GEORG-AUGUST-UNIVERSITÄT
GÖTTINGEN

Student Project

Openstack (Liberty) Integration with OpenDaylight (Lithium) with Devstack on Ubuntu 14.04LTS

Georg-August University, Göttingen

Computer Network Group

Prepared By:
Hari Raghavendar Rao Bandari
Matriculation No. 11334055

Co-Supervisor By:
M.Sc. Abhinandan S P
Computer Network Grp

Supervised By:
Dr. Mayutan Arumaithurai
Computer Network Grp

A Report on

Openstack (Liberty) Integration with OpenDaylight (Lithium) with Devstack on Ubuntu 14.04LTS

Submitted in Partial Fulfillment for the requirements of the course M.inf. 1809
(Berufsspezifische Schlüsselkompetenzen in einer forschungsbezogenen Projektarbeit)

In

M.Sc in Applied Computer Science

Submitted By:

Hari Raghavendar Rao Bandari

Matriculation No: 11334055

Under the Guidance of

Dr. Mayutan Arumaithurai

Computer Network Group, University of Göttingen

Department of Informatics

Faculty of Informatics and Computer Science

Georg-August-Universität,Göttingen

January 2016

The Abstract

"Openstack integration with OpenDaylight" is the project to integrate and setup the base platform which is required for OpenDaylight to provide Neutron as a Service. Neutron is a networking project in Openstack, which provides networking services. In order to handle networking services of openstack, Opendaylight is a software defined networking controller, It provides networking services to the openstack projects.

The main goal of this project was to setup a base platform so that Neutron as a service, analyze and test can be done without having to worry about the networking services - network, sub-network, ports, routing, Load balancing and Firewall as a service etc.

Openstack is the Open source Computing platform which is being used to set up large pool of compute, storage and networking resources within the data center. Therefore, all the resources can be supervised and controlled through the dashboard which provides administrators a full control over the above resources through the web interface(Horizon).

OpenDaylight is also open source software Defined Networking (SDN) project to build programmable networks and it is managed by a Linux foundation to advance SDN and to create a strong infrastructure for network function virtualization (NFV). it provides networking services to the "OpenStack Neutron" through certain bundles (OVSDB, VTN Manager, LISP etc.) with ODL APIs (REST) Interface. Hence ODL controller offers networking services to the openstack services through (Modular Layer 2) ML2 plugin as neutron as a backend. Therefore, we opted OVSDB bundle for opendaylight with Openstack integration.

Final Conclusion of the project is the Opendaylight provides a very easy and crucial networking services with the ease of deployment and use for end users.

TABLE OF CONTENTS

1 Introduction

OpenStack is open source software for building individual and public clouds. It is a cloud operating system which commands a large pool of compute, storage and networking resources within the data center. Therefore, all the resources can be supervised and controlled through the dashboard which provides administrators a full control over the above resources through the web interface.

OpenStack provides an Infrastructure-as-a-Service (IaaS) solution through a different service. Therefore, each service delivers an application programming interface (API) that address this merging. Therefore, list of services is available such as Dashboard, Compute, Block Storage, Networking, Object storage, Image service, Data processing service, Identity service, Orchestration, Telemetry, Database service.

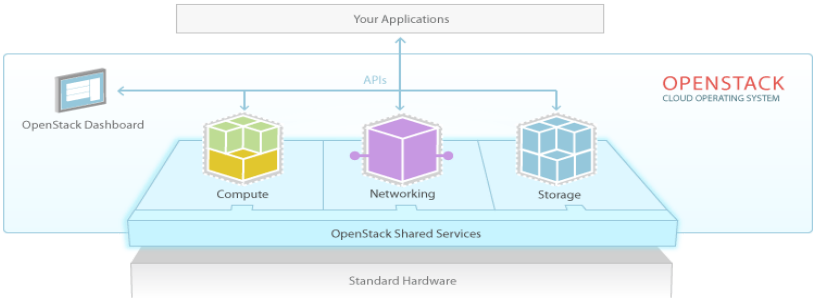


Figure 1: Openstack Architecture

OpenDaylight is also open source software Defined Networking (SDN) project to build programmable networks and it is managed by a Linux foundation to advance SDN and to create a strong infrastructure for network function virtualization (NFV). The extensive contribution to provide a functional SDN platform for users to deploy directly without need of other components. It concentrates on building a multi-vendor, multi-project ecosystem to encourage innovation and with a transparent approach towards SDN. Therefore, ODL can control Ethernet switches through OpenFlow protocols. As shown in the below image, it consists of different modules that can be combined as needed to fulfill the requirements of any given scenario. On other hand ODL provides users to evolve their own service architecture. It provides network services to cloud data center platforms such as OpenStack.

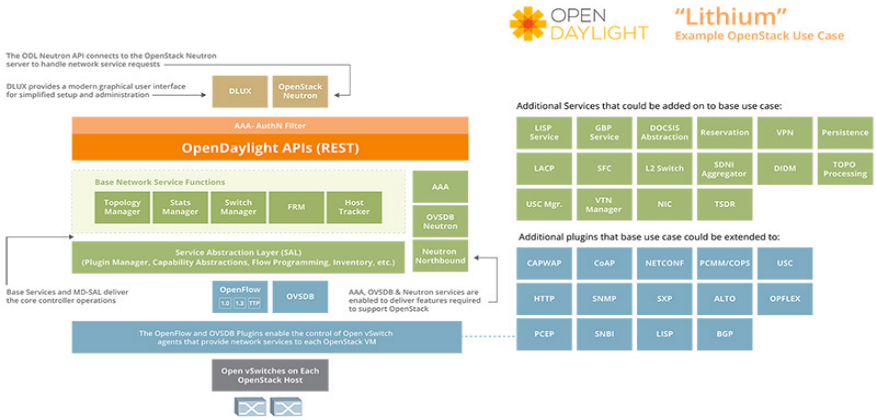


Figure 2: Opendaylight Architecture

From the above figure 2 the integration of the opendaylight with Openstack Neutron can be visualized. Therefore, above the design conveys that it provides networking services to the "OpenStack Neutron" through certain bundles (OVSDB, VTN Manager, LISP etc.) with ODL APIs (REST) Interface. Hence ODL controller offers networking services to the openstack services through (Modular Layer 2) ML2 plugin as neutron as a backend. Therefore, we opted OVSDB bundle for opendaylight with Openstack integration. Therefore, below diagram gives a graphical overview how openstack and opendaylight is integrated. All OpenDaylight core services are based on Java virtual machines (JVM) using karaf containers and OSGi framework.

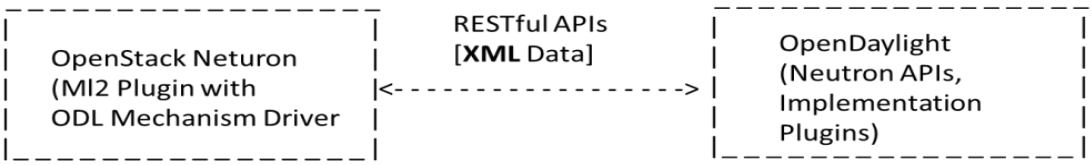


Figure 3: Openstack with Opendaylight

Above diagram represents Openstack Neutron consist of ML2 plugin with ODL Mechanism drivers act as RESTful proxy and passes all neutron APIs calls into Opendaylight. On other hand Opendaylight contains northbound REST service called Neutron API service which caches information from these proxies API calls and makes it available to other services inside of OpenDaylight. The extensive usage of ML2 plugin for OpenStack Neutron is a structure designed to take advantages of layer 2 networking technologies simultaneously, the fundamental concept behind the ML2 plugin is to classify the network type from the mechanism that understands the network type.

Therefore, ML2 plugin drivers contain different sets of network types (local, VXLAN, VLAN, flat and GRE), and along with respective mechanisms to access these networks.

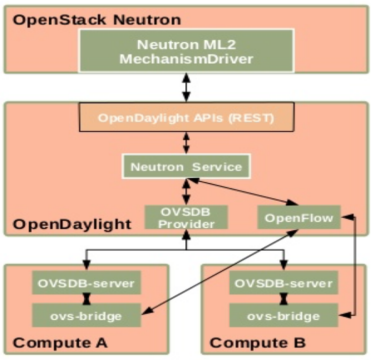


Figure 4: Neutron and Opendaylight

The OpenDaylight has various options to incorporate with OpenStack Neutron. Therefore, the northbound services intent to contribute network virtualization, it can be utilized as an implementation of Neutron network. Furthermore, Opendaylight has multiple options for Neutron API Implementation it includes OVSDB, It is used for southbound configuration of Open vSwitches on compute nodes and on other hand Opendaylight has northbound APIs to connect with Neutron to perform necessary operations. Hence, Opendaylight can handle network connectivity and setup GRE or VLAN tunnels for compute nodes. Basically OVSDB bundle is meant for implementation of southbound configuration of Open vSwitch database management protocol, it can also handle network virtualization such as Vxlan and GRE tunnels for OpenStack deployment.

Open vSwitch is also open source software resource which provides virtual switches environment to virtual machines. Therefore, Its major contribution was in OpenStack compute nodes, with the advantage of this technology the integration with OpenStack nodes and Opendaylight controller was quite painless. In this Integration Opendaylight manages respective OpenStack nodes switches through the OVSDB bundle plugin and openflow protocols. The Open vSwitch provides three bridges such as br-int, br-ex, br-tun. The openstack virtual machines get ports on br-int, br-ex is used for actual network traffic for external network to provide internet services to compute instances and br-tun is used for the tunnel interfaces between instances. Therefore, Open vSwitch creates flows rules with virtual patch cables between br-ex and br-int to provide connectivity. Furthermore, physical interfaces are added to br-ex, to creates the

management ports with the type internal so Linux can add IP address to it and iptables to perform L3 forwarding and NAT.

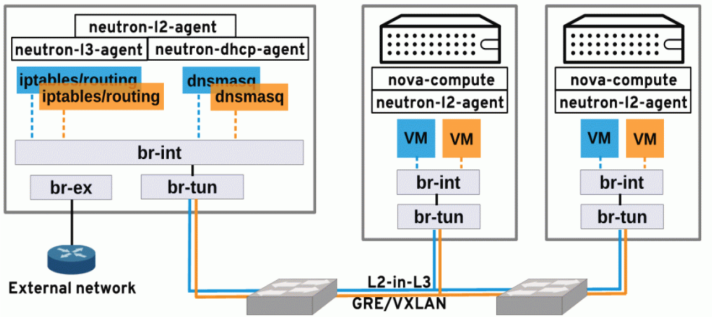


Figure 5: Openstack Neutron Architecture

2 Requirements of OpenStack and OpenDaylight

Openstack can install on Linux operating systems. Therefore, I consider Minimal Ubuntu Server Version 14.04 LTS currently which is more stable. To setup openstack for testing purpose we required three node controller node, compute node and last but not least opendaylight node, it handles all network operations such as port, network, subnet, router etc. Furthermore, you can run above OpenStack and Opendaylight services in a single machine for simplicity. To gain more knowledge about each services, I considered to execute three respective node on separate individual machine. OpenStack has many flavours of releases currently “Liberty” is more stable and with some new features services but in this deployment only on main services are focused. They are

1. Nova - Compute Service
2. Glance - Imaging Service
3. Keystone - Identity Service
4. Horizon - UI Service
5. Neutron - Networking Service

From above services Openstack Neutron service is handled by the Opendaylight to create network, subnet, routers, ports etc. OpenDaylight have different flavours of releases currently “Lithium“ is more stable. Hence, Opendaylight can provide L3 Forwarding services, Firewall as a service, Load balancing as a service to OpenStack instances. Therefore, entire setup provides L3 forwarding services along with ML2 services to OpenStack

Requirements

1. Laptop at least with 8GB RAM.
2. VirtualBox
3. Minimal Ubuntu Server 14.04 LTS
4. Devstack
5. XQuartz (X11 Forwarding) [**Note : Install on Host Machine**]
6. Wireshark (Packet Tracer)

The Environment used for this guide:

1. A 64-Bit Intel Core i5 Laptop, 16Gb Ram.
2. OS X El Capitan version 10.11.1 ("Host Machine")
3. VirtualBox 5

3 Installation Minimal Ubuntu Server 14.04 LTS on Virtualbox

Step 1: Download and install VirtualBox from official website with respective to the host operating system i.e. https://www.virtualbox.org/wiki/Downloads .

Step 2: Download "MinimalCD Ubuntu Server 14.04 LTS" from official website with appropriate stable version https://help.ubuntu.com/community/Installation/MinimalCD

Step 3: Once virtualBox is installed on the Host Machine. Start the VirtualBox.

- Click **New** button from VirtualBox Console and follow Instruction below.

  Name: Controller

  Type: Linux

  Operating System : Ubuntu 64bit.

Note: Host machine should have Virtualization technology is enabled then only 64bit is allowed to select otherwise 32bit.

- Click **Continue** button and provide **Memory Size: 2GB(2048 MB**) .
- Click **Continue** button and Select **Create a virtual hard disk now**.
- Click **Continue** button and Select **VDI(Virtualbox Disk Image)** .
- Click **Continue** button and Select **Dynamically allocated** .
- Click **Continue** button and Select **Provide 20GB Size** of the virtualbox.

- Click **Create** button. Now start **Controller Node** virtual image and select the **Minimal CD ISO** image file from the respective folder.
- Click **Open** button and follow the **Ubuntu installation instruction** from the screen.

Step 4 : Installation process for the Ubuntu Server 14.04 LTS.

- Select **Install Ubuntu Server** and press **ENTER** key.
- Select Ubuntu Operating system Language : **English** and Press **ENTER** key.
- Select Your Location from the screen if not available select **Other** and select Region selection and select the country and press **ENTER** key.
- Now select Locale setting from the screen and press **ENTER** key and select default setting.
- Now select keyboard setting with **no** option and manually select the keyboard layout.
- Now provide **Host Name** to machine i.e. **controller**.
- Now provide **User Name** i.e. **stack** provide password twice for confirmation.
- Now select **no** option for encryption for simplicity.
- Now select respective option to set the time zone if it is correct select **yes** if not **no** and select particular city and press ENTER key.
- Now select **guided - use entire disk** option from the screen and select the hard drive you wish to use and press **ENTER** key.
- Again press **ENTER** key to begin installation. Therefore, at particular stage of installation it will ask proxy setting press **ENTER** key.
- Now it will prompt security updates select **no automatic updates** (depends on user choice).
- Now it prompt with list of services select with **TAB** key **Basic Ubuntu server** and **OpenSSH server** and press **ENTER** key.
- Now press **ENTER** key i.e. **yes** when it ask about GRUB Boot loader.
- Now **Installation** is **completed**. (**Note** : Remove ISO file from respective directory and click **continue** otherwise it will again prompt with starting screen i.e. **installation ubuntu server**) .

step 5: Before going further we required **Host only Network** adapter to ssh into virtualBox.

- Press **VirtualBox** it will provide drop down menu.
- Click on **preferences** under select **Network** and click on **host-only Networks** option.
- Now on right side select first image with plus(+) sign it will create "**vboxnet0**".
- Now double click on **vboxnet0** it will prompt with a dialog box shown below.

Figure 6: VirtualBox DHCP Settings

step 6: VirtualBox Network Settings for Controller node

INTERFACE NAME	**VIRTUALBOX NETWORK ADAPTER**	**COMMENTS**
eth0 (Additional Interface)	NAT	This Interface used for Internet access to download packages etc.
eth1(Data Network)	Internal Network	This Interface used for data network between openstack nodes to communicate privately it use same network name i.e. "**vmdata"**. We should set Promiscuous mode to "**Allow All**"
eth2(External Network)	Internal Network	This Interface used for external network which provides internet to openstack vms which use same network name i.e. "**external"**. We should set Promiscuous mode to "**Allow All**"
eth3(Management Network)	Host-only	OpenStack nodes is used for API requests and responses. Typically, it is also used to access the Horizon GUI and at same time to execute CLI via SSH.

4 VirtualBox Configuration Settings

4.1 Controller Node:

step 7: select the **Controller** machine and right click on it and select **Settings** option and select **Network** option. Follow below setting.

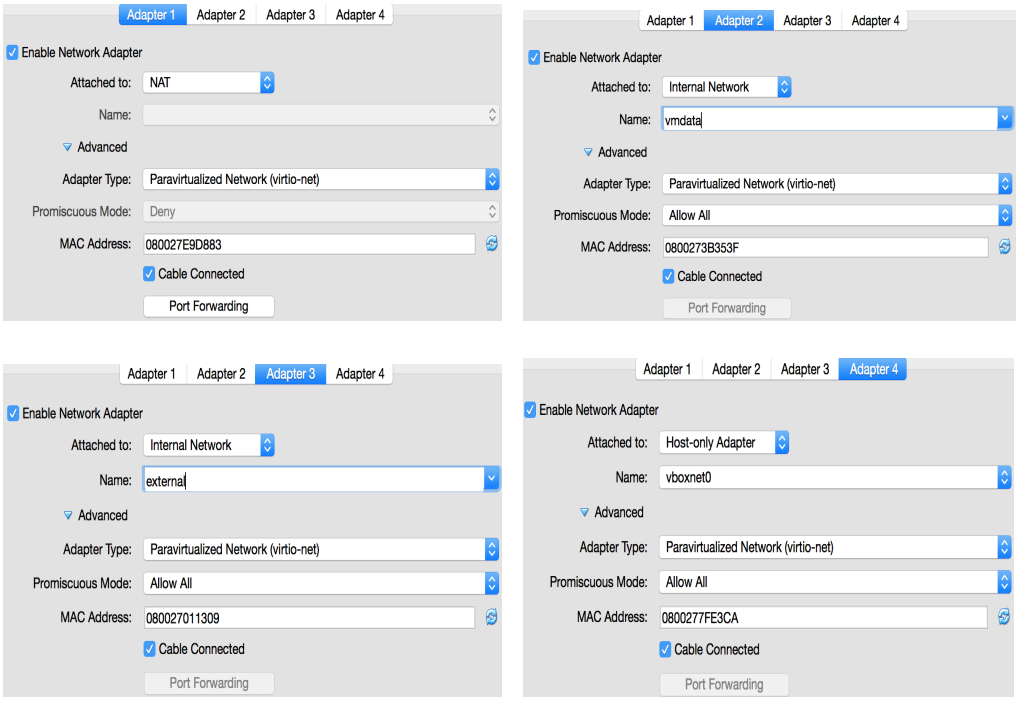


Figure 7: VirtualBox Network Settings

step 8: Now login with credential and execute below command to configure interfaces.

```
$ sudo apt-get install vim bridge-utils git-core iptables-persistent wget net-tools tar unzip
$ sudo vim /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface (Internet)
auto eth0
iface eth0 inet dhcp

# Internal Data Network for Tenant Traffic
auto eth1
iface eth1 inet static
address 192.168.254.32
```

```
netmask 255.255.255.0

# External Network for floating IP's
auto eth2
iface eth2 inet manual
    up ip link set dev $IFACE up
    down ip link set dev $IFACE down

# Management, Reachable from host via vboxnet0(Host-only Network)
auto eth3
iface eth3 inet dhcp
```

step 9: Now configure hosts and execute below command to edit the file.

```
$ sudo vim /etc/hosts
```

```
127.0.0.1 localhost controller

192.168.254.32 controller
192.168.254.33 computenode1
192.168.50.103 odl
```

step 10: Now Restart the Ubuntu Machine. To do execute below command.

```
$ sudo reboot
```

step 11: Now once machine has restarted login with credentials and execute below command.

```
$ ifconfig
$ echo "stack      ALL=(ALL)      NOPASSWD: ALL" >> /etc/sudoers
```

Now every interface will have an IP address except "**eth2**" interface.

step 12: Now **power off** the machine, just select particular machine and right click on it and select **close** then **power off.**

step 13: Now select the **controller** virtual machine and right click on it and select **clone** option and name it as **computenode1** and select **Reinitialize the MAC address for all network cards** and click **continue**.

step 14: Now Perform **step 13** again but name it as **odl**.

step 15: Now Perform **step 13** again but name it as **router-node**.

Note: After executing above 15 steps, virtualbox contains 4 virtual machine i.e. controller node, computenode1, odl and router-node.

4.2 Compute Node:

Now login with Credentials and execute the following commands

```
$ sudo vim /etc/hostname
```

Change controller to computenode1

```
computenode1
```

```
$ sudo vim /etc/hosts
```

```
127.0.0.1  localhost computenode1

192.168.254.33 computenode1
192.168.254.32 controller
192.168.50.103 odl
```

```
$ sudo vim /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface (Internet)
auto eth0
iface eth0 inet dhcp

# Internal Data Network for Tenant Traffic
auto eth1
iface eth1 inet static
address 192.168.254.33
netmask 255.255.255.0

# External Network for Floating IP's
auto eth2
iface eth2 inet manual
    up ip link set dev  $IFACE up
    down ip link set dev $IFACE down

# Management, Reachable from host via vboxnet0(Host-only Network)
auto eth3
iface eth3 inet dhcp
```

```
$ sudo reboot
```

After reboot again login with credentials

```
$ ifconfig
```

Above command will show interfaces with the IP address assigned to it.

Note: You can add more compute nodes to controller node based on your Host Machine RAM memory and execute same above commands.

4.3 OpenDaylight Node

Before login into Opendaylight machine we need to change network settings for odl as we required two interfaces.

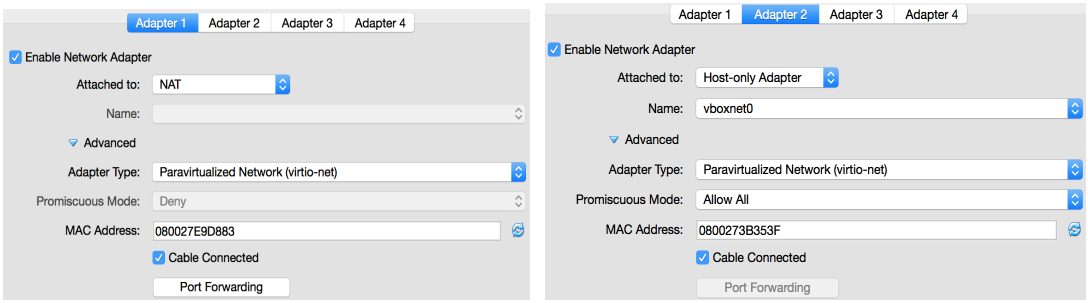


Figure 8: VirtualBox Network Settings

Now login with Credentials and execute the following commands

```
$ sudo vim /etc/host
```

Change controller to odl

```
odl
```

```
$ sudo vim /etc/hosts
```

```
127.0.0.1  localhost odl

192.168.50.103 odl
```

```
$ sudo vim /etc/network/interfaces
```

```
# The loopback network interface
auto lo
```

```
iface lo inet loopback

# The primary network interface (Internet)
auto eth0
iface eth0 inet dhcp

# Management, Reachable from host via vboxnet0(Host-only Network)
auto eth1
iface eth1 inet dhcp
```

```
$ sudo reboot
```

After reboot again login with credentials

```
$ ifconfig
```

4.4 Router Node

Before login into router-node machine we need to change network settings for router-node as we required two interfaces.

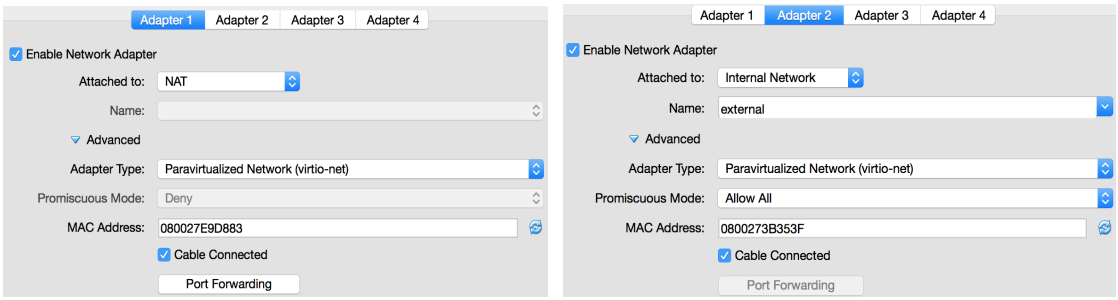


Figure 9: VirtualBox Network Settings

Now login with Credentials and execute the following commands

```
$ sudo vim /etc/host
```

Change controller to router-node

```
router-node
```

```
$ sudo vim /etc/hosts
```

```
127.0.0.1  localhost router-node
```

```
192.168.56.1 router-node_if
```

```
$ sudo vim /etc/network/interfaces
```

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface (Internet)
auto eth0
iface eth0 inet dhcp

# External Network for Floating IP's
auto eth1
iface eth1 inet static
address 192.168.56.1
netmask 255.255.255.0
```

```
$ sudo reboot
```

After reboot again login with credentials

```
$ ifconfig
```

After each node is configured below Topology details will be observed:

VM	**Services**	**eth0** **VB NAT**	**eth1** **VB Internal 1**	**eth2** **VB Internal 2**	**eth3** **VB vboxnet0**
controller	controller	10.0.2.15	192.168.254.32	0.0.0.0	192.168.50.102
computenode1	compute	10.0.2.15	192.168.254.33	0.0.0.0	192.168.50.104
odl	ODL	10.0.2.15	192.168.50.103 **VB vboxnet0**	NA	NA
router-node	router,dhcp	10.0.2.15	192.168.56.1 **VB Internal 2**	NA	NA

Note: Here odl, router-node VM has two interfaces. So typically eth1 interface VB type is different. As above configuration for internal networking interfaces, it does not require a gateway in linux interfaces configuration file.

5 Installation of Open vSwitch

Install Open vSwitch on below Node

5.1 Controller Node

5.2 Compute Node

5.3 Opendaylight Node

Note: If you just want to try ODL with OpenStack through ML2 (Modular Layer 2) plugin simply perform below command through command line interface $ sudo apt-get install openvswitch-switch then it will install openvswitch version 2.0.1. Therefore if you would like to assign further more interesting jobs to ODL like L3 (layer 3) routing, Firewall-as-service, Load Balancing-as-service then we required advanced openvswitch version >=2.3.1.

To install Openvswitch on above three nodes . Therefore Execute below commands through command line.

Remove old Openvswitch (if it is previously installed)

```
$ sudo apt-get remove openvswitch-common openvswitch-datapath-dkms openvswitch-switch
```

Download the new openvswitch source tarball

```
$ mkdir openvswitch
$ cd openvswitch
$ wget http://openvswitch.org/releases/openvswitch-2.3.1.tar.gz
$ tar zxvf openvswitch-2.3.1.tar.gz
$ cd openvswitch-2.3.1
```

Now install all the necessary dependencies which are required

```
$ sudo apt-get update
$ sudo apt-get -y install build-essential fakeroot debhelper autoconf automake libssl-dev pkg-config bzip2 openssl python-all procps python-qt4 python-zopeinterface python-twisted-conch
```

Build the Debian packages:

```
$ DEB_BUILD_OPTIONS='parallel=2 nocheck' fakeroot debian/rules binary
```

Install the packages:

```
$ cd ..
$sudo dpkg -i openvswitch-common*.deb openvswitch-datapath-dkms*.deb openvswitch-pki*.deb openvswitch-switch*.deb
$ sudo /etc/init.d/openvswitch-switch start
```

Note: If you are facing any connection database error please run above commands again, it connects to database for second time.

5.4 Installing Wireshark

Now execute the below commands to install wireshark

```
$ sudo apt-get install wireshark xorg xauth
$ sudo vim /etc/ssh/sshd_config

# change the x11forwarding to yes

X11Forwarding no ---> X11Forwarding Yes

# Link root's .XAuthority to stack
$ ln -s /home/stack/.XAuthority  /root/
```

Note: Optional to install wireshark on devstack controller Node, compute Node and Opendaylight Node.

6 Linux Network Configurations

6.1 Router Node

This router-node is acting as "router" for neutron L3 Agent, It requires to enable packet forwarding including particular iptables firewall rules to provide internet access to openstack compute node and simultaneously disable default firewall rules Uncomplicated Firewall (UFW) .

```
$ sudo vim /etc/sysctl.conf
```

Now copy and paste below code:

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

```
net.ipv4.ip_forward=1
net.ipv4.conf.default.accept_source_route = 0
```

```
$ sudo vim iptable.sh
```

Now copy and paste below code:

```
#!/bin/bash

sudo service iptables-persistent flush
sleep 1
sudo iptables -P INPUT DROP
sleep 1
sudo iptables -P FORWARD ACCEPT
sleep 1
sudo iptables -A INPUT -i lo -j ACCEPT
sudo iptables -A INPUT -m state --state RELATED,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -i eth0 -p udp -s 0/0 --dport 53 -j REJECT
sudo iptables -A INPUT -p udp --dport 53 -m state --state NEW,ESTABLISHED -j ACCEPT
sudo iptables -A INPUT -p tcp -m tcp --dport 22 -j ACCEPT
sudo iptables -A INPUT -p icmp -j ACCEPT
sudo iptables -A INPUT -j REJECT --reject-with icmp-host-prohibited
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sleep 1
sudo service iptables-persistent save
sleep 1
sudo service iptables-persistent restart
```

Now to execute above changes it required to run below bash files.

```
$ sudo vim ufwdisable.sh
```

Now copy and paste below code:

```
#!/bin/bash

sudo apt-get update
sudo apt-get upgrade -y
sudo ufw disable
sudo sysctl -p
sudo service iptables-persistent start
```

6.2 Opendaylight Node

Firstly, It is necessary to enable packet forwarding. Therefore, execute below command to edit specific lines in configuration file.

```
$ sudo vim /etc/sysctl.conf
```

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

Now, After above changes It is not yet enabled packet forwarding simply by running specific command and simultaneously it requires to stop OpenVswitch and empty the Database of open vSwitch. Therefore, ODL get full control over it and execute the below bash file.

```
$ sudo vim ovs.sh
```

Now paste below code:

```
#!/bin/bash

sudo sysctl -p
sudo ufw disable
sudo apt-get update -y
sudo apt-get dist-upgrade -y
sudo service openvswitch-switch stop
sudo rm -rf /var/log/openvswitch/*
sudo rm -rf /etc/openvswitch/conf.db
sudo service openvswitch-switch start
```

Above bash file contains linux commands and it requires to stop default firewall services (UFW) and it's important to consider iptables as default firewall rules. Simultaneously it is best practise to update the packages.

6.3 Controller Node:

Firstly, network issues may generate problems from accessing our cloud instances, It is necessary that compute instances should able to forward packets from public interface to the bridge interface. Therefore, execute below command to edit specific lines in configuration file.

Note: In devstack installation control node is also compute node.

```
$ sudo vim /etc/sysctl.conf
```

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

Now, After above changes It is not yet enabled packet forwarding simply by running specific command and simultaneously it requires to stop Open vSwitch and empty the Database of open vSwitch. Therefore, ODL get complete control over it. So execute the below bash file.

```
$ sudo vim ovs.sh
```

Now paste below code:

```
#!/bin/bash

sudo sysctl -p
sudo ufw disable
sudo apt-get update -y
sudo apt-get dist-upgrade -y
sudo service openvswitch-switch stop
sudo rm -rf /var/log/openvswitch/*
sudo rm -rf /etc/openvswitch/conf.db
sudo service openvswitch-switch start
```

Above bash file contains linux commands, it will disable default firewall services (UFW) and enables iptables as default firewall rules. Simultaneously it is best practise to update the packages. Therefore, ODL manages all the nodes.

6.4 Compute Node:

Firstly, network issues may generate problems from accessing our cloud instances, It is necessary that compute instances should able to forward packets from public interface to the bridge interface. Therefore, execute below command to edit specific lines in configuration file.

```
$ sudo vim /etc/sysctl.conf
```

```
net.ipv4.conf.default.rp_filter=0
net.ipv4.conf.all.rp_filter=0
net.bridge.bridge-nf-call-iptables=1
net.bridge.bridge-nf-call-ip6tables=1
net.ipv4.ip_forward=1
```

Now, After above changes It is not yet enabled packet forwarding simply by running specific command and simultaneously it requires to stop Open vSwitch and empty the Database of open vSwitch. Therefore, ODL get complete control over it. So execute the below bash file.

```
$ sudo vim ovs.sh
```

```
#!/bin/bash

sudo sysctl -p
sudo ufw disable
sudo apt-get update -y
sudo apt-get dist-upgrade -y
sudo service openvswitch-switch stop
sudo rm -rf /var/log/openvswitch/*
sudo rm -rf /etc/openvswitch/conf.db
sudo service openvswitch-switch start
```

Above bash file contains linux commands, it will disable default firewall services (UFW) and enables iptables as default firewall rules. Simultaneously it is best practise to update the packages. Therefore, ODL manages all the nodes.

7 Installing OpenDaylight on OpenDaylight Node

Now login with respective credentials and execute below commands to install java.

```
# sudo apt-get install openjdk-7-jdk
```

We required Maven to build ODL. Install the most recent version of Maven

```
# sudo mkdir -p /usr/local/apache-maven
```

Now download the maven source code

```
#wget
http://ftp.wayne.edu/apache/maven/maven-3/3.3.3/binaries/apache-maven-3.3.3
-bin.tar.gz
```

Now install the Maven by executing below commands.

```
# sudo mkdir /usr/local/apache-maven
# sudo mv apache-maven-3.3.3-bin.tar.gz /usr/local/apache-maven
# sudo tar -xzvf /usr/local/apache-maven/apache-maven-3.3.3-bin.tar.gz -C
/usr/local/apache-maven/
#sudo update-alternatives --install /usr/bin/mvn mvn
/usr/local/apache-maven/apache-maven-3.3.3/bin/mvn 1
# sudo update-alternatives --config mvn
# sudo apt-get install vim
# vim ~/.bashrc
```

We need to add below lines in ~/.bashrc file.

```
export M2_HOME=/usr/local/apache-maven/apache-maven-3.3.3
export MAVEN_OPTS="-Xms256m -Xmx512m" # Very important to put the "m" on the end
export JAVA_HOME=/usr/lib/jvm/java-7-openjdk-amd64 # This matches sudo update-alternatives --config java
```

Now run below command

```
source ~/.bashrc
```

Now configuring Opendaylight to so we need to download the most recent version of ODL. Therefore, perform these below commands.

```
$wget
https://nexus.opendaylight.org/content/groups/public/org/opendaylight/integration/distribution-karaf/0.3.3-Lithium-SR3/distribution-karaf-0.3.3-Lithium-SR3.tar.gz
```

Now Uncompress the downloaded file with below commands.

```
$ tar xvfz distribution-karaf-0.3.3-Lithium-SR3.tar.gz
```

Now enter into distribution file and start the ODL controller

```
$ cd distribution-karaf-0.3.3-Lithium-SR3
$ sudo vim etc/custom.properties
```

```
# enabling L3 forwarding in ODL
ovsdb.l3.fwd.enabled=yes
# providing router-node mac address to ODL node
ovsdb.l3gateway.mac=08:00:27:fb:34:e1[ router-node MAC address]
```

```
$ ./bin/karaf
```

Now ODL controller console will start see the below figure.

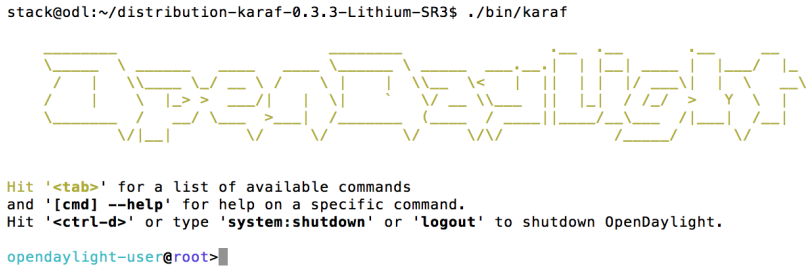


Figure 10: OpenDaylight Console

Now install features in the Opendaylight console which are required for the openstack. Therefore execute below commands in ODL console

```
feature:install odl-ovsdb-openstack odl-restconf odl-mdsal-apidocs odl-dlux-core
```

It will take few minutes to install all feature and load them. On other hand open below link in your “Web Browser”

```
http://<controller-ip>:8181/index.html

http://192.168.50.103:8181/index.html
```

Now please login with credentials

Username: admin Password: admin

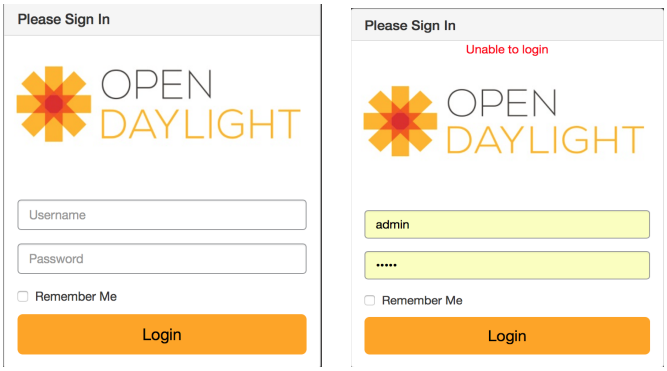


Figure 11: OpenDaylight DLUX UI

Note: Sometimes you can’t login into ODL web page because it take 5-7 minutes to load all the features. Therefore, as soon as ODL controller up and running please wait for 5-7 minutes and then login with above credentials.

After successful login below figure will be seen

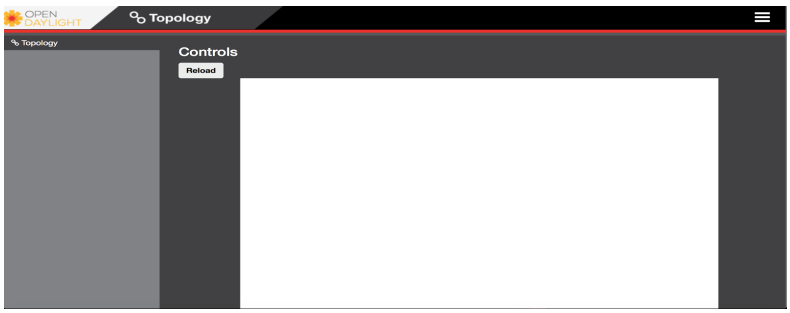


Figure 12: OpenDaylight DLUX UI

The above figure indicates that all the features which are installed are been loaded and ODL controller is ready for experiment.

8 Installing Openstack

Installing openstack for testing purpose we are using devstack as the deployment resource to install on ubuntu server 14.04LTS. It is useful to have minimal multi-node Openstack deployment.

We consider 2 cluster node for this deployment. This is also eligible for 3 cluster node, just clone the compute node and change IP address configuration.

1. one control node containing all the management services for openstack (Horizon,Neutron,Glance,cinder,keystone)
2. one compute node containing nova-compute node.
3. Neutron using OVS back-end GRE for tunnels

8.1 Devstack installation on Controller node and Compute node

8.1.1 Controller Node:

Now generate a rsa Key for ssh to login without password by running below command.

```
$ ssh-keygen -t rsa
```

After executing above command just press “Enter” key to generate default ssh key.

To install Devstack on controller node, therefore execute below commands on respective node.

```
$ sudo apt-get install git -y
$ git clone -b stable/liberty https://git.openstack.org/openstack-dev/devstack
$ cd devstack
$ git clone -b https://github.com/shague/odl_tools
$ sudo vim local.conf
```

Now paste below code into local.conf file

```
[[local|localrc]]
GIT_BASE=${GIT_BASE:-http://git.openstack.org}

HOST_IP=192.168.254.32
HOST_NAME=controller
SERVICE_HOST_NAME=$HOST_NAME
SERVICE_HOST=$HOST_IP
Q_HOST=$SERVICE_HOST
# if you are running for first time make OFFLINE=False after successful stacking make it to “OFFLINE=True”
OFFLINE=False
# if you are running for first time make “RECLONE=Yes” after successful stacking make it to “RECLONE=no”
RECLONE=Yes
VERBOSE=True

DATABASE_PASSWORD=mysql
RABBIT_PASSWORD=rabbit
SERVICE_TOKEN=service
SERVICE_PASSWORD=admin
ADMIN_PASSWORD=admin

SCREEN_LOGDIR=$DEST/logs/screen
LOGFILE=$DEST/logs/stack.sh.log
LOG_COLOR=False

VNCSERVER_PROXYCLIENT_ADDRESS=${HOST_IP}
VNCSERVER_LISTEN=0.0.0.0

# Disables all the services
disable_all_services
# Enables core compute(Glance + keystone)
enable_service g-api g-reg key
# Enables core compute(nova + vnc)
enable_service n-api n-crt n-obj n-cpu n-cond n-sch n-novnc n-xvnc n-cauth
# Enables odl as the neutron backend rather than l2 agent
enable_service neutron q-dhcp q-meta q-svc odl-compute odl-neutron
enable_service tempest
```

```
# Enables dashboard
enable_service horizon
# Additional services
enable_service mysql rabbit

#Opendaylight ml2 vlan and gre tunnels
enable_plugin networking-odl https://github.com/openstack/networking-odl stable/liberty

# if using ODL outside devstack-control, replace ODL_MODE
ODL_MODE=externalodl
ODL_PORT=8080
ODL_MGR_IP=192.168.50.103
ML2_VLAN_RANGES=physnet1:2000:2999
NEUTRON_CREATE_INITIAL_NETWORKS=False

# Disable q-l3 and uncomment the lines below if ODL is being configured to perform l3fwd
# enable_service q-l3
#disable_service q-l3
Q_L3_ENABLED=True
#The next configuration is ODL l3
ODL_L3=True
# Use the following to automatically add eth2 to br-ex
PUBLIC_INTERFACE=eth2
# Add some time for odl to start before starting neutron
# Also set ODL_BOOT_WAIT_URL to empty to use ODL_BOOT_WAIT
ODL_BOOT_WAIT=123
BRANCH=stable/liberty
GLANCE_BRANCH=$BRANCH
HORIZON_BRANCH=$BRANCH
KEYSTONE_BRANCH=$BRANCH
NOVA_BRANCH=$BRANCH
NEUTRON_BRANCH=$BRANCH
SWIFT_BRANCH=$BRANCH
##CLIFF_BRANCH=$BRANCH
##TEMPEST_BRANCH=$BRANCH
CINDER_BRANCH=$BRANCH
HEAT_BRANCH=$BRANCH
TROVE_BRANCH=$BRANCH
CEILOMETER_BRANCH=$BRANCH

[[post-config|/etc/neutron/plugins/ml2/ml2_conf.ini]]
[agent]
minimize_polling=True

[[post-config|$NEUTRON_CONF]]
[DEFAULT]
service_plugins = networking_odl.l3.l3_odl.OpenDaylightL3RouterPlugin
```

Note: After saving local.conf file then execute below command in the devstack directory only when compute node is configured.

```
$ ./stack.sh
```

Note : if your stacking is unsuccessful then perform below command and then again stack with above command.

```
$ ./unstack.sh
```

Note: if you are unsuccessful even after unstacking then make it make a clean stacking.

```
$ ./clean.sh
```

Note: After executing above command again execute below command for clean stacking.

```
$ ./stack.sh
```

Note: Now edit few files in odl_tools folder.

```
$ cd devstack
$ sudo vim odl_tools/os_addvms.sh

change the hostnames to “controller” and “computenode1”
```

8.1.2 Compute Node :

To install Devstack on compute node, therefore execute below commands on respective node.

```
$ sudo apt-get install git -y
$ git clone -b stable/liberty https://git.openstack.org/openstack-dev/devstack
$ cd devstack
$ git clone -b https://github.com/shague/odl_tools
$ sudo vim local.conf
```

Now paste below code into local.conf file

```
[[local|localrc]]
GIT_BASE=${GIT_BASE:-http://git.openstack.org}

HOST_IP=192.168.254.33
HOST_NAME=computenode1
SERVICE_HOST_NAME=controller
SERVICE_HOST=192.168.254.32
Q_HOST=$SERVICE_HOST
# if you are running for first time make OFFLINE=False after successful stacking
```

```
make it to “OFFLINE=True”
OFFLINE=False
# if you are running for first time make “RECLONE=Yes” after successful stacking
make it to “RECLONE=no”
RECLONE=yes
VERBOSE=True

# Assigning Passwords
MYSQL_PASSWORD=mysql
RABBIT_PASSWORD=rabbit
SERVICE_TOKEN=service
SERVICE_PASSWORD=admin
ADMIN_PASSWORD=admin

SCREEN_LOGDIR=/opt/stack/logs/screen
LOGFILE=/opt/stack/logs/stack.sh.log
LOG_COLOR=False

VNCSERVER_PROXYCLIENT_ADDRESS=${HOST_IP}
VNCSERVER_LISTEN=0.0.0.0

# nova-network disabled
disable_all_services
#core compute(nova+vnc)
enable_service n-cpu n-novnc
#additional service
enable_service rabbit
# Below line enable odl as neutron backend rather than the l2 agent
enable_service neutron odl-compute

# opendaylight ml2 vlan and gre tunnels
enable_plugin networking-odl https://github.com/openstack/networking-odl
stable/liberty

# If using ODL outside devstack-control, replace ODL_MODE
ODL_MODE=compute
#ODL_NETVIRT_DEBUG_LOGS=True
ODL_PORT=8080
ODL_MGR_IP=192.168.50.103
ODL_PROVIDER_MAPPINGS=physnet1:eth1
#The next config is ODL L3
ODL_L3=True
# Use the following to automatically add eth2 to br-ex
PUBLIC_INTERFACE=eth2
ODL_BOOT_WAIT=123

# Use master for latest
BRANCH=stable/liberty
GLANCE_BRANCH=$BRANCH
HORIZON_BRANCH=$BRANCH
```

```
KEYSTONE_BRANCH=$BRANCH
NOVA_BRANCH=$BRANCH
NEUTRON_BRANCH=$BRANCH
SWIFT_BRANCH=$BRANCH
##CLIFF_BRANCH=$BRANCH
##TEMPEST_BRANCH=$BRANCH
CINDER_BRANCH=$BRANCH
HEAT_BRANCH=$BRANCH
TROVE_BRANCH=$BRANCH
CEILOMETER_BRANCH=$BRANCH

[[post-config|/etc/neutron/plugins/ml2/ml2_conf.ini]]
[agent]
minimize_polling=True
```

Note : After saving local.conf file then execute below command in the devstack directory only when controller node is finished stacking.

```
$ ./stack.sh
```

Note : if your stacking is unsuccessful then perform below command and then again stack with above command.

```
$ ./unstack.sh
```

Note: if you are unsuccessful even after unstacking then make it make a clean stacking.

```
$ ./clean.sh
```

Note: After executing above command again execute below command for clean stacking.

```
$ ./stack.sh
```

8.1.3 Testing

Procedure to run all nodes:

This procedure plays a vital role in this integration because we are running all nodes independently. Therefore, the main reason to follow this procedure is that sometimes devstack nodes and ODL controller node will not be in synchronization, devstack will end up with failures. Therefore, end of the report contains how to troubleshoot certain problems.

Procedure 1:

Note: As already router-node and ODL controller node is running and active we don't required to run again below respective instructions to run again if some failures happens.

Firstly, start the router-node and then ODL controller node and execute the bash file and wait until it finishes.

```
$ ./ovs.sh
```

```
$ cd distribution-karaf-0.3.3-Lithium-SR3
$ ./bin/karaf
```

Now, please wait for 5-7 minutes and then login with credentials(username and password : admin). Therefore, Once after login into ODL webpage and it will be empty now.

Procedure 2: Secondly, Start the devstack controller node and login with the credentials(login:stack and password:odl). After successful login perform below commands.

```
$ ./ovs.sh
```

```
$ cd devstack
$ ./stack.sh
```

After the successful stacking the below output of devstack node and opendaylight DLUX web page.

```
This is your host ip: 192.168.254.32
Horizon is now available at http://192.168.254.32/
Keystone is serving at http://192.168.254.32:5000/
The default users are: admin and demo
The password: admin
```

Figure 13: Devstack Controller node Output

Procedure 3: Thirdly, login into devstack computenode1 node and start stacking with below commands.

```
$ ./ovs.sh
```

```
$ cd devstack
$ ./stack.sh
```

After the successful stacking the below output of devstack node and opendaylight DLUX User Interface.

```
This is your host ip: 192.168.254.33
```

Figure 14: Devstack Compute Node Output

Procedure 4: Perform below commands on devstack controller node and compute node to check connectivity between the ODL and Devstack nodes.

```
$ sudo ovs-vsctl show
```

```
stack@controller:~/devstack$ sudo ovs-vsctl show
d26be849-c972-45e8-8425-2debf41050f1
    Manager "tcp:192.168.50.103:6640"
        is_connected: true
    Bridge br-ex
        Controller "tcp:192.168.50.103:6653"
            is_connected: true
        fail_mode: secure
        Port br-ex
            Interface br-ex
                type: internal
    Bridge br-int
        Controller "tcp:192.168.50.103:6653"
            is_connected: true
        fail_mode: secure
        Port br-int
            Interface br-int
                type: internal
    ovs_version: "2.3.1"
```

Figure 15: OVS output

```
$ sudo ovs-ofctl -O OpenFlow13 dump-flows br-int
```

```
********** sudo ovs-ofctl -O OpenFlow13 dump-flows br-int **********
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=929.171s, table=0, n_packets=0, n_bytes=0, priority=0 actions=goto_table:20
 cookie=0x0, duration=930.053s, table=0, n_packets=0, n_bytes=0, dl_type=0x88cc actions=CONTROLLER:65535
 cookie=0x0, duration=929.171s, table=20, n_packets=0, n_bytes=0, priority=0 actions=goto_table:30
 cookie=0x0, duration=929.160s, table=30, n_packets=0, n_bytes=0, priority=0 actions=goto_table:40
 cookie=0x0, duration=929.148s, table=40, n_packets=0, n_bytes=0, priority=0 actions=goto_table:50
 cookie=0x0, duration=929.139s, table=50, n_packets=0, n_bytes=0, priority=0 actions=goto_table:60
 cookie=0x0, duration=929.119s, table=60, n_packets=0, n_bytes=0, priority=0 actions=goto_table:70
 cookie=0x0, duration=929.103s, table=70, n_packets=0, n_bytes=0, priority=0 actions=goto_table:80
 cookie=0x0, duration=929.067s, table=80, n_packets=0, n_bytes=0, priority=0 actions=goto_table:90
 cookie=0x0, duration=929.061s, table=90, n_packets=0, n_bytes=0, priority=0 actions=goto_table:100
 cookie=0x0, duration=929.053s, table=100, n_packets=0, n_bytes=0, priority=0 actions=goto_table:110
 cookie=0x0, duration=929.012s, table=110, n_packets=0, n_bytes=0, priority=0 actions=drop
```

Figure 16: OVS OpenFlow 1.3 br-int Output

```
$ sudo ovs-ofctl -O OpenFlow13 dump-flows br-ex
```

```
************** sudo ovs-ofctl -O OpenFlow13 dump-flows br-ex *************
OFPST_FLOW reply (OF1.3) (xid=0x2):
 cookie=0x0, duration=931.068s, table=0, n_packets=0, n_bytes=0, priority=0 actions=NORMAL
 cookie=0x0, duration=931.066s, table=0, n_packets=0, n_bytes=0, dl_type=0x88cc actions=CONTROLLER:65535
```

Figure 17: OVS OpenFlow 1.3 br-ex output

```
$ sudo ovs-ofctl -O OpenFlow13 dump-flows br-ex | cut -d',' -f3-
```

```
***** sudo ovs-ofctl -O OpenFlow13 dump-flows br-ex | cut -d',' -f3- *****
OFPST_FLOW reply (OF1.3) (xid=0x2):
 table=0, n_packets=0, n_bytes=0, priority=0 actions=NORMAL
 table=0, n_packets=0, n_bytes=0, dl_type=0x88cc actions=CONTROLLER:65535
```

Figure 18: OVS OpenFlow 1.3 br-ex

Procedure 5: Fourthly, Now execute particular commands on devstack controller node to create router node and compute vm's and assigning floating IP address to ping from outside openstack.

```
$ cd devstack
$ source openrc admin admin
```

step 1: Creating a flavor with name m1.nano

```
$ odl_tools/os_addnano.sh
```

Note: After executing the command it throws error due to devstack will already created while stacking it. ERROR (Conflict): Flavor with name m1.nano already exists. (HTTP 409) (Request-ID: < ID >).

step 2: Adding Administration Key to compute node.

```
$ odl_tools/os_addadminkey.sh
```

```
stack@controller:~/devstack$ odl_tools/os_addadminkey.sh
+-----------+------+-------------------------------------------------+
| Name      | Type | Fingerprint                                     |
+-----------+------+-------------------------------------------------+
| admin_key | ssh  | 5a:c9:b9:ba:3a:e8:ea:48:49:44:0d:1b:a8:e8:ed:d9 |
+-----------+------+-------------------------------------------------+
```

Figure 19: Administration Key

step 3: Creating network,subnetwork,router

```
$ odl_tools/os_addextnetrtr.sh
```

```
stack@controller:~/devstack$ odl_tools/os_addextnetrtr.sh
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 3bddcd7c-f43c-4484-92e2-088407dbad4f |
| mtu                       | 0                                    |
| name                      | ext-net                              |
| provider:network_type     | flat                                 |
| provider:physical_network | public                               |
| provider:segmentation_id  |                                      |
| router:external           | True                                 |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | a3cb18e0e26c4870a14f0c797fe340bf     |
+---------------------------+--------------------------------------+
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "192.168.56.9", "end": "192.168.56.14"}    |
| cidr              | 192.168.56.0/24                                      |
| dns_nameservers   |                                                      |
| enable_dhcp       | False                                                |
| gateway_ip        | 192.168.56.1                                         |
| host_routes       |                                                      |
| id                | de5c112e-7d1e-4352-b8ee-40c586101075                 |
| ip_version        | 4                                                    |
| ipv6_address_mode |                                                      |
| ipv6_ra_mode      |                                                      |
| name              | ext-subnet                                           |
| network_id        | 3bddcd7c-f43c-4484-92e2-088407dbad4f                 |
| subnetpool_id     |                                                      |
| tenant_id         | a3cb18e0e26c4870a14f0c797fe340bf                     |
+-------------------+------------------------------------------------------+
Created a new network:
+---------------------------+--------------------------------------+
| Field                     | Value                                |
+---------------------------+--------------------------------------+
| admin_state_up            | True                                 |
| id                        | 11bd4159-ed38-4f2d-a1da-8bf5b64796cc |
| mtu                       | 0                                    |
| name                      | vx-net                               |
| provider:network_type     | vxlan                                |
| provider:physical_network |                                      |
| provider:segmentation_id  | 1500                                 |
| router:external           | False                                |
| shared                    | False                                |
| status                    | ACTIVE                               |
| subnets                   |                                      |
| tenant_id                 | a3cb18e0e26c4870a14f0c797fe340bf     |
+---------------------------+--------------------------------------+
Created a new subnet:
+-------------------+------------------------------------------------------+
| Field             | Value                                                |
+-------------------+------------------------------------------------------+
| allocation_pools  | {"start": "10.100.5.2", "end": "10.100.5.254"}       |
| cidr              | 10.100.5.0/24                                        |
| dns_nameservers   | 8.8.8.8                                              |
| enable_dhcp       | True                                                 |
| gateway_ip        | 10.100.5.1                                           |
| host_routes       |                                                      |
| id                | a8ee0d7b-d037-4854-bf79-d024c8a0979b                 |
| ip_version        | 4                                                    |
| ipv6_address_mode |                                                      |
| ipv6_ra_mode      |                                                      |
| name              | vx-subnet                                            |
| network_id        | 11bd4159-ed38-4f2d-a1da-8bf5b64796cc                 |
| subnetpool_id     |                                                      |
| tenant_id         | a3cb18e0e26c4870a14f0c797fe340bf                     |
+-------------------+------------------------------------------------------+
Created a new router:
+-------------------------+--------------------------------------+
| Field                   | Value                                |
+-------------------------+--------------------------------------+
| admin_state_up          | True                                 |
| distributed             | False                                |
| external_gateway_info   |                                      |
| id                      | d2a5884b-b419-410b-a51e-36d5f2b4082e |
| name                    | ext-rtr                              |
| routes                  |                                      |
| status                  | ACTIVE                               |
| tenant_id               | a3cb18e0e26c4870a14f0c797fe340bf     |
+-------------------------+--------------------------------------+
Set gateway for router ext-rtr
Added interface 889b5537-a9a9-4018-9ff2-a70ee37d7379 to router ext-rtr.
```

Figure 20: Creation of two networks(ext-net, vx-net) - network, subnet, router

step 4: Creating VM's

```
$ odl_tools/os_addvms.sh
```

```
stack@controller:~/devstack$ odl_tools/os_addvms.sh
+--------------------------------------+--------------------------------------------------------------------------------+
| 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-00000001                                                              |
| OS-EXT-STS:power_state               | 0                                                                              |
| OS-EXT-STS:task_state                | scheduling                                                                     |
| OS-EXT-STS:vm_state                  | building                                                                       |
| OS-SRV-USG:launched_at               | -                                                                              |
| OS-SRV-USG:terminated_at             | -                                                                              |
| accessIPv4                           |                                                                                |
| accessIPv6                           |                                                                                |
| adminPass                            | AktwePSz4kMv                                                                   |
| config_drive                         |                                                                                |
| created                              | 2015-12-22T23:22:18Z                                                           |
| flavor                               | m1.nano (42)                                                                   |
| hostId                               |                                                                                |
| id                                   | 2b20ab78-dd36-4dfc-adaf-af1e2dd0756f                                           |
| image                                | cirros-0.3.4-x86_64-uec (929257d1-d0b2-417d-821f-98c3e5e7cb88)                 |
| key_name                             | admin_key                                                                      |
| metadata                             | {}                                                                             |
| name                                 | vmvx1                                                                          |
| os-extended-volumes:volumes_attached | []                                                                             |
| progress                             | 0                                                                              |
| security_groups                      | default                                                                        |
| status                               | BUILD                                                                          |
| tenant_id                            | a3cb18e0e26c4870a14f0c797fe340bf                                               |
| updated                              | 2015-12-22T23:22:19Z                                                           |
| user_id                              | 47e8df9725b14187af6bd275bf026f22                                               |
+--------------------------------------+--------------------------------------------------------------------------------+
Server building... 100% complete
Finished
Server building... 100% complete
Finished
+--------------------------------------+--------------------------------------------------------------------------------+
| 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-00000002                                                              |
| OS-EXT-STS:power_state               | 0                                                                              |
| OS-EXT-STS:task_state                | scheduling                                                                     |
| OS-EXT-STS:vm_state                  | building                                                                       |
| OS-SRV-USG:launched_at               | -                                                                              |
| OS-SRV-USG:terminated_at             | -                                                                              |
| accessIPv4                           |                                                                                |
| accessIPv6                           |                                                                                |
| adminPass                            | JbZ5ae9hj7dy                                                                   |
| config_drive                         |                                                                                |
| created                              | 2015-12-22T23:22:48Z                                                           |
| flavor                               | m1.nano (42)                                                                   |
| hostId                               |                                                                                |
| id                                   | 554f004e-557c-4ced-b7e0-1767ed6624d6                                           |
| image                                | cirros-0.3.4-x86_64-uec (929257d1-d0b2-417d-821f-98c3e5e7cb88)                 |
| key_name                             | admin_key                                                                      |
| metadata                             | {}                                                                             |
| name                                 | vmvx2                                                                          |
| os-extended-volumes:volumes_attached | []                                                                             |
| progress                             | 0                                                                              |
| security_groups                      | default                                                                        |
| status                               | BUILD                                                                          |
| tenant_id                            | a3cb18e0e26c4870a14f0c797fe340bf                                               |
| updated                              | 2015-12-22T23:22:49Z                                                           |
| user_id                              | 47e8df9725b14187af6bd275bf026f22                                               |
+--------------------------------------+--------------------------------------------------------------------------------+
Server building... 100% complete
Finished
+--------------------------------------+--------------------------------------------------------------------------------+
| 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-00000003                                                              |
| OS-EXT-STS:power_state               | 0                                                                              |
| OS-EXT-STS:task_state                | scheduling                                                                     |
| OS-EXT-STS:vm_state                  | building                                                                       |
| OS-SRV-USG:launched_at               | -                                                                              |
| OS-SRV-USG:terminated_at             | -                                                                              |
| accessIPv4                           |                                                                                |
| accessIPv6                           |                                                                                |
| adminPass                            | 3aZNrvAGxQG5                                                                   |
| config_drive                         |                                                                                |
| created                              | 2015-12-22T23:23:17Z                                                           |
| flavor                               | m1.nano (42)                                                                   |
| hostId                               |                                                                                |
| id                                   | 15436e8e-ff98-42d8-83c8-dc37d9c542ce                                           |
| image                                | cirros-0.3.4-x86_64-uec (929257d1-d0b2-417d-821f-98c3e5e7cb88)                 |
| key_name                             | admin_key                                                                      |
| metadata                             | {}                                                                             |
| name                                 | vmvx3                                                                          |
| os-extended-volumes:volumes_attached | []                                                                             |
| progress                             | 0                                                                              |
| security_groups                      | default                                                                        |
| status                               | BUILD                                                                          |
| tenant_id                            | a3cb18e0e26c4870a14f0c797fe340bf                                               |
| updated                              | 2015-12-22T23:23:17Z                                                           |
| user_id                              | 47e8df9725b14187af6bd275bf026f22                                               |
+--------------------------------------+--------------------------------------------------------------------------------+
Server building... 100% complete
Finished
+-------+-----------------------------------------------------------------------------------------+
| Type  | Url                                                                                     |
+-------+-----------------------------------------------------------------------------------------+
| novnc | http://192.168.254.32:6080/vnc_auto.html?token=1fac045f-a5c3-487d-8e8b-459ccd8bafda |
+-------+-----------------------------------------------------------------------------------------+
+-------+-----------------------------------------------------------------------------------------+
| Type  | Url                                                                                     |
+-------+-----------------------------------------------------------------------------------------+
| novnc | http://192.168.254.32:6080/vnc_auto.html?token=f8d13119-d7c7-4860-b97a-b2bac2d3e250 |
+-------+-----------------------------------------------------------------------------------------+
+-------+-----------------------------------------------------------------------------------------+
| Type  | Url                                                                                     |
+-------+-----------------------------------------------------------------------------------------+
| novnc | http://192.168.254.32:6080/vnc_auto.html?token=50c91c85-d6d3-47c1-ab57-e2e4984339ff |
+-------+-----------------------------------------------------------------------------------------+
```

Figure 21: Creation of VM's

step 5: Adding Floating IP's to VM's

```
$ odl_tools/os_addfloatingip.sh
```

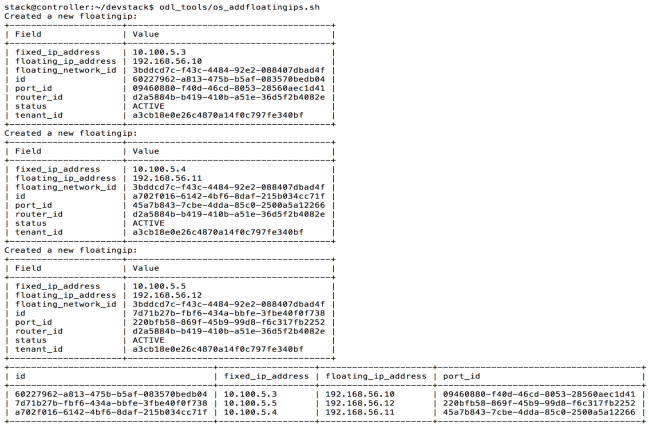


Figure 22: Adding Floating IP address to VM's

Step 6: SSH into openstack instances.

```
$ odl_tools/os_ssh.sh 10.100.5.3
```

```
$ ping 192.168.56.10
PING 192.168.56.10 (192.168.56.10): 56 data bytes
64 bytes from 192.168.56.10: seq=0 ttl=61 time=17.344 ms
64 bytes from 192.168.56.10: seq=1 ttl=61 time=9.530 ms
64 bytes from 192.168.56.10: seq=2 ttl=61 time=3.671 ms
^Z[5]+  Stopped                    ping 192.168.56.10
$ ping google.com
PING google.com (92.226.2.34): 56 data bytes
64 bytes from 92.226.2.34: seq=0 ttl=59 time=36.702 ms
64 bytes from 92.226.2.34: seq=1 ttl=59 time=31.715 ms
64 bytes from 92.226.2.34: seq=2 ttl=59 time=33.526 ms
^Z[6]+  Stopped                    ping google.com
$ ping yahoo.com
PING yahoo.com (98.138.253.109): 56 data bytes
64 bytes from 98.138.253.109: seq=0 ttl=59 time=172.170 ms
64 bytes from 98.138.253.109: seq=1 ttl=59 time=173.712 ms
64 bytes from 98.138.253.109: seq=2 ttl=59 time=169.615 ms
^Z[7]+  Stopped                    ping yahoo.com
```

Figure 23 : Ping Test Output

Now execute below commands to check again Openvswitch

```
# After creating Tenants Virtual Machines
$ sudo ovs-vsctl show
```

```
stack@controller:~/devstack$ sudo ovs-vsctl show
740d6db6-fe80-4b63-af30-7ab14bf294d6
    Manager "tcp:192.168.50.103:6640"
        is_connected: true
    Bridge br-ex
        Controller "tcp:192.168.50.103:6653"
            is_connected: true
        fail_mode: secure
        Port br-ex
            Interface br-ex
                type: internal
        Port "eth2"
            Interface "eth2"
        Port patch-int
            Interface patch-int
                type: patch
                options: {peer=patch-ext}
    Bridge br-int
        Controller "tcp:192.168.50.103:6653"
            is_connected: true
        fail_mode: secure
        Port "tap53a4dd44-97"
            Interface "tap53a4dd44-97"
        Port patch-ext
            Interface patch-ext
                type: patch
                options: {peer=patch-int}
        Port "tapb1a6744a-d0"
            Interface "tapb1a6744a-d0"
                type: internal
        Port br-int
            Interface br-int
                type: internal
        Port "vxlan-192.168.254.33"
            Interface "vxlan-192.168.254.33"
                type: vxlan
                options: {key=flow, local_ip="192.168.254.32", remote_ip="192.168.254.33"}
    ovs_version: "2.3.1"
```

Figure 24: OVS Output After Tenant's VMs

```
$ sudo ovs-vsctl list Interface | grep -E '^name|^ofport
|^mac_in_use|^external_id'
```

```
stack@controller:~$ sudo ovs-vsctl list Interface | grep -E '^name|^ofport |^mac_in_use|^external_id'
external_ids        : {}
mac_in_use          : "08:00:27:5d:dc:d4"
name                : "eth2"
ofport              : 1
external_ids        : {}
mac_in_use          : "08:00:27:5d:dc:d4"
name                : br-ex
ofport              : 65534
external_ids        : {}
mac_in_use          : "9a:2b:50:1e:29:e7"
name                : patch-int
ofport              : 2
external_ids        : {attached-mac="fa:16:3e:32:a1:56", iface-id="b1a6744a-d06c-4241-933b-2964c4aeaf60", iface-status=active}
mac_in_use          : []
name                : "tapb1a6744a-d0"
ofport              : 1
external_ids        : {}
mac_in_use          : "5a:f3:c5:5b:2d:f0"
name                : patch-ext
ofport              : 2
external_ids        : {}
mac_in_use          : "1e:ca:54:4c:77:32"
name                : "vxlan-192.168.254.33"
ofport              : 3
external_ids        : {}
mac_in_use          : "9a:f0:f1:a7:ea:4b"
name                : br-int
ofport              : 65534
external_ids        : {attached-mac="fa:16:3e:c1:fa:e6", iface-id="53a4dd44-975c-40cd-8da9-2f92d22c1306", iface-status=active, vm-id="6572b897-b281-4881-977e-7b9fd47d3533"}
mac_in_use          : "fe:16:3e:c1:fa:e6"
name                : "tap53a4dd44-97"
ofport              : 4
```

Figure 25: List of Ethernet Interfaces

9 Opendaylight and Openstack User Interface Phases

9.1 Opendaylight Dlux Interface:

Procedure 1: Opendaylight DLUX screen will be empty before stacking.

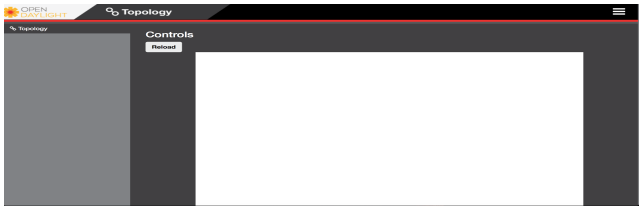


Figure 26: DLUX UI

Procedure 2: After stacking devstack controller node, if you reload DLUX, you should view two open vswitches are now connected to opendaylight.

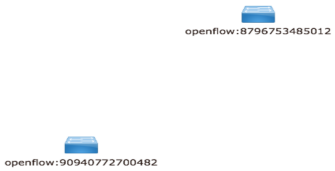


Figure 27: Openvswitch of Devstack Controller Node

Procedure 3: After stacking devstack compute node, if you reload DLUX, you should view four open vswitches are now connected to opendaylight and also compute node is managed by the controller node and is now connected to it.

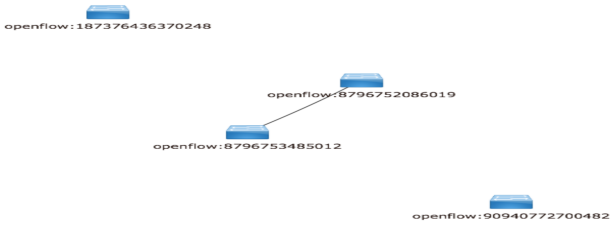


Figure 28: Openvswitch of Devstack Compute Node

Procedure 4: Here, no action will be taken place. Therefore, it's just for checking connectivity between ODL and devstack nodes.

Procedure 5: After executing command line interface commands, now we can see they are connected.

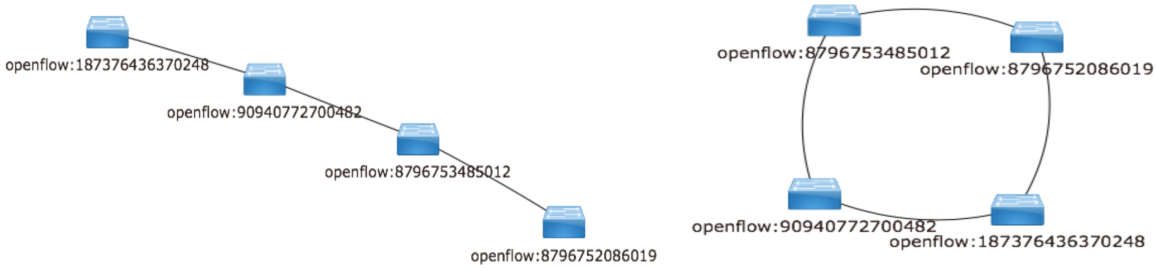


Figure 29: Openvswitch are connected (controller Node and Compute Node)

9.2 Openstack Dashboard:

Procedure 1: Not Required

Procedure 2: After stacking devstack controller node openstack “horizon” User Interface will have controller hypervisor under “System” under “Hypervisors” categories.

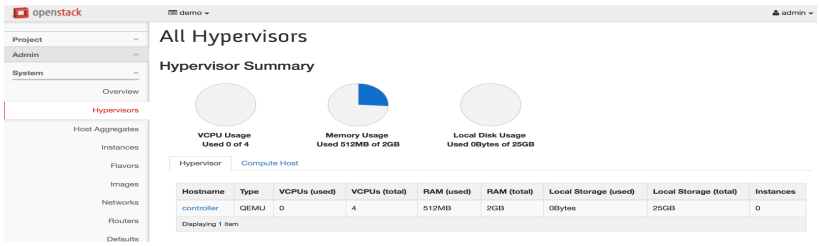


Figure 30: Openstack controller Node

Procedure 3: After stacking devstack compute node openstack horizon web page will have compute hypervisor under “System” under “Hypervisors” categories.

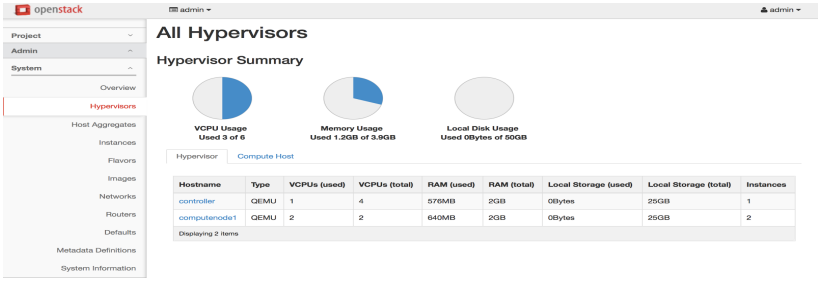


Figure 31: Openstack Compute Node

Procedure 4: Not Required

Procedure 5: Now , we have several steps.

step 1: Creating a flavor with name m1.nano

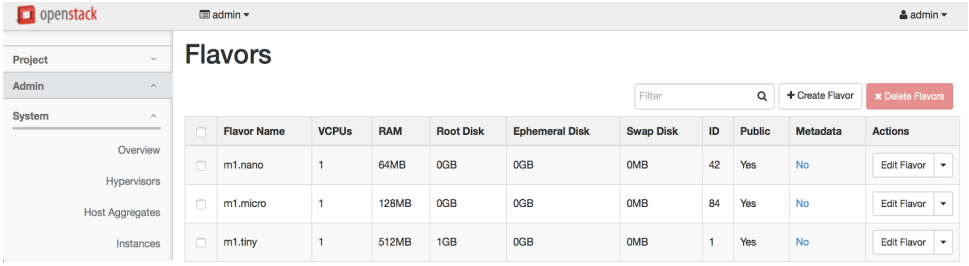


Figure 32: Flavors m1.nano

step 2: Adding Administration Key to compute node.

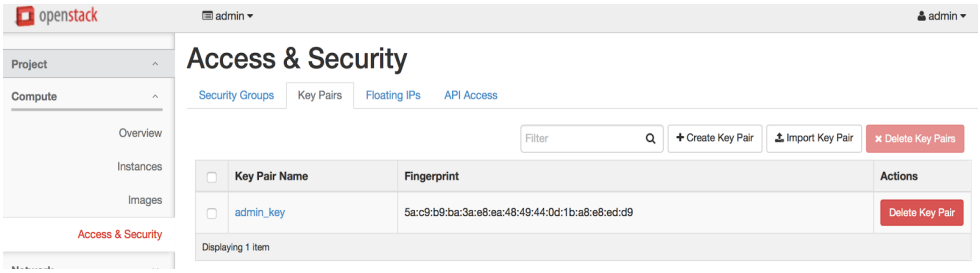
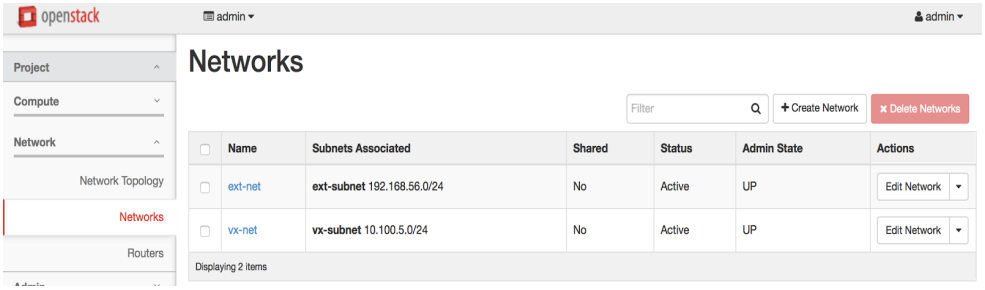


Figure 33 : Adminstration Key

step 3: Creating Networks, Subnet, Router.



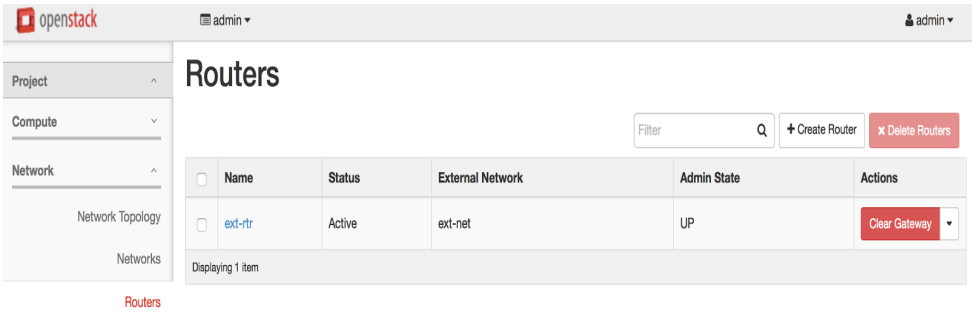


Figure 34: Creation of Networks and Routers

step 4 & 5 : Adding VM's and Floating IP's to instances to access from outside world.

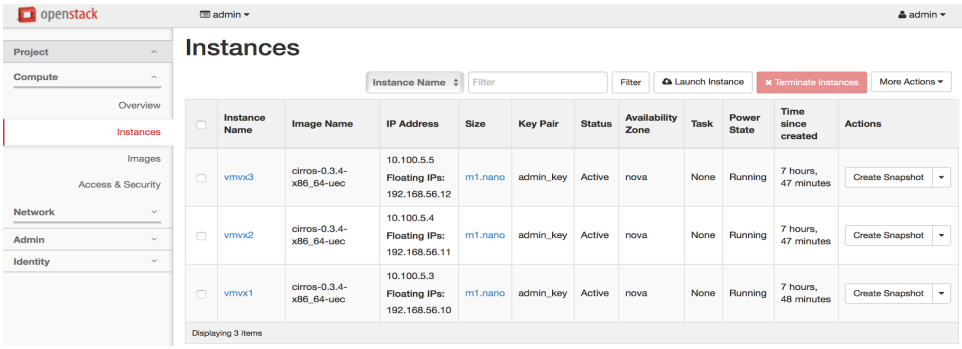


Figure 35: Added Floating IPs to VM's

Network Topology

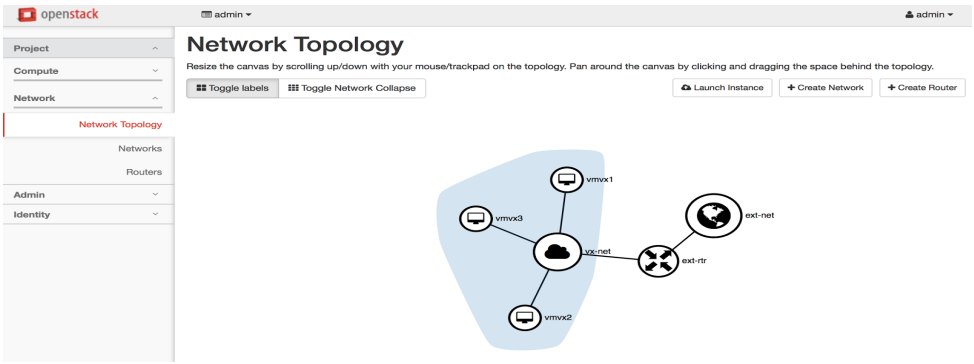


Figure 36: Network Topology

10 Troubleshooting:

Devstack Nodes:

1. Devstack Error while stacking the controller node

```
ources/__init__.py", line 849, in resolve
2015-12-06 01:08:55.026 |     raise DistributionNotFound(req, requirers)
2015-12-06 01:08:55.026 | pkg_resources.DistributionNotFound: The 'pbr!=0.7,<1.0
,>=0.6' distribution was not found and is required by oslo.i18n, python-keystone
client, oslo.utils, oslo.middleware, oslo.config, oslo.log, keystone, oslo.seria
lization, keystonemiddleware, oslo.db, oslo.messaging, oslo.concurrency
2015-12-06 01:08:55.043 | Error on exit
```

solution : Sometimes It required for updation so replace "Reclone=yes" in local.conf file if it is "Reclone=no ". or just simple run ./unstack.sh then ./clean.sh and again ./stack.sh.

2. Waiting for bridge br-int to be available error and Waiting for bridge br-ex to be available error

solution : execute below commands on devstack controller node and compute node

```
$ sudo service openvswitch-switch stop
$sudo rm -rf /var/log/openvswitch/*
$sudo rm -rf /etc/openvswitch/conf.db
$sudo service openvswitch-switch start
```

3. Cannot ssh into compute node instances.

solution: change MTU value to 1400

```
$ odl_tools/os_ssh.sh 10.100.5.3
$ ssh cirros@10.100.5.4
```

```
$ sudo ip link set eth0 mtu 1400
$ odl_tools/os_ssh.sh 10.100.5.4
```

4. Devstack Uninstall

solution:

```
$ cd devstack
$ ./unstack.sh
$./clean.sh
$ cd ..
$sudo rm -rf /opt/stack
$sudo rm -rf /etc/opt
#Delete some directories with configuration files to start from scratch.
$ sudo rm -rf /etc/nova
$ sudo rm -rf /etc/glance
$ sudo rm -rf /var/lib/nova
$ sudo rm -rf /etc/libvirt/
$ sudo rm -rf /var/lib/libvirt/
$ sudo rm -rf /etc/neutron
$ sudo rm -rf /var/lib/neutron
$ sudo rm -rf /etc/swift
$ sudo rm -rf /etc/cinder
$ sudo rm -rf /etc/keystone
$ sudo rm -rf /etc/rabbit
```

5. Devstack installation required permission to edit some files error

solution:

```
$ sudo chown 777 ~/devstack
$ sudo chown 777 /opt/stack
```

6. It will hang some time /opt/stack/nova or /opt/stack/neutron

solution:

```
$ ./unstack.sh
$ ./clean.sh
$ ./stack.sh
```

Opendaylight :

1. Feature cannot to uninstall properly so perform below commands.

solution:

```
$./bin/karaf clean
$logout
$./bin/karaf
or
```

```
$ sudo rm -rf <distribution folder>
$ tar xvzf <distribution folder.tar file>
$ cd <distribution folder>
$./bin/karaf
```

2. controller Logs

solution:

check controller log files under below path.

```
<distribution-folder>/data/log/karaf.log
```

It can also be set on karaf console:

```
log:display
```

All applications in OpenDaylight write in the log format.

```
log:set ERROR <feature name>
log:set TRACE <feature name>
bundle:list –s
bundle:list -s | grep -v Active
web:list
```

So log configuration is available in below path

```
<distribution-folder>/etc/org.ops4j.pax.logging.cfg
```

Linux commands:

```
ps -ef | grep java
netstat -pa | grep java
lsof -i:8080
lsof -i:8181
ps -ef | grep karaf
kill -9 <PID>
```

Packet capture:

```
wireshark/tshark
tcpdump
```

11 Challenges

1. The first encounter problem is laptop memory is of 4GB is not sufficient, due this reason opendaylight controller always fail to load features completely and devstack node needs required minimum 2GB.

2. The second encounter problem was fedora desktop version network connectivity issues and it takes more memory to run inside virtualbox.

3. The Third encounter problem, moved to Ubuntu desktop version still network connectivity issues it needs more memory. Therefore, opted ubuntu minimal server version.

4. The fourth encounter issue configuring Linux Network configuration settings and troubleshooting it .

5. The fifth encounter issue installing openstack through devstack as it was not stable i.e. kilo release and troubleshooting it.

6. The sixth encounter problem Opendaylight features installation number is high it won't load all features properly due to memory issue.

7. The seventh encounter problem understanding role of opendaylight with OpenStack.

8. The eight encounter problem was deploying L3 forwarding services to OpenStack nodes through Opendaylight.

9. The ninth encounter problem was messed up with iptables Firewall rules.

10. The tenth encounter problem unfortunately no proper and correct documentation for beginners.

12 Bibliography

[1] OpenStack

1.1 https://www.openstack.org/

1.2 http://docs.openstack.org/liberty/install-guide-ubuntu/

1.3 https://en.wikipedia.org/wiki/OpenStack

[2] Devstack

2.1 https://wiki.openstack.org/wiki/DevStack

2.2 http://docs.openstack.org/developer/devstack/index.html

2.3 https://github.com/openstack-dev/devstack

[3] OpenDaylight

3.1 https://www.opendaylight.org/

3.2 https://www.opendaylight.org/news/use-case/2015/07/use-case-network-services-cloud-data-center

3.3 https://en.wikipedia.org/wiki/OpenDaylight_Project

3.4 https://wiki.opendaylight.org/view/Install_On_Ubuntu_14.04

[4] OpenStack and OpenDaylight

4.1 https://wiki.opendaylight.org/view/OpenStack_and_OpenDaylight

4.2 http://blog.siliconloons.com/getting-started-with-opendaylight-and-openstack/

4.3 http://www.medianetlab.gr/publications/technical-reports/openstack-juno-opendaylight-helium-sr2-integration-over-ubuntu-14-04-lts-using-gre-tunnels/

4.4 https://wiki.opendaylight.org/view/OVSDB:Lithium_and_Openstack_on_CentOS7

[5] Open vSwitch

5.1 http://openvswitch.org/pipermail/announce/2014-December/000071.html