        |
| <b>upload</b>       | Uploads files or directories to the given container                                               |
| <b>capabilities</b> | List cluster capabilities                                                                         |

## swift examples

```
swift -A https://auth.api.rackspacecloud.com/v1.0 -U user -K api_key stat -v
swift --os-auth-url https://api.example.com/v2.0 --os-tenant-name tenant \
  --os-username user --os-password password list
swift --os-auth-token 6ee5eb33efad4e45ab46806eac010566 \
  --os-storage-url https://10.1.5.2:8080/v1/AUTH_ced809b6a4baea7aeab61a \
  list
swift list --lh
```

## swift optional arguments

|                  |                                        |
|------------------|----------------------------------------|
| <b>-version</b>  | show program's version number and exit |
| <b>-h, -help</b> | show this help message and exit        |
| <b>-s, -snet</b> | Use SERVICENET internal network        |

|                                                     |                                                                                     |
|-----------------------------------------------------|-------------------------------------------------------------------------------------|
| <b>-v, --verbose</b>                                | Print more info                                                                     |
| <b>--debug</b>                                      | Show the curl commands and results of all http queries regardless of result status. |
| <b>--info</b>                                       | Show the curl commands and results of all http queries which return an error.       |
| <b>-q, --quiet</b>                                  | Suppress status output                                                              |
| <b>-A AUTH, --auth=AUTH URL</b>                     | for obtaining an auth token                                                         |
| <b>-V AUTH_VERSION, --auth-version=AUTH_VERSION</b> | Specify a version for authentication. Defaults to 1.0.                              |
| <b>-U USER, --user=USER</b>                         | User name for obtaining an auth token.                                              |
| <b>-K KEY, --key=KEY</b>                            | Key for obtaining an auth token.                                                    |
| <b>-R RETRIES, --retries=RETRIES</b>                | The number of times to retry a failed connection.                                   |
| <b>--os-username=&lt;auth-user-name&gt;</b>         | OpenStack username. Defaults to env[OS_USERNAME].                                   |
| <b>--os-password=&lt;auth-password&gt;</b>          | OpenStack password. Defaults to env[OS_PASSWORD].                                   |
| <b>--os-tenant-id=&lt;auth-tenant-id&gt;</b>        | OpenStack tenant ID. Defaults to env[OS_TENANT_ID].                                 |
| <b>--os-tenant-name=&lt;auth-tenant-name&gt;</b>    | OpenStack tenant name. Defaults to env[OS_TENANT_NAME].                             |
| <b>--os-auth-url=&lt;auth-url&gt;</b>               | OpenStack auth URL. Defaults to env[OS_AUTH_URL].                                   |

|                                                |                                                                                                                                                                                               |
|------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-os-auth-token=&lt;auth-token&gt;</b>       | OpenStack token. Defaults to <code>env[OS_AUTH_TOKEN]</code> . Used with <code>-os-storage-url</code> to bypass the usual username/password authentication.                                   |
| <b>-os-storage-url=&lt;storage-url&gt;</b>     | OpenStack storage URL. Defaults to <code>env[OS_STORAGE_URL]</code> . Overrides the storage url returned during auth. Will bypass authentication when used with <code>-os-auth-token</code> . |
| <b>-os-region-name=&lt;region-name&gt;</b>     | OpenStack region name. Defaults to <code>env[OS_REGION_NAME]</code>                                                                                                                           |
| <b>-os-service-type=&lt;service-type&gt;</b>   | OpenStack Service type. Defaults to <code>env[OS_SERVICE_TYPE]</code>                                                                                                                         |
| <b>-os-endpoint-type=&lt;endpoint-type&gt;</b> | OpenStack Endpoint type. Defaults to <code>env[OS_ENDPOINT_TYPE]</code>                                                                                                                       |
| <b>-os-cacert=&lt;ca-certificate&gt;</b>       | Specify a CA bundle file to use in verifying a TLS (https) server certificate. Defaults to <code>env[OS_CACERT]</code>                                                                        |
| <b>-insecure</b>                               | Allow swiftclient to access servers without having to verify the SSL certificate. Defaults to <code>env[SWIFTCLIENT_INSECURE]</code> (set to 'true' to enable).                               |
| <b>-no-ssl-compression</b>                     | This option is deprecated and not used anymore. SSL compression should be disabled by default by the system SSL library                                                                       |

## swift delete command

Usage: Delete a container or objects within a container

### Positional arguments

**<container>**      Name of container to delete from

**[object]**            Name of object to delete. Specify multiple times for multiple objects

### Optional arguments

**-all**                                Delete all containers and objects

**-leave-segments**                Do not delete segments of manifest objects

**-object-threads <threads>**     Number of threads to use for deleting objects. Default is 10

**-container-threads <threads>**   Number of threads to use for deleting containers. Default is 10

## swift download command

Usage: Download objects from containers

### Positional arguments

**<container>**            Name of container to download from. To download a whole account, omit this and specify **-all**.

**[object]**                Name of object to download. Specify multiple times for multiple objects. Omit this to download all objects from the container.

### Optional arguments

**-all**                                Indicates that you really want to download everything in the account

**-marker**                            Marker to use when starting a container or account download

**-prefix <prefix>**                Only download items beginning with <prefix>

|                                                     |                                                                                                                                                          |
|-----------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-output &lt;out_file&gt;</b>                     | For a single file download, stream the output to <out_file>. Specifying "-" as <out_file> will redirect to stdout                                        |
| <b>-object-threads &lt;threads&gt;</b>              | Number of threads to use for downloading objects. Default is 10                                                                                          |
| <b>-container-threads &lt;threads&gt;</b>           | Number of threads to use for downloading containers. Default is 10                                                                                       |
| <b>-no-download</b>                                 | Perform download(s), but don't actually write anything to disk                                                                                           |
| <b>-header<br/>&lt;header_name:header_value&gt;</b> | Adds a customized request header to the query, like "Range" or "If-Match". This argument is repeatable. Example <b>-header "content-type:text/plain"</b> |
| <b>-skip-identical</b>                              | Skip downloading files that are identical on both sides                                                                                                  |

## swift list command

Usage: Lists the containers for the account or the objects for a container

### Positional arguments

**[container]**      Name of container to list object in

### Optional arguments

**-long**            Long listing format, similar to **ls -l**

**-lh**             Report sizes in human readable format similar to **ls -lh**

**-totals**         Used with **-l** or **-lh**, only report totals

**-prefix**         Only list items beginning with the prefix

**-delimiter** Roll up items with the given delimiter. For containers only. See OpenStack Swift API documentation for what this means.

## swift post command

Usage: Updates meta information for the account, container, or object.  
If the container is not found, it will be created automatically.

### Positional arguments

**[container]** Name of container to post to

**[object]** Name of object to post. Specify multiple times for multiple objects

### Optional arguments

**-read-acl <acl>** Read ACL for containers. Quick summary of ACL  
syntax: .r:\*, .r:-example.com, .r:www.example.com, account1, account2:user2

**-write-acl <acl>** Write ACL for containers. Quick summary of ACL syntax: account1  
account2:user2

**-sync-to <sync-to>** Sync To for containers, for multi-cluster replication

**-sync-key <sync-key>** Sync Key for containers, for multi-cluster replication

**-meta <name:value>** Sets a meta data item. This option may be repeated. Example: -m Color:Blue -m Size:Large

**-header <header>** Set request headers. This option may be repeated. Example -H "content-type:text/plain"

## swift stat command

Usage: Displays information for the account, container, or object

### Positional arguments

**[container]**      Name of container to stat from

**[object]**          Name of object to stat. Specify multiple times for multiple objects

### Optional arguments

**-lh**      Report sizes in human readable format similar to ls -lh

## swift upload command

Usage: Uploads specified files and directories to the given container

### Positional arguments

**<container>**      Name of container to upload to

**<file\_or\_directory>**      Name of file or directory to upload. Specify multiple times for multiple uploads

### Optional arguments

**-changed**                      Only upload files that have changed since the last upload

**-skip-identical**              Skip uploading files that are identical on both sides



|                                             |                                                                                                                                                                                           |
|---------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>-segment-size &lt;size&gt;</b>           | Upload files in segments no larger than <size> and then create a "manifest" file that will download all the segments as if it were the original file                                      |
| <b>-segment-container &lt;container&gt;</b> | Upload the segments into the specified container. If not specified, the segments will be uploaded to a <container>_segments container so as to not pollute the main <container> listings. |
| <b>-leave-segments</b>                      | Indicates that you want the older segments of manifest objects left alone (in the case of overwrites)                                                                                     |
| <b>-object-threads &lt;threads&gt;</b>      | Number of threads to use for uploading full objects. Default is 10.                                                                                                                       |
| <b>-segment-threads &lt;threads&gt;</b>     | Number of threads to use for uploading object segments. Default is 10.                                                                                                                    |
| <b>-header &lt;header&gt;</b>               | Set request headers with the syntax header:value. This option may be repeated. Example -H "content-type:text/plain".                                                                      |
| <b>-use-slo</b>                             | When used in conjunction with -segment-size will create a Static Large Object instead of the default Dynamic Large Object.                                                                |
| <b>-object-name &lt;object-name&gt;</b>     | Upload file and name object to <object-name> or upload dir and use <object-name> as object prefix instead of folder name                                                                  |

## Manage Object Storage

Will be included from the swift developer reference



# 10. Object Storage Node Quiz

## Table of Contents

|                             |     |
|-----------------------------|-----|
| Day 2, 14:25 to 14:45 ..... | 477 |
|-----------------------------|-----|

## Day 2, 14:25 to 14:45



# 11. Assessment

## Table of Contents

Day 2, 15:00 to 16:00 ..... 479

Questions ..... 479

## Day 2, 15:00 to 16:00

## Questions

Table 11.1. Assessment Question 1

| Task             |
|------------------|
| Configure a .... |

Table 11.2. Assessment Question 2

| Task             |
|------------------|
| Configure a .... |



## 12. Review of Concepts

### Table of Contents

|                             |     |
|-----------------------------|-----|
| Day 2, 16:00 to 17:00 ..... | 481 |
|-----------------------------|-----|

### Day 2, 16:00 to 17:00





## **Operator Training Guide**



# Table of Contents

|                                                                   |     |
|-------------------------------------------------------------------|-----|
| 1. Getting Started .....                                          | 1   |
| Day 1, 09:00 to 11:00, 11:15 to 12:30 .....                       | 1   |
| Overview .....                                                    | 1   |
| Review Associate Introduction .....                               | 2   |
| Review Associate Brief Overview .....                             | 4   |
| Review Associate Core Projects .....                              | 7   |
| Review Associate OpenStack Architecture .....                     | 21  |
| Review Associate Virtual Machine Provisioning Walk-Through .....  | 33  |
| 2. Getting Started Lab .....                                      | 41  |
| Day 1, 13:30 to 14:45, 15:00 to 17:00 .....                       | 41  |
| Getting the Tools and Accounts for Committing Code .....          | 41  |
| Fix a Documentation Bug .....                                     | 45  |
| Submit a Documentation Bug .....                                  | 49  |
| Create a Branch .....                                             | 49  |
| Optional: Add to the Training Guide Documentation .....           | 51  |
| 3. Getting Started Quiz .....                                     | 53  |
| Day 1, 16:40 to 17:00 .....                                       | 53  |
| 4. Controller Node .....                                          | 55  |
| Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30 .....                  | 55  |
| Review Associate Overview Horizon and OpenStack CLI .....         | 55  |
| Review Associate Keystone Architecture .....                      | 105 |
| Review Associate OpenStack Messaging and Queues .....             | 110 |
| Review Associate Administration Tasks .....                       | 121 |
| 5. Controller Node Lab .....                                      | 123 |
| Days 2 to 4, 13:30 to 14:45, 15:00 to 16:30, 16:45 to 18:15 ..... | 123 |
| Control Node Lab .....                                            | 123 |
| 6. Controller Node Quiz .....                                     | 143 |
| Days 2 to 4, 16:40 to 17:00 .....                                 | 143 |

|                                                       |     |
|-------------------------------------------------------|-----|
| 7. Network Node .....                                 | 145 |
| Days 7 to 8, 09:00 to 11:00, 11:15 to 12:30 .....     | 145 |
| Review Associate Networking in OpenStack .....        | 145 |
| Review Associate OpenStack Networking Concepts .....  | 151 |
| Review Associate Administration Tasks .....           | 153 |
| Operator OpenStack Neutron Use Cases .....            | 153 |
| Operator OpenStack Neutron Security .....             | 163 |
| Operator OpenStack Neutron Floating IPs .....         | 165 |
| 8. Network Node Lab .....                             | 167 |
| Days 7 to 8, 13:30 to 14:45, 15:00 to 17:00 .....     | 167 |
| Network Node Lab .....                                | 167 |
| 9. Network Node Quiz .....                            | 175 |
| Days 7 to 8, 16:40 to 17:00 .....                     | 175 |
| 10. Compute Node .....                                | 177 |
| Days 5 to 6, 09:00 to 11:00, 11:15 to 12:30 .....     | 177 |
| Review Associate VM Placement .....                   | 177 |
| Review Associate VM Provisioning Indepth .....        | 185 |
| Review Associate OpenStack Block Storage .....        | 189 |
| Review Associate Administration Tasks .....           | 194 |
| 11. Compute Node Lab .....                            | 195 |
| Days 5 to 6, 13:30 to 14:45, 15:00 to 17:00 .....     | 195 |
| Compute Node Lab .....                                | 195 |
| 12. Compute Node Quiz .....                           | 205 |
| Days 5 to 6, 16:40 to 17:00 .....                     | 205 |
| 13. Object Storage Node .....                         | 207 |
| Day 9, 09:00 to 11:00, 11:15 to 12:30 .....           | 207 |
| Review Associate Introduction to Object Storage ..... | 207 |
| Review Associate Features and Benefits .....          | 208 |
| Review Associate Administration Tasks .....           | 209 |
| Object Storage Capabilities .....                     | 209 |
| Object Storage Building Blocks .....                  | 211 |

---

|                                             |     |
|---------------------------------------------|-----|
| Swift Ring Builder .....                    | 222 |
| More Swift Concepts .....                   | 225 |
| Swift Cluster Architecture .....            | 229 |
| Swift Account Reaper .....                  | 233 |
| Swift Replication .....                     | 234 |
| 14. Object Storage Node Lab .....           | 237 |
| Day 9, 13:30 to 14:45, 15:00 to 17:00 ..... | 237 |
| Installing Object Node .....                | 237 |
| Configuring Object Node .....               | 239 |
| Configuring Object Proxy .....              | 242 |
| Start Object Node Services .....            | 247 |



## List of Figures

|                                                                                              |     |
|----------------------------------------------------------------------------------------------|-----|
| 1.1. Nebula (NASA) .....                                                                     | 5   |
| 1.2. Community Heartbeat .....                                                               | 9   |
| 1.3. Various Projects under OpenStack .....                                                  | 10  |
| 1.4. Programming Languages used to design OpenStack .....                                    | 12  |
| 1.5. OpenStack Compute: Provision and manage large networks of virtual machines .....        | 14  |
| 1.6. OpenStack Storage: Object and Block storage for use with servers and applications ..... | 15  |
| 1.7. OpenStack Networking: Pluggable, scalable, API-driven network and IP management .....   | 17  |
| 1.8. Conceptual Diagram .....                                                                | 23  |
| 1.9. Logical Diagram .....                                                                   | 25  |
| 1.10. Horizon Dashboard .....                                                                | 27  |
| 1.11. Initial State .....                                                                    | 36  |
| 1.12. Launch VM Instance .....                                                               | 38  |
| 1.13. End State .....                                                                        | 40  |
| 4.1. OpenStack Dashboard - Overview .....                                                    | 57  |
| 4.2. OpenStack Dashboard - Security Groups .....                                             | 60  |
| 4.3. OpenStack Dashboard - Security Group Rules .....                                        | 60  |
| 4.4. OpenStack Dashboard- Instances .....                                                    | 68  |
| 4.5. OpenStack Dashboard : Actions .....                                                     | 70  |
| 4.6. OpenStack Dashboard - Track Usage .....                                                 | 71  |
| 4.7. Keystone Authentication .....                                                           | 107 |
| 4.8. Messaging in OpenStack .....                                                            | 110 |
| 4.9. AMQP .....                                                                              | 112 |
| 4.10. RabbitMQ .....                                                                         | 115 |
| 4.11. RabbitMQ .....                                                                         | 116 |
| 4.12. RabbitMQ .....                                                                         | 117 |
| 5.1. Network Diagram .....                                                                   | 124 |
| 7.1. Network Diagram .....                                                                   | 150 |
| 7.2. Single Flat Network .....                                                               | 154 |

|                                                     |     |
|-----------------------------------------------------|-----|
| 7.3. Multiple Flat Network .....                    | 156 |
| 7.4. Mixed Flat and Private Network .....           | 158 |
| 7.5. Provider Router with Private Networks .....    | 160 |
| 7.6. Per-tenant Routers with Private Networks ..... | 162 |
| 8.1. Network Diagram .....                          | 168 |
| 10.1. Nova .....                                    | 178 |
| 10.2. Filtering .....                               | 180 |
| 10.3. Weights .....                                 | 184 |
| 10.4. Nova VM provisioning .....                    | 188 |
| 11.1. Network Diagram .....                         | 196 |
| 13.1. Object Storage(Swift) .....                   | 211 |
| 13.2. Building Blocks .....                         | 213 |
| 13.3. The Lord of the <b>Rings</b> .....            | 215 |
| 13.4. image33.png .....                             | 216 |
| 13.5. Accounts and Containers .....                 | 217 |
| 13.6. Partitions .....                              | 218 |
| 13.7. Replication .....                             | 219 |
| 13.8. When End-User uses Swift .....                | 221 |
| 13.9. Object Storage cluster architecture .....     | 230 |
| 13.10. Object Storage (Swift) .....                 | 232 |



# 1. Getting Started

## Table of Contents

|                                                                  |    |
|------------------------------------------------------------------|----|
| Day 1, 09:00 to 11:00, 11:15 to 12:30 .....                      | 1  |
| Overview .....                                                   | 1  |
| Review Associate Introduction .....                              | 2  |
| Review Associate Brief Overview .....                            | 4  |
| Review Associate Core Projects .....                             | 7  |
| Review Associate OpenStack Architecture .....                    | 21 |
| Review Associate Virtual Machine Provisioning Walk-Through ..... | 33 |

## Day 1, 09:00 to 11:00, 11:15 to 12:30

### Overview

Training would take 2.5 months self paced, (5) 2 week periods with a user group meeting, or 40 hours instructor led with 40 hours of self paced lab time.

#### Prerequisites

1. Associate guide training
2. Associate guide virtualbox scripted install completed and running

## Review Associate Introduction

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering users to provision resources through a web interface.

Cloud computing provides users with access to a shared collection of computing resources: networks for transfer, servers for storage, and applications or services for completing tasks.

The compelling features of a cloud are:

- On-demand self-service: Users can automatically provision needed computing capabilities, such as server time and network storage, without requiring human interaction with each service provider.
- Network access: Any computing capabilities are available over the network. Many different devices are allowed access through standardized mechanisms.
- Resource pooling: Multiple users can access clouds that serve other consumers according to demand.
- Elasticity: Provisioning is rapid and scales out or is based on need.
- Metered or measured service: Cloud systems can optimize and control resource use at the level that is appropriate for the service. Services include storage, processing, bandwidth, and active user accounts. Monitoring and reporting of resource usage provides transparency for both the provider and consumer of the utilized service.

Cloud computing offers different service models depending on the capabilities a consumer may require.

- SaaS: Software-as-a-Service. Provides the consumer the ability to use the software in a cloud environment, such as web-based email for example.

- PaaS: Platform-as-a-Service. Provides the consumer the ability to deploy applications through a programming language or tools supported by the cloud platform provider. An example of Platform-as-a-service is an Eclipse/Java programming platform provided with no downloads required.
- IaaS: Infrastructure-as-a-Service. Provides infrastructure such as computer instances, network connections, and storage so that people can run any software or operating system.

Terms such as public cloud or private cloud refer to the deployment model for the cloud. A private cloud operates for a single organization, but can be managed on-premise or off-premise. A public cloud has an infrastructure that is available to the general public or a large industry group and is likely owned by a cloud services company.

Clouds can also be described as hybrid. A hybrid cloud can be a deployment model, as a composition of both public and private clouds, or a hybrid model for cloud computing may involve both virtual and physical servers.

Cloud computing can help with large-scale computing needs or can lead consolidation efforts by virtualizing servers to make more use of existing hardware and potentially release old hardware from service. Cloud computing is also used for collaboration because of its high availability through networked computers. Productivity suites for word processing, number crunching, and email communications, and more are also available through cloud computing. Cloud computing also avails additional storage to the cloud user, avoiding the need for additional hard drives on each user's desktop and enabling access to huge data storage capacity online in the cloud.

When you explore OpenStack and see what it means technically, you can see its reach and impact on the entire world.

OpenStack is an open source software for building private and public clouds which delivers a massively scalable cloud operating system.

OpenStack is backed up by a global community of technologists, developers, researchers, corporations and cloud computing experts.



## Review Associate Brief Overview

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter. It is all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.

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
- feature rich.

To check out more information on OpenStack visit <http://goo.gl/Ye9DFT>

### OpenStack Foundation:

The OpenStack Foundation, established September 2012, is an independent body providing shared resources to help achieve the OpenStack Mission by protecting, empowering, and promoting OpenStack software and the community around it. This includes users, developers and the entire ecosystem. For more information visit <http://goo.gl/3uvmNX>.

### Who's behind OpenStack?

Founded by Rackspace Hosting and NASA, OpenStack has grown to be a global software community of developers collaborating on a standard and massively scalable open source cloud operating system. The OpenStack Foundation promotes the development, distribution and adoption of the OpenStack cloud operating system. As the independent home for OpenStack, the Foundation has already attracted more than 7,000 individual members from 100 countries and 850 different organizations. It has also secured more than \$10 million in funding and is ready to fulfill the OpenStack mission of becoming the ubiquitous cloud computing platform. Checkout <http://goo.gl/BZHJKd> for more on the same.

**Figure 1.1. Nebula (NASA)**



The goal of the OpenStack Foundation is to serve developers, users, and the entire ecosystem by providing a set of shared resources to grow the footprint of public and private OpenStack clouds, enable technology vendors targeting the platform and assist developers in producing the best cloud software in the industry.

**Who uses OpenStack?**

Corporations, service providers, VARS, SMBs, researchers, and global data centers looking to deploy large-scale cloud deployments for private or public clouds leveraging the support and resulting technology of a global open source community. This is just three years into OpenStack, it's new, it's yet to mature and has immense possibilities. How do I say that? All these 'buzz words' will fall into a properly solved jigsaw puzzle as you go through this article.

**It's Open Source:**

All of the code for OpenStack is freely available under the Apache 2.0 license. Anyone can run it, build on it, or submit changes back to the project. This open development model is one of the best ways to foster badly-needed cloud standards, remove the fear of proprietary lock-in for cloud customers, and create a large ecosystem that spans cloud providers.

**Who it's for:**

Enterprises, service providers, government and academic institutions with physical hardware that would like to build a public or private cloud.

**How it's being used today:**

Organizations like CERN, Cisco WebEx, DreamHost, eBay, The Gap, HP, MercadoLibre, NASA, PayPal, Rackspace and University of Melbourne have deployed OpenStack clouds to achieve control, business agility and cost savings without the licensing fees and terms of proprietary software. For complete user stories visit <http://goo.gl/aF4lsL>, this should give you a good idea about the importance of OpenStack.

## Review Associate Core Projects

### Project history and releases overview.

OpenStack is a cloud computing project that provides an Infrastructure-as-a-Service (IaaS). It is free open source software released under the terms of the Apache License. The project is managed by the OpenStack Foundation, a non-profit corporate entity established in September 2012 to promote OpenStack software and its community.

More than 200 companies joined the project, among which are AMD, Brocade Communications Systems, Canonical, Cisco, Dell, EMC, Ericsson, Groupe Bull, HP, IBM, Inktank, Intel, NEC, Rackspace Hosting, Red Hat, SUSE Linux, VMware, and Yahoo!

The technology consists of a series of interrelated projects that control pools of processing, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering its users to provision resources through a web interface.

The OpenStack community collaborates around a six-month, time-based release cycle with frequent development milestones. During the planning phase of each release, the community gathers for the OpenStack Design Summit to facilitate developer working sessions and assemble plans.

In July 2010 Rackspace Hosting and NASA jointly launched an open-source cloud-software initiative known as OpenStack. The OpenStack project intended to help organizations which offer cloud-computing services running on standard hardware. The first official release, code-named Austin, appeared four months later, with plans to release regular updates of the software every few months. The early code came from the NASA Nebula platform and from the Rackspace Cloud Files platform. In July 2011, Ubuntu Linux developers adopted OpenStack.

### OpenStack Releases

| Release Name | Release Date | Included Components |
|--------------|--------------|---------------------|
|--------------|--------------|---------------------|

|          |                   |                                                                             |
|----------|-------------------|-----------------------------------------------------------------------------|
| Austin   | 21 October 2010   | Nova, Swift                                                                 |
| Bexar    | 3 February 2011   | Nova, Glance, Swift                                                         |
| Cactus   | 15 April 2011     | Nova, Glance, Swift                                                         |
| Diablo   | 22 September 2011 | Nova, Glance, Swift                                                         |
| Essex    | 5 April 2012      | Nova, Glance, Swift, Horizon, Keystone                                      |
| Folsom   | 27 September 2012 | Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder                     |
| Grizzly  | 4 April 2013      | Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder                     |
| Havana   | 17 October 2013   | Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder                     |
| Icehouse | April 2014        | Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, (More to be added) |

Some OpenStack users include:

- PayPal / eBay
- NASA
- CERN
- Yahoo!
- Rackspace Cloud
- HP Public Cloud
- MercadoLibre.com
- AT&T



- KT (formerly Korea Telecom)
- Deutsche Telekom
- Wikimedia Labs
- Hostalia of Telefonica Group
- SUSE Cloud solution
- Red Hat OpenShift PaaS solution
- Zadara Storage
- Mint Services
- GridCentric

OpenStack is a true and innovative open standard. For more user stories, see <http://goo.gl/aF4IsL>.

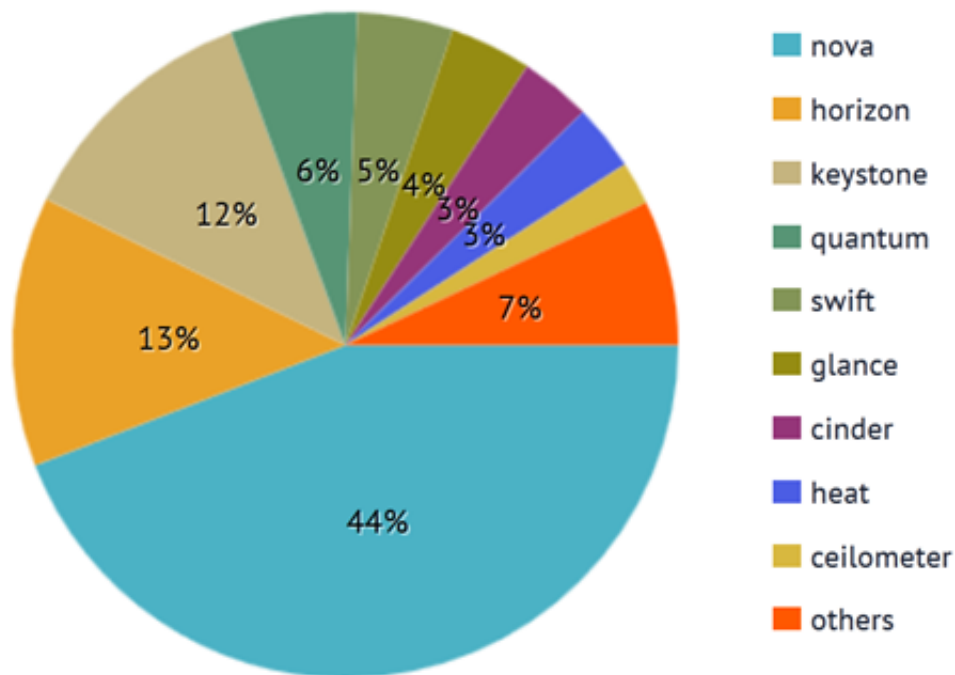
### Release Cycle

**Figure 1.2. Community Heartbeat**



OpenStack is based on a coordinated 6-month release cycle with frequent development milestones. You can find a link to the current development release schedule [here](#). The Release Cycle is made of four major stages:

**Figure 1.3. Various Projects under OpenStack**



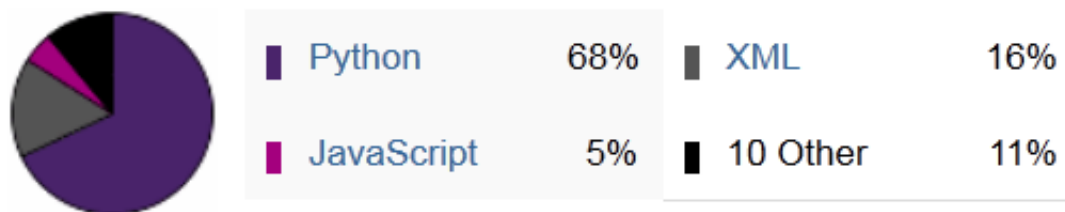
The creation of OpenStack took an estimated 249 years of effort (COCOMO model).

In a nutshell, OpenStack has:

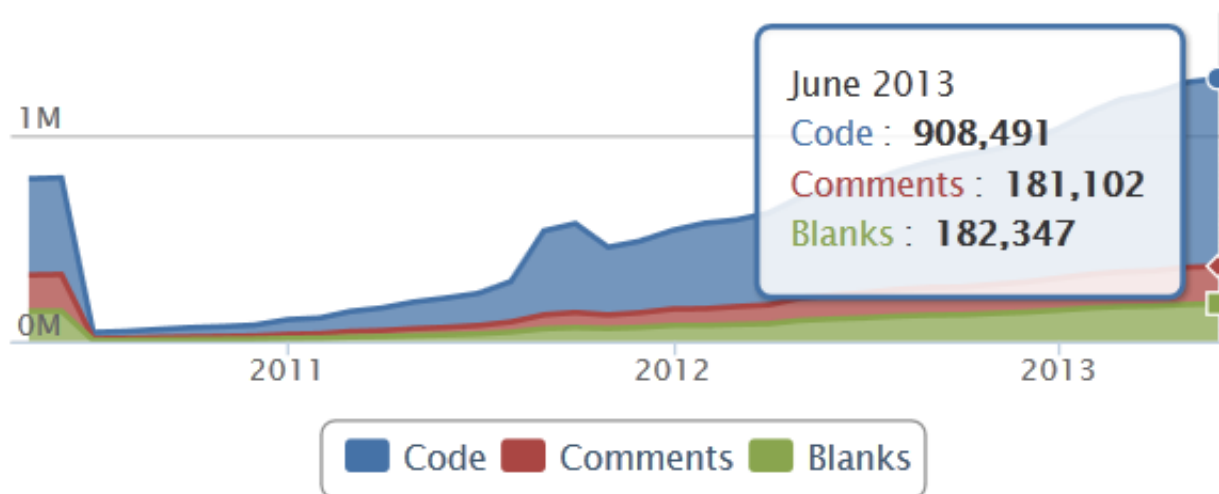
- 64,396 commits made by 1,128 contributors, with its first commit made in May, 2010.

- 908,491 lines of code. OpenStack is written mostly in Python with an average number of source code comments.
- A code base with a long source history.
- Increasing Y-O-Y commits.
- A very large development team comprised of people from around the world.

**Figure 1.4. Programming Languages used to design OpenStack**  
**Languages**



## Lines of Code



For an overview of OpenStack refer to <http://www.openstack.org> or <http://goo.gl/4q7nVI>. Common questions and answers are also covered here.

### **Core Projects Overview**

Let's take a dive into some of the technical aspects of OpenStack. Its scalability and flexibility are just some of the awesome features that make it a rock-solid cloud computing platform. The OpenStack core projects serve the community and its demands.

Being a cloud computing platform, OpenStack consists of many core and incubated projects which makes it really good as an IaaS cloud computing platform/Operating System. The following points are the main components necessary to call it an OpenStack Cloud.

### **Components of OpenStack**

OpenStack has a modular architecture with various code names for its components. OpenStack has several shared services that span the three pillars of compute, storage and networking, making it easier to implement and operate your cloud. These services - including identity, image management and a web interface - integrate the OpenStack components with each other as well as external systems to provide a unified experience for users as they interact with different cloud resources.

### **Compute (Nova)**

The OpenStack cloud operating system enables enterprises and service providers to offer on-demand computing resources, by provisioning and managing large networks of virtual machines. Compute resources are accessible via APIs for developers building cloud applications and via web interfaces for administrators and users. The compute architecture is designed to scale horizontally on standard hardware.

**Figure 1.5. OpenStack Compute: Provision and manage large networks of virtual machines**



OpenStack Compute (Nova) is a cloud computing fabric controller (the main part of an IaaS system). It is written in Python and uses many external libraries such as Eventlet (for concurrent programming), Kombu (for AMQP communication), and SQLAlchemy (for database access). Nova's architecture is designed to scale horizontally on standard hardware with no proprietary hardware or software requirements and provide the ability to integrate with legacy systems and third party technologies. It is designed to manage and automate pools of computer resources and can work with widely available virtualization technologies, as well as bare metal and high-performance computing (HPC) configurations. KVM and XenServer are available choices for hypervisor technology, together with Hyper-V and Linux container technology such as LXC. In addition to different hypervisors, OpenStack runs on ARM.

**Popular Use Cases:**

- Service providers offering an IaaS compute platform or services higher up the stack
- IT departments acting as cloud service providers for business units and project teams
- Processing big data with tools like Hadoop
- Scaling compute up and down to meet demand for web resources and applications
- High-performance computing (HPC) environments processing diverse and intensive workloads

**Object Storage(Swift)**

In addition to traditional enterprise-class storage technology, many organizations now have a variety of storage needs with varying performance and price requirements. OpenStack has support for both Object Storage and Block Storage, with many deployment options for each depending on the use case.

**Figure 1.6. OpenStack Storage: Object and Block storage for use with servers and applications**



OpenStack Object Storage (Swift) is a scalable redundant storage system. Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used.

Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. Block Storage allows block devices to be exposed and connected to compute instances for expanded storage, better performance and integration with enterprise storage platforms, such as NetApp, Nexenta and SolidFire.

A few details on OpenStack's Object Storage

- OpenStack provides redundant, scalable object storage using clusters of standardized servers capable of storing petabytes of data

- Object Storage is not a traditional file system, but rather a distributed storage system for static data such as virtual machine images, photo storage, email storage, backups and archives. Having no central "brain" or master point of control provides greater scalability, redundancy and durability.
- Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster.
- Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.

### **Block Storage(Cinder)**

OpenStack Block Storage (Cinder) provides persistent block level storage devices for use with OpenStack compute instances. The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs. In addition to local Linux server storage, it can use storage platforms including Ceph, CloudByte, Coraid, EMC (VMAX and VNX), GlusterFS, IBM Storage (Storwize family, SAN Volume Controller, and XIV Storage System), Linux LIO, NetApp, Nexenta, Scalality, SolidFire and HP (Store Virtual and StoreServ 3Par families). Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage. Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

#### **A few points on OpenStack Block Storage:**

- OpenStack provides persistent block level storage devices for use with OpenStack compute instances.
- The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs.



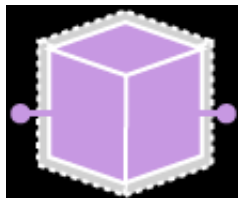
- In addition to using simple Linux server storage, it has unified storage support for numerous storage platforms including Ceph, NetApp, Nexenta, SolidFire, and Zadara.
- Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage.
- Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

### Networking(Neutron)

Today's data center networks contain more devices than ever before. From servers, network equipment, storage systems and security appliances, many of which are further divided into virtual machines and virtual networks. The number of IP addresses, routing configurations and security rules can quickly grow into the millions. Traditional network management techniques fall short of providing a truly scalable, automated approach to managing these next-generation networks. At the same time, users expect more control and flexibility with quicker provisioning.

OpenStack Networking is a pluggable, scalable and API-driven system for managing networks and IP addresses. Like other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing data center assets. OpenStack Networking ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self-service, even over their network configurations.

**Figure 1.7. OpenStack Networking: Pluggable, scalable, API-driven network and IP management**



OpenStack Networking (Neutron, formerly Quantum) is a system for managing networks and IP addresses. Like other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing data center assets. OpenStack Networking ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self-service, even over their network configurations.

OpenStack Neutron provides networking models for different applications or user groups. Standard models include flat networks or VLANs for separation of servers and traffic. OpenStack Networking manages IP addresses, allowing for dedicated static IPs or DHCP. Floating IPs allow traffic to be dynamically re routed to any of your compute resources, which allows you to redirect traffic during maintenance or in the case of failure. Users can create their own networks, control traffic and connect servers and devices to one or more networks. Administrators can take advantage of software-defined networking (SDN) technology like OpenFlow to allow for high levels of multi-tenancy and massive scale. OpenStack Networking has an extension framework allowing additional network services, such as intrusion detection systems (IDS), load balancing, firewalls and virtual private networks (VPN) to be deployed and managed.

#### Networking Capabilities

- OpenStack provides flexible networking models to suit the needs of different applications or user groups. Standard models include flat networks or VLANs for separation of servers and traffic.
- OpenStack Networking manages IP addresses, allowing for dedicated static IPs or DHCP. Floating IPs allow traffic to be dynamically re-routed to any of your compute resources, which allows you to redirect traffic during maintenance or in the case of failure.
- Users can create their own networks, control traffic and connect servers and devices to one or more networks.
- The pluggable backend architecture lets users take advantage of commodity gear or advanced networking services from supported vendors.
- Administrators can take advantage of software-defined networking (SDN) technology like OpenFlow to allow for high levels of multi-tenancy and massive scale.

- OpenStack Networking has an extension framework allowing additional network services, such as intrusion detection systems (IDS), load balancing, firewalls and virtual private networks (VPN) to be deployed and managed.

### **Dashboard(Horizon)**

OpenStack Dashboard (Horizon) provides administrators and users a graphical interface to access, provision and automate cloud-based resources. The design allows for third party products and services, such as billing, monitoring and additional management tools. Service providers and other commercial vendors can customize the dashboard with their own brand.

The dashboard is just one way to interact with OpenStack resources. Developers can automate access or build tools to manage their resources using the native OpenStack API or the EC2 compatibility API.

### **Identity Service(Keystone)**

OpenStack Identity (Keystone) provides a central directory of users mapped to the OpenStack services they can access. It acts as a common authentication system across the cloud operating system and can integrate with existing backend directory services like LDAP. It supports multiple forms of authentication including standard username and password credentials, token-based systems, and Amazon Web Services log in credentials such as those used for EC2.

Additionally, the catalog provides a query-able list of all of the services deployed in an OpenStack cloud in a single registry. Users and third-party tools can programmatically determine which resources they can access.

The OpenStack Identity Service enables administrators to:

- Configure centralized policies across users and systems
- Create users and tenants and define permissions for compute, storage, and networking resources by using role-based access control (RBAC) features
- Integrate with an existing directory, like LDAP, to provide a single source of authentication across the enterprise

The OpenStack Identity Service enables users to:

- List the services to which they have access
- Make API requests
- Log into the web dashboard to create resources owned by their account

### **Image Service(Glance)**

OpenStack Image Service (Glance) provides discovery, registration and delivery services for disk and server images. Stored images can be used as a template. They can also be used to store and catalog an unlimited number of backups. The Image Service can store disk and server images in a variety of back-ends, including OpenStack Object Storage. The Image Service API provides a standard REST interface for querying information about disk images and lets clients stream the images to new servers.

Capabilities of the Image Service include:

- Administrators can create base templates from which their users can start new compute instances
- Users can choose from available images, or create their own from existing servers
- Snapshots can also be stored in the Image Service so that virtual machines can be backed up quickly

A multi-format image registry, the image service allows uploads of private and public images in a variety of formats, including:

- Raw
- Machine (kernel/ramdisk outside of image, also known as AMI)
- VHD (Hyper-V)
- VDI (VirtualBox)

- qcow2 (Qemu/KVM)
- VMDK (VMWare)
- OVF (VMWare, others)

To checkout the complete list of Core and Incubated projects under OpenStack check out OpenStack's Launchpad Project Page here : <http://goo.gl/ka4SrV>

### **Amazon Web Services compatibility**

OpenStack APIs are compatible with Amazon EC2 and Amazon S3 and thus client applications written for Amazon Web Services can be used with OpenStack with minimal porting effort.

### **Governance**

OpenStack is governed by a non-profit foundation and its board of directors, a technical committee and a user committee.

The foundation's stated mission is by providing shared resources to help achieve the OpenStack Mission by Protecting, Empowering, and Promoting OpenStack software and the community around it, including users, developers and the entire ecosystem. Though, it has little to do with the development of the software, which is managed by the technical committee - an elected group that represents the contributors to the project, and has oversight on all technical matters.

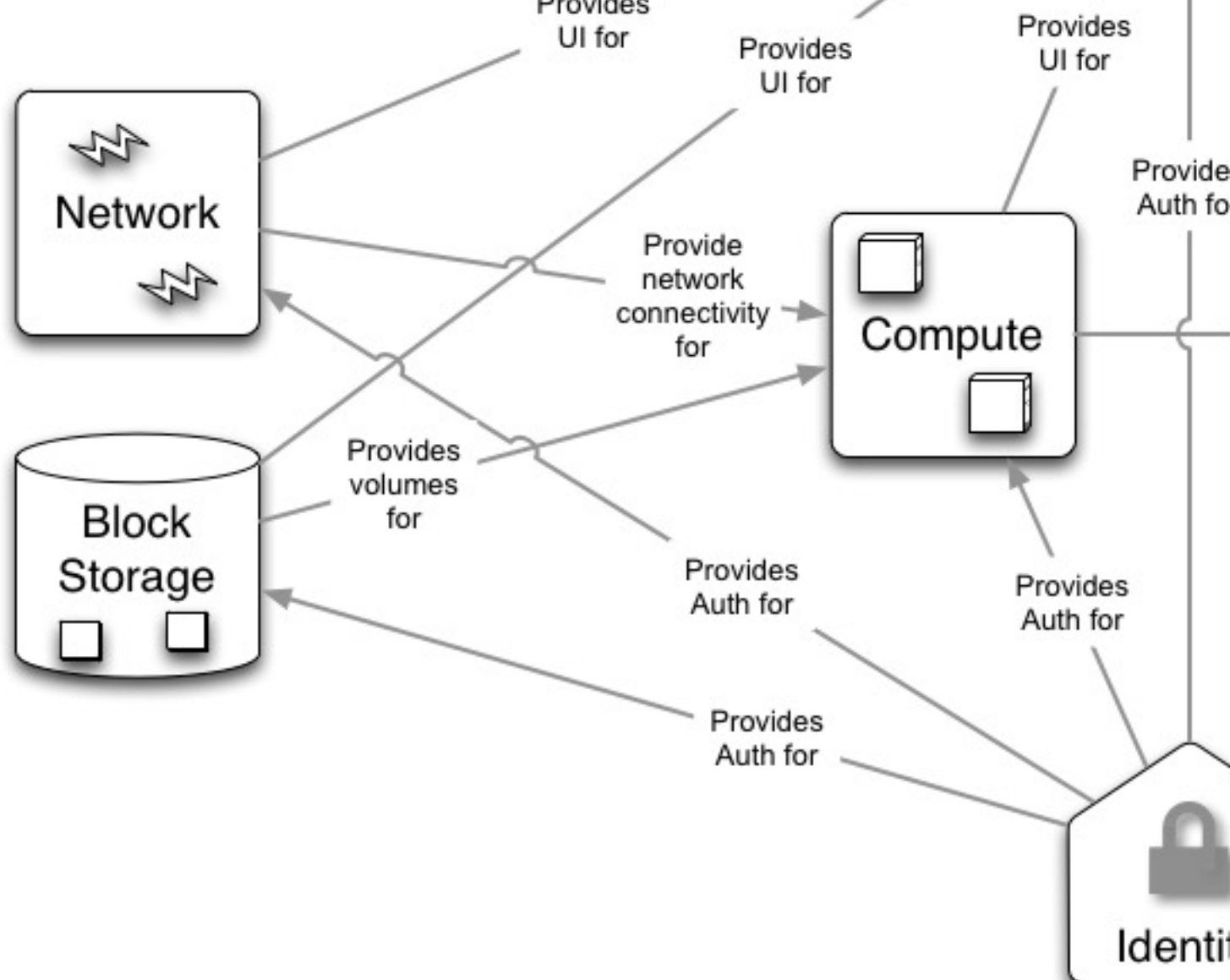
## **Review Associate OpenStack Architecture**

### **Conceptual Architecture**

The OpenStack project as a whole is designed to deliver a massively scalable cloud operating system. To achieve this, each of the constituent services are designed to work together to provide a complete

Infrastructure-as-a-Service (IaaS). This integration is facilitated through public application programming interfaces (APIs) that each service offers (and in turn can consume). While these APIs allow each of the services to use another service, it also allows an implementer to switch out any service as long as they maintain the API. These are (mostly) the same APIs that are available to end users of the cloud.

Conceptually, you can picture the relationships between the services as so:



- Dashboard ("Horizon") provides a web front end to the other OpenStack services
- Compute ("Nova") stores and retrieves virtual disks ("images") and associated metadata in Image ("Glance")
- Network ("Neutron") provides virtual networking for Compute.
- Block Storage ("Cinder") provides storage volumes for Compute.
- Image ("Glance") can store the actual virtual disk files in the Object Store("Swift")
- All the services authenticate with Identity ("Keystone")

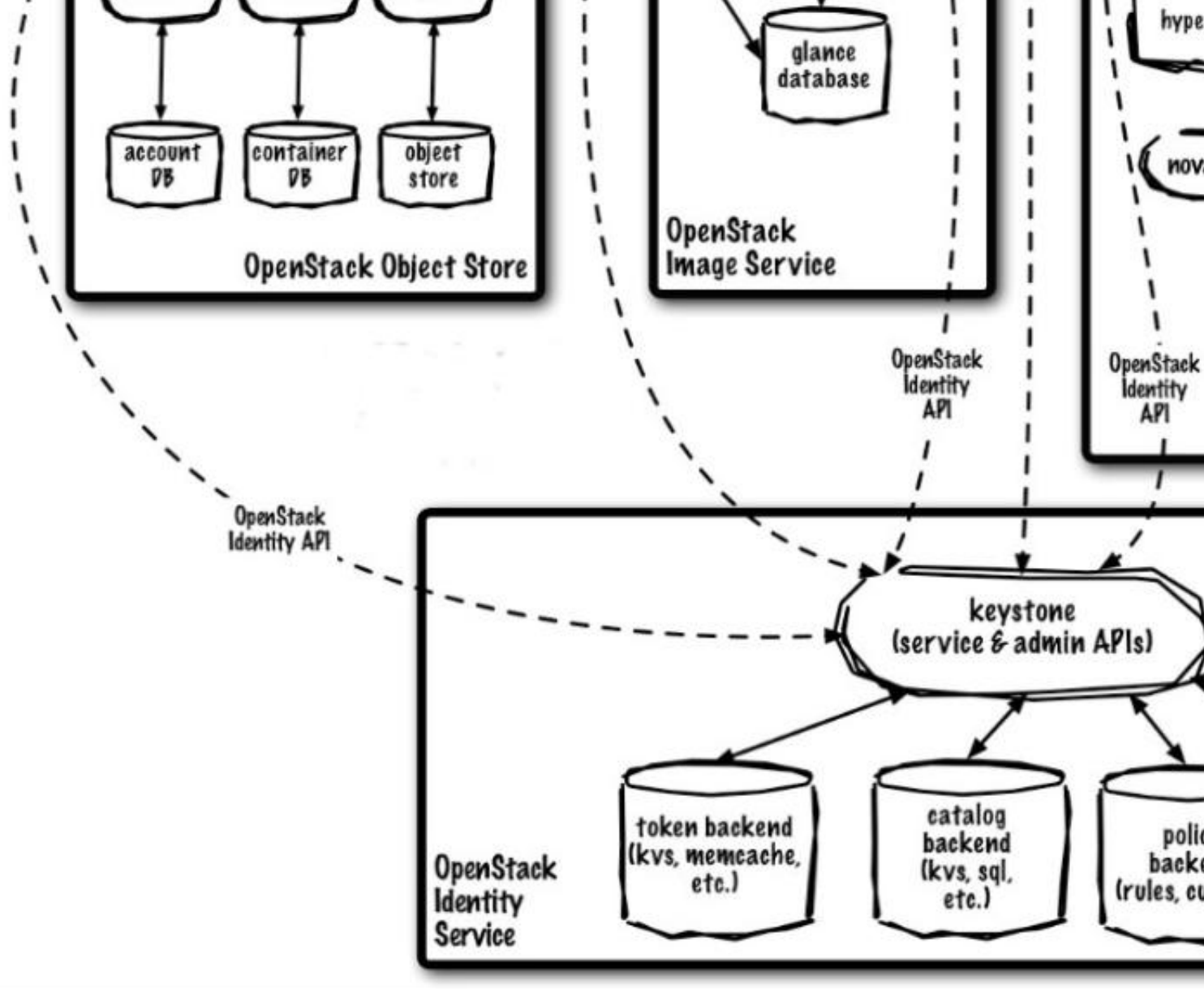
This is a stylized and simplified view of the architecture, assuming that the implementer is using all of the services together in the most common configuration. It also only shows the "operator" side of the cloud – it does not picture how consumers of the cloud may actually use it. For example, many users will access object storage heavily (and directly).

### **Logical Architecture**

This picture is consistent with the conceptual architecture above:



OpenStack



- End users can interact through a common web interface (Horizon) or directly to each service through their API
- All services authenticate through a common source (facilitated through keystone)
- Individual services interact with each other through their public APIs (except where privileged administrator commands are necessary)

In the sections below, we'll delve into the architecture for each of the services.

### **Dashboard**

Horizon is a modular Django web application that provides an end user and administrator interface to OpenStack services.

PROJECT  
**invisible\_to\_admin** ▼

## Manage Compute

Overview

Instances & Volumes

Access & Security

Images & Snapshots

## Object Store

Containers

|                          | Name | IP Address | Size      |
|--------------------------|------|------------|-----------|
| <input type="checkbox"/> | test |            | 512MB RAM |

Displaying 1 item

## Volumes

|                    | Name | Description |
|--------------------|------|-------------|
| Displaying 0 items |      |             |

As with most web applications, the architecture is fairly simple:

- Horizon is usually deployed via `mod_wsgi` in Apache. The code itself is separated into a reusable python module with most of the logic (interactions with various OpenStack APIs) and presentation (to make it easily customizable for different sites).
- A database (configurable as to which one) which relies mostly on the other services for data. It also stores very little data of its own.

From a network architecture point of view, this service will need to be customer accessible as well as be able to talk to each service's public APIs. If you wish to use the administrator functionality (i.e. for other services), it will also need connectivity to their Admin API endpoints (which should be non-customer accessible).

## Compute

Nova is the most complicated and distributed component of OpenStack. A large number of processes cooperate to turn end user API requests into running virtual machines. Below is a list of these processes and their functions:

- `nova-api` accepts and responds to end user compute API calls. It supports OpenStack Compute API, Amazon's EC2 API and a special Admin API (for privileged users to perform administrative actions). It also initiates most of the orchestration activities (such as running an instance) as well as enforces some policy (mostly quota checks).
- The `nova-compute` process is primarily a worker daemon that creates and terminates virtual machine instances via hypervisor's APIs (XenAPI for XenServer/XCP, `libvirt` for KVM or QEMU, `VMwareAPI` for VMware, etc.). The process by which it does so is fairly complex but the basics are simple: accept actions from the queue and then perform a series of system commands (like launching a KVM instance) to carry them out while updating state in the database.
- `nova-volume` manages the creation, attaching and detaching of z volumes to compute instances (similar functionality to Amazon's Elastic Block Storage). It can use volumes from a variety of providers such as iSCSI or Rados Block Device in Ceph. A new OpenStack project, Cinder, will eventually replace nova-

volume functionality. In the Folsom release, nova-volume and the Block Storage service will have similar functionality.

- The nova-network worker daemon is very similar to nova-compute and nova-volume. It accepts networking tasks from the queue and then performs tasks to manipulate the network (such as setting up bridging interfaces or changing iptables rules). This functionality is being migrated to Neutron, a separate OpenStack project. In the Folsom release, much of the functionality will be duplicated between nova-network and Neutron.
- The nova-schedule process is conceptually the simplest piece of code in OpenStack Nova: it takes a virtual machine instance request from the queue and determines where it should run (specifically, which compute server host it should run on).
- The queue provides a central hub for passing messages between daemons. This is usually implemented with RabbitMQ today, but could be any AMQP message queue (such as Apache Qpid). New to the Folsom release is support for Zero MQ.
- The SQL database stores most of the build-time and runtime state for a cloud infrastructure. This includes the instance types that are available for use, instances in use, networks available and projects. Theoretically, OpenStack Nova can support any database supported by SQL-Alchemy but the only databases currently being widely used are SQLite3 (only appropriate for test and development work), MySQL and PostgreSQL.
- Nova also provides console services to allow end users to access their virtual instance's console through a proxy. This involves several daemons (nova-console, nova-novncproxy and nova-consoleauth).

Nova interacts with many other OpenStack services: Keystone for authentication, Glance for images and Horizon for web interface. The Glance interactions are central. The API process can upload and query Glance while nova-compute will download images for use in launching images.

### **Object Store**

The swift architecture is very distributed to prevent any single point of failure as well as to scale horizontally. It includes the following components:

- Proxy server (swift-proxy-server) accepts incoming requests via the OpenStack Object API or just raw HTTP. It accepts files to upload, modifications to metadata or container creation. In addition, it will also serve files or container listing to web browsers. The proxy server may utilize an optional cache (usually deployed with memcache) to improve performance.
- Account servers manage accounts defined with the object storage service.
- Container servers manage a mapping of containers (i.e folders) within the object store service.
- Object servers manage actual objects (i.e. files) on the storage nodes.
- There are also a number of periodic processes which run to perform housekeeping tasks on the large data store. The most important of these is the replication services, which ensures consistency and availability through the cluster. Other periodic processes include auditors, updaters and reapers.

Authentication is handled through configurable WSGI middleware (which will usually be Keystone).

### Image Store

The Glance architecture has stayed relatively stable since the Cactus release. The biggest architectural change has been the addition of authentication, which was added in the Diablo release. Just as a quick reminder, Glance has four main parts to it:

- glance-api accepts Image API calls for image discovery, image retrieval and image storage.
- glance-registry stores, processes and retrieves metadata about images (size, type, etc.).
- A database to store the image metadata. Like Nova, you can choose your database depending on your preference (but most people use MySQL or SQLite).
- A storage repository for the actual image files. In the diagram above, Swift is shown as the image repository, but this is configurable. In addition to Swift, Glance supports normal filesystems, RADOS block devices, Amazon S3 and HTTP. Be aware that some of these choices are limited to read-only usage.

There are also a number of periodic processes which run on Glance to support caching. The most important of these is the replication services, which ensures consistency and availability through the cluster. Other periodic processes include auditors, updaters and reapers.

As you can see from the diagram in the Conceptual Architecture section, Glance serves a central role to the overall IaaS picture. It accepts API requests for images (or image metadata) from end users or Nova components and can store its disk files in the object storage service, Swift.

### **Identity**

Keystone provides a single point of integration for OpenStack policy, catalog, token and authentication.

- Keystone handles API requests as well as providing configurable catalog, policy, token and identity services.
- Each Keystone function has a pluggable backend which allows different ways to use the particular service. Most support standard backends like LDAP or SQL, as well as Key Value Stores (KVS).

Most people will use this as a point of customization for their current authentication services.

### **Network**

Neutron provides "network connectivity as a service" between interface devices managed by other OpenStack services (most likely Nova). The service works by allowing users to create their own networks and then attach interfaces to them. Like many of the OpenStack services, Neutron is highly configurable due to its plug-in architecture. These plug-ins accommodate different networking equipment and software. As such, the architecture and deployment can vary dramatically. In the above architecture, a simple Linux networking plug-in is shown.

- neutron-server accepts API requests and then routes them to the appropriate Neutron plug-in for action.
- Neutron plug-ins and agents perform the actual actions such as plugging and unplugging ports, creating networks or subnets and IP addressing. These plug-ins and agents differ depending on the vendor and

technologies used in the particular cloud. Neutron ships with plug-ins and agents for: Cisco virtual and physical switches, NEC OpenFlow products, Open vSwitch, Linux bridging, the Ryu Network Operating System, and VMware NSX.

- The common agents are L3 (layer 3), DHCP (dynamic host IP addressing) and the specific plug-in agent.
- Most Neutron installations will also make use of a messaging queue to route information between the neutron-server and various agents as well as a database to store networking state for particular plug-ins.

Neutron will interact mainly with Nova, where it will provide networks and connectivity for its instances.

### **Block Storage**

Cinder separates out the persistent block storage functionality that was previously part of OpenStack Compute (in the form of nova-volume) into its own service. The OpenStack Block Storage API allows for manipulation of volumes, volume types (similar to compute flavors) and volume snapshots.

- cinder-api accepts API requests and routes them to cinder-volume for action.
- cinder-volume acts upon the requests by reading or writing to the Cinder database to maintain state, interacting with other processes (like cinder-scheduler) through a message queue and directly upon block storage providing hardware or software. It can interact with a variety of storage providers through a driver architecture. Currently, there are drivers for IBM, SolidFire, NetApp, Nexenta, Zadara, linux iSCSI and other storage providers.
- Much like nova-scheduler, the cinder-scheduler daemon picks the optimal block storage provider node to create the volume on.
- Cinder deployments will also make use of a messaging queue to route information between the cinder processes as well as a database to store volume state.

Like Neutron, Cinder will mainly interact with Nova, providing volumes for its instances.



# Review Associate Virtual Machine Provisioning Walk-Through

More Content To be Added ...

OpenStack Compute gives you a tool to orchestrate a cloud, including running instances, managing networks, and controlling access to the cloud through users and projects. The underlying open source project's name is Nova, and it provides the software that can control an Infrastructure-as-a-Service (IaaS) cloud computing platform. It is similar in scope to Amazon EC2 and Rackspace Cloud Servers. 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 web-based API.

## Hypervisors

OpenStack Compute requires a hypervisor and Compute controls the hypervisors through an API server. The process for selecting a hypervisor usually means prioritizing and making decisions based on budget and resource constraints as well as the inevitable list of supported features and required technical specifications. The majority of development is done with the KVM and Xen-based hypervisors. Refer to <http://wiki.openstack.org/HypervisorSupportMatrix> <http://goo.gl/n7AXnC> for a detailed list of features and support across the hypervisors.

With OpenStack Compute, you can orchestrate clouds using multiple hypervisors in different zones. The types of virtualization standards that may be used with Compute include:

- KVM- Kernel-based Virtual Machine (visit <http://goo.gl/70dvRb>)
- LXC- Linux Containers (through libvirt) (visit <http://goo.gl/Ous3ly>)
- QEMU- Quick EMUlator (visit <http://goo.gl/WWV9IL>)
- UML- User Mode Linux (visit <http://goo.gl/4HAKJj>)

- VMWare vSphere 4.1 update 1 and newer (visit <http://goo.gl/0DBeo5>)
- Xen- Xen, Citrix XenServer and Xen Cloud Platform (XCP) (visit <http://goo.gl/yXP9t1>)
- Bare Metal- Provisions physical hardware via pluggable sub-drivers. (visit <http://goo.gl/exfeSg>)

### Users and Tenants (Projects)

The OpenStack Compute system is designed to be used by many different cloud computing consumers or customers, basically tenants on a shared system, using role-based access assignments. Roles control the actions that a user is allowed to perform. In the default configuration, most actions do not require a particular role, but this is configurable by the system administrator editing the appropriate `policy.json` file that maintains the rules. For example, a rule can be defined so that a user cannot allocate a public IP without the admin role. A user's access to particular images is limited by tenant, but the username and password are assigned per user. Key pairs granting access to an instance are enabled per user, but quotas to control resource consumption across available hardware resources are per tenant.

While the original EC2 API supports users, OpenStack Compute adds the concept of tenants. Tenants are isolated resource containers forming the principal organizational structure within the Compute service. They consist of a separate VLAN, volumes, instances, images, keys, and users. A user can specify which tenant he or she wishes to be known as by appending `:project_id` to his or her access key. If no tenant is specified in the API request, Compute attempts to use a tenant with the same ID as the user.

For tenants, quota controls are available to limit the:

- Number of volumes which may be created
- Total size of all volumes within a project as measured in GB
- Number of instances which may be launched
- Number of processor cores which may be allocated

- Floating IP addresses (assigned to any instance when it launches so the instance has the same publicly accessible IP addresses)
- Fixed IP addresses (assigned to the same instance each time it boots, publicly or privately accessible, typically private for management purposes)

## Images and Instances

This introduction provides a high level overview of what images and instances are and description of the life-cycle of a typical virtual system within the cloud. There are many ways to configure the details of an OpenStack cloud and many ways to implement a virtual system within that cloud. These configuration details as well as the specific command-line utilities and API calls to perform the actions described are presented in the Image Management and Volume Management chapters.

Images are disk images which are templates for virtual machine file systems. The OpenStack Image Service is responsible for the storage and management of images within OpenStack.

Instances are the individual virtual machines running on physical compute nodes. The OpenStack Compute service manages instances. Any number of instances maybe started from the same image. Each instance is run from a copy of the base image so runtime changes made by an instance do not change the image it is based on. Snapshots of running instances may be taken which create a new image based on the current disk state of a particular instance.

When starting an instance a set of virtual resources known as a flavor must be selected. Flavors define how many virtual CPUs an instance has and the amount of RAM and size of its ephemeral disks. OpenStack provides a number of predefined flavors which cloud administrators may edit or add to. Users must select from the set of available flavors defined on their cloud.

Additional resources such as persistent volume storage and public IP address may be added to and removed from running instances. The examples below show the cinder-volume service which provide persistent block storage as opposed to the ephemeral storage provided by the instance flavor.

Here is an example of the life cycle of a typical virtual system within an OpenStack cloud to illustrate these concepts.

### Initial State

#### Images and Instances

The following diagram shows the system state prior to launching an instance. The image store fronted by the Image Service has some number of predefined images. In the cloud, there is an available compute node with available vCPU, memory and local disk resources. Plus there are a number of predefined volumes in the cinder - volume service.

Figure 2.1. Base image state with no running instances

**Figure 1.11. Initial State**

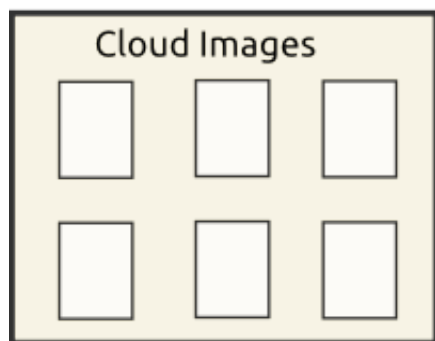


Image Store



Compute Node

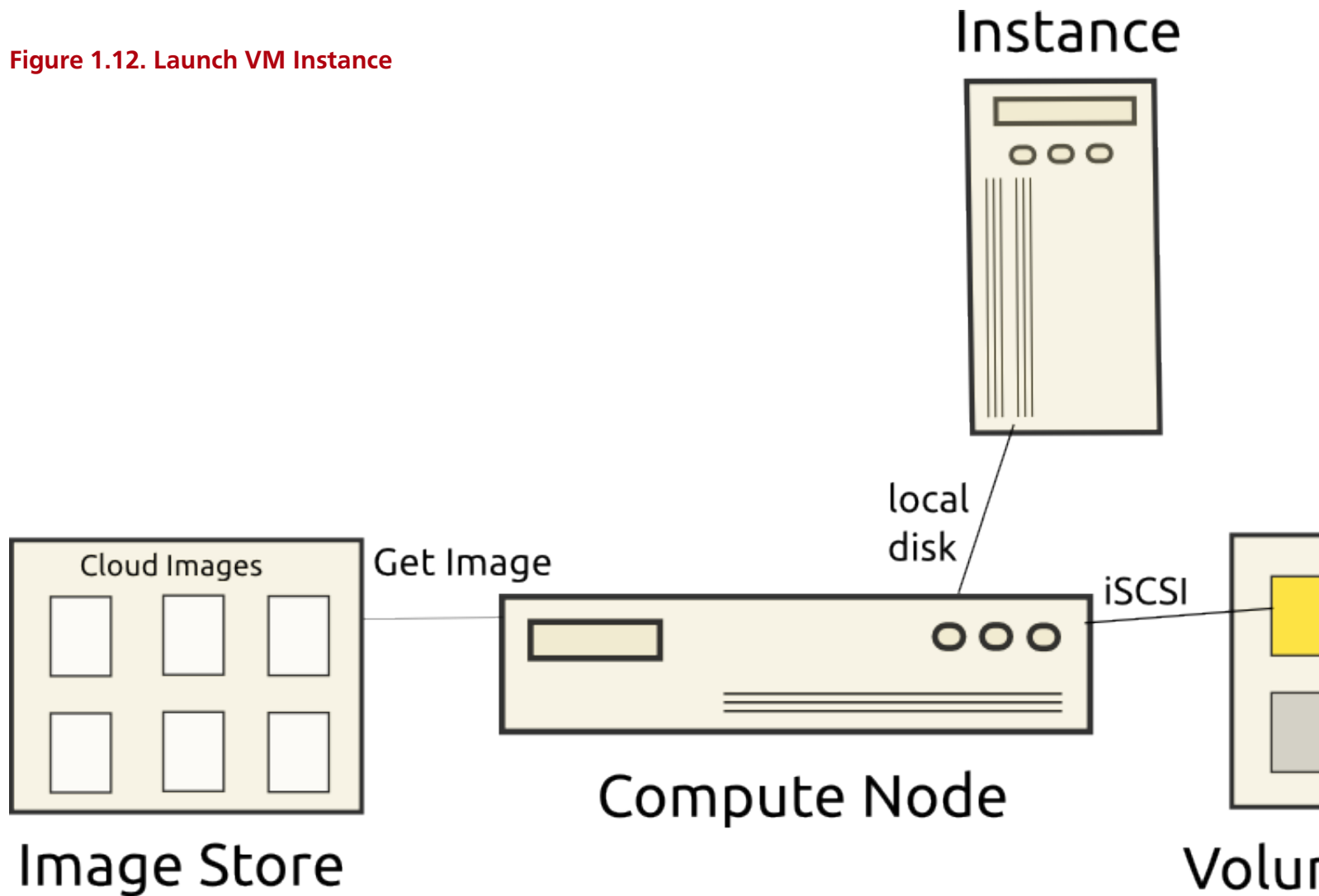


Block

Launching an instance

To launch an instance, the user selects an image, a flavor, and other optional attributes. In this case the selected flavor provides a root volume (as all flavors do) labeled vda in the diagram and additional ephemeral storage labeled vdb in the diagram. The user has also opted to map a volume from the cinder -volume store to the third virtual disk, vdc, on this instance.

Figure 2.2. Instance creation from image and run time state

**Figure 1.12. Launch VM Instance**

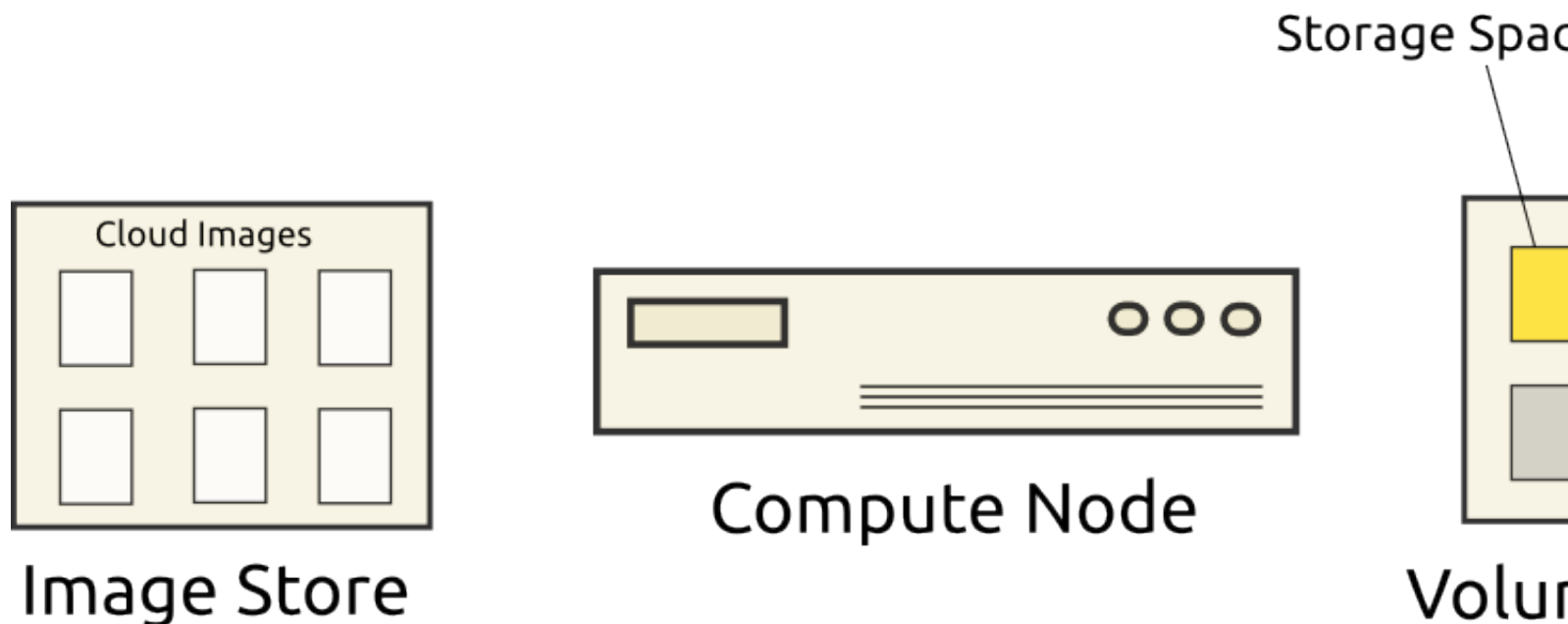
The OpenStack system copies the base image from the image store to local disk which is used as the first disk of the instance (vda). Having small images will result in faster start up of your instances as less data needs to be copied across the network. The system also creates a new empty disk image to present as the second disk (vdb). Be aware that the second disk is an empty disk with an ephemeral life as it is destroyed when you delete the instance. The compute node attaches to the requested cinder -volume using iSCSI and maps this to the third disk (vdc) as requested. The vCPU and memory resources are provisioned and the instance is booted from the first drive. The instance runs and changes data on the disks indicated in red in the diagram.

There are many possible variations in the details of the scenario, particularly in terms of what the backing storage is and the network protocols used to attach and move storage. One variant worth mentioning here is that the ephemeral storage used for volumes vda and vdb in this example may be backed by network storage rather than local disk. The details are left for later chapters.

### **End State**

Once the instance has served its purpose and is deleted all state is reclaimed, except the persistent volume. The ephemeral storage is purged. Memory and vCPU resources are released. And of course the image has remained unchanged throughout.

Figure 2.3. End state of image and volume after instance exits

**Figure 1.13. End State**

Once you launch a VM in OpenStack, there's something more going on in the background. To understand what's happening behind the dashboard, let's take a deeper dive into OpenStack's VM provisioning. For launching a VM, you can either use the command-line interfaces or the OpenStack dashboard.



## 2. Getting Started Lab

### Table of Contents

|                                                          |    |
|----------------------------------------------------------|----|
| Day 1, 13:30 to 14:45, 15:00 to 17:00 .....              | 41 |
| Getting the Tools and Accounts for Committing Code ..... | 41 |
| Fix a Documentation Bug .....                            | 45 |
| Submit a Documentation Bug .....                         | 49 |
| Create a Branch .....                                    | 49 |
| Optional: Add to the Training Guide Documentation .....  | 51 |

### Day 1, 13:30 to 14:45, 15:00 to 17:00

## Getting the Tools and Accounts for Committing Code



#### Note

First create a GitHub account at [github.com](https://github.com).



#### Note

Check out <https://wiki.openstack.org/wiki/Documentation/HowTo> for more extensive setup instructions.

1. Download and install Git from <http://git-scm.com/downloads>.

2. Create your local repository directory:

```
$ mkdir /Users/username/code/
```

3. **Install SourceTree**

- a. <http://www.sourcetreeapp.com/download/>.
- b. Ignore the Atlassian Bitbucket and Stack setup.
- c. Add your GitHub username and password.
- d. Set your local repository location.

4. **Install an XML editor**

- a. You can download a 30 day trial of Oxygen. The floating licenses donated by OxygenXML have all been handed out. [http://www.oxygenxml.com/download\\_oxygenxml\\_editor.html](http://www.oxygenxml.com/download_oxygenxml_editor.html)
- b. AND/OR PyCharm <http://download.jetbrains.com/python/pycharm-community-3.0.1.dmg>
- c. AND/OR You can use emacs or vi editors.

Here are some great resources on DocBook and Emacs' NXML mode:

- <http://paul.frields.org/2011/02/09/xml-editing-with-emacs/>
- [https://fedoraproject.org/wiki/How\\_to\\_use\\_Emacs\\_for\\_XML\\_editing](https://fedoraproject.org/wiki/How_to_use_Emacs_for_XML_editing)
- <http://infohost.nmt.edu/tcc/help/pubs/nxml/>

If you prefer vi, there are ways to make DocBook editing easier:

- [https://fedoraproject.org/wiki/Editing\\_DocBook\\_with\\_Vi](https://fedoraproject.org/wiki/Editing_DocBook_with_Vi)

## 5. Install Maven

- a. Create the apache-maven directory:

```
# mkdir /usr/local/apache-maven
```

- b. Copy the latest stable binary from <http://maven.apache.org/download.cgi> into /usr/local/apache-maven.

- c. Extract the distribution archive to the directory you wish to install Maven:

```
# cd /usr/local/apache-maven/  
# tar -xvzf apache-maven-x.x.x-bin.tar.gz
```

The apache-maven-x.x.x subdirectory is created from the archive file, where x.x.x is your Maven version.

- d. Add the M2\_HOME environment variable:

```
$ export M2_HOME=/usr/local/apache-maven/apache-maven-x.x.x
```

- e. Add the M2 environment variable:

```
$ export M2=$M2_HOME/bin
```

- f. Optionally, add the MAVEN\_OPTS environment variable to specify JVM properties. Use this environment variable to specify extra options to Maven:

```
$ export MAVEN_OPTS=' -Xms256m -XX:MaxPermSize=1024m -Xmx1024m '
```

- g. Add the M2 environment variable to your path:

```
$ export PATH=$M2:$PATH
```

- h. Make sure that JAVA\_HOME is set to the location of your JDK and that \$JAVA\_HOME/bin is in your PATH environment variable.
- i. Run the mvn command to make sure that Maven is correctly installed:

```
$ mvn --version
```

- 6. Create a Launchpad account: Visit [https://login.launchpad.net/+new\\_account](https://login.launchpad.net/+new_account). After you create this account, the follow-up page is slightly confusing. It does not tell you that you are done. (It gives you the opportunity to change your -password, but you do not have to.)
- 7. Add at least one SSH key to your account profile. To do this, follow the instructions on <https://help.launchpad.net/YourAccount/CreatingAnSSHKeyPair>".
- 8. Join The OpenStack Foundation: Visit <https://www.openstack.org/join>. Among other privileges, this membership enables you to vote in elections and run for elected positions in The OpenStack Project. When you sign up for membership, make sure to give the same e-mail address you will use for code contributions because the primary e-mail address in your foundation profile must match the preferred e-mail that you set later in your Gerrit contact information.
- 9. Validate your Gerrit identity: Add your public key to your gerrit identity by going to <https://review.openstack.org>, click the **Sign In** link, if you are not already logged in. At the top-right corner of the page select settings, then add your public ssh key under **SSH Public Keys**.

The CLA: Every developer and contributor needs to sign the Individual Contributor License agreement. Visit <https://review.openstack.org/> and click the **Sign In** link at the top-right corner of the page. Log in with your Launchpad ID. You can preview the text of the Individual CLA.

- 10. Add your SSH keys to your GitHub account profile (the same one that was used in Launchpad). When you copy and paste the SSH key, include the ssh-rsa algorithm and computer identifier. If this is your first time setting up git and Github, be sure to run these steps in a Terminal window:

```
$ git config --global user.name "Firstname Lastname"
```

```
$ git config --global user.email "your_email@youremail.com"
```

11. Install git-review. If pip is not already installed, run `easy_install pip` as root to install it on a Mac or Ubuntu.

```
# pip install git-review
```

12. Change to the directory:

```
$ cd /Users/username/code
```

13. Clone the openstack-manuals repository:

```
$ git clone http://github.com/openstack/openstack-manuals.git
```

14. Change directory to the pulled repository:

```
$ cd openstack-manuals
```

15. Test the ssh key setup:

```
$ git review -s
```

Then, enter your Launchpad account information.

## Fix a Documentation Bug

- 1.



### Note

For this example, we are going to assume bug 1188522 and change 33713

2. Bring up <https://bugs.launchpad.net/openstack-manuals>
3. Select an unassigned bug that you want to fix. Start with something easy, like a syntax error.

4. Using oXygen, open the `/Users/username/code/openstack-manuals/doc/admin-guide-cloud/bk-admin-guide-cloud.xml` master page for this example. It links together the rest of the material. Find the page with the bug. Open the page that is referenced in the bug description by selecting the content in the author view. Verify you have the correct page by visually inspecting the html page and the xml page.
5. In the shell,  

```
$ cd /Users/username/code/openstack-manuals/doc/admin-guide-cloud/
```
6. Verify that you are on master:  

```
$ git checkout master
```
7. Create your working branch off master:  

```
$ git checkout -b bug/1188522
```
8. Verify that you have the branch open through SourceTree
9. Correct the bug through oXygen. Toggle back and forth through the different views at the bottom of the editor.
10. After you fix the bug, run maven to verify that the documentation builds successfully. To build a specific guide, look for a `pom.xml` file within a subdirectory, switch to that directory, then run the **mvn** command in that directory:  

```
$ mvn clean generate-sources
```
11. Verify that the HTML page reflects your changes properly. You can open the file from the command line by using the **open** command  

```
$ open target/docbkx/webhelp/local/openstack-training/index.html
```
12. Add the changes:

```
$ git add .
```

13. Commit the changes:

```
$ git commit -a -m "Removed reference to volume scheduler in the computer scheduler config  
and admin pages, bug 1188522"
```

14. Build committed changes locally by using **tox**. As part of the review process, Jenkins runs gating scripts to check that the patch is fine. Locally, you can use the **tox** tool to run the same checks and ensure that a patch works. Install the tox package and run it from the top level directory which has the `tox.ini` file.

```
# pip install tox  
$ tox
```

Jenkins runs the following four checks. You can run them individually:

- a. Niceness tests (for example, to see extra whitespaces). Verify that the niceness check succeeds.

```
$ tox -e checkniceness
```

- b. Syntax checks. Verify that the syntax check succeeds.

```
$ tox -e checksyntax
```

- c. Check that no deleted files are referenced. Verify that the check succeeds.

```
$ tox -e checkdeletions
```

- d. Build the manuals. It also generates a directory `publish-docs/` that contains the built files for inspection. You can also use `doc/local-files.html` for looking at the manuals. Verify that the build succeeds.

```
$ tox -e checkbuild
```

15. Submit the bug fix to Gerrit:

```
$ git review
```

16. Track the Gerrit review process at <https://review.openstack.org/#/c/33713>. Follow and respond inline to the Code Review requests and comments.
17. Your change will be tested, track the Jenkins testing process at <https://jenkins.openstack.org>
18. If your change is rejected, complete the following steps:
  - a. Respond to the inline comments if any.
  - b. Update the status to work in progress.
  - c. Checkout the patch from the Gerrit change review:

```
$ git review -d 33713
```
  - d. Follow the recommended tweaks to the files.
  - e. Rerun:

```
$ mvn clean generate-sources
```
  - f. Add your additional changes to the change log:

```
$ git commit -a --amend
```
  - g. Final commit:

```
$ git review
```
  - h. Update the Jenkins status to change completed.



19. Follow the jenkins build progress at <https://jenkins.openstack.org/view/Openstack-manuals/> . Note if the build process fails, the online documentation will not reflect your bug fix.

## Submit a Documentation Bug

1. Bring up <https://bugs.launchpad.net/openstack-manuals/+filebug>.
2. Give your bug a descriptive name.
3. Verify if asked that it is not a duplicate.
4. Add some more detail into the description field.
5. Once submitted, select the assigned to pane and select "assign to me" or "sarob".
6. Follow the instructions for fixing a bug in the Fix a Documentation Bug section.

## Create a Branch



### Note

This section uses the submission of this training material as the example.

1. Create a bp/training-manuals branch:  

```
$ git checkout -b bp/training-manuals
```
2. From the openstack-manuals repository, use the template `user-story-includes-template.xml` as the starting point for your user story. File `bk001-ch003-associate-general.xml` has at least one other included user story that you can use for additional help.
3. Include the user story xml file into the `bk001-ch003-associate-general.xml` file. Follow the syntax of the existing `xi:include` statements.

4. When your editing is completed. Double check Oxygen doesn't have any errors you are not expecting.
5. Run maven locally to verify the build will run without errors. Look for a `pom.xml` file within a subdirectory, switch to that directory, then run the **mvn** command in that directory:

```
$ mvn clean generate-sources
```

6. Add your changes into git:

```
$ git add .
```

7. Commit the changes with good syntax. After entering the commit command, VI syntax applies, use "i" to insert and Esc to break out. ":wq" to write and quit.

```
$ git commit -a
      my very short summary
more details go here. A few sentences would be nice.
blueprint training-manuals
```

8. Build committed changes locally using **tox**. As part of the review process, Jenkins runs gating scripts to check that the patch is fine. Locally, you can use the **tox** tool to run the same checks and ensure that a patch works. Install the tox package and run it from the top level directory which has the `tox.ini` file.

```
# pip install tox
$ tox
```

9. Submit your patch for review:

```
$ git review
```

10. One last step. Go to the review page listed after you submitted your review and add the training core team as reviewers; Sean Roberts and Colin McNamara.
11. More details on branching can be found here under [Gerrit Workflow](#) and the [Git docs](#).

## Optional: Add to the Training Guide Documentation

1. **Getting Accounts and Tools:** We cannot do this without operators and developers using and creating the content. Anyone can contribute content. You will need the tools to get started. Go to the [Getting Tools and Accounts](#) page.
2. **Pick a Card:** Once you have your tools ready to go, you can assign some work to yourself. Go to the [Training Trello/KanBan storyboard](#) and assign a card / user story from the Sprint Backlog to yourself. If you do not have a Trello account, no problem, just create one. Email seanrob@yahoo-inc.com and you will have access. Move the card from the Sprint Backlog to Doing.
3. **Create the Content:** Each card / user story from the KanBan story board will be a separate chunk of content you will add to the openstack-manuals repository openstack-training sub-project.
4. Open the file `st-training-guides.xml` with your XML editor. All the content starts with the set file `st-training-guides.xml`. The XML structure follows the hierarchy Set -> Book -> Chapter -> Section. The `st-training-guides.xml` file holds the set level. Notice the set file uses `xi:include` statements to include the books. We want to open the associate book. Open the associate book and you will see the chapter include statements. These are the chapters that make up the Associate Training Guide book.
5. Create a branch by using the card number as **associate-card-XXX** where XXX is the card number. Review [Creating a Branch](#) again for instructions on how to complete the branch merge.
6. Copy the `user-story-includes-template.xml` to `associate-card-XXX.xml`.
7. Open the `bk001-ch003-associate-general.xml` file and add `<xi:include href="associate-card-XXX.xml">`.
8. Side by side, open **associate-card-XXX.xml** with your XML editor and open the *Ubuntu 12.04 Install Guide* with your HTML browser.
9. Find the HTML content to include. Find the XML file that matches the HTML. Include the whole page using a simple href like `<xi:include href="associate-card-XXX.xml">` or include a section using xpath like

`<xi:include href="../../basic-install/src/basic-install_controller-common.xml" xpointer="xmlns(db=http://docbook.org/ns/docbook) xpath(//*[@xml:id = 'controller-os'])">` . Review the `user-story-  
includes-template.xml` file for the whole syntax.

10. Copy in other content sources including the Aptira content, a description of what the section aims to teach, diagrams, and quizzes. If you include content from another source like Aptira content, add a paragraph that references the file and/or HTTP address from where the content came.
11. Verify the code is good by running **`mvn clean generate-sources`** and by reviewing the local HTML in <file:///Users/username/code/openstack-manuals/doc/training-guides/target/docbkx/webhelp/training-guides/content/>.
12. Merge the branch.
13. Move the card from Doing to Done.

## 3. Getting Started Quiz

### Table of Contents

|                             |    |
|-----------------------------|----|
| Day 1, 16:40 to 17:00 ..... | 53 |
|-----------------------------|----|

### Day 1, 16:40 to 17:00



## 4. Controller Node

### Table of Contents

|                                                           |     |
|-----------------------------------------------------------|-----|
| Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30 .....          | 55  |
| Review Associate Overview Horizon and OpenStack CLI ..... | 55  |
| Review Associate Keystone Architecture .....              | 105 |
| Review Associate OpenStack Messaging and Queues .....     | 110 |
| Review Associate Administration Tasks .....               | 121 |

### Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30

### Review Associate Overview Horizon and OpenStack CLI

#### How can I use an OpenStack cloud?

As an OpenStack cloud end user, you can provision your own resources within the limits set by administrators. The examples in this guide show you how to complete these tasks by using the OpenStack dashboard and command-line clients. The dashboard, also known as horizon, is a Web-based graphical interface. The command-line clients let you run simple commands to create and manage resources in a cloud and automate tasks by using scripts. Each of the core OpenStack projects has its own command-line client.

You can modify these examples for your specific use cases.

In addition to these ways of interacting with a cloud, you can access the OpenStack APIs indirectly through cURL commands or open SDKs, or directly through the APIs. You can automate access or build tools to manage resources and services by using the native OpenStack APIs or the EC2 compatibility API.

To use the OpenStack APIs, it helps to be familiar with HTTP/1.1, RESTful web services, the OpenStack services, and JSON or XML data serialization formats.

## OpenStack dashboard

As a cloud end user, the OpenStack dashboard lets you to provision your own resources within the limits set by administrators. You can modify these examples to create other types and sizes of server instances.

### Overview

The following requirements must be fulfilled to access the OpenStack dashboard:

- The cloud operator has set up an OpenStack cloud.
- You have a recent Web browser that supports HTML5. It must have cookies and JavaScript enabled. To use the VNC client for the dashboard, which is based on noVNC, your browser must support HTML5 Canvas and HTML5 WebSockets. For more details and a list of browsers that support noVNC, see <https://github.com/kanaka/noVNC/blob/master/README.md> and <https://github.com/kanaka/noVNC/wiki/Browser-support>, respectively.

Learn how to log in to the dashboard and get a short overview of the interface.

### Log in to the dashboard

To log in to the dashboard

1. Ask your cloud operator for the following information:

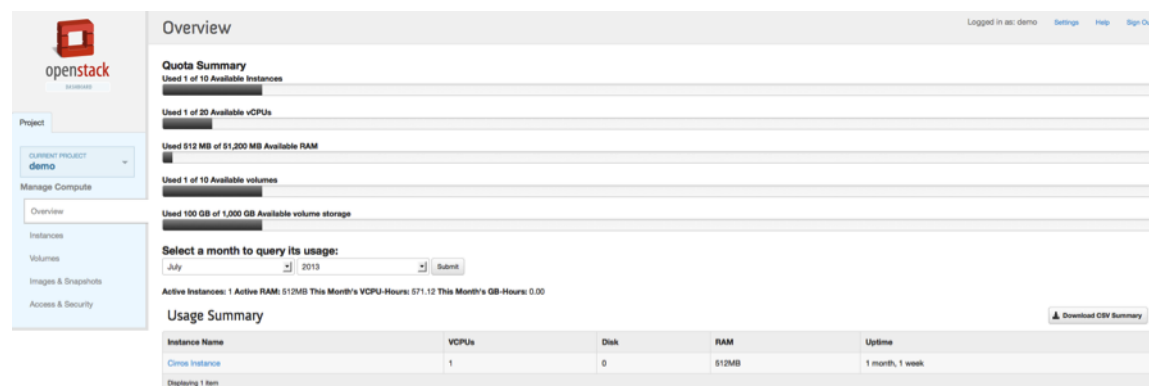
- The hostname or public IP address from which you can access the dashboard.
- The dashboard is available on the node that has the nova-dashboard server role.



- The username and password with which you can log in to the dashboard.
1. Open a Web browser that supports HTML5. Make sure that JavaScript and cookies are enabled.
  2. As a URL, enter the host name or IP address that you got from the cloud operator.
  3. [https://IP\\_ADDRESS\\_OR\\_HOSTNAME/](https://IP_ADDRESS_OR_HOSTNAME/)
  4. On the dashboard log in page, enter your user name and password and click Sign In.

After you log in, the following page appears:

**Figure 4.1. OpenStack Dashboard - Overview**



The top-level row shows the username that you logged in with. You can also access Settings or Sign Out of the Web interface.

If you are logged in as an end user rather than an admin user, the main screen shows only the Project tab.

## OpenStack dashboard – Project tab

This tab shows details for the projects, or projects, of which you are a member.

Select a project from the drop-down list on the left-hand side to access the following categories:

### **Overview**

Shows basic reports on the project.

### **Instances**

Lists instances and volumes created by users of the project.

From here, you can stop, pause, or reboot any instances or connect to them through virtual network computing (VNC).

### **Volumes**

Lists volumes created by users of the project.

From here, you can create or delete volumes.

### **Images & Snapshots**

Lists images and snapshots created by users of the project, plus any images that are publicly available. Includes volume snapshots. From here, you can create and delete images and snapshots, and launch instances from images and snapshots.

### **Access & Security**

On the **Security Groupstab**, you can list, create, and delete security groups and edit rules for security groups.

On the **Keypairstab**, you can list, create, and import keypairs, and delete keypairs.

On the **Floating IPstab**, you can allocate an IP address to or release it from a project.

On the **API Accesstab**, you can list the API endpoints.

### **Manage images**

During setup of OpenStack cloud, the cloud operator sets user permissions to manage images. Image upload and management might be restricted to only cloud administrators or cloud operators. Though you can complete most tasks with the OpenStack dashboard, you can manage images through only the glance and nova clients or the Image Service and Compute APIs.

### **Set up access and security**

Before you launch a virtual machine, you can add security group rules to enable users to ping and SSH to the instances. To do so, you either add rules to the default security group or add a security group with rules. For information, see [the section called "Add security group rules"](#).

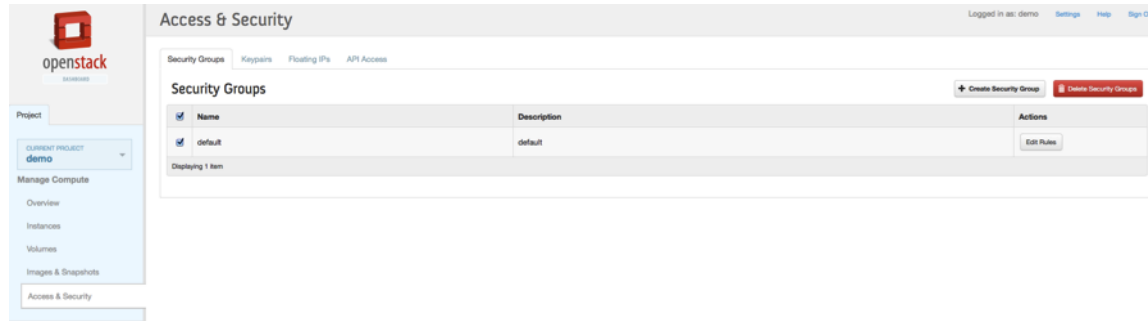
Keypairs are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the cloud-init package. For information, see [the section called "Add keypairs"](#).

### **Add security group rules**

The following procedure shows you how to add rules to the default security group.

To add rules to the default security group

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Projecttab.
3. Click the Access & Securitycategory.
4. The dashboard shows the security groups that are available for this project.

**Figure 4.2. OpenStack Dashboard - Security Groups**

1. Select the default security group and click Edit Rules.
2. The Security Group Rulespage appears:

**Figure 4.3. OpenStack Dashboard - Security Group Rules**

1. Add a TCP rule
2. Click Add Rule.

3. The Add Rulewindow appears.

1. In the IP Protocol list, select TCP.

2. In the Open list, select Port.

3. In the Port box, enter 22.

4. In the Source list, select CIDR.

5. In the CIDR box, enter 0.0.0.0/0.

6. Click Add.

7. Port 22 is now open for requests from any IP address.

8. If you want to accept requests from a particular range of IP addresses, specify the IP address block in the CIDR box.

1. Add an ICMP rule

2. Click Add Rule.

3. The Add Rulewindow appears.

1. In the IP Protocol list, select ICMP.

2. In the Type box, enter -1.

3. In the Code box, enter -1.

4. In the Source list, select CIDR.

5. In the CIDR box, enter 0.0.0.0/0.

6. Click Add.

### **Add keypairs**

Create at least one keypair for each project. If you have generated a keypair with an external tool, you can import it into OpenStack. The keypair can be used for multiple instances that belong to a project.

To add a keypair

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Projecttab.
3. Click the Access & Securitycategory.
4. Click the Keypairstab. The dashboard shows the keypairs that are available for this project.
5. To add a keypair
6. Click Create Keypair.
7. The Create Keypairwindow appears.
  1. In the Keypair Namebox, enter a name for your keypair.
  2. Click Create Keypair.
  3. Respond to the prompt to download the keypair.
1. To import a keypair
2. Click Import Keypair.
3. The Import Keypairwindow appears.

1. In the Keypair Namebox, enter the name of your keypair.
2. In the Public Keybox, copy the public key.
3. Click Import Keypair.
1. Save the \*.pem file locally and change its permissions so that only you can read and write to the file:
2. `$ chmod 0600 MY_PRIV_KEY.pem`
3. Use the `ssh-add` command to make the keypair known to SSH:
4. `$ ssh-add MY_PRIV_KEY.pem`

The public key of the keypair is registered in the Nova database.

The dashboard lists the keypair in the Access & Security category.

### **Launch instances**

Instances are virtual machines that run inside the cloud. You can launch an instance directly from one of the available OpenStack images or from an image that you have copied to a persistent volume. The OpenStack Image Service provides a pool of images that are accessible to members of different projects.

#### **Launch an instance from an image**

When you launch an instance from an image, OpenStack creates a local copy of the image on the respective compute node where the instance is started.

To launch an instance from an image

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.

3. Click the Images & Snapshotcategory.
4. The dashboard shows the images that have been uploaded to OpenStack Image Service and are available for this project.
5. Select an image and click Launch.
6. In the Launch Imagewindow, specify the following:
  1. Enter an instance name to assign to the virtual machine.
  2. From the Flavordrop-down list, select the size of the virtual machine to launch.
  3. Select a keypair.
  4. In case an image uses a static root password or a static key set (neither is recommended), you do not need to provide a keypair to launch the instance.
  5. In Instance Count, enter the number of virtual machines to launch from this image.
  6. Activate the security groups that you want to assign to the instance.
  7. Security groups are a kind of cloud firewall that define which incoming network traffic should be forwarded to instances. For details, see[the section called "Add security group rules"](#).
  8. If you have not created any specific security groups, you can only assign the instance to the default security group.
  9. If you want to boot from volume, click the respective entry to expand its options.  
Set the options as described in[http://docs.openstack.org/user-guide/content/dashboard\\_launch\\_instances.html#dashboard\\_launch\\_instances\\_from\\_volumethe](http://docs.openstack.org/user-guide/content/dashboard_launch_instances.html#dashboard_launch_instances_from_volumethe) [section called "Launch an instance from a volume"](#).



1. Click Launch Instance. The instance is started on any of the compute nodes in the cloud.

After you have launched an instance, switch to the Instances category to view the instance name, its (private or public) IP address, size, status, task, and power state.

Figure 5. OpenStack dashboard – Instances

If you did not provide a keypair, security groups, or rules so far, by default the instance can only be accessed from inside the cloud through VNC at this point. Even pinging the instance is not possible. To access the instance through a VNC console, see [http://docs.openstack.org/user-guide/content/instance\\_console.html](http://docs.openstack.org/user-guide/content/instance_console.html) the section called “Get a console to an instance”.

### Launch an instance from a volume

You can launch an instance directly from an image that has been copied to a persistent volume.

In that case, the instance is booted from the volume, which is provided by nova-volume, through iSCSI.

For preparation details, see [http://docs.openstack.org/user-guide/content/dashboard\\_manage\\_volumes.html#create\\_or\\_delete\\_volumes](http://docs.openstack.org/user-guide/content/dashboard_manage_volumes.html#create_or_delete_volumes) the section called “Create or delete a volume”.

To boot an instance from the volume, especially note the following steps:

- To be able to select from which volume to boot, launch an instance from an arbitrary image. The image you select does not boot. It is replaced by the image on the volume that you choose in the next steps.
- In case you want to boot a Xen image from a volume, note the following requirement: The image you launch in must be the same type, fully virtualized or paravirtualized, as the one on the volume.
- Select the volume or volume snapshot to boot from.
- Enter a device name. Enter vda for KVM images or xvda for Xen images.

To launch an instance from a volume

You can launch an instance directly from one of the images available through the OpenStack Image Service or from an image that you have copied to a persistent volume. When you launch an instance from a volume, the procedure is basically the same as when launching an instance from an image in OpenStack Image Service, except for some additional steps.

1. Create a volume as described in [http://docs.openstack.org/user-guide/content/dashboard\\_manage\\_volumes.html#create\\_or\\_delete\\_volumesthe](http://docs.openstack.org/user-guide/content/dashboard_manage_volumes.html#create_or_delete_volumesthe) section called "Create or delete a volume".
2. It must be large enough to store an unzipped image.
3. Create an image.
4. For details, see [Creating images manually](#) in the *OpenStack Virtual Machine Image Guide*.
5. Launch an instance.
6. Attach the volume to the instance as described in [http://docs.openstack.org/user-guide/content/dashboard\\_manage\\_volumes.html#attach\\_volumes\\_to\\_instancethe](http://docs.openstack.org/user-guide/content/dashboard_manage_volumes.html#attach_volumes_to_instancethe) section called "Attach volumes to instances".
7. Assuming that the attached volume is mounted as `/dev/vdb`, use one of the following commands to copy the image to the attached volume:
  - For a raw image:
  - `$ cat IMAGE >/dev/null`
  - Alternatively, use `dd`.
  - For a non-raw image:

- `$ qemu-img convert -O raw IMAGE /dev/vdb`
  - For a \*.tar.bz2 image:
  - `$ tar xjfO IMAGE >/dev/null`
1. Only detached volumes are available for booting. Detach the volume.
  2. To launch an instance from the volume, continue with [http://docs.openstack.org/user-guide/content/dashboard\\_launch\\_instances.html#dashboard\\_launch\\_instances\\_from\\_image](http://docs.openstack.org/user-guide/content/dashboard_launch_instances.html#dashboard_launch_instances_from_image) the section called "Launch an instance from an image".
  3. You can launch an instance directly from one of the images available through the OpenStack Image Service. When you do that, OpenStack creates a local copy of the image on the respective compute node where the instance is started.
  4. SSH in to your instance

To SSH into your instance, you use the downloaded keypair file.

To SSH into your instance

1. Copy the IP address for your instance.
2. Use the SSH command to make a secure connection to the instance. For example:
3. `$ ssh -i MyKey.pem ubuntu@10.0.0.2`
4. A prompt asks, "Are you sure you want to continue connection (yes/no)?" Type yes and you have successfully connected.

### **Manage instances**

#### **Create instance snapshots**

**Figure 4.4. OpenStack Dashboard- Instances**

The screenshot shows the OpenStack Dashboard's 'Instances' page. On the left is a sidebar with the OpenStack logo and navigation links: Project, Manage Compute, Overview, Instances (selected), Volumes, Images & Snapshots, and Access & Security. The main content area is titled 'Instances' and shows a table of two instances. The first instance is 'Instance from volume' with IP 10.0.0.3 and size m1.tiny | 512MB RAM | 1 VCPU | 0 Disk. The second is 'myserver' with IP 10.0.0.2 and size m1.small | 2GB RAM | 1 VCPU | 20GB Disk. Both are 'Active' and 'Running'. The 'Actions' column for each instance has a 'Create Snapshot' button. At the top right of the main area are buttons for 'Launch Instance' and 'Terminate Instances'. The top right of the dashboard shows 'Logged in as: demo' with links for 'Settings' and 'Sign Out'.

| Instance Name        | IP Address | Size                                    | Keypair | Status | Task | Power State | Actions         |
|----------------------|------------|-----------------------------------------|---------|--------|------|-------------|-----------------|
| Instance from volume | 10.0.0.3   | m1.tiny   512MB RAM   1 VCPU   0 Disk   | -       | Active | None | Running     | Create Snapshot |
| myserver             | 10.0.0.2   | m1.small   2GB RAM   1 VCPU   20GB Disk | -       | Active | None | Running     | Create Snapshot |

To create instance snapshots

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.
3. Click the Instances category.
4. The dashboard lists the instances that are available for this project.
5. Select the instance of which to create a snapshot. From the Actions drop-down list, select Create Snapshot.
6. In the Create Snapshot window, enter a name for the snapshot. Click Create Snapshot. The dashboard shows the instance snapshot in the Images & Snapshots category.
7. To launch an instance from the snapshot, select the snapshot and click Launch. Proceed with <http://docs.openstack.org/user-guide/content/>

[dashboard\\_launch\\_instances.html#dashboard\\_launch\\_instances\\_from\\_image](#) the section called “Launch an instance from an image”.

### **Control the state of an instance**

To control the state of an instance

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.
3. Click the Instances category.
4. The dashboard lists the instances that are available for this project.
5. Select the instance for which you want to change the state.
6. In the More drop-down list in the Actions column, select the state.
7. Depending on the current state of the instance, you can choose to pause, un-pause, suspend, resume, soft or hard reboot, or terminate an instance.

**Figure 4.5. OpenStack Dashboard : Actions**

+

Launch Instance

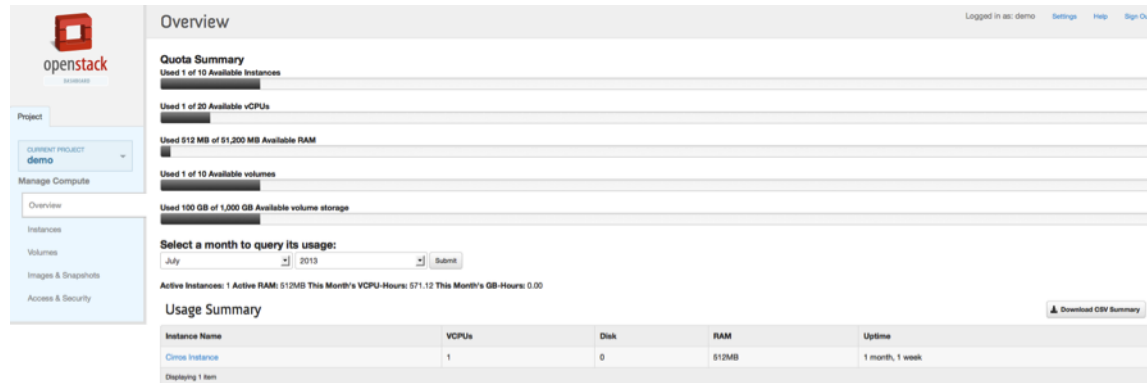
Terminate Instances

|  | Keypair    | Status | Task | Power State | Actions                                                                                                                                                                                                                                                                                                                                                                |
|--|------------|--------|------|-------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|  | OS-keypair | Active | None | Running     | <div>Create Snapshot</div> <div>More ▾</div> <div><div>Associate Floating IP</div><div>Disassociate Floating IP</div><div>Edit Instance</div><div>Edit Security Groups</div><div>Console</div><div>View Log</div><div>Pause Instance</div><div>Suspend Instance</div><div>Soft Reboot Instance</div><div>Hard Reboot Instance</div><div>Terminate Instance</div></div> |

## Track usage

Use the dashboard's Overviewcategory to track usage of instances for each project.

**Figure 4.6. OpenStack Dashboard - Track Usage**



You can track costs per month by showing metrics like number of VCPUs, disks, RAM, and uptime of all your instances.

To track usage

1. If you are a member of multiple projects, select a project from the drop-down list at the top of the Project tab.
2. Select a month and click Submit to query the instance usage for that month.
3. Click Download CSV Summary to download a CSV summary.

## Manage volumes

Volumes are block storage devices that you can attach to instances. They allow for persistent storage as they can be attached to a running instance, or detached and attached to another instance at any time.

In contrast to the instance's root disk, the data of volumes is not destroyed when the instance is deleted.

### **Create or delete a volume**

To create or delete a volume

1. Log in to the OpenStack dashboard.
2. If you are a member of multiple projects, select a Project from the drop-down list at the top of the tab.
3. Click the Volumes category.
4. To create a volume
  1. Click Create Volume.
  2. In the window that opens, enter a name to assign to a volume, a description (optional), and define the size in GBs.
  3. Confirm your changes.
  4. The dashboard shows the volume in the Volumes category.
1. To delete one or multiple volumes
  1. Activate the checkboxes in front of the volumes that you want to delete.
  2. Click Delete Volumes and confirm your choice in the pop-up that appears.
  3. A message indicates whether the action was successful.



After you create one or more volumes, you can attach them to instances.

You can attach a volume to one instance at a time.

View the status of a volume in the Instances & Volumes category of the dashboard: the volume is either available or In-Use.

### **Attach volumes to instances**

To attach volumes to instances

1. Log in to OpenStack dashboard.
2. If you are a member of multiple projects, select a Project from the drop-down list at the top of the tab.
3. Click the Volumes category.
4. Select the volume to add to an instance and click Edit Attachments.
5. In the Manage Volume Attachments window, select an instance.
6. Enter a device name under which the volume should be accessible on the virtual machine.
7. Click Attach Volume to confirm your changes. The dashboard shows the instance to which the volume has been attached and the volume's device name.
8. Now you can log in to the instance, mount the disk, format it, and use it.
9. To detach a volume from an instance
  1. Select the volume and click Edit Attachments.
  2. Click Detach Volume and confirm your changes.
  3. A message indicates whether the action was successful.

## OpenStack command-line clients

### Overview

You can use the OpenStack command-line clients to run simple commands that make API calls and automate tasks by using scripts. Internally, each client command runs cURL commands that embed API requests. The OpenStack APIs are RESTful APIs that use the HTTP protocol, including methods, URIs, media types, and response codes.

These open-source Python clients run on Linux or Mac OS X systems and are easy to learn and use. Each OpenStack service has its own command-line client. On some client commands, you can specify a `debug` parameter to show the underlying API request for the command. This is a good way to become familiar with the OpenStack API calls.

The following command-line clients are available for the respective services' APIs:

`cinder`(`python-cinderclient`)

Client for the Block Storage service API. Use to create and manage volumes.

`glance`(`python-glanceclient`)

Client for the Image Service API. Use to create and manage images.

`keystone`(`python-keystoneclient`)

Client for the Identity Service API. Use to create and manage users, tenants, roles, endpoints, and credentials.

`nova`(`python-novaclient`)

Client for the Compute API and its extensions. Use to create and manage images, instances, and flavors.

`neutron`(`python-neutronclient`)

Client for the Networking API. Use to configure networks for guest servers. This client was previously known as neutron.

swift(python-swiftclient)

Client for the Object Storage API. Use to gather statistics, list items, update metadata, upload, download and delete files stored by the object storage service. Provides access to a swift installation for ad hoc processing.

heat(python-heatclient)

Client for the Orchestration API. Use to launch stacks from templates, view details of running stacks including events and resources, and update and delete stacks.

### Install the OpenStack command-line clients

To install the clients, install the prerequisite software and the Python package for each OpenStack client.

#### Install the clients

Use pip to install the OpenStack clients on a Mac OS X or Linux system. It is easy and ensures that you get the latest version of the client from the <http://pypi.python.org/pypi> Python Package Index. Also, pip lets you update or remove a package. After you install the clients, you must source an openrc file to set required environment variables before you can request OpenStack services through the clients or the APIs.

To install the clients

1. You must install each client separately.
2. Run the following command to install or update a client package:

```
# pip install [--update] python-<project>client
```

Where <project> is the project name and has one of the following values:

- nova. Compute API and extensions.
  - neutron. Networking API.
  - keystone. Identity Service API.
  - glance. Image Service API.
  - swift. Object Storage API.
  - cinder. Block Storage service API.
  - heat. Orchestration API.
3. For example, to install the nova client, run the following command:
- ```
# pip install python-novaclient
```
4. To update the nova client, run the following command:
- ```
# pip install --upgrade python-novaclient
```
5. To remove the nova client, run the following command:
- ```
# pip uninstall python-novaclient
```
6. Before you can issue client commands, you must download and source the openrc file to set environment variables. Proceed to [the section called "OpenStack RC file"](#).

### Get the version for a client

After you install an OpenStack client, you can search for its version number, as follows:

```
$ pip freeze | grep python-
```

```
python-glanceclient==0.4.0python-keystoneclient==0.1.2-e git+https://github.com/openstack/python-  
novaclient.git@077cc0bf22e378c4c4b970f2331a695e440a939f#egg=python_novaclient-devpython-  
neutronclient==0.1.1python-swiftclient==1.1.1
```

You can also use the `yolk -l` command to see which version of the client is installed:

```
$ yolk -l | grep python-novaclient
```

```
python-novaclient - 2.6.10.27 - active development (/Users/your.name/src/cloud-servers/src/src/python-  
novaclient)python-novaclient - 2012.1 - non-active
```

### OpenStack RC file

To set the required environment variables for the OpenStack command-line clients, you must download and source an environment file, `openrc.sh`. It is project-specific and contains the credentials used by OpenStack Compute, Image, and Identity services.

When you source the file and enter the password, environment variables are set for that shell. They allow the commands to communicate to the OpenStack services that run in the cloud.

You can download the file from the OpenStack dashboard as an administrative user or any other user.

To download the OpenStack RC file

1. Log in to the OpenStack dashboard.
2. On the Projecttab, select the project for which you want to download the OpenStack RC file.
3. Click Access & Security. Then, click Download OpenStack RC Fileand save the file.
4. Copy the `openrc.sh` file to the machine from where you want to run OpenStack commands.
5. For example, copy the file to the machine from where you want to upload an image with a glance client command.

6. On any shell from where you want to run OpenStack commands, source the `openrc.sh` file for the respective project.
7. In this example, we source the `demo-openrc.sh` file for the demo project:
8. `$ source demo-openrc.sh`
9. When you are prompted for an OpenStack password, enter the OpenStack password for the user who downloaded the `openrc.sh` file.
10. When you run OpenStack client commands, you can override some environment variable settings by using the options that are listed at the end of the nova helpoutput. For example, you can override the `OS_PASSWORD` setting in the `openrc.sh` file by specifying a password on a nova command, as follows:
11. `$ nova --password <password> image-list`
12. Where password is your password.

### **Manage images**

During setup of OpenStack cloud, the cloud operator sets user permissions to manage images.

Image upload and management might be restricted to only cloud administrators or cloud operators.

After you upload an image, it is considered golden and you cannot change it.

You can upload images through the glance client or the Image Service API. You can also use the nova client to list images, set and delete image metadata, delete images, and take a snapshot of a running instance to create an image.

### **Manage images with the glance client**

To list or get details for images

1. To list the available images:
2. `$ glance image-list`
3. You can use `grep` to filter the list, as follows:
4. `$ glance image-list | grep 'cirros'`
5. To get image details, by name or ID:
6. `$ glance image-show myCirrosImage`

To add an image

- The following example uploads a CentOS 6.3 image in qcow2 format and configures it for public access:
- `$glance image-create --name centos63-image --disk-format=qcow2 --container-format=bare --is-public=True ./centos63.qcow2`

To create an image

1. Write any buffered data to disk.
2. For more information, see the [Taking Snapshots](#) in the OpenStack Operations Guide.
3. To create the image, list instances to get the server ID:
4. `$ nova list`
5. In this example, the server is named `myCirrosServer`. Use this server to create a snapshot, as follows:
6. `$ nova image-create myCirrosServer myCirrosImage`
7. The command creates a qemu snapshot and automatically uploads the image to your repository. Only the tenant that creates the image has access to it.

8. Get details for your image to check its status:

9. `$ nova image-show IMAGE`

10. The image status changes from `SAVING` to `ACTIVE`. Only the tenant who creates the image has access to it.

To launch an instance from your image

- To launch an instance from your image, include the image ID and flavor ID, as follows:

- `$ nova boot newServer --image 7e5142af-1253-4634-bcc6-89482c5f2e8a --flavor 3`

### **Troubleshoot image creation**

- You cannot create a snapshot from an instance that has an attached volume. Detach the volume, create the image, and re-mount the volume.
- Make sure the version of qemu you are using is version 0.14 or greater. Older versions of qemu result in an "unknown option -s" error message in the nova-compute.log.
- Examine the `/var/log/nova-api.log` and `/var/log/nova-compute.log` log files for error messages.

### **Set up access and security for instances**

When you launch a virtual machine, you can inject a key pair, which provides SSH access to your instance. For this to work, the image must contain the cloud-init package. Create at least one key pair for each project. If you generate a keypair with an external tool, you can import it into OpenStack. You can use the key pair for multiple instances that belong to that project. In case an image uses a static root password or a static key set – neither is recommended – you must not provide a key pair when you launch the instance.

A security group is a named collection of network access rules that you use to limit the types of traffic that have access to instances. When you launch an instance, you can assign one or more security groups to it. If you do not create security groups, new instances are automatically assigned to the default security group,



unless you explicitly specify a different security group. The associated rules in each security group control the traffic to instances in the group. Any incoming traffic that is not matched by a rule is denied access by default. You can add rules to or remove rules from a security group. You can modify rules for the default and any other security group.

You must modify the rules for the default security group because users cannot access instances that use the default group from any IP address outside the cloud.

You can modify the rules in a security group to allow access to instances through different ports and protocols. For example, you can modify rules to allow access to instances through SSH, to ping them, or to allow UDP traffic – for example, for a DNS server running on an instance. You specify the following parameters for rules:

- Source of traffic. Enable traffic to instances from either IP addresses inside the cloud from other group members or from all IP addresses.
- Protocol. Choose TCP for SSH, ICMP for pings, or UDP.
- Destination port on virtual machine. Defines a port range. To open a single port only, enter the same value twice. ICMP does not support ports: Enter values to define the codes and types of ICMP traffic to be allowed.

Rules are automatically enforced as soon as you create or modify them.

You can also assign a floating IP address to a running instance to make it accessible from outside the cloud. You assign a floating IP address to an instance and attach a block storage device, or volume, for persistent storage.

### **Add or import keypairs**

To add a key

You can generate a keypair or upload an existing public key.

1. To generate a keypair, run the following command:
2. `$ nova keypair-add KEY_NAME > MY_KEY.pem`
3. The command generates a keypair named KEY\_NAME, writes the private key to the MY\_KEY.pem file, and registers the public key at the Nova database.
4. To set the permissions of the MY\_KEY.pem file, run the following command:
5. `$ chmod 600 MY_KEY.pem`
6. The command changes the permissions of the MY\_KEY.pem file so that only you can read and write to it.

To import a key

1. If you have already generated a keypair with the public key located at `~/.ssh/id_rsa.pub`, run the following command to upload the public key:
2. `$ nova keypair-add --pub_key ~/.ssh/id_rsa.pub KEY_NAME`
3. The command registers the public key at the Nova database and names the keypair KEY\_NAME.
4. List keypairs to make sure that the uploaded keypair appears in the list:
5. `$ nova keypair-list`

### **Configure security groups and rules**

To configure security groups

1. To list all security groups
2. To list security groups for the current project, including descriptions, enter the following command:

3. `$ nova secgroup-list`
4. To create a security group
5. To create a security group with a specified name and description, enter the following command:
6. `$ nova secgroup-create SEC_GROUP_NAME GROUP_DESCRIPTION`
7. To delete a security group
8. To delete a specified group, enter the following command:
9. `$ nova secgroup-delete SEC_GROUP_NAME`

To configure security group rules

Modify security group rules with the `nova secgroup-*-rule` commands.

1. On a shell, source the OpenStack RC file. For details, see [http://docs.openstack.org/user-guide/content/cli\\_openrc.html](http://docs.openstack.org/user-guide/content/cli_openrc.html) the section called "OpenStack RC file".
2. To list the rules for a security group
3. `$ nova secgroup-list-rules SEC_GROUP_NAME`
4. To allow SSH access to the instances
5. Choose one of the following sub-steps:
  1. Add rule for all IPs
  2. Either from all IP addresses (specified as IP subnet in CIDR notation as 0.0.0.0/0):
  3. `$ nova secgroup-add-rule SEC_GROUP_NAME tcp 22 22 0.0.0.0/0`

1. Add rule for security groups
2. Alternatively, you can allow only IP addresses from other security groups (source groups) to access the specified port:
3. 

```
$ nova secgroup-add-group-rule --ip_proto tcp --from_port 22 \ --to_port 22 SEC_GROUP_NAME SOURCE_GROUP_NAME
```
1. To allow ping to the instances
2. Choose one of the following sub-steps:
  1. To allow ping from IPs
  2. Specify all IP addresses as IP subnet in CIDR notation: 0.0.0.0/0. This command allows access to all codes and all types of ICMP traffic, respectively:
  3. 

```
$ nova secgroup-add-rule SEC_GROUP_NAME icmp -1 -1 0.0.0.0/0
```
  4. To allow ping from other security groups
  5. To allow only members of other security groups (source groups) to ping instances:
  6. 

```
$ nova secgroup-add-group-rule --ip_proto icmp --from_port -1 \ --to_port -1 SEC_GROUP_NAME SOURCE_GROUP_NAME
```
  1. To allow access through UDP port
  2. To allow access through a UDP port, such as allowing access to a DNS server that runs on a VM, complete one of the following sub-steps:
    1. To allow UDP access from IPs
    2. Specify all IP addresses as IP subnet in CIDR notation: 0.0.0.0/0.

3. `$ nova secgroup-add-rule SEC_GROUP_NAME udp 53 53 0.0.0.0/0`
4. To allow UDP access
5. To allow only IP addresses from other security groups (source groups) to access the specified port:
6. `$ nova secgroup-add-group-rule --ip_proto udp --from_port 53 \ --to_port 53 SEC_GROUP_NAME SOURCE_GROUP_NAME`
1. To delete a security group rule, specify the same arguments that you used to create the rule.
2. To delete the security rule that you created in [Step 3.a](#):
3. `$ nova secgroup-delete-rule SEC_GROUP_NAME tcp 22 22 0.0.0.0/0`
4. To delete the security rule that you created in [Step 3.b](#):
5. `$ nova secgroup-delete-group-rule --ip_proto tcp --from_port 22 \ --to_port 22 SEC_GROUP_NAME SOURCE_GROUP_NAME`

### Launch instances

Instances are virtual machines that run inside the cloud.

Before you can launch an instance, you must gather parameters such as the image and flavor from which you want to launch your instance.

You can launch an instance directly from one of the available OpenStack images or from an image that you have copied to a persistent volume. The OpenStack Image Service provides a pool of images that are accessible to members of different projects.

### Gather parameters to launch an instance

To launch an instance, you must specify the following parameters:

- The instance source, which is an image or snapshot. Alternatively, you can boot from a volume, which is block storage, to which you've copied an image or snapshot.
- The image or snapshot, which represents the operating system.
- A name for your instance.
- The flavor for your instance, which defines the compute, memory, and storage capacity of nova computing instances. 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, see Section 1.5, "Managing Flavors," (# User Guide for Administrators ).
- User Data is a special key in the metadata service which holds a file that cloud aware applications within the guest instance can access. For example the [cloudinit](#) system is an open source package from Ubuntu that handles early initialization of a cloud instance that makes use of this user data.
- Access and security credentials, which include one or both of the following credentials:
  - A key-pair for your instance, which are SSH credentials that are injected into images when they are launched. For this to work, the image must contain the cloud-init package. Create at least one keypair for each project. If you already have generated a key-pair with an external tool, you can import it into OpenStack. You can use the keypair for multiple instances that belong to that project. For details, refer to Section 1.5.1, Creating or Importing Keys.
  - A security group, which defines which incoming network traffic is forwarded to instances. Security groups hold a set of firewall policies, known as security group rules. For details, see xx.
- If needed, you can assign a floating (public) IP address to a running instance and attach a block storage device, or volume, for persistent storage. For details, see Section 1.5.3, Managing IP Addresses and Section 1.7, Managing Volumes.

After you gather the parameters you need to launch an instance, you can launch it from an [image](#) or a [volume](#).

To gather the parameters to launch an instance

1. On a shell, source the OpenStack RC file.
2. List the available flavors:
3. `$ nova flavor-list`
4. Note the ID of the flavor that you want to use for your instance.
5. List the available images:
6. `$ nova image-list`
7. You can also filter the image list by using `grep` to find a specific image, like this:
8. `$ nova image-list | grep 'kernel'`
9. Note the ID of the image that you want to boot your instance from.
10. List the available security groups:  
`$ nova secgroup-list --all-tenants`
1. If you have not created any security groups, you can assign the instance to only the default security group.
2. You can also list rules for a specified security group:
3. `$ nova secgroup-list-rules default`
4. In this example, the default security group has been modified to allow HTTP traffic on the instance by permitting TCP traffic on Port 80.
5. List the available keypairs.

6. `$ nova keypair-list`
7. Note the name of the keypair that you use for SSH access.

### **Launch an instance from an image**

Use this procedure to launch an instance from an image.

To launch an instance from an image

1. Now you have all parameters required to launch an instance, run the following command and specify the server name, flavor ID, and image ID. Optionally, you can provide a key name for access control and security group for security. You can also include metadata key and value pairs. For example you can add a description for your server by providing the `--meta description="My Server"` parameter.
2. You can pass user data in a file on your local system and pass it at instance launch by using the flag `--user-data <user-data-file>`.
3. `$ nova boot --flavor FLAVOR_ID --image IMAGE_ID --key_name KEY_NAME --user-data mydata.file \ --security_group SEC_GROUP NAME_FOR_INSTANCE --meta KEY=VALUE --meta KEY=VALUE`
4. The command returns a list of server properties, depending on which parameters you provide.
5. A status of BUILD indicates that the instance has started, but is not yet online.
6. A status of ACTIVE indicates that your server is active.
7. Copy the server ID value from the id field in the output. You use this ID to get details for or delete your server.
8. Copy the administrative password value from the adminPass field. You use this value to log into your server.
9. Check if the instance is online:



10\$ nova list

11.This command lists all instances of the project you belong to, including their ID, their name, their status, and their private (and if assigned, their public) IP addresses.

12.If the status for the instance is ACTIVE, the instance is online.

13.To view the available options for the nova listcommand, run the following command:

14\$ nova help list

15.If you did not provide a keypair, security groups, or rules, you can only access the instance from inside the cloud through VNC. Even ping the instance is not possible.

### **Launch an instance from a volume**

After you[create a bootable volume](#), you[launch an instance from the volume](#).

To launch an instance from a volume

1. To create a bootable volume
2. To create a volume from an image, run the following command:
3. # cinder create --image-id 397e713c-b95b-4186-ad46-6126863ea0a9 --display-name my-bootable-vol 8
4. Optionally, to configure your volume, see the [Configuring Image Service and Storage for Compute](#) chapter in the OpenStack Configuration Reference.
5. To list volumes
6. Enter the following command:
7. \$ nova volume-list

8. Copy the value in the ID field for your volume.

1. To launch an instance

2. Enter the nova boot command with the `--block_device_mapping` parameter, as follows:

3. `$ nova boot --flavor <flavor> --block_device_mapping  
    <dev_name>=<id>:<type>:<size>:<delete_on_terminate> <name>`

4. The command arguments are:

5. `--flavor flavor`

6. The flavor ID.

7. `--block_device_mapping dev- name=id:type:size:delete-on-terminate`

- `dev-name`. A device name where the volume is attached in the system at `/dev/dev_name`. This value is typically `vda`.
- `id`. The ID of the volume to boot from, as shown in the output of `nova volume-list`.
- `type`. Either `snap` or any other value, including a blank string. `snap` means that the volume was created from a snapshot.
- `size`. The size of the volume, in GBs. It is safe to leave this blank and have the Compute service infer the size.
- `delete-on-terminate`. A boolean that indicates whether the volume should be deleted when the instance is terminated. You can specify
  - True or 1
  - False or 0

name

1. The name for the server.
2. For example, you might enter the following command to boot from a volume with ID bd7cf584-45de-44e3-bf7f-f7b50bf235e. The volume is not deleted when the instance is terminated:
3. 

```
$ nova boot --flavor 2 --image 397e713c-b95b-4186-ad46-6126863ea0a9 --block_device_mapping vda=bd7cf584-45de-44e3-bf7f-f7b50bf235e:::0 myInstanceFromVolume
```
4. Now when you list volumes, you can see that the volume is attached to a server:
5. 

```
$ nova volume-list
```
6. Additionally, when you list servers, you see the server that you booted from a volume:
7. 

```
$ nova list
```

### **Manage instances and hosts**

Instances are virtual machines that run inside the cloud.

### **Manage IP addresses**

Each instance can have a private, or fixed, IP address and a public, or floating, one.

Private IP addresses are used for communication between instances, and public ones are used for communication with the outside world.

When you launch an instance, it is automatically assigned a private IP address that stays the same until you explicitly terminate the instance. Rebooting an instance has no effect on the private IP address.

A pool of floating IPs, configured by the cloud operator, is available in OpenStack Compute.

You can allocate a certain number of these to a project: The maximum number of floating IP addresses per project is defined by the quota.

You can add a floating IP address from this set to an instance of the project. Floating IP addresses can be dynamically disassociated and associated with other instances of the same project at any time.

Before you can assign a floating IP address to an instance, you first must allocate floating IPs to a project. After floating IP addresses have been allocated to the current project, you can assign them to running instances.

One floating IP address can be assigned to only one instance at a time. Floating IP addresses can be managed with the nova `*floating-ip-*` commands, provided by the `python-novaclient` package.

To list pools with floating IP addresses

- To list all pools that provide floating IP addresses:
- `$ nova floating-ip-pool-list`

To allocate a floating IP address to the current project

- The output of the following command shows the freshly allocated IP address:
- `$ nova floating-ip-pool-list`
- If more than one pool of IP addresses is available, you can also specify the pool from which to allocate the IP address:
- `$ floating-ip-create POOL_NAME`

To list floating IP addresses allocated to the current project

- If an IP is already associated with an instance, the output also shows the IP for the instance, the fixed IP address for the instance, and the name of the pool that provides the floating IP address.

- `$ nova floating-ip-list`

To release a floating IP address from the current project

- The IP address is returned to the pool of IP addresses that are available for all projects. If an IP address is currently assigned to a running instance, it is automatically disassociated from the instance.
- `$ nova floating-ip-delete FLOATING_IP`

To assign a floating IP address to an instance

- To associate an IP address with an instance, one or multiple floating IP addresses must be allocated to the current project. Check this with:
- `$ nova floating-ip-list`
- In addition, you must know the instance's name (or ID). To look up the instances that belong to the current project, use the `nova list` command.
- `$ nova add-floating-ip INSTANCE_NAME_OR_ID FLOATING_IP`
- After you assign the IP with `nova add-floating-ip` and configure security group rules for the instance, the instance is publicly available at the floating IP address.

To remove a floating IP address from an instance

- To remove a floating IP address from an instance, you must specify the same arguments that you used to assign the IP.
- `$ nova remove-floating-ip INSTANCE_NAME_OR_ID FLOATING_IP`

### **Change the size of your server**

You change the size of a server by changing its flavor.

To change the size of your server

1. List the available flavors:
2. `$ nova flavor-list`
3. Show information about your server, including its size:
4. `$ nova show myCirrosServer`
5. The size of the server is m1.small (2).
6. To resize the server, pass the server ID and the desired flavor to the nova resizecommand. Include the `-poll` parameter to report the resize progress.
7. `$ nova resize myCirrosServer 4 -poll`
8. Instance resizing... 100% completeFinished
9. Show the status for your server:
10. `$ nova list`
11. When the resize completes, the status becomes VERIFY\_RESIZE. To confirm the resize:
12. `$ nova resize-confirm 6beefcf7-9de6-48b3-9ba9-e11b343189b3`
13. The server status becomes ACTIVE.
14. If the resize fails or does not work as expected, you can revert the resize:
15. `$ nova resize-revert 6beefcf7-9de6-48b3-9ba9-e11b343189b3`
16. The server status becomes ACTIVE.

**Stop and start an instance**

Use one of the following methods to stop and start an instance.

**Pause and un-pause an instance**

To pause and un-pause a server

- To pause a server, run the following command:
- `$ nova pause SERVER`
- This command stores the state of the VM in RAM. A paused instance continues to run in a frozen state.
- To un-pause the server, run the following command:
- `$ nova unpause SERVER`

**Suspend and resume an instance**

To suspend and resume a server

Administrative users might want to suspend an infrequently used instance or to perform system maintenance.

1. When you suspend an instance, its VM state is stored on disk, all memory is written to disk, and the virtual machine is stopped. Suspending an instance is similar to placing a device in hibernation; memory and vCPUs become available.
2. To initiate a hypervisor-level suspend operation, run the following command:
3. `$ nova suspend SERVER`
4. To resume a suspended server:
5. `$ nova resume SERVER`

## Reboot an instance

You can perform a soft or hard reboot of a running instance. A soft reboot attempts a graceful shutdown and restart of the instance. A hard reboot power cycles the instance.

To reboot a server

- By default, when you reboot a server, it is a soft reboot.
- `$ nova reboot SERVER`

To perform a hard reboot, pass the `–hard` parameter, as follows:

```
$ nova reboot –hard SERVER
```

## Evacuate instances

If a cloud compute node fails due to a hardware malfunction or another reason, you can evacuate instances to make them available again.

You can choose evacuation parameters for your use case.

To preserve user data on server disk, you must configure shared storage on the target host. Also, you must validate that the current VM host is down. Otherwise the evacuation fails with an error.

To evacuate your server

1. To find a different host for the evacuated instance, run the following command to lists hosts:
2. `$ nova host-list`
3. You can pass the instance password to the command by using the `–password <pwd>` option. If you do not specify a password, one is generated and printed after the command finishes successfully. The following command evacuates a server without shared storage:



4. `$ nova evacuate evacuated_server_name host_b`
5. The command evacuates an instance from a down host to a specified host. The instance is booted from a new disk, but preserves its configuration including its ID, name, uid, IP address, and so on. The command returns a password:
6. To preserve the user disk data on the evacuated server, deploy OpenStack Compute with shared filesystem.
7. `$ nova evacuate evacuated_server_name host_b --on-shared-storage`

### **Delete an instance**

When you no longer need an instance, you can delete it.

To delete an instance

1. List all instances:
2. `$ nova list`
3. Use the following command to delete the newServer instance, which is in ERROR state:
4. `$ nova delete newServer`
5. The command does not notify that your server was deleted.
6. Instead, run the nova list command:
7. `$ nova list`
8. The deleted instance does not appear in the list.

### **Get a console to an instance**

To get a console to an instance

To get a VNC console to an instance, run the following command:

```
$ nova get-vnc-console myCirrosServer xvpvnc
```

The command returns a URL from which you can access your instance:

### **Manage bare metal nodes**

If you use the bare metal driver, you must create a bare metal node and add a network interface to it. You then launch an instance from a bare metal image. You can list and delete bare metal nodes. When you delete a node, any associated network interfaces are removed. You can list and remove network interfaces that are associated with a bare metal node.

#### Commands

- `baremetal-interface-add`
- Adds a network interface to a bare metal node.
- `baremetal-interface-list`
- Lists network interfaces associated with a bare metal node.
- `baremetal-interface-remove`
- Removes a network interface from a bare metal node.
- `baremetal-node-create`
- Creates a bare metal node.
- `baremetal-node-delete`

- Removes a bare metal node and any associated interfaces.
- `baremetal-node-list`
- Lists available bare metal nodes.
- `baremetal-node-show`
- Shows information about a bare metal node.

To manage bare metal nodes

1. Create a bare metal node.
2. `$ nova baremetal-node-create --pm_address=1.2.3.4 --pm_user=ipmi --pm_password=ipmi $(hostname -f) 1 512 10 aa:bb:cc:dd:ee:ff`
3. Add network interface information to the node:
4. `$ nova baremetal-interface-add 1 aa:bb:cc:dd:ee:ff`
5. Launch an instance from a bare metal image:
6. `$ nova boot --image my-baremetal-image --flavor my-baremetal-flavor test`
7. | ... wait for instance to become active ...
8. You can list bare metal nodes and interfaces. When a node is in use, its status includes the UUID of the instance that runs on it:
9. `$ nova baremetal-node-list`
10. Show details about a bare metal node:

11\$ nova baremetal-node-show 1

### **Show usage statistics for hosts and instances**

You can show basic statistics on resource usage for hosts and instances.

To show host usage statistics

1. List the hosts and the nova-related services that run on them:
2. \$ nova host-list
3. Get a summary of resource usage of all of the instances running on the host.
4. \$ nova host-describe devstack-grizzly
5. The cpu column shows the sum of the virtual CPUs for instances running on the host.
6. The memory\_mb column shows the sum of the memory (in MB) allocated to the instances that run on the hosts.
7. The disk\_gb column shows the sum of the root and ephemeral disk sizes (in GB) of the instances that run on the hosts.

To show instance usage statistics

1. Get CPU, memory, I/O, and network statistics for an instance.
2. First, list instances:
3. \$ nova list
4. Then, get diagnostic statistics:
5. \$ nova diagnostics myCirrosServer

6. Get summary statistics for each tenant:

7. `$ nova usage-list`

8. Usage from 2013-06-25 to 2013-07-24:

### **Create and manage networks**

Before you run commands, set the following environment variables:

```
export OS_USERNAME=admin export OS_PASSWORD=password export OS_TENANT_NAME=admin export OS_AUTH_URL=http://localhost:5000/v2.0
```

To create and manage networks

1. List the extensions of the system:

2. `$ neutron ext-list -c alias -c name`

3. Create a network:

4. `$ neutron net-create net1`

5. Created a new network:

6. Create a network with specified provider network type:

7. `$ neutron net-create net2 --provider:network-type local`

8. Created a new network:

9. Just as shown previous, the unknown option `--provider:network-type` is used to create a local provider network.

10. Create a subnet:

11\$ neutron subnet-create net1 192.168.2.0/24 --name subnet1

12.Created a new subnet:

13.In the previous command, net1 is the network name, 192.168.2.0/24 is the subnet's CIDR. They are positional arguments. --name subnet1 is an unknown option, which specifies the subnet's name.

14.Create a port with specified IP address:

15\$ neutron port-create net1 --fixed-ip ip\_address=192.168.2.40

16.Created a new port:

17.In the previous command, net1 is the network name, which is a positional argument. --fixed-ip ip\_address=192.168.2.40 is an option, which specifies the port's fixed IP address we wanted.

18.Create a port without specified IP address:

19\$ neutron port-create net1

20.Created a new port:

21.We can see that the system will allocate one IP address if we don't specify the IP address in command line.

22.Query ports with specified fixed IP addresses:

23\$ neutron port-list --fixed-ips ip\_address=192.168.2.2 ip\_address=192.168.2.40

24.--fixed-ips ip\_address=192.168.2.2 ip\_address=192.168.2.40 is one unknown option.

25.How to find unknown options?The unknown options can be easily found by watching the output of create\_xxx or show\_xxx command. For example, in the port creation command, we see the fixed\_ips fields, which can be used as an unknown option.

## Create and manage stacks

To create a stack from an example template file

1. To create a stack, or template, from an [example template file](#), run following command:
2. `$ heat stack-create mystack --template-file=/path/to/heat/templates/WordPress_Single_Instance.template--parameters="InstanceType=m1.large;DBUsername=wp;DBPassword=verybadpassword;KeyName=heat_key;LinuxDistrib`
3. The `--parameters` values that you specify depend on which parameters are defined in the template. If the template file is hosted on a website, you can specify the URL with `--template-url` parameter instead of the `--template-file` parameter.
4. The command returns the following output:
5. You can also use the `stack-create` command to validate a template file without creating a stack from it.
6. To do so, run the following command:
7. `$ heat stack-create mystack --template-file=/path/to/heat/templates/WordPress_Single_Instance.template`
8. If validation fails, the response returns an error message.

To list stacks

- To see which stacks are visible to the current user, run the following command:
- `$ heat stack-list`

To view stack details

To explore the state and history of a particular stack, you can run a number of commands.

1. To show the details of a stack, run the following command:
2. `$ heat stack-show mystack`
3. A stack consists of a collection of resources. To list the resources, including their status, in a stack, run the following command:
4. `$ heat resource-list mystack`
5. To show the details for the specified resource in a stack, run the following command:
6. `$ heat resource-show mystack WikiDatabase`
7. Some resources have associated metadata which can change throughout the life-cycle of a resource:
8. `$ heat resource-metadata mystack WikiDatabase`
9. A series of events is generated during the life-cycle of a stack. This command will display those events.
10. `$ heat event-list mystack`
11. To show the details for a particular event, run the following command:
12. `$ heat event-show WikiDatabase 1`

To update a stack

- To update an existing stack from a modified template file, run a command like the following command:
- `$ heat stack-update mystack --template-file=/path/to/heat/templates/WordPress_Single_Instance_v2.template --parameters="InstanceType=m1.large;DBUsername=wp;DBPassword=verybadpassword;KeyName=heat_key;LinuxDistribution=F"`



- Some resources are updated in-place, while others are replaced with new resources.

## Review Associate Keystone Architecture

The Identity service performs these functions:

- User management. Tracks users and their permissions.
- Service catalog. Provides a catalog of available services with their API endpoints.

To understand the Identity Service, you must understand these concepts:

### User

Digital representation of a person, system, or service who uses OpenStack cloud services. Identity authentication services will validate that incoming request are being made by the user who claims to be making the call. Users have a login and may be assigned tokens to access resources. Users may be directly assigned to a particular tenant and behave as if they are contained in that tenant.

### Credentials

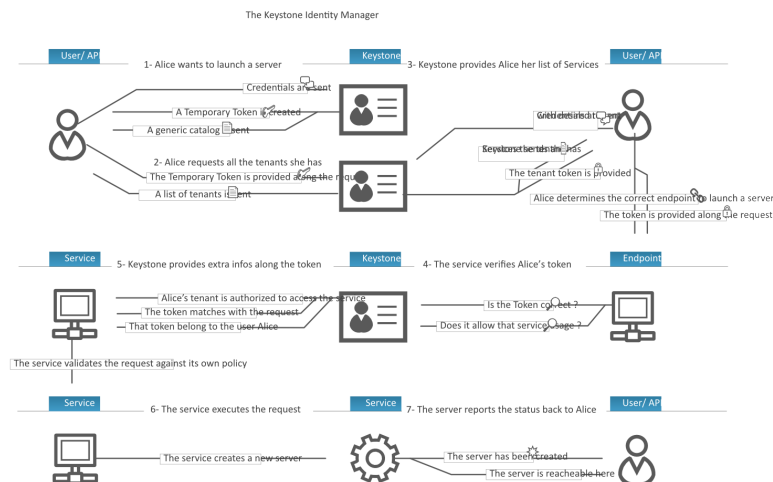
Data that is known only by a user that proves who they are. In the Identity Service, examples are:

- Username and password
- Username and API key
- An authentication token provided by the Identity Service

### Authentication

The act of confirming the identity of a user. The Identity Service confirms an incoming request by validating a set of credentials supplied by the user. These credentials are initially a username and password or a username and API key. In response to these credentials, the Identity Service issues

|                 |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                 | <p>the user an authentication token, which the user provides in subsequent requests.</p>                                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <b>Token</b>    | <p>An arbitrary bit of text that is used to access resources. Each token has a scope which describes which resources are accessible with it. A token may be revoked at anytime and is valid for a finite duration.</p> <p>While the Identity Service supports token-based authentication in this release, the intention is for it to support additional protocols in the future. The intent is for it to be an integration service foremost, and not aspire to be a full-fledged identity store and management solution.</p> |
| <b>Tenant</b>   | <p>A container used to group or isolate resources and/or identity objects. Depending on the service operator, a tenant may map to a customer, account, organization, or project.</p>                                                                                                                                                                                                                                                                                                                                         |
| <b>Service</b>  | <p>An OpenStack service, such as Compute (Nova), Object Storage (Swift), or Image Service (Glance). Provides one or more endpoints through which users can access resources and perform operations.</p>                                                                                                                                                                                                                                                                                                                      |
| <b>Endpoint</b> | <p>An network-accessible address, usually described by URL, from where you access a service. If using an extension for templates, you can create an endpoint template, which represents the templates of all the consumable services that are available across the regions.</p>                                                                                                                                                                                                                                              |
| <b>Role</b>     | <p>A personality that a user assumes that enables them to perform a specific set of operations. A role includes a set of rights and privileges. A user assuming that role inherits those rights and privileges.</p> <p>In the Identity Service, a token that is issued to a user includes the list of roles that user can assume. Services that are being called by that user determine how they interpret the set of roles a user has and which operations or resources each role grants access to.</p>                     |

**Figure 4.7. Keystone Authentication**

## User management

The main components of Identity user management are:

- Users
- Tenants
- Roles

A user represents a human user, and has associated information such as username, password and email. This example creates a user named "alice":

```
$ keystone user-create --name=alice --pass=mypassword123 --email=alice@example.com
```

A tenant can be a project, group, or organization. Whenever you make requests to OpenStack services, you must specify a tenant. For example, if you query the Compute service for a list of running instances, you get a list of all running instances for the specified tenant. This example creates a tenant named "acme":

```
$ keystone tenant-create --name=acme
```

A role captures what operations a user is permitted to perform in a given tenant. This example creates a role named "compute-user":

```
$ keystone role-create --name=compute-user
```

The Identity service associates a user with a tenant and a role. To continue with our previous examples, we may wish to assign the "alice" user the "compute-user" role in the "acme" tenant:

```
$ keystone user-list
```

```
$ keystone user-role-add --user=892585 --role=9a764e --tenant-id=6b8fd2
```

A user can be assigned different roles in different tenants. For example, Alice may also have the "admin" role in the "Cyberdyne" tenant. A user can also be assigned multiple roles in the same tenant.

The `/etc/[SERVICE_CODENAME]/policy.json` file controls what users are allowed to do for a given service. For example, `/etc/nova/policy.json` specifies the access policy for the Compute service, `/etc/glance/policy.json` specifies the access policy for the Image Service, and `/etc/keystone/policy.json` specifies the access policy for the Identity service.

The default `policy.json` files in the Compute, Identity, and Image Service recognize only the admin role: all operations that do not require the admin role will be accessible by any user that has any role in a tenant.

If you wish to restrict users from performing operations in, say, the Compute service, you need to create a role in the Identity service and then modify `/etc/nova/policy.json` so that this role is required for Compute operations.

For example, this line in `/etc/nova/policy.json` specifies that there are no restrictions on which users can create volumes: if the user has any role in a tenant, they will be able to create volumes in that tenant.

## Service Management

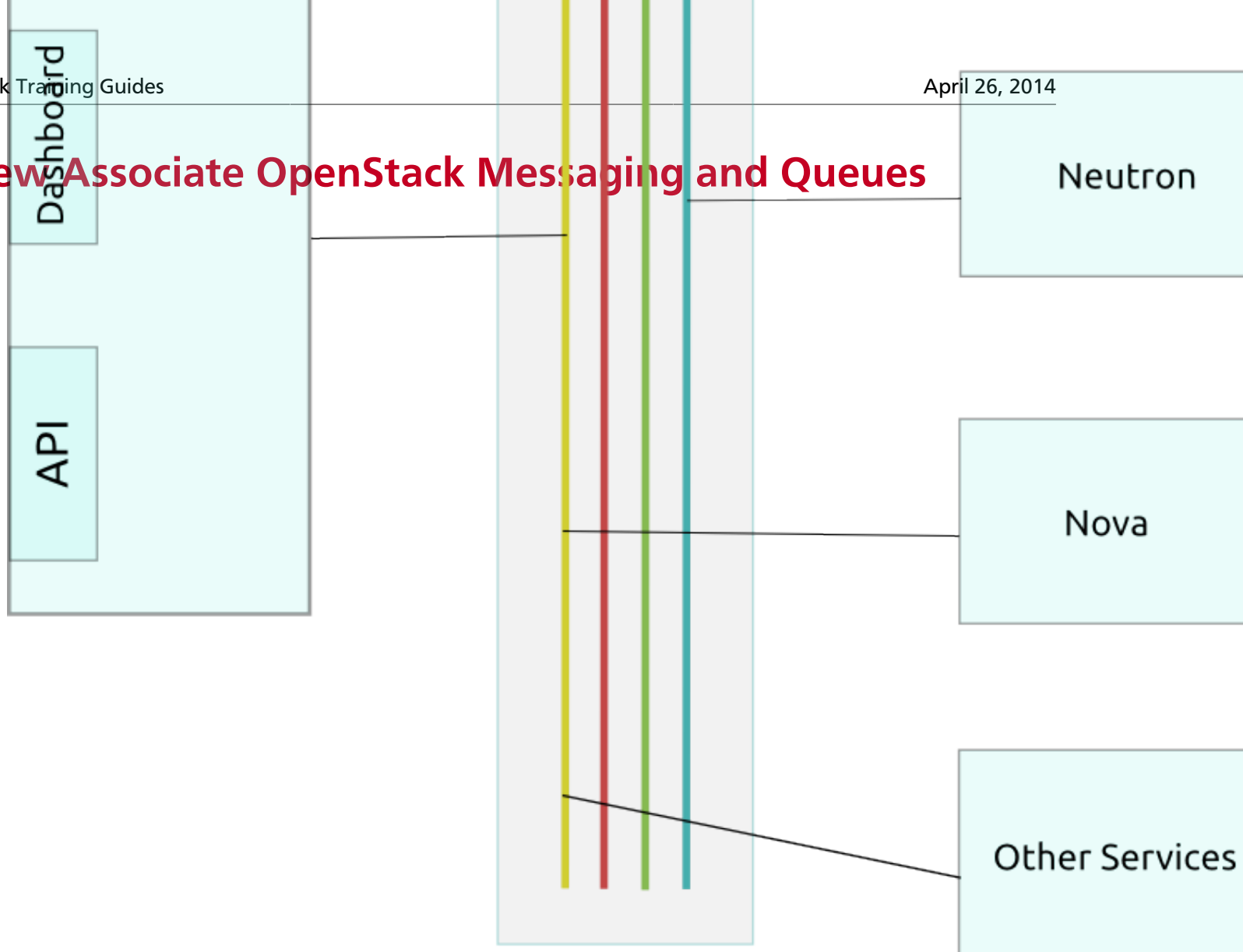
The Identity Service provides the following service management functions:

- Services
- Endpoints

The Identity Service also maintains a user that corresponds to each service, such as a user named `nova`, for the Compute service) and a special service tenant, which is called `service`.

The commands for creating services and endpoints are described in a later section.

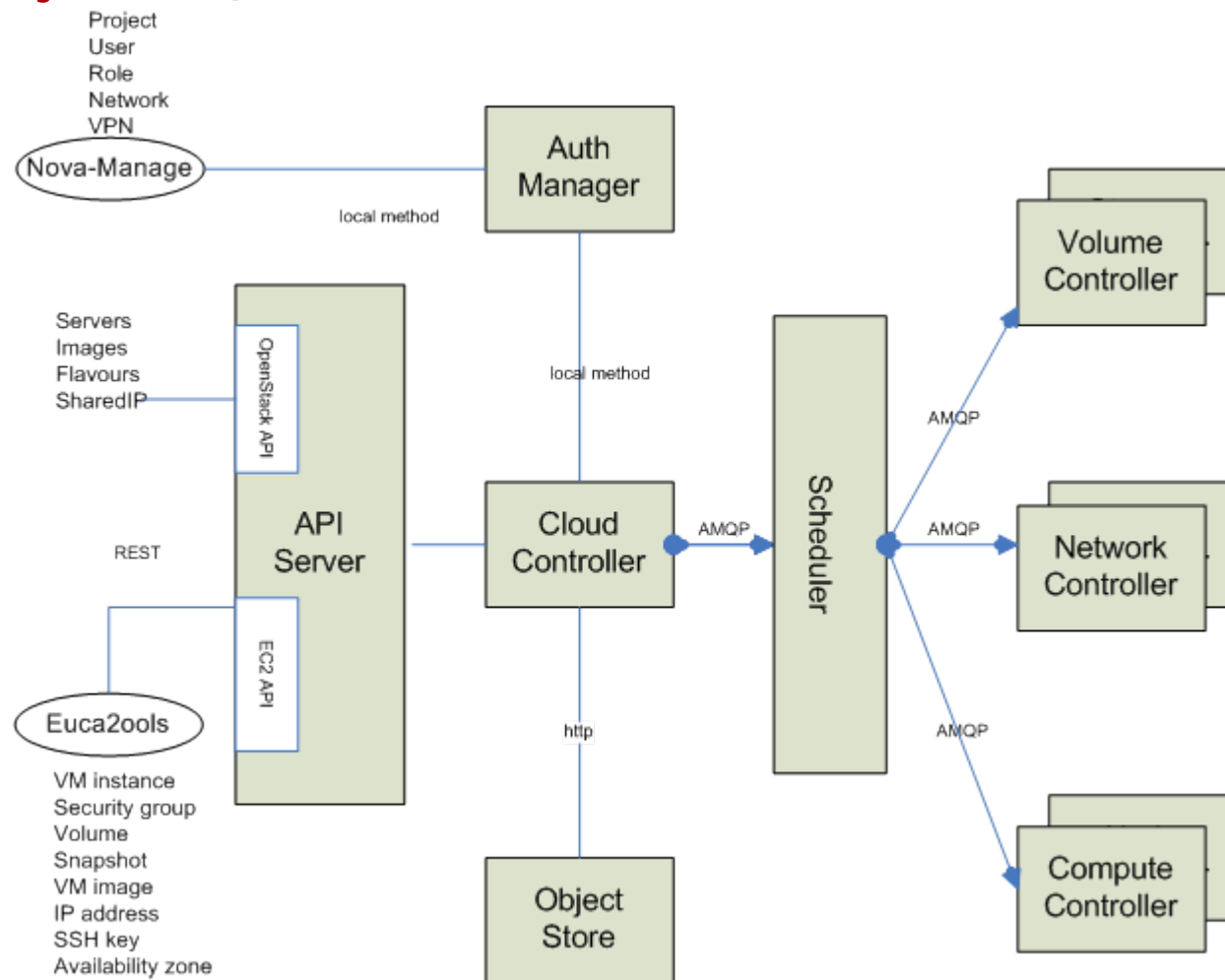
# Review Associate OpenStack Messaging and Queues



AMQP is the messaging technology chosen by the OpenStack cloud. The AMQP broker, either RabbitMQ or Qpid, sits between any two Nova components and allows them to communicate in a loosely coupled fashion. More precisely, Nova components (the compute fabric of OpenStack) use Remote Procedure Calls (RPC hereinafter) to communicate to one another; however such a paradigm is built atop the publish/subscribe paradigm so that the following benefits can be achieved:

- Decoupling between client and servant (such as the client does not need to know where the servant reference is).
- Full a-synchronism between client and servant (such as the client does not need the servant to run at the same time of the remote call).
- Random balancing of remote calls (such as if more servants are up and running, one-way calls are transparently dispatched to the first available servant).

Nova uses direct, fanout, and topic-based exchanges. The architecture looks like the one depicted in the figure below:

**Figure 4.9. AMQP**



Nova implements RPC (both request+response, and one-way, respectively nicknamed 'rpc.call' and 'rpc.cast') over AMQP by providing an adapter class which take cares of marshaling and un-marshaling of messages into function calls. Each Nova service, such as Compute, Scheduler, and so on, creates two queues at the initialization time, one which accepts messages with routing keys 'NODE-TYPE.NODE-ID', for example, compute.hostname, and another, which accepts messages with routing keys as generic 'NODE-TYPE', for example compute. The former is used specifically when Nova-API needs to redirect commands to a specific node like 'euca-terminate instance'. In this case, only the compute node whose host's hypervisor is running the virtual machine can kill the instance. The API acts as a consumer when RPC calls are request/response, otherwise is acts as publisher only.

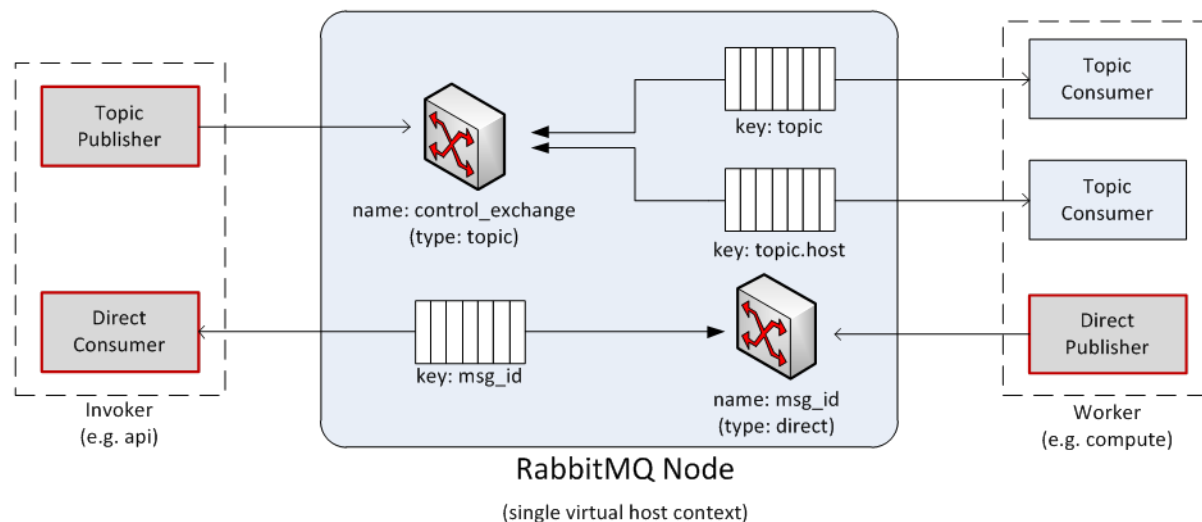
### Nova RPC Mappings

The figure below shows the internals of a message broker node (referred to as a RabbitMQ node in the diagrams) when a single instance is deployed and shared in an OpenStack cloud. Every component within Nova connects to the message broker and, depending on its personality, such as a compute node or a network node, may use the queue either as an Invoker (such as API or Scheduler) or a Worker (such as Compute or Network). Invokers and Workers do not actually exist in the Nova object model, but in this example they are used as an abstraction for the sake of clarity. An Invoker is a component that sends messages in the queuing system using **rpc.call** and **rpc.cast**. A worker is a component that receives messages from the queuing system and replies accordingly to rcp.call operations.

Figure 2 shows the following internal elements:

- **Topic Publisher:** A Topic Publisher comes to life when an rpc.call or an rpc.cast operation is executed; this object is instantiated and used to push a message to the queuing system. Every publisher connects always to the same topic-based exchange; its life-cycle is limited to the message delivery.
- **Direct Consumer:** A Direct Consumer comes to life if (an only if) a rpc.call operation is executed; this object is instantiated and used to receive a response message from the queuing system; Every consumer connects to a unique direct-based exchange via a unique exclusive queue; its life-cycle is limited to the message delivery; the exchange and queue identifiers are determined by a UUID generator, and are marshaled in the message sent by the Topic Publisher (only rpc.call operations).

- **Topic Consumer:** A Topic Consumer comes to life as soon as a Worker is instantiated and exists throughout its life-cycle; this object is used to receive messages from the queue and it invokes the appropriate action as defined by the Worker role. A Topic Consumer connects to the same topic-based exchange either via a shared queue or via a unique exclusive queue. Every Worker has two topic consumers, one that is addressed only during `rpc.cast` operations (and it connects to a shared queue whose exchange key is 'topic') and the other that is addressed only during `rpc.call` operations (and it connects to a unique queue whose exchange key is 'topic.host').
- **Direct Publisher:** A Direct Publisher comes to life only during `rpc.call` operations and it is instantiated to return the message required by the request/response operation. The object connects to a direct-based exchange whose identity is dictated by the incoming message.
- **Topic Exchange:** The Exchange is a routing table that exists in the context of a virtual host (the multi-tenancy mechanism provided by Qpid or RabbitMQ); its type (such as topic vs. direct) determines the routing policy; a message broker node will have only one topic-based exchange for every topic in Nova.
- **Direct Exchange:** This is a routing table that is created during `rpc.call` operations; there are many instances of this kind of exchange throughout the life-cycle of a message broker node, one for each `rpc.call` invoked.
- **Queue Element:** A Queue is a message bucket. Messages are kept in the queue until a Consumer (either Topic or Direct Consumer) connects to the queue and fetch it. Queues can be shared or can be exclusive. Queues whose routing key is 'topic' are shared amongst Workers of the same personality.

**Figure 4.10. RabbitMQ**

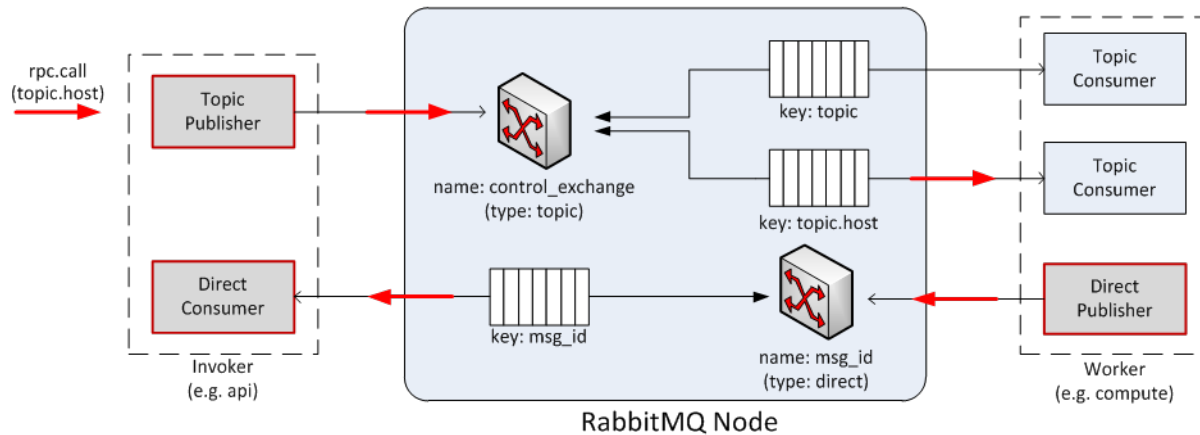
### RPC Calls

The diagram below shows the message flow during an rp.call operation:

1. A Topic Publisher is instantiated to send the message request to the queuing system; immediately before the publishing operation. A Direct Consumer is instantiated to wait for the response message.
2. Once the message is dispatched by the exchange, it is fetched by the Topic Consumer dictated by the routing key (such as 'topic.host') and passed to the Worker in charge of the task.
3. Once the task is completed, a Direct Publisher is allocated to send the response message to the queuing system.

4. Once the message is dispatched by the exchange, it is fetched by the Direct Consumer dictated by the routing key (such as 'msg\_id') and passed to the Invoker.

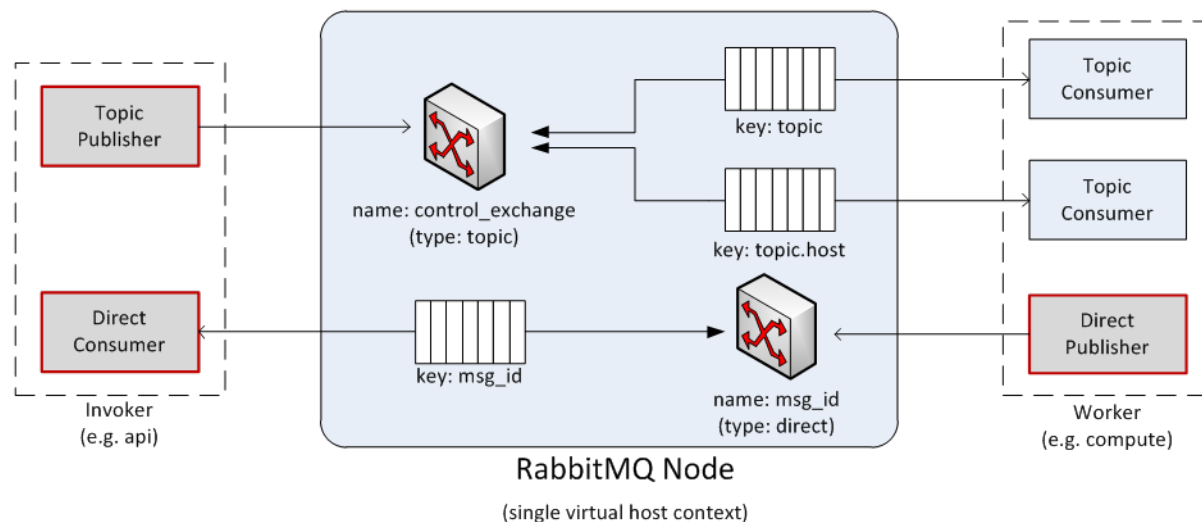
**Figure 4.11. RabbitMQ**



## RPC Casts

The diagram below the message flow during an `rp.cast` operation:

1. A Topic Publisher is instantiated to send the message request to the queuing system.
2. Once the message is dispatched by the exchange, it is fetched by the Topic Consumer dictated by the routing key (such as 'topic') and passed to the Worker in charge of the task.

**Figure 4.12. RabbitMQ**

### AMQP Broker Load

At any given time the load of a message broker node running either Qpid or RabbitMQ is a function of the following parameters:

- **Throughput of API calls:** the number of API calls (more precisely rpc.call ops) being served by the OpenStack cloud dictates the number of direct-based exchanges, related queues and direct consumers connected to them.
- **Number of Workers:** there is one queue shared amongst workers with the same personality; however there are as many exclusive queues as the number of workers; the number of workers dictates also the number of routing keys within the topic-based exchange, which is shared amongst all workers.

The figure below shows the status of a RabbitMQ node after Nova components' bootstrap in a test environment. Exchanges and queues being created by Nova components are:

- Exchanges
  1. nova (topic exchange)
- Queues
  1. compute.phantom (phantom is the hostname)
  2. compute
  3. network.phantom (phantom is the hostname)
  4. network
  5. scheduler.phantom (phantom is the hostname)
  6. scheduler

### **RabbitMQ Gotchas**

Nova uses Kombu to connect to the RabbitMQ environment. Kombu is a Python library that in turn uses AMQPLib, a library that implements the standard AMQP 0.8 at the time of writing. When using Kombu, Invokers and Workers need the following parameters in order to instantiate a Connection object that connects to the RabbitMQ server (please note that most of the following material can be also found in the Kombu documentation; it has been summarized and revised here for the sake of clarity):

- **Hostname:** The hostname to the AMQP server.
- **UserId:** A valid username used to authenticate to the server.
- **Password:** The password used to authenticate to the server.

- **Virtual\_host:** The name of the virtual host to work with. This virtual host must exist on the server, and the user must have access to it. Default is `"/"`.
- **Port:** The port of the AMQP server. Default is 5672 (amqp).

The following parameters are default:

- **Insist:** Insist on connecting to a server. In a configuration with multiple load-sharing servers, the Insist option tells the server that the client is insisting on a connection to the specified server. Default is False.
- **Connect\_timeout:** The timeout in seconds before the client gives up connecting to the server. The default is no timeout.
- **SSL:** Use SSL to connect to the server. The default is False.

More precisely consumers need the following parameters:

- **Connection:** The above mentioned Connection object.
- **Queue:** Name of the queue.
- **Exchange:** Name of the exchange the queue binds to.
- **Routing\_key:** The interpretation of the routing key depends on the value of the `exchange_type` attribute.
- **Direct exchange:** If the routing key property of the message and the `routing_key` attribute of the queue are identical, then the message is forwarded to the queue.
- **Fanout exchange:** Messages are forwarded to the queues bound the exchange, even if the binding does not have a key.
- **Topic exchange:** If the routing key property of the message matches the routing key of the key according to a primitive pattern matching scheme, then the message is forwarded to the queue. The message routing

key then consists of words separated by dots (".", like domain names), and two special characters are available; star ("") and hash ("#"). The star matches any word, and the hash matches zero or more words. For example ".stock.#" matches the routing keys "usd.stock" and "eur.stock.db" but not "stock.nasdaq".

- **Durable:** This flag determines the durability of both exchanges and queues; durable exchanges and queues remain active when a RabbitMQ server restarts. Non-durable exchanges/queues (transient exchanges/queues) are purged when a server restarts. It is worth noting that AMQP specifies that durable queues cannot bind to transient exchanges. Default is True.
- **Auto\_delete:** If set, the exchange is deleted when all queues have finished using it. Default is False.
- **Exclusive:** Exclusive queues (such as non-shared) may only be consumed from by the current connection. When exclusive is on, this also implies auto\_delete. Default is False.
- **Exchange\_type:** AMQP defines several default exchange types (routing algorithms) that covers most of the common messaging use cases.
- **Auto\_ack:** Acknowledgement is handled automatically once messages are received. By default auto\_ack is set to False, and the receiver is required to manually handle acknowledgment.
- **No\_ack:** It disables acknowledgement on the server-side. This is different from auto\_ack in that acknowledgement is turned off altogether. This functionality increases performance but at the cost of reliability. Messages can get lost if a client dies before it can deliver them to the application.
- **Auto\_declare:** If this is True and the exchange name is set, the exchange will be automatically declared at instantiation. Auto declare is on by default. Publishers specify most the parameters of consumers (they do not specify a queue name), but they can also specify the following:
- **Delivery\_mode:** The default delivery mode used for messages. The value is an integer. The following delivery modes are supported by RabbitMQ:
- **1 or "transient":** The message is transient. Which means it is stored in memory only, and is lost if the server dies or restarts.



- **2 or “persistent”**: The message is persistent. Which means the message is stored both in-memory, and on disk, and therefore preserved if the server dies or restarts.

The default value is 2 (persistent). During a send operation, Publishers can override the delivery mode of messages so that, for example, transient messages can be sent over a durable queue.

## Review Associate Administration Tasks



## 5. Controller Node Lab

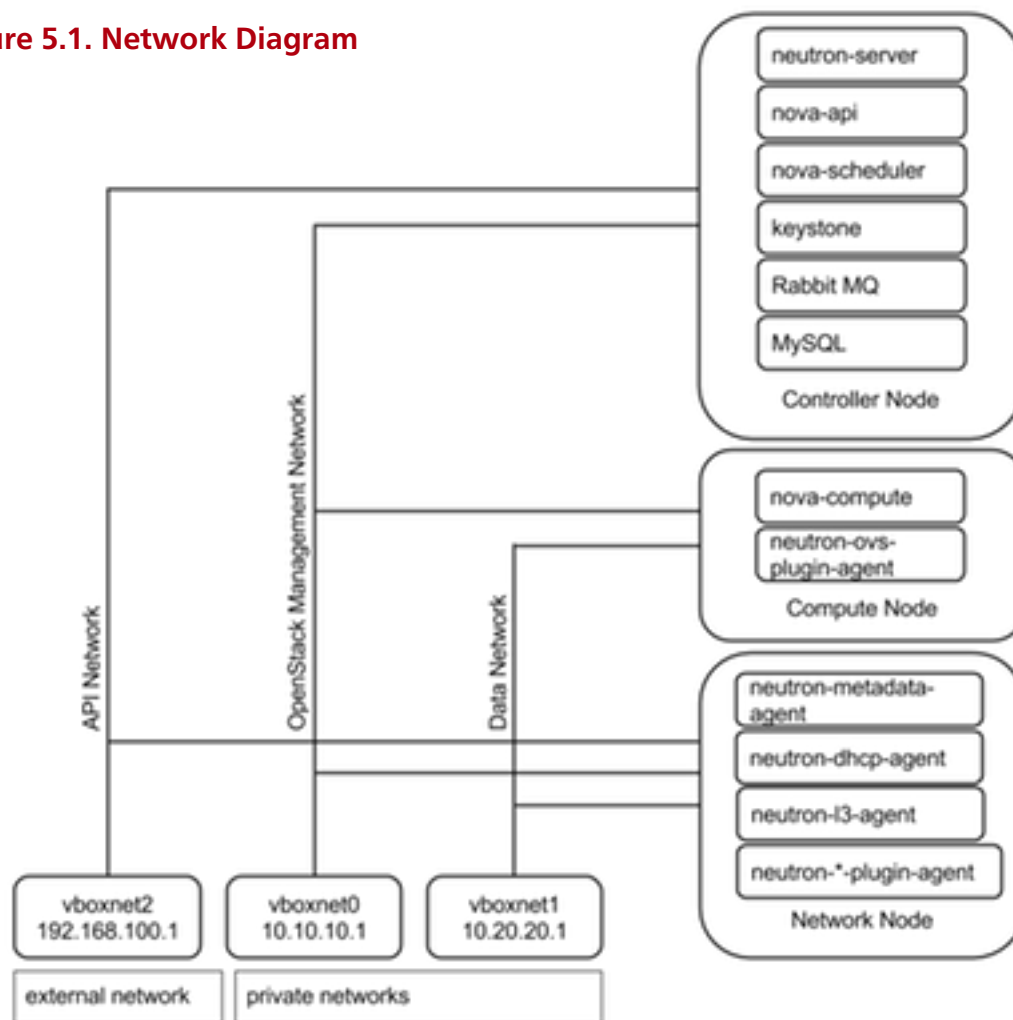
### Table of Contents

|                                                                   |     |
|-------------------------------------------------------------------|-----|
| Days 2 to 4, 13:30 to 14:45, 15:00 to 16:30, 16:45 to 18:15 ..... | 123 |
| Control Node Lab .....                                            | 123 |

**Days 2 to 4, 13:30 to 14:45, 15:00 to 16:30, 16:45 to 18:15**

### Control Node Lab

**Network Diagram :**

**Figure 5.1. Network Diagram**

Publicly editable image source at [https://docs.google.com/drawings/d/1GX3FXmkz3c\\_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing](https://docs.google.com/drawings/d/1GX3FXmkz3c_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing)

**Vboxnet0, Vboxnet1, Vboxnet2** - are virtual networks setup up by virtual box with your host machine. This is the way your host can communicate with the virtual machines. These networks are in turn used by virtual box VM's for OpenStack networks, so that OpenStack's services can communicate with each other.

### Controller Node

Start your Controller Node the one you setup in previous section.

#### Preparing Ubuntu 13.04/12.04

- After you install Ubuntu Server, go in sudo mode

```
$ sudo su
```

- Add Havana repositories:

```
# apt-get install ubuntu-cloud-keyring python-software-properties software-properties-common  
python-keyring
```

```
# echo deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/icehouse main  
>> /etc/apt/sources.list.d/icehouse.list
```

- Update your system:

```
# apt-get update  
# apt-get upgrade  
# apt-get dist-upgrade
```

### Networking :

Configure your network by editing /etc/network/interfaces file

- Open /etc/network/interfaces and edit file as mentioned:

```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).
# This file is configured for OpenStack Control Node by dguitarbite.
# Note: Selection of the IP addresses is important, changing them may break some of
# OpenStack Related services,
# As these IP addresses are essential for communication between them.

# The loopback network interface - for Host-Onlyroot
auto lo
iface lo inet loopback

# Virtual Box vboxnet0 - OpenStack Management Network
# (Virtual Box Network Adapter 1)
auto eth0
iface eth0 inet static
address 10.10.10.51
netmask 255.255.255.0
gateway 10.10.10.1

# Virtual Box vboxnet2 - for exposing OpenStack API over external network
# (Virtual Box Network Adapter 2)
auto eth1
iface eth1 inet static
address 192.168.100.51
netmask 255.255.255.0
gateway 192.168.100.1

# The primary network interface - Virtual Box NAT connection
# (Virtual Box Network Adapter 3)
auto eth2
iface eth2 inet dhcp
```

- After saving the interfaces file, restart the networking service

```
# service networking restart
```

```
# ifconfig
```

- You should see the expected network interface cards having the required IP Addresses.

### SSH from HOST

- Create an SSH key pair for your Control Node. Follow the same steps as you did in the starting section of the article for your host machine.
- To SSH into the Control Node from the Host Machine type the below command.

```
$ ssh control@10.10.10.51
```

```
$ sudo su
```

- Now you can have access to your host clipboard.

### My SQL

- Install MySQL:

```
# apt-get install -y mysql-server python-mysqldb
```

- Configure mysql to accept all incoming requests:

```
# sed -i 's/127.0.0.1/0.0.0.0/g' /etc/mysql/my.cnf
```

```
# service mysql restart
```

### RabbitMQ

- Install RabbitMQ:

```
# apt-get install -y rabbitmq-server
```

- Install NTP service:

```
# apt-get install -y ntp
```

- Create these databases:

```
$ mysql -u root -p
```

```
mysql> CREATE DATABASE keystone;
```

```
mysql> GRANT ALL ON keystone.* TO 'keystoneUser'@'%' IDENTIFIED BY 'keystonePass';
```

```
mysql> CREATE DATABASE glance;
```

```
mysql> GRANT ALL ON glance.* TO 'glanceUser'@'%' IDENTIFIED BY 'glancePass';
```

```
mysql> CREATE DATABASE neutron;
```

```
mysql> GRANT ALL ON neutron.* TO 'neutronUser'@'%' IDENTIFIED BY 'neutronPass';
```

```
mysql> CREATE DATABASE nova;
```

```
mysql> GRANT ALL ON nova.* TO 'novaUser'@'%' IDENTIFIED BY 'novaPass';
```

```
mysql> CREATE DATABASE cinder;
```

```
mysql> GRANT ALL ON cinder.* TO 'cinderUser'@'%' IDENTIFIED BY 'cinderPass';
```

```
mysql> quit;
```

## Other

- Install other services:

```
# apt-get install -y vlan bridge-utils
```

- Enable IP\_Forwarding:

```
# sed -i 's/#net.ipv4.ip_forward=1/net.ipv4.ip_forward=1/' /etc/sysctl.conf
```



- Also add the following two lines into `/etc/sysctl.conf`:

```
net.ipv4.conf.all.rp_filter=0
```

```
net.ipv4.conf.default.rp_filter=0
```

- To save you from reboot, perform the following

```
# sysctl net.ipv4.ip_forward=1
```

```
# sysctl net.ipv4.conf.all.rp_filter=0
```

```
# sysctl net.ipv4.conf.default.rp_filter=0
```

```
# sysctl -p
```

## Keystone

Keystone is an OpenStack project that provides Identity, Token, Catalog and Policy services for use specifically by projects in the OpenStack family. It implements OpenStack's Identity API.

- Install Keystone packages:

```
# apt-get install -y keystone
```

- Adapt the connection attribute in the `/etc/keystone/keystone.conf` to the new database:

```
connection = mysql://keystoneUser:keystonePass@10.10.10.51/keystone
```

- Restart the identity service then synchronize the database:

```
# service keystone restart
```

```
# keystone-manage db_sync
```

- Fill up the keystone database using the below two scripts:

[keystone\\_basic.sh](#)

[keystone\\_endpoints\\_basic.sh](#)

- Run Scripts:

```
$ chmod +x keystone_basic.sh
```

```
$ chmod +x keystone_endpoints_basic.sh
```

```
$ ./keystone_basic.sh
```

```
$ ./keystone_endpoints_basic.sh
```

- Create a simple credentials file

```
nano Credentials.sh
```

- Paste the following:

```
$ export OS_TENANT_NAME=admin
```

```
$ export OS_USERNAME=admin
```

```
$ export OS_PASSWORD=admin_pass
```

```
$ export OS_AUTH_URL="http://192.168.100.51:5000/v2.0/"
```

- Load the above credentials:

```
$ source Credentials.sh
```

- To test Keystone, we use a simple CLI command:

```
$ keystone user-list
```

## Glance

The OpenStack Glance project provides services for discovering, registering, and retrieving virtual machine images. Glance has a RESTful API that allows querying of VM image metadata as well as retrieval of the actual image.

VM images made available through Glance can be stored in a variety of locations from simple file systems to object-storage systems like the OpenStack Swift project.

Glance, as with all OpenStack projects, is written with the following design guidelines in mind:

- Component based architecture: Quickly adds new behaviors
- Highly available: Scales to very serious workloads
- Fault tolerant: Isolated processes avoid cascading failures
- Recoverable: Failures should be easy to diagnose, debug, and rectify
- Open standards: Be a reference implementation for a community-driven api
- Install Glance

```
# apt-get install -y glance
```

- Update /etc/glance/glance-api-paste.ini

```
[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
delay_auth_decision = true
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = service_pass
```

- Update the `/etc/glance/glance-registry-paste.ini`

```
[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = service_pass
```

- Update the `/etc/glance/glance-api.conf`

```
sql_connection = mysql://glanceUser:glancePass@10.10.10.51/glance
[keystone_authtoken]
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = service_pass

[paste_deploy]
flavor = keystone
```

- Update the `/etc/glance/glance-registry.conf`

```
sql_connection = mysql://glanceUser:glancePass@10.10.10.51/glance
[keystone_authtoken]
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = glance
admin_password = service_pass

[paste_deploy]
flavor = keystone
```

- Restart the glance-api and glance-registry services:

```
# service glance-api restart; service glance-registry restart
```

- Synchronize the Glance database:

```
# glance-manage db_sync
```

- To test Glance, upload the "cirros cloud image" directly from the internet:

```
$ glance image-create --name OS4Y_Cirros --is-public true --container-format bare --disk-format qcow2 --location https://launchpad.net/cirros/trunk/0.3.0/+download/cirros-0.3.0-x86_64-disk.img
```

- Check if the image is successfully uploaded:

```
$ glance image-list
```

## Neutron

Neutron is an OpenStack project to provide "network connectivity as a service" between interface devices (e.g., vNICs) managed by other OpenStack services (e.g., nova).

- Install the Neutron Server and the Open vSwitch package collection:

```
# apt-get install -y neutron-server
```

- Edit the /etc/neutron/plugins/openvswitch/ovs\_neutron\_plugin.ini:

```
[database]
connection = mysql://neutronUser:neutronPass@10.10.10.51/neutron

#Under the OVS section
[ovs]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
[agent]
tunnel_types = gre

#Firewall driver for realizing neutron security group function
[securitygroup]
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

- Edit the /etc/neutron/api-paste.ini:

```
[filter:authtoken]
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriverpaste.
filter_factory = keystoneclient.middleware.auth_token:filter_factory
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = neutron
admin_password = service_pass
```

- Edit the /etc/neutron/neutron.conf:

```
rabbit_host = 10.10.10.51
[keystone_authtoken]
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = neutron
admin_password = service_pass
signing_dir = /var/lib/neutron/keystone-signing

[database]
connection = mysql://neutronUser:neutronPass@10.10.10.51/neutron
```

- Restart Neutron services:

```
# service neutron-server restart
```

## Nova

Nova is the project name for OpenStack Compute, a cloud computing fabric controller, the main part of an IaaS system. Individuals and organizations can use Nova to host and manage their own cloud computing systems. Nova originated as a project out of NASA Ames Research Laboratory.

Nova is written with the following design guidelines in mind:

- Component based architecture: Quickly adds new behaviors.
- Highly available: Scales to very serious workloads.
- Fault-Tolerant: Isolated processes avoid cascading failures.
- Recoverable: Failures should be easy to diagnose, debug, and rectify.
- Open standards: Be a reference implementation for a community-driven api.

- API compatibility: Nova strives to be API-compatible with popular systems like Amazon EC2.

- Install nova components:

```
# apt-get install -y nova-novncproxy novnc nova-api nova-ajax-console-proxy nova-cert nova-conductor nova-consoleauth nova-doc nova-scheduler python-novaclient
```

- Edit /etc/nova/api-paste.ini

```
[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = service_pass
signing_dir = /tmp/keystone-signing-nova

# Workaround for https://bugs.launchpad.net/nova/+bug/1154809
auth_version = v2.0
```

- Edit /etc/nova/nova.conf

```
[DEFAULT]
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/run/lock/nova
verbose=True
api_paste_config=/etc/nova/api-paste.ini
compute_scheduler_driver=nova.scheduler.simple.SimpleScheduler
rabbit_host=10.10.10.51
nova_url=http://10.10.51:8774/v1.1/
sql_connection=mysql://novaUser:novaPass@10.10.10.51/nova
root_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf
```



```
# Auth
use_deprecated_auth=false
auth_strategy=keystone

# Imaging service
glance_api_servers=10.10.10.51:9292
image_service=nova.image.glance.GlanceImageService

# Vnc configuration
novnc_enabled=true
novncproxy_base_url=http://192.168.1.51:6080/vnc_auto.html
novncproxy_port=6080
vncserver_proxycient_address=10.10.10.51
vncserver_listen=0.0.0.0

# Network settings
network_api_class=nova.network.neutronv2.api.API
neutron_url=http://10.10.10.51:9696
neutron_auth_strategy=keystone
neutron_admin_tenant_name=service
neutron_admin_username=neutron
neutron_admin_password=service_pass
neutron_admin_auth_url=http://10.10.10.51:35357/v2.0
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
linuxnet_interface_driver=nova.network.linux_net.LinuxOVSInterfaceDriver

#If you want Neutron + Nova Security groups
firewall_driver=nova.virt.firewall.NoopFirewallDriver
security_group_api=neutron
#If you want Nova Security groups only, comment the two lines above and
uncomment line -1-.
#-1-firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver

#Metadata
service_neutron_metadata_proxy = True
neutron_metadata_proxy_shared_secret = helloOpenStack
```

```
# Compute #  
compute_driver=libvirt.LibvirtDriver  
  
# Cinder #  
volume_api_class=nova.volume.cinder.API  
osapi_volume_listen_port=5900
```

- Synchronize your database:

```
# nova-manage db sync
```

- Restart nova-\* services (all nova services):

```
# cd /etc/init.d/; for i in $( ls nova-* ); do service $i restart; done
```

- Check for the smiling faces on nova - \* services to confirm your installation:

```
# nova-manage service list
```

## Cinder

Cinder is an OpenStack project to provide “block storage as a service”.

- Component based architecture: Quickly adds new behavior.
- Highly available: Scales to very serious workloads.
- Fault-Tolerant: Isolated processes avoid cascading failures.
- Recoverable: Failures should be easy to diagnose, debug and rectify.
- Open standards: Be a reference implementation for a community-driven API.
- API compatibility: Cinder strives to be API-compatible with popular systems like Amazon EC2.

- Install Cinder components:

```
# apt-get install -y cinder-api cinder-scheduler cinder-volume iscsitarget open-iscsi  
iscsitarget-dkms
```

- Configure the iSCSI services:

```
# sed -i 's/false/true/g' /etc/default/iscsitarget
```

- Restart the services:

```
# service iscsitarget start
```

```
# service open-iscsi start
```

- Edit /etc/cinder/api-paste.ini:

```
[filter:authtoken]  
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory  
service_protocol = http  
service_host = 192.168.100.51  
service_port = 5000  
auth_host = 10.10.10.51  
auth_port = 35357  
auth_protocol = http  
admin_tenant_name = service  
admin_user = cinder  
admin_password = service_pass  
signing_dir = /var/lib/cinder
```

- Edit /etc/cinder/cinder.conf:

```
[DEFAULT]
rootwrap_config=/etc/cinder/rootwrap.conf
sql_connection = mysql://cinderUser:cinderPass@10.10.10.51/cinder
api_paste_config = /etc/cinder/api-paste.ini
iscsi_helper=ietadm
volume_name_template = volume-%s
volume_group = cinder-volumes
verbose = True
auth_strategy = keystone
iscsi_ip_address=10.10.10.51
rpc_backend = cinder.openstack.common.rpc.impl_kombu
rabbit_host = 10.10.10.51
rabbit_port = 5672
```

- Then, synchronize Cinder database:

```
# cinder-manage db sync
```

- Finally, create a volume group and name it cinder-volumes:

```
# dd if=/dev/zero of=cinder-volumes bs=1 count=0 seek=2G
```

```
# losetup /dev/loop2 cinder-volumes
```

```
# fdisk /dev/loop2
```

```
Command (m for help): n
```

```
Command (m for help): p
```

```
Command (m for help): 1
```

```
Command (m for help): t
```

```
Command (m for help): 8e
```

```
Command (m for help): w
```

- Proceed to create the physical volume then the volume group:

```
# pvcreate /dev/loop2
```

```
# vgcreate cinder-volumes /dev/loop2
```

- Note: Be aware that this volume group gets lost after a system reboot. If you do not want to perform this step again, make sure that you save the machine state and do not shut it down.
- Restart the Cinder services:

```
# cd /etc/init.d/; for i in $( ls cinder-* ); do service $i restart; done
```

- Verify if Cinder services are running:

```
# cd /etc/init.d/; for i in $( ls cinder-* ); do service $i status; done
```

## Horizon

Horizon is the canonical implementation of OpenStack's dashboard, which provides a web-based user interface to OpenStack services including Nova, Swift, Keystone, etc.

- To install Horizon, proceed with the following steps:

```
# apt-get install -y openstack-dashboard memcached
```

- If you do not like the OpenStack Ubuntu Theme, you can remove it with help of the below command:

```
# dpkg --purge openstack-dashboard-ubuntu-theme
```

- Reload Apache and memcached:

```
# service apache2 restart; service memcached restart
```



## 6. Controller Node Quiz

### Table of Contents

|                                   |     |
|-----------------------------------|-----|
| Days 2 to 4, 16:40 to 17:00 ..... | 143 |
|-----------------------------------|-----|

### Days 2 to 4, 16:40 to 17:00





## 7. Network Node

### Table of Contents

|                                                      |     |
|------------------------------------------------------|-----|
| Days 7 to 8, 09:00 to 11:00, 11:15 to 12:30 .....    | 145 |
| Review Associate Networking in OpenStack .....       | 145 |
| Review Associate OpenStack Networking Concepts ..... | 151 |
| Review Associate Administration Tasks .....          | 153 |
| Operator OpenStack Neutron Use Cases .....           | 153 |
| Operator OpenStack Neutron Security .....            | 163 |
| Operator OpenStack Neutron Floating IPs .....        | 165 |

### Days 7 to 8, 09:00 to 11:00, 11:15 to 12:30

## Review Associate Networking in OpenStack

### Networking in OpenStack

OpenStack Networking provides a rich tenant-facing API for defining network connectivity and addressing in the cloud. The OpenStack Networking project gives operators the ability to leverage different networking technologies to power their cloud networking. It is a virtual network service that provides a powerful API to define the network connectivity and addressing used by devices from other services, such as OpenStack Compute. It has a rich API which consists of the following components.

- **Network:** An isolated L2 segment, analogous to VLAN in the physical networking world.

- **Subnet:** A block of v4 or v6 IP addresses and associated configuration state.
- **Port:** A connection point for attaching a single device, such as the NIC of a virtual server, to a virtual network. Also describes the associated network configuration, such as the MAC and IP addresses to be used on that port.

You can configure rich network topologies by creating and configuring networks and subnets, and then instructing other OpenStack services like OpenStack Compute to attach virtual devices to ports on these networks. In particular, OpenStack Networking supports each tenant having multiple private networks, and allows tenants to choose their own IP addressing scheme, even if those IP addresses overlap with those used by other tenants. This enables very advanced cloud networking use cases, such as building multi-tiered web applications and allowing applications to be migrated to the cloud without changing IP addresses.

### Plugin Architecture: Flexibility to Choose Different Network Technologies

Enhancing traditional networking solutions to provide rich cloud networking is challenging. Traditional networking is not designed to scale to cloud proportions or to configure automatically.

The original OpenStack Compute network implementation assumed a very basic model of performing all isolation through Linux VLANs and IP tables. OpenStack Networking introduces the concept of a plug-in, which is a pluggable back-end implementation of the OpenStack Networking API. A plug-in can use a variety of technologies to implement the logical API requests. Some OpenStack Networking plug-ins might use basic Linux VLANs and IP tables, while others might use more advanced technologies, such as L2-in-L3 tunneling or OpenFlow, to provide similar benefits.

The current set of plug-ins include:

- **Big Switch, Floodlight REST Proxy:** <http://www.openflowhub.org/display/floodlightcontroller/Quantum+REST+Proxy+Plugin>
- **Brocade Plugin**
- **Cisco:** Documented externally at: <http://wiki.openstack.org/cisco-quantum>

- **Hyper-V Plugin**
- **Linux Bridge:** Documentation included in this guide and <http://wiki.openstack.org/Quantum-Linux-Bridge-Plugin>
- **Midonet Plugin**
- **NEC OpenFlow:** <http://wiki.openstack.org/Quantum-NEC-OpenFlow-Plugin>
- **Open vSwitch:** Documentation included in this guide.
- **PLUMgrid:** <https://wiki.openstack.org/wiki/Plumgrid-quantum>
- **Ryu:** <https://github.com/osrg/ryu/wiki/OpenStack>
- **VMware NSX:** Documentation include in this guide, [NSX Product Overview](#) , and [NSX Product Support](#).

Plugins can have different properties in terms of hardware requirements, features, performance, scale, operator tools, etc. Supporting many plug-ins enables the cloud administrator to weigh different options and decide which networking technology is right for the deployment.

### Components of OpenStack Networking

To deploy OpenStack Networking, it is useful to understand the different components that make up the solution and how those components interact with each other and with other OpenStack services.

OpenStack Networking is a standalone service, just like other OpenStack services such as OpenStack Compute, OpenStack Image Service, OpenStack Identity service, and the OpenStack Dashboard. Like those services, a deployment of OpenStack Networking often involves deploying several processes on a variety of hosts.

The main process of the OpenStack Networking server is quantum-server, which is a Python daemon that exposes the OpenStack Networking API and passes user requests to the configured OpenStack Networking

plug-in for additional processing. Typically, the plug-in requires access to a database for persistent storage, similar to other OpenStack services.

If your deployment uses a controller host to run centralized OpenStack Compute components, you can deploy the OpenStack Networking server on that same host. However, OpenStack Networking is entirely standalone and can be deployed on its own server as well. OpenStack Networking also includes additional agents that might be required depending on your deployment:

- **plugin agent (quantum-\*agent):**Runs on each hypervisor to perform local vswitch configuration. Agent to be run depends on which plug-in you are using, as some plug-ins do not require an agent.
- **dhcp agent (quantum-dhcp-agent):**Provides DHCP services to tenant networks. This agent is the same across all plug-ins.
- **l3 agent (quantum-l3-agent):**Provides L3/NAT forwarding to provide external network access for VMs on tenant networks. This agent is the same across all plug-ins.

These agents interact with the main quantum-server process in the following ways:

- Through RPC. For example, rabbitmq or qpid.
- Through the standard OpenStack Networking API.

OpenStack Networking relies on the OpenStack Identity Project (Keystone) for authentication and authorization of all API request.

OpenStack Compute interacts with OpenStack Networking through calls to its standard API. As part of creating a VM, nova-compute communicates with the OpenStack Networking API to plug each virtual NIC on the VM into a particular network.

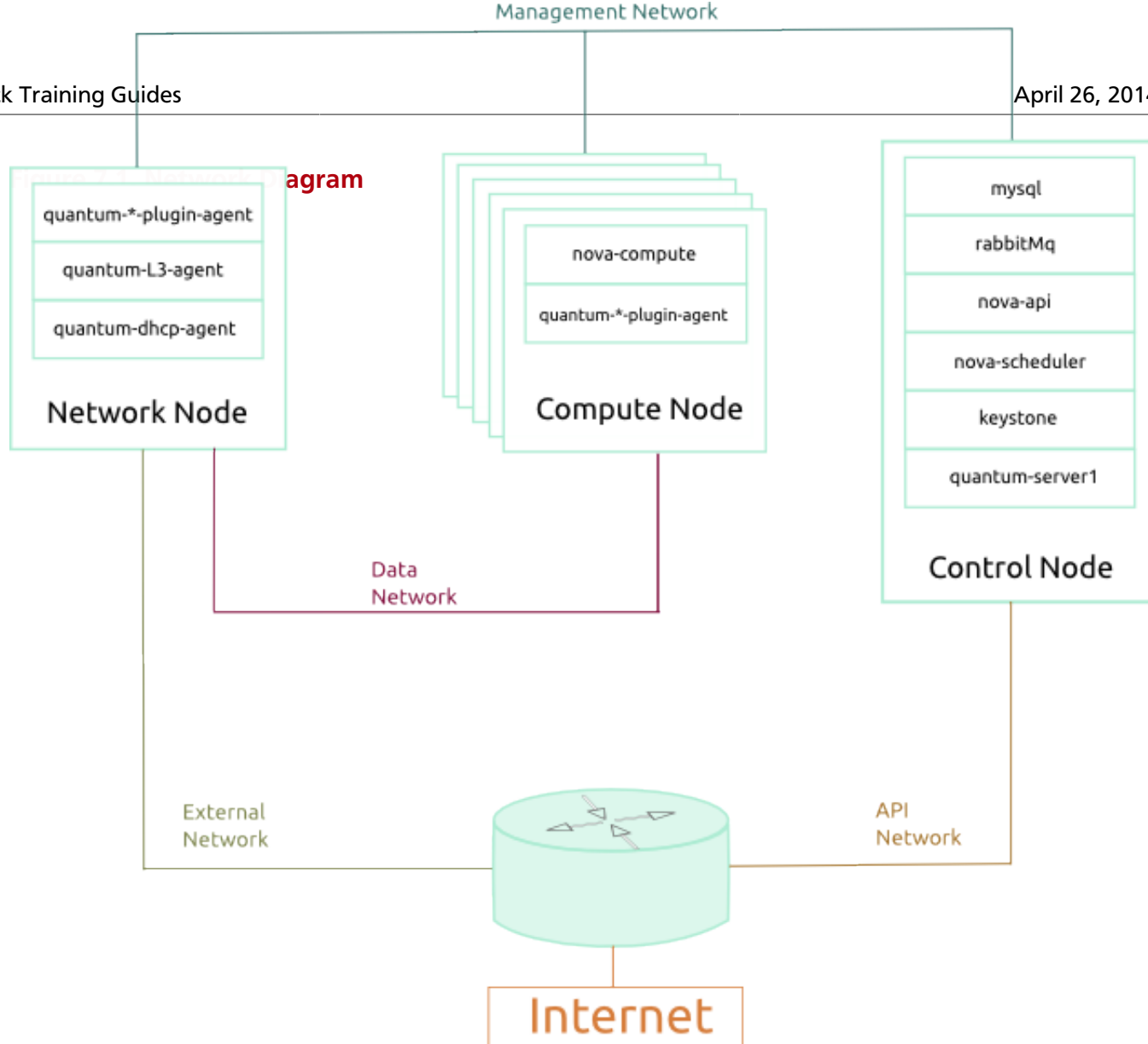
The OpenStack Dashboard (Horizon) has integration with the OpenStack Networking API, allowing administrators and tenant users, to create and manage network services through the Horizon GUI.

### **Place Services on Physical Hosts**

Like other OpenStack services, OpenStack Networking provides cloud administrators with significant flexibility in deciding which individual services should run on which physical devices. On one extreme, all service daemons can be run on a single physical host for evaluation purposes. On the other, each service could have its own physical hosts, and some cases be replicated across multiple hosts for redundancy.

In this guide, we focus primarily on a standard architecture that includes a “cloud controller” host, a “network gateway” host, and a set of hypervisors for running VMs. The “cloud controller” and “network gateway” can be combined in simple deployments, though if you expect VMs to send significant amounts of traffic to or from the Internet, a dedicated network gateway host is suggested to avoid potential CPU contention between packet forwarding performed by the quantum-l3-agent and other OpenStack services.

### **Network Connectivity for Physical Hosts**



A standard OpenStack Networking setup has up to four distinct physical data center networks:

- **Management network:**Used for internal communication between OpenStack Components. The IP addresses on this network should be reachable only within the data center.
- **Data network:**Used for VM data communication within the cloud deployment. The IP addressing requirements of this network depend on the OpenStack Networking plug-in in use.
- **External network:**Used to provide VMs with Internet access in some deployment scenarios. The IP addresses on this network should be reachable by anyone on the Internet.
- **API network:**Exposes all OpenStack APIs, including the OpenStack Networking API, to tenants. The IP addresses on this network should be reachable by anyone on the Internet. This may be the same network as the external network, as it is possible to create a subnet for the external network that uses IP allocation ranges to use only less than the full range of IP addresses in an IP block.

## Review Associate OpenStack Networking Concepts

### Network Types

The OpenStack Networking configuration provided by the Rackspace Private Cloud cookbooks allows you to choose between VLAN or GRE isolated networks, both provider- and tenant-specific. From the provider side, an administrator can also create a flat network.

The type of network that is used for private tenant networks is determined by the `network_type` attribute, which can be edited in the Chef `override_attributes`. This attribute sets both the default provider network type and the only type of network that tenants are able to create. Administrators can always create flat and VLAN networks. GRE networks of any type require the `network_type` to be set to `gre`.

### Namespaces

For each network you create, the Network node (or Controller node, if combined) will have a unique network namespace (netns) created by the DHCP and Metadata agents. The netns hosts an interface and IP addresses

for dnsmasq and the quantum-ns-metadata-proxy. You can view the namespaces with the `ip netns [list]`, and can interact with the namespaces with the `ip netns exec <namespace> <command>` command.

## Metadata

Not all networks or VMs need metadata access. Rackspace recommends that you use metadata if you are using a single network. If you need metadata, you may also need a default route. (If you don't need a default route, no-gateway will do.)

To communicate with the metadata IP address inside the namespace, instances need a route for the metadata network that points to the dnsmasq IP address on the same namespaced interface. OpenStack Networking only injects a route when you do not specify a gateway-ip in the subnet.

If you need to use a default route and provide instances with access to the metadata route, create the subnet without specifying a gateway IP and with a static route from 0.0.0.0/0 to your gateway IP address. Adjust the DHCP allocation pool so that it will not assign the gateway IP. With this configuration, dnsmasq will pass both routes to instances. This way, metadata will be routed correctly without any changes on the external gateway.

## OVS Bridges

An OVS bridge for provider traffic is created and configured on the nodes where single-network-node and single-compute are applied. Bridges are created, but physical interfaces are not added. An OVS bridge is not created on a Controller-only node.

When creating networks, you can specify the type and properties, such as Flat vs. VLAN, Shared vs. Tenant, or Provider vs. Overlay. These properties identify and determine the behavior and resources of instances attached to the network. The cookbooks will create bridges for the configuration that you specify, although they do not add physical interfaces to provider bridges. For example, if you specify a network type of GRE, a br-tun tunnel bridge will be created to handle overlay traffic.



## Review Associate Administration Tasks

TBD

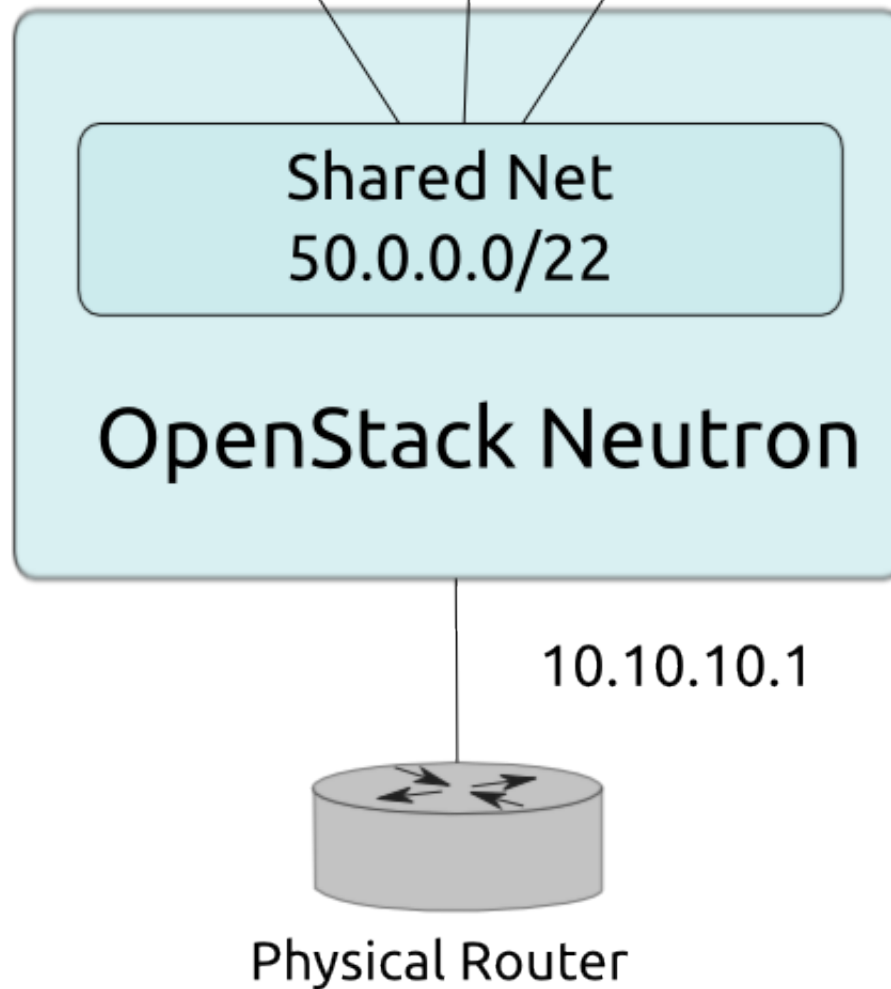
## Operator OpenStack Neutron Use Cases

As of now you must be wondering, how to use these awesome features that OpenStack Networking has given to us.

### **Use Case: Single Flat Network**

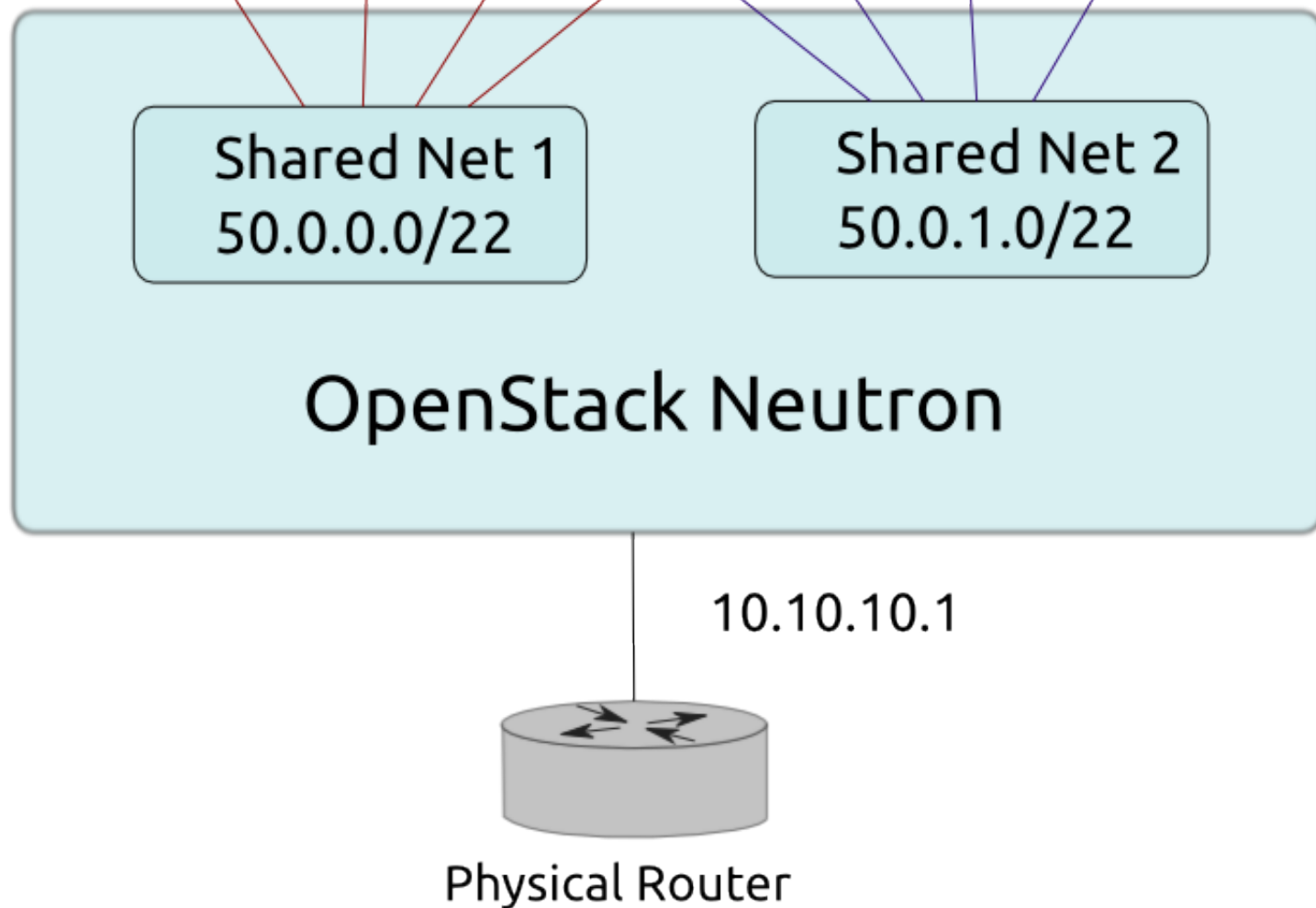
In the simplest use case, a single OpenStack Networking network exists. This is a "shared" network, meaning it is visible to all tenants via the OpenStack Networking API. Tenant VMs have a single NIC, and receive a fixed IP address from the subnet(s) associated with that network. This essentially maps to the FlatManager and FlatDHCPManager models provided by OpenStack Compute. Floating IPs are not supported.

It is common that an OpenStack Networking network is a "provider network", meaning it was created by the OpenStack administrator to map directly to an existing physical network in the data center. This allows the provider to use a physical router on that data center network as the gateway for VMs to reach the outside world. For each subnet on an external network, the gateway configuration on the physical router must be manually configured outside of OpenStack.

**Figure 7.2. Single Flat Network**

**Use Case: Multiple Flat Network**

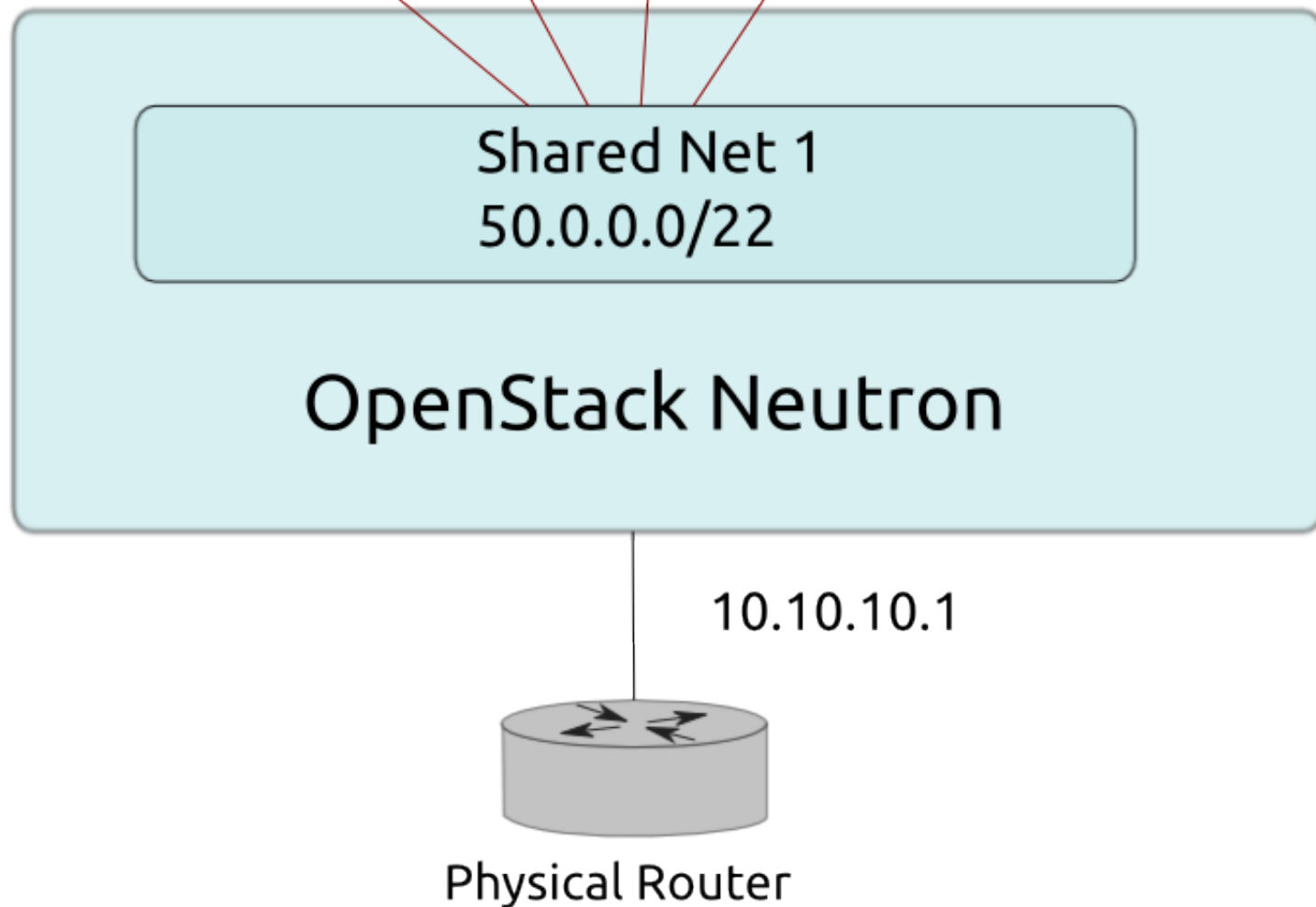
This use case is very similar to the above Single Flat Network use case, except that tenants see multiple shared networks via the OpenStack Networking API and can choose which network (or networks) to plug into.

**Figure 7.3. Multiple Flat Network**

**Use Case: Mixed Flat and Private Network**

This use case is an extension of the above flat network use cases, in which tenants also optionally have access to private per-tenant networks. In addition to seeing one or more shared networks via the OpenStack Networking API, tenants can create additional networks that are only visible to users of that tenant. When creating VMs, those VMs can have NICs on any of the shared networks and/or any of the private networks belonging to the tenant. This enables the creation of "multi-tier" topologies using VMs with multiple NICs. It also supports a model where a VM acting as a gateway can provide services such as routing, NAT, or load balancing.

**Figure 7.4. Mixed Flat and Private Network**



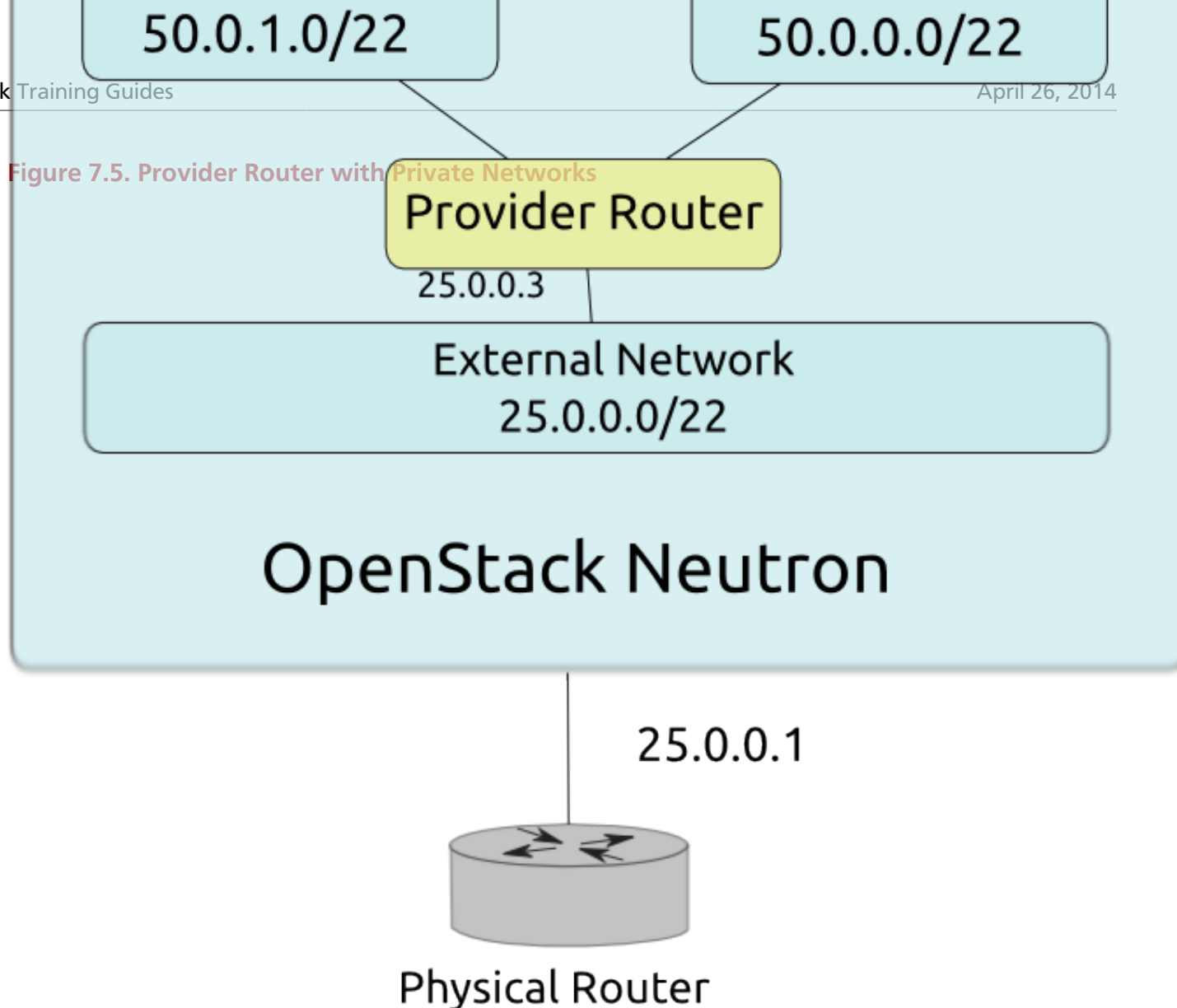
**Use Case: Provider Router with Private Networks**

This use provides each tenant with one or more private networks, which connect to the outside world via an OpenStack Networking router. The case where each tenant gets exactly one network in this form maps to the same logical topology as the VlanManager in OpenStack Compute (of course, OpenStack Networking doesn't require VLANs). Using the OpenStack Networking API, the tenant would only see a network for each private network assigned to that tenant. The router object in the API is created and owned by the cloud admin.

This model supports giving VMs public addresses using "floating IPs", in which the router maps public addresses from the external network to fixed IPs on private networks. Hosts without floating IPs can still create outbound connections to the external network, as the provider router performs SNAT to the router's external IP. The IP address of the physical router is used as the gateway\_ip of the external network subnet, so the provider has a default router for Internet traffic.

The router provides L3 connectivity between private networks, meaning that different tenants can reach each others instances unless additional filtering, such as security groups, is used. Because there is only a single router, tenant networks cannot use overlapping IPs. Thus, it is likely that the admin would create the private networks on behalf of tenants.

Figure 7.5. Provider Router with Private Networks

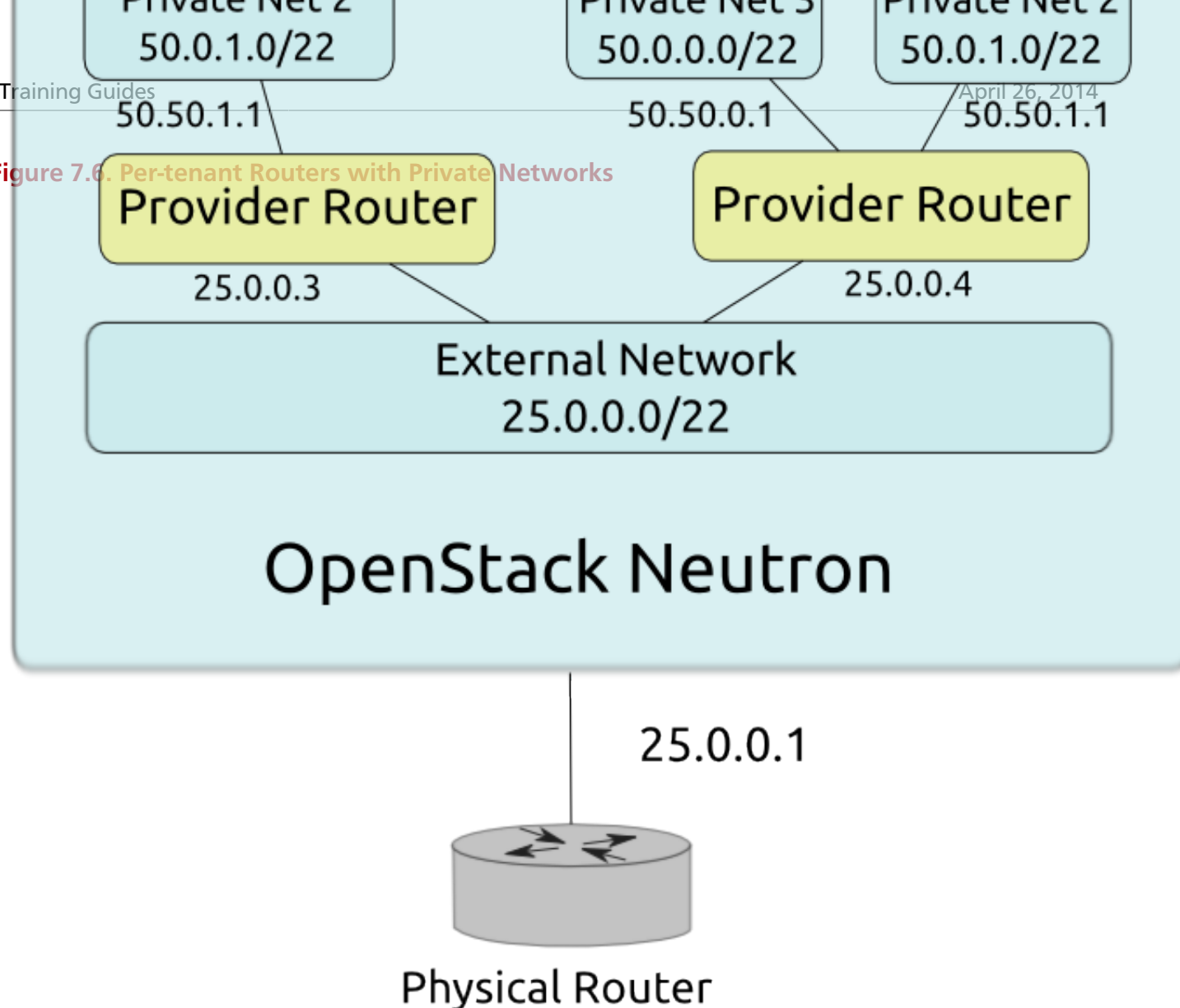




**Use Case: Per-tenant Routers with Private Networks**

A more advanced router scenario in which each tenant gets at least one router, and potentially has access to the OpenStack Networking API to create additional routers. The tenant can create their own networks, potentially uplinking those networks to a router. This model enables tenant-defined multi-tier applications, with each tier being a separate network behind the router. Since there are multiple routers, tenant subnets can be overlapping without conflicting, since access to external networks all happens via SNAT or Floating IPs. Each router uplink and floating IP is allocated from the external network subnet.

Figure 7.6. Per-tenant Routers with Private Networks



# Operator OpenStack Neutron Security

## Security Groups

Security groups and security group rules allows administrators and tenants the ability to specify the type of traffic and direction (ingress/egress) that is allowed to pass through a port. A security group is a container for security group rules.

When a port is created in OpenStack Networking it is associated with a security group. If a security group is not specified the port will be associated with a 'default' security group. By default this group will drop all ingress traffic and allow all egress. Rules can be added to this group in order to change the behaviour.

If one desires to use the OpenStack Compute security group APIs and/or have OpenStack Compute orchestrate the creation of new ports for instances on specific security groups, additional configuration is needed. To enable this, one must configure the following file `/etc/nova/nova.conf` and set the config option `security_group_api=neutron` on every node running nova-compute and nova-api. After this change is made restart nova-api and nova-compute in order to pick up this change. After this change is made one will be able to use both the OpenStack Compute and OpenStack Network security group API at the same time.

## Authentication and Authorization

OpenStack Networking uses the OpenStack Identity service (project name keystone) as the default authentication service. When OpenStack Identity is enabled Users submitting requests to the OpenStack Networking service must provide an authentication token in X-Auth-Token request header. The aforementioned token should have been obtained by authenticating with the OpenStack Identity endpoint. For more information concerning authentication with OpenStack Identity, please refer to the OpenStack Identity documentation. When OpenStack Identity is enabled, it is not mandatory to specify `tenant_id` for resources in create requests, as the tenant identifier will be derived from the Authentication token. Please note that the default authorization settings only allow administrative users to create resources on behalf of a different tenant. OpenStack Networking uses information received from OpenStack Identity to authorize user requests. OpenStack Networking handles two kind of authorization policies:

- **Operation-based:** policies specify access criteria for specific operations, possibly with fine-grained control over specific attributes;
- **Resource-based:** whether access to specific resource might be granted or not according to the permissions configured for the resource (currently available only for the network resource). The actual authorization policies enforced in OpenStack Networking might vary from deployment to deployment.

The policy engine reads entries from the `policy.json` file. The actual location of this file might vary from distribution to distribution. Entries can be updated while the system is running, and no service restart is required. That is to say, every time the policy file is updated, the policies will be automatically reloaded. Currently the only way of updating such policies is to edit the policy file. Please note that in this section we will use both the terms "policy" and "rule" to refer to objects which are specified in the same way in the policy file; in other words, there are no syntax differences between a rule and a policy. We will define a policy something which is matched directly from the OpenStack Networking policy engine, whereas we will define a rule as the elements of such policies which are then evaluated. For instance in `create_subnet: [{"admin_or_network_owner"}]`, `create_subnet` is regarded as a policy, whereas `admin_or_network_owner` is regarded as a rule.

Policies are triggered by the OpenStack Networking policy engine whenever one of them matches an OpenStack Networking API operation or a specific attribute being used in a given operation. For instance the `create_subnet` policy is triggered every time a `POST /v2.0/subnets` request is sent to the OpenStack Networking server; on the other hand `create_network:shared` is triggered every time the `shared` attribute is explicitly specified (and set to a value different from its default) in a `POST /v2.0/networks` request. It is also worth mentioning that policies can be also related to specific API extensions; for instance `extension:provider_network:set` will be triggered if the attributes defined by the Provider Network extensions are specified in an API request.

An authorization policy can be composed by one or more rules. If more rules are specified, evaluation policy will be successful if any of the rules evaluates successfully; if an API operation matches multiple policies, then all the policies must evaluate successfully. Also, authorization rules are recursive. Once a rule is matched, the rule(s) can be resolved to another rule, until a terminal rule is reached.

The OpenStack Networking policy engine currently defines the following kinds of terminal rules:

- **Role-based rules:** evaluate successfully if the user submitting the request has the specified role. For instance "role:admin" is successful if the user submitting the request is an administrator.
- **Field-based rules:** evaluate successfully if a field of the resource specified in the current request matches a specific value. For instance "field:networks:shared=True" is successful if the attribute shared of the network resource is set to true.
- **Generic rules:** compare an attribute in the resource with an attribute extracted from the user's security credentials and evaluates successfully if the comparison is successful. For instance "tenant\_id:%(tenant\_id)s" is successful if the tenant identifier in the resource is equal to the tenant identifier of the user submitting the request.

## Operator OpenStack Neutron Floating IPs

OpenStack Networking has the concept of Fixed IPs and Floating IPs. Fixed IPs are assigned to an instance on creation and stay the same until the instance is explicitly terminated. Floating ips are ip addresses that can be dynamically associated with an instance. This address can be disassociated and associated with another instance at any time.

Various tasks carried out by Floating IP's as of now.

- create IP ranges under a certain group, only available for admin role.
- allocate an floating IP to a certain tenant, only available for admin role.
- deallocate an floating IP from a certain tenant
- associate an floating IP to a given instance
- disassociate an floating IP from a certain instance

Just as shown by the above figure, we will have nova-network-api to support nova client floating commands. nova-network-api will invoke neutron cli lib to interactive with neutron server via API. The data about floating IPs will be stored in to neutron DB. Neutron Agent, which is running on compute host will enforce the floating IP.

### Multiple Floating IP Pools

The L3 API in OpenStack Networking supports multiple floating IP pools. In OpenStack Networking, a floating IP pool is represented as an external network and a floating IP is allocated from a subnet associated with the external network. Since each L3 agent can be associated with at most one external network, we need to invoke multiple L3 agent to define multiple floating IP pools. 'gateway\_external\_network\_id' in L3 agent configuration file indicates the external network that the L3 agent handles. You can run multiple L3 agent instances on one host.

In addition, when you run multiple L3 agents, make sure that `handle_internal_only_routers` is set to `True` only for one L3 agent in an OpenStack Networking deployment and set to `False` for all other L3 agents. Since the default value of this parameter is `True`, you need to configure it carefully.

Before starting L3 agents, you need to create routers and external networks, then update the configuration files with UUID of external networks and start L3 agents.

For the first agent, invoke it with the following `l3_agent.ini` where `handle_internal_only_routers` is `True`.

## 8. Network Node Lab

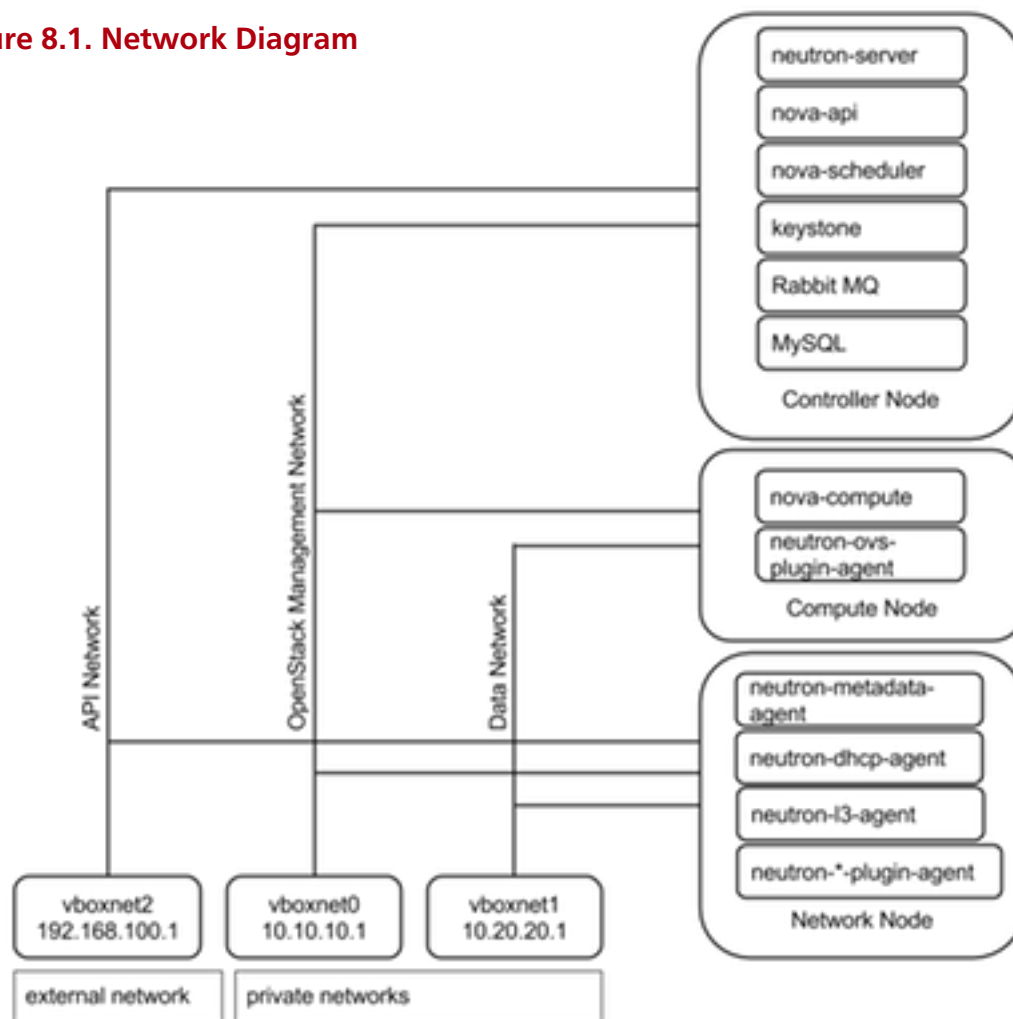
### Table of Contents

|                                                   |     |
|---------------------------------------------------|-----|
| Days 7 to 8, 13:30 to 14:45, 15:00 to 17:00 ..... | 167 |
| Network Node Lab .....                            | 167 |

### Days 7 to 8, 13:30 to 14:45, 15:00 to 17:00

### Network Node Lab

#### 1. Network Diagram :

**Figure 8.1. Network Diagram**



Publicly editable image source at [https://docs.google.com/drawings/d/1GX3FXmkz3c\\_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing](https://docs.google.com/drawings/d/1GX3FXmkz3c_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing)

**Vboxnet0, Vboxnet1, Vboxnet2** - are virtual networks setup up by virtual box with your host machine. This is the way your host can communicate with the virtual machines. These networks are in turn used by virtual box VM's for OpenStack networks, so that OpenStack's services can communicate with each other.

### Network Node

Start your Controller Node the one you setup in previous section.

#### Preparing Ubuntu 12.04

- After you install Ubuntu Server, go in sudo mode

```
$ sudo su
```

- Add Havana repositories:

```
# apt-get install ubuntu-cloud-keyring python-software-properties software-properties-common  
python-keyring
```

```
# echo deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/icehouse main  
>> /etc/apt/sources.list.d/icehouse.list
```

- Update your system:

```
# apt-get update
```

```
# apt-get upgrade
```

```
# apt-get dist-upgrade
```

- Install NTP and other services:

```
# apt-get install ntp vlan bridge-utils
```

- Configure NTP Server to Controller Node:

```
# sed -i 's/server 0.ubuntu.pool.ntp.org/#server0.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

```
# sed -i 's/server 1.ubuntu.pool.ntp.org/#server1.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

```
# sed -i 's/server 2.ubuntu.pool.ntp.org/#server2.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

```
# sed -i 's/server 3.ubuntu.pool.ntp.org/#server3.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

- Enable IP Forwarding by adding the following to `/etc/sysctl.conf`:

```
net.ipv4.ip_forward=1
net.ipv4.conf.all.rp_filter=0
net.ipv4.conf.default.rp_filter=0
```

- Run the following commands:

```
# sysctl net.ipv4.ip_forward=1
```

```
# sysctl net.ipv4.conf.all.rp_filter=0
```

```
# sysctl net.ipv4.conf.default.rp_filter=0
```

```
# sysctl -p
```

## Open vSwitch

- Install Open vSwitch Packages:

```
# apt-get install -y openvswitch-switch openvswitch-datapath-dkms
```

- Create the bridges:

```
# ovs-vsctl add-br br-int
```

```
# ovs-vsctl add-br br-ex
```

## Neutron

- Neutron:

```
# apt-get install neutron-server neutron-dhcp-agent neutron-plugin-openvswitch-agent  
neutron-l3-agent
```

- Edit /etc/neutron/api-paste.ini:

```
[filter:authtoken]  
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory  
auth_host = 10.10.10.51  
auth_port = 35357  
auth_protocol = http  
admin_tenant_name = service  
admin_user = neutron  
admin_password = service_pass
```

- Edit /etc/neutron/plugins/openvswitch/ovs\_neutron\_plugin.ini:

```
#Under the database section  
[DATABASE]  
connection = mysql://neutronUser:neutronPass@10.10.10.51/neutron  
#Under the OVS section  
[OVS]  
tenant_network_type = gre  
tunnel_id_ranges = 1:1000  
integration_bridge = br-int  
tunnel_bridge = br-tun  
local_ip = 10.10.10.51  
enable_tunneling = True
```

```
tunnel_type = gre
[agent]
tunnel_types = gre
#Firewall driver for realizing quantum security group function
[SECURITYGROUP]
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

- Edit /etc/neutron/metadata\_agent.ini:

```
# The Neutron user information for accessing the Neutron API.
auth_url = http://10.10.10.51:35357/v2.0
auth_region = RegionOne
admin_tenant_name = service
admin_user = neutron
admin_password = service_pass
# IP address used by Nova metadata server
nova_metadata_ip = 10.10.10.51
# TCP Port used by Nova metadata server
nova_metadata_port = 8775
metadata_proxy_shared_secret = helloOpenStack
```

- Edit /etc/neutron/dhcp\_agent.ini:

```
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
```

- Edit /etc/neutron/l3\_agent.ini:

```
[DEFAULT]
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge = br-ex
```

- Edit /etc/neutron/neutron.conf:

```
rabbit_host = 10.10.10.51
#And update the keystone_auth token section
[keystone_auth token]
auth_host = 10.10.10.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = neutron
admin_password = service_pass
signing_dir = /var/lib/neutron/keystone-signing
[database]
connection = mysql://neutronUser:neutronPass@10.10.10.51/neutron
```

- Edit /etc/sudoers.d/neutron\_sudoers::

```
#Modify the neutron user
neutron ALL=NOPASSWD: ALL
```

- Restart Services:

```
# for i in neutron-dhcp-agent neutron-metadata-agent neutron-  
plugin-agent neutron-l3-agent neutron-server; do service $i  
restart; done
```

- Edit Network Interfaces file /etc/network/interfaces:

```
auto eth2
iface eth2 inet manual
up ifconfig $IFACE 0.0.0.0 up
up ip link set $IFACE promisc on
down ip link set $IFACE promisc off
down ifconfig $IFACE down

auto br-ex
iface br-ex inet static
address 192.168.100.52
netmask 255.255.255.0
gateway 192.168.100.1
dns-nameservers 8.8.8.8
```

- Update your system:

```
# ovs-vsctl add-port br-ex eth2
```

## 9. Network Node Quiz

### Table of Contents

|                                   |     |
|-----------------------------------|-----|
| Days 7 to 8, 16:40 to 17:00 ..... | 175 |
|-----------------------------------|-----|

### Days 7 to 8, 16:40 to 17:00





# 10. Compute Node

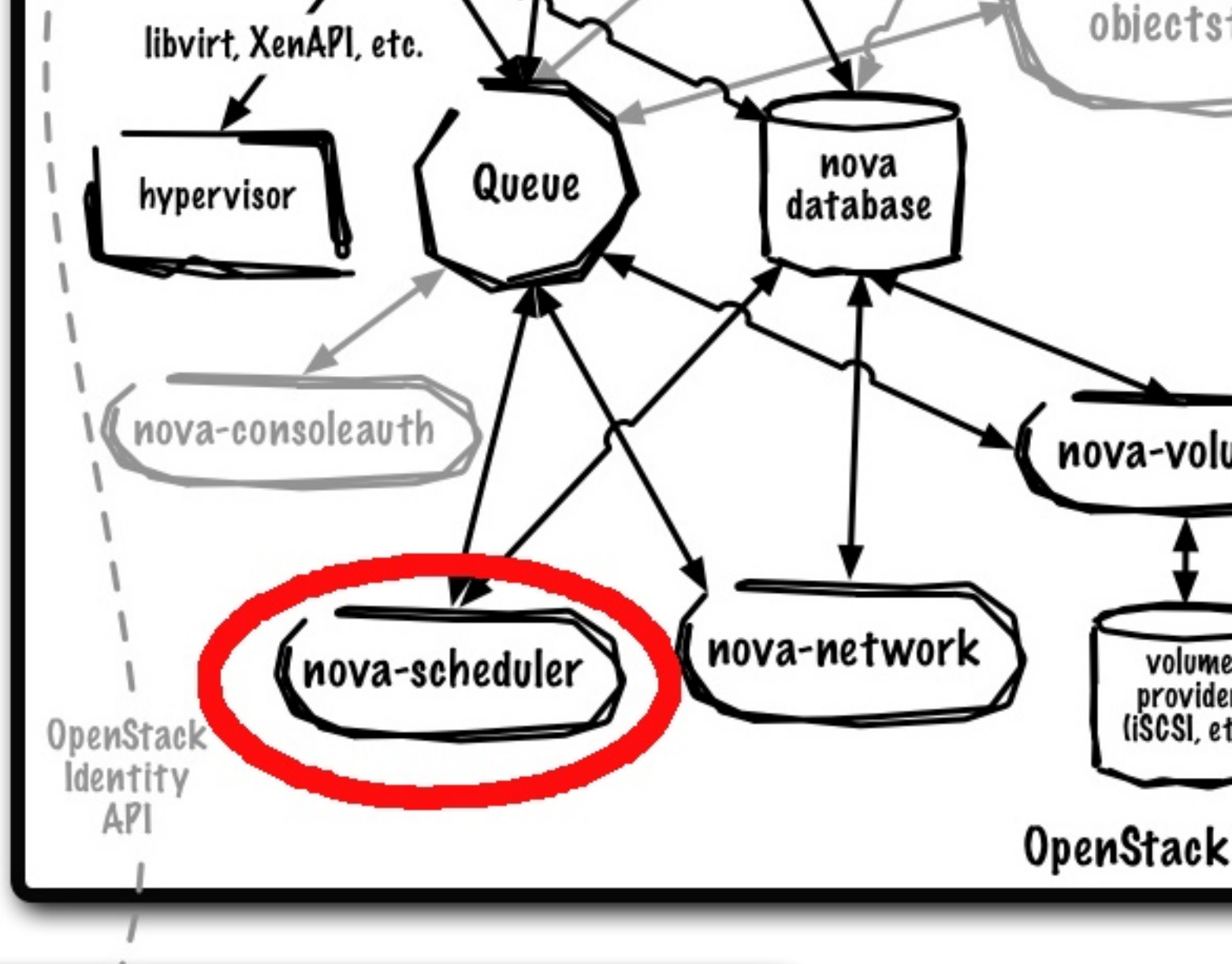
## Table of Contents

|                                                   |     |
|---------------------------------------------------|-----|
| Days 5 to 6, 09:00 to 11:00, 11:15 to 12:30 ..... | 177 |
| Review Associate VM Placement .....               | 177 |
| Review Associate VM Provisioning Indepth .....    | 185 |
| Review Associate OpenStack Block Storage .....    | 189 |
| Review Associate Administration Tasks .....       | 194 |

## Days 5 to 6, 09:00 to 11:00, 11:15 to 12:30

### Review Associate VM Placement

Compute uses the nova-scheduler service to determine how to dispatch compute and volume requests. For example, the nova-scheduler service determines which host a VM should launch on. The term host in the context of filters means a physical node that has a nova -compute service running on it. You can configure the scheduler through a variety of options.



Just as shown by above figure, nova-scheduler interacts with other components through queue and central database repo. For scheduling, queue is the essential communications hub.

All compute nodes (also known as hosts in terms of OpenStack) periodically publish their status, resources available and hardware capabilities to nova-scheduler through the queue. nova-scheduler then collects this data and uses it to make decisions when a request comes in.

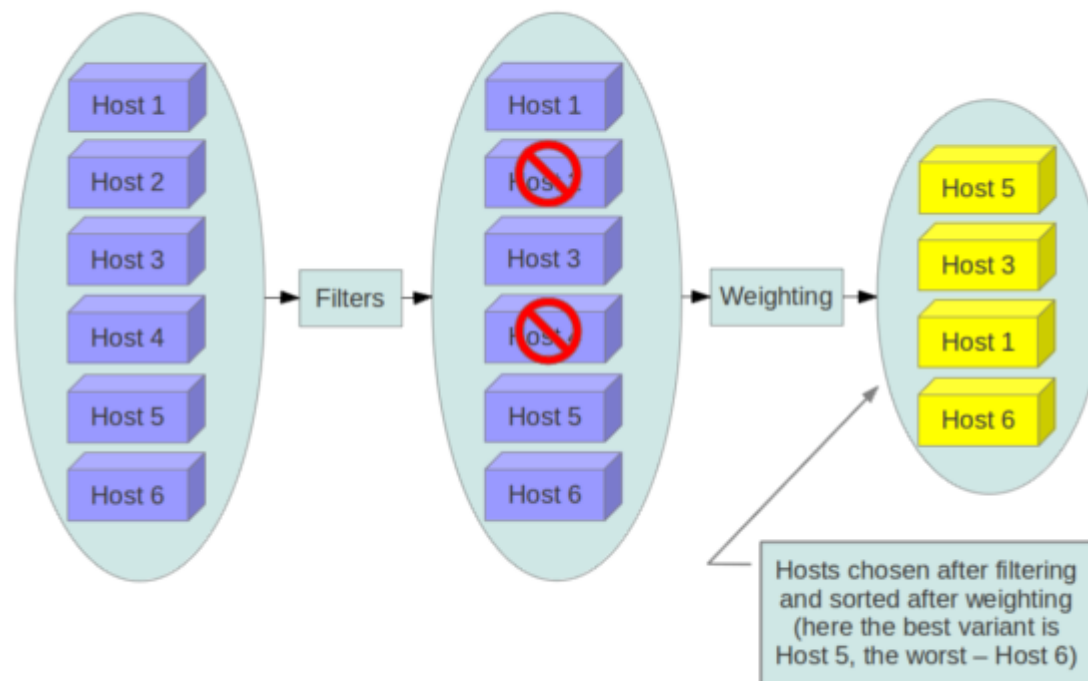
By default, the compute scheduler is configured as a filter scheduler, as described in the next section. In the default configuration, this scheduler considers hosts that meet all the following criteria:

- Are in the requested availability zone (AvailabilityZoneFilter).
- Have sufficient RAM available (RamFilter).
- Are capable of servicing the request (ComputeFilter).

### **Filter Scheduler**

The Filter Scheduler supports filtering and weighting to make informed decisions on where a new instance should be created. This Scheduler supports only working with Compute Nodes.

### **Filtering**

**Figure 10.2. Filtering**

During its work, Filter Scheduler first makes a dictionary of unfiltered hosts, then filters them using filter properties and finally chooses hosts for the requested number of instances (each time it chooses the most weighed host and appends it to the list of selected hosts).

If it turns up, that it can't find candidates for the next instance, it means that there are no more appropriate hosts where the instance could be scheduled.

If we speak about filtering and weighting, their work is quite flexible in the Filter Scheduler. There are a lot of filtering strategies for the Scheduler to support. Also you can even implement your own algorithm of filtering.

There are some standard filter classes to use (`nova.scheduler.filters`):

- `AllHostsFilter` - frankly speaking, this filter does no operation. It passes all the available hosts.
- `ImagePropertiesFilter` - filters hosts based on properties defined on the instance's image. It passes hosts that can support the specified image properties contained in the instance.
- `AvailabilityZoneFilter` - filters hosts by availability zone. It passes hosts matching the availability zone specified in the instance properties.
- `ComputeCapabilitiesFilter` - checks that the capabilities provided by the host Compute service satisfy any extra specifications associated with the instance type. It passes hosts that can create the specified instance type.
- The extra specifications can have a scope at the beginning of the key string of a key/value pair. The scope format is `scope:key` and can be nested, i.e. `key_string := scope:key_string`. Example like `capabilities:cpu_info: features` is valid scope format. A key string without any `:` is non-scope format. Each filter defines its valid scope, and not all filters accept non-scope format.
- The extra specifications can have an operator at the beginning of the value string of a key/value pair. If there is no operator specified, then a default operator of `s==` is used. Valid operators are:  
`*` = (equal to or greater than as a number; same as `vcpus` case) `* ==` (equal to as a number) `* !=` (not equal to as a number) `* >=` (greater than or equal to as a number) `* <=` (less than or equal to as a number) `* s==` (equal to as a string) `* s!=` (not equal to as a string) `* s>=` (greater than or equal to as a string) `* s>` (greater than as a string) `* s<=` (less than or equal to as a string) `* s<` (less than as a string) `* <in>` (substring) `* <or>` (find one of these) Examples are: `">= 5"`, `"s== 2.1.0"`, `"<in> gcc"`, and `"<or> fpu <or> gpu"`

```
class RamFilter(filters.BaseHostFilter):
    """Ram Filter with over subscription flag"""

    def host_passes(self, host_state, filter_properties):
        """Only return hosts with sufficient available RAM."""

        instance_type = filter_properties.get('instance_type')
        requested_ram = instance_type['memory_mb']
        free_ram_mb = host_state.free_ram_mb
        total_usable_ram_mb = host_state.total_usable_ram_mb
        used_ram_mb = total_usable_ram_mb - free_ram_mb
        return total_usable_ram_mb * FLAGS.ram_allocation_ratio - used_ram_mb >= requested_ram
```

Here `ram_allocation_ratio` means the virtual RAM to physical RAM allocation ratio (it is 1.5 by default). Really, nice and simple.

Next standard filter to describe is `AvailabilityZoneFilter` and it isn't difficult too. This filter just looks at the availability zone of compute node and availability zone from the properties of the request. Each Compute service has its own availability zone. So deployment engineers have an option to run scheduler with availability zones support and can configure availability zones on each compute host. This classes method `host_passes` returns True if availability zone mentioned in request is the same on the current compute host.

The `ImagePropertiesFilter` filters hosts based on the architecture, hypervisor type, and virtual machine mode specified in the instance. E.g., an instance might require a host that supports the arm architecture on a qemu compute host. The `ImagePropertiesFilter` will only pass hosts that can satisfy this request. These instance properties are populated from properties define on the instance's image. E.g. an image can be decorated with these properties using `glance image-update img-uuid --property architecture=arm --property hypervisor_type=qemu` Only hosts that satisfy these requirements will pass the `ImagePropertiesFilter`.

`ComputeCapabilitiesFilter` checks if the host satisfies any `extra_specs` specified on the instance type. The `extra_specs` can contain key/value pairs. The key for the filter is either non-scope format (i.e. no : contained), or scope format in capabilities scope (i.e. `capabilities:xxx:yyy`). One example of capabilities scope is `capabilities:cpu_info:features`, which will match host's cpu features capabilities. The `ComputeCapabilitiesFilter`

will only pass hosts whose capabilities satisfy the requested specifications. All hosts are passed if no `extra_specs` are specified.

ComputeFilter is quite simple and passes any host whose Compute service is enabled and operational.

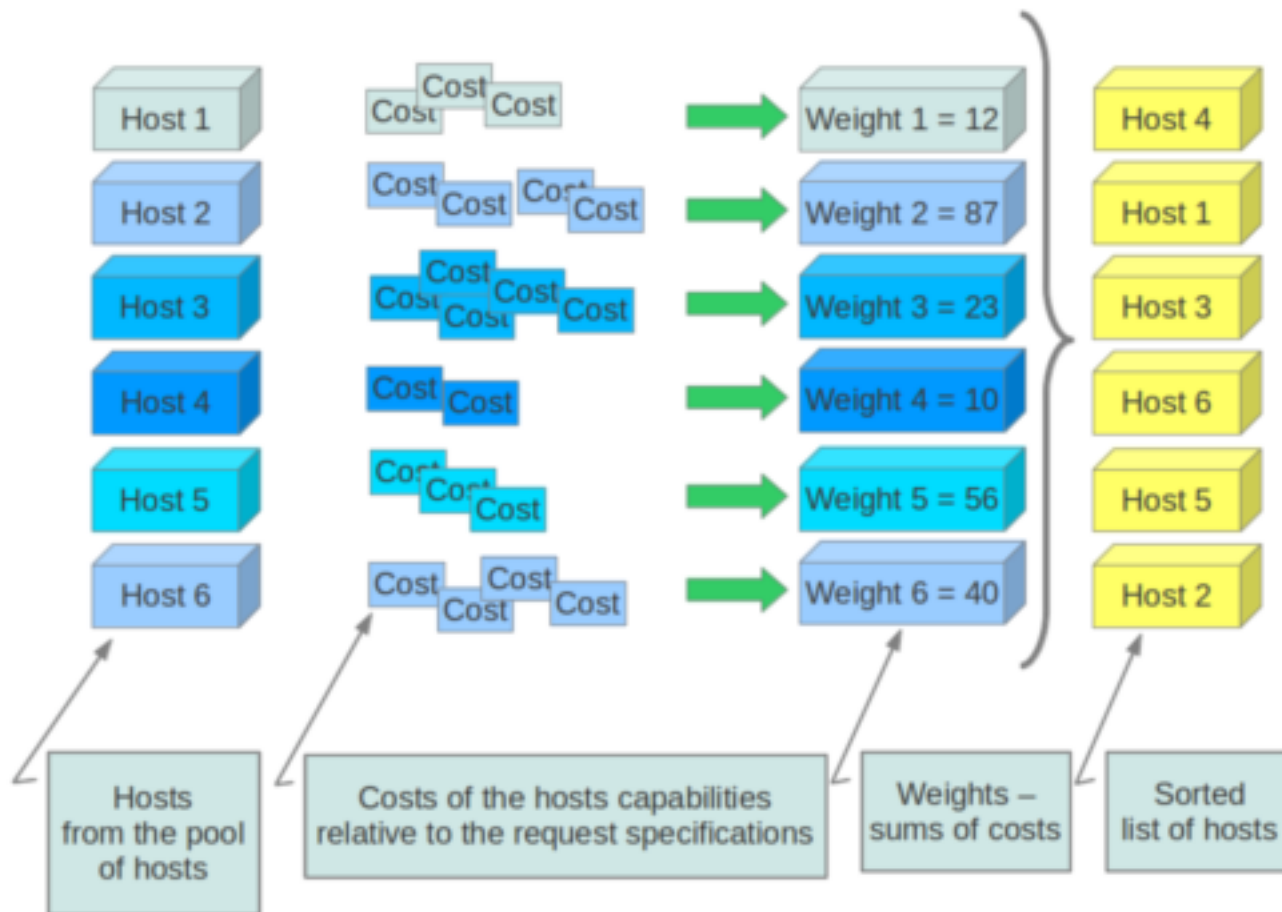
Now we are going to IsolatedHostsFilter. There can be some special hosts reserved for specific images. These hosts are called isolated. So the images to run on the isolated hosts are also called isolated. This Scheduler checks if `image_isolated` flag named in instance specifications is the same that the host has.

## Weights

Filter Scheduler uses so-called weights during its work.

The Filter Scheduler weights hosts based on the config option `scheduler_weight_classes`, this defaults to `nova.scheduler.weights.all_weighters`, which selects the only weigher available – the `RamWeigher`. Hosts are then weighted and sorted with the largest weight winning.

Filter Scheduler finds local list of acceptable hosts by repeated filtering and weighing. Each time it chooses a host, it virtually consumes resources on it, so subsequent selections can adjust accordingly. It is useful if the customer asks for the same large amount of instances, because weight is computed for each instance requested.

**Figure 10.3. Weights**



In the end Filter Scheduler sorts selected hosts by their weight and provisions instances on them.

## Review Associate VM Provisioning Indepth

The request flow for provisioning an instance goes like this:

1. The dashboard or CLI gets the user credentials and authenticates with the Identity Service via REST API.

The Identity Service authenticates the user with the user credentials, and then generates and sends back an auth-token which will be used for sending the request to other components through REST-call.

2. The dashboard or CLI converts the new instance request specified in **launch instance** or **nova-boot** form to a REST API request and sends it to nova-api.
3. nova-api receives the request and sends a request to the Identity Service for validation of the auth-token and access permission.

The Identity Service validates the token and sends updated authentication headers with roles and permissions.

4. nova-api checks for conflicts with nova-database.

nova-api creates initial database entry for a new instance.

5. nova-api sends the rpc.call request to nova-scheduler expecting to get updated instance entry with host ID specified.
6. nova-scheduler picks up the request from the queue.
7. nova-scheduler interacts with nova-database to find an appropriate host via filtering and weighing.  
nova-scheduler returns the updated instance entry with the appropriate host ID after filtering and weighing.

nova-scheduler sends the rpc.cast request to nova-compute for launching an instance on the appropriate host.

8. nova-compute picks up the request from the queue.

9. nova-compute sends the rpc.call request to nova-conductor to fetch the instance information such as host ID and flavor (RAM, CPU, Disk).

10. nova-conductor picks up the request from the queue.

11. nova-conductor interacts with nova-database.

nova-conductor returns the instance information.

nova-compute picks up the instance information from the queue.

12. nova-compute performs the REST call by passing the auth-token to glance-api. Then, nova-compute uses the Image ID to retrieve the Image URI from the Image Service, and loads the image from the image storage.

13. glance-api validates the auth-token with keystone.

nova-compute gets the image metadata.

14. nova-compute performs the REST-call by passing the auth-token to Network API to allocate and configure the network so that the instance gets the IP address.

15. neutron-server validates the auth-token with keystone.

nova-compute retrieves the network info.

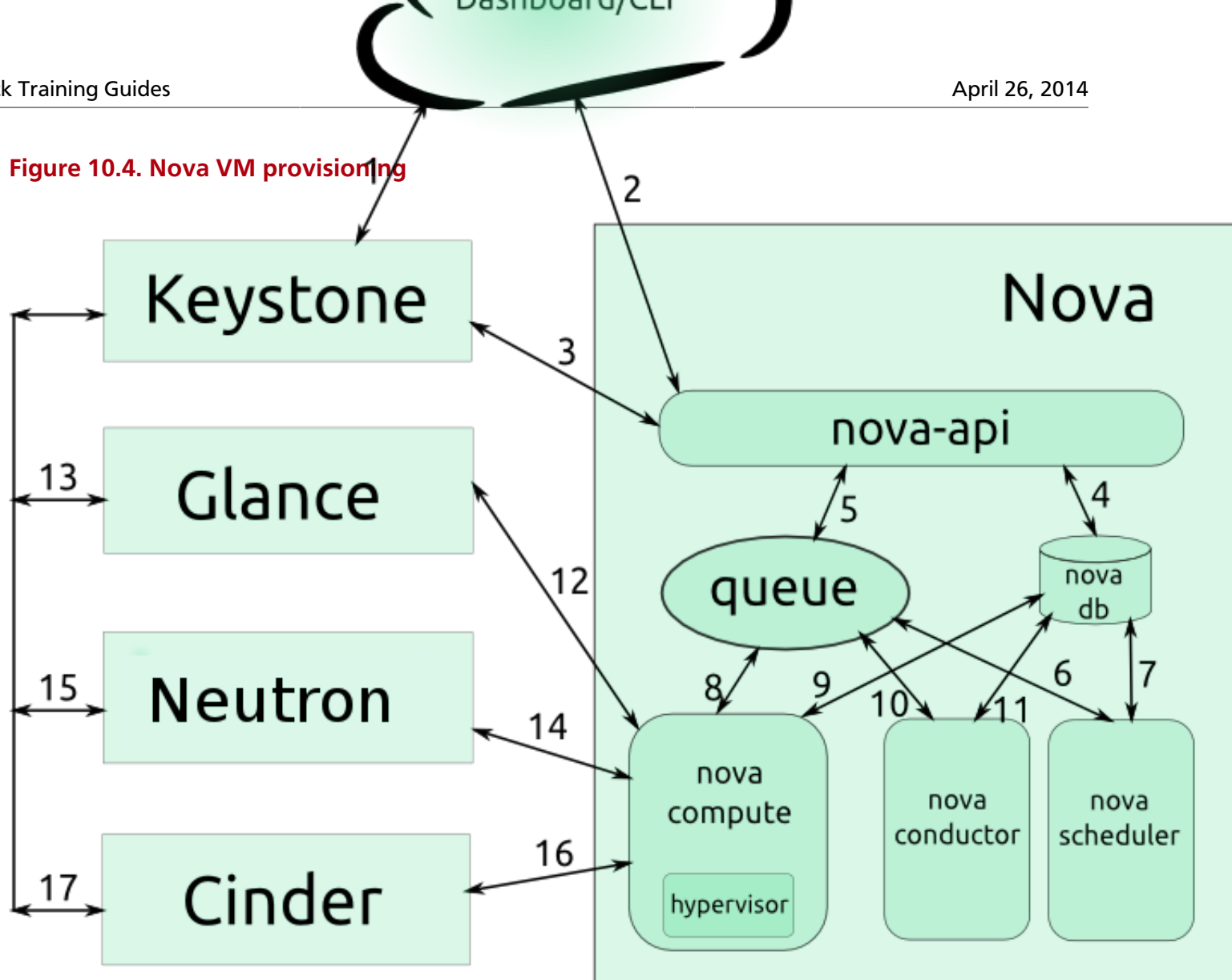
16. nova-compute performs the REST call by passing the auth-token to Volume API to attach volumes to the instance.

17.cinder - api validates the auth-token with keystone.

nova - compute retrieves the block storage info.

18.nova - compute generates data for the hypervisor driver and executes the request on the hypervisor (via libvirt or API).

Figure 10.4. Nova VM provisioning



## Review Associate OpenStack Block Storage

### Block Storage and OpenStack Compute

OpenStack provides two classes of block storage, "ephemeral" storage and persistent "volumes". Ephemeral storage exists only for the life of an instance, it will persist across reboots of the guest operating system but when the instance is deleted so is the associated storage. All instances have some ephemeral storage. Volumes are persistent virtualized block devices independent of any particular instance. Volumes may be attached to a single instance at a time, but may be detached or reattached to a different instance while retaining all data, much like a USB drive.

### Ephemeral Storage

Ephemeral storage is associated with a single unique instance. Its size is defined by the flavor of the instance.

Data on ephemeral storage ceases to exist when the instance it is associated with is terminated. Rebooting the VM or restarting the host server, however, will not destroy ephemeral data. In the typical use case an instance's root filesystem is stored on ephemeral storage. This is often an unpleasant surprise for people unfamiliar with the cloud model of computing.

In addition to the ephemeral root volume all flavors except the smallest, m1.tiny, provide an additional ephemeral block device varying from 20G for the m1.small through 160G for the m1.xlarge by default - these sizes are configurable. This is presented as a raw block device with no partition table or filesystem. Cloud aware operating system images may discover, format, and mount this device. For example the cloud-init package included in Ubuntu's stock cloud images will format this space as an ext3 filesystem and mount it on /mnt. It is important to note this a feature of the guest operating system. OpenStack only provisions the raw storage.

### Volume Storage

Volume storage is independent of any particular instance and is persistent. Volumes are user created and within quota and availability limits may be of any arbitrary size.

When first created volumes are raw block devices with no partition table and no filesystem. They must be attached to an instance to be partitioned and/or formatted. Once this is done they may be used much like an external disk drive. Volumes may attached to only one instance at a time, but may be detached and reattached to either the same or different instances.

It is possible to configure a volume so that it is bootable and provides a persistent virtual instance similar to traditional non-cloud based virtualization systems. In this use case the resulting instance may still have ephemeral storage depending on the flavor selected, but the root filesystem (and possibly others) will be on the persistent volume and thus state will be maintained even if the instance is shutdown. Details of this configuration are discussed in the [OpenStack End User Guide](#).

Volumes do not provide concurrent access from multiple instances. For that you need either a traditional network filesystem like NFS or CIFS or a cluster filesystem such as GlusterFS. These may be built within an OpenStack cluster or provisioned outside of it, but are not features provided by the OpenStack software.

The OpenStack Block Storage service works via the interaction of a series of daemon processes named cinder-\* that reside persistently on the host machine or machines. The binaries can all be run from a single node, or spread across multiple nodes. They can also be run on the same node as other OpenStack services.

**The current services available in OpenStack Block Storage are:**

- **cinder-api** - The cinder-api service is a WSGI app that authenticates and routes requests throughout the Block Storage system. It supports the OpenStack API's only, although there is a translation that can be done via Nova's EC2 interface which calls in to the cinderclient.
- **cinder-scheduler** - The cinder-scheduler is responsible for scheduling/routing requests to the appropriate volume service. As of Grizzly; depending upon your configuration this may be simple round-robin scheduling to the running volume services, or it can be more sophisticated through the use of the Filter Scheduler. The Filter Scheduler is the default in Grizzly and enables filter on things like Capacity, Availability Zone, Volume Types and Capabilities as well as custom filters.
- **cinder-volume** - The cinder-volume service is responsible for managing Block Storage devices, specifically the back-end devices themselves.

- **cinder-backup** - The cinder-backup service provides a means to back up a Cinder Volume to OpenStack Object Store (SWIFT).

### **Introduction to OpenStack Block Storage**

OpenStack Block Storage provides persistent High Performance Block Storage resources that can be consumed by OpenStack Compute instances. This includes secondary attached storage similar to Amazon's Elastic Block Storage (EBS). In addition images can be written to a Block Storage device and specified for OpenStack Compute to use a bootable persistent instance.

There are some differences from Amazon's EBS that one should be aware of. OpenStack Block Storage is not a shared storage solution like NFS, but currently is designed so that the device is attached and in use by a single instance at a time.

### **Backend Storage Devices**

OpenStack Block Storage requires some form of back-end storage that the service is built on. The default implementation is to use LVM on a local Volume Group named "cinder-volumes". In addition to the base driver implementation, OpenStack Block Storage also provides the means to add support for other storage devices to be utilized such as external Raid Arrays or other Storage appliances.

### **Users and Tenants (Projects)**

The OpenStack Block Storage system is designed to be used by many different cloud computing consumers or customers, basically tenants on a shared system, using role-based access assignments. Roles control the actions that a user is allowed to perform. In the default configuration, most actions do not require a particular role, but this is configurable by the system administrator editing the appropriate policy.json file that maintains the rules. A user's access to particular volumes is limited by tenant, but the username and password are assigned per user. Key pairs granting access to a volume are enabled per user, but quotas to control resource consumption across available hardware resources are per tenant.

**For tenants, quota controls are available to limit the:**

- Number of volumes which may be created
- Number of snapshots which may be created
- Total number of Giga Bytes allowed per tenant (shared between snapshots and volumes)

### **Volumes Snapshots and Backups**

This introduction provides a high level overview of the two basic resources offered by the OpenStack Block Storage service. The first is Volumes and the second is Snapshots which are derived from Volumes.

#### **Volumes**

Volumes are allocated block storage resources that can be attached to instances as secondary storage or they can be used as the root store to boot instances. Volumes are persistent R/W Block Storage devices most commonly attached to the compute node via iSCSI.

#### **Snapshots**

A Snapshot in OpenStack Block Storage is a read-only point in time copy of a Volume. The Snapshot can be created from a Volume that is currently in use (via the use of '-force True') or in an available state. The Snapshot can then be used to create a new volume via create from snapshot.

#### **Backups**

A Backup is an archived copy of a Volume currently stored in Object Storage (Swift).

### **Managing Volumes**

Cinder is the OpenStack service that allows you to give extra block level storage to your OpenStack Compute instances. You may recognize this as a similar offering from Amazon EC2 known as Elastic Block Storage (EBS). The default Cinder implementation is an iSCSI solution that employs the use of Logical Volume Manager (LVM) for Linux. Note that a volume may only be attached to one instance at a time. This is not a 'shared storage' solution like a SAN or NFS on which multiple servers can attach to. It's also important to note that



Cinder also includes a number of drivers to allow you to use a number of other vendor's back-end storage devices in addition to or instead of the base LVM implementation.

Here is brief walk-through of a simple create/attach sequence, keep in mind this requires proper configuration of both OpenStack Compute via `cinder.conf` and OpenStack Block Storage via `cinder.conf`.

1. The volume is created via `cinder create`; which creates an LV into the volume group (VG) "cinder-volumes"
2. The volume is attached to an instance via `nova volume-attach`; which creates a unique iSCSI IQN that will be exposed to the compute node
3. The compute node which run the concerned instance has now an active iSCSI session; and a new local storage (usually a `/dev/sdX` disk)
4. `libvirt` uses that local storage as a storage for the instance; the instance get a new disk (usually a `/dev/vdX` disk)

### **Block Storage Capabilities**

- OpenStack provides persistent block level storage devices for use with OpenStack compute instances.
- The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs.
- In addition to using simple Linux server storage, it has unified storage support for numerous storage platforms including Ceph, NetApp, Nexenta, SolidFire, and Zadara.
- Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage.
- Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

## **Review Associate Administration Tasks**

# 11. Compute Node Lab

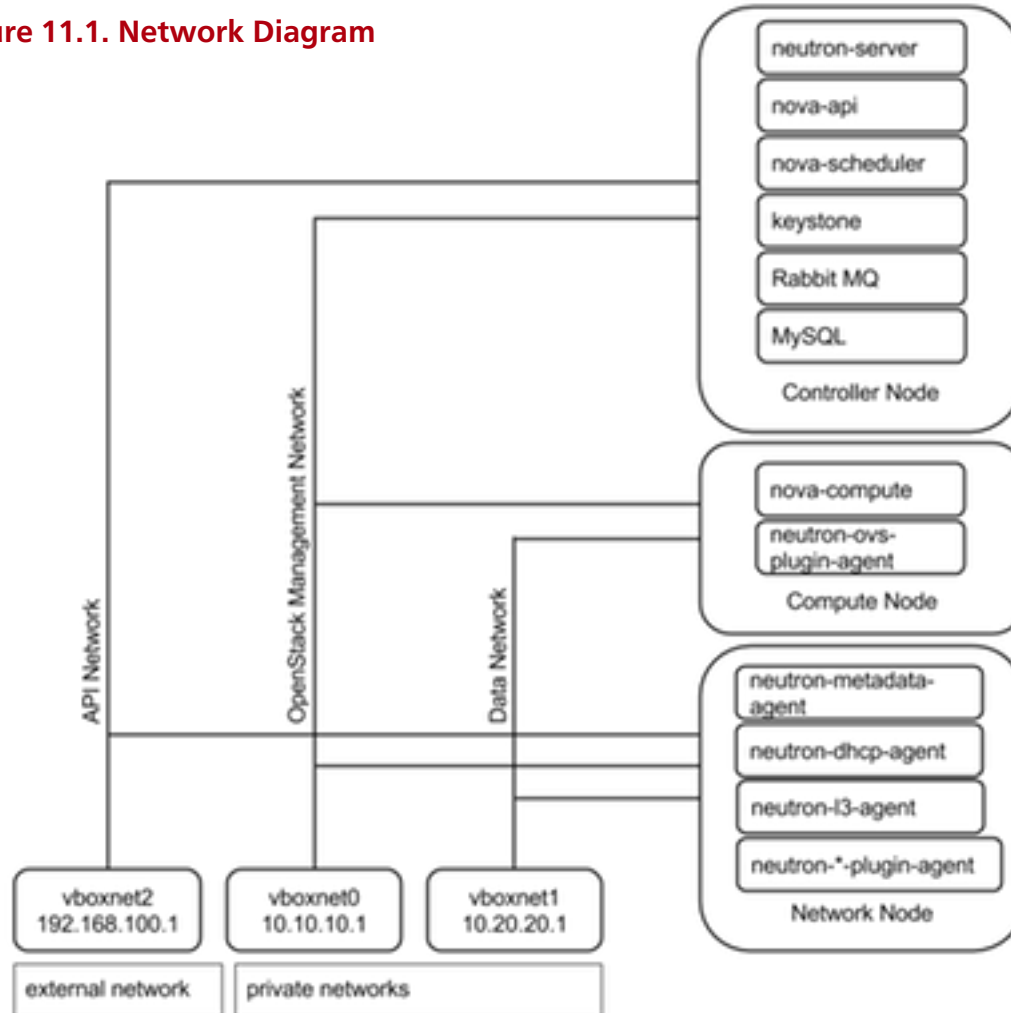
## Table of Contents

|                                                   |     |
|---------------------------------------------------|-----|
| Days 5 to 6, 13:30 to 14:45, 15:00 to 17:00 ..... | 195 |
| Compute Node Lab .....                            | 195 |

## Days 5 to 6, 13:30 to 14:45, 15:00 to 17:00

## Compute Node Lab

### 1. Network Diagram :

**Figure 11.1. Network Diagram**

Publicly editable image source at [https://docs.google.com/drawings/d/1GX3FXmkz3c\\_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing](https://docs.google.com/drawings/d/1GX3FXmkz3c_tUDpZXUVMpylxicWuHs5fNsHvYNjwNNk/edit?usp=sharing)

**Vboxnet0, Vboxnet1, Vboxnet2** - are virtual networks setup up by virtual box with your host machine. This is the way your host can communicate with the virtual machines. These networks are in turn used by virtual box VM's for OpenStack networks, so that OpenStack's services can communicate with each other.

### Compute Node

Start your Controller Node (the one you setup in the previous section).

#### Preparing Ubuntu 12.04

- After you install Ubuntu Server, go in sudo mode

```
$ sudo su
```

- Add Havana repositories:

```
# apt-get install ubuntu-cloud-keyring python-software-properties software-properties-common  
python-keyring
```

```
# echo deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/icehouse main  
>> /etc/apt/sources.list.d/icehouse.list
```

- Update your system:

```
# apt-get update
```

```
# apt-get upgrade
```

```
# apt-get dist-update
```

- Install NTP and other services:

```
# apt-get install ntp vlan bridge-utils
```

- Configure NTP Server to Controller Node:

```
# sed -i 's/server 0.ubuntu.pool.ntp.org/#server0.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

```
# sed -i 's/server 1.ubuntu.pool.ntp.org/#server1.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

```
# sed -i 's/server 2.ubuntu.pool.ntp.org/#server2.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

```
# sed -i 's/server 3.ubuntu.pool.ntp.org/#server3.ubuntu.pool.ntp.org/g' /etc/ntp.conf
```

- Enable IP Forwarding by adding the following to `/etc/sysctl.conf`

```
net.ipv4.ip_forward=1  
net.ipv4.conf.all.rp_filter=0  
net.ipv4.conf.default.rp_filter=0
```

- Run the following commands:

```
# sysctl net.ipv4.ip_forward=1
```

```
# sysctl net.ipv4.conf.all.rp_filter=0
```

```
# sysctl net.ipv4.conf.default.rp_filter=0
```

```
# sysctl -p
```

## KVM

- Install KVM:

```
# apt-get install -y kvm libvirt-bin pm-utils
```

- Edit `/etc/libvirt/qemu.conf`

```
cgroup_device_acl = [  
    "/dev/null", "/dev/full", "/dev/zero",  
    "/dev/random", "/dev/urandom",  
    "/dev/ptmx", "/dev/kvm", "/dev/kqemu",  
    "/dev/rtc", "/dev/hpet", "/dev/net/tun"  
]
```

- Delete Default Virtual Bridge

```
# virsh net-destroy default
```

```
# virsh net-undefine default
```

- To Enable Live Migration Edit /etc/libvirt/libvirtd.conf

```
listen_tls = 0  
listen_tcp = 1  
auth_tcp = "none"
```

- Edit /etc/init/libvirt-bin.conf

```
env libvirtd_opts="-d -l"
```

- Edit /etc/default/libvirt-bin

```
libvirtd_opts="-d -l"
```

- Restart libvirt

```
# service dbus restart
```

```
# service libvirt-bin restart
```

## Neutron and OVS

- Install Open vSwitch

```
# apt-get install -y openvswitch-switch openvswitch-datapath-dkms
```

- Create bridges:

```
# ovs-vsctl add-br br-int
```

- Neutron

Install the Neutron Open vSwitch agent:

```
# apt-get -y install neutron-plugin-openvswitch-agent
```

- Edit /etc/neutron/plugins/openvswitch/ovs\_neutron\_plugin.ini

```
#Under the database section
[database]
connection = mysql://neutronUser:neutronPass@192.168.100.51/neutron
#Under the OVS section
[ovs]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
integration_bridge = br-int
tunnel_bridge = br-tun
local_ip = 10.10.10.53
enable_tunneling = True
tunnel_type=gre
[agent]
tunnel_types = gre
#Firewall driver for realizing quantum security group function
[SECURITYGROUP]
firewall_driver =
neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
```

- Edit /etc/neutron/neutron.conf



```
rabbit_host = 192.168.100.51
#And update the keystone_auth token section
[keystone_auth token]
auth_host = 192.168.100.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = quantum
admin_password = service_pass
signing_dir = /var/lib/quantum/keystone-signing
[database]
connection = mysql://neutronUser:neutronPass@192.168.100.51/neutron
```

- Restart all the services:

```
# service neutron-plugin-openvswitch-agent restart
```

## Nova

- Install Nova

```
# apt-get install nova-compute-kvm python-guestfs
```

```
# chmod 0644 /boot/vmlinuz*
```

- Edit /etc/nova/api-paste.ini

```
[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory
auth_host = 192.168.100.51
auth_port = 35357
auth_protocol = http
admin_tenant_name = service
admin_user = nova
admin_password = service_pass
signing_dirname = /tmp/keystone-signing-nova
# Workaround for https://bugs.launchpad.net/nova/+bug/1154809
auth_version = v2.0
```

- Edit /etc/nova/nova-compute.conf

```
[DEFAULT]
libvirt_type=qemu
libvirt_ovs_bridge=br-int
libvirt_vif_type=ethernet
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
libvirt_use_virtio_for_bridges=True
```

- Edit /etc/nova/nova.conf

```
[DEFAULT]
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/run/lock/nova
verbose=True
api_paste_config=/etc/nova/api-paste.ini
compute_scheduler_driver=nova.scheduler.simple.SimpleScheduler
rabbit_host=192.168.100.51
nova_url=http://192.168.100.51:8774/v1.1/
sql_connection=mysql://novaUser:novaPass@192.168.100.51/nova
root_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf
# Auth
```

```
use_deprecated_auth=false
auth_strategy=keystone
# Imaging service
glance_api_servers=192.168.100.51:9292
image_service=nova.image.glance.GlanceImageService
# Vnc configuration
novnc_enabled=true
novncproxy_base_url=http://192.168.100.51:6080/vnc_auto.html
novncproxy_port=6080
vncserver_proxycient_address=10.10.10.53
vncserver_listen=0.0.0.0
# Network settings
network_api_class=nova.network.neutronv2.api.API
neutron_url=http://192.168.100.51:9696
neutron_auth_strategy=keystone
neutron_admin_tenant_name=service
neutron_admin_username=neutron
neutron_admin_password=service_pass
neutron_admin_auth_url=http://192.168.100.51:35357/v2.0
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
linuxnet_interface_driver=nova.network.linux_net.LinuxOVSInterfaceDriver
#If you want Neutron + Nova Security groups
firewall_driver=nova.virt.firewall.NoopFirewallDriver
security_group_api=neutron
#If you want Nova Security groups only, comment the two lines above and uncomment line -1-.
#-1-firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver
#Metadata
service_neutron_metadata_proxy = True
neutron_metadata_proxy_shared_secret = helloOpenStack
# Compute #
compute_driver=libvirt.LibvirtDriver
# Cinder #
volume_api_class=nova.volume.cinder.API
osapi_volume_listen_port=5900
cinder_catalog_info=volume:cinder:internalURL
```

- Restart nova services

```
# cd /etc/init.d/; for i in $( ls nova-* ); do service $i restart; done
```

- List nova services (Check for the Smiley Faces to know if the services are running):

```
# nova-manage service list
```

## 12. Compute Node Quiz

### Table of Contents

|                                   |     |
|-----------------------------------|-----|
| Days 5 to 6, 16:40 to 17:00 ..... | 205 |
|-----------------------------------|-----|

### Days 5 to 6, 16:40 to 17:00



## 13. Object Storage Node

### Table of Contents

|                                                       |     |
|-------------------------------------------------------|-----|
| Day 9, 09:00 to 11:00, 11:15 to 12:30 .....           | 207 |
| Review Associate Introduction to Object Storage ..... | 207 |
| Review Associate Features and Benefits .....          | 208 |
| Review Associate Administration Tasks .....           | 209 |
| Object Storage Capabilities .....                     | 209 |
| Object Storage Building Blocks .....                  | 211 |
| Swift Ring Builder .....                              | 222 |
| More Swift Concepts .....                             | 225 |
| Swift Cluster Architecture .....                      | 229 |
| Swift Account Reaper .....                            | 233 |
| Swift Replication .....                               | 234 |

### Day 9, 09:00 to 11:00, 11:15 to 12:30

### Review Associate Introduction to Object Storage

OpenStack Object Storage (code-named Swift) is open source software for creating redundant, scalable data storage using clusters of standardized servers to store petabytes of accessible data. It is a long-term storage system for large amounts of static data that can be retrieved, leveraged, and updated. Object Storage uses a distributed architecture with no central point of control, providing greater scalability, redundancy and

permanence. Objects are written to multiple hardware devices, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally by adding new nodes. Should a node fail, OpenStack works to replicate its content from other active nodes. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.

Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. Block Storage allows block devices to be exposed and connected to compute instances for expanded storage, better performance and integration with enterprise storage platforms, such as NetApp, Nexenta and SolidFire.

## Review Associate Features and Benefits

| Features                                                                                                           | Benefits                                                                                                                                                             |
|--------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>Leverages commodity hardware</b>                                                                                | No lock-in, lower price/GB                                                                                                                                           |
| <b>HDD/node failure agnostic</b>                                                                                   | Self healingReliability, data redundancy protecting from failures                                                                                                    |
| <b>Unlimited storage</b>                                                                                           | Huge & flat namespace, highly scalable read/write accessAbility to serve content directly from storage system                                                        |
| <b>Multi-dimensional scalability</b> (scale out architecture)Scale vertically and horizontally-distributed storage | Backup and archive large amounts of data with linear performance                                                                                                     |
| <b>Account/Container/Object structure</b> No nesting, not a traditional file system                                | Optimized for scaleScales to multiple petabytes, billions of objects                                                                                                 |
| <b>Built-in replication</b> 3x+ <b>data redundancy</b> compared to 2x on RAID                                      | Configurable number of accounts, container and object copies for high availability                                                                                   |
| <b>Easily add capacity</b> unlike RAID resize                                                                      | Elastic data scaling with ease                                                                                                                                       |
| <b>No central database</b>                                                                                         | Higher performance, no bottlenecks                                                                                                                                   |
| <b>RAID not required</b>                                                                                           | Handle lots of small, random reads and writes efficiently                                                                                                            |
| <b>Built-in management utilities</b>                                                                               | Account Management: Create, add, verify, delete usersContainer Management: Upload, download, verifyMonitoring: Capacity, host, network, log trawling, cluster health |



|                                           |                                                                                |
|-------------------------------------------|--------------------------------------------------------------------------------|
| Drive auditing                            | Detect drive failures preempting data corruption                               |
| Expiring objects                          | Users can set an expiration time or a TTL on an object to control access       |
| Direct object access                      | Enable direct browser access to content, such as for a control panel           |
| Realtime visibility into client requests  | Know what users are requesting                                                 |
| Supports S3 API                           | Utilize tools that were designed for the popular S3 API                        |
| Restrict containers per account           | Limit access to control usage by user                                          |
| Support for NetApp, Nexenta, SolidFire    | Unified support for block volumes using a variety of storage systems           |
| Snapshot and backup API for block volumes | Data protection and recovery for VM data                                       |
| Standalone volume API available           | Separate endpoint and API for integration with other compute systems           |
| Integration with Compute                  | Fully integrated to Compute for attaching block volumes and reporting on usage |

## Review Associate Administration Tasks

### Object Storage Capabilities

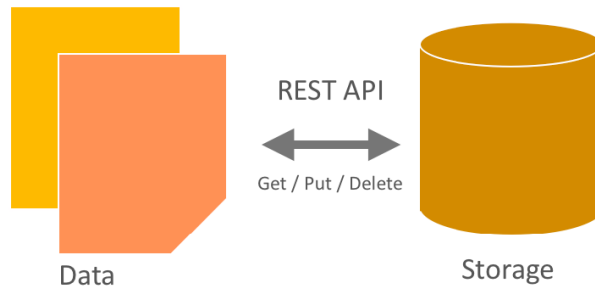
- OpenStack provides redundant, scalable object storage using clusters of standardized servers capable of storing petabytes of data
- Object Storage is not a traditional file system, but rather a distributed storage system for static data such as virtual machine images, photo storage, email storage, backups and archives. Having no central "brain" or master point of control provides greater scalability, redundancy and durability.
- Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster.
- Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack

uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.

### **Swift Characteristics**

The key characteristics of Swift include:

- All objects stored in Swift have a URL
- All objects stored are replicated 3x in as-unique-as-possible zones, which can be defined as a group of drives, a node, a rack etc.
- All objects have their own metadata
- Developers interact with the object storage system through a RESTful HTTP API
- Object data can be located anywhere in the cluster
- The cluster scales by adding additional nodes – without sacrificing performance, which allows a more cost-effective linear storage expansion vs. fork-lift upgrades
- Data doesn't have to be migrated to an entirely new storage system
- New nodes can be added to the cluster without downtime
- Failed nodes and disks can be swapped out with no downtime
- Runs on industry-standard hardware, such as Dell, HP, Supermicro etc.

**Figure 13.1. Object Storage(Swift)**

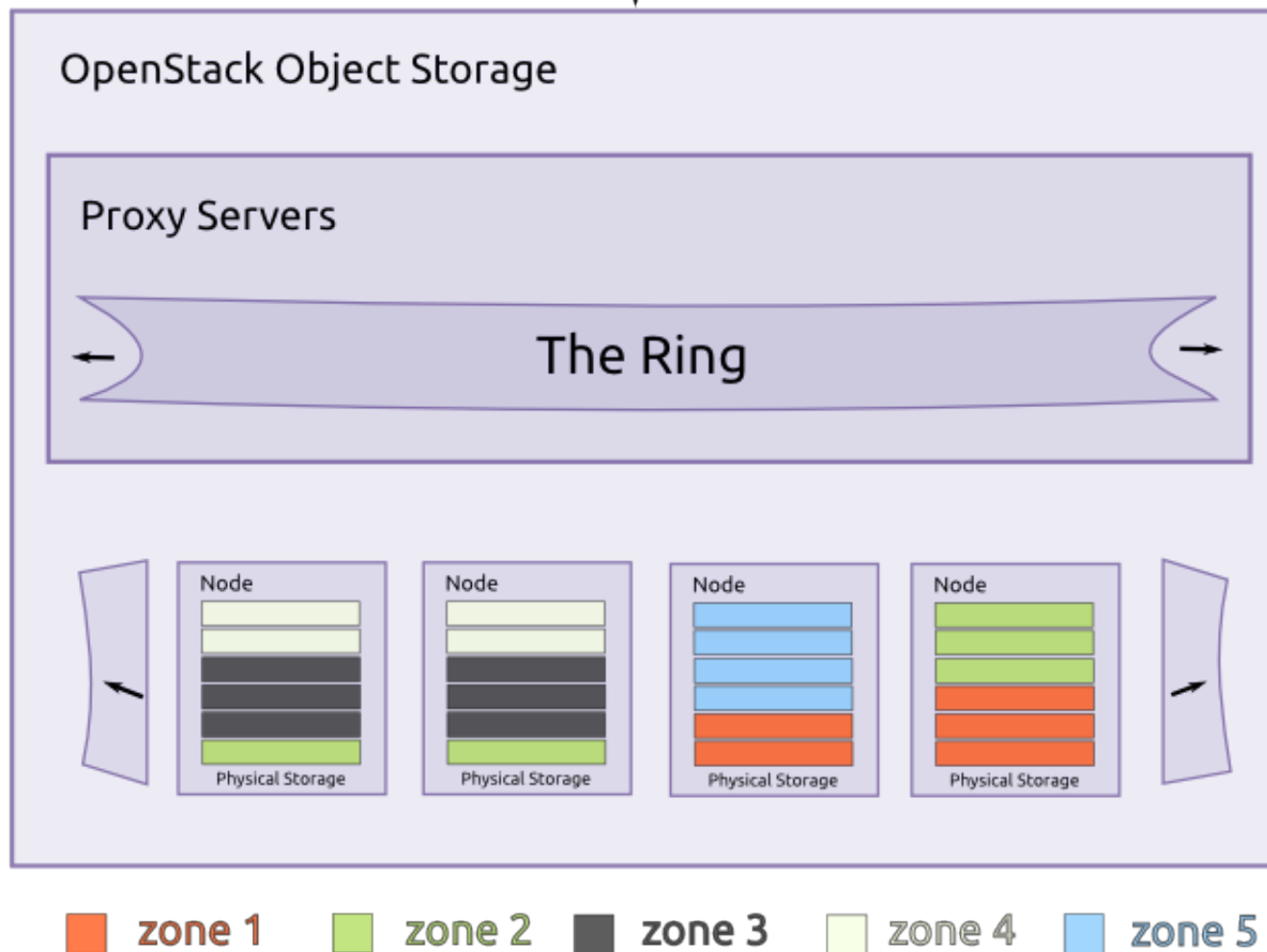
Developers can either write directly to the Swift API or use one of the many client libraries that exist for all popular programming languages, such as Java, Python, Ruby and C#. Amazon S3 and RackSpace Cloud Files users should feel very familiar with Swift. For users who have not used an object storage system before, it will require a different approach and mindset than using a traditional filesystem.

## Object Storage Building Blocks

The components that enable Swift to deliver high availability, high durability and high concurrency are:

- **Proxy Servers:**Handles all incoming API requests.
- **Rings:**Maps logical names of data to locations on particular disks.
- **Zones:**Each Zone isolates data from other Zones. A failure in one Zone doesn't impact the rest of the cluster because data is replicated across the Zones.
- **Accounts & Containers:**Each Account and Container are individual databases that are distributed across the cluster. An Account database contains the list of Containers in that Account. A Container database contains the list of Objects in that Container

- **Objects:**The data itself.
- **Partitions:**A Partition stores Objects, Account databases and Container databases. It's an intermediate 'bucket' that helps manage locations where data lives in the cluster.

**Figure 13.2. Building Blocks**

## Proxy Servers

The Proxy Servers are the public face of Swift and handle all incoming API requests. Once a Proxy Server receives a request, it will determine the storage node based on the URL of the object, such as `https://swift.example.com/v1/account/container/object`. The Proxy Servers also coordinate responses, handle failures and coordinate timestamps.

Proxy servers use a shared-nothing architecture and can be scaled as needed based on projected workloads. A minimum of two Proxy Servers should be deployed for redundancy. Should one proxy server fail, the others will take over.

## The Ring

A ring represents a mapping between the names of entities stored on disk and their physical location. There are separate rings for accounts, containers, and objects. When other components need to perform any operation on an object, container, or account, they need to interact with the appropriate ring to determine its location in the cluster.

The Ring maintains this mapping using zones, devices, partitions, and replicas. Each partition in the ring is replicated, by default, 3 times across the cluster, and the locations for a partition are stored in the mapping maintained by the ring. The ring is also responsible for determining which devices are used for hand off in failure scenarios.

Data can be isolated with the concept of zones in the ring. Each replica of a partition is guaranteed to reside in a different zone. A zone could represent a drive, a server, a cabinet, a switch, or even a data center.

The partitions of the ring are equally divided among all the devices in the OpenStack Object Storage installation. When partitions need to be moved around, such as when a device is added to the cluster, the ring ensures that a minimum number of partitions are moved at a time, and only one replica of a partition is moved at a time.

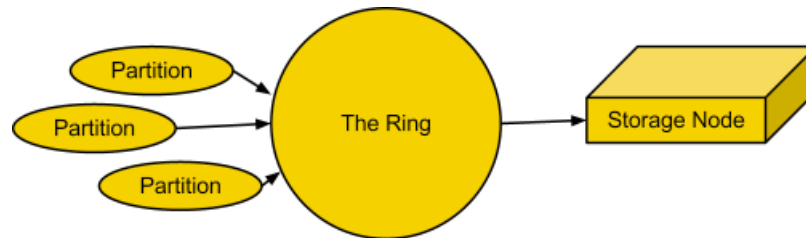
Weights can be used to balance the distribution of partitions on drives across the cluster. This can be useful, for example, when different sized drives are used in a cluster.

The ring is used by the Proxy server and several background processes (like replication).

The Ring maps Partitions to physical locations on disk. When other components need to perform any operation on an object, container, or account, they need to interact with the Ring to determine its location in the cluster.

The Ring maintains this mapping using zones, devices, partitions, and replicas. Each partition in the Ring is replicated three times by default across the cluster, and the locations for a partition are stored in the mapping maintained by the Ring. The Ring is also responsible for determining which devices are used for handoff should a failure occur.

**Figure 13.3. The Lord of the Rings**



The Ring maps partitions to physical locations on disk.

The rings determine where data should reside in the cluster. There is a separate ring for account databases, container databases, and individual objects but each ring works in the same way. These rings are externally managed, in that the server processes themselves do not modify the rings, they are instead given new rings modified by other tools.

The ring uses a configurable number of bits from a path's MD5 hash as a partition index that designates a device. The number of bits kept from the hash is known as the partition power, and 2 to the partition power indicates the partition count. Partitioning the full MD5 hash ring allows other parts of the cluster to work in

batches of items at once which ends up either more efficient or at least less complex than working with each item separately or the entire cluster all at once.

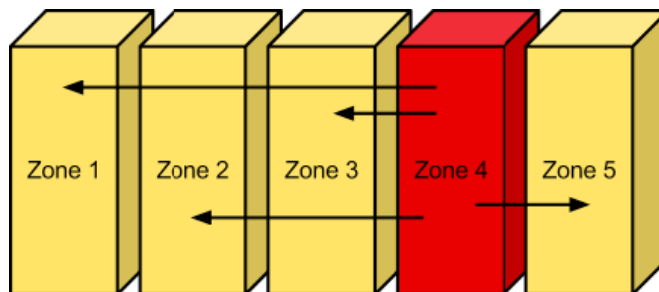
Another configurable value is the replica count, which indicates how many of the partition->device assignments comprise a single ring. For a given partition number, each replica's device will not be in the same zone as any other replica's device. Zones can be used to group devices based on physical locations, power separations, network separations, or any other attribute that would lessen multiple replicas being unavailable at the same time.

### Zones: Failure Boundaries

Swift allows zones to be configured to isolate failure boundaries. Each replica of the data resides in a separate zone, if possible. At the smallest level, a zone could be a single drive or a grouping of a few drives. If there were five object storage servers, then each server would represent its own zone. Larger deployments would have an entire rack (or multiple racks) of object servers, each representing a zone. The goal of zones is to allow the cluster to tolerate significant outages of storage servers without losing all replicas of the data.

As we learned earlier, everything in Swift is stored, by default, three times. Swift will place each replica "as-uniquely-as-possible" to ensure both high availability and high durability. This means that when choosing a replica location, Swift will choose a server in an unused zone before an unused server in a zone that already has a replica of the data.

**Figure 13.4.** image33.png



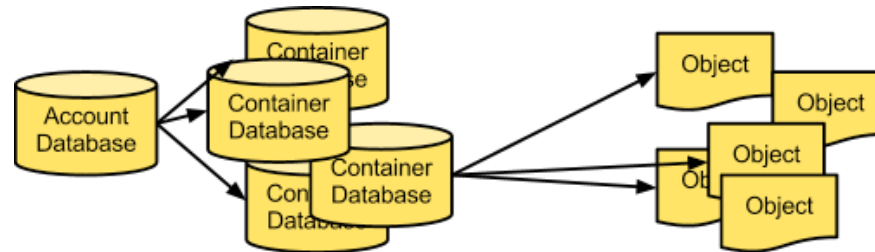


When a disk fails, replica data is automatically distributed to the other zones to ensure there are three copies of the data

### Accounts & Containers

Each account and container is an individual SQLite database that is distributed across the cluster. An account database contains the list of containers in that account. A container database contains the list of objects in that container.

**Figure 13.5. Accounts and Containers**



To keep track of object data location, each account in the system has a database that references all its containers, and each container database references each object

### Partitions

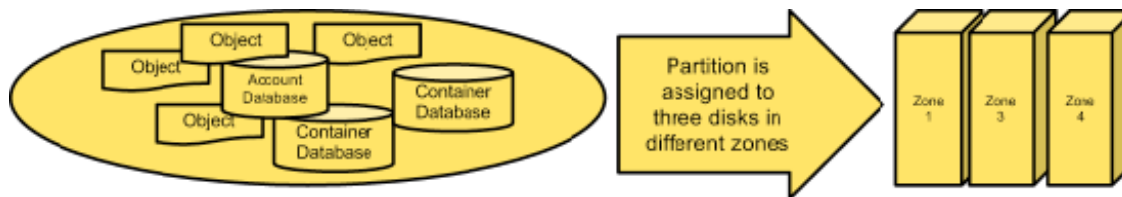
A Partition is a collection of stored data, including Account databases, Container databases, and objects. Partitions are core to the replication system.

Think of a Partition as a bin moving throughout a fulfillment center warehouse. Individual orders get thrown into the bin. The system treats that bin as a cohesive entity as it moves throughout the system. A bin full of things is easier to deal with than lots of little things. It makes for fewer moving parts throughout the system.

The system replicators and object uploads/downloads operate on Partitions. As the system scales up, behavior continues to be predictable as the number of Partitions is a fixed number.

The implementation of a Partition is conceptually simple – a partition is just a directory sitting on a disk with a corresponding hash table of what it contains.

**Figure 13.6. Partitions**



\*Swift partitions contain all data in the system.

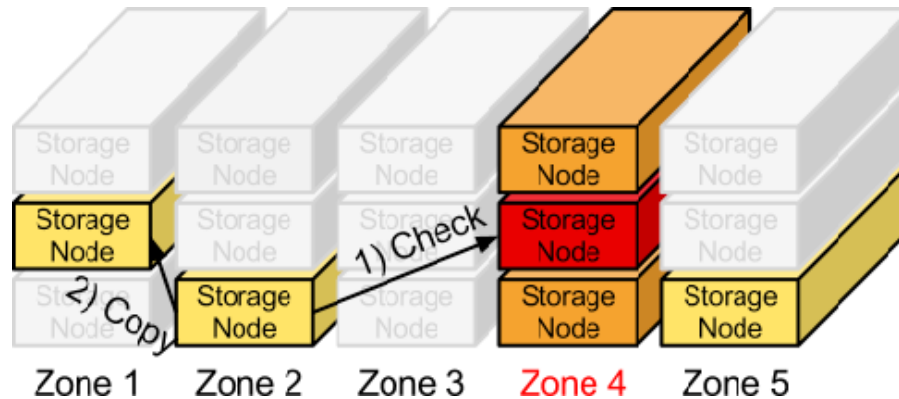
### Replication

In order to ensure that there are three copies of the data everywhere, replicators continuously examine each Partition. For each local Partition, the replicator compares it against the replicated copies in the other Zones to see if there are any differences.

How does the replicator know if replication needs to take place? It does this by examining hashes. A hash file is created for each Partition, which contains hashes of each directory in the Partition. Each of the three hash files is compared. For a given Partition, the hash files for each of the Partition's copies are compared. If the hashes are different, then it is time to replicate and the directory that needs to be replicated is copied over.

This is where the Partitions come in handy. With fewer "things" in the system, larger chunks of data are transferred around (rather than lots of little TCP connections, which is inefficient) and there are a consistent number of hashes to compare.

The cluster has eventually consistent behavior where the newest data wins.

**Figure 13.7. Replication**

\*If a zone goes down, one of the nodes containing a replica notices and proactively copies data to a handoff location.

To describe how these pieces all come together, let's walk through a few scenarios and introduce the components.

### Bird-eye View

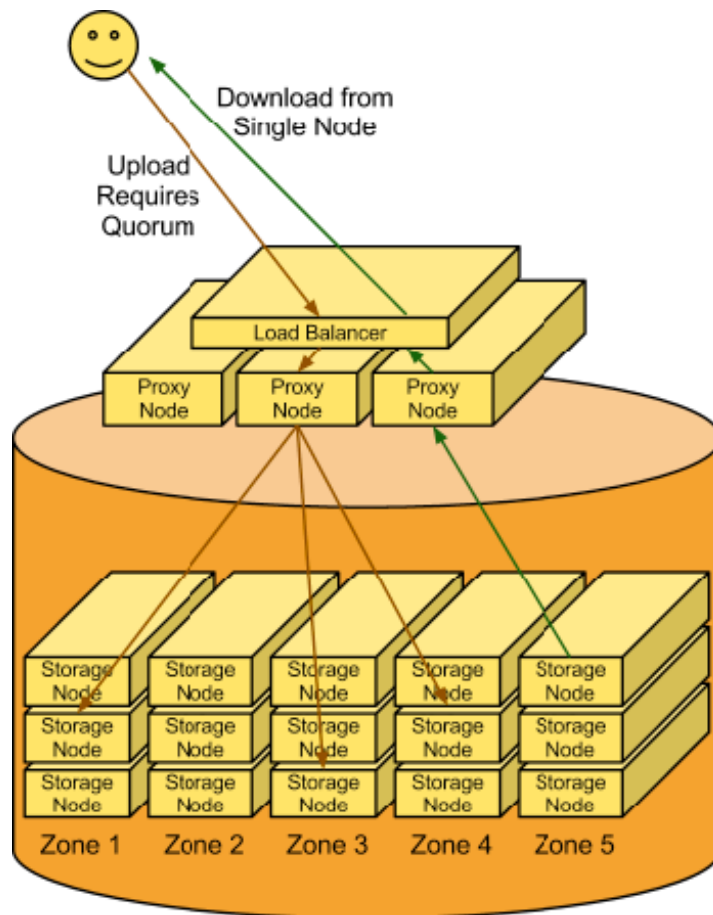
#### Upload

A client uses the REST API to make a HTTP request to PUT an object into an existing Container. The cluster receives the request. First, the system must figure out where the data is going to go. To do this, the Account name, Container name and Object name are all used to determine the Partition where this object should live.

Then a lookup in the Ring figures out which storage nodes contain the Partitions in question.

The data then is sent to each storage node where it is placed in the appropriate Partition. A quorum is required – at least two of the three writes must be successful before the client is notified that the upload was successful.

Next, the Container database is updated asynchronously to reflect that there is a new object in it.

**Figure 13.8. When End-User uses Swift****Download**

A request comes in for an Account/Container/object. Using the same consistent hashing, the Partition name is generated. A lookup in the Ring reveals which storage nodes contain that Partition. A request is made to one of the storage nodes to fetch the object and if that fails, requests are made to the other nodes.

## Swift Ring Builder

The rings are built and managed manually by a utility called the ring-builder. The ring-builder assigns partitions to devices and writes an optimized Python structure to a gzipped, serialized file on disk for shipping out to the servers. The server processes just check the modification time of the file occasionally and reload their in-memory copies of the ring structure as needed. Because of how the ring-builder manages changes to the ring, using a slightly older ring usually just means one of the three replicas for a subset of the partitions will be incorrect, which can be easily worked around.

The ring-builder also keeps its own builder file with the ring information and additional data required to build future rings. It is very important to keep multiple backup copies of these builder files. One option is to copy the builder files out to every server while copying the ring files themselves. Another is to upload the builder files into the cluster itself. Complete loss of a builder file will mean creating a new ring from scratch, nearly all partitions will end up assigned to different devices, and therefore nearly all data stored will have to be replicated to new locations. So, recovery from a builder file loss is possible, but data will definitely be unreachable for an extended time.

### Ring Data Structure

The ring data structure consists of three top level fields: a list of devices in the cluster, a list of lists of device ids indicating partition to device assignments, and an integer indicating the number of bits to shift an MD5 hash to calculate the partition for the hash.

### Partition Assignment List

This is a list of array('H') of devices ids. The outermost list contains an array('H') for each replica. Each array('H') has a length equal to the partition count for the ring. Each integer in the array('H') is an index into the above list of devices. The partition list is known internally to the Ring class as `_replica2part2dev_id`.

So, to create a list of device dictionaries assigned to a partition, the Python code would look like: `devices = [self.devs[part2dev_id[partition]] for part2dev_id in self._replica2part2dev_id]`

That code is a little simplistic, as it does not account for the removal of duplicate devices. If a ring has more replicas than devices, then a partition will have more than one replica on one device; that's simply the pigeonhole principle at work.

`array('H')` is used for memory conservation as there may be millions of partitions.

### Fractional Replicas

A ring is not restricted to having an integer number of replicas. In order to support the gradual changing of replica counts, the ring is able to have a real number of replicas.

When the number of replicas is not an integer, then the last element of `_replica2part2dev_id` will have a length that is less than the partition count for the ring. This means that some partitions will have more replicas than others. For example, if a ring has 3.25 replicas, then 25% of its partitions will have four replicas, while the remaining 75% will have just three.

### Partition Shift Value

The partition shift value is known internally to the Ring class as `_part_shift`. This value is used to shift an MD5 hash to calculate the partition on which the data for that hash should reside. Only the top four bytes of the hash is used in this process. For example, to compute the partition for the path `/account/container/object` the Python code might look like: `partition = unpack_from('>I', md5('/account/container/object').digest())[0] >> self._part_shift`

For a ring generated with `part_power P`, the partition shift value is `32 - P`.

### Building the Ring

The initial building of the ring first calculates the number of partitions that should ideally be assigned to each device based on the device's weight. For example, given a partition power of 20, the ring will have 1,048,576

partitions. If there are 1,000 devices of equal weight they will each desire 1,048.576 partitions. The devices are then sorted by the number of partitions they desire and kept in order throughout the initialization process.

Note: each device is also assigned a random tiebreaker value that is used when two devices desire the same number of partitions. This tiebreaker is not stored on disk anywhere, and so two different rings created with the same parameters will have different partition assignments. For repeatable partition assignments, `RingBuilder.rebalance()` takes an optional seed value that will be used to seed Python's pseudo-random number generator.

Then, the ring builder assigns each replica of each partition to the device that desires the most partitions at that point while keeping it as far away as possible from other replicas. The ring builder prefers to assign a replica to a device in a regions that has no replicas already; should there be no such region available, the ring builder will try to find a device in a different zone; if not possible, it will look on a different server; failing that, it will just look for a device that has no replicas; finally, if all other options are exhausted, the ring builder will assign the replica to the device that has the fewest replicas already assigned. Note that assignment of multiple replicas to one device will only happen if the ring has fewer devices than it has replicas.

When building a new ring based on an old ring, the desired number of partitions each device wants is recalculated. Next the partitions to be reassigned are gathered up. Any removed devices have all their assigned partitions unassigned and added to the gathered list. Any partition replicas that (due to the addition of new devices) can be spread out for better durability are unassigned and added to the gathered list. Any devices that have more partitions than they now desire have random partitions unassigned from them and added to the gathered list. Lastly, the gathered partitions are then reassigned to devices using a similar method as in the initial assignment described above.

Whenever a partition has a replica reassigned, the time of the reassignment is recorded. This is taken into account when gathering partitions to reassign so that no partition is moved twice in a configurable amount of time. This configurable amount of time is known internally to the `RingBuilder` class as `min_part_hours`. This restriction is ignored for replicas of partitions on devices that have been removed, as removing a device only happens on device failure and there's no choice but to make a reassignment.



The above processes don't always perfectly rebalance a ring due to the random nature of gathering partitions for reassignment. To help reach a more balanced ring, the rebalance process is repeated until near perfect (less 1% off) or when the balance doesn't improve by at least 1% (indicating we probably can't get perfect balance due to wildly imbalanced zones or too many partitions recently moved).

## More Swift Concepts

### Containers and Objects

A container is a storage compartment for your data and provides a way for you to organize your data. You can think of a container as a folder in Windows or a directory in UNIX. The primary difference between a container and these other file system concepts is that containers cannot be nested. You can, however, create an unlimited number of containers within your account. Data must be stored in a container so you must have at least one container defined in your account prior to uploading data.

The only restrictions on container names is that they cannot contain a forward slash (/) or an ascii null (%00) and must be less than 257 bytes in length. Please note that the length restriction applies to the name after it has been URL encoded. For example, a container name of Course Docs would be URL encoded as Course %20Docs and therefore be 13 bytes in length rather than the expected 11.

An object is the basic storage entity and any optional metadata that represents the files you store in the OpenStack Object Storage system. When you upload data to OpenStack Object Storage, the data is stored as-is (no compression or encryption) and consists of a location (container), the object's name, and any metadata consisting of key/value pairs. For instance, you may chose to store a backup of your digital photos and organize them into albums. In this case, each object could be tagged with metadata such as Album : Caribbean Cruise or Album : Aspen Ski Trip.

The only restriction on object names is that they must be less than 1024 bytes in length after URL encoding. For example, an object name of C++final(v2).txt should be URL encoded as C%2B%2Bfinal%28v2%29.txt and therefore be 24 bytes in length rather than the expected 16.

The maximum allowable size for a storage object upon upload is 5 GB and the minimum is zero bytes. You can use the built-in large object support and the swift utility to retrieve objects larger than 5 GB.

For metadata, you should not exceed 90 individual key/value pairs for any one object and the total byte length of all key/value pairs should not exceed 4 KB (4096 bytes).

### Language-Specific API Bindings

A set of supported API bindings in several popular languages are available from the Rackspace Cloud Files product, which uses OpenStack Object Storage code for its implementation. These bindings provide a layer of abstraction on top of the base REST API, allowing programmers to work with a container and object model instead of working directly with HTTP requests and responses. These bindings are free (as in beer and as in speech) to download, use, and modify. They are all licensed under the MIT License as described in the COPYING file packaged with each binding. If you do make any improvements to an API, you are encouraged (but not required) to submit those changes back to us.

The API bindings for Rackspace Cloud Files are hosted at <http://github.com/rackspace/rackspace>. Feel free to coordinate your changes through github or, if you prefer, send your changes to [cloudfiles@rackspacecloud.com](mailto:cloudfiles@rackspacecloud.com). Just make sure to indicate which language and version you modified and send a unified diff.

Each binding includes its own documentation (either HTML, PDF, or CHM). They also include code snippets and examples to help you get started. The currently supported API binding for OpenStack Object Storage are:

- PHP (requires 5.x and the modules: cURL, FileInfo, mbstring)
- Python (requires 2.4 or newer)
- Java (requires JRE v1.5 or newer)
- C#/.NET (requires .NET Framework v3.5)
- Ruby (requires 1.8 or newer and mime-tools module)

There are no other supported language-specific bindings at this time. You are welcome to create your own language API bindings and we can help answer any questions during development, host your code if you like, and give you full credit for your work.

### **Proxy Server**

The Proxy Server is responsible for tying together the rest of the OpenStack Object Storage architecture. For each request, it will look up the location of the account, container, or object in the ring (see below) and route the request accordingly. The public API is also exposed through the Proxy Server.

A large number of failures are also handled in the Proxy Server. For example, if a server is unavailable for an object PUT, it will ask the ring for a hand-off server and route there instead.

When objects are streamed to or from an object server, they are streamed directly through the proxy server to or from the user – the proxy server does not spool them.

You can use a proxy server with account management enabled by configuring it in the proxy server configuration file.

### **Object Server**

The Object Server is a very simple blob storage server that can store, retrieve and delete objects stored on local devices. Objects are stored as binary files on the filesystem with metadata stored in the file's extended attributes (xattrs). This requires that the underlying filesystem choice for object servers support xattrs on files. Some filesystems, like ext3, have xattrs turned off by default.

Each object is stored using a path derived from the object name's hash and the operation's timestamp. Last write always wins, and ensures that the latest object version will be served. A deletion is also treated as a version of the file (a 0 byte file ending with ".ts", which stands for tombstone). This ensures that deleted files are replicated correctly and older versions don't magically reappear due to failure scenarios.

### **Container Server**

The Container Server's primary job is to handle listings of objects. It does not know where those objects are, just what objects are in a specific container. The listings are stored as SQLite database files, and replicated across the cluster similar to how objects are. Statistics are also tracked that include the total number of objects, and total storage usage for that container.

### **Account Server**

The Account Server is very similar to the Container Server, excepting that it is responsible for listings of containers rather than objects.

### **Replication**

Replication is designed to keep the system in a consistent state in the face of temporary error conditions like network outages or drive failures.

The replication processes compare local data with each remote copy to ensure they all contain the latest version. Object replication uses a hash list to quickly compare subsections of each partition, and container and account replication use a combination of hashes and shared high water marks.

Replication updates are push based. For object replication, updating is just a matter of rsyncing files to the peer. Account and container replication push missing records over HTTP or rsync whole database files.

The replicator also ensures that data is removed from the system. When an item (object, container, or account) is deleted, a tombstone is set as the latest version of the item. The replicator will see the tombstone and ensure that the item is removed from the entire system.

To separate the cluster-internal replication traffic from client traffic, separate replication servers can be used. These replication servers are based on the standard storage servers, but they listen on the replication IP and only respond to REPLICATE requests. Storage servers can serve REPLICATE requests, so an operator can transition to using a separate replication network with no cluster downtime.

Replication IP and port information is stored in the ring on a per-node basis. These parameters will be used if they are present, but they are not required. If this information does not exist or is empty for a particular node, the node's standard IP and port will be used for replication.

### **Updaters**

There are times when container or account data can not be immediately updated. This usually occurs during failure scenarios or periods of high load. If an update fails, the update is queued locally on the file system, and the updater will process the failed updates. This is where an eventual consistency window will most likely come in to play. For example, suppose a container server is under load and a new object is put in to the system. The object will be immediately available for reads as soon as the proxy server responds to the client with success. However, the container server did not update the object listing, and so the update would be queued for a later update. Container listings, therefore, may not immediately contain the object.

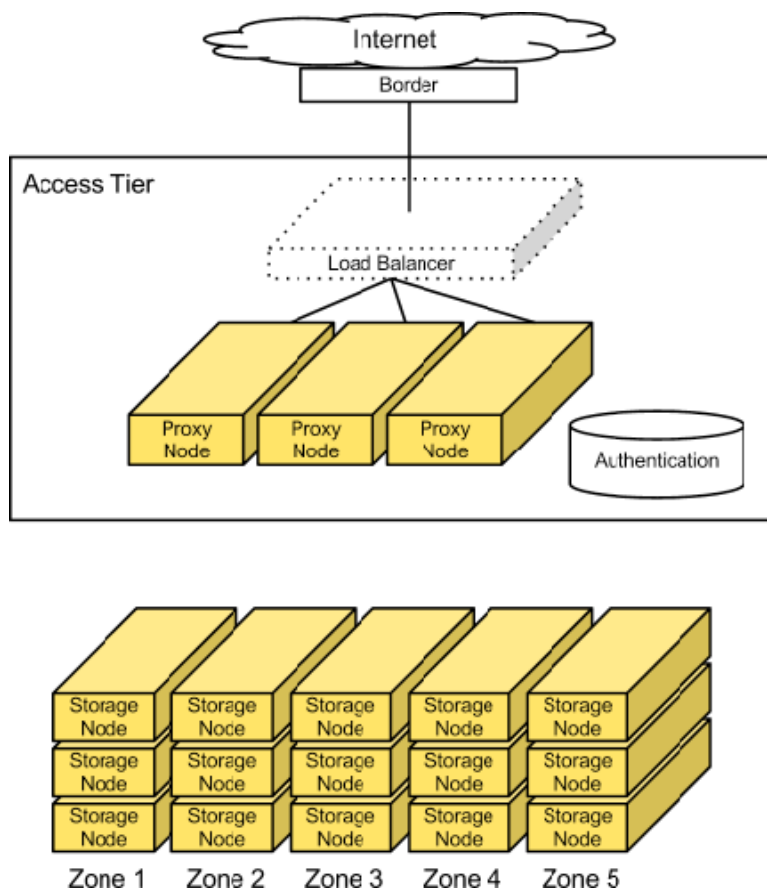
In practice, the consistency window is only as large as the frequency at which the updater runs and may not even be noticed as the proxy server will route listing requests to the first container server which responds. The server under load may not be the one that serves subsequent listing requests – one of the other two replicas may handle the listing.

### **Auditors**

Auditors crawl the local server checking the integrity of the objects, containers, and accounts. If corruption is found (in the case of bit rot, for example), the file is quarantined, and replication will replace the bad file from another replica. If other errors are found they are logged. For example, an object's listing cannot be found on any container server it should be.

## **Swift Cluster Architecture**

### **Access Tier**

**Figure 13.9. Object Storage cluster architecture**

Large-scale deployments segment off an "Access Tier". This tier is the "Grand Central" of the Object Storage system. It fields incoming API requests from clients and moves data in and out of the system. This tier is

composed of front-end load balancers, ssl- terminators, authentication services, and it runs the (distributed) brain of the Object Storage system — the proxy server processes.

Having the access servers in their own tier enables read/write access to be scaled out independently of storage capacity. For example, if the cluster is on the public Internet and requires SSL-termination and has high demand for data access, many access servers can be provisioned. However, if the cluster is on a private network and it is being used primarily for archival purposes, fewer access servers are needed.

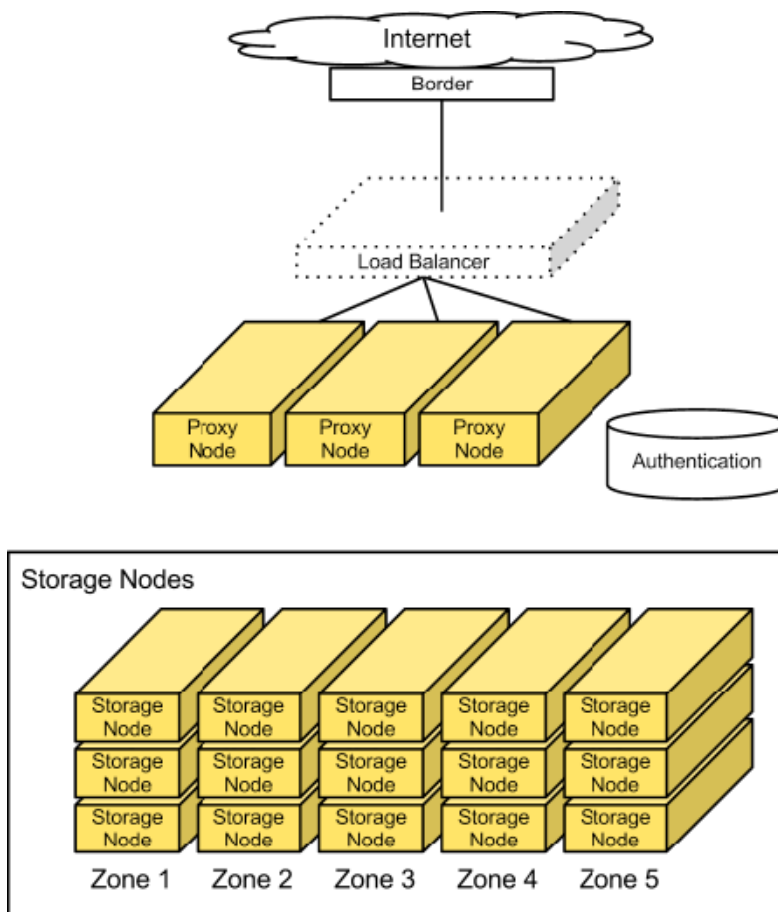
A load balancer can be incorporated into the access tier, because this is an HTTP addressable storage service.

Typically, this tier comprises a collection of 1U servers. These machines use a moderate amount of RAM and are network I/O intensive. It is wise to provision them with two high-throughput (10GbE) interfaces, because these systems field each incoming API request. One interface is used for 'front-end' incoming requests and the other for 'back-end' access to the Object Storage nodes to put and fetch data.

### **Factors to consider**

For most publicly facing deployments as well as private deployments available across a wide-reaching corporate network, SSL is used to encrypt traffic to the client. SSL adds significant processing load to establish sessions between clients; it adds more capacity to the access layer that will need to be provisioned. SSL may not be required for private deployments on trusted networks.

### **Storage Nodes**

**Figure 13.10. Object Storage (Swift)**



The next component is the storage servers themselves. Generally, most configurations should provide each of the five Zones with an equal amount of storage capacity. Storage nodes use a reasonable amount of memory and CPU. Metadata needs to be readily available to quickly return objects. The object stores run services not only to field incoming requests from the Access Tier, but to also run replicators, auditors, and reapers. Object stores can be provisioned with a single gigabit or a 10-gigabit network interface depending on expected workload and desired performance.

Currently, a 2 TB or 3 TB SATA disk delivers good performance for the price. Desktop-grade drives can be used where there are responsive remote hands in the datacenter, and enterprise-grade drives can be used where this is not the case.

### Factors to Consider

Desired I/O performance for single-threaded requests should be kept in mind. This system does not use RAID, so each request for an object is handled by a single disk. Disk performance impacts single-threaded response rates.

To achieve apparent higher throughput, the object storage system is designed with concurrent uploads/downloads in mind. The network I/O capacity (1GbE, bonded 1GbE pair, or 10GbE) should match your desired concurrent throughput needs for reads and writes.

## Swift Account Reaper

The Account Reaper removes data from deleted accounts in the background.

An account is marked for deletion by a reseller issuing a DELETE request on the account's storage URL. This simply puts the value DELETED into the status column of the account\_stat table in the account database (and replicas), indicating the data for the account should be deleted later.

There is normally no set retention time and no undelete; it is assumed the reseller will implement such features and only call DELETE on the account once it is truly desired the account's data be removed. However,

in order to protect the Swift cluster accounts from an improper or mistaken delete request, you can set a `delay_reaping` value in the `[account-reaper]` section of the `account-server.conf` to delay the actual deletion of data. At this time, there is no utility to undelete an account; one would have to update the account database replicas directly, setting the status column to an empty string and updating the `put_timestamp` to be greater than the `delete_timestamp`. (On the TODO list is writing a utility to perform this task, preferably through a ReST call.)

The account reaper runs on each account server and scans the server occasionally for account databases marked for deletion. It will only trigger on accounts that server is the primary node for, so that multiple account servers aren't all trying to do the same work at the same time. Using multiple servers to delete one account might improve deletion speed, but requires coordination so they aren't duplicating efforts. Speed really isn't as much of a concern with data deletion and large accounts aren't deleted that often.

The deletion process for an account itself is pretty straightforward. For each container in the account, each object is deleted and then the container is deleted. Any deletion requests that fail won't stop the overall process, but will cause the overall process to fail eventually (for example, if an object delete times out, the container won't be able to be deleted later and therefore the account won't be deleted either). The overall process continues even on a failure so that it doesn't get hung up reclaiming cluster space because of one troublesome spot. The account reaper will keep trying to delete an account until it eventually becomes empty, at which point the database reclaim process within the `db_replicator` will eventually remove the database files.

Sometimes a persistent error state can prevent some object or container from being deleted. If this happens, you will see a message such as "Account <name> has not been reaped since <date>" in the log. You can control when this is logged with the `reap_warn_after` value in the `[account-reaper]` section of the `account-server.conf` file. By default this is 30 days.

## Swift Replication

Because each replica in swift functions independently, and clients generally require only a simple majority of nodes responding to consider an operation successful, transient failures like network partitions can quickly

cause replicas to diverge. These differences are eventually reconciled by asynchronous, peer-to-peer replicator processes. The replicator processes traverse their local filesystems, concurrently performing operations in a manner that balances load across physical disks.

Replication uses a push model, with records and files generally only being copied from local to remote replicas. This is important because data on the node may not belong there (as in the case of handoffs and ring changes), and a replicator can't know what data exists elsewhere in the cluster that it should pull in. It's the duty of any node that contains data to ensure that data gets to where it belongs. Replica placement is handled by the ring.

Every deleted record or file in the system is marked by a tombstone, so that deletions can be replicated alongside creations. The replication process cleans up tombstones after a time period known as the consistency window. The consistency window encompasses replication duration and how long transient failure can remove a node from the cluster. Tombstone cleanup must be tied to replication to reach replica convergence.

If a replicator detects that a remote drive has failed, the replicator uses the `get_more_nodes` interface for the ring to choose an alternate node with which to synchronize. The replicator can maintain desired levels of replication in the face of disk failures, though some replicas may not be in an immediately usable location. Note that the replicator doesn't maintain desired levels of replication when other failures, such as entire node failures occur, because most failure are transient.

Replication is an area of active development, and likely rife with potential improvements to speed and accuracy.

There are two major classes of replicator - the db replicator, which replicates accounts and containers, and the object replicator, which replicates object data.

### **DB Replication**

The first step performed by db replication is a low-cost hash comparison to determine whether two replicas already match. Under normal operation, this check is able to verify that most databases in the system are

already synchronized very quickly. If the hashes differ, the replicator brings the databases in sync by sharing records added since the last sync point.

This sync point is a high water mark noting the last record at which two databases were known to be in sync, and is stored in each database as a tuple of the remote database id and record id. Database ids are unique amongst all replicas of the database, and record ids are monotonically increasing integers. After all new records have been pushed to the remote database, the entire sync table of the local database is pushed, so the remote database can guarantee that it is in sync with everything with which the local database has previously synchronized.

If a replica is found to be missing entirely, the whole local database file is transmitted to the peer using `rsync(1)` and vested with a new unique id.

In practice, DB replication can process hundreds of databases per concurrency setting per second (up to the number of available CPUs or disks) and is bound by the number of DB transactions that must be performed.

### **Object Replication**

The initial implementation of object replication simply performed an `rsync` to push data from a local partition to all remote servers it was expected to exist on. While this performed adequately at small scale, replication times skyrocketed once directory structures could no longer be held in RAM. We now use a modification of this scheme in which a hash of the contents for each suffix directory is saved to a per-partition hashes file. The hash for a suffix directory is invalidated when the contents of that suffix directory are modified.

The object replication process reads in these hash files, calculating any invalidated hashes. It then transmits the hashes to each remote server that should hold the partition, and only suffix directories with differing hashes on the remote server are rsynced. After pushing files to the remote server, the replication process notifies it to recalculate hashes for the rsynced suffix directories.

Performance of object replication is generally bound by the number of uncached directories it has to traverse, usually as a result of invalidated suffix directory hashes. Using write volume and partition counts from our running systems, it was designed so that around 2% of the hash space on a normal node will be invalidated per day, which has experimentally given us acceptable replication speeds.

# 14. Object Storage Node Lab

## Table of Contents

|                                             |     |
|---------------------------------------------|-----|
| Day 9, 13:30 to 14:45, 15:00 to 17:00 ..... | 237 |
| Installing Object Node .....                | 237 |
| Configuring Object Node .....               | 239 |
| Configuring Object Proxy .....              | 242 |
| Start Object Node Services .....            | 247 |

## Day 9, 13:30 to 14:45, 15:00 to 17:00

## Installing Object Node

1. Create a `swift` user that the Object Storage Service can use to authenticate with the Identity Service. Choose a password and specify an email address for the `swift` user. Use the `service` tenant and give the user the `admin` role:

```
$ keystone user-create --name=swift --pass=SWIFT_PASS \  
--email=swift@example.com  
$ keystone user-role-add --user=swift --tenant=service --role=admin
```

2. Create a service entry for the Object Storage Service:

```
$ keystone service-create --name=swift --type=object-store \  

```

```
--description="OpenStack Object Storage"
```

| Property    | Value                            |
|-------------|----------------------------------|
| description | OpenStack Object Storage         |
| id          | eede9296683e4b5ebfa13f5166375ef6 |
| name        | swift                            |
| type        | object-store                     |



### Note

The service ID is randomly generated and is different from the one shown here.

- Specify an API endpoint for the Object Storage Service by using the returned service ID. When you specify an endpoint, you provide URLs for the public API, internal API, and admin API. In this guide, the controller host name is used:

```
$ keystone endpoint-create \
  --service-id=$(keystone service-list | awk '/ object-store / {print $2}') \
  --publicurl='http://controller:8080/v1/AUTH_$(tenant_id)s' \
  --internalurl='http://controller:8080/v1/AUTH_$(tenant_id)s' \
  --adminurl=http://controller:8080
```

| Property    | Value                                         |
|-------------|-----------------------------------------------|
| adminurl    | http://controller:8080/                       |
| id          | 9e3ce428f82b40d38922f242c095982e              |
| internalurl | http://controller:8080/v1/AUTH_\$(tenant_id)s |
| publicurl   | http://controller:8080/v1/AUTH_\$(tenant_id)s |
| region      | regionOne                                     |
| service_id  | eede9296683e4b5ebfa13f5166375ef6              |

- Create the configuration directory on all nodes:

```
# mkdir -p /etc/swift
```

5. Create `/etc/swift/swift.conf` on all nodes:

```
[swift-hash]  
# random unique string that can never change (DO NOT LOSE)  
swift_hash_path_suffix = fLibertyGibbitZ
```



### Note

The suffix value in `/etc/swift/swift.conf` should be set to some random string of text to be used as a salt when hashing to determine mappings in the ring. This file must be the same on every node in the cluster!

Next, set up your storage nodes and proxy node. This example uses the Identity Service for the common authentication piece.

## Configuring Object Node



### Note

Object Storage works on any file system that supports Extended Attributes (XATTRS). XFS shows the best overall performance for the swift use case after considerable testing and benchmarking at Rackspace. It is also the only file system that has been thoroughly tested. See the [OpenStack Configuration Reference](#) for additional recommendations.

1. Install storage node packages:

```
# apt-get install swift swift-account swift-container swift-object xfsprogs
```

```
# yum install openstack-swift-account openstack-swift-container \  
openstack-swift-object xfsprogs xinetd
```

```
# zypper install openstack-swift-account openstack-swift-container \  
openstack-swift-object python-xml xfsprogs xinetd
```

2. For each device on the node that you want to use for storage, set up the XFS volume (/dev/sdb is used as an example). Use a single partition per drive. For example, in a server with 12 disks you may use one or two disks for the operating system which should not be touched in this step. The other 10 or 11 disks should be partitioned with a single partition, then formatted in XFS.

```
# fdisk /dev/sdb  
# mkfs.xfs /dev/sdb1  
# echo "/dev/sdb1 /srv/node/sdb1 xfs noatime,nodiratime,nobarrier,logbufs=8 0 0" >> /etc/  
fstab  
# mkdir -p /srv/node/sdb1  
# mount /srv/node/sdb1  
# chown -R swift:swift /srv/node
```

3. Create /etc/rsyncd.conf:

Replace the content of /etc/rsyncd.conf with:

```
uid = swift  
gid = swift  
log file = /var/log/rsyncd.log  
pid file = /var/run/rsyncd.pid  
address = STORAGE_LOCAL_NET_IP  
  
[account]  
max connections = 2  
path = /srv/node/  
read only = false  
lock file = /var/lock/account.lock  
  
[container]  
max connections = 2  
path = /srv/node/
```



```
read only = false
lock file = /var/lock/container.lock

[object]
max connections = 2
path = /srv/node/
read only = false
lock file = /var/lock/object.lock
```

4. (Optional) If you want to separate rsync and replication traffic to replication network, set `STORAGE_REPLICATION_NET_IP` instead of `STORAGE_LOCAL_NET_IP`:

```
address = STORAGE_REPLICATION_NET_IP
```

5. Edit the following line in `/etc/default/rsync`:

```
RSYNC_ENABLE=true
```

6. Edit the following line in `/etc/xinetd.d/rsync`:

```
disable = false
```

7. Start the rsync service:

```
# service rsync start
```

Start the xinetd service:

```
# service xinetd start
```

Start the xinetd service and configure it to start when the system boots:

```
# service xinetd start
# chkconfig xinetd on
```

**Note**

The rsync service requires no authentication, so run it on a local, private network.

8. Create the swift recon cache directory and set its permissions:

```
# mkdir -p /var/swift/recon  
# chown -R swift:swift /var/swift/recon
```

## Configuring Object Proxy

The proxy server takes each request and looks up locations for the account, container, or object and routes the requests correctly. The proxy server also handles API requests. You enable account management by configuring it in the `/etc/swift/proxy-server.conf` file.

**Note**

The Object Storage processes run under a separate user and group, set by configuration options, and referred to as `swift:swift`. The default user is `swift`.

1. Install swift-proxy service:

```
# apt-get install swift-proxy memcached python-keystoneclient python-swiftclient python-webob
```

```
# yum install openstack-swift-proxy memcached python-swiftclient python-keystone-auth-token
```

```
# zypper install openstack-swift-proxy memcached python-swiftclient python-keystoneclient python-xml
```

2. Modify memcached to listen on the default interface on a local, non-public network. Edit this line in the `/etc/memcached.conf` file:

```
-l 127.0.0.1
```

Change it to:

```
-l PROXY_LOCAL_NET_IP
```

3. Modify memcached to listen on the default interface on a local, non-public network. Edit the `/etc/sysconfig/memcached` file:

```
OPTIONS="-l PROXY_LOCAL_NET_IP"
```

```
MEMCACHED_PARAMS="-l PROXY_LOCAL_NET_IP"
```

4. Restart the memcached service:

```
# service memcached restart
```

5. Start the memcached service and configure it to start when the system boots:

```
# service memcached start
```

```
# chkconfig memcached on
```

6. Create Edit `/etc/swift/proxy-server.conf`:

```
[DEFAULT]
bind_port = 8080
user = swift

[pipeline:main]
pipeline = healthcheck cache authtoken keystoneauth proxy-server

[app:proxy-server]
use = egg:swift#proxy
allow_account_management = true
account_autocreate = true
```

```
[filter:keystoneauth]
use = egg:swift#keystoneauth
operator_roles = Member,admin,swiftoperator

[filter:authtoken]
paste.filter_factory = keystoneclient.middleware.auth_token:filter_factory

# Delaying the auth decision is required to support token-less
# usage for anonymous referrers ('.r:*').
delay_auth_decision = true

# cache directory for signing certificate
signing_dir = /home/swift/keystone-signing

# auth_* settings refer to the Keystone server
auth_protocol = http
auth_host = controller
auth_port = 35357

# the service tenant and swift username and password created in Keystone
admin_tenant_name = service
admin_user = swift
admin_password = SWIFT_PASS

[filter:cache]
use = egg:swift#memcache

[filter:catch_errors]
use = egg:swift#catch_errors

[filter:healthcheck]
use = egg:swift#healthcheck
```



## Note

If you run multiple memcache servers, put the multiple IP:port listings in the [filter:cache] section of the /etc/swift/proxy-server.conf file:

```
10.1.2.3:11211,10.1.2.4:11211
```

Only the proxy server uses memcache.

7. Create the account, container, and object rings. The builder command creates a builder file with a few parameters. The parameter with the value of 18 represents  $2^{18}$ , the value that the partition is sized to. Set this "partition power" value based on the total amount of storage you expect your entire ring to use. The value 3 represents the number of replicas of each object, with the last value being the number of hours to restrict moving a partition more than once.

```
# cd /etc/swift
# swift-ring-builder account.builder create 18 3 1
# swift-ring-builder container.builder create 18 3 1
# swift-ring-builder object.builder create 18 3 1
```

8. For every storage device on each node add entries to each ring:

```
# swift-ring-builder account.builder add
  zZONE-STORAGE_LOCAL_NET_IP:6002[RSTORAGE_REPLICATION_NET_IP:6005]/DEVICE 100
# swift-ring-builder container.builder add
  zZONE-STORAGE_LOCAL_NET_IP_1:6001[RSTORAGE_REPLICATION_NET_IP:6004]/DEVICE 100
# swift-ring-builder object.builder add
  zZONE-STORAGE_LOCAL_NET_IP_1:6000[RSTORAGE_REPLICATION_NET_IP:6003]/DEVICE 100
```



## Note

You must omit the optional *STORAGE\_REPLICATION\_NET\_IP* parameter if you do not want to use dedicated network for replication.

For example, if a storage node has a partition in Zone 1 on IP 10.0.0.1, the storage node has address 10.0.1.1 from replication network. The mount point of this partition is /srv/node/sdb1, and the path in /etc/rsyncd.conf is /srv/node/, the DEVICE would be sdb1 and the commands are:

```
# swift-ring-builder account.builder add z1-10.0.0.1:6002R10.0.1.1:6005/sdb1 100
# swift-ring-builder container.builder add z1-10.0.0.1:6001R10.0.1.1:6004/sdb1 100
# swift-ring-builder object.builder add z1-10.0.0.1:6000R10.0.1.1:6003/sdb1 100
```



### Note

If you assume five zones with one node for each zone, start ZONE at 1. For each additional node, increment ZONE by 1.

9. Verify the ring contents for each ring:

```
# swift-ring-builder account.builder
# swift-ring-builder container.builder
# swift-ring-builder object.builder
```

10. Rebalance the rings:

```
# swift-ring-builder account.builder rebalance
# swift-ring-builder container.builder rebalance
# swift-ring-builder object.builder rebalance
```



### Note

Rebalancing rings can take some time.

11. Copy the `account.ring.gz`, `container.ring.gz`, and `object.ring.gz` files to each of the Proxy and Storage nodes in /etc/swift.
12. Make sure the swift user owns all configuration files:

```
# chown -R swift:swift /etc/swift
```

13. Restart the Proxy service:

```
# service swift-proxy restart
```

14. Start the Proxy service and configure it to start when the system boots:

```
# service openstack-swift-proxy start  
# chkconfig openstack-swift-proxy on
```

## Start Object Node Services

Now that the ring files are on each storage node, you can start the services. On each storage node, run the following command:

```
# for service in \  
  swift-object swift-object-replicator swift-object-updater swift-object-auditor \  
  swift-container swift-container-replicator swift-container-updater swift-container-auditor \  
  swift-account swift-account-replicator swift-account-reaper swift-account-auditor; do \  
  service $service start; done
```

```
# for service in \  
  openstack-swift-object openstack-swift-object-replicator openstack-swift-object-updater  
  openstack-swift-object-auditor \  
  openstack-swift-container openstack-swift-container-replicator openstack-swift-container-  
  updater openstack-swift-container-auditor \  
  openstack-swift-account openstack-swift-account-replicator openstack-swift-account-reaper  
  openstack-swift-account-auditor; do \  
  service $service start; chkconfig $service on; done
```



### Note

To start all swift services at once, run the command:

```
# swift-init all start
```

To know more about `swift-init` command, run:

```
$ man swift-init
```



## **Developer Training Guide**



## Table of Contents

|                                                                 |    |
|-----------------------------------------------------------------|----|
| 1. Getting Started .....                                        | 1  |
| Day 1, 09:00 to 11:00, 11:15 to 12:30 .....                     | 1  |
| Overview .....                                                  | 1  |
| Review Operator Introduction .....                              | 2  |
| Review Operator Brief Overview .....                            | 4  |
| Review Operator Core Projects .....                             | 7  |
| Review Operator OpenStack Architecture .....                    | 21 |
| Review Operator Virtual Machine Provisioning Walk-Through ..... | 33 |
| 2. Getting Started Lab .....                                    | 41 |
| Day 1, 13:30 to 14:45, 15:00 to 17:00 .....                     | 41 |
| Getting the Tools and Accounts for Committing Code .....        | 41 |
| Fix a Documentation Bug .....                                   | 45 |
| Submit a Documentation Bug .....                                | 49 |
| Create a Branch .....                                           | 49 |
| Optional: Add to the Training Guide Documentation .....         | 51 |
| 3. Getting Started Quiz .....                                   | 53 |
| Day 1, 16:40 to 17:00 .....                                     | 53 |
| 4. Developer APIs in Depth .....                                | 55 |
| Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30 .....                | 55 |
| 5. Developer APIs in Depth Lab Day Two .....                    | 57 |
| Day 2, 13:30 to 14:45, 15:00 to 16:30 .....                     | 57 |
| 6. Developer APIs in Depth Day Two Quiz .....                   | 59 |
| Day 2, 16:40 to 17:00 .....                                     | 59 |
| 7. Developer APIs in Depth Lab Day Three .....                  | 61 |
| Day 3, 13:30 to 14:45, 15:00 to 16:30 .....                     | 61 |
| 8. Developer APIs in Depth Day Three Quiz .....                 | 63 |
| Day 3, 16:40 to 17:00 .....                                     | 63 |
| 9. Developer How To Participate Lab Day Four .....              | 65 |

|                                                                                          |    |
|------------------------------------------------------------------------------------------|----|
| Day 4, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 65 |
| 10. Developer APIs in Depth Day Four Quiz .....                                          | 67 |
| Day 4, 16:40 to 17:00 .....                                                              | 67 |
| 11. Developer How To Participate .....                                                   | 69 |
| Day 5 to 9, 09:00 to 11:00, 11:15 to 12:30 .....                                         | 69 |
| 12. Developer How To Participate Lab Day Five .....                                      | 71 |
| Day 5, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 71 |
| 13. Developer How To Participate Day Five Quiz .....                                     | 73 |
| Day 5, 16:40 to 17:00 .....                                                              | 73 |
| 14. Developer How To Participate Lab Day Six .....                                       | 75 |
| Day 6, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 75 |
| 15. Developer How To Participate Day Six Quiz .....                                      | 77 |
| Day 6, 16:40 to 17:00 .....                                                              | 77 |
| 16. Developer How To Participate Lab Day Seven .....                                     | 79 |
| Day 7, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 79 |
| 17. Developer How To Participate Day Seven Quiz .....                                    | 81 |
| Day 7, 16:40 to 17:00 .....                                                              | 81 |
| 18. Developer How To Participate Lab Day Eight .....                                     | 83 |
| Day 8, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 83 |
| 19. Developer How To Participate Day Eight Quiz .....                                    | 85 |
| Day 8, 16:40 to 17:00 .....                                                              | 85 |
| 20. Developer How To Participate Lab Day Nine .....                                      | 87 |
| Day 9, 13:30 to 14:45, 15:00 to 16:30 .....                                              | 87 |
| 21. Developer How To Participate Day Nine Quiz .....                                     | 89 |
| Day 9, 16:40 to 17:00 .....                                                              | 89 |
| 22. Assessment .....                                                                     | 91 |
| Day 10, 9:00 to 11:00, 11:15 to 12:30, hands on lab 13:30 to 14:45, 15:00 to 17:00 ..... | 91 |
| Questions .....                                                                          | 91 |
| 23. Developer How To Participate Bootcamp .....                                          | 93 |
| One Day with Focus on Contribution .....                                                 | 93 |
| Overview .....                                                                           | 93 |

|                                          |    |
|------------------------------------------|----|
| Morning Classroom 10:00 to 11:15 .....   | 94 |
| Morning Lab 11:30 to 12:30 .....         | 95 |
| Morning Quiz 12:30 to 12:50 .....        | 95 |
| Afternoon Classroom 13:30 to 14:45 ..... | 95 |
| Afternoon Lab 15:00 to 17:00 .....       | 96 |
| Afternoon Quiz 17:00 to 17:20 .....      | 96 |



## List of Figures

|                                                                                              |    |
|----------------------------------------------------------------------------------------------|----|
| 1.1. Nebula (NASA) .....                                                                     | 5  |
| 1.2. Community Heartbeat .....                                                               | 9  |
| 1.3. Various Projects under OpenStack .....                                                  | 10 |
| 1.4. Programming Languages used to design OpenStack .....                                    | 12 |
| 1.5. OpenStack Compute: Provision and manage large networks of virtual machines .....        | 14 |
| 1.6. OpenStack Storage: Object and Block storage for use with servers and applications ..... | 15 |
| 1.7. OpenStack Networking: Pluggable, scalable, API-driven network and IP management .....   | 17 |
| 1.8. Conceptual Diagram .....                                                                | 23 |
| 1.9. Logical Diagram .....                                                                   | 25 |
| 1.10. Horizon Dashboard .....                                                                | 27 |
| 1.11. Initial State .....                                                                    | 36 |
| 1.12. Launch VM Instance .....                                                               | 38 |
| 1.13. End State .....                                                                        | 40 |





## List of Tables

|                                   |    |
|-----------------------------------|----|
| 22.1. Assessment Question 1 ..... | 91 |
| 22.2. Assessment Question 2 ..... | 91 |



# 1. Getting Started

## Table of Contents

|                                                                 |    |
|-----------------------------------------------------------------|----|
| Day 1, 09:00 to 11:00, 11:15 to 12:30 .....                     | 1  |
| Overview .....                                                  | 1  |
| Review Operator Introduction .....                              | 2  |
| Review Operator Brief Overview .....                            | 4  |
| Review Operator Core Projects .....                             | 7  |
| Review Operator OpenStack Architecture .....                    | 21 |
| Review Operator Virtual Machine Provisioning Walk-Through ..... | 33 |

## Day 1, 09:00 to 11:00, 11:15 to 12:30

### Overview

Training would take 2.5 months self paced, (5) 2 week periods with a user group meeting, or 40 hours instructor led with 40 hours of self paced lab time.

#### Prerequisites

1. Associate guide training
2. Associate guide virtualbox scripted install completed and running

## Review Operator Introduction

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering users to provision resources through a web interface.

Cloud computing provides users with access to a shared collection of computing resources: networks for transfer, servers for storage, and applications or services for completing tasks.

The compelling features of a cloud are:

- On-demand self-service: Users can automatically provision needed computing capabilities, such as server time and network storage, without requiring human interaction with each service provider.
- Network access: Any computing capabilities are available over the network. Many different devices are allowed access through standardized mechanisms.
- Resource pooling: Multiple users can access clouds that serve other consumers according to demand.
- Elasticity: Provisioning is rapid and scales out or is based on need.
- Metered or measured service: Cloud systems can optimize and control resource use at the level that is appropriate for the service. Services include storage, processing, bandwidth, and active user accounts. Monitoring and reporting of resource usage provides transparency for both the provider and consumer of the utilized service.

Cloud computing offers different service models depending on the capabilities a consumer may require.

- SaaS: Software-as-a-Service. Provides the consumer the ability to use the software in a cloud environment, such as web-based email for example.

- PaaS: Platform-as-a-Service. Provides the consumer the ability to deploy applications through a programming language or tools supported by the cloud platform provider. An example of Platform-as-a-service is an Eclipse/Java programming platform provided with no downloads required.
- IaaS: Infrastructure-as-a-Service. Provides infrastructure such as computer instances, network connections, and storage so that people can run any software or operating system.

Terms such as public cloud or private cloud refer to the deployment model for the cloud. A private cloud operates for a single organization, but can be managed on-premise or off-premise. A public cloud has an infrastructure that is available to the general public or a large industry group and is likely owned by a cloud services company.

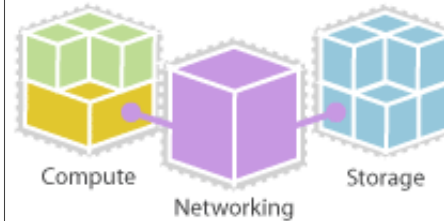
Clouds can also be described as hybrid. A hybrid cloud can be a deployment model, as a composition of both public and private clouds, or a hybrid model for cloud computing may involve both virtual and physical servers.

Cloud computing can help with large-scale computing needs or can lead consolidation efforts by virtualizing servers to make more use of existing hardware and potentially release old hardware from service. Cloud computing is also used for collaboration because of its high availability through networked computers. Productivity suites for word processing, number crunching, and email communications, and more are also available through cloud computing. Cloud computing also avails additional storage to the cloud user, avoiding the need for additional hard drives on each user's desktop and enabling access to huge data storage capacity online in the cloud.

When you explore OpenStack and see what it means technically, you can see its reach and impact on the entire world.

OpenStack is an open source software for building private and public clouds which delivers a massively scalable cloud operating system.

OpenStack is backed up by a global community of technologists, developers, researchers, corporations and cloud computing experts.



## Review Operator Brief Overview

OpenStack is a cloud operating system that controls large pools of compute, storage, and networking resources throughout a datacenter. It is all managed through a dashboard that gives administrators control while empowering their users to provision resources through a web interface.

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
- feature rich.

To check out more information on OpenStack visit <http://goo.gl/Ye9DFT>

### OpenStack Foundation:

The OpenStack Foundation, established September 2012, is an independent body providing shared resources to help achieve the OpenStack Mission by protecting, empowering, and promoting OpenStack software and the community around it. This includes users, developers and the entire ecosystem. For more information visit <http://goo.gl/3uvmNX>.

### Who's behind OpenStack?

Founded by Rackspace Hosting and NASA, OpenStack has grown to be a global software community of developers collaborating on a standard and massively scalable open source cloud operating system. The OpenStack Foundation promotes the development, distribution and adoption of the OpenStack cloud operating system. As the independent home for OpenStack, the Foundation has already attracted more than 7,000 individual members from 100 countries and 850 different organizations. It has also secured more than \$10 million in funding and is ready to fulfill the OpenStack mission of becoming the ubiquitous cloud computing platform. Checkout <http://goo.gl/BZHJKd> for more on the same.

**Figure 1.1. Nebula (NASA)**



The goal of the OpenStack Foundation is to serve developers, users, and the entire ecosystem by providing a set of shared resources to grow the footprint of public and private OpenStack clouds, enable technology vendors targeting the platform and assist developers in producing the best cloud software in the industry.

**Who uses OpenStack?**

Corporations, service providers, VARS, SMBs, researchers, and global data centers looking to deploy large-scale cloud deployments for private or public clouds leveraging the support and resulting technology of a global open source community. This is just three years into OpenStack, it's new, it's yet to mature and has immense possibilities. How do I say that? All these 'buzz words' will fall into a properly solved jigsaw puzzle as you go through this article.

**It's Open Source:**

All of the code for OpenStack is freely available under the Apache 2.0 license. Anyone can run it, build on it, or submit changes back to the project. This open development model is one of the best ways to foster badly-needed cloud standards, remove the fear of proprietary lock-in for cloud customers, and create a large ecosystem that spans cloud providers.

**Who it's for:**

Enterprises, service providers, government and academic institutions with physical hardware that would like to build a public or private cloud.

**How it's being used today:**

Organizations like CERN, Cisco WebEx, DreamHost, eBay, The Gap, HP, MercadoLibre, NASA, PayPal, Rackspace and University of Melbourne have deployed OpenStack clouds to achieve control, business agility and cost savings without the licensing fees and terms of proprietary software. For complete user stories visit <http://goo.gl/aF4lsL>, this should give you a good idea about the importance of OpenStack.



## Review Operator Core Projects

### Project history and releases overview.

OpenStack is a cloud computing project that provides an Infrastructure-as-a-Service (IaaS). It is free open source software released under the terms of the Apache License. The project is managed by the OpenStack Foundation, a non-profit corporate entity established in September 2012 to promote OpenStack software and its community.

More than 200 companies joined the project, among which are AMD, Brocade Communications Systems, Canonical, Cisco, Dell, EMC, Ericsson, Groupe Bull, HP, IBM, Inktank, Intel, NEC, Rackspace Hosting, Red Hat, SUSE Linux, VMware, and Yahoo!

The technology consists of a series of interrelated projects that control pools of processing, storage, and networking resources throughout a data center, all managed through a dashboard that gives administrators control while empowering its users to provision resources through a web interface.

The OpenStack community collaborates around a six-month, time-based release cycle with frequent development milestones. During the planning phase of each release, the community gathers for the OpenStack Design Summit to facilitate developer working sessions and assemble plans.

In July 2010 Rackspace Hosting and NASA jointly launched an open-source cloud-software initiative known as OpenStack. The OpenStack project intended to help organizations which offer cloud-computing services running on standard hardware. The first official release, code-named Austin, appeared four months later, with plans to release regular updates of the software every few months. The early code came from the NASA Nebula platform and from the Rackspace Cloud Files platform. In July 2011, Ubuntu Linux developers adopted OpenStack.

### OpenStack Releases

| Release Name | Release Date | Included Components |
|--------------|--------------|---------------------|
|--------------|--------------|---------------------|

|          |                   |                                                                             |
|----------|-------------------|-----------------------------------------------------------------------------|
| Austin   | 21 October 2010   | Nova, Swift                                                                 |
| Bexar    | 3 February 2011   | Nova, Glance, Swift                                                         |
| Cactus   | 15 April 2011     | Nova, Glance, Swift                                                         |
| Diablo   | 22 September 2011 | Nova, Glance, Swift                                                         |
| Essex    | 5 April 2012      | Nova, Glance, Swift, Horizon, Keystone                                      |
| Folsom   | 27 September 2012 | Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder                     |
| Grizzly  | 4 April 2013      | Nova, Glance, Swift, Horizon, Keystone, Quantum, Cinder                     |
| Havana   | 17 October 2013   | Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder                     |
| Icehouse | April 2014        | Nova, Glance, Swift, Horizon, Keystone, Neutron, Cinder, (More to be added) |

Some OpenStack users include:

- PayPal / eBay
- NASA
- CERN
- Yahoo!
- Rackspace Cloud
- HP Public Cloud
- MercadoLibre.com
- AT&T

- KT (formerly Korea Telecom)
- Deutsche Telekom
- Wikimedia Labs
- Hostalia of Telefonica Group
- SUSE Cloud solution
- Red Hat OpenShift PaaS solution
- Zadara Storage
- Mint Services
- GridCentric

OpenStack is a true and innovative open standard. For more user stories, see <http://goo.gl/aF4IsL>.

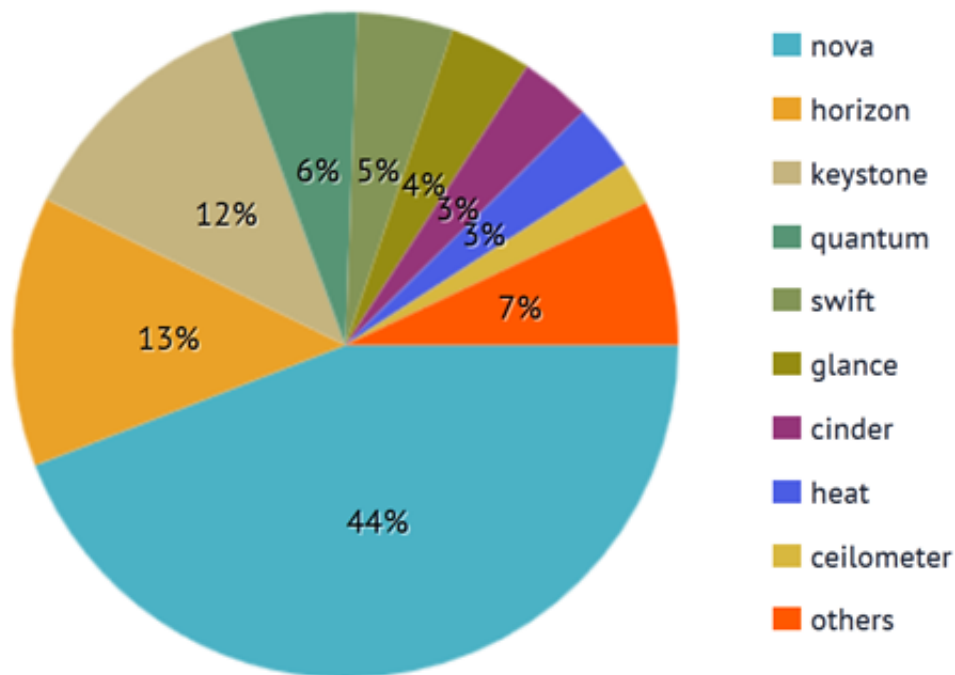
### Release Cycle

**Figure 1.2. Community Heartbeat**



OpenStack is based on a coordinated 6-month release cycle with frequent development milestones. You can find a link to the current development release schedule [here](#). The Release Cycle is made of four major stages:

**Figure 1.3. Various Projects under OpenStack**



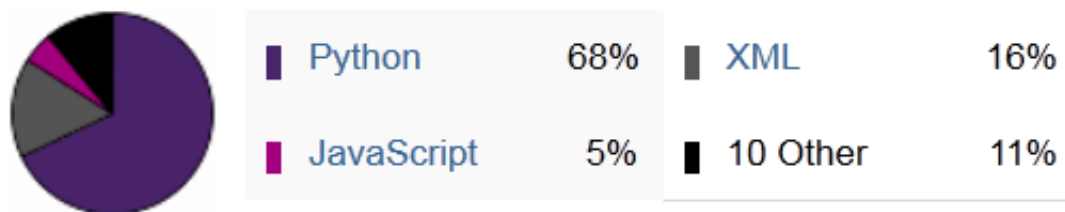
The creation of OpenStack took an estimated 249 years of effort (COCOMO model).

In a nutshell, OpenStack has:

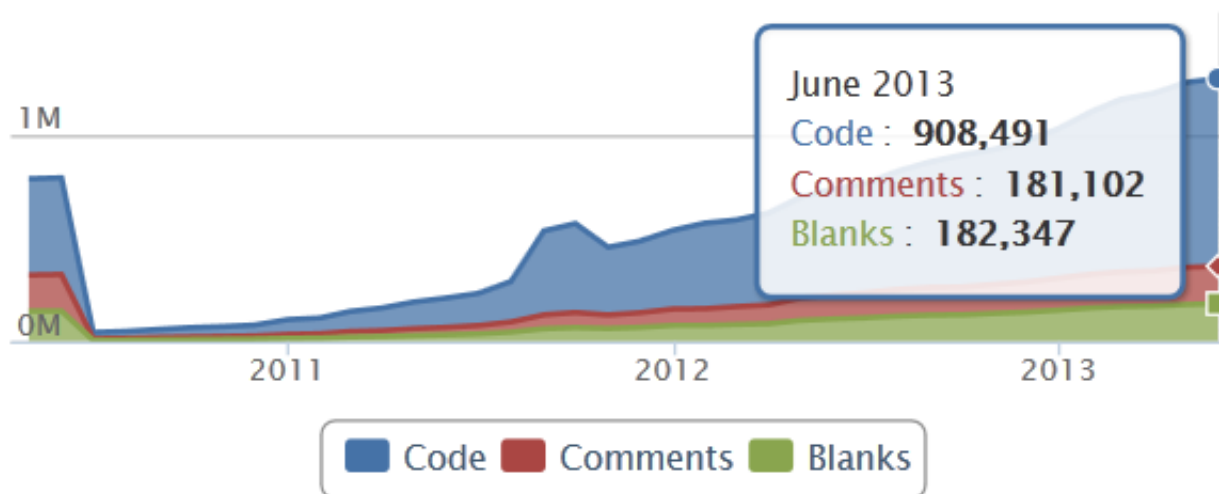
- 64,396 commits made by 1,128 contributors, with its first commit made in May, 2010.

- 908,491 lines of code. OpenStack is written mostly in Python with an average number of source code comments.
- A code base with a long source history.
- Increasing Y-O-Y commits.
- A very large development team comprised of people from around the world.

**Figure 1.4. Programming Languages used to design OpenStack**  
**Languages**



## Lines of Code



For an overview of OpenStack refer to <http://www.openstack.org> or <http://goo.gl/4q7nVI>. Common questions and answers are also covered here.

### **Core Projects Overview**

Let's take a dive into some of the technical aspects of OpenStack. Its scalability and flexibility are just some of the awesome features that make it a rock-solid cloud computing platform. The OpenStack core projects serve the community and its demands.

Being a cloud computing platform, OpenStack consists of many core and incubated projects which makes it really good as an IaaS cloud computing platform/Operating System. The following points are the main components necessary to call it an OpenStack Cloud.

### **Components of OpenStack**

OpenStack has a modular architecture with various code names for its components. OpenStack has several shared services that span the three pillars of compute, storage and networking, making it easier to implement and operate your cloud. These services - including identity, image management and a web interface - integrate the OpenStack components with each other as well as external systems to provide a unified experience for users as they interact with different cloud resources.

### **Compute (Nova)**

The OpenStack cloud operating system enables enterprises and service providers to offer on-demand computing resources, by provisioning and managing large networks of virtual machines. Compute resources are accessible via APIs for developers building cloud applications and via web interfaces for administrators and users. The compute architecture is designed to scale horizontally on standard hardware.

**Figure 1.5. OpenStack Compute: Provision and manage large networks of virtual machines**



OpenStack Compute (Nova) is a cloud computing fabric controller (the main part of an IaaS system). It is written in Python and uses many external libraries such as Eventlet (for concurrent programming), Kombu (for AMQP communication), and SQLAlchemy (for database access). Nova's architecture is designed to scale horizontally on standard hardware with no proprietary hardware or software requirements and provide the ability to integrate with legacy systems and third party technologies. It is designed to manage and automate pools of computer resources and can work with widely available virtualization technologies, as well as bare metal and high-performance computing (HPC) configurations. KVM and XenServer are available choices for hypervisor technology, together with Hyper-V and Linux container technology such as LXC. In addition to different hypervisors, OpenStack runs on ARM.

**Popular Use Cases:**

- Service providers offering an IaaS compute platform or services higher up the stack
- IT departments acting as cloud service providers for business units and project teams
- Processing big data with tools like Hadoop
- Scaling compute up and down to meet demand for web resources and applications
- High-performance computing (HPC) environments processing diverse and intensive workloads

**Object Storage(Swift)**



In addition to traditional enterprise-class storage technology, many organizations now have a variety of storage needs with varying performance and price requirements. OpenStack has support for both Object Storage and Block Storage, with many deployment options for each depending on the use case.

**Figure 1.6. OpenStack Storage: Object and Block storage for use with servers and applications**



OpenStack Object Storage (Swift) is a scalable redundant storage system. Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster. Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used.

Object Storage is ideal for cost effective, scale-out storage. It provides a fully distributed, API-accessible storage platform that can be integrated directly into applications or used for backup, archiving and data retention. Block Storage allows block devices to be exposed and connected to compute instances for expanded storage, better performance and integration with enterprise storage platforms, such as NetApp, Nexenta and SolidFire.

A few details on OpenStack's Object Storage

- OpenStack provides redundant, scalable object storage using clusters of standardized servers capable of storing petabytes of data

- Object Storage is not a traditional file system, but rather a distributed storage system for static data such as virtual machine images, photo storage, email storage, backups and archives. Having no central "brain" or master point of control provides greater scalability, redundancy and durability.
- Objects and files are written to multiple disk drives spread throughout servers in the data center, with the OpenStack software responsible for ensuring data replication and integrity across the cluster.
- Storage clusters scale horizontally simply by adding new servers. Should a server or hard drive fail, OpenStack replicates its content from other active nodes to new locations in the cluster. Because OpenStack uses software logic to ensure data replication and distribution across different devices, inexpensive commodity hard drives and servers can be used in lieu of more expensive equipment.

### **Block Storage(Cinder)**

OpenStack Block Storage (Cinder) provides persistent block level storage devices for use with OpenStack compute instances. The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs. In addition to local Linux server storage, it can use storage platforms including Ceph, CloudByte, Coraid, EMC (VMAX and VNX), GlusterFS, IBM Storage (Storwize family, SAN Volume Controller, and XIV Storage System), Linux LIO, NetApp, Nexenta, Scalality, SolidFire and HP (Store Virtual and StoreServ 3Par families). Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage. Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

#### **A few points on OpenStack Block Storage:**

- OpenStack provides persistent block level storage devices for use with OpenStack compute instances.
- The block storage system manages the creation, attaching and detaching of the block devices to servers. Block storage volumes are fully integrated into OpenStack Compute and the Dashboard allowing for cloud users to manage their own storage needs.

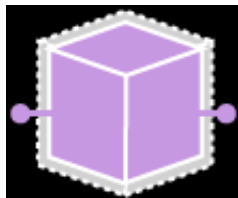
- In addition to using simple Linux server storage, it has unified storage support for numerous storage platforms including Ceph, NetApp, Nexenta, SolidFire, and Zadara.
- Block storage is appropriate for performance sensitive scenarios such as database storage, expandable file systems, or providing a server with access to raw block level storage.
- Snapshot management provides powerful functionality for backing up data stored on block storage volumes. Snapshots can be restored or used to create a new block storage volume.

### Networking(Neutron)

Today's data center networks contain more devices than ever before. From servers, network equipment, storage systems and security appliances, many of which are further divided into virtual machines and virtual networks. The number of IP addresses, routing configurations and security rules can quickly grow into the millions. Traditional network management techniques fall short of providing a truly scalable, automated approach to managing these next-generation networks. At the same time, users expect more control and flexibility with quicker provisioning.

OpenStack Networking is a pluggable, scalable and API-driven system for managing networks and IP addresses. Like other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing data center assets. OpenStack Networking ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self-service, even over their network configurations.

**Figure 1.7. OpenStack Networking: Pluggable, scalable, API-driven network and IP management**



OpenStack Networking (Neutron, formerly Quantum) is a system for managing networks and IP addresses. Like other aspects of the cloud operating system, it can be used by administrators and users to increase the value of existing data center assets. OpenStack Networking ensures the network will not be the bottleneck or limiting factor in a cloud deployment and gives users real self-service, even over their network configurations.

OpenStack Neutron provides networking models for different applications or user groups. Standard models include flat networks or VLANs for separation of servers and traffic. OpenStack Networking manages IP addresses, allowing for dedicated static IPs or DHCP. Floating IPs allow traffic to be dynamically re routed to any of your compute resources, which allows you to redirect traffic during maintenance or in the case of failure. Users can create their own networks, control traffic and connect servers and devices to one or more networks. Administrators can take advantage of software-defined networking (SDN) technology like OpenFlow to allow for high levels of multi-tenancy and massive scale. OpenStack Networking has an extension framework allowing additional network services, such as intrusion detection systems (IDS), load balancing, firewalls and virtual private networks (VPN) to be deployed and managed.

#### Networking Capabilities

- OpenStack provides flexible networking models to suit the needs of different applications or user groups. Standard models include flat networks or VLANs for separation of servers and traffic.
- OpenStack Networking manages IP addresses, allowing for dedicated static IPs or DHCP. Floating IPs allow traffic to be dynamically re-routed to any of your compute resources, which allows you to redirect traffic during maintenance or in the case of failure.
- Users can create their own networks, control traffic and connect servers and devices to one or more networks.
- The pluggable backend architecture lets users take advantage of commodity gear or advanced networking services from supported vendors.
- Administrators can take advantage of software-defined networking (SDN) technology like OpenFlow to allow for high levels of multi-tenancy and massive scale.

- OpenStack Networking has an extension framework allowing additional network services, such as intrusion detection systems (IDS), load balancing, firewalls and virtual private networks (VPN) to be deployed and managed.

### **Dashboard(Horizon)**

OpenStack Dashboard (Horizon) provides administrators and users a graphical interface to access, provision and automate cloud-based resources. The design allows for third party products and services, such as billing, monitoring and additional management tools. Service providers and other commercial vendors can customize the dashboard with their own brand.

The dashboard is just one way to interact with OpenStack resources. Developers can automate access or build tools to manage their resources using the native OpenStack API or the EC2 compatibility API.

### **Identity Service(Keystone)**

OpenStack Identity (Keystone) provides a central directory of users mapped to the OpenStack services they can access. It acts as a common authentication system across the cloud operating system and can integrate with existing backend directory services like LDAP. It supports multiple forms of authentication including standard username and password credentials, token-based systems, and Amazon Web Services log in credentials such as those used for EC2.

Additionally, the catalog provides a query-able list of all of the services deployed in an OpenStack cloud in a single registry. Users and third-party tools can programmatically determine which resources they can access.

The OpenStack Identity Service enables administrators to:

- Configure centralized policies across users and systems
- Create users and tenants and define permissions for compute, storage, and networking resources by using role-based access control (RBAC) features
- Integrate with an existing directory, like LDAP, to provide a single source of authentication across the enterprise

The OpenStack Identity Service enables users to:

- List the services to which they have access
- Make API requests
- Log into the web dashboard to create resources owned by their account

### **Image Service(Glance)**

OpenStack Image Service (Glance) provides discovery, registration and delivery services for disk and server images. Stored images can be used as a template. They can also be used to store and catalog an unlimited number of backups. The Image Service can store disk and server images in a variety of back-ends, including OpenStack Object Storage. The Image Service API provides a standard REST interface for querying information about disk images and lets clients stream the images to new servers.

Capabilities of the Image Service include:

- Administrators can create base templates from which their users can start new compute instances
- Users can choose from available images, or create their own from existing servers
- Snapshots can also be stored in the Image Service so that virtual machines can be backed up quickly

A multi-format image registry, the image service allows uploads of private and public images in a variety of formats, including:

- Raw
- Machine (kernel/ramdisk outside of image, also known as AMI)
- VHD (Hyper-V)
- VDI (VirtualBox)

- qcow2 (Qemu/KVM)
- VMDK (VMWare)
- OVF (VMWare, others)

To checkout the complete list of Core and Incubated projects under OpenStack check out OpenStack's Launchpad Project Page here : <http://goo.gl/ka4SrV>

### **Amazon Web Services compatibility**

OpenStack APIs are compatible with Amazon EC2 and Amazon S3 and thus client applications written for Amazon Web Services can be used with OpenStack with minimal porting effort.

### **Governance**

OpenStack is governed by a non-profit foundation and its board of directors, a technical committee and a user committee.

The foundation's stated mission is by providing shared resources to help achieve the OpenStack Mission by Protecting, Empowering, and Promoting OpenStack software and the community around it, including users, developers and the entire ecosystem. Though, it has little to do with the development of the software, which is managed by the technical committee - an elected group that represents the contributors to the project, and has oversight on all technical matters.

## **Review Operator OpenStack Architecture**

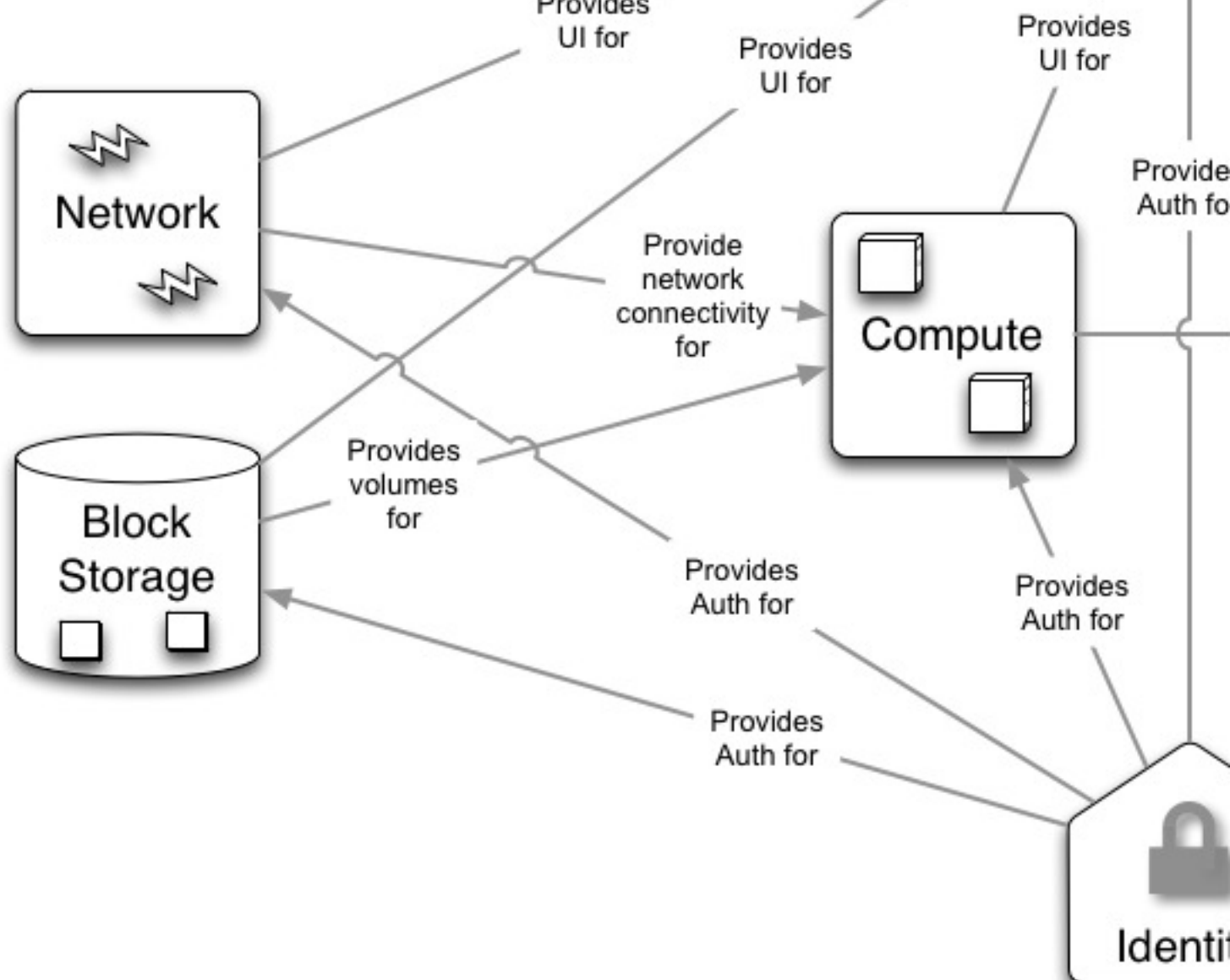
### **Conceptual Architecture**

The OpenStack project as a whole is designed to deliver a massively scalable cloud operating system. To achieve this, each of the constituent services are designed to work together to provide a complete

Infrastructure-as-a-Service (IaaS). This integration is facilitated through public application programming interfaces (APIs) that each service offers (and in turn can consume). While these APIs allow each of the services to use another service, it also allows an implementer to switch out any service as long as they maintain the API. These are (mostly) the same APIs that are available to end users of the cloud.

Conceptually, you can picture the relationships between the services as so:





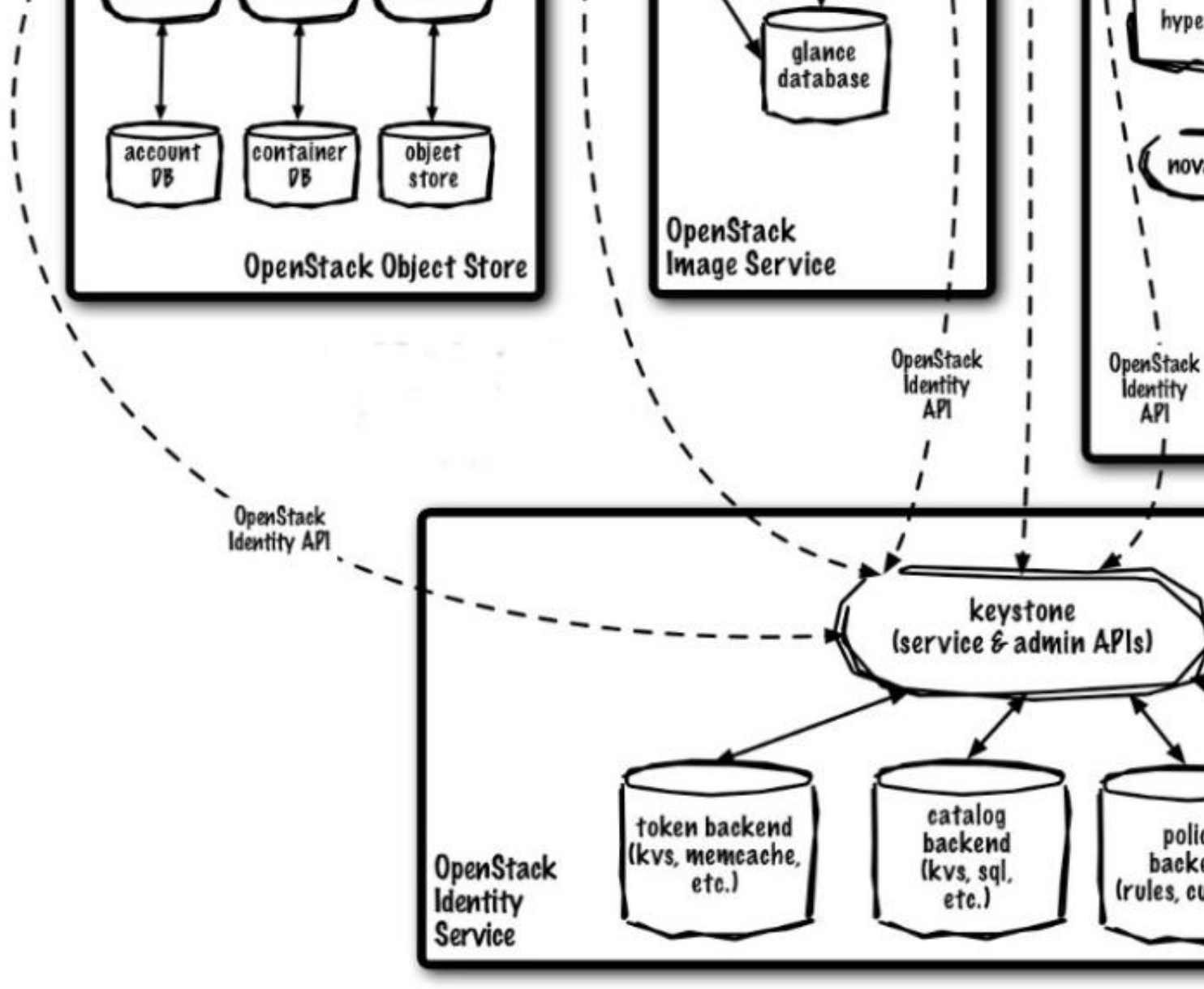
- Dashboard ("Horizon") provides a web front end to the other OpenStack services
- Compute ("Nova") stores and retrieves virtual disks ("images") and associated metadata in Image ("Glance")
- Network ("Neutron") provides virtual networking for Compute.
- Block Storage ("Cinder") provides storage volumes for Compute.
- Image ("Glance") can store the actual virtual disk files in the Object Store("Swift")
- All the services authenticate with Identity ("Keystone")

This is a stylized and simplified view of the architecture, assuming that the implementer is using all of the services together in the most common configuration. It also only shows the "operator" side of the cloud – it does not picture how consumers of the cloud may actually use it. For example, many users will access object storage heavily (and directly).

### **Logical Architecture**

This picture is consistent with the conceptual architecture above:

OpenStack



- End users can interact through a common web interface (Horizon) or directly to each service through their API
- All services authenticate through a common source (facilitated through keystone)
- Individual services interact with each other through their public APIs (except where privileged administrator commands are necessary)

In the sections below, we'll delve into the architecture for each of the services.

### **Dashboard**

Horizon is a modular Django web application that provides an end user and administrator interface to OpenStack services.

PROJECT  
**invisible\_to\_admin** ▼

## Manage Compute

Overview

Instances & Volumes

Access & Security

Images & Snapshots

## Object Store

Containers

|                          | Name | IP Address | Size      |
|--------------------------|------|------------|-----------|
| <input type="checkbox"/> | test |            | 512MB RAM |

Displaying 1 item

## Volumes

|                    | Name | Description |
|--------------------|------|-------------|
| Displaying 0 items |      |             |

As with most web applications, the architecture is fairly simple:

- Horizon is usually deployed via `mod_wsgi` in Apache. The code itself is separated into a reusable python module with most of the logic (interactions with various OpenStack APIs) and presentation (to make it easily customizable for different sites).
- A database (configurable as to which one) which relies mostly on the other services for data. It also stores very little data of its own.

From a network architecture point of view, this service will need to be customer accessible as well as be able to talk to each service's public APIs. If you wish to use the administrator functionality (i.e. for other services), it will also need connectivity to their Admin API endpoints (which should be non-customer accessible).

## Compute

Nova is the most complicated and distributed component of OpenStack. A large number of processes cooperate to turn end user API requests into running virtual machines. Below is a list of these processes and their functions:

- `nova-api` accepts and responds to end user compute API calls. It supports OpenStack Compute API, Amazon's EC2 API and a special Admin API (for privileged users to perform administrative actions). It also initiates most of the orchestration activities (such as running an instance) as well as enforces some policy (mostly quota checks).
- The `nova-compute` process is primarily a worker daemon that creates and terminates virtual machine instances via hypervisor's APIs (XenAPI for XenServer/XCP, `libvirt` for KVM or QEMU, `VMwareAPI` for VMware, etc.). The process by which it does so is fairly complex but the basics are simple: accept actions from the queue and then perform a series of system commands (like launching a KVM instance) to carry them out while updating state in the database.
- `nova-volume` manages the creation, attaching and detaching of z volumes to compute instances (similar functionality to Amazon's Elastic Block Storage). It can use volumes from a variety of providers such as iSCSI or Rados Block Device in Ceph. A new OpenStack project, Cinder, will eventually replace nova-

volume functionality. In the Folsom release, nova-volume and the Block Storage service will have similar functionality.

- The nova-network worker daemon is very similar to nova-compute and nova-volume. It accepts networking tasks from the queue and then performs tasks to manipulate the network (such as setting up bridging interfaces or changing iptables rules). This functionality is being migrated to Neutron, a separate OpenStack project. In the Folsom release, much of the functionality will be duplicated between nova-network and Neutron.
- The nova-schedule process is conceptually the simplest piece of code in OpenStack Nova: it takes a virtual machine instance request from the queue and determines where it should run (specifically, which compute server host it should run on).
- The queue provides a central hub for passing messages between daemons. This is usually implemented with RabbitMQ today, but could be any AMQP message queue (such as Apache Qpid). New to the Folsom release is support for Zero MQ.
- The SQL database stores most of the build-time and runtime state for a cloud infrastructure. This includes the instance types that are available for use, instances in use, networks available and projects. Theoretically, OpenStack Nova can support any database supported by SQL-Alchemy but the only databases currently being widely used are SQLite3 (only appropriate for test and development work), MySQL and PostgreSQL.
- Nova also provides console services to allow end users to access their virtual instance's console through a proxy. This involves several daemons (nova-console, nova-novncproxy and nova-consoleauth).

Nova interacts with many other OpenStack services: Keystone for authentication, Glance for images and Horizon for web interface. The Glance interactions are central. The API process can upload and query Glance while nova-compute will download images for use in launching images.

### **Object Store**

The swift architecture is very distributed to prevent any single point of failure as well as to scale horizontally. It includes the following components:

- Proxy server (swift-proxy-server) accepts incoming requests via the OpenStack Object API or just raw HTTP. It accepts files to upload, modifications to metadata or container creation. In addition, it will also serve files or container listing to web browsers. The proxy server may utilize an optional cache (usually deployed with memcache) to improve performance.
- Account servers manage accounts defined with the object storage service.
- Container servers manage a mapping of containers (i.e folders) within the object store service.
- Object servers manage actual objects (i.e. files) on the storage nodes.
- There are also a number of periodic processes which run to perform housekeeping tasks on the large data store. The most important of these is the replication services, which ensures consistency and availability through the cluster. Other periodic processes include auditors, updaters and reapers.

Authentication is handled through configurable WSGI middleware (which will usually be Keystone).

### Image Store

The Glance architecture has stayed relatively stable since the Cactus release. The biggest architectural change has been the addition of authentication, which was added in the Diablo release. Just as a quick reminder, Glance has four main parts to it:

- glance-api accepts Image API calls for image discovery, image retrieval and image storage.
- glance-registry stores, processes and retrieves metadata about images (size, type, etc.).
- A database to store the image metadata. Like Nova, you can choose your database depending on your preference (but most people use MySQL or SQLite).
- A storage repository for the actual image files. In the diagram above, Swift is shown as the image repository, but this is configurable. In addition to Swift, Glance supports normal filesystems, RADOS block devices, Amazon S3 and HTTP. Be aware that some of these choices are limited to read-only usage.



There are also a number of periodic processes which run on Glance to support caching. The most important of these is the replication services, which ensures consistency and availability through the cluster. Other periodic processes include auditors, updaters and reapers.

As you can see from the diagram in the Conceptual Architecture section, Glance serves a central role to the overall IaaS picture. It accepts API requests for images (or image metadata) from end users or Nova components and can store its disk files in the object storage service, Swift.

### **Identity**

Keystone provides a single point of integration for OpenStack policy, catalog, token and authentication.

- Keystone handles API requests as well as providing configurable catalog, policy, token and identity services.
- Each Keystone function has a pluggable backend which allows different ways to use the particular service. Most support standard backends like LDAP or SQL, as well as Key Value Stores (KVS).

Most people will use this as a point of customization for their current authentication services.

### **Network**

Neutron provides "network connectivity as a service" between interface devices managed by other OpenStack services (most likely Nova). The service works by allowing users to create their own networks and then attach interfaces to them. Like many of the OpenStack services, Neutron is highly configurable due to its plug-in architecture. These plug-ins accommodate different networking equipment and software. As such, the architecture and deployment can vary dramatically. In the above architecture, a simple Linux networking plug-in is shown.

- neutron-server accepts API requests and then routes them to the appropriate Neutron plug-in for action.
- Neutron plug-ins and agents perform the actual actions such as plugging and unplugging ports, creating networks or subnets and IP addressing. These plug-ins and agents differ depending on the vendor and

technologies used in the particular cloud. Neutron ships with plug-ins and agents for: Cisco virtual and physical switches, NEC OpenFlow products, Open vSwitch, Linux bridging, the Ryu Network Operating System, and VMware NSX.

- The common agents are L3 (layer 3), DHCP (dynamic host IP addressing) and the specific plug-in agent.
- Most Neutron installations will also make use of a messaging queue to route information between the neutron-server and various agents as well as a database to store networking state for particular plug-ins.

Neutron will interact mainly with Nova, where it will provide networks and connectivity for its instances.

### **Block Storage**

Cinder separates out the persistent block storage functionality that was previously part of OpenStack Compute (in the form of nova-volume) into its own service. The OpenStack Block Storage API allows for manipulation of volumes, volume types (similar to compute flavors) and volume snapshots.

- cinder-api accepts API requests and routes them to cinder-volume for action.
- cinder-volume acts upon the requests by reading or writing to the Cinder database to maintain state, interacting with other processes (like cinder-scheduler) through a message queue and directly upon block storage providing hardware or software. It can interact with a variety of storage providers through a driver architecture. Currently, there are drivers for IBM, SolidFire, NetApp, Nexenta, Zadara, linux iSCSI and other storage providers.
- Much like nova-scheduler, the cinder-scheduler daemon picks the optimal block storage provider node to create the volume on.
- Cinder deployments will also make use of a messaging queue to route information between the cinder processes as well as a database to store volume state.

Like Neutron, Cinder will mainly interact with Nova, providing volumes for its instances.

# Review Operator Virtual Machine Provisioning Walk-Through

More Content To be Added ...

OpenStack Compute gives you a tool to orchestrate a cloud, including running instances, managing networks, and controlling access to the cloud through users and projects. The underlying open source project's name is Nova, and it provides the software that can control an Infrastructure-as-a-Service (IaaS) cloud computing platform. It is similar in scope to Amazon EC2 and Rackspace Cloud Servers. 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 web-based API.

## Hypervisors

OpenStack Compute requires a hypervisor and Compute controls the hypervisors through an API server. The process for selecting a hypervisor usually means prioritizing and making decisions based on budget and resource constraints as well as the inevitable list of supported features and required technical specifications. The majority of development is done with the KVM and Xen-based hypervisors. Refer to <http://wiki.openstack.org/HypervisorSupportMatrix> <http://goo.gl/n7AXnC> for a detailed list of features and support across the hypervisors.

With OpenStack Compute, you can orchestrate clouds using multiple hypervisors in different zones. The types of virtualization standards that may be used with Compute include:

- KVM- Kernel-based Virtual Machine (visit <http://goo.gl/70dvRb>)
- LXC- Linux Containers (through libvirt) (visit <http://goo.gl/Ous3ly>)
- QEMU- Quick EMUlator (visit <http://goo.gl/WWV9IL>)
- UML- User Mode Linux (visit <http://goo.gl/4HAKJj>)

- VMWare vSphere 4.1 update 1 and newer (visit <http://goo.gl/0DBeo5>)
- Xen- Xen, Citrix XenServer and Xen Cloud Platform (XCP) (visit <http://goo.gl/yXP9t1>)
- Bare Metal- Provisions physical hardware via pluggable sub-drivers. (visit <http://goo.gl/exfeSg>)

### Users and Tenants (Projects)

The OpenStack Compute system is designed to be used by many different cloud computing consumers or customers, basically tenants on a shared system, using role-based access assignments. Roles control the actions that a user is allowed to perform. In the default configuration, most actions do not require a particular role, but this is configurable by the system administrator editing the appropriate `policy.json` file that maintains the rules. For example, a rule can be defined so that a user cannot allocate a public IP without the admin role. A user's access to particular images is limited by tenant, but the username and password are assigned per user. Key pairs granting access to an instance are enabled per user, but quotas to control resource consumption across available hardware resources are per tenant.

While the original EC2 API supports users, OpenStack Compute adds the concept of tenants. Tenants are isolated resource containers forming the principal organizational structure within the Compute service. They consist of a separate VLAN, volumes, instances, images, keys, and users. A user can specify which tenant he or she wishes to be known as by appending `:project_id` to his or her access key. If no tenant is specified in the API request, Compute attempts to use a tenant with the same ID as the user.

For tenants, quota controls are available to limit the:

- Number of volumes which may be created
- Total size of all volumes within a project as measured in GB
- Number of instances which may be launched
- Number of processor cores which may be allocated

- Floating IP addresses (assigned to any instance when it launches so the instance has the same publicly accessible IP addresses)
- Fixed IP addresses (assigned to the same instance each time it boots, publicly or privately accessible, typically private for management purposes)

## Images and Instances

This introduction provides a high level overview of what images and instances are and description of the life-cycle of a typical virtual system within the cloud. There are many ways to configure the details of an OpenStack cloud and many ways to implement a virtual system within that cloud. These configuration details as well as the specific command-line utilities and API calls to perform the actions described are presented in the Image Management and Volume Management chapters.

Images are disk images which are templates for virtual machine file systems. The OpenStack Image Service is responsible for the storage and management of images within OpenStack.

Instances are the individual virtual machines running on physical compute nodes. The OpenStack Compute service manages instances. Any number of instances maybe started from the same image. Each instance is run from a copy of the base image so runtime changes made by an instance do not change the image it is based on. Snapshots of running instances may be taken which create a new image based on the current disk state of a particular instance.

When starting an instance a set of virtual resources known as a flavor must be selected. Flavors define how many virtual CPUs an instance has and the amount of RAM and size of its ephemeral disks. OpenStack provides a number of predefined flavors which cloud administrators may edit or add to. Users must select from the set of available flavors defined on their cloud.

Additional resources such as persistent volume storage and public IP address may be added to and removed from running instances. The examples below show the cinder-volume service which provide persistent block storage as opposed to the ephemeral storage provided by the instance flavor.

Here is an example of the life cycle of a typical virtual system within an OpenStack cloud to illustrate these concepts.

### Initial State

#### Images and Instances

The following diagram shows the system state prior to launching an instance. The image store fronted by the Image Service has some number of predefined images. In the cloud, there is an available compute node with available vCPU, memory and local disk resources. Plus there are a number of predefined volumes in the cinder - volume service.

Figure 2.1. Base image state with no running instances

**Figure 1.11. Initial State**

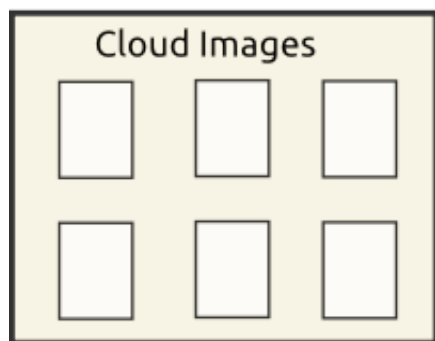


Image Store



Compute Node

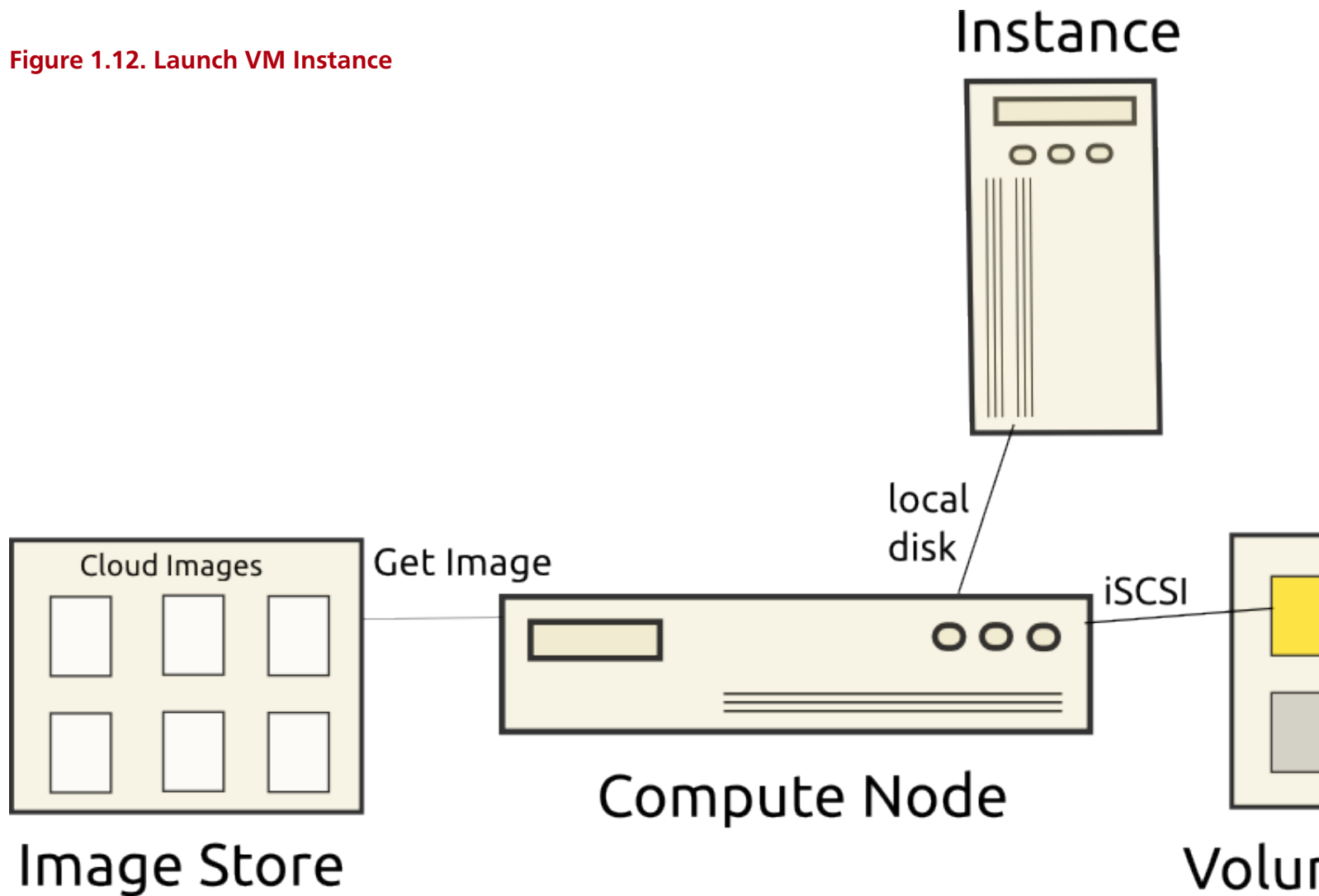


Block Storage

Launching an instance

To launch an instance, the user selects an image, a flavor, and other optional attributes. In this case the selected flavor provides a root volume (as all flavors do) labeled `vda` in the diagram and additional ephemeral storage labeled `vdb` in the diagram. The user has also opted to map a volume from the `cinder` -volume store to the third virtual disk, `vdc`, on this instance.

Figure 2.2. Instance creation from image and run time state

**Figure 1.12. Launch VM Instance**



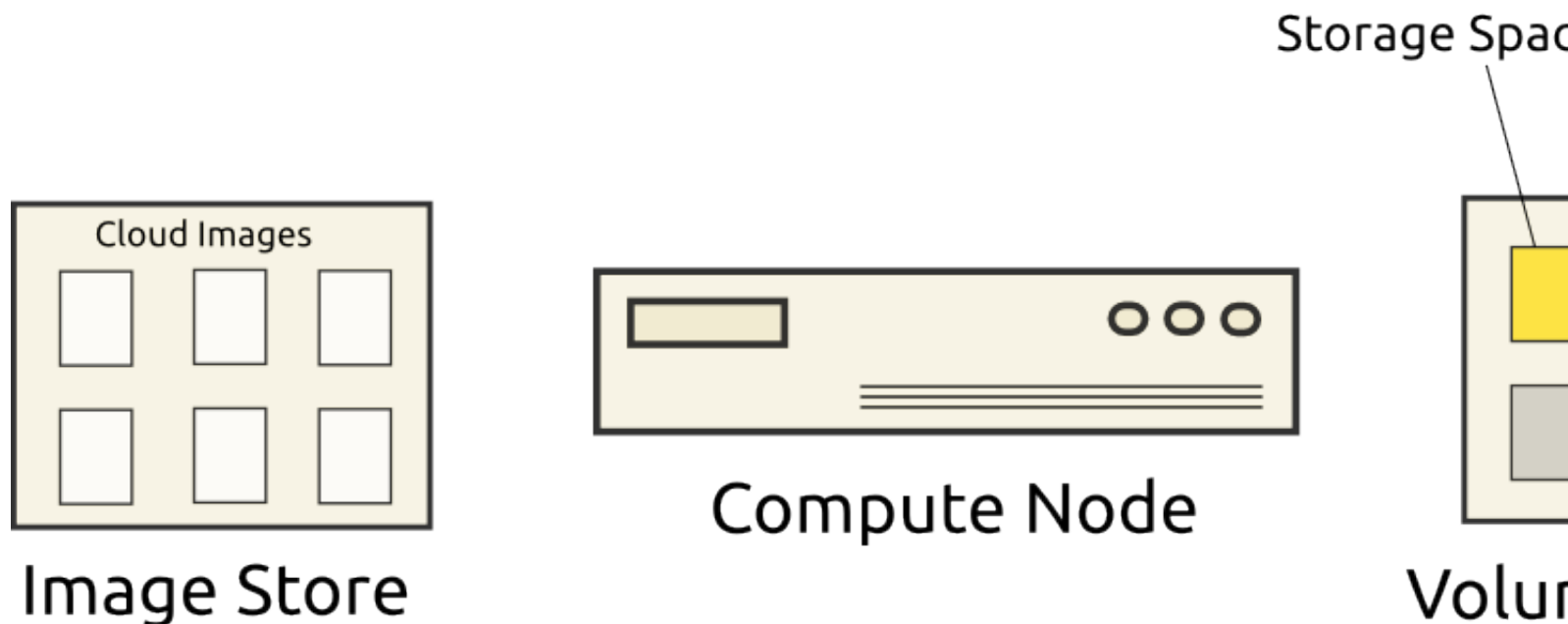
The OpenStack system copies the base image from the image store to local disk which is used as the first disk of the instance (vda). Having small images will result in faster start up of your instances as less data needs to be copied across the network. The system also creates a new empty disk image to present as the second disk (vdb). Be aware that the second disk is an empty disk with an ephemeral life as it is destroyed when you delete the instance. The compute node attaches to the requested cinder -volume using iSCSI and maps this to the third disk (vdc) as requested. The vCPU and memory resources are provisioned and the instance is booted from the first drive. The instance runs and changes data on the disks indicated in red in the diagram.

There are many possible variations in the details of the scenario, particularly in terms of what the backing storage is and the network protocols used to attach and move storage. One variant worth mentioning here is that the ephemeral storage used for volumes vda and vdb in this example may be backed by network storage rather than local disk. The details are left for later chapters.

### **End State**

Once the instance has served its purpose and is deleted all state is reclaimed, except the persistent volume. The ephemeral storage is purged. Memory and vCPU resources are released. And of course the image has remained unchanged throughout.

Figure 2.3. End state of image and volume after instance exits

**Figure 1.13. End State**

Once you launch a VM in OpenStack, there's something more going on in the background. To understand what's happening behind the dashboard, let's take a deeper dive into OpenStack's VM provisioning. For launching a VM, you can either use the command-line interfaces or the OpenStack dashboard.

## 2. Getting Started Lab

### Table of Contents

|                                                          |    |
|----------------------------------------------------------|----|
| Day 1, 13:30 to 14:45, 15:00 to 17:00 .....              | 41 |
| Getting the Tools and Accounts for Committing Code ..... | 41 |
| Fix a Documentation Bug .....                            | 45 |
| Submit a Documentation Bug .....                         | 49 |
| Create a Branch .....                                    | 49 |
| Optional: Add to the Training Guide Documentation .....  | 51 |

### Day 1, 13:30 to 14:45, 15:00 to 17:00

### Getting the Tools and Accounts for Committing Code



#### Note

First create a GitHub account at [github.com](https://github.com).



#### Note

Check out <https://wiki.openstack.org/wiki/Documentation/HowTo> for more extensive setup instructions.

1. Download and install Git from <http://git-scm.com/downloads>.

2. Create your local repository directory:

```
$ mkdir /Users/username/code/
```

3. **Install SourceTree**

- a. <http://www.sourcetreeapp.com/download/>.
- b. Ignore the Atlassian Bitbucket and Stack setup.
- c. Add your GitHub username and password.
- d. Set your local repository location.

4. **Install an XML editor**

- a. You can download a 30 day trial of Oxygen. The floating licenses donated by OxygenXML have all been handed out. [http://www.oxygenxml.com/download\\_oxygenxml\\_editor.html](http://www.oxygenxml.com/download_oxygenxml_editor.html)
- b. AND/OR PyCharm <http://download.jetbrains.com/python/pycharm-community-3.0.1.dmg>
- c. AND/OR You can use emacs or vi editors.

Here are some great resources on DocBook and Emacs' NXML mode:

- <http://paul.frields.org/2011/02/09/xml-editing-with-emacs/>
- [https://fedoraproject.org/wiki/How\\_to\\_use\\_Emacs\\_for\\_XML\\_editing](https://fedoraproject.org/wiki/How_to_use_Emacs_for_XML_editing)
- <http://infohost.nmt.edu/tcc/help/pubs/nxml/>

If you prefer vi, there are ways to make DocBook editing easier:

- [https://fedoraproject.org/wiki/Editing\\_DocBook\\_with\\_Vi](https://fedoraproject.org/wiki/Editing_DocBook_with_Vi)

## 5. Install Maven

- a. Create the apache-maven directory:

```
# mkdir /usr/local/apache-maven
```

- b. Copy the latest stable binary from <http://maven.apache.org/download.cgi> into /usr/local/apache-maven.

- c. Extract the distribution archive to the directory you wish to install Maven:

```
# cd /usr/local/apache-maven/  
# tar -xvzf apache-maven-x.x.x-bin.tar.gz
```

The apache-maven-x.x.x subdirectory is created from the archive file, where x.x.x is your Maven version.

- d. Add the M2\_HOME environment variable:

```
$ export M2_HOME=/usr/local/apache-maven/apache-maven-x.x.x
```

- e. Add the M2 environment variable:

```
$ export M2=$M2_HOME/bin
```

- f. Optionally, add the MAVEN\_OPTS environment variable to specify JVM properties. Use this environment variable to specify extra options to Maven:

```
$ export MAVEN_OPTS=' -Xms256m -XX:MaxPermSize=1024m -Xmx1024m'
```

- g. Add the M2 environment variable to your path:

```
$ export PATH=$M2:$PATH
```

- h. Make sure that JAVA\_HOME is set to the location of your JDK and that \$JAVA\_HOME/bin is in your PATH environment variable.
- i. Run the mvn command to make sure that Maven is correctly installed:

```
$ mvn --version
```

- 6. Create a Launchpad account: Visit [https://login.launchpad.net/+new\\_account](https://login.launchpad.net/+new_account). After you create this account, the follow-up page is slightly confusing. It does not tell you that you are done. (It gives you the opportunity to change your -password, but you do not have to.)
- 7. Add at least one SSH key to your account profile. To do this, follow the instructions on <https://help.launchpad.net/YourAccount/CreatingAnSSHKeyPair>".
- 8. Join The OpenStack Foundation: Visit <https://www.openstack.org/join>. Among other privileges, this membership enables you to vote in elections and run for elected positions in The OpenStack Project. When you sign up for membership, make sure to give the same e-mail address you will use for code contributions because the primary e-mail address in your foundation profile must match the preferred e-mail that you set later in your Gerrit contact information.
- 9. Validate your Gerrit identity: Add your public key to your gerrit identity by going to <https://review.openstack.org>, click the **Sign In** link, if you are not already logged in. At the top-right corner of the page select settings, then add your public ssh key under **SSH Public Keys**.

The CLA: Every developer and contributor needs to sign the Individual Contributor License agreement. Visit <https://review.openstack.org/> and click the **Sign In** link at the top-right corner of the page. Log in with your Launchpad ID. You can preview the text of the Individual CLA.

- 10. Add your SSH keys to your GitHub account profile (the same one that was used in Launchpad). When you copy and paste the SSH key, include the ssh-rsa algorithm and computer identifier. If this is your first time setting up git and Github, be sure to run these steps in a Terminal window:

```
$ git config --global user.name "Firstname Lastname"
```

```
$ git config --global user.email "your_email@youremail.com"
```

11. Install git-review. If pip is not already installed, run `easy_install pip` as root to install it on a Mac or Ubuntu.

```
# pip install git-review
```

12. Change to the directory:

```
$ cd /Users/username/code
```

13. Clone the openstack-manuals repository:

```
$ git clone http://github.com/openstack/openstack-manuals.git
```

14. Change directory to the pulled repository:

```
$ cd openstack-manuals
```

15. Test the ssh key setup:

```
$ git review -s
```

Then, enter your Launchpad account information.

## Fix a Documentation Bug

- 1.



### Note

For this example, we are going to assume bug 1188522 and change 33713

2. Bring up <https://bugs.launchpad.net/openstack-manuals>
3. Select an unassigned bug that you want to fix. Start with something easy, like a syntax error.

4. Using oXygen, open the `/Users/username/code/openstack-manuals/doc/admin-guide-cloud/bk-admin-guide-cloud.xml` master page for this example. It links together the rest of the material. Find the page with the bug. Open the page that is referenced in the bug description by selecting the content in the author view. Verify you have the correct page by visually inspecting the html page and the xml page.
5. In the shell,  

```
$ cd /Users/username/code/openstack-manuals/doc/admin-guide-cloud/
```
6. Verify that you are on master:  

```
$ git checkout master
```
7. Create your working branch off master:  

```
$ git checkout -b bug/1188522
```
8. Verify that you have the branch open through SourceTree
9. Correct the bug through oXygen. Toggle back and forth through the different views at the bottom of the editor.
10. After you fix the bug, run maven to verify that the documentation builds successfully. To build a specific guide, look for a `pom.xml` file within a subdirectory, switch to that directory, then run the **mvn** command in that directory:  

```
$ mvn clean generate-sources
```
11. Verify that the HTML page reflects your changes properly. You can open the file from the command line by using the **open** command  

```
$ open target/docbkx/webhelp/local/openstack-training/index.html
```
12. Add the changes:



```
$ git add .
```

13. Commit the changes:

```
$ git commit -a -m "Removed reference to volume scheduler in the computer scheduler config  
and admin pages, bug 1188522"
```

14. Build committed changes locally by using **tox**. As part of the review process, Jenkins runs gating scripts to check that the patch is fine. Locally, you can use the **tox** tool to run the same checks and ensure that a patch works. Install the tox package and run it from the top level directory which has the `tox.ini` file.

```
# pip install tox  
$ tox
```

Jenkins runs the following four checks. You can run them individually:

- a. Niceness tests (for example, to see extra whitespaces). Verify that the niceness check succeeds.

```
$ tox -e checkniceness
```

- b. Syntax checks. Verify that the syntax check succeeds.

```
$ tox -e checksyntax
```

- c. Check that no deleted files are referenced. Verify that the check succeeds.

```
$ tox -e checkdeletions
```

- d. Build the manuals. It also generates a directory `publish-docs/` that contains the built files for inspection. You can also use `doc/local-files.html` for looking at the manuals. Verify that the build succeeds.

```
$ tox -e checkbuild
```

15. Submit the bug fix to Gerrit:

```
$ git review
```

16. Track the Gerrit review process at <https://review.openstack.org/#/c/33713>. Follow and respond inline to the Code Review requests and comments.
17. Your change will be tested, track the Jenkins testing process at <https://jenkins.openstack.org>
18. If your change is rejected, complete the following steps:
  - a. Respond to the inline comments if any.
  - b. Update the status to work in progress.
  - c. Checkout the patch from the Gerrit change review:

```
$ git review -d 33713
```
  - d. Follow the recommended tweaks to the files.
  - e. Rerun:

```
$ mvn clean generate-sources
```
  - f. Add your additional changes to the change log:

```
$ git commit -a --amend
```
  - g. Final commit:

```
$ git review
```
  - h. Update the Jenkins status to change completed.

19. Follow the jenkins build progress at <https://jenkins.openstack.org/view/Openstack-manuals/> . Note if the build process fails, the online documentation will not reflect your bug fix.

## Submit a Documentation Bug

1. Bring up <https://bugs.launchpad.net/openstack-manuals/+filebug>.
2. Give your bug a descriptive name.
3. Verify if asked that it is not a duplicate.
4. Add some more detail into the description field.
5. Once submitted, select the assigned to pane and select "assign to me" or "sarob".
6. Follow the instructions for fixing a bug in the Fix a Documentation Bug section.

## Create a Branch



### Note

This section uses the submission of this training material as the example.

1. Create a bp/training-manuals branch:  

```
$ git checkout -b bp/training-manuals
```
2. From the openstack-manuals repository, use the template `user-story-includes-template.xml` as the starting point for your user story. File `bk001-ch003-associate-general.xml` has at least one other included user story that you can use for additional help.
3. Include the user story xml file into the `bk001-ch003-associate-general.xml` file. Follow the syntax of the existing `xi:include` statements.

4. When your editing is completed. Double check Oxygen doesn't have any errors you are not expecting.
5. Run maven locally to verify the build will run without errors. Look for a `pom.xml` file within a subdirectory, switch to that directory, then run the **mvn** command in that directory:

```
$ mvn clean generate-sources
```

6. Add your changes into git:

```
$ git add .
```

7. Commit the changes with good syntax. After entering the commit command, VI syntax applies, use "i" to insert and Esc to break out. ":wq" to write and quit.

```
$ git commit -a
      my very short summary
more details go here. A few sentences would be nice.
blueprint training-manuals
```

8. Build committed changes locally using **tox**. As part of the review process, Jenkins runs gating scripts to check that the patch is fine. Locally, you can use the **tox** tool to run the same checks and ensure that a patch works. Install the tox package and run it from the top level directory which has the `tox.ini` file.

```
# pip install tox
$ tox
```

9. Submit your patch for review:

```
$ git review
```

10. One last step. Go to the review page listed after you submitted your review and add the training core team as reviewers; Sean Roberts and Colin McNamara.
11. More details on branching can be found here under [Gerrit Workflow](#) and the [Git docs](#).

## Optional: Add to the Training Guide Documentation

1. **Getting Accounts and Tools:** We cannot do this without operators and developers using and creating the content. Anyone can contribute content. You will need the tools to get started. Go to the [Getting Tools and Accounts](#) page.
2. **Pick a Card:** Once you have your tools ready to go, you can assign some work to yourself. Go to the [Training Trello/KanBan storyboard](#) and assign a card / user story from the Sprint Backlog to yourself. If you do not have a Trello account, no problem, just create one. Email seanrob@yahoo-inc.com and you will have access. Move the card from the Sprint Backlog to Doing.
3. **Create the Content:** Each card / user story from the KanBan story board will be a separate chunk of content you will add to the openstack-manuals repository openstack-training sub-project.
4. Open the file `st-training-guides.xml` with your XML editor. All the content starts with the set file `st-training-guides.xml`. The XML structure follows the hierarchy Set -> Book -> Chapter -> Section. The `st-training-guides.xml` file holds the set level. Notice the set file uses `xi:include` statements to include the books. We want to open the associate book. Open the associate book and you will see the chapter include statements. These are the chapters that make up the Associate Training Guide book.
5. Create a branch by using the card number as **associate-card-XXX** where XXX is the card number. Review [Creating a Branch](#) again for instructions on how to complete the branch merge.
6. Copy the `user-story-includes-template.xml` to `associate-card-XXX.xml`.
7. Open the `bk001-ch003-associate-general.xml` file and add `<xi:include href="associate-card-XXX.xml">`.
8. Side by side, open **associate-card-XXX.xml** with your XML editor and open the *Ubuntu 12.04 Install Guide* with your HTML browser.
9. Find the HTML content to include. Find the XML file that matches the HTML. Include the whole page using a simple href like `<xi:include href="associate-card-XXX.xml">` or include a section using xpath like

`<xi:include href="../../basic-install/src/basic-install_controller-common.xml" xpointer="xmlns(db=http://docbook.org/ns/docbook) xpath(//*[@xml:id = 'controller-os'])">` . Review the `user-story-  
includes-template.xml` file for the whole syntax.

10. Copy in other content sources including the Aptira content, a description of what the section aims to teach, diagrams, and quizzes. If you include content from another source like Aptira content, add a paragraph that references the file and/or HTTP address from where the content came.
11. Verify the code is good by running **`mvn clean generate-sources`** and by reviewing the local HTML in <file:///Users/username/code/openstack-manuals/doc/training-guides/target/docbkx/webhelp/training-guides/content/>.
12. Merge the branch.
13. Move the card from Doing to Done.

## 3. Getting Started Quiz

### Table of Contents

|                             |    |
|-----------------------------|----|
| Day 1, 16:40 to 17:00 ..... | 53 |
|-----------------------------|----|

### Day 1, 16:40 to 17:00





## 4. Developer APIs in Depth

### Table of Contents

Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30 ..... 55

**Day 2 to 4, 09:00 to 11:00, 11:15 to 12:30**



## 5. Developer APIs in Depth Lab Day Two

### Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 2, 13:30 to 14:45, 15:00 to 16:30 ..... | 57 |
|---------------------------------------------|----|

### Day 2, 13:30 to 14:45, 15:00 to 16:30

#### Pre-Requisites.

1. [Git Basics](#)
2. [Gerrit Basics](#)
3. [Jenkins](#)



## 6. Developer APIs in Depth Day Two Quiz

### Table of Contents

|                             |    |
|-----------------------------|----|
| Day 2, 16:40 to 17:00 ..... | 59 |
|-----------------------------|----|

### Day 2, 16:40 to 17:00



## 7. Developer APIs in Depth Lab Day Three

### Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 3, 13:30 to 14:45, 15:00 to 16:30 ..... | 61 |
|---------------------------------------------|----|

### Day 3, 13:30 to 14:45, 15:00 to 16:30





## 8. Developer APIs in Depth Day Three Quiz

### Table of Contents

|                             |    |
|-----------------------------|----|
| Day 3, 16:40 to 17:00 ..... | 63 |
|-----------------------------|----|

### Day 3, 16:40 to 17:00



## 9. Developer How To Participate Lab Day Four

### Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 4, 13:30 to 14:45, 15:00 to 16:30 ..... | 65 |
|---------------------------------------------|----|

### Day 4, 13:30 to 14:45, 15:00 to 16:30



# 10. Developer APIs in Depth Day Four Quiz

## Table of Contents

|                             |    |
|-----------------------------|----|
| Day 4, 16:40 to 17:00 ..... | 67 |
|-----------------------------|----|

## Day 4, 16:40 to 17:00



# 11. Developer How To Participate

## Table of Contents

Day 5 to 9, 09:00 to 11:00, 11:15 to 12:30 ..... 69

## Day 5 to 9, 09:00 to 11:00, 11:15 to 12:30





## 12. Developer How To Participate Lab Day Five

### Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 5, 13:30 to 14:45, 15:00 to 16:30 ..... | 71 |
|---------------------------------------------|----|

### Day 5, 13:30 to 14:45, 15:00 to 16:30



## 13. Developer How To Participate Day Five Quiz

### Table of Contents

|                             |    |
|-----------------------------|----|
| Day 5, 16:40 to 17:00 ..... | 73 |
|-----------------------------|----|

### Day 5, 16:40 to 17:00



## 14. Developer How To Participate Lab Day Six

### Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 6, 13:30 to 14:45, 15:00 to 16:30 ..... | 75 |
|---------------------------------------------|----|

### Day 6, 13:30 to 14:45, 15:00 to 16:30



# 15. Developer How To Participate Day Six Quiz

## Table of Contents

|                             |    |
|-----------------------------|----|
| Day 6, 16:40 to 17:00 ..... | 77 |
|-----------------------------|----|

## Day 6, 16:40 to 17:00





# 16. Developer How To Participate Lab Day Seven

## Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 7, 13:30 to 14:45, 15:00 to 16:30 ..... | 79 |
|---------------------------------------------|----|

## Day 7, 13:30 to 14:45, 15:00 to 16:30



# 17. Developer How To Participate Day Seven Quiz

## Table of Contents

|                             |    |
|-----------------------------|----|
| Day 7, 16:40 to 17:00 ..... | 81 |
|-----------------------------|----|

## Day 7, 16:40 to 17:00



# 18. Developer How To Participate Lab Day Eight

## Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 8, 13:30 to 14:45, 15:00 to 16:30 ..... | 83 |
|---------------------------------------------|----|

## Day 8, 13:30 to 14:45, 15:00 to 16:30



# 19. Developer How To Participate Day Eight Quiz

## Table of Contents

|                             |    |
|-----------------------------|----|
| Day 8, 16:40 to 17:00 ..... | 85 |
|-----------------------------|----|

## Day 8, 16:40 to 17:00





## 20. Developer How To Participate Lab Day Nine

### Table of Contents

|                                             |    |
|---------------------------------------------|----|
| Day 9, 13:30 to 14:45, 15:00 to 16:30 ..... | 87 |
|---------------------------------------------|----|

### Day 9, 13:30 to 14:45, 15:00 to 16:30



# 21. Developer How To Participate Day Nine Quiz

## Table of Contents

|                             |    |
|-----------------------------|----|
| Day 9, 16:40 to 17:00 ..... | 89 |
|-----------------------------|----|

## Day 9, 16:40 to 17:00



# 22. Assessment

## Table of Contents

Day 10, 9:00 to 11:00, 11:15 to 12:30, hands on lab 13:30 to 14:45, 15:00 to 17:00 ..... 91

Questions ..... 91

## Day 10, 9:00 to 11:00, 11:15 to 12:30, hands on lab 13:30 to 14:45, 15:00 to 17:00

## Questions

Table 22.1. Assessment Question 1

|                  |
|------------------|
| Task             |
| Configure a .... |

Table 22.2. Assessment Question 2

|                  |
|------------------|
| Task             |
| Configure a .... |



## 23. Developer How To Participate Bootcamp

### Table of Contents

|                                          |    |
|------------------------------------------|----|
| One Day with Focus on Contribution ..... | 93 |
| Overview .....                           | 93 |
| Morning Classroom 10:00 to 11:15 .....   | 94 |
| Morning Lab 11:30 to 12:30 .....         | 95 |
| Morning Quiz 12:30 to 12:50 .....        | 95 |
| Afternoon Classroom 13:30 to 14:45 ..... | 95 |
| Afternoon Lab 15:00 to 17:00 .....       | 96 |
| Afternoon Quiz 17:00 to 17:20 .....      | 96 |

### One Day with Focus on Contribution

#### Overview

Training will take 6 hours with labs and quizzes.

##### Prerequisites

1. Some knowledge of Python and/or Perl
2. Editor on a self-supplied laptop with either Eclipse with pydev, vim, emacs, or pycharm
3. Run through the Operator Training Guide Getting Started Lab in full. This will walk each trainee through installing the accounts and tools required for the bootcamp.

## Morning Classroom 10:00 to 11:15

Understanding the local tools in-depth

- Pycharm editor
- Git
- Sourcetree
- Maven

Understanding the remote tools in-depth

- git-review
- github
- gerrit
- jenkins
- gearman
- jeepy
- zuul
- launchpad

CI Pipeline Workflow Overview

- Understanding the submission process in-depth



- Review submission syntax
- Gerrit etiquette
- Resubmission

## Morning Lab 11:30 to 12:30

TBD

## Morning Quiz 12:30 to 12:50

Online moodle test for theory, bit of syntax and terms, retake until 100%

Content TBD

## Afternoon Classroom 13:30 to 14:45

Understanding the CI Pipeline in-depth

- Gerrit Workflow
- Common jenkins tests
- Reviewing and understanding zuul
- Understanding jenkins output
- Understanding jenkins system manual (devstack)
- automated (tempest) integration tests

## **Afternoon Lab 15:00 to 17:00**

TBD

## **Afternoon Quiz 17:00 to 17:20**

Online moodle test for theory, bit of syntax and terms, retake until 100%

Content TBD

## **Architect Training Guide**



# Table of Contents

|                                               |   |
|-----------------------------------------------|---|
| 1. Architect Training Guide Coming Soon ..... | 1 |
|-----------------------------------------------|---|



# 1. Architect Training Guide Coming Soon

TBD

