1. OpenStack Basic Install

Table of Contents

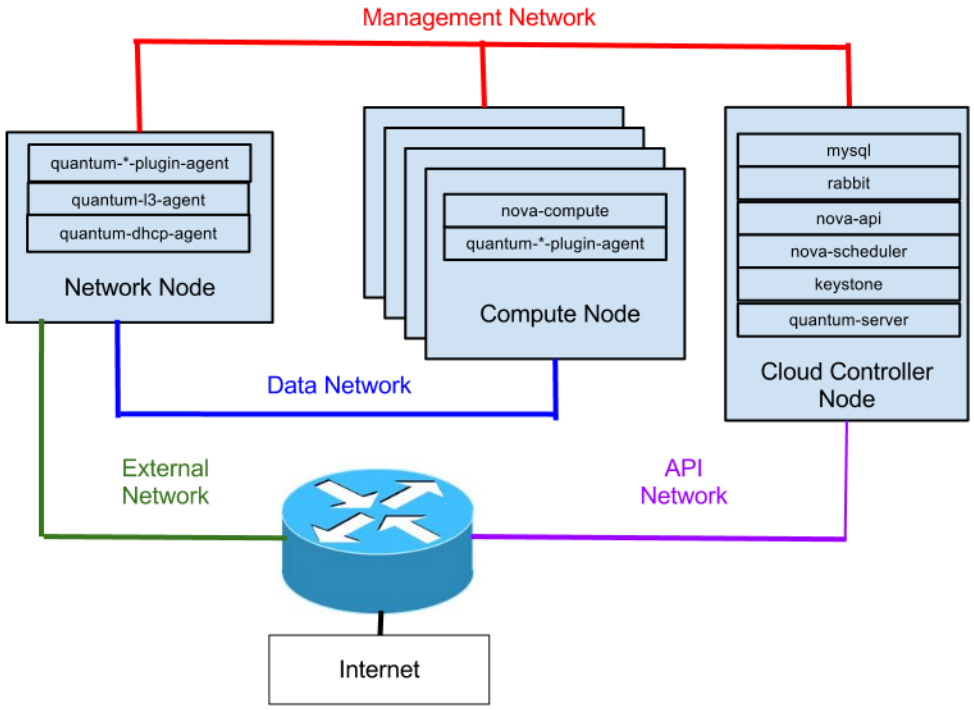
Introduction

This document helps anyone who wants to deploy OpenStack Grizzly for development purposes with Ubuntu 12.04 LTS (using the Ubuntu Cloud Archive).

We are going to install a three-node setup with one controller, one network and one compute node.

Of course, you can setup as many computes nodes as you want. This document is a good start for beginners in OpenStack who want to install a testing infrastructure.

Architecture



A standard Quantum 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 Quantum plugin 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 Quantum 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 quantum 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.

Requirements

You need at least three machines (virtual or physical) with Ubuntu 12.04 (LTS) installed.

Table 1.1. Architecture and node information

	controller	**network**	**compute**
Hostname	grizzly-controller	grizzly-network	grizzly-compute
Services	MySQL, RabbitMQ, Nova, Cinder, Glance, Keystone, Quantum	Quantum-L3-agent, Quantum-DHCP-agent, Quantum Agent with Open-vSwitch	nova-compute, KVM, nova-api, Quantum Agent with Open-vSwitch
Minimum number of disks	2	1	1
External + API network	7.7.7.7/24	7.7.7.8/24	-
Management network	192.168.0.1/24	192.168.0.2/24	192.168.0.3/24
Data network	-	10.10.10.1/24	10.10.10.2/24
Total number of NIC	**2**	**3**	**2**

Controller Node

Introduction

The Controller node will provide :

- Databases (with MySQL)
- Queues (with RabbitMQ)
- Keystone
- Glance
- Nova (without nova-compute)
- Cinder
- Quantum Server (with Open-vSwitch plugin)
- Dashboard (with Horizon)

Common services

Operating System

1. Install Ubuntu with this parameters :
   - Time zone : **UTC**
   - Hostname : **controller**
   - Packages : **OpenSSH-Server**

   After OS Installation, reboot the server.
2. Since the default OpenStack release in Ubuntu 12.04 LTS is older, we are going to use the Ubuntu Cloud Archive for Grizzly :

```
apt-get install ubuntu-cloud-keyring
```

Edit **/etc/apt/sources.list.d/cloud-archive.list** :

```
deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/grizzly
 main
```

Upgrade the system (and reboot if you need) :

```
apt-get update && apt-get upgrade
```

3. Configure the network :

   - Edit **/etc/network/interfaces** file :

```
# Management Network
auto eth0
    iface eth0 inet static
    address 192.168.0.1
    netmask 255.255.255.0

# API + Public Network
    auto eth1
    iface eth1 inet static
    address 7.7.7.7
    netmask 255.255.255.0
    gateway 7.7.7.1
    dns-nameservers 8.8.8.8
```

   - Edit **/etc/sysctl.conf** :

```
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
```

   Then, restart network service :

```
service networking restart
```

   - Edit the **/etc/hosts** file and add **controller**, **networknode** and **compute1** hostnames with correct IP.

```
127.0.0.1       localhost
127.0.1.1       controller
192.168.0.1     controller
192.168.0.2     network
192.168.0.3     compute
```

4. Install Configure NTP :

   - Install the package :

```
apt-get install -y ntp
```

   - Configure **/etc/ntp.conf** file :

```
server ntp.ubuntu.com iburst
server 127.127.1.0
fudge 127.127.1.0 stratum 10
```

   - Restart the service :

```
service ntp restart
```

MySQL Database Service

1. Install the packages :

```
apt-get install mysql-server python-mysqldb
```

2. Allow connection from the network :

```
sed -i 's/127.0.0.1/0.0.0.0/g' /etc/mysql/my.cnf
```

3. Restart the service :

```
service mysql restart
```

4. Create Databases, Users, Rights :

```
mysql -u root -password <<EOF
CREATE DATABASE nova;
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'localhost' \
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'192.168.0.1' \
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'192.168.0.2' \
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON nova.* TO 'nova'@'192.168.0.3' \
IDENTIFIED BY 'password';
CREATE DATABASE cinder;
GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'localhost' \
IDENTIFIED BY 'password';
CREATE DATABASE glance;
GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
IDENTIFIED BY 'password';
CREATE DATABASE keystone;
GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' \
IDENTIFIED BY 'password';
CREATE DATABASE quantum;
GRANT ALL PRIVILEGES ON quantum.* TO 'quantum'@'localhost' \
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON quantum.* TO 'quantum'@'192.168.0.2' \
IDENTIFIED BY 'password';
GRANT ALL PRIVILEGES ON quantum.* TO 'quantum'@'192.168.0.3' \
IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
EOF
```

RabbitMQ Messaging Service

1. Install the packages :

```
apt-get install rabbitmq-server
```

2. Change the default password :

```
rabbitmqctl change_password guest password
```

Keystone

1. Install the packages :

```
apt-get install keystone python-keystone python-keystoneclient
```

2. Edit **/etc/keystone/keystone.conf** :

```
[DEFAULT]
admin_token = password
bind_host = 0.0.0.0
public_port = 5000
admin_port = 35357
compute_port = 8774
verbose = True
debug = True
log_file = keystone.log
log_dir = /var/log/keystone
log_config = /etc/keystone/logging.conf

[sql]
connection = mysql://keystone:password@localhost:3306/keystone
idle_timeout = 200

[identity]
driver = keystone.identity.backends.sql.Identity

[catalog]
driver = keystone.catalog.backends.sql.Catalog

(...)
```

3. Restart Keystone and create the tables in the database :

```
service keystone restart
keystone-manage db_sync
```

4. Load environment variables :

   - Create **novarc** file :

```
export OS_TENANT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=password
export OS_AUTH_URL="http://localhost:5000/v2.0/"
export SERVICE_ENDPOINT="http://localhost:35357/v2.0"
export SERVICE_TOKEN=password
```

   - Export the variables :

```
source novarc
echo "source novarc">>.bashrc
```

5. Download the data script and fill Keystone database with data (users, tenants, services) :

```
./keystone-data.sh
```

6. Download the endpoint script and create the endpoints (for projects) :

```
./keystone-endpoints.sh
```

If an IP address of the management network on the controller node is different from this example, please use the following:

```
./keystone-endpoints.sh -K <ip address of the management network>
```

Glance

1. Install the packages :

```
apt-get install glance glance-api glance-registry python-glanceclient
 glance-common
```

2. Configure Glance :

   - Edit **/etc/glance/glance-api.conf** and **/etc/glance/glance-registry.conf** files and modify :

     ```
     sql_connection = mysql://glance:password@localhost/glance
     admin_tenant_name = service
     admin_user = glance
     admin_password = password
     ```

     For **glance-api.conf**, modify :

     ```
     notifier_strategy = rabbit
     rabbit_password = password
     ```

   - Restart Glance services :

     ```
     service glance-api restart && service glance-registry restart
     ```

   - Create Glance tables into the database :

     ```
     glance-manage db_sync
     ```

   - Download and import Ubuntu 12.04 LTS UEC Image :

     ```
     glance image-create \
         --location http://uec-images.ubuntu.com/releases/12.04/release/
     ubuntu-12.04-server-cloudimg-amd64-disk1.img \
         --is-public true --disk-format qcow2 --container-format bare --name
      "Ubuntu"
     ```

   - Check if the image has been introduced in the index :

     ```
     glance image-list

     +--------------------------------------+--------+------------
     +-----------------+-----------+--------+
     | ID                                   | Name   | Disk Format | Container
      Format | Size      | Status |
     +--------------------------------------+--------+------------
     +-----------------+-----------+--------+
     | 0d2664d3-cda9-4937-95b2-909ecf8ea362 | Ubuntu | qcow2       | bare
            | 233701376 | active |
     +--------------------------------------+--------+------------
     +-----------------+-----------+--------+
     ```

- You can also install Glance Replicator. More informations about it here.

Nova

1. Install the packages :

```
apt-get install nova-api nova-cert nova-common nova-conductor \
    nova-scheduler python-nova python-novaclient nova-consoleauth novnc \
    nova-novncproxy
```

2. Configure Nova :

- Edit **/etc/nova/api-paste.ini** file and modify :

```
admin_tenant_name = service
admin_user = nova
admin_password = password
```

```
============================================================
[composite:osapi_volume]
use = call:nova.api.openstack.urlmap:urlmap_factory
/: osvolumeversions
/v1: openstack_volume_api_v1
============================================================

============================================================
[composite:openstack_volume_api_v1]
use = call:nova.api.auth:pipeline_factory
noauth = faultwrap sizelimit noauth ratelimit osapi_volume_app_v1
keystone = faultwrap sizelimit authtoken keystonecontext ratelimit
 osapi_volume_app_v1
keystone_nolimit = faultwrap sizelimit authtoken keystonecontext
 osapi_volume_app_v1
============================================================

============================================================
[app:osapi_volume_app_v1]
paste.app_factory = nova.api.openstack.volume:APIRouter.factory
============================================================

============================================================
[pipeline:osvolumeversions]
pipeline = faultwrap osvolumeversionapp

[app:osvolumeversionapp]
paste.app_factory = nova.api.openstack.volume.versions:Versions.factory
============================================================
```

- Edit **/etc/nova/nova.conf** file and modify :

```
[DEFAULT]

# MySQL Connection #
sql_connection=mysql://nova:password@192.168.0.1/nova

# nova-scheduler #
rabbit_password=password
scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler
```

```
# nova-api #
cc_host=192.168.0.1
auth_strategy=keystone
s3_host=192.168.0.1
ec2_host=192.168.0.1
nova_url=http://192.168.0.1:8774/v1.1/
ec2_url=http://192.168.0.1:8773/services/Cloud
keystone_ec2_url=http://192.168.0.1:5000/v2.0/ec2tokens
api_paste_config=/etc/nova/api-paste.ini
allow_admin_api=true
ec2_private_dns_show_ip=True
dmz_cidr=169.254.169.254/32
ec2_dmz_host=192.168.0.1
metadata_host=192.168.0.1
metadata_listen=0.0.0.0
enabled_apis=ec2,osapi_compute,metadata

# Networking #
network_api_class=nova.network.quantumv2.api.API
quantum_url=http://192.168.0.1:9696
quantum_auth_strategy=keystone
quantum_admin_tenant_name=service
quantum_admin_username=quantum
quantum_admin_password=password
quantum_admin_auth_url=http://192.168.0.1:35357/v2.0
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
linuxnet_interface_driver=nova.network.linux_net.LinuxOVSInterfaceDriver
firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver

# Cinder #
volume_api_class=nova.volume.cinder.API

# Glance #
glance_api_servers=192.168.0.1:9292
image_service=nova.image.glance.GlanceImageService

# novnc #
novnc_enable=true
novncproxy_base_url=http://7.7.7.7:6080/vnc_auto.html
vncserver_proxyclient_address=192.168.0.1
vncserver_listen=0.0.0.0

# Misc #
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/var/lock/nova
root_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf
verbose=true
```

- Create Nova tables into the database :

```
nova-manage db sync
```

- Restart Nova services :

```
service nova-api restart
service nova-cert restart
service nova-consoleauth restart
service nova-scheduler restart
service nova-novncproxy restart
```

Cinder

1. Install the packages :

```
apt-get install -y cinder-api cinder-scheduler cinder-volume iscsitarget \
    open-iscsi iscsitarget-dkms python-cinderclient linux-headers-`uname -r`
```

2. Configure & start the iSCSI services :

```
sed -i 's/false/true/g' /etc/default/iscsitarget
service iscsitarget start
service open-iscsi start
```

3. Configure Cinder :

   - Edit **/etc/cinder/cinder.conf** file and modify :

```
[DEFAULT]
sql_connection = mysql://cinder:password@localhost:3306/cinder
rabbit_password = password
```

   - Edit **/etc/cinder/api-paste.ini** file and modify :

```
admin_tenant_name = service
admin_user = cinder
admin_password = password
```

   - Create the volume (on the second disk) :

```
fdisk /dev/sdb

[Create a Linux partition]

pvcreate /dev/sdb1
vgcreate cinder-volumes /dev/sdb1
```

   - Create Cinder tables into the database :

```
cinder-manage db sync
```

   - Restart the services :

```
service cinder-api restart
service cinder-scheduler restart
service cinder-volume restart
```

Quantum

1. Install the packages :

```
apt-get install quantum-server
```

2. Configure Quantum services :

   - Edit **/etc/quantum/quantum.conf** file and modify :

```
core_plugin = \
    quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2
```

```
auth_strategy = keystone
fake_rabbit = False
rabbit_password = password
```

- Edit **/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini** file and modify :

```
[DATABASE]
sql_connection = mysql://quantum:password@localhost:3306/quantum
[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
```

Note

It's more handy to choose **tunnel mode** since you don't have to configure your physical switches for VLANs.

- Edit **/etc/quantum/api-paste.ini** file and modify :

```
admin_tenant_name = service
admin_user = quantum
admin_password = password
```

3. Start the services :

```
service quantum-server restart
```

Dashboard (Horizon)

Install the packages :

```
apt-get install apache2 libapache2-mod-wsgi openstack-dashboard \
    memcached python-memcache
```

OpenStack Dashboard is now available at **http://<controller_node>/horizon**. We can login with **admin** / **password** credentials or **demo** / **password**.

Network Node

Introduction

The Network node will provide :

- Virtual Bridging (Open-vSwitch + Quantum Agent) with tunneling
- DHCP Server (Quantum DHCP Agent)
- Virtual Routing (Quantum L3 Agent)

Common services

Operating System

1. Install Ubuntu with this parameters :

- Time zone : **UTC**
- Hostname : **grizzly-network**
- Packages : **OpenSSH-Server**

After OS Installation, reboot the server.

2. Since the default OpenStack version in Ubuntu 12.04 LTS is older, we are going to use Cloud Archives for Grizzly :

```
apt-get install ubuntu-cloud-keyring
```

Edit **/etc/apt/sources.list.d/cloud-archive.list** :

```
deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/grizzly
 main
```

Upgrade the system (and reboot if you need) :

```
apt-get update && apt-get upgrade
```

3. Configure the network :

- Edit **/etc/network/interfaces** file :

```
# Management Network
auto eth0
    iface eth0 inet static
    address 192.168.0.2
    netmask 255.255.255.0
    gateway 192.168.0.254
    dns-nameservers 8.8.8.8

# Data Network
    auto eth1
    iface eth1 inet static
    address 10.10.10.1
    netmask 255.255.255.0

# Public Bridge
    auto eth2
    iface eth2 inet manual
    up ifconfig $IFACE 0.0.0.0 up
    up ip link set $IFACE promisc on
    down ifconfig $IFACE down
```

- Edit **/etc/sysctl.conf** :

```
net.ipv4.ip_forward=1
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
```

Then, restart network service :

```
service networking restart
```

- Edit the **/etc/hosts** file and add **grizzly-controller**, **grizzly-network** and **grizzly-compute** hostnames with correct IP.

4. Install Configure NTP :

   - Install the package :

     ```
     apt-get install -y ntp
     ```

   - Configure **/etc/ntp.conf** file :

     ```
     server 192.168.0.1
     ```

   - Restart the service :

     ```
     service ntp restart
     ```

Network Services

Open-vSwitch

1. Install the packages :

   ```
   apt-get install quantum-plugin-openvswitch-agent \
   quantum-dhcp-agent quantum-l3-agent
   ```

2. Start Open vSwitch:

   ```
   service openvswitch-switch start
   ```

3. Create Virtual Bridging :

   ```
   ovs-vsctl add-br br-int
   ovs-vsctl add-br br-ex
   ovs-vsctl add-port br-ex eth2
   ip link set up br-ex
   ```

Quantum

Configure Quantum services :

- Edit **/etc/quantum/l3_agent.ini** file and modify :

  ```
  auth_url = http://192.168.0.1:35357/v2.0
  admin_tenant_name = service
  admin_user = quantum
  admin_password = password
  metadata_ip = 192.168.0.1
  use_namespaces = False
  ```

- Edit **/etc/quantum/api-paste.ini** file and modify :

  ```

  auth_host = 192.168.0.1
  admin_tenant_name = service
  admin_user = quantum
  admin_password = password
  ```

- Edit **/etc/quantum/quantum.conf** file and modify :

  ```
  core_plugin = \
      quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2
  ```

```
auth_strategy = keystone
fake_rabbit = False
rabbit_host = 192.168.0.1
rabbit_password = password
```

- Edit **/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini** file and modify :

```
[DATABASE]
sql_connection = mysql://quantum:password@192.168.0.1:3306/quantum
[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
enable_tunneling = True
integration_bridge = br-int
tunnel_bridge = br-tun
local_ip = 10.10.10.1
```

Note

It's more handy to choose **tunnel mode** since you don't have to configure your physical switches for VLANs.

- Edit **/etc/quantum/dhcp_agent.ini** file and add :

```
use_namespaces = False
```

Start the services :

```
service quantum-plugin-openvswitch-agent start
service quantum-dhcp-agent restart
service quantum-l3-agent restart
```

Virtual Networking

Create Virtual Networking

1. Load environment variables :
   - Create **novarc** file :

     ```
     export OS_TENANT_NAME=admin
     export OS_USERNAME=admin
     export OS_PASSWORD=password
     export OS_AUTH_URL="http://192.168.0.1:5000/v2.0/"
     export SERVICE_ENDPOINT="http://192.168.0.1:35357/v2.0"
     export SERVICE_TOKEN=password
     ```

   - Export the variables :

     ```
     source novarc
     echo "source novarc">>.bashrc
     ```

2. Download the Quantum script. We are using the "**Provider Router with Private Networks**" use-case.
3. Edit the script belong your networking (public network, floatings IP).
4. Execute the script.

L3 Configuration

- Copy the external network ID :

```
quantum net-list
```

- Edit /etc/quantum/l3_agent.ini and paste the ID :

```
gateway_external_network_id = ID
```

- Copy the provider router ID :

```
quantum router-list
```

- Edit /etc/quantum/l3_agent.ini and paste the ID :

```
router_id = ID
```

- Restart L3 Agent :

```
service quantum-l3-agent restart
```

Compute Node

Introduction

The Compute node will provide :

- Hypervisor (KVM)
- nova-compute
- Quantum OVS Agent

Common services

1. Install Ubuntu with this parameters :
   - Time zone : **UTC**
   - Hostname : **grizzly-compute**
   - Packages : **OpenSSH-Server**

   After OS Installation, reboot the server .
2. Since the default OpenStack version in Ubuntu 12.04 LTS is older, we are going to use Cloud Archives for Grizzly :

   ```
   apt-get install ubuntu-cloud-keyring
   ```

   Edit **/etc/apt/sources.list.d/cloud-archive.list** :

   ```
   deb http://ubuntu-cloud.archive.canonical.com/ubuntu precise-updates/grizzly
    main
   ```

   Upgrade the system (and reboot if you need) :

```
apt-get update && apt-get upgrade
```

3. Configure the network :

   - Edit **/etc/network/interfaces** file :

```
# Management Network
auto eth0
    iface eth0 inet static
    address 192.168.0.3
    netmask 255.255.255.0
    gateway 192.168.0.254
    dns-nameservers 8.8.8.8

# Data Network
auto eth1
    iface eth1 inet static
    address 10.10.10.2
    netmask 255.255.255.0
```

   - Edit **/etc/sysctl.conf** :

```
net.ipv4.conf.all.rp_filter = 0
net.ipv4.conf.default.rp_filter = 0
```

   Then, restart network service :

```
service networking restart
```

   - Edit the **/etc/hosts** file and add **grizzly-controller**, **grizzly-network** and **grizzly-compute** hostnames with correct IP.

4. Install & Configure NTP :

   - Install the package :

```
apt-get install -y ntp
```

   - Configure **/etc/ntp.conf** file :

```
server 192.168.0.1
```

   - Restart the service :

```
service ntp restart
```

Hypervisor

1. Install the packages that we need :

```
apt-get install -y kvm libvirt-bin pm-utils
```

2. Configure libvirt :

   - Edit **/etc/libvirt/qemu.conf** file and add :

```
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"]
```

- Disable **KVM default virtual bridge** to avoid any confusion :

```
virsh net-destroy default
virsh net-undefine default
```

- Allow **Live Migrations** :

  Edit **/etc/libvirt/libvirtd.conf** file :

```
listen_tls = 0
listen_tcp = 1
auth_tcp = "none"
```

  Modify libvirtd_opts variable in **/etc/init/libvirt-bin.conf** file :

```
env libvirtd_opts="-d -l"
```

  Edit **/etc/default/libvirt-bin** file :

```
libvirtd_opts="-d -l"
```

3. Restart libvirt :

```
service libvirt-bin restart
```

Nova

1. Install the packages :

```
apt-get install nova-compute-kvm
```

2. Configure Nova :

   - Edit **/etc/nova/api-paste.ini** file and modify :

```
auth_host = 192.168.0.1
admin_tenant_name = service
admin_user = nova
admin_password = password
```

   - Edit **/etc/nova/nova-compute.conf** file and modify :

```
[DEFAULT]
libvirt_type=kvm
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** file and modify :

```
[DEFAULT]

# MySQL Connection #
sql_connection=mysql://nova:password@192.168.0.1/nova

# nova-scheduler #
```

```
rabbit_host=192.168.0.1
rabbit_password=password
scheduler_driver=nova.scheduler.filter_scheduler.FilterScheduler

# nova-api #
cc_host=192.168.0.1
auth_strategy=keystone
s3_host=192.168.0.1
ec2_host=192.168.0.1
nova_url=http://192.168.0.1:8774/v1.1/
ec2_url=http://192.168.0.1:8773/services/Cloud
keystone_ec2_url=http://192.168.0.1:5000/v2.0/ec2tokens
api_paste_config=/etc/nova/api-paste.ini
allow_admin_api=true
ec2_private_dns_show_ip=True
dmz_cidr=169.254.169.254/32
ec2_dmz_host=192.168.0.1
metadata_host=192.168.0.1
metadata_listen=0.0.0.0
enabled_apis=metadata

# Networking #
network_api_class=nova.network.quantumv2.api.API
quantum_url=http://192.168.0.1:9696
quantum_auth_strategy=keystone
quantum_admin_tenant_name=service
quantum_admin_username=quantum
quantum_admin_password=password
quantum_admin_auth_url=http://192.168.0.1:35357/v2.0
libvirt_vif_driver=nova.virt.libvirt.vif.LibvirtHybridOVSBridgeDriver
linuxnet_interface_driver=nova.network.linux_net.LinuxOVSInterfaceDriver
firewall_driver=nova.virt.libvirt.firewall.IptablesFirewallDriver

# Compute #
compute_driver=libvirt.LibvirtDriver
connection_type=libvirt

# Cinder #
volume_api_class=nova.volume.cinder.API

# Glance #
glance_api_servers=192.168.0.1:9292
image_service=nova.image.glance.GlanceImageService

# novnc #
novnc_enable=true
novncproxy_base_url=http://7.7.7.7:6080/vnc_auto.html
vncserver_proxyclient_address=192.168.0.3
vncserver_listen=0.0.0.0

# Misc #
logdir=/var/log/nova
state_path=/var/lib/nova
lock_path=/var/lock/nova
root_helper=sudo nova-rootwrap /etc/nova/rootwrap.conf
verbose=true
```

- Restart Nova services :

```
service nova-compute restart
```

Quantum

Open vSwitch

1. Install the packages:

```
apt-get install -y openvswitch-switch
```

2. Start Open vSwitch service

```
service openvswitch-switch start
```

3. Configure Virtual Bridging

```
ovs-vsctl add-br br-int
```

Quantum

1. Install the packages :

```
apt-get install -y quantum-plugin-openvswitch-agent
```

2. Edit **/etc/quantum/quantum.conf** file and modify :

```
core_plugin = \
    quantum.plugins.openvswitch.ovs_quantum_plugin.OVSQuantumPluginV2
auth_strategy = keystone
fake_rabbit = False
rabbit_host = 192.168.0.1
rabbit_password = password
```

3. Edit **/etc/quantum/plugins/openvswitch/ovs_quantum_plugin.ini** file and modify :

```
[DATABASE]
sql_connection = mysql://quantum:password@192.168.0.1:3306/quantum
[OVS]
tenant_network_type = gre
tunnel_id_ranges = 1:1000
integration_bridge = br-int
tunnel_bridge = br-tun
local_ip = 10.10.10.2
enable_tunneling = True
```

4. Start the Agent :

```
service quantum-plugin-openvswitch-agent restart
```

Create your first VM

1. You can now use OpenStack API or the Dashboard to manage your own IaaS : http://192.168.0.1/horizon with demo / password credentials.

2. Edit the security group "Default" to allow ICMP and SSH.

3. Create a personal keypair.

4. In the Dashboard, go to "Instances" and click "Launch Instance" for spawning a new VM.

5. We have to configure floating IP (using demo tenant). To do that using the CLI, you need to get the ext_net ID and the port_id of your VM :

```
quantum net-list -- --router:external True
quantum port-list -- --device_id <vm-uuid>
```

6. Now, we are going to create a floating-IP attached to the virtual port of our VM and routed to the external network :

```
quantum floatingip-create --port_id <port_id> <ext_net_id>
```

7. That's it! You should be able to ping your VM using floating IP.

Conclusion

We have built a basic architecture for advanced testing purpose. This kind of architecture is close to the production, without High Availability (HA) and some services such as those for running OpenStack Object Storage. You can of course add as many compute nodes as you want. If you need more specific help, please read the official documentation of each project or write a post to an OpenStack mailing list.