International Journal of Science and Research (IJSR)
ISSN (Online): 2319-7064
Index Copernicus Value (2013): 6.14 | Impact Factor (2013): 4.438

Open Daylight Integration with Openstack

Basivireddy D

M. Tech Cloud Computing, Department of Information Technology,
SRM University, SRM Nagar, Kattankulathur, Tamil Nadu, India, 603203

Abstract: This paper explain about some advantages of Open Daylight on Openstack and integration of both open source projects.

Keywords: OpenDaylight, Openstack, Software Defined Networking, Neutron, ML2 Plug-in**.**

1. Introduction

OpenDaylight is a Java-based open source SDN controller. Software-defined Network (SDN) is a recent architectural framework for networking, which decouples the network control plane from the data plane at physical topology and provides a programmatic interface to network control. In this paper we address advantages of OpenDaylight on Openstack like Policy based traffic steering on Openstack and Integration of both Open source projects.

I begin by giving an overview of Openstack in Section 2. Next describe about OpenDaylight in Section 3. After that I discuss how to integrate OpenDaylight with Openstack in Section 4. I discuss about advantages of integration in Section 5 and describe our conclusions and future work in Section 6 .

2. Openstack

Openstack is a cloud computing platform and open source system. Openstack provides an Infrastructure as a Service (IaaS) and constitutes of resources such as compute, storage and network resources. Openstack is a manager of multiple hypervisors. It is a collection of tools for managing and orchestrating cloud resources. Openstack keeps its services as decoupled as possible. This is designed to provide massive scalability.

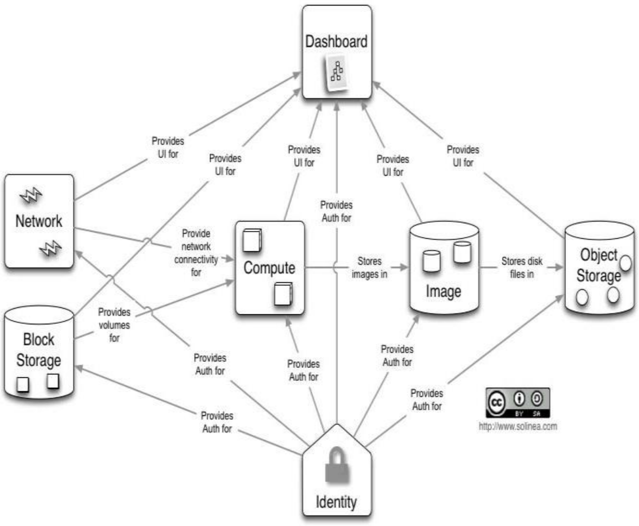


Figure 1: Loosely coupled architecture of Openstack

2.1 Openstack Services

- Dashboard also knows as Horizon provides a web based user interface access to compute, storage and network resources required by virtual machines. Users can provision and relinquish resources from horizon.
- Compute also knows as Nova avails computing services for creating deleting and managing virtual machines.
- Network also knows as Neutron provides virtual networking services to Nova. It allows users to create tenant specific networks.
- Image also knows as Glance can store the actual virtual disk files in the Object Store also know as swift
- Identity management service also knows as Keystone authenticates and authorizes all components of the Openstack.

2.2 Openstack Neutron

Neutron is an Openstack project to provide "networking as a service" between interface devices (e.g., vNICs) managed by other Openstack services. It enable innovation plugins (open and closed source) that introduce advanced network capabilities like use L2-in-L3 tunneling to avoid VLAN limits, provide end-to-end QoS guarantees, used monitoring protocols like NetFlow. Let anyone build advanced network services (open and closed source) that plug into Openstack tenant networks like LB-aaS, VPN-aaS, firewall-aaS, IDS-aaS, data-center-interconnect-aaS.

The Modular Layer 2 (ml2) plugin is a framework allowing Openstack Networking to simultaneously utilize the variety of layer 2 networking technologies found in complex real-world data centers. It currently works with the existing open vs witch, linux bridge, and hyperv L2 agents, and is intended to replace and deprecate the monolithic plugins associated with those L2 agents. The ml2 framework is also intended to greatly simplify adding support for new L2 networking technologies, requiring much less initial and ongoing effort than would be required to add a new monolithic core plugin. A modular agent may be developed as a follow-on effort.

3. OpenDaylight

The OpenDaylight Project is a collaborative open source project hosted by The Linux Foundation. The goal of the project is to accelerate the adoption of software-defined networking (SDN) and create a solid foundation for Network

Functions Virtualization.

Software-defined Network (SDN) is a recent architectural framework for networking, which decouples the network control plane from the data plane at physical topology and provides a programmatic interface to network control. SDN has the potential in simplifying the network management, improving the efficiency of network utilization, and enabling network innovations. It is an approach to computer networking that allows network administrators to manage network services through abstraction of lower level functionality. This is done by decoupling the system that makes decisions about where traffic is sent (the control plane) from the underlying systems that forward traffic to the selected destination (the data plane). The inventors and vendors of these systems claim that this simplifies networking.[1] SDN requires some method for the control plane to communicate with the data plane. One such mechanism, OpenFlow, is often misunderstood to be equivalent to SDN, but other mechanisms could also fit into the concept.

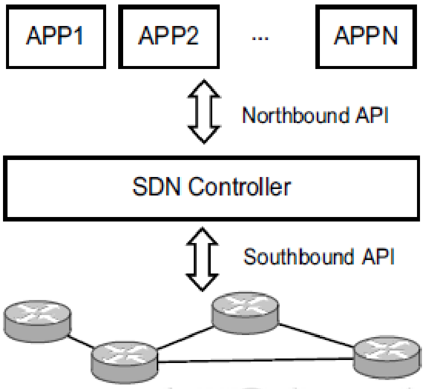
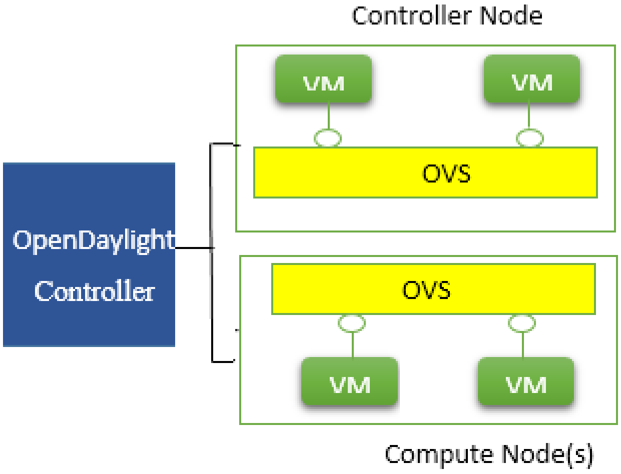


Figure 2: High Level SDN Architecture

4. Integrate OpenDaylight with Openstack

Openstack keeps its services as decoupled as possible. This is designed to provide massive scalability. As show in figure install Openstack compute+controller services on 172.21.207.134 and one more compute node with 172.21.207.135 and OpenDaylight on 172.21.207.136. These configurations not mandatory. It’s as reference.

Architecture:



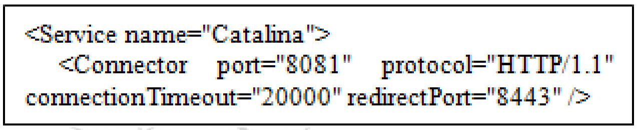
OpenDaylight is an open platform for network programmability to enable Software-Defined Networking (SDN). OpenDaylight has driver for Neutron ML2 (Modular Layer 2) plugin to enable communication between Neutron and OpenDaylight. On the SDN controller side, OpenDaylight has northbound APIs to interact with Neutron and use OVSDB (Open vSwitch Database Management Protocol) for southbound configuration of vSwitches on compute nodes. Thus OpenDaylight can manage network connectivity and initiate GRE or VXLAN tunnels for compute nodes.

Step-by-step detail for neutron integration with OpenDaylight.

1) Run OpenDaylight Controller
2) Configure ovsdb to connect to OpenDaylight and provide network information
3) Start Openstack
4) Enable ML2 plugin with OpenDaylight controller
5) Create network and launch VM instances

4.1 Run OpenDaylight Controller

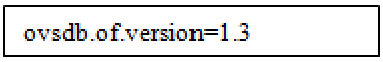
Get OpenDaylight virtualization edition from OpenDaylight Release Download. Follow the Installation Guide to install. OpenDaylight can either run on same host of Openstack control node or on separate Host. The host must have at least 3GB memory if running computer node and ODL controller. Both Openstack Swift and OpenDaylight use port 8080 for webpage GUI. You can either disable Openstack swift or move OpenDaylight webserver to different port. To move ODL webserver to different port, edit Connector port value in configuration/tomcat-server.xml under OpenDaylight directory:



```
<Service name="Catalina">
   <Connector   port="8081"   protocol="HTTP/1.1"
connectionTimeout="20000" redirectPort="8443" />
```

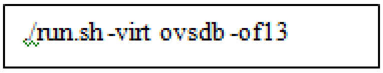
4.2 Configure ovsdb to connect to OpenDaylight and provide network information

Make sure it has following line for OpenFlow 1.3 config in the configuration/config.ini file. If not, add this line to config.ini file (or uncomment it if it is commented):



```
ovsdb.of.version=1.3
```

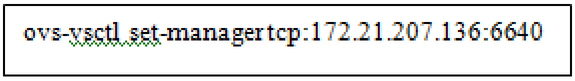
Start the controller with OVSDB and OpenFlow 1.3 option



```
./run.sh -virt ovsdb -of13
```

4.3 Configure ovsdb to connect to OpenDaylight and provide network information

The set-manager instructs ovsdb to connect to OpenDaylight as a manager.

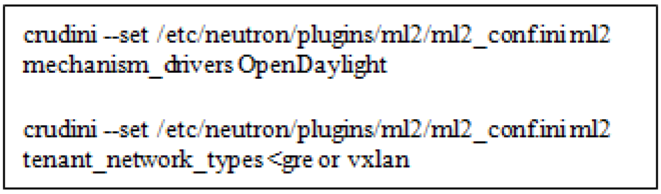


```
ovs-vsctl set-manager tcp:172.21.207.136:6640
```

4.4 Start Openstack

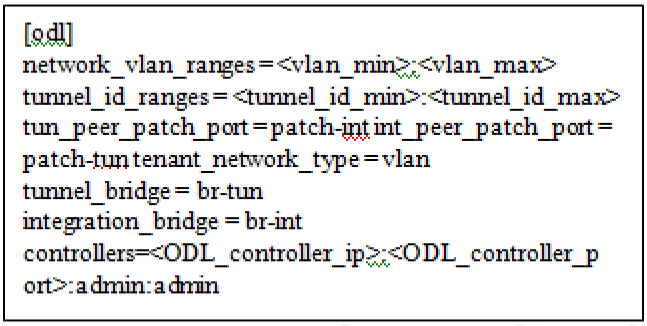
4.5 Enable ML2 plugin with OpenDaylight controller

Modify /etc/neutron/plugins/ml2/ml2_conf.ini in Openstack control node:



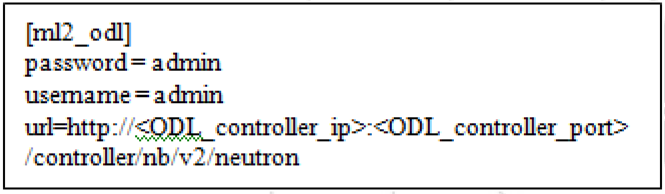
```
crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2
mechanism_drivers OpenDaylight

crudini --set /etc/neutron/plugins/ml2/ml2_conf.ini ml2
tenant_network_types <gre or vxlan
```

Configure the odl section in /etc/neutron/plugins/ml2/ml2_conf.ini in Openstack control node



```
[odl]
network_vlan_ranges=<vlan_min>:<vlan_max>
tunnel_id_ranges=<tunnel_id_min>:<tunnel_id_max>
tun_peer_patch_port=patch-int int_peer_patch_port=
patch-tun tenant_network_type=vlan
tunnel_bridge= br-tun
integration_bridge = br-int
controllers=<ODL_controller_ip>:<ODL_controller_p
ort>:admin:admin
```

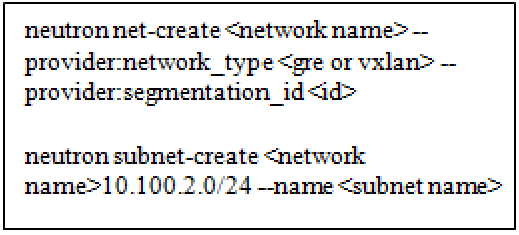
Configure the ml2_odl section in /etc/neutron/plugins/ml2_conf.ini in Openstack control node



```
[ml2_odl]
password=admin
username=admin
url=http://<ODL_controller_ip>:<ODL_controller_port>
/controller/nb/v2/neutron
```

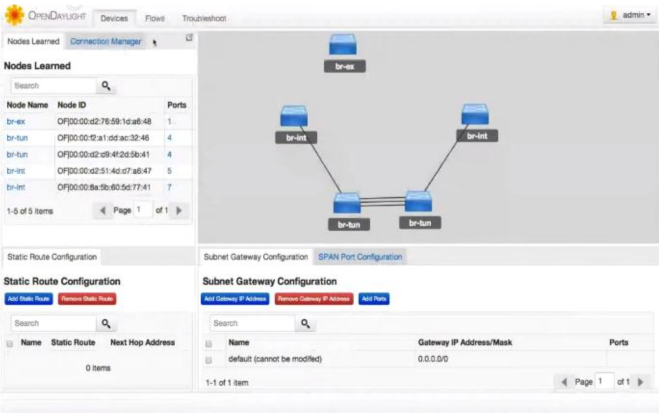
4.6 Create network and launch VM instances

Create network and attach subnet by neutron commands



```
neutron net-create <network name> --
provider:network_type <gre or vxlan> --
provider:segmentation_id <id>

neutron subnet-create <network
name>10.100.2.0/24 --name <subnet name>
```

Add VM instance to the subnet by running instances. VM on different compute nodes should be able to ping each other through GRE or VXLAN tunnels provisioned by ODL controller.



Nodes Learned

Node Name	Node ID	Ports
br-ex	OF|00:00:d2:76:59:1d:a6:48	1
br-tun	OF|00:00:f2:a1:dd:ac:32:46	4
br-tun	OF|00:00:d2:d9:4f:2d:5b:41	4
br-int	OF|00:00:d2:51:4d:d7:a6:47	5
br-int	OF|00:00:8a:5b:60:5d:77:41	7

1-5 of 5 items

Static Route Configuration

Name	Static Route	Next Hop Address

0 items

Subnet Gateway Configuration

Name	Gateway IP Address/Mask	Ports
default (cannot be modifed)	0.0.0.0/0	

1-1 of 1 item

5. Integration Advantages

- Visual representation of nodes and hosts in network topology
- Programmability
- Policies apply
- Policy based traffic steering on Openstack by integrating OpenDaylight
- Traffic steering on underlying network from the OpenDaylight controller
- Enables applications to create their own policy based on application properties such as $ cost , bandwidth , latency , time of day , SLA.

6. Conclusion and Future Work

In this paper I have addressed need of integration of OpenDaylight with Openstack and integration of both open source projects. In future I have plane to apply policy based traffic steering on Openstack using OpenDaylight DLUX . . It's Enables applications to create their own policy based on application properties such as cost , bandwidth , latency and SLA . Policy Based Traffic Steering means Network administrator selects data flows on the ground of analyses of network services and applications. As result administrator creates set of classification rules with specification of data flow attributes.

References

[1] Adrian Lara, "Simplifying network management using Software Defined Networking and OpenFlow," Journal of IEEE Xplore

[2] Dinesh C. Verma, "Policy-Based Management of Content Distribution Networks," Journal of IEEE Xplore

[3] Shafqat Ur Rehman, " Network-Wide Traffic Visibility in OF@TEIN SDN Testbed using sFlow," Journal of IEEE Xplore

[4] OpenDaylight: Java-based OpenFlow Controller, http://www.OpenDaylight.org/

[5] Openstack Nova configuration , Openstack wiki, http://docs.Openstack.org/juno/config-reference/content/section_compute-config-samples.html

[6] OpenFlow™: http://www.openflowswitch.org

[7] Openstack project website. Available: http://www.Openstack.org

Author Profile

Basivireddy pursuing the M.TECH in Cloud Computing from SRM University from 2013 to 2015. Now I am doing Internship in Alcatel-Lucent India Pvt Limited Company.