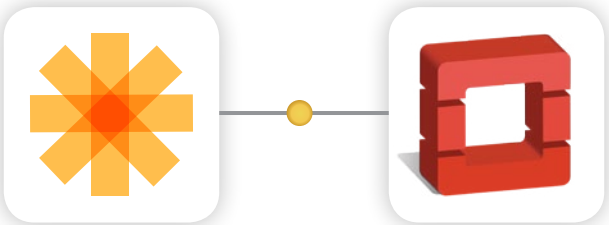
Open Daylight SDN Integration with open stack



OpenDaylight is a collaborative open source project hosted by the Linux Foundation. The goal of this project is to accelerate the adoption of software defined networking (SDN), creating a solid foundation for network functions virtualisation (NFV). OpenStack is a FOSS software platform for cloud computing, generally deployed as Infrastructure-as-a-Service (IaaS). This article gives a step-by-step tutorial on integrating OpenDaylight and OpenStack.

The adoption of open source technologies is on the rise, which is evident from the popularity of projects like OpenStack and OpenDaylight. While each project tries to address specific problems, a combination of OpenStack and OpenDaylight projects might be something to look out for.

OpenDaylight started with the release of Hydrogen and is now in its third release cycle, named Lithium. There are more than 60 projects available or being incubated with OpenDaylight Lithium that could satisfy all the networking requirements demanded by SDN. There are also some dedicated projects for OpenStack that are designed and integrated with OpenDaylight.

In this article, I will walk you through the steps in deploying OpenDaylight and integrating it with OpenStack to address use cases specific to networking infrastructure.

1. Assumption

Before getting started, I assume the reader understands OpenStack (kilo) and the set-up is already done.

2. Installing OpenDaylight

OpenDaylight Controller runs in a JVM. So install OpenJDK7 using the command below:

```
$ apt-get install openjdk-7-jdk
```

Download the latest OpenDaylight-Lithium package from the official repo, as follows:

```
$ 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
```

Uncompress it as the root user, and start OpenDaylight using the following command:

```
$ tar xvfz distribution-karaf-0.3.3-Lithium-SR3.tar.gz
$ cd distribution-karaf-0.3.3-Lithium-SR3/
$ ./bin/start  # Start OpenDaylight as a server process and
Wait for some time before running the next command below.
```

Connect to the Karaf shell, as follows:

```
$ ./bin/client  # connecting to the OpenDaylight with the
client
```

Now, you should be in OpenDaylight's console. Install all the required features:

```
opendaylight-user@root> feature:install odl-base-all odl-aaa-
authn odl-restconf odl-nsf-all
odl-adsal-northbound odl-mdsal-apidocs odl-ovsdb-openstack
odl-ovsdb-northbound odldlux-core
#Feature installation may take some time to install.
```

Once the installation is complete, you can check whether everything is working fine by using the *curl* call shown below:

```
$ curl -u admin:admin http://<OPENDAYLIGHT SERVER's IP>:8080/
controller/nb/v2/neutron/networks
```

The response should be an empty network list if OpenDaylight is working properly.

Also, you should be able to log in to the *dlux* interface on *http://<OPENDAYLIGHT SERVER's IP>:8181/index.html* - the default username and password is 'admin/admin' (see Figure 2).

Additionally, you can find the log details at the location below:

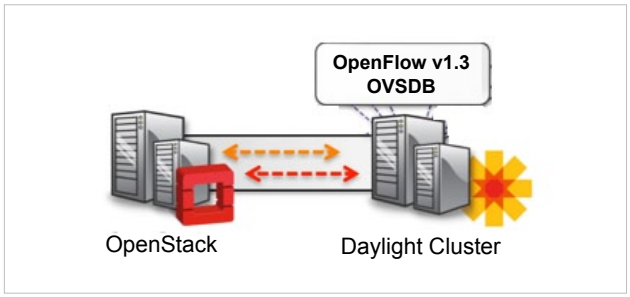


Figure 1: OpenDaylight-OpenStack

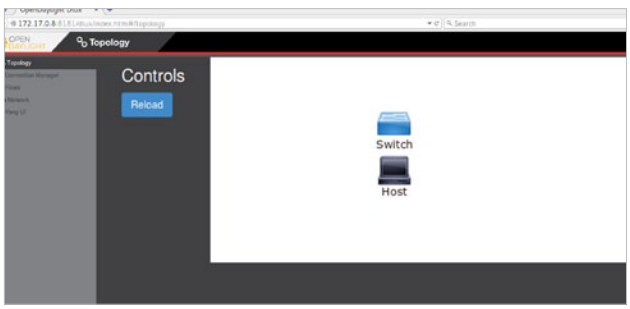


Figure 2: *Dlux* login screenshot

```
$ tail -f data/log/karaf.log
$ tail -f logs/web_access_log_2015-12.txt
```

Now, you have a working OpenDaylight-Lithium set-up; so, let's get into the integration part.

3. Erase all VMs, networks, routers and ports in the controller node

Since you already have a working OpenStack set-up, you may test for VM provisioning, but before integrating OpenStack with OpenDaylight, you must clean up all the unwanted data from the OpenStack database. When using OpenDaylight as the Neutron back-end, ODL expects to be the only source for Open vSwitch configuration. Because of this, it is necessary to remove existing OpenStack and Open vSwitch (OVS) configurations to give OpenDaylight a clear field.

The following steps will guide you through the cleaning process:

```
• Delete the instances
$ nova list
$ nova delete <instance names>

• Remove links from subnets to routers
$ neutron subnet-list
$ neutron router-list
$ neutron router-port-list <router name>
$ neutron router-interface-delete <router name> <subnet ID or
name>

• Delete subnets, nets, routers
$ neutron subnet-delete <subnet name>
$ neutron net-list
$ neutron net-delete <net name>
$ neutron router-delete <router name>

• Check that all ports have been cleared - at this point, this
should be an empty list
$ neutron port-list
```

While Neutron is managing the OVS instances on compute and control nodes, OpenDaylight and Neutron can be in conflict. To prevent issues, turn off the Neutron server on the network controller and Neutron's Open vSwitch agents on all hosts.

```
$ service neutron-server stop
```

4. Configuring Open vSwitches in network and compute nodes

The Neutron plugin in every node must be removed because only OpenDaylight will be controlling the Open vSwitches. So on each host, we will clear the pre-existing Open vSwitch configuration and set OpenDaylight to manage the Open vSwitch.

```
$ apt-get purge neutron-plugin-openvswitch-agent
$ service openvswitch-switch stop
$ rm -rf /var/log/openvswitch/*
$ rm -rf /etc/openvswitch/conf.db     #Clear openvswitch
database and start it again.
$ service openvswitch-switch start
$ ovs-vsctl show   #This command must return the empty set
except OpenVswitch ID and
it's Version.
```

5. Connecting Open vSwitch with OpenDaylight

Use the following commands to make OpenDaylight manage Open vSwitch:

```
$ ovs-vsctl set Open_vSwitch <OPENVSWITCH ID> other_
config={'local_ip'='<TUNNEL INTERFACE IP>'}
$ ovs-vsctl set-manager tcp:<OPENDAYLIGHT MANAGEMENT IP>:6640
```

You can get the Open vSwitch ID from the command *ovs-vsctl show* and the tunnel IP from the OpenStack *ml2_conf.ini* file.

Execute the above command in all the nodes (network and compute nodes) that have Open vSwitch installed.

Create the bridge *br-ex*, which is needed for the external network for OpenStack in the network (Neutron) node.

```
$ ovs-vsctl add-br br-ex
$ ovs-vsctl add-port br-ex <INTERFACE NAME OF EXTERNAL NETWORK>
#commonly eth0 or p2p1
$ ovs-vsctl show
```

The above command will show that you are connected to the

OpenDaylight server, and OpenDaylight will automatically create a *br-int* bridge.

```
[root@vinoth ~]# ovs-vsctl show
9e3b34cb-fefc-4br4-828s-084b3e55rtfd
Manager "tcp:192.XXX.X.XXX:XXXX"
Is_connected: true
Bridge br-int
Controller "tcp:192.XXX.X.XXX:XXXX"
fail_mode: secure
Port br-int
Interface br-int
ovs_version: "2.1.3"
```

6. Configuring *ml2_conf.ini* for the OpenDaylight driver

Edit *vi /etc/neutron/plugins/ml2/ml2_conf.ini* in all the required nodes and modify the following configuration. Leave the other configurations as they are.

```
[ml2]
type_drivers = flat,vxlan
tenant_network_types = vxlan
mechanism_drivers = opendaylight

[ml2_odl]
password = admin
username = admin
url = http://<OPENDAYLIGHT SERVER's IP>:8080/controller/nb/
v2/neutron
```

7. Configuring the Neutron database

Reset the Neutron database, as follows:

```
$ mysql -uroot –p
$ drop database neutron;
$ create database neutron;
$ grant all privileges on neutron.* to 'neutron'@'localhost'
identified by '<YOUR NEUTRON
PASSWORD>';
$ grant all privileges on neutron.* to 'neutron'@'%'
identified by '<YOUR NEUTRON
PASSWORD>';
$ exit
$ 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
```

Restart the Neutron server, using the following command:

```
$ service neutron-server start
```

8. Installing the *networking_odl* Python module

Important: At this stage you will get a message stating "Neutron service failed to start." Don't worry; it's a temporary issue. Since you have enabled OpenDaylight as a *mechanism_driver* but not yet installed the Python module for it, you need to install the *networking_odl* Python module, as follows:

```
$ apt-get install python-pip
$ pip install networking_odl
```

Now, restart the Neutron server and check its status. It should be running without errors.

9. Verifying the integration

We are almost done with integrating OpenStack with OpenDaylight. Now, it's time to verify it.

Create initial networks in OpenStack and check whether they are reflected in OpenDaylight.

You could create the network using the OpenStack horizon dashboard also. Shown below are the commands to create the same in CLI:

```
$ neutron router-create router1
$ neutron net-create private
$ neutron subnet-create private --name=private_subnet
172.0.0.0/24
$ neutron router-interface-add router1 private_subnet
$ nova boot --flavor <flavor> --image <image id> --nic net-
id=<network id> vinoth-vm1
```

Once the network creation is done, you will be able to see that the same network has been created in OpenDaylight by using the following command:

```
$ curl -u admin:admin http://${OPENDAYLIGHT's SERVER}:8080/
controller/nb/v2/neutron/networks
```

The above command will return the network information which is created in OpenDaylight through OpenStack. If the integration has been done correctly, you should now be able to ping VM. END

References

[1] *http://www.hellovinoth.com/*
[2] *http://www.cloudenablers.com/blog/*
[3] *https://www.opendaylight.org/*
[4] *http://docs.openstack.org/*

By: Vinoth Kumar Selvaraj

The author is a DevOps engineer at Cloudenablers Pvt Ltd, a cloud technology start-up based at Chennai, India. He holds a bachelor's degree in computer science and engineering, and is also a freelance reviewer with PACKTPUB Publishers, for books related to OpenStack. He blogs at *http://www.hellovinoth.com*