Introduction

This is the second part in three part series of blog posts where I am sharing my experience installing Pivotal Cloud Foundry on Openstack. I will be installing PCF on open source version of openstack Kilo release.

In the first part, I have documented the following

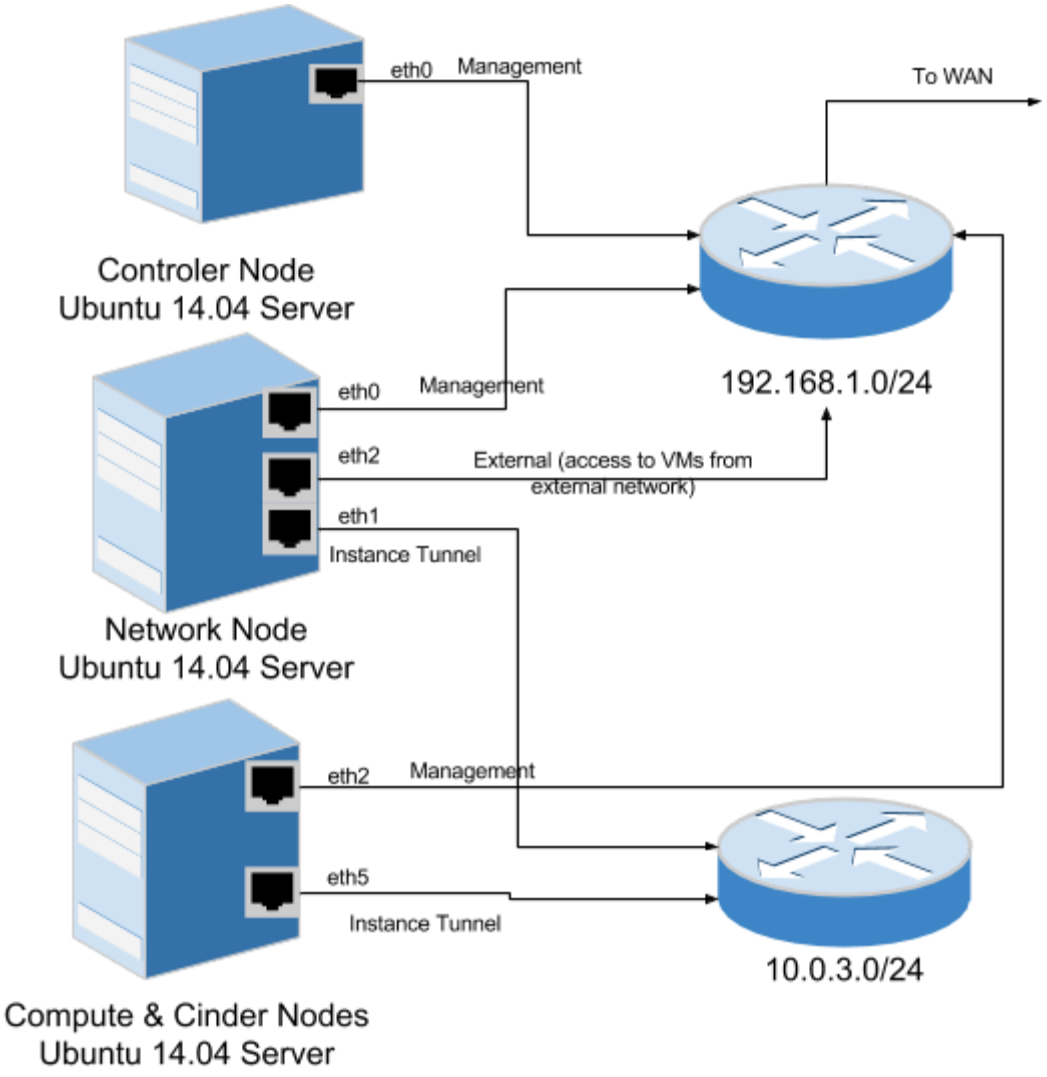
1. Basic server configuration
2. Basic network configuration
3. Installation and configuration of following openstack services
    a. Identity Service - Keystone
    b. Image Service - Glance
    c. Compute Service - Nova

In this blog post, I will explain

1. Installation and configuration of following openstack services
    a. Network Service - Neutron
    b. Block Storage Service - Cinder
    c. Dashboard Service - Horizon

Overview of Network configuration

To refresh our memory following is the physical network architecture from the first part of this series



Following is our network adapter configurations

Controller Node

```
auto eth0
iface eth0 inet static
address 192.168.1.8
netmask 255.255.255.0
gateway 192.168.1.1
```

Network Node

```
auto eth0
iface eth0 inet static
address 192.168.1.9
network 192.168.1.0
netmask 255.255.255.0

auto eth1
iface eth1 inet static
```

```
address 10.0.3.4
network 10.0.3.0
netmask 255.255.255.0

auto eth2
iface eth2 inet manual
       up ip link set dev $IFACE up
       down ip link set dev $IFACE down
```

Compute Node

```
auto eth2
iface eth2 inet static
address 192.168.1.2
netmask 255.255.255.0

auto eth5
iface eth5 inet static
address 10.0.3.5
network 10.0.3.0
netmask 255.255.255.0
```

Install and configure Controller Node

1. **Install and configure Neutron - Openstack Networking Service**

   The following script will configure

```
$ mysql -u root -p
       MariaDB [(none)]> CREATE DATABASE neutron;
       MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO
       'neutron'@'localhost' IDENTIFIED BY 'neutronDbPwd';
       MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%'
       IDENTIFIED BY 'neutronDbPwd';
       MariaDB [(none)]> exit
$ source admin-openrc.sh
$ openstack user create --password-prompt neutron
       <Enter neutronPwd for password>
$ openstack role add --project service --user neutron admin
$ openstack service create --name neutron --description "OpenStack Networking"
network
$ openstack endpoint create \
       --publicurl http://controller:9696 \
       --adminurl http://controller:9696 \
       --internalurl http://controller:9696 \
       --region RegionOne \
       network
# apt-get install -y neutron-server neutron-plugin-ml2 python-neutronclient
# vi /etc/neutron/neutron.conf
       [DEFAULT]
       ...
       rpc_backend=rabbit
       verbose = False
       auth_strategy = keystone
```

```
core_plugin = ml2
service_plugins = router
allow_overlapping_ips = True

notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
nova_url = http://controller:8774/v2

[database]
...
connection = mysql://neutron:neutronDbPwd@controller/neutron


[oslo_messaging_rabbit]
...
rabbit_host=controller
rabbit_userid=openstack
rabbit_password=rabbitPwd

[keystone_authtoken]
...
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = neutron
password = neutronPwd


[nova]
...
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
region_name = RegionOne
project_name = service
username = nova
password = novaPwd
# vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
...
type_drivers = flat,vlan,gre,vxlan
tenant_network_types = gre
mechanism_drivers = openvswitch

[ml2_type_gre]
...
```

```
        tunnel_id_ranges = 1:1000

        [securitygroup]
        …
        enable_security_group = True
        enable_ipset = True
        firewall_driver =
        neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver
# vi /etc/nova/nova.conf
        [DEFAULT]
        …
        network_api_class = nova.network.neutronv2.api.API
        security_group_api = neutron
        linuxnet_interface_driver =
        nova.network.linux_net.LinuxOVSInterfaceDriver
        firewall_driver = nova.virt.firewall.NoopFirewallDriver

        [neutron]
        …
        url = http://controller:9696
        auth_strategy = keystone
        admin_auth_url = http://controller:35357/v2.0
        admin_tenant_name = service
        admin_username = neutron
        admin_password = neutronPwd

        service_metadata_proxy = True
        metadata_proxy_shared_secret = testMetadataSecret

# su -s /bin/sh -c "neutron-db-manage --config-file /etc/neutron/neutron.conf
--config-file /etc/neutron/plugins/ml2/ml2_conf.ini upgrade head" neutron
# for a in nova-api neutron-server ; do service $a restart; done
```

2. Install and configure Cinder - Block Storage Service

```
$ mysql -u root -p
        MariaDB [(none)]> CREATE DATABASE cinder;
        MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO
        'cinder'@'localhost' IDENTIFIED BY 'cinderDbPwd';
        MariaDB [(none)]> GRANT ALL PRIVILEGES ON cinder.* TO 'cinder'@'%'
        IDENTIFIED BY 'cinderDbPwd';
        MariaDB [(none)]> exit

$ openstack user create --password-prompt cinder
        Password: cinderPwd
$ openstack role add --project service --user cinder admin
$ openstack service create --name cinder --description "OpenStack Block Storage"
volume
$ openstack service create --name cinderv2 --description "OpenStack Block
Storage" volumev2
```

```
$ openstack endpoint create \
        --publicurl http://controller:8776/v2/%\(tenant_id\)s \
        --internalurl http://controller:8776/v2/%\(tenant_id\)s \
        --adminurl http://controller:8776/v2/%\(tenant_id\)s \
        --region RegionOne \
        volume
$ openstack endpoint create \
        --publicurl http://controller:8776/v2/%\(tenant_id\)s \
        --internalurl http://controller:8776/v2/%\(tenant_id\)s \
        --adminurl http://controller:8776/v2/%\(tenant_id\)s \
        --region RegionOne \
        volumev2
# apt-get install -y cinder-api cinder-scheduler python-cinderclient
# vi /etc/cinder/cinder.conf
        [DEFAULT]
        verbose = True
        auth_strategy = keystone

        rpc_backend = rabbit
        my_ip = 192.168.1.8

        [database]
        connection = mysql://cinder:cinderDbPwd@controller/cinder

        [oslo_messaging_rabbit]
        rabbit_host = controller
        rabbit_userid = openstack
        rabbit_password = rabbitPwd

        [keystone_authtoken]
        auth_uri = http://controller:5000
        auth_url = http://controller:35357
        auth_plugin = password
        project_domain_id = default
        user_domain_id = default
        project_name = service
        username = cinder
        password = cinderPwd

        [oslo_concurrency]
        lock_path = /var/lock/cinder
# su -s /bin/sh -c "cinder-manage db sync" cinder
# for a in cinder-scheduler cinder-api ; do service $a restart; done
```

```
# rm -f /var/lib/cinder/cinder.sqlite
$ echo "export OS_VOLUME_API_VERSION=2" | tee -a admin-openrc.sh demo-openrc.sh
```

3. **Install and configure Horizon - Dashboard Service**

   Following scripts will install openstack dashboard service.

```
# apt-get install -y openstack-dashboard

# vi /etc/openstack-dashboard/local_settings.py
        ...
        OPENSTACK_HOST = "controller"
        ALLOWED_HOSTS = '*'
        CACHES = {
           'default': {
               'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
               'LOCATION': '127.0.0.1:11211',
           }
        }
        OPENSTACK_KEYSTONE_DEFAULT_ROLE = "user"
        TIME_ZONE = "US/Mountain"
# service apache2 reload
```

Install and configure Network Node

Network node provisions ip addresses to VMs and enables access to VMs from outside network. Neutron server is the primary component that enables this magic. Configuring openstack networking is the most complex operation in openstack installation, IMO. In this blog, I have used openvSwitch as neutron plugin, GRE protocol to for traffic to/from VMs.

```
# vi /etc/sysctl.conf
        net.ipv4.conf.default.rp_filter=0
        net.ipv4.conf.all.rp_filter=0
        net.ipv4.ip_forward=1
# sysctl -p
# apt-get install -y neutron-plugin-ml2 neutron-plugin-openvswitch-agent
neutron-l3-agent neutron-dhcp-agent neutron-metadata-agent
# vi /etc/neutron/neutron.conf
        [DEFAULT]
        ...
        rpc_backend = rabbit
        verbose = False
        auth_strategy = keystone
        core_plugin = ml2
        service_plugins = router
        allow_overlapping_ips = True

        [database]
        ...
        connection = mysql://neutron:neutronDbPwd@controller/neutron

        [oslo_messaging_rabbit]
```

```
        ...
        rabbit_host = controller
        rabbit_userid = openstack
        rabbit_password = rabbitPwd
# vi /etc/neutron/plugins/ml2/ml2_conf.ini
        [ml2]
        ...
        type_drivers = flat,vlan,gre,vxlan
        tenant_network_types = gre
        mechanism_drivers = openvswitch

        [ml2_type_flat]
        ...
        flat_networks = external

        [ml2_type_gre]
        ...
        tunnel_id_ranges = 1:1000

        [securitygroup]
        ...
        enable_security_group = True
        enable_ipset = True
        firewall_driver =
        neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver

        [ovs]
        ...
        #local_ip = INSTANCE_TUNNELS_INTERFACE_IP_ADDRESS
        local_ip = 10.0.3.4
        bridge_mappings = external:br-ex

        [agent]
        ...
        tunnel_types = gre
# vi /etc/neutron/l3_agent.ini
        [DEFAULT]
        ...
        verbose = False
        interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
        external_network_bridge =
        router_delete_namespaces = True
# vi /etc/neutron/dhcp_agent.ini
        [DEFAULT]
        ...

        verbose = False
        interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
        dhcp_driver = neutron.agent.linux.dhcp.Dnsmasq
        dhcp_delete_namespaces = True
```

```
        #Following line is optional
        dnsmasq_config_file = /etc/neutron/dnsmasq-neutron.conf
# vi /etc/neutron/dnsmasq-neutron.conf
        dhcp-option-force=26,1454
# pkill dnsmasq
# vi /etc/neutron/metadata_agent.ini
        [DEFAULT]
        ...
        auth_uri = http://controller:5000
        auth_url = http://controller:35357
        auth_region = RegionOne
        auth_plugin = password
        project_domain_id = default
        user_domain_id = default
        project_name = service
        username = neutron
        password = neutronPwd

        nova_metadata_ip = controller
        metadata_proxy_shared_secret = testMetadataSecret
# service openvswitch-switch restart
# ovs-vsctl add-br br-ex
# ovs-vsctl add-port br-ex eth2 <eth2 is INERFACE_NAME for external network>
# for a in openvswitch-switch neutron-plugin-openvswitch-agent neutron-l3-agent
neutron-dhcp-agent neutron-metadata-agent; do service $a restart; done
```

Install and configure Network Service in Compute Node

Following scripts will install and configure neutron network services in compute node

```
# vi /etc/sysctl.conf
        net.ipv4.conf.all.rp_filter=0
        net.ipv4.conf.default.rp_filter=0
        net.bridge.bridge-nf-call-iptables=1
        net.bridge.bridge-nf-call-ip6tables=1
# sysctl -p
# apt-get install -y neutron-plugin-ml2 neutron-plugin-openvswitch-agent
# vi /etc/neutron/neutron.conf
        //comment out any connection parameters in [database] section
        [DEFAULT]
        verbose = True
        rpc_backend = rabbit

        auth_strategy = keystone

        core_plugin = ml2
        service_plugins = router
        allow_overlapping_ips = True

        [database]
```

```
        ...
        connection = mysql://neutron:neutronDbPwd@controller/neutron

        [keystone_authtoken]
        auth_uri = http://controller:5000
        auth_url = http://controller:35357
        auth_plugin = password
        project_domain_id = default
        user_domain_id = default
        project_name = service
        username = neutron
        password = neutronPwd


        [oslo_messaging_rabbit]
        rabbit_host = controller
        rabbit_userid = openstack
        rabbit_password = rabbitPwd

# vi /etc/neutron/plugins/ml2/ml2_conf.ini
        [ml2]
        type_drivers = flat,vlan,gre,vxlan
        tenant_network_types = gre
        mechanism_drivers = openvswitch

        [ml2_type_gre]
        tunnel_id_ranges = 1:1000

        [securitygroup]
        enable_security_group = True
        enable_ipset = True
        firewall_driver =
        neutron.agent.linux.iptables_firewall.OVSHybridIptablesFirewallDriver

        [ovs]
        local_ip = 10.0.3.5

        [agent]
        tunnel_types = gre
# service openvswitch-switch restart
# vi /etc/nova/nova.conf
        [DEFAULT]
        ...
        network_api_class = nova.network.neutronv2.api.API
        security_group_api = neutron
```

```
        linuxnet_interface_driver = nova.network.linux_net.LinuxOVSInterfaceDriver
        firewall_driver = nova.virt.firewall.NoopFirewallDriver

        [neutron]
        ...
        url = http://controller:9696
        auth_strategy = keystone
        admin_auth_url = http://controller:35357/v2.0
        admin_tenant_name = service
        admin_username = neutron
        admin_password = neutronPwd
#
```

Install and Configure Cinder Services in Compute Node

Typically storage nodes are installed separately from compute nodes. However for this installation, we will use storage services in compute node. Following will install storage services.

```
# apt-get install -y lvm2
# pvcreate /dev/cciss/c0d1 <Use fdisk -l to find the correct device>
# vgcreate cinder-volumes /dev/cciss/c0d1
# vi /etc/lvm/lvm.conf
devices {
        filter = [ "a/c0d1/,r/.*/" ]
...

# apt-get install -y cinder-volume python-mysqldb
# vi /etc/cinder/cinder.conf
        [DEFAULT]
        verbose = False
        auth_strategy = keystone

        rpc_backend = rabbit
        my_ip = 192.168.1.2
        enabled_backends = lvm
        glance_host = controller

        [database]
        connection = mysql://cinder:cinderDbPwd@controller/cinder

        [oslo_messaging_rabbit]
        rabbit_host = controller
        rabbit_userid = openstack
        rabbit_password = rabbitPwd

        [keystone_authtoken]
```

```
auth_uri = http://controller:5000
auth_url = http://controller:35357
auth_plugin = password
project_domain_id = default
user_domain_id = default
project_name = service
username = cinder
password = cinderPwd

[lvm]
volume_driver = cinder.volume.drivers.lvm.LVMVolumeDriver
volume_group = cinder-volumes
iscsi_protocol = iscsi
iscsi_helper = tgtadm

[oslo_concurrency]
lock_path = /var/lock/cinder
# for a in tgt cinder-volume; do service $a restart; done
# rm -f /var/lib/cinder/cinder.sqlite
```

Verifying the Configuration

Finally we are ready to install PCF on our openstack installation. But before that, let us verify how we have done so far. If the verification steps are successful, it will be exciting. If not, good luck troubleshooting.

1. Create Internal network for demo tenant

   The following scripts will create internal network for demo tenant. All the virtual machines created in demo tenant will have ip addresses in 11.0.0.0/24 network.

```
$ source demo-openrc.sh
$ neutron net-create demo-net
$ neutron subnet-create demo-net 11.0.0.0/24 \
        --name demo-subnet --gateway 11.0.0.1 \
        --dns-nameserver 192.168.1.1
$ neutron net-list
+--------------------------------------+----------+-------------------------------------------------+
| id                                   | name     | subnets                                         |
+--------------------------------------+----------+-------------------------------------------------+
| 017af6a5-2adb-41bc-a678-7f8fc5fbe34c | demo-net | 38d97cc8-b17c-443e-a795-92b213a3a5e9 11.0.0.0/24 |
+--------------------------------------+----------+-------------------------------------------------+
$ neutron subnet-list
+--------------------------------------+-------------+-------------+--------------------------------------------+
| id                                   | name        | cidr        | allocation_pools                           |
+--------------------------------------+-------------+-------------+--------------------------------------------+
| 38d97cc8-b17c-443e-a795-92b213a3a5e9 | demo-subnet | 11.0.0.0/24 | {"start": "11.0.0.2", "end": "11.0.0.254"} |
+--------------------------------------+-------------+-------------+--------------------------------------------+
```

2. Create External Network

   The following scripts will create floating ip pool in the range 192.168.1.151 to 192.168.1.249

```
$ source admin-openrc.sh
```

```
$ neutron net-create ext-net --router:external --provider:physical_network
external --provider:network_type flat
$ neutron subnet-create ext-net 192.168.1.0/24 --name pcf-ext-subnet \
       --allocation-pool start=192.168.1.150,end=192.168.1.250 \
       --disable-dhcp --gateway 192.168.1.1 \
       --dns-nameserver 192.168.1.1
$ neutron subnet-show pcf-ext-subnet
+-------------------+-----------------------------------------------------+
| Field             | Value                                               |
+-------------------+-----------------------------------------------------+
| allocation_pools  | {"start": "192.168.1.150", "end": "192.168.1.250"} |
| cidr              | 192.168.1.0/24                                      |
| dns_nameservers   | 192.168.1.1                                         |
| enable_dhcp       | False                                               |
| gateway_ip        | 192.168.1.1                                         |
| host_routes       |                                                     |
| id                | 4aaf838f-847a-46ac-8fab-3d47728aa790                |
| ip_version        | 4                                                   |
| ipv6_address_mode |                                                     |
| ipv6_ra_mode      |                                                     |
| name              | pcf-ext-subnet                                      |
| network_id        | 0a316294-3b7d-4718-886c-e985872f5376                |
| subnetpool_id     |                                                     |
| tenant_id         | 9ae23c9777b444799424d4473052394a                    |
+-------------------+-----------------------------------------------------+
```

3. Create router
   Following script create router, adds the demo-subnet to the router's switch port and sets the ext-net network as gateway

```
$ source demo-openrc.sh
$ neutron router-create demo-router
$ neutron router-interface-add demo-router demo-subnet
$ neutron router-gateway-set demo-router ext-net
```

3. Create Security group that allow icmp and ssh to virtual machines

```
$ source demo-openrc.sh
$ nova secgroup-create demoSecurityGroup "Demo Security Group"
$ nova secgroup-add-rule demoSecurityGroup icmp -1 -1 0.0.0.0/0
$ nova secgroup-add-rule demoSecurityGroup tcp 22 22 0.0.0.0/0
```

4. Create key pair and save private key.

```
$ nova keypair-add demoKey > demoKey.pem
$ chmod 400 demoKey.pem
```

5. Create Virtual Machine
   Now that network is setup we are ready to spin up a virtual machine based on cirros image that we created in part 1 of this series. Following will create a virtual machine based on cirros image

```
$ nova boot --flavor m1.tiny --image cirros-0.3.4-x86_64 --key-name demoKey
--security-group demoSecurityGroup demoCirros
$ nova list
```

```
+--------------------------------------+------------+--------+------------+-------------+------------------+
| ID                                   | Name       | Status | Task State | Power State | Networks         |
+--------------------------------------+------------+--------+------------+-------------+------------------+
| 4086759e-9453-48af-ac45-0bf7d8b09402 | demoCirros | ACTIVE | -          | Running     | demo-net=11.0.0.3 |
+--------------------------------------+------------+--------+------------+-------------+------------------+
```

6. Create a volume

```
$ source demo-openrc.sh
$ cinder create --name demo-volume1 1
$ cinder list
+--------------------------------------+-----------+--------------+------+-------------+----------+-------------+
|                  ID                  |   Status  |     Name     | Size | Volume Type | Bootable | Attached to |
+--------------------------------------+-----------+--------------+------+-------------+----------+-------------+
| 7674195f-0bba-46a5-99eb-13b7696eefcd | available | demo-volume1 |  1   |     None    |  false   |             |
+--------------------------------------+-----------+--------------+------+-------------+----------+-------------+
```

7. Attach the volume to Virtual Machine

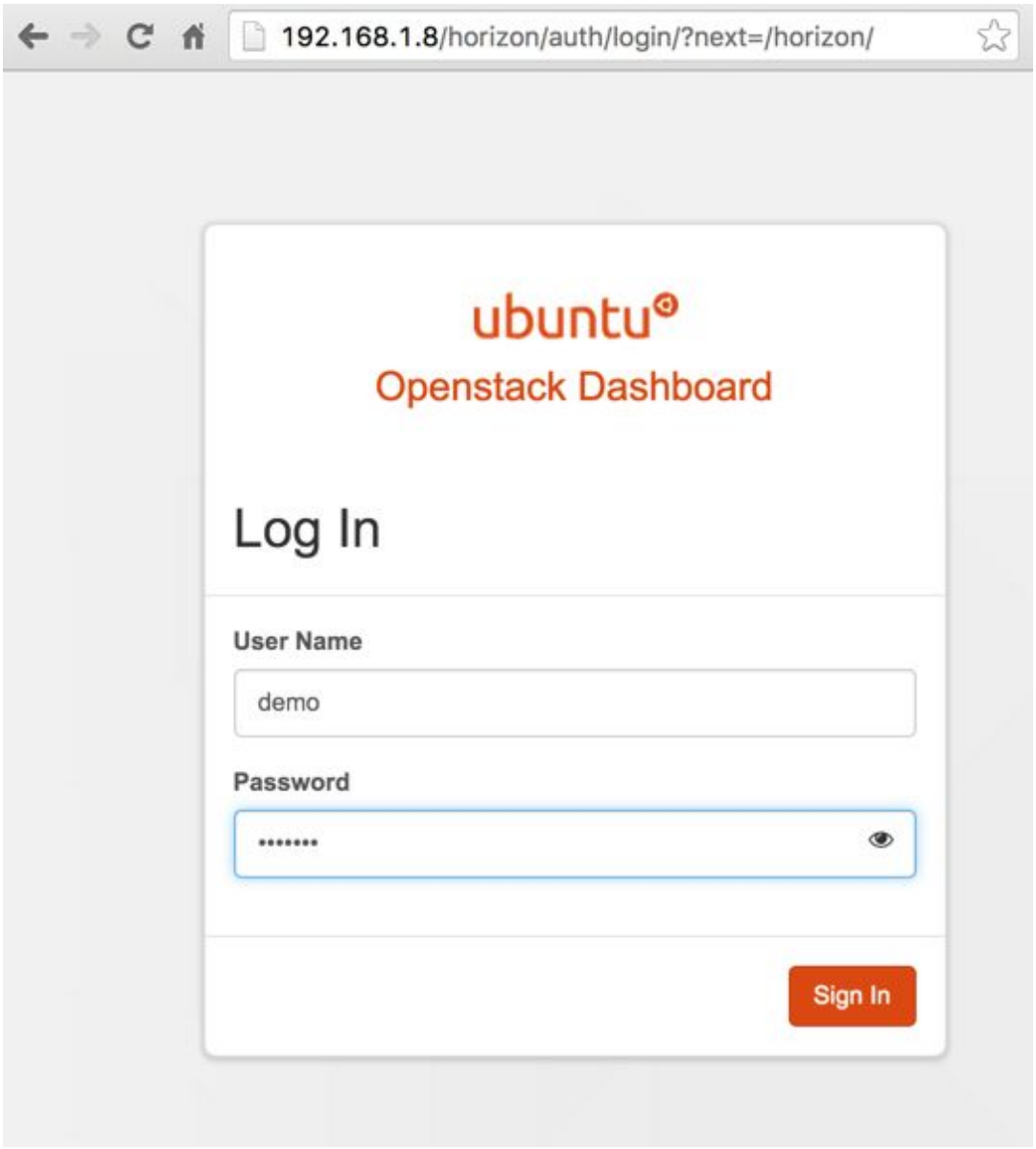
```
$ nova volume-attach demoCirros 7674195f-0bba-46a5-99eb-13b7696eefcd /dev/vdb
+----------+--------------------------------------+
| Property | Value                                |
+----------+--------------------------------------+
| device   | /dev/vdb                             |
| id       | 7674195f-0bba-46a5-99eb-13b7696eefcd |
| serverId | dbd6815c-74b1-4096-8743-1f170ccbcde8 |
| volumeId | 7674195f-0bba-46a5-99eb-13b7696eefcd |
+----------+--------------------------------------+
```

8. Allocate Floating IP, connect from outside network and view the attached volume

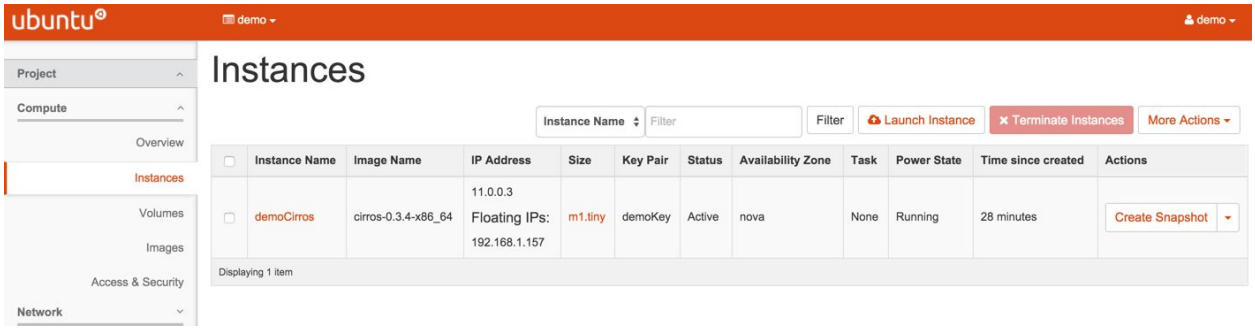
```
$ nova floating-ip-create ext-net
+--------------------------------------+--------------+-----------+----------+---------+
| Id                                   | IP           | Server Id | Fixed IP | Pool    |
+--------------------------------------+--------------+-----------+----------+---------+
| 0309a899-8df3-49ae-b035-8ea913df90c4 | 192.168.1.157 | -         | -        | ext-net |
+--------------------------------------+--------------+-----------+----------+---------+
$ nova floating-ip-associate demoCirros 192.168.1.157
$ nova list
+--------------------------------------+------------+--------+------------+-------------+---------------------------------+
| ID                                   | Name       | Status | Task State | Power State | Networks                        |
+--------------------------------------+------------+--------+------------+-------------+---------------------------------+
| 4086759e-9453-48af-ac45-0bf7d8b09402 | demoCirros | ACTIVE | -          | Running     | demo-net=11.0.0.3, 192.168.1.157 |
+--------------------------------------+------------+--------+------------+-------------+---------------------------------+
$ ssh -i demoKey.pem cirros@192.168.1.157
        The authenticity of host '192.168.1.157 (192.168.1.157)' can't be established.
        RSA key fingerprint is ac:2b:98:95:22:76:3b:35:13:a0:85:32:39:89:cd:7d.
        Are you sure you want to continue connecting (yes/no)? yes
        Warning: Permanently added '192.168.1.157' (RSA) to the list of known hosts.
        $ uname -a
                Linux democirros 3.2.0-80-virtual #116-Ubuntu SMP Mon Mar 23 17:28:52
                UTC 2015 x86_64 GNU/Linux
        $ ping -c 1 www.google.com
                PING www.google.com (216.58.217.36): 56 data bytes
                64 bytes from 216.58.217.36: seq=0 ttl=55 time=21.277 ms

                --- www.google.com ping statistics ---
                1 packets transmitted, 1 packets received, 0% packet loss
                round-trip min/avg/max = 21.277/21.277/21.277 ms
        $ sudo fdisk -l
                Disk /dev/vda: 1073 MB, 1073741824 bytes
                255 heads, 63 sectors/track, 130 cylinders, total 2097152 sectors
```

```
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

   Device Boot      Start         End      Blocks   Id  System
/dev/vda1   *       16065     2088449     1036192+  83  Linux

Disk /dev/vdb: 1073 MB, 1073741824 bytes
16 heads, 63 sectors/track, 2080 cylinders, total 2097152 sectors
Units = sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disk identifier: 0x00000000

Disk /dev/vdb doesn't contain a valid partition table
```

9. Login to Dashboard and view the instance, network, volume etc.
   Now that we have successfully created new virtual machine, we can now login to dashboard and see what we have done
   a. In browser window, go to http://192.168.1.8/horizon and login using demo/demoPwd

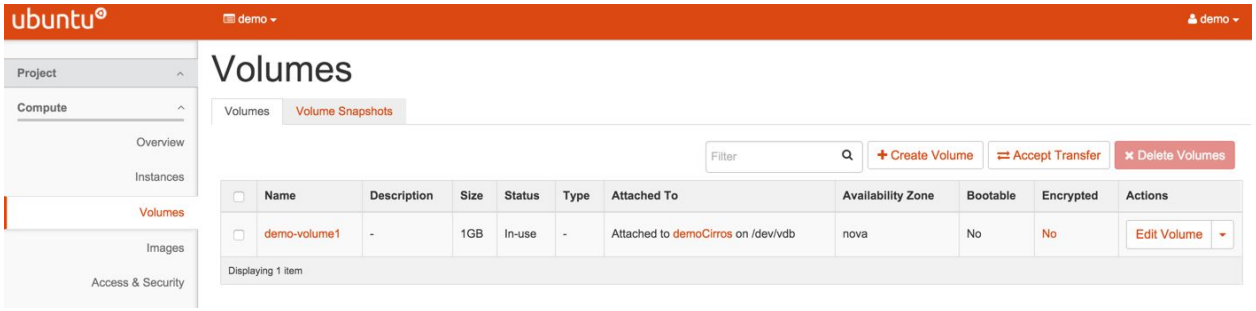


b. Go To instances and see the cirros instance that we created using command line

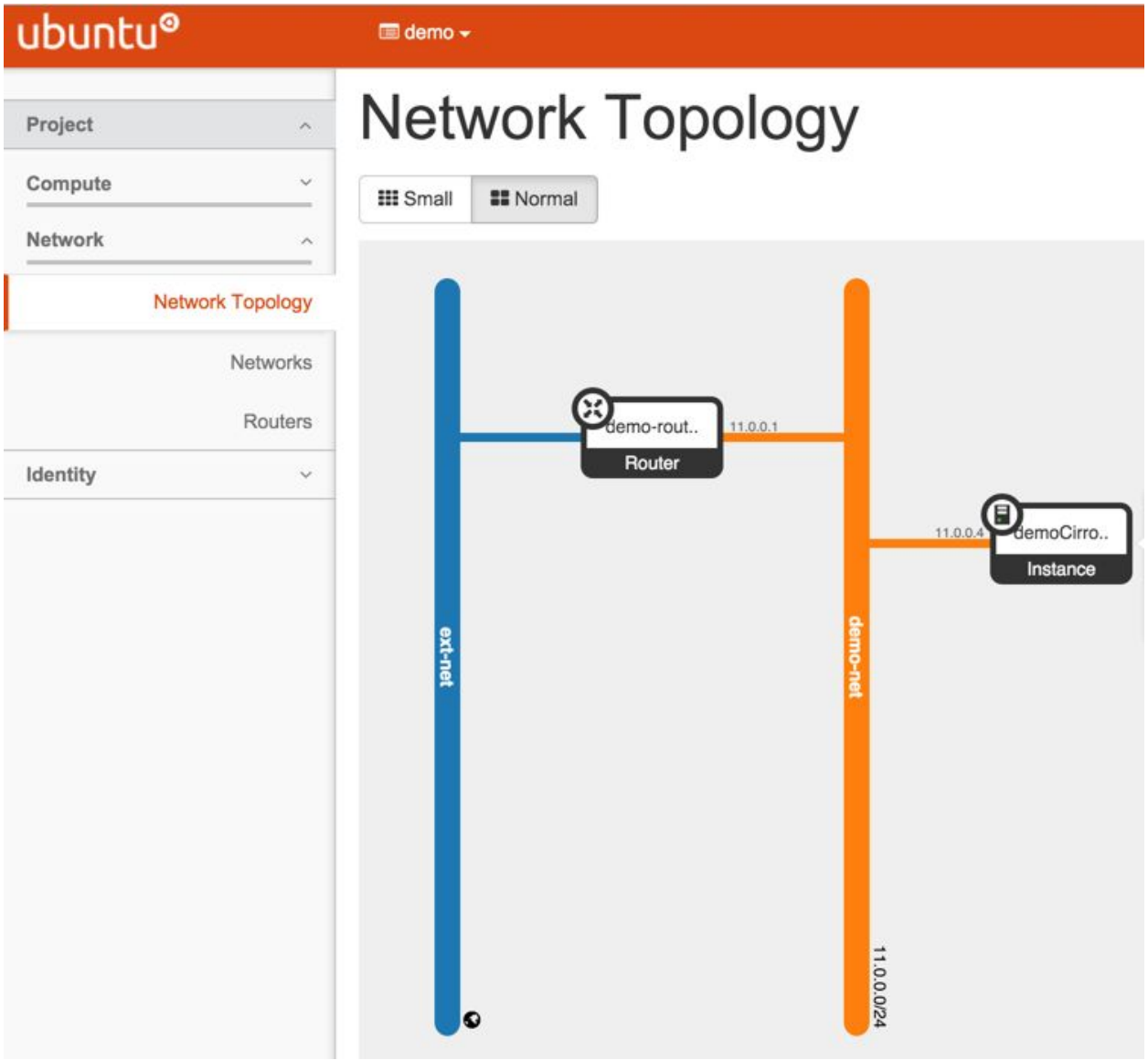


c. Explore other options and we can view the images, keypair, security rules etc.

d. Go to Volumes → Volumes to verify that the volume attached in the command line is visible as shown below



e. Go to Network -> Network Topology and view the network topology



Next Step

Now that we have fully operational openstack installation, we are finally ready to install Pivotal Cloud Foundry on openstack. The next series in this blog will prepare the openstack for installing PCF and install PCF.