OpenStack Mitaka and

OpenDayLight Berylium Integration

Kasidit Chanchio, Vasinee Siripoon, Somkiat Kosolsombat

vasabilab

28 July 2016

In this document, we refer to a reference architecture depicted in Figure 1 below, where there are four ubuntu 14.04 machines running OpenStack Mitaka and one machine running OpenDayLight Beryllium. The gateway machine is a simulated gateway of the network where OpenStack and ODL hosts reside.

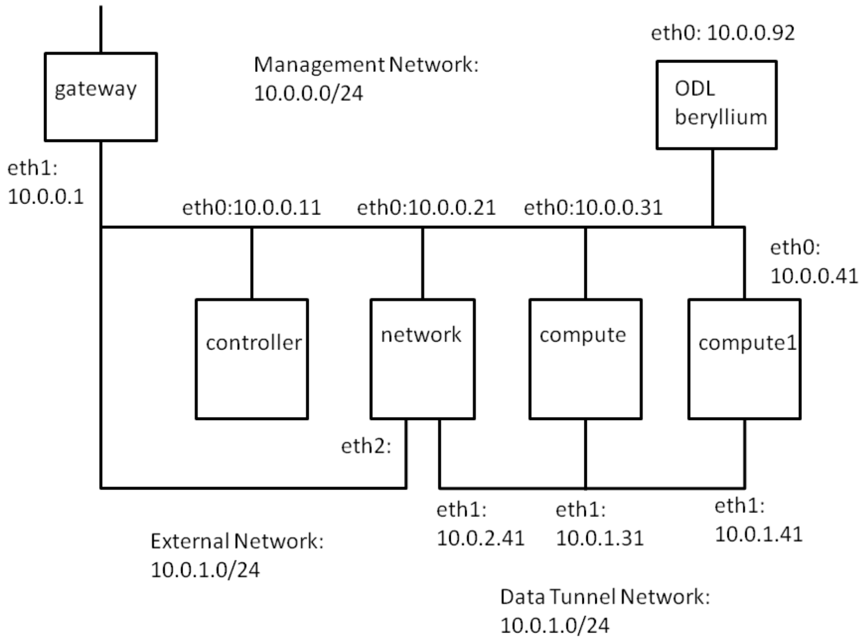

Figure 1.

We supposed the OpenStack installation with neutron is running. This document uses "gre" tunneling configuration for neutron's data tunnel network. The configuration for "vxlan" mechanisms will be given in the pink sideline.

For vxlan, it is recommended to change the MTU of every NIC attached to the data tunnel network (i.e. eth1: 10.0.1.21, eth1: 10.0.1.31, eth1: 10.0.1.41 in Figure 1) to 1600 bytes for performance.

1. ODL installation

On the ODL machine (IP 10.0.0.92), install java and download and install ODL beryllium using the following commands.

```
$ sudo apt-get install openjdk-7-jdk
$ wget https://nexus.opendaylight.org/content/groups/public/org/opendaylight/
integration/distribution-karaf/0.4.2-Beryllium-SR2/distribution-karaf-0.4.2-
Beryllium-SR2.tar.gz
$ tar xvfz distribution-karaf-0.4.2-Beryllium-SR2.tar.gz
$ cd distribution-karaf-0.4.2-Beryllium-SR2/
$ sudo ./bin/start
$ sudo ./bin/client –u karaf
```

In the ODL shell, install the following features:

```
opendaylight-user@root>feature:install odl-ovsdb-openstack
opendaylight-user@root>feature:install odl-dlux-core
opendaylight-user@root>feature:install odl-dlux-all
```

You can check what features were installed with the following commands.

```
opendaylight-user@root>feature:list | grep dlux
  < showing the list of features installed … >
opendaylight-user@root>feature:list | grep openstack
  < showing the list of features installed … >
opendaylight-user@root>
```

Check the installation from one of the remote machine in the OpenStacK pool, says the controller machine.

```
openstack@controller:~$ curl -u admin:admin
http://10.0.0.92:8080/controller/nb/v2/neutron/networks
{
   "networks" : [ ]
}openstack@controller:~$
```

2. Erase all VMs and network stuffs from OpenStack

Terminate all the VMs and network related stuffs such as network, subnet, routers, Floating IPs of every user from the OpenStack installation. You can do this via Horizon dashboard or command line interfaces.

Then, stop neutron with

```
openstack@controller:~$ sudo service neutron-server stop
```

There are two way to continue OpenStack and OpenDayLight integration. First, you can script provede in the openstack-mitaka-installer-0.3.3.tar file or configure the integration manually.

If you want to use the script, please continue on Step 11. In case you want to do it manually, you can follow the next step.

3. Remove neutron plugin on every network and compute node

Purge the neutron plugin agent and delete existing openvswitch configurations. Restart the openvswitch agent anew.

```
openstack@network:~$ sudo apt-get purge neutron-openvswitch-agent
openvswitch-switch stop/waiting
openstack@network:~$ sudo rm -rf /var/log/openvswitch/
openstack@network:~$ sudo rm -rf /etc/openvswitch/conf.db
openstack@network:~$ sudo mkdir /var/log/openvswitch/
openstack@network:~$ sudo service openvswitch-switch start
openvswitch-switch start/running
openstack@network:~$ sudo ovs-vsctl show
2c63096f-74f3-46eb-9904-00305ef84106
    ovs_version: "2.3.1"
openstack@network:~$
```

Do the same on every network node and compute node. Next, identify the IP address of the NIC on each machine that connect to the data tunnel network and input the openvswitch's configuration.

```
openstack@network:~$ grep local_ip /etc/neutron/plugins/ml2/ml2_conf.ini
local_ip = 10.0.1.21
openstack@network:~$
openstack@network:~$ openstack@network:~$ sudo ovs-vsctl set Open_vSwitch
2c63096f-74f3-46eb-9904-00305ef84106  other_config={'local_ip'='10.0.1.21'}
```

Instruct openvswitch to make a connection to the OpenDayLight server with the command below.

```
openstack@network:~$ sudo ovs-vsctl set-manager tcp:10.0.0.92:6640
```

Again, do the same on every network and compute node. If your controller node is also a network node, you may have to do this on the controller as well.

4. Configure external network on the network node

Do the following on the network node.

```
openstack@network:~$ sudo ovs-vsctl add-br br-ex
openstack@network:~$ sudo ovs-vsctl add-port br-ex eth3
openstack@network:~$ sudo ovs-vsctl show
2c63096f-74f3-46eb-9904-00305ef84106
```

```
    Manager "tcp:10.0.0.92:6640"
        is_connected: true
    Bridge br-ex
        Port br-ex
            Interface br-ex
                type: internal
        Port "eth3"
            Interface "eth3"
    Bridge br-int
        Controller "tcp:10.0.0.92:6653"
            is_connected: true
        fail_mode: secure
        Port br-int
            Interface br-int
                type: internal
    ovs_version: "2.3.1"
openstack@network:~$
```

5. Config ml2_conf.ini on controller, network and compute node

Next, modify the `/etc/neutron/plugins/ml2/ml2_conf.ini` file on

- controller,
- network,
- compute,
- compute1

On the controller node:

```
openstack@network:~$ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = opendaylight

[ml2_odl]
password = admin
username = admin
url = http://10.0.0.92:8080/controller/nb/v2/neutron
[ml2_type_gre]
tunnel_id_ranges = 1:1000
```

For vxlan, the ml2_conf.ini should contain:

```
openstack@network:~$ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,vxlan
tenant_network_types = vxlan
mechanism_drivers = opendaylight

[ml2_odl]
password = admin
username = admin
url = http://10.0.0.92:8080/controller/nb/v2/neutron
[ml2_type_vxlan]
vni_ranges = 1:1000
```

On the network nodes do the following. Note that the [agent] section is different from the sane file on the controller node above, assuming the controller node does not have compute node capability and is not connecting to the data tunnel network.

```
openstack@network:~$ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = opendaylight

[ml2_odl]
password = admin
username = admin
url = http://10.0.0.92:8080/controller/nb/v2/neutron
[ml2_type_gre]
tunnel_id_ranges = 1:1000
[ovs]
local_ip = 10.0.1.21
bridge_mappings = external:br-ex
[agent]
tunnel_types = gre
```

For vxlan, the ml2_conf.ini file on the network node is:

```
openstack@network:~$ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,vxlan
tenant_network_types = vxlan
mechanism_drivers = opendaylight
[ml2_odl]
password = admin
username = admin
url = http://10.0.0.92:8080/controller/nb/v2/neutron
[ml2_type_vxlan]
vni_ranges = 1:1000
[ovs]
local_ip = 10.0.1.21
bridge_mappings = external:br-ex
[agent]
tunnel_types = vxlan
```

On the compute nodes modify the ml2_conf.ini as follows. The example file below is for the compute node that has data tunnel network IP as 10.0.1.31. You have to change the "local_ip" parameter for other compute nodes accordingly. (Do the comute1's configuration yourself.)

```
openstack@compute:~$ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,gre
tenant_network_types = gre
mechanism_drivers = opendaylight
[ml2_odl]
password = admin
username = admin
url = http://10.0.0.92:8080/controller/nb/v2/neutron
[ml2_type_gre]
tunnel_id_ranges = 1:1000
[ovs]
local_ip = 10.0.1.31
[agent]
tunnel_types = gre
```

For vxlan, the ml2_conf.ini file on the network node is:

```
openstack@network:~$ sudo vi /etc/neutron/plugins/ml2/ml2_conf.ini
[ml2]
type_drivers = flat,vxlan
tenant_network_types = vxlan
mechanism_drivers = opendaylight
[ml2_odl]
password = admin
username = admin
url = http://10.0.0.92:8080/controller/nb/v2/neutron

[ml2_type_vxlan]
vni_ranges = 1:1000
[ovs]
local_ip = 10.0.1.31
[agent]
tunnel_types = vxlan
```

6. Config l3_agent.ini on network node

Next, indicate "external_network_bridge = br_ex" in the l3_agent.ini file of the network node. Note that from our experience, **you must complete the step 5 above before doing this step.** This requirement, however, may not be required in your situations.

```
openstack@network:~$ sudo vi /etc/neutron/l3_agent.ini
…
interface_driver = neutron.agent.linux.interface.OVSInterfaceDriver
external_network_bridge = br_ex
router_delete_namespaces = True
verbose = True
```

7. Reset neutron database on controller node

Log in as a user and use the following commands on the controller node. These commands reset neutron database and configure new parameters to it.

```
openstack@controller:~$ source ./admin_openrc.sh
openstack@controller:~$ mysql -u root -pmysqlpassword
MariaDB [(none)]> DROP DATABASE neutron;
Query OK, 157 rows affected (1.76 sec)

MariaDB [(none)]> CREATE DATABASE neutron;
Query OK, 1 row affected (0.00 sec)
```

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'localhost'
IDENTIFIED BY 'b6d473ff35f93e98e191';
Query OK, 0 rows affected (0.05 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON neutron.* TO 'neutron'@'%'
IDENTIFIED BY 'b6d473ff35f93e98e191';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> exit
Bye
openstack@controller:$
openstack@controller:$ sudo 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
```

8. REBOOT controller, network nodes and compute nodes.

You may restart the neutron server on the controller node now.

```
openstack@controller:$ sudo service neutron-server restart
neutron-server start/running, process 7669
openstack@controller:$
```

From our experience, you may want to reboot the controller, network, and compute nodes here. Make sure the controller start first following by network and compute nodes consequently.

9. Install the networking_odl python module.

Next, do the following on the controller.

```
openstack@controller:$ sudo apt-get install git
openstack@controller:$ git clone https://github.com/openstack/networking-odl
-b stable/mitaka
openstack@controller:$ sudo python setup.py install
openstack@controller:$ sudo service neutron-server restart
neutron-server stop/waiting
neutron-server start/running, process 12502
```

10. Verify installation

Next, create router, network and subnet on the controller's CLI or use web UI in horizon. We will create an external network and a private network in this example.

```
openstack@controller:$ . ./admin_openrc.sh
```

In our setup, the file contain the followings:

```
openstack@controller:$ neutron net-create ext-net --router:external --
provider:physical_network external --provider:network_type flat

openstack@controller:$ neutron subnet-create ext-net 10.0.0.0/24 --name ext-
subnet --allocation-pool start=10.0.0.100,end=10.0.0.200 --disable-dhcp --
gateway 10.0.0.1

openstack@controller:$ neutron net-create demo-net

openstack@controller:$ neutron subnet-create demo-net 192.168.1.0/24 --name
demo-subnet --gateway 192.168.1.1

openstack@controller:$ neutron router-create demo-router

openstack@controller:$ neutron router-interface-add demo-router demo-subnet

openstack@controller:$ neutron router-gateway-set demo-router ext-net

openstack@controller:$ ping 10.0.0.100
```

11. Installation using installer scripts

Assuming that you have installed OpenStack Mitaka with classic open vswitch network, you can do the following.

- Delete all virtual networks and routers in OpenStack.
- Login to the openstack account on the controller and make sure the time is correct

```
$ date
```

- Set the ODL_IP parameter in the install_paramrc.sh file and run the exe-config-installer.sh script

```
$ cd openstack-mitaka-installer-0.3.3
$ vi install-paramrc.sh
…
export ODL_IP=10.0.0.92
…
$ ./exe-config-installer.sh
```

- run the following scripts in the controller node.

```
$ cd OPSIinstaller/installer
$ ./OSODL-force-set-controller-time.sh
$ ./ OSODL-force-redo-set-openstack-nodes.sh
$ ./ OSODL-installer-00-stop-neutron.sh
$ ./ OSODL-installer-01-purge-neutron-agent.sh
$ ./ OSODL-installer-02-set-ovs-manager.sh
$ ./ OSODL-installer-03-copy-ml2.sh
$ ./ OSODL-installer-04-set-neutron.sh
```

Shutdown controller, network, compute and compute1 node, and restart the cotroller, network, compute and compute1 in order.

```
$ ./ OSODL-installer-05-pip-init-network.sh
```

Test your installation by running the cirros VM in openstack.

Cheers.