Cisco DFA Application Programmer's API Guide 7.x

Cisco Dynamic Fabric Automation
January 2014

Cisco Systems, Inc.
www.cisco.com

Cisco has more than 200 offices worldwide. Addresses, phone numbers, and fax numbers are listed on the Cisco website at www.cisco.com/go/offices.

Text Part Number: OL-31170-01

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Any Internet Protocol (IP) addresses and phone numbers used in this document are not intended to be actual addresses and phone numbers. Any examples, command display output, network topology diagrams, and other figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses or phone numbers in illustrative content is unintentional and coincidental.

© 2014 Cisco Systems, Inc. All rights reserved.

Preface

Organization

This guide includes the following sections:

Section	Title	Description
1	DFA REST API	Describes the REST API's for Cisco Dynamic Fabric Automation.

Conventions

This document uses the following conventions:

Convention	Indication
bold font	Commands and keywords and user-entered text appear in **bold** font.
italic font	Document titles, new or emphasized terms, and arguments for which you supply values are in *italic* font.
[ ]	Elements in square brackets are optional.
{x | y | z }	Required alternative keywords are grouped in braces and separated by vertical bars.
[x | y | z]	Optional alternative keywords are grouped in brackets and separated by vertical bars.
string	A nonquoted set of characters. Do not use quotation marks around the string or the string will include the quotation marks.
`courier` font	Terminal sessions and information the system displays appear in `courier` font.
< >	Nonprinting characters such as passwords are in angle brackets.
[ ]	Default responses to system prompts are in square brackets.
!, #	An exclamation point (!) or a pound sign (#) at the beginning of a line of code indicates a comment line.

Note Means *reader take note*. Notes contain helpful suggestions or references to material not covered in the manual.

REVIEW DRAFT—CISCO CONFIDENTIAL

Tip Means *the following information will help you solve a problem*. The tips information might not be troubleshooting or even an action, but could be useful information, similar to a Timesaver.

Caution Means *reader be careful*. In this situation, you might perform an action that could result in equipment damage or loss of data.

Timesaver Means *the described action saves time*. You can save time by performing the action described in the paragraph.

Warning **IMPORTANT SAFETY INSTRUCTIONS**

This warning symbol means danger. You are in a situation that could cause bodily injury. Before you work on any equipment, be aware of the hazards involved with electrical circuitry and be familiar with standard practices for preventing accidents. Use the statement number provided at the end of each warning to locate its translation in the translated safety warnings that accompanied this device.

SAVE THESE INSTRUCTIONS

Warning **Statements using this symbol are provided for additional information and to comply with regulatory and customer requirements.**

CONTENTS

Cisco Dynamic Fabric Automation API

Overview

The Cisco Dynamic Fabric Automation (DFA) APIs for third party applications enables you to programmatically control Cisco Dynamic Fabric Automation (DFA). The DFA API supports POAP (Power On Auto Provisioning), Auto Config, DHCP and Cable plan features.

POAP allows devices to boot up with temporary IP address (assigned by DHCP server), to download the POAP boot-up script (also assigned by DHCP server) which will further download the required kick-start and system image, and the device configuration file from the specific TFTP server indicated in the boot-up script.

In the Cisco Dynamic Fabric Automation architecture, the virtual machine (VM) facing interface on leaf switches is automatically configured and de-configured by the auto config. It detects the server/VM boot-up, retrieves pre-defined network parameters from the asset database, and applies the generated configuration. When the VM is moved or shutdown, the auto config also updates/removes the dynamic configuration.

Cable plan provides a Netmap of port-to-port cable connectivity data that can be imported into the switches of the DFA. The XML cable plan acts as a lookup table, and if a port is not connected to its corresponding destination port as per the plan, the switch should flag an error and notify the customers.

Note All the REST API operations can also be performed using the DCNM GUI as DCNM uses these REST APIs to render the GUI. Therefore it is not mandatory to programmatically controlDFA via third party applications since DFA can be controlled via the DCNM GUI. However if it is necessary to augment or replace DCNM GUIs with customized controls, the RESTAPIs provide an open interface for third party applications to control DFA programmaticallly.

Software Architecture

In a DFA datacenter, Cisco DCNM will be the central point of management for the fabric and for the network auto-configuration. PoAP templates are used to auto-configure the spine and leaf network devices and configuration profiles are used to auto-configure the organizations, networks and services.

DCNM works as the network controller in conjunction with any instances of compute/storage orchestrators and service controllers to provide an open and extensible integrated virtual and physical data center. Organizations and networks can be created directly using the Cisco DCNM GUIs or through the compute/storage orchestrators. In both cases the external APIs discussed in this document are used to retrieve information and create/retrieve/update/delete configuration profile instances into the network

REVIEW DRAFT—CISCO CONFIDENTIAL

Asset Database (LDAP). The leaf devices in turn fetch configuration from the Asset Database and self-configure themselves. Service controllers like PNSC can also get organization/network information from DCNM through the APIs, as well as update the network configuration profiles for services integration.

The general workflow is as follows:

- Create Organizations and Partitions
- As part of this, edge services may be deployed
- Create Segments
- As part of this, segment services may be deployed
- Deploy application workload

Throughout this process the network and network services can be automated using DFA. For an animated description of the DFA architecture and workflow refer to http://www.youtube.com/watch?v=MNnv2Y_k6EY.

Note You can also use your own REST based third party orchestrators or applications to interface with DCNM to control DFA..

REST APIs

DCNM will provide REST API, and the REST approach emphasizes on using the resource name as part of the URL. In this release, the response of the REST API will be encoded in JSON format (see http://www.json.org for validation).

The query parameters for the HTTP GET will be appended to the URL after the symbol "?"; the input for the HTTP POST/PUT/DELETE will be specified in the payload with the URL-encoded. REST APIs support both HTTP and HTTPS.

This section contains the high level description for the REST API, during implementation the REST APIs are subject to change.

DCNM REST API supports "application/json" for the Content-Type.

The following parameter types are mentioned in this document:

- A - Array
- S - String
- O – Object

The DFA REST APIs control the following categories of functionalities:

- Authentication
- Auto-configuration
- DHCP
- POAP
- Cable Plan

For detailed information about the Cisco DFA REST APIs, see the *Cisco DCNM 7.0 REST API Guide*.

REVIEW DRAFT—CISCO CONFIDENTIAL

Authentication

The Authentication REST APIs can be used by an external application to authenticate itself to the DCNM in order to programmatically control the DFA cluster. After calling logon to get the token, all the subsequent REST API requests need to set the DCNM-Token field with the token in the HTTP header.

The Authentication REST APIs are mentioned inTable 1-1 on page 1-3.

Table 1-1 Authentication REST API

API Function	**HTTP Method**	**Resource at URL** https://dcnm-ip/rest/
Logon	POST	`/logon`
Logout	POST	`/logout`

REVIEW DRAFT—CISCO CONFIDENTIAL

Auto Configuration

Overview

Auto Config REST APIs can be used by an external application to programmatically manage the configuration profiles used to deploy organizations, partitions and networks in the DFA cluster.

REVIEW DRAFT—CISCO CONFIDENTIAL

Auto Config Database Schema

Organization

Attributes	Type	Require
organizationName	String	Mandatory
description	String	Optional
orchestrationSource	String	Optional

Partition

Attributes	Type	Require
partitionName	String	Mandatory
partitionSegmentId	String	Mandatory
description	String	Optional
serviceNodeIpAddress	String	Optional
organizationName	String	Mandatory
dnsServer	String	Optional
secondaryDNSServer	String	Optional
configArg	String	Optional

Network

Attributes	Type	Require
organizationName	String	Mandatory
dvsId	String	Optional
staticIpStart	String	Optional
networkRole	String	Mandatory
gateway	String	Optional
netmaskLength	String	Optional
gatewayIpv6Adddress	String	Optional
prefixLength	String	Optional
secondaryGateway	String	Optional
staticIpEnd	String	Optional
vSwitchControllerNetworkId	String	Optional
networkName	String	Mandatory
segmentId	String	Mandatory

REVIEW DRAFT—CISCO CONFIDENTIAL

Attributes	Type	Require
vlanId	String	Mandatory
mobilityDomainId	String	Mandatory
description	String	Optional
profileName	String	Mandatory
vSwitchControllerId	String	Optional
configArg	String	Optional
partitionName	String	Mandatory
dhcpScope	subnet	Optional
	gateway	Optional
	ipRange	Optional

Profile

Attributes	Type	Require
forwardingMode	String	Mandatory
profileName	String	Mandatory
description	String	Optional
configCommands	String	Mandatory

Auto Config REST APIs

The Auto Config REST APIs are mentioned in Table 2-1 on page 2-3

Table 2-1 Auto Config REST APIs

API Function	HTTP Method	Resource at URL https://dcnm-ip/rest/
List organizations	GET	`/auto-config/organizations`
List organizations - Detailed	GET	/auto-config/organizations?detail=true
Create an organization	POST	`/auto-config/organizations`
Get an organization	GET	`/auto-config/organizations/organization-name`
Update an organization	PUT	`auto-config/organizations/organization-name`
Delete an organization	DELETE	`auto-config/organizations/organization-name`
List partitions	GET	`/auto-config/organizations/organization-name/partitions`
Create a partition	POST	`/auto-config/organizations/organization-name/partitions`

REVIEW DRAFT—CISCO CONFIDENTIAL

API Function	HTTP Method	Resource at URL https://dcnm-ip/rest/
Get a partition	GET	/auto-config/organizations/organization-name/partitions/partition-name
Update a partition	PUT	/auto-config/organizations/organization-name/partitions/partition-name
Delete a partition	DELETE	/auto-config/organizations/organization-name/partitions/partition-name
List networks	GET	/auto-config/organizations/organization-name/partitions/partition-name/networks
Create a network	POST	/auto-config/organizations/organization-name/partitions/partition-name/networks
Get a network	GET	/auto-config/organizations/organization-name/partitions/partition-name/networks/{network-id}
Update a network	PUT	/auto-config/organizations/organization-name/partitions/partition-name/networks/{network-id}
Delete a network	DELETE	/auto-config/organizations/organization-name/partitions/partition-name/networks/{network-id}
List profiles	GET	/auto-config/profiles
Create a profile	POST	/auto-config/profiles
Get a profile	GET	/auto-config/profiles/profile-name
Update a profile	PUT	/auto-config/profiles/profile-name
Delete a profile	DELETE	/auto-config/profiles/profile-name
Get auto config settings	GET	/auto-config/settings
Update auto config settings	PUT	/auto-config/settings

Configuration Profiles Construct

The Dynamic Fabric Automation (DFA) network auto-configuration requires the use of configuration profiles (config profiles) to instantiate the required network onto a leaf. Config-profile templates are parameterized templates that allow for the instantiation of specific network config-profiles. They are analogous to a class in object oriented terminology. The combination of a class object with specific parameters represents an instance of that object.

Currently DFA supports two types of templates (aka classes); **org:partition config profile** and the **network config profile**. The network config-profile can optionally "include" the **org:partition** config profile in order to create a composite object which is an **organization** network config-profile. Many network config-profiles may include the same partition config profile if many networks belong to the same organization. If a network config-profile does not include a partition by name, then it exists in the default organization.

```
config-profile <network profile name>
[include profile <partition profile name>]
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
end
```

Note **config-profile + <*parameters*> → org network config-profile instance.**

Once a network config-profile instance is populated in DCNM, it can then be pulled dynamically by leafs to instantiate the network onto the leafs.

Typically, config-profiles will be defined by the network administrator. A config-profile has a `ProfileName` and set of parameterized commands associated with it. The parameters are stored in the LDAP database. The REST APIs , can be employed to programmatically create, read, delete, and update these profiles and parameters from external applications as documented in the section in Appendix B, "Auto Config Examples".

Note The set of config-profiles created by one set of applications (Example; network administration applications) can further be made available to another set of applications (Example; compute/storage orchestration engines such as Openstack, UCS Director or any 3rd party applications) through the documented REST APIs.

Example 2:
The following is an example of a partition config-profile included by a network config-profile that can be used or created through the REST APIs.

```
config profile vrf-common
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv4 unicast
     route-target import 111:222
     route-target both auto
    address-family ipv6 unicast
     route-target import 111:222
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       redistribute direct route-map FABRIC-RMAP-REDIST-SUBNET
       maximum-paths ibgp 2
      address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       redistribute direct route-map FABRIC-RMAP-REDIST-SUBNET
       maximum-paths ibgp 2
end

config profile defaultNetworkIpv4EfProfile
vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   fabric forwarding mode proxy-gateway
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
   no ip redirects
   no shutdown
  include profile vrf-common
end

When this config-profile is instantiated with specific parameters it may
appear as follows:

   vrf context Pepsi
    vni 802004
    rd auto
    address-family ipv4 unicast
     route-target import 111:222
     route-target both auto
    address-family ipv6 unicast
     route-target import 111:222
     route-target both auto
    router bgp 100
     vrf Pepsi
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       redistribute direct route-map FABRIC-RMAP-REDIST-SUBNET
       maximum-paths ibgp 2
      address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       redistribute direct route-map FABRIC-RMAP-REDIST-SUBNET
       maximum-paths ibgp 2

vlan 3000
   vn-segment 11000
   mode fabricpath
interface vlan 3000
   vrf member Pepsi
   ip address 1.1.1.254/24
   fabric forwarding mode proxy-gateway
   no ip redirects
   no shutdown
```

Pre-Packaged Config-Profiles

To cover most of the common scenarios, the following table lists the set of config-profiles that will come pre-packaged into the DCNM profiles database

REVIEW DRAFT—CISCO CONFIDENTIAL

Table 2-2 DCNM Pre-Packaged Config-Profiles

Profile Type	Profile Name	Forwarding Mode	Profile Description
Network	defaultNetworkIpv4EfProfile	proxy-gateway	Profile for an IPv4 enabled network segment in the non-default partition with DFA Enhanced Forwarding mode.
Network	defaultNetworkIpv4TfProfile	anycast-gateway	Profile for an IPv4 enabled network segment in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkIpv6EfProfile	proxy-gateway	Profile for an IPv6 enabled network segment in the non-default partition with DFA Enhanced Forwarding mode.
Network	defaultNetworkIpv6TfProfile	anycast-gateway	Profile for an IPv6 enabled network segment in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkIpv4v6EfProfile	proxy-gateway	Profile for an IPv4 and IPv6 enabled network segment in the non-default partition with DFA Enhanced Forwarding mode.
Network	defaultNetworkIpv4v6TfProfile	anycast-gateway	Profile for an IPv4 and IPv6 enabled network segment in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkdefault PartitionIpv4EfProfile	proxy-gateway	Profile for an IPv4 enabled network segment in the non-default partition with an edge service node with DFA Enhanced Forwarding mode.
Network	defaultNetworkIpv4v6TfProfile	anycast-gateway	Profile for an IPv4 and IPv6 enabled network segment in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkdefaultPartitionIpv4EfProfile	proxy-gateway	Profile for an IPv6 enabled network segment in the default partition with DFA Enhanced Forwarding mode.

REVIEW DRAFT—CISCO CONFIDENTIAL

Profile Type	Profile Name	Forwarding Mode	Profile Description
Network	defaultNetworkdefaultPartitionIpv6TfProfile	anycast-gateway	Profile for an IPv6 enabled network segment in the default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkdefaultPartitionIpv4v6EfProfile	proxy-gateway	Profile for an IPv4 and IPv6 enabled network segment in the default partition with DFA Enhanced Forwarding mode
Network	defaultNetworkdefaultPartitionIpv4v6TfProfile	anycast-gateway	Profile for an IPv4 and IPv6 enabled network segment in the default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkL2GblVlanProfile	none	Profile for L2 network that is a global vlan
Network	defaultNetworkL2Profile	none	Profile for L2 network segment where DFA L3 routing is not enabled. Another node (service node or router) attached to a leaf node can do the routing as needed.
Network	defaultNetworkIpv4TfGblVlanProfile	anycast-gateway	Profile for a global vlan based IPv4 enabled network in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkIpv6TfGblVlanProfile	anycast-gateway	Profile for a global vlan based IPv6 enabled network in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkIpv4v6TfGblVlanProfile	anycast-gateway	Profile for a global vlan based IPv4 and IPv6 enabled network in the non-default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkdefaultPartitionIpv4TfGblVlanProfile	anycast-gateway	Profile for a global vlan based IPv4 enabled network in the default partition with DFA Traditional Forwarding mode

REVIEW DRAFT—CISCO CONFIDENTIAL

Profile Type	Profile Name	Forwarding Mode	Profile Description
Network	defaultNetworkdefaultPartitionIpv6TfGblVlanProfile	anycast-gateway	Profile for a global vlan based IPv6 enabled network in the default partition with DFA Traditional Forwarding mode.
Network	defaultNetworkdefaultPartitionIpv4v6TfGblVlanProfile	anycast-gateway	Profile for a global vlan based IPv4 and IPv6 enabled network in the default partition with DFA Traditional Forwarding mode
Network	defaultNetworkIpv4EfEdgeServiceProfile	Proxy-gateway	Profile for an IPv4 enabled network segment in the non-default partition with an edge service node in DFA enhanced forwarding mode.
Network	defaultNetworkIpv4TfEdgeServiceProfile	anycast-gateway	Profile for an IPv4 enabled network segment in the non-default partition with an edge service node with DFA Traditional forwarding mode.
Network	defaultNetworkIpv4v6EfEdgeServiceProfile	proxy-gateway	Profile for an IPv4 and IPv6 enabled network segment in the non-default partition with an edge service node with DFA Enhanced Forwarding mode.
Network	defaultNetworkIpv4v6TfEdgeServiceProfile	anycast-gateway	Profile for an IPv4 and IPv6 enabled network segment in the non-default partition with an edge service node with DFA Traditional forwarding mode.
Network	externalNetworkIpv4TfStaticRoutingProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default external partition used for Edge service node with static routing in DFA traditional forwarding mode.
Network	serviceNetworkIpv4TfStaticRoutingProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default partition used for service node interface with static routing in DFA traditional forwarding mode.

REVIEW DRAFT—CISCO CONFIDENTIAL

Profile Type	Profile Name	Forwarding Mode	Profile Description
Network	serviceNetworkIpv4TfDynamic RoutingProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default partition used for service node interface with dynamic routing in DFA traditional forwarding mode
Network	externalNetworkIpv4TfDynamic RoutingProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default external partition used for Edge service node with dynamic routing in DFA traditional forwarding mode
Network	serviceNetworkIpv4TfL3Vpath ServiceNodeProfile	anycast-gateway	Profile for an IPv4 enabled vPath L3 mode service network segment used for vPath service nodes in DFA traditional forwarding mode
Network	serviceNetworkIpv4EfL3Vpath ServiceClassifierProfile	proxy-gateway	Profile for an IPv4 enabled vPath L3 mode service network segment used for service classifiers in DFA enhanced forwarding mode
Network	serviceNetworkL2VpathProfile	none	Profile for vPath L2 mode service network segment used for vPath service nodes
Network	serviceNetworkIpv4TfLBProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default partition used for a Load Balancer service node in one-armed routed mode.
Network	defaultNetworkIpv4EfLBProfile	proxy-gateway	Profile for an IPv4 enabled network segment in DFA enhanced forwarding mode in the non-default partition that has a Load Balancer service node
Network	defaultNetworkIpv4TfLBProfile	anycast-gateway	Profile for an IPv4 enabled network segment in DFA traditional forwarding mode in the non-default partition that has a Load Balancer service node.

REVIEW DRAFT—CISCO CONFIDENTIAL

Profile Type	Profile Name	Forwarding Mode	Profile Description
Network	serviceNetworkIpv4TfLBChainlLBESProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default partition comprising a service chain with a Load Balancer (LB) and Tenant Edge Firewall both in routed mode with dynamic routing enabled. This service segment is used for the LB
Network	serviceNetworkIpv4TfEdgeServicesChainlLBESProfile	anycast-gateway	Profile for an IPv4 enabled service network segment in the non-default partition comprising a service chain with a Load Balancer (LB) and Tenant Edge Firewall both in routed mode with dynamic routing enabled. This service segment is used for the tenant edge firewall
Network	defaultNetworkIpv4EfChainLBESProfile	proxy-gateway	Profile for an IPv4 enabled network segment in DFA enhanced forwarding mode in the non-default partition comprising a service chain with a Load Balancer (LB) and Tenant Edge Firewall both in routed mode with dynamic routing enabled
Network	defaultNetworkIpv4TfChainLBESProfile	anycast-gateway	Profile for an IPv4 enabled network segment in DFA traditional forwarding mode in the non-default partition comprising a service chain with a Load Balancer (LB) and Tenant Edge Firewall both in routed mode with dynamic routing enabled

For information on the default configuration profiles in the DCNM DFA package, see Appendix A, “Pre-Packaged Config Profiles”.

For information on how to use the REST APIs for auto configuration, see Appendix B, “Auto Config Examples”.

REVIEW DRAFT—CISCO CONFIDENTIAL

DHCP

DHCP REST APIs

The DHCP REST APIs can be used by an external application to manage the DHCP settings used with Power On Auto Provisioning of the network devices

The DHCP REST APIs are mentioned in Table 3-1 on page 3-1

Table 3-1 DHCP REST APIs

API Function	**HTTP Method**	**Resource at URL** https://dcnm-ip/rest/
List POAP scopes	GET	`/poap/dhcp/scopes`
Create a POAP scope	POST	`/poap/dhcp/scopes`
Update a POAP scope	PUT	`/poap/dhcp/scopes/{scope-name}`
Delete a POAP scope	DELETE	`/poap/dhcp/scopes/{scope-name}`

REVIEW DRAFT—CISCO CONFIDENTIAL

Power On Auto Provisioning

Overview

POAP (Power On Auto Provisioning) automates the process of upgrading software images and installing configuration files on Cisco Nexus switches that are being deployed in the network.

When a Cisco Nexus switch with the POAP feature boots and does not find the startup configuration, the switch enters POAP mode, locates the DCNM DHCP server and bootstraps itself with its interface IP address, gateway, and DCNM DNS server IP addresses. It also obtains the IP address of the DCNM server to download the configuration script that is run on the switch to download and install the appropriate software image and device configuration file.

The DCNM 7.0 release will support the web services API for external applications to manage POAP, as an alternate means but similar to how the same API set is used internally by the DCNM GUI for the same functionality. Therefore customers can choose to externally manage POAP programmatically via the DCNM web services APIs, or directly via the DCNM GUI.

Note You will need to install DCNM before running these APIs.

DHCP Scope Creation

DHCP scope is a well-defined term in DHCP arena. It is used to define a policy for giving out IP addresses and other options to hosts on a specific IP subnet. Here, we use DCHP scope for the POAP function to distribute IPv4 address, default gateway, DNS sever IP address, PYTHON bootscript, TFTP server IP address (or other supported protocol + access credential + server, e.g. http://<dcnm-server-ip>/scripts) which stores the bootscript.

By default, a DHCP scope for the management vlan facing interface (eth1) will be created. For DFA use, it is only required to edit the management vlan facing scope with the corresponding IP address range for the devices management addresses.

Add the boot, startup, image server information

This feature allows the user to specify the servers & credentials used to access the device images and the uploaded or DCNM generated/published device configuration. The server serving the images could be different from the one serving the configurations. For the case that the same server is serving both images and configurations, the user is required to provide the server IP address and credentials twice for each

REVIEW DRAFT—CISCO CONFIDENTIAL

server because the root directory holding the images or configuration files could be different. By default, the DCNM server will be the default image and configuration server. There will be two DCNM server addresses, one for config, one for image.

POAP Definition

A POAP device requires the following elements to make it work:

- A device configuration (startup config), which is either provided by the user or generated by DCNM through template instantiation
- Device system and kick start images
- Device Recipe (device deployment plan) which contains
  - Image information -- the location (server & directory) and name of the device images to use
  - Troubleshooting policy – enables debugging and set debugging level, turn-on/off remote logging. It is assumed that the device log will be uploaded to DCNM server.
  - Extra misc. CLI commands to be executed before the device reboot
- Server List file (dcnm-server-list.cfg) – defines the list of servers and their access-credential and path to upload/download files or images.
- DHCP Scope setting – specifies the IPv4 address allocated to a switch device temporarily during the POAP process, what bootscript to use and which TFTP server stores it
- A bootup script (poap-dcnm.py), which is referenced in the DCHP scope, stored in a TFTP server(by default on DCNM server at /var/lib/tftpboot) and loaded by the device. It will download further information (device recipe) or entity (device images, startup config) to complete the POAP process.

POAP Templates

DCNM 7.0 comes pre-packaged with 3 default templates for DFA:

- **The Leaf Template** – used for switch devices with interfaces facing the hosts
- **The Spine Template** –used for switch devices serving as spine switches
- **The Border Leaf Template** – used for switch devices with interfaces facing the DC Interconnet

POAP REST APIs

The POAP REST APIs can be used by an external application to manage the servers and files used for Power On Auto Provisioning of the network devices

The POAP REST APIs are mentioned in Table 4-1 on page 4-2

Table 4-1 POAP REST APIs

API Function	**HTTP Method**	**Resource at URL** https://dcnm-ip/rest/
List POAP settings	GET	`/poap/settings`
Create new POAP settings	POST	`/poap/settings`
Get POAP settings	GET	`/poap/settings`
Update POAP settings	PUT	`/poap/settings/settings-name`

REVIEW DRAFT—CISCO CONFIDENTIAL

API Function	HTTP Method	Resource at URL https://dcnm-ip/rest/
Delete POAP settings	DELETE	/poap/settings/settings-name
List servers	GET	/poap/servers
Create a server	POST	/poap/servers
Get servers	GET	/poap/servers/server-name
Update servers	PUT	/poap/servers/server-name
Delete Servers	DELETE	/poap/servers/server-name
List switch definitions	GET	/poap/switch-definitions
Create switch definitions	POST	/poap/switch-definitions
Get a switch definition	GET	/poap/switch-definitions/{serial-number}
Update a switch definition	PUT	/poap/switch-definitions/switch-id
Delete a switch definition	DELETE	/poap/switch-definitions/switch-id
Publish a switch definition	POST	/poap/published-switch-definitions/{serial-number}

For information on the default configuration profiles in the DCNM DFA package, see Appendix E, “POAP Examples”

REVIEW DRAFT—CISCO CONFIDENTIAL

CHAPTER 5

Cable Plan

Cable Plan REST APIs

The Cable Plan REST APIs can be used by an external application to programmatically manage the cable plan according to which the network device cabling is verified

The cable plan REST APIs are mentioned in Table 5-1 on page 5-1.

Table 5-1 Cable Plan REST APIs

API Function	**HTTP Method**	**Resource at URL** https://dcnm-ip/rest/
Capture a cable plan	GET	`/cable-plans/discovery`
Generate a cable plan	GET	`/cable-plans/poap`
Save a cable plan	POST	`/cable-plans`
Delete a cable plan	DELETE	`/cable-plans`
Get a cable plan	GET	`/cable-plans`
Import a cable plan	POST	`/cable-plans/import`
Export a cable plan	GET	`/cable-plans/xml`
Deploy a cable plan	POST	`/cable-plans/fabric`
Revoke a cable plan	POST	`/cable-plans/revoke`

Table 5-2

REVIEW DRAFT—CISCO CONFIDENTIAL

Pre-Packaged Config Profiles

Introduction

There are two types of Config Profiles pre-packaged and available by default in DCNM:

- Partition Config-Profile
- Network Config-Profile

Pre-Packaged Partition Config-Profile Details

This section lists the contents of the partition config-profiles that will be part of the DCNM profiles database that ships with the DFA solution. Note that new parameterized profiles can be added on top of this existing set. In addition, the existing profiles may be modified so that they can be tailored to the specific data center needs (with some caution).

```
configure profile vrf-common
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv4 unicast
     route-target both auto
    address-family ipv6 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
      maximum-paths ibgp 2
     address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       maximum-paths ibgp 2
end
configure profile vrf-common-services
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
 vrf context $vrfName
  ip route 0.0.0.0/0 $include_serviceIpAddress pref 1
  vni $include_l3_segid
  rd auto
  address-family ipv4 unicast
   route-target both auto
  address-family ipv6 unicast
   route-target both auto
  router bgp $asn
   vrf $vrfName
    address-family ipv4 unicast
     redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
     maximum-paths ibgp 2
   address-family ipv6 unicast
     redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
     maximum-paths ibgp 2
end
```

Pre-Packaged Network Config-Profile Details

This section lists the contents of the network config-profiles that will be part of the DCNM profiles database which ships with the DFA solution.

Caution The new parameterized profiles can be added on top of this existing set. In addition, the existing profiles may be modified so that they can be tailored to the specific data center needs.

```
config profile defaultNetworkIpv4EfProfile
  vlan $vlanId
    vn-segment $segmentId
    mode fabricpath
  interface vlan $vlanId
    vrf member $vrfName
    ip address $gatewayIpAddress/$netMaskLength
    ip dhcp relay address $dhcpServerAddr use-vrf default
    fabric forwarding mode proxy-gateway
    no ip redirects
    no shutdown
  include profile vrf-common
end

config profile defaultNetworkIpv4TfProfile
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
ip dhcp relay address $dhcpServerAddr use-vrf default
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  include profile vrf-common
end

config profile defaultNetworkIpv6EfProfile
  interface vlan $vlanId
   vrf member $vrfName
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no ipv6 redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  include profile vrf-common
end

config profile defaultNetworkIpv6TfProfile
  interface vlan $vlanId
   vrf member $vrfName
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  include profile vrf-common
end

config profile defaultNetworkIpv4v6EfProfile
  interface vlan $vlanId
   vrf member $vrfName
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no ip redirects
   no ipv6 redirects
  no shutdown
 vlan $vlanId
  vn-segment $segmentId
  mode fabricpath
 include profile vrf-common
end

config profile defaultNetworkIpv4v6TfProfile
 interface vlan $vlanId
  vrf member $vrfName
  ip address $gatewayIpAddress/$netMaskLength
  ipv6 address $gatewayIpv6Address/$prefixLength
  fabric forwarding mode anycast-gateway
  no shutdown
 vlan $vlanId
  vn-segment $segmentId
  mode fabricpath
 include profile vrf-common
end

config profile defaultNetworkIpv4EfEdgeServiceProfile
 interface vlan $vlanId
  vrf member $vrfName
  ip address $gatewayIpAddress/$netMaskLength
#  ip dhcp relay address $dhcpServerAddr use-vrf default
  fabric forwarding mode proxy-gateway
  no ip redirects
  no shutdown
 vlan $vlanId
  vn-segment $segmentId
  mode fabricpath
 include profile vrf-common-services
end

config profile defaultNetworkIpv4TfEdgeServiceProfile
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
#  ip dhcp relay address $dhcpServerAddr use-vrf default
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  include profile vrf-common-services
end

config profile defaultNetworkIpv4v6EfServiceProfile
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no ip redirects
   no ipv6 redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  include profile vrf-common-services
end

config profile defaultNetworkIpv4v6TfServiceProfile
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
  include profile vrf-common-services
end
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
config profile defaultNetworkL2Profile
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end

config profile defaultNetworkdefaultPartitionIpv4EfProfile
  interface vlan $vlanId
   ip address $gatewayIpAddress/$netMaskLength
#  ip dhcp relay address $dhcpServerAddr
   fabric forwarding mode proxy-gateway
   no ip redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end

config profile defaultNetworkdefaultPartitionIpv4TfProfile
  interface vlan $vlanId
   ip address $gatewayIpAddress/$netMaskLength
#  ip dhcp relay address $dhcpServerAddr
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end

config profile defaultNetworkdefaultPartitionIpv6EfProfile
  interface vlan $vlanId
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no ipv6 redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
config profile defaultNetworkdefaultPartitionIpv6TfProfile
  interface vlan $vlanId
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end

config profile defaultNetworkdefaultPartitionIpv4v6EfProfile
  interface vlan $vlanId
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no ip redirects
   no ipv6 redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end

config profile defaultNetworkdefaultPartitionIpv4v6TfProfile
  interface vlan $vlanId
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode anycast-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
   mode fabricpath
end
```

REVIEW DRAFT—CISCO CONFIDENTIAL

Auto Config Examples

How to Use the DFA Rest APIs to Control Auto-Configuration

This example shows how to use DCNM REST APIs for auto-configuration.

Note While reviewing the actual API calls, a section is available to provide the sample.

Steps to Create a Network

Step 1 Logon to get a token

Step 2 Create a Profile

Step 3 Create an Organization

Step 4 Create a network under the partition.

Step 5 Logout

Example to call Logon

```
URL: https://10.77.247.111/rest/logon
HTTP Method: POST
Resource Name: /rest/logon
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Authorization: Basic YWRtaW46YWRtaW4xMjM=
Content-Length: 28
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
{"expirationTime": 1000000}
HTTP/1.1
Status Code: 200 OK
Content-Type: application/json
{
    "token" : " NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ "
}
```

Example to create a profile

```
URL: http://10.77.247.111/rest/auto-config/profiles
HTTP Method: POST
Resource Name: /rest/auto-config/profiles
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{
      "profileName":" GoldProfile",
         "description":"Gold profile",
         "vlanId":"$vlanId",
         "segmentId":"$segmentId",
        "interfaceVlanId":"$vlanId",
        "gatewayIpAddress":"$gatewayIpAddress",
        "netMaskLength":"$netMaskLength",
        "forwardingMode":"$forwardingMode",
        "ipDhcp":"$ipDhcp",
"configCommands":" vlan $vlanId  \nvn-segment $segmentId\ninterface Vlan
$vlanId\nip address     $gatewayIpAddress/$netMaskLength\nfabric
forwarding mode $forwardingMode\nip dhcp relay address $ipDhcp \nno
shutdown "
}
Status Code: 202 Accepted
Content-Type: application/json
{
           "tprofileName" : "GoldProfile"
}
```

Example to create an organization

```
URL: http://10.77.247.111/rest/auto-config/organizations
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
HTTP Method: POST
Resource Name: /rest/auto-config/organizations
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{
  "organizationName" : "organization1",
  "description" : "organization1",
  "profileName" : "GoldProfile",
  "forwardingMode" : "anycast-gateway",
  "configArg" : [ "$netMaskLength=24",
"$gatewayIpAddress=10.10.10.1","$partitionName=partition1","
$forwardingMode=anycast-gateway"," $networkName=Net-Dev-101" ]
  "orchestrationSource" : "vCloud Director",
}
Status Code: 202 Accepted
Content-Type: application/json
{
           "organizationName" : "organization1"
}
```

Example to list organizations

```
URL: http://10.77.247.111/rest/auto-config/organizations
HTTP Method: GET
Resource Name: /rest/auto-config/organizations?detail=true
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
Cache-Control: no-cache
Accept: application/json
Status Code: 200 OK
Content-Type: application/json
{
  "organizations" : [ {
"organizationName" : "organization1",
 "description" : "organization1",
 "profileName" : "GoldProfile",
 "forwardingMode" : "anycast-gateway",
 "configArg" : [ "$netMaskLength=24",
"$gatewayIpAddress=10.10.10.1",”$partitionName=partition1”,”
$forwardingMode=anycast-gateway”,” $networkName=Net-Dev-101” ]
  "orchestrationSource" : "vCloud Director",
  }, {
 "organizationName" : "organization2",
 "description" : "organization2",
 "profileName" : "SilverProfile",
 "forwardingMode" : "anycast-gateway",
"configArg" : [ "$netMaskLength=24",
"$gatewayIpAddress=20.20.20.1",”$partitionName=partition2”,
”$forwardingMode=anycast-gateway”, “$networkName=Net-Dev-102” ],
  "orchestrationSource" : "vCloud Director"
  } ]
}
```

Example to create a Partition

```
URL:http://10.77.247.111/rest/auto-config/organizations/organization1/pa
rtitions
HTTP Method: POST
Resource Name: /rest/auto-config/organizations/organization1/partitions
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{
  "partitionName" : "partition1",
  "description" : "partition1",
  "profileName" : "GoldProfile",
  "forwardingMode" : "anycast-gateway",
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
"configArg" : [ "$netMaskLength=24",
"$gatewayIpAddress=10.10.10.1","$partitionName=partition1","
$forwardingMode=anycast-gateway"," $networkName=Net-Dev-101" ],
  "vpnId" : "100",
  "organizationName" : "organization1",
}
Status Code: 202 Accepted
Content-Type: application/json
{
            "partitionName" : "partition1"
       }
```

Example to create a network

```
URL: http://10.77.247.111/rest/auto-config/organizations/
organization1/partitions/partition1/networks
HTTP Method: POST
Resource Name:
/rest/auto-config/organizations/organization1/partitions/partition1/netw
orks
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{
  "segmentId" : "100001",
  "description" : "segment1",
  "profileName" : "defaultNetworkIpv4EfProfile",
  "forwardingMode" : "anycast-gateway",
"configArg" : [ "$netMaskLength=24",
"$gatewayIpAddress=10.10.10.1","$partitionName=partition1","
$forwardingMode=anycast-gateway"," $networkName=Net-Dev-101" ],
  "hostMacAddress" : "08:00:69:02:01:10",
  "partitionName" : "partition1",
}
Status Code: 202 Accepted
Content-Type: application/json
{
            "segmentId" : "100001"
}
```

REVIEW DRAFT—CISCO CONFIDENTIAL

Example to Logout

URL: http://10.77.247.111/rest/logout
HTTP Method: POST
Resource Name: /rest/logout
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
Status Code: 202 Accepted

DFA REST APIs for OpenStack

Introduction

This section explains how the DFA REST APIs were used to integrate OpenStack with DFA. It's not very different from how any application can be integrated with DFA using the DFA REST APIs.

The DFA plugins that are patched into the OpenStack, implement this documented interaction between OpenStack and DFA. The following steps explain how the Project, Network and VM creations in Openstack are communicated to DFA.

Step 1 Logon to the REST API to get a security token

This step is not seen by the Project user.

Step 2 Create a Project

This is the organization and partition in DFA schema terminology.

Step 3 Get the network config-profile list for a given Project.

This step is not seen by the Project user but these profiles will become available to the project user from a drop-down menu in Horizon during network creation.

Step 4 Create a network for that Project specifying one of the available network config-profiles

Step 5 Get the network config-profile contents

This step is not see by the Project user.

Step 6 Launch the Virtual Machine.

The network config-profile contents will be sent to the OVS where the VM is attached.

Step 7 Logout

Step 1: Logon to get a token

```
https://10.77.247.111/rest/logon

POST /rest/logon

Host: 10.77.247.111

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3 Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

REVIEW DRAFT—CISCO CONFIDENTIAL

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Keep-Alive: 115

Connection: keep-alive

Content-Type: application/json; charset=UTF-8

Authorization: Basic YWRtaW46YWRtaW4xMjM=

Content-Length: 28

Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328

Pragma: no-cache

Cache-Control: no-cache

{"expirationTime": 1000000}

HTTP/1.1 200 OK

Content-Type: application/json

{ "token" : " NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ " }

Step 2: Create Project (organization and partition)

The OpenStack Project name will be mapped to both Organization and Partition in DCNM.

Create an organization

http://10.77.247.111/rest/auto-config/organizations

POST /rest/auto-config/organizations

Host: 10.77.247.111

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.3

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Keep-Alive: 115

Connection: keep-alive

Content-Type: application/json; charset=UTF-8

Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ

Content-Length: 179

Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328

Pragma: no-cache

Cache-Control: no-cache

REVIEW DRAFT—CISCO CONFIDENTIAL

```
{
 "organizationName" : "organization1",
  "description" : "organization1",
  "orchestrationSource" : "OpenStack Controller 5",
}
202 Accepted
Content-Type: application/json
{
            "organizationName" : "organization1"
       }
```

Create a Partition

```
http://10.77.247.111/rest/auto-config/organizations/organization1/partitions
POST /rest/auto-config/organizations/organization1/partitions
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3) Gecko/20100401 Firefox/3.6.3
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{
  "partitionName" : "organization1",
  "description" : "organization1",
  "organizationName" : "organization1",
}
202 Accepted
Content-Type: application/json
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
{
            "partitionName" : "organization1"
       }
```

Step 3: Get Network Profile list

Using this REST API, OpenStack gets a list of network config-profiles and display them on a Horizon dashboard pull-down menu for the project user to select one when creating a network.

```
http://10.77.247.111/rest/auto-config/profiles
POST /rest/auto-config/profiles
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
202 Accepted
Content-Type: application/json
{
"profileName" : "defaultNetworkIpv4EfEdgeServiceProfile"
"profileName" : "defaultNetworkIPv4TfEdgeServiceProfile"
"profileName" : "defaultNetworkL2Profile"
       }
```

Step 4: Create a network for that Tenant

```
POST
/rest/auto-config/organizations/organization1/partitions/organization
1/networks
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
http://10.77.247.111/rest/auto-config/organizations/
organization1/partitions/organization1/networks

POST
/rest/auto-config/organizations/organization1/partitions/organization
1/networks

Host: 10.77.247.111

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3

Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Keep-Alive: 115

Connection: keep-alive

Content-Type: application/json; charset=UTF-8

Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ

Content-Length: 179

Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328

Pragma: no-cache

Cache-Control: no-cache

{
  "NetworkName" : "Network1",
  "segmentId" : "100001",
  "description" : "One of the networks in Organization1",
  "profileName" : "defaultNetworkIpv4EfEdgeServiceProfile",
 "configArg" : [ "$netMaskLength=24",
"$gatewayIpAddress=10.10.10.1" ],
  "partitionName" : "organization1"
}

202 Accepted

Content-Type: application/json

{
           "segmentId" : "100001"
      }
```

Step 5: Get the profile info

```
http://10.77.247.111/rest/auto-config/profiles/
defaultNetworkIpv4EfEdgeServiceProfile
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
GET /rest/auto-config/profiles/ POST /rest/auto-config/profiles/
defaultNetworkIpv4EfEdgeServiceProfile

Host: 10.77.247.111

User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3

Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7


Keep-Alive: 115

Connection: keep-alive

Content-Type: application/json; charset=UTF-8

Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ

Content-Length: 179

Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328

Pragma: no-cache

202 Accepted

Content-Type: application/json

{

   "profileName" : "defaultNetworkIpv4EfEdgeServiceProfile",

"configCommands":"vlan $vlanId\rvn-segment $segmentId\rinterface vlan
$vlanId\rvrf member $vrfName\rip address
$gatewayIpAddress/$netMaskLength\rno ip redirect\rfabric forwarding
mode proxy-gateway\rip dhcp relay address 172.23.244.28 use-vrf
default\rno shutdown\rinclude profile vrf-common",

"params":["$vlanId","$segmentId","$vrfName","$gatewayIpAddress","$net
MaskLength"]

}
```

Step 6: Launch a VM

The VM launch is communicated to the OVS and LLDPad of the physical host selected by OpenStack. The VM's information along with the associated segment ID is used for establishing a VDP session between the LLDPad and DFA leaf switch and the subsequently acquired vlan ID from VDP is used to program data flows in OVS.

Step 7: log out

```
POST /rest/logout
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
http://10.77.247.111/rest/logout
POST /rest/logout
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
202 Accepted
```

REVIEW DRAFT—CISCO CONFIDENTIAL

Pre-Packaged POAP Template

Introduction

There are three POAP network device templates pre-packaged and available by default in DCNM:

- Default Leaf Template
- Default Spine Template
- Default Border Leaf Template

Default Leaf Template

```
##template properties
name = Base_Leaf_Template;
description = This file specifies the template configuration for leaf
switch;
userDefined = false;
supportedPlatforms = N7K, N6K;
templateType = POAP;
published = true;
##
##template variables
@(IsSwitchName=true, UseDNSReverseLookup=true, IsMandatory=true,
Description="The host name of the switch")
string SWITCH_NAME;


@(IsManagementIP=true, IsMandatory=true, IsVPCPeerLinkSrc=true,
Description="Management IP address used by DCNM to monitor this
device")
ipV4Address MGMT_IP;


@(IsMandatory=true, Description="Management Prefix")
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
integer MGMT_PREFIX;

@(IsMandatory=true, Description="Default Gateway IP address")
ipV4Address DEFAULT_GATEWAY;

@(IsMandatory=true, Description="Plain text or 5 encrypted")
string ADMIN_PASSWORD;

@(IsMandatory=true, IsSwitchRole=true, Description="The role of the
switch. e.g. leaf, spine")
string SWITCH_ROLE {
  defaultValue=leaf;
};

@(IsMandatory=true, IsFabricPort=true, Description="The comma and dash
separated list of fabric ports")
interfaceRange FABRIC_INTERFACES;

@(IsMandatory=true, IsHostPort=true, Description="The comma and dash
separated list of host ports")
interfaceRange HOST_INTERFACES;

@(IsMandatory=true, Description="Backbone VLAN ID")
integer BACKBONE_VLAN;

@(IsMandatory=true, Description="Backbone IP address/prefix")
string BACKBONE_IP;

@(IsMandatory=true, Description="Backbone IPv6 address/prefix")
string BACKBONE_IPV6;

@(IsMandatory=true)
string BGP_ROUTER_IP;

@(IsMandatory=true)
string BGP_RR_IP;

@(IsMandatory=true, Description="IP Address of the Auto-config LDAP
Server")
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
ipV4Address LDAP_SERVER_IP;

@(IsMandatory=true, Description="IP Address of the XMPP Server")
ipV4Address XMPP_SERVER_IP;

@(IsMandatory=true, Description="FQDN of the XMPP Server")
string XMPP_SERVER;

@(IsMandatory=true, Description="Space separated XMPP Spine Group
Names")
string XMPP_GROUPS;

@(IsMandatory=true, Description="Password")
string XMPP_PASSWORD;

@(IsMandatory=true, Description="True if VPC should be configured")
boolean ENABLE_VPC;

@(IsVPCDomainID=true)
integer VPC_DOMAIN_ID;

@(IsVPCPeerLinkDst=true)
ipV4Address VPC_PEER_DST;

@(IsVPCPeerLinkPortChannel=true, IsVPCPort=true)
integer VPC_PEER_LINK_PORT_CHANNEL_NUMBER;

@(IsVPCPeerLinkPort=true)
interfaceRange VPC_PEER_LINK_IF_NAMES;

@(IsVPC=true)
struct VPC {
  @(IsVPCID=true,IsVPCPortChannel=true)
  integer ID;
  @(IsVPCPort=true)
  interface IF_NAME;
} VPC_ARRAY[];
##
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
##template content
license grace-period
hostname $$SWITCH_NAME$$

install feature-set fabric
feature-set fabric
feature fabric forwarding

install feature-set fabricpath
feature-set fabricpath

!#Cable Management
feature lldp
feature cable-management
fabric connectivity tier 1

feature bgp
feature interface-vlan
feature vn-segment-vlan-based
feature dhcp
feature evb
feature pim
if( $$ENABLE_VPC$$ == "true") {
feature vpc
}

!#feature SPoM
feature fabric access

!### Vinci Multicast Forwarding (NGMVPN)
feature fabric multicast

!### Vinci passive PIM
ip multicast fabric-forwarding

fabric forwarding identifier 1
fabric forwarding anycast-gateway-mac 2020.0000.00AA
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
fabric forwarding switch-role leaf

username admin password $$ADMIN_PASSWORD$$ role network-admin
no password strength-check
ip domain-lookup

!### Configure IP host for SPoM XMPP server below
ip host $$XMPP_SERVER$$ $$XMPP_SERVER_IP$$

!### Configure SPoM XMPP Server below
fabric access server $$XMPP_SERVER$$ vrf management password
$$XMPP_PASSWORD$$

!### Subscribe this device to this XMPP group
fabric access group $$XMPP_GROUPS$$

fabric database profile-map global
  ethernet-tag encapsulation dot1q default dynamic
  ethernet-tag encapsulation vni default dynamic
  vdp vni default dynamic

!### Configure fabric database location
!### db-table "ou=segments,dc=cisco,dc=com"
!### is a variable that should match the table
!### name that is populated in the LDAP database.
fabric database type asset
  server protocol ldap ip $$LDAP_SERVER_IP$$
     db-table ou=segments,dc=cisco,dc=com key-type 1

!### Enable global mobility-domain only when you
!### are going to use (vlan, mobility_domain) to search ADBM database
!### fabric database mobility-domain %mobility_domain

route-map FABRIC-RMAP-REDIST-HOST deny 10
  match interface Vlan $$BACKBONE_VLAN$$
route-map FABRIC-RMAP-REDIST-HOST permit 20
  match ip address HOSTS
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
route-map FABRIC-RMAP-REDIST-V6HOST  deny 10
  match interface Vlan $$BACKBONE_VLAN$$
route-map FABRIC-RMAP-REDIST-V6HOST permit 20
  match ip address V6HOSTS

ip dhcp snooping
service dhcp
ip dhcp relay
ip dhcp relay information option
ip dhcp relay information option vpn

interface Vlan $$BACKBONE_VLAN$$
  no shutdown
  ip address $$BACKBONE_IP$$
  ipv6 address $$BACKBONE_IPV6$$
  fabric forwarding control-segment

foreach FABRIC_INTERFACE in $$FABRIC_INTERFACES$$ {
interface @FABRIC_INTERFACE
  no shutdown
  switchport
  switchport mode fabricpath
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
}

foreach HOST_INTERFACE in $$HOST_INTERFACES$$ {
interface @HOST_INTERFACE
  switchport
  switchport mode trunk
  no shutdown
}

route-map ALL-PATHS permit 10
  set path-selection all advertise

vlan $$BACKBONE_VLAN$$
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  mode fabricpath

router bgp 100
  router-id $$BGP_ROUTER_IP$$
  address-family ipv4 unicast
    redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
    maximum-paths ibgp 2
    additional-paths receive
    additional-paths selection route-map ALL-PATHS
  address-family ipv6 unicast
    redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
    maximum-paths ibgp 2
    additional-paths receive
    additional-paths selection route-map ALL-PATHS
  address-family vpnv4 unicast
    additional-paths receive
  address-family vpnv6 unicast
    additional-paths receive
  address-family ipv4 mvpn
    additional-paths receive
  address-family ipv6 mvpn
    additional-paths receive

  neighbor $$BGP_RR_IP$$ remote-as 100
    address-family ipv4 unicast
      send-community both
    address-family ipv6 unicast
      send-community both
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended
    address-family ipv4 mvpn
      send-community both
    address-family ipv6 mvpn
      send-community both

vrf context management
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  ip route 0.0.0.0/0 $$DEFAULT_GATEWAY$$

interface mgmt0
  vrf member management
  no cdp enable
  ip address $$MGMT_IP$$/$$MGMT_PREFIX$$

system fabric dynamic-vlans 2500-3500

!###l3vm uses bootflash:platform.inf for the core vlans.
!###Please refer to CSCuj12763 for additional details
system fabric core-vlans  2500-2999

line console
  exec-timeout 0
line vty

if( $$ENABLE_VPC$$ == "true") {
vpc domain $$VPC_DOMAIN_ID$$
  peer-keepalive destination $$VPC_PEER_DST$$ source $$MGMT_IP$$ vrf
management
 delay restore 150
 auto-recovery
 fabricpath switch-id 200
 ip arp synchronize
 ipv6 nd synchronize

interface port-channel$$VPC_PEER_LINK_PORT_CHANNEL_NUMBER$$
 description "vpc-peer-link"
 switchport mode fabricpath
 spanning-tree port type network
 vpc peer-link
 fabricpath isis metric 200

foreach VPC_PEER_LINK_IF_NAME in $$VPC_PEER_LINK_IF_NAMES$$ {

interface @VPC_PEER_LINK_IF_NAME
  switchport mode fabricpath
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  channel-group $$VPC_PEER_LINK_PORT_CHANNEL_NUMBER$$
}

foreach VPC in $$VPC_ARRAY$${

interface port-channel@VPC.ID
  switchport mode trunk
  vpc @VPC.ID

interface @VPC.IF_NAME
  switchport mode trunk
  channel-group @VPC.ID
}
}

!### Sample config for setting interface
!### address e.g. for connecting to
!### 101.101.101.91 ldap server

!interface Ethernet 1/18
!  no switchport
!  ip address 101.101.101.22/24
!  no shutdown

configure profile vrf_tenant_profile
   vlan $id
      mode fabricpath
      vn-segment $segment
   interface vlan $id
      vrf member $vrfname
      ip address $$BACKBONE_IP$$
      ipv6 address $$BACKBONE_IPV6$$
      no shut
exit

configure profile vrf-common
   vrf context $vrfName
    vni $include_l3_segid
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
    rd auto
    address-family ipv4 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       maximum-paths ibgp 2
exit

config profile GoldProfile
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   ip dhcp relay address $dhcpServerAddr use-vrf default
   fabric forwarding mode proxy-gateway
   no ip redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
  include profile vrf-common
end

configure profile vrf-common-v6
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv6 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
       maximum-paths ibgp 2
exit

config profile GoldProfile-v6
  interface vlan $vlanId
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
   vrf member $vrfName
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
  include profile vrf-common-v6
end

configure profile vrf-common-v4nv6
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv4 unicast
     route-target both auto
    address-family ipv6 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       maximum-paths ibgp 2
      address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
       maximum-paths ibgp 2
exit

config profile GoldProfile-v4nv6
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   ip dhcp relay address $dhcpServerAddr use-vrf default
   fabric forwarding mode proxy-gateway
   no ip redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  include profile vrf-common-v4nv6
end
##
```

Default Spine Template

```
##template properties
name = Base_Spine_Template;
description = This file specifies the template configuration for spine
switch;
userDefined = false;
supportedPlatforms = N7K, N6K;
templateType = POAP;
published = true;
##

##template variables
@(IsSwitchName=true, UseDNSReverseLookup=true, IsMandatory=true,
Description="The host name of the switch")
string SWITCH_NAME;

@(IsManagementIP=true, IsMandatory=true, Description="Management IP
address used by DCNM to monitor this device")
ipV4Address MGMT_IP;

@(IsMandatory=true, Description="Management Prefix")
integer MGMT_PREFIX;

@(IsMandatory=true, Description="Default Gateway IP address")
ipV4Address DEFAULT_GATEWAY;

@(IsMandatory=true, Description="Plain text or 5 encrypted")
string ADMIN_PASSWORD;

@(IsMandatory=true, IsSwitchRole=true, Description="The role of the
switch. e.g. leaf, spine")
string SWITCH_ROLE {
  defaultValue=spine;
};
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
@(IsMandatory=true, IsFabricPort=true, Description="The comma and dash
separated list of fabric ports")
interfaceRange FABRIC_INTERFACES;

@(IsMandatory=true, Description="Backbone VLAN ID")
integer BACKBONE_VLAN;

@(IsMandatory=true, Description="Backbone IP address")
ipV4Address BACKBONE_IP;

@(IsMandatory=true, Description="Backbone Prefix")
integer BACKBONE_PREFIX;

@(IsMandatory=true, Description="IP Address of the XMPP Server")
ipV4Address XMPP_SERVER_IP;

@(IsMandatory=true, Description="FQDN of the XMPP Server")
string XMPP_SERVER;

@(IsMandatory=true, Description="Space separated XMPP Spine Group
Names")
string XMPP_GROUPS;

@(IsMandatory=true)
string XMPP_PASSWORD;
##

##template content
license grace-period
hostname $$SWITCH_NAME$$

!### Traditional Forwarding
install feature-set fabricpath
feature-set fabricpath
feature interface-vlan

!### Enhanced Forwarding
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
install feature-set fabric
feature-set fabric
feature fabric forwarding

!### Cable Management
feature lldp
feature cable-management
fabric connectivity tier 2

feature interface-vlan

!#feature SPoM
feature fabric access

fabric forwarding identifier 1
fabric forwarding switch-role spine

username admin password $$ADMIN_PASSWORD$$ role network-admin
no password strength-check
ip domain-lookup
!### Configure IP host for SPoM XMPP server below
ip host $$XMPP_SERVER$$ $$XMPP_SERVER_IP$$

!### Configure SPoM XMPP Server below
fabric access server $$XMPP_SERVER$$ vrf management password
$$XMPP_PASSWORD$$

!### Subscribe this device to this XMPP group
fabric access group $$XMPP_GROUPS$$

interface Vlan $$BACKBONE_VLAN$$
  no shutdown
  ip address $$BACKBONE_IP$$/$$BACKBONE_PREFIX$$
  fabric forwarding control-segment

foreach FABRIC_INTERFACE in $$FABRIC_INTERFACES$$ {
interface @FABRIC_INTERFACE
  switchport
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  switchport mode fabricpath
  no shutdown
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
}

vlan $$BACKBONE_VLAN$$
  mode fabricpath

vrf context management
  ip route 0.0.0.0/0 $$DEFAULT_GATEWAY$$

interface mgmt0
  vrf member management
  no cdp enable
  ip address $$MGMT_IP$$/$$MGMT_PREFIX$$

line console
  exec-timeout 0
line vty
##
```

REVIEW DRAFT—CISCO CONFIDENTIAL

POAP Examples

How to use the DFA REST APIs to Control POAP

You can use any programming language that supports Web services to create a Web services client that invokes the DCNM POAP Web services API. This section just lists out the example HTTP request and response for reference.

The examples will show the following steps to configure POAP:

Step 1 Call logon to get a Dcnm-Token

Step 2 Update a POAP scope

Step 3 Create a config/image server

Step 4 Get a group

Step 5 List POAP templates

Step 6 Generate config with a template

Step 7 Create POAP definition

Step 8 Logout

Secure Logon

The logon API takes “username:password” with Basic base64 encoded in HTTP Authorization header, like “Authorization: Basic QWxhZGRpbjpvcGVuIHNlc2FtZQ==”. It will return the Token in the payload, and this token will be used as Dcnm-Token in the subsequent API.

```
http://10.77.247.111/rest/logon
POST /rest/logon
```

Request

```
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Authorization: Basic YWRtaW46YWRtaW4xMjM=
Content-Length: 28
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{"expirationTime": 1000000}
```

Response

```
HTTP/1.1 200 OK
Content-Type: application/json
{
    "Token" : " NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ "
}
```

Update a POAP scope

```
http://10.77.247.111/rest/poap/dhcp/scopes/scope1
PUT /rest/poap/dhcp/scopes/scope1
```

Request

```
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
Content-Length: 28
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
{
"scopeName":"scope1",
"subnet":"4.4.4.0/24",
"ipRange":"4.4.4.6-4.4.4.8",
"maxLeaseTime":"3600",
"bootFilename":"poap_dcnm.py",
"domainNameServers":"2.2.2.2",
"routers":"2.2.2.2",
"tftpServerName":"100.100.100.140"
}
3.2.2 Response
HTTP/1.1 202 Accepted
```

Create a config/image server

```
http://10.77.247.111/rest/poap/servers
POST /rest/poap/servers
```

Request

```
Host: 10.77.247.111
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us,en;q=0.5
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 115
Connection: keep-alive
Content-Type: application/json; charset=UTF-8
Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ
Content-Length: 179
Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328
Pragma: no-cache
Cache-Control: no-cache
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
{
"id":0,
"serverName":"TEST_SERVER",
"url":"scp://10.77.247.151/var/lib/tftpboot/dcnm",
"userName":"root",
"password":"admin",
"lastUpdateTime":1379358179262
}
```

Response

```
202 Accepted
Content-Type: application/json
{
"id":0,
"serverName":"TEST_SERVER",
"lastUpdateTime":0}
  }
```

Get a group

This is a SOAP API call, will need to get the "memDbId" for the group which POAP devices belong to. In Create POAP Definition, page E-48, this "memDbId" will be used as "lanGroup" in the request.

```
http:// 10.77.247.111:80/DbAdminWSService/DbAdminWS
3.4.1 Request:
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <ns0:token
xmlns:ns0="http://dcnm/headers">NUVyu5Y0YHB5PfWoiQqNY64FVVw6yv2l</ns0
:token>
    <ns0:session xmlns:ns0="http://dcnm/headers">0</ns0:session>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tns:getGroupNavigation
xmlns:tns="http://ep.san.jaxws.dcbu.cisco.com/"/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

Response

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
   <env:Header/>
   <env:Body>
      <tns:getGroupNavigationResponse
xmlns:tns="http://ep.san.jaxws.dcbu.cisco.com/">
         <result><![CDATA[<groupList name='Data Center'>
<group isBranch="true"  name="Default_LAN" state ="unchecked"
selectable= "true" memDbId="2" type="1" ></group>
<group isBranch="true"  name="Default_SAN" state ="unchecked"
selectable= "false" memDbId="1" type="2" >
</group></groupList>]]></result>
      </tns:getGroupNavigationResponse>
   </env:Body>
</env:Envelope>
```

List POAP templates

This is a SOAP API call to get the template, in this example, it will get the Base_Leaf_Template that will later be used to generate the configuration.

```
http://10.77.247.139:80/ConfigTemplateWSService/ConfigTemplateWS
```

Request

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <ns0:token
xmlns:ns0="http://dcnm/headers">NUVyu5Y0YHB5PfWoiQqNY64FVVw6yv2l</ns0
:token>
    <ns0:session xmlns:ns0="http://dcnm/headers">0</ns0:session>
  </SOAP-ENV:Header>
  <SOAP-ENV:Body>
    <tns:getAllPublishedTemplates
xmlns:tns="http://ep.san.jaxws.dcbu.cisco.com/"/>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
   <env:Header/>
   <env:Body>
      <tns:getAllPublishedTemplatesResponse
xmlns:tns="http://ep.san.jaxws.dcbu.cisco.com/">
         <result>
            <item>
               <assignedInstanceClassId>0</assignedInstanceClassId>
               <instanceClassId>27</instanceClassId>

<instanceName>com.cisco.dcbu.dcm.model.cfgtemplate.ConfigTemplate:nam
e=Leaf_Template:type=false</instanceName>
            <description>This file specifies the template configuration
for leaf switch with annotations</description>
               <fileName>leaf_annotations.template</fileName>
               <name>Leaf_Template</name>
               <parameters>
                  <annotations>
         <entry>
                        <key>Description</key>
                        <value>"The host name of the switch"</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>UseDNSReverseLookup</key>
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>IsSwitchName</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>SWITCH_NAME</name>
                  <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                <parameters>
                   <annotations>
                      <entry>
                         <key>Description</key>
                         <value>"Vlan interfaces allocated to this VDC
or device"</value>
                      </entry>
                      <entry>
                         <key>IsMandatory</key>
                         <value>true</value>
                      </entry>
                   </annotations>
                   <name>ALLOC_INTERFACES</name>
                   <parameterType>string[]</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                   <structureParameters/>
                </parameters>
                <parameters>
                   <annotations>
                      <entry>
                         <key>Description</key>
                         <value>"DNS name of the XMPP server or XMPP
domain name"</value>
                      </entry>
                      <entry>
                         <key>IsMandatory</key>
                         <value>true</value>
                      </entry>
                   </annotations>
                   <name>DNS_HOST_NAME</name>
                   <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                   <structureParameters/>
                </parameters>
                <parameters>
                   <annotations>
                      <entry>
                         <key>Description</key>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <value>"IP Address for the XMPP DNS name or
domain name"</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>DNS_HOST_IP</name>
                  <parameterType>ipV4Address</parameterType>
                <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
                        <value>"Management IP address used by DCNM to
monitor this device"</value>
                     </entry>
                     <entry>
                        <key>IsManagementIP</key>
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>MGMT_IP</name>
                  <parameterType>ipV4Address</parameterType>
                <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <value>"The prefix in integer format. e.g.
24"</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>MGMT_PREFIX</name>
                  <parameterType>integer</parameterType>
               <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
                        <value>"Default Gateway IP address"</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>DEFAULT_GATEWAY</name>
                  <parameterType>ipV4Address</parameterType>
               <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
                        <value>"VLAN facing the server or VMs"</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        </entry>
                     </annotations>
                     <name>CUSTOMER_FACING_VLAN_ID</name>
                     <parameterType>integer</parameterType>
                   <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                    <value>"IP address facing the server or VMs"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>CUSTOMER_FACING_IP</name>
                     <parameterType>ipV4Address</parameterType>
                   <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                      <value>"IP subnet facing the server or VMs. The
prefix in integer format. e.g. 24"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>CUSTOMER_FACING_PREFIX</name>
                     <parameterType>integer</parameterType>
                   <parameterTypeStructure>false</parameterTypeStructure>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                    <structureParameters/>
                </parameters>
                <parameters>
                    <annotations>
                        <entry>
                            <key>IsMandatory</key>
                            <value>true</value>
                        </entry>
                    </annotations>
                    <name>FABRIC_FORWARDING_ID</name>
                    <parameterType>integer</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
                </parameters>
                <parameters>
                    <annotations>
                        <entry>
                            <key>IsMandatory</key>
                            <value>true</value>
                        </entry>
                    </annotations>
                    <name>ANYCAST_GATEWAY_MAC</name>
                    <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
                </parameters>
                <parameters>
                    <annotations>
                        <entry>
                            <key>Description</key>
                            <value>"The role of the switch. e.g. leaf,
spine"</value>
                        </entry>
                        <entry>
                            <key>IsMandatory</key>
                            <value>true</value>
                        </entry>
                    </annotations>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                    <name>SWITCH_ROLE</name>
                    <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
                 </parameters>
                 <parameters>
                    <annotations>
                       <entry>
                          <key>IsTier</key>
                          <value>true</value>
                       </entry>
                       <entry>
                          <key>Description</key>
                          <value>"Tier number of the switch. Tier 1 is
leaf, 2 is spine"</value>
                       </entry>
                       <entry>
                          <key>IsMandatory</key>
                          <value>true</value>
                       </entry>
                    </annotations>
                    <name>TIER_NUMBER</name>
                    <parameterType>integer</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
                 </parameters>
                 <parameters>
                    <annotations>
                       <entry>
                          <key>Description</key>
                          <value>"List of VLANs allowed separated by
comma"</value>
                       </entry>
                       <entry>
                          <key>IsMandatory</key>
                          <value>true</value>
                       </entry>
                    </annotations>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <name>VLAN_LIST</name>
                        <parameterType>string</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
                              <key>Description</key>
                              <value>"The comma separated list of fabric
ports"</value>
                           </entry>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
                           <entry>
                              <key>IsFabricPort</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>FABRIC_INTERFACES</name>
                        <parameterType>interfaceRange</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>BGP_NEIGHBOR_IP</name>
                        <parameterType>ipV4Address</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
<parameters>
   <annotations>
      <entry>
         <key>IsMandatory</key>
         <value>true</value>
      </entry>
   </annotations>
   <name>BGP_REMOTE_AS</name>
   <parameterType>integer</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations>
      <entry>
         <key>IsMandatory</key>
         <value>true</value>
      </entry>
   </annotations>
   <name>XMPP_GROUPS_SEP_BY_SPACE</name>
   <parameterType>string</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations>
      <entry>
         <key>IsMandatory</key>
         <value>true</value>
      </entry>
   </annotations>
   <name>XMPP_SERVER</name>
   <parameterType>string</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>XMPP_PASSWORD</name>
                     <parameterType>string</parameterType>
                    <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                           <value>"True if the cable plan should be
enabled"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>ENABLE_CABLE_PLAN</name>
                     <parameterType>boolean</parameterType>
                    <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <published>true</published>
                  <supportedPlatforms>N7K,N6K</supportedPlatforms>
                  <templateType>POAP</templateType>
                  <timestamp>2013-09-30 08:04:08</timestamp>
                  <userDefined>false</userDefined>
               </item>
               <item>
                  <assignedInstanceClassId>0</assignedInstanceClassId>
                  <instanceClassId>31</instanceClassId>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
<instanceName>com.cisco.dcbu.dcm.model.cfgtemplate.ConfigTemplate:nam
e=Base_Leaf_Template:type=false</instanceName>
            <description>This file specifies the template configuration
for leaf switch</description>
                <fileName>base_leaf.template</fileName>
                <name>Base_Leaf_Template</name>
                <parameters>
                   <annotations>
                      <entry>
                         <key>Description</key>
                         <value>"The host name of the switch"</value>
                      </entry>
                      <entry>
                         <key>IsMandatory</key>
                         <value>true</value>
                      </entry>
                      <entry>
                         <key>UseDNSReverseLookup</key>
                         <value>true</value>
                      </entry>
                      <entry>
                         <key>IsSwitchName</key>
                         <value>true</value>
                      </entry>
                   </annotations>
                   <name>SWITCH_NAME</name>
                   <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                   <structureParameters/>
                </parameters>
                <parameters>
                   <annotations>
                      <entry>
                         <key>Description</key>
                         <value>"Management IP address used by DCNM to
monitor this device"</value>
                      </entry>
                      <entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                <key>IsVPCPeerLinkSrc</key>
                <value>true</value>
            </entry>
            <entry>
                <key>IsManagementIP</key>
                <value>true</value>
            </entry>
            <entry>
                <key>IsMandatory</key>
                <value>true</value>
            </entry>
        </annotations>
        <name>MGMT_IP</name>
        <parameterType>ipV4Address</parameterType>
     <parameterTypeStructure>false</parameterTypeStructure>
        <structureParameters/>
    </parameters>
    <parameters>
        <annotations>
            <entry>
                <key>Description</key>
                <value>"Management Prefix"</value>
            </entry>
            <entry>
                <key>IsMandatory</key>
                <value>true</value>
            </entry>
        </annotations>
        <name>MGMT_PREFIX</name>
        <parameterType>integer</parameterType>
     <parameterTypeStructure>false</parameterTypeStructure>
        <structureParameters/>
    </parameters>
    <parameters>
        <annotations>
            <entry>
                <key>Description</key>
                <value>"Default Gateway IP address"</value>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>DEFAULT_GATEWAY</name>
                     <parameterType>ipV4Address</parameterType>
                  <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                           <value>"Plain text or 5 encrypted"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>ADMIN_PASSWORD</name>
                     <parameterType>string</parameterType>
                  <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                           <value>"The role of the switch. e.g. leaf,
spine"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <entry>
                           <key>IsSwitchRole</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <defaultValue>leaf</defaultValue>
                     <name>SWITCH_ROLE</name>
                     <parameterType>string</parameterType>
                  <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                           <value>"The comma and dash separated list of
fabric ports"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
                           <value>true</value>
                        </entry>
                        <entry>
                           <key>IsFabricPort</key>
                           <value>true</value>
                        </entry>
                     </annotations>
                     <name>FABRIC_INTERFACES</name>
                     <parameterType>interfaceRange</parameterType>
                  <parameterTypeStructure>false</parameterTypeStructure>
                     <structureParameters/>
                  </parameters>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                           <value>"The comma and dash separated list of
host ports"</value>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            </entry>
            <entry>
               <key>IsMandatory</key>
               <value>true</value>
            </entry>
            <entry>
               <key>IsHostPort</key>
               <value>true</value>
            </entry>
         </annotations>
         <name>HOST_INTERFACES</name>
         <parameterType>interfaceRange</parameterType>
       <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
               <key>Description</key>
               <value>"Backbone VLAN ID"</value>
            </entry>
            <entry>
               <key>IsMandatory</key>
               <value>true</value>
            </entry>
         </annotations>
         <name>BACKBONE_VLAN</name>
         <parameterType>integer</parameterType>
       <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
               <key>Description</key>
               <value>"Backbone IP address/prefix"</value>
            </entry>
            <entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>BACKBONE_IP</name>
      <parameterType>string</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>Description</key>
            <value>"Backbone IPv6 address/prefix"</value>
         </entry>
         <entry>
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>BACKBONE_IPV6</name>
      <parameterType>string</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>BGP_ROUTER_IP</name>
      <parameterType>string</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <annotations>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>BGP_RR_IP</name>
                        <parameterType>string</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
                              <key>Description</key>
                              <value>"IP Address of the Auto-config LDAP
Server"</value>
                           </entry>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>LDAP_SERVER_IP</name>
                        <parameterType>ipV4Address</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
                              <key>Description</key>
                             <value>"IP Address of the XMPP Server"</value>
                           </entry>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                    </annotations>
                    <name>XMPP_SERVER_IP</name>
                    <parameterType>ipV4Address</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
                 </parameters>
                 <parameters>
                    <annotations>
                       <entry>
                          <key>Description</key>
                          <value>"FQDN of the XMPP Server"</value>
                       </entry>
                       <entry>
                          <key>IsMandatory</key>
                          <value>true</value>
                       </entry>
                    </annotations>
                    <name>XMPP_SERVER</name>
                    <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
                 </parameters>
                 <parameters>
                    <annotations>
                       <entry>
                          <key>Description</key>
                          <value>"Space separated XMPP Spine Group
Names"</value>
                       </entry>
                       <entry>
                          <key>IsMandatory</key>
                          <value>true</value>
                       </entry>
                    </annotations>
                    <name>XMPP_GROUPS</name>
                    <parameterType>string</parameterType>
                 <parameterTypeStructure>false</parameterTypeStructure>
                    <structureParameters/>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
</parameters>
<parameters>
   <annotations>
      <entry>
         <key>Description</key>
         <value>"Password"</value>
      </entry>
      <entry>
         <key>IsMandatory</key>
         <value>true</value>
      </entry>
   </annotations>
   <name>XMPP_PASSWORD</name>
   <parameterType>string</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations>
      <entry>
         <key>Description</key>
       <value>"True if VPC should be configured"</value>
      </entry>
      <entry>
         <key>IsMandatory</key>
         <value>true</value>
      </entry>
   </annotations>
   <name>ENABLE_VPC</name>
   <parameterType>boolean</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations>
      <entry>
         <key>IsVPCDomainID</key>
         <value>true</value>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            </entry>
         </annotations>
         <name>VPC_DOMAIN_ID</name>
         <parameterType>integer</parameterType>
      <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
               <key>IsVPCPeerLinkDst</key>
               <value>true</value>
            </entry>
         </annotations>
         <name>VPC_PEER_DST</name>
         <parameterType>ipV4Address</parameterType>
      <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
               <key>IsVPCPeerLinkPortChannel</key>
               <value>true</value>
            </entry>
            <entry>
               <key>IsVPCPort</key>
               <value>true</value>
            </entry>
         </annotations>
         <name>VPC_PEER_LINK_PORT_CHANNEL_NUMBER</name>
         <parameterType>integer</parameterType>
      <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <key>IsVPCPeerLinkPort</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>VPC_PEER_LINK_IF_NAMES</name>
                  <parameterType>interfaceRange</parameterType>
               <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations/>
                  <name>VPC_ARRAY</name>
                  <parameterType>structureArray</parameterType>
                <parameterTypeStructure>true</parameterTypeStructure>
                  <structureParameters>
                     <entry>
                        <key>ID</key>
                        <value>
                           <annotations>
                              <entry>
                                 <key>IsVPCPortChannel</key>
                                 <value>true</value>
                              </entry>
                              <entry>
                                 <key>IsVPCID</key>
                                 <value>true</value>
                              </entry>
                           </annotations>
                           <name>ID</name>
                           <parameterType>integer</parameterType>

<parameterTypeStructure>false</parameterTypeStructure>
                           <structureParameters/>
                        </value>
                     </entry>
                     <entry>
                        <key>IF_NAME</key>
                        <value>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                              <annotations>
                                 <entry>
                                    <key>IsVPCPort</key>
                                    <value>true</value>
                                 </entry>
                              </annotations>
                              <name>IF_NAME</name>
                              <parameterType>interface</parameterType>

<parameterTypeStructure>false</parameterTypeStructure>
                              <structureParameters/>
                           </value>
                        </entry>
                     </structureParameters>
                  </parameters>
                  <published>true</published>
                  <supportedPlatforms>N7K,N6K</supportedPlatforms>
                  <templateType>POAP</templateType>
                  <timestamp>2013-09-30 08:04:08</timestamp>
                  <userDefined>false</userDefined>
               </item>
               <item>
                  <assignedInstanceClassId>0</assignedInstanceClassId>
                  <instanceClassId>36</instanceClassId>

<instanceName>com.cisco.dcbu.dcm.model.cfgtemplate.ConfigTemplate:nam
e=Base_Spine_Template:type=false</instanceName>
            <description>This file specifies the template configuration
for spine switch</description>
                  <fileName>base_spine.template</fileName>
                  <name>Base_Spine_Template</name>
                  <parameters>
                     <annotations>
                        <entry>
                           <key>Description</key>
                           <value>"The host name of the switch"</value>
                        </entry>
                        <entry>
                           <key>IsMandatory</key>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>UseDNSReverseLookup</key>
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>IsSwitchName</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>SWITCH_NAME</name>
                  <parameterType>string</parameterType>
               <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
                        <value>"Management IP address used by DCNM to
monitor this device"</value>
                     </entry>
                     <entry>
                        <key>IsManagementIP</key>
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>MGMT_IP</name>
                  <parameterType>ipV4Address</parameterType>
               <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
      <annotations>
         <entry>
            <key>Description</key>
            <value>"Management Prefix"</value>
         </entry>
         <entry>
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>MGMT_PREFIX</name>
      <parameterType>integer</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>Description</key>
            <value>"Default Gateway IP address"</value>
         </entry>
         <entry>
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>DEFAULT_GATEWAY</name>
      <parameterType>ipV4Address</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>Description</key>
            <value>"Plain text or 5 encrypted"</value>
         </entry>
         <entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>ADMIN_PASSWORD</name>
                  <parameterType>string</parameterType>
                <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
                        <value>"The role of the switch. e.g. leaf,
spine"</value>
                     </entry>
                     <entry>
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                     <entry>
                        <key>IsSwitchRole</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <defaultValue>spine</defaultValue>
                  <name>SWITCH_ROLE</name>
                  <parameterType>string</parameterType>
                <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
                  <annotations>
                     <entry>
                        <key>Description</key>
                        <value>"The comma and dash separated list of
fabric ports"</value>
                     </entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            <entry>
               <key>IsMandatory</key>
               <value>true</value>
            </entry>
            <entry>
               <key>IsFabricPort</key>
               <value>true</value>
            </entry>
         </annotations>
         <name>FABRIC_INTERFACES</name>
         <parameterType>interfaceRange</parameterType>
       <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
               <key>Description</key>
               <value>"Backbone VLAN ID"</value>
            </entry>
            <entry>
               <key>IsMandatory</key>
               <value>true</value>
            </entry>
         </annotations>
         <name>BACKBONE_VLAN</name>
         <parameterType>integer</parameterType>
       <parameterTypeStructure>false</parameterTypeStructure>
         <structureParameters/>
      </parameters>
      <parameters>
         <annotations>
            <entry>
               <key>Description</key>
               <value>"Backbone IP address"</value>
            </entry>
            <entry>
               <key>IsMandatory</key>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            <value>true</value>
         </entry>
      </annotations>
      <name>BACKBONE_IP</name>
      <parameterType>ipV4Address</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>Description</key>
            <value>"Backbone Prefix"</value>
         </entry>
         <entry>
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>BACKBONE_PREFIX</name>
      <parameterType>integer</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>Description</key>
            <value>"IP Address of the XMPP Server"</value>
         </entry>
         <entry>
            <key>IsMandatory</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>XMPP_SERVER_IP</name>
      <parameterType>ipV4Address</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
                              <key>Description</key>
                              <value>"FQDN of the XMPP Server"</value>
                           </entry>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>XMPP_SERVER</name>
                        <parameterType>string</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
                              <key>Description</key>
                              <value>"Space separated XMPP Spine Group
Names"</value>
                           </entry>
                           <entry>
                              <key>IsMandatory</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>XMPP_GROUPS</name>
                        <parameterType>string</parameterType>
                     <parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </parameters>
                     <parameters>
                        <annotations>
                           <entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <key>IsMandatory</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>XMPP_PASSWORD</name>
                  <parameterType>string</parameterType>
                <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <published>true</published>
               <supportedPlatforms>N7K,N6K</supportedPlatforms>
               <templateType>POAP</templateType>
               <timestamp>2013-09-30 08:04:08</timestamp>
               <userDefined>false</userDefined>
            </item>
            <item>
               <assignedInstanceClassId>0</assignedInstanceClassId>
               <instanceClassId>38</instanceClassId>

<instanceName>com.cisco.dcbu.dcm.model.cfgtemplate.ConfigTemplate:nam
e=VPC_Template:type=false</instanceName>
         <description>This file specifies the template configuration
for vpc with annotations</description>
               <fileName>vpc_annotations.template</fileName>
               <name>VPC_Template</name>
               <parameters>
                  <annotations>
                     <entry>
                        <key>IsVPCDomainID</key>
                        <value>true</value>
                     </entry>
                  </annotations>
                  <name>VPC_DOMAIN_ID</name>
                  <parameterType>integer</parameterType>
                <parameterTypeStructure>false</parameterTypeStructure>
                  <structureParameters/>
               </parameters>
               <parameters>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
      <annotations>
         <entry>
            <key>IsVPCPeerLinkSrc</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>VPC_PEER_SRC</name>
      <parameterType>ipV4Address</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>IsVPCPeerLinkDst</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>VPC_PEER_DST</name>
      <parameterType>ipV4Address</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
   </parameters>
   <parameters>
      <annotations>
         <entry>
            <key>IsVPCPeerLinkPortChannel</key>
            <value>true</value>
         </entry>
         <entry>
            <key>IsVPCPort</key>
            <value>true</value>
         </entry>
      </annotations>
      <name>VPC_PEER_LINK_PORT_CHANNEL_NUMBER</name>
      <parameterType>integer</parameterType>
   <parameterTypeStructure>false</parameterTypeStructure>
      <structureParameters/>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
</parameters>
<parameters>
   <annotations/>
   <name>VPC_PEER_LINK_PORT_CHANNEL_DESCRIPTION</name>
   <parameterType>string</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations>
      <entry>
         <key>IsVPCPeerLinkPort</key>
         <value>true</value>
      </entry>
   </annotations>
   <name>VPC_PEER_LINK_IF_NAMES</name>
   <parameterType>interfaceRange</parameterType>
 <parameterTypeStructure>false</parameterTypeStructure>
   <structureParameters/>
</parameters>
<parameters>
   <annotations/>
   <name>VPC_ARRAY</name>
   <parameterType>structureArray</parameterType>
  <parameterTypeStructure>true</parameterTypeStructure>
   <structureParameters>
      <entry>
         <key>ID</key>
         <value>
            <annotations>
               <entry>
                  <key>IsVPCID</key>
                  <value>true</value>
               </entry>
            </annotations>
            <name>ID</name>
            <parameterType>integer</parameterType>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
<parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </value>
                  </entry>
                  <entry>
                     <key>PORT_CHANNEL_NUMBER</key>
                     <value>
                        <annotations>
                           <entry>
                              <key>IsVPCPortChannel</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>PORT_CHANNEL_NUMBER</name>
                        <parameterType>integer</parameterType>

<parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </value>
                  </entry>
                  <entry>
                     <key>IF_NAME</key>
                     <value>
                        <annotations>
                           <entry>
                              <key>IsVPCPort</key>
                              <value>true</value>
                           </entry>
                        </annotations>
                        <name>IF_NAME</name>
                        <parameterType>interface</parameterType>

<parameterTypeStructure>false</parameterTypeStructure>
                        <structureParameters/>
                     </value>
                  </entry>
                  <entry>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        <key>DESCRIPTION</key>
                        <value>
                           <annotations/>
                           <name>DESCRIPTION</name>
                           <parameterType>string</parameterType>

<parameterTypeStructure>false</parameterTypeStructure>
                           <structureParameters/>
                        </value>
                     </entry>
                  </structureParameters>
               </parameters>
               <published>true</published>
               <supportedPlatforms>N7K,N6K</supportedPlatforms>
               <templateType>POAP</templateType>
               <timestamp>2013-09-30 08:04:08</timestamp>
               <userDefined>false</userDefined>
            </item>
         </result>
      </tns:getAllPublishedTemplatesResponse>
   </env:Body>
</env:Envelope>
```

Generate Configuration with the template

This is a SOAP API call to generate the configuration and later used to create the POAP definition. In this example, it's using the "Base_Leaf_Template" to generate the configuration.

http://10.77.247.139:80/ConfigTemplateWSService/ConfigTemplateWS

Request

```
<SOAP-ENV:Envelope
xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <SOAP-ENV:Header>
    <ns0:token
xmlns:ns0="http://dcnm/headers">ZO29rayKhzlieynOjb2N+b1x7pRHNyqB</ns0
:token>
    <ns0:session xmlns:ns0="http://dcnm/headers">0</ns0:session>
  </SOAP-ENV:Header>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  <SOAP-ENV:Body>
    <tns:getPopulatedCommands
xmlns:tns="http://ep.san.jaxws.dcbu.cisco.com/">
      <arg0>
        <instanceClassId>31</instanceClassId>

<instanceName>com.cisco.dcbu.dcm.model.cfgtemplate.ConfigTemplate:nam
e=Base_Leaf_Template:type=false</instanceName>
      </arg0>
      <arg1>
        <item>SWITCH_NAME</item>
        <item>MGMT_IP</item>
        <item>MGMT_PREFIX</item>
        <item>DEFAULT_GATEWAY</item>
        <item>ADMIN_PASSWORD</item>
        <item>SWITCH_ROLE</item>
        <item>FABRIC_INTERFACES</item>
        <item>HOST_INTERFACES</item>
        <item>BACKBONE_VLAN</item>
        <item>BACKBONE_IP</item>
        <item>BACKBONE_IPV6</item>
        <item>BGP_ROUTER_IP</item>
        <item>BGP_RR_IP</item>
        <item>LDAP_SERVER_IP</item>
        <item>XMPP_SERVER_IP</item>
        <item>XMPP_SERVER</item>
        <item>XMPP_GROUPS</item>
        <item>XMPP_PASSWORD</item>
        <item>ENABLE_VPC</item>
        <item>VPC_DOMAIN_ID</item>
        <item>VPC_PEER_DST</item>
        <item>VPC_PEER_LINK_PORT_CHANNEL_NUMBER</item>
        <item>VPC_PEER_LINK_IF_NAMES</item>
        <item>VPC_ARRAY</item>
      </arg1>
      <arg2>
        <item>leaf-SR123456</item>
        <item>172.22.31.23</item>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            <item>24</item>
            <item>172.22.31.1</item>
            <item>cisco123</item>
            <item>leaf</item>
            <item>eth1/4-8</item>
            <item>eth1/10-12</item>
            <item>12</item>
            <item>172.22.31.15</item>
            <item>2345::2346</item>
            <item>172.22.31.10</item>
            <item>172.22.31.12</item>
            <item>172.22.31.2</item>
            <item>172.22.31.3</item>
            <item>xmpp.cisco.com</item>
            <item>leaf_group</item>
            <item>xmpp_123</item>
            <item>True</item>
            <item>3</item>
            <item>172.22.31.24</item>
            <item>2</item>
            <item>eth3/4</item>
            <item>{{2,eth4/4}}</item>
         </arg2>
         <arg3>true</arg3>
      </tns:getPopulatedCommands>
   </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

Response

Note Replace $$ with empty in the below response

```
<env:Envelope xmlns:env="http://schemas.xmlsoap.org/soap/envelope/">
   <env:Header/>
   <env:Body>
      <tns:getPopulatedCommandsResponse
xmlns:tns="http://ep.san.jaxws.dcbu.cisco.com/">
         <result>license grace-period
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
hostname $$leaf-SR123456$$
install feature-set fabric
feature-set fabric
feature fabric forwarding
install feature-set fabricpath
feature-set fabricpath
!#Cable Management
feature lldp
feature cable-management
fabric connectivity tier 1
feature bgp
feature interface-vlan
feature vn-segment-vlan-based
feature dhcp
feature evb
feature pim
!#feature SPoM
feature fabric access
!### Vinci Multicast Forwarding (NGMVPN)
feature fabric multicast
!### Vinci passive PIM
ip multicast fabric-forwarding
fabric forwarding identifier 1
fabric forwarding anycast-gateway-mac 2020.0000.00AA
fabric forwarding switch-role leaf
username admin password $$cisco123$$ role network-admin
no password strength-check
ip domain-lookup
!### Configure IP host for SPoM XMPP server below
ip host $$xmpp.cisco.com$$ $$172.22.31.3$$
!### Configure SPoM XMPP Server below
fabric access server $$xmpp.cisco.com$$ vrf management password
$$xmpp_123$$
!### Subscribe this device to this XMPP group
fabric access group $$leaf_group$$
fabric database profile-map global
  ethernet-tag encapsulation dot1q default dynamic
  ethernet-tag encapsulation vni default dynamic
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  vdp vni default dynamic
!### Configure fabric database location
!### db-table "ou=segments,dc=cisco,dc=com"
!### is a variable that should match the table
!### name that is populated in the LDAP database.
fabric database type asset
  server protocol ldap ip $$172.22.31.2$$
     db-table ou=segments,dc=cisco,dc=com key-type 1
!### Enable global mobility-domain only when you
!### are going to use (vlan, mobility_domain) to search ADBM database
!### fabric database mobility-domain %mobility_domain
route-map FABRIC-RMAP-REDIST-HOST deny 10
  match interface Vlan $$12$$
route-map FABRIC-RMAP-REDIST-HOST permit 20
  match ip address HOSTS

route-map FABRIC-RMAP-REDIST-V6HOST  deny 10
  match interface Vlan $$12$$
route-map FABRIC-RMAP-REDIST-V6HOST permit 20
  match ip address V6HOSTS
ip dhcp snooping
service dhcp
ip dhcp relay
ip dhcp relay information option
ip dhcp relay information option vpn
interface Vlan $$12$$
  no shutdown
  ip address $$172.22.31.15$$
  ipv6 address $$2345::2346$$
  fabric forwarding control-segment
interface $$eth1/7$$
  no shutdown
  switchport
  switchport mode fabricpath
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
interface $$eth1/8$$
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  no shutdown
  switchport
  switchport mode fabricpath
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
interface $$eth1/6$$
  no shutdown
  switchport
  switchport mode fabricpath
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
interface $$eth1/5$$
  no shutdown
  switchport
  switchport mode fabricpath
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
interface $$eth1/4$$
  no shutdown
  switchport
  switchport mode fabricpath
  fabricpath isis hello-interval 100
  fabricpath isis retransmit-interval 10
  fabricpath isis retransmit-throttle-interval 200
interface $$eth1/10$$
  switchport
  switchport mode trunk
  no shutdown
interface $$eth1/11$$
  switchport
  switchport mode trunk
  no shutdown
interface $$eth1/12$$
  switchport
  switchport mode trunk
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  no shutdown
route-map ALL-PATHS permit 10
  set path-selection all advertise
vlan $$12$$
  mode fabricpath
router bgp 100
  router-id $$172.22.31.10$$
  address-family ipv4 unicast
    redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
    maximum-paths ibgp 2
    additional-paths receive
    additional-paths selection route-map ALL-PATHS
  address-family ipv6 unicast
    redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
    maximum-paths ibgp 2
    additional-paths receive
    additional-paths selection route-map ALL-PATHS
  address-family vpnv4 unicast
    additional-paths receive
  address-family vpnv6 unicast
    additional-paths receive
  address-family ipv4 mvpn
    additional-paths receive
  address-family ipv6 mvpn
    additional-paths receive
  neighbor $$172.22.31.12$$ remote-as 100
    address-family ipv4 unicast
      send-community both
    address-family ipv6 unicast
      send-community both
    address-family vpnv4 unicast
      send-community extended
    address-family vpnv6 unicast
      send-community extended
    address-family ipv4 mvpn
      send-community both
    address-family ipv6 mvpn
      send-community both
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
vrf context management
  ip route 0.0.0.0/0 $$172.22.31.1$$
interface mgmt0
  vrf member management
  no cdp enable
  ip address $$172.22.31.23$$/$$24$$
system fabric dynamic-vlans 2500-3500
!###l3vm uses bootflash:platform.inf for the core vlans.
!###Please refer to CSCuj12763 for additional details
system fabric core-vlans  2500-2999
line console
  exec-timeout 0
line vty
!### Sample config for setting interface
!### address e.g. for connecting to
!### 101.101.101.91 ldap server
!interface Ethernet 1/18
!  no switchport
!  ip address 101.101.101.22/24
!  no shutdown
configure profile vrf_tenant_profile
   vlan $id
      mode fabricpath
      vn-segment $segment
   interface vlan $id
      vrf member $vrfname
      ip address $$172.22.31.15$$
      ipv6 address $$2345::2346$$
      no shut
exit
configure profile vrf-common
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv4 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       maximum-paths ibgp 2
exit
config profile GoldProfile
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   ip dhcp relay address $dhcpServerAddr use-vrf default
   fabric forwarding mode proxy-gateway
   no ip redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
  include profile vrf-common
end

configure profile vrf-common-v6
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv6 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
       maximum-paths ibgp 2
exit

config profile GoldProfile-v6
  interface vlan $vlanId
   vrf member $vrfName
   ipv6 address $gatewayIpv6Address/$prefixLength
   fabric forwarding mode proxy-gateway
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
  include profile vrf-common-v6
end

configure profile vrf-common-v4nv6
   vrf context $vrfName
    vni $include_l3_segid
    rd auto
    address-family ipv4 unicast
     route-target both auto
    address-family ipv6 unicast
     route-target both auto
    router bgp $asn
     vrf $vrfName
      address-family ipv4 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-HOST
       maximum-paths ibgp 2
      address-family ipv6 unicast
       redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST
       maximum-paths ibgp 2
exit

config profile GoldProfile-v4nv6
  interface vlan $vlanId
   vrf member $vrfName
   ip address $gatewayIpAddress/$netMaskLength
   ipv6 address $gatewayIpv6Address/$prefixLength
   ip dhcp relay address $dhcpServerAddr use-vrf default
   fabric forwarding mode proxy-gateway
   no ip redirects
   no shutdown
  vlan $vlanId
   vn-segment $segmentId
  include profile vrf-common-v4nv6
end</result>
      </tns:getPopulatedCommandsResponse>
   </env:Body>
</env:Envelope>
```

REVIEW DRAFT—CISCO CONFIDENTIAL

Create POAP Definition

This REST API call will generate the DCNM POAP definition, save the definition into DCNM database and publish the configuration to the devices. This will use the configuration generated from step 6 and pass them in the request which shown in blue color.

REST URL: http://10.77.247.139/rest/poap/poap-definitions/

HTTP Method:POST

Request

```
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 6.1; en-US; rv:1.9.2.3)
Gecko/20100401 Firefox/3.6.3

Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip,deflate

Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7

Keep-Alive: 115

Connection: keep-alive

Content-Type: application/json; charset=UTF-8

     Dcnm-Token: NUVyu5Y0YHDv0tT6xFuIPd+Cu9OA1XCQ

Content-Length: 28

Cookie: JSESSIONID=4C975316B01A44215861475E1B5F9328

Pragma: no-cache

Cache-Control: no-cache

{

"poapTemplateCol":[{"templateContent":"license grace-period\nhostname
leaf-SR123456\n\ninstall feature-set fabric\nfeature-set
fabric\nfeature fabric forwarding\n\ninstall feature-set
fabricpath\nfeature-set fabricpath\n\n!#Cable Management\nfeature
lldp\nfeature cable-management\nfabric connectivity tier 1\n\nfeature
bgp\nfeature interface-vlan\nfeature vn-segment-vlan-based\nfeature
dhcp\nfeature evb\nfeature pim\n\n!#feature SPoM\nfeature fabric
access\n\n!### Vinci Multicast Forwarding (NGMVPN)\nfeature fabric
multicast\n\n!### Vinci passive PIM\nip multicast
fabric-forwarding\n\nfabric forwarding identifier 1\nfabric forwarding
anycast-gateway-mac 2020.0000.00AA\nfabric forwarding switch-role
leaf\n\nusername admin password cisco123 role network-admin\nno
password strength-check\nip domain-lookup\n\n!### Configure IP host for
SPoM XMPP server below\nip host xmpp.cisco.com 172.22.31.3\n\n!###
Configure SPoM XMPP Server below\nfabric access server xmpp.cisco.com
vrf management password xmpp_123\n\n!### Subscribe this device to this
XMPP group \nfabric access group leaf_group\n\nfabric database
profile-map global\n  ethernet-tag encapsulation dot1q default
dynamic\n  ethernet-tag encapsulation vni default dynamic\n  vdp vni
default dynamic\n\n!### Configure fabric database location\n!###
```

REVIEW DRAFT—CISCO CONFIDENTIAL

db-table \"ou=segments,dc=cisco,dc=com\"\n!### is a variable that should match the table\n!### name that is populated in the LDAP database.\nfabric database type asset\n server protocol ldap ip 172.22.31.2\n   db-table ou=segments,dc=cisco,dc=com key-type 1\n\n!### Enable global mobility-domain only when you \n!### are going to use (vlan, mobility_domain) to search ADBM database   \n!### fabric database mobility-domain %mobility_domain\n\nroute-map FABRIC-RMAP-REDIST-HOST deny 10\n match interface Vlan 12\nroute-map FABRIC-RMAP-REDIST-HOST permit 20\n match ip address HOSTS \n\nroute-map FABRIC-RMAP-REDIST-V6HOST deny 10\n match interface Vlan 12\nroute-map FABRIC-RMAP-REDIST-V6HOST permit 20\n match ip address V6HOSTS\n\nip dhcp snooping\nservice dhcp\nip dhcp relay\nip dhcp relay information option\nip dhcp relay information option vpn\n\ninterface Vlan 12\n no shutdown\n ip address 172.22.31.15\n ipv6 address 2345::2346\n fabric forwarding control-segment\n\n\ninterface eth1/7\n no shutdown\n switchport\n switchport mode fabricpath\n fabricpath isis hello-interval 100\n fabricpath isis retransmit-interval 10\n fabricpath isis retransmit-throttle-interval 200\n\ninterface eth1/8\n no shutdown\n switchport\n switchport mode fabricpath\n fabricpath isis hello-interval 100\n fabricpath isis retransmit-interval 10\n fabricpath isis retransmit-throttle-interval 200\n\ninterface eth1/6\n no shutdown\n switchport\n switchport mode fabricpath\n fabricpath isis hello-interval 100\n fabricpath isis retransmit-interval 10\n fabricpath isis retransmit-throttle-interval 200\n\ninterface eth1/5\n no shutdown\n switchport\n switchport mode fabricpath\n fabricpath isis hello-interval 100\n fabricpath isis retransmit-interval 10\n fabricpath isis retransmit-throttle-interval 200\n\ninterface eth1/4\n no shutdown\n switchport\n switchport mode fabricpath\n fabricpath isis hello-interval 100\n fabricpath isis retransmit-interval 10\n fabricpath isis retransmit-throttle-interval 200\n\n\ninterface eth1/10\n switchport \n switchport mode trunk\n no shutdown\n\ninterface eth1/11\n switchport \n switchport mode trunk\n no shutdown\n\ninterface eth1/12\n switchport \n switchport mode trunk\n no shutdown\n\nroute-map ALL-PATHS permit 10\n set path-selection all advertise\n\nvlan 12\n mode fabricpath\n\nrouter bgp 100\n router-id 172.22.31.10\n address-family ipv4 unicast\n redistribute hmm route-map FABRIC-RMAP-REDIST-HOST \n   maximum-paths ibgp 2 \n   additional-paths receive\n   additional-paths selection route-map ALL-PATHS\n address-family ipv6 unicast\n   redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST\n   maximum-paths ibgp 2\n   additional-paths receive\n   additional-paths selection route-map ALL-PATHS\n address-family vpnv4 unicast\n   additional-paths receive\n address-family vpnv6 unicast\n   additional-paths receive\n address-family ipv4 mvpn\n   additional-paths receive\n address-family ipv6 mvpn\n   additional-paths receive\n\n neighbor 172.22.31.12 remote-as 100\n   address-family ipv4 unicast\n   send-community both\n   address-family ipv6 unicast\n   send-community both\n   address-family vpnv4 unicast\n   send-community extended\n   address-family vpnv6 unicast\n   send-community extended\n   address-family ipv4 mvpn\n   send-community both\n   address-family ipv6 mvpn\n   send-community both\n\nvrf context management\n ip route 0.0.0.0/0

REVIEW DRAFT—CISCO CONFIDENTIAL

172.22.31.1\n\ninterface mgmt0\n vrf member management\n no cdp enable\n ip address 172.22.31.23/24\n\nsystem fabric dynamic-vlans 2500-3500\n\n!###l3vm uses bootflash:platform.inf for the core vlans. \n!###Please refer to CSCuj12763 for additional details\nsystem fabric core-vlans 2500-2999\n\nline console\n exec-timeout 0\nline vty\n\n\n!### Sample config for setting interface\n!### address e.g. for connecting to\n!### 101.101.101.91 ldap server\n\n!interface Ethernet 1/18\n! no switchport\n! ip address 101.101.101.22/24\n! no shutdown\n\nconfigure profile vrf_tenant_profile\n   vlan $id\n mode fabricpath\n   vn-segment $segment\n   interface vlan $id\n vrf member $vrfname\n   ip address 172.22.31.15\n   ipv6 address 2345::2346\n   no shut\nexit\n\nconfigure profile vrf-common\n   vrf context $vrfName\n   vni $include_l3_segid\n   rd auto\n address-family ipv4 unicast\n   route-target both auto\n   router bgp $asn\n   vrf $vrfName\n   address-family ipv4 unicast\n redistribute hmm route-map FABRIC-RMAP-REDIST-HOST\n   maximum-paths ibgp 2 \nexit\n\nconfig profile GoldProfile\n interface vlan $vlanId\n vrf member $vrfName\n   ip address $gatewayIpAddress/$netMaskLength\n ip dhcp relay address $dhcpServerAddr use-vrf default\n   fabric forwarding mode proxy-gateway\n   no ip redirects\n   no shutdown\n vlan $vlanId\n   vn-segment $segmentId\n include profile vrf-common\nend\n\nconfigure profile vrf-common-v6\n   vrf context $vrfName\n   vni $include_l3_segid\n   rd auto\n   address-family ipv6 unicast\n   route-target both auto\n   router bgp $asn\n   vrf $vrfName\n   address-family ipv6 unicast\n   redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST\n   maximum-paths ibgp 2 \nexit\n\nconfig profile GoldProfile-v6\n interface vlan $vlanId\n vrf member $vrfName\n   ipv6 address $gatewayIpv6Address/$prefixLength\n   fabric forwarding mode proxy-gateway\n   no shutdown\n vlan $vlanId\n   vn-segment $segmentId\n include profile vrf-common-v6\nend\n\nconfigure profile vrf-common-v4nv6\n vrf context $vrfName\n   vni $include_l3_segid\n rd auto\n   address-family ipv4 unicast\n   route-target both auto\n address-family ipv6 unicast\n   route-target both auto\n   router bgp $asn\n   vrf $vrfName\n   address-family ipv4 unicast\n redistribute hmm route-map FABRIC-RMAP-REDIST-HOST\n   maximum-paths ibgp 2 \n   address-family ipv6 unicast\n   redistribute hmm route-map FABRIC-RMAP-REDIST-V6HOST\n   maximum-paths ibgp 2 \nexit\n\nconfig profile GoldProfile-v4nv6\n interface vlan $vlanId\n vrf member $vrfName\n   ip address $gatewayIpAddress/$netMaskLength\n ipv6 address $gatewayIpv6Address/$prefixLength\n   ip dhcp relay address $dhcpServerAddr use-vrf default\n   fabric forwarding mode proxy-gateway\n   no ip redirects\n   no shutdown\n vlan $vlanId\n vn-segment $segmentId\n include profile vrf-common-v4nv6\nend\n","templateName":"Base_Leaf_Template","methodType":"POST","id":0,"templateNVPairs":"{\"0\":{\"text\":\"leaf-SR123456\",\"annotations\":[{\"value\":\"\\\"The host name of the switch\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"IsMandatory\"},{\"value\":\"true\",\"key\":\"UseDNSReverseLookup\"},{\"value\":\"true\",\"key\":\"IsSwitchName\"}],\"name\":\"SWITCH_NAME\"},\"1\":{\"text\":\"172.22.31.23\",\"annotations\":[{\"value\":\"\\\"Management IP address used by DCNM to monitor this device\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"I

REVIEW DRAFT—CISCO CONFIDENTIAL

sVPCPeerLinkSrc\"},{\"value\":\"true\",\"key\":\"IsManagementIP\"},{\
"value\":\"true\",\"key\":\"IsMandatory\"}],\"name\":\"MGMT_IP\"},\"2
\":{\"text\":\"24\",\"annotations\":[{\"value\":\"\\\"Management
Prefix\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"I
sMandatory\"}],\"name\":\"MGMT_PREFIX\"},\"3\":{\"text\":\"172.22.31.
1\",\"annotations\":[{\"value\":\"\\\"Default Gateway IP
address\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"
IsMandatory\"}],\"name\":\"DEFAULT_GATEWAY\"},\"4\":{\"text\":\"cisco
123\",\"annotations\":[{\"value\":\"\\\"Plain text or 5
encrypted\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":
\"IsMandatory\"}],\"name\":\"ADMIN_PASSWORD\"},\"5\":{\"text\":\"leaf
\",\"annotations\":[{\"value\":\"\\\"The role of the switch. e.g. leaf,
spine\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"Is
Mandatory\"},{\"value\":\"true\",\"key\":\"IsSwitchRole\"}],\"name\":
\"SWITCH_ROLE\"},\"6\":{\"text\":\"eth1/4-8\",\"annotations\":[{\"val
ue\":\"\\\"The comma and dash separated list of fabric
ports\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"Is
Mandatory\"},{\"value\":\"true\",\"key\":\"IsFabricPort\"}],\"name\":
\"FABRIC_INTERFACES\"},\"7\":{\"text\":\"eth1/10-12\",\"annotations\"
:[{\"value\":\"\\\"The comma and dash separated list of host
ports\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"Is
Mandatory\"},{\"value\":\"true\",\"key\":\"IsHostPort\"}],\"name\":\"
HOST_INTERFACES\"},\"8\":{\"text\":\"12\",\"annotations\":[{\"value\"
:\"\\\"Backbone VLAN
ID\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"IsMan
datory\"}],\"name\":\"BACKBONE_VLAN\"},\"9\":{\"text\":\"172.22.31.15
\",\"annotations\":[{\"value\":\"\\\"Backbone IP
address/prefix\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"k
ey\":\"IsMandatory\"}],\"name\":\"BACKBONE_IP\"},\"10\":{\"text\":\"2
345::2346\",\"annotations\":[{\"value\":\"\\\"Backbone IPv6
address/prefix\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"k
ey\":\"IsMandatory\"}],\"name\":\"BACKBONE_IPV6\"},\"11\":{\"text\":\
"172.22.31.10\",\"annotations\":[{\"value\":\"true\",\"key\":\"IsMand
atory\"}],\"name\":\"BGP_ROUTER_IP\"},\"12\":{\"text\":\"172.22.31.12
\",\"annotations\":[{\"value\":\"true\",\"key\":\"IsMandatory\"}],\"n
ame\":\"BGP_RR_IP\"},\"13\":{\"text\":\"172.22.31.2\",\"annotations\"
:[{\"value\":\"\\\"IP Address of the Auto-config LDAP
Server\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"I
sMandatory\"}],\"name\":\"LDAP_SERVER_IP\"},\"14\":{\"text\":\"172.22
.31.3\",\"annotations\":[{\"value\":\"\\\"IP Address of the XMPP
Server\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"I
sMandatory\"}],\"name\":\"XMPP_SERVER_IP\"},\"15\":{\"text\":\"xmpp.c
isco.com\",\"annotations\":[{\"value\":\"\\\"FQDN of the XMPP
Server\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"I
sMandatory\"}],\"name\":\"XMPP_SERVER\"},\"16\":{\"text\":\"leaf_grou
p\",\"annotations\":[{\"value\":\"\\\"Space separated XMPP Spine Group
Names\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\":\"Is
Mandatory\"}],\"name\":\"XMPP_GROUPS\"},\"17\":{\"text\":\"xmpp_123\"
,\"annotations\":[{\"value\":\"\\\"Password\\\"\",\"key\":\"Descripti
on\"},{\"value\":\"true\",\"key\":\"IsMandatory\"}],\"name\":\"XMPP_P
ASSWORD\"},\"18\":{\"text\":\"True\",\"annotations\":[{\"value\":\"\\
\"True if VPC should be
configured\\\"\",\"key\":\"Description\"},{\"value\":\"true\",\"key\"

REVIEW DRAFT—CISCO CONFIDENTIAL


```
:\"IsMandatory\"}],\"name\":\"ENABLE_VPC\"},\"19\":{\"text\":\"3\",\"
annotations\":[{\"value\":\"true\",\"key\":\"IsVPCDomainID\"}],\"name
\":\"VPC_DOMAIN_ID\"},\"20\":{\"text\":\"172.22.31.24\",\"annotations
\":[{\"value\":\"true\",\"key\":\"IsVPCPeerLinkDst\"}],\"name\":\"VPC
_PEER_DST\"},\"21\":{\"text\":\"2\",\"annotations\":[{\"value\":\"tru
e\",\"key\":\"IsVPCPeerLinkPortChannel\"},{\"value\":\"true\",\"key\"
:\"IsVPCPort\"}],\"name\":\"VPC_PEER_LINK_PORT_CHANNEL_NUMBER\"},\"22
\":{\"text\":\"eth3/4\",\"annotations\":[{\"value\":\"true\",\"key\":
\"IsVPCPeerLinkPort\"}],\"name\":\"VPC_PEER_LINK_IF_NAMES\"},\"23\":{
\"text\":\"{{2,eth4/4}}\",\"annotations\":[],\"name\":\"VPC_ARRAY\"}}
"}],"poapSwitchCol":[{"switchName":"leaf-SR123456","publish":"true","
lanGroup":2,"deviceType":"N6K","systemImageName":"n6000-system-image.
bin","virutalDeviceContextName":"vdc","kickstartImageName":"n6000-kic
k-image.bin","methodType":"POST","imageServerId":1,"serialNumber":"SR
123456","switchStatus":"Not
Discovered","configServerId":1,"username":"admin","mgmtIp":"172.22.31
.23","publishStatus":"Published","tier":0,"id":0,"password":"cisco123
"}]
}
```


Response

```
HTTP/1.1 202 Accepted
```

vCD Sample Script

<Introduction Required>

```
"""
.. module:: vCDclient
        :platform: Linux, Windows
      :synopsis: Reference module script to demonstrate the interaction
between VMware vCD
                  and Cisco DCNM via VMware vCloud AMQP notification,
REST APIs and DCNM
                   REST APIs.


.. moduleauthor:: Cisco DCNM team


.. note:: The configuration parameters need to be specified in
:file:`vCDclient-ini.conf` file
                before running this script.
"""
import sys, ConfigParser, time
import urllib2
import contextlib
import base64
import json
import requests
try:
        import xml.etree.cElementTree as et
except ImportError:
        import xml.etree.ElementTree as et
# for AMQP
import pika
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
import logging
from logging import StreamHandler, FileHandler
logger = logging.getLogger('vCDclient')
class AMQPClient():
        """ This AMQP client class listens to vCD's AMQP notification
and interacts
       with VMware vCloud Director (vCD) and vShield Manager (vSM) for
further tenant
        and network information. It also communicates with DCNM to
populate network data.
        """
        def __init__(self, params, client_vcds, client_dcnm):
                """Create a new instance of AMQP client.
              :param dict params: AMQP configuration parameters, e.g.
AMQP server ip, port,
                                          user name, password, name of
AMQP exchange and queue for vCD notification.
                :param list client_vcds: vCD client instances.
                :param object client_dcnm: DCNM instance.
                :raises: ValueError
                """
                # extract from input params
                self._server_ip = params.get('ip')
                self._port = int(params.get('port'))
                self._user = params.get('user')
                self._pwd = params.get('password')
                # exchange, queue name for receiving vCD events
               self._vcd_exchange_name = params.get('vcdexchangename')
                self._vcd_queue_name = params.get('vcdqueuename')
                self._client_vcds = client_vcds
                self._client_vcd = None
                self._client_dcnm = client_dcnm
             if (not self._server_ip) or (not self._vcd_exchange_name)
or (not self._vcd_queue_name):
                        raise ValueError, '[AMQPClient] Input IP, vCD
exchange name or vCD queue name parameter is not specified'
                logger.info('[AMQPClient] AMQP server: %s, exchange
name: %s, queue name: %s.' % (self._server_ip, self._vcd_exchange_name,
self._vcd_queue_name))
                logger_pika = logging.getLogger('pika')
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                logger_pika.setLevel(logging.CRITICAL)
        def _cb_vcd_msg(self, ch, method, properties, body):
                """ Callback function to process vCD
organization/VDC/network
                creation/update/deletion AMQP messages being received.
             It also communicates with vCD and vSM to extract detailed
info
                and passed them to DCNM via DCNM REST APIs.
                :param pika.channel.Channel ch: The channel instance.
                :param method method: The method
                """
                if 'true.' not in method.routing_key:
                        # send acknowledgement
                      ch.basic_ack(delivery_tag = method.delivery_tag)
                        return
                if 'network' in method.routing_key:
                        self._process_org_vdc_network_msg(ch, method,
properties, body)
                        # send acknowledgement
                      ch.basic_ack(delivery_tag = method.delivery_tag)
                        return
                key_org_create = 'com.vmware.vcloud.event.org.create'
               # no need to process vCD org modify event, as the only
field - org name is not editable in vCD
                key_org_delete = 'com.vmware.vcloud.event.org.delete'
             key_org_vdc_create = 'com.vmware.vcloud.event.vdc.create'
             key_org_vdc_update = 'com.vmware.vcloud.event.vdc.modify'
             key_org_vdc_delete = 'com.vmware.vcloud.event.vdc.delete'
                if (key_org_create in method.routing_key) or
(key_org_delete in method.routing_key):
                        tenant_name = self._parse_vcd_org_event(body)
                        if tenant_name:
                                # add tenant entry
                            if (key_org_create in method.routing_key):

self._client_dcnm.create_org(tenant_name)
                                else:

self._client_dcnm.delete_org(tenant_name)
               elif  method.routing_key.endswith(key_org_vdc_create) \
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                                    or
method.routing_key.endswith(key_org_vdc_update) \
                                    or
method.routing_key.endswith(key_org_vdc_delete):
                            (tenant_name, vdc_name) =
self._parse_vcd_org_vdc_event(body)
                            if tenant_name and vdc_name:
                                    if
method.routing_key.endswith(key_org_vdc_create):

self._client_dcnm.create_update_partition(tenant_name, vdc_name, True)
                                    elif
method.routing_key.endswith(key_org_vdc_update):

self._client_dcnm.create_update_partition(tenant_name, vdc_name,
False)
                                    else:

self._client_dcnm.delete_partition(tenant_name, vdc_name)
                # send acknowledgement
                ch.basic_ack(delivery_tag = method.delivery_tag)
        def _process_org_vdc_network_msg(self, ch, method, properties,
body):
                """ Process vCD vDC network creation/update/deletion
AMQP message
                being received.
              It also communicates with vCD and vSM to extract detailed
info
                and pass them to DCNM via DCNM REST APIs.
                :param pika.channel.Channel ch: The channel
                """
                if 'true.' not in method.routing_key:
                        return
                key_vdc_network_create_complete =
'com.vmware.vcloud.event.task.complete.networkCreateOrgVdcNetwork'
                key_network_delete_complete =
'com.vmware.vcloud.event.task.complete.networkDelete'
                key_network_update_complete =
'com.vmware.vcloud.event.task.complete.networkUpdateNetwork'
                key_vapp_network_deploy =
'com.vmware.vcloud.event.network.deploy'
                key_vapp_network_undeploy =
'com.vmware.vcloud.event.network.undeploy'
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                if (key_vdc_network_create_complete not in
method.routing_key) \
                        and (key_vdc_network_create_complete not in
method.routing_key) \
                        and (key_network_delete_complete not in
method.routing_key) \
                        and (key_network_update_complete not in
method.routing_key) \
                        and (key_vapp_network_deploy not in
method.routing_key) \
                        and (key_vapp_network_undeploy not in
method.routing_key):
                                return
                is_vapp_network = False
                is_create_network = True
                self._parse_vcd_event(body)
                if (key_vapp_network_deploy in method.routing_key) \
                        or (key_vapp_network_undeploy in
method.routing_key):
                        is_vapp_network = True
               if (key_network_update_complete in method.routing_key):
                        is_create_network = False
              if (key_network_delete_complete in method.routing_key) \
                                or (key_vapp_network_undeploy in
method.routing_key):
                        network_info =
self._client_vcd.process_network_delete_message(body,
is_vapp_network)
                       self._client_dcnm.delete_network(network_info)
                else:
                        network_info =
self._client_vcd.process_network_create_update_message(body,
is_vapp_network)

self._client_dcnm.create_update_network(network_info,
is_create_network)
        def _parse_vcd_event(self, msg):
                """Parse vCD event to find the input vCD instance with
matched IP address.
                :param str msg: The received vCD AMQP notification.
                :returns: object -- The matched vCD instance.
                """
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                if not msg:
                        return
                # entity resolver
                root = et.fromstring(msg)
                # find entityResolver
               node_resolver = root.find('.//*[@rel="entityResolver"]')
                url_resolver = ''
                if node_resolver is not None:
                        url_resolver = node_resolver.attrib['href']
                vcd_ip = url_resolver.split('/')[2]
                for client_vcd in self._client_vcds:
                        if vcd_ip == client_vcd._vcd_ip:
                               logger.debug('[AMQPClient] vCD IP: %s'
% vcd_ip)
                                self._client_vcd = client_vcd
                                break
        def _parse_vcd_org_event(self, msg):
            """Parse vCD organization event to extract the organization
name.
               :param str msg: The vCD organization AMQP notification.
                :returns: str -- The organization name.
                """
                if not msg:
                        return
                org_name = None
                # entity resolver
                root = et.fromstring(msg)
                # find org element
                node_org =
root.find('.//*[@rel="entity"][@type="vcloud:org"]')
                if node_org is not None:
                        org_name = node_org.attrib['name']
                return org_name
        def _parse_vcd_org_vdc_event(self, msg):
                """Parse vCD VDC event to extract the organization and
vDC name.
                :param str msg: The vCD VDC AMQP notification.
              :returns: tuple (str, str) -- (organization name, VDC name)
                """
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                if not msg:
                        return
                org_name = None
                vdc_name = None
                # entity resolver
                root = et.fromstring(msg)
                # find vdc element
                node_vdc =
root.find('.//*[@rel="entity"][@type="vcloud:vdc"]')
                if node_vdc is not None:
                        vdc_name = node_vdc.attrib['name']
                        node_orgs =
root.findall('.//*[@rel="up"][@type="vcloud:org"]')
                        for node_org in node_orgs:
                                org_name = node_org.attrib['name']
                                if org_name != 'System':
                                        break;
                return (org_name, vdc_name)
        def process_amqp_msgs(self):
                """Process AMQP queue messages.
              It connects to AMQP server and calls callbacks to process
VMware events,
               i.e. routing key containing '.vmware.', once they arrive
in the queue.
                """
                # specify the key of interest
                key = '#.vmware.#'
                self._conn = None
                consume_channel = None
                try:
                        credentials = None
                        if self._user:
                                credentials =
pika.PlainCredentials(self._user, self._pwd)
                        # create connection, channel
                        self._conn =
pika.BlockingConnection(pika.ConnectionParameters(host =
self._server_ip, port = self._port, credentials = credentials))
                        # create channels for consuming
                        consume_channel = self._conn.channel()
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        # declare vCD exchange
                        vcd_exchange =
consume_channel.exchange_declare(exchange = self._vcd_exchange_name,
type = 'topic', durable = True, auto_delete = False)
                       result = consume_channel.queue_declare(queue =
self._vcd_queue_name, durable = True, auto_delete = False)
                        consume_channel.queue_bind(exchange =
self._vcd_exchange_name, queue = self._vcd_queue_name, routing_key =
key)
                        # for info only
                        msg_count = result.method.message_count
                      logger.info('[AMQPClient] The exchange %r queue
%r has totally %d messages. ' % (self._vcd_exchange_name,
self._vcd_queue_name, msg_count))
                        print ' [*] About to retrieve messages. Press
Ctl-C to exit'
                        # consume messages
                     consume_channel.basic_consume(self._cb_vcd_msg,
queue = self._vcd_queue_name)
                        consume_channel.start_consuming()
                except KeyboardInterrupt:
                        print '\n Received Ctl-C.'
                finally:
                      # don't call cancel or close due to pika's error
#                       if consume_channel:
#                               consume_channel.cancel()
#                               consume_channel.close()
                        if self._conn:
                                self._conn.close()
class VCDWSClient():
        """ This vCD Web Service client class interacts with vCD and
vSM for detailed tenant
        and network information.
        """
        def __init__(self, params_vcd, params_vsm):
                """Create a new instance of vCD client.
                :param dict params_vcd: vCD configuration parameters,
e.g. vCD ip and user name.
                :param dict params_vsm: vSM configuration parameters,
e.g. vSM ip and user name.
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                :raises: ValueError                :param dict params_vcd:
vCD configuration parameters, e.g. vCD ip and user name.
                 """
                 self._req_header = {'Accept':'application/*+xml'}
                 self._url_login = None
                 self._version_num = 1.5
                 # vCD namespace
                 self._NS_VCD = 'http://www.vmware.com/vcloud/'
                 # format {vcd_network_entity_id: segment_id}
                 self._network_mapping = {}
                 # hard-coded tenant mapping
                 self._tenant_mapping = {}
                 # url timeout: 10 seconds
                 self._TIMEOUT_URL_OPEN = 10
                 # extract from input params
                 self._vcd_ip = params_vcd.get('ip')
                 # vCD user format: userName@org
                self._vcd_user = '%s@system' % (params_vcd.get('user'))
                 self._vcd_pwd = params_vcd.get('password')
                 self._vsm_ip = params_vsm.get('ip')
                 self._vsm_user = params_vsm.get('user')
                 self._vsm_pwd = params_vsm.get('password')
                 if (not self._vcd_ip) or (not self._vcd_user) or (not
self._vcd_pwd):
                         raise ValueError, '[VCDWSClient] Input vCD IP,
user name or password parameter is not specified'
               elif (not self._vsm_ip) or (not self._vsm_user) or (not
self._vsm_pwd):
                         raise ValueError, '[VCDWSClient] Input vSM IP,
user name or password parameter is not specified'
                 logger.info('[vCDWSClient] vCD IP: %s, vCD User: %s,
vSM IP: %s, vSM User: %s.' % (self._vcd_ip, self._vcd_user,
self._vsm_ip, self._vsm_user))
        def get_tenant_vdc_network(self):
                 """ Retrieve all the organization, VDC and networks.
                This method does not return all the data in one shot;
instead for code efficiency, it returns
               organization, VDC or network during its looping through
all the network related data in vCD.
                 :returns: * tuple (str) -- Organization name
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        * Or tuple (str, str) -- (Organization name,
VDC name)
                        * Or tuple (str, str, dict) -- (Organization
name, VDC name, network info)
                :note: The caller needs to loop through the returned
data until no more data is available.
                """
                try:
                        #login
                        self._login()
                        # retrieve tenants list
                    tenants_url = 'https://%s/api/org' % (self._vcd_ip)
                        tenants_msg = self._get_response(tenants_url)
                        tenants_root = et.fromstring(tenants_msg)
                        # extract namespace
                        tag_vcd = tenants_root.tag
                        ns_vcd = ''
                        if tag_vcd.startswith('{'):
                                ns_vcd = tag_vcd[1:].split('}')[0]
                        # extract tenant nodes
                     tenant_nodes = tenants_root.findall('.//{%s}Org'
% ns_vcd)
                        for tenant_node in tenant_nodes:
                              tenant_name = tenant_node.attrib['name']
                               tenant_url = tenant_node.attrib['href']
                               # ignore system built-in tenant: System
                                if tenant_name == 'System':
                                        continue
                                yield (tenant_name)
                           tenant_msg = self._get_response(tenant_url)
                               tenant_root = et.fromstring(tenant_msg)
                                vdcs_name = []
                                vdcs_nodes =
tenant_root.findall('.//*[@type="application/vnd.vmware.vcloud.vdc+xm
l"]')
                                for vdc_node in vdcs_nodes:
                                    vdc_name = vdc_node.attrib['name']
                                        yield (tenant_name, vdc_name)
                                tenant_id = tenant_url.split('/')[-1]
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                                networks_nodes =
tenant_root.findall('.//*[@type="application/vnd.vmware.vcloud.orgNet
work+xml"]')
                                for network_node in networks_nodes:
                                        network_url =
network_node.attrib['href']
                                        network_info =
self._compose_network_info(tenant_id, tenant_name, network_url)
                                       yield (tenant_name, vdc_name,
network_info)
                except (urllib2.HTTPError, urllib2.URLError) as e:
                        if isinstance(e, urllib2.HTTPError):
                               reason = 'Error reaching URL (%s) with
code %s.' % (e.url, e.code)
                        else:
                               reason = 'Error reaching URL (%s) with
reason %s.' % (e.url, e.reason)
                        logger.exception(reason)
                finally:
                        self._logout()
        def process_network_create_update_message(self, msg,
is_vapp_network):
                """Process vCD's network creation and update event.
              It retrieves detailed network info from vCD via vCD REST
APIs, adds the vCD network Id
              to class network mapping table (for segment Id retrieval
later) and deletes the vSE (due
                to some duplicated features offered by DFA leaf nodes).
                :param str msg: The vCD's AMQP notification.
             :param bool is_vapp_network: The flag to indicate whether
the input message
                                            is related to vApp network.
             :returns: dict -- Network data which includes tenant_name
(organization name),
                                       network_name, segment_id as keys.
                """
                logger.info('[VCDWSClient] Start processing network
create/update message ...')
                network_info = None
                tenant_name = None
                network_name = None
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                segment_id = None
                try:
                   # parse the event first, so as to generate DCNM event
                        network_event_info =
self._parse_vcd_network_event(msg)
                        if not network_event_info:
                                return
                        # login
                        self._login()
                        tenant_id = network_event_info['tenant_id']
                       tenant_name = network_event_info['tenant_name']
                       urn_network = network_event_info['urn_network']
                        network_id = network_event_info['network_id']
                     network_link =  self._get_network_link(urn_network)
                   network_info = self._compose_network_info(tenant_id,
tenant_name, network_link)
                        if not network_info:
                                return
                        segment_id = network_info['segment_id']
                        network_name = network_info['network_name']
                     # add network Id to global network mapping table,
with segment Id being filled later
                        self._set_segment_id(network_id, segment_id,
is_vapp_network)
                        self._logout()
                        # delete vSM edge (vSE)
                       self._delete_vsm_edge(tenant_id, network_name)
                    # adding a 10s delay to ensure that vSE is deleted
                        # before ldap is populated
                        time.sleep(10)
                except (urllib2.HTTPError, urllib2.URLError) as e:
                        if isinstance(e, urllib2.HTTPError):
                                reason = 'Error reaching URL (%s) with
code %s.' % (e.url, e.code)
                        else:
                                reason = 'Error reaching URL (%s) with
reason %s.' % (e.url, e.reason)
                        logger.exception(reason)
                return network_info
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
        def process_network_delete_message(self, msg, is_vapp_network):
                """Process vCD's network deletion event.
               It retrieves detailed network info from vCD via vCD REST
APIs, and finds the matched
                segment Id from class network mapping table based on
extracted vCD network Id.
                :param str msg: The vCD's AMQP notification.
              :param bool is_vapp_network: The flag to indicate whether
the input message
                                                is related to vApp network.
              :returns: dict -- Network data which includes tenant_name
(organization name),
                                                and segment_id as keys.
                """
                logger.info('[VCDWSClient] Start processing network
delete message ...')
               network_event_info = self._parse_vcd_network_event(msg)
                if not network_event_info:
                        return
                network_id = network_event_info['network_id']
                segment_id = self._lookup_segment_id(network_id,
is_vapp_network, True)
                network_info = {'tenant_name':
network_event_info['tenant_name'],
                                'segment_id': segment_id
                                }
                return network_info
        def _lookup_segment_id(self, vcd_network_entity_id,
is_vapp_network, remove_entry):
               """Find segment Id based on input vCD network entry Id.
                DFA uses segment Id to identify the network, whereas
vCD uses UUID formatted network
               Id to identify the network. So a network mapping table
needs to be maintained to
                map between vCD network Id and DFA segment Id.
               :param str vcd_nework_entry_id: The network Id used by
vCD to identify the network.
              :param bool is_vapp_network: The flag to indicate whether
the input message
                                                is related to vApp network.
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                :param bool remove_entry: The flag to indicate whether
to remove the entry from class
                                          network mapping table.
                :returns: str -- The matched segment Id if found.

                """

                for network_id, (segment_id, vapp_network_flag) in
self._network_mapping.iteritems():
                        if (network_id == vcd_network_entity_id) and
(vapp_network_flag == is_vapp_network):
                                if remove_entry:
                                 del self._network_mapping[network_id]
                                return segment_id


        def _set_segment_id(self, vcd_network_entity_id, segment_id,
is_vapp_network):
                """Add segment Id and vCD network entry Id entry to
internal mapping table
                for network deletion message processing later.

                :param str vcd_nework_entry_id: The network Id used by
vCD to identify the network.
                :param str segment_id: The segment Id used by DFA to
identify the network.
                :param bool is_vapp_network: The flag to indicate whether
the input message
                                            is related to vApp network.

                """

                if not segment_id:
                        return
                for network_id, (network_segment_id, vapp_network_flag)
in self._network_mapping.iteritems():
                        if segment_id == network_segment_id:
                                return
                self._network_mapping.update({vcd_network_entity_id:
(segment_id, is_vapp_network)})
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
        def _compose_request_header(self, url):
                """Compose HTTP request header.

                :param str url: The URI that the request is sent to.
                :returns: Request -- The HTTP request object.

                """

                req = urllib2.Request(url)
                for key, value in self._req_header.iteritems():
                        req.add_header(key, value)
                return req

        def _get_response(self, url):
                """Generalize the HTTP(S) request/response processing.

                :param str url: The URI that the request is sent to.
                :returns: str -- The HTTP response body message.

                """

                req = self._compose_request_header(url)
                content = None
                try:
                        with contextlib.closing(urllib2.urlopen(req,
timeout = self._TIMEOUT_URL_OPEN)) as res:
                                content = res.read()
                except urllib2.URLError as e:
                      # add url to the exception for caller to display
                        e.url = url
                        raise
                return content


        def _parse_vcd_network_event(self, msg):
                """Parse vCD network event.
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                :param str msg: The vCD AMQP notification.
               :returns: dict -- The extracted network data which have
tenant_id (organization Id),
                                        tenant_name (organization name),
network_id (vCD network
                                          entry Id) and urn_network (URN
of vCD network entry) as keys.

                """

                if not msg:
                        return

                # entity resolver
                root = et.fromstring(msg)

                # find tenant Id (org)
                node_org = root.find('.//*[@type="vcloud:org"]')
                org_id = node_org.attrib['id'].split(':')[-1]
                org_name = node_org.attrib['name']

                # find entityResolver
              node_resolver = root.find('.//*[@rel="entityResolver"]')
                url_resolver = ''
                if node_resolver is not None:
                        url_resolver = node_resolver.attrib['href']

                # find network id
              node_network = root.find('.//*[@type="vcloud:network"]')
                network_id = ''
                if node_network is not None:
                        network_id = node_network.attrib['id']

                urn_network = ''.join([url_resolver, network_id])

                vcd_network = {'tenant_id': org_id,
                               'tenant_name': org_name,
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                                'network_id': network_id,
                                'urn_network': urn_network
                                }
                return vcd_network


        def _get_network_link(self, urn_network):
                """Retrive network reference link.

                It sends HTTP(S) GET request to vCD and extracts the
network reference
                link from the response body.

               :param str urn_network: The URN of network entry in vCD.
                :returns: str -- The URI of network reference link.

                """

                network_link = None
                # resolve network entity
                entity_resolver_res = self._get_response(urn_network)

                # find the first href for network
                if entity_resolver_res:
                        root = et.fromstring(entity_resolver_res)
                  node_alternate = root.find('.//*[@rel="alternate"]')
                        if node_alternate is not None:
                          network_link = node_alternate.attrib['href']

                return network_link


        def _vsm_ws_request(self, vsm_url, delete_action = False):
                """Send HTTP(S) GET or DELETE request to vSM.

                This method is called to retrieve networks (virtual
wires), edges and delete individual edge.
                :param str vsm_url: The URL of vSM resource.
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                :param bool delete_action: The flag to indicate whether
it is DELETE request.

                :returns: str -- The HTTP(S) response body.

                """

                req = urllib2.Request(vsm_url)
                req.add_header('Accept', 'application/*+xml')
                if delete_action:
                        req.get_method = lambda:'DELETE'

                # use base64
                base64string = base64.encodestring('%s:%s' %
(self._vsm_user, self._vsm_pwd))[:-1]
            req.add_header("Authorization", "Basic %s" % base64string)
                content = None
                try:
                        with contextlib.closing(urllib2.urlopen(req,
timeout = self._TIMEOUT_URL_OPEN)) as res:
                                content = res.read()
                except urllib2.URLError as e:
                      # add url to the exception for caller to display
                        e.url = 'vSM: ' + vsm_url
                        raise
                return content


        def _get_vsm_segment_id(self, tenant_id, network_name):
                """Retrieve segment Id from vSM which has the matched
organization Id and network name.

                :param str tenant_id: The organization Id in vCD.
                :param str network_name: The network name.
               :returns: tuple -- (segment Id in vSM, port profile name
in N1KV)

                """
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                url_networks = 'https://%s/api/2.0/vdn/virtualwires' %
(self._vsm_ip)
                 vsm_networks = self._vsm_ws_request(url_networks)
                 segment_id = None
                 port_profile_n1kv_name = None
                 if vsm_networks:
                         # network is vSM's virtual wire
                         root = et.fromstring(vsm_networks)
                         for network in root.findall('.//virtualWire'):
                                 tenant_id_value =
network.find('tenantId').text
                                name_value = network.find('name').text
                                if((tenant_id_value == tenant_id) and
(network_name in name_value)):
                                        # find segment Id
                               segment_id = network.find('vdnId').text
                                 # compose N1KV (vDS) port profile name

#<virtualWire><objectId>virtualwire-6</objectId>
                                        virtualwire_id =
network.find('objectId').text

#<vdsContextWithBacking><switch><objectId>dvs-136
                                     dvs = network.find('.//switch')
                                      dvs_id = ''
                                      if dvs:
                                    dvs_id = dvs.find('objectId').text
                                        port_profile_name =
'vxw-%s-%s-sid-%s-%s' % (dvs_id, virtualwire_id, segment_id,
name_value)
                                        # example:
vxw-dvs-136-virtualwire-6-sid-10003-dvs.VCDVSNetPepsiInternal-50bc778
c-2fcd-454a
                                # N1kV port profile name max length: 80
                                        port_profile_n1kv_name =
port_profile_name[:80]
                                        break


                return (segment_id, port_profile_n1kv_name)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
        def _delete_vsm_edge(self, tenant_id, network_name):
                """Delete vSE by sending HTTP(S) DELETE to vSM edge
resource.

                :param str tenant_id: The organization Id in vCD.
                :param str network_name: The network name.
                :returns: str -- The vSE edge Id used in vSM.

                """

                edge_id = None

               url_edges = 'https://%s/api/3.0/edges' % (self._vsm_ip)
                try:
                        vsm_edges = self._vsm_ws_request(url_edges)
                        if vsm_edges:
                                root = et.fromstring(vsm_edges)
                           for edge in root.findall('.//edgeSummary'):
                                   name_value = edge.find('name').text
                                node_tenant_id = edge.find('tenantId')
                                        if node_tenant_id is None:
                                                continue
                                 tenant_id_value = node_tenant_id.text
                                    if((tenant_id_value == tenant_id)
and (network_name in name_value)):
                                                # find edge Id
                                                edge_id =
edge.find('objectId').text
                                                break
                        # delete edge
                        if edge_id:
                             url_edge = 'https://%s/api/3.0/edges/%s'
% (self._vsm_ip, edge_id)
                                self._vsm_ws_request(url_edge, True)
                              logger.debug('[vCDWSClient] Delete vSM
edge. Edge Id: %s.' % edge_id)

                except urllib2.HTTPError as e:
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        logger.exception('[vCDWSClient] Error code: %s
' % e.code)
                except urllib2.URLError as e:
                        logger.exception('[vCDWSClient] Error reaching
URL (%s) with reason %s.' % (e.url, e.reason))

                return edge_id


        def _compose_network_info(self, tenant_id, tenant_name,
network_url):
                """Retrive detailed network info and parse/compose
network data.

             :param str tenant_id: The organization (tenant) Id in vCD.
                :param str tenant_name: The organization name.
                :param str network_url: The URI that the HTTP(S) GET
request is sent to
                                        for detailed network info.

               :returns: dict -- The detailed network data which have
network_id (organization Id),
                                    tenant_name, vrf_name (VDC name),
network_name, segment_id,
                                       gateway, netmask, port_profile
(on N1KV), dns, ip_start and
                                        ip_end as keys.

                """
                if not network_url:
                        return
                vcd_network = self._get_response(network_url)
                node_vcd_network = et.fromstring(vcd_network)

                # find VDC name (network's parent)
                node_vdc =
node_vcd_network.find('.//*[@type="application/vnd.vmware.vcloud.vdc+
xml"]')
                vdc_name = ''
                if node_vdc is not None:  # vCD 1.5 does not have VDC
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        vdc_url =
node_vcd_network.find('.//*[@type="application/vnd.vmware.vcloud.vdc+
xml"]').attrib['href']
                        vdc_msg = self._get_response(vdc_url)
                        vdc_name = et.fromstring(vdc_msg).get('name')


                network_name = node_vcd_network.get('name')


                # contact vSM for segment id
                (segment_id, port_profile_n1kv_name) =
self._get_vsm_segment_id(tenant_id, network_name)


                # extract namespace
                tag_vcd = node_vcd_network.tag
                if tag_vcd.startswith('{'):
                        ns_vcd = tag_vcd[1:].split('}')[0]
                # find subnet info
                node_ipscope = node_vcd_network.find('.//{%s}IpScope'
% ns_vcd)
              gateway = node_ipscope.find('{%s}Gateway' % ns_vcd).text

              netmask = node_ipscope.find('{%s}Netmask' % ns_vcd).text
                node_dns = node_ipscope.find('{%s}Dns1' % ns_vcd)
                dns = ''
                if node_dns is not None:
                        dns = node_dns.text


                gateways = gateway.split('.')
              node_ip_start = node_ipscope.find('.//{%s}StartAddress'
% ns_vcd)


                if node_ip_start is not None:
                        ip_start = node_ip_start.text
                else:
                      ip_start =  '.'.join((gateways[0], gateways[1],
gateways[2], str(int(gateways[3])+ 1)))


                node_ip_end = node_ipscope.find('.//{%s}EndAddress' %
ns_vcd)
                if node_ip_end is not None:
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        ip_end = node_ip_end.text
                else:
                        netmasks = netmask.split('.')
                        ip_end_last_num = 255 - int(gateways[3]) -
int(netmasks[3])
                        ip_end = '.'.join((gateways[0], gateways[1],
gateways[2], str(ip_end_last_num)))

                network_info = {'tenant_id': tenant_id,
                                'tenant_name': tenant_name,
                                'vrf_name': vdc_name,
                                'segment_id': segment_id,
                                'network_name' : network_name,
                                'gateway': gateway,
                                'netmask': netmask,
                              'port_profile' : port_profile_n1kv_name,
                                'dns': dns,
                                 'ip_start': ip_start,
                                 'ip_end': ip_end }
                logger.debug('[vCDWSClient] Network info: %s ' %
network_info )

                return network_info


        def _login(self):
                """Find out vCD version and log into vCD.

              vCD returns session token in login response header after
successful login, and that token
                will be added to the class request header field to be
used for subsequent request
                composition.

                """

                # for error report
                url = ''
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                try:
                        if not self._url_login:
                                # first time to retrieve login url
                               url_version = 'http://%s/api/versions'
% (self._vcd_ip)
                            ns_version = '%sversions' % (self._NS_VCD)

                               url = url_version
                               url_login = None
                               with
contextlib.closing(urllib2.urlopen(url_version, timeout =
self._TIMEOUT_URL_OPEN)) as res:
                              root_version = et.fromstring(res.read())
                                        versions =
root_version.findall('.//{%s}VersionInfo' % (ns_version))
                                        for version_info in versions:
                                                version =
version_info.find('{%s}Version' % (ns_version)).text
                                                if float(version) >
self._version_num:
                                                    self._version_num
= float(version)
                                                        url_login =
version_info.find('{%s}LoginUrl' % (ns_version)).text

                               if not url_login:
                                        self._url_login = None
                                        return
                            # change url to https as vCD only supports
https
                        self._url_login = url_login.replace('http://',
'https://')

                         # basic authentication from session login
                    pwd_mgr = urllib2.HTTPPasswordMgrWithDefaultRealm()
                         pwd_mgr.add_password(None, self._url_login,
self._vcd_user, self._vcd_pwd)
                       handler = urllib2.HTTPBasicAuthHandler(pwd_mgr)

                         opener = urllib2.build_opener(handler)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        # add header
                        opener.addheaders = [('Accept',
'application/*+xml;version=%s' % str(self._version_num))]
                        url = self._url_login
                        with
contextlib.closing(opener.open(self._url_login, data = '', timeout =
self._TIMEOUT_URL_OPEN)) as res:
                                # session Id
                                session_id =
res.info().getheader('x-vcloud-authorization')
                                # update global request header
                                self._req_header =
{'Accept':'application/*+xml;version=%s' % self._version_num,
'x-vcloud-authorization': session_id }



                except urllib2.URLError as e:
                      # add url to the exception for caller to display
                        e.url = 'login: ' + url
                        raise



        def _logout(self):
                """Log out from vCD.
                """

                if not self._url_login:
                        return
                # replace 'sessions' to 'session'
                url_logout = self._url_login.replace('sessions',
'session')
                req = self._compose_request_header(url_logout)
                req.get_method = lambda:'DELETE'
                try:
                        with contextlib.closing(urllib2.urlopen(req,
timeout = self._TIMEOUT_URL_OPEN)) as res:
                                logger.debug('[vCDWSClient] Logout
result: %s' % (res.read()))
                except urllib2.URLError as e:
                      # add url to the exception for caller to display
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        e.url = 'logout: ' + url_logout
                        raise


class DCNMClient():
        """ This DCNM client class interacts with DCNM via DCNM REST
API to populate
        organization (tenant), partition (vrf, VDC) and network data.
        """

        def __init__(self, params_dcnm, params_tenant):
                """Create a new instance of DCNM client.

               :param dict params_dcnm: DCNM configuration parameters,
e.g. DCNM server IP, user name.
                :param dict params_tenant: Default parameters for
organization (tenant) such as orchestration source.

                """

                self._ip = params_dcnm.get('ip')
                self._user = params_dcnm.get('user')
                self._pwd = params_dcnm.get('password')
              if (not self._ip) or (not self._user) or (not self._pwd):
                        raise ValueError, '[DCNMClient] Input DCNM IP,
user name or password parameter is not specified'
                logger.info('[DCNMClient] DCNM IP: %s, User: %s.' %
(self._ip, self._user))

                # tenant defaults
                self._default_forwarding_mode =
params_tenant.get('defaultforwardingmode', 'proxy-gateway')
                self._default_profile =
params_tenant.get('defaultprofilename', 'GoldProfile')
                self._orchestration_source =
params_tenant.get('orchestrationsource', 'vCloud Director')

                # url timeout: 10 seconds
                self._TIMEOUT_RESPONSE = 10
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
        def create_org(self, org_name):
                """Create organization (tenant) by sending POST request
to DCNM auto-config organizations resource.

                :param str org_name: The name of organization to be
created.

                """

                url = 'http://%s/rest/auto-config/organizations' %
(self._ip)
                payload = {'organizationName': org_name,
                                'profileName': self._default_profile,
                                'forwardingMode':
self._default_forwarding_mode,
                                'orchestrationSource':
self._orchestration_source
                                }

               self._send_request('POST', url, payload, 'organization')

        def delete_org(self, org_name):
             """Delete organization (tenant) by sending DELETE request
to DCNM auto-config organizations resource.

                :param str org_name: The name of organization to be
deleted.

                """

                url = 'http://%s/rest/auto-config/organizations/%s' %
(self._ip, org_name)
                self._send_request('DELETE', url, '', 'organization')



        def create_update_partition(self, org_name, partition_name,
is_create = True):
               """Create or update partition (vrf, VDC) by sending POST
or PUT request to
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                DCNM auto-config partitions resource.

                :param str org_name: The organization name.
                :param str partition_name: The partition name.
               :param bool is_create: The flag to indicate whether to
create organization.

                """

                url =
'http://%s/rest/auto-config/organizations/%s/partitions' % (self._ip,
org_name)
                composed_partition_name =
self._compose_partition_name(org_name, partition_name)

                operation = 'POST'
                if not is_create:
                        operation = 'PUT'
                        url =
'http://%s/rest/auto-config/organizations/%s/partitions/%s' %
(self._ip, org_name, composed_partition_name)

                payload = {'organizationName': org_name,
                             'partitionName': composed_partition_name,
                           'profileName': self._default_profile,
                           'forwardingMode':
self._default_forwarding_mode
                           }
              self._send_request(operation, url, payload, 'partition')


        def delete_partition(self, org_name, partition_name):
              """Delete partition (vrf, VDC) by sending DELETE request
to DCNM auto-config partitions resource.

                :param str org_name: The organization name.
                :param str partition_name: The partition name.

                """
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                composed_partition_name =
self._compose_partition_name(org_name, partition_name)
                url =
'http://%s/rest/auto-config/organizations/%s/partitions/%s' %
(self._ip, org_name, composed_partition_name)
                self._send_request('DELETE', url, '', 'partition')



      def create_update_network(self, network_info, is_create = True):
                """Create or update network by sending POST or PUT
request to DCNM auto-config networks resource.

             :param dict network_info: The network info which includes
tenant_name (organization name),
                                       vrf_name (partition name),
segment_id, gateway and netmask.
               :param bool is_create: The flag to indicate whether to
create network.

                """

                org_name = network_info.get('tenant_name', '')
                partition_name = network_info.get('vrf_name', '')
                composed_partition_name =
self._compose_partition_name(org_name, partition_name)

                url =
'http://%s/rest/auto-config/organizations/%s/partitions/%s/networks'
% (self._ip, org_name, composed_partition_name)
                operation = 'POST'
                if not is_create:
                        operation = 'PUT'
                        url =
'http://%s/rest/auto-config/organizations/%s/partitions/%s/networks/s
egment/%s' % (self._ip, org_name, composed_partition_name, segment_id)

                segment_id = network_info['segment_id']
                gateway = network_info.get('gateway', '')
                netmask = network_info.get('netmask', '')
                netmask_len = 24
                if netmask != '':
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        netmask_len = self._convert_netmask(netmask)

                config_args = []
                config_args.append('$vrfName=%s' %
composed_partition_name)
                config_args.append('$segmentId=%s' % segment_id)
                config_args.append('$netMaskLength=%d' % netmask_len)
                config_args.append('$gatewayIpAddress=%s' % gateway)
                config_args.append('$forwardingMode=%s' %
self._default_forwarding_mode)
                config_args = ';'.join(config_args)

                ip_start = network_info.get('ip_start', '')
                ip_end = network_info.get('ip_end', '')
                subnet = gateway[:gateway.rfind('.') + 1] + '0'

            dhcp_scopes = {'ipRange': ('%s-%s' % (ip_start, ip_end)),
                           'subnet': ('%s/%d' % (subnet, netmask_len)),
                                'routers': gateway,
                                'segmentID': segment_id
                                }

             payload = {'networkName': network_info['network_name'],
                              'partitionName': composed_partition_name,
                                'profileName': self._default_profile,
                                'forwardingMode':
self._default_forwarding_mode,
                                'segmentId': segment_id,
                                'configArg': config_args,
                                'dhcpScope': dhcp_scopes
                                }
              self._send_request(operation, url, payload, 'network')


        def delete_network(self, network_info):
                """Delete network by sending DELETE request to DCNM
auto-config networks resource.
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
            :param dict network_info: The network info which includes
tenant_name (organization name),
                                      vrf_name (partition name) and
segment_id.

                """

                if 'segment_id' not in network_info:
                        return
                # Note: vCD network deletion notification does not
contain VDC info, so waiting for DCNM to add search API to make vCD
network deletion work.
                org_name = network_info.get('tenant_name', '')
                partition_name = network_info.get('vrf_name', '')
                composed_partition_name =
self._compose_partition_name(org_name, partition_name)
                segment_id = network_info['segment_id']
                url =
'http://%s/rest/auto-config/organizations/%s/partitions/%s/networks/s
egment/%s' % (self._ip, org_name, composed_partition_name, segment_id)

                self._send_request('DELETE', url, '', 'network')


        def _compose_partition_name(self, org_name, partition_name):
                """Compose partition name.

                :param str org_name: The organization name.
                :param str partition_name: The partition name.
               :returns: str -- The name with 'orgName_partitionName'
format to avoid possible
                                     duplicated partition name among
different organization.

                """

               # combine org and partition name as partition/vrf name
is MUST attribute in LDAP partition/vrf table
                return org_name + '_' + partition_name
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
        def _send_request(self, operation, url, payload, desc):
                """Generalize the HTTP(S) request, which includes POST,
PUT, DELETE.

                :param str operaion: The HTTP verb with value of POST,
PUT or DELETE.
                :param str url: The URI that the request is sent to.
                :param dict payload: The data to be put in the request
body. It will
                                                be converted into JSON format
before being sent out.
                :param str desc: The descrption to be recorded in log
message.
                :returns: Response -- The response object from HTTP(S)
request.

                :notes: It logs into DCNM, sends HTTP(S) request, and
log out from DCNM.

                """

                res = None
                try:
                        payload_json = None
                        if payload and payload != '':
                                payload_json = json.dumps(payload)
                        self._login()
                        if operation == 'POST':
                                res = requests.post(url, data =
payload_json, headers = self._req_headers, timeout =
self._TIMEOUT_RESPONSE)
                                desc += ' creation'
                        elif operation == 'PUT':
                          res = requests.put(url, data = payload_json,
headers = self._req_headers, timeout = self._TIMEOUT_RESPONSE)
                                desc += ' update'
                        elif operation == 'DELETE':
                                res = requests.delete(url, data =
payload_json, headers = self._req_headers, timeout =
self._TIMEOUT_RESPONSE)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                                desc += ' deletion'

                            logger.debug('\n [DCNMClient] REST Response
code: %d, content: %s \n' % (res.status_code, res.content))
                            if res and res.status_code >= 200:
                                  logger.info('[DCNMClient] Sent %s to %s
successfully.' % (desc, url))
                            else:
                                    logger.error('[DCNMClient] Sent %s to
%s unsuccessfully.' % (desc, url))

                            self._logout()
                    except requests.ConnectionError as e:
                           # add url to the exception for caller to display
                            print 'Error connecting to ', url
                            logger.exception(str(e))
                            raise
                    except requests.HTTPError as e:
                            print 'HTTP error'
                            logger.exception(str(e))
                    except requests.Timeout as e:
                            print 'Timeout error'
                            logger.exception(str(e))

                    return res


            def _login(self):
                    """Log into DCNM by calling POST request to DCNM logon
resource.

                 DCNM returns DCNM token in login response after successful
login, and that token
                    will be added to the class request header field to be
used for subsequent request composition.

                    """

                    url_login = 'http://%s/rest/logon' % (self._ip)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                expiration_time = 100000


                payload = {'expirationTime': expiration_time}
                self._req_headers = {'Accept': 'application/json',
'Content-Type': 'application/json; charset=UTF-8'}
            res = requests.post(url_login, data = json.dumps(payload),
headers = self._req_headers, auth = (self._user, self._pwd), timeout =
self._TIMEOUT_RESPONSE)
                logger.debug('[DCNMClient] Login response: %s' %
(res.content))
                session_id = ''
                if res and res.status_code >= 200:
                        session_id = res.json().get('token')
                # update global request header
                self._req_headers.update({'Dcnm-Token': session_id })



        def _logout(self):
                """Log out from DCNM by calling POST request to DCNM
logout resource
                """

                url_logout = 'http://%s/rest/logout' % (self._ip)
               requests.post(url_logout, headers = self._req_headers,
timeout = self._TIMEOUT_RESPONSE)



        def _convert_netmask(self, netmask):
                """Convert netmask from dotted decimal to bitmask.

              :param str netmask: The netmask in dotted decimal format.
                :returns: int -- The bitmask (length).
                """

                arr = netmask.split('.')
                arr = map(int, arr)
                return reduce(lambda x, y: x + (y + 1)/32, arr, 0)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
def read_config_file(config_file):
        """Read initial configuration file.

        :param config_file: Configuration file name.

        """

        config_params = {}

        parser = ConfigParser.ConfigParser()
        parser.readfp(open(config_file))

        for section in parser.sections():
                section_params = {}
                for option in parser.options(section):
                        values = parser.get(section, option)
                        if ';' in values:
                                values = values.split(';')
                        section_params.update({option: values})
                config_params.update({section: section_params})

        return config_params


def set_logger():
        """Set logger with log file name and log message format.

        The log messages are written to 'vcdclient.log' file.
        """

        default_formatter = logging.Formatter('%(asctime)s
%(levelname)s: %(message)s')
        handler_console = StreamHandler()
        handler_console.setFormatter(default_formatter)
        handler_console.setLevel(logging.DEBUG)

        handler_file = FileHandler('vcdclient.log', 'a')
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
        handler_file.setFormatter(default_formatter)

        logger.addHandler(handler_console)
        logger.addHandler(handler_file)


def query_process_vcd_tenants(client_vcds, client_dcnm):
       """ Retrieve the current tenants and network info from vCD, and
calls DCNM to
        create or update organization, partition and network data.

        :param client_vcds: List of vCD instances
        :param client_dcnm: DCNM instance

        """

        # no need to proceed if DCNM client is not present
        if not client_dcnm:
                return

        logger.info('Retrieving tenants (organizations), VDC
(partitions) and network info from vCD.')
        for client_vcd in client_vcds:
              """ Delete partition (vrf) by sending DELETE request to
DCNM auto-config partitions resource.
               """
              for tenant_info in client_vcd.get_tenant_vdc_network():
                       # retrieve all the {tenant: vdc/vrf} from vCD
                       # tenant
                    if (type(tenant_info) == str) or (len(tenant_info)
== 1):
                                (tenant_name) = tenant_info
                                # add tenant entry
                                client_dcnm.create_org(tenant_name)
                       elif len(tenant_info) == 2:
                                (tenant_name, vdc_name) = tenant_info
                                # add vrf entry

client_dcnm.create_update_partition(tenant_name, vdc_name, True)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        elif len(tenant_info) == 3:
                                (tenant_name, vdc_name, segment_data)
= tenant_info
                                # add segment entry

client_dcnm.create_update_network(segment_data, True)



if __name__ == '__main__':
        """Main function for vCDclient flow.

       It reads configuration parameters from the 'vCDclient-ini.conf'
file, retrieves all the
       organization, VDC and network data from VMware vCD, passes the
info to DCNM, and processes
        AMQP notification from vCD.

        """

        config_file_name = 'vCDclient-ini.conf'

        set_logger()

        try:
                config_params = read_config_file(config_file_name)

                # get config params
                params_log = config_params.get('Log')
                params_amqp = config_params.get('AMQP')
                params_vcd = config_params.get('vCD')
                params_vsm = config_params.get('vSM')
                params_dcnm = config_params.get('DCNM')
                params_tenant = config_params.get('Tenant')

                # set logger level
                log_levels = {
                        'DEBUG': logging.DEBUG,
                        'INFO': logging.INFO,
                        'WARNING': logging.WARNING,
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                        'ERROR': logging.ERROR,
                        'CRITICAL': logging.CRITICAL
                }
                logger.setLevel(log_levels.get(params_log['level'],
logging.INFO))

                # check config parameters
                if (not params_amqp) or (not params_vcd) or (not
params_vsm):
                        logger.error('Section [AMQP], [vCD] or [vSM] is
missing in ini.conf file.')
                        exit(1)

                logger.info('Parsed config file %s' % config_file_name)

                client_vcds = []
                params_vcd_values = params_vcd.itervalues().next()

                if isinstance(params_vcd_values, list):
                        len_param_vcd = len(params_vcd_values)
                        for i in range(len_param_vcd):
                                param_vcd = {}
                                for k in params_vcd:
                                param_vcd.update({k: params_vcd[k][i]})
                                logger.debug('vCD input parameters: %s.'
% param_vcd)
                                param_vsm = {}
                                for k in params_vsm:
                                param_vsm.update({k: params_vsm[k][i]})
                                logger.debug('vSM input parameters: %s'
% param_vsm)
                                client_vcds.append((VCDWSClient(param_vcd,
param_vsm)))
                else:
                        client_vcds.append((VCDWSClient(params_vcd,
params_vsm)))

                client_dcnm = DCNMClient(params_dcnm, params_tenant)
                query_process_vcd_tenants(client_vcds, client_dcnm)
```

REVIEW DRAFT—CISCO CONFIDENTIAL

```
                client_amqp = AMQPClient(params_amqp, client_vcds,
client_dcnm)
                client_amqp.process_amqp_msgs()

                logger.info('Exit the program.\n')
                exit(0)

        except Exception as e:
                logger.exception(str(e))
                logger.info('Exit the program.\n')
                exit(1)
```

REVIEW DRAFT—CISCO CONFIDENTIAL