OpenStack for Cisco DFA Install Guide for Using Pre-built OpenStack for Cisco DFA Images

Table of Contents

1 Hardware and Software Requirements

OpenStack for DFA software is currently packaged with COI (Cisco OpenStack Installer). The Cisco OpenStack Installer is qualified on:

- Ubuntu 12.04 LTS serves as a base operating system.
- KVM serves as the hypervisor.
- Cisco UCS C-Series serve as physical compute/storage hardware
- OpenStack for DFA software is based on open source software Grizzly release with Cisco DFA plugin developed at Cisco

We have developed an alternative mechanism that allows a user to install OpenStack for DFA software from pre-built/cloned images used for build server, OpenStack controller and compute nodes in the COI installation and deployment model. The target UCS needs to have a minimum of 500.1G hard disk space.

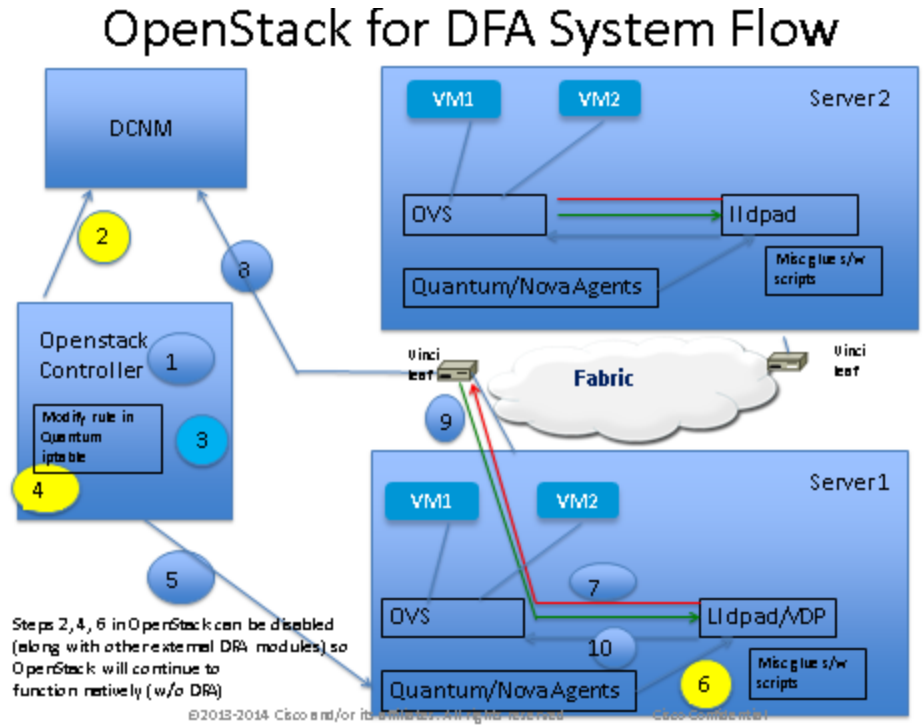
2 Pre-Requisite

As a pre-requisite to run OpenStack for DFA, please follow Dynamic Fabric Automation Deployment Guideline document to bring up DFA system first, including Nexus switches and DCNM software.

3 OpenStack for DFA Overview

OpenStack serves as one of orchestrators of the cloud enabled through DFA. For this release, all the orchestration is done using OpenStack's dashboard Horizon graphic user interface.

The following diagram provides an overview of the system in the DFA context:



The control flow can be summarized as follows:

1. Create a Network of type tunnel with tunnel ID range > 4K. (In OpenStack, tunnel ID is Segment ID)

2. Network Information (Subnet/mask, tenant name), is sent to DCNM via DCNM REST APIs

3. Instance (VM) is launched, specifying the Network that the instance will be a part of.

4. The current security rules in Openstack blocks any incoming DHCP frames from the outside world as well as add host specific iptable rules that will only allow frames with source IP addresses assigned by Openstack. The host specific iptable rules in quantum agent is modified to unblock it.

5. Network information (Subnet/mask, tenant name) and VM information is sent to the compute node.

6. Lldpad (VDP) gets notified about the VM and the segment_id associated with the VM.

7. VDP communicates with the Leaf passing the VM's information along with the Segment ID.

8. Leaf contacts DCNM with the Segment_id for retrieving the Network attributes

9. Leaf responds back with the VLAN to be used for tagging the VM's traffic

10. Lldpad module configures OVS for tagging the packets from the VM destined to the network with the value provided by the Leaf. => VM's VNIC is operational ONLY at this point

4 Before You Start

- We assume you have some general understanding of DFA (Dynamic Fabric Automation) system architecture.
- Please refer the Server/UCS Topology and OpenStack for DFA Installation section for pre-requisites
- Ensure CIMC connectivity is there
- Ensure that DCNM is reachable via the Openstack Controller as well as from the Vinci cluster

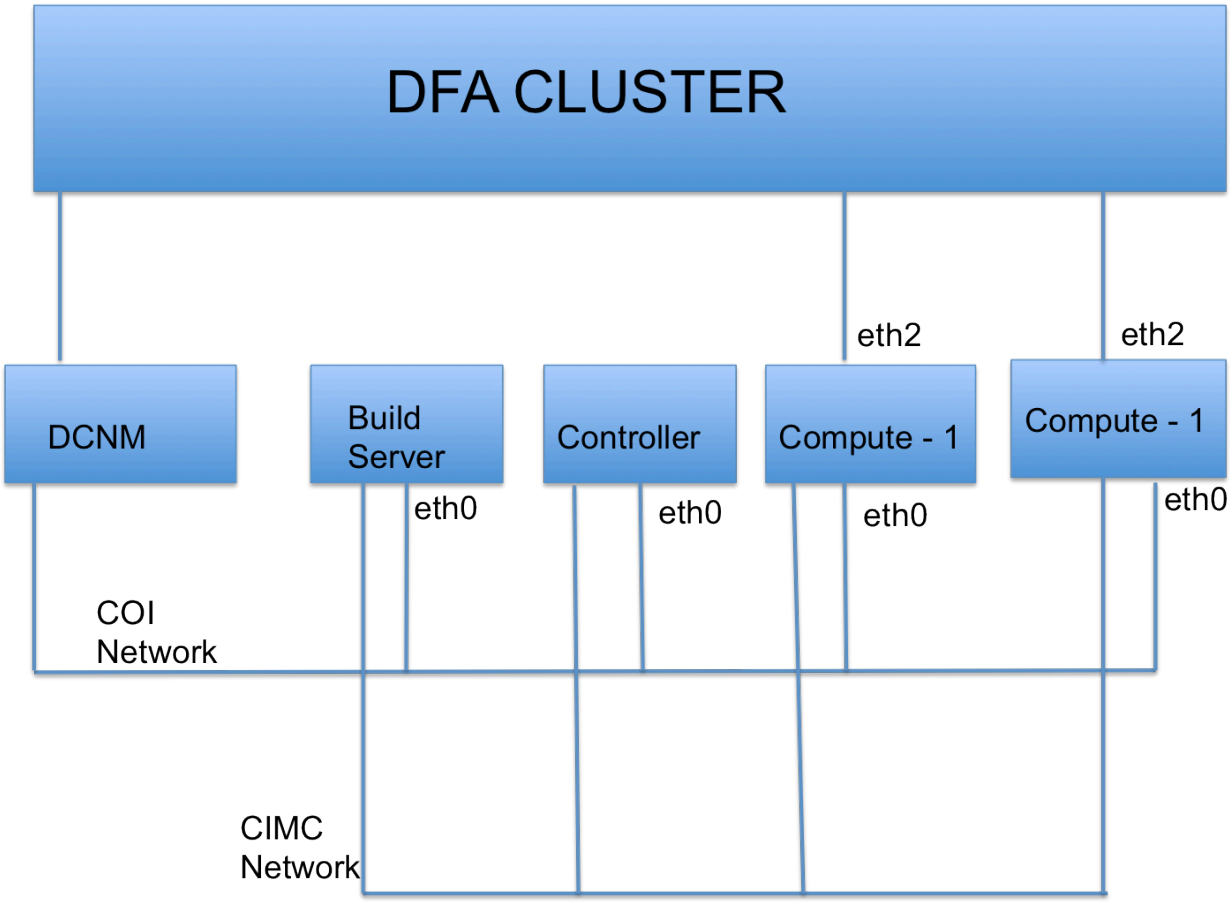
5 Servers/UCS Topology and OpenStack for DFA Installation

5.1 UCS resources required

- One UCS is dedicated for the puppet build server.
- One UCS is dedicated for the OpenStack Controller where the dashboard Horizon is running for doing orchestration.
- As many UCS as the number of Computes are needed. VM's run in the Computes.
- Ensure that all the nodes are configured with CIMC and they are reachable.

5.2 Servers/UCS topology

A sample topology is shown below. This is a critical step as it lays down your foundation so it is highly recommended that users follow the same wiring scheme.



Ensure that all the UCS' are connected to each other. The sample topology has the build server, the controller and the computes connected through eth0. This is referred as the COI network.

The computes have their (same) uplink interface(s) connected to the DFA cluster.

All the CIMC ports of the OpenStack nodes including the build server are connected to the CIMC network.

The DCNM is also connected to the COI network.

5.3 Installation

The installation procedure described here includes the following major steps. But more precisely, you don't specifically install COI-DFA. It is the pre-built/cloned images that already have a COI OpenStack for DFA installation and the procedure described here takes you to bring them up on your target UCS servers for OpenStack.

Step 1: Copy the 3 cloned images and the Clonezilla utility (iso image) to a repository server that can be reached by the target UCSs. The five prebuilt/cloned images are at:

http://software.cisco.com/download/release.html?mdfid=281722751&flowid=36334&softwareid=282088134&release=7.0%281%29&relind=AVAILABLE&rellifecycle=&reltype=latest.

NOTE:
The build node image is split into the following three binary files:
openstack_dfa_1_0_build_node_part1.tgz
openstack_dfa_1_0_build_node_part2.tgz
openstack_dfa_1_0_build_node_part3.tgz

Before proceeding, use the following command to reassemble the three files into a single TAR image:

```
cat openstack_dfa_1_0_build_node_part* >openstack_dfa_1_0_build_node.tgz
```

The preceding operation above takes a while to complete. Do not continue until the single TAR image is available.

The open source Clonezilla s/w for restoring the cloned/pre-built images is at:
http://www.clonezilla.org/downloads.php (we used clonezilla-live-20131125-saucy-amd64.iso).

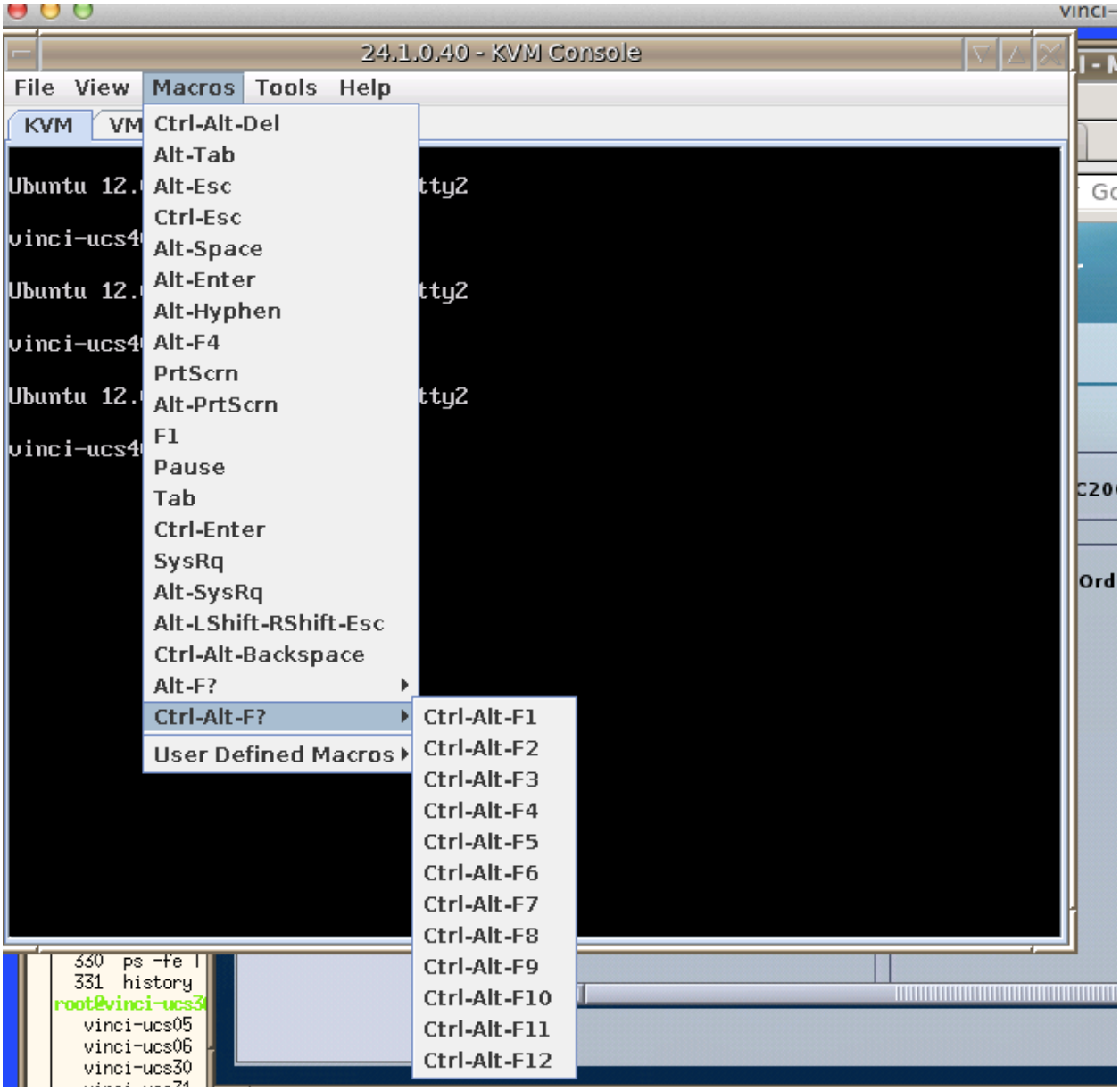
The original hostnames: ucs05, ucs34, and uc36, are in the OpenStack setup where the images were cloned/pre-built.

Step 2: Please follow Quick Guide to Clonezilla at
http://www.cisco.com/en/US/docs/switches/datacenter/dfa/openstack/install/guide/clonezilla-image-restore.pdf for instructions for restoring a cloned UCS image to a target UCS system. From the repository server above, launch the browser and remotely access to the CIMC port IP address (http;//<your UCS's CIMC port IP address>) of your target UCS, then use Cisco Integrated Management Controller for UCS to proceed.

Assuming the UCS' are all wired correctly, their connectivity is good and the steps described below are followed strictly, restoring a cloned UCS system each will take about 15-20 minutes. So if you have two compute nodes to install, then the total number of target systems will be four and that can take about 1 hour if the restoring execution is done here flawlessly. But this is not really related to OpenStack installation and it is simply a cloning restoring process.

The following steps are after clonezilla restoration and after rebooting the nodes:

On KVM console use CTRL-ALT-F2 to get the login window:



Build server Login/password: cisco/insecure (this is also sudo password)
Control/compute login/password : localadmin/cisco123 (this is also sudo password)

Step 3: Post-cloning COI/OpenStack fix-up that is detailed in the following section. Potentially most parts of this section will be automated so no/minimum manual step by step processing will be required in the future

5.3.1 Bring down COI service modules

These steps have to be done in KVM console:

After login
(After login to the nodes, always do: sudo –H bash)

a. power up the build node first, stop the apache2 service
   do: `service apache2 stop`
   `do: service cobbler stop`
b. power up controller node, stop the puppet service,
   do: `service puppet stop`
c. power up compute node and stop puppet service
   do: `service puppet stop`

5.3.2 Restore target UCS host names

Edit /etc/hosts :
- On all nodes : Add one entry per node in the setup:
  <ip> <hostname.domain.name> <hostname>
  <build ip address> <build hostname.domain.name> <build hostname>
  <controller ip address> <controller hostname.domain.name> <controller hostname>

  <compute ip address> <compute hostname.domain.name> <compute hostname>

…

Edit /etc/hostname :
- On all nodes : modify the hostname to new one.

Edit /etc/network/interfaces :
- On all nodes : Add network IP address for public interface.

As an example:

```
auto eth2
iface eth2 inet static
        address 172.28.12.88
        netmask 255.255.255.0
        network 172.28.12.0
        broadcast 172.28.12.255
        gateway 172.28.12.1
        # dns-* options are implemented by the resolvconf package, if
installed
        dns-nameservers 172.28.12.131
        dns-search cisco.com
```

5.3.2.1 Edit /etc/puppet/puppet.conf on build server node

- On build node : modify 'server' and 'certname' to the new build server's name.
- On Control and compute node: modify ‘server’ to the new build server’s name

5.3.2.2 On All Nodes:

- route del default gw <gateway ip>
- service networking restart
- route add default gw <gateway ip>
- hostname <the node's host name>

5.3.3 Fix-up on Build Server

There is one step in 5.3.3.2 AND 5.3.3.3 that applies to all nodes

5.3.3.1 Edit /etc/puppet/manifests/site.pp

This is a critical step that decides whether the installation will be successful.

On build node modify the site.pp for the new setup.

This is one of the most critical steps in the process. Read through the site.pp in detail, it's well documented. Take your time in filling this file. If there's a mistake in filling information in this file, the subsequent steps of puppet apply or even bringing up the controller/computes may not work. And the error messages are not at all intuitive. Also, follow the sample diff pasted below.

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

A sample diff is given below. Here vinci-ucs116 is the target build server, vinci-ucs119 is the target controller, vinci-ucs117 and vinci-ucs120 are the target computes.

These hosts will be changed to your target build server, controller and computes.

```
diff site.pp.example site.pp
9c9
< #$proxy = "http://proxy-server:port-number"
---
> $proxy = "http://proxy.esl.cisco.com:80"
42c42
< $build_node_name        = 'build-server'
---
> $build_node_name        = 'vinci-ucs116'
49c49
< $ntp_servers = ['time-server.domain.name']
---
> $ntp_servers = ['ntp.esl.cisco.com']
55,56c55,56
< $cobbler_node_ip        = '192.168.242.100'
< $node_subnet            = '192.168.242.0'
---
> $cobbler_node_ip        = '172.28.8.186'
> $node_subnet            = '172.28.8.0'
60c60
< $node_gateway           = '192.168.242.1'
---
> $node_gateway           = '172.28.8.1'
67c67
< $domain_name            = 'domain.name'
---
> $domain_name            = 'cisco.com'
94c94
```

```
< $password_crypted       =
'$6$UfgWxrIv$k4KfzAEMqMg.fppmSOTd0usI4j6gfjs0962.JXsoJRWa5wMz8yQk4SfInn4.WZ3L/MCt5u.62tHDGB36EhiK
F1'
---
> $password_crypted       =
'$6$UfgWxrIv$ge2goE1NYynQRW2yRimZLsjp6qwfruhmMCi4NCQHmClSVvhYlGo4yZiSDxjAm8433cU9t1hjlOGFjCbWbbR2
0.'
111,113c111,113
< $controller_node_address       = '192.168.242.10'
< $controller_node_network       = '192.168.242.0'
< $controller_hostname           = 'control-server'
---
> $controller_node_address       = '172.28.8.189'
> $controller_node_network       = '172.28.8.0'
> $controller_hostname           = 'vinci-ucs119'
118c118
< $db_allowed_network            = '192.168.242.%'
---
> $db_allowed_network            = '172.28.8.0'
162c162
< $service_address = $ipaddress_eth0
---
> $service_address = '172.28.8.189'
200,201c200,201
< $admin_password          = 'Cisco123'
< $keystone_db_password    = 'keystone_db_pass'
---
> $admin_password          = 'cisco123'
> $keystone_db_password    = 'cisco123'
203c203
< $mysql_root_password     = 'mysql_db_pass'
---
> $mysql_root_password     = 'cisco123'
205,206c205,206
< $nova_db_password        = 'nova_pass'
< $nova_user_password      = 'nova_pass'
---
> $nova_db_password        = 'cisco123'
> $nova_user_password      = 'cisco123'
208,209c208,209
< $glance_db_password      = 'glance_pass'
< $glance_user_password    = 'glance_pass'
---
> $glance_db_password      = 'cisco123'
> $glance_user_password    = 'cisco123'
212,216c212,216
< $cinder_user_password    = 'cinder_pass'
< $cinder_db_password      = 'cinder_pass'
< $quantum_user_password   = 'quantum_pass'
< $quantum_db_password     = 'quantum_pass'
< $rabbit_password         = 'openstack_rabbit_password'
---
> $cinder_user_password    = 'cisco123'
> $cinder_db_password      = 'cisco123'
> $quantum_user_password   = 'cisco123'
> $quantum_db_password     = 'cisco123'
> $rabbit_password         = 'cisco123'
218c218
< $swift_password          = 'swift_pass'
---
> $swift_password          = 'cisco123'
257c257
< # $quantum_core_plugin = 'ovs'
---
> $quantum_core_plugin = 'ovs'
304c304
< $test_file_image_type = 'kvm'
---
> $test_file_image_type = 'cirros'
343,346c343,346
<   cobbler_node { 'control-server':
<     mac            => '00:11:22:33:44:55',
<     ip             => '192.168.242.10',
<     power_address  => '192.168.242.110',
---
>   cobbler_node { 'vinci-ucs119':
```

```
>     mac            => '7c:ad:74:6f:72:0c',
>     ip             => '172.28.8.189',
>     power_address  => '24.1.0.119',
360,363c360,372
<   cobbler_node { 'compute-server01':
<     mac            => '11:22:33:44:55:66',
<     ip             => '192.168.242.21',
<     power_address  => '192.168.242.121',
---
>   cobbler_node { 'vinci-ucs120':
>     mac            => '7c:ad:74:6f:69:84',
>     ip             => '172.28.8.190',
>     power_address  => '24.1.0.120',
>     power_user     => 'admin',
>     power_password => 'password',
>     power_type     => 'ipmitool',
>   }
>
>   cobbler_node { 'vinci-ucs117':
>     mac            => '7c:ad:74:6f:7b:10',
>     ip             => '172.28.8.187',
>     power_address  => '24.1.0.117',
494c503
< node build-server inherits build-node { }
---
> node vinci-ucs116 inherits build-node { }
506c515
< $dcnm_ip_var = "10.1.1.1"
---
> $dcnm_ip_var = "172.28.11.156"
508c517
< $dcnm_pass_val = "xxxxx"
---
> $dcnm_pass_val = "cisco123"
511c520
< $mysql_host_val = "30.1.1.1"
---
> $mysql_host_val = "172.28.10.177"
513c522
< $mysql_pass_val = "xxxxx"
---
> $mysql_pass_val = "cisco123"
517,520c526,533
< #This is the start of the segment ID base. Ensure that this doesn't conflict
< #with other orchestrators segment ID
< $dfa_tunnel_base_var = "30000"
<
---
> $dfa_tunnel_base_var = "60000"
> #If a compute node is not directly connected to a leaf, but through any other
> #physical bridge device, then VDP frames will not reach the leaf. In such cases,
> #VDP frames should use other reserved MCast DMAC that will not be consumed by
> #the intermediate bridge. The Leaf and intermediate bridge also may need to be
> #appropriately configured. If there's no such configuration in your setup, leave
> #the below field as it is, otherwise, enter the Mcast DMAC that you plan to use.
> $non_nearest_brdmac = "01:88:C2:00:00:00"
528c541
< node 'control-server' inherits os_base {
---
> node 'vinci-ucs119' inherits os_base {
530c543
<     #enable_dhcp_agent     => false,
---
>     enable_dhcp_agent     => false,
536,545c549,560
<       uplink_intf    => 'eth1',
<       dcnm_ip_addr   => $dcnm_ip_var,
<       dcnm_username  => $dcnm_user_val,
<       dcnm_password  => $dcnm_pass_val,
<       mysql_host     => $mysql_host_val,
<       mysql_user     => $mysql_user_val,
<       mysql_password => $mysql_pass_val,
<       compute        => 'false',
<       gateway_mac    => $gateway_mac_val,
<       dfa_tunnel_base => $dfa_tunnel_base_var,
---
```

```
>       uplink_intf            => 'eth1',
>       dcnm_ip_addr           => $dcnm_ip_var,
>       dcnm_username          => $dcnm_user_val,
>       dcnm_password          => $dcnm_pass_val,
>       mysql_host             => $mysql_host_val,
>       mysql_user             => $mysql_user_val,
>       mysql_password         => $mysql_pass_val,
>       compute                => 'false',
>       gateway_mac            => $gateway_mac_val,
>       dfa_tunnel_base        => $dfa_tunnel_base_var,
>       non_nearest_bridge     => 'false',
>       non_nearest_bridge_mac => $non_nearest_brdmac,
597c612,646
< node 'compute-server01' inherits os_base {
---
> node 'vinci-ucs120' inherits os_base {
>   class { 'compute':
>     internal_ip        => '172.28.8.190',
>     #enable_dhcp_agent => true,
>     #enable_l3_agent   => true,
>     enable_ovs_agent  => true,
>   }
>   class { 'dfa':
>       # Modify this to the upstream interface of the compute server connected
>       # to the leaf. Currently, only one interface can be connected to the
>       # leaf
>       uplink_intf            => 'eth2',
>       dcnm_ip_addr           => $dcnm_ip_var,
>       dcnm_username          => $dcnm_user_val,
>       dcnm_password          => $dcnm_pass_val,
>       mysql_host             => $mysql_host_val,
>       mysql_user             => $mysql_user_val,
>       mysql_password         => $mysql_pass_val,
>       compute                => 'true',
>       gateway_mac            => $gateway_mac_val,
>       dfa_tunnel_base        => $dfa_tunnel_base_var,
>       # If Compute is not directly connected to leaf, but through any other
>       # intermediate physical bridge, enter 'true' here. It is case-sensitive.
>       non_nearest_bridge     => 'false',
>       non_nearest_bridge_mac => $non_nearest_brdmac,
>   }
>
>   # Reduce OVS log verbosity.
>   coe::ovs{ ['netdev', 'netdev_linux']:
>     facility    => 'file',
>     log_level   => 'err',
>   }
> }
>
> node 'vinci-ucs117' inherits os_base {
599c648
<     internal_ip        => '192.168.242.21',
---
>     internal_ip        => '172.28.8.187',
602c651
<     #enable_ovs_agent  => true,
---
>     enable_ovs_agent  => true,
608,617c657,670
<       uplink_intf    => 'eth3',
<       dcnm_ip_addr   => $dcnm_ip_var,
<       dcnm_username  => $dcnm_user_val,
<       dcnm_password  => $dcnm_pass_val,
<       mysql_host     => $mysql_host_val,
<       mysql_user     => $mysql_user_val,
<       mysql_password => $mysql_pass_val,
<       compute        => 'true',
<       gateway_mac    => $gateway_mac_val,
<       dfa_tunnel_base => $dfa_tunnel_base_var,
---
>       uplink_intf            => 'eth2',
>       dcnm_ip_addr           => $dcnm_ip_var,
>       dcnm_username          => $dcnm_user_val,
>       dcnm_password          => $dcnm_pass_val,
>       mysql_host             => $mysql_host_val,
>       mysql_user             => $mysql_user_val,
```

```
>       mysql_password         => $mysql_pass_val,
>       compute                => 'true',
>       gateway_mac            => $gateway_mac_val,
>       dfa_tunnel_base        => $dfa_tunnel_base_var,
>       # If Compute is not directly connected to leaf, but through any other
>       # intermediate physical bridge, enter 'true' here. It is case-sensitive.
>       non_nearest_bridge     => 'false',
>       non_nearest_bridge_mac => $non_nearest_brdmac,
```

Please make sure you set the following in build server's `/etc/puppet/manifests/site.pp` file matches the anycast_mac configured (either through DCNM PoAP screen or manual configuration) in the leaf switch connected to the OpenStack compute node

```
$gateway_mac_val = "00:00:DE:AD:BE:EF"
```

```
(switch)# sh run | inc anycast
fabric forwarding anycast-gateway-mac 0000.DEAD.BEEF
```

5.3.3.2 Build PKI for puppet

This step additionally need to be done on control and compute nodes.

- On all nodes : `rm -rf /var/lib/puppet/ssl`
  (or in case of keeping a backup: mv /var/lib/puppet/ssl /root/puppet_ssl_backup)
- On build node run: `puppet master`
  check for existence of /var/lib/puppet/ssl
- On build node : Kill the puppet process using pid of the process.
  `kill `ps -fe | grep puppet | awk '{print $2}'``

5.3.3.3 Edit /etc/apache2/sites-enabled/puppetmaster on build server node:

- On build node : modify the following line to have with new build server name:

SSLCertificateFile /var/lib/puppet/ssl/certs/<build server name>.cisco.com.pem
SSLCertificateKeyFile /var/lib/puppet/ssl/private_keys/<build server name>.cisco.com.pem

5.3.3.4 Apply puppet on build server node and restart dnsmasq

- On build node run: `puppet apply -v /etc/puppet/manifests/site.pp`

This step takes a short while to complete so be patient

verify apache2 is running by doing the following - `service apache2 status`

Restart dnsmasq :

- On build node run : `service dnsmasq restart`

5.3.4 Fix-up on control node

Keystone PKI update and certificate clean-up on control node need to be done here.

- On control node run :
- * `service keystone  stop`
  * `rm -rf /etc/keystone/ssl` (or mv /etc/keystone/ssl /root/keystone_ssl_backup)
  * `sudo -u keystone keystone-manage pki_setup`
  * `service keystone start`

verify:
`service keystone status` -- service is running

CA certificate clean up:

* `rm -rf /var/lib/quantum/keystone-signing` (or mv /var/lib/quanutm/keystone-signing /root/quantum_keystone-signing_bakcup)
* `rm -rf /var/lib/nova/CA` (or mv /var/lib/nova/CA /root/nova_CA_backup)

5.3.5 Starting Puppet on controller and computes

Do this on control and all compute nodes: `service puppet start`

And wait for the puppet run to completion before rebooting. Something like this should be seen:

"puppet-agent[2380]: Finished catalog run in 500 seconds" in /var/log/syslog (search this line)

5.3.5.1 In order to confirm keystone is started correctly on controller:

Make sure that openrc has this line:

export OS_AUTH_URL=http://<your controller ip address>:5000/v2.0/)

do: `source /root/openrc`

```
keystone service-list     -- has output
keystone endpoint-list    -- has output
```

5.3.6 Reboot the nodes

Suggest the following sequence:

1. power down the compute nodes first
2. reboot control node and check if you can login from your Chrome browser with admin/cisco123
3. power up your compute nodes

After Compute node is rebooted and is up, wait for the puppet to run and then ensure that the daemons LLDPAD and Packet Capture are running as follows and also wait for the puppet run to complete as given above:

```
pidof lldpad
pidof pktcpt
```

5.4 Images for Launching a VM

There are two images that can be used for launching an instance from the image list:

- cirros image login/password -- cirros/cubswin:)
- Ubuntu-mini image login/password -- ubuntu/ubuntu

5.5 Configuration in the Leaf or DCNM

5.5.1 Leaf Interface Config and Global Config

- For the leaf interface connected to the UCS server, “fabric forwarding port-tracking” configuration needs to be done:

  int e1/35
  switchport mode trunk

- Need to set the speed of the interface connected to compute node correctly (currently DCNM PoAP screen does not allow you to configure this)

  fabric forwarding port-tracking
  speed 1000
  no shut

- Add the global command:

  evb batch-response disable

- Again, make sure the leaf node connected to the OpenStack compute node needs to have the same anycast gateway mac as in the site.pp file of the build server:

  N6k-23# sh run | inc anycast

  fabric forwarding anycast-gateway-mac 0000.DEAD.BEEF

- There are some LC's where the interface connecting the leaf and UCS does not come up. Try the following workaround.

  1. On the leaf, load the debug plugin.
  2. cd /mnt/pss
  3. rm norcal_disable_qsfp
  4. Reload the leaf

5.5.2 DHCP Configuration in DCNM

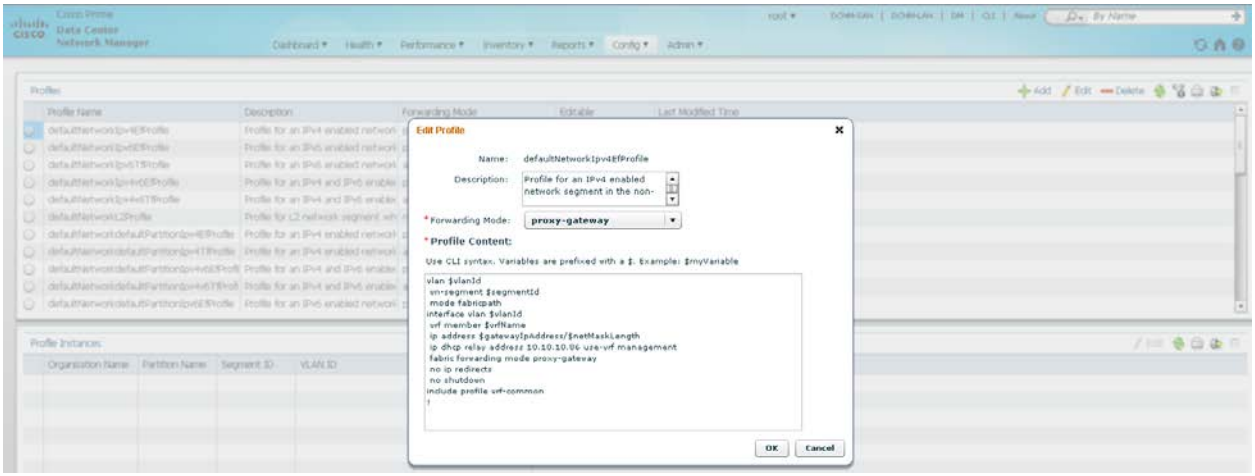
- For auto-configuration to work, the profile's DHCP address needs to be modified manually in DCNM. This is because Openstack does not populate the DHCP address to DCNM, since it has no way of knowing this information. For example, under a profile, you will see a config like the below:

  ```
  ip dhcp relay address $dhcpServerAddr use-vrf management
  ```

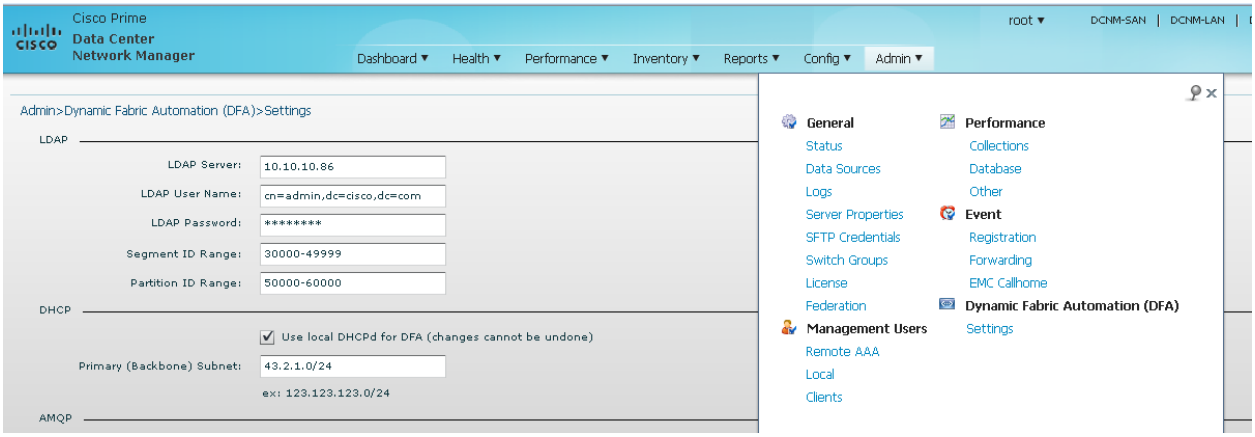
  Modify this to the actual IP address of the DHCP server. Like:

  ```
  ip dhcp relay address 10.10.2.86 use-vrf management
  ```

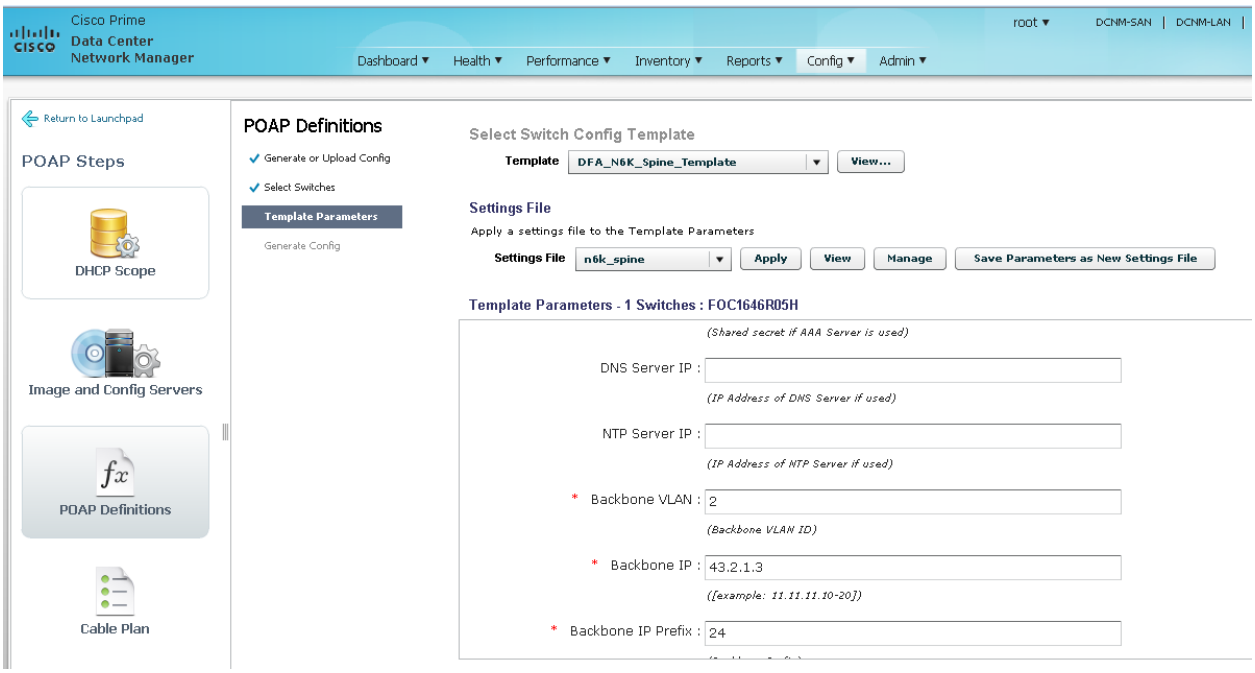
  Go to “Config”, then DFA Profiles:



- Also fill the DFA backbone vlan subnet value in the “Settings” of Dynamic Fabric Automation (DFA) section under Admin tap in DCNM console.



- Need make sure that Primary (Backbone) subnet with “Use local DHCPd for DFA” checked is consistent with the backbone vlan subnet for the leaf and spine nodes in the DFA fabric. If the fabric is brought up from PoAP in DCNM, the configuration screen will look like the following:



6 Create Project and Launch VM

The information provided in this section is generic to OpenStack and it is provided here for your convenience with the exception of “ConfigProfile”, which is DFA specific.

6.1 KNOWN CAVEATS

- When a project is created, do not have hyphen "-" in the project name.
- Bulk Create and Delete of VM's is supported only to an extent. Pls refer the caveats section.
- Do not give space for a network name or project name or a VM name.

6.2 Steps to create a Project:

- Login to Horizon as admin, password “cisco123”. Click on "projects" and "create project"
- Enter a Name for the project, DO NOT have hyphen "-" in the project name.
- Click on "Create Project".

6.3 Steps to create a User for the project:

- Click on Users and then do "Create User"
- Fill in all the fields, select the project you just created and select the role as "admin". The Network information will not be populated to DCNM correctly, if you fail to specify the role as "admin".

6.4 Steps to create the network:

- Login as the user just created.
- Click on the "project" tab.
- Click on "Networks" and then "Create Network". Specify a Name for the network and go to the subnet tab. This is MANDATORY.
- Specify a Network Address for the subnet.
- Click on the "Subnet Detail" tab and then uncheck the "Enable DHCP" option.
- Go to the "ConfigProfile" tab and select the profile from the pull down menu

6.5 Steps to launch the VM:

- Click on "Instances" and then "Launch Instance"
- Click on "Image" drop down menu and select the image. There will be cirros image by default.
- Give a name for the Instance.
- Go to the "Networking" tab and select the network from "Available network".
- Click "Launch".

7 Scalability

OpenStack is supposed to support many compute nodes from the control node running the Horizon dashboard. This should not be related to the added DFA functionality.

By default without specific configuration, OpenStack supports 10 VMs per compute node. This is the recommended number for a user. Our QA has qualified about 20 or less VM's/instances per each OpenStack compute node.

Batching VM creation and deletion with OpenStack should be used with great caution. We support 10 or less VMs per batch creation and deletion on two compute nodes (i.e., tested). Batching capability beyond that is best-effort.

8 Known Limitations and Caveats

- Currently live-migration/vmotion is not supported as it is not supported in COI Grizzly release
- OpenStack does not support IPv6 yet
- No service integration supported
- All added DFA functionality in OpenStack is done through the Horizon dashboard for this release.
- If a compute node with live VMs gets reloaded, OpenStack will lose these VM's
- Currently a single port of a server running OpenStack is allowed to connect to a DFA leaf port

- It is required that the compute nodes should always be connected to the build server at least during reboots and puppet should not be disabled
- vm's vnic interface up/down not detected by vdp (use VM deletion/creation as potential workaround)
- Observed LLDPad does not come up after COI installation so a manual reload is necessary as a workaround for now
- All the subnet creation for network should be done through Horizon project/network menu. Other way of creating network subnet might not work.
- All orchestration tasks are done through the Horizon dashboard for this release
- VM provisioned through Horizon will always get its IP address via DHCP server in DCNM
- Cirros and mini Ubuntu images are currently tested

9 Technical Support Model

- OpenStack is based on open source and is generally supported through its community using a best effort approach.
- All general COI installation support will be from Cisco COI team. See the Before You Start section above.
- COI-DFA part support will be from Cisco DFA team.
- DFA part of the OpenStack DFA support will come from Cisco DFA team.