A Simple Guide to Install OpenStack Icehouse on CentOS

A two node Architecture

Techglimpse.com

Contents

Introduction

The word "OpenStack" is quite popular these days. You might have heard this word, even if you are not working in the area of Cloud computing. I have been working in Grid computing domain for the past 10 years and never understood the fuss behind OpenStack. But this buzz word kept tapping my ears wherever I go – Seriously, in my office, conferences and popular websites that kept promoting OpenStack. Finally, I was pushed (Yeah, forcefully) into the world of OpenStack and these big questions took a toll on me – What is Openstack, how it's implemented and how does it work? Well, the only way to understand OpenStack is to start with an installation.

At first sight, getting started with OpenStack looked easy, but it's not. Yes, the official website of OpenStack has a good documentation, but the real problem I faced was, the terminologies used – Horizon, Glance, Keystone, Nova, Neutron, Swift, Heat, Cinder, Ceilometer….Uff, why hell they are named like that?. And then how to install it? There are several other terms linked to the installation – RDO, Packstack, Devstack, different architectures (one node, two node and multinode), legacy networking etc…Seriously, I had to run back to Google to understand all those (particularly this guy has done an amazing job - Click Here)[1]. Well, if you ask me to write about my experience with OpenStack, then this article will run down to 100 of pages. So I'll stop here and lets quickly go to the topics.

What is OpenStack?

OpenStack is an open source Cloud computing platform that provides Infrastructure as a Service (IaaS). If you are a Grid computing expert, then OpenStack is something similar to popular Grid middleware such as Globus Toolkit, Glite, Oracle Grid Engine, Unicore etc… OpenStack is basically a middleware that allows you to manage cloud computing resources more efficiently and of course, effectively.

OpenStack services

Horizon: A dashboard service that provides a web portal for interacting and managing the underlying services, virtual instances, users and network.
Nova: A compute service that helps in managing compute instances – which includes spawning, scheduling, booting and terminating a virtual machine.
Keystone: An Identity service that provides authentication and authorization.

Neutron: A network service that allows you to create and manage network.

Glance: An Image service that helps to store and fetch virtual machine images.

Database service: Provides database as a service.

Swift and Cinder: Provides storage as a service.

Telemetry: A service that helps you to manage billing and benchmarking.

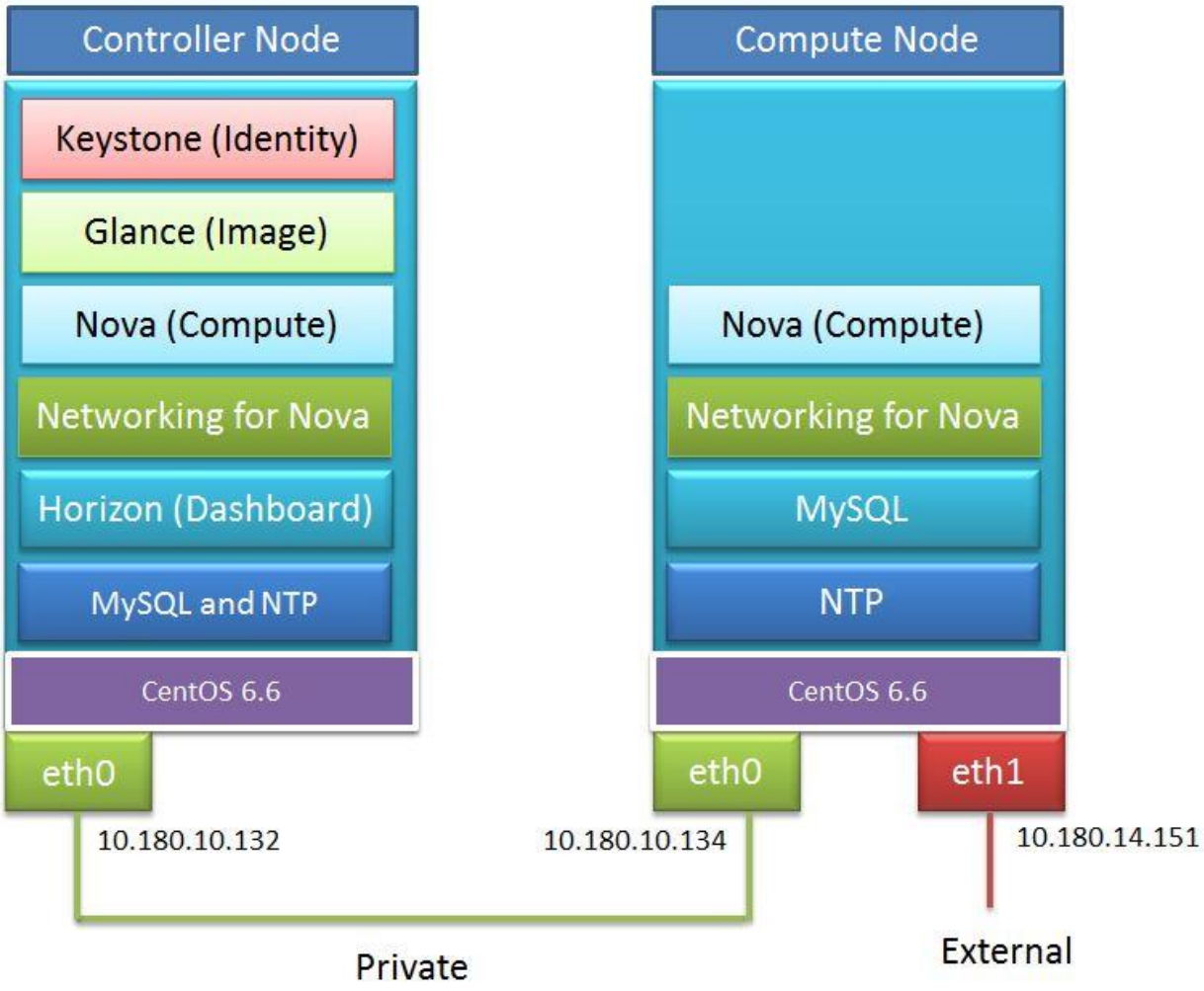
There are plenty other services…read more at http://openstack.org.[2]

Architecture

If you are a beginner, then the ideal way is to start with **All-In-One Single Machine** installation of OpenStack. If you have a little knowledge on OpenStack installation with All-In-One Single Machine and its configurations, now you can better go to two-node architecture – one node called as 'Controller' and the other as 'Compute'. In this tutorial, we are going to install OpenStack Icehouse version on CentOS 6.6 operating system. The services to be installed on controller node are Keystone, Glance, Nova, Networking (legacy nova-networking), Horizon and the compute node will have Nova and Networking (legacy nova-networking).

Note: OpenStack beginner? It's advised to start with Nova legacy networking. Understanding and configuring "Neutron" is a big ask at this stage. However, you should consider migrating from legacy networking to Neutron at a later stage, which you can think of as a three-node architecture where, controller, compute and network management would be on each node separately.

Below is the diagram that depicts my test bed. All of my installation steps are based on the below architecture.



OpenStack Icehouse – Two node Architecture

Note: Since we are going to use Legacy Nova-network, we need only one interface in controller node for management purpose and two interfaces on compute node (one for management that falls in the same network of controller node and the other interface is for external, which will be used for VM communications).

Well, we are good to go ahead with the installation steps.

How to install and Configure OpenStack Icehouse on CentOS?

I'll use, **'controller-hostname'** as the hostname and '10.180.10.132' (Management IP) for my Controller node and **"compute-hostname"** as the hostname and '10.180.10.134' (Management IP) for Compute node. The external IP of compute node is going to be '10.180.14.151'. Refer to the above architecture diagram.

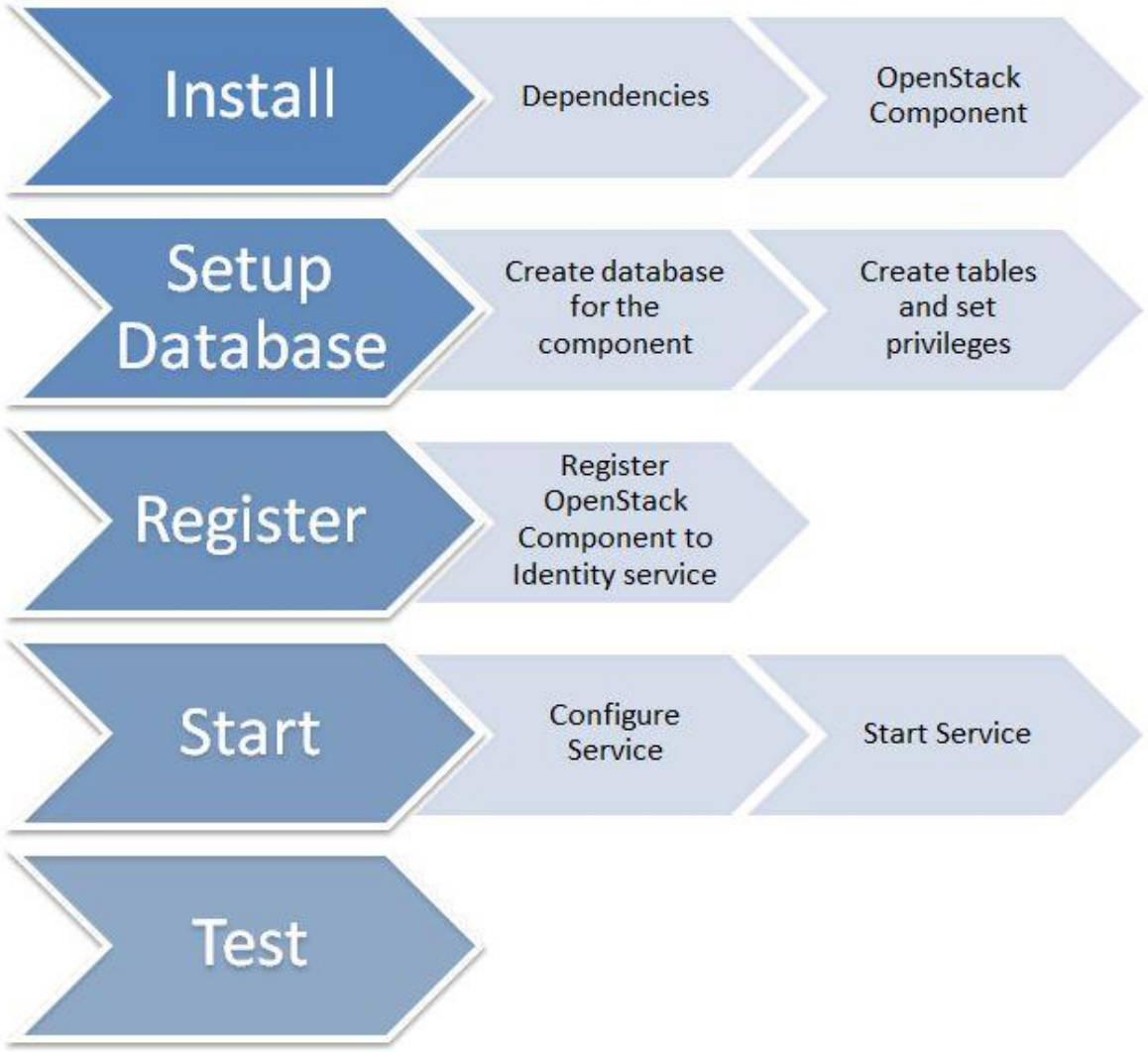
Note: *Remember to replace those with the corresponding FQDN or IP addresses.*
Note: *During the installation, wherever you find 'setpassword', replace it with the corresponding service password.*

I'll be using 'YUM' for most part of the installation. So make sure the machines are connected to the internet and yum is configured. In case, if you have problem in configuring yum, then the below guides will be of great help.

- How to configure Yum repository on CentOS - Click Here[3]
- Yum commands that you must know - Click Here[4]

The output of the few commands has been truncated to avoid confusion and to reduce the length of this tutorial.

Installation of any component (or service) in OpenStack involves below steps:



OpenStack Icehouse Installation and Configuration Flow

Update OS on both Controller and Compute node

It's always advised to upgrade the operating system before installing any component. It will resolve lot of package dependency headaches.

```
[root@controller-hostname ]# yum update
[root@compute-hostname ]# yum update
```

Install and Configure NTP on both Controller and Compute node

What is NTP?

NTP stands for Network Time Protocol, which actually send time signals over the network to synchronize NTP clients (the description is enough for now and you can find better explanation at ntp.org). NTP can be installed using '**yum**' command.

Install NTP

```
#yum install ntp
.....
Installed:
ntp.x86_64 0:4.2.6p5-2.el6.centos
Dependency Installed:
ntpdate.x86_64 0:4.2.6p5-2.el6.centos
Complete!
```

Configure 'ntpd' to start during the boot

```
# chkconfig ntpd on
```

Update NTP

```
# ntpdate pool.ntp.org
```

Start 'ntpd' daemon

```
# service ntpd start
 or
# /etc/init.d/ntpd start
```

NTPSTAT

You can also run ntpstat after configuring 'server' in /etc/ntp.conf.

```
# ntpstat
unsynchronised
time server re-starting
polling server every 8 s
```

Configure NTP

```
# vim /etc/ntp.conf
```

and add the below lines…

```
server 0.centos.pool.ntp.org
server 1.centos.pool.ntp.org
server 2.centos.pool.ntp.org
server 3.centos.pool.ntp.org
```

By default, the public NTP pool servers (for CentOS) would have been added to the ntp.conf file. The 'server' attribute can be used to add your organization's NTP server as well.

For example,

```
server ntp.yourorganization.org
```

That's it!

Install MySQL database on both Controller and Compute node

Note: MySQL is now called as 'Mariadb'.

```
[root@controller-hostname ]# yum install mysql mysql-server MySQL-python
[root@compute-hostname ]# yum install mysql mysql-server MySQL-python
```

You will have to look at '/etc/my.cnf' to ensure that the MySQL service will bind on Management IP address of both the controller and compute nodes.

On Controller node:

```
[root@controller-hostname ]# vi /etc/my.cnf
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
bind-address=10.180.10.132
symbolic-links=0
[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

On compute node:

```
[root@compute-hostname ]# vi /etc/my.cnf
[mysqld]
datadir=/var/lib/mysql
socket=/var/lib/mysql/mysql.sock
user=mysql
bind-address=10.180.10.134
symbolic-links=0
[mysqld_safe]
log-error=/var/log/mysqld.log
pid-file=/var/run/mysqld/mysqld.pid
```

Start the mysql server:

```
[root@controller-hostname ~]# service mysqld start
[root@compute-hostname ~]# service mysqld start
```

Setup MySQL…

```
[root@controller-hostname ~]# mysql_install_db
[root@compute-hostname ~]# mysql_install_db
```

It's always advised to run '*mysql_secure_installation*' after installing MySQL.

```
[root@controller-hostname ~]# mysql_secure_installation
[root@compute-hostname ~]# mysql_secure_installation
```

If you are not sure what mysql_secure_installation does, then here's a quick guide that explains about it in detail - Secure MySQL – Click Here[5]

Install Base OpenStack Icehouse packages on both Controller and Compute node

Setup yum repository and install base Icehouse packages. This step has to be performed on both the controller and compute nodes.

On Controller node:

```
[root@controller-hostname ~]# yum install yum-plugin-priorities
[root@controller-hostname~]# yum install http://repos.fedorapeople.org/repos/openstack/openstack-icehouse/rdo-release-icehouse-4.noarch.rpm
[root@controller-hostname ~]# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-RDO-Icehouse
[root@controller-hostname~]# yum -y install http://fedora.cu.be/epel/7/x86_64/e/epel-release-7-2.noarch.rpm
[root@controller-hostname~]# yum install openstack-utils openstack-selinux
```

On compute node:

```
[root@compute-hostname ~]# yum install yum-plugin-priorities
[root@compute-hostname~]# yum install http://repos.fedorapeople.org/repos/openstack/openstack-icehouse/rdo-release-icehouse-4.noarch.rpm
[root@compute-hostname ~]# rpm --import /etc/pki/rpm-gpg/RPM-GPG-KEY-RDO-Icehouse
[root@compute-hostname~]#yum -y install http://fedora.cu.be/epel/7/x86_64/e/epel-release-7-2.noarch.rpm
[root@compute-hostname ~]# yum install openstack-utils openstack-selinux
```

Install Message Broker Service on Controller node

OpenStack services needs a message broker for its communication. Currently OpenStack supports – Rabbit-mq and Apache Qpid. In this tutorial, we'll be installing Qpid-server.

Apache Qpid allows services to send and receive messages over AMQP (Advanced Message Queuing Protocol).

```
[root@controller-hostname ~]# yum install qpid-cpp-server
```

Configure qpid-cpp-server to work without the need of authentication (This will ease up our installation process. But you should enable authentication for security reasons)

```
[root@controller-hostname ~]# echo "auth=no">/etc/qpid/qpidd.conf
```

Start the message broker…

```
[root@controller-hostname ~]# service qpidd start
```

Installing and Configuring Keystone (Identity Service) on Controller node

As I told earlier, the Keystone service plays a major role in OpenStack – providing authentication and authorization. It also means that, every service of OpenStack (including identity service) has to be registered with KeyStone.

Install keystone packages…

```
[root@controller-hostname ~]# yum install openstack-keystone python-keystoneclient
```

Create a database for keystone, so that the service can store its state and data…

```
[root@controller-hostname ~]# mysql -u root -p
mysql> CREATE DATABASE keystone;
mysql> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'controller-hostname' IDENTIFIED BY
'setpassword';
mysql> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY 'setpassword';
```

Note: *Remember to replace 'setpassword' with your own password for keystone.*

Create tables for keystone database…

```
[root@controller-hostname ~]# su -s /bin/sh -c "keystone-manage db_sync" keystone
```

Now, tell keystone that, this is what you should use to make a database connection

```
[root@controller-hostname ~]# openstack-config --set /etc/keystone/keystone.conf database connection
mysql://keystone:setpassword@controller-hostname/keystone
```

Note: *Remember to replace 'setpassword' with the suitable password of 'keystone'@mysql*

Create authentication token to authenticate other services with keystone. To do that, we'll use openssl to generate a random HEX value and store it in a variable called 'ADMIN_TOKEN'

```
[root@controller-hostname ~]# ADMIN_TOKEN=$(openssl rand -hex 10)
[root@controller-hostname ~]# echo $ADMIN_TOKEN
e9393f7ac1886f0c1a20
```

Note: *Make a copy of the generated token, as you need this later (not required, if you are not going to kill your terminal session in the mid-way)*

Configure keystone to use the authentication token generated in the previous step…

```
[root@controller-hostname ~]# openstack-config --set /etc/keystone/keystone.conf DEFAULT admin_token
$ADMIN_TOKEN
```

Configure keystone to provide PKI based authentication tokens..

```
[root@controller-hostname ~]# keystone-manage pki_setup --keystone-user keystone --keystone-group
keystone
```

Set ownership and permission for 'keystone' user…

```
[root@controller-hostname ~]# chown -R keystone:keystone /etc/keystone/ssl
[root@controller-hostname ~]# chmod -R o-rwx /etc/keystone/ssl
```

All done for keystone, lets start the service…

```
[root@controller-hostname ~]# service openstack-keystone start
```

Make sure the service is really running by checking its status

```
[root@controller-hostname ~]# service openstack-keystone status
keystone (pid 10245) is running...
```

In case, after starting, if the service is dead, then you should fix that issue at this point itself.
Set the service to start automatically during system boot…

```
[root@controller-hostname ~]# chkconfig openstack-keystone on
```

Automatically clear expired keystone tokens…

There might be many expired tokens in the keystone during its operation, thus increasing the size of the keystone database. It's a good idea to clear those at regular interval. To do that, we'll set cron jobs as below:

```
[root@controller-hostname ~]# echo '@hourly /usr/bin/keystone-manage token_flush
>/var/log/keystone/keystone-tokenflush.log 2&>&1' >> /var/spool/cron/keystone
[root@controller-hostname ~]# crontab -l -u keystone
@hourly /usr/bin/keystone-manage token_flush > /var/log/keystone/keystone-tokenflush.log 2&>&1
```

Create users, tenant and roles for keystone

Before executing keystone commands, you should set certain environment variables. To do that, create a file named 'admin-openrc.sh' and set environment variables as below:

```
[root@controller-hostname ~]# vi admin-openrc.sh
export OS_USERNAME=admin
export OS_PASSWORD=setpassword
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://controller-hostname:35357/v2.0
[root@controller-hostname ~]# source admin-openrc.sh
```

Lets create a user…

```
[root@controller-hostname ~]# keystone user-create --name=admin --pass=setpassword --
email=keystone@controller-hostname
```

Create a role…

```
[root@controller-hostname ~]# keystone role-create --name=admin
```

Create a tenant…

```
[root@controller-hostname ~]# keystone tenant-create --name=admin --description="Admin Tenant"
```

We'll have to link the user with the role and tenant created in previous steps…

```
[root@controller-hostname ~]# keystone user-role-add --user=admin --role=admin --tenant=admin
```

Register the keystone (identity service) to the KeyStone service

```
[root@controller-hostname ~]# keystone tenant-create --name=service --description="Service Tenant"
[root@controller-hostname ~]# keystone service-create --name=keystone --type=identity --
description="OpenStack Identity"
[root@controller-hostname ~]# keystone endpoint-create --service-id=$(keystone service-list | awk '/
compute / {print $2}') --publicurl=http://controller-hostame:8774/v2/%\(tenant_id\)s --
internalurl=http://controller-hostame:8774/v2/%\(tenant_id\)s --adminurl=http://controller-
hostame:8774/v2/%\(tenant_id\)s
```

That's it; the KeyStone configuration is done. In case, if you face any problem during keystone installation and configuration, then jump to **Appendix 1** to find out the Common keystone service errors you may face and its solutions.

Install Glance (Image Service) on Controller node

The image service will be installed on the Controller node to host all the images that are to be used to boot VMs on the Compute node.

```
[root@controller-hostname ~]# yum install openstack-glance python-glanceclient
```

Setup database 'glance'…

```
[root@controller-hostname ~]# mysql -u root -p
mysql> CREATE DATABASE glance;
mysql> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'controller-hostname' IDENTIFIED BY
'setpassword';
mysql > GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' IDENTIFIED BY 'setpassword';
```

Note: *Remember to change 'setpassword'*

Create tables for 'glance' database…

```
[root@controller-hostname ~]# su -s /bin/sh -c "glance-manage db_sync" glance
```

Configure glance service to read database credentials…

```
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf database connection
mysql://glance:setpassword@controller-hostname/glance
[root@controller-hostname ~]# glance-registry.conf database connection
mysql://glance:setpassword@controller-hostname/glance
```

Create Glance user in KeyStone…

```
[root@controller-hostname ~]# keystone user-create --name=glance --pass=setpassword --
email=glance@controller-hostname
```

Note: *Remember to change 'setpassword'*

Add glance user to the role 'admin'

```
[root@controller-hostname ~]# keystone user-role-add --user=glance --tenant=service --role=admin
```

Configure authentication for Glance…

```
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken
auth_uri http://controller-hostname:5000
```

```
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken auth_host controller-hostname
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken auth_port 35357
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken auth_protocol http
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken admin_tenant_name service
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken admin_user glance
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf keystone_authtoken admin_password setpassword
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-api.conf paste_deploy flavor keystone
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken auth_uri http://controller-hostname:5000
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken auth_host controller-hostname
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken auth_port 35357
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken auth_protocol http
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken admin_tenant_name service
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken admin_user glance
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf keystone_authtoken admin_password setpassword
[root@controller-hostname ~]# openstack-config --set /etc/glance/glance-registry.conf paste_deploy flavor keystone
```

Note: *Remember to change 'setpassword'*

Register Glance service to KeyStone…

```
[root@controller-hostname ~]# keystone service-create --name=glance --type=image --description="OpenStack Image Service"
```

```
[root@controller-hostname ~]# keystone --os-token=fd465d38e342ddc68be3 --os-endpoint=http://controller-hostname:35357/v2.0 endpoint-create --service-id=$(keystone service-list | awk '/ image / {print $2}') --publicurl=http://controller-hostname:9292 --internalurl=http://controller-hostname:9292 --adminurl=http://controller-hostname:9292
```

Start glance services…

```
[root@controller-hostname ~]# service openstack-glance-api start
[root@controller-hostname ~]# service openstack-glance-registry start
[root@controller-hostname ~]# chkconfig openstack-glance-api on
[root@controller-hostname ~]# chkconfig openstack-glance-registry on
```

Note: *If any of the service fails to start, then check the log files under /var/log/glance.*

That's it; the Glance Installation and configuration is done. In case, if you face any problem during Glance installation and configuration or during service start-up, then jump to **Appendix 2** to find out the Common errors you may face during Glance Image Service installation, configuration and the service start-up and its solutions.

Test Glance service…

We shall now download Cirros image to test the glance service. Cirros is a tiny linux operating system which sizes in few MBs, so it's quite easy to download and add it to the Glance service.

```
[root@controller-hostname ~]# wget http://cdn.download.cirros-cloud.net/0.3.2/cirros-0.3.2-x86_64-disk.img
[root@controller-hostname ~]# glance image-create --name=cirros --disk-format=qcow2 --container-format=bare --is-public=true < cirros-0.3.2-x86_64-disk.img
```

Add CentOS image to the glance service

```
[root@controller-hostname ~]# wget http://cloud.centos.org/centos/6/images/CentOS-6-x86_64-GenericCloud-20141129_01.qcow2.xz
```

Note: xz is a general purpose data compression utility that provides high compression ratio. That's the reason, the size of .xz file is less and easy to download.

If you are not sure how to uncompress .xz files, then here's a tutorial to help you - How to uncompress .xz file format in Linux using tar and xz utilities?[6]

After uncompressing .xz image, add the image to Glance service.

```
[root@controller-hostname ~]# glance image-create --name 'Centos-6' --disk-format qcow2 --container-format bare --is-public=true < CentOS-6-x86_64-GenericCloud-20141129_01.qcow2
```

View the list of images added to the Glance services…

```
[root@controller-hostname ~]# glance image-show "cirros"

+------------------+--------------------------------------+
| Property | Value |
+------------------+--------------------------------------+
| checksum | 64d7c1cd2b6f60c92c14662941cb7913 |
| container_format | bare |
| created_at | 2015-02-06T12:47:39 |
| deleted | False |
| disk_format | qcow2 |
| id | a338631b-3bb8-43ba-a700-d4648c040a05 |
| is_public | True |
| min_disk | 0 |
| min_ram | 0 |
| name | cirros |
| owner | e7c679bc36ec4c298cf68ecf6d49c1b3 |
| protected | False |
| size | 13167616 |
| status | active |
| updated_at | 2015-02-06T12:47:40 |
+------------------+--------------------------------------+
[root@controller-hostname ~]# glance image-show "Centos-6"
+------------------+--------------------------------------+
| Property | Value |
+------------------+--------------------------------------+
| checksum | 62ac2565e3527377860361f57639f334 |
| container_format | bare |
| created_at | 2015-02-13T06:18:52 |
| deleted | False |
```

```
| disk_format | qcow2 |
| id | ac7ffb6d-1594-4a4c-94e7-9d8e70a120a8 |
| is_public | True |
| min_disk | 0 |
| min_ram | 0 |
| name | Centos-6 |
| owner | e7c679bc36ec4c298cf68ecf6d49c1b3 |
| protected | False |
| size | 1151533056 |
| status | active |
| updated_at | 2015-02-13T06:20:29 |
+------------------+--------------------------------------+
```

Install Nova Service on Controller node

Let us now move on to install Nova on Controller node…

```
[root@controller-hostname ~]# yum install openstack-nova-api openstack-nova-cert openstack-nova-conductor openstack-nova-console openstack-nova-novncproxy openstack-nova-scheduler python-novaclient
```

Create 'nova' database…

```
[root@controller-hostname ~]# mysql -u root -p
mysql> CREATE DATABASE nova;
mysql> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'controller-hostname' IDENTIFIED BY 'setpassword';
mysql> GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'%' IDENTIFIED BY 'setpassword';
```

Note: *Remember to change 'setpassword'*

Tell Nova to use corresponding database and provide credentials for it…

```
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf database connection mysql://nova:setpassword@controller-hostname/nova
```

Connect Message broker (Qpid) to Nova…

```
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf DEFAULT rpc_backend qpid
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf DEFAULT qpid_hostname controller-hostname
```

Create tables for 'nova' database

```
[root@controller-hostname ~]# su -s /bin/sh -c "nova-manage db sync" nova
```

Create user for nova in KeyStone

```
[root@controller-hostname ~]# keystone user-create --name=nova --pass=setpassword --email=nova@controller-hostname
[root@controller-hostname ~]# keystone user-role-add --user=nova --tenant=service --role=admin
```

Configure nova authentication with KeyStone

```
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf DEFAULT auth_strategy
keystone
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_uri
http://controller-hostname:5000
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_host
controller-hostname
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken
auth_protocol http
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_port
35357
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_user
nova
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken
admin_tenant_name service
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf keystone_authtoken
admin_password setpassword
```

Register nova with KeyStone

```
[root@controller-hostname ~]# keystone service-create --name=nova --type=compute --
description="OpenStack Compute"
[root@controller-hostname ~]# keystone endpoint-create --service-id=$(keystone service-list | awk '/
compute / {print $2}') --publicurl=http://controller-hostname:8774/v2/%\(tenant_id\)s --
internalurl=http://controller-hostname:8774/v2/%\(tenant_id\)s --adminurl=http://controller-
hostname:8774/v2/%\(tenant_id\)s
```

Start Nova related services…

```
[root@controller-hostname ~]# service openstack-nova-api start
[root@controller-hostname ~]# chkconfig openstack-nova-api on
[root@controller-hostname ~]# service openstack-nova-cert start
[root@controller-hostname ~]# chkconfig openstack-nova-cert on
[root@controller-hostname ~]# service openstack-nova-scheduler start
[root@controller-hostname ~]# chkconfig openstack-nova-scheduler on
[root@controller-hostname ~]# service  openstack-nova-novncproxy start
[root@controller-hostname ~]# chkconfig openstack-nova-novncproxy on
[root@controller-hostname ~]# service  openstack-nova-consoleauth start
[root@controller-hostname ~]# chkconfig openstack-nova-consoleauth on
```

```
[root@controller-hostname ~]# service  openstack-nova-conductor start
[root@controller-hostname ~]# chkconfig openstack-nova-conductor on
[root@controller-hostname ~]# service  openstack-nova-metadata-api start
[root@controller-hostname ~]# chkconfig openstack-nova-metadata-api on
```

Note: *After starting the service, check its status and make sure the service is really running. If any of the service failed to start, check the log files under /var/log/nova.*

Nova commands to test the configuration and services…

To list all the images those are stored in Glance service…

```
[root@controller-hostname ~]# nova image-list
+--------------------------------------+--------------+--------+--------------------------------------+
| ID | Name | Status | Server |
+--------------------------------------+--------------+--------+--------------------------------------+
| ac7ffb6d-1594-4a4c-94e7-9d8e70a120a8 | Centos-6 | ACTIVE | |
| a338631b-3bb8-43ba-a700-d4648c040a05 | cirros | ACTIVE | |
| e8c477ae-7c74-497d-9d9b-5fea1035d899 | testvm-snap1 | ACTIVE | aa7c5535-4259-42f0-8d74-
5b26f0d731de |
+--------------------------------------+--------------+--------+--------------------------------------+
```

To list all the services…

```
[root@controller-hostname ~]# nova service-list
+-----------------+------------------------+----------+---------+-------+----------------------------+-----------------+
| Binary | Host | Zone | Status | State | Updated_at | Disabled Reason |
+-----------------+------------------------+----------+---------+-------+----------------------------+-----------------+
| nova-cert | controller-hostname | internal | enabled | up | 2015-02-16T11:06:09.000000 | - |
| nova-scheduler | controller-hostname | internal | enabled | up | 2015-02-16T11:06:07.000000 | - |
| nova-consoleauth | controller-hostname | internal | enabled | up | 2015-02-16T11:06:11.000000 | - |
| nova-conductor | controller-hostname | internal | enabled | up | 2015-02-16T11:06:12.000000 | - |
| nova-console | controller-hostname | internal | enabled | up | 2015-02-16T11:06:12.000000 | - |
+-----------------+------------------------+----------+---------+-------+----------------------------+-----------------+
```

In case, if you face any problem during Glance installation and configuration or during service start-up, then jump to **Appendix 3** to find out the Common errors you may face during Nova Service installation, configuration and the service start-up and its solutions.

Setup Nova-Networking for Controller node

In this two node setup, we are going to install Nova networking (Legacy) on Controller and Compute nodes. It means, you need not worry about the complex Neutron service.

Configure Nova to use nova legacy networking…

```
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf DEFAULT network_api_class nova.network.api.API
[root@controller-hostname ~]# openstack-config --set /etc/nova/nova.conf DEFAULT security_group_api nova
```

Restart nova services

```
[root@controller-hostname ~]# service openstack-nova-api restart
[root@controller-hostname ~]# service openstack-nova-scheduler restart
[root@controller-hostname ~]# service  openstack-nova-conductor restart
```

Create a network for VMs

Now Nova knows that it should use legacy networking for communication, but you have to create a network and a pool of IP addresses that VMs should be assigned to. Just scroll up to see the architecture diagram where you'll find external IP (10.180.14.151) assigned to Compute node. It means, we need to create a subnet in 10.180.14.0 network for VMs to use. In this tutorial, let us assume that we need 30 IP addresses for allocation. So the subnet details goes as below…

```
------------------------------------------------
        TCP/IP NETWORK INFORMATION
------------------------------------------------
IP Entered = ..................: 10.180.14.160
CIDR = ........................: /27
Netmask = .....................: 255.255.255.224
Wildcard Bits = ...............: 0.0.0.31
------------------------------------------------
Network Address = .............: 10.180.14.160
Broadcast Address = ...........: 10.180.14.191
Usable IP Addresses = .........: 30
First Usable IP Address = .....: 10.180.14.161
```

```
Last Usable IP Address = .....: 10.180.14.190
```

For the subnet '10.180.14.160/27', the first usable IP address is 10.180.14.161 and the last usable IP address is 10.180.14.190.

The below command will create a network called 'private'

```
[root@controller-hostname ~]# nova network-create private --bridge br100 --multi-host T --fixed-range-v4
10.180.14.160/27
[root@controller-hostname ~]# nova net-list
+--------------------------------------+---------+------------------+
| ID | Label | CIDR |
+--------------------------------------+---------+------------------+
| 60dfd46a-4649-4758-8b8d-88cc562b9b39 | private | 10.180.14.160/27 |
+--------------------------------------+---------+------------------+
```

Add a security group to the created network to allow SSH connections

```
[root@controller-hostname ~]# nova secgroup-add-rule default tcp 22 22 0.0.0.0/0
+-------------+-----------+---------+-----------+--------------+
| IP Protocol | From Port | To Port | IP Range | Source Group |
+-------------+-----------+---------+-----------+--------------+
| tcp | 22 | 22 | 0.0.0.0/0 | |
+-------------+-----------+---------+-----------+--------------+
```

Install Dashboard (Horizon) on Controller node

Now it's time to install Horizon service on controller node – provides a portal to manage instances, services, network etc…

```
[root@controller-hostname ~]# yum install memcached python-memcached mod_wsgi openstack-dashboard
```

Change the values of CACHES['default']['LOCATION'] as below

```
# vi /etc/openstack-dashboard/local_settings
CACHES = {
'default': {
'BACKEND' : 'django.core.cache.backends.memcached.MemcachedCache',
'LOCATION' : '127.0.0.1:11211'
}
}
```

The above changes should match the address and port specified in /etc/sysconfig/memcached.

Also modify 'ALLOWED_HOSTS' attribute to accept connection from your desktop – as that will allow you to access the dashboard via browser.

You have to make sure SELINUX allows connection to the web server

```
[root@controller-hostname ~]# setsebool -P httpd_can_network_connect on
```

Start web server and memcached

```
[root@controller-hostname ~]# service memcached start
[root@controller-hostname ~]# service httpd start
[root@controller-hostname ~]# chkconfig memcached on
[root@controller-hostname ~]# chkconfig httpd on
```

Try accessing the dashboard service – http://controller-hostname/dashboard

Finally, we are done with our installation on Controller node and we'll now move on to install Nova on Compute node.

Install Nova on Compute node

Install nova compute service on Compute node..

```
[root@compute-hostname ]# yum install openstack-nova-compute
```

Connect to nova database on controller node…

```
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf database connection
mysql://nova:setpassword@controller-hostname/nova
```

Setup authentication…

```
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT auth_strategy keystone
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_uri
http://controller-hostname:5000
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_host
controller-hostname
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_protocol
http
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken auth_port
35357
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken admin_user
nova
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken
admin_tenant_name service
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf keystone_authtoken
admin_password setpassword
```

Configure Nova to use Qpid message broker for communication

```
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT rpc_backend qpid
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT qpid_hostname
controller-hostname
```

Compute Node should know where the Glance is running, so we need to configure that as well.

```
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT glance_host controller-
hostname
```

Start messagebus, libvirtd and nova-compute services

```
[root@compute-hostname ]# service libvirtd start
[root@compute-hostname ]# chkconfig libvirtd on
[root@compute-hostname ]# chkconfig messagebus on
[root@compute-hostname ]# service messagebus start
[root@compute-hostname ]# service openstack-nova-compute start
[root@compute-hostname ]# chkconfig openstack-nova-compute on
```

Install Nova-Networking for Computer node

```
[root@compute-hostname ]# yum install openstack-nova-network openstack-nova-api
```

Configure network api, security group, firewall, network size, dhcp etc..

```
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT network_api_class
nova.network.api.API
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT security_group_api nova
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT network_manager
nova.network.manager.FlatDHCPManager
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT firewall_driver
nova.virt.libvirt.firewall.IptablesFirewallDriver
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT network_size 254
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT allow_same_net_traffic
False
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT multi_host True
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT send_arp_for_ha True
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT share_dhcp_address True
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT force_dhcp_release True
```

Configure Network Bridge and interfaces

If you remember we created a network bridge 'br100' in controller node. For this tutorial, the external IP address of the compute node is 10.180.14.151 (configured on eth1 network interface) and the subnet is '10.180.14.160/27'

```
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT flat_network_bridge
br100
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT flat_interface eth1
[root@compute-hostname ]# openstack-config --set /etc/nova/nova.conf DEFAULT public_interface eth1
```

Note: *Remember to change the network interface (eth1) to the corresponding interface of your compute node.*

Start the services…

```
[root@compute-hostname ]# service openstack-nova-api start
[root@compute-hostname ]# service openstack-nova-network start
[root@compute-hostname ]# service openstack-nova-metadata-api start
[root@compute-hostname ]# chkconfig openstack-nova-api on
[root@compute-hostname ]# chkconfig openstack-nova-network on
[root@compute-hostname ]# chkconfig openstack-nova-api on
```

Well, we are done with installations on both the Controller and Compute node. In case, if you face any problem during Glance installation and configuration or during service start-up, then jump to **Appendix 3** to find out the Common errors you may face during Nova Service installation, configuration and the service start-up and its solutions.

Commands to know

Before trying to boot VM's, you should know the below commands.

List of networks created…

```
[root@controller-hostname ~]# nova net-list
+--------------------------------------+---------+-----------------+
| ID | Label | CIDR |
+--------------------------------------+---------+-----------------+
| 60dfd46a-4649-4758-8b8d-88cc562b9b39 | private | 10.180.14.160/27 |
+--------------------------------------+---------+-----------------+
```

List of images stored in Glance…

```
[root@controller-hostname ~]# nova image-list
+--------------------------------------+--------------+--------+--------------------------------------+
| ID | Name | Status | Server |
+--------------------------------------+--------------+--------+--------------------------------------+
| ac7ffb6d-1594-4a4c-94e7-9d8e70a120a8 | Centos-6 | ACTIVE | |
| a338631b-3bb8-43ba-a700-d4648c040a05 | cirros | ACTIVE | |
| e8c477ae-7c74-497d-9d9b-5fea1035d899 | tgvm-snap1 | ACTIVE | aa7c5535-4259-42f0-8d74-
5b26f0d731de |
+--------------------------------------+--------------+--------+--------------------------------------+
```

List of flavors available…

```
[root@controller-hostname ~]# nova flavor-list
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| ID | Name | Memory_MB | Disk | Ephemeral | Swap | VCPUs | RXTX_Factor | Is_Public |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
| 1 | m1.tiny | 512 | 1 | 0 | | 1 | 1.0 | True |
| 2 | m1.small | 2048 | 20 | 0 | | 1 | 1.0 | True |
| 3 | m1.medium | 4096 | 40 | 0 | | 2 | 1.0 | True |
| 4 | m1.large | 8192 | 80 | 0 | | 4 | 1.0 | True |
| 5 | m1.xlarge | 16384 | 160 | 0 | | 8 | 1.0 | True |
+----+-----------+-----------+------+-----------+------+-------+-------------+-----------+
```

List of security groups created…

```
[root@controller-hostname ~]# nova secgroup-list
+----+---------+-------------+
| Id | Name | Description |
+----+---------+-------------+
| 1 | default | default |
+----+---------+-------------+
```

You may want to **generate SSH keys** to allow users to login to the newly created instance.

```
[root@controller-hostname ~]# ssh-keygen
```

Add the SSH public key to the nova keypair list.

```
[root@controller-hostname ~]# nova keypair-add --pub-key ~/.ssh/id_rsa.pub test-key
```

View the list of key pairs created…

```
[root@controller-hostname ~]# nova keypair-list
+----------+-------------------------------------------------+
| Name | Fingerprint |
+----------+-------------------------------------------------+
| test-key | b0:e1:ff:a5:1b:b0:ff:14:d5:46:13:bc:b6:ba:97:9b |
+----------+-------------------------------------------------+
```

Create an Instance

Based on the output from above commands, you can create a new instance as below:

```
[root@controller-hostname ~]# nova boot --flavor m1.small --image Centos-6 --nic net-id=60dfd46a-4649-4758-8b8d-88cc562b9b39 --security-group default --key-name test-key myfirstvm
+--------------------------------------+--------------------------------------------------+
| Property | Value |
+--------------------------------------+--------------------------------------------------+
| OS-DCF:diskConfig | MANUAL |
| OS-EXT-AZ:availability_zone | nova |
| OS-EXT-SRV-ATTR:host | - |
| OS-EXT-SRV-ATTR:hypervisor_hostname | - |
| OS-EXT-SRV-ATTR:instance_name | instance-00000016 |
| OS-EXT-STS:power_state | 0 |
| OS-EXT-STS:task_state | - |
| OS-EXT-STS:vm_state | building |
| OS-SRV-USG:launched_at | - |
| OS-SRV-USG:terminated_at | - |
| accessIPv4 | |
| accessIPv6 | |
| adminPass | hEMdKAnLs6XX |
| config_drive | |
| created | 2015-02-18T08:56:32Z |
| flavor | m1.small (2) |
| hostId | |
| id | a9735dd7-c601-4209-a86a-0575711239d1 |
| image | Centos-6 (ac7ffb6d-1594-4a4c-94e7-9d8e70a120a8) |
| key_name | test-key |
| metadata | {} |
| name | myfirstvm |
| os-extended-volumes:volumes_attached | [] |
| progress | 0 |
| security_groups | default |
| status | BUILD |
| tenant_id | e7c679bc36ec4c298cf68ecf6d49c1b3 |
| updated | 2015-02-18T08:56:32Z |
```

```
| user_id | 8607e0ccc8ee407daf50c1985616b153 |
+-------------------------------------+--------------------------------------------------+
```

Check if the state of the VM is "ACTIVE" using the below command…

```
[root@controller-hostname ~]# nova list
+--------------------------------------+-----------------+--------+------------+-------------+----------------------+
| ID | Name | Status | Task State | Power State | Networks |
+--------------------------------------+-----------------+--------+------------+-------------+----------------------+
| 8aaa3873-09c5-48f0-9d1e-cb4401e44583 | Cluster-headnode | ACTIVE | - | Running |
private=10.180.14.162 |
| a9735dd7-c601-4209-a86a-0575711239d1 | myfirstvm | ACTIVE | - | Running | private=10.180.14.163 |
+--------------------------------------+-----------------+--------+------------+-------------+----------------------+
```

The new VM called "myfirstvm" is running and the allocated IP address is 10.180.14.163.

Login to an instance

```
[root@controller-hostname ~]# cd .ssh/

[root@controller-hostname .ssh]# ssh -i id_rsa centos@10.180.14.163
Last login: Wed Feb 18 09:01:07 2015 from 10.180.10.132
[centos@myfirstvm ~]$ hostname
myfirstvm
```

Note: Since we installed "Horizon" service on controller node, you should be able to create an instance via OpenStack dashboard. That's super easy!

FAQ's

What is the password for Cirros?

In case, if you have used Cirros image to create an instance, then you must login with username as 'cirros' and password as 'cubswin:)' (of course, without single quotes).

Information about running VM's

To know information about the running VM, you can just issue the below command.

```
[root@controller-hostname ~]# nova show myfirstvm
+--------------------------------------+----------------------------------------------------------+
| Property | Value |
+--------------------------------------+----------------------------------------------------------+
| OS-DCF:diskConfig | MANUAL |
| OS-EXT-AZ:availability_zone | nova |
| OS-EXT-SRV-ATTR:host | gcompute.blr.cdac.in |
| OS-EXT-SRV-ATTR:hypervisor_hostname | gcompute.blr.cdac.in |
| OS-EXT-SRV-ATTR:instance_name | instance-00000016 |
| OS-EXT-STS:power_state | 1 |
| OS-EXT-STS:task_state | - |
| OS-EXT-STS:vm_state | active |
| OS-SRV-USG:launched_at | 2015-02-18T08:56:54.000000 |
| OS-SRV-USG:terminated_at | - |
| accessIPv4 | |
| accessIPv6 | |
| config_drive | |
| created | 2015-02-18T08:56:32Z |
| flavor | m1.small (2) |
| hostId | bd672087b1933d00d10e9d4f85cbac0326ebf3de73a0ce093c0d1838 |
| id | a9735dd7-c601-4209-a86a-0575711239d1 |
| image | Centos-6 (ac7ffb6d-1594-4a4c-94e7-9d8e70a120a8) |
| key_name | test-key |
| metadata | {} |
| name | myfirstvm |
| os-extended-volumes:volumes_attached | [] |
```

```
| private network | 10.180.14.163 |
| progress | 0 |
| security_groups | default |
| status | ACTIVE |
| tenant_id | e7c679bc36ec4c298cf68ecf6d49c1b3 |
| updated | 2015-02-18T08:56:41Z |
| user_id | 8607e0ccc8ee407daf50c1985616b153 |
+--------------------------------------+----------------------------------------------------------+
```

Stop an Instance

If you want to stop a running instance, here's how you can do that.

```
[root@controller-hostname ~]# nova stop myfirstvm
```

If you want to delete an instance completely, then here's the command.

```
[root@controller-hostname ~]# nova delete myfirstvm
```

All these steps could be done via OpenStack Dashboard. *That's super easy!*

How to add another compute node to the OpenStack environment?

It's simple! Just follow the same steps that we used for *Compute-hostname* node.

Appendix 1 - Common keystone service errors and its solutions

Error 1:

How to Fix OpenStack's Keystone Authentication Error – HTTP 500?

I have installed keystone package, but hit with an error while creating an admin user for the cloud. I used the below command.

```
keystone user-create --name=admin --pass=<admin_password> --email=<admin_email>
```

But the above command returned an error message:

```
An unexpected error prevented the server from fulfilling your request. (HTTP 500)
```

Solution:

- You should check the keystone log-file. By default, the keystone logfile is located at "/var/log/keystone/keystone.log" and see what caused HTTP 500 error.
- Check if you are referring to the right 'OS_SERVICE_ENDPOINT' and OS_SERVICE_TOKEN.

For example:

```
#export OS_SERVICE_ENDPOINT=http://localhost:35357/v2.0
#export OS_SERVICE_TOKEN=<valid_token>
```

You should also compare 'OS_SERVICE_ENDPOINT' and 'OS_SERVICE_TOKEN' entries in '/etc/keystone/keystone.conf' file.

Check if MySQL credentials for keystone are set properly.

```
#vi /etc/keystone/keystone.conf
```

And check for the correct entries under [database]

```
[database]
connection = mysql://keystoneUsername:keystonePassword@controller-hostname/keystone
```

In most cases, people make mistakes while configuring the above settings. In case, if the MySQL credentials for keystone is wrong, you should see an error message (/var/log/keystone/keystone.log) like the one shown below.

```
OperationalError: (OperationalError) (2005, "Unknown MySQL server host 'controller' (1)") None None
```

Check if keystone database contains necessary tables: This could be one reason.

For instance, while configuring keystone did you create tables for the keystone database? To verify, login to mysql as below.

```
#mysql -u keystone -p
mysql>use keystone
mysql>show tables
```

If you find no tables in keystone database, then you should create it first. In this case, you should see a corresponding error in /var/log/keystone/keystone.log file. The sample error is posted below:

```
ProgrammingError: (ProgrammingError) (1146, "Table 'keystone.domain' doesn't exist") 'SELECT domain.id AS domain_id, domain.name AS domain_name, domain.enabled AS domain_enabled, domain.extra AS domain_extra \nFROM domain \nWHERE domain.id = %s' ('default',)
```

In case if you see the above error, then create tables for keystone database using the below command.

```
# keystone-manage db_sync keystone
```

Now verify if the tables are created in keystone database.

```
mysql> show tables;
+------------------------+
| Tables_in_keystone |
+------------------------+
| credential |
| domain |
| ec2_credential |
| endpoint_v2 |
| endpoint_v3 |
| metadata |
```

```
| migrate_version |
| policy |
| role |
| service |
| tenant |
| token |
| user |
| user_domain_metadata |
| user_tenant_membership |
+------------------------+
15 rows in set (0.00 sec)
```

Now try creating an admin user by running the below command

```
keystone user-create --name=admin --pass=<admin_password> --email=<admin_email>
```

Error 2:

CRITICAL keystone [-] ConfigFileNotFound: The Keystone configuration file keystone-paste.ini could not be found

I have installed keystone package, but hit with an error while starting the keystone service. I used the below command.

```
# service openstack-keystone start
Stopping keystone:                                 [FAILED]
```

When looked at the keystone log, observed the below error recorded

```
#tailf /var/log/keystone/keystone.log
CRITICAL keystone [-] ConfigFileNotFound: The Keystone configuration file keystone-paste.ini could not be found.
TRACE keystone Traceback (most recent call last):
TRACE keystone   File "/usr/bin/keystone-all", line 113, in
TRACE keystone     paste_config = config.find_paste_config()
TRACE keystone   File "/usr/lib/python2.6/site-packages/keystone/config.py", line 90, in find_paste_config
TRACE keystone     raise exception.ConfigFileNotFound(config_file=paste_config_value)
TRACE keystone ConfigFileNotFound: The Keystone configuration file keystone-paste.ini could not be found.
```

Solution:

As the error message says, verify whether *keystone-paste.ini* file exists and identified properly by the*keystone.conf* file.

Open */etc/keystone/keystone.conf* and see if '*config_file*' is referring to keystone-paste.ini file.

```
[paste_deploy]
config_file=/usr/share/keystone/keystone-dist-paste.ini
```

Try starting the service again.

```
#service openstack-keystone start
```

Appendix 2 - Errors during OpenStack Image Service GLANCE configuration and solutions

Error 1:

Error when 'glance-manage db_sync glance' executed

Here's the snapshot of the error:

```
$ glance-manage db_sync glance
/usr/lib/python2.6/site-packages/glance/cmd/manage.py:41: DeprecationWarning: The oslo namespace
package is deprecated. Please use oslo_config instead.
  from oslo.config import cfg
/usr/lib64/python2.6/site-packages/Crypto/Util/number.py:57: PowmInsecureWarning: Not using
mpz_powm_sec.  You should rebuild using libgmp >= 5 to avoid timing attack vulnerability.
  _warn("Not using mpz_powm_sec.  You should rebuild using libgmp >= 5 to avoid timing attack
vulnerability.", PowmInsecureWarning)
2015-02-03 22:00:02.848 8671 CRITICAL glance [-] DbMigrationError: version should be an integer
```

Solution:

This has something to do with the version of GMP installed. As you can see from the above error, **PyCrypto needs libgmp version >=5**. To fix this, follow the below steps:

Download and install gmp-6.0.0.a[7]

```
$ bunzip2 gmp-6.0.0a.tar.bz2
$ tar xvf gmp-6.0.0a.tar
$ cd gmp-6.0.0a.tar
$ ./configure
$ make
$ make check
$ make install
```

Now you have the right version of GMP installed and with these libraries you can install PyCrypto. To do that, run the below command:

```
$ pip install --ignore-installed PyCrypto
```

Once PyCrpto is installed, you can try Glance database sync again.

Error 2:

ImportError: /usr/lib64/python2.6/site-packages/Crypto/Cipher/_AES.so: undefined symbol: rpl_malloc

When '*glance-manage db_sync glance*' executed, I was hit with the below error message:

```
$ glance-manage db_sync glance
su -s /bin/sh -c "glance-manage db_sync" glance
Traceback (most recent call last):
  File "/usr/bin/glance-manage", line 6, in
    from glance.cmd.manage import main
  File "/usr/lib/python2.6/site-packages/glance/cmd/manage.py", line 45, in
    from glance.db import migration as db_migration
  File "/usr/lib/python2.6/site-packages/glance/db/__init__.py", line 21, in
    from glance.common import crypt
  File "/usr/lib/python2.6/site-packages/glance/common/crypt.py", line 23, in
    from Crypto.Cipher import AES
  File "/usr/lib64/python2.6/site-packages/Crypto/Cipher/AES.py", line 50, in
    from Crypto.Cipher import _AES
ImportError: /usr/lib64/python2.6/site-packages/Crypto/Cipher/_AES.so: undefined symbol: rpl_malloc
```

Solution:

Previously, we fixed libgmp, PyCrypto error by installing GMP version 6.0.0.a and pip install PyCrypto. But that didn't work as expected. It seems like PyCrypto should be recompiled from source and should not be installed via pip.

Download latest version of PyCrypto[8] **and recompile**

Untar PyCrypto and set the below environment variable.

```
$ export ac_cv_func_malloc_0_nonnull=yes
```

Configure, Build and Install

```
$ ./configure
$ python setup.py build
$ python setup.py install
```

Once PyCrypto is installed, try '*glance-manage db_sync glance*' again.

Error 3:

ValueError: Tables "migrate_version" have non utf8 collation, please make sure all tables are CHARSET=utf8

```
# su -s /bin/sh -c "glance-manage db_sync" glance
```

Note: Whenever you sync database for any service, make sure the corresponding tables are created. For example, when I synced 'glance' database using the above command, no tables were created except 'migrate_version' and the error log reported as shown in the title of this para.

```
mysql> show tables;
+------------------+
| Tables_in_glance |
+------------------+
| migrate_version  |
+------------------+
1 row in set (0.00 sec)
```

Solution:

Login to mysql database as below:

```
$ mysql -u root -p
mysql > use glance
mysql > alter table migrate_version convert to character set utf8 collate utf8_unicode_ci;
```

Now, try "db_sync" again.

```
mysql> show tables;
+------------------+
| Tables_in_glance |
```

```
+------------------+
| image_locations  |
| image_members    |
| image_properties |
| image_tags       |
| images           |
| migrate_version  |
| task_info        |
| tasks            |
+------------------+
8 rows in set (0.00 sec)
```

Error 4:

CRITICAL glance [-] DbMigrationError: version should be an integer

I was still left with few errors while syncing the Glance database.

```
$ glance-manage db_sync glance
/usr/lib/python2.6/site-packages/glance/cmd/manage.py:41: DeprecationWarning: The oslo namespace package is deprecated. Please use oslo_config instead.
  from oslo.config import cfg
2015-02-04 09:46:14.847 13116 CRITICAL glance [-] DbMigrationError: version should be an integer
```

Solution:

Instead of executing '*glance-manage*' with two arguments, try with one argument '*db_sync*' alone.

```
# glance-manage db_sync
```

It worked for me.

Error 5:

Expecting an auth URL via either –os-auth-url or env[OS_AUTH_URL

```
$ keystone user-create --name=glance --pass=<glance_password> --email=<admin_email>
Expecting an auth URL via either --os-auth-url or env[OS_AUTH_URL
```

Solution

Probably the OpenStack authentication URLs are not set. Try setting the below environment variables.

```
export OS_USERNAME=admin
export OS_PASSWORD=keystone_password
export OS_TENANT_NAME=admin
export OS_AUTH_URL=http://controller.node.in:35357/v2.0
```

Once the above variables are set in your shell, try registering the glance service to the keystone again. It worked for me.

```
$ keystone user-create --name=glance --pass=glance_password --email=admin_email
Expecting an auth URL via either --os-auth-url or env[OS_AUTH_URL

You should see table like the one below
+----------+----------------------------------+
| Property |              Value               |
+----------+----------------------------------+
|  email   |         admin_email              |
| enabled  |              True                |
|    id    | c8c0212a16084a65469609c857914f8c |
|   name   |             glance               |
| username |              glance              |
+----------+----------------------------------+
```

Error 6:

Error while starting the glance-api service

```
$ service openstack-glance-api start [FAILED]
```

The error log */var/log/glance/openstack-glance-api-startup.log* revealed the below message:

```
Failed to configure store correctly: Store filesystem could not be configured correctly. Reason: Specify at least 'filesystem_store_datadir' or 'filesystem_store_datadirs' option Disabling add method.
```

Solution:

- Open *glance-api.conf* file. It should be located at */etc/glance/glance-api.conf*
- Look out for '*filesystem_store_datadir*' and set '*/var/lib/glance/images/*' as value. If already set and commented, just uncomment it.

Try starting the service.

Error 7:

Unable to locate paste config file for glance-api

```
$ service openstack-glance-api start [FAILED]
```

Solution:

- Open *glance-api.conf* file. It should be located at */etc/glance/glance-api.conf*
- Lookout for the line with 'paste configuration file' text and uncomment the below line

```
config_file=/usr/share/glance/glance-api-dist-paste.ini
```

Try starting the service again.

Error 8:

CRITICAL glance [-] UnicodeError: Message objects do not support str() because they may contain non-ascii characters. Please use unicode() or translate() instead.

```
$ service openstack-glance-registry start [FAILED]
```

The log file '*/var/log/glance/openstack-glance-registry-startup.log*' revealed the below message:

Solution:

The error occured in the file :

```
'/usr/lib/python2.6/site-packages/glance/openstack/common/gettextutils.py'
```

Which was raising '**UnicodeError**'. You will find more information about the error here[9].

To fix the issue:

1. Go to this link[10] and copy the modified 'gettextutils.py' file.
2. Take a backup of the existing **gettextutils.py** file as below

```
cp /usr/lib/python2.6/site-packages/glance/openstack/common/gettextutils.py /usr/lib/python2.6/site-packages/glance/openstack/common/gettextutils.py.bak
```

3. Replace the file '*gettextutils.py*' with the one copied from step 1.
4. Start the glance-registry service as below:

```
$ service openstack-glance-registry start
```

Note: If the above command fails with a syntax error, then probably you copied some unwanted characters while copying the modified gettextutils.py script in step 1. For example, the copied file contained some characters such as '*[docs]*'. To fix the issue, open '*gettextutils.py*', search and remove those unwanted characters. Save the file and start the glance-registry service again.

In case, if the service failed to start, then lookout for the fresh error message in the log files.

```
# tailf /var/log/glance/openstack-glance-registry-startup.log
Traceback (most recent call last):
  File "/usr/bin/glance-registry", line 6, in <module>
```

```
  from glance.cmd.registry import main
 File "/usr/lib/python2.6/site-packages/glance/cmd/__init__.py", line 17, in <module>
  gettextutils.install('glance', lazy=True)
TypeError: install() got an unexpected keyword argument 'lazy'
```

To fix the above error:

```
$vim /usr/lib/python2.6/site-packages/glance/cmd/__init__.py
```

And lookout for the function ' gettextutils.install('glance', lazy=True)' and replace it with the below one:

```
gettextutils.install('glance')
```

Save the file and try starting the glance-registry service.

Error 9:

Unable to locate paste config file for glance-registry.

```
# service openstack-glance-registry start
Starting openstack-glance-registry:                [FAILED]
```

The log file revealed the below error:

```
# tailf /var/log/glance/openstack-glance-registry-startup.log
ERROR: Unable to locate paste config file for glance-registry.
```

Solution:

- Open '**vim /etc/glance/glance-registry.conf**' and lookout for the line with 'paste configuration file' text.
- Uncomment 'config_file'. For example: '**config_file=/usr/share/glance/glance-registry-dist-paste.ini**' and try starting the service again. It worked for me.

Error 10:

DeprecationWarning: The oslo namespace package is deprecated. Please use oslo_config instead. from oslo.config import cfg

Solution

Open the below files one by one and replace 'oslo.config' with 'oslo_config' and save it. (Note dot is replaced with underscore)

```
/usr/lib/python2.6/site-packages/glance/openstack/common/log.py
/usr/lib/python2.6/site-packages/glance/common/wsgi.py:37
/usr/lib/python2.6/site-packages/oslo_config/cfg.py:333
/usr/lib/python2.6/site-packages/oslo_config/cfg.py:333
/usr/lib/python2.6/site-packages/glance/common/config.py:26
/usr/lib/python2.6/site-packages/paste/deploy/loadwsgi.py:22
```

Try starting the service and see if you get those warnings.

Appendix 3 - Errors during OpenStack Nova service configuration and solutions

Error 1:

ERROR: ('Connection aborted.', error(111, 'Connection refused'))

```
# nova image-list
ERROR: ('Connection aborted.', error(111, 'Connection refused'))
```

Solution:

When I tried to image-list via nova, the connection refused error was thrown. This was primarily due to various reasons,

Check if all the nova related services are running properly.

For example: when you start any nova services as 'service nova-api start', the command might return green status as "[OK]'. But actually the service might not have been started properly. Below is one example.

```
# /etc/init.d/openstack-nova-api start
Starting openstack-nova-api:                [  OK  ]
# /etc/init.d/openstack-nova-api status
openstack-nova-api dead but pid file exists
```

In above case, you should check the corresponding service log under /var/log/nova/api.log to know the exact error message.

In my case, none of the nova service was starting properly and I had to dig all of those service logs.

```
openstack-nova-api is running...
openstack-nova-cert is running...
openstack-nova-conductor is running...
openstack-nova-console is running...
openstack-nova-consoleauth is running...
openstack-nova-metadata-api is running
openstack-nova-novncproxy is running...
openstack-nova-scheduler is running...
```

Check if the firewall on the controller node is blocking the service ports.

Error 2:

CRITICAL nova [-] RequiredOptError: value required for option: lock_path

The above error message was captured from */var/log/nova/api.log* file and it was triggered when*openstack-nova-api* service failed to start.

Solution:

```
# vim /etc/nova/nova.conf
```

And uncomment 'lock_path' : *lock_path=/var/lib/nova/tmp.* Save the file and start the service.

Error 3:

ERROR nova.openstack.common.threadgroup [-] [Errno 13] Permission denied: '/usr/lib/python2.6/site-packages/CA'

The above error message was logged in */var/log/nova/cert.log* and it was triggered when '*openstack-nova-cert*' failed to start.

Solution:

```
# vim /etc/nova/nova.conf
```

and check if '**ca_path**' is pointing to the right directory and user 'nova' has permission to it.

Error 4:

ERROR nova.wsgi [-] Could not bind to 0.0.0.0:8775 CRITICAL nova [-] error: [Errno 98] Address already in use

```
# service openstack-nova-metadata-api status
openstack-nova-metadata-api dead but pid file exists
```

And /var/log/nova/metadata-api.log reported the above error message.

Solution:

Surprisingly, the service '*openstack-nova-metadata-api*' was already running. Because, 'openstack-nova-api' starts 'metadata-api' service along with it and thus the service 'metadata-api' fails to bind to the port. All you need to do is, check if 'nova-api' is configured to start 'metadata-api' along with it. To do that,

```
# vim /etc/nova/nova.conf
```

And lookout for '**enabled_apis = osapi_compute,metadata**'

If you find metadata in enabled_apis, then whenever nova-api is started, it will also start metadata api.

In case, if you want to start nova-metadata-api individually, then remove metadata from 'enabled_apis'.

```
'enabled_apis = osapi_compute'
# service openstack-nova-api start
# service openstack-nova-metadata-api start
```

Now the service should start individually.

Error 5:

Error: compute driver option required but not specified

```
# service openstack-nova-compute status
openstack-nova-compute dead but pid file exists
```

Solution:

```
#vim /etc/nova/nova.conf
```

And set '**compute_driver**'.

Error 6:

TRACE nova.openstack.common.threadgroup OSError: [Errno 2] No such file or directory: '/usr/lib/python2.6/site-packages/instances'

```
# service openstack-nova-compute status
openstack-nova-compute dead but pid file exists
```

Solution:

```
#vim /etc/nova/nova.conf
```

And set '**instances_path=/var/lib/nova/instances**'

References

1. http://jensd.be/?p=289
2. http://openstack.org
3. http://techglimpse.com/configure-yum-rhel-centos-repository/
4. http://techglimpse.com/yum-install-search-uninstall-commands-and-examples/
5. http://techglimpse.com/mysql-mariadb-automatic-secure-installation-script/
6. http://techglimpse.com/linux-xz-command-unzip-xz-tar-files/
7. https://gmplib.org/#DOWNLOAD
8. https://www.dlitz.net/software/pycrypto/
9. http://lists.openstack.org/pipermail/openstack-dev/2014-June/038795.html
10. http://docs.openstack.org/developer/keystone/_modules/keystone/openstack/common/gettextutils.html