This article describes how to use the COI (Cisco OpenStack Installer) to setup a working OpenStack cluster, which utilizes Nexus ToR switches in VLAN mode.

Contents

Setting up the Nexus switch

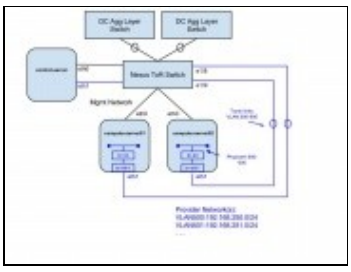
The following items need to be considered before proceeding with the Cisco Nexus Plugin deployment on the OpenStack Grizzly release.

- Your Nexus switch must be connected to a management network separate from the OpenStack data network. By default, we use the eth0 interface for this purpose. The plugin communicates with the switch over this network to set up your data flows.
- The switch must have SSH login and XML API enabled.
- Each compute host on the cloud should be connected to a port on the switch over a dedicated interface just for OpenStack data traffic. By default we use eth1 for this.
- The link from the node running the DHCP agent (by default, this is the OpenStack control node) should be a trunk link trunking all VLANs used for OpenStack traffic.
- Inter switch links should be trunk links trunking all VLANs.

Diagram

Figure 1 Can be used to visualize the layout of the OpenStack compute nodes and the attached Nexus switches.

*Figure 1:*Nexus Plugin Diagram



Installing the COI Cache and Build Servers

Below are the manual install steps for creating a build server.

Perform an update/upgrade and install Puppet, Git and ipmitool on the freshly installed Ubuntu 12.04 LTS node:

```
apt-get update && apt-get dist-upgrade -y && apt-get install -y puppet git ipmitool
```

NOTE: The system may need to be restarted after applying the updates. Get the Cisco OpenStack Installer (COI) example manifests. Under the grizzly-manifests GitHub repository you will find different branches, so select the one that matches your topology plans most closely. In the following examples the multi-node branch will be used, which is likely the most common topology:

```
git clone ?b multi-node https://github.com/CiscoSystems/grizzly-manifests
cd ~/grizzly-manifests
```

With a proxy:

```
https_proxy=http://proxy.example.com:80 git clone ?b multi-node https://github.com/CiscoSystems/gr
cd ~/grizzly-manifests
```

Copy the puppet manifests from ~/grizzly-manifests/manifests/ to /etc/puppet/manifests/

```
cp ~/grizzly-manifests/manifests/* /etc/puppet/manifests
```

Copy the puppet templates from ~grizzly-manifests/templates/ to /etc/puppet/templates/

```
cp ~/grizzly-manifests/templates/* /etc/puppet/templates
```

Get the Cisco OpenStack Installer puppet modules from Cisco's GitHub repository:

```
(cd /etc/puppet/manifests; python /etc/puppet/manifests/puppet_modules.py)
```

With a proxy:

```
(cd /etc/puppet/manifests; http_proxy=http://proxy.example.com:80 https_proxy=http://proxy.example
```

Run the puppet manifest site.pp file routine.

IMPORTANT! You must copy site.pp.example to site.pp and then edit it as appropriate for your installation. It is internally documented.

```
cp /etc/puppet/manifests/site.pp.example /etc/puppet/manifests/site.pp
vi /etc/puppet/manifests/site.pp
```

In addition to the basic site.pp modification for your environment, changes for the Nexus plugin needs to be made. Here are the items that may require customization in order to enable the use of the Nexus switch in VLAN mode. Other customizations for your site, such as host names, network address ranges, proxies and repositories are not explained here, but are documented as comments in the site.pp file. A few explanations follow the example:

```
$public_interface          = 'eth0'
$external_interface        = 'eth1'
$ovs_vlan_ranges           = 'physnet1:500:600'
$ovs_bridge_uplinks        = ['br-eth1:eth1']
$ovs_bridge_mappings       = ['physnet1:br-eth1']
$quantum_core_plugin       = 'cisco'
$cisco_vswitch_plugin      = 'ovs'
$cisco_nexus_plugin        = 'nexus'
$nexus_config = {
                  '10.121.10.39' => {
                                     'compute-server01' => '1/8',
                                     'compute-server02' => '1/9'
                                   },
                  '10.121.10.40' => {
                                     'compute-server03' => '1/10',
                                     'compute-server04' => '1/11',
                                     'compute-server05' => '1/12',
                                   }
                }
$nexus_credentials = ['10.121.10.39/my_username/my_password',
                      '10.121.10.40/my_username/my_password?]
$tenant_network_type = 'vlan'
```

Before specifying the VLAN address ranges in the **$ovs_vlan_ranges** parameter, please log into the switch to confirm that the specified address range is indeed free to use. The **$nexus_config** value is mapped, where for each Nexus switch (here identified by their IP addresses 10.121.13.39 and 10.121.13.40) we define a further map, specifying by hostname the various OpenStack compute servers, which are attached to that switch, as well as the port on that switch, to which the eth1 interface of that compute server is connected. In **$nexus_credentials** we specify for each of the Nexus switches a username and password for a user with management credentials.

Example Nexus ToR Configuration

```
interface Ethernet1/1
  description to N7k-agg-1
  switchport mode trunk
  switchport trunk allowed vlan 13,500-600

interface Ethernet1/2
  description to N7k-agg-2
  switchport mode trunk
  switchport trunk allowed vlan 13,500-600

interface Ethernet1/5
  description to control-server Management
  switchport mode trunk
  switchport trunk allowed vlan 13
  speed 1000

interface Ethernet1/6
  description to control-server Data
```

```
  switchport mode trunk
  switchport trunk allowed vlan 13,500-600
  speed 1000

interface Ethernet1/7
  description to compute-server01 Management
  switchport mode trunk
  switchport trunk allowed vlan 13
  speed 1000

interface Ethernet1/8
  description to compute-server01 Data
  switchport mode trunk
  speed 1000

interface mgmt0
  ip address 10.121.10.39/24

vrf context management
  ip route 0.0.0.0/0 10.121.10.1
```

The Nexus example configuration above has a trunk link configured to each upstream Nexus 7000 aggregation layer switch that includes the VLAN ranges that are defined in the site.pp (500-600). In this example the control node ("control-server") has two physical interfaces just like the compute nodes do. One for the management interface (VLAN 13) and the other for the data interface which is a trunk interface that is preconfigured with the same VLANs being used as the provider network ranges (500-600). This is needed if the DHCP agent is being deployed on the control server as it will need connectivity to assign address to instances on the compute nodes that are attached to VLANs 500-600. Also notice that the compute nodes have two physical interfaces and the data interface does not have a **switchport trunk allowed** command configured. This command will be implemented by the Cisco Nexus plugin when the first instance is launched for each VLAN defined in the site.pp file.

Quantum/Neutron Commands for Network/Subnet Creation

Create a provider network for VLAN 500. Name the network whatever you want (here we used "vlan500"). The provider network type is "vlan" which coincides with the site.pp entry **$tenant_network_type = 'vlan'**. The provider physical network is **physnet1** on eth1 which was mapped in the site.pp entry **$ovs_bridge_mappings = ['physnet1:br-eth1']**. The provider segmentation id of "500" simply refers to VLAN 500 defined in the site.pp entry **$ovs_vlan_ranges = 'physnet1:500:600'**. Finally, we defined that this is an external network with an upstream router.

```
quantum net-create vlan500 --provider:network_type vlan --provider:physical_network physnet1 --pro
```

Create a subnet for VLAN 500. Name the subnet whatever you want (here we used "subnet500"). Note: You don't have to actually enter a name for the subnet. Since our upstream aggregation layer switches are using HSRP (VRRP/GLBP are other options) and the addresses used on those switches are .1 = standby address, .2 = N7k-agg-1, .3 = N7k-agg-2, we need to create an allocation pool range that begins after those addresses. By default, the addresses for OpenStack use would begin at .1 and this would cause an IP address conflict. In this example we begin at .10 and end at .250 to account for any other management addresses that may be used. "vlan500" is the network name that we are attaching the subnet to. The subnet range is 192.168.250.0/24 and the DNS server address being assigned to the instances attached to the network is 10.121.12.10

```
quantum subnet-create --name subnet500 --allocation-pool start=192.168.250.10,end=192.168.250.250
```

The Nexus Plugin in Action

The Nexus Top of Rack switch has been configured, the Quantum/Neutron network and subnet have been defined and after you launch an instance and attach it to the network, we can see that the Cisco Nexus plugin has used the **$nexus_config** and **$nexus_credentials** information to login to the Cisco Nexus switch and defined a VLAN 500 and modified the appropriate interface for that VLAN.

```
vlan 500
  name q-500

interface Ethernet1/8
  description to compute-server01 Data
  switchport mode trunk
  switchport trunk allowed vlan 500
  speed 1000
```

All that is needed now is to continue configuring Quantum/Neutron networks and subnets and attached instances to those networks.

The Cisco Nexus switch configuration would be changed each time a new instance is created for a newly defined VLAN as the following example shows for VLAN 600:

```
vlan 500
  name q-500

vlan 600
  name q-600

interface Ethernet1/8
  description to compute-server01 Data
  switchport mode trunk
  switchport trunk allowed vlan 500,600
  speed 1000
```

Support

Email: openstack-support@cisco.com

Authors

Shannon McFarland

Juergen Brendel

Arvind Somya