The Network Times
Handbook Series – Part V

Cisco SD-WAN

A Practical Guide to Understand the Basics of Cisco Viptela Based SD-WAN solution

Toni Pasanen, CCIE#28158

Cisco SD-WAN

A Practical Guide to Understand the Basics of Cisco Viptela Based SD-WAN solution

Toni Pasanen, CCIE 28158

Published - 23 May 2021

About the Editor:

Toni Pasanen. CCIE No. 28158 (RS), Distinguished Engineer at Fujitsu Finland. Toni started his IT carrier in 1998 at Tieto, where he worked as a Service Desk Specialist moving via the LAN team to the Data Center team as a 3rd. Level Network Specialist. Toni joined Teleware (Cisco Learning partner) in 2004, where he spent two years teaching network technologies focusing on routing/switching and MPLS technologies. Toni joined Tieto again in 2006, where he spent the next six years as a Network Architect before joining Fujitsu. In his current role, Toni works closely with customers helping them in selecting the right network solutions not only from the technology perspective but also from the business perspective. He is also the author of books:

Virtual Extensible LAN – VXLAN: The Practical Guide to VXLAN Solution –2019

LISP Control-Plane in Campus Fabric. A Practical Guide to Understand the Operation of Campus Fabric– 2020

VXLAN Fabric with BGP EVPN Control-Plane. Design Considerations – 2020

Object-Based Approach to Cisco ACI: The Logic Behind the Application Centric Infrastructure - 2020.

About This Book

I wrote this book primarily for those of you who want to learn the basics of a Cisco SD-WAN solution. The first two chapters explain how to set up your local test environment. You will learn how to install certificates to vBond, vManage, vSmart, and vEdge and how to set up underlay and Out-of-Band management network connections. Chapter three introduces a centralized control plane with the Overlay Management Protocol (OMP) running between vEdges and vSmart. After reading this chapter, you should be familiar with how overlay tunnels between vEdges are built by using TLOC routes. Besides, you learn how Sevice VPN routing information is advertised by using OMP routes. This chapter also introduces the data-plane operation explaining how Service VPNs are segmented in the WAN data plane by using labels. The fourth chapter explains the end-to-end control-plane operation and data-plane encapsulation over the SR-MPLS transport network. Besides, you will learn some of the basic concepts of the IS-IS Segment-Routing extension. The fifth chapter introduces how you can implement Hub and Spoke overlay topology by filtering TLOC routes using Centralized Policy. You will learn how Lists are used within a Control Policy and how the Control Policy, in turn, is attached to Centralized Policy. The sixth chapter introduces Feature Templates in which you set protocol and feature specific values. Then it shows how Feature Templates are attached to Device Template which in turn is attached to devices. This chapter also introduces CLI Templates with device-specific variables. Chapter seven explains how you can use TLOC Extension for transport network connection in dual-homed sites. Chapter eight discusses LAN side BGP routing. It shows how to implement BGP using Feature Templates as well as CLI configuration. It also explains how you can build a Centralized Policy to filter out unnecessary OMP routes. Chapter nine explains how to do preference-based traffic engineering, again with Centralized Policy. Chapter 10 introduces Application-Aware Routing. It starts by explaining how BFD probes are used for tunnel Health Monitoring and how the AAR uses the RTT of BFD probes to monitor the Path Quality. The last chapter explains how to build a Direct Cloud Access (DCA), where the remote-sites user can access applications hosted in Cloud provider networks using the local Internet connection while all other traffic is sent over SD-WAN overlay tunnels. You will also learn how to filter DCA traffic with a centrally managed Zone-Based Firewall.

Network topologies used in this book are built with the minimum amount of devices in order to keep examples as simple as possible.

Disclaimers

The content of this book is based only on the author's own experience and testing results. Its content is neither validated nor accepted by Cisco or any other organization or person. This book is meant to be neither design nor an implementation guide. After reading this book, readers should do their technology validation before using it in a production environment.

Table of Contents

Chapter 1: Setting Up On-Prem Controllers

Introduction

This section explains the process how to build an on-premise Cisco Viptela based SD-WAN control plane system. It starts by setting up an enterprise Certificate Server using a Cisco CSR1000V cloud router. Next, it goes through the process of root certificate generation. The rest of the chapter explains the initial configuration and certification installation processes from vManage, vBond, and vSmart viewpoints.

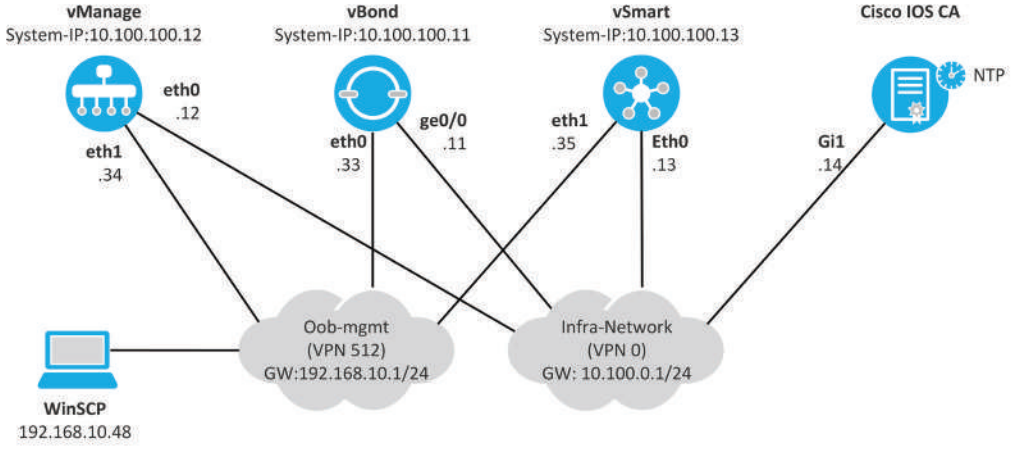


Figure 1-1: *Control-Plane Components Topology.*

Note! I am using EVE-NG running on an ESXi host. You can find installation instructions from eve-ng.net *Documentation > How To's > Cisco SDWAN Viptela image set.*

Configuring IOS-XE Certification Server

In order to onboard vEdges to the SD-WAN system and building a control plane connection between vBond, vManage, and vSmart we need certificates. The focus of this section is to explain how Cisco IOS-XE can be used as a Certification Authority.

Enabling HTTP Server and NTP

The mandatory pre-request is to enable the HTTP server on Certification Authority (CA). In addition, Clock times have to be synchronized between vManage, vBond, vSmart, and Certificate Server, otherwise, there will be problems with certificates. I'm using IOS-XE as *Network Time Protocol (NTP)* master, this way we turn on a hardware clock on the router and provide a time source for other devices.

```
ip http-server
ntp master
```

Example 1-1: Enabling *HTTP and NTP Master services on IOS-XE.*

Certificate Server Configuration

First, we generate an RSA key pair for the IOS-XE Certificate Server (CS). We are using 2048 bits modulus size for the RSA key. As a next step, we start the Certificate Server configuration. The cs-label used with the server must match the label used with the RSA key configuration (We are using the label PKI). We are using a flash as a file database where we are going to store each issued certificate with their serial number and subject name. The CA issuer DN name is set to rootca.nwkt.local. Besides, we are using the SHA-256 hash function for the signature that our CS uses to sign self-signed certificates. We are using the pkcs12 archive format for the CA keys and certificates and the file is encrypted with the password Cisco123. Certificates are generated automatically. As the last step, we turn on the Certificate Server and export the CA certificate into flash in PEM format.

```
CA-Server(config)#crypto key generate rsa label PKI modulus 2048
% The key modulus size is 2048 bits
% Generating 2048 bit RSA keys, keys will be non-exportable...
[OK] (elapsed time was 1 seconds)
CA-Server(config)#crypto pki server PKI
CA-Server(cs-server)#database url flash:
% Server database url was changed. You need to move the
% existing database to the new location.
CA-Server(cs-server)#database level complete
CA-Server(cs-server)#issuer-name cn=root.nwkt.local
CA-Server(cs-server)#hash sha256
CA-Server(cs-server)#database archive pkcs12 password Cisco123
CA-Server(cs-server)#grant auto
Mar 14 13:12:42.854: %PKI-6-CS_GRANT_AUTO: All enrollment requests will be
auto.
CA-Server(cs-server)#no shut
%Some server settings cannot be changed after CA certificate generation.
%Exporting Certificate Server signing certificate and keys...
%Certificate Server enabled.
Mar 14 13:12:57.634: %PKI-6-CS_ENABLED: Certificate server now enabled.
```

Example 1-2: Enabling *HTTP and NTP Master services on IOS-XE.*

When configurations are done, we can export the CA certificate (root certificate) in PEM format to the terminal by using the command ***crypto pki PKI pem terminal.***

```
CA-Server(config)#crypto pki export PKI pem terminal
% The specified trustpoint is not enrolled (PKI).
% Only export the CA certificate in PEM format.
% CA certificate:
-----BEGIN CERTIFICATE-----
MIIDFjCCAf6gAwIBAgIBATANBgkqhkiG9w0BAQsFADAcMRowGAYDVQQDExFyb290
Y2EubndrdC5sb2NhbDAeFw0yMTAzMTIxMjUwNTBaFw0yNDAzMTExMjUwNTBaMBwx
GjAYBgNVBAMTEXJvb3RjYS5ud2t0LmxvY2FsMIIBIjANBgkqhkiG9w0BAQEFAAOC
AQ8AMIIBCgKCAQEAnQ79KFAbXXMZmzO0yQUbrky8NFQDoj3wB2Hj4B92wpuVVPhk
m9lSHaljTV/5VarG+i4OOBLtPnzS3XTV7TyvO5bwGcCFd0EA0jDMMOLSjGiilr4R
5nO4U3gJMrtXVxn6v7LvsA1Sw5j646T4dlI4CPFElTpNXMSSRBSWlHMuoG9CLFKm
Rf53VQQlCSO3NmM/2qSBDsjbmDkoE7UMDrgZPwezdvRbJgUvH3EbwJycFUVsA+h5
oQ53YPMwq8F9DPUL85Gdi6I7mbUswTMZX/DyjCd7p8TB8bNJcESnMHaY9gDkmGuc
HfGRLcXSv0LIqQPnkyU08/X2zVaokUmHbeEmcQIDAQABo2MwYTAPBgNVHRMBAf8E
BTADAQH/MA4GA1UdDwEB/wQEAwIBhjAfBgNVHSMEGDAWgBTSwYkk0XxUfShDc060
d/h/LXehrzAdBgNVHQ4EFgQU0sGJJNF8VH0oQ3NOtHf4fy13oa8wDQYJKoZIhvcN
AQELBQADggEBAHLAkh4Tlui1wi3bu8hnM9F47pJ54uQUjC08QqyG5wb5lGX2qGZj
nnqbrI0n9avVJGltib9FJiB1vVir2EpwYAuXtFrmQQWHfwepUdwfA5UT873eWdFA
qE1YyApd4TgaQ4auQlzF5TJ8l3bWTFjZFaOmHw4DBfaN5EMOvcJv1uM5fiqGbIVp
35QkOlf5vKOCe1RgBKErOC0p0NgVM1ZiHtOrhDW5bSLSaoKPSaKoYneoLZFU8XWj
HVzK+NCyVdBnBSLRk2x4U64qZweGCUNUjiYYASoFQAB+kseiFclsAbdm/TZeUwRQ
MvHw+tv9vSmReiOTpVTllyF+VGzYdJ4Xu+I=
-----END CERTIFICATE-----
```

Example 1-3: *Exporting the CA certificate.*

vManage Configuration

When booting up the vManage for the first time, we need to define the storage device. During the vManage installation process in EVE-NG, we install an additional 100G storage HDD, which can be seen as *1) vdb* in the initial setup process.

```
viptela 20.3.3

System Initializing. Please wait to login...

vmanage login: admin
Password:
Welcome to Viptela CLI
admin connected from 127.0.0.1 using console on vmanage
You must set an initial admin password.
Password:
Re-enter password:
Available storage devices:
vdb     100GB
hdc     3GB
1) vdb
2) hdc
Select storage device to use: 1
Would you like to format vdb? (y/n): y
mke2fs 1.43.8 (1-Jan-2018)
/dev/vdb contains a ext3 file system
        last mounted on Sun Mar 14 08:10:51 2021
Creating filesystem with 26214400 4k blocks and 6553600 inodes
Filesystem UUID: 021ed637-2944-4665-a817-2248d217ae3a
Superblock backups stored on blocks:
        32768, 98304, 163840, 229376, 294912, 819200, 884736, 1605632, 2654208,
        4096000, 7962624, 11239424, 20480000, 23887872

Allocating group tables: done
Writing inode tables: done
Creating journal (131072 blocks): done
Writing superblocks and filesystem accounting information: done

Extracting vManage extra-packages
vManage Extra-Package extracted to /tmp_install/extra-packages/20.3.3/
vManage Extra-Package Extraction Complete

Broadcast message from root@vmanage (somewhere) (Sun Mar 14 14:57:53 2021):

Sun Mar 14 14:57:53 UTC 2021: The system is going down for reboot NOW!
```

Example 1-4: *vManage Initial Setup.*

Example 1-5 shows the initial configuration of vManage.

```
vmanage# sh run
system
 host-name             vmanage
 admin-tech-on-failure
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  usergroup tenantadmin
  !
  user admin
   password
$6$Stk7TwMMEy7Qi82x$j.WtL3.WseQgOhAPMtULUfaT9T5ihxYJJI.BXHJj.BzdPapd9TCElFF0M
Zm3daFrE2ClwX9DS5c0jTAASjiW8.
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
!
vpn 0
 interface eth0
  ip dhcp-client
  ipv6 dhcp-client
  no shutdown
 !
!
vpn 512
!
```

Example 1-5: *vManage Initial Configuration.*

System Information

All our control components are on the same site using the site-ID 100. The system-ip address identifies devices just like a Router Identifier (RID), it doesn't have to be routable but it has to be unique. We are using the organization name nwkt. vBond IP address in VPN 0 (Infrastructure VPN) is 10.100.100.11. As the last step, we define the time source and its VPN. Note that the VPN is just like VRF, it is a virtual routing instance. Changes are implemented and saved with the command ***commit.***

```
vmanage# conf t
Entering configuration mode terminal
vmanage(config)# system
vmanage(config-system)# site-id 100
vmanage(config-system)# system-ip 10.100.100.12
vmanage(config-system)# organization-name nwkt
vmanage(config-system)# vbond 10.100.0.11
vmanage(config-system)# ntp server 10.100.0.14
vmanage(config-server-10.100.0.14)# vpn 0
vmanage(config-server-10.100.0.14)# commit
Commit complete.
```

Example 1-6: *vManage System Information.*

VPN Configuration

VPN 0 is used for control plane connections. Interface eth 0 is used as a tunnel interface with IP address 10.100.0.12/24 attached to it. We can allow (or deny) services like ssh, dhcp, ntp, netconf, dns by listing them separately. However, I'm allowing all services with the command ***allow-service all***. The two last steps are enabling interface and assign VPN-specific default route. VPN 512 used for *Out of Band (OoB)* management connection. I'm using DHCP for IP address assignment but in the production environment, you should use a statically configured IP address.

```
vmanage(config-system)# vpn 0
vmanage(config-vpn-0)#  interface eth0
vmanage(config-interface-eth0)# ip address 10.100.0.12/24
vmanage(config-interface-eth0)# tunnel-interface
vmanage(config-tunnel-interface)# allow-service all
vmanage(config-tunnel-interface)# no shutdown
vmanage(config-tunnel-interface)# ip route 0.0.0.0/0 10.100.0.1
vmanage(config-vpn-0)# vpn 512
vmanage(config-vpn-512)#  interface eth1
vmanage(config-interface-eth1)# ip dhcp-client
vmanage(config-interface-eth1)# no shutdown
vmanage(config-interface-eth1)# ip route 0.0.0.0/0 192.168.10.1
vmanage(config-vpn-512)# commit
Commit complete.
```

Example 1-7: *VPN0 and VPN512 Configuration of vManage.*

The example below shows the interface eth1 IP address setting and its operational status as well as other interface-related information.

```
vmanage# sh int eth1
interface vpn 512 interface eth1 af-type ipv4
 ip-address      192.168.10.34/24
 if-admin-status Up
 if-oper-status  Up
 encap-type      null
 port-type       mgmt
 hwaddr          50:00:00:04:00:01
 speed-mbps      1000
 duplex          full
 uptime          0:00:01:53
 rx-packets      500
 tx-packets      23
```

Example 1-8: *Interface IP Address Verification.*

Certification enrollment

After attaching the IP address to the management interface we can log on to vManage. We are using the default username/password combination admin/admin.

Figure 1-2: *Log in to vManage.*

Navigate to the *Administration/Settings* window. Before the actual certification enrollment process, we fill in the organization name and the vBond IP address.

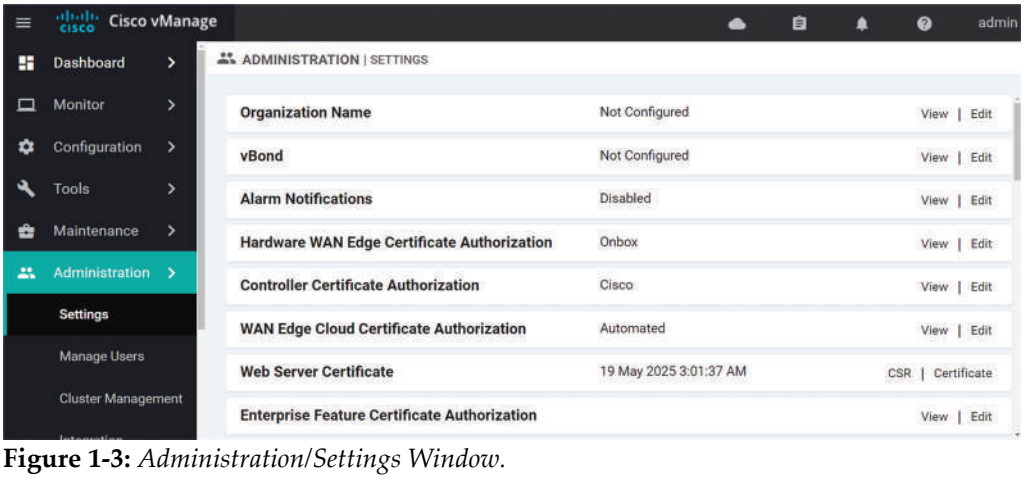


Figure 1-3: *Administration/Settings Window.*

Choose the Edit from *Organization Name* row and apply changes if they are blank (we specify an organization in the initial setup via CLI). Click the *Save* button to commit changes.

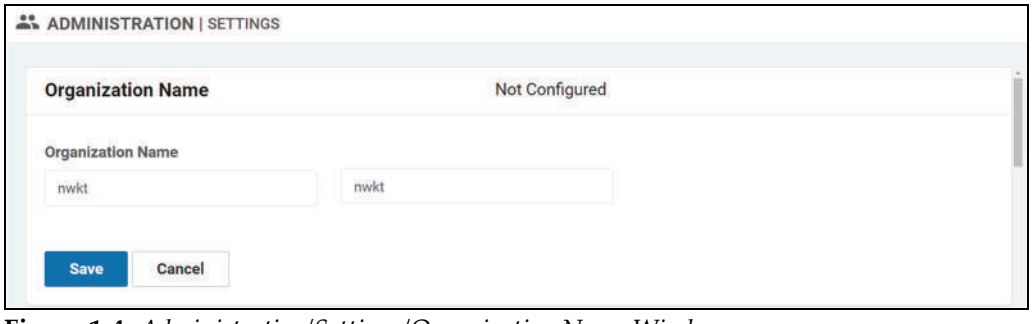


Figure 1-4: *Administration/Settings/Organization Name Window.*

Next, configure the IP address of the vBond node changes if it is not shown here (we also specify an IP address of vBond in the initial setup process). Click the *Save* button to commit changes.

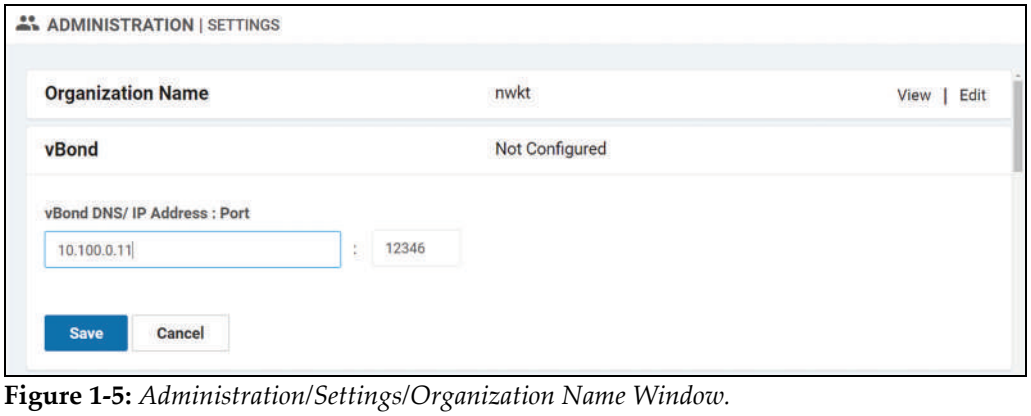


Figure 1-5: *Administration/Settings/Organization Name Window.*

After adding the Organization Name and vBond IP address we are good to go to the certificate enrollment process. Select the *Edit* option for the *Controller certificate Authorization* row. The *Certification Signed by* field has a default value *Cisco Automated (Recommended)*. Select the *Enterprise Root Certificate* option. Copy the root certificate from the IOS-XE certificate Server and paste it to the Certificate field. Fill in Doman Name, Organization Unit, Organization, City, State, Email, and Country Code fields and select the Validity time. Click the *Import & Save* button.

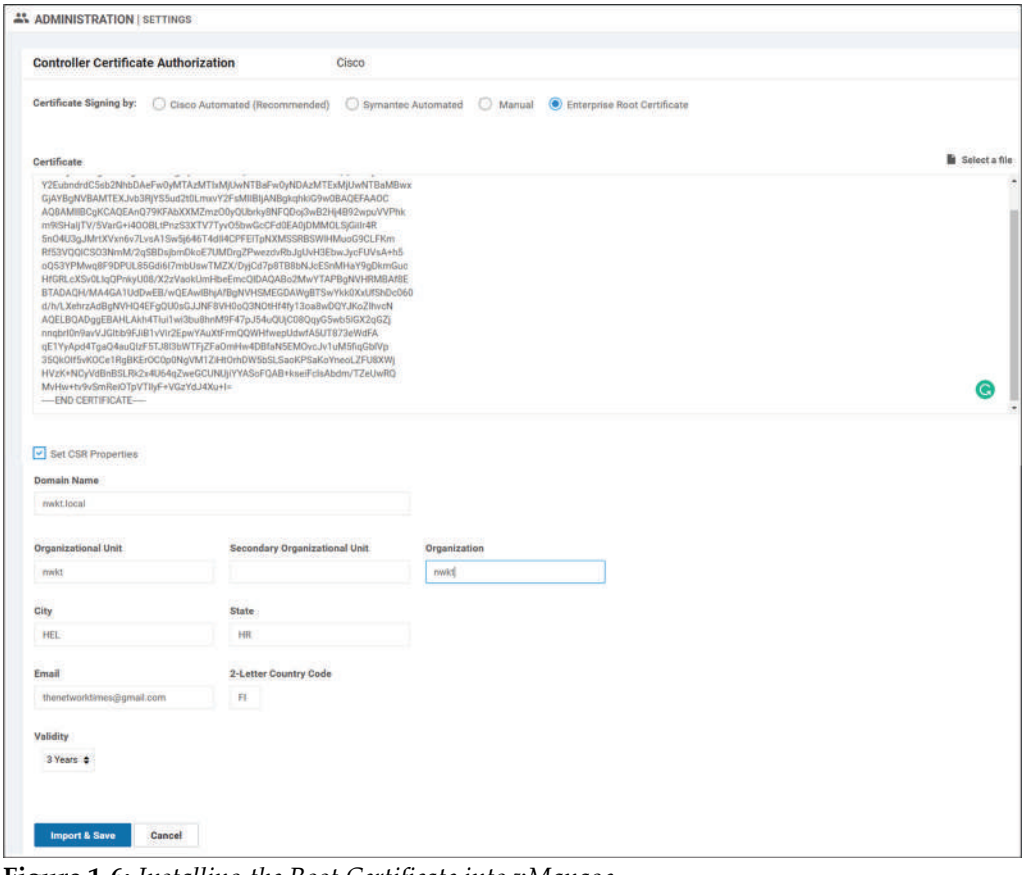


Figure 1-6: *Installing the Root Certificate into vManage.*

You will get a warning message that you can accept by clicking the Proceed button.

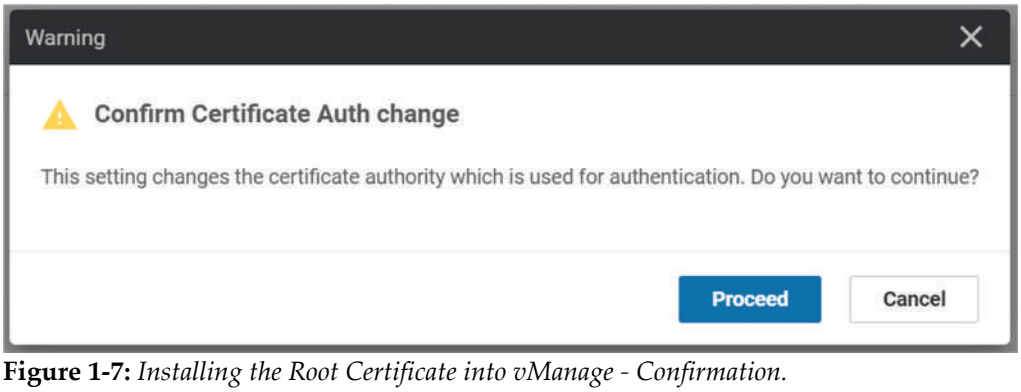


Figure 1-7: *Installing the Root Certificate into vManage - Confirmation.*

The figure below verifies our configuration changes.

Figure 1-8: *Verification.*

When the root certificate is installed, we generate a *Certificate Signing Request (CSR)*. Navigate *to Configuration > Certificates* window and select *Controllers* sheet. Select […] at the end of the vManage row and choose the *Generate CSR* option.

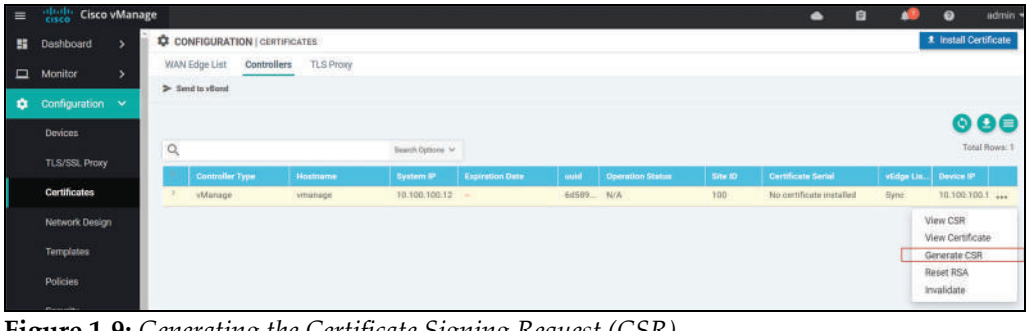


Figure 1-9: *Generating the Certificate Signing Request (CSR).*

Figure 1-10 shows the generated CSR. Copy it to the clipboard and click the Close button.

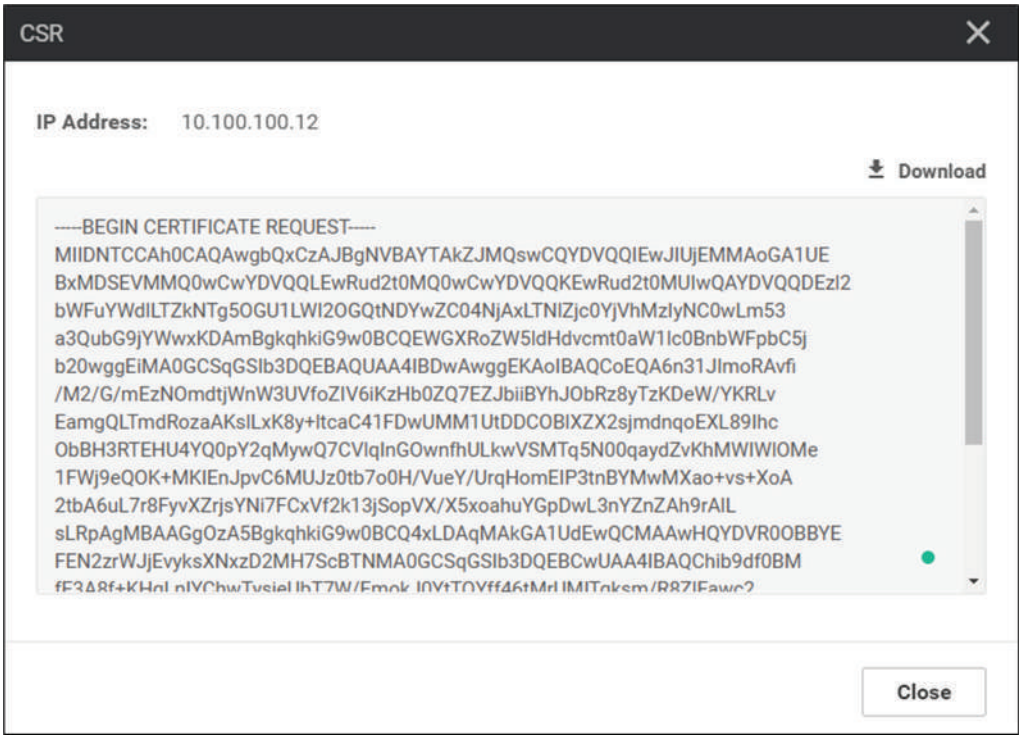


Figure 1-10: *Generating the Certificate Signing Request (CSR).*

Login to IOS-XE certificate Server. Type the command ***crypto pki server PKI request pkcs10 terminal*** and press enter. Now paste the CSR into the terminal window. After that type quit. In case your prompt stays at the end of line "*-----END CERTIFICATE REQUEST-----*" you need to press enter before typing the command ***quit***. Copy the vManage Specific Granted Certificate into the clipboard.

```
CA-Server#crypto pki server PKI request pkcs10 terminal
PKCS10 request in base64 or pem

% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIDNTCCAh0CAQAwgbQxCzAJBgNVBAYTAkZJMQswCQYDVQQIEwJIUjEMMAoGA1UE
BxMDSEVMMQ0wCwYDVQQLEwRud2t0MQ0wCwYDVQQKEwRud2t0MUIwQAYDVQQDEzl2
bWFuYWdlLTZkNTg5OGU1LWI2OGQtNDYwZC04NjAxLTNlZjc0YjVhMzIyNC0wLm53
a3QubG9jYWwxKDAmBgkqhkiG9w0BCQEWGXRoZW5ldHdvcmt0aW1lc0BnbWFpbC5j
b20wggEiMA0GCSqGSIb3DQEBAQUAA4IBDwAwggEKAoIBAQCoEQA6n31JImoRAvfi
```

```
/M2/G/mEzNOmdtjWnW3UVfoZIV6iKzHb0ZQ7EZJbiiBYhJObRz8yTzKDeW/YKRLv
EamgQLTmdRozaAKslLxK8y+ltcaC41FDwUMM1UtDDCOBlXZX2sjmdnqoEXL89Ihc
ObBH3RTEHU4YQ0pY2qMywQ7CVlqInGOwnfhULkwVSMTq5N00qaydZvKhMWIWlOMe
1FWj9eQOK+MKIEnJpvC6MUJz0tb7o0H/VueY/UrqHomEIP3tnBYMwMXao+vs+XoA
2tbA6uL7r8FyvXZrjsYNi7FCxVf2k13jSopVX/X5xoahuYGpDwL3nYZnZAh9rAlL
sLRpAgMBAAGgOzA5BgkqhkiG9w0BCQ4xLDAqMAkGA1UdEwQCMAAwHQYDVR0OBBYE
FEN2zrWJjEvyksXNxzD2MH7ScBTNMA0GCSqGSIb3DQEBCwUAA4IBAQChib9df0BM
fE3A8f+KHqLnIYChwTysjeUbT7W/EmokJ0YtTQYff46tMrUMITqksm/R8ZlFawc2
f92OkMKdfeyQvuo37+oJX6CBdmvwqRWT7AOT64qADDxIOuye52H35APbfaQpm5Me
dLy98oqrDrD66ExRFq7FXYnqCRxGLaL263/HSlSroLGYwlwc7ED4aiuoW1NKI/or
BP0aLnbQl2GPJ5cwJrPW6Np9wqrosXzCrvHQqzFdTRRpfFZPMHhoMoWVo8YCwVnR
hiwO0EqnAN/NVBWcuhsENph5VEMUalPRFJ1rMQ3bG40gePjWDsV15IhW1EN/m0l6
dj8u9qiwaGoz
-----END CERTIFICATE REQUEST-----
quit
% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIDnzCCAoegAwIBAgIBCDANBgkqhkiG9w0BAQsFADAcMRowGAYDVQQDExFyb290
Y2EubndrdC5sb2NhbDAeFw0yMTAzMTQxNTM2NDVaFw0yMjAzMTQxNTM2NDVaMIG0
MQswCQYDVQQGEwJGSTELMAkGA1UECBMCSFIxDDAKBgNVBAcTA0hFTDENMAsGA1UE
CxMEbndrdDENMAsGA1UEChMEbndrdDFCMEAGA1UEAxM5dm1hbmFnZS02ZDU4OThl
NS1iNjhkLTQ2MGQtODYwMS0zZWY3NGI1YTMyMjQtMC5ud2t0LmxvY2FsMSgwJgYJ
KoZIhvcNAQkBFhl0aGVuZXR3b3JrdGltZXNAZ21haWwuY29tMIIBIjANBgkqhkiG
9w0BAQEFAAOCAQ8AMIIBCgKCAQEAqBEAOp99SSJqEQL34vzNvxv5hMzTpnbY1p1t
1FX6GSFeoisx29GUOxGSW4ogWISTm0c/Mk8yg3lv2CkS7xGpoEC05nUaM2gCrJS8
SvMvpbXGguNRQ8FDDNVLQwwjgZV2V9rI5nZ6qBFy/PSIXDmwR90UxB1OGENKWNqj
MsEOwlZaiJxjsJ34VC5MFUjE6uTdNKmsnWbyoTFiFpTjHtRVo/XkDivjCiBJyabw
ujFCc9LW+6NB/1bnmP1K6h6JhCD97ZwWDMDF2qPr7Pl6ANrWwOri+6/Bcr12a47G
DYuxQsVX9pNd40qKVV/1+caGobmBqQ8C952GZ2QIfawJS7C0aQIDAQABo1MwUTAP
BgNVHRMBAf8EBTADAQH/MB8GA1UdIwQYMBaAFNLBiSTRfFR9KENzTrR3+H8td6Gv
MB0GA1UdDgQWBBRDds61iYxL8pLFzccw9jB+0nAUzTANBgkqhkiG9w0BAQsFAAOC
AQEAEOSIOqdA4p9rejub2wzJWlgagDy/vCnmvvuOwO7Nz177ktzP82KX6VUe+PSY
P8ArXlAbH0YNmMl0ukZVD3xb3q5q0YKGzxt4JwLSEbqn7BMY6DsGhkrnJp/5WYVQ
dhjX1JQbvzjGXjX+cYnqCKLwMIPiwRk/vrUxJzS6bfxPUHpiXpWX8VnsxIX49iU7
WW3CI1SqyABfhAwrrmXo7fyFTk+ng6FbcNTj69G7ciLZnV7ckVQYH8CYKgwxEezD
xr7ogUIRAbZqn3/J/s6fLsr6MQ8xlVuG7+adZlLLeSYnbjjR/AublSWS0oOLLzN3
vRqA519Xr6m5CsqWcgLu4sXOAw==
-----END CERTIFICATE-----

CA-Server#
```

Example 1-9: *Generating Granted Certificate from the Certificate Signing Request.*

Go back to the vManage management window and navigate to *Configuration* > *Certificates* window and select *Controller* sheet. Click the *Install Certificate* button.

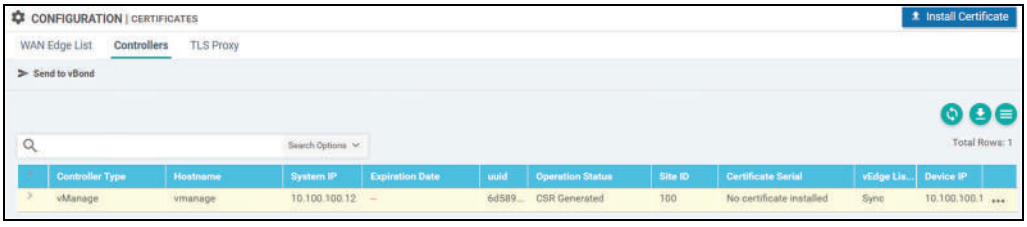


Figure 1-11: *Installing the Granted Certificate into vManage.*

Paste the Granted Certificate into the Install *Certificate Text* in the *Install Certificate* window and click the *Install* button.

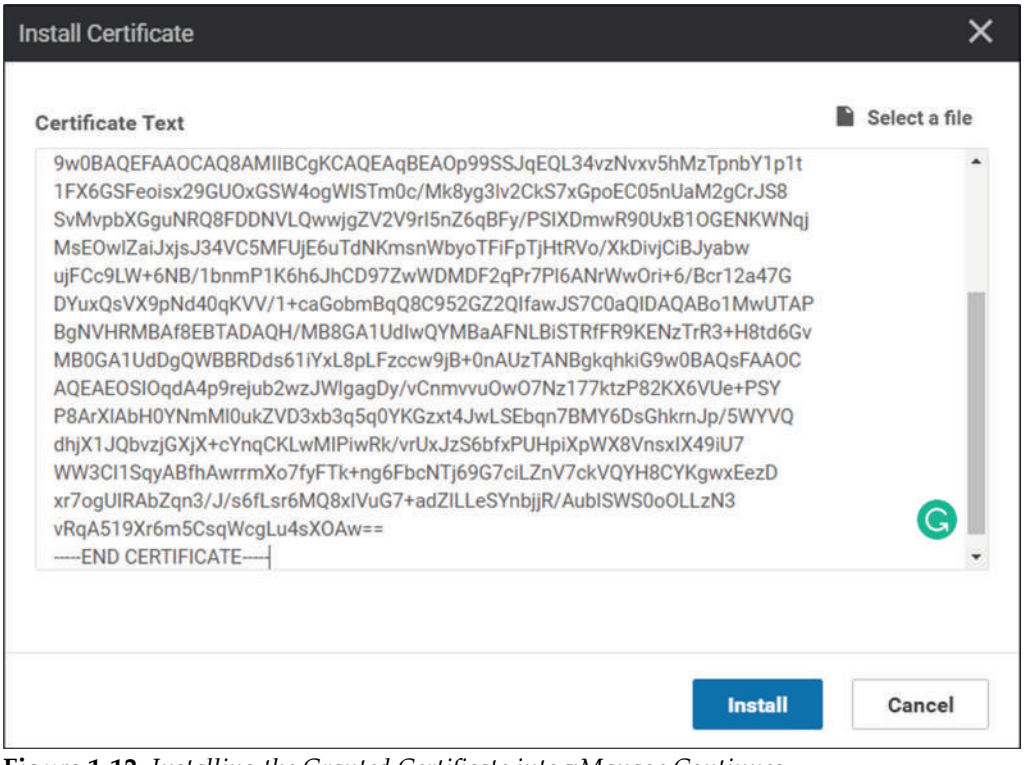


Figure 1-12: *Installing the Granted Certificate into vManage Continues.*

The figure below illustrates the installation progress and its result which is Success. At this phase the vManage certification process is ready.

Figure 1-13: *The Granted Certificate Installation Progress.*

We can verify certificate status from the *Controllers* sheet in *Configuration* > *Devices* windows. From there we can see that the *Certificate Status* column is *Installed*.

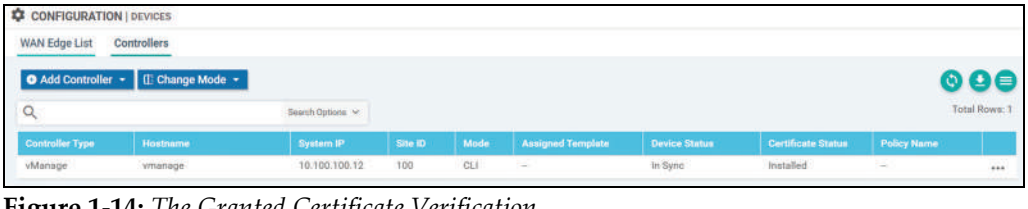


Figure 1-14: *The Granted Certificate Verification.*

vBond Initial Configuration

vBond initial configuration is shown in the example below. The very first configuration step is to provide a password for the admin user account.

```
viptela 20.3.2

vedge login: admin
Password:
Welcome to Viptela CLI
admin connected from 127.0.0.1 using console on vedge
You must set an initial admin password.
Password:
Re-enter password:
vedge# sh run
system
 host-name                vedge
 admin-tech-on-failure
```

```
 no route-consistency-check
 vbond ztp.viptela.com
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  usergroup tenantadmin
  !
  user admin
   password
$6$meJoeGPXSl/jhFSK$hwZ5Uo3pLhiRLm9AD7eRAutUkRknCLgJyrhzFdJ74qaZyYS3
yoKP2xrBwAo6slmRrOUiuI9ejmk9LmftljbIp/
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
!
omp
 no shutdown
 graceful-restart
 advertise connected
 advertise static
!
security
 ipsec
  authentication-type ah-sha1-hmac sha1-hmac
 !
!
vpn 0
 interface ge0/0
```

```
  ip dhcp-client
  ipv6 dhcp-client
  tunnel-interface
   encapsulation ipsec
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
   allow-service https
  !
  no shutdown
 !
!
vpn 512
 interface eth0
  ip dhcp-client
  ipv6 dhcp-client
  no shutdown
 !
!
vedge#
```

Example 1-10: *The Initial Configuration of vBond.*

System Information

There are only two differences in the system configuration of vBond compared to vManage. The system-ip is 10.100.100.11 and the vbond configuration is defined as local. Other than these everything else is common with vManage system configuration. Note that vBond uses the same image as vEdge and that is why the prompt is *vedge* by default.

```
vedge# conf t
Entering configuration mode terminal
vedge(config)# system
vedge(config-system)# host-name vbond
vedge(config-system)# site-id 100
vedge(config-system)# system-ip 10.100.100.11
vedge(config-system)# organization-name nwkt
vedge(config-system)# vbond 10.100.0.11 local
vedge(config-system)# ntp server 10.100.0.14
vedge(config-server-10.100.0.14)# vpn 0
```

Example 1-11: *The System Configuration of vBond.*

VPN Configuration

The VPN configuration follows the same procedure as what we did with vManage. We attached interfaces to VPN 0 and VPN 512 and give IP addresses and the VPN-specific gateways. The vEdge image is used for vBond devices and that is why the interface naming is different. In the vManage, we attached the interface eth0 to VPN 0 while in vBond we are going to use interface ge0/0 instead. Note that interface ge0/0 has to be a non-tunnel interface, that is why there is a command no tunnel-interface.

```
vedge(config-system)# vpn 0
vedge(config-vpn-0)#  interface ge0/0
vedge(config-interface-ge0/0)# ip address 10.100.0.11/24
vedge(config-interface-ge0/0)# no tunnel-interface
vedge(config-interface-ge0/0)# no shut
vedge(config-interface-ge0/0)# ip route 0.0.0.0/0 10.100.0.1
vedge(config-vpn-0)#
vedge(config-vpn-0)# vpn 512
vedge(config-vpn-512)# vpn 512
vedge(config-vpn-512)#  interface eth0
vedge(config-interface-eth0)# ip dhcp-client
vedge(config-interface-eth0)# no shut
vedge(config-interface-eth0)# ip route 0.0.0.0/0 192.168.10.1
vedge(config-vpn-512)# commit
Commit complete.
```

Example 1-12: *VPN0 and VPN512 Configuration of vBond.*

The example below shows the interface eth0 IP address setting and its operational status as well as other interface-related information. Note that after committing changes, the host name is vbond.

```
vbond# sh int eth0
interface vpn 512 interface eth0 af-type ipv4
 ip-address        192.168.10.33/24
 if-admin-status   Up
 if-oper-status    Up
 if-tracker-status NA
 encap-type        null
 port-type         service
 mtu               1500
 hwaddr            50:00:00:01:00:00
 speed-mbps        1000
 duplex            full
 tcp-mss-adjust    1416
 uptime            0:01:03:35
 rx-packets        5840
 tx-packets        389
```

Example 1-13: *The IP Address Verification of vBond.*

Certification enrollment

There couple of ways to install the root certificate into vBond. I'm using the WinSCP application for that. First, you need to copy the root certificate from the IOS-XE Certificate Server as shown in the example 1-3. Then paste it to notepad or your favorite text editor, and save it as PKI.pem. After that, copy the file into the vBond directory */home/admin*.

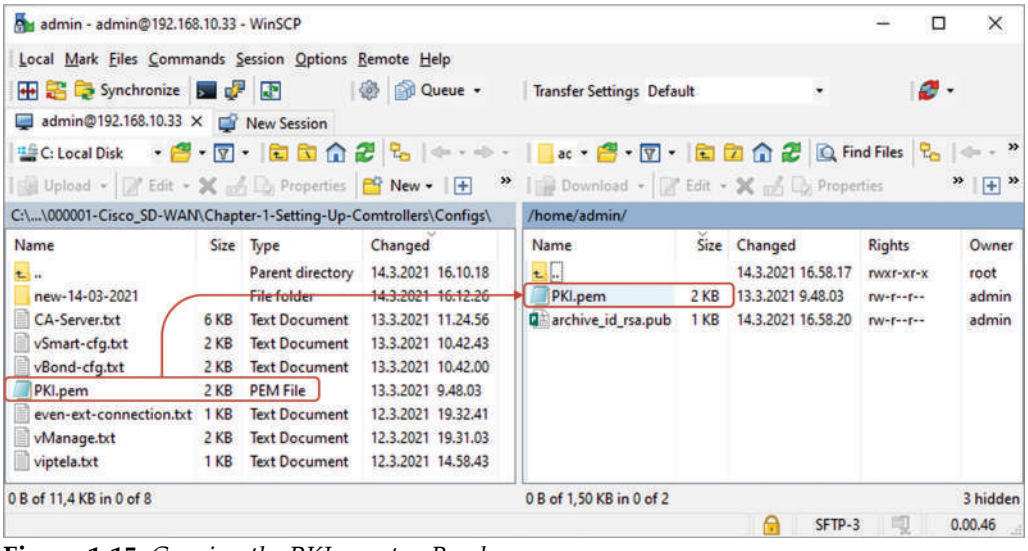


Figure 1-15: *Copying the PKI.pem to vBond.*

After copying the file into the */home/admin* directory, it is installed by using the command ***request root-cert-chain install /home/admin/PKI.pem***.

```
vbond# request root-cert-chain install /home/admin/PKI.pem
Uploading root-ca-cert-chain via VPN 0
Copying ... /home/admin/PKI.pem via VPN 0
Updating the root certificate chain..
Successfully installed the root certificate chain
vbond#
```

Example 1-14: *Installing the Root certificate into vBond.*

Next, we will add vBond to the controller list in the vManage management console. vBond can be added to the controller list by navigating to *Configuration* > *Devices* window and from there by selecting the *Controller* sheet. Select vBond from the *Add Controller* drop-down menu.

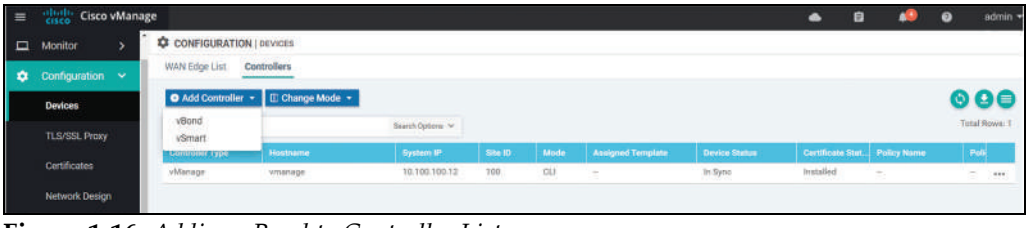


Figure 1-16: *Adding vBond to Controller List.*

Give the vBond IP address and user credentials, check the *Generate CSR* box and click the *Add* button to commit changes.

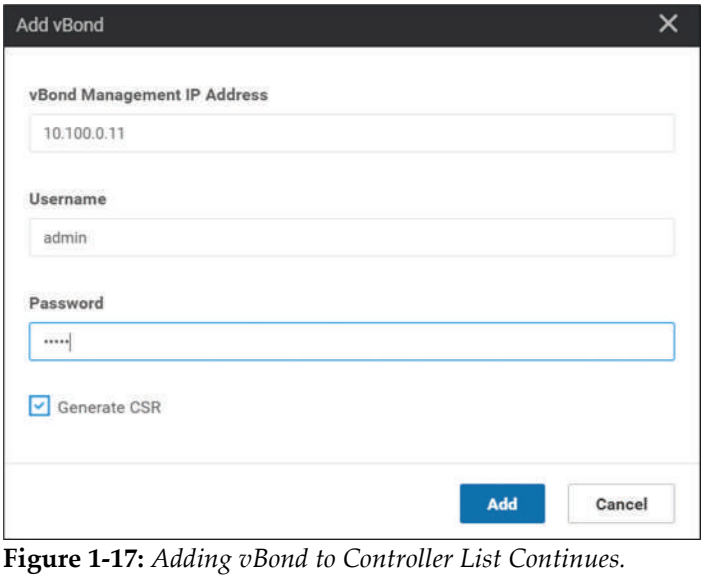


Figure 1-17: *Adding vBond to Controller List Continues.*

The figure below shows that vBond is now listed in the controller list.

Figure 1-18: *Adding vBond to Controller List Continues.*

After installing the root certificate, navigate *to the Configuration > Certificates* window and select the *Controllers* sheet. Select [...] at the end of the vBond row and choose the *View CSR* option.

Figure 1-19: *Adding vBond to Controller List Continues.*

Copy the CSR into the clipboard.

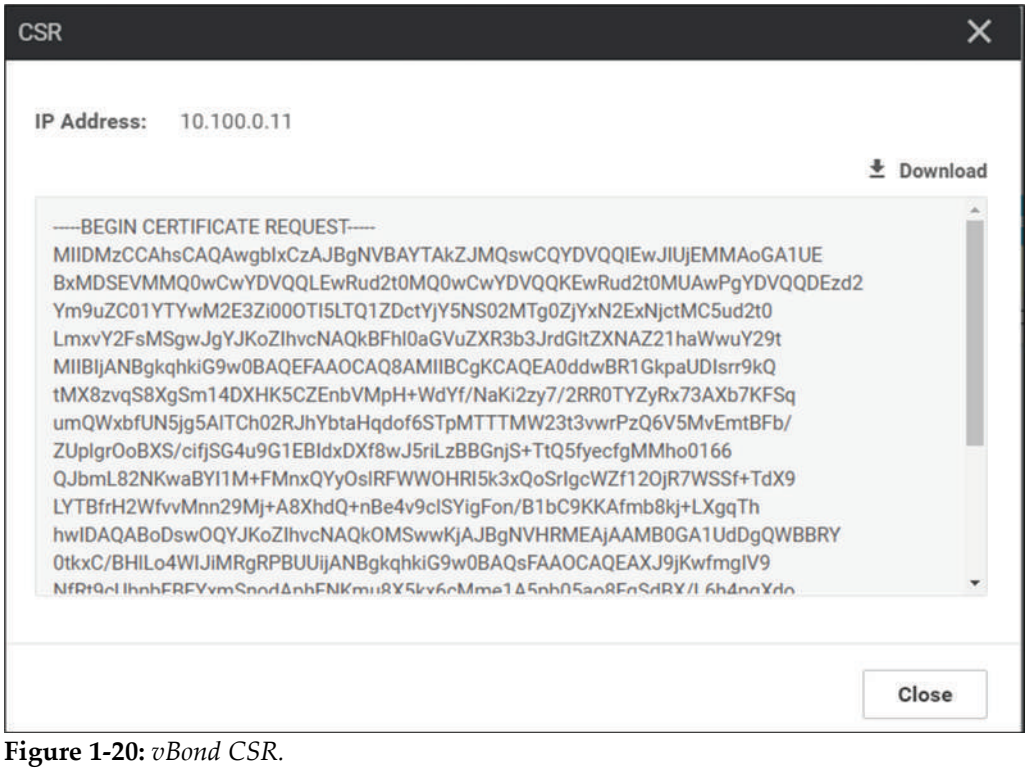


Figure 1-20: *vBond CSR.*

Login to IOS-XE certificate Server. Type the command ***crypto pki server PKI request pkcs10 terminal*** and press enter. Now paste the vBond CSR into the terminal window. After that type quit. In case your prompt stays at the end of line *"-----END CERTIFICATE REQUEST-----"* you need to press enter before typing the command ***quit***. Copy the vManage Specific Granted Certificate into the clipboard. This is the same process we used with vManage.

```
CA-Server#crypto pki server PKI request pkcs10 terminal
PKCS10 request in base64 or pem

% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIDMzCCAhsCAQAwgbIxCzAJBgNVBAYTAkZJMQswCQYDVQQIEwJIUjEMMAoGA1UE
BxMDSEVMMQ0wCwYDVQQLEwRud2t0MQ0wCwYDVQQKEwRud2t0MUAwPgYDVQQDEzd2
Ym9uZC01YTYwM2E3Zi00OTI5LTQ1ZDctYjY5NS02MTg0ZjYxN2ExNjctMC5ud2t0
LmxvY2FsMSgwJgYJKoZIhvcNAQkBFhl0aGVuZXR3b3JrdGltZXNAZ21haWwuY29t
MIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA0ddwBR1GkpaUDIsrr9kQ
tMX8zvqS8XgSm14DXHK5CZEnbVMpH+WdYf/NaKi2zy7/2RR0TYZyRx73AXb7KFSq
umQWxbfUN5jg5AlTCh02RJhYbtaHqdof6STpMTTTMW23t3vwrPzQ6V5MvEmtBFb/
ZUplgrOoBXS/cifjSG4u9G1EBIdxDXf8wJ5riLzBBGnjS+TtQ5fyecfgMMho0166
QJbmL82NKwaBYI1M+FMnxQYyOslRFWWOHRI5k3xQoSrIgcWZf12OjR7WSSf+TdX9
LYTBfrH2WfvvMnn29Mj+A8XhdQ+nBe4v9clSYigFon/B1bC9KKAfmb8kj+LXgqTh
hwIDAQABoDswOQYJKoZIhvcNAQkOMSwwKjAJBgNVHRMEAjAAMB0GA1UdDgQWBBRY
0tkxC/BHlLo4WIJiMRgRPBUUijANBgkqhkiG9w0BAQsFAAOCAQEAXJ9jKwfmgIV9
NfRt9cUbnbFBFYxmSnodAphFNKmu8X5kx6cMme1A5pb05ao8FgSdBX/L6h4pqXdo
cVHY1lSzwJ8HfXgSwKJPmZOtYLUotXPKZAe9d+pJx+nrQdq/gZ8VEvMcO7z9pelu
qIp1o1fAd/ZADLuHYiQGjzdejtYnfTt5uMc/3puGI2k4J+YKZjlBthCz8QouHOT0
Q40X3S5AUC8Ts36nGePUH8W4pw5OBJjmaNtzIVWMpo4MLcxlkVmwRjwDvz6AxSAO
hgh4rxuon+SLPlE74TVZAThXVAXiktLABxN/4m5YREJIDLPxtLCAGl+jwfPX6XTR
Fsqn5FJj4g==
-----END CERTIFICATE REQUEST-----
quit
% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIDnTCCAoWgAwIBAgIBCTANBgkqhkiG9w0BAQsFADAcMRowGAYDVQQDExFyb290
Y2EubndrdC5sb2NhbDAeFw0yMTAzMTQxNjEzNTZaFw0yMjAzMTQxNjEzNTZaMIGy
MQswCQYDVQQGEwJGSTELMAkGA1UECBMCSFIxDDAKBgNVBAcTA0hFTDENMAsGA1UE
CxMEbndrdDENMAsGA1UEChMEbndrdDFAMD4GA1UEAxM3dmJvbmQtNWE2MDNhN2Yt
NDkyOS00NWQ3LWI2OTUtNjE4NGY2MTdhMTY3LTAubndrdC5sb2NhbDEoMCYGCSqG
SIb3DQEJARYZdGhlbmV0d29ya3RpbWVzQGdtYWlsLmNvbTCCASIwDQYJKoZIhvcN
AQEBBQADggEPADCCAQoCggEBANHXcAUdRpKWlAyLK6/ZELTF/M76kvF4EpteA1xy
uQmRJ21TKR/lnWH/zWiots8u/9kUdE2Gckce9wF2+yhUqrpkFsW31DeY4OQJUwod
NkSYWG7Wh6naH+kk6TE00zFtt7d78Kz80OleTLxJrQRW/2VKZYKzqAV0v3In40hu
LvRtRASHcQ13/MCea4i8wQRp40vk7UOX8nnH4DDIaNNeukCW5i/NjSsGgWCNTPhT
J8UGMjrJURVljh0SOZN8UKEqyIHFmX9djo0e1kkn/k3V/S2EwX6x9ln77zJ59vTI
/gPF4XUPpwXuL/XJUmIoBaJ/wdWwvSigH5m/JI/i14Kk4YcCAwEAAaNTMFEwDwYD
VR0TAQH/BAUwAwEB/zAfBgNVHSMEGDAWgBTSwYkk0XxUfShDc060d/h/LXehrzAd
BgNVHQ4EFgQUWNLZMQvwR5S6OFiCYjEYETwVFIowDQYJKoZIhvcNAQELBQADggEB
AGgygNkmi0oTX+8kqfrn88ydNSpK5kgPjMzDn0ktQ7KtLVHutGVetdDTA6c8OARi
```

```
sybjMwLcuMZxahJ5H62FPqQEZ0MPol7qm6We4DOOftLbPvppTP8hTfLj4/R2Aqxq
KJ9andY/K8vkuv/NH+sgdiNtyNlx+hY5dZjDYDXzVxSntoL3aD+qMWHCPTKDpg5/
QT+/9u76wX5wMVfKY3+XzcGmDqIyDxsqLn66w3ADcLshOjXGokxwtDvzKPpAXU1e
XPxZJsM5LEd4+VxlcKsswkX385KKDVj2mYYm7uxB8Tlt71hlsHKaBBiIeUq9ai69
2HOp/VzmfXqWsQ/anUWxDG0=
-----END CERTIFICATE-----

CA-Server#
```

Example 1-15: *Generating the Granted Certificate for vBond.*

Go back to the vManage management window and navigate to *Configuration* > *Certificates* window and select *Controller* sheet. Select the vBond row and click the *Install Certificate* button.

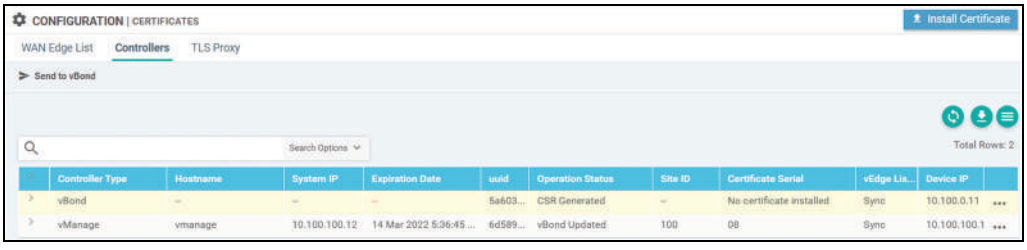


Figure 1-21: *Installing the Granted Certificate into vBond.*

Paste the Granted Certificate into the Install *Certificate Text* in the *Install Certificate* window and click the *Install* button.

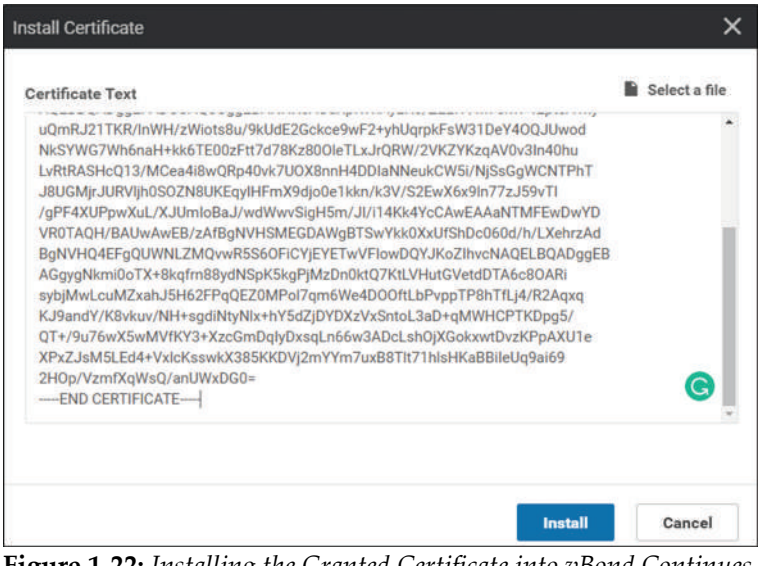


Figure 1-22: *Installing the Granted Certificate into vBond Continues.*

The figure below illustrates the installation progress and its result which is Success. At this phase, the vBond certification process is done.

Figure 1-23: *The Granted Certificate Installation Progress.*

We can verify certificate status from the *Controllers* sheet in *Configuration* > *Devices* windows. From there we can see that the *Certificate Status* column is *Installed*.

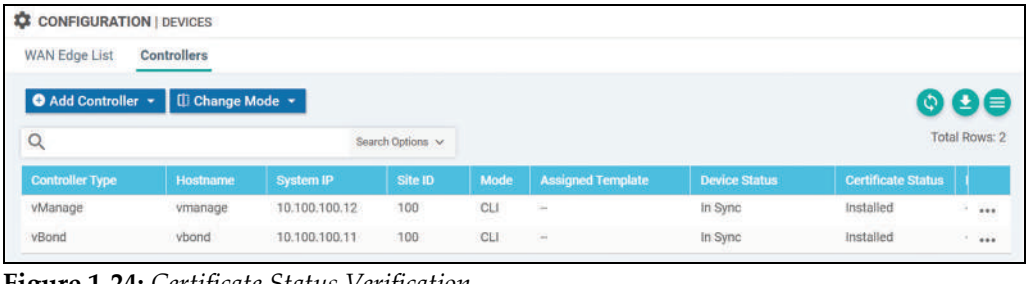


Figure 1-24: *Certificate Status Verification.*

vSmart Initial Configuration

Example 1-16 illustrates the default configuration of vSmart.

```
vsmart# sh run
system
 host-name             vsmart
 admin-tech-on-failure
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  usergroup tenantadmin
  !
  user admin
   password
$6$jKzSSqC2GCJveJV4$VxMCv59Qv2J.lDd2luqXXJ9dUuv3izVKXPEbE3b43AAry3n6ptI7DqunO
0y0TzxaUVRGAUZ7E/ySEiWdyt8/60
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
!
omp
 no shutdown
 graceful-restart
!
vpn 0
```

```
 interface eth0
  ip dhcp-client
  ipv6 dhcp-client
  no shutdown
 !
!
vpn 512
```

Example 1-16: *The Initial Configuration of vSmart.*

System Information

As with vManage and vBond, we set the system settings. It is just like what we did with vManage, the only difference is the unique system-ip.

```
vsmart# conf t
Entering configuration mode terminal
vsmart(config)# system
vsmart(config-system)# site-id 100
vsmart(config-system)# system-ip 10.100.100.13
vsmart(config-system)# organization-name nwkt
vsmart(config-system)# vbond 10.100.0.11
vsmart(config-system)# ntp server 10.100.0.14
vsmart(config-server-10.100.0.14)# vpn 0
```

Example 1-17: *The System Configuration of vSmart.*

VPN Configuration

VPN configuration follows the same process as what we did with vManage and vBond.

```
vsmart(config-system)# vpn 0
vsmart(config-vpn-0)#  interface eth0
vsmart(config-interface-eth0)# ip address 10.100.0.13/24
vsmart(config-interface-eth0)# tunnel-interface
vsmart(config-tunnel-interface)# allow-service all
vsmart(config-tunnel-interface)# no shutdown
vsmart(config-tunnel-interface)# ip route 0.0.0.0/0 10.100.0.1
vsmart(config-vpn-0)# vpn 512
vsmart(config-vpn-512)# interface eth1
vsmart(config-interface-eth1)# ip dhcp-client
vsmart(config-interface-eth1)# no shutdown
vsmart(config-interface-eth1)# commit
Commit complete.
```

Example 1-18: *VPN0 and VPN512 Configuration of vSmart.*

The example below shows the interface eth1 IP address setting and its operational status as well as other interface-related information.

```
vsmart# sh int eth1
interface vpn 512 interface eth1 af-type ipv4
 ip-address      192.168.10.35/24
 if-admin-status Up
 if-oper-status  Up
 encap-type      null
 port-type       mgmt
 hwaddr          50:00:00:03:00:01
 speed-mbps      1000
 duplex          full
 uptime          0:00:09:35
 rx-packets      2208
 tx-packets      97
```

Example 1-19:*The IP Address Verification of vBond.*

Certification enrollment

The root certificate installation into vSmart is the same as what was explained with vBond. I'm using the WinSCP to copy the root certificate into the vSmart directory */home/admin*.

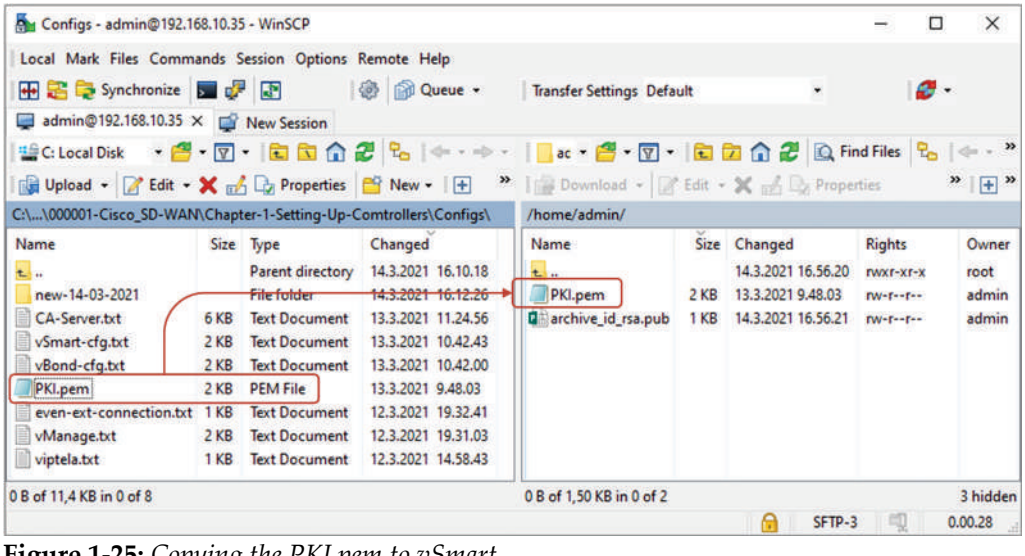


Figure 1-25: *Copying the PKI.pem to vSmart.*

After copying the file into the */home/admin* directory, it is installed by using the command ***request root-cert-chain install /home/admin/PKI.pem***.

```
vsmart# request root-cert-chain install /home/admin/PKI.pem
Uploading root-ca-cert-chain via VPN 0
Copying ... /home/admin/PKI.pem via VPN 0
Updating the root certificate chain..
Successfully installed the root certificate chain
vsmart#
```

Example 1-20: *Installing the Root certificate into vSmart.*

Next, we will add vSmart to the controller list in the vManage management console. vSmart can be added to the controller list by navigating to *Configuration > Devices* window and from there by selecting the *Controller* sheet. Select vSmart from the *Add Controller* drop-down menu.

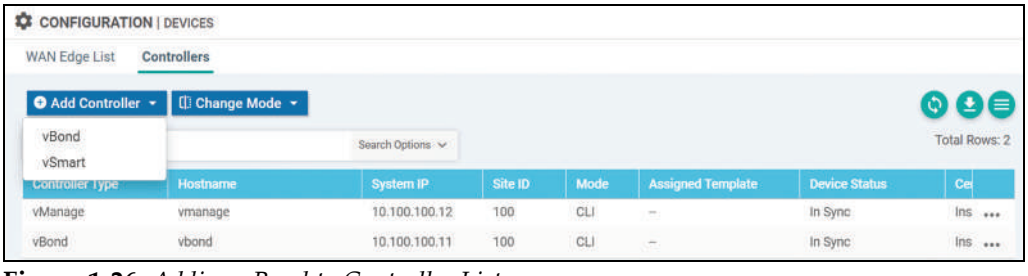


Figure 1-26: *Adding vBond to Controller List.*

Give the vSmart IP address and user credentials, check the *Generate CSR* box and click the *Add* button to commit changes.

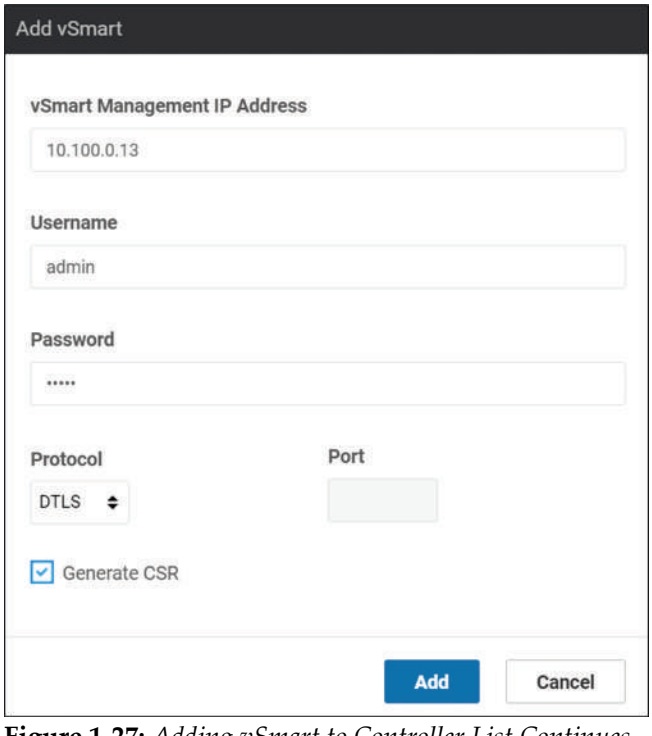


Figure 1-27: *Adding vSmart to Controller List Continues.*

The figure below shows that vSmart is now listed in the controller list.

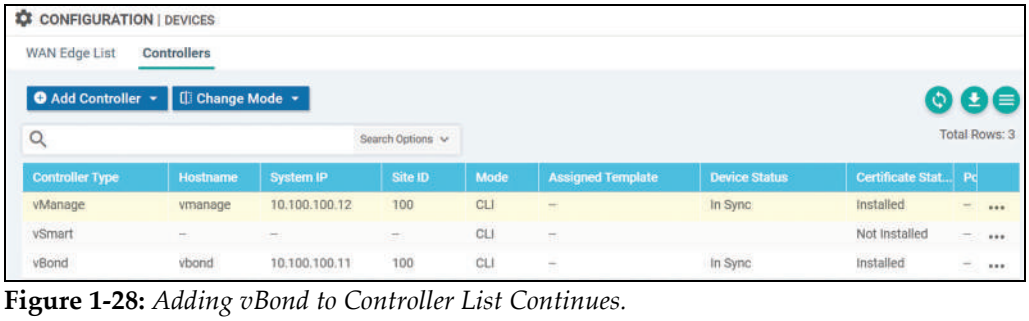


Figure 1-28: *Adding vBond to Controller List Continues.*

After installing the root certificate, navigate *to the Configuration > Certificates* window and select the *Controllers* sheet. Select *Option* menu [**...**] at the end of the vSmart row and choose the *View CSR* option.

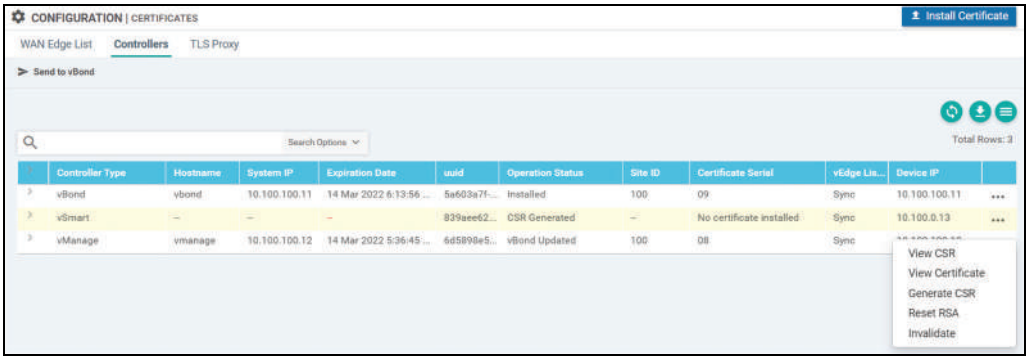


Figure 1-29: *Adding vSmart to Controller List Continues.*

Copy the CSR into the clipboard.

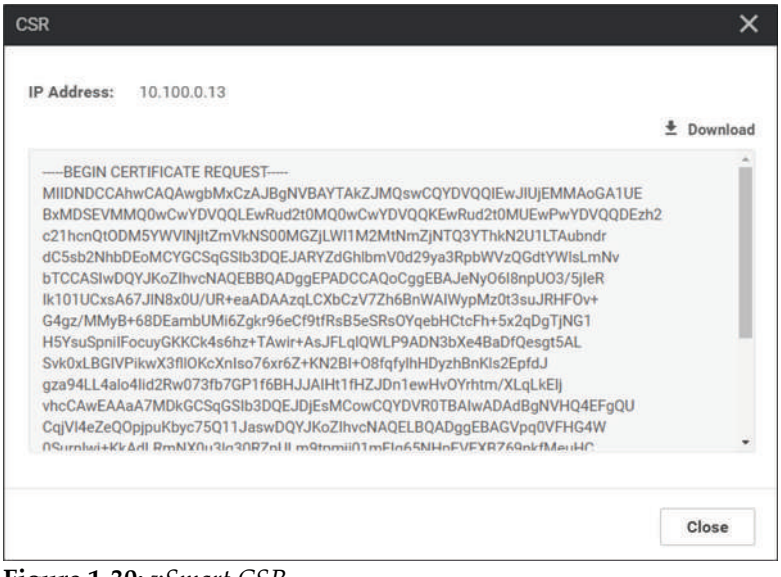


Figure 1-30: *vSmart CSR.*

Login to IOS-XE certificate Server. Type the command ***crypto pki server PKI request pkcs10 terminal*** and press enter. Now paste the vSmart CSR into the terminal window. After that type quit. In case your prompt stays at the end of line *"-----END CERTIFICATE REQUEST-----"* you need to press enter before typing the command ***quit***. Copy the vManage Specific Granted Certificate into the clipboard.

```
CA-Server#crypto pki server PKI request pkcs10 terminal
PKCS10 request in base64 or pem
% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIDNDCCAhwCAQAwgbMxCzAJBgNVBAYTAkZJMQswCQYDVQQIEwJIUjEMMAoGA1UE
BxMDSEVMMQ0wCwYDVQQLEwRud2t0MQ0wCwYDVQQKEwRud2t0MUEwPwYDVQQDEzh2
c21hcnQtODM5YWVlNjItZmVkNS00MGZjLWI1M2MtNmZjNTQ3YThkN2U1LTAubndr
dC5sb2NhbDEoMCYGCSqGSIb3DQEJARYZdGhlbmV0d29ya3RpbWVzQGdtYWlsLmNv
bTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAJeNyO6I8npUO3/5jIeR
Ik101UCxsA67JlN8x0U/UR+eaADAAzqLCXbCzV7Zh6BnWAIWypMz0t3suJRHFOv+
G4gz/MMyB+68DEambUMi6Zgkr96eCf9tfRsB5eSRsOYqebHCtcFh+5x2qDgTjNG1
H5YsuSpniIFocuyGKKCk4s6hz+TAwir+AsJFLqlQWLP9ADN3bXe4BaDfQesgt5AL
Svk0xLBGIVPikwX3fllOKcXnIso76xr6Z+KN2BI+O8fqfylhHDyzhBnKls2EpfdJ
gza94LL4alo4Iid2Rw073fb7GP1f6BHJJAIHt1fHZJDn1ewHvOYrhtm/XLqLkEIj
vhcCAwEAAaA7MDkGCSqGSIb3DQEJDjEsMCowCQYDVR0TBAIwADAdBgNVHQ4EFgQU
CqjVl4eZeQOpjpuKbyc75Q11JaswDQYJKoZIhvcNAQELBQADggEBAGVpq0VFHG4W
0Surnlwj+KkAdLRmNX0u3lg30RZpULm9tpmji01mFIq65NHpEVEXBZ69nkfMeuHC
6Lj//pdzNueIiH+/FfQOk6IdS0+cmVxjF2xXrhats1HYNdiiSGdVcK93gZgdWqJY
Fm6l2x9CiusYFfhvDfkJKcFJ6z3AUSoi4KijREgkrpnk+yZ+m2Tm/DjgA5YLGaDb
p/DBsJh6M6aI0zbgebyrxiJEAJri1l7bKmx+5h0Hu1VaUMllP7TkgnhYLDG4Mdz2
W1STwRO5RNbY6qnoVdybKiL/6BNhyuH9r+DMeY+THQea+rx+N61xetEL/hmlugth
XVDYrXrEdtE=
-----END CERTIFICATE REQUEST-----
quit
% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIDnjCCAoagAwIBAgIBCjANBgkqhkiG9w0BAQsFADAcMRowGAYDVQQDExFyb290
Y2EubndrdC5sb2NhbDAeFw0yMTAzMTQxNjUyNTNaFw0yMjAzMTQxNjUyNTNaMIGz
MQswCQYDVQQGEwJGSTELMAkGA1UECBMCSFIxDDAKBgNVBAcTA0hFTDENMAsGA1UE
CxMEbndrdDENMAsGA1UEChMEbndrdDFBMD8GA1UEAxM4dnNtYXJ0LTgzOWFlZTYy
LWZlZDUtNDBmYy1iNTNjLTZmYzU0N2E4ZDdlNS0wLm53a3QubG9jYWwxKDAmBgkq
hkiG9w0BCQEWGXRoZW5ldHdvcmt0aW1lc0BnbWFpbC5jb20wggEiMA0GCSqGSIb3
DQEBAQUAA4IBDwAwggEKAoIBAQCXjcjuiPJ6VDt/+YyHkSJNdNVAsbAOuyZTfMdF
P1EfnmgAwAM6iwl2ws1e2YegZ1gCFsqTM9Ld7LiURxTr/huIM/zDMgfuvAxGpm1D
IumYJK/engn/bX0bAeXkkbDmKnmxwrXBYfucdqg4E4zRtR+WLLkqZ4iBaHLshiig
pOLOoc/kwMIq/gLCRS6pUFiz/QAzd213uAWg30HrILeQC0r5NMSwRiFT4pMF935Z
TinF5yLKO+sa+mfijdgSPjvH6n8pYRw8s4QZypbNhKX3SYM2veCy+GpaOCIndkcN
O932+xj9X+gRySQCB7dXx2SQ59XsB7zmK4bZv1y6i5BCI74XAgMBAAGjUzBRMA8G
A1UdEwEB/wQFMAMBAf8wHwYDVR0jBBgwFoAU0sGJJNF8VH0oQ3NOtHf4fy13oa8w
HQYDVR0OBBYEFAqo1ZeHmXkDqY6bim8nO+UNdSWrMA0GCSqGSIb3DQEBCwUAA4IB
AQAJK3ojgtVWCw2DnoJ44paGEuI9TF7YdKqsJnM3rrFU9zwuoqUNHcEQtvrUmMFZ
Pffo8sF3OMOPLAZ99kTvBDVAKZM0MFwlC619XdB3XyPjbkmsgA5oc12tH+bVEfFW
41qo8xNU65/BA0ysdh1Qt7VT/PcmsliPDY81ltmtxX3S84gQCwlAZTtI5o0ocCdX
KuC8SgICacecfjy8NlcOpNduZsKFYY7e1bUuCDuWMcb9Fk3Rf6NCXhI6c1oWO9xu
XTgHpf0mBOlQyU7540/ubb/DEldCVbCisd3V9rqk+L4QnFYte+LZr3j7Gf/2KILb
IrhZ+4j7/DdwVEKnHv3xAD+9
-----END CERTIFICATE-----
```

Example 1-21: *Generating the Granted Certificate for vSmart.*

Go back to the vManage management window and navigate to *Configuration* > *Certificates* window and select *Controller* sheet. Select the vSmart row and click the *Install Certificate* button. Paste the Granted Certificate into the Install *Certificate Text* in the *Install Certificate* window and click the *Install* button.

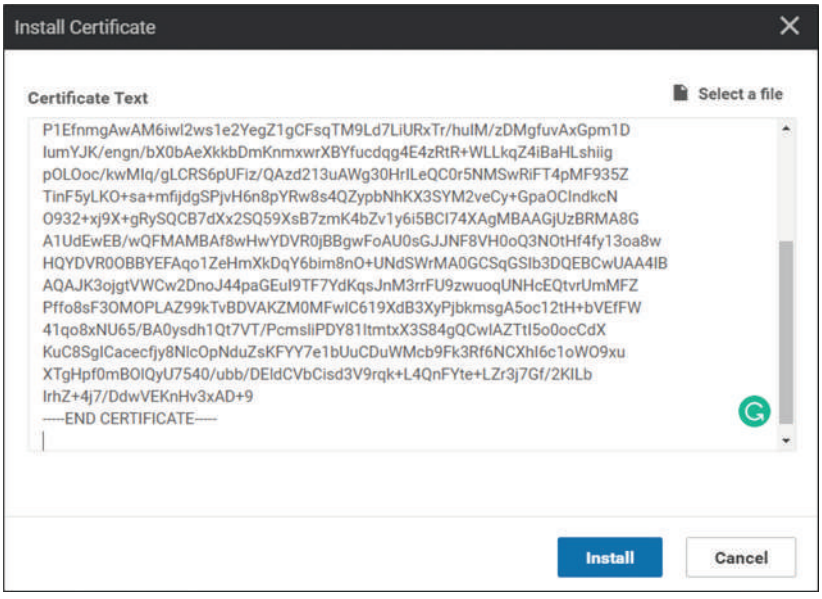


Figure 1-31: *Installing the Granted Certificate into vSmart Continues.*

The figure below illustrates the installation progress and its result which is Success. At this phase, the vSmart certification process is ready.

Figure 1-32: *The Granted Certificate Installation Progress.*

We can verify certificate status from the *Controllers* sheet in *Configuration* > *Devices* windows. From there we can see that the *Certificate Status* column is *Installed*.

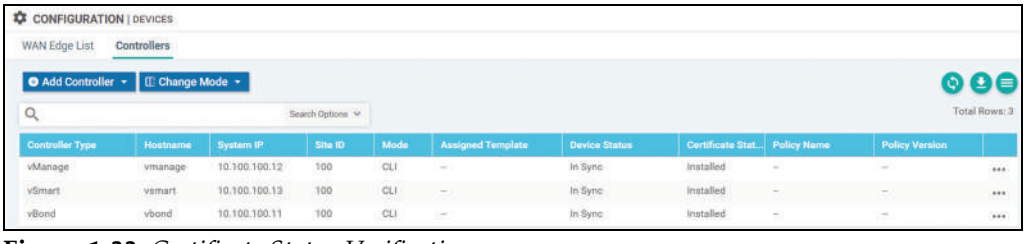


Figure 1-33: *Certificate Status Verification.*

Control Connection Verification

You can verify the control connections by navigating to *Monitor* > *Network*. Select the WAN-Edge tab and click the device which DTLS connections you wish to very. Figure 1-34 verifies the DTL connection from the vMange perspective.

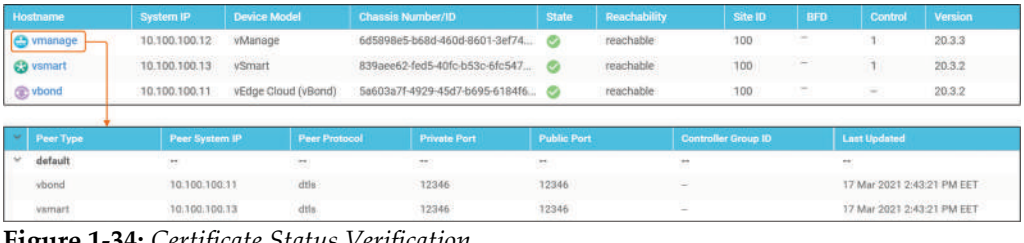


Figure 1-34: *Certificate Status Verification.*

Chapter 2: Manual vEdge Provision

Introduction

This chapter explains how we can provision vEdge devices manually. It starts by explaining how to build an initial system and tunnel interface configurations. Then it goes through the various certificate installation steps (CA root certificate, Certificate Signing Request (CSR), and granted certificate). After the initial configuration and certificate process section, this chapter shows how we can verify the Control Plane operation. Figure 2-1 illustrates our example topology. For simplicity, there are only two vEdge devices used in this chapter.

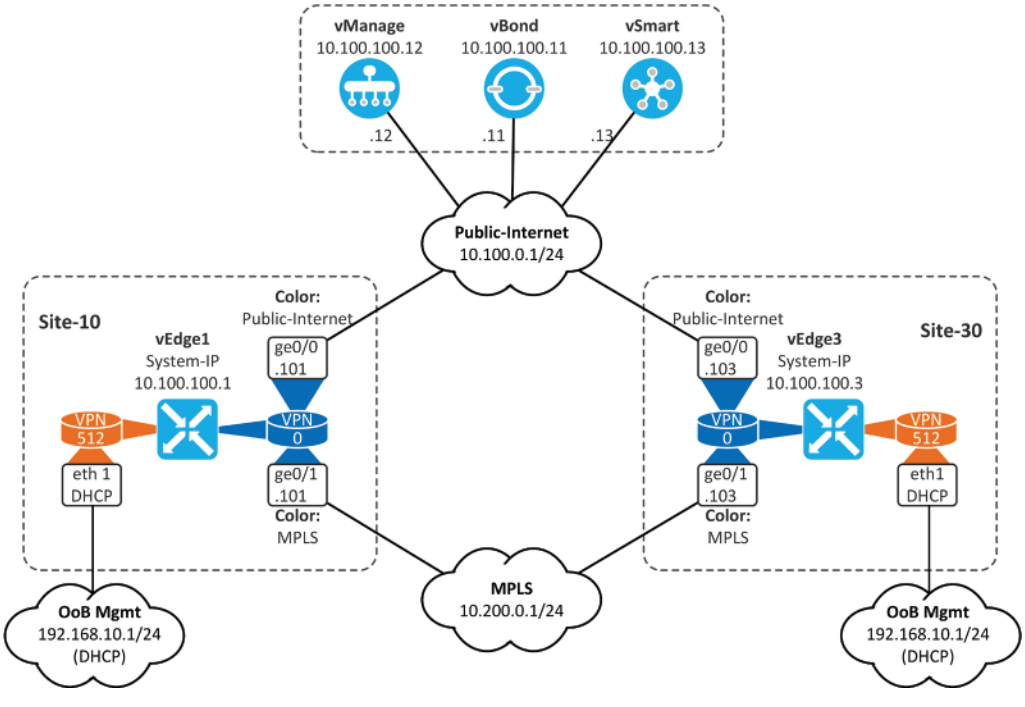


Figure 2-1: *SD-WAN Topology.*

vEdge Configuration

System Information

The only difference in vEdge initial system configuration compared to vManagem vBond, and vSmart is the device-specific host-name and system-ip values.

```
vedge# conf t
Entering configuration mode terminal
vedge(config)# system
vedge(config-system)# host-name vEdge-1
vedge(config-system)# site-id 10
vedge(config-system)# system-ip 10.100.100.101
vedge(config-system)# organization-name nwkt
vedge(config-system)# vbond 10.100.0.11
vedgeconfig-system)# ntp server 10.100.0.14
vedge(config-server-10.100.0.14)# vpn 0
vedge(config-server-10.100.0.14)# exit
vedge(config-ntp)# exit
vedge(config-system)# commit
Commit complete.
vEdge-1(config-system)#
```

Example 2-1: *The Initial Configuration of vEdge-1-Step#1: System Configuration.*

Underlay Network: VPN 0

VPN 0 is always used only for Underlay Network connections. The VPN 0 is like a Front-Door VRF in Cisco Intelligent WAN (iWAN) SD-WAN solution. In our example, interfaces ge0/0 and ge0/1 on vEdges are attached to VPN 0 and IP addresses are bind to them statically. Both interfaces are used for IPSec tunneling. The definition *color* identifies the transport connections. The color of the interface ge0/0 is *public-internet* while the color of interface g0/1 is *mpls*. The command ***color [color] restrict*** means that tunnels are only established between the TLOCs (Transport Locator) belonging to the same color. If the *restrict* option is left out, the vEdge tries to establish a tunnel with every remote TLOCs (MPLS-to-MPLS and MPLS-Public-Internet, Public-Internet-to-MPLS, and Public-Internet-to-Public-Internet) it learns via Overlay Management Protocol (OMP). These kinds of failed tunnels are shown in *Dashboard > Main Dashboard* windows in the *Side Health* field as *Partial WAN Connectivity*. The Main dashboard is shown in figure 2-4 on page 43 (without any issues).

Note that there are 22 pre-defined colors which are divided into *Public* and *Private* colors based on their Network Address Translation (NAT) solution. If NAT is required (or might later be required) use public colors (3g, lte, biz-internet, public-internet, blue, green, red, bronze, silver, gold, custom1-3). If NAT is not needed, use private colors (metro-ethernet, mpls, private 1-6).

The command ***max-control-connection 0*** under tunnel-interface assigned to the interface ge0/1 means that vEdge doesn't try to establish control connections to vManage, vBond, or vSmart. Without the command, vEdge-1 and vEdge-3 will try to establish control connections over MPLS without success (no route). These kinds of failures are shown in *Dashboard > Main Dashboard* windows in the *Control Status* field as *Partial*.

I have also configured VPN 512 for Out of Band Management which is used for copying the root certificate file into vEdges by using WinSCP in the same way that was done with control plane devices vBond and vSmart.

```
vEdge-1(config-system)# vpn 0
vEdge-1(config-vpn-0)#  interface ge0/0
vEdge-1(config-interface-ge0/0)# ip address 10.100.0.101/24
vEdge-1(config-interface-ge0/0)# tunnel-interface
vEdge-1(config-tunnel-interface)# color public-internet restrict
vEdge-1(config-tunnel-interface)# encapsulation ipsec
vEdge-1(config-tunnel-interface)# allow-service all
vEdge-1(config-tunnel-interface)# no shutdown
vEdge-1(config-tunnel-interface)# exit
vEdge-1(config-interface-ge0/0)# exit
vEdge-1(config-vpn-0)# interface ge0/1
vEdge-1(config-interface-ge0/1)# ip address 10.200.0.101/24
vEdge-1(config-interface-ge0/1)# tunnel-interface
vEdge-1(config-tunnel-interface)# color mpls restrict
vEdge-1(config-tunnel-interface)# max-control-connections 0
vEdge-1(config-tunnel-interface)# encapsulation ipsec
vEdge-1(config-tunnel-interface)# allow-service all
vEdge-1(config-tunnel-interface)# no shutdown
vEdge-1(config-tunnel-interface)# exit
vEdge-1(config-interface-ge0/1)# exit
vEdge-1(config-system)# vpn 512
vEdge-1(config-vpn-512)# interface eth0
vEdge-1(config-interface-eth0)# ip dhcp-client
vEdge-1(config-interface-eth0)# no shutdown
vEdge-1(config-interface-eth0)# !
vEdge-1(config-interface-eth0)# ip route 0.0.0.0/0 192.168.10.1
vEdge-1(config-vpn-512)# commit
```

Example 2-2: The *Initial Configuration of vEdge-1-Step#2: VPN Specific Interface Settings.*

Certification enrollment

The CA root certificate enrollment process follows the same principles as what was introduced in chapter 1 with vBond and vMange. I have already copied the PKI.pem file into the vEdge directory home/admin, where it can be installed.

```
vEdge-3# request root-cert-chain install home/admin/PKI.pem
Uploading root-ca-cert-chain via VPN 0
Copying ... /home/admin/PKI.pem via VPN 0
Updating the root certificate chain..
Successfully installed the root certificate chain
```

Example 2-3: *Certificate Enrollment-Step#1: Installing the CA Root Certificate.*

Next, we generate a Certificate Signing Request (CSR) with the name csr.txt into the home/admin directory.

```
vEdge-3# request csr upload home/admin/csr.txt
Uploading CSR via VPN 0
Enter organization-unit name            : nwkt
Re-enter organization-unit name          : nwkt
Generating private/public pair and CSR for this vedge device
Generating CSR for this vedge device   ........[DONE]
Copying ... /home/admin/csr.txt via VPN 0
CSR upload successful
```

Example 2-4: *Certificate Enrollment-Step#2: Generating CSR.*

Then we go into the Linux shell and verify that the file is created. After that, we printed it out and copy it to the clipboard.

```
vEdge-3# vshell
vEdge-3:~$ ls
PKI.pem  archive_id_rsa.pub  csr.txt
vEdge-3:~$ more csr.txt
-----BEGIN CERTIFICATE REQUEST-----
MIIDQjCCAioCAQAwgcExCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlh
MREwDwYDVQQHEwhTYW4gSm9zZTENMAsGA1UECxMEbndrdDEUMBIGA1UEChMLVmlw
dGVsYSBMTEMxQTA/BgNVBAMTOHZlZGdlLTg3NzAwZDE0LTBiNGItNGRjYi1hOTNk
LTQ3YzEyYTViYzMyMS0wLnZpcHRlbGEuY29tMSIwIAYJKoZIhvcNAQkBFhNzdXBw
b3J0QHZpcHRlbGEuY29tMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
qrdJX6GajIK9kZY1Go358nYcyCxh6ie8w6uXvefUhXytsMc2cIVbiTimhYTrRB5+
0Ag2R6WjoohJrlpvDom/mJuNo6GuNjzJOXEhEOLVH2zDwvTnI3nhMmiCWLTNMkGe
NeQSKbHu7VwJ9/zbpPYJgmtsTVF4pzXVN3XVIRAq65VaZ9Tlg4kZUdFXnrKolx0p
hW8DB9QhIL9CO5fGmXuK+Ahc810gV6ObAy+fS4Gbt9fAZToXUK65ToeHWjTzgIsD
3OV4Wieae9PAJd2TZEGismiSONv8JeYVjxXIpzdk+j4NXBv9QgpZxOoEnRpTl/ds
nEfjNpAmHL4m6eaQh2XrJwIDAQABoDswOQYJKoZIhvcNAQkOMSwwKjAJBgNVHRME
AjAAMB0GA1UdDgQWBBSnmgfFMK/zFlb0zYjfdgMZojPhzDANBgkqhkiG9w0BAQsF
AAOCAQEAUsxQlaor8Clt4aDhK3S+t/jo+48i2f1WvKro7CLMC5mB9cKrmugM+fhl
F33mnQtR6/F08yAnMXzaPQuHIN7P6f4nZ0lYNGa8HWXAanlkjqKILoYBfB4LJ7G8
z9UZM03e/0myjCdik2oLhPO0tza/fZ0izf5UzBKcgGWJzn9fPhbgKD+g5ejaF4Vk
kyYYZGnWK7C5okdGqsEv6MMkWxmS5u7hAI3j82gmV7drFL/qK/z23NngBq9Rtz3B
8pCjQDOIUvpWrJQRgY2yAHDsp0kOgQ2UuJdRAvmuaUPr1krmrjdM2+EpItfl2SB4
+5SVRtKDDRiQQG/hLKBGsX+OSMS7zA==
-----END CERTIFICATE REQUEST-----
vEdge-3:~$
```

Example 2-5: *Certificate Enrollment-Step#3: printing the CSR.*

Use the command ***crypto pki server PKI request pkcs10 terminal*** in order to paste CSR into our IOS-XE CA-Server. When done, type ***quit*** and then press enter to get Granted Certificate. Copy the Granted Certificate into the clipboard.

```
CA-Server#crypto pki server PKI request pkcs10 terminal
PKCS10 request in base64 or pem

% Enter Base64 encoded or PEM formatted PKCS10 enrollment request.
% End with a blank line or "quit" on a line by itself.
-----BEGIN CERTIFICATE REQUEST-----
MIIDQjCCAioCAQAwgcExCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlh
MREwDwYDVQQHEwhTYW4gSm9zZTENMAsGA1UECxMEbndrdDEUMBIGA1UEChMLVmlw
dGVsYSBMTEMxQTA/BgNVBAMTOHZlZGdlLTg3NzAwZDE0LTBiNGItNGRjYi1hOTNk
LTQ3YzEyYTViYzMyMS0wLnZpcHRlbGEuY29tMSIwIAYJKoZIhvcNAQkBFhNzdXBw
b3J0QHZpcHRlbGEuY29tMIIBIjANBgkqhkiG9w0BAQEFAAOCAQ8AMIIBCgKCAQEA
qrdJX6GajIK9kZY1Go358nYcyCxh6ie8w6uXvefUhXytsMc2cIVbiTimhYTrRB5+
0Ag2R6WjoohJrlpvDom/mJuNo6GuNjzJOXEhEOLVH2zDwvTnI3nhMmiCWLTNMkGe
NeQSKbHu7VwJ9/zbpPYJgmtsTVF4pzXVN3XVIRAq65VaZ9Tlg4kZUdFXnrKolx0p
```

```
hW8DB9QhIL9CO5fGmXuK+Ahc810gV6ObAy+fS4Gbt9fAZToXUK65ToeHWjTzgIsD
3OV4Wieae9PAJd2TZEGismiSONv8JeYVjxXIpzdk+j4NXBv9QgpZxOoEnRpTl/ds
nEfjNpAmHL4m6eaQh2XrJwIDAQABoDswOQYJKoZIhvcNAQkOMSwwKjAJBgNVHRME
AjAAMB0GA1UdDgQWBBSnmgfFMK/zFlb0zYjfdgMZojPhzDANBgkqhkiG9w0BAQsF
AAOCAQEAUsxQlaor8Clt4aDhK3S+t/jo+48i2f1WvKro7CLMC5mB9cKrmugM+fhl
F33mnQtR6/F08yAnMXzaPQuHIN7P6f4nZ0lYNGa8HWXAanlkjqKILoYBfB4LJ7G8
z9UZM03e/0myjCdik2oLhPO0tza/fZ0izf5UzBKcgGWJzn9fPhbgKD+g5ejaF4Vk
kyYYZGnWK7C5okdGqsEv6MMkWxmS5u7hAI3j82gmV7drFL/qK/z23NngBq9Rtz3B
8pCjQDOIUvpWrJQRgY2yAHDsp0kOgQ2UuJdRAvmuaUPr1krmrjdM2+EpItfl2SB4
+5SVRtKDDRiQQG/hLKBGsX+OSMS7zA==
-----END CERTIFICATE REQUEST-----
quit
% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIDrDCCApSgAwIBAgIBDzANBgkqhkiG9w0BAQsFADAcMRowGAYDVQQDExFyb290
Y2EubndrdC5sb2NhbDAeFw0yMTAzMTgxNTM4NDdaFw0yMjAzMTgxNTM4NDdaMIHB
MQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTERMA8GA1UEBxMIU2Fu
IEpvc2UxDTALBgNVBAsTBG53a3QxFDASBgNVBAoTC1ZpcHRlbGEgTExDMUEwPwYD
VQQDEzh2ZWRnZS04NzcwMGQxNC0wYjRiLTRkY2ItYTkzZC00N2MxMmE1YmMzMjEt
MC52aXB0ZWxhLmNvbTEiMCAGCSqGSIb3DQEJARYTc3VwcG9ydEB2aXB0ZWxhLmNv
bTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKq3SV+hmoyCvZGWNRqN
+fJ2HMgsYeonvMOrl73n1IV8rbDHNnCFW4k4poWE60QeftAINkelo6KISa5abw6J
v5ibjaOhrjY8yTlxIRDi1R9sw8L05yN54TJogli0zTJBnjXkEimx7u1cCff826T2
CYJrbE1ReKc11Td11SEQKuuVWmfU5YOJGVHRV56yqJcdKYVvAwfUISC/QjuXxpl7
ivgIXPNdIFejmwMvn0uBm7fXwGU6F1CuuU6Hh1o084CLA9zleFonmnvTwCXdk2RB
orJokjjb/CXmFY8VyKc3ZPo+DVwb/UIKWcTqBJ0aU5f3bJxH4zaQJhy+JunmkIdl
6ycCAwEAAaNTMFEwDwYDVR0TAQH/BAUwAwEB/zAfBgNVHSMEGDAWgBTSwYkk0XxU
fShDc060d/h/LXehrzAdBgNVHQ4EFgQUp5oHxTCv8xZW9M2I33YDGaIz4cwwDQYJ
KoZIhvcNAQELBQADggEBAATPB1iYqcg0o9n7sgyA2DGFayZYEmIHS7R/2YMbFaQ9
yrbzD9lKoISOvyHDT4SSRdFpBYM2ZHMFdM62JsRqpCZz5Leswv++Y4DMbu5t8MKW
arDYzhYVmggTlmMQ+BLU1WOE+gJn4gIxVNKJcphWJOA5oQDcVnlTbB5iKCDiIfVx
0GWV8UKLNpFV8pypmUIUz7UDAx7UQcSgOYsvzJ80C2sM1XFIuF8+C/ieFUxXaz3i
QPlHUvFTE3Tz9uTKpk/lN0UDib9RxfXUOqHX+UQ8ok80xQN5Td6IsHUWSn97q8Pm
GCz80wPh9hmfrPq0Dvz5ak0Gde5foIPfoKCNkKDfIvw=
-----END CERTIFICATE-----

CA-Server#
```

Example 2-6: *Certificate Enrollment-Step#4: Generating Granted Certificate.*

Create a certificate file cert.txt by first using the command ***cat <<"" > cert.txt*** and then by pasting the granted certificate from the clipboard.

```
vEdge-3:~$ cat <<"" > cert.txt
> -----BEGIN CERTIFICATE-----
> MIIDrDCCApSgAwIBAgIBDzANBgkqhkiG9w0BAQsFADAcMRowGAYDVQQDExFyb290
> Y2EubndrdC5sb2NhbDAeFw0yMTAzMTgxNTM4NDdaFw0yMjAzMTgxNTM4NDdaMIHB
> MQswCQYDVQQGEwJVUzETMBEGA1UECBMKQ2FsaWZvcm5pYTERMA8GA1UEBxMIU2Fu
> IEpvc2UxDTALBgNVBAsTBG53a3QxFDASBgNVBAoTC1ZpcHRlbGEgTExDMUEwPwYD
> VQQDEzh2ZWRnZS04NzcwMGQxNC0wYjRiLTRkY2ItYTkzZC00N2MxMmE1YmMzMjEt
> MC52aXB0ZWxhLmNvbTEiMCAGCSqGSIb3DQEJARYTc3VwcG9ydEB2aXB0ZWxhLmNv
> bTCCASIwDQYJKoZIhvcNAQEBBQADggEPADCCAQoCggEBAKq3SV+hmoyCvZGWNRqN
> +fJ2HMgsYeonvMOrl73n1IV8rbDHNnCFW4k4poWE60QeftAINkelo6KISa5abw6J
> v5ibjaOhrjY8yTlxIRDi1R9sw8L05yN54TJogli0zTJBnjXkEimx7u1cCff826T2
> CYJrbE1ReKc11Td11SEQKuuVWmfU5YOJGVHRV56yqJcdKYVvAwfUISC/QjuXxpl7
> ivgIXPNdIFejmwMvn0uBm7fXwGU6F1CuuU6Hh1o084CLA9zleFonmnvTwCXdk2RB
> orJokjjb/CXmFY8VyKc3ZPo+DVwb/UIKWcTqBJ0aU5f3bJxH4zaQJhy+JunmkIdl
> 6ycCAwEAAaNTMFEwDwYDVR0TAQH/BAUwAwEB/zAfBgNVHSMEGDAWgBTSwYkk0XxU
> fShDc060d/h/LXehrzAdBgNVHQ4EFgQUp5oHxTCv8xZW9M2I33YDGaIz4cwwDQYJ
> KoZIhvcNAQELBQADggEBAATPB1iYqcg0o9n7sgyA2DGFayZYEmIHS7R/2YMbFaQ9
> yrbzD9lKoISOvyHDT4SSRdFpBYM2ZHMFdM62JsRqpCZz5Leswv++Y4DMbu5t8MKW
> arDYzhYVmggTlmMQ+BLU1WOE+gJn4gIxVNKJcphWJOA5oQDcVnlTbB5iKCDiIfVx
> 0GWV8UKLNpFV8pypmUIUz7UDAx7UQcSgOYsvzJ80C2sM1XFIuF8+C/ieFUxXaz3i
> QPlHUvFTE3Tz9uTKpk/lN0UDib9RxfXUOqHX+UQ8ok80xQN5Td6IsHUWSn97q8Pm
> GCz80wPh9hmfrPq0Dvz5ak0Gde5foIPfoKCNkKDfIvw=
> -----END CERTIFICATE-----
>
>
vEdge-3:~$ exit
vEdge-3#
```

Example 2-7: *Certificate Enrollment-Step#5: Creating Certificate File.*

Next, install the certificate.

```
vEdge-3# request certificate install  home/admin/cert.txt
Installing certificate via VPN 0
Copying ... /home/admin/cert.txt via VPN 0
cp -f "/usr/share/viptela/tmp_csr/server.key" "/usr/share/viptela/server.key"
moving temp Cert "/usr/share/viptela/server.crt.tmp" to Cert
"/usr/share/viptela/vedge_certs/client_0F.crt"
Successfully installed the certificate
```

Example 2-8: *Certificate Enrollment-Step#6: Installing Granted Certificate.*

After installing certificates, we can register devices into vMange. This is done by assigning both the chassis number and the serial number of vEdge to vMange and vBond. The process is shown in examples 2-9, 2-10, and 2-11.

```
vEdge-3# show certificate serial
Chassis number: 87700d14-0b4b-4dcb-a93d-47c12a5bc321 serial number: 0F
```

Example 2-9: *Certificate Enrollment-Step#7: vEdge Serial Number Verification.*

```
vmanage#request vedge add chassis-num 87700d14-0b4b-4dcb-a93d-47c12a5bc321 serial-num
0F
status success
```

Example 2-10: *Certificate Enrollment-Step#8: Adding vEdge into vManage.*

```
vbond# request vedge add chassis-num 87700d14-0b4b-4dcb-a93d-47c12a5bc321 serial-num
0F
status success
```

Example 2-11: *Certificate Enrollment-Step#9: Adding vEdge into vBond.*

We can verify that vEdges are registered into vMange by navigating to the *Configuration/Devices* window and by selecting the *WAN Edge List* tab. As can be seen from the figure below, the state of both vEdges is successful.

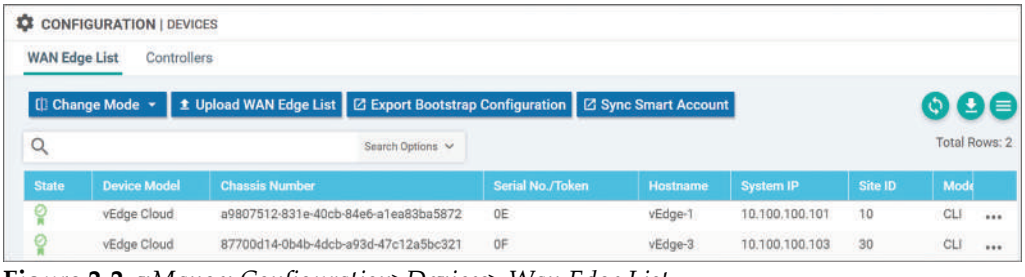


Figure 2-2: *vMange: Configuration>Devices> Wan Edge List.*

As the last step, we need to send the updated vEdge list to other control devices by navigating the *Configuration>Certificates* window and selecting the *Send to Controllers* from the *WAN Edge List* tab.

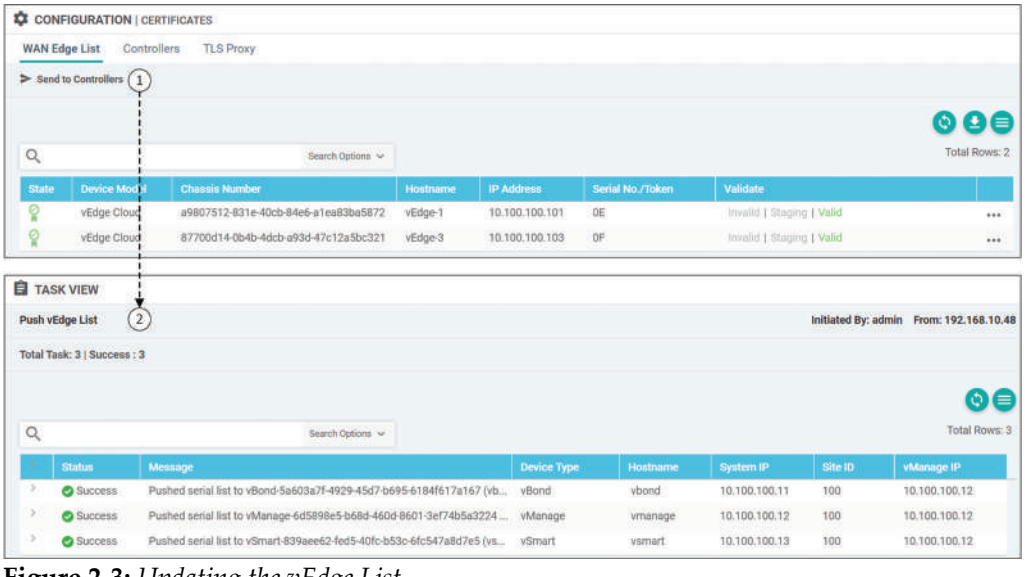


Figure 2-3: *Updating the vEdge List.*

Figure 2-4 verifies that our example SD-WAN infrastructure is now up and running.

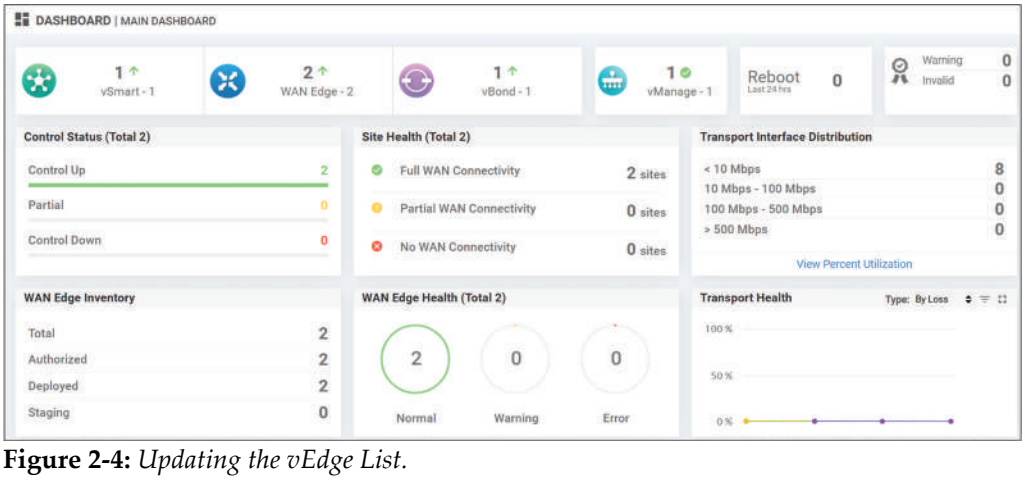


Figure 2-4: *Updating the vEdge List.*

Onboarding Process

Figure 2-5 illustrates the onboarding process after initial configuration and successful certification enrollment from the vEdge-1 perspective. (1) vEdge-1 does bidirectional authentication with vBond. (2) Devices establish DTLS tunnel. (3) vBond instructs vEdge-1 how to reach vManage and vSmart devices over the DTLS tunnel. (4) vEdge does bidirectional authentication with vManage. (5) Devices establish DTLS tunnel. (6) vMange sends the configuration defined in configuration templates to vEdge-1 over the DTLS tunnel. (7) vEdge does bidirectional authentication with vSmart. (8) Devices establish DTLS tunnel. (9) vEdge and vSmart exchange OMP routing information over DTLS tunnel. We don't have any configured service VPN at this phase, so only TLOC routes are advertised (10) vEdge tears down the DTLS tunnel with vBond.

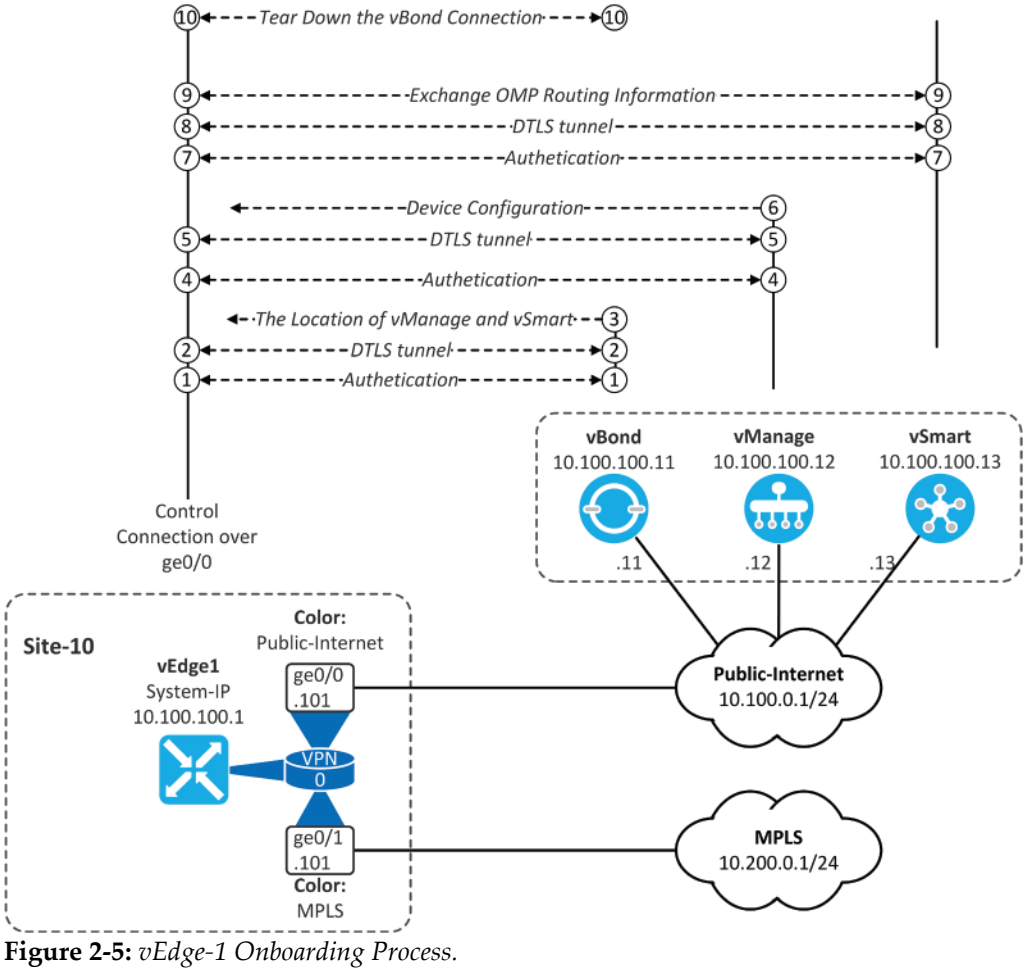


Figure 2-5: *vEdge-1 Onboarding Process.*

The control connections are established over DTLS tunnels while Data Plane tunnels are using IPSec tunnels by default. Note that control connections are only established over the Public-Internet and IPSec tunnels are only established between the interfaces with the same color. We will get back on these in later sections.

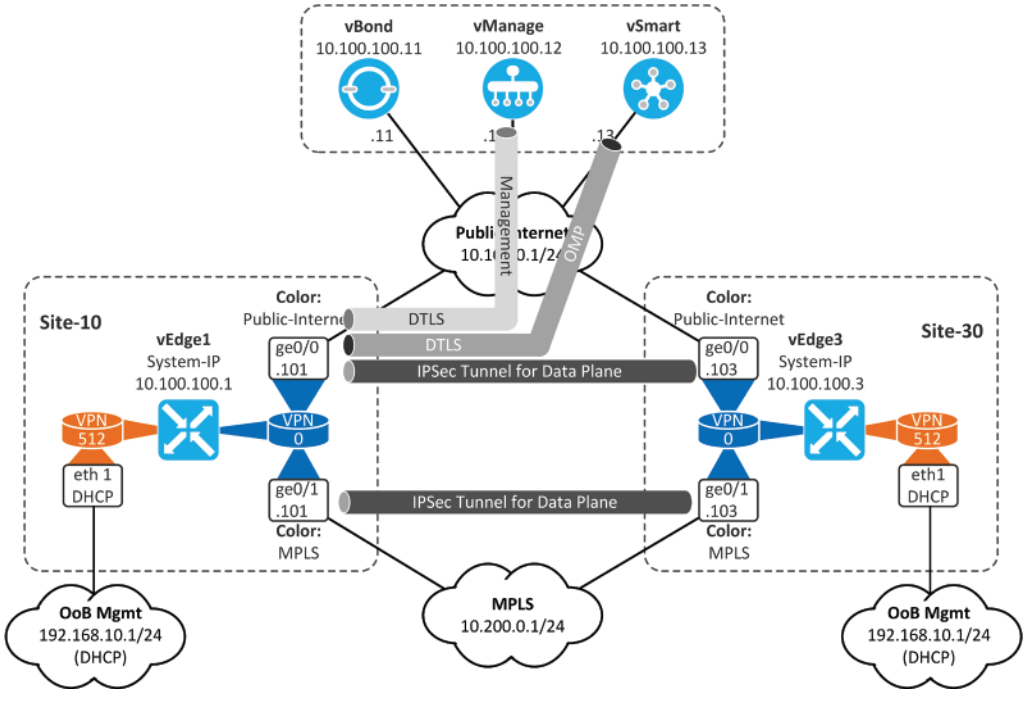


Figure 2-6: *vEdge-1 Tunnels for Management Plane, Control Plane, and Data Plane.*

Control Connection Verification

We can verify vEdge-1 control connections from the vManage GUI by navigating to *Monitor* > *Network* and choosing vEdge-1 from the *Host Name* list. The current Control Connection can be verified by scrolling down to *Control Connection* on the left menu. Figure 2-7 shows that vEdge-1 has DTLS tunnels to vSmart and vManage, both using public-internet.

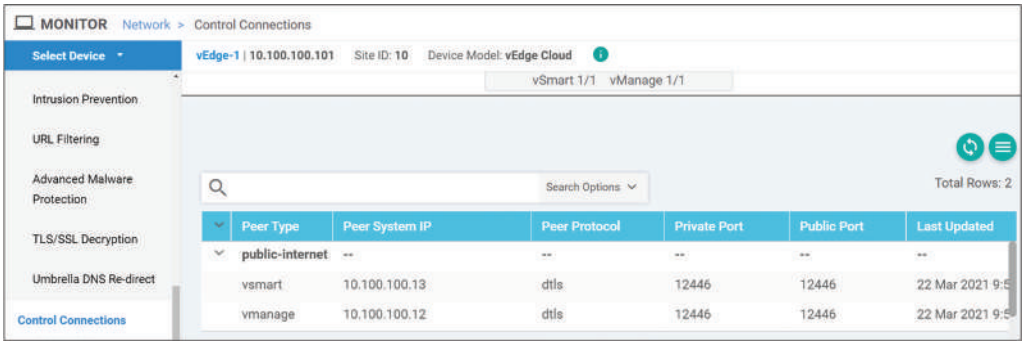


Figure 2-7: *GUI-based vEdge Control Connections Verifications.*

You can also check the control connections by selecting the *Troubleshooting* from the left menu and then by selecting the *Control Connections (Live view)* from the *Connectivity* sections.

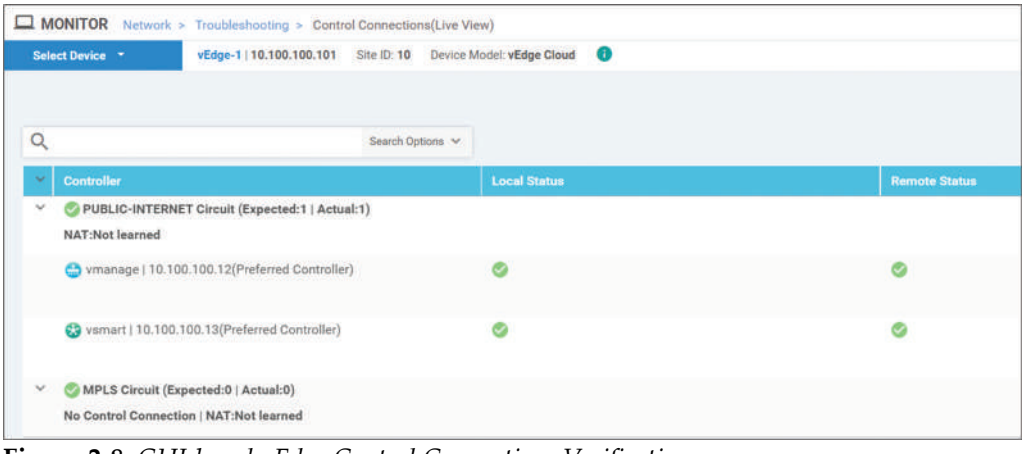


Figure 2-8: *GUI-based vEdge Control Connections Verifications.*

CLI-based verification can be done by using the command ***show control connections details***. vBond is also listed in output but with the System-IP 0.0.0.0.

```
vEdge-1# show control connections detail
""
-----------------------------------------------------------------------------------
 LOCAL-COLOR- public-internet SYSTEM-IP- 10.100.100.13   PEER-PERSONALITY- vsmart
-----------------------------------------------------------------------------------
site-id             100
domain-id           1
protocol            dtls
private-ip          10.100.0.13
private-port        12446
public-ip           10.100.0.13
public-port         12446

state               up [Local Err: NO_ERROR] [Remote Err: NO_ERROR]
```

```
uptime               0:01:50:35
hello interval       1000
hello tolerance      12000
controller-grp-id    0

  <Tx and Rx statistics ommited>

------------------------------------------------------------------------------------
 LOCAL-COLOR- public-internet SYSTEM-IP- 0.0.0.0   PEER-PERSONALITY- vbond
------------------------------------------------------------------------------------
site-id              0
domain-id            0
protocol             dtls
private-ip           10.100.0.11
private-port         12346
public-ip            10.100.0.11
public-port          12346

state                up [Local Err: NO_ERROR] [Remote Err: NO_ERROR]
uptime               0:01:50:53
hello interval       1000
hello tolerance      12000
controller-grp-id    0

<Tx and Rx statistics ommited>

------------------------------------------------------------------------------------
 LOCAL-COLOR- public-internet SYSTEM-IP- 10.100.100.12   PEER-PERSONALITY- vmanage
------------------------------------------------------------------------------------
site-id              100
domain-id            0
protocol             dtls
private-ip           10.100.0.12
private-port         12446
public-ip            10.100.0.12
public-port          12446

state                up [Local Err: NO_ERROR] [Remote Err: NO_ERROR]
uptime               0:01:50:37
hello interval       1000
hello tolerance      12000
controller-grp-id    0

 <Tx and Rx statistics ommited>
```

Example 2-12: *CLI-based vEdge Control Connections Verifications.*

You can find current OMP peers by selecting *Real Time* from the left menu and selecting *OMP peers* from the *Device Options:* filed. From the figure below we can see that vEdge-1 has OMP peering with vSmart. Note that you can select what columns you want to see on the screen by clicking the green icon right next to the refresh icon on the right corner.

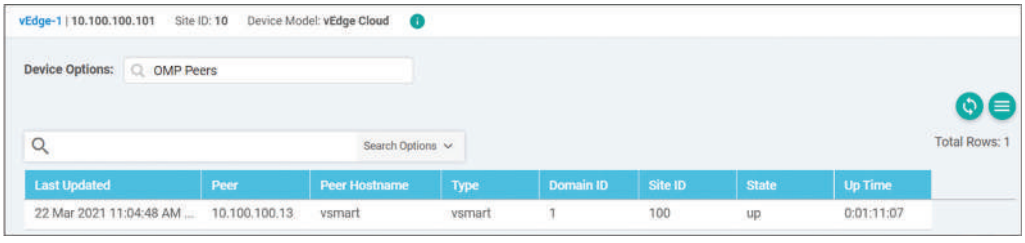


Figure 2-9: *GUI-based vEdge OMP Peer Verifications.*

The same thing can also be verified from the vEdge-1 CLI by using the command ***show omp peers***. The output also shows that vEdge-1 doesn't have received, installed, or sent (R/I/S) any prefixes from the vSmart. This is because we haven't added any service VPNs into example SD-WAN infrastructure yet.

```
vEdge-1# show omp peers
R -> routes received
I -> routes installed
S -> routes sent

                        DOMAIN    OVERLAY   SITE
PEER            TYPE    ID        ID        ID        STATE    UPTIME           R/I/S
-----------------------------------------------------------------------------------
10.100.100.13   vsmart  1         1         100       up       0:04:37:47       0/0/0
```

Example 2-13: *Verifying OMP Peers.*

The only routing information exchanged between vEdges and vSmart via OMP is Transport Locations (TLOCs) information which describes the System-IP (10.100.100.101), Color (mpls/public-internet), and Encapsulation type (IPSEC). Figure 2-10 shows the basic operation of TLOC advertisement. vEdge-1 advertises two TLOCs, one for the color *mpls* and the other one to color *public-internet*. Updates are sent only to vSmart (vEdges doesn't establish OMP peering between themself) that reflects the update to vEdge-3. In that sense, vSmart is like an iBGP route-reflector. The OMP TLOC update carries also a set of attributes, just like BGP Network Layer Reachability Information (NLRI). Attributes *Public-IP* and *Private-IP* addresses are used as destination IP addresses in the tunnel header. Which one is used depends on the TLOC color. The color *mpls* is a *private* color, which means that private-ip is always used in the data plane even though the device is behind the NAT device. The second TLOC with the color *public-internet* is *public* color and with these TLOCs the public-IP address is used. There is no NAT in our example network, so the *public-ip* and *private-ip* attributes are the same on both TLOCs. Note that all OMP updates are sent over DTLS tunnels.

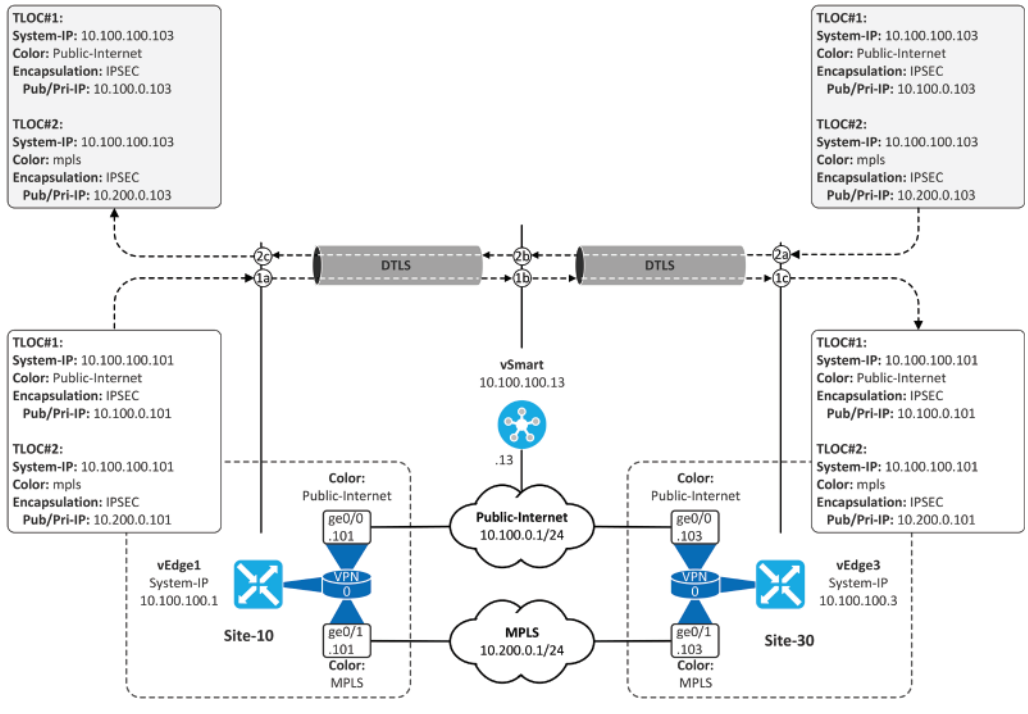


Figure 2-10: *TLOC Updates.*

The example below shows the OMP TLOC update generated by vEdge about its local TLOCs. Note that the *restrict* TLOC attribute states that the receiving vEdge should only try to establish an IPSec tunnel between the same colors.

```
vEdge-1# show omp tlocs advertised detail

----------------------------------------------------
tloc entries for 10.100.100.101
                 mpls
                 ipsec
----------------------------------------------------
           RECEIVED FROM:
peer            0.0.0.0
status          C,Red,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.200.0.101
     public-port       12346
     private-ip        10.200.0.101
```

```
     private-port      12346
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           10
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
    gen-id             0x80000008
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border             not set
     unknown-attr-len  not set
            ADVERTISED TO:
peer    10.100.100.13
    Attributes:
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.200.0.101
     public-port       12346
     private-ip        10.200.0.101
     private-port      12346
     public-ip       ::
     public-port     0
     private-ip      ::
     private-port     0
     domain-id         not set
     site-id           10
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
     gen-id            0x80000008
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border              not set
     unknown-attr-len  not set

-------------------------------------------------
tloc entries for 10.100.100.101
                 public-internet
                 ipsec
```

```
------------------------------------------------
            RECEIVED FROM:
peer            0.0.0.0
status          C,Red,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.100.0.101
     public-port       12346
     private-ip        10.100.0.101
     private-port      12346
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           10
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
    gen-id             0x80000008
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border             not set
     unknown-attr-len  not set
            ADVERTISED TO:
peer    10.100.100.13
    Attributes:
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.100.0.101
     public-port       12346
     private-ip        10.100.0.101
     private-port      12346
     public-ip       ::
     public-port     0
     private-ip      ::
     private-port     0
     domain-id         not set
     site-id           10
     overlay-id        not set
     preference        0
     tag               not set
```

```
     stale             not set
     weight            1
     version           3
     gen-id            0x80000008
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border              not set
     unknown-attr-len  not set
vEdge-1#
```

Example 2-14: *TLOCs Advertised by vEdge-1.*

The example below shows the TLOC generated by vEdge-1 and reflected by vSamrt from the vEdge-3 perspective.

```
vEdge-3# show omp tlocs received

----------------------------------------------------
tloc entries for 10.100.100.101
                 mpls
                 ipsec
----------------------------------------------------
           RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.200.0.101
     public-port       12346
     private-ip        10.200.0.101
     private-port      12346
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           10
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
    gen-id             0x80000008
```

```
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border             not set
     unknown-attr-len  not set

------------------------------------------------
tloc entries for 10.100.100.101
                 public-internet
                 ipsec
------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.100.0.101
     public-port       12346
     private-ip        10.100.0.101
     private-port      12346
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           10
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
    gen-id             0x80000008
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border             not set
     unknown-attr-len  not set

------------------------------------------------
tloc entries for 10.100.100.103
                 mpls
                 ipsec
------------------------------------------------
            RECEIVED FROM:
peer            0.0.0.0
```

```
status          C,Red,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.200.0.103
     public-port       12346
     private-ip        10.200.0.103
     private-port      12346
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           30
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
    gen-id             0x80000007
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border             not set
     unknown-attr-len  not set

------------------------------------------------
tloc entries for 10.100.100.103
                 public-internet
                 ipsec
------------------------------------------------
            RECEIVED FROM:
peer            0.0.0.0
status          C,Red,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     encap-key         not set
     encap-proto       0
     encap-spi         258
     encap-auth        sha1-hmac,ah-sha1-hmac
     encap-encrypt     aes256
     public-ip         10.100.0.103
     public-port       12346
     private-ip        10.100.0.103
     private-port      12346
     public-ip         ::
```

```
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           30
     overlay-id        not set
     preference        0
     tag               not set
     stale             not set
     weight            1
     version           3
    gen-id             0x80000007
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
     border             not set
     unknown-attr-len  not set
vEdge-3#
```

Example 2-15: *TLOCs Received by vEdge-3.*

Using information received from TLOC updates, vEdges can build IPSec tunnels between themselves. You can find current IPsec peers by selecting *Real Time* from the left menu and selecting *IPsec Outbound Connections* in the *Device Options:* field.

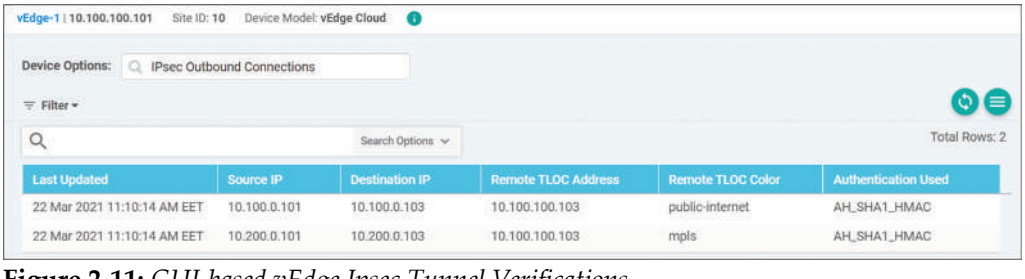


Figure 2-11: *GUI-based vEdge Ipsec Tunnel Verifications.*

You can also monitor device-specific Ipsec tunnels by selecting the *Tunnel* option under the *WAN* section.

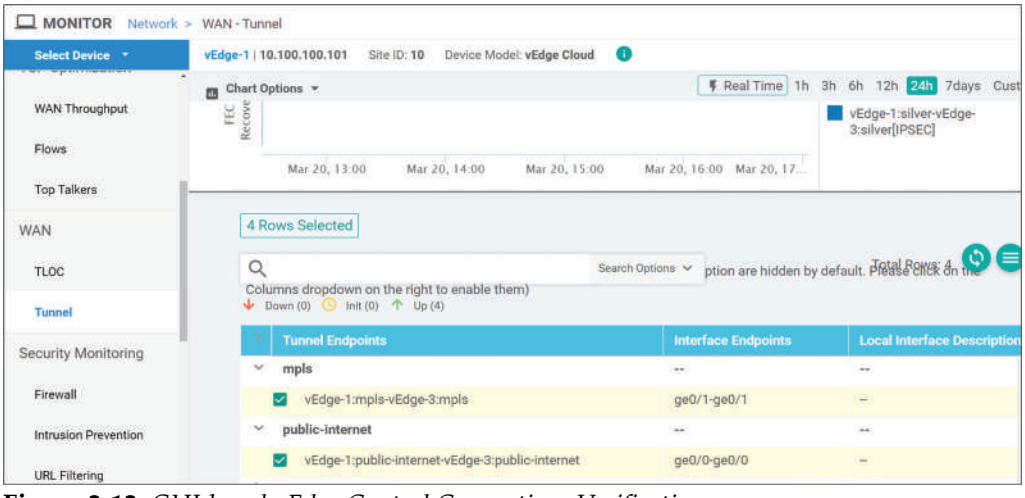


Figure 2-12: *GUI-based vEdge Control Connections Verifications.*

vEdges sends user data over these IPsec tunnels and they also monitor the tunnel quality using a slightly modified Bidirect Forwarding Detection (BFD) solution.

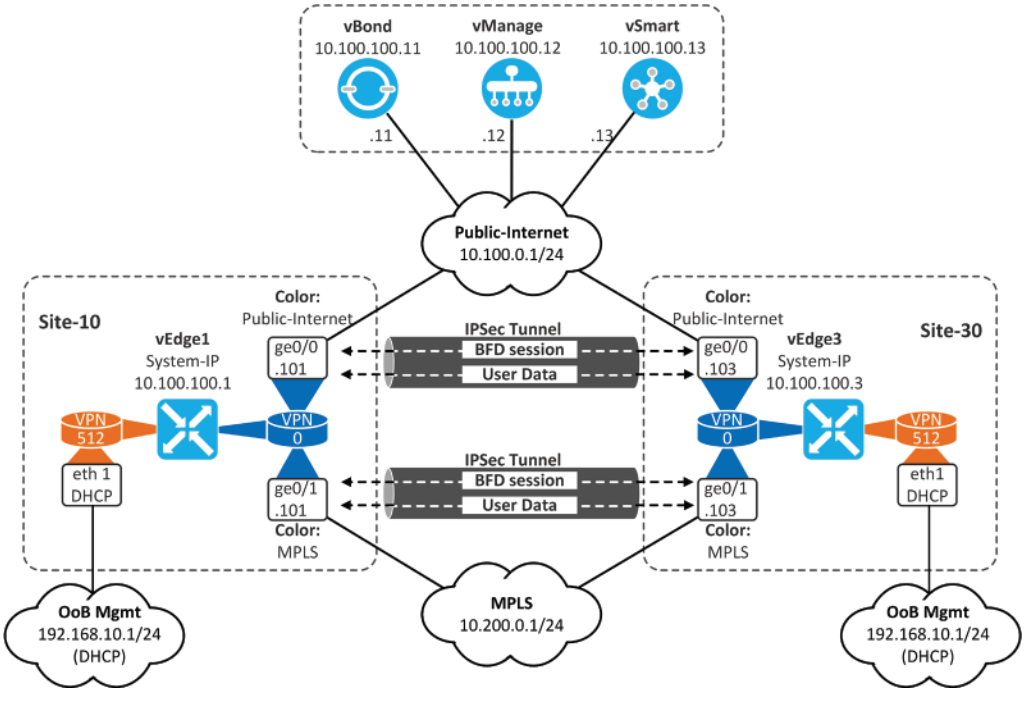


Figure 2-13: *Ipsec Tunnels Between vEdges.*

BFD sessions can be seen by selecting *Real Time* from the left menu and selecting *IPsec Outbound Connections* in the *Device Options:* field.

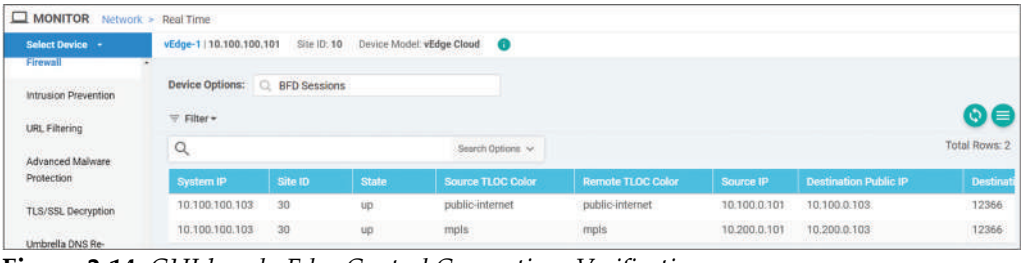


Figure 2-14: *GUI-based vEdge Control Connections Verifications.*

The SD-WAN infra is now ready. The focus of the next chapter is to show how to implement customers into our example network.

Chapter 3: Overlay Management Protocol

Introduction

This chapter introduces the operation of the Overlay Management Protocol (OMP). It starts by introducing TLOC Routes which are used for establishing tunnels between vEdges. Next, it explains OMP Routes which in turn are used for advertising Service VPN specific networks reachability information. I am also going to show the data plane encapsulation when data is sent between the hosts in site 10 and site 30. The purpose of the data plane section is to show how the label attribute advertised within OMP routing advertisements is used to identify customer VPN. In order to see inside captured packets, I am using GRE tunnels instead of IPSec. Figure 3-1 illustrates the example topology used in this chapter. The customer VPN 10 is used on both sites. Site 10 subnet is 172.16.10.0/24 and site 30 subnet is 172.16.30.0/24. Interface ge0/0 in both vEdges is connected to the Public-Internet, and interface ge0/1 is the connected to MPLS transport network where the customer has its dedicated MPLS VPN.

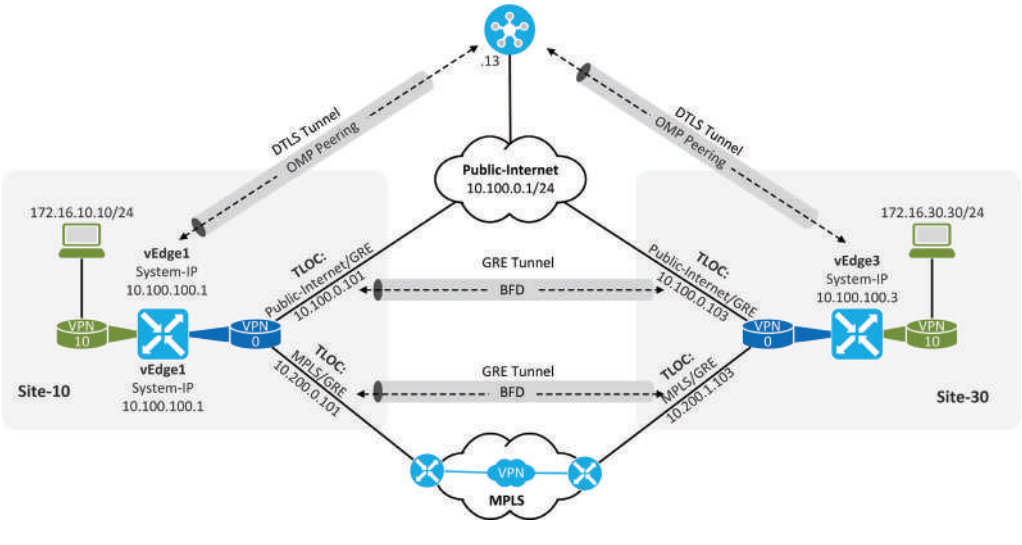


Figure 3-1: *SD-WAN Example Topology.*

Service VPN Configuration

This short section shows how to configure service VPNs in vEdge by using CLI. First, we create a VPN and attach an interface to it. Next, we assign an IP address to the interface, enable it and then commit changes. Examples 3-1 and 3-2 show the configuration steps.

```
vEdge-1(config)# vpn 10
vEdge-1(config-vpn-10)# interface ge0/2
vEdge-1(config-interface-ge0/2)# ip address 172.16.10.1/24
vEdge-1(config-interface-ge0/2)# no shutdown
vEdge-1(config-interface-ge0/2)# commit
```

Example 3-1: *VPN 10 Configuration on vEdge-1.*

```
vEdge-3(config)# vpn 10
vEdge-3(config-vpn-10)# interface ge0/3
vEdge-3(config-interface-ge0/3)# ip address 172.16.30.1/24
vEdge-3(config-interface-ge0/3)# no shutdown
vEdge-3(config-interface-ge0/3)# commit
```

Example 3-2: *VPN 10 Configuration on vEdge-3.*

TLOC Routes

Transport Location Identifier (TLOC) route advertisement carries the Overlay Network reachability information. The TLOC entry consists of three main elements and a set of attributes. The first element is the system IP address that is used, as its name describes, to identify the system/device. These addresses are only known by the SD-WAN vEdge devices and the control plane components. The Underlay Network devices are unaware of these IP addresses. The next TLOC element is the color which is used for identifying the transport network. The third element in the TLOC entry describes the tunneling protocol. I am using the Generic Route Encapsulation (GRE) protocol in order to show unencrypted data plane traffic. In real life, the other option, IPSec is the obvious choice. Each TLOC routes has also a set of attributes, just like we have in BGP.

The *Public-IP* address attribute describes the routable destination IP address used in the outer IP header (tunnel header). The *Private-IP* attribute in turn describes the physical IP address of the vEdge interface connected to a specific transport network. In a solution where vEdge is behind the NAT device, the Private-IP is translated to Public-IP which we are using in the tunnel header. When vEdge receives the TLOC routing advertisement, it knows that the advertising vEdge is behind the NAT device if these two addresses are different. The *Site-Id* attribute is self-explanatory, it identifies the site. The *Restrict* attribute in our example is set to one, which implies that we only want to build a tunnel (GRE or IPSec) between the vEdge interfaces connected to the same transport network. By default, this attribute is not set, which means that vEdges tries to establish full mesh tunnels between all transport interfaces no matter to which transport network (defined by color) they belong. As an example, without setting the restrict attribute vEdge1 in figure 3-2 tries to build GRE tunnels with vEdge3 from Public-Internet to Public-Internet but also from Public-Internet to MPLS and the same cross color connection over MPLS transport. This will not work since there is no routing between these two transport networks.

vEdges has OMP peering over the DTLS tunnel only with vSmart(s) that in turn distributes the received information to other vEdges. In figure 3-2 vEdge1 sends the TLOC route to vSmart that forwards it to vEdge3. Based on the TLOC routing information it builds two tunnels, one over MPLS and the other one over Public-Internet to vEdge using the GRE encapsulation. It uses the public-IP address as a next-hop address in the tunnel IP header. When the tunnel is up, vEdges start monitoring the tunnel transport capability by using Bi-directional Forwarding Detection (BFD). These tunnels are then used for Service VPN traffic sent between vEdges.

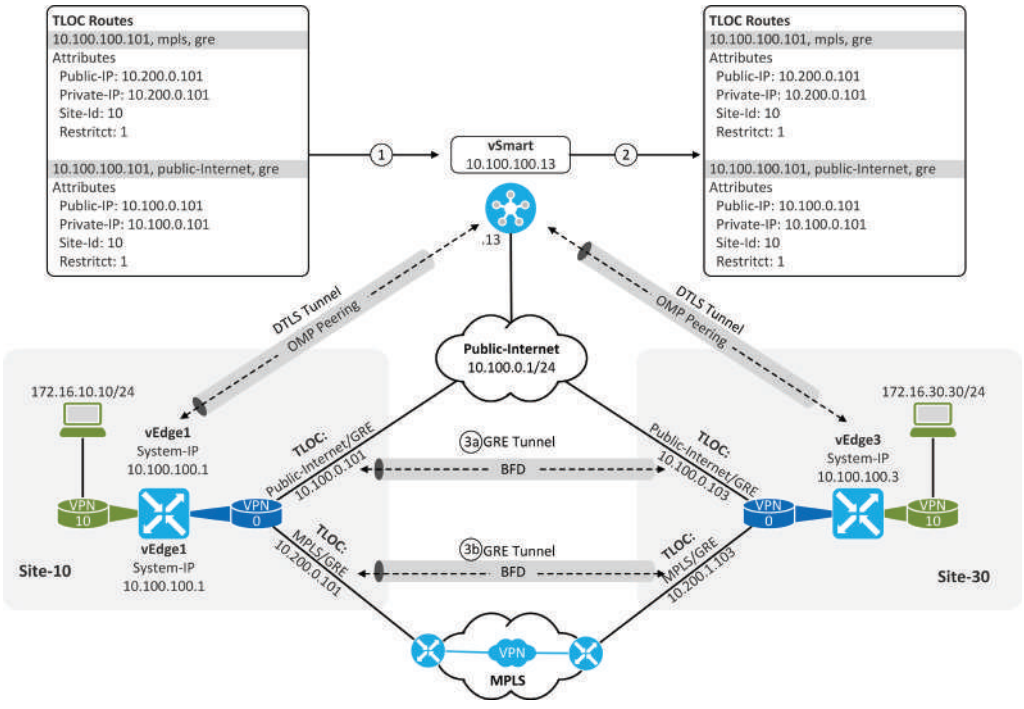


Figure 3-2: *OMP TLOC Peering and TLOC Route Advertisement.*

Examples 3-3, 3-4, and 3-5 verify that OMP peering is established only between the vSmart and vEdges.

```
vEdge-1# show omp peers
R -> routes received
I -> routes installed
S -> routes sent

                        DOMAIN    OVERLAY   SITE
PEER            TYPE    ID        ID        ID        STATE    UPTIME           R/I/S
-----------------------------------------------------------------------------------
10.100.100.13   vsmart  1         1         100       up       0:00:21:58       2/2/2
```

Example 3-3: *OMP peers of vEdge1.*

```
vEdge-3# show omp peers
R -> routes received
I -> routes installed
S -> routes sent

                        DOMAIN    OVERLAY   SITE
PEER            TYPE    ID        ID        ID        STATE    UPTIME           R/I/S
-----------------------------------------------------------------------------------
10.100.100.13   vsmart  1         1         100       up       0:00:23:01       2/2/2
```

Example 3-4: *OMP peers of vEdge3.*

```
vsmart# show omp peers
R -> routes received
I -> routes installed
S -> routes sent

                        DOMAIN    OVERLAY   SITE
PEER            TYPE    ID        ID        ID        STATE    UPTIME           R/I/S
-----------------------------------------------------------------------------------
10.100.100.101  vedge   1         1         10        up       0:00:23:42       2/0/2
10.100.100.103  vedge   1         1         30        up       0:00:23:40       2/0/2
```

Example 3-5: *OMP peers of vSmart.*

Example 3-6 shows the TLOC route originated by vEdge and advertised to vSmart.

```
vEdge-1# show omp tlocs advertised detail | exclude not | nomore

-------------------------------------------------
tloc entries for 10.100.100.101
                 mpls
                 gre
-------------------------------------------------
           RECEIVED FROM:
peer            0.0.0.0
status          C,Red,R
    Attributes:
     attribute-type    installed
     encap-key         0
     public-ip         10.200.0.101
     public-port       0
     private-ip        10.200.0.101
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     site-id           10
     preference        0
     weight            1
     version           3
    gen-id             0x80000010
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
           ADVERTISED TO:
peer    10.100.100.13
    Attributes:
     encap-key         0
     public-ip         10.200.0.101
     public-port       0
     private-ip        10.200.0.101
     private-port      0
```

```
     public-ip       ::
     public-port     0
     private-ip      ::
     private-port     0
     site-id           10
     preference        0
     weight            1
     version           3
     gen-id            0x80000010
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group

------------------------------------------------
tloc entries for 10.100.100.101
                 public-internet
                 gre
------------------------------------------------
            RECEIVED FROM:
peer            0.0.0.0
status          C,Red,R
    Attributes:
     attribute-type    installed
     encap-key         0
     public-ip         10.100.0.101
     public-port       0
     private-ip        10.100.0.101
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     site-id           10
     preference        0
     weight            1
     version           3
    gen-id             0x80000010
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
            ADVERTISED TO:
peer    10.100.100.13
    Attributes:
     encap-key         0
     public-ip         10.100.0.101
     public-port       0
     private-ip        10.100.0.101
     private-port      0
     public-ip       ::
     public-port     0
     private-ip      ::
     private-port     0
     site-id           10
     preference        0
```

```
     weight            1
     version           3
     gen-id            0x80000010
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
```

Example 3-6: *TLOC Route advertised to vSmart by vEdge1.*

Example 3-7 verify that vSmart has received two TLOC routes from vEdge1 which is also has advertised to vEdge3. The status of these routes indicates that vSmart has selected (C = chosen) this TLOC route to be installed (I) into the routing table and the next-hop address defined Public-IP attribute is reachable (R). Note that vSmart installs all valid TLOC routes into RIB but it does not install OMP routes describing Service VPNs into the RIB.

```
vsmart# show omp tlocs detail | exclude not | nomore | until tloc\ entries\ for\
10.100.100.103

----------------------------------------------------
tloc entries for 10.100.100.101
                 mpls
                 gre
----------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.101
status          C,I,R
    Attributes:
     attribute-type    installed
     encap-key         0
     public-ip         10.200.0.101
     public-port       0
     private-ip        10.200.0.101
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     site-id           10
     preference        0
     weight            1
     version           3
    gen-id             0x8000000f
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
            ADVERTISED TO:
peer    10.100.100.103
    Attributes:
```

```
     encap-key         0
     public-ip         10.200.0.101
     public-port       0
     private-ip        10.200.0.101
     private-port      0
     public-ip       ::
     public-port     0
     private-ip      ::
     private-port     0
     site-id           10
     preference        0
     weight            1
     version           3
     gen-id            0x8000000f
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group

--------------------------------------------------
tloc entries for 10.100.100.101
                 public-internet
                 gre
--------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.101
status          C,I,R
    Attributes:
     attribute-type    installed
     encap-key         0
     public-ip         10.100.0.101
     public-port       0
     private-ip        10.100.0.101
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     site-id           10
     preference        0
     weight            1
     version           3
    gen-id             0x8000000f
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
           ADVERTISED TO:
peer    10.100.100.103
    Attributes:
     encap-key         0
     public-ip         10.100.0.101
     public-port       0
     private-ip        10.100.0.101
     private-port      0
     public-ip       ::
     public-port     0
```

```
     private-ip      ::
     private-port     0
     site-id           10
     preference        0
     weight            1
     version           3
     gen-id            0x8000000f
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group

---------------------------------------------------
tloc entries for 10.100.100.103
```

Example 3-7: *TLOC Route Entry about vEdge1 TLOCs in vSmart.*

The example below verifies that vEdge3 has received the TLOC route and that it is also installed into the routing table (status C, I, R). Based on the *restrict* attribute vEdge3 only builds the GRE tunnels with vEdge only between the TLOCs sharing the same color (transport network identifier). vEdge uses its TLOC specific public-IP address as a source IP and the public IP received via TLOC route as a destination IP in GRE tunnel header when sending data packets.

```
vEdge-3# show omp tlocs ip 10.100.100.101 detail | exclude not\ set | nomore

---------------------------------------------------
tloc entries for 10.100.100.101
                 mpls
                 gre
---------------------------------------------------
           RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
    Attributes:
     attribute-type    installed
     encap-key         0
     public-ip         10.200.0.101
     public-port       0
     private-ip        10.200.0.101
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     site-id           10
     preference        0
     weight            1
     version           3
    gen-id             0x8000000f
     carrier           default
     restrict          1
```

```
     on-demand          0
     groups             [ 0 ]
     bandwidth          0
     qos-group          default-group

------------------------------------------------
tloc entries for 10.100.100.101
                 public-internet
                 gre
------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
    Attributes:
     attribute-type    installed
     encap-key         0
     public-ip         10.100.0.101
     public-port       0
     private-ip        10.100.0.101
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     site-id           10
     preference        0
     weight            1
     version           3
    gen-id             0x8000000f
     carrier           default
     restrict          1
     on-demand          0
     groups            [ 0 ]
     bandwidth         0
     qos-group         default-group
```

Example 3-8: *TLOC Route Entry about vEdge1 TLOCs in vEdge3.*

Example 3-9 shows that vEdge has two GRE tunnels with vEdge1 (10.100.100.101), one over Public-Internet and one over MPLS transport. There are no tunnels between different local-remote colors.

Tunnel Verification

```
vEdge-3# show tunnel statistics
tunnel stats gre 10.100.0.103 10.100.0.101 0 0
 system-ip      10.100.100.101
 local-color    public-internet
 remote-color   public-internet
 tunnel-mtu     1471
 tx_pkts        7027
 tx_octets      751722
 rx_pkts        7026
 rx_octets      653168
 tcp-mss-adjust 1391
tunnel stats gre 10.200.1.103 10.200.0.101 0 0
 system-ip      10.100.100.101
 local-color    mpls
 remote-color   mpls
 tunnel-mtu     1471
 tx_pkts        7061
 tx_octets      808604
 rx_pkts        7026
 rx_octets      653129
 tcp-mss-adjust 1391
```

Example 3-9: *GRE tunnel statistics.*

vEdge has also started the BFD session with vEdge1. Note, in order to install OMP routes about service VPN networks, there has to be an active BFD session to system-ID described in TLOC.

```
vEdge-3# show bfd sessions system-ip 10.100.100.101 | beg 10.

10.100.100.101 10  up public-internet public-internet  10.100.0.103  10.100.0.101  gre
10.100.100.101 10  up mpls            mpls             10.200.1.103  10.200.0.101  gre
```

Example 3-10: *BFD session to vEdge verification on vEdge3.*

Examples 3-11 and 3-12 shows that we have IP reachability over the transport networks between the Public-IP addresses.

```
vEdge-3# ping vpn 0 10.200.0.101 source 10.200.1.103 count 3
Ping in VPN 0
PING 10.200.0.101 (10.200.0.101) from 10.200.1.103 : 56(84) bytes of data.
64 bytes from 10.200.0.101: icmp_seq=1 ttl=61 time=2.54 ms
64 bytes from 10.200.0.101: icmp_seq=2 ttl=61 time=2.35 ms
64 bytes from 10.200.0.101: icmp_seq=3 ttl=61 time=2.26 ms

--- 10.200.0.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2003ms
rtt min/avg/max/mdev = 2.267/2.390/2.548/0.123 ms
```

Example 3-11: *BFD session to vEdge verification on vEdge3.*

```
vEdge-3# ping vpn 0 10.100.0.101 source 10.100.0.103 count 3
Ping in VPN 0
PING 10.100.0.101 (10.100.0.101) from 10.100.0.103 : 56(84) bytes of data.
64 bytes from 10.100.0.101: icmp_seq=1 ttl=64 time=1.40 ms
64 bytes from 10.100.0.101: icmp_seq=2 ttl=64 time=1.29 ms
64 bytes from 10.100.0.101: icmp_seq=3 ttl=64 time=5.46 ms

--- 10.100.0.101 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2001ms
rtt min/avg/max/mdev = 1.294/2.721/5.462/1.938 ms
```

Example 3-12: *BFD session to vEdge verification on vEdge3.*

OMP Routes

While TLOC routes are used for tunnel setup between vEdges, OMP routes are used for advertising reachability information related to Service VPN subnets. The OMP route entry includes VPN and the subnet information as well as a set of attributes. In our example, vEdge1 advertises its Service VPN 10 subnet 172.16.10.0/24 to vSmart. The advertisement carries attributes that describe the TLOCs and the label value that is used in data plane encapsulation as VPN identifier. vSmart verifies the reachability of the system-ID of TLOC (based on the TLOC route described in the previous section) before forwarding the advertisement to vEdge3. vSmart doesn't install the route in its RIB because it is not in the data path i.e. it never has to route packets to Service VPN subnets.

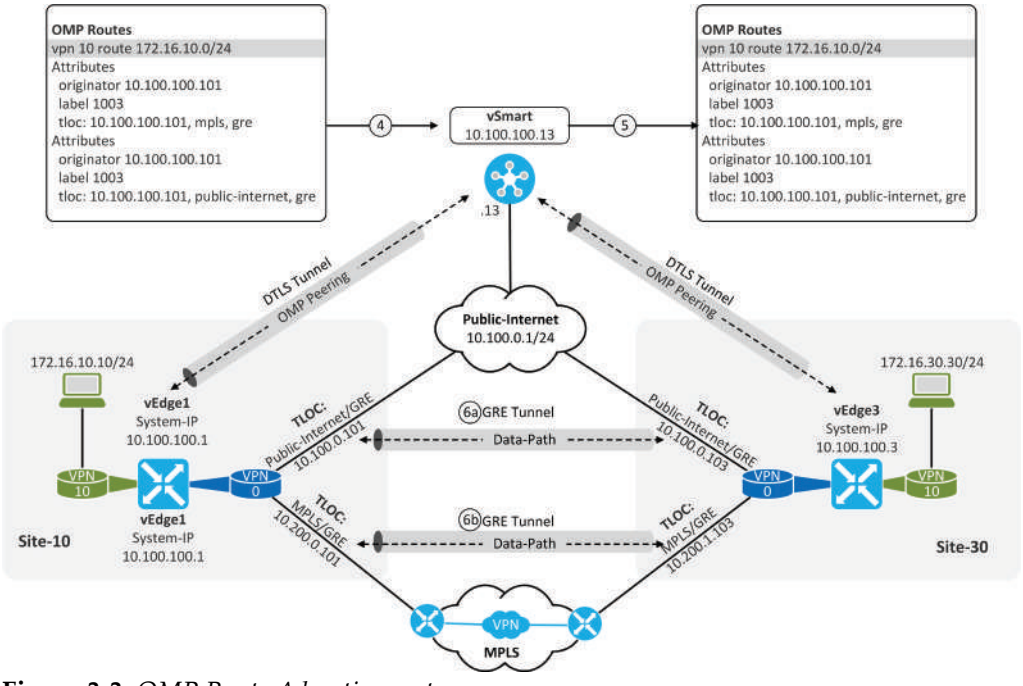


Figure 3-3: *OMP Route Advertisement.*

Example 3-13 shows the OMP route about VPN 10 subnet 172.16.10.0/24 from the vEdge perspective. The subnet is reachable through both transport networks. The VPN 10 is identified with label 1003. The *Red* in the status field means that route is redistributed into OMP (connected routes are redistributed automatically).

```
vEdge-1# show omp routes advertised detail | exclude not | nomore

-------------------------------------------------
omp route entries for vpn 10 route 172.16.10.0/24
-------------------------------------------------
            RECEIVED FROM:
peer            0.0.0.0
path-id         34
label           1003
status          C,Red,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, mpls, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
            RECEIVED FROM:
peer            0.0.0.0
```

```
path-id         37
label           1003
status          C,Red,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, public-internet, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
            ADVERTISED TO:
peer    10.100.100.13
    Attributes:
     originator       10.100.100.101
     label            1003
     path-id          34
     tloc             10.100.100.101, mpls, gre
     site-id          10
     overlay-id        1
     origin-proto     connected
     origin-metric    0
    Attributes:
     originator       10.100.100.101
     label            1003
     path-id          37
     tloc             10.100.100.101, public-internet, gre
     site-id          10
     overlay-id        1
     origin-proto     connected
     origin-metric    0
```

Example 3-13: *OMP route about VPN 10 subnet 172.16.10.0/24 on vEdge1.*

Example 3-14 shows that vSmart has received the OMP route. That status shows that route is selected (C) and the system-ID is reachable (R) so the update is valid and vSmart can forward the update to vEdge3.

```
vsmart# show omp routes received | until 172.16.30.0 | exclude not\ set | nomore

-------------------------------------------------
omp route entries for vpn 10 route 172.16.10.0/24
-------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.101
path-id         34
label           1003
status          C,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, mpls, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
            RECEIVED FROM:
peer            10.100.100.101
```

```
path-id         37
label           1003
status          C,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, public-internet, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0

-----------------------------------------------------
omp route entries for vpn 10 route 172.16.30.0/24
```

Example 3-14: *Received OMP route about VPN 10 subnet 172.16.10.0/24 on vSmart.*

Example 3-15 shows that vSmart has advertised the OMP routes to vEdge3.

```
vsmart# show omp routes advertised detail | until 172.16.30.0 | exclude not| nomore

-----------------------------------------------------
omp route entries for vpn 10 route 172.16.10.0/24
-----------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.101
path-id         34
label           1003
status          C,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, mpls, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
            RECEIVED FROM:
peer            10.100.100.101
path-id         37
label           1003
status          C,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, public-internet, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
            ADVERTISED TO:
peer    10.100.100.103
    Attributes:
     originator       10.100.100.101
     label            1003
     path-id          1
     tloc             10.100.100.101, mpls, gre
     site-id          10
     overlay-id        1
     origin-proto     connected
     origin-metric    0
```

```
    Attributes:
     originator       10.100.100.101
     label            1003
     path-id          2
     tloc             10.100.100.101, public-internet, gre
     site-id          10
     overlay-id        1
     origin-proto     connected
     origin-metric    0

----------------------------------------------------
omp route entries for vpn 10 route 172.16.30.0/24
```

Example 3-15: *Advertised OMP route about VPN 10 subnet 172.16.10.0/24 on vSmart.*

Example 3-16 shows that vEdge3 has received an update from vSmart. It also shows that vEdge3 has installed a route into the RIB. In order to install an OMP route to RIB, the TLOC IP address has to be reachable meaning there has to be a valid TLOC route that describes the Public-IP of TLOC IP. Besides, there also has to be an active BFD session with the originating vEdge1.

```
vEdge-3# show omp routes received | until 172.16.30.0 | exclude not\ set | nomore
----------------------------------------------------
omp route entries for vpn 10 route 172.16.10.0/24
----------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
path-id         1
label           1003
status          C,I,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, mpls, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
            RECEIVED FROM:
peer            10.100.100.13
path-id         2
label           1003
status          C,I,R
    Attributes:
     originator       10.100.100.101
     type             installed
     tloc             10.100.100.101, public-internet, gre
     overlay-id        1
     site-id          10
     origin-proto     connected
     origin-metric    0
```

Example 3-16: *Received OMP route about VPN 10 subnet 172.16.10.0/24 on vEdge3.*

Example 3-17 shows the routing entry about the VPN 10 subnet 172.16.10.0/24 on vEdge3 RIB. Note that the Administrative Distance (AD) is the same 250 as what IOS devices use for NHRP routes. If you are using IOS-XE devices in your SD-WAN solution, then the AD is 251.

```
vEdge-3# show ip routes vpn 10 detail | until 172.16.30.0/24 | nomore
Codes Proto-sub-type:
  IA -> ospf-intra-area, IE -> ospf-inter-area,
  E1 -> ospf-external1, E2 -> ospf-external2,
  N1 -> ospf-nssa-external1, N2 -> ospf-nssa-external2,
  e -> bgp-external, i -> bgp-internal
Codes Status flags:
  F -> fib, S -> selected, I -> inactive,
  B -> blackhole, R -> recursive, L -> import

------------------------------------------
 VPN 10      PREFIX 172.16.10.0/24
------------------------------------------
 proto           omp
 distance        250
 metric          0
 uptime          0:00:56:59
 tloc-ip         10.100.100.101
 tloc-color      mpls
 tloc-encap      gre
 nexthop-label   1003
 status          F,S

------------------------------------------
 VPN 10      PREFIX 172.16.10.0/24
------------------------------------------
 proto           omp
 distance        250
 metric          0
 uptime          0:00:56:59
 tloc-ip         10.100.100.101
 tloc-color      public-internet
 tloc-encap      gre
 nexthop-label   1003
 status          F,S

------------------------------------------
 VPN 10      PREFIX 172.16.30.0/24
```

Example 3-17: *RIB Verification in vEdge3.*

IP Reachability Verification

Examples 3-18 and 3-19 verifies that we have IP connectivity between subnets in site 10 and 30.

```
vEdge-3# ping vpn 10 172.16.10.10 source 172.16.30.1 count 2
Ping in VPN 10
PING 172.16.10.10 (172.16.10.10) from 172.16.30.1 : 56(84) bytes of data.
64 bytes from 172.16.10.10: icmp_seq=1 ttl=63 time=3.35 ms
64 bytes from 172.16.10.10: icmp_seq=2 ttl=63 time=2.68 ms
```

Example 3-18: *Dataplane Testing by Pinging from vEdge3 to VPN 10 IP address on vEdge1.*

```
Srv10> ping 172.16.10.10

84 bytes from 172.16.10.10 icmp_seq=1 ttl=62 time=4.143 ms
84 bytes from 172.16.10.10 icmp_seq=2 ttl=62 time=3.159 ms
```

Example 3-19: *Dataplane Testing by Pinging from host 172.16.30.30 to 172.16.10.10.*

Data Plane

Figure 3-4 shows the encapsulated ICMP messages when sent over transport networks. When the data path goes over the Public-Internet, there is only the tunnel header which in our case is GRE with VPN label. The packet sent over the MPLS transport first has the MPLS Transport label used within the MPLS network for label switching packets from ingress PE to egress PE. Then there is a VPN label that identifies the customer VPN in PE devices. Within these MPLS headers, there is a GRE tunnel header.

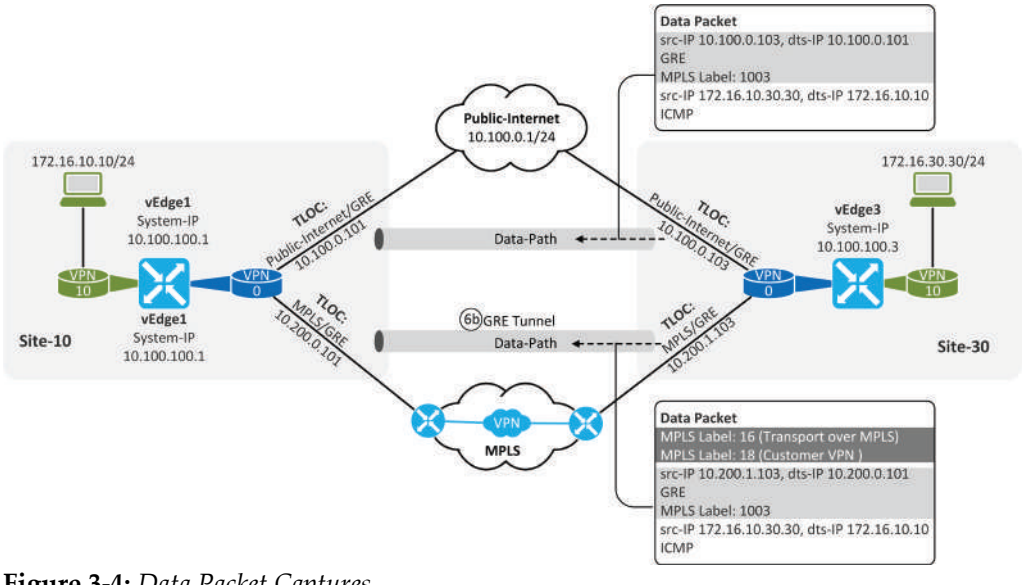


Figure 3-4: *Data Packet Captures.*

Capture 3-1 shows the encapsulated ICMP message from 172.16.30.30 to 172.16.10.10 over the Public-Internet transport network. The MPLS label is received with the OMP route and it is used as a VPN identifier in remote vEdge.

```
Internet Protocol Version 4, Src: 10.100.0.103, Dst: 10.100.0.101
Generic Routing Encapsulation (MPLS label switched packet)
MultiProtocol Label Switching Header, Label: 1003, Exp: 0, S: 1, TTL: 64
    0000 0000 0011 1110 1011 .... .... .... = MPLS Label: 1003
    .... .... .... .... .... 000. .... .... = MPLS Experimental Bits: 0
    .... .... .... .... .... ...1 .... .... = MPLS Bottom Of Label Stack: 1
    .... .... .... .... .... .... 0100 0000 = MPLS TTL: 64
Internet Protocol Version 4, Src: 172.16.30.30, Dst: 172.16.10.10
Internet Control Message Protocol
```

Capture 3-1: *Data Packet Captures from the Public-Internet transport.*

Capture 3-2 shows the encapsulated ICMP message from 172.16.30.30 to 172.16.10.10 over the MPLS transport network. So basically we are creating a GRE tunnel inside MPLS VPN. This increases overall complexity because in the MPLS transport network we need LDP or Segment Routing for exchanging label binding information for label switching (outer MPLS label) and BGP VPNv4 Unicast for exchanging the customer VRF reachability information (inner/customer MPLS label).

```
MultiProtocol Label Switching Header, Label: 16, Exp: 0, S: 0, TTL: 62
    0000 0000 0000 0001 0000 .... .... .... = MPLS Label: 16
    .... .... .... .... .... 000. .... .... = MPLS Experimental Bits: 0
    .... .... .... .... .... ...0 .... .... = MPLS Bottom Of Label Stack: 0
    .... .... .... .... .... .... 0011 1110 = MPLS TTL: 62
MultiProtocol Label Switching Header, Label: 18, Exp: 0, S: 1, TTL: 62
    0000 0000 0000 0001 0010 .... .... .... = MPLS Label: 18
    .... .... .... .... .... 000. .... .... = MPLS Experimental Bits: 0
    .... .... .... .... .... ...1 .... .... = MPLS Bottom Of Label Stack: 1
    .... .... .... .... .... .... 0011 1110 = MPLS TTL: 62
Internet Protocol Version 4, Src: 10.200.1.103, Dst: 10.200.0.101
Generic Routing Encapsulation (MPLS label switched packet)
MultiProtocol Label Switching Header, Label: 1003, Exp: 0, S: 1, TTL: 64
    0000 0000 0011 1110 1011 .... .... .... = MPLS Label: 1003
    .... .... .... .... .... 000. .... .... = MPLS Experimental Bits: 0
    .... .... .... .... .... ...1 .... .... = MPLS Bottom Of Label Stack: 1
    .... .... .... .... .... .... 0100 0000 = MPLS TTL: 64
Internet Protocol Version 4, Src: 172.16.30.30, Dst: 172.16.10.10
Internet Control Message Protocol
```

Capture 3-2: *Data Packet Captures from the MPLS transport.*

Summary

If we compare the control plane operation in the Viptela based SD-WAN solution to the VXLAN Fabric DC solution, we can see a lot of similarities. TLOC routes are used for advertising the tunnel endpoint public IP address while this task in VXLAN fabric is done by control plane protocol used in the underlay network (OSPF, IS-IS, or BGP). The customer subnet and VPN identifier (label) and encapsulation used in the data plane are advertised with OMP routes in Viptela SD-WAN, while in VXLAN fabric the same task is done by MP-BGP L3VPN EVPN update where the customer VPN is identified with Virtual Network Identifier (VNI). The Viptela based SD-WAN solution uses a centralized control plane model where updates are sent to vSamart that forwards valid updates to other vEdges. The same model is also used with VXLAN fabric, Leaf switches send updates only to Spine switches which forwards valid updates to other Leaf switches or Super-Spine switches in case we are using three tiers, 5-stage Clos fabric. So why there is again one new control-plane protocol doing the same job that can be done with existing, standard protocols? The OMP also distributes the encryption keys for tunnel encryption. BGP at the time of writing is not capable of doing that. Besides, OMP messages are sent over DTLS tunnels while BGP update messages are sent as clear text.

Chapter 4: Consideration When Using MPLS Transport

Introduction

In order to have IP connectivity between hosts A and B over the underlay transport network, we need to build a tunnel (IPSec or GRE) between the Public IP addresses of vEdge devices (TLOC Routes). Then we also need VPN-specific subnet routing information (OMP Routes) to be able to route traffic over the tunnel. This chapter discusses the role and operation of various protocols involved in Control Plane operations when an MPLS Transport network is used as an Underlay Network for SD-WAN solution. The first section introduces the *Segment Routing* solution for building a *Label Switch Path (LSP)* between PE routers over the MPLS backbone by using the IS-IS routing protocol for both routing and label distribution. The second section explains how to build L3VPN between vEdge Public IP addresses over the LSP. Figure 4-1 shows the high-level routing model used in this chapter.

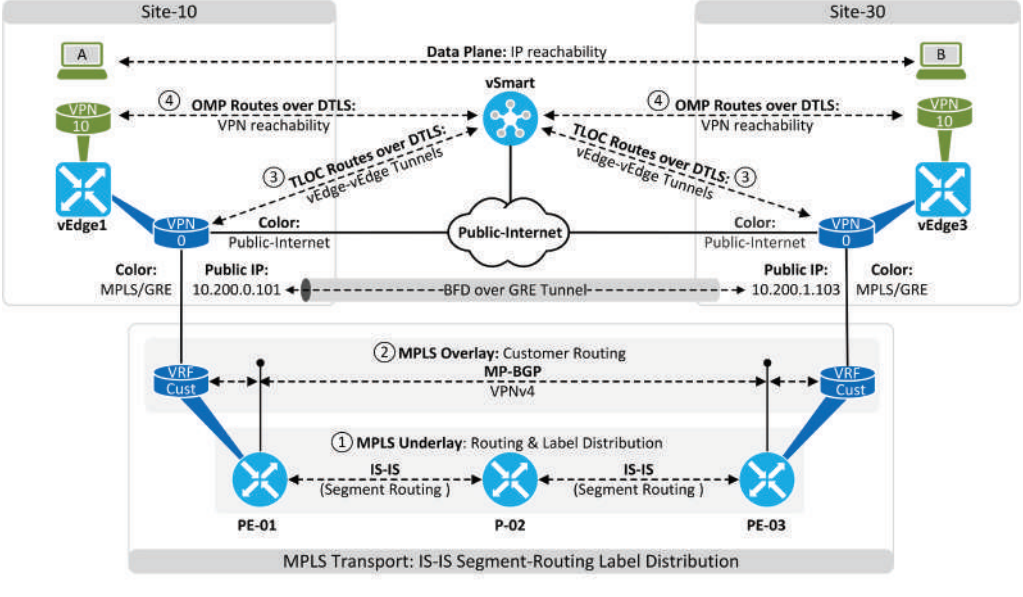


Figure 4-1: *Control Plane Model.*

Building a Label Switch Path

In order to build a *Labels Switch Path (LSP)* between PE devices, we need a routing protocol for IP reachability and MPLS labels for subnet/label binding. This is because forwarding decision within MPLS network is based on labels, not IP addresses. Label/Subnet binding information in traditional MPLS networks is done by using *Label Distribution Protocol (LDP)*. Having one protocol for routing and another for the label distribution, however, increases overall complexity. The Segment-Routing is developed for simplifying the solution. It uses a routing protocol for advertising both IP reachability information and MPLS label-related information. The reason why I use the term "IP reachability" instead of the term "route" is that the most common IGP protocols in MPLS Underlay are OSPF and IS-IS and which don't advertise routes but link-state information. Segment-Routing in turn doesn't advertise label/subnet binding information, it advertises a label range and then the index number for each destination.

Segment Routing Global Block (SRGB)

Segment-Routing Global Block (SRGB) defines the label range from where a Link-State IGP allocates *IGP Prefix-SID* (Prefix Segment Identifier). In practice, IGP Prefix-SID is a node-specific Prefix to SID mapping identifier (Label) within the MPLS network. We are using the SRGB 16000 – 23999 in our examples (figure 4-2). When the IS-IS Segment-Routing protocol extension is enabled on a router, it advertises the SRGB label range (8000) within *SR Capability sub-TLV (Type-2)* carried in the *IS-IS Router Capability TLV (Type-242)*. The SRGB base label (16000) in turn is advertised by using the *Prefix-SID/Label sub-TLV (Type-1)* carried within the *SR Capability sub-TLV.*

Note: IS-IS uses *Links State Protocol (LSP) Data Units* for advertising information to its IS-IS neighbors. Information is encoded as Type/Length/Value (TLV) fields.

IGP Prefix Segment (Prefix-SID)

IS-IS Segment-Routing extension uses the *Extended IP reachability TLV (Type-135)* for advertising the node IP and its metric. Its *Prefix-SID sub-TLV (Type-3)* describes the *index number/label value* that is used by receiving IS-IS routers for calculating the label associated with the advertised node IP address. The Index/value is derived from the statically configured value, either as an absolute label value or as an index number. Note that the value has to be a unique, node-specific value. As an example, in PE-01 we are using absolute label value 16001. The index value is calculated by subtracting SRGB base value 16000 from the defined absolute label value 16001 which gives us an index value 1 (16001 – 16000 = 1). The receiving IS-IS router calculates the label value by adding the index value to the SRGB base value which gives the label 16001 (16000 + 1 = 16001). If the absolute label value is signaled, every router should use the same SRGB.

The IS-IS Segment Routing configuration of PE-01, P-02, and PE-02 can be found at the end of this chapter.

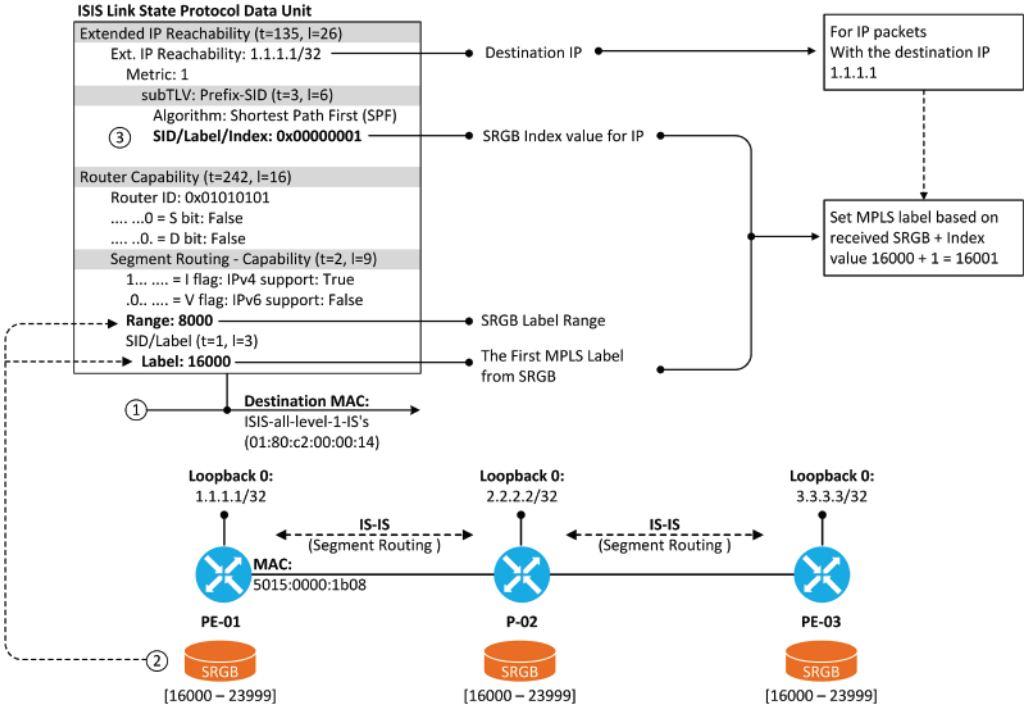


Figure 4-2: *MPLS Transport Underlay: IS-IS Link State Packet sent by PE-01.*

Capture 4-1 shows the complete IS-IS LSP advertised by PE-01.

```
IEEE 802.3 Ethernet
    Destination: ISIS-all-level-1-IS's (01:80:c2:00:00:14)
    Source: 50:15:00:00:1b:08 (50:15:00:00:1b:08)
    Length: 134
Logical-Link Control
ISO 10589 ISIS InTRA Domain Routeing Information Exchange Protocol
ISO 10589 ISIS Link State Protocol Data Unit
    PDU length: 131
    Remaining lifetime: 1199
    LSP-ID: 0000.0000.0001.00-00
    Sequence number: 0x00000018
    Checksum: 0xac69 [correct]
    [Checksum Status: Good]
    Type block(0x01): Partition Repair:0, Attached bits:0, Overload bit:0, IS type:1
    Area address(es) (t=1, l=2)
    Protocols supported (t=129, l=1)
    Traffic Engineering Router ID (t=134, l=4)
    IP Interface address(es) (t=132, l=4)
        Type: 132
        Length: 4
        IPv4 interface address: 1.1.1.1
    Hostname (t=137, l=5)
    Extended IS reachability (t=22, l=30)
    Extended IP Reachability (t=135, l=26)
        Type: 135
        Length: 26
        Ext. IP Reachability: 10.1.2.0/24
            Metric: 40
            0... .... = Distribution: Up
            .0.. .... = Sub-TLV: No
            ..01 1000 = Prefix Length: 24
            IPv4 prefix: 10.1.2.0
            no sub-TLVs present
        Ext. IP Reachability: 1.1.1.1/32
            Metric: 1
            0... .... = Distribution: Up
            .1.. .... = Sub-TLV: Yes
            ..10 0000 = Prefix Length: 32
            IPv4 prefix: 1.1.1.1
            SubCLV Length: 8
            subTLV: Prefix-SID (t=3, l=6)
                Code: Prefix-SID (3)
                Length: 6
                Flags: 0x40, Node-SID
                Algorithm: Shortest Path First (SPF) (0)
                SID/Label/Index: 0x00000001
    Router Capability (t=242, l=16)
        Type: 242
        Length: 16
        Router ID: 0x01010101
        .... ...0 = S bit: False
        .... ..0. = D bit: False
        Segment Routing - Capability (t=2, l=9)
            1... .... = I flag: IPv4 support: True
            .0.. .... = V flag: IPv6 support: False
            Range: 8000
            SID/Label (t=1, l=3)
                Label: 16000
```

Capture 4-1: *IS-IS Link State Packet sent by PE-01.*

Example 4-1 shows that PE-03 knows the SRGB range as well as Prefix-SID/Index associated with IP 1.1.1.1/32.

```
PE-03# sh isis database detail PE-01.00-00
IS-IS Process: SR LSP database VRF: default
IS-IS Level-1 Link State Database
  LSPID                 Seq Number   Checksum  Lifetime   A/P/O/T
  PE-01.00-00           0x00000012   0x6AB1    968        0/0/0/1
    Instance      :  0x0000000D
    Area Address  :  49
    NLPID         :  0xCC
    Router ID     :  1.1.1.1
    IP Address    :  1.1.1.1
    Hostname      :  PE-01              Length : 5
    Extended IS   :  P-02.00            Metric : 40
      Interface IP Address :  10.1.2.1
      IP Neighbor Address :  10.1.2.2
      ADJ-SID         :  16                 Flags  : V/L,  Weight 1
    Extended IP   :         1.1.1.1/32  Metric : 1           (U)
      Prefix-SID  :                 1   Algo   : 0    Flags : N
    Extended IP   :        10.1.2.0/24  Metric : 40          (U)
    Capability    : Router-Id 1.1.1.1          Flags 0x0
      SR-Range        :  16000 - 23999 (8000) Flags I--
    Digest Offset :  0

IS-IS Level-2 Link State Database
  LSPID                 Seq Number   Checksum  Lifetime   A/P/O/T
```

Example 4-1: *show isis database detail PE-01.00-00 on PE-03.*

The IPv4 Forwarding Equivalency Class (FEC) under VRF default in example 4-2 shows that PE-03 uses MPLS label 16001 in the outer MPLS header when sending data packets to PE-01 (1.1.1.1/32). Deaggregatin FEC is customer VRF related and there we can see the customer VPN Label advertised by MP-BGP. We will focus on that in the next section. The Adjacency SID (ADJ SID) describes the Inter-Router links.

```
PE-03#  sh mpls switching detail

VRF default
IPv4 FEC
 In-Label                    : 16001
 Out-Label stack             : 16001
 FEC                         : 1.1.1.1/32
 Out interface               : Eth1/3
 Next hop                    : 10.2.3.2
 Input traffic statistics    : 0 packets, 0 bytes
 Output statistics per label : label 16001, 0 packets, 0 bytes

Deaggregation FEC type
```

```
 In-Label                        : 492287
 VRF                             : Customer-77
 Address-Family                  : IPv4
 Input traffic statistics        : 0 packets 0 bytes

ADJ SID
 In-Label                        : 16
 Out-Label stack                 : 3
 FEC                             : 10.2.3.2
 Out interface                   : Eth1/3
 Next hop                        : 10.2.3.2
 Input traffic statistics        : 0 packets, 0 bytes
 Output statistics per label     : label 3, 0 packets, 0 bytes


*Label statistics accurate as of 72 seconds ago

Block       Label-Range
1           16000 - 23999
```

Example 4-2: *show isis database detail PE-01.00-00 on PE-03.*

Example 4-3 shows that PE-03 is installed the information in its RIB. Whenever PE-03 has something to send to 1.1.1.1/32, it pushes the label 16001 as a top label for the packet. In case that traffic is received from the customer VRF, PE-03 also adds the VPN label as a bottom of stack label.

```
PE-03# show ip route 1.1.1.1 isis-SR detail
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

1.1.1.1/32, ubest/mbest: 1/0
    *via 10.2.3.2, Eth1/3, [115/81], 00:05:37, isis-SR, L1 (mpls)
         MPLS[0]: Label=16001 E=0 TTL=255 S=0
         client-specific data: 41
```

Example 4-3: *show ip route 1.1.1.1 isis-SR detail on PE-03.*

Examples 4-4 and 4-5 show the same show commands from the PE-01 perspective.

```
PE-01# sh isis database detail PE-03.00-00
IS-IS Process: SR LSP database VRF: default
IS-IS Level-1 Link State Database
  LSPID                 Seq Number   Checksum  Lifetime   A/P/O/T
  PE-03.00-00           0x00000012   0xF5F7    1182       0/0/0/1
    Instance      :  0x0000000D
    Area Address  :  49
    NLPID         :  0xCC
    Router ID     :  3.3.3.3
    IP Address    :  3.3.3.3
    Hostname      :  PE-03              Length : 5
    Extended IS   :  P-02.00            Metric : 40
      Interface IP Address :  10.2.3.3
      IP Neighbor Address :  10.2.3.2
      ADJ-SID         :  16                Flags  : V/L,  Weight 1
    Extended IP   :         3.3.3.3/32  Metric : 1           (U)
      Prefix-SID  :                 3  Algo   : 0    Flags : N
    Extended IP   :        10.2.3.0/24  Metric : 40          (U)
    Capability    : Router-Id 3.3.3.3          Flags 0x0
      SR-Range        :  16000 - 23999 (8000) Flags I--
    Digest Offset :  0

<snipped>
```

Example 4-4: *show isis database detail PE-03.00-00 on PE-01.*

```
PE-01# sh mpls switching detail

VRF default
IPv4 FEC
 In-Label                      : 16003
 Out-Label stack               : 16003
 FEC                           : 3.3.3.3/32
 Out interface                 : Eth1/2
 Next hop                      : 10.1.2.2
 Input traffic statistics      : 0 packets, 0 bytes
 Output statistics per label   : label 16003, 0 packets, 0 bytes

Deaggregation FEC type
 In-Label                      : 492287
 VRF                           : Customer-77
 Address-Family                : IPv4
 Input traffic statistics      : 0 packets 0 bytes

ADJ SID
 In-Label                      : 16
 Out-Label stack               : 3
 FEC                           : 10.1.2.2
 Out interface                 : Eth1/2
 Next hop                      : 10.1.2.2
 Input traffic statistics      : 0 packets, 0 bytes
 Output statistics per label   : label 3, 0 packets, 0 bytes
<snipped>
```

Example 4-5: *sh mpls switching detail on PE-01.*

MP-BGP: Advertising Customer Routes

MP-BGP is used for advertising customer routes between PE devices. BGP uses IPv4/Labeled VPN Unicast afi/safi for advertising VPNv4 addresses (Route-Distinguisher:IPv4 prefix). BGP Update carries the IPv4 prefix, its associated RD, and VPN Label value. The VPN label value within Segment-Routing enabled devices is taken from the SR Dynamic range (default range 24000 - 1048575). Besides, MP-BGP Update carries Route-Target extended community which is used for BGP export/import policy. In our example, PE-01 advertises its customer-specific subnet 10.200.0.0/24 with RD 65077:77 and VPN Label 492287. The BGP Update message is label switched across the MPLS transport meaning P-02 forwards packet based on MPLS label 16003, which is the label used with the destination IP address 3.3.3.3/32.

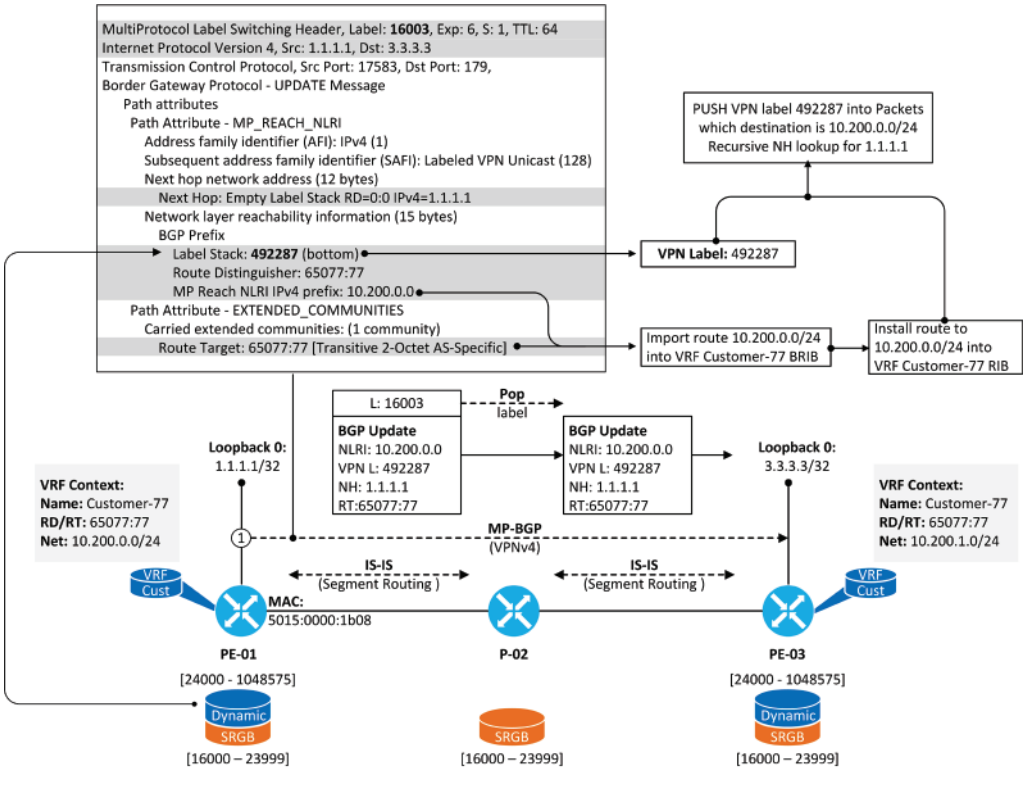


Figure 4-3: MPLS Transport Overlay: *MP-BGP Update by PE-01.*

Capture 4-2 shows the complete MP-BGP packet sent by PE-01.

```
MultiProtocol Label Switching Header, Label: 16003, Exp: 6, S: 1, TTL: 64
Internet Protocol Version 4, Src: 1.1.1.1, Dst: 3.3.3.3
Transmission Control Protocol, Src Port: 17583, Dst Port: 179, Seq: 106, Ack: 228,
Len: 132
Border Gateway Protocol - UPDATE Message
    Marker: ffffffffffffffffffffffffffffffff
    Length: 84
    Type: UPDATE Message (2)
    Withdrawn Routes Length: 0
    Total Path Attribute Length: 61
    Path attributes
        Path Attribute - MP_REACH_NLRI
            Flags: 0x90, Optional, Extended-Length, Non-transitive, Complete
            Type Code: MP_REACH_NLRI (14)
            Length: 32
            Address family identifier (AFI): IPv4 (1)
            Subsequent address family identifier (SAFI): Labeled VPN Unicast (128)
            Next hop network address (12 bytes)
                Next Hop: Empty Label Stack RD=0:0 IPv4=1.1.1.1
            Number of Subnetwork points of attachment (SNPA): 0
            Network layer reachability information (15 bytes)
                BGP Prefix
                    Prefix Length: 112
                    Label Stack: 492287 (bottom)
                    Route Distinguisher: 65077:77
                    MP Reach NLRI IPv4 prefix: 10.200.0.0
        Path Attribute - ORIGIN: IGP
        Path Attribute - AS_PATH: empty
        Path Attribute - LOCAL_PREF: 100
        Path Attribute - EXTENDED_COMMUNITIES
            Flags: 0xc0, Optional, Transitive, Complete
            Type Code: EXTENDED_COMMUNITIES (16)
            Length: 8
            Carried extended communities: (1 community)
                Route Target: 65077:77 [Transitive 2-Octet AS-Specific]
```

Capture 4-2: *MP-BGP update sent by PE-01.*

Example 4-6 show the BGP table of PE-03 concerning subnet 10.200.0.0./24 that is attach to customer VRF in PE-01. We can see that VPN label 492287 is associated with network 10.200.0.0/24 with the next-hop 1.1.1.1.

```
PE-03# sh bgp vpnv4 unicast 10.200.0.0/24
BGP routing table information for VRF default, address family VPNv4 Unicast
Route Distinguisher: 65077:77    (VRF Customer-77)
BGP routing table entry for 10.200.0.0/24, version 7
Paths: (1 available, best #1)
Flags: (0x8008001a) (high32 00000000) on xmit-list, is in urib, is best urib
route, is in HW
  vpn: version 11, (0x00000000100002) on xmit-list

  Advertised path-id 1, VPN AF advertised path-id 1
  Path type: internal, path is valid, imported same remote RD, is best path,
in
rib
  AS-Path: NONE, path sourced internal to AS
    1.1.1.1 (metric 81) from 1.1.1.1 (1.1.1.1)
      Origin IGP, MED not set, localpref 100, weight 0
      Received label 492287
      Extcommunity: RT:65077:77

  VRF advertise information:
  Path-id 1 not advertised to any peer

  VPN AF advertise information:
  Path-id 1 not advertised to any peer
```

Example 4-6: *sh bgp vpnv4 unicast 10.200.0.0/24 on PE-03.*

The information is installed from the BGP table into the routing table. Examples 4-7 and 4-8 illustrate the recursive next-hop resolution and verify that PE-03 uses MPLS label 16001 when forwarding customer traffic over the MPLS transport network.

```
PE-03# show ip route detail vrf Customer-77 | sec 10.200.0.0
10.200.0.0/24, ubest/mbest: 1/0
    *via 1.1.1.1%default, [200/0], 00:25:37, bgp-65077, internal, tag 65077
(mpls-vpn)
         MPLS[0]: Label=492287 E=0 TTL=0 S=0 (VPN)
         client-specific data: 2
         recursive next hop: 1.1.1.1/32%default
         extended route information: BGP origin AS 65077 BGP peer AS 65077
```

Example 4-7 *show ip route detail vrf Customer-77 | sec 10.200.0.0 on PE-03.*

```
PE-03#  show ip route 1.1.1.1 detail
IP Route Table for VRF "default"
'*' denotes best ucast next-hop
'**' denotes best mcast next-hop
'[x/y]' denotes [preference/metric]
'%<string>' in via output denotes VRF <string>

1.1.1.1/32, ubest/mbest: 1/0
    *via 10.2.3.2, Eth1/3, [115/81], 00:26:43, isis-SR, L1 (mpls)
         MPLS[0]: Label=16001 E=0 TTL=255 S=0
         client-specific data: 41
```

Example 4-8 *show ip route 1.1.1.1 detail on PE-03.*

When the MPLS Underlay Network routing and label binding process is done and the BGP Updates are sent, vEdges have IP connectivity and they can establish a tunnel between them and sent BFD messages by using it. At this phase the TLOC Route sent to vSmart is valid. Figure 4-4 shows that BFD messages are encapsulated with a label stack where the inner VPN label defines the customer VRF and the outer MPLS label is used for forwarding packets to the destination.

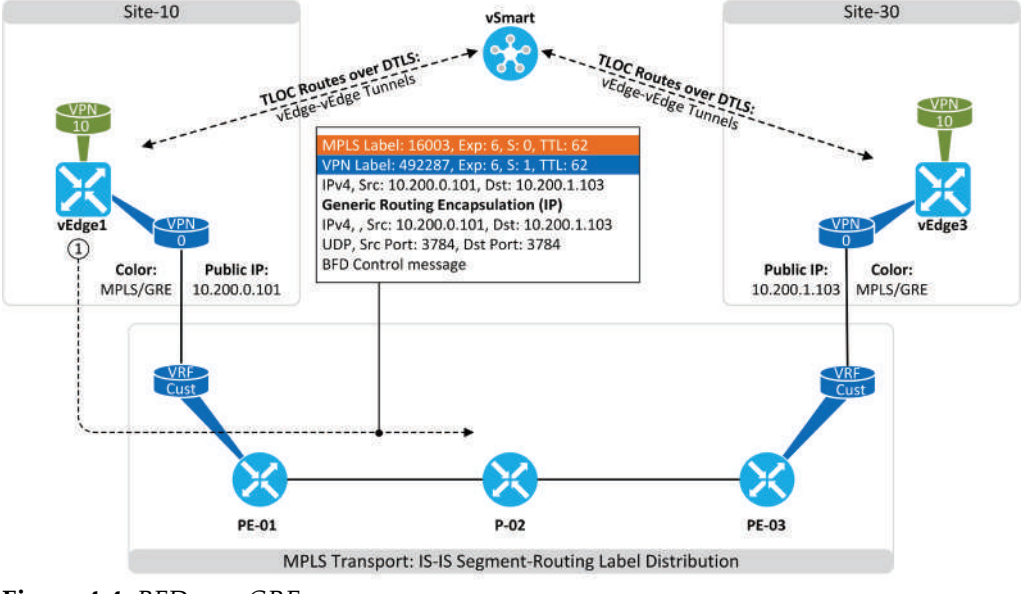


Figure 4-4: *BFD over GRE.*

Capture 4-3 shows the complete captured packet.

```
MultiProtocol Label Switching Header, Label: 16003, Exp: 6, S: 0, TTL: 62
MultiProtocol Label Switching Header, Label: 492287, Exp: 6, S: 1, TTL: 62
Internet Protocol Version 4, Src: 10.200.0.101, Dst: 10.200.1.103
Generic Routing Encapsulation (IP)
Internet Protocol Version 4, Src: 10.200.0.101, Dst: 10.200.1.103
User Datagram Protocol, Src Port: 3784, Dst Port: 3784
BFD Control message
```

Capture 4-3: *BFD Message Sent by PE-01.*

When the GRE tunnel is established between vEdge-1 and vEdge-3 user data can be routed over it. Just for the recap, VPN routes in vEdges are advertised to vSmart as OMP routes including all valid TLOCs (Public IP address, Colour, and Encapsulation) that can be used for routing packets towards the advertised VPN subnet. The TLOC is valid when BFD messages can be exchanged between vEdges attached to the same color. As can be seen from figure 4-5 and capture 4-4 there are three label values when data is sent across the MPLS transport network; one for the LSP between PE devices, one as an MPLS customer VRF identifier, and one for the client VPN identifier in vEdges.

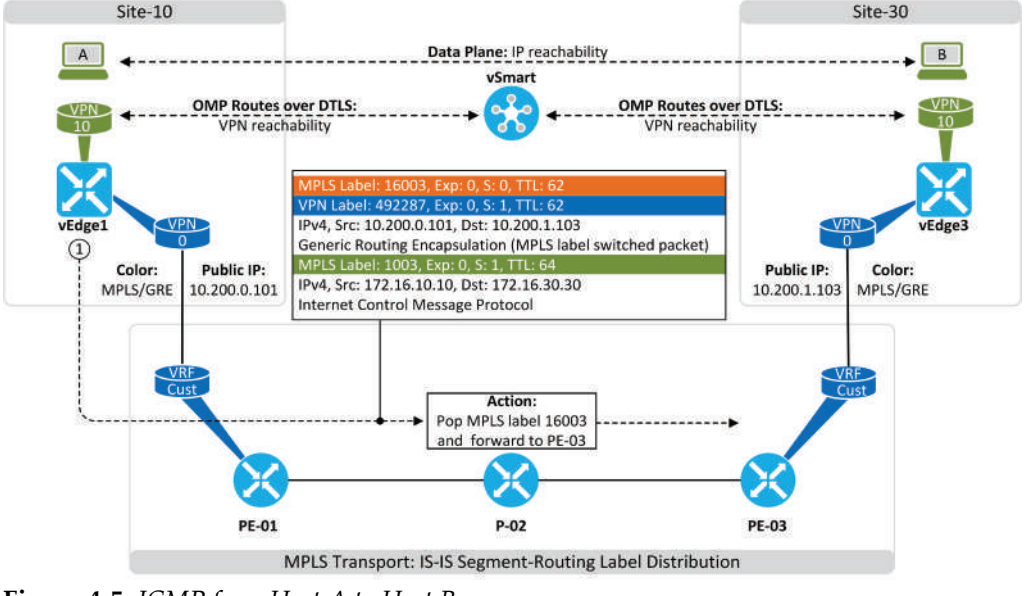


Figure 4-5: *ICMP from Host A to Host B.*

Capture 4-4 shows the complete captured packet.

```
MultiProtocol Label Switching Header, Label: 16003, Exp: 0, S: 0, TTL: 62
MultiProtocol Label Switching Header, Label: 492287, Exp: 0, S: 1, TTL: 62
Internet Protocol Version 4, Src: 10.200.0.101, Dst: 10.200.1.103
Generic Routing Encapsulation (MPLS label switched packet)
MultiProtocol Label Switching Header, Label: 1003, Exp: 0, S: 1, TTL: 64
Internet Protocol Version 4, Src: 172.16.10.10, Dst: 172.16.30.30
Internet Control Message Protocol
```

Capture 4-4: *ICMP Request Sent by Host A to Host B.*

Summary

When we are running SD-WAN over the MPLS transport network we need to understand what information is needed and how it is advertised. First, we need to build an LSP between PE devices. In our example, this was done by using the IS-IS Segment-Routing protocol extension. Then we need IP connectivity between vEdge devices. This is done by advertising networks associated with customer VRF, where vEdges are connected, by using MP-BGP. In addition, we need to advertise TLOC routes and OMP routes to vSmart and from there to other vEdges. It is crucial to understand the relationship between control plane protocols IS-IS, BGP, and OMP and this way to understand how the system works.

The reason why I wrote this chapter is that I wanted readers to understand the additional complexity coming with MPLS transport compared to Internet transport where we only rely on routing. My intent is not to say that don't use MPLS transport.

MPLS device configurations

```
PE-01# sh run
hostname PE-01
install feature-set mpls
feature-set mpls
feature bgp
feature isis
feature mpls l3vpn
feature mpls segment-routing
feature mpls oam
feature mpls segment-routing traffic-engineering

segment-routing
  mpls
    connected-prefix-sid-map
      address-family ipv4
        1.1.1.1/32 absolute 16001
vrf context Customer-77
  rd 65077:77
  address-family ipv4 unicast
    route-target import 65077:77
    route-target export 65077:77
vrf context management

interface Ethernet1/1
  no switchport
  vrf member Customer-77
  ip address 10.200.0.1/24
  no shutdown

interface Ethernet1/2
  no switchport
  ip address 10.1.2.1/24
  isis network point-to-point
  ip router isis SR
  mpls ip forwarding
  no shutdown
!
interface loopback0
  ip address 1.1.1.1/32
  ip router isis SR
icam monitor scale

router isis SR
  net 49.0000.0000.0001.00
  is-type level-1
  address-family ipv4 unicast
    segment-routing mpls
router bgp 65077
  router-id 1.1.1.1
  address-family ipv4 unicast
```

```
  address-family vpnv4 unicast
  neighbor 3.3.3.3
    remote-as 65077
    update-source loopback0
    address-family ipv4 unicast
    address-family vpnv4 unicast
      send-community extended
  vrf Customer-77
    address-family ipv4 unicast
      network 10.200.0.0/24
```

```
PE-03# sh run
hostname PE-03
install feature-set mpls
feature-set mpls
feature bgp
feature isis
feature mpls l3vpn
feature mpls segment-routing
feature mpls oam
feature mpls segment-routing traffic-engineering

vlan 1
segment-routing
  mpls
    connected-prefix-sid-map
      address-family ipv4
        3.3.3.3/32 absolute 16003

vrf context Customer-77
  rd 65077:77
  address-family ipv4 unicast
    route-target import 65077:77
    route-target export 65077:77
vrf context management

interface Ethernet1/1
  no switchport
  vrf member Customer-77
  ip address 10.200.1.1/24
  no shutdown

interface Ethernet1/3
  no switchport
  ip address 10.2.3.3/24
  isis network point-to-point
  ip router isis SR
  mpls ip forwarding
  no shutdown

interface loopback0
  ip address 3.3.3.3/32
  ip router isis SR
```

```
icam monitor scale

line console
line vty
boot nxos bootflash:/nxos.9.3.5.bin sup-1

router isis SR
  net 49.0000.0000.0003.00
  is-type level-1
  address-family ipv4 unicast
    segment-routing mpls
router bgp 65077
  router-id 3.3.3.3
  address-family ipv4 unicast
  address-family vpnv4 unicast
  neighbor 1.1.1.1
    remote-as 65077
    update-source loopback0
    address-family vpnv4 unicast
      send-community extended
  vrf Customer-77
    address-family ipv4 unicast
      network 10.200.1.0/24
```

```
P-02# sh run
hostname P-02
install feature-set mpls
feature-set mpls
feature bgp
feature isis
feature mpls l3vpn
feature mpls segment-routing
feature mpls oam
feature mpls segment-routing traffic-engineering

vlan 1
segment-routing
  mpls
    connected-prefix-sid-map
      address-family ipv4
        2.2.2.2/32 absolute 16002

interface Ethernet1/2
  no switchport
  ip address 10.1.2.2/24
  isis network point-to-point
  ip router isis SR
  mpls ip forwarding
  no shutdown

interface Ethernet1/3
  no switchport
  ip address 10.2.3.2/24
  isis network point-to-point
```

```
  ip router isis SR
  mpls ip forwarding
  no shutdown

interface loopback0
  ip address 2.2.2.2/32
  ip router isis SR

router isis SR
  net 49.0000.0000.0002.00
  is-type level-1
  address-family ipv4 unicast
    segment-routing mpls
```

Chapter 5: Policies – Topology: Hub and Spoke

Introduction

Cisco Viptela SD-WAN solution builds a full-mesh topology between vEdge devices by default when there are no Control Policies implemented. This means that vEdges tries to build an IPSec/GRE tunnel to every reachable TLOC public IP addresses no matter which site or color (transport network) TLOCs belong to. We have already change the default behavior by using the *restrict* option (chapter 2) under tunnel interfaces. In this way, tunnels are only established between TLOCs belonging to the same color. In this chapter, we are going to create a Hub and Spoke topology by implementing a Control Policy where the vSmart advertises TLOC/OMP routes from site 30 to sites 10 and 20 and TLOC/OMP routes from sites 10 and 20 to site 30. vSmart doesn't advertise TLOC/OMP routes between sites 10 and 20. Site 10 and 20 will be our Branch/Remote sites and site 30 will be the Hub/DataCenter site.

Figure 5-1 recaps the operation of the Overlay Management Protocol (OMP). vEdge1 in site 10 advertises TLOC route advertisement to vSmart where it describes its System Id, transport color, and encapsulation method as well as Public/Private IP and restricts attributes (among several other attributes). vSmart forwards TLOC routes received from vEdge1 to both vEdge2 and vEdge3. vEdge1 also advertises OMP routes where it describes the reachability information about its local subnet 172.16.10.0/24 bound to VPN10.

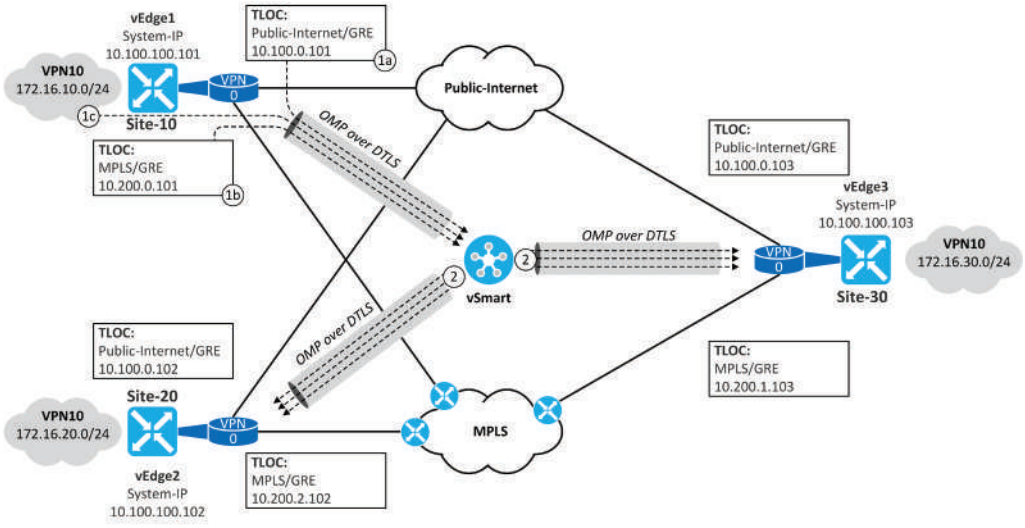


Figure 5-1: *TLOC Route advertisement.*

When vEdge2 and vEdge3 receive TLOC route advertisement, they can establish GRE tunnels with vEdge-1 between the same colors (restrict option) using the received public IP address as a tunnel destination. They also start the BFD session over each tunnel to monitor the tunnel's health. When tunnels are up and running vEdges can send data by using a public IP address as a destination IP address in the tunnel header. Figure 5-2 illustrates how vEdges establish color-to-color tunnels between themselves. vEdge1 has an mpls-to-mpls tunnel with both vEdge2 and vEdge3 as well as an internet-to-internet tunnel with both vEdges. vEdge2 and vEdge3 have also tunnels between themselves.

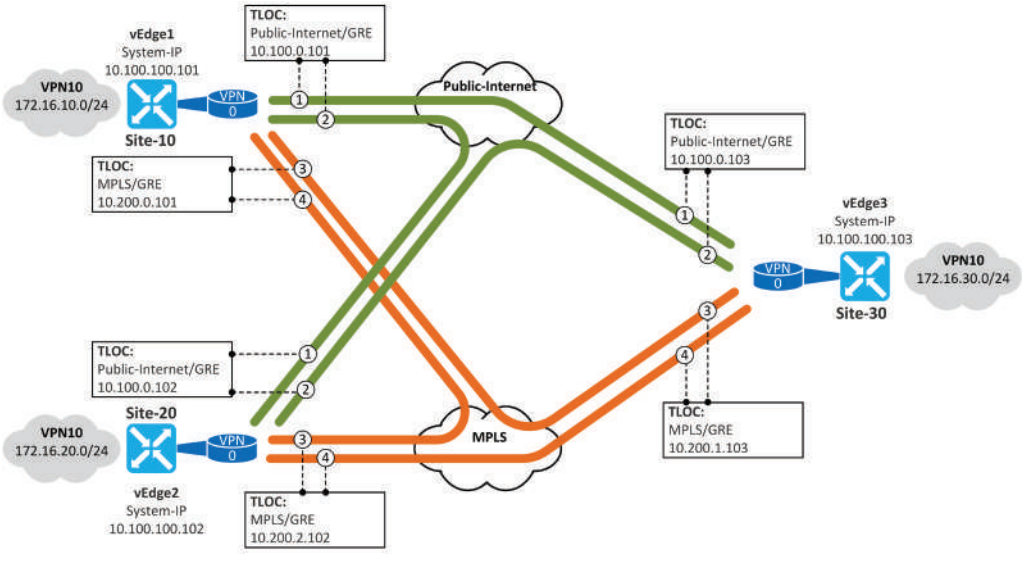


Figure 5-2: *Full Mesh Tunnels Between vEdges.*

Figure 5-3 shows the TLOC routes received by vEdge1. You can verify the real-time information of vEdge1 by selecting vEdge1 from the device list in the *Monitor/Network* window. Next, select the *OMP Received TLOCS* option from the *Device Option* field. As an example, we can see that vEdge1 has received two TLOC routes from vSmart (10.100.100.13) originated by vEdge2 10.10.100.102.

vEdge-1 | 10.100.100.101 Site ID: 10 Device Model: vEdge Cloud

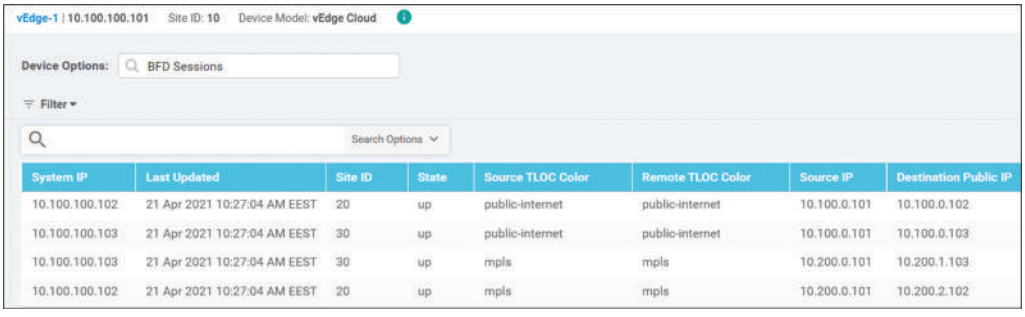
Device Options: OMP Received TLOCS

Search Options

IP	Color	Encap	From Peer	Public IP	Private IP	Site Id	BFD Status
10.100.100.101	mpls	gre	0.0.0.0	10.200.0.101	10.200.0.101	10	up
10.100.100.101	public-internet	gre	0.0.0.0	10.100.0.101	10.100.0.101	10	up
10.100.100.102	mpls	gre	10.100.100.13	10.200.2.102	10.200.2.102	20	up
10.100.100.102	public-internet	gre	10.100.100.13	10.100.0.102	10.100.0.102	20	up
10.100.100.103	mpls	gre	10.100.100.13	10.200.1.103	10.200.1.103	30	up
10.100.100.103	public-internet	gre	10.100.100.13	10.100.0.103	10.100.0.103	30	up

Figure 5-3: *Received TLOC Routes from vEdge1 Perspective.*

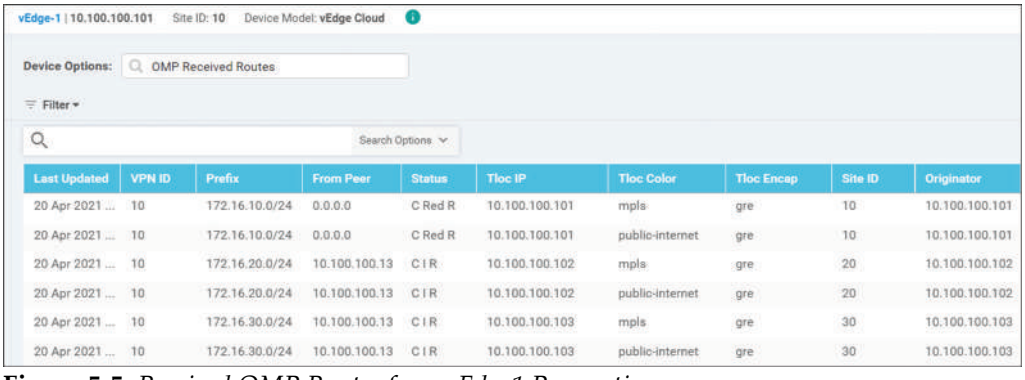
Figure 5-4 verifies that vEdge also has BFD sessions between vEdge2 and vEdge3 over both transport networks.



System IP	Last Updated	Site ID	State	Source TLOC Color	Remote TLOC Color	Source IP	Destination Public IP
10.100.100.102	21 Apr 2021 10:27:04 AM EEST	20	up	public-internet	public-internet	10.100.0.101	10.100.0.102
10.100.100.103	21 Apr 2021 10:27:04 AM EEST	30	up	public-internet	public-internet	10.100.0.101	10.100.0.103
10.100.100.103	21 Apr 2021 10:27:04 AM EEST	30	up	mpls	mpls	10.200.0.101	10.200.1.103
10.100.100.102	21 Apr 2021 10:27:04 AM EEST	20	up	mpls	mpls	10.200.0.101	10.200.2.102

Figure 5-4: *BFD Sessions of vEdge1.*

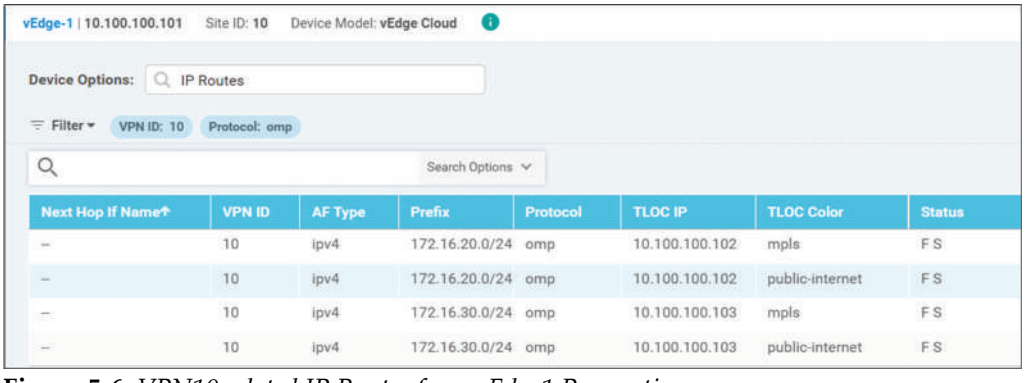
Figure 5-5 shows that vEdge1 has received VPN10 specific OMP routes from both sites over both transport networks.



Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color	Tloc Encap	Site ID	Originator
20 Apr 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	mpls	gre	10	10.100.100.101
20 Apr 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	public-internet	gre	10	10.100.100.101
20 Apr 2021 ...	10	172.16.20.0/24	10.100.100.13	C I R	10.100.100.102	mpls	gre	20	10.100.100.102
20 Apr 2021 ...	10	172.16.20.0/24	10.100.100.13	C I R	10.100.100.102	public-internet	gre	20	10.100.100.102
20 Apr 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.103	mpls	gre	30	10.100.100.103
20 Apr 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.103	public-internet	gre	30	10.100.100.103

Figure 5-5: *Received OMP Routes from vEdge1 Perspective.*

We can also see that vEdge1 has installed VPN 10 specific OMP routes into VPN 10 RIB and FIB.



Next Hop If Name	VPN ID	AF Type	Prefix	Protocol	TLOC IP	TLOC Color	Status
–	10	ipv4	172.16.20.0/24	omp	10.100.100.102	mpls	F S
–	10	ipv4	172.16.20.0/24	omp	10.100.100.102	public-internet	F S
–	10	ipv4	172.16.30.0/24	omp	10.100.100.103	mpls	F S
–	10	ipv4	172.16.30.0/24	omp	10.100.100.103	public-internet	F S

Figure 5-6: *VPN10 related IP Routes from vEdge1 Perspective.*

The example below verifies that we have IP connectivity from host 172.16.10.10 (site 10) to host 172.16.30.30 (site 30) and to host 172.16.20.20 (site 20).

```
PC1-10> ping 172.16.30.30

84 bytes from 172.16.30.30 icmp_seq=1 ttl=62 time=2.646 ms
84 bytes from 172.16.30.30 icmp_seq=2 ttl=62 time=4.439 ms
84 bytes from 172.16.30.30 icmp_seq=3 ttl=62 time=2.830 ms
84 bytes from 172.16.30.30 icmp_seq=4 ttl=62 time=2.663 ms
84 bytes from 172.16.30.30 icmp_seq=5 ttl=62 time=2.298 ms

PC1-10> ping 172.16.20.20

84 bytes from 172.16.20.20 icmp_seq=1 ttl=62 time=7.359 ms
84 bytes from 172.16.20.20 icmp_seq=2 ttl=62 time=5.029 ms
84 bytes from 172.16.20.20 icmp_seq=3 ttl=62 time=2.071 ms
84 bytes from 172.16.20.20 icmp_seq=4 ttl=62 time=2.637 ms
84 bytes from 172.16.20.20 icmp_seq=5 ttl=62 time=3.321 ms
```

Example 5-1: *IP Connectivity Verification*

vSmart - from CLI mode to vManaged mode

The default management mode for Viptela components is CLI mode. We can't use NETCONF to push policies made in vMange to vSmart unless we change the mode to vMange mode. Figure 5-7 shows the failure notification when trying to activate the *Centralized Policy* when vSmart is in CLI mode.

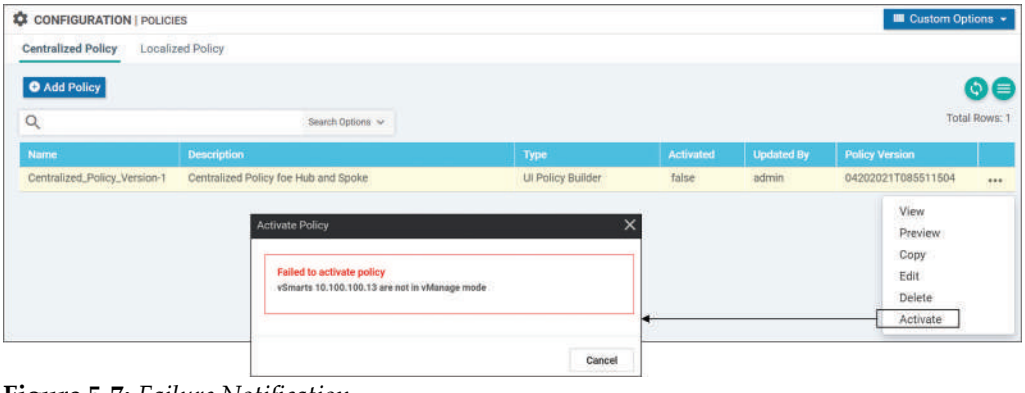


Figure 5-7: *Failure Notification.*

We can verify the management mode either from the vSmart CLI or by navigating to *configuration/devices* and by selecting the *Controllers* tab from the vManage GUI. Example 5-2 shows that the *vManged* state is *false* and the *Configuration Template* is set to *None*.

```
vsmart# show system status | i beg Pers

Personality:              vsmart
Model name:               vsmart
Services:                 None
vManaged:                 false
Commit pending:           false
Configuration template:   None
Policy template:          None
Policy template version:  None
Chassis serial number:    None
```

Example 5-2: *Verifying Manage Mode Used in vSmart by Using the Device CLI.*

Figure 5-8 shows that vSmart mode is CLI and there is no assigned template.

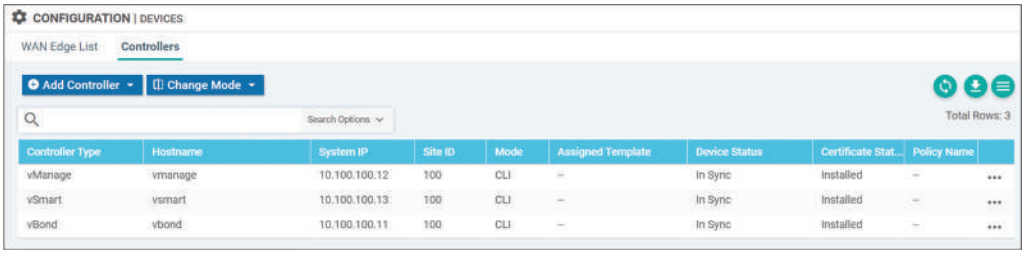


Figure 5-8: *Verifying Manage Mode Used in vSmart via vMange GUI.*

Create CLI Template

In order to change the CLI mode to the vManaged mode navigate to the *configuration/templates* window and select the *Device*. Open the *Create Template* drop-down menu and choose the *CLI Template* option.

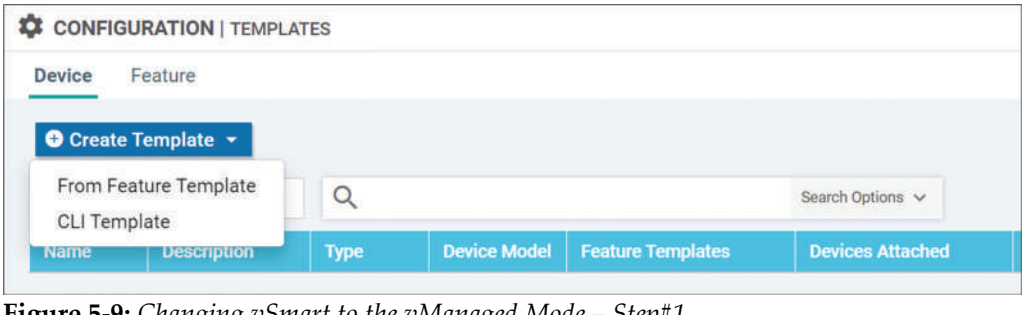


Figure 5-9: *Changing vSmart to the vManaged Mode – Step#1.*

Then select the *vSmart* from the *Device Model* drop-down menu and fill the *Template Name* and *Description* fields. You can either copy the vSmart configuration from the vSmart CLI or by selecting vSmart from the *Load Running config from the reachable device* drop-down menu. Click the *Add* button when done.

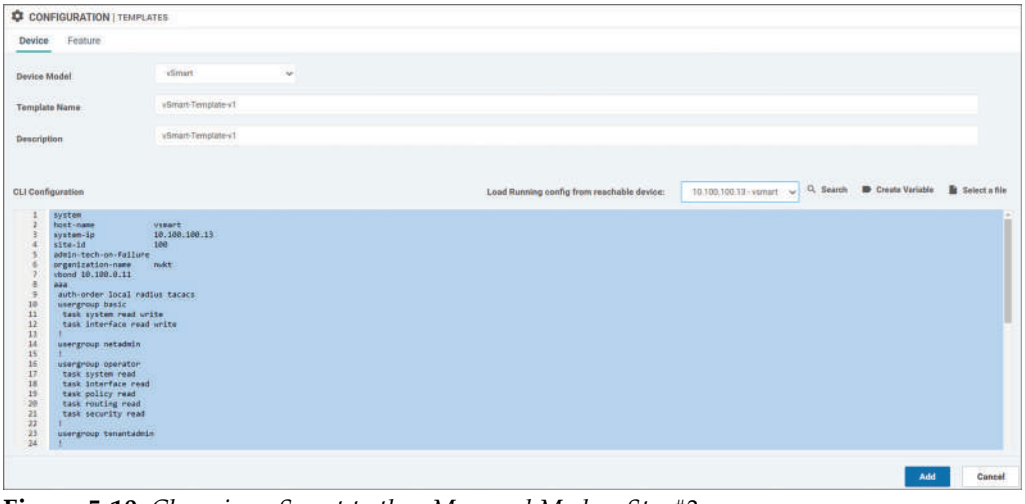


Figure 5-10: *Changing vSmart to the vManaged Mode – Step#2.*

Attach CLI Template to vSmart

Attach the CLI template to vSmart by selecting the *Attach Device* optin from the *Options* menu […].

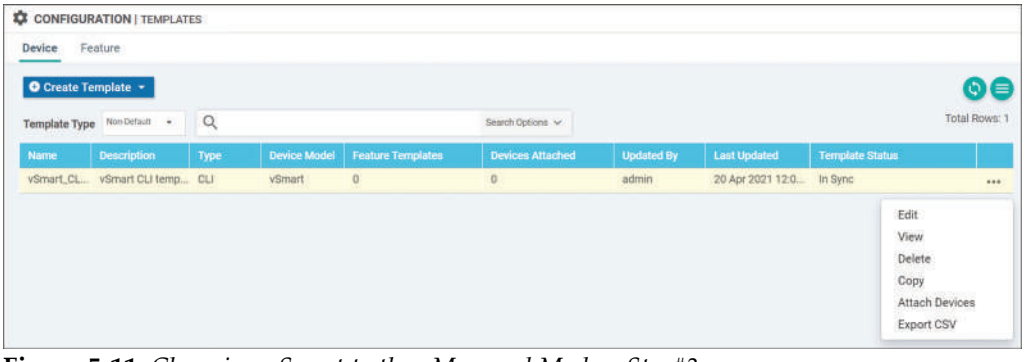


Figure 5-11: *Changing vSmart to the vManaged Mode – Step#3.*

Select *vsmart* from the *Available Devices* column and press the *Arrow* button to move it to *Selected Devices* columns. Then click the *Attach* button.

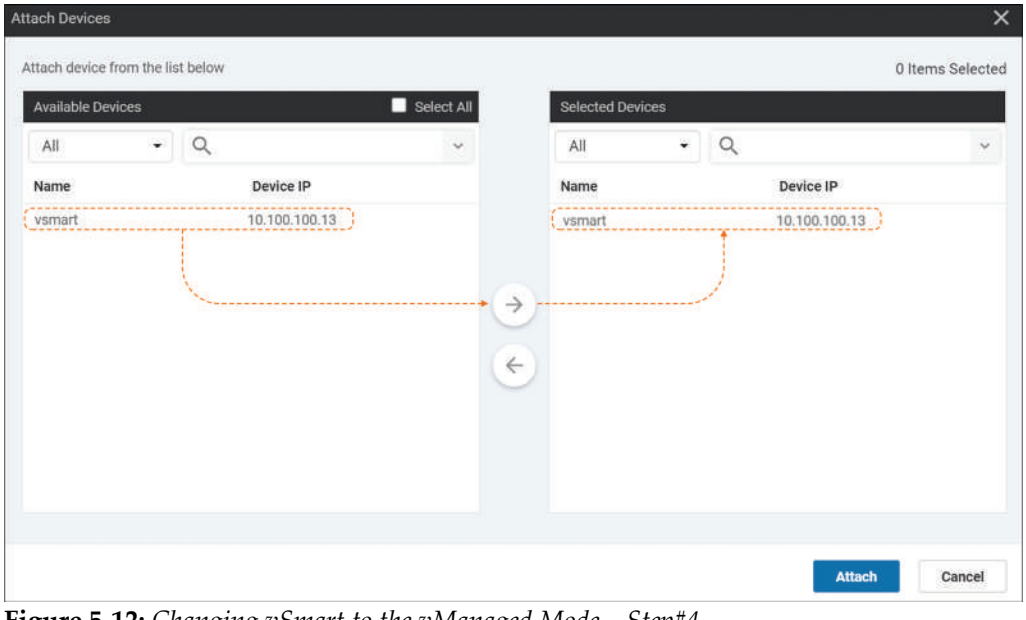


Figure 5-12: *Changing vSmart to the vManaged Mode – Step#4.*

As the last step, select the *Configure Devices* button.

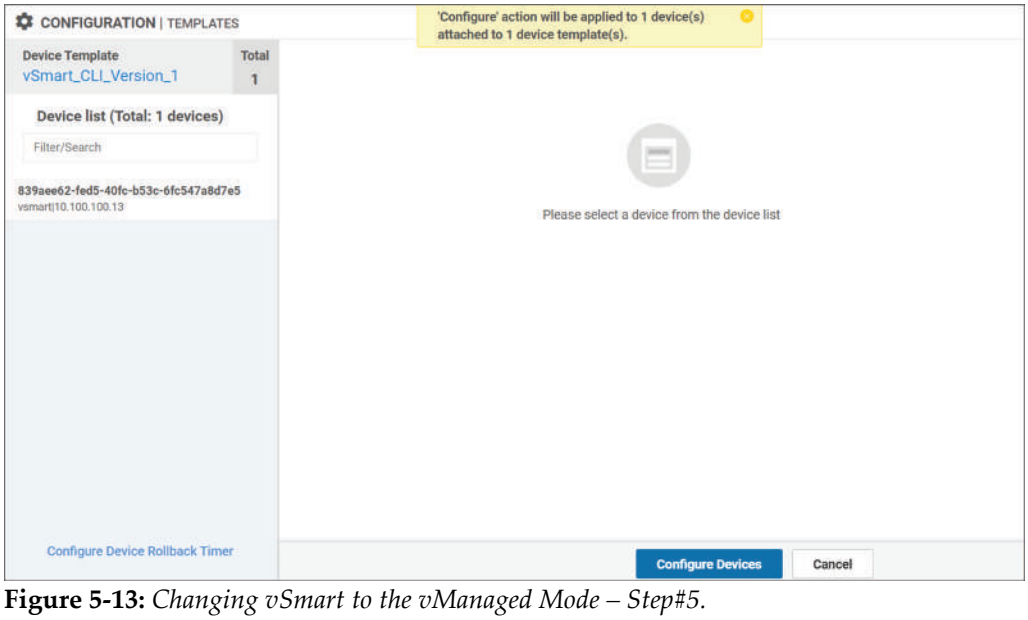


Figure 5-13: *Changing vSmart to the vManaged Mode – Step#5.*

Figure 5-14 shows the successful template installation.

Figure 5-14: *Changing vSmart to the vManaged Mode – Verification: Success.*

Note! The failure shown in figure 5-15 may occur if you copied the configuration and attached it to the CLI template configuration.

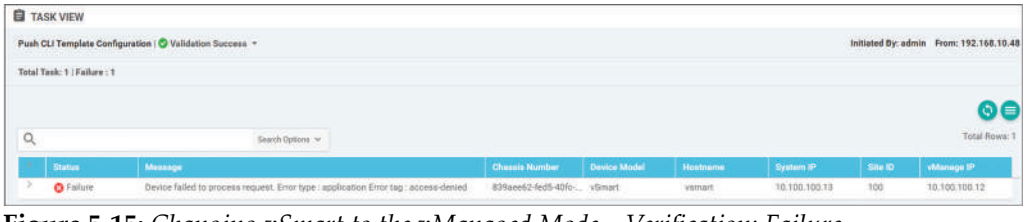


Figure 5-15: *Changing vSmart to the vManaged Mode – Verification: Failure.*

We can verify successful changes from the CLI (example 5-3) or the vManage GUI (figure 5-16).

```
vsmart# show system status | i beg Pers
Personality:              vsmart
Model name:               vsmart
Services:                 None
vManaged:                 true
Commit pending:           true
Configuration template:   vSmart-Template-v1
Policy template:          None
Policy template version:  None
Chassis serial number:    None
```

Example 5-3: *Verifying Manage Mode Used in vSmart by Using the vSmart CLI.*

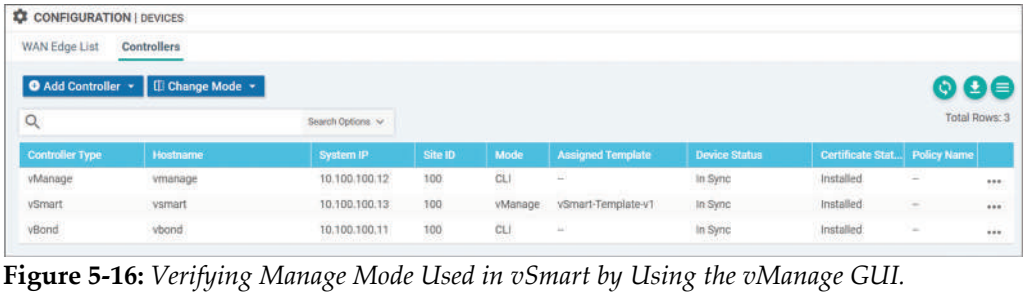


Figure 5-16: *Verifying Manage Mode Used in vSmart by Using the vManage GUI.*

Policy Configuration

Figure 5-17 illustrates the SD-WAN Policy model. It has two main policy categories, a) *Localized Policies*, and b) *Centralized Policies*. The focus of this chapter is Centralized Policies. Centralized Policies has two sub-categories, a) *Data Policies* and *Control/Topology policies*. Our intent is to restrict traffic between site 10 and site 20 so we are going to build a *Centralized Policy* where we are using *Control Policy* to build a Hub and Spoke topology where sites 10 and 20 are restricted from each other by controlling TLOC/OMP route advertisements.

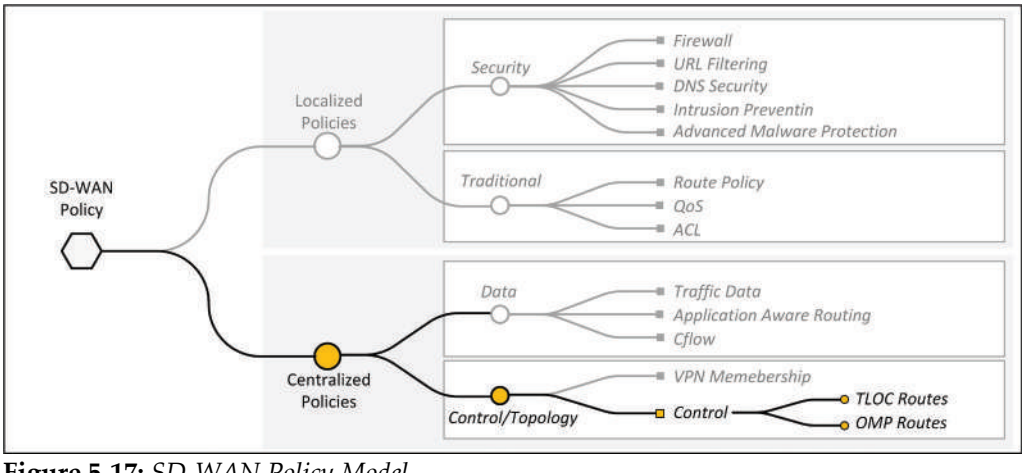


Figure 5-17: *SD-WAN Policy Model.*

Figure 5-18 illustrates the TLOC/OMP route advertisement directions. The vSmart can advertise TLOC/OMP routes received from Branch Sites (site 10 and site 20) to DataCenter (site 30). Besides, vSmart can advertise TLOC/OMP routes from DataCenter to Branch Sites. However, vSmart doesn't advertise TLOC/OMP routes received from one Branch Site to another. This way we create Hub and Spoke topology where Branch Sites are restricted from each other and no data can be sent between them. The whole Control Policy is based on routing control and we are not using any data traffic filtering.

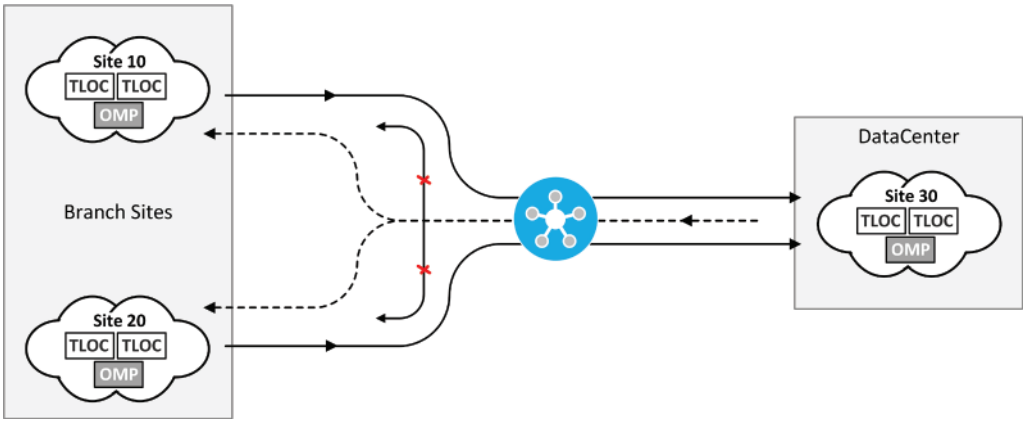


Figure 5-18: *TLOC/OMP Routes Advertisement Direction.*

Example 5-4 shows the configuration we are going to build by using vManage GUI. It also shows the structure and configuration flow of the Centralized Policy. Note that the Centralized Policy doesn't have any name, we can only use one Central Policy. *In the first step,* we create a set of *lists* where we define a *Group of Interest*. A list can include application, color, data prefix, policer, prefix, site, SLA class, TLOC, or VPN. We are going to use two *site* lists and a *prefix* list. One site list for DataCenter (site 30) and the other one for the Remote-Sites (sites 10 and 20). The prefix list matches all networks. In *the second step,* we create a *Control Policy* that has a set of sequences where we match TLOCs and OMP routes from site 30 listed under site list DataCenter and where we set the action *Accept*. At the end of the Control Policy, we have implicit reject. As *the last step*, we apply the Control Policy to sites listed in the site-list Remote-Sites as an outbound policy. By doing this, we allow vSmart to send TLOC and OMP routes originated from the DataCenter to Remote-Sites but not TLOC and OMP routes from site 10 to site 20 and the other way around. There is no policy applied towards site 30, so vSmart forwards all TLOC and OMP routes to site 30.

```
Policy                                          # Centrlazied Policy
 Lists                                          # Group of Interest
  site-list DataCenter
   site-id 30
  !
  site-list Remote-Sites
   site-id 10
   site-id 20
  !
  prefix-list _AnyIpv4PrefixList
   ip-prefix 0.0.0.0/0 le 32
  !
 !
 control-policy HUB_AND_SPOKE_POLICY        # Control Policy
  sequence 1
   match tloc
    site-list DataCenter
   !
   action accept
   !
  !
  sequence 11
   match route
    prefix-list _AnyIpv4PrefixList
    site-list   DataCenter
   !
   action accept
   !
  !
  default-action reject
 !
!
apply-policy                          # Apply Policy
 site-list Remote-Sites
  control-policy HUB_AND_SPOKE_POLICY out
```

Example 5-4: *Verifying Manage Mode Used in vSmart by Using the vSmart CLI.*

Step-1: Create Site-List

Open the vMange GUI and navigate to *configurations/policies* and select the *Centralized Policy* tab. Click the *Add Policy* button.

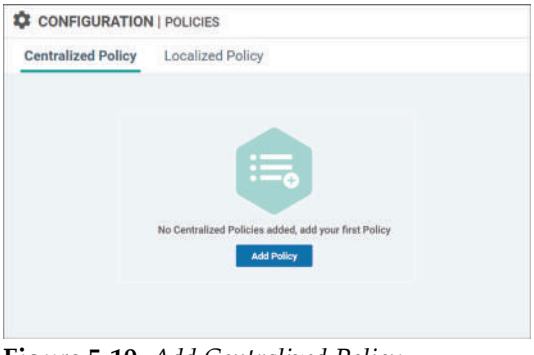


Figure 5-19: *Add Centralized Policy.*

Select *Site* from the left list pane. Click the *New Site List* button. I have named the first site list as Remote-Sites and added sites 10 and 20 to the list.

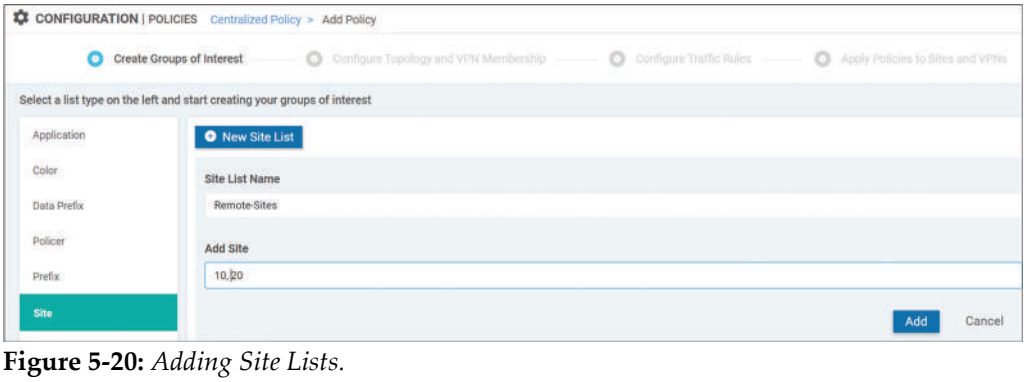


Figure 5-20: *Adding Site Lists.*

I have also created the DataCenter site list in the same way. Figure 5-21 shows our two site lists. To move forward to *Configure Topology and VPN Membership section*, click the *Next* button at the bottom (not shown in the figure).

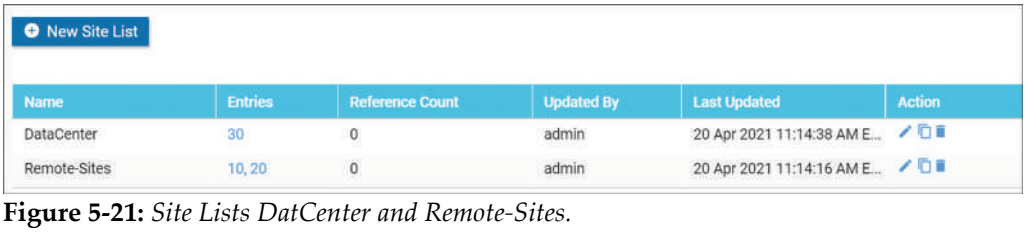


Figure 5-21: *Site Lists DatCenter and Remote-Sites.*

Example 5-5 shows the configuration related to site lists. The configuration is sent to vSmart as a part of the complete policy configuration when we activate the policy as the last step. Note that there is also an auto-generated prefix-list with prefix 0.0.0.0/0 le 32 which covers all subnets including 172.16.30.0/24.

```
policy
 lists
  site-list DataCenter
   site-id 30
  !
  site-list Remote-Sites
   site-id 10
   site-id 20
  !
  prefix-list _AnyIpv4PrefixList
   ip-prefix 0.0.0.0/0 le 32
```

Example 5-5: *Site List Configuration.*

Step-2: Create Control Policy

Open the *Add Topology* drop-down menu and select *Custom Control (Route & TLOC)* option.

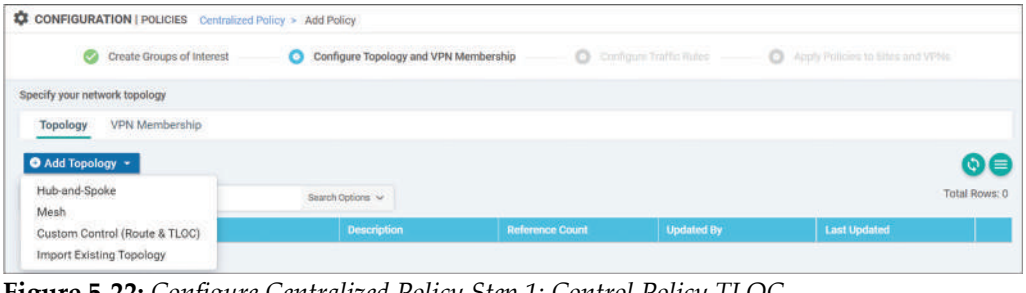


Figure 5-22: *Configure Centralized Policy Step 1: Control Policy TLOC.*

Give the name and description and click the *Sequence Type* button. This will create the first sequence under Control Policy.

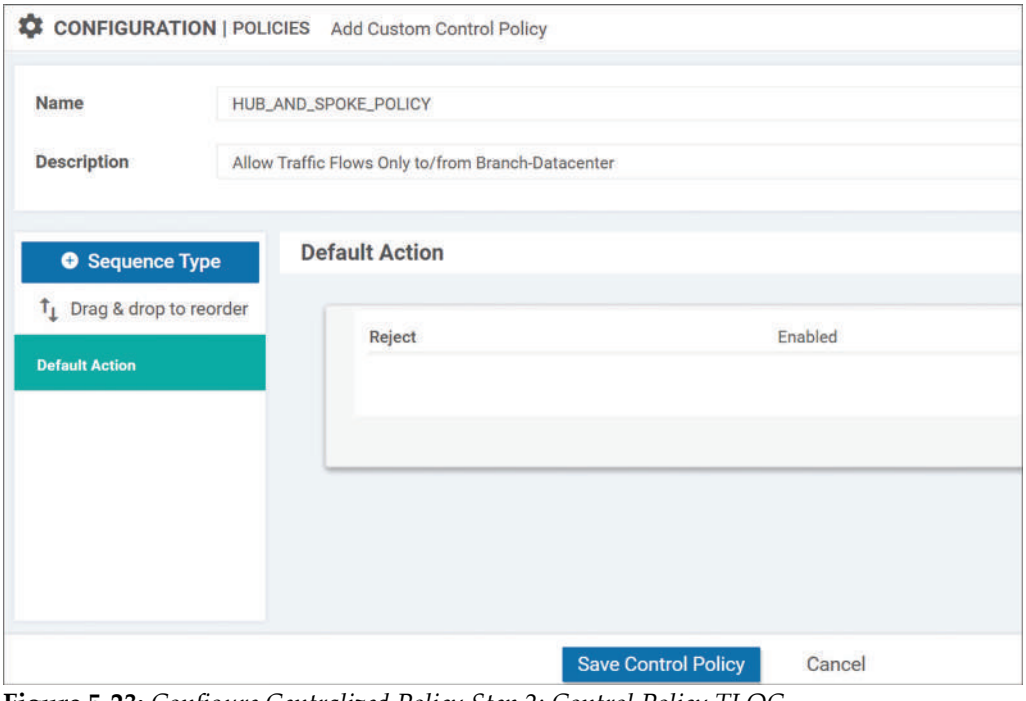


Figure 5-23: *Configure Centralized Policy Step 2: Control Policy TLOC.*

Select the *TLOC* option from the *Add Control Policy* pop-up window. By doing this the first sequence matches to TLOCs.

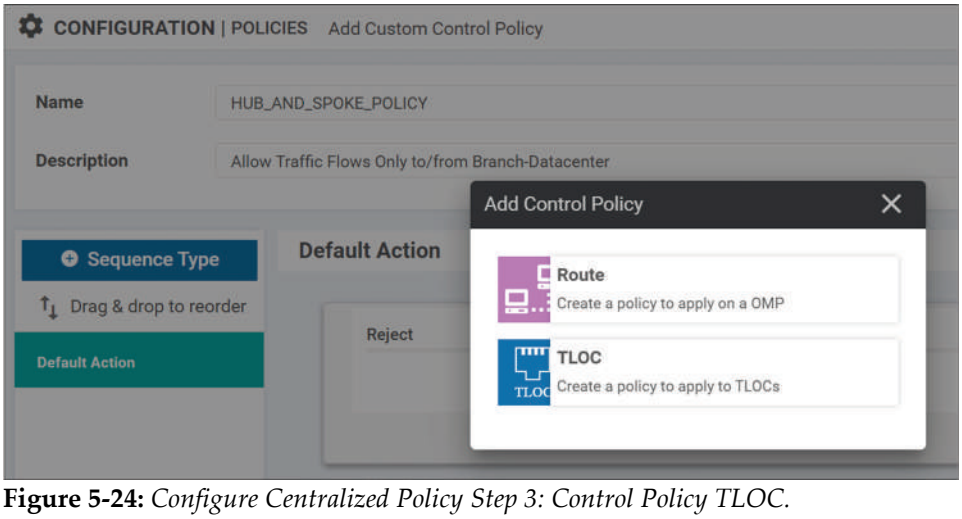


Figure 5-24: *Configure Centralized Policy Step 3: Control Policy TLOC.*

Click the *Action* selector and choose the *Accept* option.

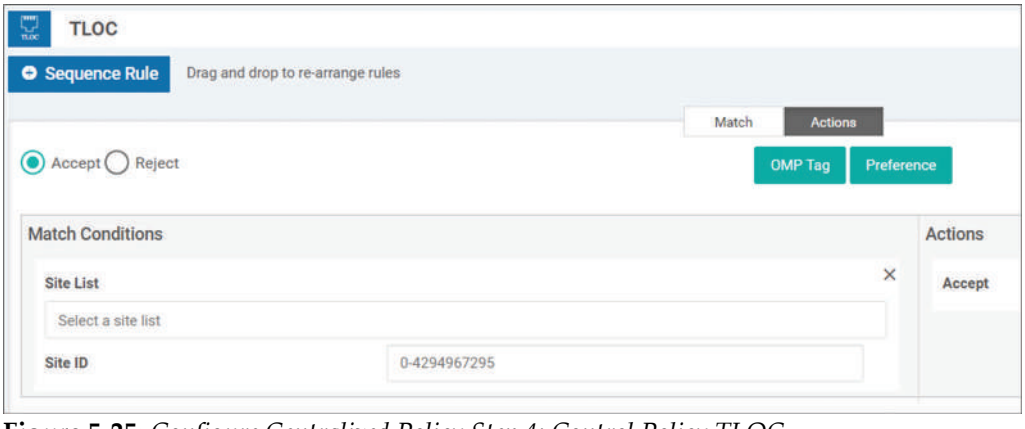


Figure 5-25: *Configure Centralized Policy Step 4: Control Policy TLOC.*

Click the *Match* selector and open the *Site List* drop-down menu from the *Match Conditions* window. Select the *DataCenter*.

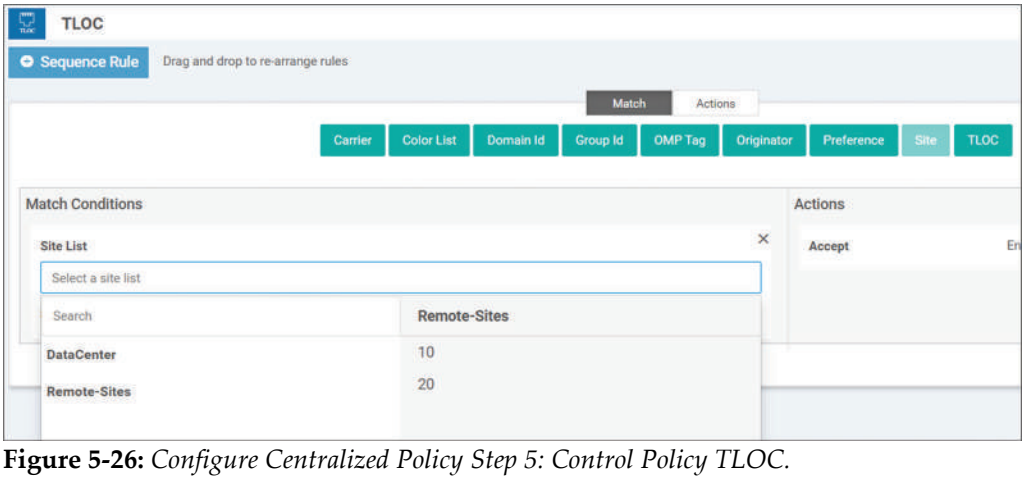


Figure 5-26: *Configure Centralized Policy Step 5: Control Policy TLOC.*

Click the *Save Match and Actions* button.

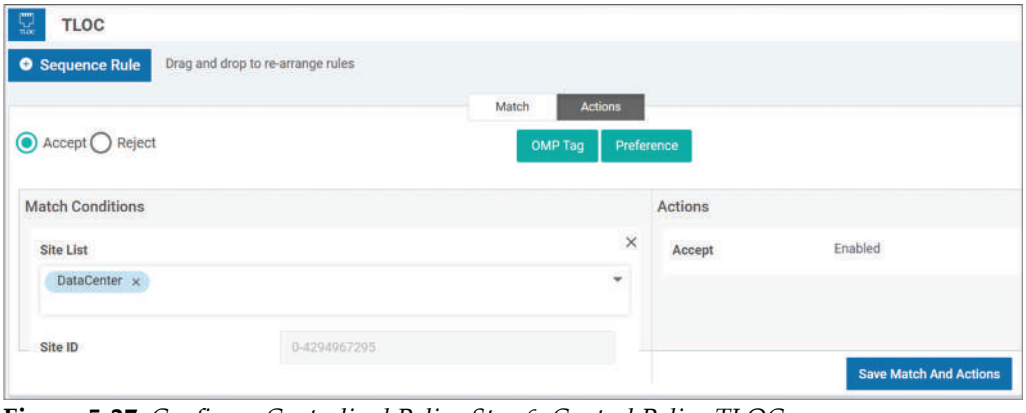


Figure 5-27: *Configure Centralized Policy Step 6: Control Policy TLOC.*

Figure 5-28 shows the first sequence TLOC. We created a sequence inside the *Control Policy* HUB_AND_SPOKE_POLICY where we match to TLOCs of DataCenter and set the action to permit.

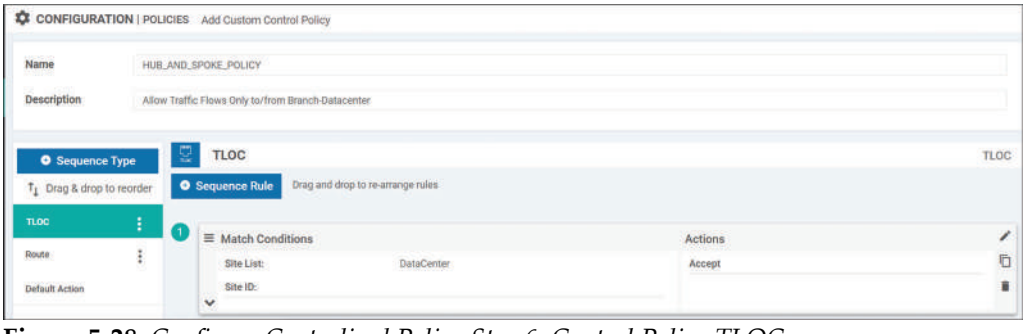


Figure 5-28: *Configure Centralized Policy Step 6: Control Policy TLOC.*

Create a new sequence under the same Control Policy. This sequence will match to OMP Routes. In addition, this sequence also matched the default IP prefix because we are not explicitly defined any IP prefix.

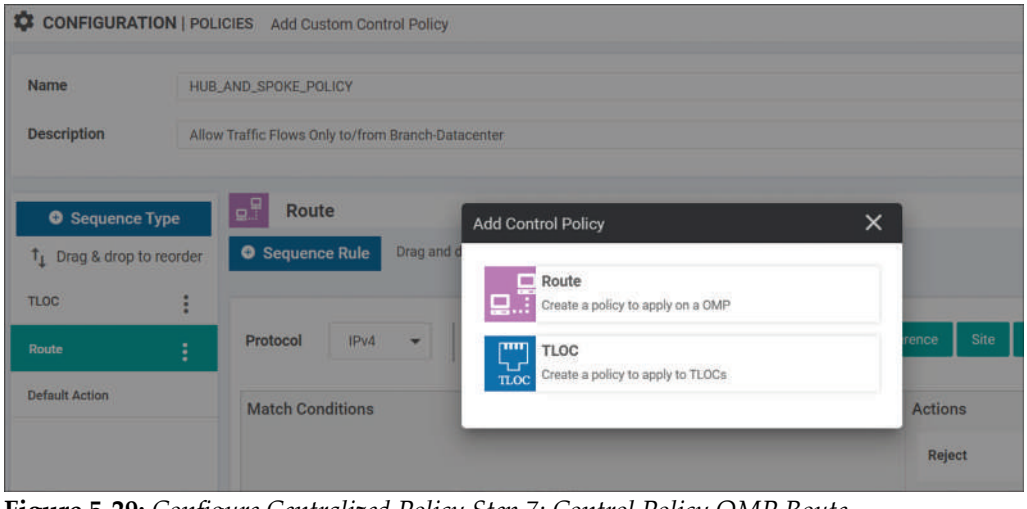


Figure 5-29: *Configure Centralized Policy Step 7: Control Policy OMP Route.*

The process of creating a route sequence is the same as what we did with the sequence related to TLOC. Set the *Match* and *Action* conditions.

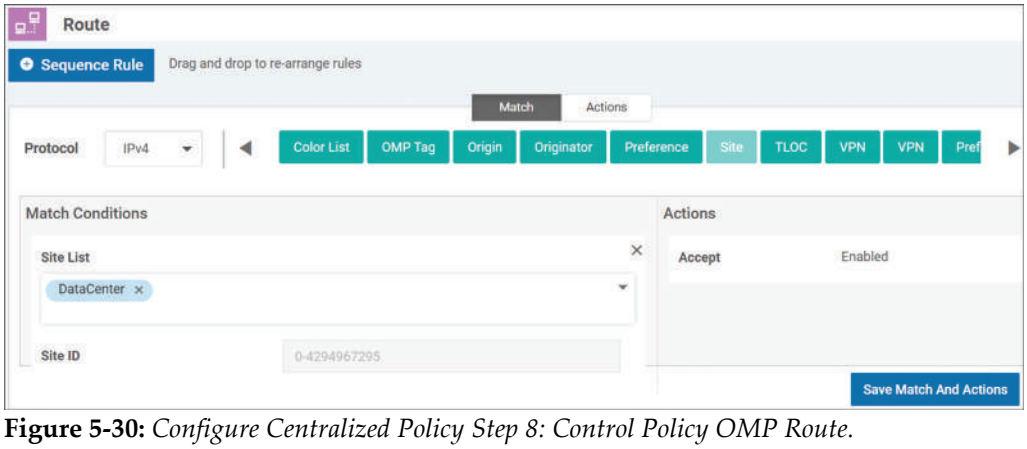


Figure 5-30: *Configure Centralized Policy Step 8: Control Policy OMP Route.*

Figure 5-31 shows that the sequence related to TLOC is listed before the sequence related to OMP Route.

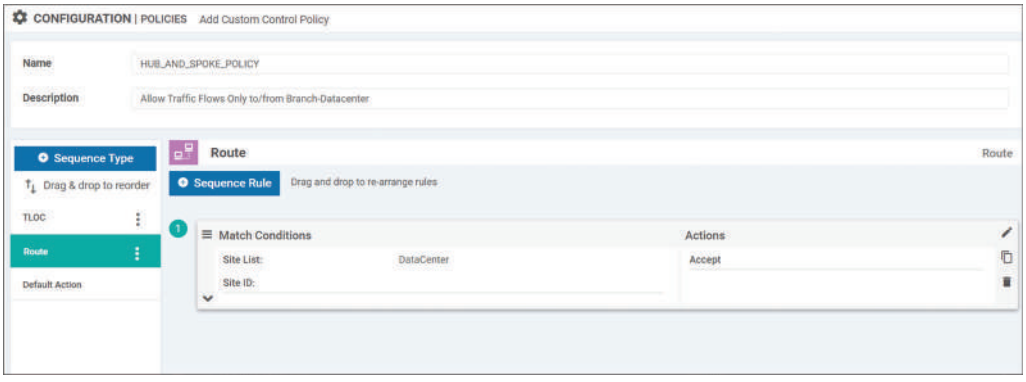


Figure 5-31: *Configure Centralized Policy Step 9: Control Policy OMP Route.*

Example 5-5 shows the configuration of the Control Policy. The auto-generated prefix-list with the prefix 0.0.0.0/0 is added under the *match route* (OMP route) section because we did not specify the exact subnet.

```
control-policy HUB_AND_SPOKE_POLICY
  sequence 1
   match tloc
    site-list DataCenter
   !
   action accept
   !
  !
  sequence 11
   match route
    prefix-list _AnyIpv4PrefixList
    site-list   DataCenter
   !
   action accept
   !
  !
  default-action reject
```

Example 5-6: *Configuration Related to TLOC/OMP Control-Policy Sequences.*

By clicking the *Next* button, you will be forwarded to *Configure Traffic Rules* window.

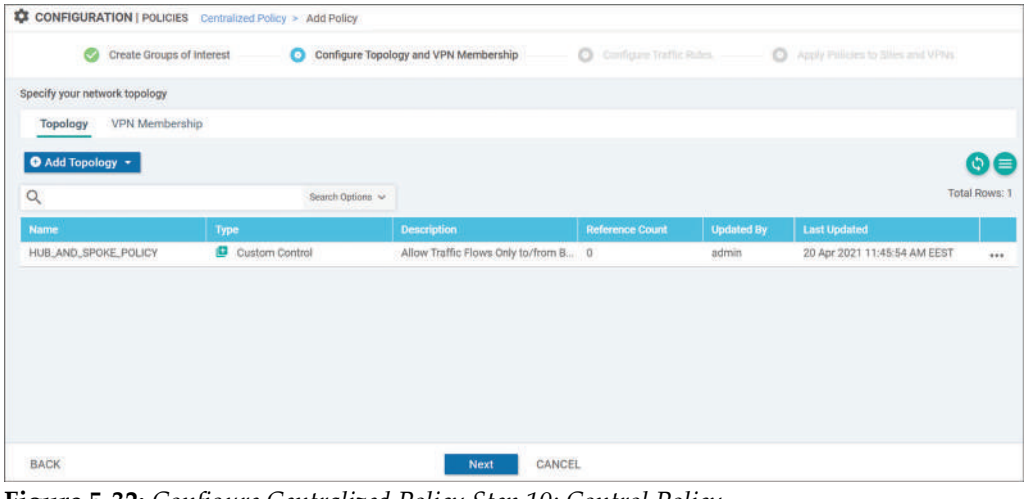


Figure 5-32: *Configure Centralized Policy Step 10: Control Policy.*

We are not using any traffic rules so click the *Next* button to move to *Apply Policy to Sites and VPN* window.

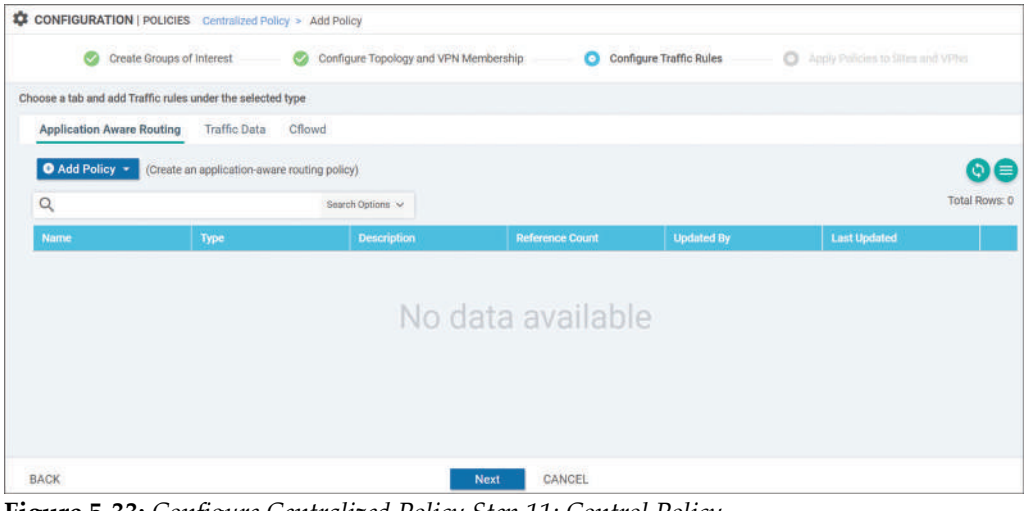


Figure 5-33: *Configure Centralized Policy Step 11: Control Policy.*

Step-3: Apply Control Policy

Give the name and description to *Centralized Policy*. Note that you can activate only one *Centralized Policy* at a time and that is why I am using the _Version-1 definition at the end of the name. Select site list Remote-Sites from the *Outbound Site List* field. By doing this we add a *Control Policy* Hub_and_Spoke_Policy toward sites 10 and 20 defined in the Remote-Sites site-list. Remember that the main policy is a *Centralized Policy* with the sub-policy *Control Policy* which is used for setting up the topology. Click the *Add* button.

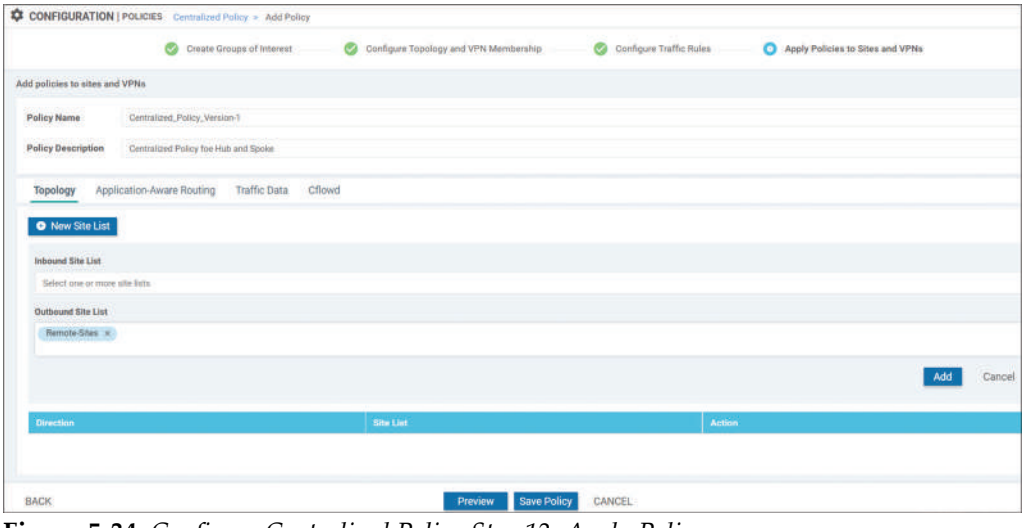


Figure 5-34: *Configure Centralized Policy Step 12: Apply Policy.*

Example 5-7 shows the CLI configuration generated in *Apply Policies to Sites and VPNs* phase.

```
apply-policy
 site-list Remote-Sites
  control-policy HUB_AND_SPOKE_POLICY out
```

Example 5-7: *Configure Centralized Policy Step 12 from the CLI: Apply Policy.*

Figure 5-35 shows the ***Control Policy*** *Hub_and_Spoke_Policy* is attached to the ***Centralized Policy*** *Centralized-Policy_Version-1* as an outbound policy towards sites listed in the site list Remote-Sites. You can check the configuration that has been created by clicking the *Preview* button. Click the *Save Policy* button.

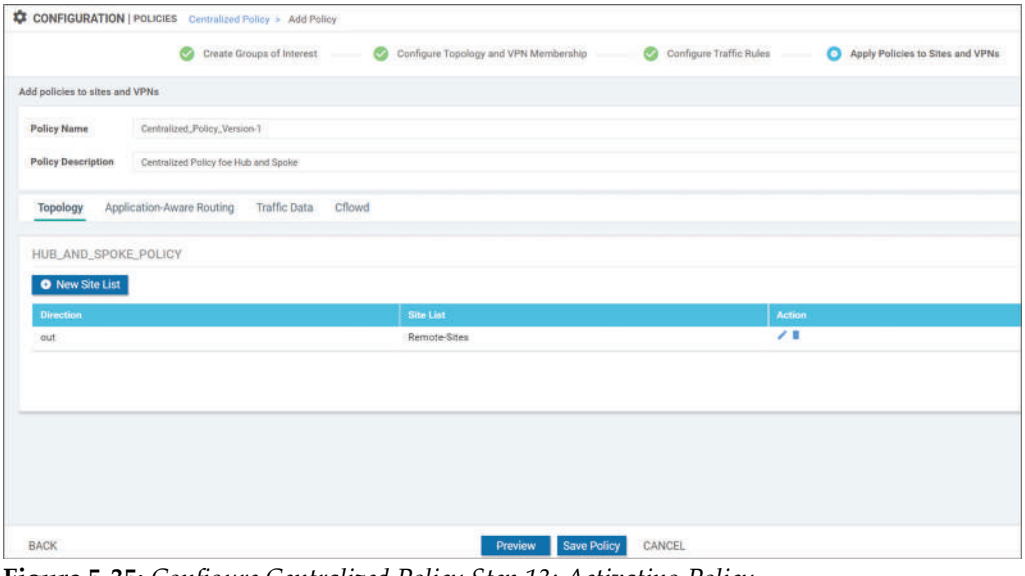


Figure 5-35: *Configure Centralized Policy Step 13: Activating Policy.*

Step-4: Activate Centralized Policy

As the final step, we activate the *Centralized Policy* by selecting *Activate* from the *Options* drop-down menu […].

Figure 5-36: *Configure Centralized Policy Step 14: Activating Policy.*

You will be asked to confirm the activation. Click the *Activate* button.

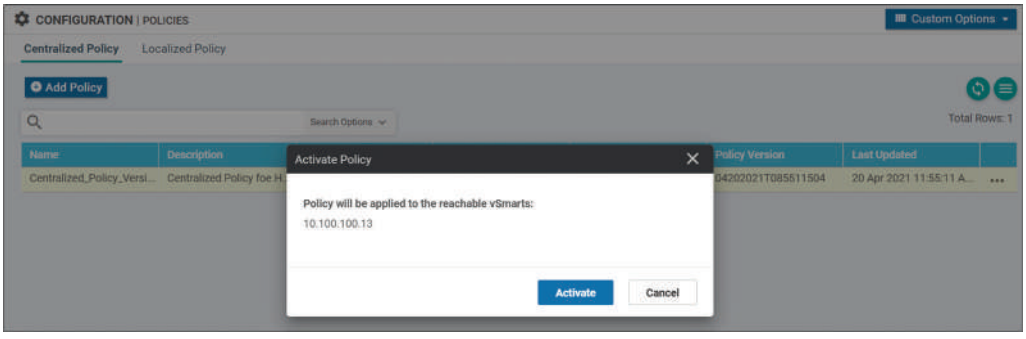


Figure 5-37: *Configure Centralized Policy Step 14: Activating Policy.*

Figure 5-38 shows that the Centralized Policy is now applied successfully to vSmart.

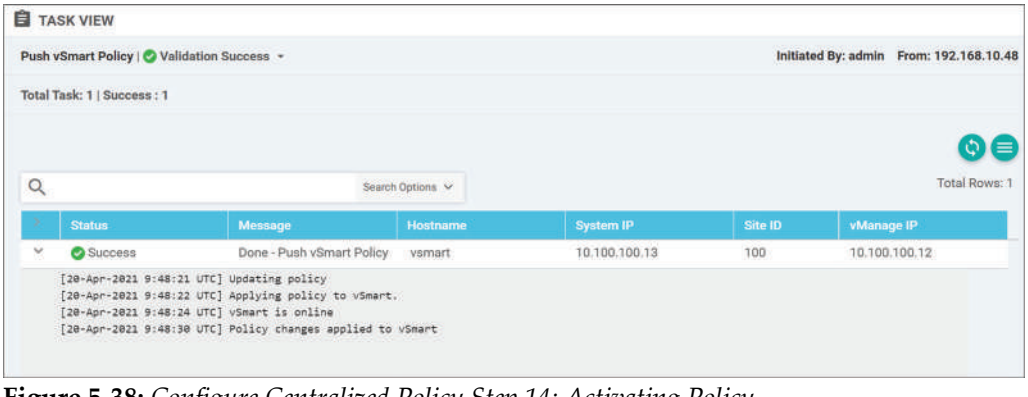


Figure 5-38: *Configure Centralized Policy Step 14: Activating Policy.*

Policy Verification

We can verify that our Centralized Policy is in effect from the vMange GUI by verifying which TLOC and OMP routes vEdge1 has received from the vSmart. Figure 5-39 shows that in addition to local TLOCs, vEdge1 has received only site 30 TLOCs (DataCenter).

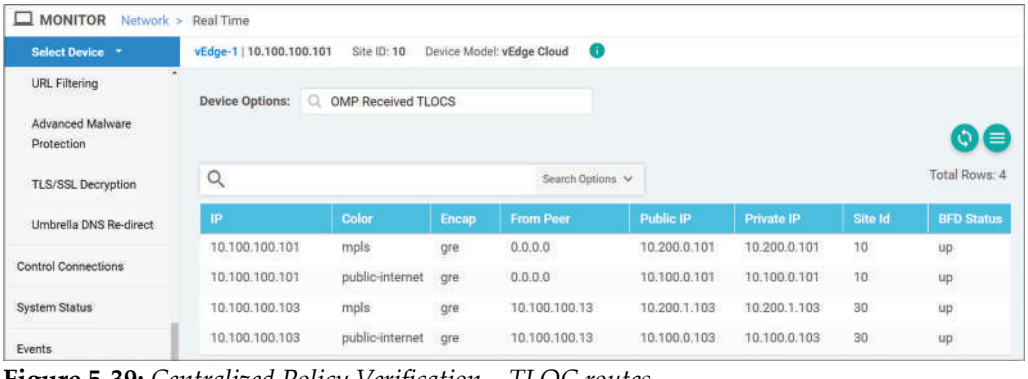


Figure 5-39: *Centralized Policy Verification – TLOC routes.*

We can also see that vEdge1 has only received site 30 related OMP routes from the vSmart.

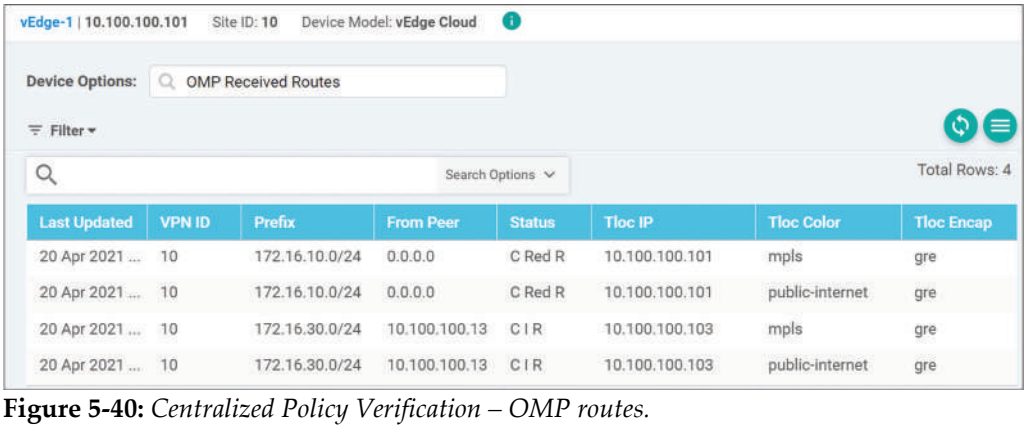


Figure 5-40: *Centralized Policy Verification – OMP routes.*

Figure 5-41 verifies that vEdge1 has installed OMP route 172.16.30.0/24 to VPN10 Routing Information Base (RIB).

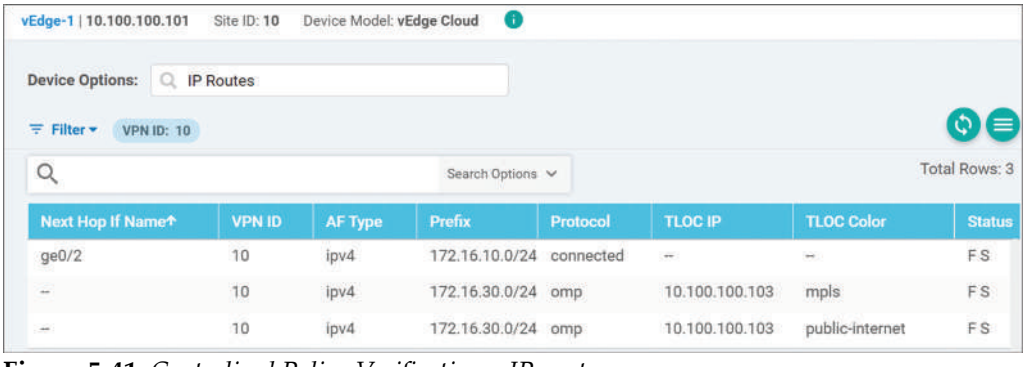


Figure 5-41: *Centralized Policy Verification – IP routes.*

We can also see that there is only two BFD session from vEdge1 to vEdge-3, one over Public-Internet transport network and the other one over MPLS transport network.

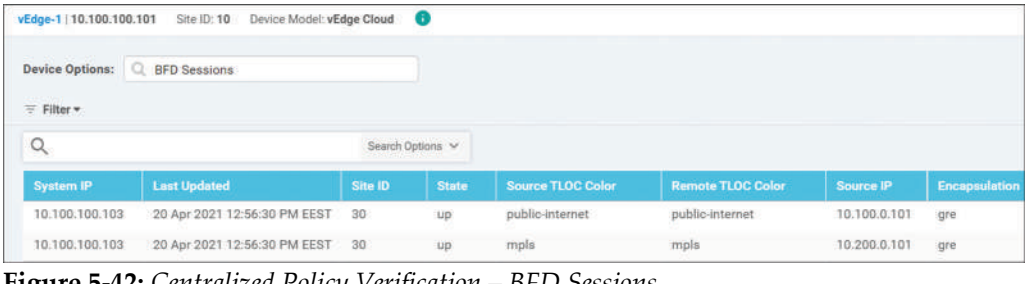


Figure 5-42: *Centralized Policy Verification – BFD Sessions.*

Figure 5-43 illustrates the topology we created with Centralized Topology. There are no tunnels between vEdge1 and vEdge2.

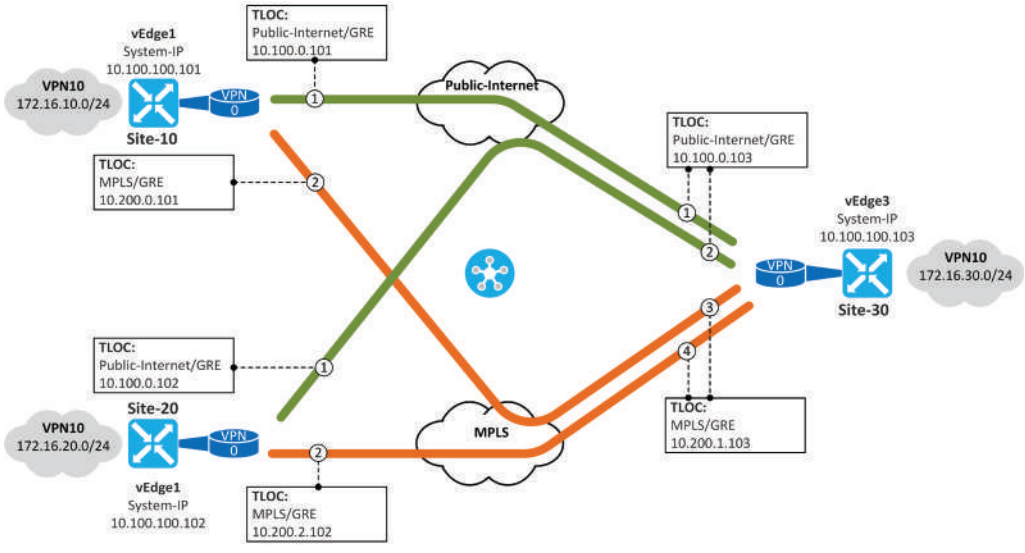


Figure 5-43: *Final Topology – Hub and Spoke.*

Example 5-8 shows that we can't anymore ping the end-point 172.16.20.20 but we still can ping end-point 172.16.30.30.

```
PC1-10> ping 172.16.20.20

*172.16.10.1 icmp_seq=1 ttl=64 time=0.140 ms (ICMP type:3, code:0, Destination network
unreachable)
*172.16.10.1 icmp_seq=2 ttl=64 time=0.126 ms (ICMP type:3, code:0, Destination network
unreachable)
*172.16.10.1 icmp_seq=3 ttl=64 time=0.131 ms (ICMP type:3, code:0, Destination network
unreachable)
*172.16.10.1 icmp_seq=4 ttl=64 time=0.128 ms (ICMP type:3, code:0, Destination network
unreachable)
*172.16.10.1 icmp_seq=5 ttl=64 time=0.210 ms (ICMP type:3, code:0, Destination network
unreachable)

PC1-10> ping 172.16.30.30

84 bytes from 172.16.30.30 icmp_seq=1 ttl=62 time=2.646 ms
84 bytes from 172.16.30.30 icmp_seq=2 ttl=62 time=4.439 ms
84 bytes from 172.16.30.30 icmp_seq=3 ttl=62 time=2.830 ms
84 bytes from 172.16.30.30 icmp_seq=4 ttl=62 time=2.663 ms
84 bytes from 172.16.30.30 icmp_seq=5 ttl=62 time=2.298 ms
```

Example 5-8: *Data Plane Testing.*

Example 5-9 shows the complete vSmart configuration

```
vsmart# sh run
system
 host-name             vsmart
 system-ip             10.100.100.13
 site-id               100
 admin-tech-on-failure
 organization-name     nwkt
 vbond 10.100.0.11
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  usergroup tenantadmin
  !
  user admin
   password
$6$jKzSSqC2GCJveJV4$VxMCv59Qv2J.lDd2luqXXJ9dUuv3izVKXPEbE3b43AAry3n6
ptI7DqunO0y0TzxaUVRGAUZ7E/ySEiWdyt8/60
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
```

```
omp
 no shutdown
 graceful-restart
!
vpn 0
 interface eth0
  ip address 10.100.0.13/24
  ipv6 dhcp-client
  tunnel-interface
   allow-service all
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   allow-service netconf
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.0.1
!
vpn 512
 interface eth1
  ip dhcp-client
  no shutdown
 !
!
policy
 lists
  site-list DataCenter
   site-id 30
  !
  site-list Remote-Sites
   site-id 10
   site-id 20
  !
  prefix-list _AnyIpv4PrefixList
   ip-prefix 0.0.0.0/0 le 32
  !
 !
 control-policy HUB_AND_SPOKE_POLICY
  sequence 1
   match tloc
    site-list DataCenter
   !
   action accept
   !
  !
  sequence 11
   match route
    prefix-list _AnyIpv4PrefixList
    site-list   DataCenter
   !
```

```
    action accept
     !
   !
   default-action reject
  !
!
apply-policy
 site-list Remote-Sites
  control-policy HUB_AND_SPOKE_POLICY out
 !
!
vsmart#
```

Example 5-9: *Complete vSmart Configuration.*

Spoke-to-Spoke traffic

If we want to allow traffic flows between spoke sites without direct tunneling we can do the OMP route summarization in the Hub site. In our example, I have added a null route under VPN10 configuration for network 172.16.0.0/10. This static route is automatically redistributed into the OMP route advertisement process. Our Centralized Policy allows OMP Routes from site 30 to be advertised to sites 10 and 20.

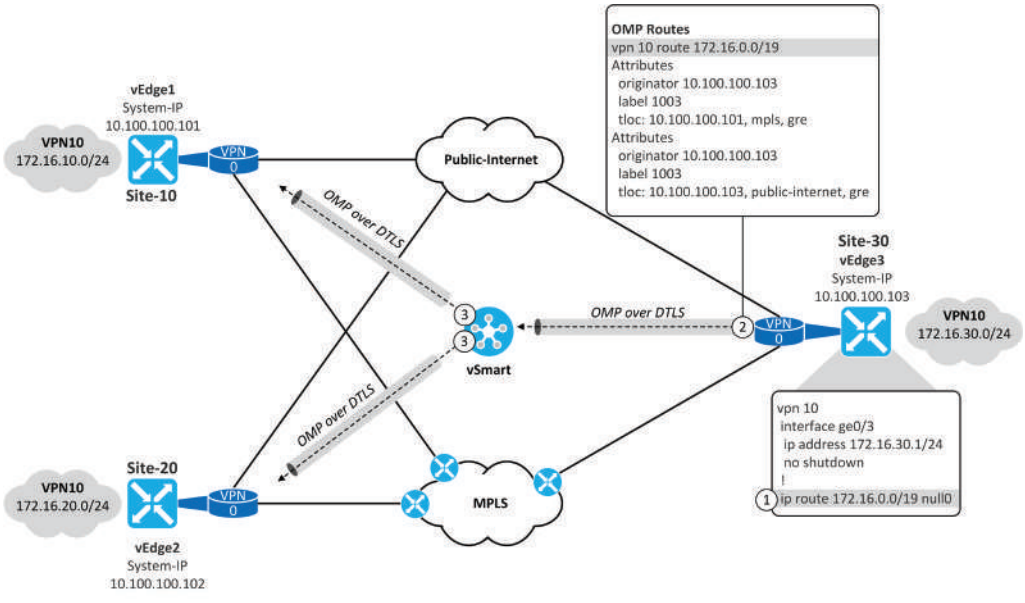


Figure 5-44: *Spoke-to-Spoke: OMP Summary Route Advertisement.*

Figure 5-45 shows that vEdge3 advertises the OMP Summary route 172.16.0.0/19 to vSmart.

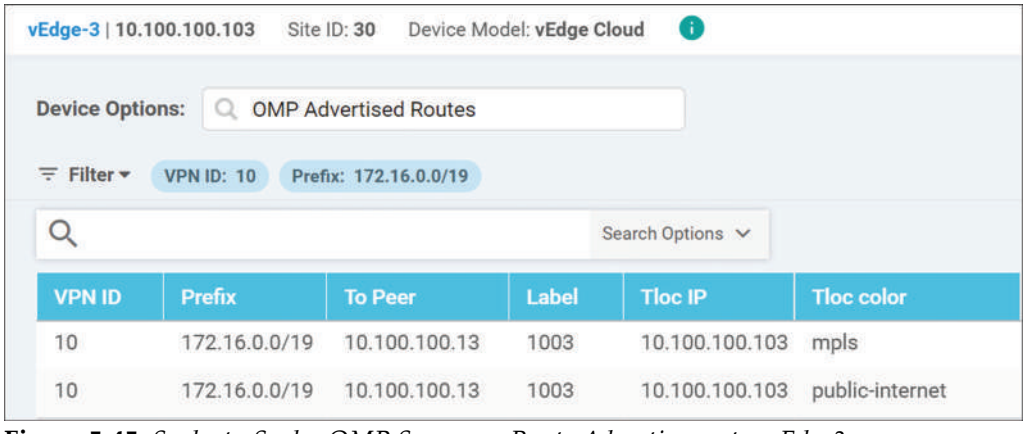


Figure 5-45: *Spoke-to-Spoke: OMP Summary Route Advertisement – vEdge3.*

Figure 5-46 shows that vSmart in turn advertises the OMP Summary route 172.16.0.0/19 to both vEdge1 and vEdge2.

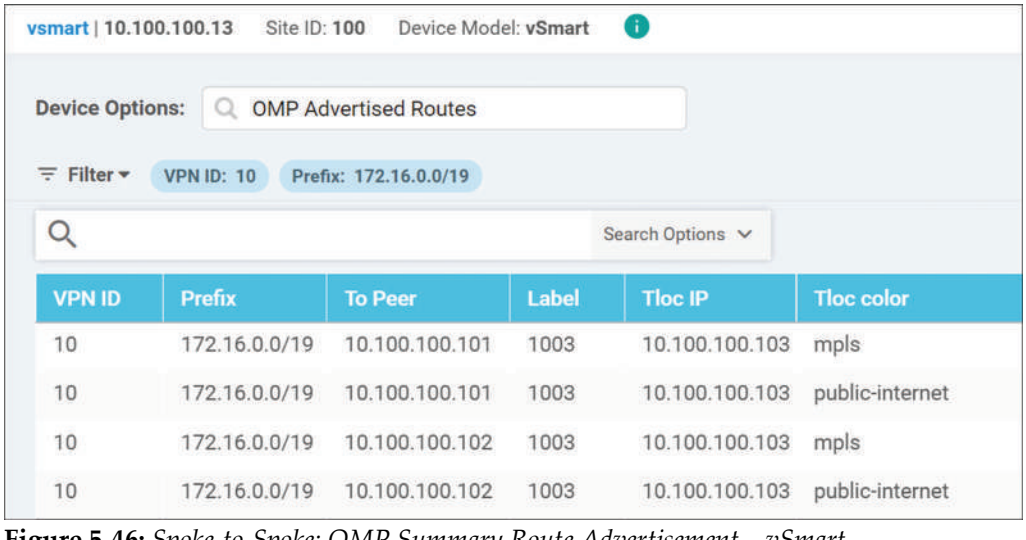


Figure 5-46: *Spoke-to-Spoke: OMP Summary Route Advertisement – vSmart.*

Figures 5-47 and 5-48 verifies that both vEdge1 and vEdge2 have received the OMP Summary route 172.16.0.0/19 from vSmart.

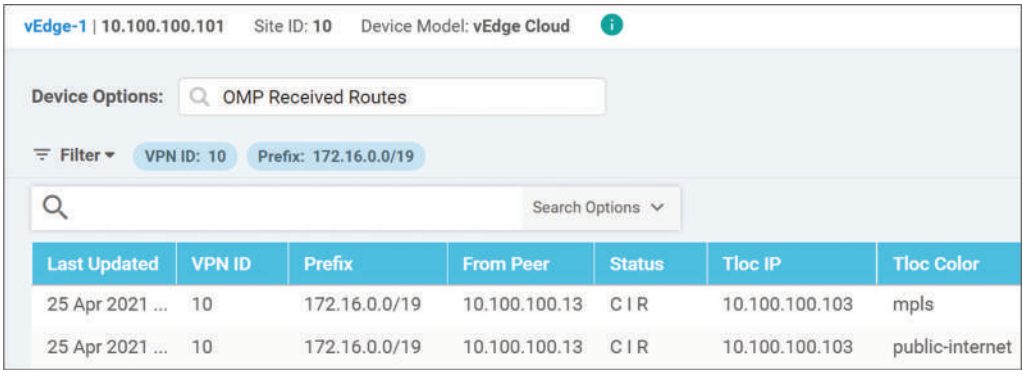


Figure 5-47: *Spoke-to-Spoke: Received OMP Summary Route – vEdge1.*

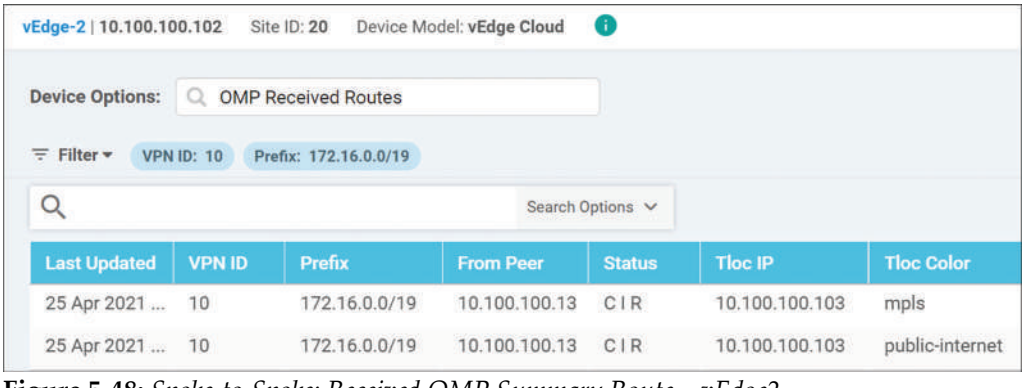


Figure 5-48: *Spoke-to-Spoke: Received OMP Summary Route – vEdge2.*

Figure 5-49 verifies that we have a data plane connection also between sites 10 and 20 though we don't have a GRE tunnel or BFD session between sites (figures 5-50 and 5-51).

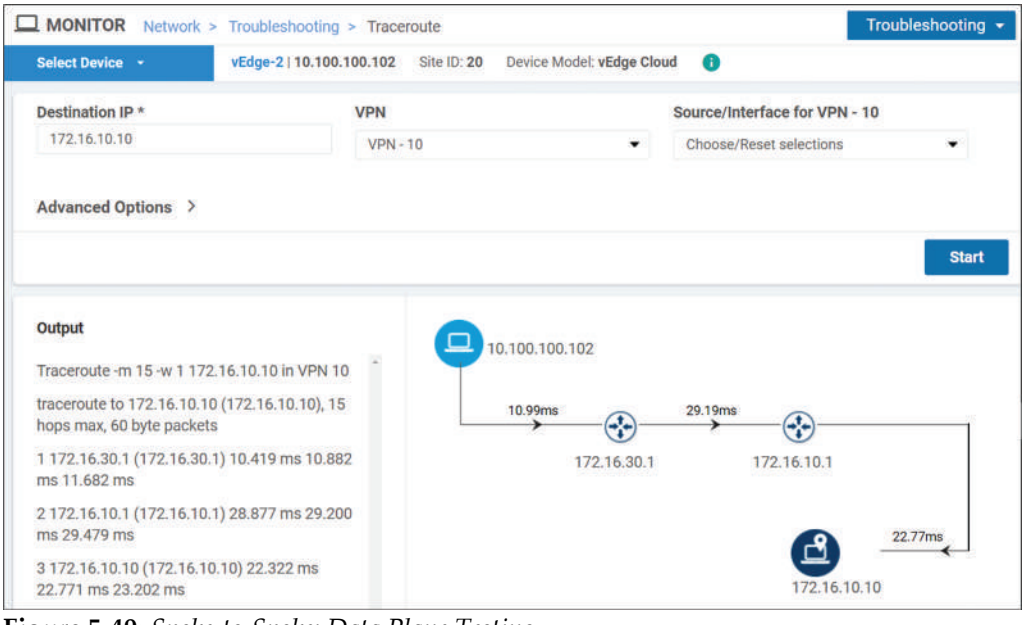


Figure 5-49: *Spoke-to-Spoke: Data Plane Testing.*

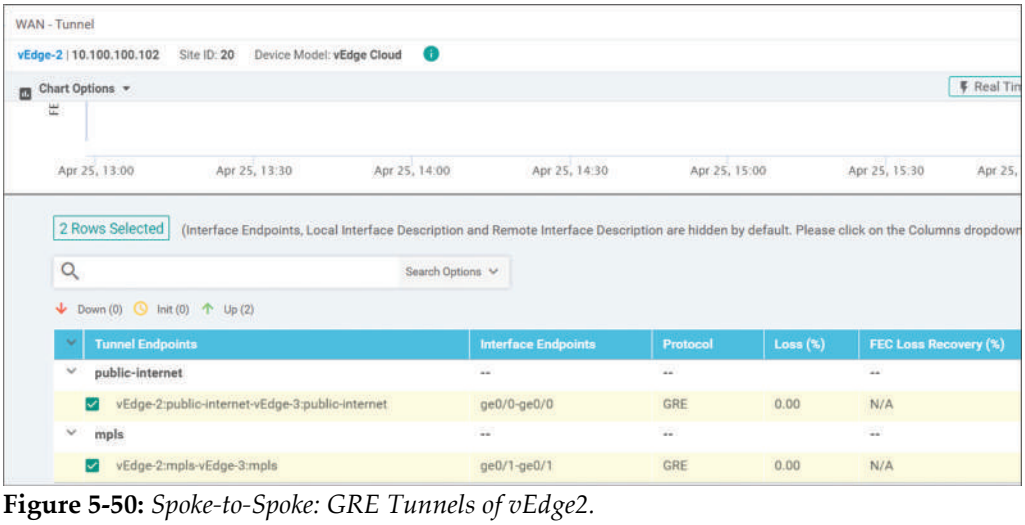


Figure 5-50: *Spoke-to-Spoke: GRE Tunnels of vEdge2.*

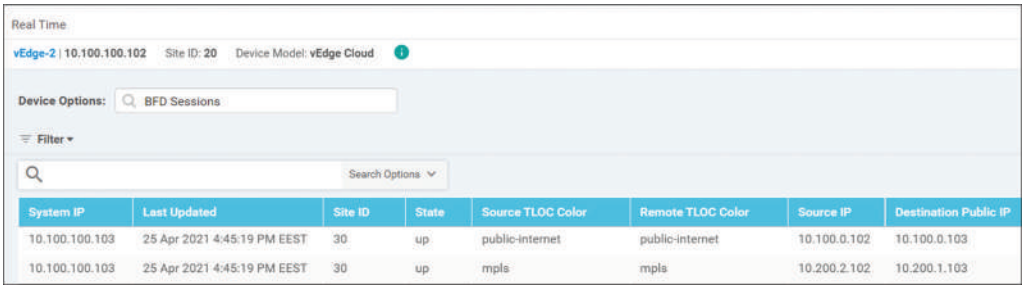


Figure 5-51: *Spoke-to-Spoke: BFD Sessions of vEdge2.*

Summary

This chapter started by explaining tasks required to change vSmart from CLI Managed to vManaged mode. Next, it explained how to construct a Centralized Policy that has a Control Policy which rejects TLOC and OMP routes received from site 10 to be advertised to site 20 and the other way around. As a result, there is no GRE tunnel between site 10 and site 20 and they are restricted from each other. The last section describes how we can allow traffic between site 10 and site 20 by using OMP route summarization in site 30.

Chapter 6: Feature and CLI Templates

Introduction

This chapter introduces two template-based configuration models a) *Feature Templates,* and b) *CLI Templates*. Each protocol/feature has its unique *Feature Templates* where we can define related attributes. We can set the value of each attribute as global or device-specific. Feature Templates are then attached to the *Device Template* which in turn is applied to the device. Our focus is on vEdge System settings and VPN-specific configuration. We can also build Device Templates by using *CLI Template* that, unlike the Feature Template, includes all device configuration in one configuration.

Feature Templates

Figure 6-1 illustrates configuration components from the vEdge2 perspective. It has three VPNs; a) VPN0 for WAN, b) VPN512 for OoB Mgmt, and c) Service VPN10 for the customer. Each of these VPNs has an attached interface(s) with a set of parameters depending on the VPN type. vEdge2 has also basic System settings like system-Id and site-Id.

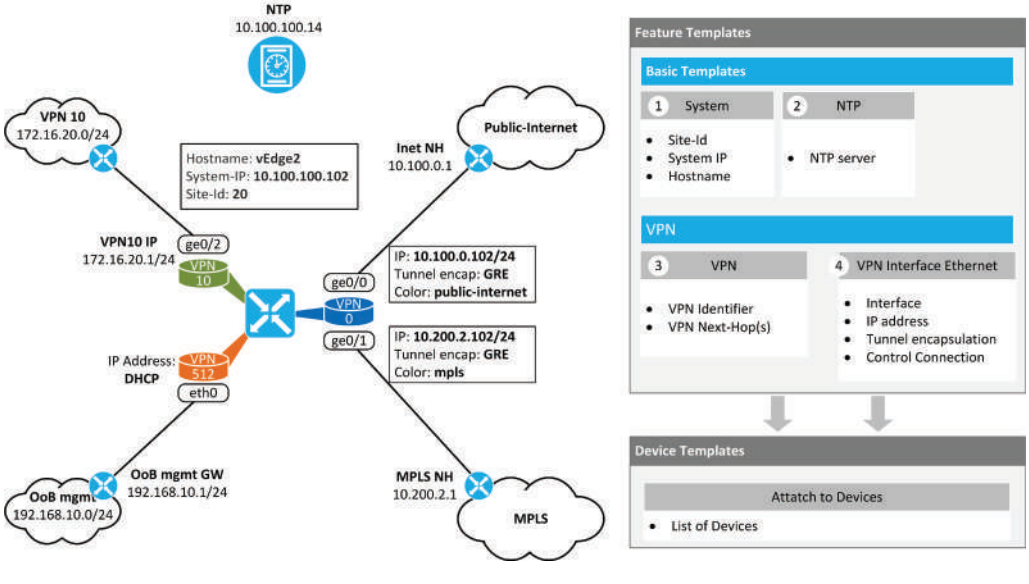


Figure 6-1: *Device Template based on Feature Templates.*

I have already created nine Feature Templates. There are separate templates for the system and NTP as well as for VPNs and interfaces attached to VPNs. I will show how to create the System and NTP template and VPN0 related templates.

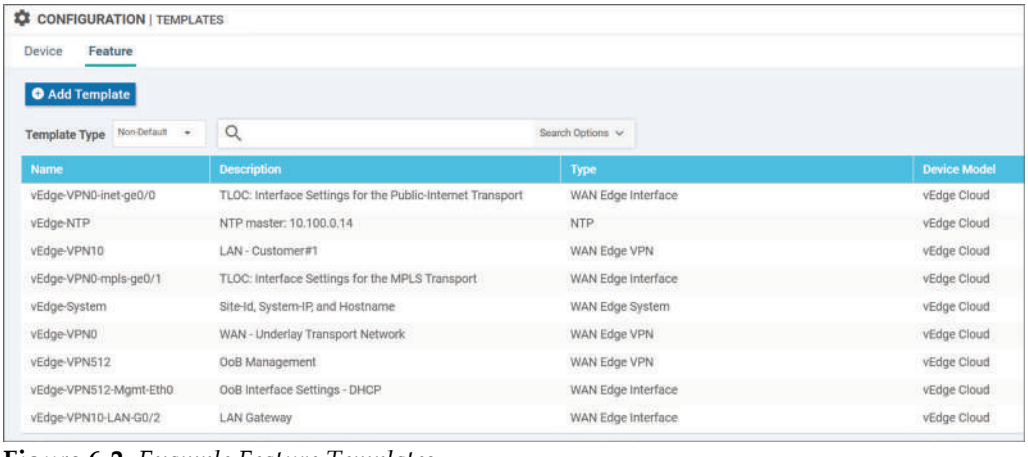


Figure 6-2: *Example Feature Templates.*

You can create Feature Templates by navigating to the C*onfiguration/Templates* window where you first select the platform and then select the Feature Template from the right window. We are first going to do the System template.

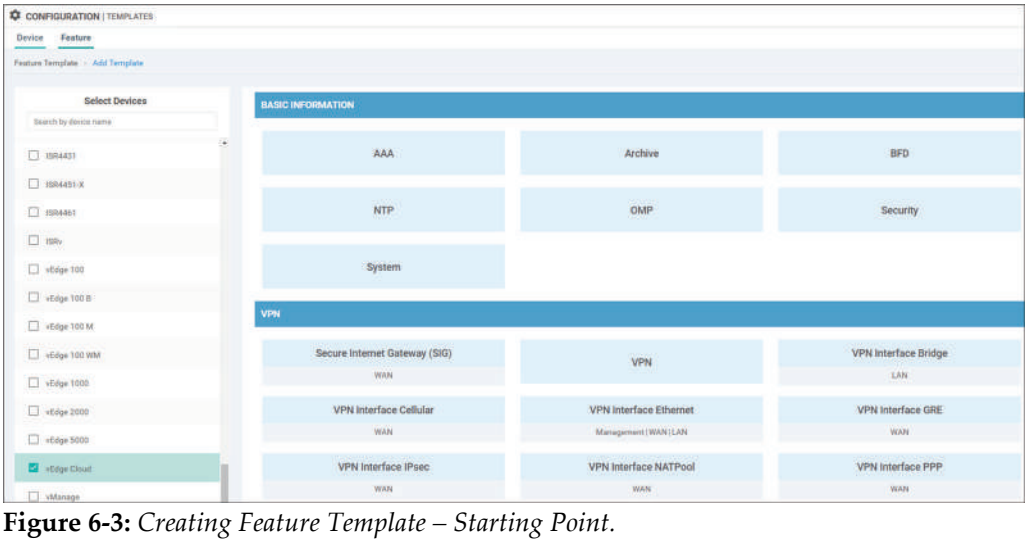


Figure 6-3: *Creating Feature Template – Starting Point.*

System and NTP Templates

Select the *System* box from the *Basic Information* section. Fill in the *Template Name* and *Description* fields. The icon in front of the *Site ID*, *System IP*, and *Hostname* drop-down menu means that the value is device-specific and needs to be filled when the Device Template is attached to vEdge. The text within square brackets shows how this value is shown in the *Update Device Template* window (figure 6-28 on page 146) during the process where Device Template is attached to vEdge. You can modify the text string as you like but remember that if you use the text string "Mickey Mouse", you will be asked to fill the "Mickey Mouse" field. With this in mind, remember to use descriptive texts. Note that the icon in front of the *Overlay Id* and *Timezone* drop-down menu means that the default value is shown in the value field.

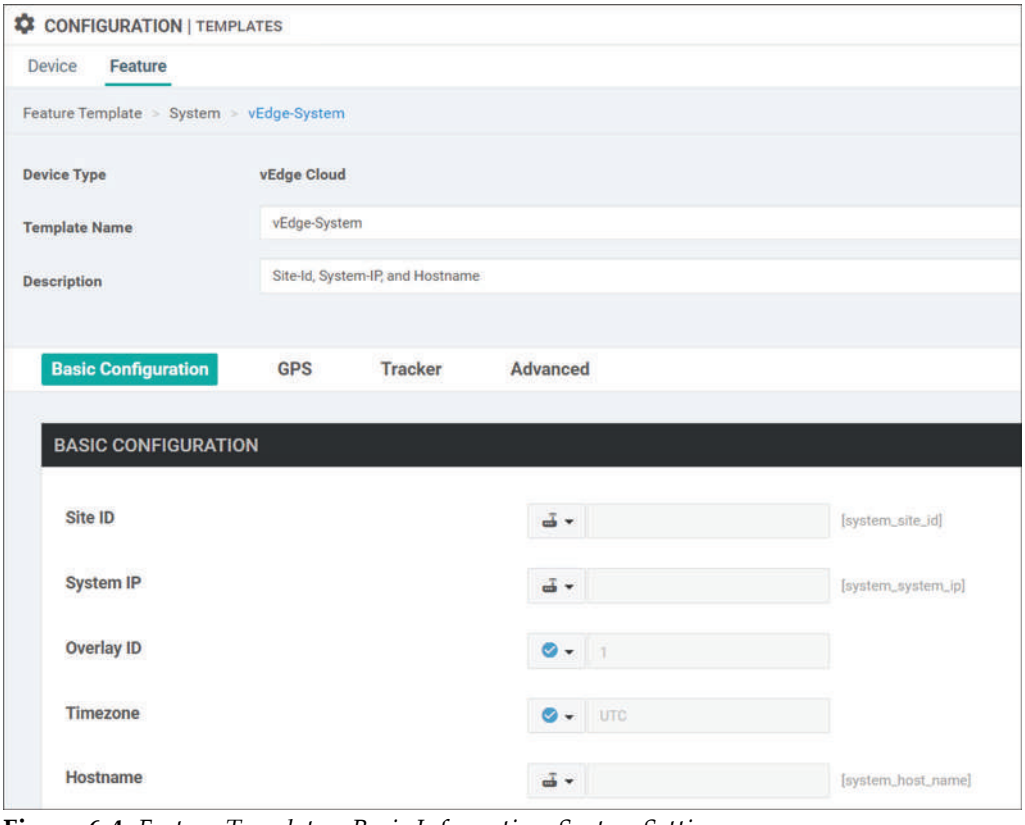


Figure 6-4: *Feature Template – Basic Information: System Settings.*

At this point, we have three device-specific variables: *[system_site_id]*, *[system_system_ip]*, and *[system_host_name]*.

Figure 6-5 shows the NTP Server defined in the *NTP Feature Template*. The globe symbol in front of the server IP address means that this value is fixed, global for all devices which are using this template and it's not being asked during the Device Template attachment process. You can also use device-specific values if needed.

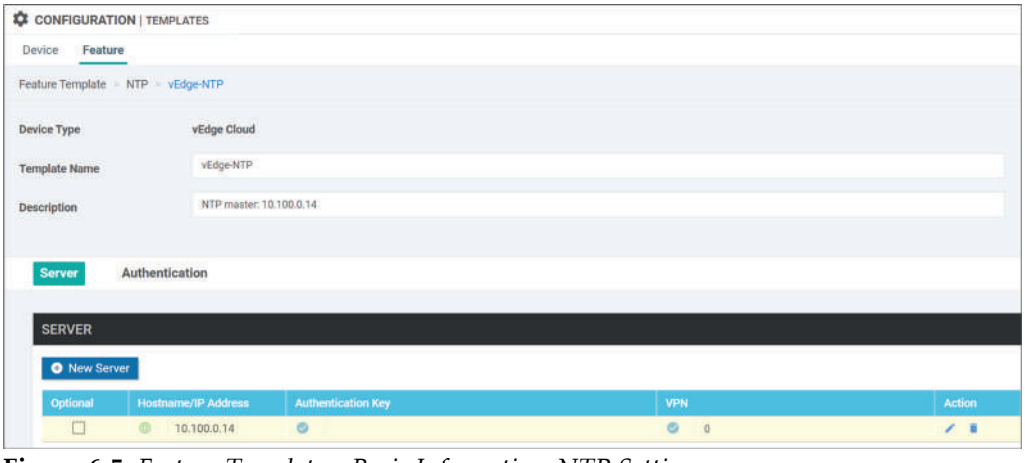


Figure 6-5: *Feature Template – Basic Information: NTP Settings.*

VPN Template

Figure 6-6 illustrates the VPN configuration window. Fill in the *Template Name* and *Description* fields. Choose the *Basic Configuration* and give the VPN Id. Note that I have already done all the configuration and the view in figure 6-7 is the *Edit* view of the template.

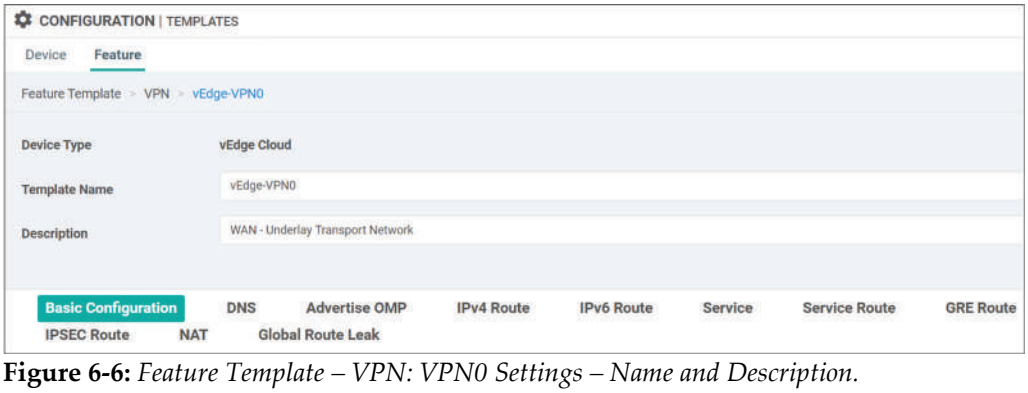


Figure 6-6: *Feature Template – VPN: VPN0 Settings – Name and Description.*

Example 6-7 shows the Basic Configuration Section where we give the VPN Id. By default, there is no name for VPN.

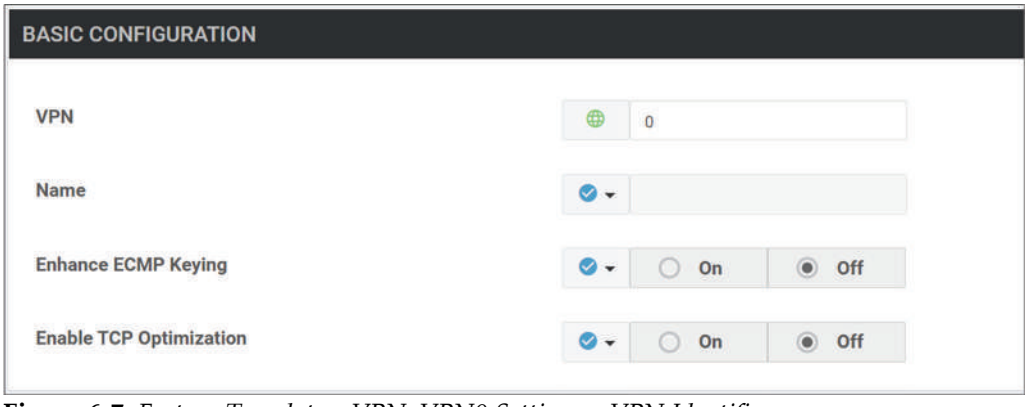


Figure 6-7: *Feature Template – VPN: VPN0 Settings – VPN Identifier.*

VPN0 is attached to two transport networks. We need to define the VPN-specific next-hop addresses under the *IPv4 Route* section. Figure 6-8 shows that we have two next-hops for 0.0.0.0/0.

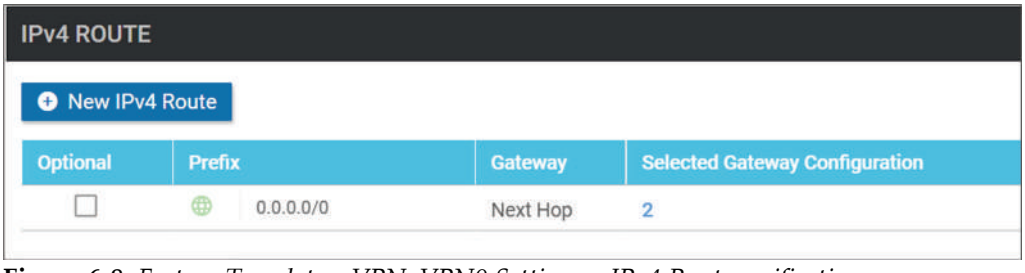


Figure 6-8: *Feature Template – VPN: VPN0 Settings – IPv4 Route verification.*

You can add routes by clicking the *New IPv4 Route* button. Figure 6-9 shows the IPv4 route configuration window. We have two next-hop with device-specific values, each device usually will use a different NH for the default GW. I have modified the default text strings to describe the usage of the default route. Because these values are device-specific values, they are asked during the Device Template attachment process. Now we have five device-specific variables: *[system_site_id]*, *[system_system_ip]*, *[system_host_name]*, *[VPN0-Default-NH-MPLS]*, and *[VPN0-Default-NH-Inet]*.

The Feature Templates for the OoB Management VPN512 and the Customer VPN10 are configured in the same way.

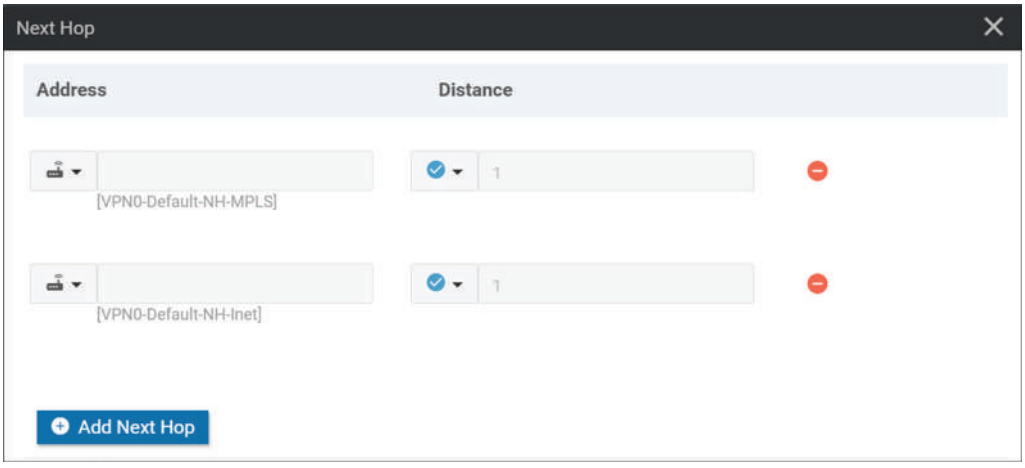


Figure 6-9: *Feature Template – VPN: VPN0 Settings – IPv4 Route Configuration.*

VPN Interface

In the previous section, we configure the VPN0 Feature Template. Now we configure templates for interfaces connected to Public-Internet. The configuration flow for the MPLS transport network follows the same principles. Fill *Template Name* and *Description* fields and go to the *Basic Configuration* section.

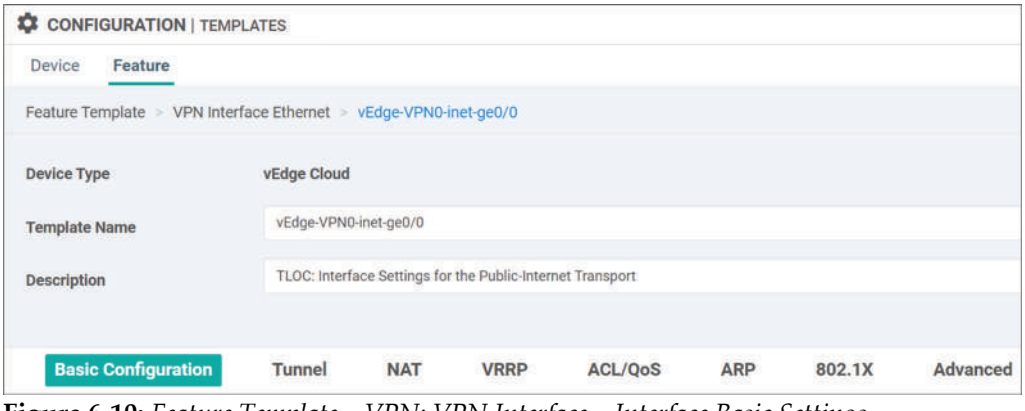


Figure 6-10: *Feature Template – VPN: VPN Interface – Interface Basic Settings.*

The global value for the *Shutdown* field is set to *No*, which enables the interface. We are using interface ge0/0 in every vEdge for Public-Internet transport so we can define the *Interface Name* attribute as a global value. We can also use global value for the *Description*. Interface IP address, in turn, is device-specific. I have changed the description to *[VPN-G0/0-Inet]*. Now we have six device-specific variables: *[system_site_id]*, *[system_system_ip]*, *[system_host_name]*, *[VPN0-Default-NH-MPLS]*, *[VPN0-Default-NH-Inet]*, *and [VPN0-G0/0-Inet]*.

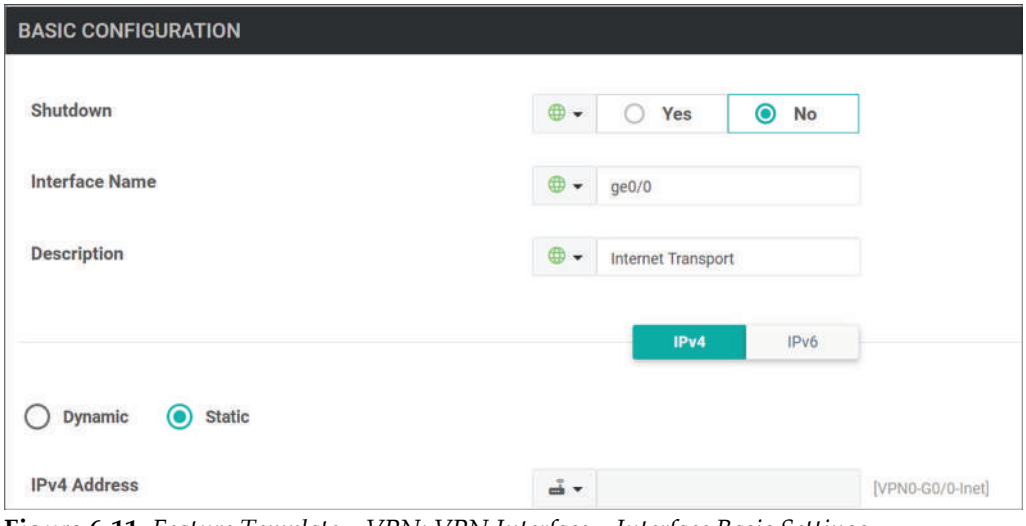


Figure 6-11: *Feature Template – VPN: VPN Interface – Interface Basic Settings.*

All our interface Tunnel configurations are defined as global values. Select radio button to *On* for *Tunnel Interface, Restrict, and Control Connectin* fields. Select the *public-internet* from the *Color* drop-down menu. We are using public-internet for control connection (vBond, vManage, and vSmart), so set the *Control Connection* global value to *On*.

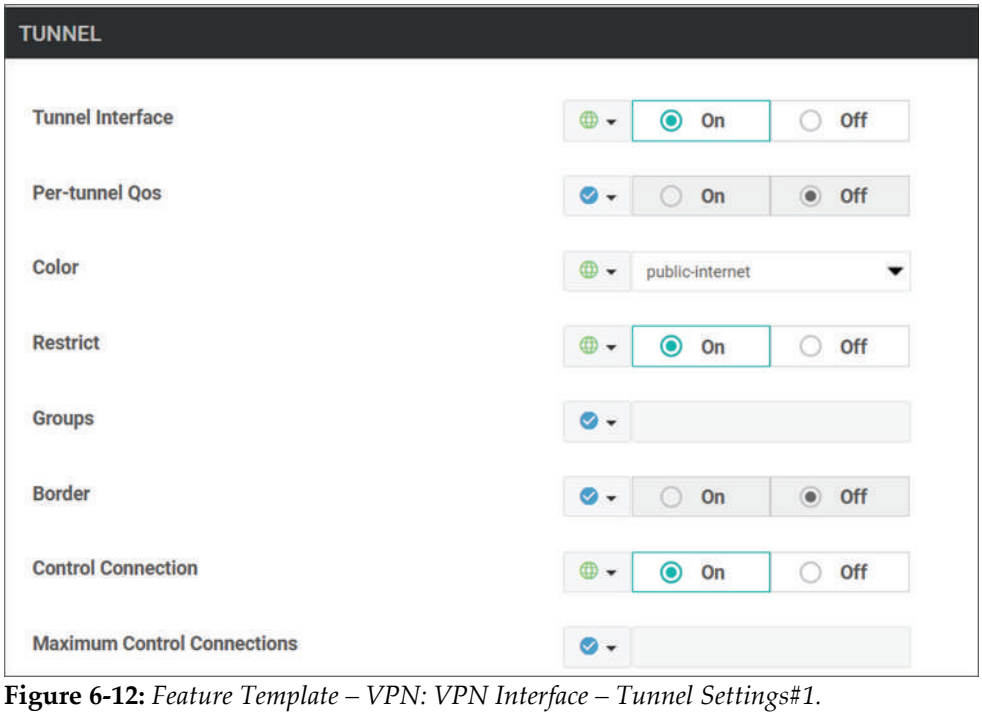


Figure 6-12: *Feature Template – VPN: VPN Interface – Tunnel Settings#1.*

Scroll down to the *Tunnel* Section. Enable all services. At the end of the Tunnel section, click the *Advanced Options*. Here you change the tunnel encapsulation mode from the default IPSec to GRE.

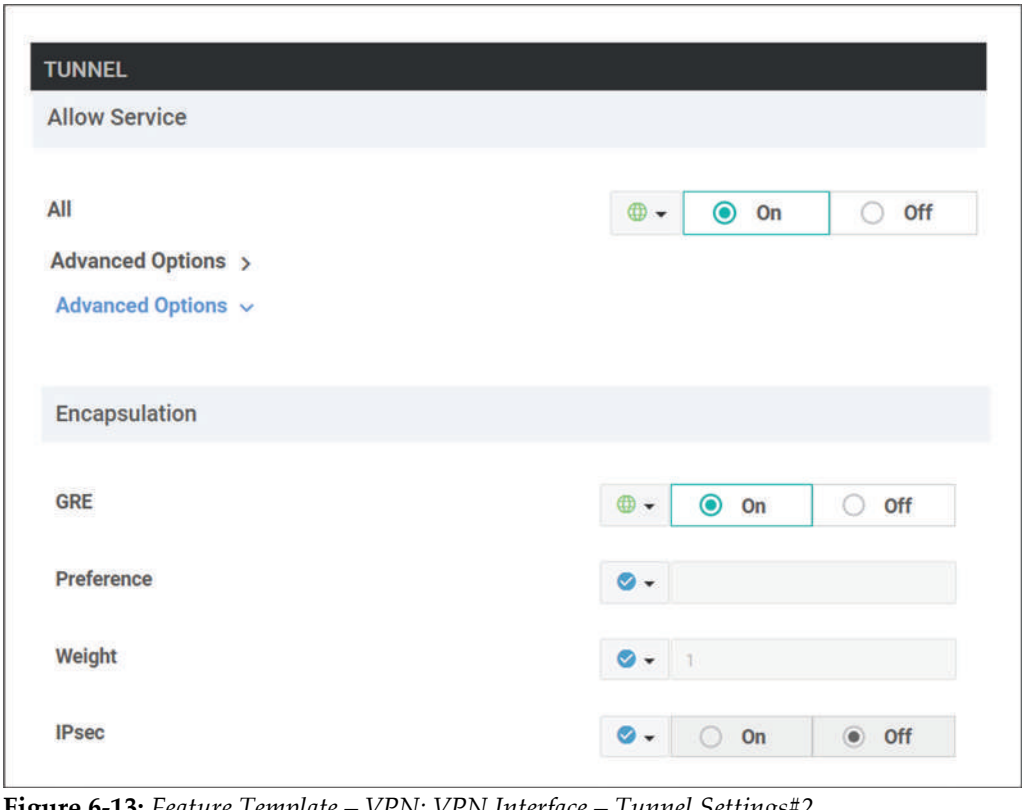


Figure 6-13: *Feature Template – VPN: VPN Interface – Tunnel Settings#2.*

I have already created VPN Interface templates for VPN512 and VPN10. Now we have done all the Feature templates. The next step is to create the *Device Template* where we are going to attach our Feature Templates.

Device Templates

The next step is to create a Device Template and attach all previously defined Feature Templates to it. Navigate back to the *configuration/templates* window and select the *Device* tab. Open the *Create Template* menu and select the *From Feature Template* option. As the figure below shows, we already have a CLI template for vSmart.

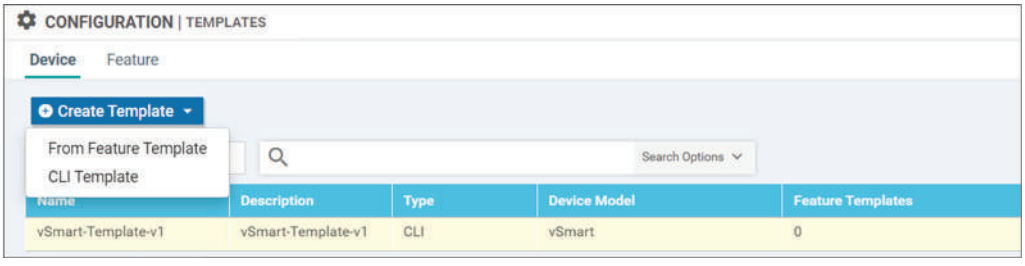


Figure 6-14: *Device Template – From Feature Template: Create Template.*

Select the *vEdge Cloud* from the *Device Model* drop-down menu and fill in the *Template Name* and *Description* fields.

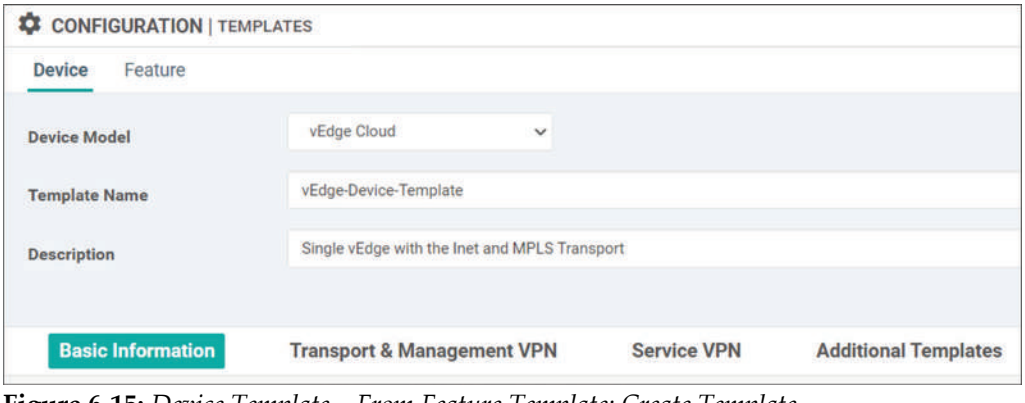


Figure 6-15: *Device Template – From Feature Template: Create Template.*

System and NTP

Go to the *Basic Information* section. Figure 6-16 shows the default values.

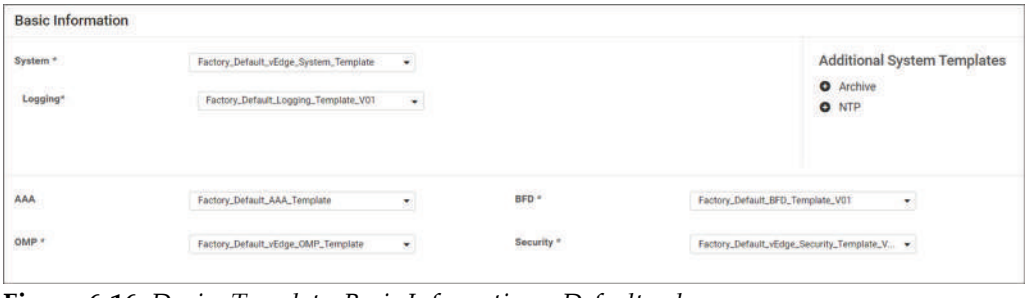


Figure 6-16: *Device Template: Basic Information – Default values.*

Open the *System* drop-down menu and select our previously created Feature Template *vEdge-System* (figure 6-4). Note that the *Description* field is visible on the right-hand side.

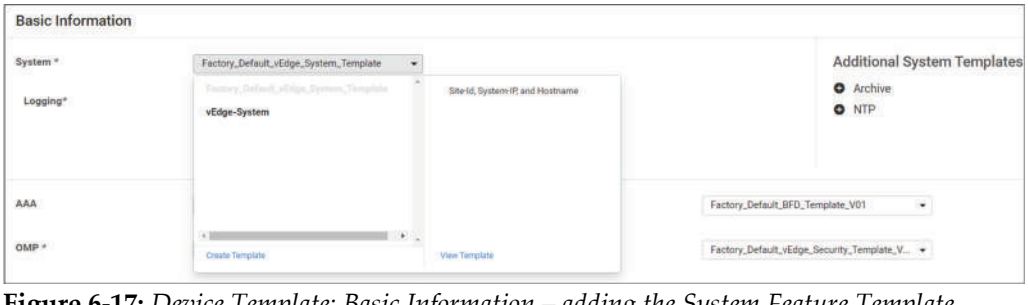


Figure 6-17: *Device Template: Basic Information – adding the System Feature Template.*

Click the *NTP* plus-sign under the *Additional System Template* section. This adds an NTP drop-down menu. Open the NTP drop-down menu and select our Feature Template *vEdge-NTP*.

Figure 6-18: *Device Template: Basic Information – adding the NTP Feature Template.*

VPN Template

Scroll down to the *Transport & Management VPN* section. Figure 6-19 shows the default fields and their values.

Figure 6-19: *Device Template: VPNInformation – adding the VPN and VPN Interface Feature Templates.*

Open the VPN0* drop-down menu and select our *vEdge-VPN0* Feature Template. Next click the *VPN Interface* plus-sign under the *Additional VPN 0 Template* section. This adds a VPN Interface drop-down menu. Open the VPN Interface drop-down menu and select our Feature Template *vEdge-VPN0-mpls-ge0/0*. Click the VPN Interface plus sign again and select the other *VPN Interface* Feature Template *vEdge-VPN0-inet-ge0/0*. Repeat these processes for the OoB Management VPN512. At this point, it should be clear why the Feature Template naming needs to be clear and descriptive.

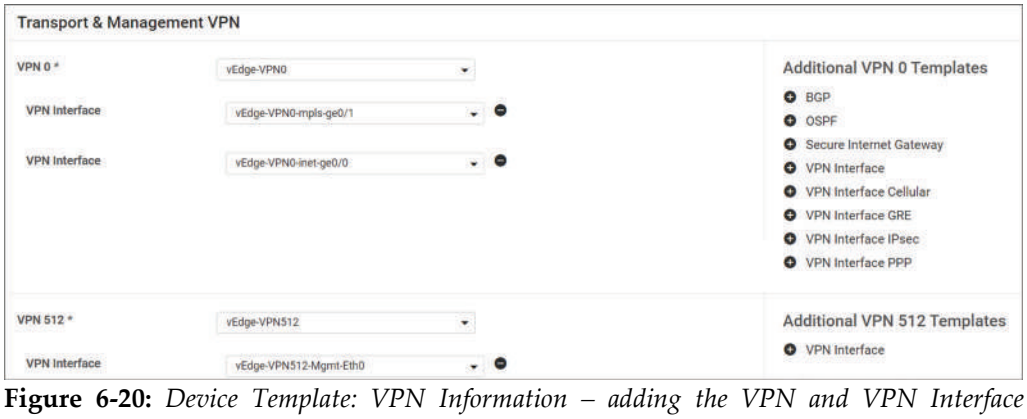


Figure 6-20: *Device Template: VPN Information – adding the VPN and VPN Interface Feature Templates.*

Scroll down to the *Service VPN* section and click the *Add VPN* button.

Figure 6-21: *Device Template: VPN Information – adding the Service VPN.*

The first step is to select the Service VPN. Our *vEdge-VPN10* is listed in the *Available VPN Templates* window. Move it to the *Selected VPN Templates* window by clicking the arrow.

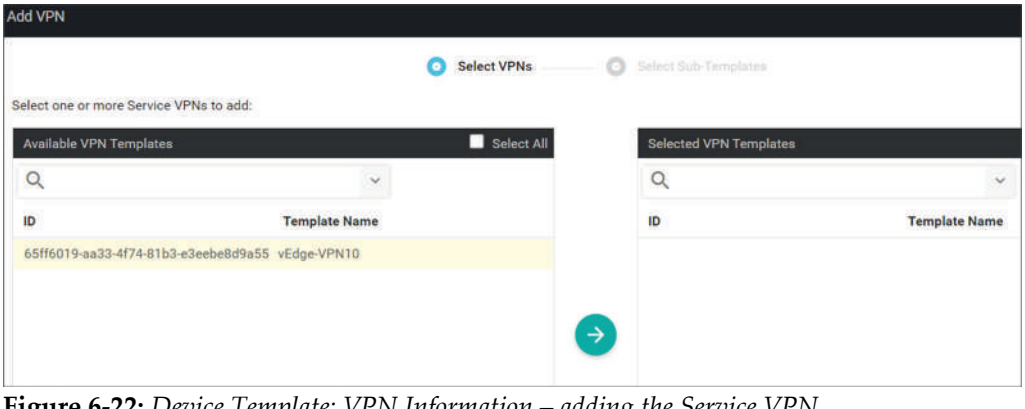


Figure 6-22: *Device Template: VPN Information – adding the Service VPN.*

As a next step, we add *VPN Interface* as a Sub-Template. Click the *VPN Interface* under the *Additional VPN Templates* section and select our Feature Template *vEdge-VPN10-LAN-G0/2* from the drop-down menu. Click the *Add* button.

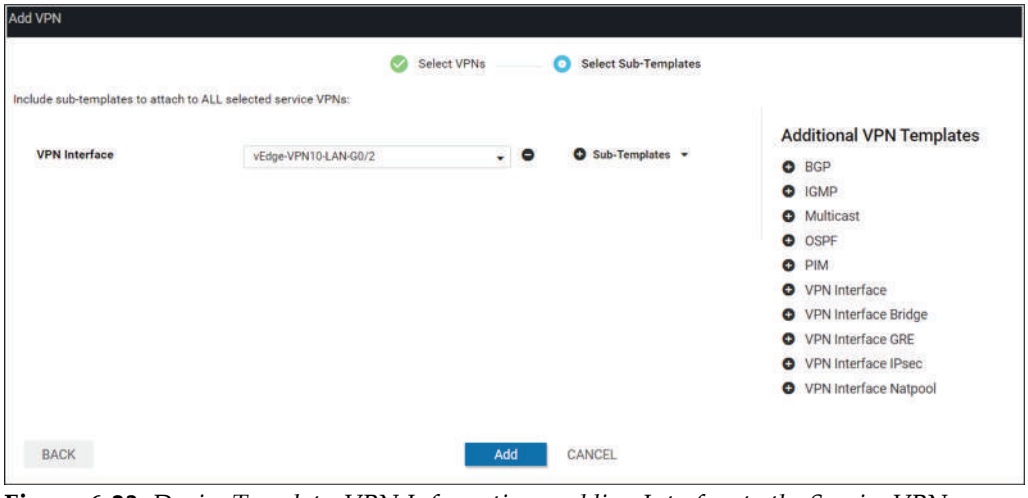


Figure 6-23: *Device Template: VPN Information – adding Interface to the Service VPN.*

As the last step click the Create button.

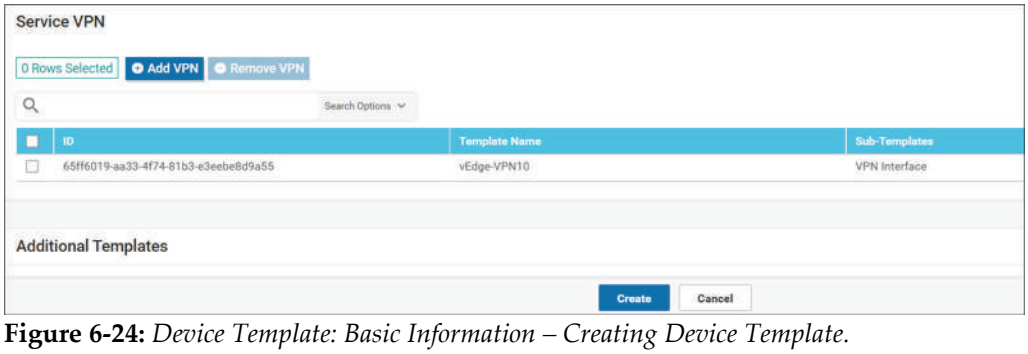


Figure 6-24: *Device Template: Basic Information – Creating Device Template.*

Attach Device Template to vEdge

Select our Device Template *vEdge-Device-Template* and select the *Attach Device* option from the Options menu […].

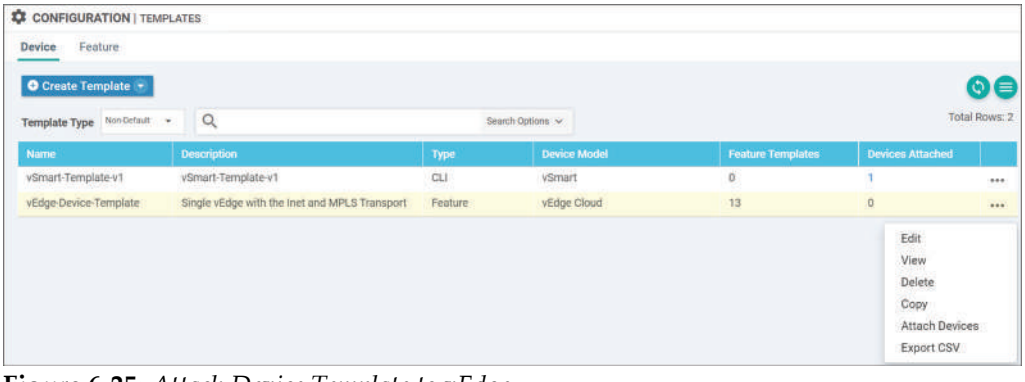


Figure 6-25: *Attach Device Template to vEdge.*

Select the vEdge2 from The *Available Devices* section and move it to the *Selected Device* section. Click the *Attach* button.

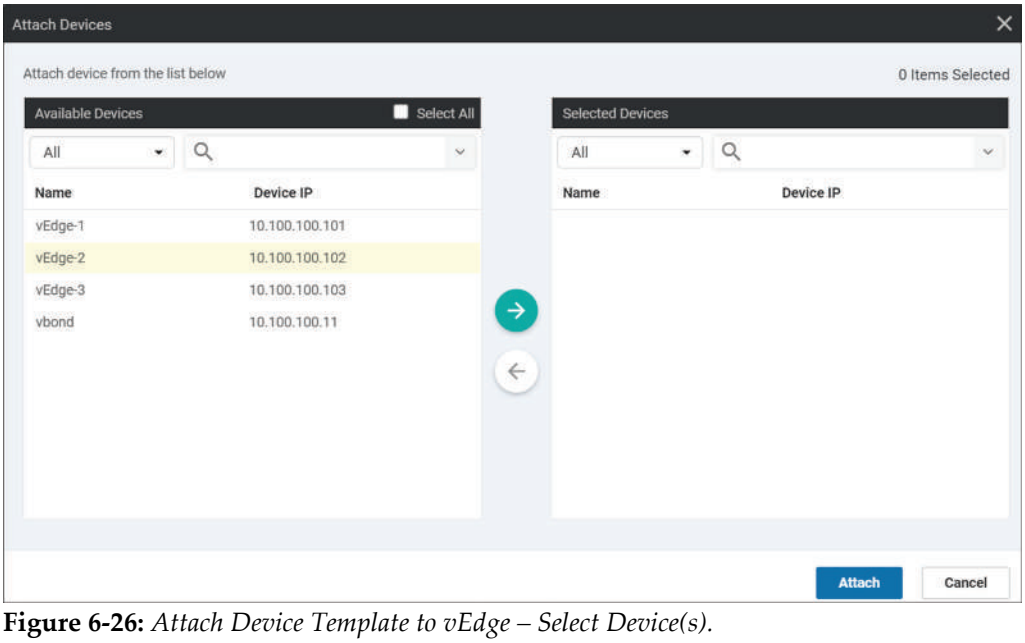


Figure 6-26: *Attach Device Template to vEdge – Select Device(s).*

The next step is to fill device-specific values from the vEdge2 perspective. Select the *Edit Device Template* from the *Options* menu […]. From the System and VPN0 point of view these are a) *[system_site_id]*, *[system_system_ip]*, *[system_host_name]*, *[VPN0-Default-NH-MPLS]*, *[VPN0-Default-NH-Inet]*, *and [VPN0-G0/0-Inet]* which we defined for Feature Template. Note that if you click the Next button before editing, you will get an error message.

Note that you can fill the required fields also in this window.

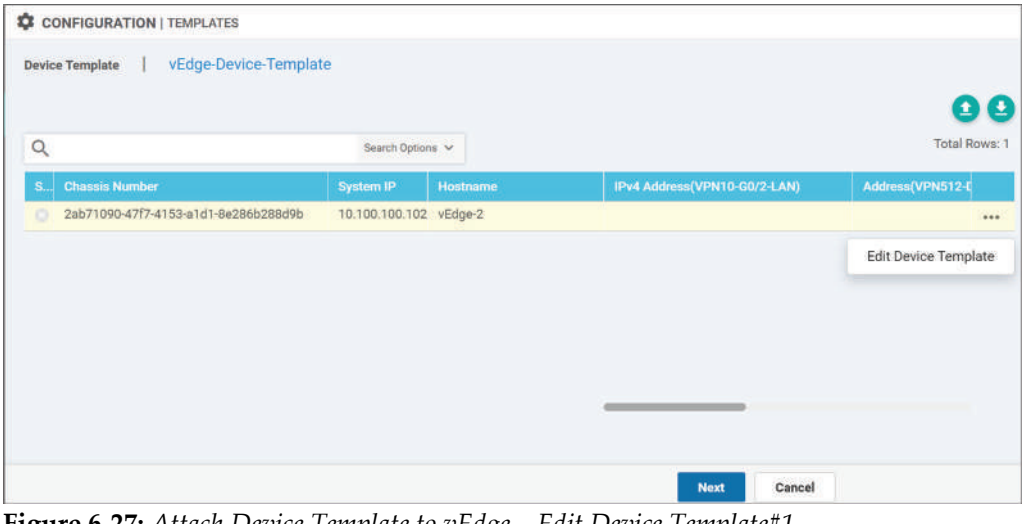


Figure 6-27: *Attach Device Template to vEdge – Edit Device Template#1.*

You can see all our device-specific fields in the *Update Device Template* window. If you didn't use descriptive names there's no way to recognize which fields are related to which VPNs or Interfaces. Note that *Address* fields are related to Next-Hop addresses while *IP address* fields are related to Interface IP address. Let's take VPN0 as an example. Our VPN0 Feature Templates include two Next-Hops, one towards the Public-Internet transport network which we'll be asked to fill in the *Address (VPN0-Default-NH-Inet)* and the *Address (VPN0-Default-NH-MPLS)* fields. Then we have two *VPN Interface* Feature Templates which are sub-templates for the VPN template and are shown in the windows as *IPv4 Address (VPN0-G0/0-Inet)* and *IPv4 Address (VPN0-G0/0-mpls)*. These fields describe the interface IP addresses and subnet masks. After filling in all fields, click the *Update* button.

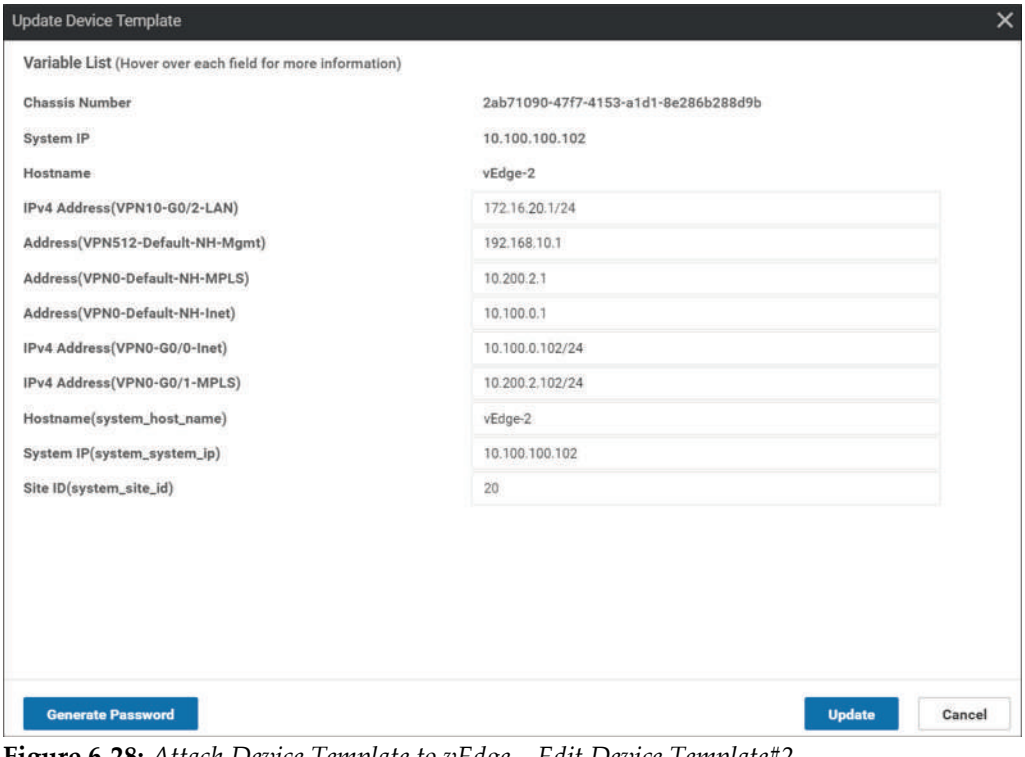


Figure 6-28: *Attach Device Template to vEdge – Edit Device Template#2.*

Click the *Next* button to proceed.

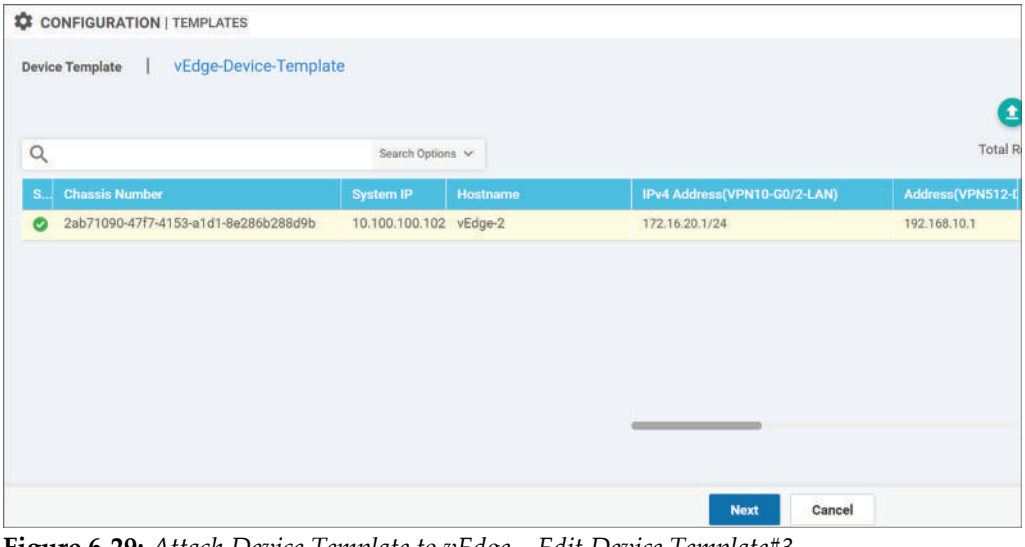


Figure 6-29: *Attach Device Template to vEdge – Edit Device Template#3.*

Before adding the configuration to vEdge, you can see the configuration by selecting vEdge2 from the left *Device List* section and clicking the *Config Preview* button.

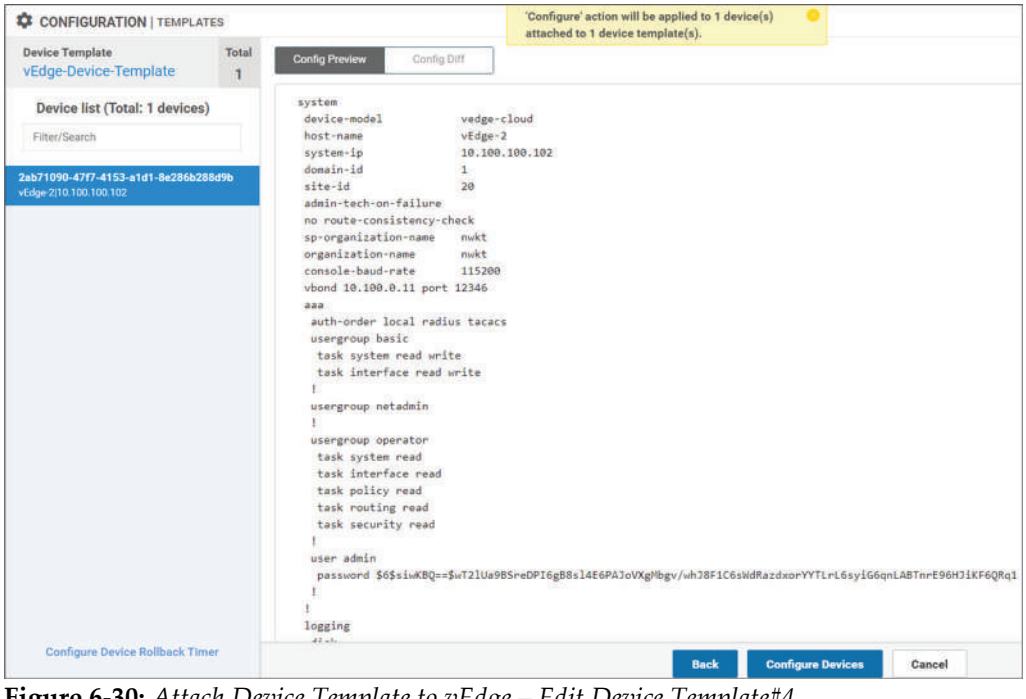


Figure 6-30: *Attach Device Template to vEdge – Edit Device Template#4.*

Example 6-1 below shows the configuration that will be pushed to vEdge. Highlighted parts are the configurations that were specified as device-specific in Feature Templates

```
system
   device-model            vedge-cloud
   host-name               vEdge-2
   system-ip               10.100.100.102
   domain-id               1
   site-id                 20
   admin-tech-on-failure
   no route-consistency-check
   sp-organization-name    nwkt
   organization-name       nwkt
   console-baud-rate       115200
   vbond 10.100.0.11 port 12346
   aaa
```

```
   <snipped>
   !
  !
  logging
   disk
    enable
   !
  !
  ntp
   server 10.100.0.14
    version 4
   exit
  !
 !
 omp
  <snipped>
 !
 security
  ipsec
   authentication-type sha1-hmac ah-sha1-hmac
  !
 !
 vpn 0
  interface ge0/0
   description "Internet Transport"
   ip address 10.100.0.102/24
   tunnel-interface
    encapsulation gre
    color public-internet restrict
    control-connections
    allow-service all
    <snipped>
   !
   no shutdown
  !
  interface ge0/1
   description "MPLS Transport"
   ip address 10.200.2.102/24
   tunnel-interface
    encapsulation gre
    color mpls
    max-control-connections 0
    no control-connections
    allow-service all
    <snipped>
```

```
    !
    no shutdown
   !
   ip route 0.0.0.0/0 10.100.0.1
   ip route 0.0.0.0/0 10.200.2.1
  !
  vpn 10
   interface ge0/2
    ip address 172.16.20.1/24
    no shutdown
   !
  !
  vpn 512
   interface eth0
    description Management
    ip dhcp-client
    no shutdown
   !
   ip route 0.0.0.0/0 192.168.10.1
  !
 !
!
```

Example 6-1: *Complete Configuration.*

You can also compare the new and the old configuration by clicking the *Config Diff* button.

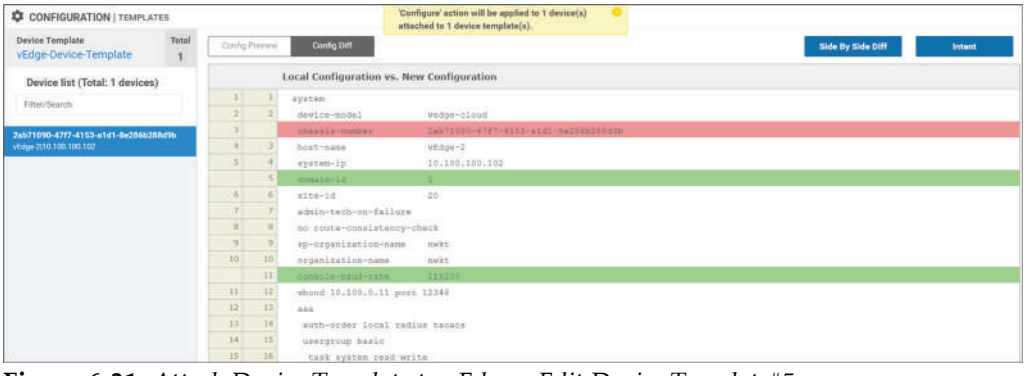


Figure 6-31: *Attach Device Template to vEdge – Edit Device Template#5.*

Click the *Configure Device* button. The *Task View* window shows the configuration process.

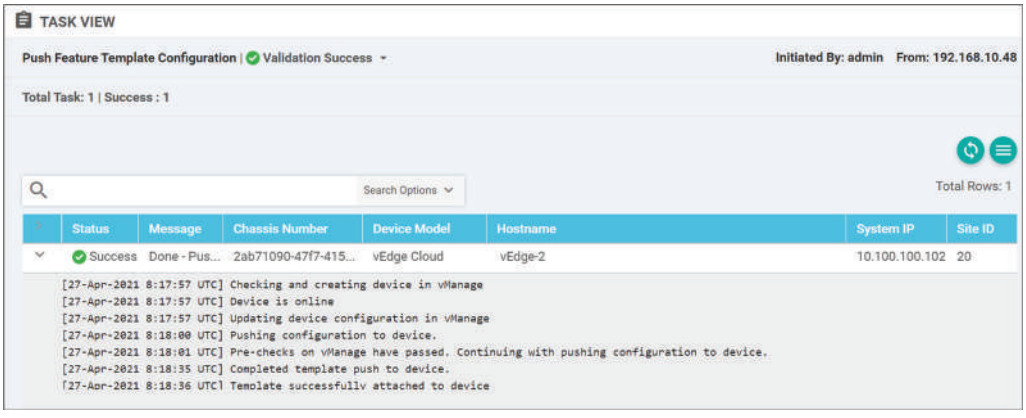


Figure 6-32: *Attach Device Template to vEdge – Configuring Device.*

Verification

Navigate to *configuration/devices* and select *WAN Edge List*. The *Mode* column shows that vEdge2 is in vMange mode while the other vEdges are still in CLI mode. The *Assigned Template* column shows our Device Template *vEdge-Device-Template*.

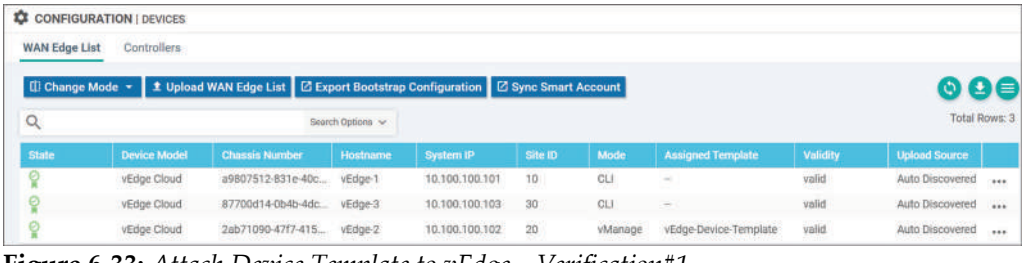


Figure 6-33: *Attach Device Template to vEdge – Verification#1.*

Figure 6-34 verifies that vEdge2 has two BFD sessions with vEdge3. Remember that we have implemented the Hub and Spoke topology where vEdge3 is the Hub and vEdge1 and vEdge2 are Spokes.

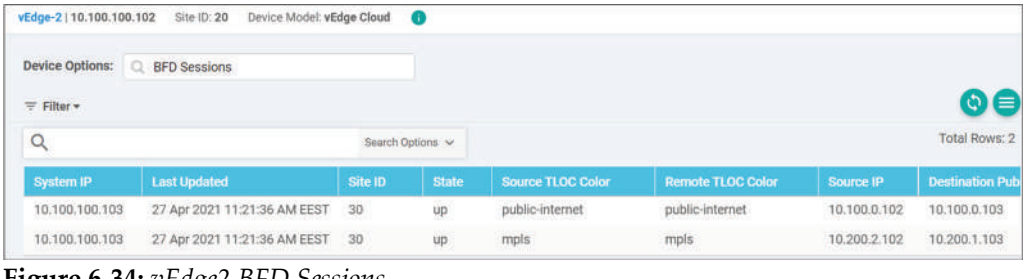


Figure 6-34: *vEdge2 BFD Sessions.*

Figure 6-35 shows that vEdge has IP routes 172.16.30.0/24 and 172.16.0.0.0/19 so the control plane is working.

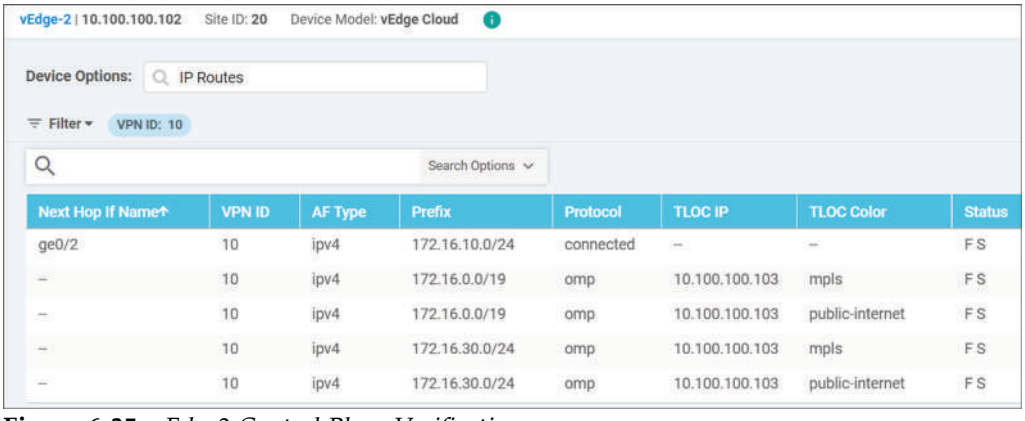


Figure 6-35: *vEdge2 Control Plane Verification.*

Figure 6-36 shows that also Data Plane is working as expected. The traceroute is taking from the vEdge2 to vEdge1 VPN10 LAN interface 172.16.10.1.

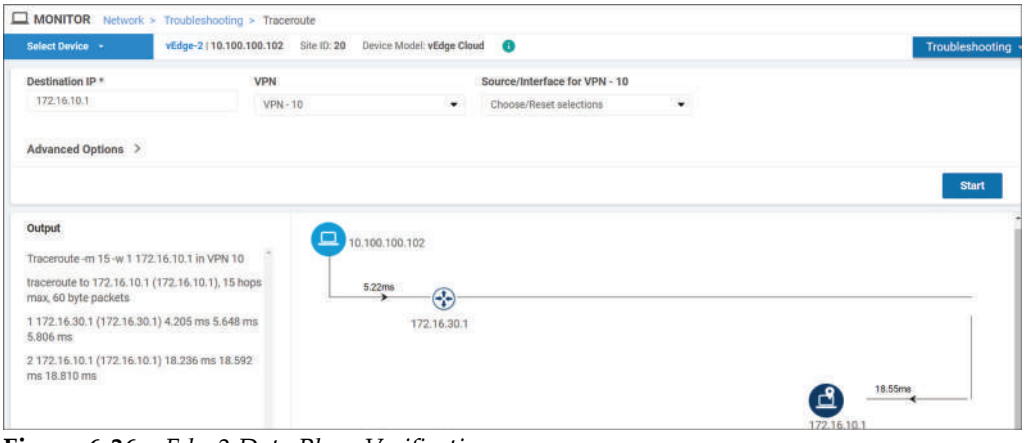


Figure 6-36: *vEdge2 Data Plane Verification.*

Detach Device Template from vEdge

In case you want to revert vEdge to back the *CLI Mode* you just need to detach the Device Template from the device. Navigate to the *configuration/template* window and select the *Device* tab. Then choose the template and select *Detach Devices* from the Options menu […].

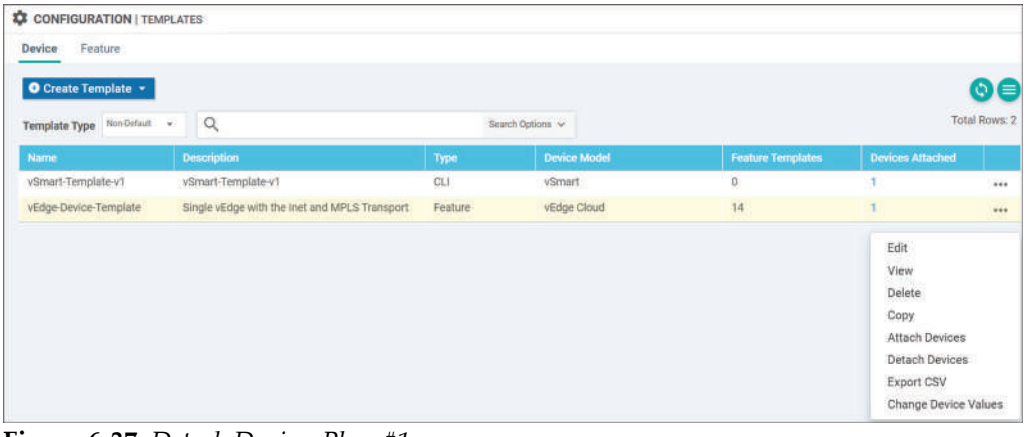


Figure 6-37: *Detach Device- Phase#1.*

Figures 6-38 and 6-39 show the process.

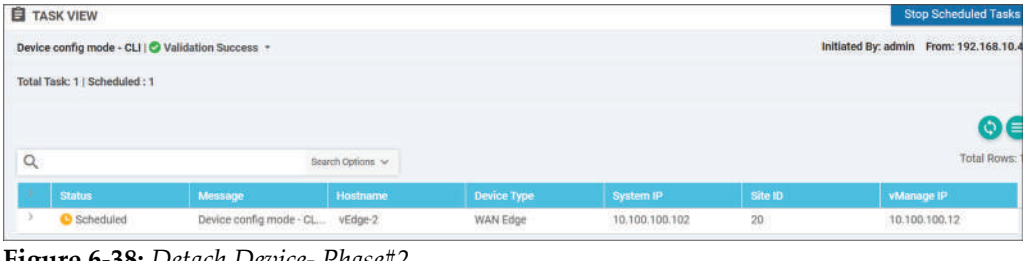


Figure 6-38: *Detach Device- Phase#2.*

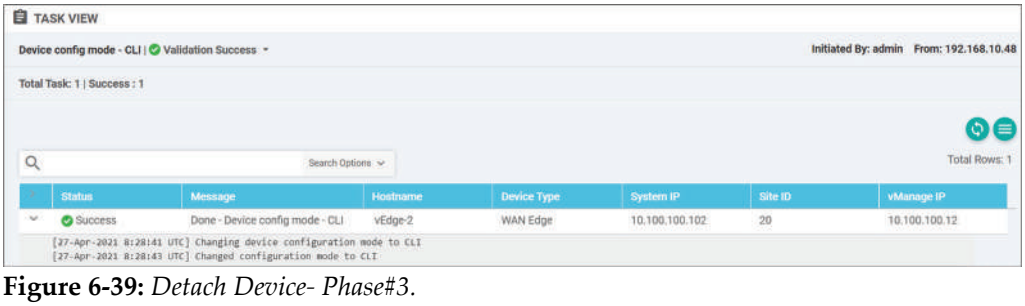


Figure 6-39: *Detach Device- Phase#3.*

CLI Templates with Variables

In the previous section, we implement the Device Template by using Feature Templates. You can also use CLI Template-based Device Template. Select the CLI Template from the *Create Template* menu.

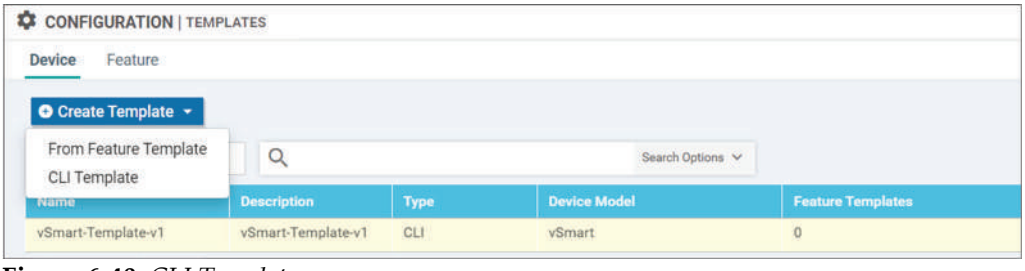


Figure 6-40: *CLI Template.*

Fill in the *Device Model* and *Description* fields. We want to create a CLI Template that can be used with all our vEdge devices and that is why we enable device-specific settings by using variables. As a first step, we need to download the base configuration file. I have selected vEdge-2 from the *Load Running config from reachable device* drop-down menu. When the configuration is loaded, we need to select the values from the configuration that we want to be configurable during the template implementation process. In figure 6-41 these device-specific fields are host-name, system-ip, and site-id. To make the value vEdge-2 in the host-name field to be configurable, we need to first highlight it and then click the *Create Variable* selector.

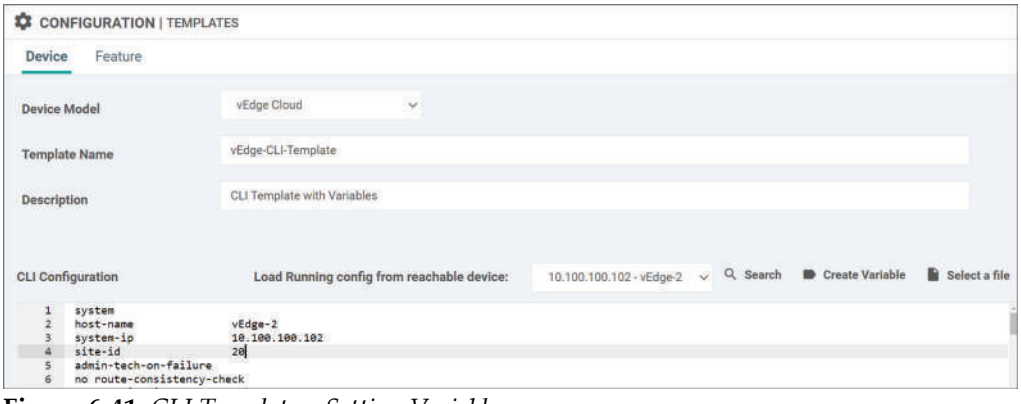


Figure 6-41: *CLI Template – Setting Variables.*

Figure 6-42 shows the output when we have set all three variables. Remember to use descriptive naming just like what we did with Feature Templates because you will be asked to fill these fields when the Device Template is attached to vEdge devices. Even though figure 6-36 only shows the partial configuration you have to set all device-specific settings as variables.

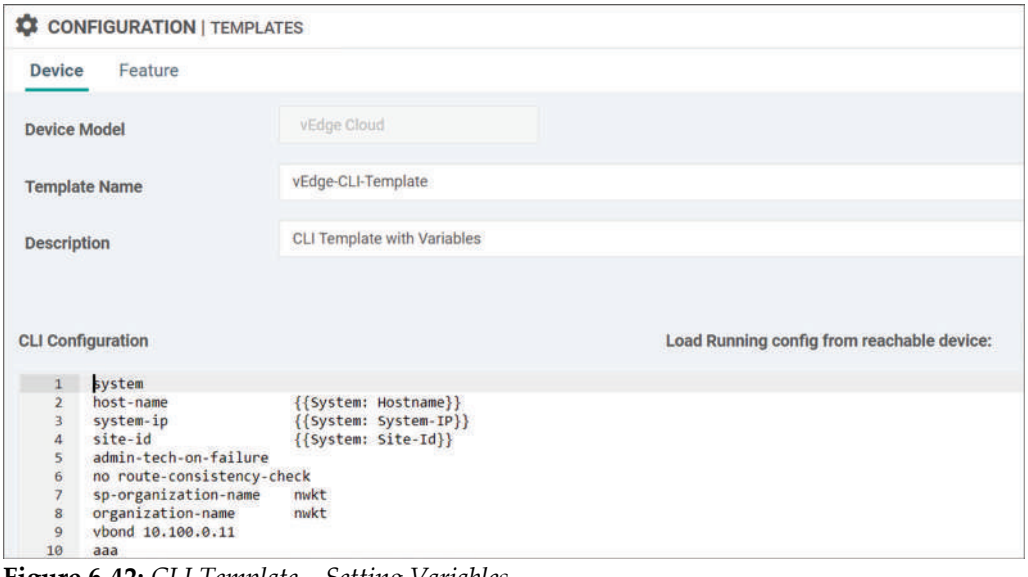


Figure 6-42: *CLI Template – Setting Variables.*

Attach Template to vEdge

Attaching CLI-based Device Template to vEdge follows the same workflow that we use with Device Template based on Feature Template. Select our Device Template *vEdge-CLI-Template* and select the *Attach Device* option from the *Options* menu […].

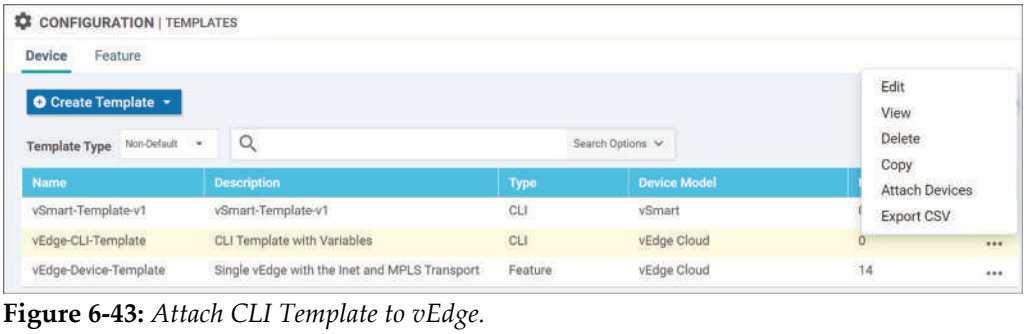


Figure 6-43: *Attach CLI Template to vEdge.*

Select the vEdge2 from The *Available Devices* section and move it to the *Selected Device* section. Click the *Attach* button.

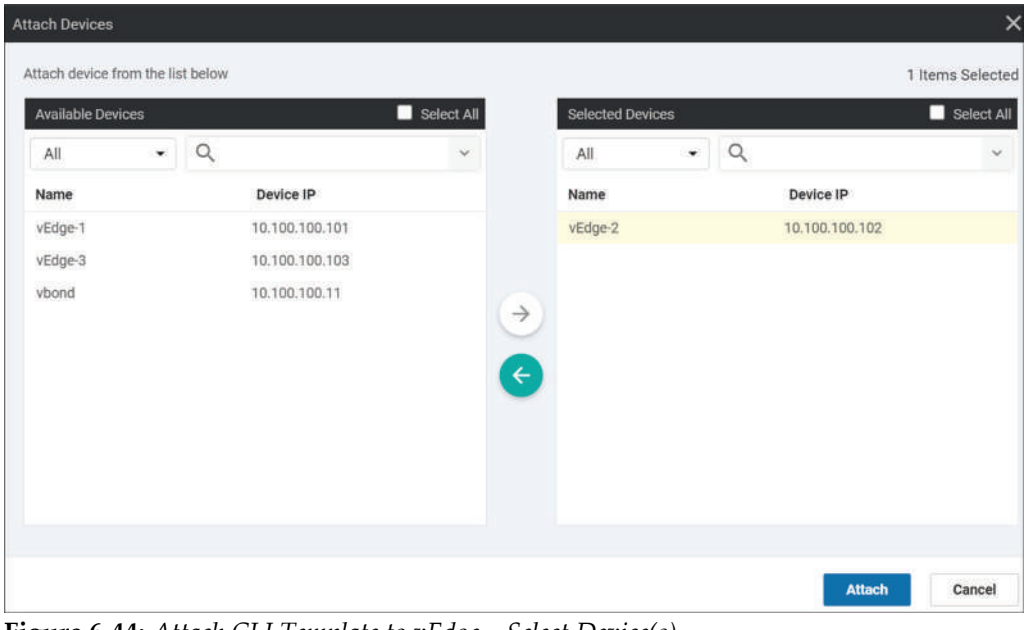


Figure 6-44: *Attach CLI Template to vEdge – Select Device(s).*

The next step is to fill device-specific values from the vEdge2 perspective. Select the *Edit Device Template* from the *Options* menu […].

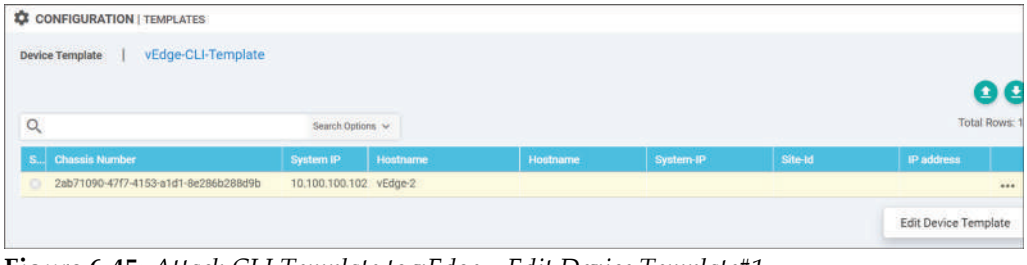


Figure 6-45: *Attach CLI Template to vEdge – Edit Device Template#1.*

Fill in all variables.

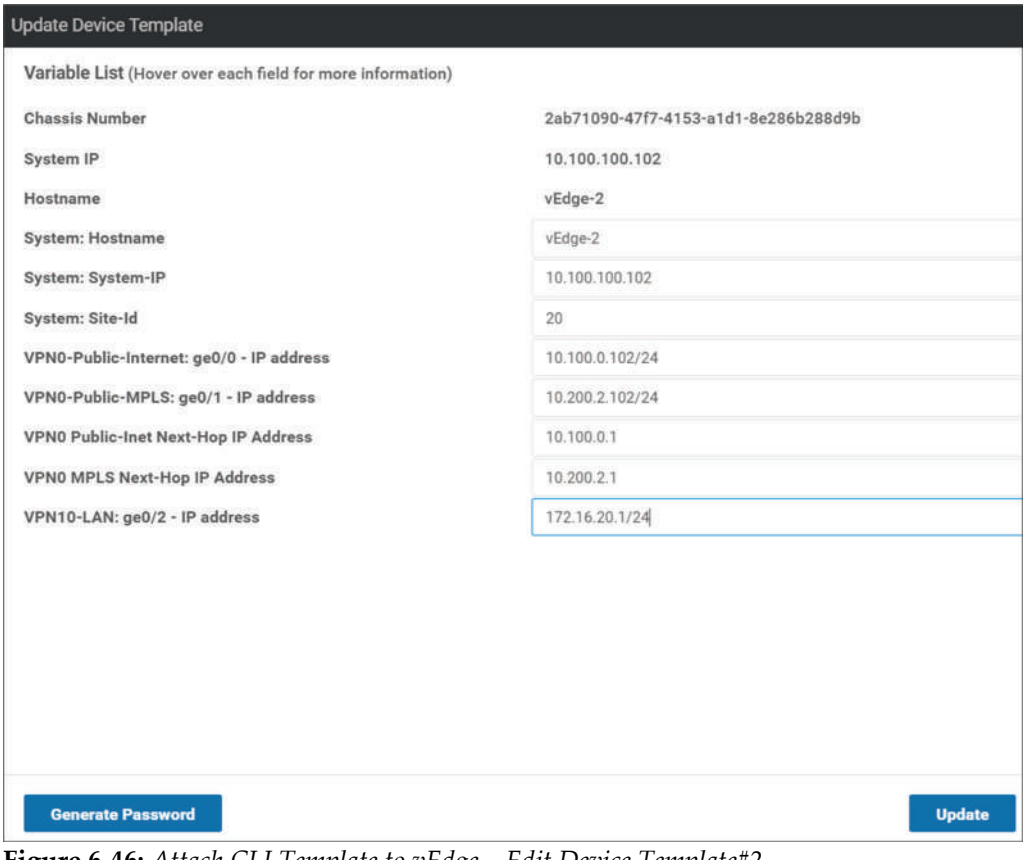


Figure 6-46: *Attach CLI Template to vEdge – Edit Device Template#2.*

You can check the configuration by selecting the vEdge2 from the left *Device List* section and clicking the *Config Preview* button.

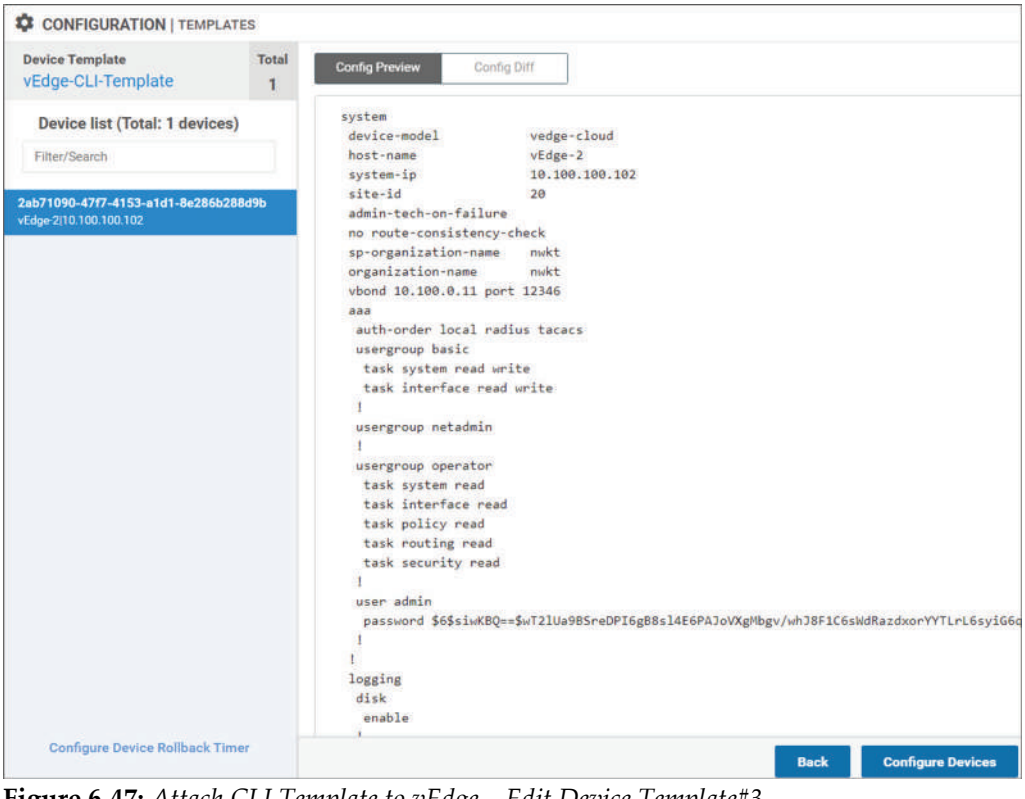


Figure 6-47: *Attach CLI Template to vEdge – Edit Device Template#3.*

Click the *Configure Device* button. The *Task View* window shows the configuration process.

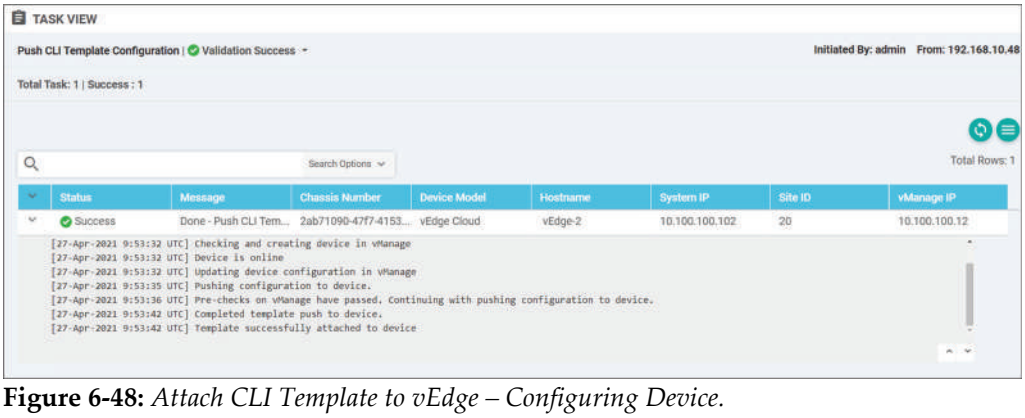


Figure 6-48: *Attach CLI Template to vEdge – Configuring Device.*

Verification

Figure 6-49 verifies that Data Plane is working as expected. The traceroute is taking from the vEdge2 to host 172.16.10.10 attached to vEdge1 VPN10.

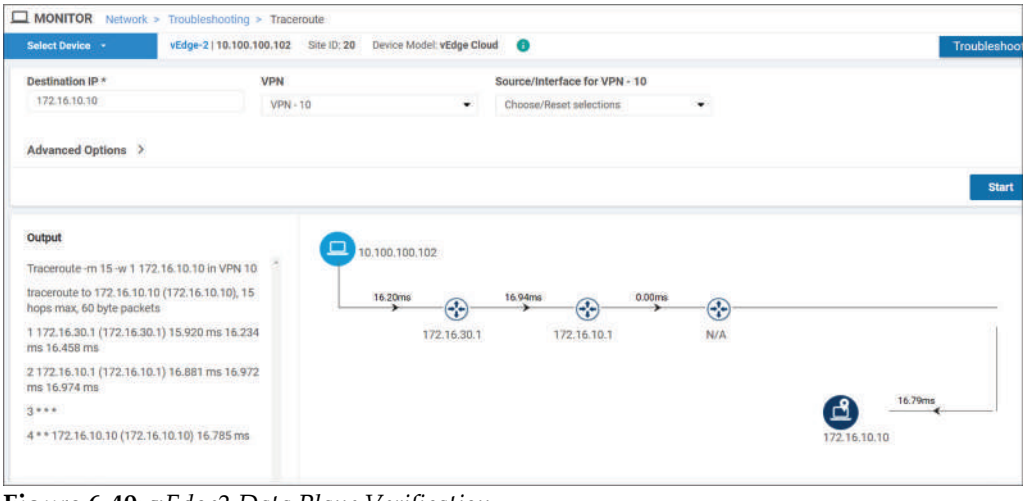


Figure 6-49: *vEdge2 Data Plane Verification.*

Summary

This chapter showed how to configure vEdge devices using Device Template. The first part introduced the Feature Template where each property/protocol is configured as individual components that are then combined under the Device Template. The second part explained how to use CLI based Device template where we create one configuration file with configurable variables.

Chapter 7: TLOC Extension

Introduction

Dual-Homed site with two or more vEdge devices can be connected to external WAN networks by using a) one direct WAN link per vEdge, b) direct links to some or all WAN cloud or c) a combination of direct WAN link(s) and indirect WAN links(s) via another local vEdge device by using TLOC Extension. This chapter introduces the TLOC Extension. Figure 7-1 illustrates the example network topology where site-10 is a Branch-site and site-30 is a Hub-site. vEdge2 on site-30 is connected to the Public-Internet while vEdge3 is connected to the MPLS network. Between the vEdges, there are also dedicated links for TLOC Extension. Interface ge0/1 on vEdge2 is connected to vEdge3 interface g0/3. From the vEdge2 perspective, this link is a direct link to the MPLS network while vEdge3 stitch the interface to its interface ge0/1 which is connected to the MPLS network. The same solution is used for connecting vEdge3 to the Public-Internet, interface ge0/0 on vEdge3 connected to vEdge2 interface ge0/3 is just like a regular WAN link from the vEdge3 perspective while vEdge2 bind this interface to its interface ge0/0 connected to the Public Internet. This way both vEdge2 and vEdge3 can set up tunnels with vEdge1 by using both Public-Interface and MPLS networks.

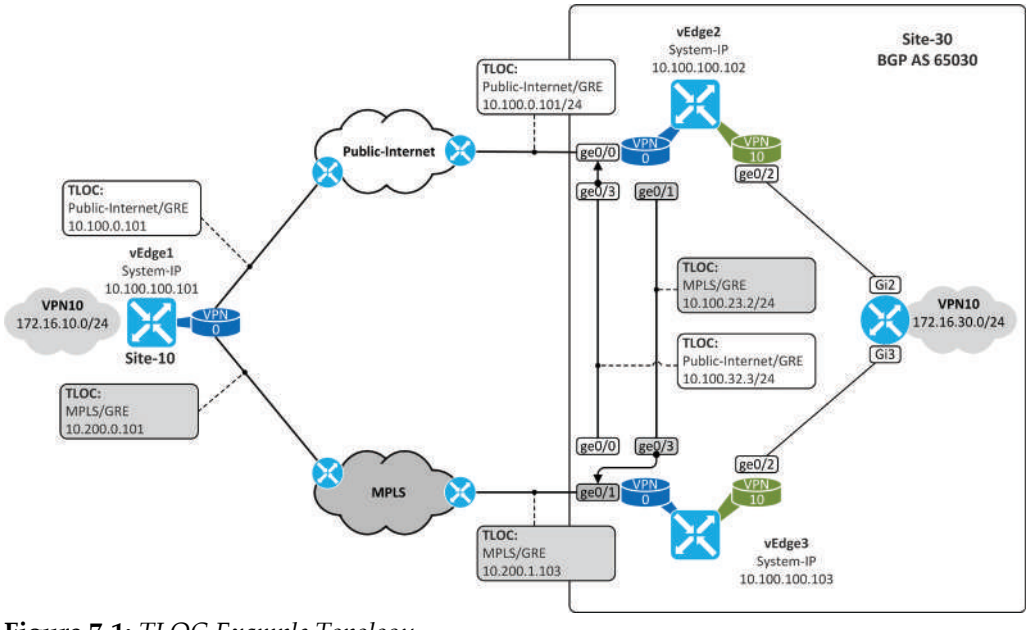


Figure 7-1: *TLOC Example Topology.*

Configuring TLOCs by using CLI

Figure 7-2 shows the CLI commands needed in order to create the TLOC Extension link. You can turn on the TLOC extension vEdge3 interface ge0/3 with interface-sub command *tloc-extensin ge0/1*. After the configuration, vEdge2 can use also the MPLS connection. vEdge2 uses its interface ge0/1 IP address 10.100.23.2 as a public-ip and private-ip in TLOC route advertisements about color MPLS and that is why the address 10.100.23.2 has to routable from the MPLS network perspective. We can also use NAT for relaxing the routing requirements. Then the public-ip will be the IP address 10.200.1.103 attached to vEdge3 interface ge0/1 while the private-ip is still 10.100.23.2.

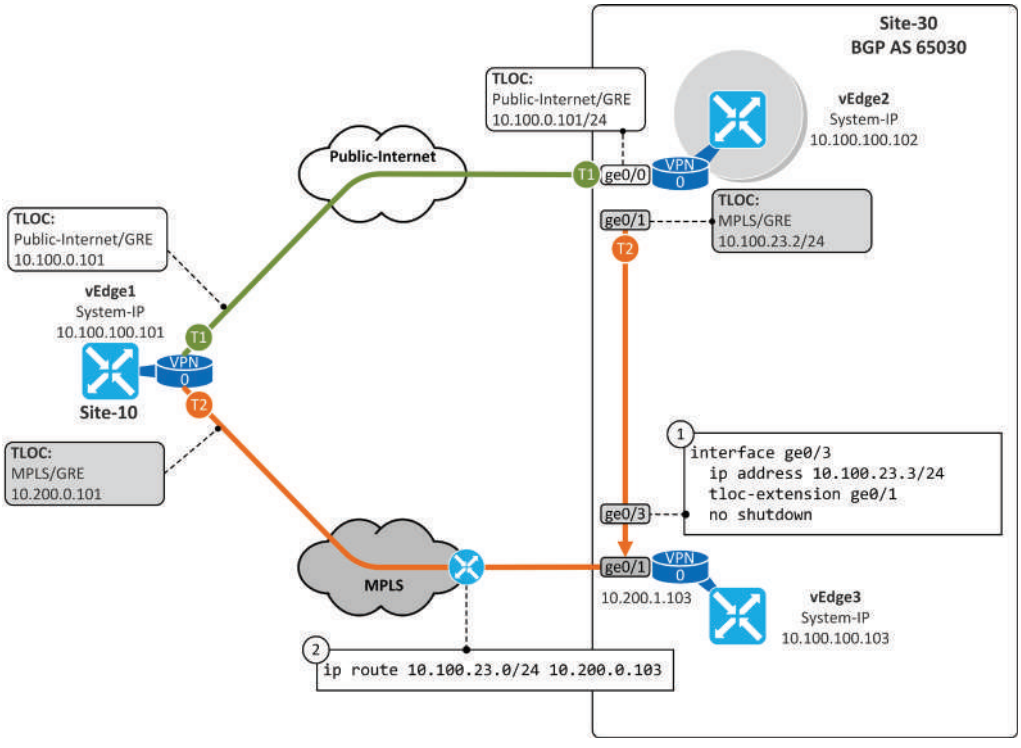


Figure 7-2: *TLOC Extension in vEdge3.*

After the TLOC Extenstion configuration, we can see that the Real-Time monitoring verifies that vEdge2 has BFD sessions with vEdge1 over the Public-Internet and the MPLS transport networks.

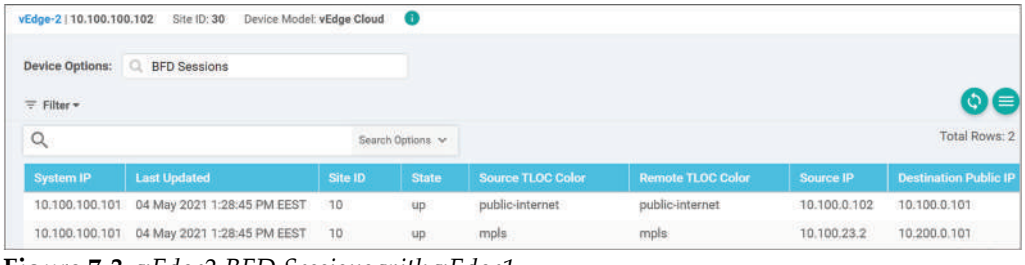


Figure 7-3: *vEdge2 BFD Sessions with vEdge1.*

Figure 7-4 verifies that GRE tunnels are also up over both transport networks.

Tunnel Endpoints	Interface Endpoints	Protocol	State
mpls	--	--	--
vEdge-2:mpls-vEdge-1:mpls	ge0/1-ge0/1	GRE	↑
public-internet	--	--	--
vEdge-2:public-internet-vEdge-1:public-internet	ge0/0-ge0/0	GRE	↑

Figure 7-4: *vEdge2 GRE Tunnels with vEdge1.*

Figure 7-5 shows the TLOC Extension configuration from the vEdge3 perspective. Its interface ge0/0 is nothing more than a normal WAN link while the TLOC Extension configuration is done under the interface ge0/3 on vEdge2. If NAT is not used, the IP address 10.100.32.3 has to be a routable, public address.

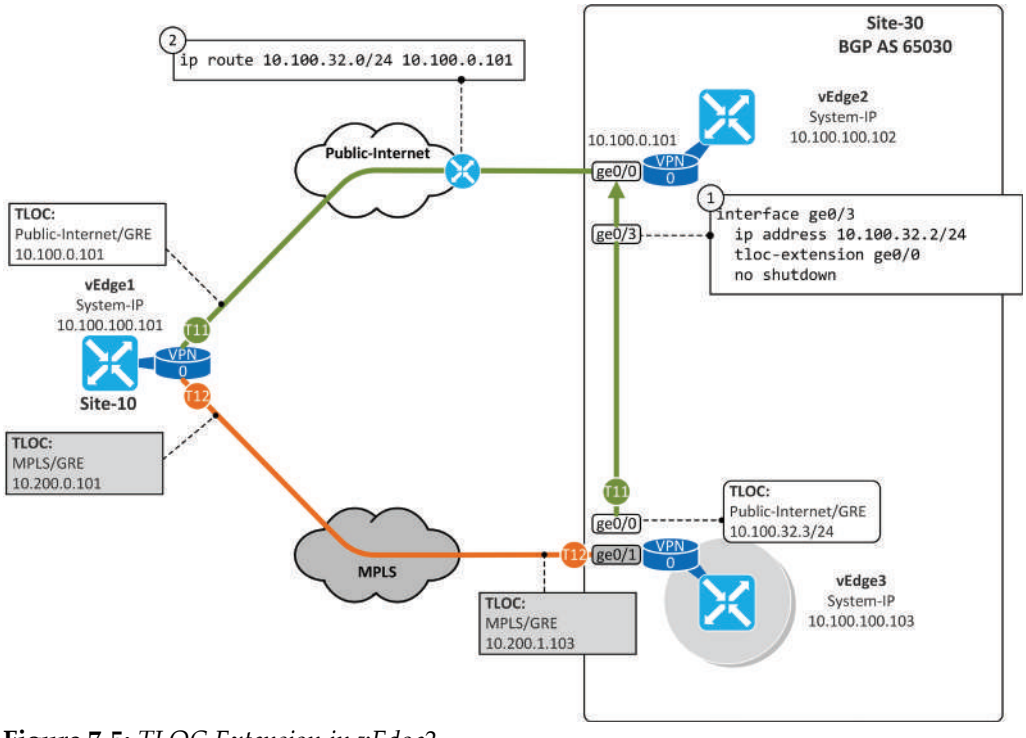


Figure 7-5: *TLOC Extension in vEdge2.*

Template Based TLOC Extension

Figure 7-6 shows the structure of the *Device Template* and *Feature Templates* attached to it. Note that the initial Feature Templates and Device Template were done in chapter 6. We now have a new Feature Template "TLOC-Ext_for_MPLS-G0/3" where in addition to IP address and no shut definition we are using TLOC extension to bind ge0/3 to interface ge0/1. This new Feature Template is then attached to Transport VPN 0 as *VPN Interface*.

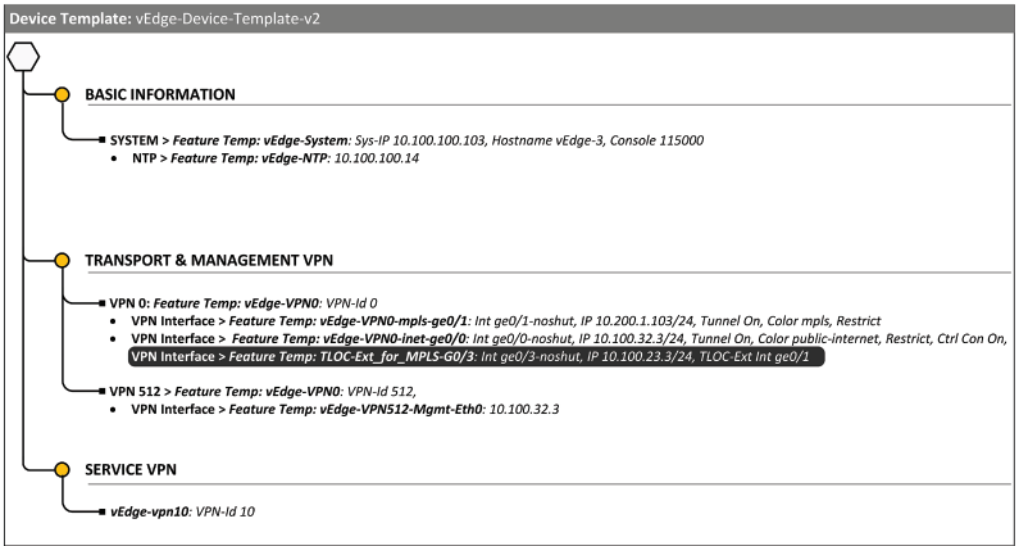


Figure 7-6: *Device Template and Feature Templates.*

The Real-Time monitoring of vEdge3 in figure 7-7 verifies that vEdge3 has BFD sessions with vEdge1 over the Public-Internet and the MPLS transport networks.

Figure 7-7: *vEdge3 BFD Sessions with vEdge1.*

Figure 7-8 verifies that GRE tunnels are also up over both transport networks.

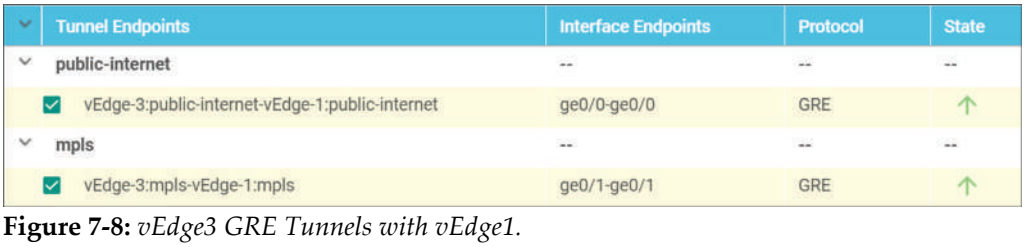


Figure 7-8: *vEdge3 GRE Tunnels with vEdge1.*

Figure 7-9 illustrates the complete GRE tunneling. vEdge1 have GRE tunnels with both vEdge2 and vEdge3 over both transport network.

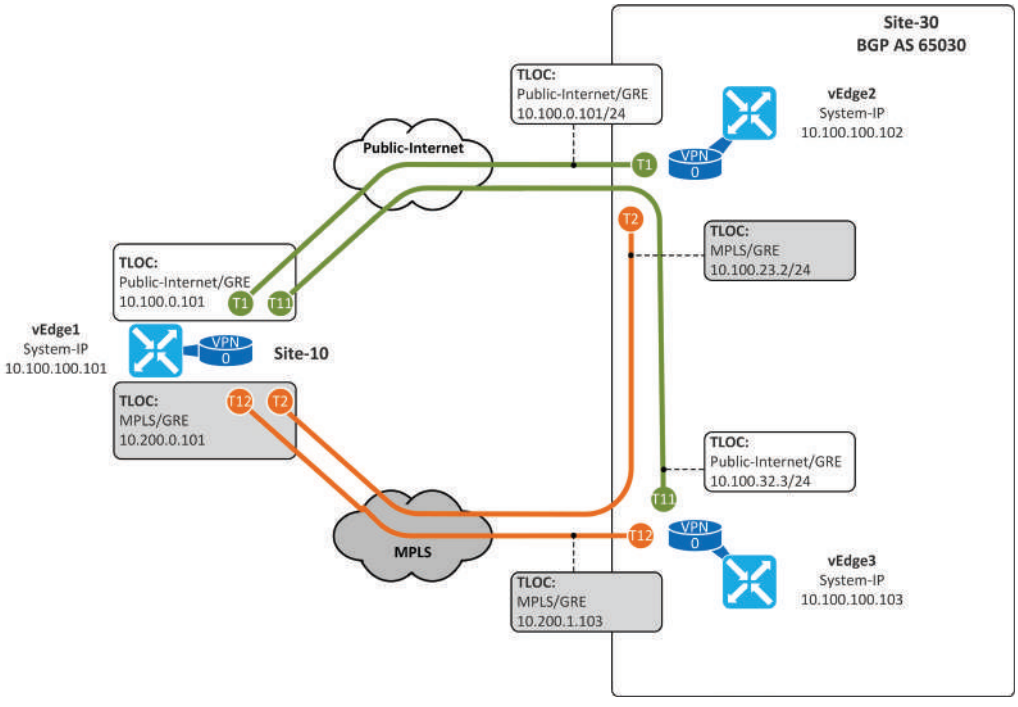


Figure 7-9: *Overall GRE Tunnels Between vEdge Devices.*

The Real-Time monitoring of vEdge1 in figure 7-10 verifies that vEdge1 has BFD sessions with vEdges in site 30 over the Public-Internet and the MPLS transport networks.

System IP	Last Updated	Site ID	State	Source TLOC Color	Remote TLOC Color	Source IP	Destination Public IP
10.100.100.102	04 May 2021 2:07:09 PM EEST	30	up	public-internet	public-internet	10.100.0.101	10.100.0.102
10.100.100.103	04 May 2021 2:07:09 PM EEST	30	up	public-internet	public-internet	10.100.0.101	10.100.32.3
10.100.100.102	04 May 2021 2:07:09 PM EEST	30	up	mpls	mpls	10.200.0.101	10.100.23.2
10.100.100.103	04 May 2021 2:07:09 PM EEST	30	up	mpls	mpls	10.200.0.101	10.200.1.103

Figure 7-10: *vEdge1 BFD Sessions with vEdges on Site-30.*

Benefits of TLOC Extension

The benefit of the TLOC extension is that you can reduce the connection to service provider networks without impacting the tunneling design. In figure 7-11 both vEdges on site-30 have established GRE tunnels over MPLS and Public-Internet transport networks. If the link between vEdge2 and LAN router is down, the traffic flows from site-30 to site-10 can still be load-balanced. However, TLOC Extension does not protect against vEdge device failure. In case we lose vEdge2, we will also lose the connection to the Public-Internet.

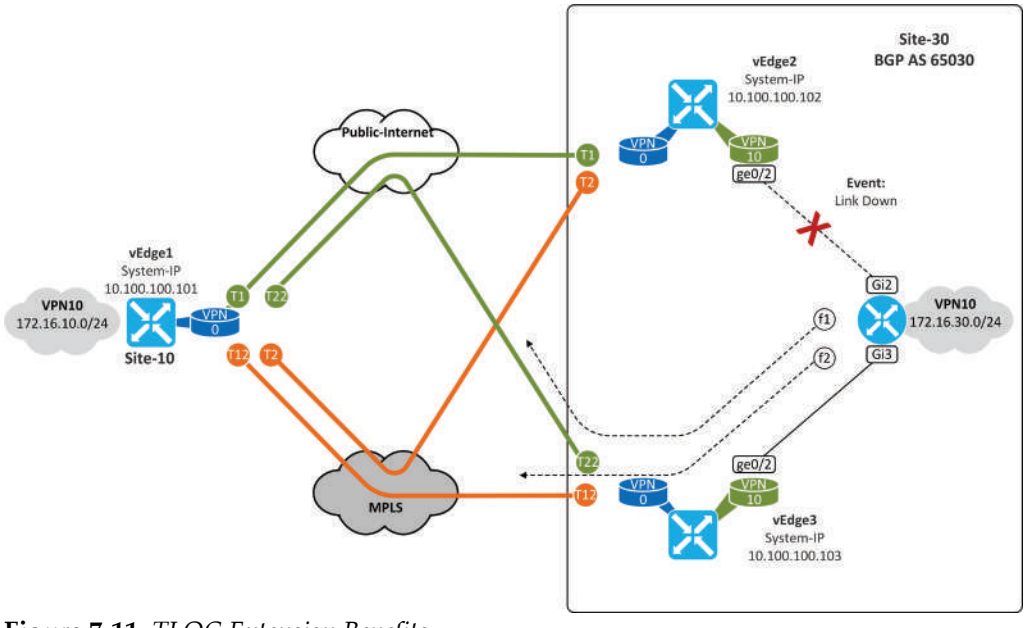


Figure 7-11: *TLOC Extension Benefits.*

Chapter 8: BGP Routing in LAN

Introduction

This chapter discusses the interaction of LAN Control-Plane and WAN Control-Plane on SD-WAN solution. You can use distance vector (EIGRP), link-state (IS-IS and OSPF), or path-vector (iBGP or eBGP) protocols in addition to static routing on the LAN side. Our focus is on iBGP based routing solutions. Figure 8-1 shows the topology and IP addressing scheme used in this chapter. Note that we are still using TLOC Extension implemented in chapter 7. For simplicity, iBGP peering is configured straight between the interface. This way we avoid having two control plane protocols in LAN. We are going to configure BGP settings on vEdge2 by using CLI while we are using feature templates with vEdge3. After BGP configuration, we are going to do some route optimization by using Centralized Policies in vSmart.

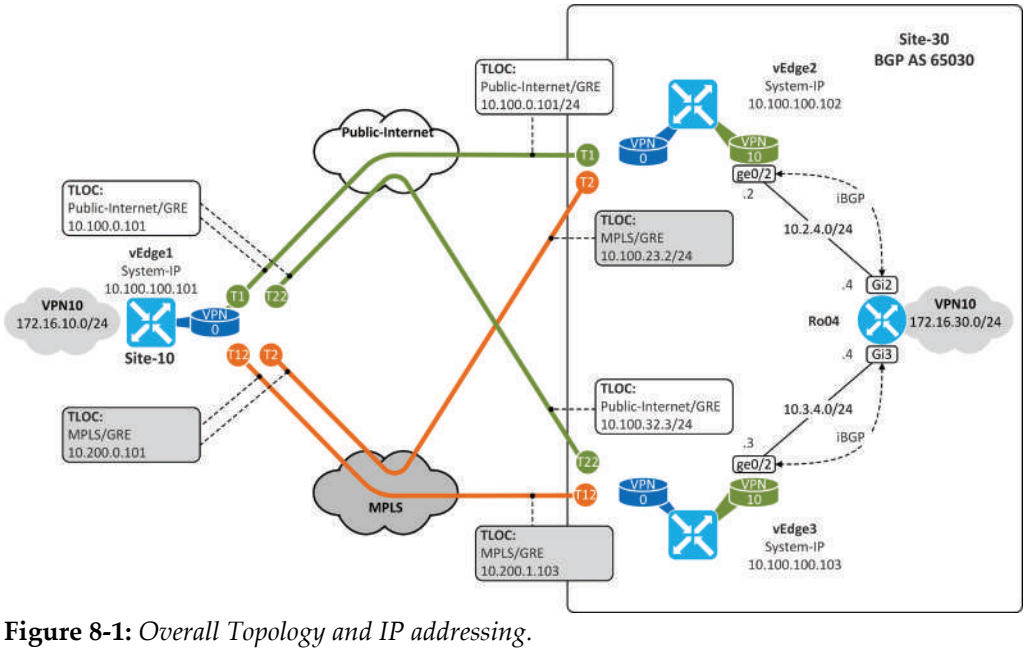


Figure 8-1: *Overall Topology and IP addressing.*

BGP Configuration Using CLI

Example 8-1 shows the BGP and OMP configuration steps related to the LAN side on vEdge2. First, we attach the interface ge0/2 to VPN 10 and then assign an IP address and enable the interface. Under the VPN 10 specific BGP section, we define the BGP neighbor IP address and the neighbor AS number. We also configure mutual redistribution from BGP to OMP and the other way around. Don't forget to commit changes.

```
vEdge-2(config)# vpn 10
vEdge-2(config-vpn-10)# interface ge0/2
vEdge-2(config-interface-ge0/2)# ip address 10.2.4.2/24
vEdge-2(config-interface-ge0/2)# no shutdown

vEdge-2(config-vpn-10)# router
vEdge-2(config-router)#  bgp 65030
vEdge-2(config-bgp-65030)# address-family ipv4-unicast
vEdge-2(config-address-family-ipv4-unicast)# redistribute omp
vEdge-2(config-address-family-ipv4-unicast)# neighbor 10.2.4.4
vEdge-2(config-neighbor-10.2.4.4)# no shutdown
vEdge-2(config-neighbor-10.2.4.4)# remote-as 65030
vEdge-2(config-neighbor-10.2.4.4)# address-family ipv4-unicast

vEdge-2(config)# omp
vEdge-2(config-omp)# advertise bgp
```

Example 8-1: *vEdge2 BGP and OMP CLI Configuration.*

Feature Template Based BGP Configuration

Figure 8-2 highlights the BGP and OMP feature templates related to the LAN side on vEdge3 from the complete feature template chart. As a first step, we need to set up LAN facing interface ge0/2 (feature template *vEdge3-Ro04*). Then we create another feature template for BGP where we define the AS Number, maximum-paths used for ECMP (Equal Cost Multi-Pathing), and the neighbor IP address and its AS number and OMP redistribution (feature template *BGP-Settings*). These two templates are then attached to VPN 10. The third new feature template is used for advertising BGP routes into OMP (feature template *OMP-Settings*). The configuration of the feature template was discussed in chapter 6. All feature templates are attached to device template v*Edge-Device-template-v2* which is then attached to vEdge3.

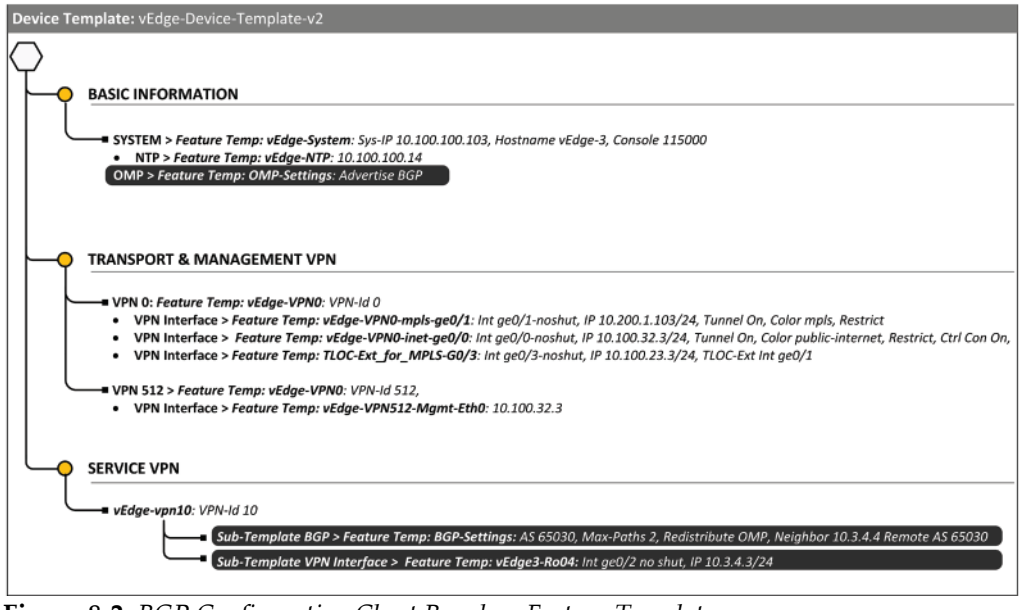


Figure 8-2: *BGP Configuration Chart Based on Feature Template.*

Example 8-2 shows the CLI configuration generated from the device template. The BGP and OMP configuration sections are highlighted.

```
vEdge-3# sh run
system
 host-name               vEdge-3
 system-ip               10.100.100.103
 site-id                 30
 admin-tech-on-failure
 no route-consistency-check
 sp-organization-name    nwkt
 organization-name       nwkt
 vbond 10.100.0.11
 aaa
 <snipped>
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
omp
 no shutdown
 graceful-restart
 advertise bgp
 advertise connected
 advertise static
!
security
```

```
 ipsec
  authentication-type sha1-hmac ah-sha1-hmac
 !
!
vpn 0
 interface ge0/0
  description "Internet Transport"
  ip address 10.100.32.3/24
  tunnel-interface
   encapsulation gre
   color public-internet restrict
   allow-service all
   <snipped>
  !
  no shutdown
 !
 interface ge0/1
  description "MPLS Transport"
  ip address 10.200.1.103/24
  tunnel-interface
   encapsulation gre
   color mpls restrict
   max-control-connections 0
   no control-connections
   allow-service all
   <snipped>
  !
  no shutdown
 !
 interface ge0/3
  description    TLOC-Ext-to-MPLS
  ip address 10.100.23.3/24
  tloc-extension ge0/1
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.32.2
 ip route 0.0.0.0/0 10.200.1.1
!
vpn 10
 router
  bgp 65030
   address-family ipv4-unicast
    maximum-paths paths 2
    redistribute omp
   !
   neighbor 10.3.4.4
    no shutdown
    remote-as 65030
    address-family ipv4-unicast
    !
   !
  !
 !
 interface ge0/2
```

```
  ip address 10.3.4.3/24
  no shutdown
 !
!
vpn 512
 interface eth0
  description Management
  ip dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
vEdge-3#
```

Example 8-2: *CLI Configuration Derived from the Device Template.*

Example 8-3 illustrates the LAN Ro04 BGP configuration.

```
Ro04(config)#router bgp 65030
Ro04(config-router)# bgp router-id 4.4.4.4
Ro04(config-router)# bgp log-neighbor-changes
Ro04(config-router)# neighbor 10.2.4.2 remote-as 65030
Ro04(config-router)# neighbor 10.3.4.3 remote-as 65030
Ro04(config-router)# !
Ro04(config-router)# address-family ipv4
Ro04(config-router-af)# network 172.16.30.0 mask 255.255.255.0
Ro04(config-router-af)# neighbor 10.2.4.2 activate
Ro04(config-router-af)# neighbor 10.3.4.3 activate
Ro04(config-router-af)# maximum-paths ibgp 2
Ro04(config-router-af)# exit-address-family
```

Example 8-3: *LAN Ro04 CLI configuration.*

Verification

Examples 8-4 and 8-5 verifies that both vEdges have established BGP adjacency with LAN Ro04.

```
vEdge-2# show bgp neighbor vpn 10
bgp bgp-neighbor vpn 10 10.2.4.4
 as          65030
 msg-rcvd    43
 msg-sent    39
 outQ        0
 uptime      0:00:35:25
 state       established
 last-uptime "Tue May  4 12:58:33 2021"
 address-family 0
  afi ipv4-unicast
```

Example 8-4: *BGP Peering Verificatin on vEdge2.*

```
vEdge-3# show bgp neighbor vpn 10
bgp bgp-neighbor vpn 10 10.3.4.4
 as          65030
 msg-rcvd    33
 msg-sent    31
 outQ        0
 uptime      0:00:27:31
 state       established
 last-uptime "Tue May  4 13:06:23 2021"
 address-family 0
  afi ipv4-unicast
```

Example 8-5: *BGP Peering Verificatin on vEdge3.*

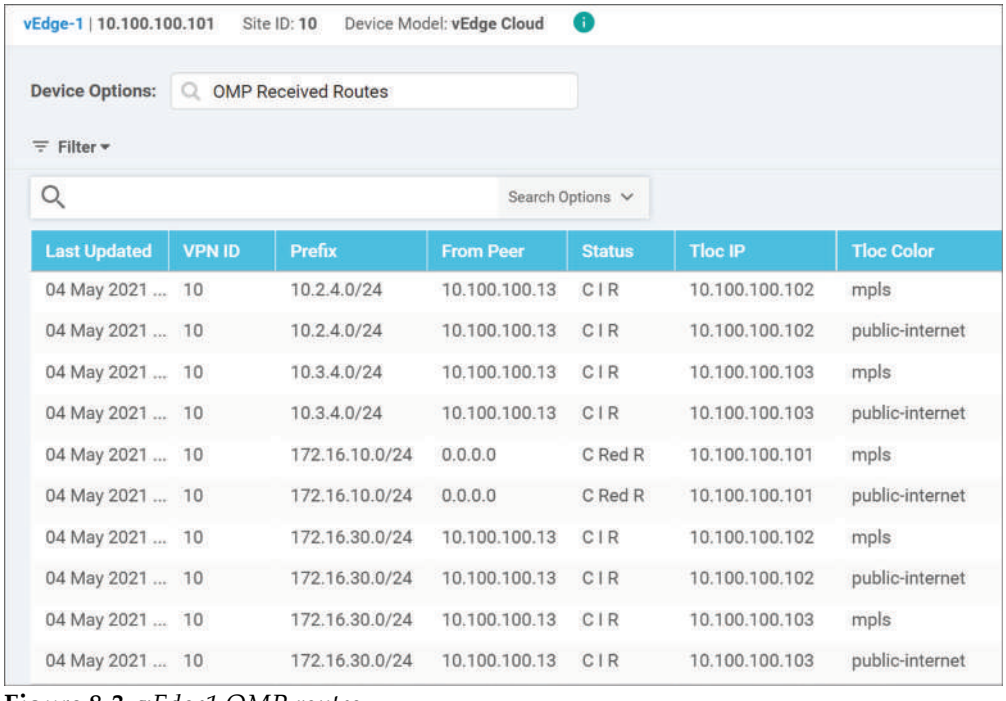
Example 8-6 shows that Ro04 has now learned the route to 172.16.10.0/24 via BGP from both vEdges and it is using multi-pathing meaning it can do flow-based traffic load-balancing.

```
Ro04#sh ip bgp 172.16.10.0
BGP routing table entry for 172.16.10.0/24, version 4
Paths: (2 available, best #2, table default)
Multipath: iBGP
  Not advertised to any peer
  Refresh Epoch 1
  Local
    10.3.4.3 from 10.3.4.3 (10.100.100.103)
      Origin incomplete, metric 1000, localpref 50, valid, internal, multipath(oldest)
      Extended Community: SoO:0:30
      rx pathid: 0, tx pathid: 0
  Refresh Epoch 1
  Local
    10.2.4.2 from 10.2.4.2 (10.100.100.102)
      Origin incomplete, metric 1000, localpref 50, valid, internal, multipath, best
      Extended Community: SoO:0:30
      rx pathid: 0, tx pathid: 0x0
```

Example 8-6: *BGP Peering Verificatin on Ro04.*

Route Optimization

Figure 8-3 shows the real-time OMP routes received by vEdge1. We can see that subnets 10.2.4.0/24 (the link between vEdge2 and Ro04) and 10.3.4.0/24 (the link between vEdge3 and Ro04) are received from vSmart. OMP advertises connected and static routes by default so this was expected behavior.



vEdge-1 | 10.100.100.101 Site ID: 10 Device Model: vEdge Cloud

Device Options: OMP Received Routes

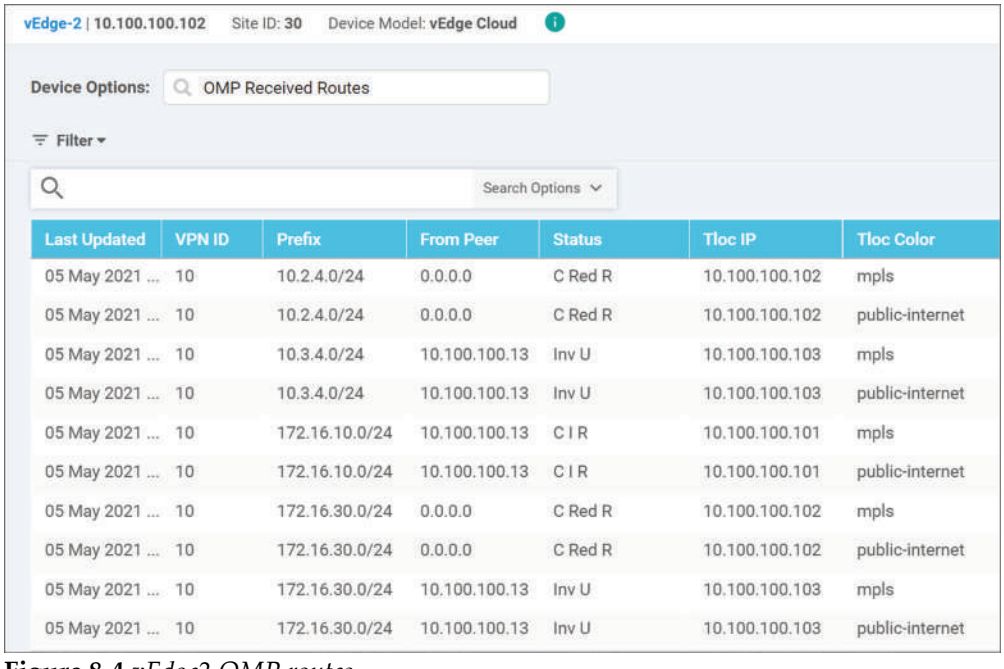
Filter

Search Options

Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
04 May 2021 ...	10	10.2.4.0/24	10.100.100.13	C I R	10.100.100.102	mpls
04 May 2021 ...	10	10.2.4.0/24	10.100.100.13	C I R	10.100.100.102	public-internet
04 May 2021 ...	10	10.3.4.0/24	10.100.100.13	C I R	10.100.100.103	mpls
04 May 2021 ...	10	10.3.4.0/24	10.100.100.13	C I R	10.100.100.103	public-internet
04 May 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	mpls
04 May 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	public-internet
04 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.102	mpls
04 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.102	public-internet
04 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.103	mpls
04 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.103	public-internet

Figure 8-3: *vEdge1 OMP routes.*

Figure 8-4 shows that the subnet 10.3.4.0/24 is also advertised to vEdge2 by vSmart.



vEdge-2 | 10.100.100.102 Site ID: 30 Device Model: vEdge Cloud

Device Options: OMP Received Routes

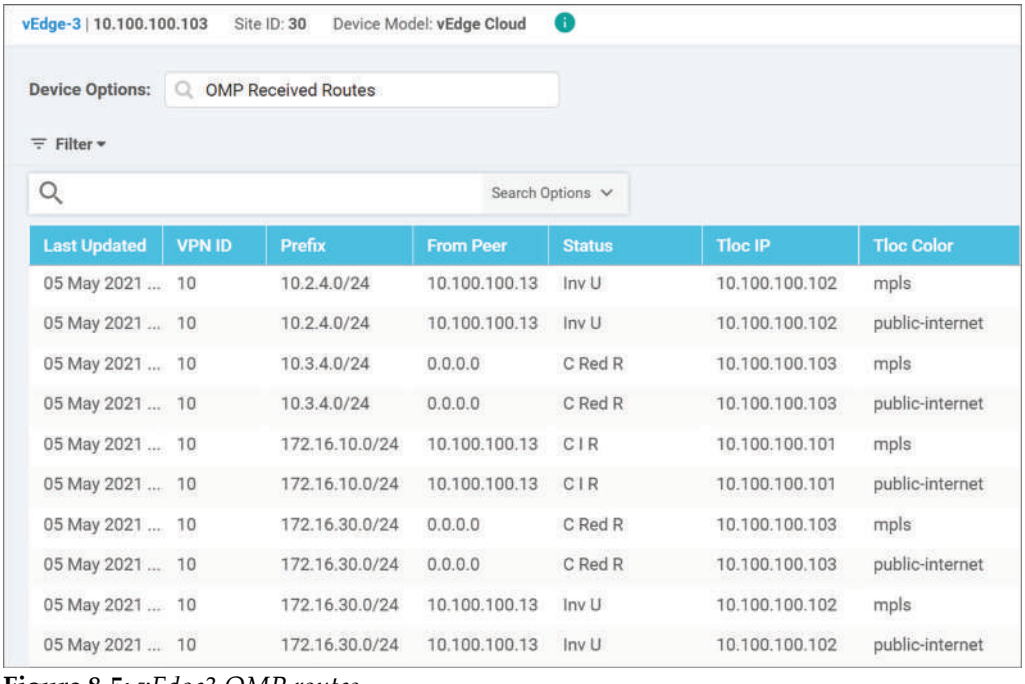
Filter

Search Options

Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
05 May 2021 ...	10	10.2.4.0/24	0.0.0.0	C Red R	10.100.100.102	mpls
05 May 2021 ...	10	10.2.4.0/24	0.0.0.0	C Red R	10.100.100.102	public-internet
05 May 2021 ...	10	10.3.4.0/24	10.100.100.13	Inv U	10.100.100.103	mpls
05 May 2021 ...	10	10.3.4.0/24	10.100.100.13	Inv U	10.100.100.103	public-internet
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	mpls
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	public-internet
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.102	mpls
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.102	public-internet
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.103	mpls
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.103	public-internet

Figure 8-4 *vEdge2 OMP routes.*

Figure 8-5 in turn shows that the subnet 10.2.4.0/24 is advertised to vEdge3 by vSmart.



Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
05 May 2021 ...	10	10.2.4.0/24	10.100.100.13	Inv U	10.100.100.102	mpls
05 May 2021 ...	10	10.2.4.0/24	10.100.100.13	Inv U	10.100.100.102	public-internet
05 May 2021 ...	10	10.3.4.0/24	0.0.0.0	C Red R	10.100.100.103	mpls
05 May 2021 ...	10	10.3.4.0/24	0.0.0.0	C Red R	10.100.100.103	public-internet
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	mpls
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	public-internet
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.103	mpls
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.103	public-internet
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.102	mpls
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.102	public-internet

Figure 8-5: *vEdge3 OMP routes.*

Subnets 10.2.4.0/24 and 10.3.4.0/24 will never be the data flow destination, so we do not need to advertise them. There are several ways and places to do the filtering. One way of doing that is to disable connected route advertisement from OMP. Example 8-7 shows the CLI configuration.

```
vEdge-2(config)# omp
vEdge-2(config-omp)# no advertise connected
vEdge-2(config-omp)# commit
```

Example 8-7: *Filtering Connected Routes from OMP.*

vEdge3 is in vManaged mode which means that changes have to be done by modifying templates. We have previously created a feature template *OMP-Settings* which we are using for advertising BGP routes into OMP. This feature template is then attached to basic settings for our device template. Basically what we need to do is to disable connected route advertisement. Figure 8-6 shows the complete template chart where the OMP feature template is highlighted.

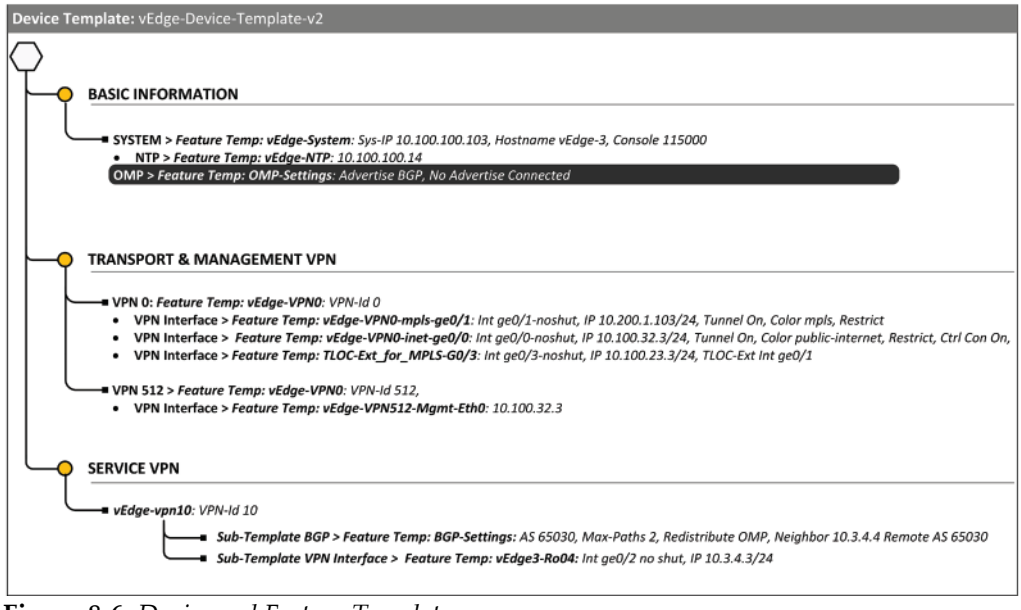
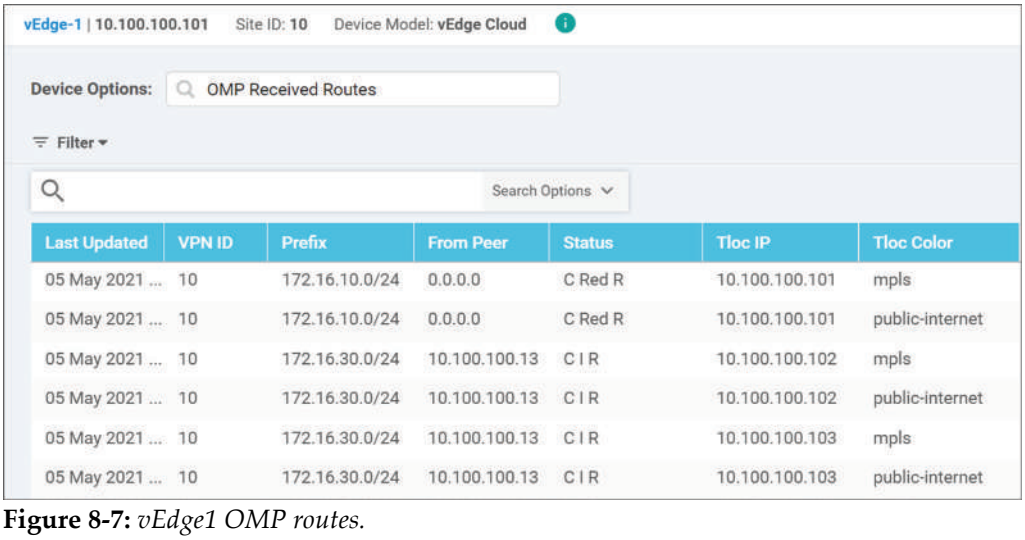


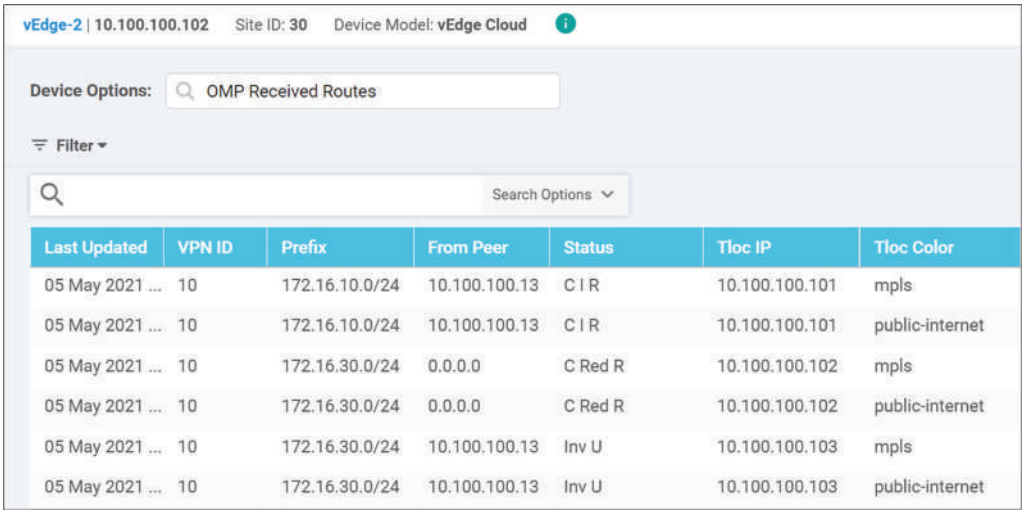
Figure 8-6: *Device and Feature Templates.*

After modification of the *OMP-Settings* feature template, we can see that vSmart doesn't advertise subnets 10.2.4.0/24 and 10.3.4.0/24 anymore (figure 8-7, 8-8, and 8-9).



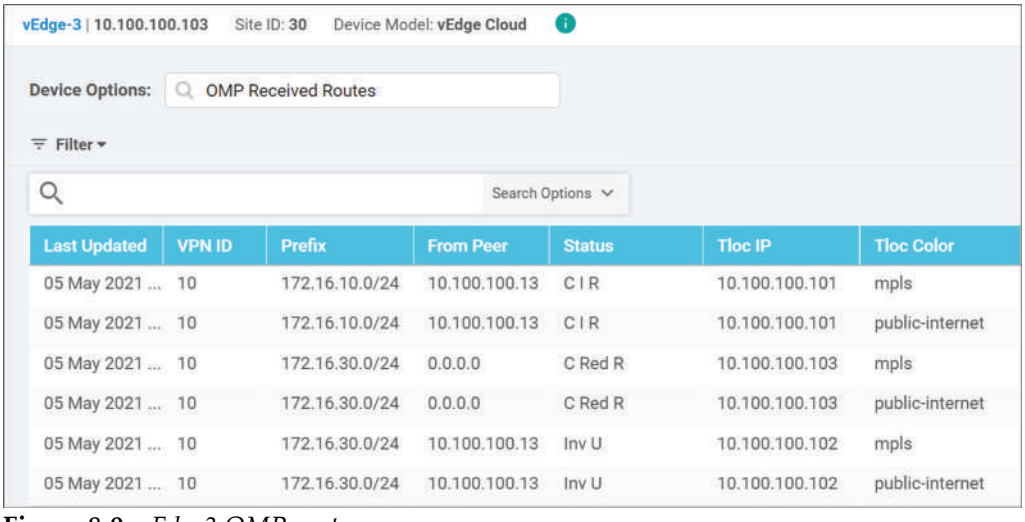
Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
05 May 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	mpls
05 May 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	public-internet
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.102	mpls
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.102	public-internet
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.103	mpls
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.103	public-internet

Figure 8-7: *vEdge1 OMP routes.*



Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	mpls
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	public-internet
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.102	mpls
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.102	public-internet
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.103	mpls
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.103	public-internet

Figure 8-8: *vEdge2 OMP routes.*



Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	mpls
05 May 2021 ...	10	172.16.10.0/24	10.100.100.13	C I R	10.100.100.101	public-internet
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.103	mpls
05 May 2021 ...	10	172.16.30.0/24	0.0.0.0	C Red R	10.100.100.103	public-internet
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.102	mpls
05 May 2021 ...	10	172.16.30.0/24	10.100.100.13	Inv U	10.100.100.102	public-internet

Figure 8-9: *vEdge3 OMP routes.*

If we take a look at figures 8-8 and 8-9 we can see that vSmart has sent OMP advertisements about subnet 172.16.30.0/24 to vEdge2 and vEdge3. Both devices have marked the route as Invalid with unreachable TLOC (U). vEdge2 in our case will never use vEdge3 as a next-hop for forwarding data flows to subnet 172.16.30.0/24, and the other way around. This due to fact that we have neither VLAN 10 Inter-vEdge nor GRE tunnel between vEdge2 and vEdge3. That is why we can prevent vSmart from advertising OMP routes originated by vEdge2 and vEdge3 back to site 30.

Figure 8-10 illustrates the OMP route propagation about subnet 172.16.30.0/24. Both vEdge2 and vEdge3 advertise the route to vSmart, which in turn forwards OMP advertisement to all vEdges.

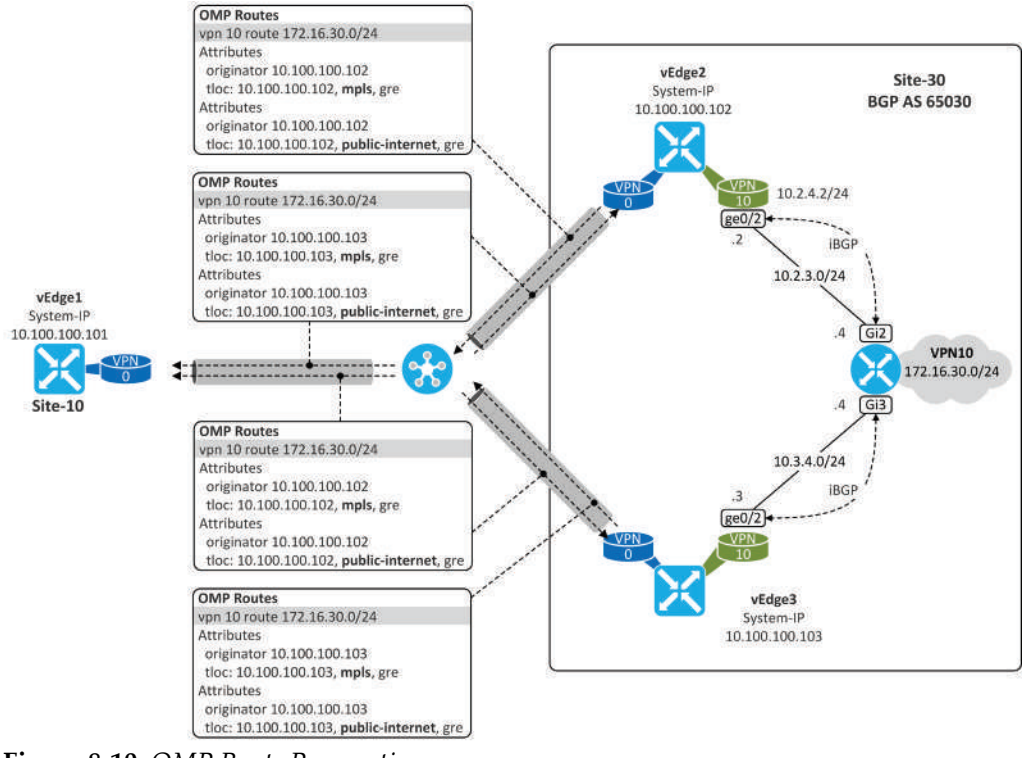


Figure 8-10: *OMP Route Propagation.*

Centralize Policy for OMP Route Filtering

Figure 8-11 recaps the SD-WAN policy model. OMP route policies are part of the *Control Policies* which in turn belongs to the Centralized Policy.

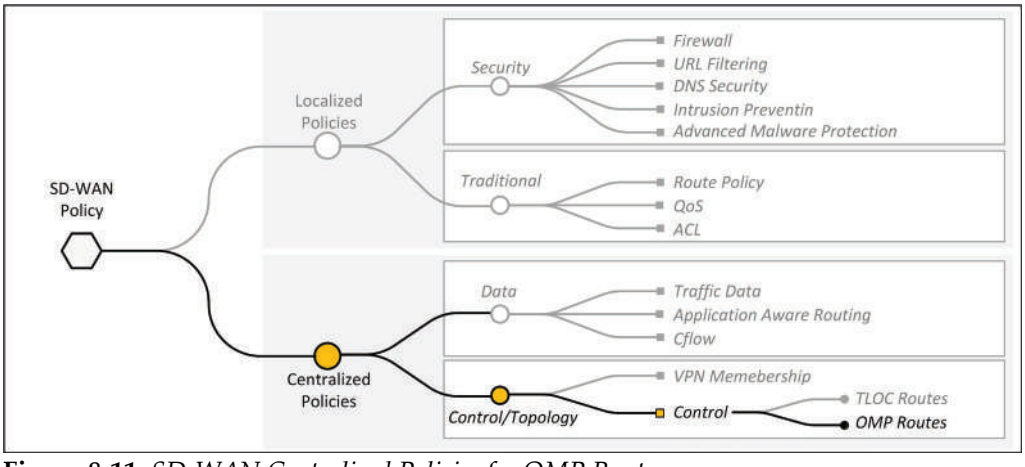


Figure 8-11: *SD-WAN Centralized Policies for OMP Routes.*

Figure 8-12 illustrates the components related to our OMP Route policy and its configuration steps. We start by defining the *Group of Interest*, which in our case are 1a) Site30, and 1b) subnet 172.16.30.0/24. Then we need to specify our intent, which is 2) advertise all other prefixes excluding prefix 172.16.30.0/24, 3) to Datacenter still advertising all prefixes to site10.

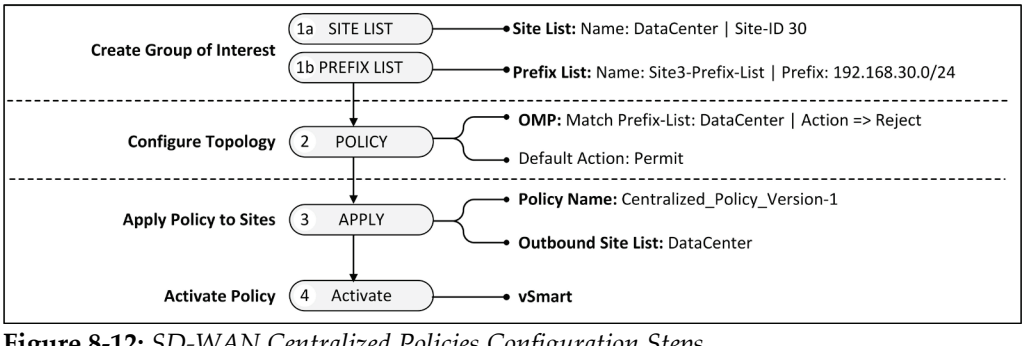


Figure 8-12: *SD-WAN Centralized Policies Configuration Steps.*

We have an existing Central Policy *Centralized_Policy_version1* which was used for creating Hub and Spoke topology in chapter 5. Instead of using it, we create a new Centralized Policy. Click the *Add Policy* button (figure 8-13).

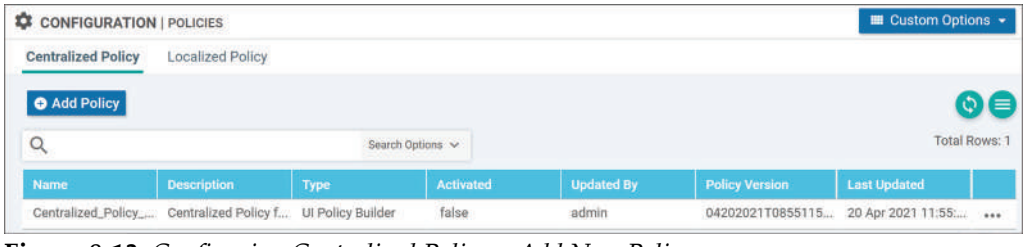


Figure 8-13: *Configuring Centralized Policy – Add New Polic.*

You can add a new prefix-list by clicking the *New Prefix List* selector. I have already added the prefix-list *Site30-Prefix-List* that defines subnet 12.16.30.0/24,

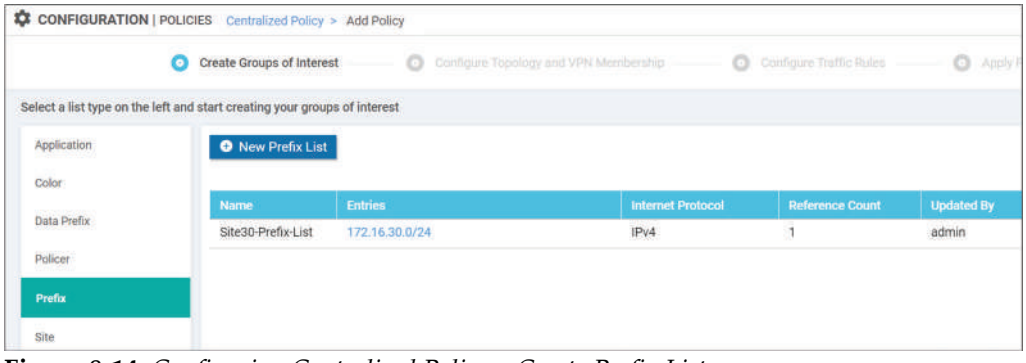


Figure 8-14: *Configuring Centralized Policy – Create Prefix List.*

I have also added a site-list DataCenter pointing to site 30.

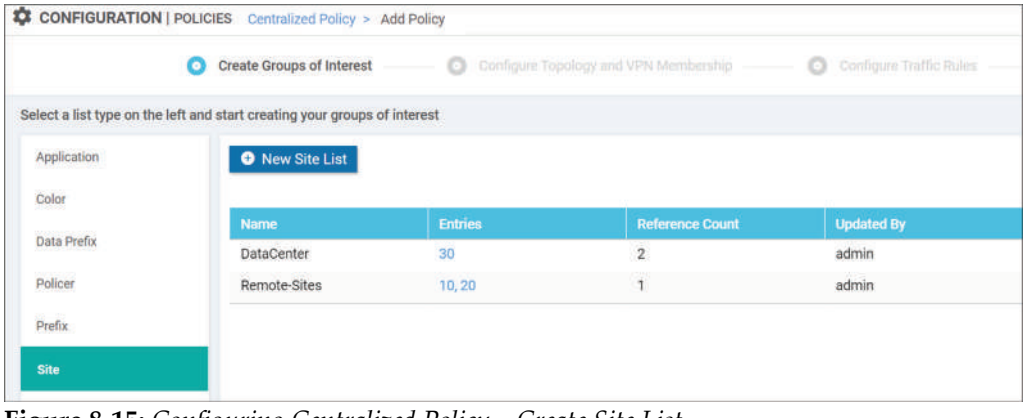


Figure 8-15: *Configuring Centralized Policy – Create Site List.*

After creating prefix-list and site-list, we can build a *Route Policy*. Select the *Custom Control (Route & TLOC)* option from the *Add Topology* drop-down menu.

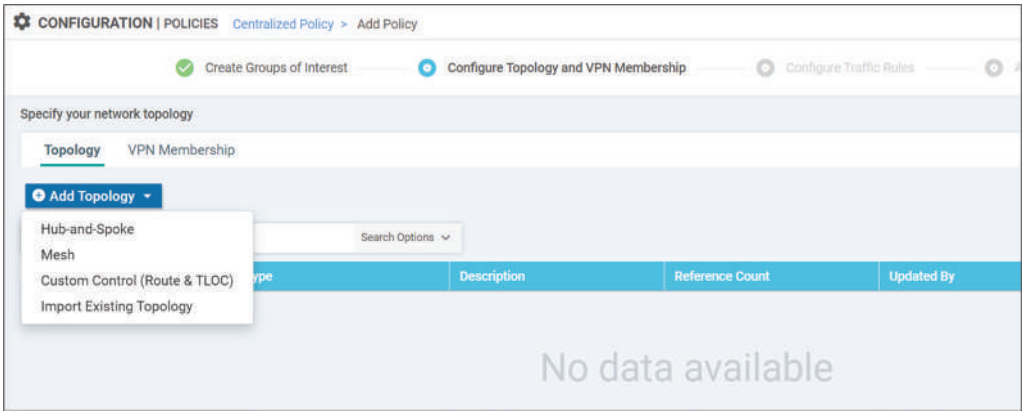


Figure 8-16: *Configuring Centralized Policy – Route Policy.*

Fill in *Name* and *Device* fields. Set the default action to *Accept* and click the *Save Match And Action* button.

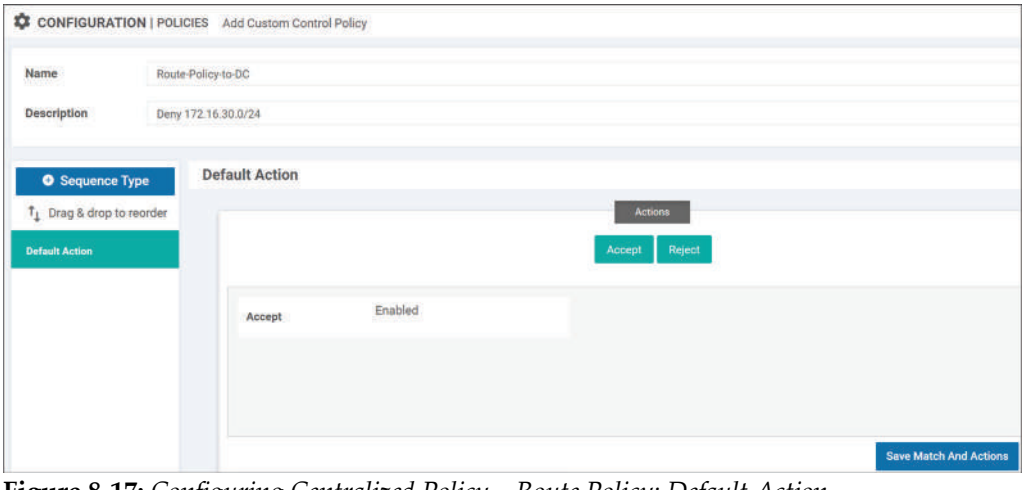


Figure 8-17: *Configuring Centralized Policy – Route Policy: Default Action.*

Click the *Sequence Type* selector and choose *Route* from the *Add Control Policy* pop-up window.

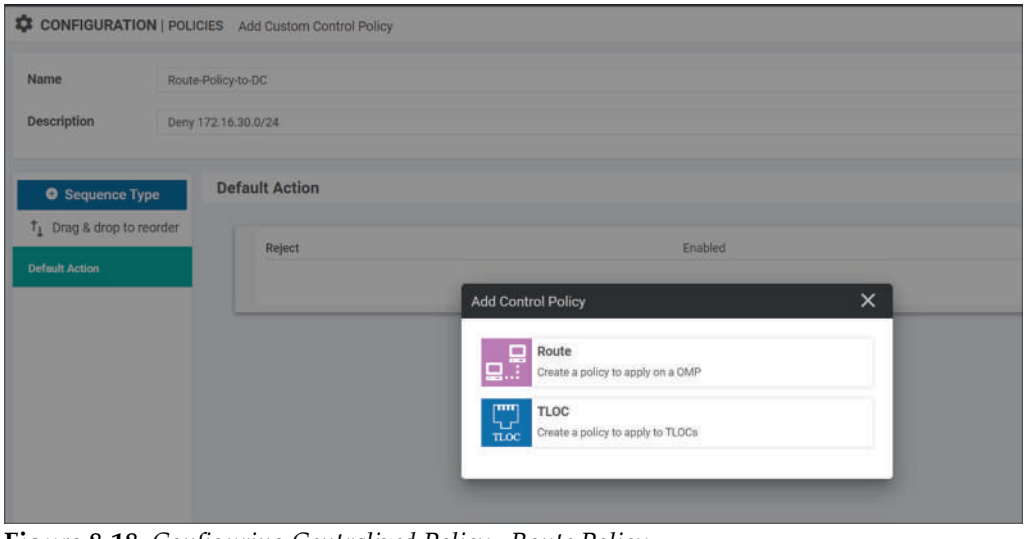


Figure 8-18: *Configuring Centralized Policy –Route Policy.*

Click the *Actions* selector and select the *Reject* option. Then select the *Match* selector and choose the *Site30-Prefix-List* from the *Prefix-List* drop-down menu on *Match Conditions* section. When done, click the *Save Match And Action button*.

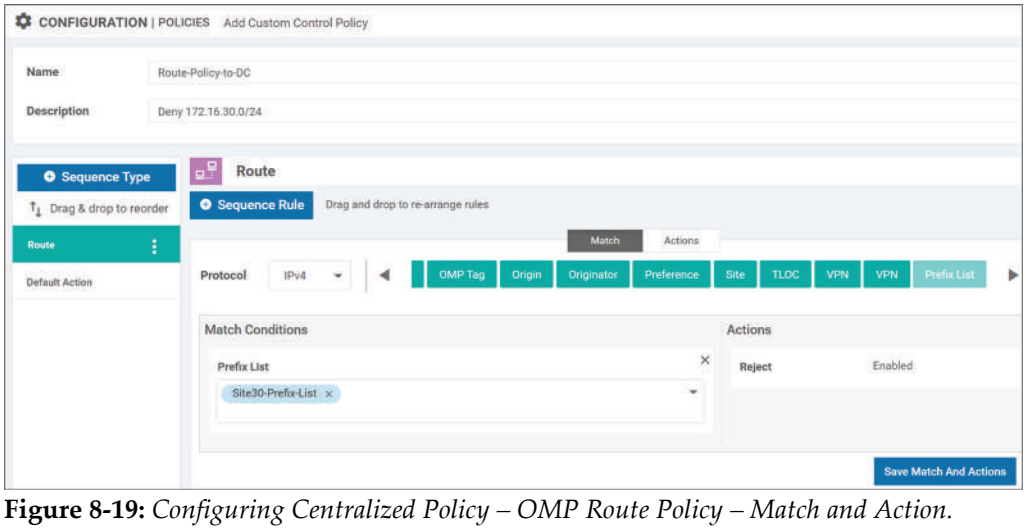


Figure 8-19: *Configuring Centralized Policy – OMP Route Policy – Match and Action.*

When done, save the Control Policy by clicking the *Save Control Policy* button.

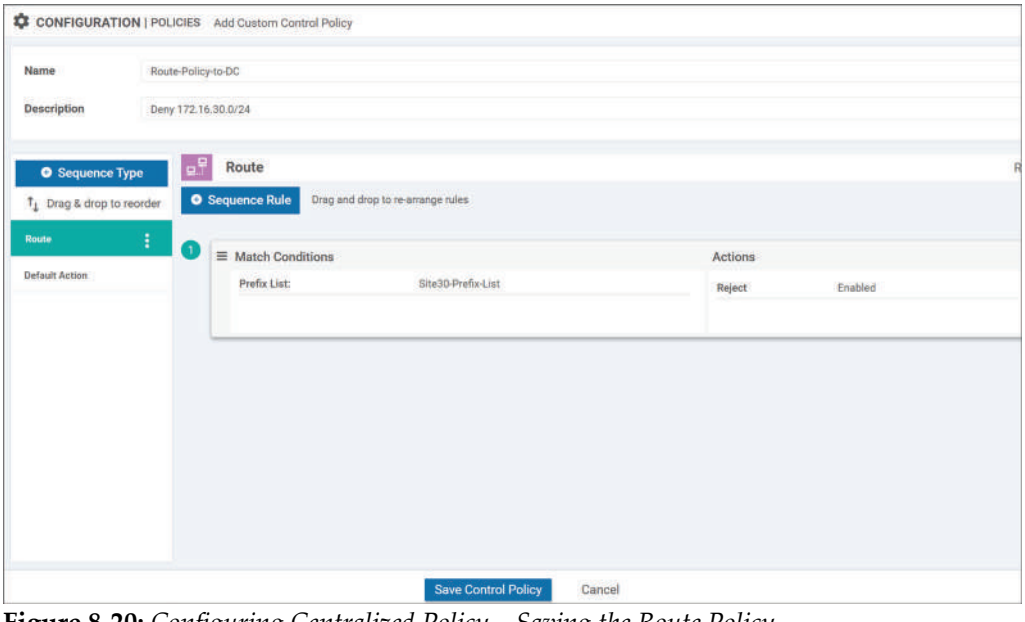


Figure 8-20: *Configuring Centralized Policy – Saving the Route Policy.*

Click the *Next* button.

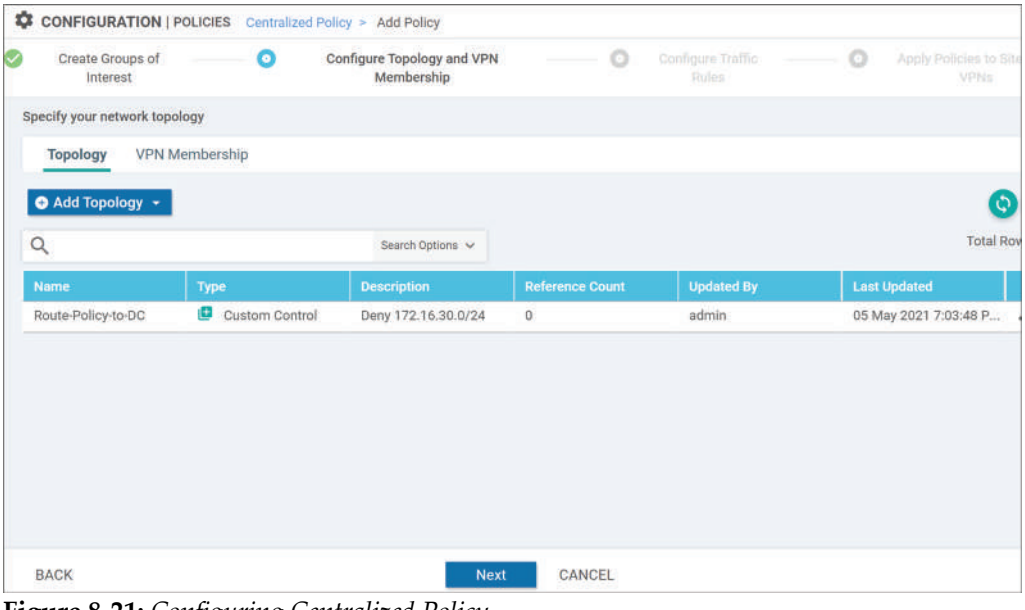


Figure 8-21: *Configuring Centralized Policy.*

We are not going to configure any Application-Aware Routing policy so click the *Next* button again.

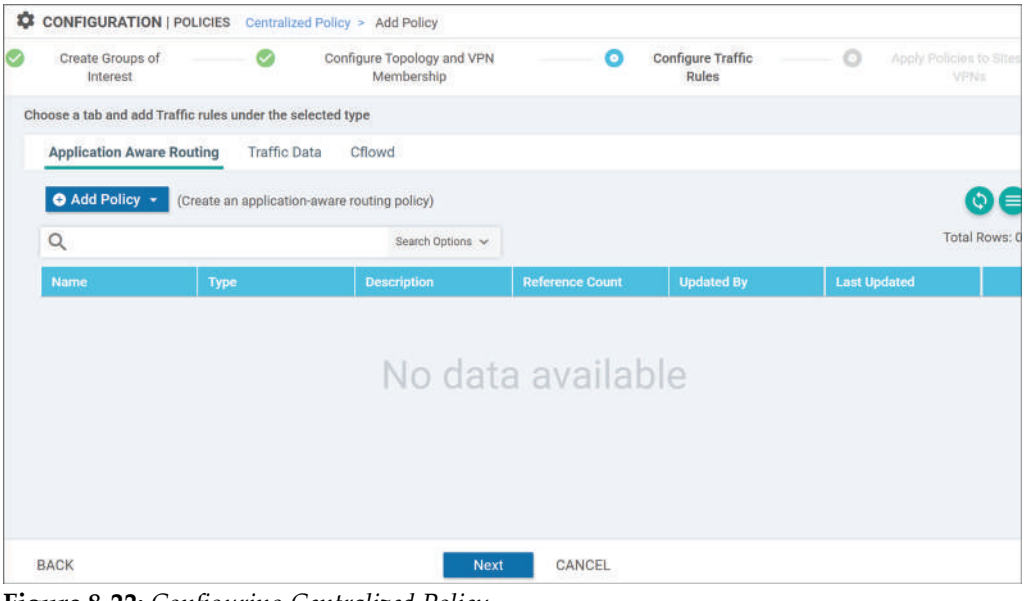


Figure 8-22: *Configuring Centralized Policy.*

The last window *Add policies to site and VPN* is where we bind our ***Control Policy*** *Route-Policy-to-DC* to ***Centralized Policy*** *Centralized-Policy-v1* (the main policy). Select the site-list *DataCenter* (site 30) from the *Outbound Site List* section (policy direction) and click the *Add* button.

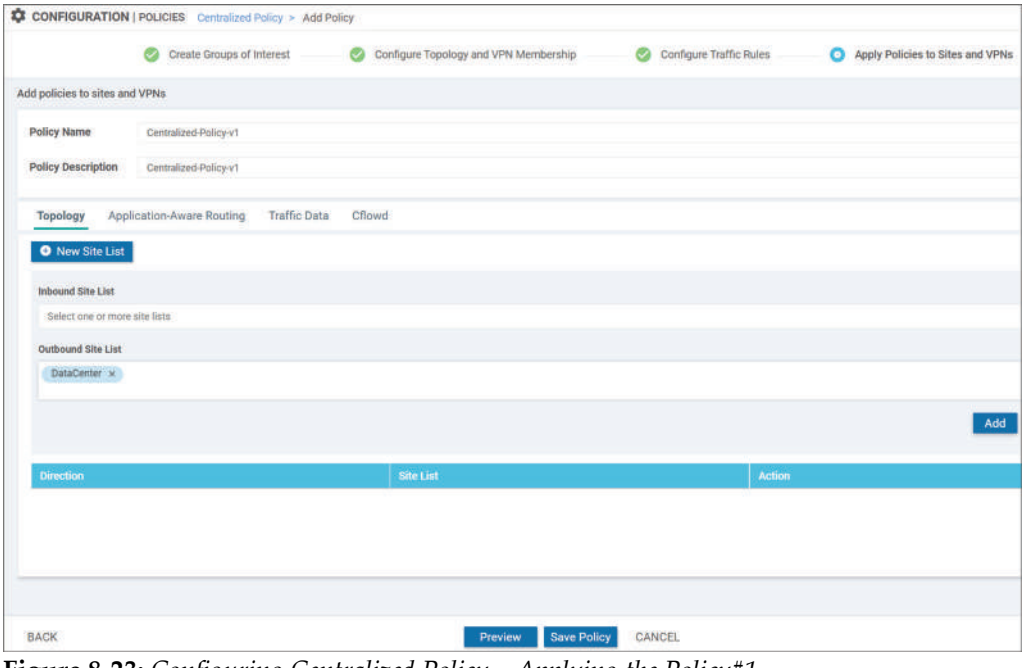


Figure 8-23: *Configuring Centralized Policy – Applying the Policy#1.*

You can preview the CLI configuration generated by our Centralized Policy by clicking the *Preview* button. You can save the policy in this window or the *Preview* window by clicking the *Save Policy* button.

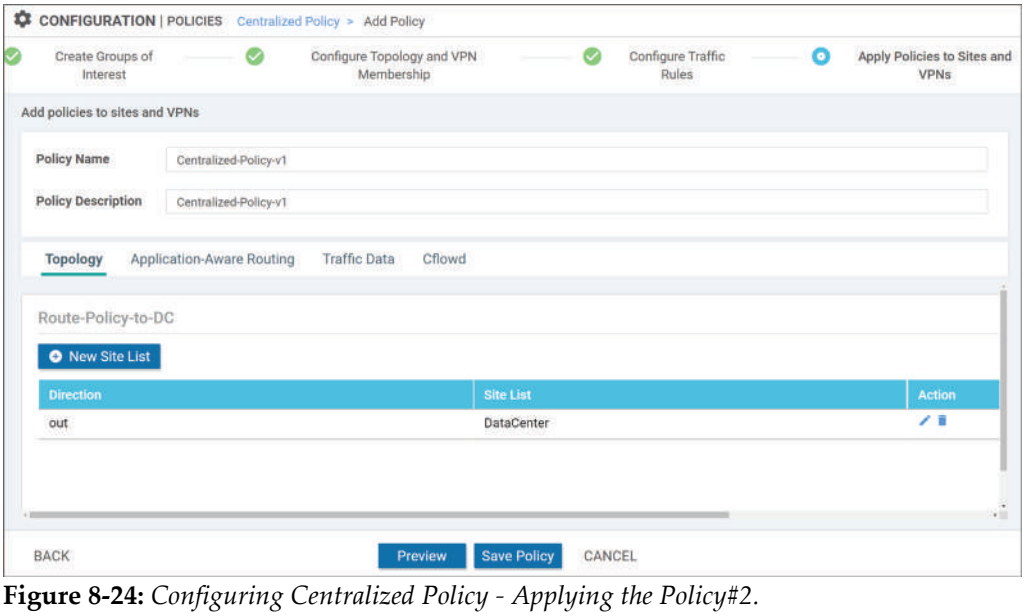


Figure 8-24: *Configuring Centralized Policy - Applying the Policy#2.*

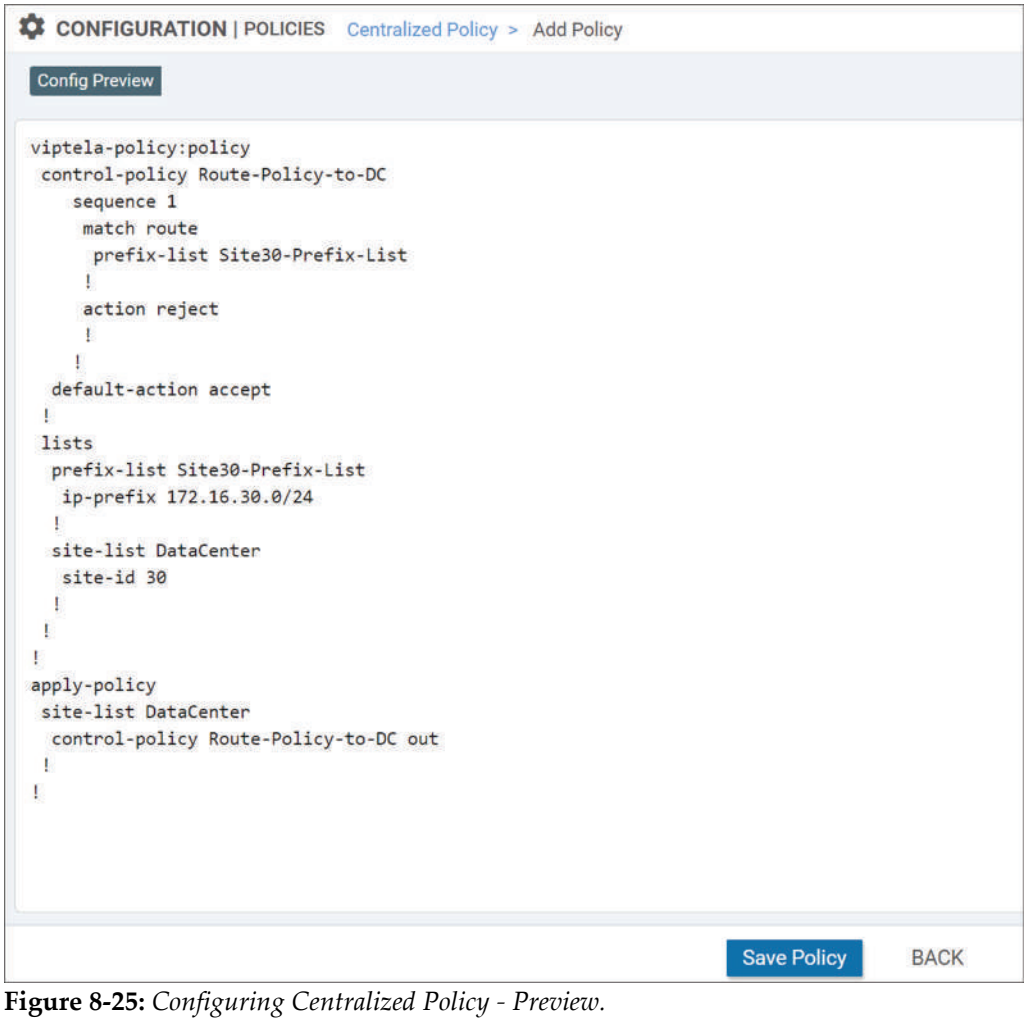


Figure 8-25: *Configuring Centralized Policy - Preview.*

As the last step, we need to deactivate the current Centralized Policy and activate the new one (figure 8-26). This can be done by selecting the Deactivate/Activate from the policy-related options menu […].

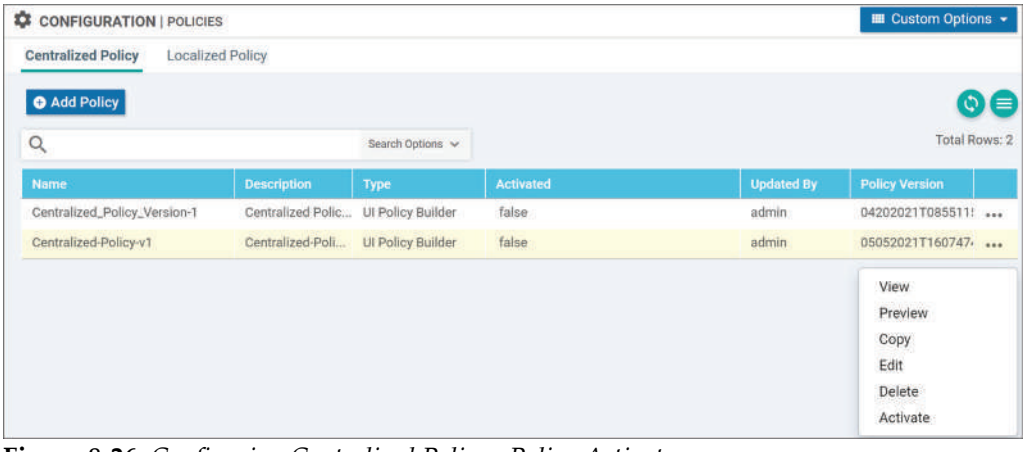


Figure 8-26: *Configuring Centralized Policy –Policy Activate.*

You need to confirm the activation process.

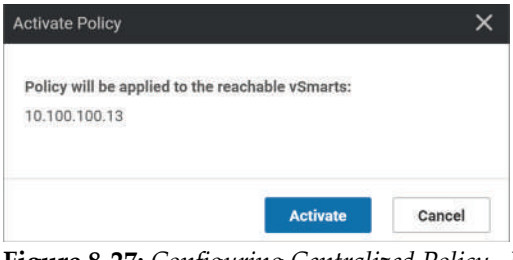


Figure 8-27: *Configuring Centralized Policy –Policy Activate.*

Figure 8-28 shows the progress of the process.

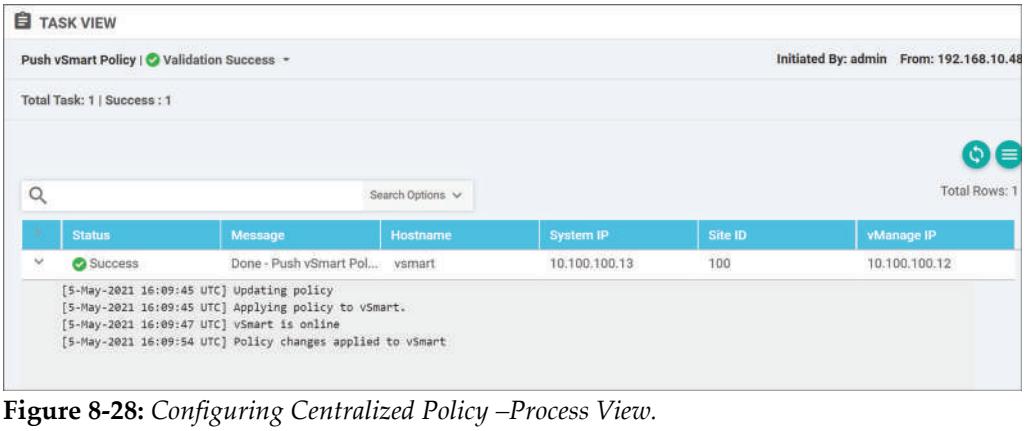


Figure 8-28: *Configuring Centralized Policy –Process View.*

Figure 8-29 verifies that vEdge1 still gets the OMP route about 172.16.30.0/24 while neither vEdge2 (figure 8-30) nor vEdge3 (figure 8-31) doesn't.

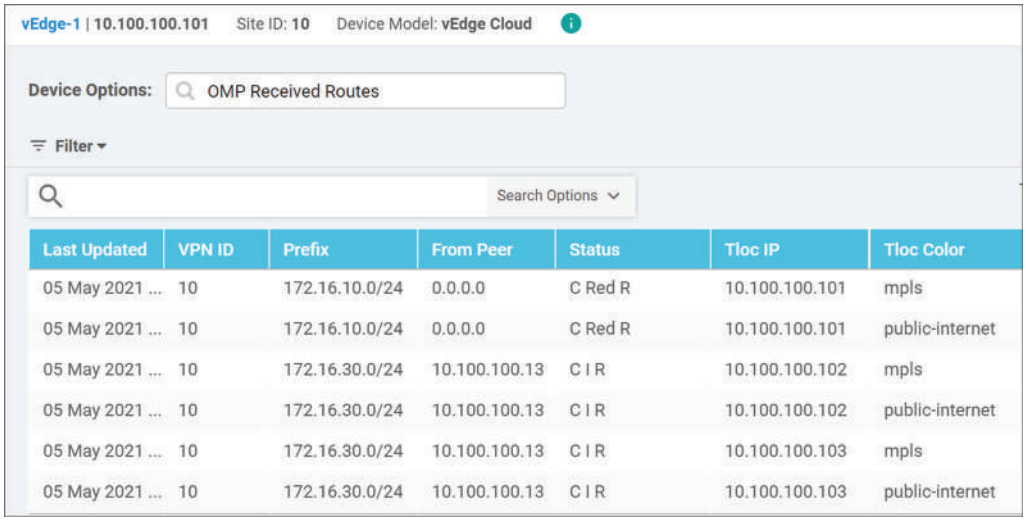


Figure 8-29: *OMP Routes – vEdge1.*

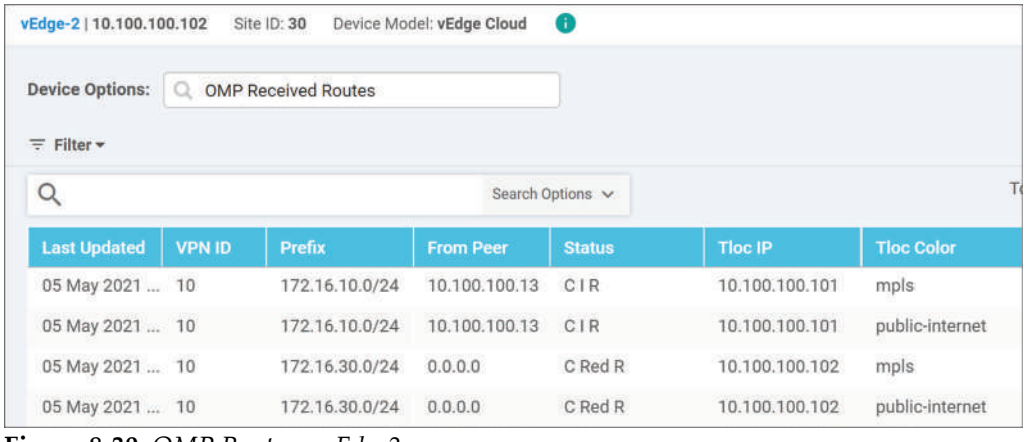


Figure 8-30: *OMP Routes – vEdge2.*

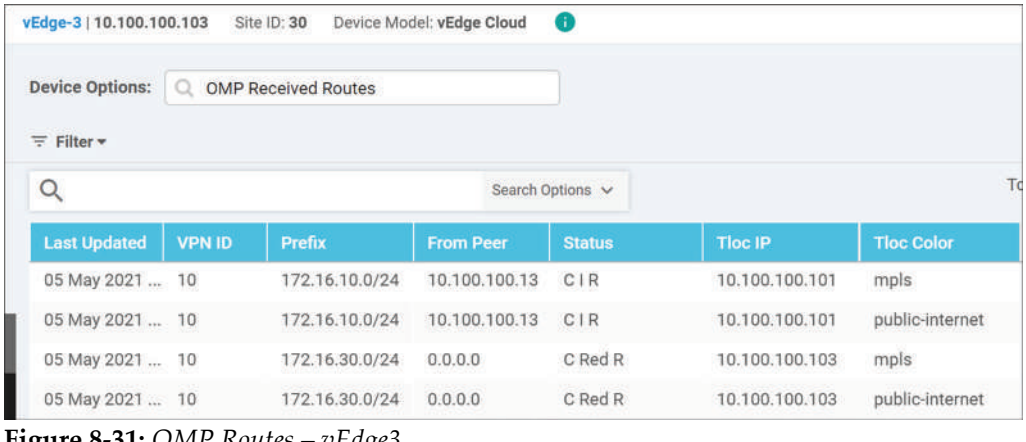


Figure 8-31: *OMP Routes – vEdge3.*

Chapter 9: Traffic Engineering

Introduction

vEdge1 in site 10 has established four tunnels to site 30. All four tunnels are used for ECMP by default. This chapter explains how we can prefer some of these tunnels while the rest are only used when preferred tunnels fail. The first section explains the TLOC specific precedence settings by using the Centralized Policies. The second section introduces how we can set tunnel-specific precedence by using Feature Templates. The last section explains how we can use a set of tunnels for forwarding packets based on the destination prefix.

Centralized Policy – Precedence

Figure 9-1 illustrates the default traffic forwarding model where all tunnels are used for flow-based load-balancing. vEdge1 has received TLOC routes from vEdge2 and vEdge3 where they describe their transport network (mpls, public-internet), tunneling mode (GRE), and public IP address used in the outer IP header (used as a next-hop in OMP routing updates). vEdge1 has also received VPN10 specific OMP routing update about network 172.16.30.0/24 from both vEdges on site 30.

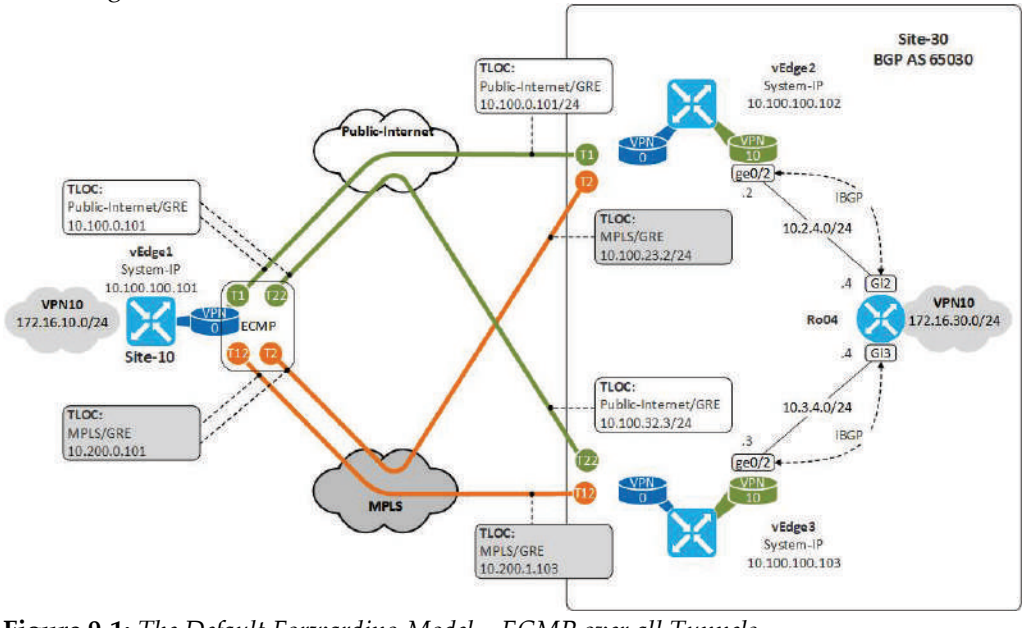


Figure 9-1: *The Default Forwarding Model – ECMP over all Tunnels.*

Figure 9-2 shows there are four entries about the subnet 172.16.30.0/24 installed in vEdge1 VPN10 Routing Information Base (RIB).

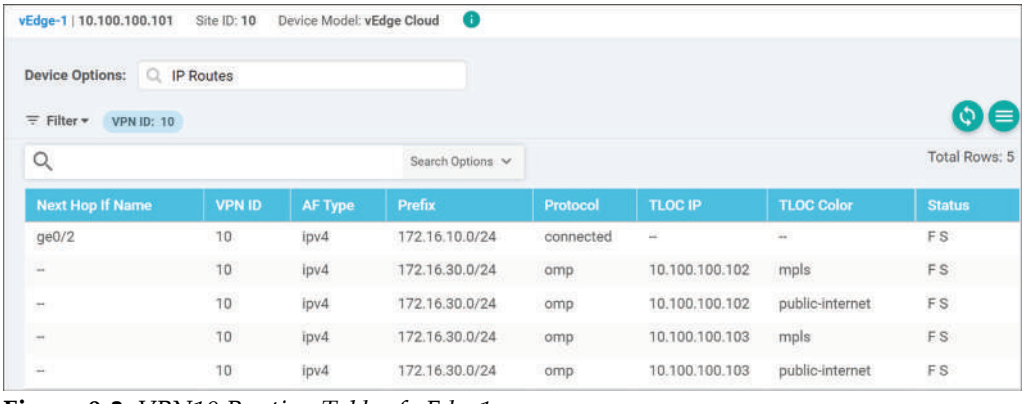


Figure 9-2: *VPN10 Routing Table of vEdge1.*

Figure 9-3 shows the simulated youtube application data flow from the vEdge1 VPN10 LAN interface 172.16.10.1 to the destination 172.16.30.1. We can see that four tunnels are used for traffic forwarding.

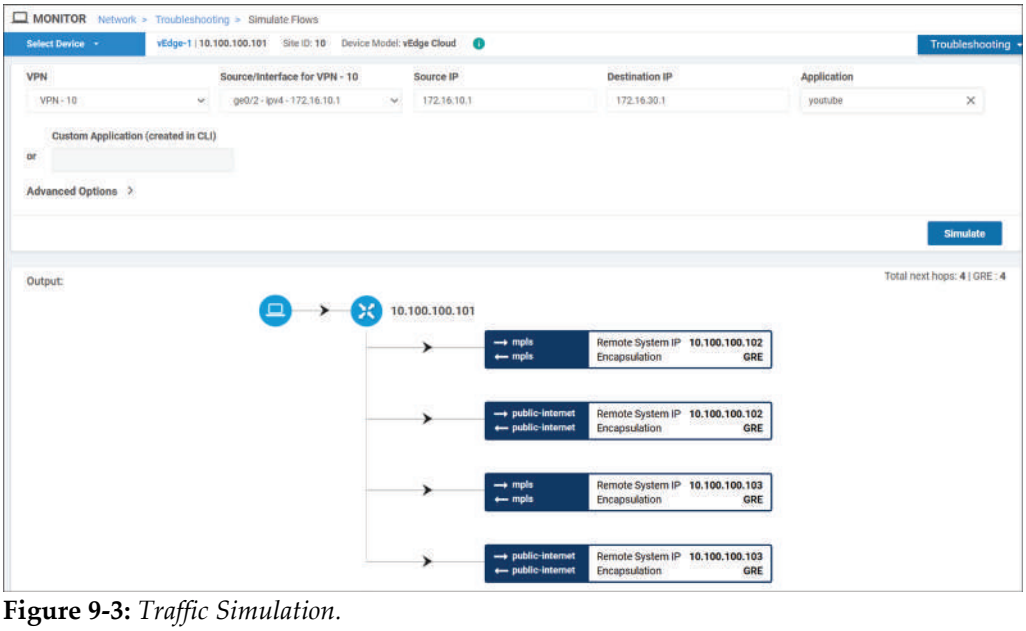


Figure 9-3: *Traffic Simulation.*

In the *Centralized Policy* based precedence model, vSmart does all the magic. It adds precedence 200 to TLOC updates from vEdge2 and precedence 100 to TLOC updates from vEdge3 before forwarding TLOC routes to vEdge1.

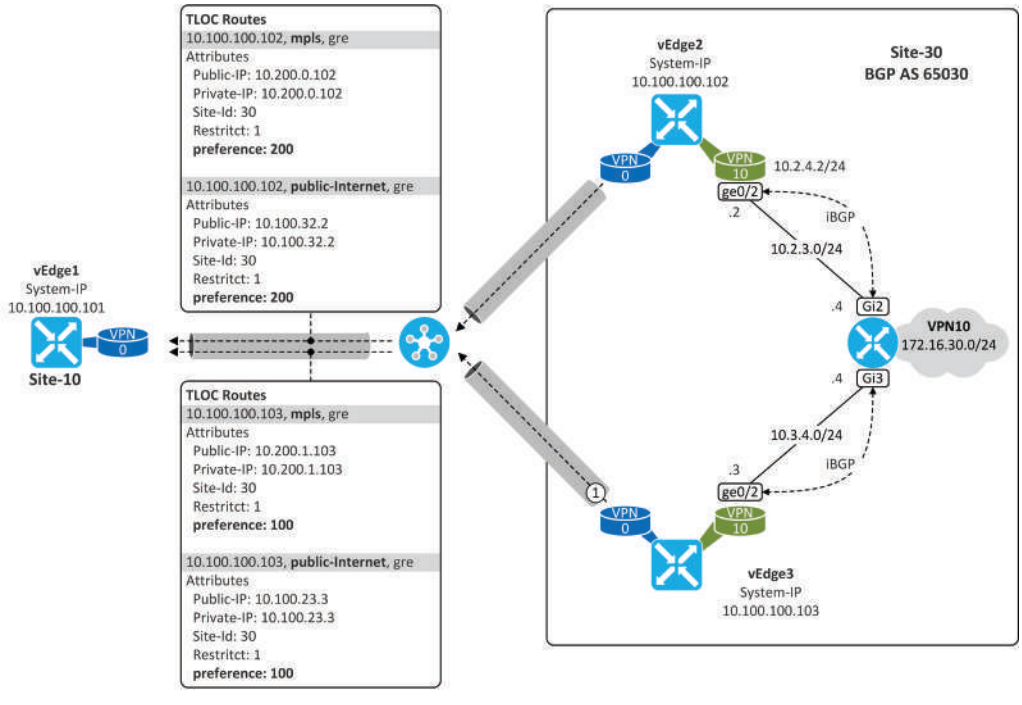


Figure 9-4: *Centralized Policy model.*

When vEdge1 receives TLOC routes from vSmart, it only uses the public-ip address carried within TLOC updates with the highest precedence value as a valid next-hop for destination subnets received within OMP updates. Figure 9-5 shows that vEdge1 only uses tunnels to vEdge2 because both TLOC updates originated by vEdge2 have precedence 200 while TLOC updates originated by vEdge3 have precedence value 100. Note that the precedence value is not shown if you take a look at the vEdge2 OMP routes because the precedence is set as an OMP attribute in the TLOC route.

Note that we are only going to show how the do TLOC route precedence-based traffic engineering for the traffic from site10 to site30. We are using BGP multi-path in Ro04, so it will do flow-based load-balancing between vEdge2 and vEdge3 when forwarding packets to site10. In order to prefer vEdge2, we need to disable BGP multi-pathing and then use the BGP-specific egress policy such as Weight or Local-Preference. In our example, the simple way to prefer vEdge2 as the next-hop is using weight on Ro04.

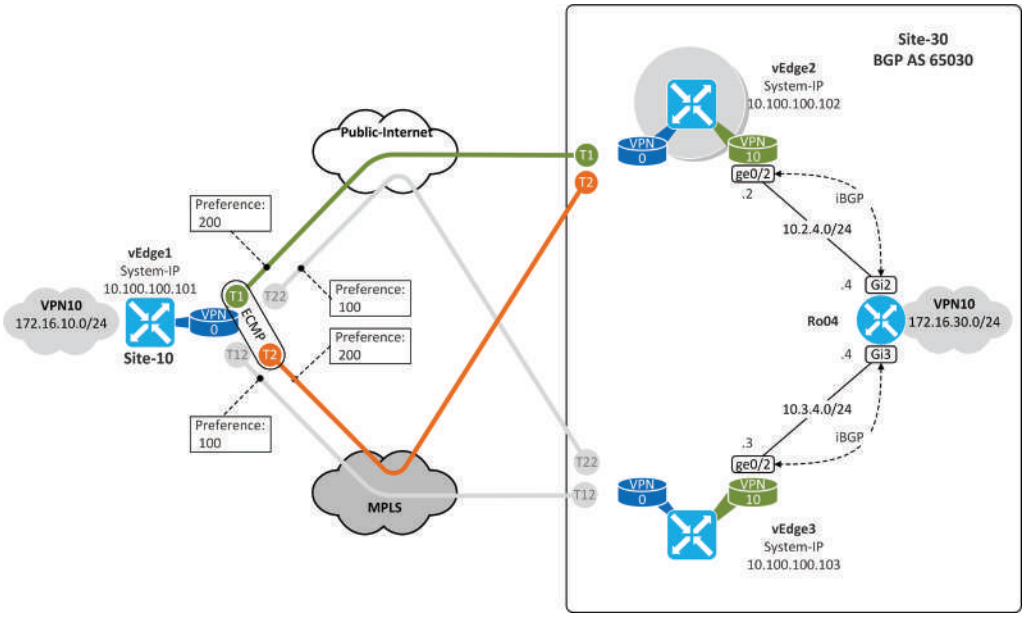


Figure 9-5: *ECMP to Site30 over Tunnels to vEdge2.*

TLOC List Configuration

As a first step, we are going to add TLOC lists. Navigate to the *configuration/policies* window and select the *Topology* option from the *Custom Option* drop-down menu.

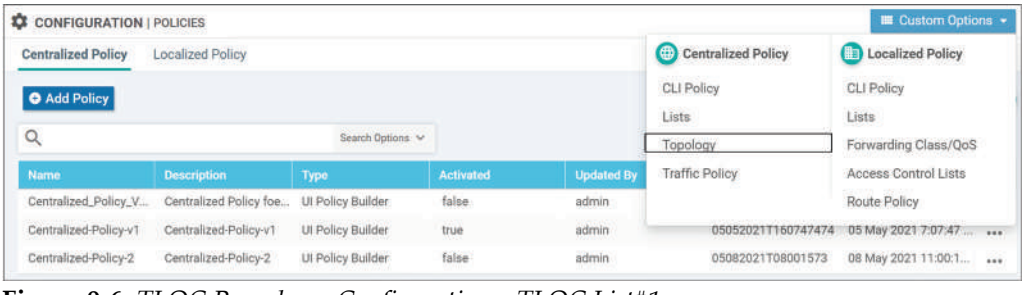


Figure 9-6: *TLOC Precedence Configuration – TLOC List#1.*

Select the TLOC from the Group of Interest section and click the *New TLOC List* button.

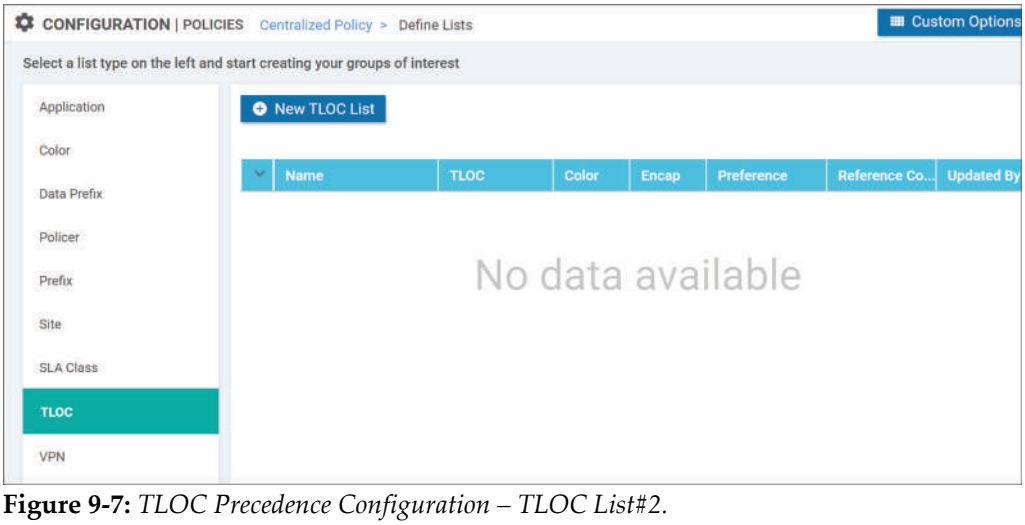


Figure 9-7: *TLOC Precedence Configuration – TLOC List#2.*

We are going to create a TLOC list for all vEdges where each list includes both TLOCs. Figure 9-8 shows how to configure the TLOC list for vEdge2. Note that the TLOC IP is the System-Id used by vEdge2.

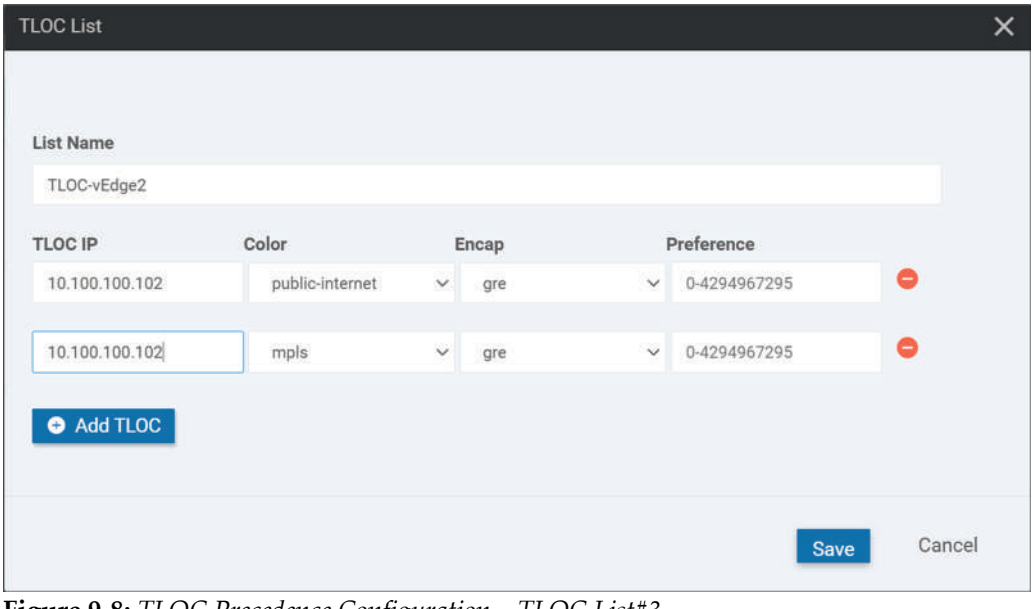


Figure 9-8: *TLOC Precedence Configuration – TLOC List#3.*

Figure 9-9 shows all vEdge-specific TLOC lists.

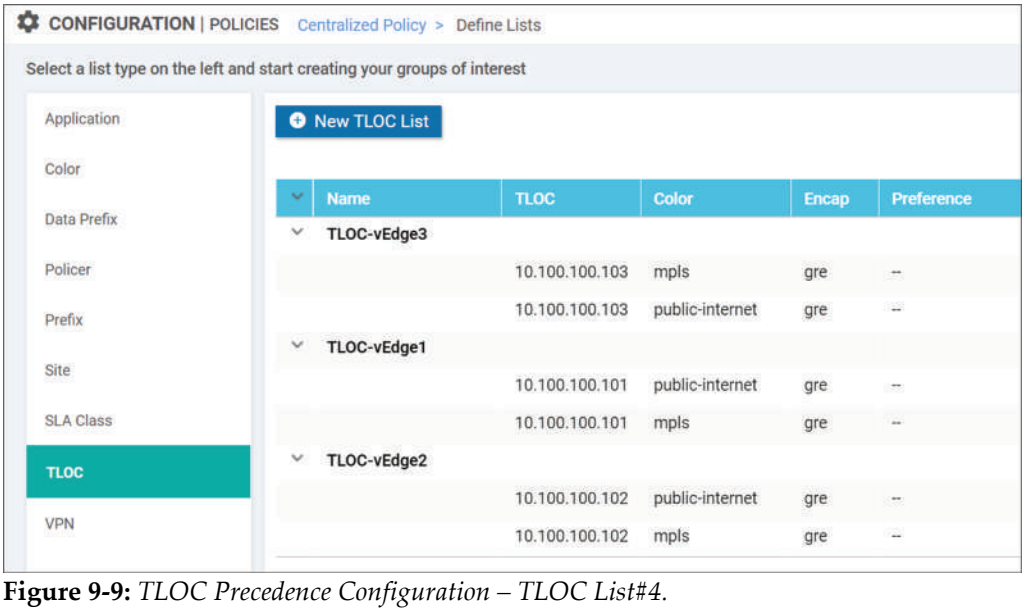


Figure 9-9: *TLOC Precedence Configuration – TLOC List#4.*

TLOC Control Policy Configuration

When TLOC lists are done, we can start to build a new Control Policy under Centralized Policy. I have copied our previously used Centralized Policy and then detached it from the vSmart. We are going to do all modifications to the new Centralized Policy. Select the *Centralized Policy-2* and choose the *Edit* from the Options menu […].

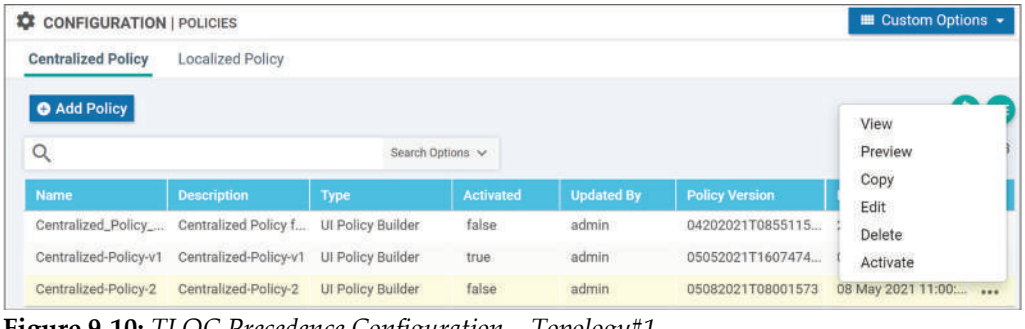


Figure 9-10: *TLOC Precedence Configuration – Topology#1.*

When you enter the *Edit Policy* window the *Policy Application* tab is selected by default. Click the *Topology* tab.

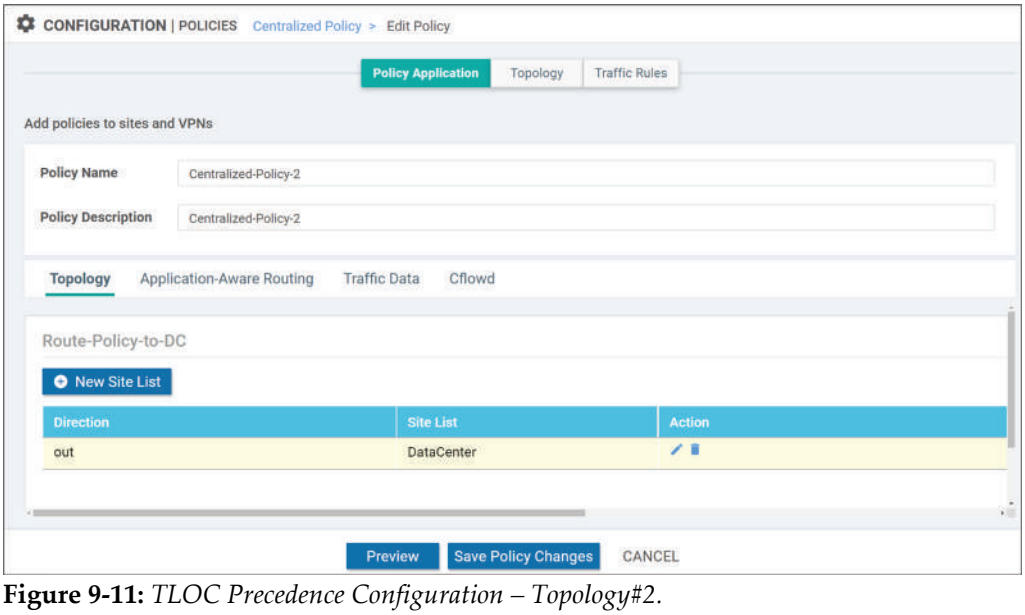


Figure 9-11: *TLOC Precedence Configuration – Topology#2.*

Open the *Add Topology* drop-down menu and select *Custom Control (Route & TLOC)* option.

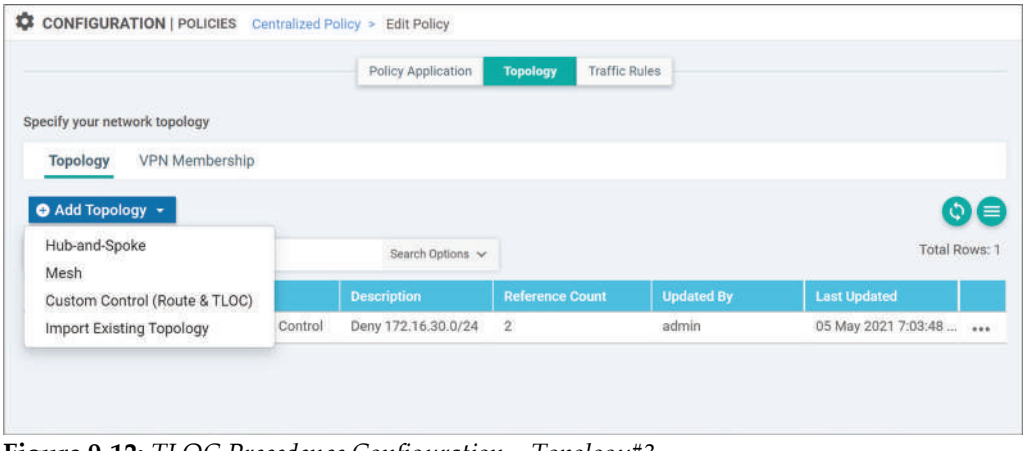


Figure 9-12: *TLOC Precedence Configuration – Topology#3.*

Change the default action from *Reject* to *Accept* by selecting *Default Action* from the *Sequence Type* section and the by choosing an *Accept* option.

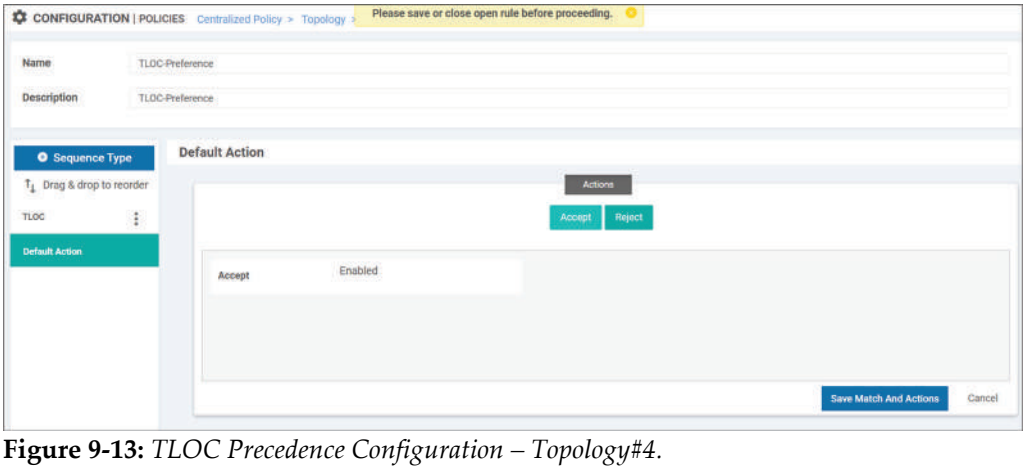


Figure 9-13: *TLOC Precedence Configuration – Topology#4.*

Next, select the *TLOC* option from the *Sequence Type* section. Click the *Match* selector and choose *TLOC*. Select the previously created TLOC list *TLOC-vEdge2* from the *TLOC List* in the *Match Conditions* section.

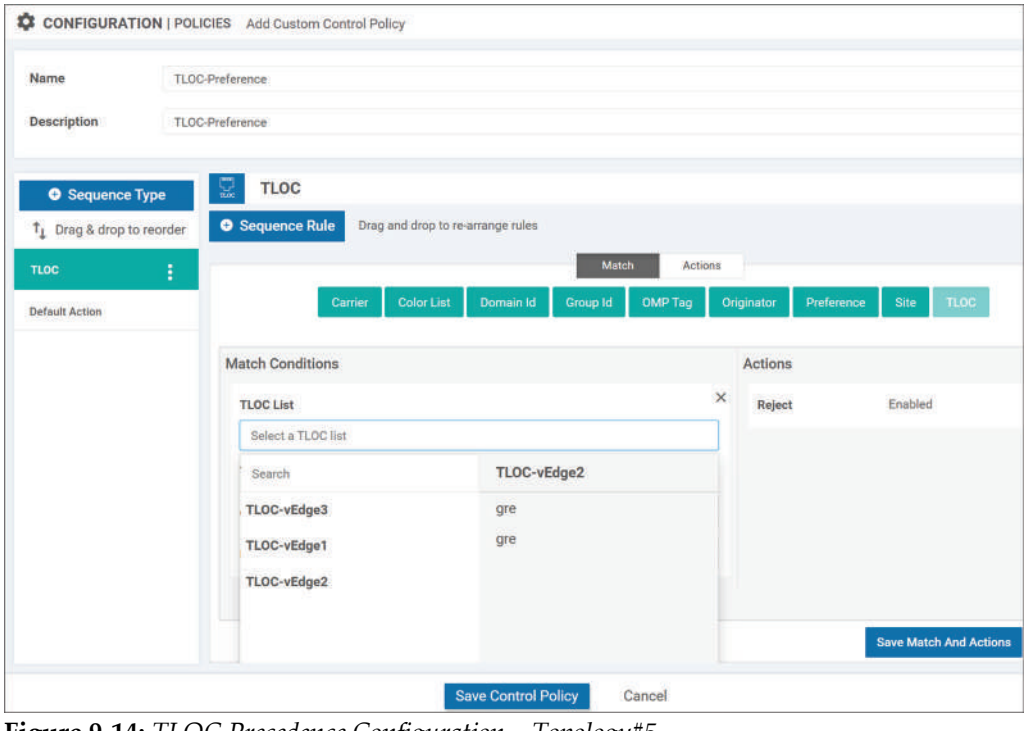


Figure 9-14: *TLOC Precedence Configuration – Topology#5.*

Click the *Actions* button and mark the *Accept* radio button. Click the *Preference* button and give preference value 200. Save changes by clicking the *Save Match And Action* button. Do the same process by using TLOC list TLOC-vEdge3 but set the preference to 100.

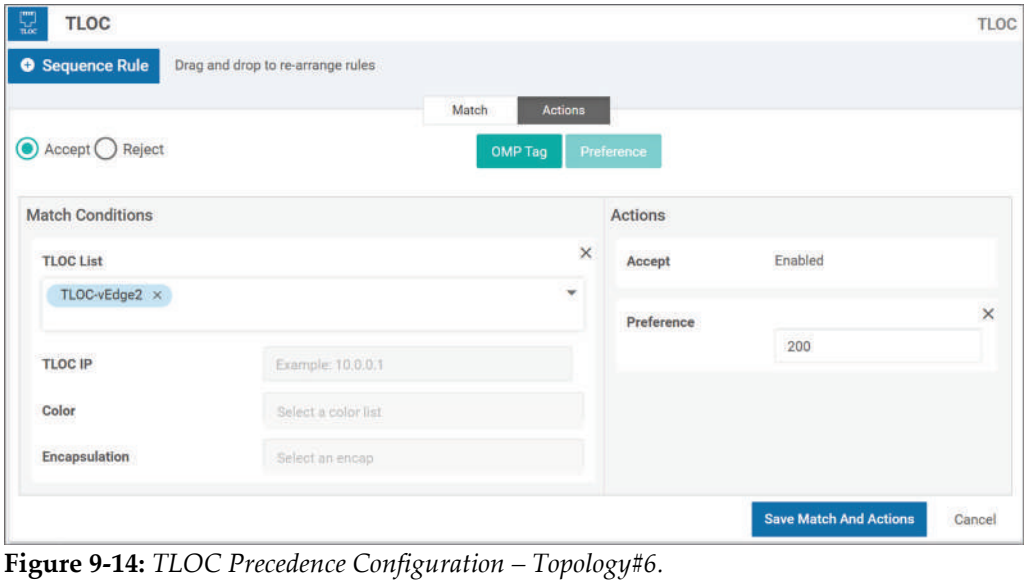


Figure 9-14: *TLOC Precedence Configuration – Topology#6.*

Figure 9-15 shows our rule. TLOCs defined in list TLOC-vEdge2 will be given preference value 200, while TLOCs defined in list TLOC-vEdge3 will be given preference 100 when this rule is applied as an outbound policy towards vEdge1.

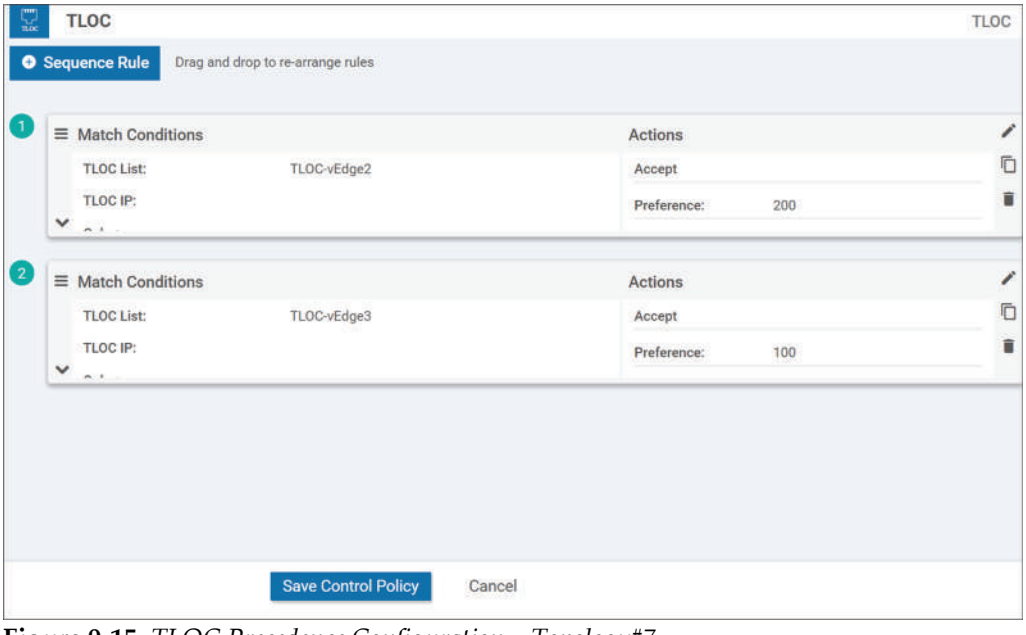


Figure 9-15: *TLOC Precedence Configuration – Topology#7.*

You can see that our topology *TLOC-Preference* is now listed in the Centralized PolicyTopology view. You can navigate to this page by opening the *Custom Options* drop-down menu and by selecting the *Topology* option under the *Centralized Policy* section.

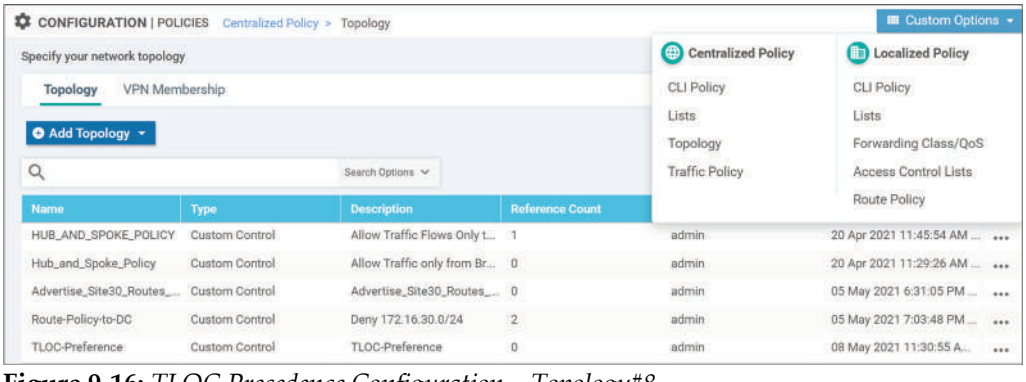


Figure 9-16: *TLOC Precedence Configuration – Topology#8.*

Applying Control Policy

Next, we need to apply the TLOC-Precedence topology to our Centralized Policy. Navigate back to the *Configuration/Policies* window and select the policy *Centralized Policy-2*. Click the *Options* selector […] and choose *Edit*.

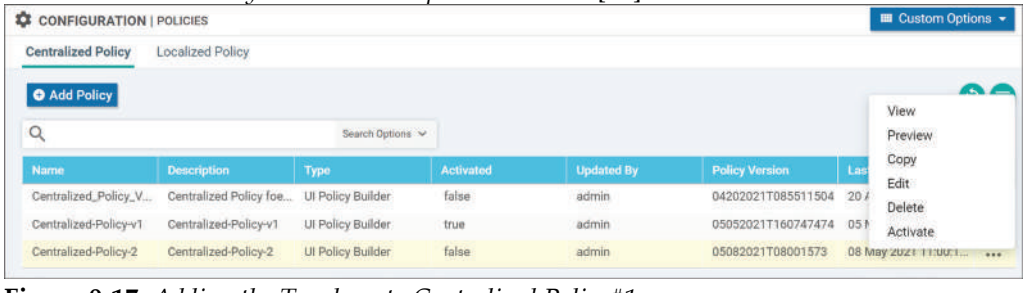


Figure 9-17: *Adding the Topology to Centralized Policy#1.*

Select the *Topology* tab and open the *Add Topology* drop-down menu. Choose the *Import Existing Topology*.

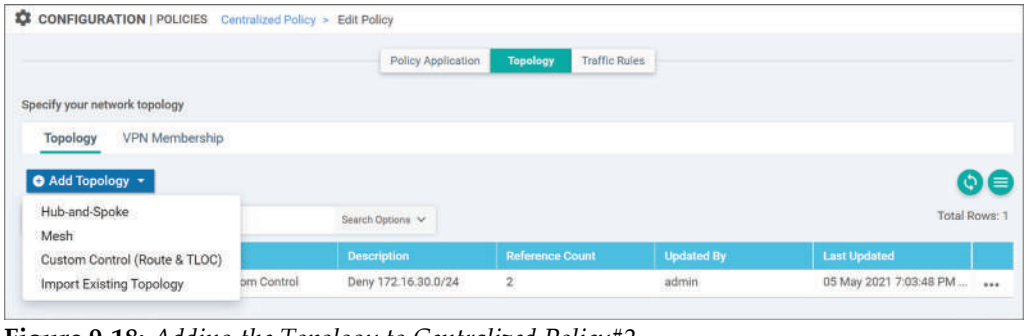


Figure 9-18: *Adding the Topology to Centralized Policy#2.*

Select *Custom Control (Route and TLOC)* radio button from the *Policy Type* section and choose our *TLOC-Preference* topology from the *Policy* drop-down menu.

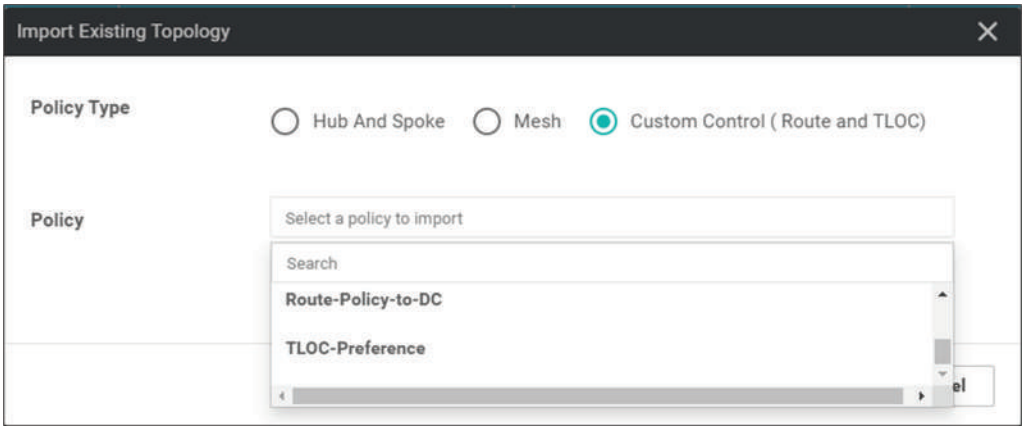


Figure 9-19: *Adding the Topology to Centralized Policy#3.*

Figure 9-16 shows out two topologies. The first one, *Route-Policy-to-DC*, is related to route optimization explained in chapter 8 section *Route-Optimization* and it is used for reject routing updates from site 30 to be sent back to the site.

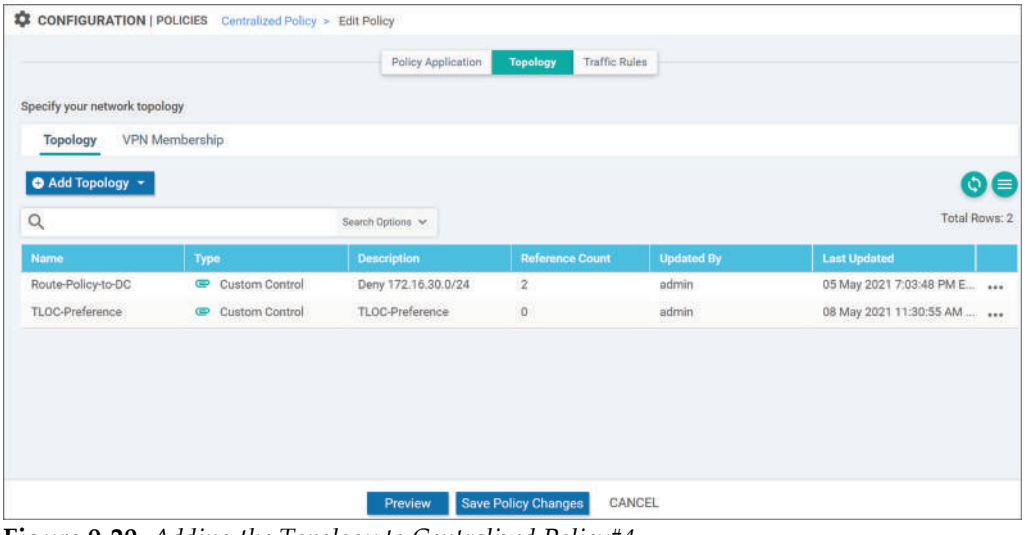


Figure 9-20: *Adding the Topology to Centralized Policy#4.*

Select the *Policy Application* button. Now you can see our *TLOC-Preference* policy listed as topology policy. Next, we need to apply the policy. Click the *New Site List* button and select the *Remote-Sites* list as an *Outbound Site List* and click the *Add* button. Figure 9-21 shows that now the *TLOC-Preference* topology policy is applied as outbound policy when TLOC route updates are sent to sites listed in the *Remote-Site* site list.

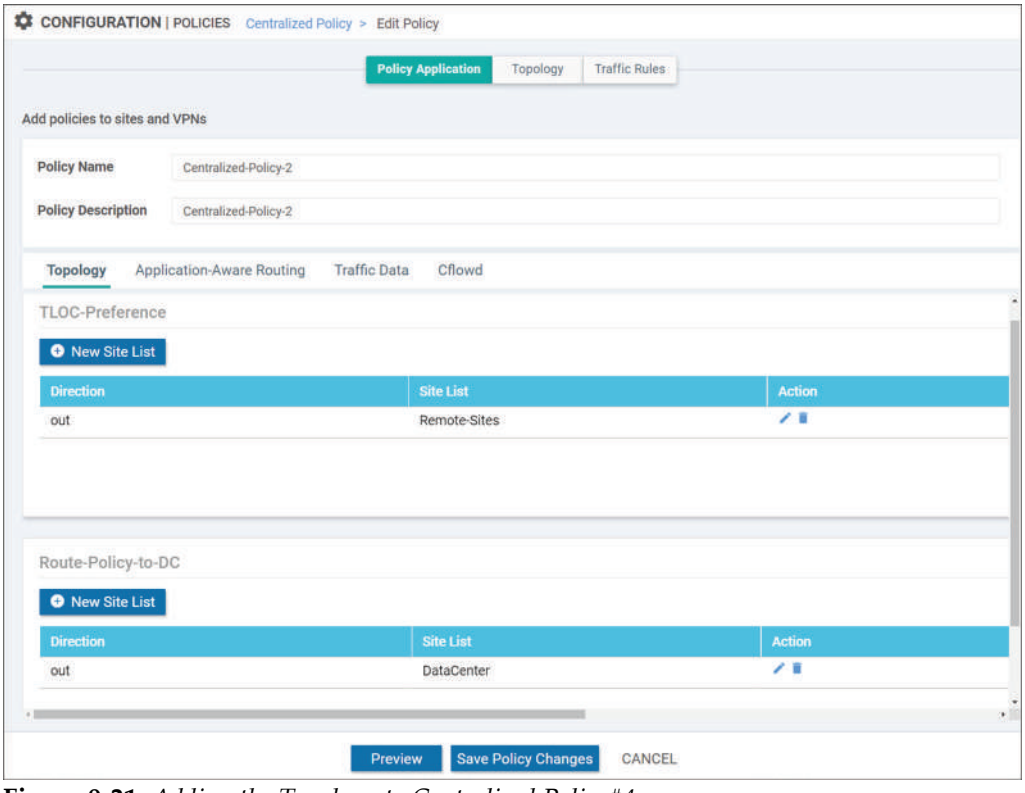


Figure 9-21: *Adding the Topology to Centralized Policy#4.*

You can preview the configuration before applying it. Example 9-1 shows the *Centralized Policy* that will be implemented in vSmart after applying changes. We have created a Control Policy *TLOC-Preference* where we set the preference value 200 for TLOCs attached to TLOC list *TLOC-vEdge2* and preference value 100 for TLOCs attached to TLOC list *TLOC-vEdge3*. The Control Policy TLOC-Preference is then applied as an outbound Control Policy towards sites listed in the Site List *Remote-Sites*.

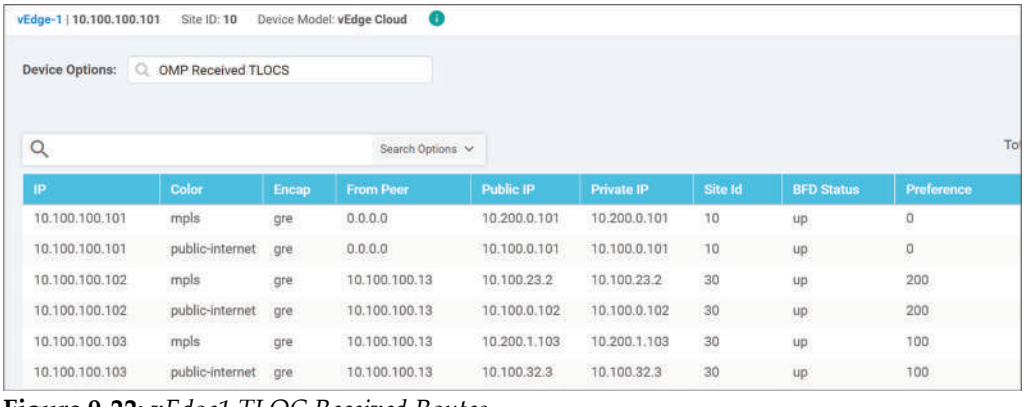
```
viptela-policy:policy
 control-policy Route-Policy-to-DC
    sequence 1
     match route
      prefix-list Site30-Prefix-List
     !
     action reject
     !
    !
  default-action accept
 !
 control-policy TLOC-Preference
    sequence 1
```

```
     match tloc
      tloc-list TLOC-vEdge2
     !
     action accept
      set
       preference 200
      !
     !
    !
    sequence 11
     match tloc
      tloc-list TLOC-vEdge3
     !
     action accept
      set
       preference 100
      !
     !
    !
  default-action accept
 !
 lists
  prefix-list Site30-Prefix-List
   ip-prefix 172.16.30.0/24
  !
  site-list DataCenter
   site-id 30
  !
  site-list Remote-Sites
   site-id 10
   site-id 20
  !
  tloc-list TLOC-vEdge2
   tloc 10.100.100.102 color public-internet encap gre
   tloc 10.100.100.102 color mpls encap gre
  !
  tloc-list TLOC-vEdge3
   tloc 10.100.100.103 color mpls encap gre
   tloc 10.100.100.103 color public-internet encap gre
  !
 !
!
apply-policy
 site-list Remote-Sites
  control-policy TLOC-Preference out
 !
 site-list DataCenter
  control-policy Route-Policy-to-DC out
 !
!
```

Example 9-1: *Complete Centralized Policy.*

Verification Control Policy Configuration

Figure 9-22 and examples from 9-2 to 9-5 verifies our that our policy works as expected. Both TLOCs originated by vEdge2 have a preference 200 while preference 100 is added to TLOCs originated by vEdge3.



IP	Color	Encap	From Peer	Public IP	Private IP	Site Id	BFD Status	Preference
10.100.100.101	mpls	gre	0.0.0.0	10.200.0.101	10.200.0.101	10	up	0
10.100.100.101	public-internet	gre	0.0.0.0	10.100.0.101	10.100.0.101	10	up	0
10.100.100.102	mpls	gre	10.100.100.13	10.100.23.2	10.100.23.2	30	up	200
10.100.100.102	public-internet	gre	10.100.100.13	10.100.0.102	10.100.0.102	30	up	200
10.100.100.103	mpls	gre	10.100.100.13	10.200.1.103	10.200.1.103	30	up	100
10.100.100.103	public-internet	gre	10.100.100.13	10.100.32.3	10.100.32.3	30	up	100

Figure 9-22: *vEdge1 TLOC Received Routes.*

```
vEdge-1# show omp tlocs ip 10.100.100.102 color public-internet received |
until prefe | exclude encap | nomore

--------------------------------------------------
tloc entries for 10.100.100.102
                 public-internet
                 gre
--------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     public-ip         10.100.0.102
     public-port       0
     private-ip        10.100.0.102
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           30
     overlay-id        not set
     preference        200
```

Example 9-2: *vEdge1 TLOC Received Routes: TLOC 10.100.100.102 public-internet gre.*

```
vEdge-1# show omp tlocs ip 10.100.100.102 color mpls received | until prefe |
exclude encap | nomore

--------------------------------------------------
tloc entries for 10.100.100.102
                 mpls
                 gre
--------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     public-ip         10.100.23.2
     public-port       0
     private-ip        10.100.23.2
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           30
     overlay-id        not set
     preference        200
```

Example 9-3: *vEdge1 TLOC Received Routes: TLOC 10.100.100.102 mpls gre.*

```
vEdge-1# show omp tlocs ip 10.100.100.103 color mpls received | until prefe |
exclude encap | nomore

--------------------------------------------------
tloc entries for 10.100.100.103
                 mpls
                 gre
--------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     public-ip         10.200.1.103
     public-port       0
     private-ip        10.200.1.103
     private-port      0
```

```
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           30
     overlay-id        not set
     preference        100
```

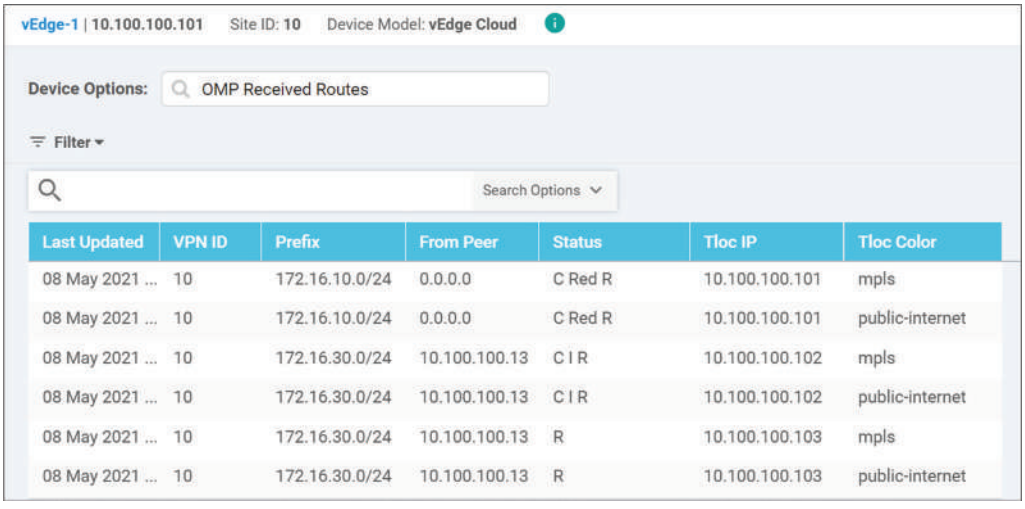
Example 9-4: *vEdge1 TLOC Received Routes: TLOC 10.100.100.103 mpls gre.*

```
vEdge-1# show omp tlocs ip 10.100.100.103 color public-internet received |
until prefe | exclude encap | nomore

--------------------------------------------------
tloc entries for 10.100.100.103
                 public-internet
                 gre
--------------------------------------------------
            RECEIVED FROM:
peer            10.100.100.13
status          C,I,R
loss-reason     not set
lost-to-peer    not set
lost-to-path-id not set
    Attributes:
     attribute-type    installed
     public-ip         10.100.32.3
     public-port       0
     private-ip        10.100.32.3
     private-port      0
     public-ip         ::
     public-port       0
     private-ip        ::
     private-port      0
     bfd-status        up
     domain-id         not set
     site-id           30
     overlay-id        not set
     preference        100
```

Example 9-5: *vEdge1 TLOC Received Routes: TLOC 10.100.100.102 public-internet gre.*

Figure 9-23 and example 9-6 verify that vEdge1 has installed only routes originated by vEdge2 into RIB. Routes originated by vEdge3 are not installed into the RIB.



Last Updated	VPN ID	Prefix	From Peer	Status	Tloc IP	Tloc Color
08 May 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	mpls
08 May 2021 ...	10	172.16.10.0/24	0.0.0.0	C Red R	10.100.100.101	public-internet
08 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.102	mpls
08 May 2021 ...	10	172.16.30.0/24	10.100.100.13	C I R	10.100.100.102	public-internet
08 May 2021 ...	10	172.16.30.0/24	10.100.100.13	R	10.100.100.103	mpls
08 May 2021 ...	10	172.16.30.0/24	10.100.100.13	R	10.100.100.103	public-internet

Figure 9-23: *vEdge1 VPN10 RIB.*

```
vEdge-1# show ip routes vpn 10 detail
Codes Proto-sub-type:
  IA -> ospf-intra-area, IE -> ospf-inter-area,
  E1 -> ospf-external1, E2 -> ospf-external2,
  N1 -> ospf-nssa-external1, N2 -> ospf-nssa-external2,
  e -> bgp-external, i -> bgp-internal
Codes Status flags:
  F -> fib, S -> selected, I -> inactive,
  B -> blackhole, R -> recursive, L -> import

""------------------------------------------
 VPN 10      PREFIX 172.16.10.0/24
------------------------------------------
 proto           connected
 distance        0
 metric          0
 uptime          0:02:54:05
 nexthop-ifname  ge0/2
 status          F,S

------------------------------------------
 VPN 10      PREFIX 172.16.30.0/24
------------------------------------------
 proto           omp
 distance        250
 metric          0
 uptime          0:00:01:50
 tloc-ip         10.100.100.102
 tloc-color      mpls
 tloc-encap      gre
 nexthop-label   1003
 status          F,S
```

```
------------------------------------------
 VPN 10      PREFIX 172.16.30.0/24
------------------------------------------
 proto           omp
 distance        250
 metric          0
 uptime          0:00:01:50
 tloc-ip         10.100.100.102
 tloc-color      public-internet
 tloc-encap      gre
 nexthop-label   1003
 status          F,S
```

Example 9-6: *vEdge1 VPN10 RIB.*

Now when we simulate the Youtube data flow from site 10 to site 30, we can see that only two tunnels between sites 10 and 30 are in use.

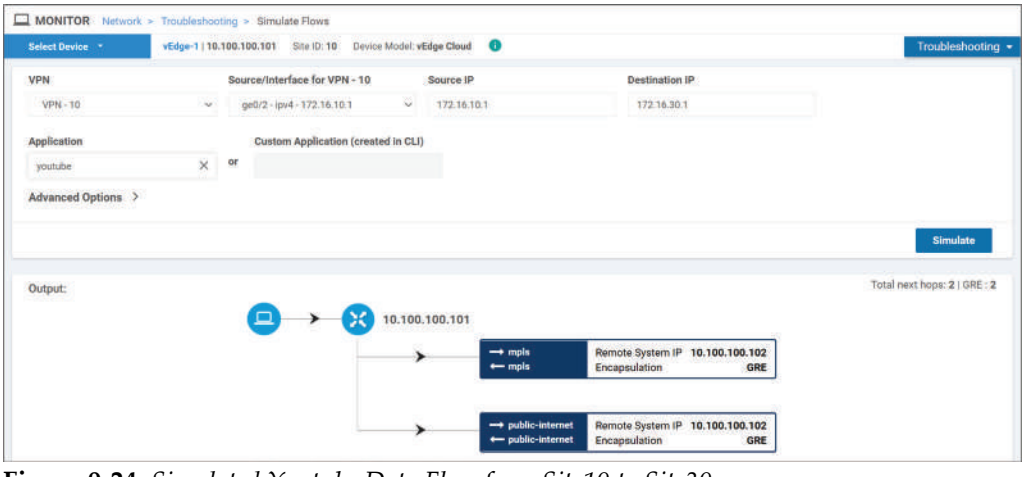


Figure 9-24: *Simulated Youtube Data Flow from Site10 to Site30.*

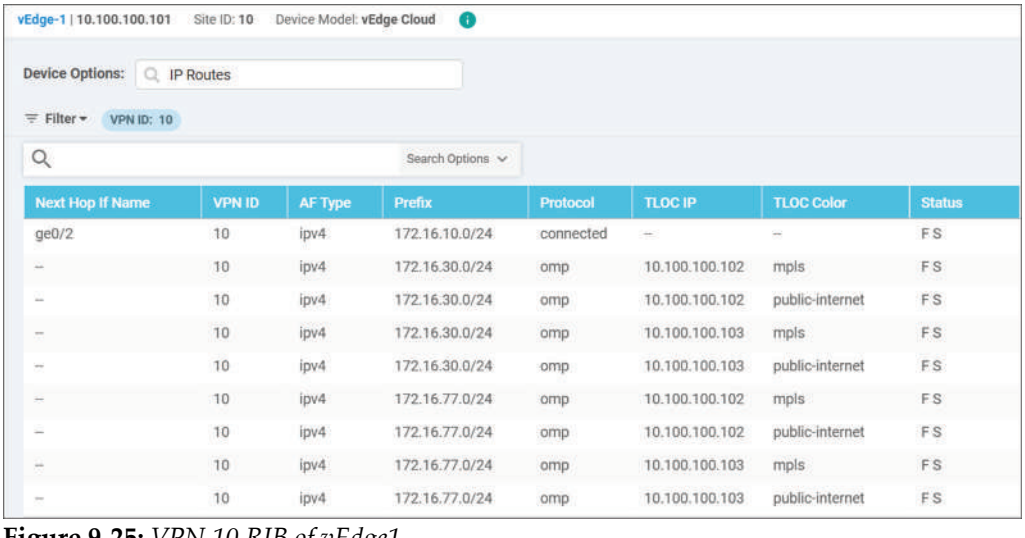
The trace in example 9-7 taken from the end-point PC1-10 attached to VPN10 in Site10 verifies that we still have IP connectivity between sites.

```
PC1-10> trace 172.16.30.1
trace to 172.16.30.1, 8 hops max, press Ctrl+C to stop
 1   172.16.10.1   0.342 ms  0.150 ms  0.135 ms
 2   10.2.4.2   4.098 ms  3.655 ms  2.252 ms
 3   *10.2.4.4   2.749 ms (ICMP type:3, code:3, Destination port unreachable)
*
```

Example 9-7: *Trace from site10 to Site30.*

Feature Template – Precedence & Prefix

This section explains how we can use the TLOC route preference attribute to build subnet-specific overlay topologies. I have added subnet 172.16.77.0/24 to site 30. Figure 9-25 shows that vEdge1 has installed both routes 172.16.30.0/24 and 172.16.77.0/24 in its VPN10 RIB. Note that I have detached the Centralized Policy which we configure in the previous section.



Next Hop If Name	VPN ID	AF Type	Prefix	Protocol	TLOC IP	TLOC Color	Status
ge0/2	10	ipv4	172.16.10.0/24	connected	–	–	F S
–	10	ipv4	172.16.30.0/24	omp	10.100.100.102	mpls	F S
–	10	ipv4	172.16.30.0/24	omp	10.100.100.102	public-internet	F S
–	10	ipv4	172.16.30.0/24	omp	10.100.100.103	mpls	F S
–	10	ipv4	172.16.30.0/24	omp	10.100.100.103	public-internet	F S
–	10	ipv4	172.16.77.0/24	omp	10.100.100.102	mpls	F S
–	10	ipv4	172.16.77.0/24	omp	10.100.100.102	public-internet	F S
–	10	ipv4	172.16.77.0/24	omp	10.100.100.103	mpls	F S
–	10	ipv4	172.16.77.0/24	omp	10.100.100.103	public-internet	F S

Figure 9-25: *VPN 10 RIB of vEdge1.*

Route Control Policy Configuration

Figure 9-26 shows the *Route* control policy *Prefix-Based-Preference* where we match both prefix-list and TLOC lists. Note that the prefix-list list configuration is done in the same way as what we did in the *TLOC List Configuration* section but instead of TLOC we will select *Prefix* from the Group of Interest section. The prefix-list *Site-30-Prefix-List-1* includes subnet 172.16.77.0/24 and the prefix-list *Site-30-Prefix-List-2* includes subnet 172.16.30.0/24. The first sequence states that subnet 172.16.77.0/24 when originated by vEdge2 gets the OMP preference 200. The second sequence states that subnet 172.16.30.0/24 when originated by vEdge2 gets the OMP preference 100 and so on.

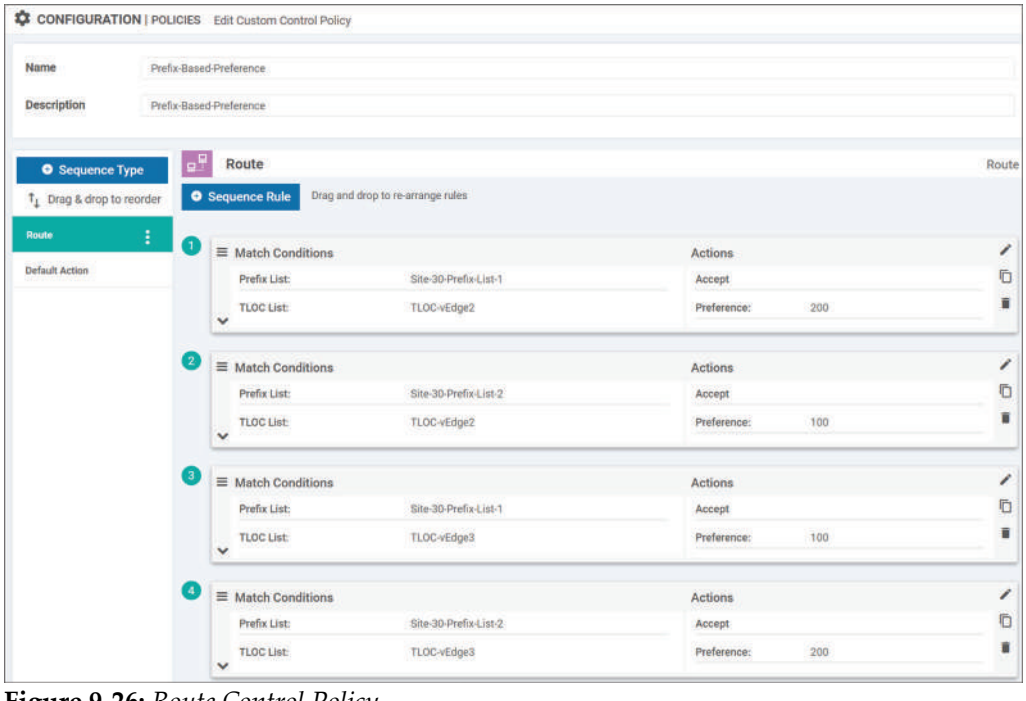


Figure 9-26: *Route Control Policy.*

The process of applying the Route Control Policy is the same as we did with TLOC Control Policy. I have once again copied the current Centralized Policy and renamed it to Central-Policy-3. Select it and choose the *Edit* from the Options menu […].

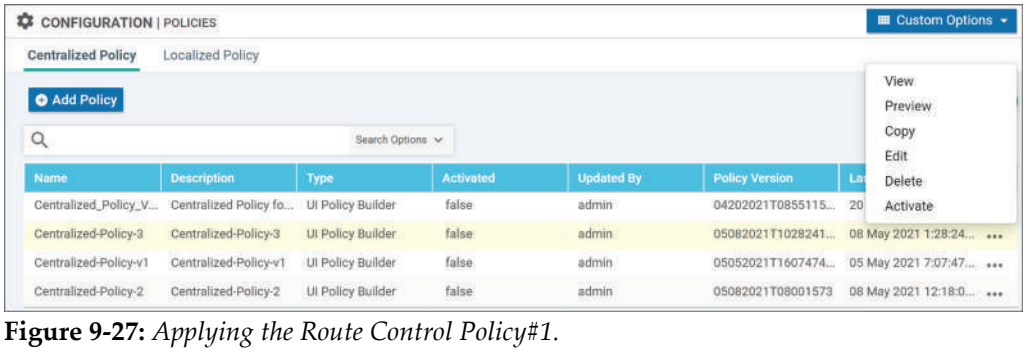


Figure 9-27: *Applying the Route Control Policy#1.*

Our Route Control Policy is now listed in the *Policy Application* tab. I have already applied the policy towards sites defined in the site-list *Remote-Access*. Click the *Save Policy Changes* button.

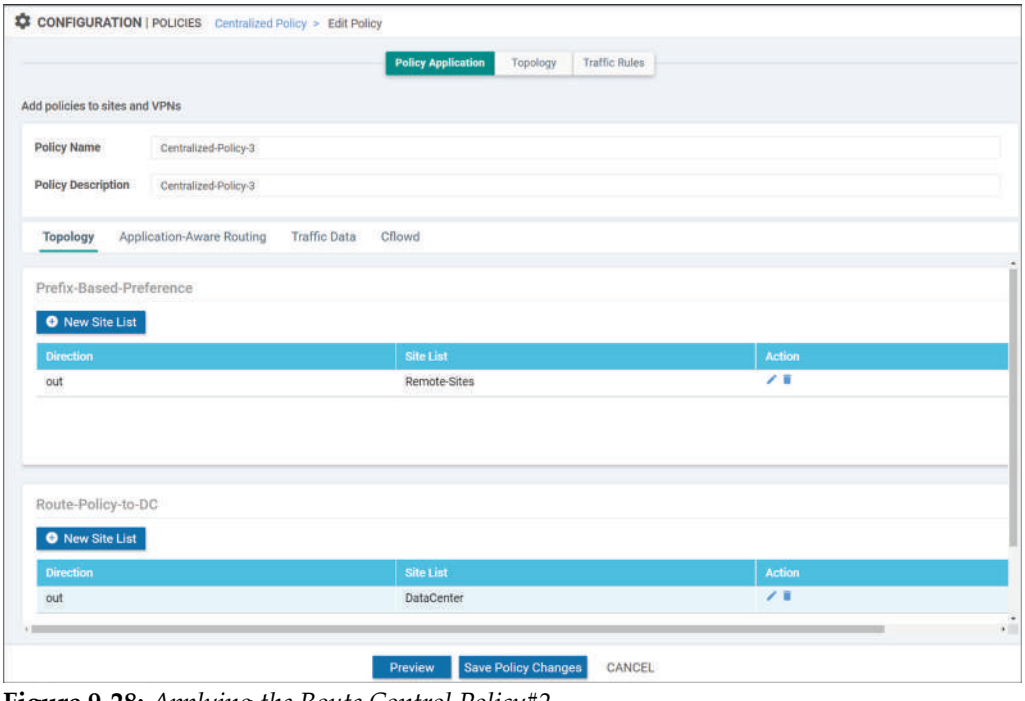


Figure 9-28: *Applying the Route Control Policy#2.*

Figure 9-29 shows that our Route Control Policy works as expected. vEdge1 receives all the OMP routes but it doesn't install either a route to subnet 172.16.30.0/24 originated by vEdge2 or a route to 172.16.77.0/24 originated by vEdge3 into VPN10 RIB (the status of these is only Received = R).

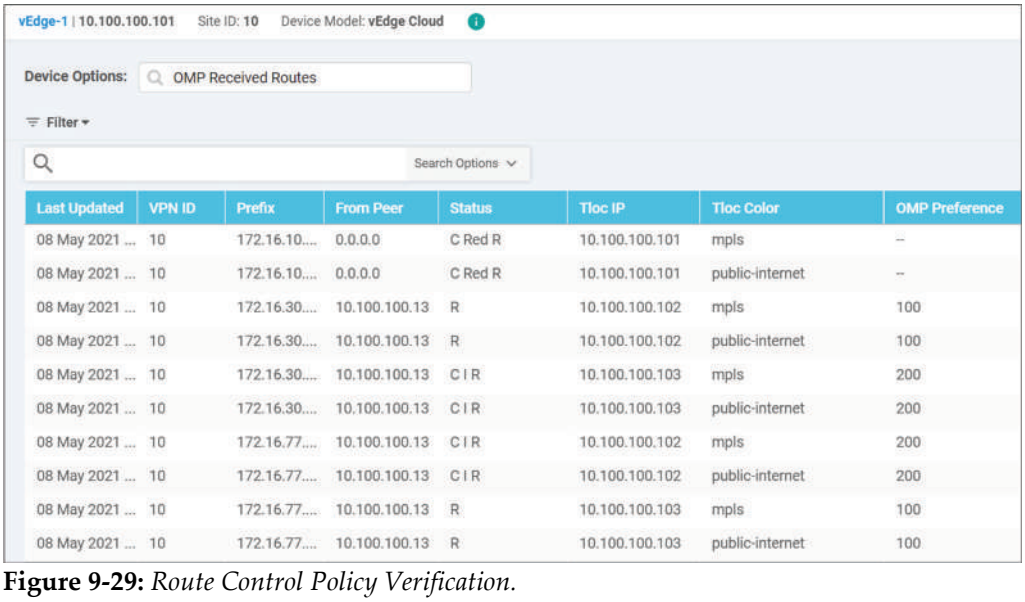


Figure 9-29: *Route Control Policy Verification.*

Figure 9-30 verifies that route to 172.16.30.0/24 is only available through the tunnels to vEdge3 while figure 9-31 shows that the subnet 172.16.77.0/24 is reachable over tunnels to vEdge2.

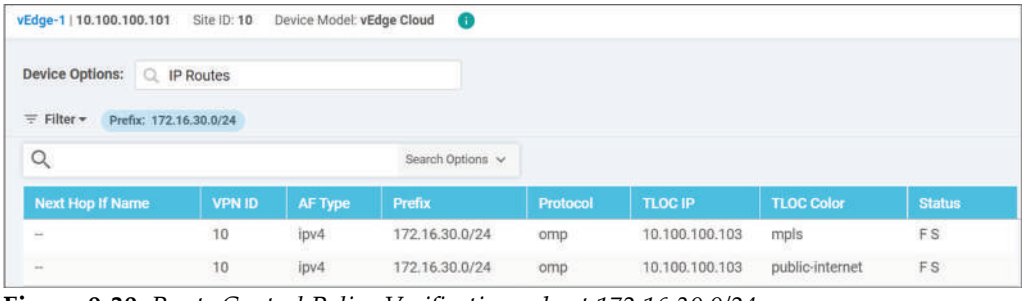


Figure 9-30: *Route Control Policy Verification subnet 172.16.30.0/24*

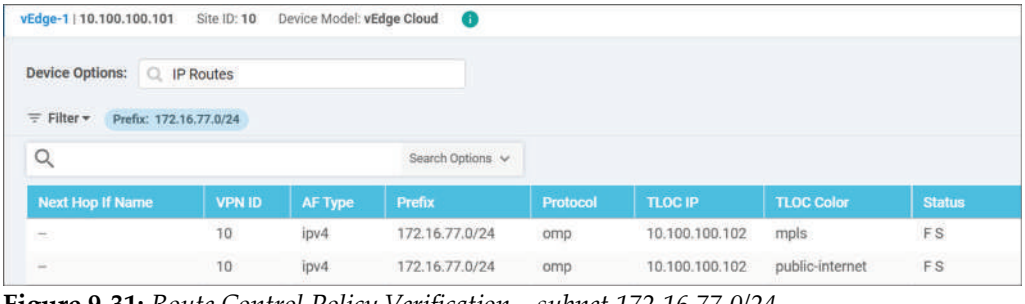


Figure 9-31: *Route Control Policy Verification – subnet 172.16.77.0/24.*

Example 9-8 shows the Centralized Policy configuration implemented in vSmart.

```
viptela-policy:policy
 control-policy Route-Policy-to-DC
    sequence 1
     match route
      prefix-list Site30-Prefix-List
     !
     action reject
     !
    !
  default-action accept
 !
 control-policy Prefix-Based-Preference
    sequence 1
     match route
      prefix-list Site-30-Prefix-List-1
      tloc-list TLOC-vEdge2
     !
     action accept
      set
```

```
      preference 200
     !
    !
   !
   sequence 11
    match route
     prefix-list Site-30-Prefix-List-2
     tloc-list TLOC-vEdge2
    !
    action accept
     set
      preference 100
     !
    !
   !
   sequence 21
    match route
     prefix-list Site-30-Prefix-List-1
     tloc-list TLOC-vEdge3
    !
    action accept
     set
      preference 100
     !
    !
   !
   sequence 31
    match route
     prefix-list Site-30-Prefix-List-2
     tloc-list TLOC-vEdge3
    !
    action accept
     set
      preference 200
     !
    !
   !
 default-action accept
!
lists
 prefix-list Site-30-Prefix-List-1
  ip-prefix 172.16.77.0/24
 !
 prefix-list Site-30-Prefix-List-2
  ip-prefix 172.16.30.0/24
 !
 prefix-list Site30-Prefix-List
  ip-prefix 172.16.30.0/24
 !
 site-list DataCenter
  site-id 30
 !
 site-list Remote-Sites
  site-id 10
```

```
   site-id 20
  !
  tloc-list TLOC-vEdge2
   tloc 10.100.100.102 color public-internet encap gre
   tloc 10.100.100.102 color mpls encap gre
  !
  tloc-list TLOC-vEdge3
   tloc 10.100.100.103 color mpls encap gre
   tloc 10.100.100.103 color public-internet encap gre
  !
 !
!
apply-policy
 site-list Remote-Sites
  control-policy Prefix-Based-Preference out
 !
 site-list DataCenter
  control-policy Route-Policy-to-DC out
 !
!
```

Example 9-8: *Complete Centralized Policy*

Example 9-9 verifies the data-plane operation. Trace from subnet 172.16.10.0 located in site 10 to subnet 172.16.30.1 goes via vEdge3 (IP address 10.3.4.3) and the to subnet 172.16.7.0/24 via vEdge2 (IP address 10.2.4.2).

```
PC1-10> trace 172.16.30.1
trace to 172.16.30.1, 8 hops max, press Ctrl+C to stop
 1   172.16.10.1   0.209 ms  0.076 ms  0.109 ms
 2   10.3.4.3   4.579 ms  4.459 ms  4.694 ms
 3   *10.3.4.4   3.792 ms (ICMP type:3, code:3, Destination port unreachable)
*


PC1-10> trace 172.16.77.1
trace to 172.16.77.1, 8 hops max, press Ctrl+C to stop
 1   172.16.10.1   0.377 ms  0.155 ms  0.267 ms
 2   10.2.4.2   4.800 ms  5.369 ms  2.493 ms
 3   *10.2.4.4   3.013 ms (ICMP type:3, code:3, Destination port unreachable)
*
```

Example 9-9: *Data-Plane Testing*

Feature Template – Precedence

The last section of this chapter briefly describes how we can set the Tunnel Based Preference by using Feature Templates. We have done Feature Template *vEdge-VPN0-inet-ge0/0* in chapter 6 and now we are going to edit that one. We can add preference by selecting the *Tunnel* section and by opening *Advanced Options*. There we can set the tunnel-specific preference value as a global or device-specific value. Note that there is also a *Weight* attribute that is used for weighted flow-based load-balancing over two tunnels having the same preference. If you set the weight 25 for one tunnel and 75 for another, you will end up forwarding every fourth flow over the tunnel with a weight value of 25.

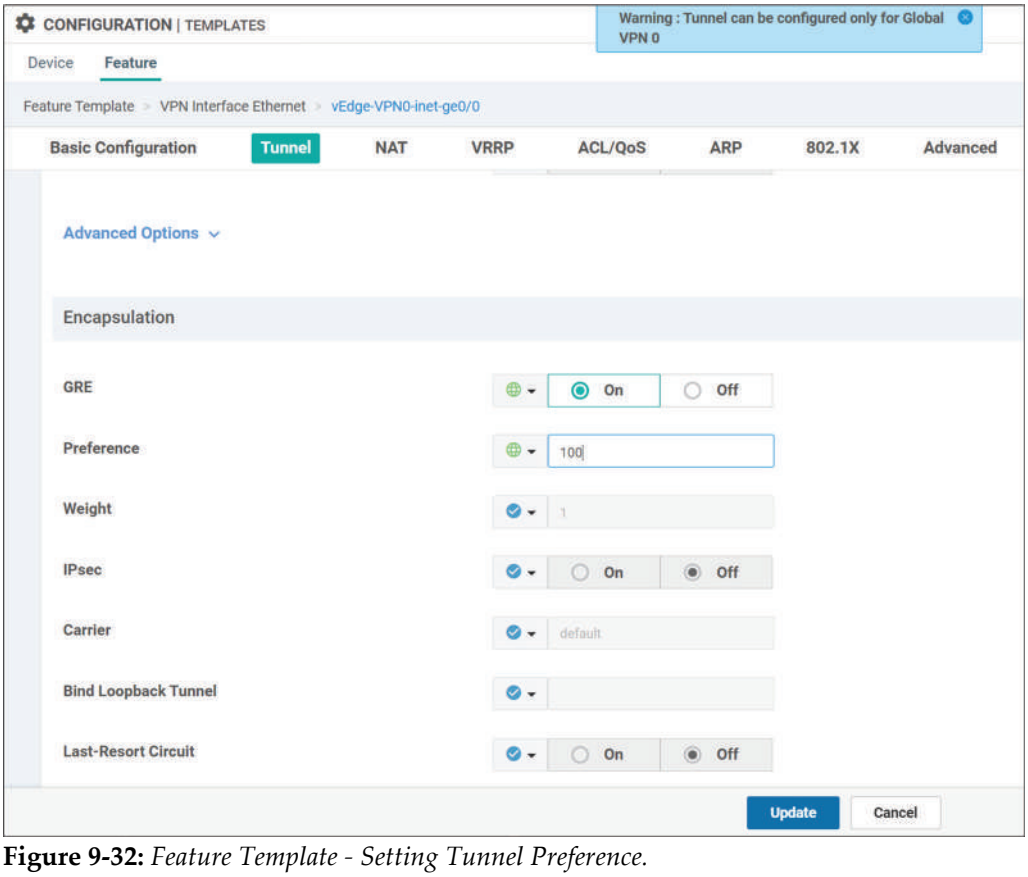


Figure 9-32: *Feature Template - Setting Tunnel Preference.*

Figure 9-28 shows the configuration preview.

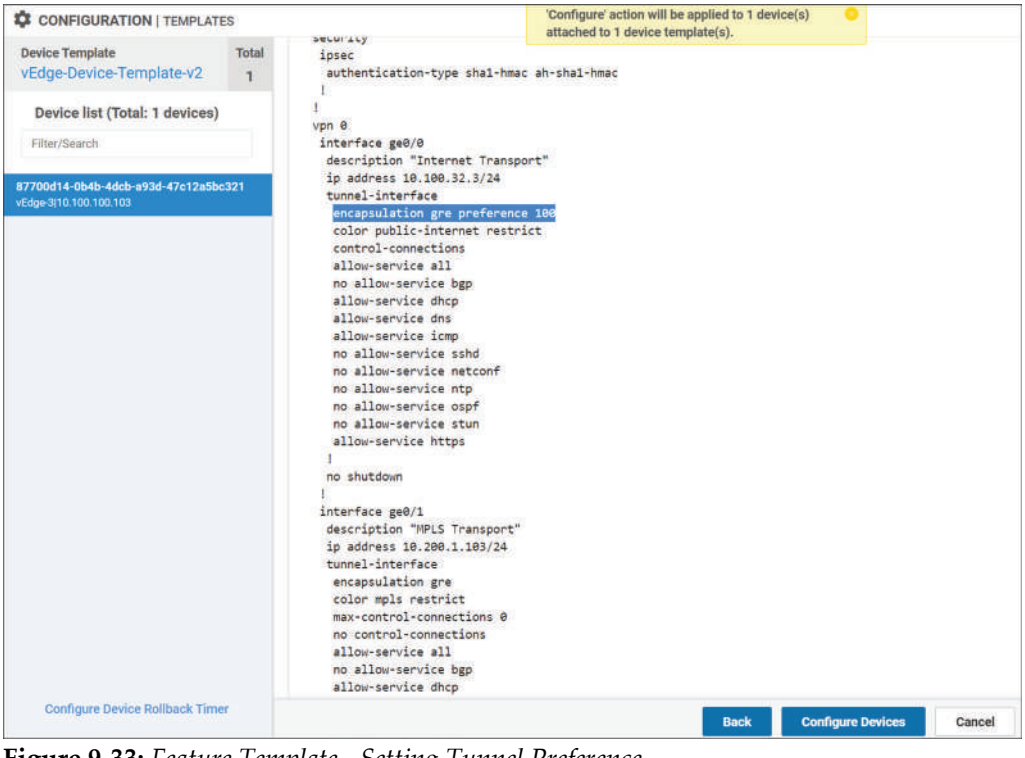


Figure 9-33: *Feature Template - Setting Tunnel Preference.*

Summary

This chapter shows how we can do Trafic Engineering using Centralized Policies and Feature Templates.

Chapter 10: Application-Aware Routing

Introduction

vEdges use BFD probes for *Tunnel Health* monitoring. Both vEdges in figure 10-1 send BFD probes in 1000ms frequency (Hello Interval). If seven sequential probe replies are missing (Multiplier), the state of the tunnel is changed to down. The *Application-Aware Routing* (AAR) in turn uses BFD probes for observation *Path Quality*. In our example, the AAR poll interval value is 5000ms, which means that vEdges calculate an average Packet Loss, Latency, and Jitter values from the received BFD echoes within a five-second time frame. The calculated values in the first poll interval are Packet Loss: 20%, Latency: 5ms, and Jitter: 1ms. Because this was the first App-Route poll interval, it is not compared against threshold values defined in sla-class NWKT_CRITICAL. Changing the path whenever the result of the single App-Route poll interval is non-compliant might cause path flapping where application data is shifted from path to path repeatedly. This behavior can be avoided by using an *App-Route Multiplier* where the shifting decision is based on the average value calculated from more than one App-Route poll interval. We are using App-Route Multiplier 2 in the example shown in figure 10-1. The calculated values in the second poll interval are Packet Loss: 0%, Latency: 6ms, and Jitter: 1ms. The next step is to calculate average values from these two newest App-Route poll intervals, which gives us Packet Loss: (20+0/2) = 10%, Latency: (5+6)/2 = 5.5ms, and Jitter: (1+1)/2 1ms. These values are lower than what is defined in sla-class NWKT_CRITICAL so we can use the path. In the third iteration round (not shown in the figure) average values are calculated based on the second and the third App-Route poll interval, the first one is excluded based on multiplier 2 (only two latest poll intervals are used in our case), and so on. The default values for BFD are Tx Interval 1000ms and Multiplier value 7. The default values for AAR are poll interval 600 000ms (10 minutes), and multiplier value 6.

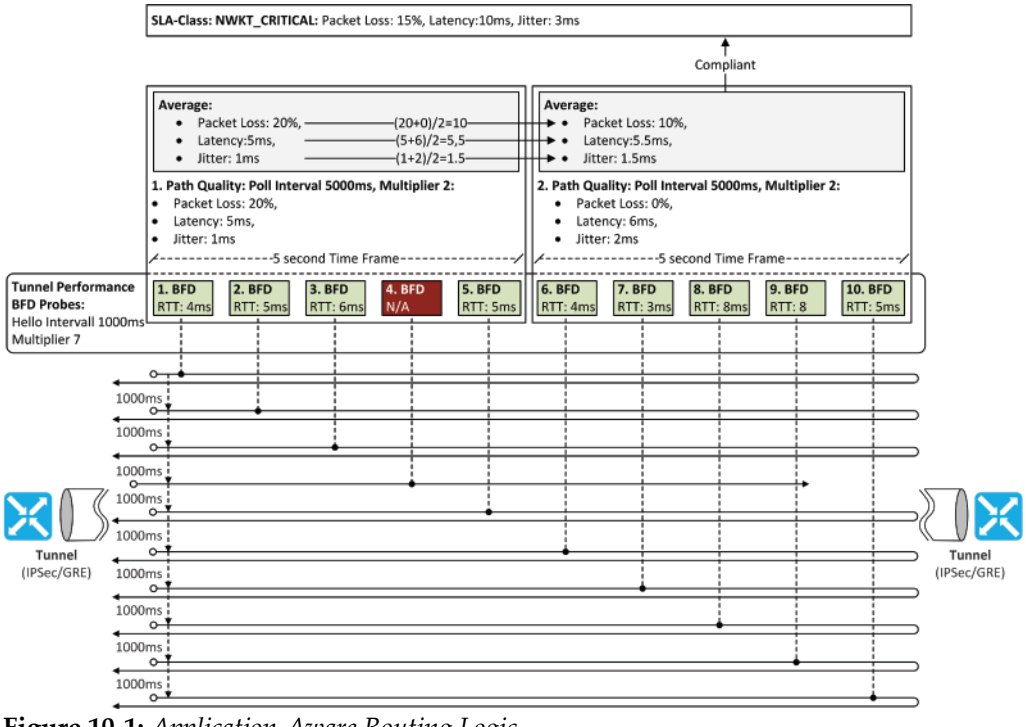


Figure 10-1: *Application-Aware Routing Logic.*

Tunnel Health Monitoring

BFD Settings

You can verify BFD transit interval and multiplier values from the device-specific Real-Time/BFD Session view by navigating to the *monitoring/network/(device)* page. Figure 10-2 shows the BFD values used in vEdge-1. Note that if BFD values are different in the remote device, the higher values will be used by both vEdges.

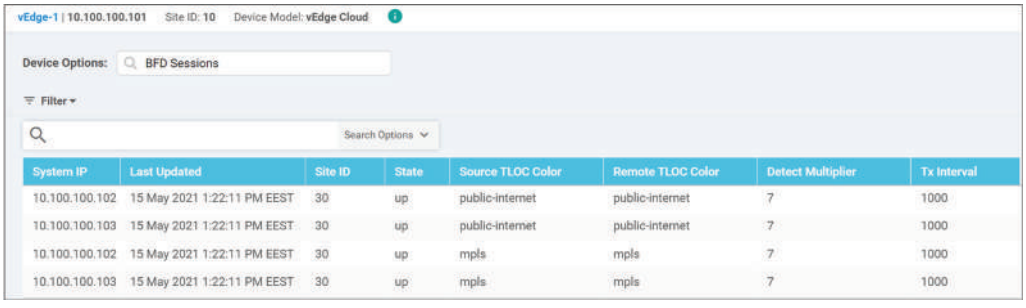


Figure 10-2: *vEdge-1 one Real-Time BFD Sessions.*

The default BFD values can be set in the Feature Template where you can also set color-specific BFD values. Besides, you can modify the Application-Aware Routing multiplier and poll interval values in the Basic Configuration section. We are leaving both AAR and BFD settings to their default, global values.

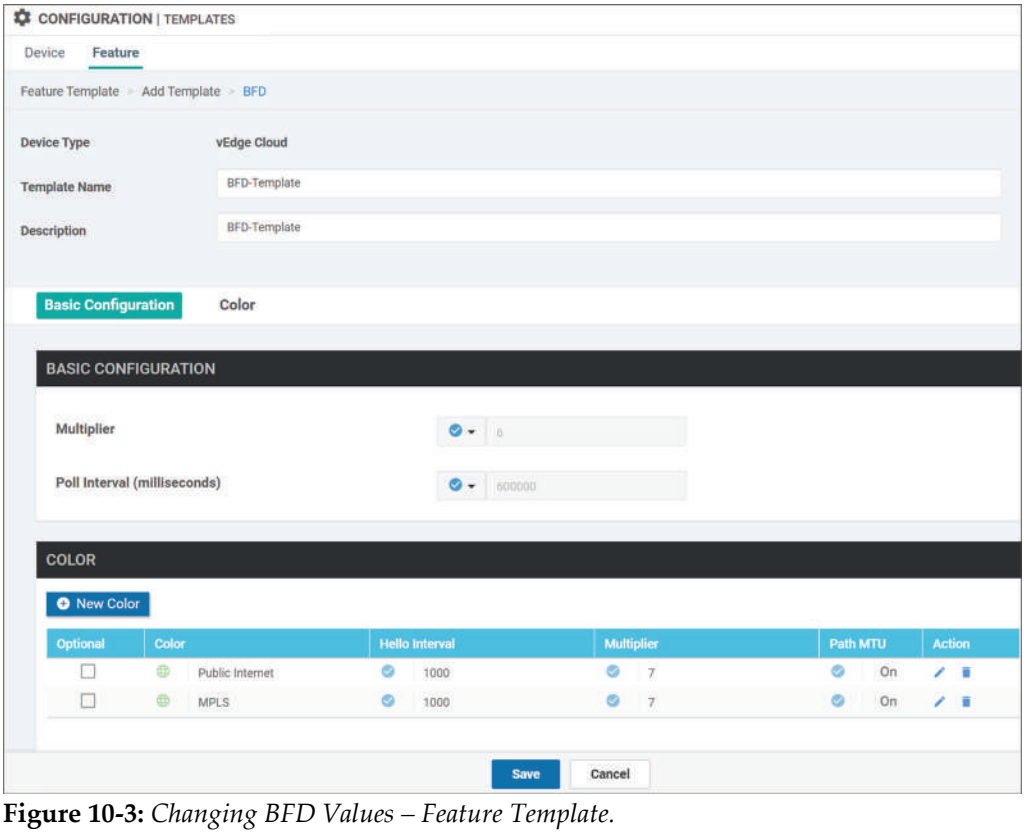


Figure 10-3: *Changing BFD Values – Feature Template.*

After creating the Feature Template you can add it to the Device Template in the Basic Information section where you select the template from the BFD drop-down menu.

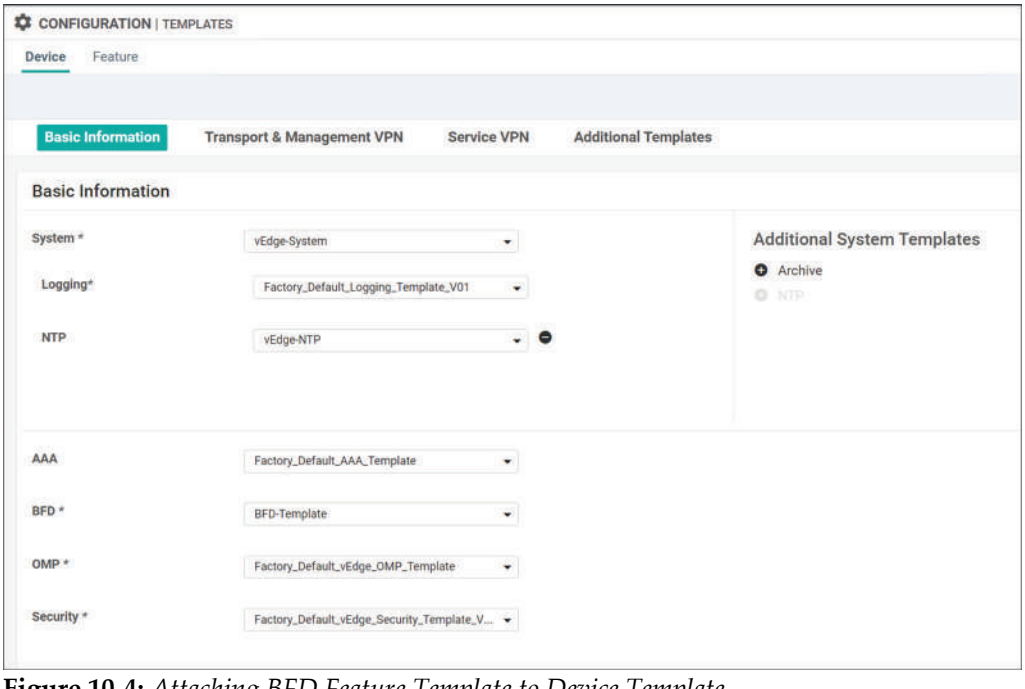


Figure 10-4: *Attaching BFD Feature Template to Device Template.*

Tunnel Switch-Over Process

Figure 10-5 shows the stabile situation where the tunneled packets, including BFD probes, are label switched through PE-1-West and PE-2-East routers in the MPLS transport network. Note that we are still using TLOC-Extension configured in chapter 7. That is why vEdge-1 is connected to the MPLS transport network through the vEdge-2 which in turn is connected to the Public-Internet transport network through the vEdge-1. We are using default BFD transmit interval and multiplier values in our example.

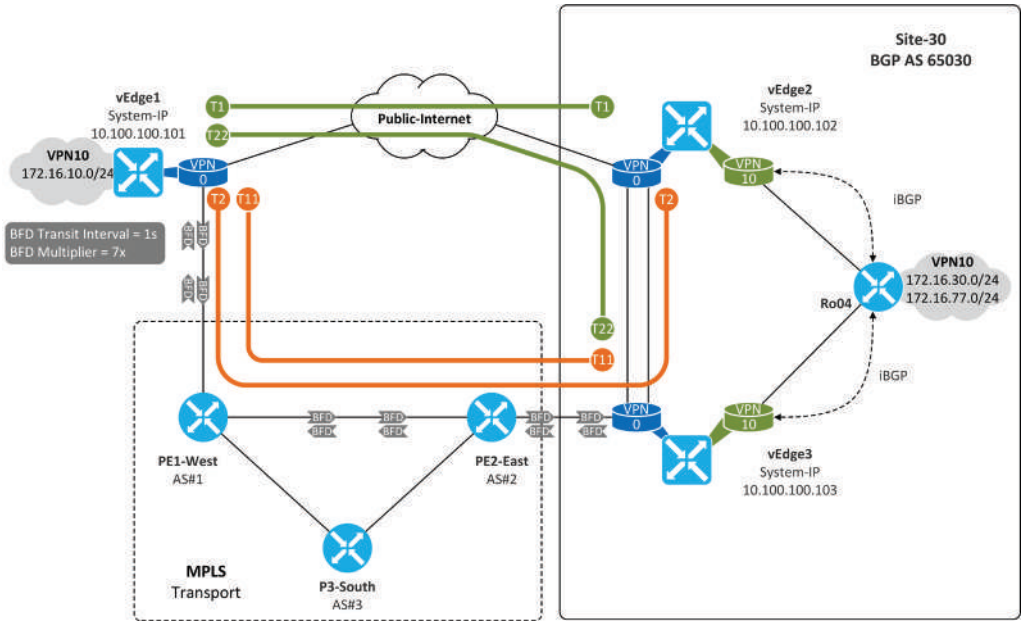


Figure 10-5: *Tunnel Health Monitoring - Stabile Situation.*

Figure 10-6 verifies that all BFD echoes sent by vEdge-1 are successfully echoed back over both transport networks.

vEdge-1 | 10.100.100.101 Site ID: 10 Device Model: vEdge Cloud

Device Options: Tunnel BFD Statistics

Search Options

Hostname	Last Updated	Tunnel Protocol	Source IP	Destination IP	BFD Echo Tx Pkts	BFD Echo Rx Pkts
vEdge-1	15 May 2021 1:59:33 PM EEST	gre	10.100.0.101	10.100.0.102	550	550
vEdge-1	15 May 2021 1:59:33 PM EEST	gre	10.100.0.101	10.100.32.3	551	551
vEdge-1	15 May 2021 1:59:33 PM EEST	gre	10.200.0.101	10.100.23.2	550	550
vEdge-1	15 May 2021 1:59:33 PM EEST	gre	10.200.0.101	10.200.1.103	551	551

Figure 10-6: *BFD Statistics.*

Figure 10-7 illustrates a link failure event between PE1-West and PE2-East. This is done by shutting down the interface on PE1-West and it reacts immediately to failure event by a) removing the inter-router subnet from the RIB, b) tearing down the BGP adjacency with P2-East, and c) starts using the path via P3-South towards PE2-East. So Technically it continues forwarding BFD Echoes received from vEdge-1. The BGP processes and RIB updates are shown in example 10-1. However, from the PE2-East perspective, this is an indirect link-failure meaning its interface towards PE1-West still remains up. Its BGP Hold timer is set to 20 seconds and it will wait that the hold timer expires before it tears down the BGP adjacency with PE1-West, and updating its RIB. Until then it will forward BFD echo replies towards PE1-West. By comparing the timestamps on example 10-1 and 10-2 you can see that the difference in the RIB update is approximately 17 seconds. vEdge1, however, will exclude the path over MPLS color from the ECMP after seven missed BFD Echo replies and start using only tunnels over the Public-Internet transport network.

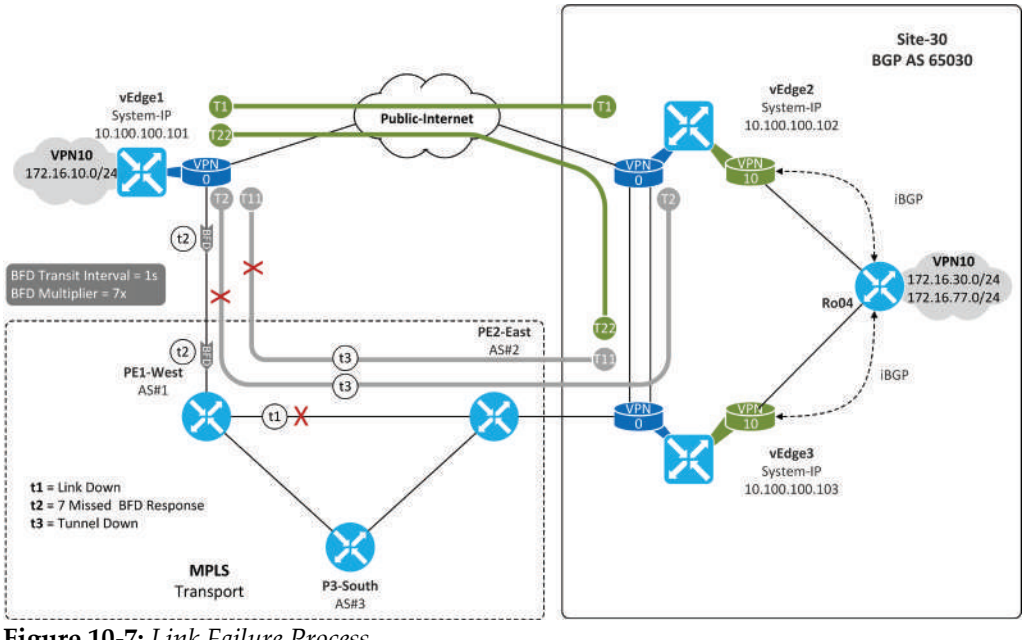


Figure 10-7: *Link Failure Process.*

```
PE1-West
*May 15 11:14:43: is_up: GigabitEthernet2 0 state: 6 sub state: 1 line: 1
*May 15 11:14:43.006: %BGP-5-NBR_RESET: Neighbor 10.1.2.2 reset (Interface flap)
*May 15 11:14:43: RT: interface GigabitEthernet2 removed from routing table
*May 15 11:14:43: RT: del 10.1.2.0 via 0.0.0.0, connected metric [0/0]
*May 15 11:14:43: RT: delete subnet route to 10.1.2.0/24
*May 15 11:14:43: RT: interface GigabitEthernet2 topo state DOWN, afi 0
*May 15 11:14:43: RT: del 10.1.2.1 via 0.0.0.0, connected metric [0/0]
*May 15 11:14:43: RT: delete subnet route to 10.1.2.1/32
*May 15 11:14:43.009: %BGP-5-ADJCHANGE: neighbor 10.1.2.2 Down Interface flap
*May 15 11:14:43.009: %BGP_SESSION-5-ADJCHANGE: neighbor 10.1.2.2 IPv4 Unicast
                      topology base removed from session  Interface flap
*May 15 11:14:43: RT: updating bgp 10.100.23.0/24 (0x0) [local lbl/ctx:1048577/0x0]  :
                      via 10.1.3.3   0 1048577 1048577

*May 15 11:14:43: RT: closer admin distance for 10.100.23.0, flushing 1 routes
*May 15 11:14:43: RT: add 10.100.23.0/24 via 10.1.3.3, bgp metric [20/0]
*May 15 11:14:43: RT: updating bgp 10.200.1.0/24 (0x0) [local lbl/ctx:1048577/0x0]  :
                      via 10.1.3.3   0 1048577 1048577

*May 15 11:14:43: RT: closer admin distance for 10.200.1.0, flushing 1 routes
*May 15 11:14:43: RT: add 10.200.1.0/24 via 10.1.3.3, bgp metric [20/0]
*May 15 11:14:44.952: %LINK-5-CHANGED: Interface GigabitEthernet2, changed state to
administratively down
*May 15 11:14:44: is_up: GigabitEthernet2 0 state: 6 sub state: 1 line: 1
*May 15 11:14:45.953: %LINEPROTO-5-UPDOWN: Line protocol on Interface
GigabitEthernet2, changed state to down
*May 15 11:14:45: is_up: GigabitEthernet2 0 state: 6 sub state: 1 line: 1
```

Example 10-1: *BGP Notifications in PE1-West.*

```
PE2-East#
*May 15 11:15:00.026: %BGP-3-NOTIFICATION: sent to neighbor 10.1.2.1 4/0 (hold time
expired) 0 bytes
*May 15 11:15:00.027: %BGP-5-NBR_RESET: Neighbor 10.1.2.1 reset (BGP Notification
sent)
*May 15 11:15:00.027: %BGP-5-ADJCHANGE: neighbor 10.1.2.1 Down BGP Notification sent
*May 15 11:15:00.027: %BGP_SESSION-5-ADJCHANGE: neighbor 10.1.2.1 IPv4 Unicast
topology base removed from session  BGP Notification sent
*May 15 11:15:00: RT: updating bgp 10.200.0.0/24 (0x0) [local lbl/ctx:1048577/0x0]  :
    via 10.2.3.3   0 1048577 1048577

*May 15 11:15:00: RT: closer admin distance for 10.200.0.0, flushing 1 routes
*May 15 11:15:00: RT: add 10.200.0.0/24 via 10.2.3.3, bgp metric [20/0]
```

Example 10-2: *BGP Notifications in PE2-East.*

Figure 10-8 verifies that tunnels over MPLS transport are down from vEdge-1 to vEdges in site 30.

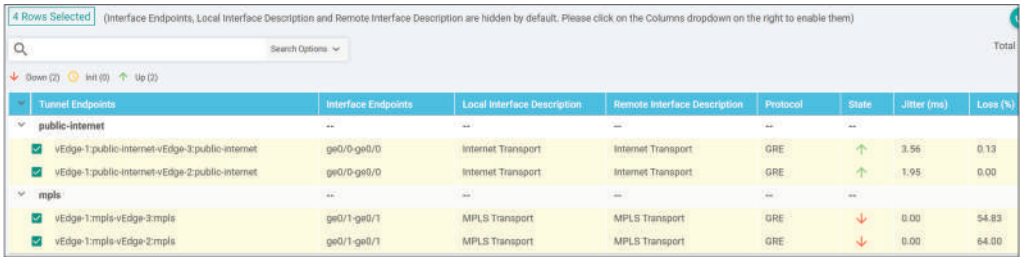


Figure 10-8: *Verification.*

After routing tables in both PE1-West and PE2-East are updated, the BFD probes start flowing over the MPLS transport network and tunnels can again be used for ECMP.

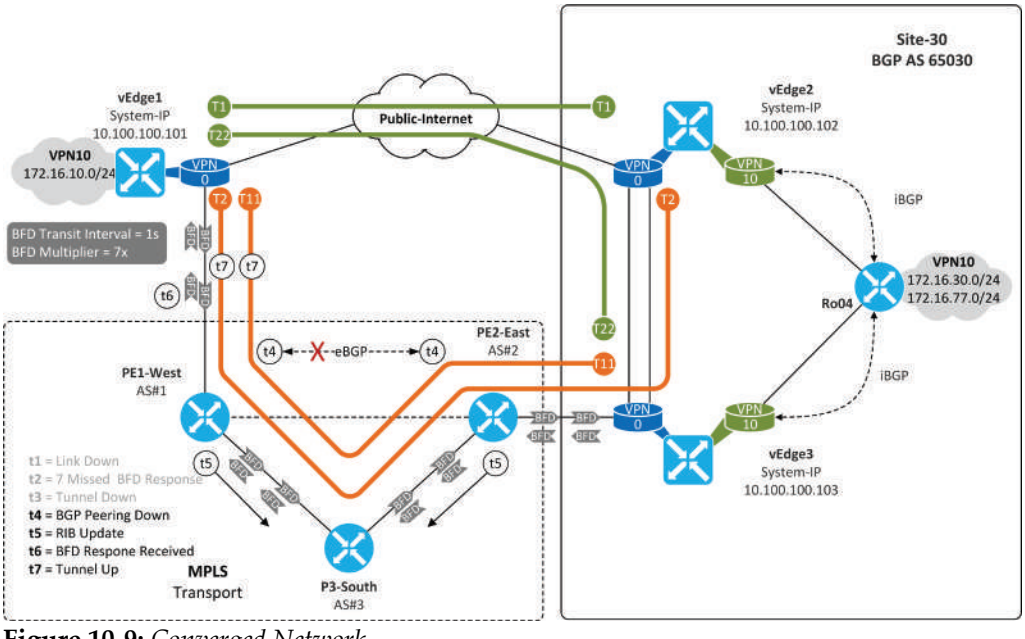


Figure 10-9: *Converged Network.*

Figure 10-10 verifies that vEdge1 has sent more BFD echoes than what it has received from the MPLS transport network (Destination IP 10.200.23.2 = MPLS color of vEdge2, destination (IP 10.200.1.103 = MPLS color of vEdge3).

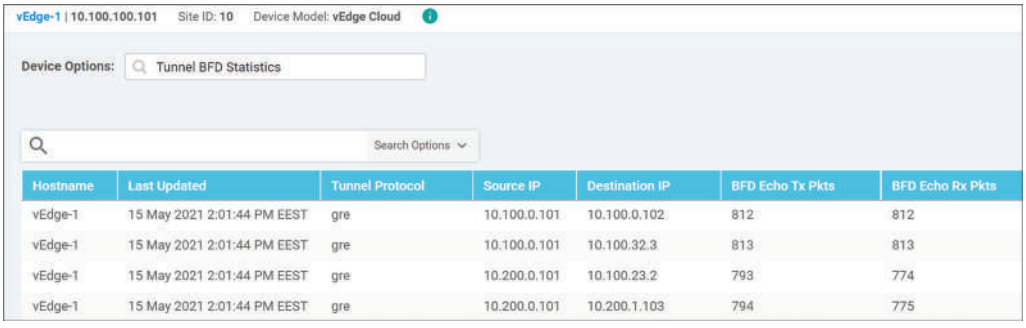


Figure 10-10: *BFD Statistics on vEdge-1.*

After convergence, we can see that all tunnels are up.

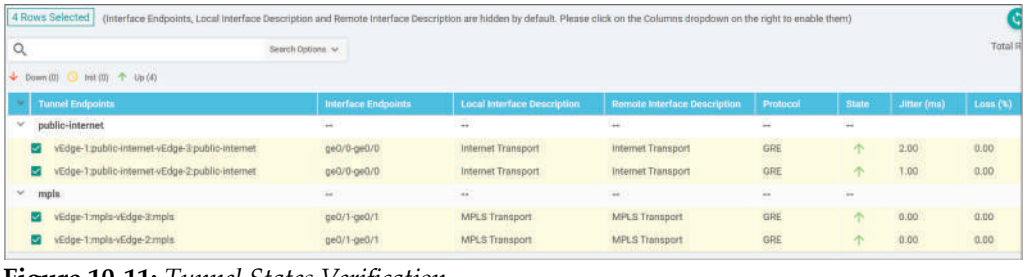


Figure 10-11: *Tunnel States Verification.*

Path Quality Monitoring

SLA-Class & Traffic Policy

This section explains how to configure SLA-Class. Navigate to *Configuration/Policies/Central Policies* page. Open the *Custom Options* drop-down menu and select the *List* option from the *Centralized Policy* section.

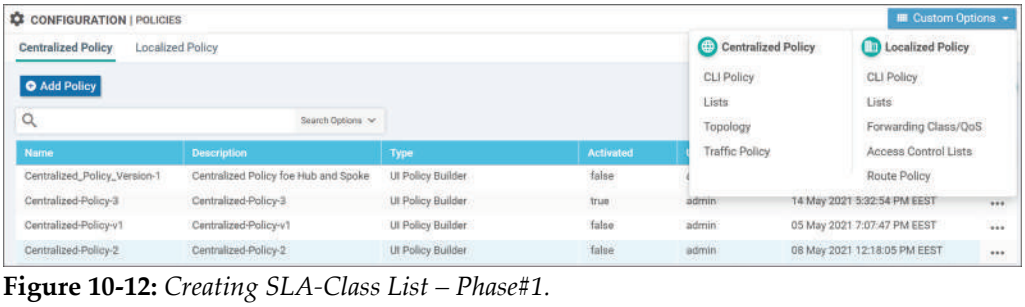


Figure 10-12: *Creating SLA-Class List – Phase#1.*

Choose the *SLA Class* from the list field. Give the name to list and set the Packet Loss, Latency, and Jitter threshold values and click the *Add* button.

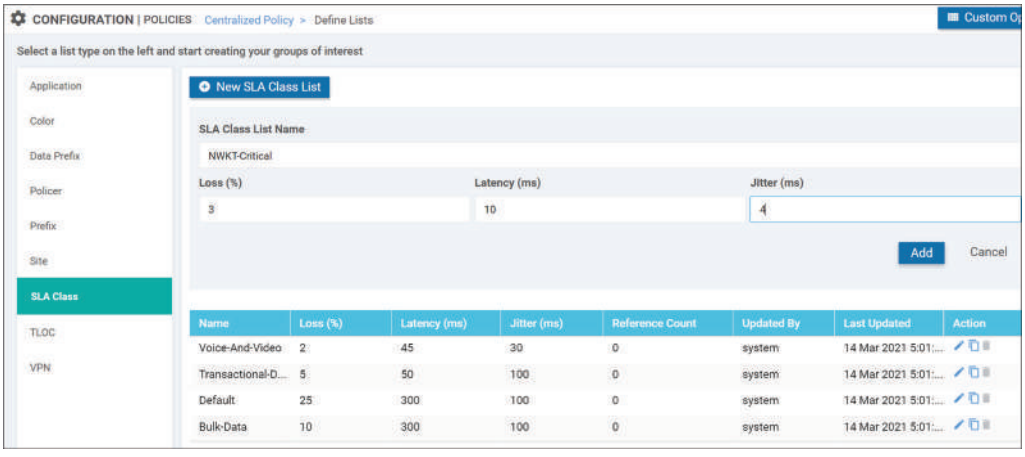


Figure 10-13: *Creating SLA-Class List – Phase#2.*

Example 10-14 shows that our new SLA Class NWKT_CRITICAL is listed among the default SLA Classes. Open the *Custom Options* drop-down menu again and select the *Traffic Policy* option from the *Centralized Policy* section. This leads you to the *Application-Aware Routing* window.

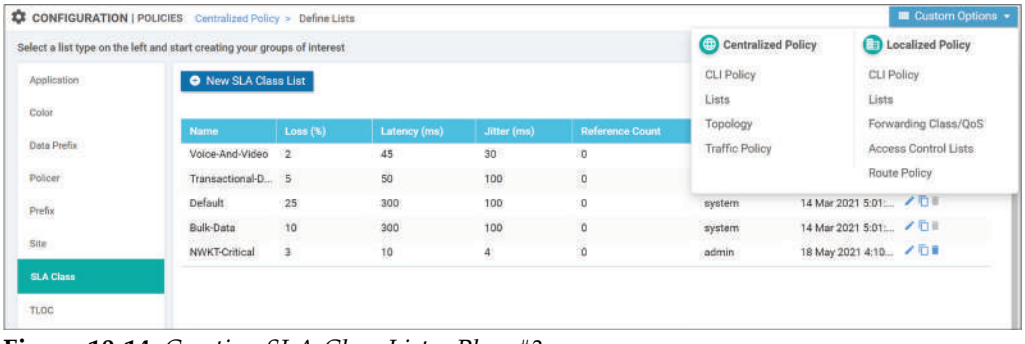


Figure 10-14: *Creating SLA-Class List – Phase#3.*

Click the *Add Policy* drop-down menu and select the only available option *Create New*.

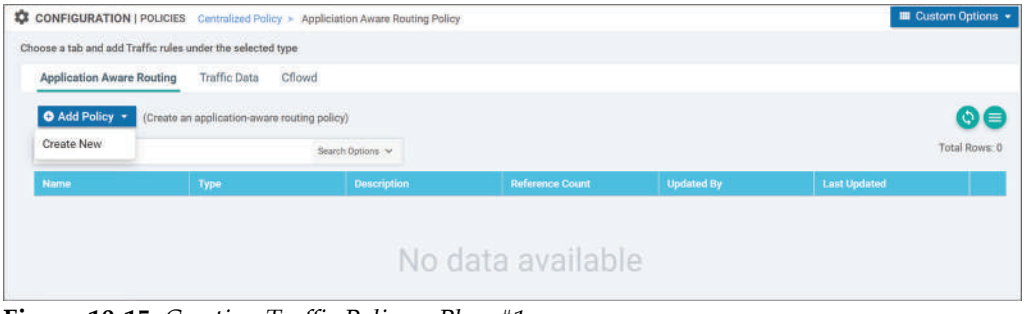


Figure 10-15: *Creating Traffic Policy – Phase#1.*

Click the *Sequence Type* button. Next, click the *Sequence Rule* button under the *App Route* section to create a new rule.

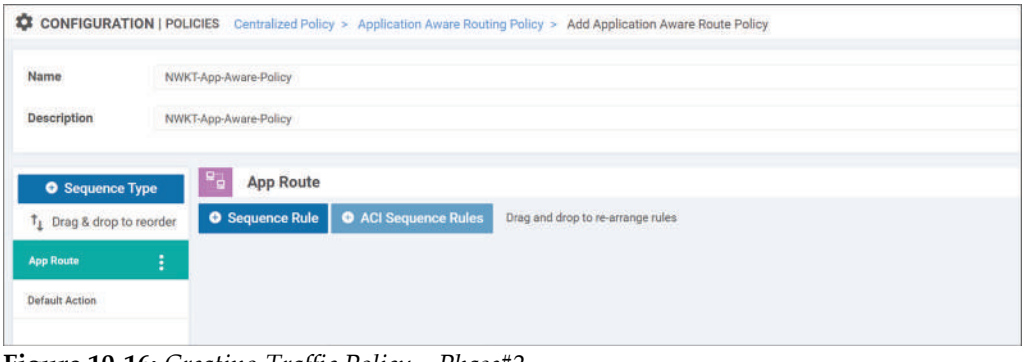


Figure 10-16: *Creating Traffic Policy – Phase#2.*

We want this policy to be applied to the application running on the subnet 172.16.30.0/24 and we also want to use the MPLS transport network as the primary path. We could (and should) also select an application but for simplicity, I am only using a subnet as an example. First, fill in the *Name* and *Description* fields. Select the *Destination Prefix* from the *Match* options and type the network address to the *Destination IP Prefix* field. Go to the *Action* section and select *NWKT-Critical* from the *SLA Class* menu. Besides, select the *mpls* as the *Preferred color*. Click the *Save Match And Actions* button.

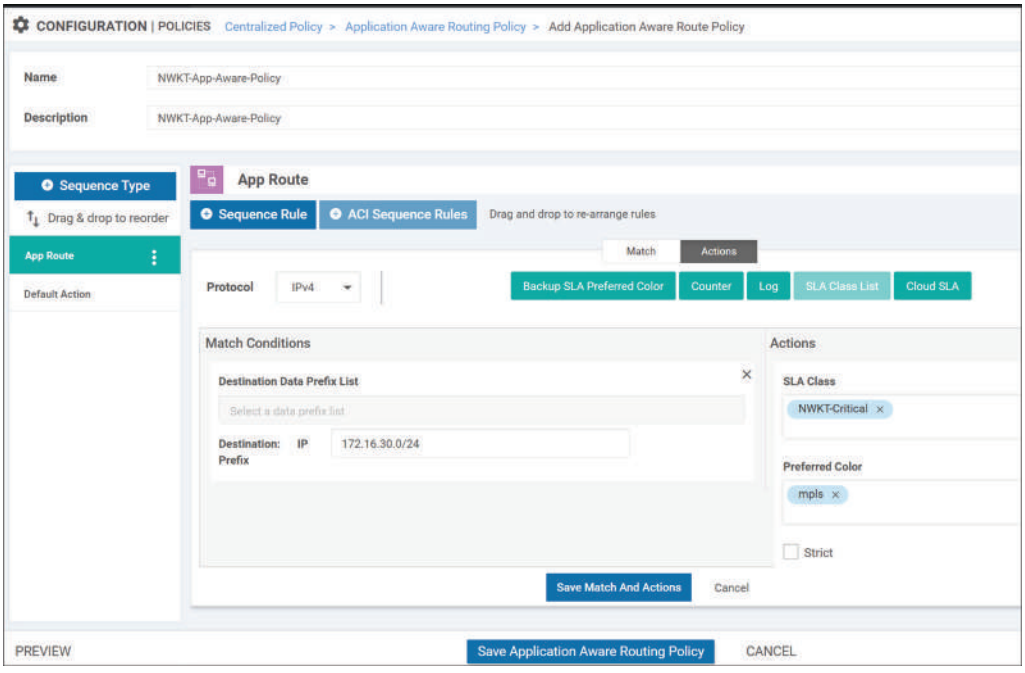


Figure 10-17: *Creating Traffic Policy – Phase#3.*

Figure 10-18 verifies our changes.

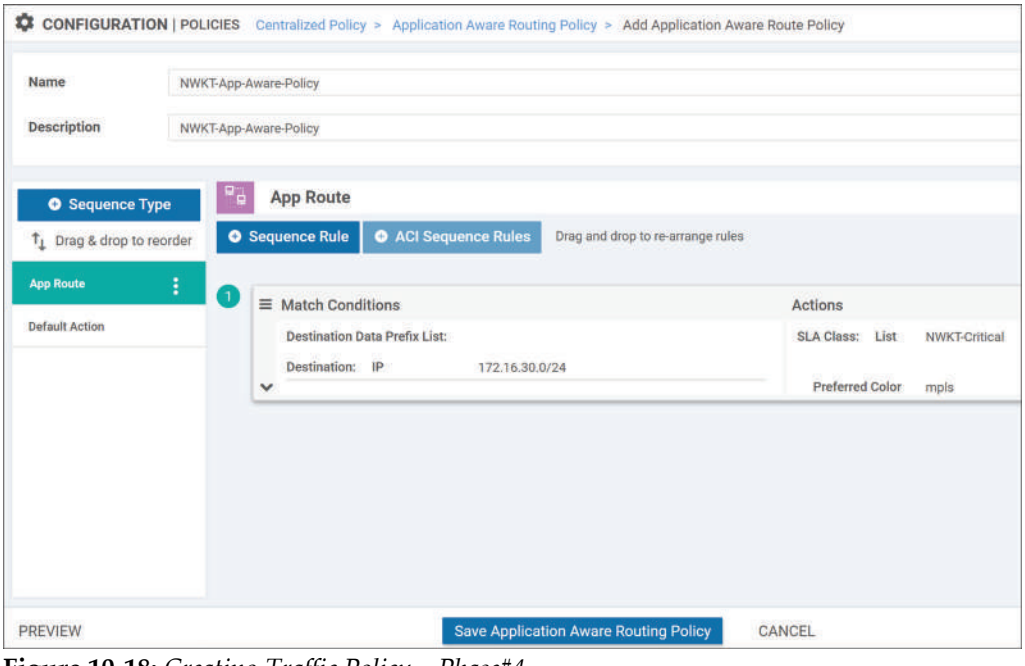


Figure 10-18: *Creating Traffic Policy – Phase#4.*

Click the *Save Application Aware Routing* Policy button. Now the new Traffic Policy is listed on the *Application Aware Routing* page.

Figure 10-19: *Creating Traffic Policy – Phase#5.*

Centralized Policy

As the last step, we need to apply the *Traffic Policy* into the *Centralized Policy*. Navigate back to the *Centralized Policy* window. We are currently using the policy Centralized-Policy-3, and we are going to add the new AAR Policy to it. Select the Edit option […].

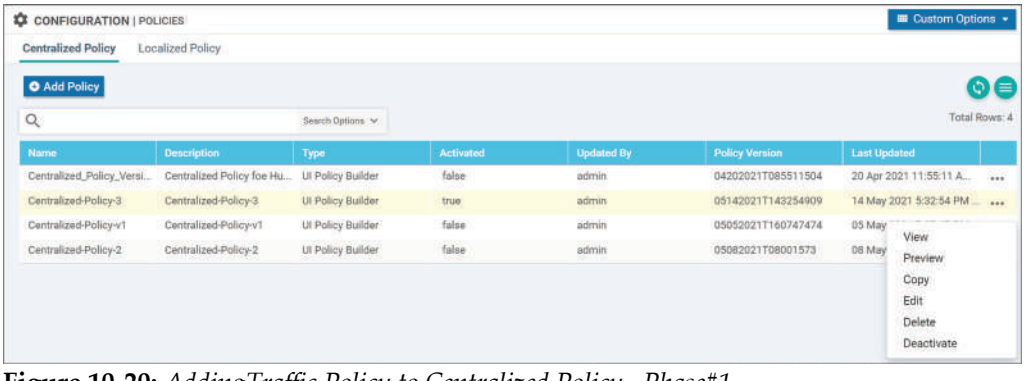


Figure 10-20: *AddingTraffic Policy to Centralized Policy– Phase#1.*

Select the *Traffic Rules* page and in the *Application Aware Routing* window select *Import Existing* from the *Add Policy* drop-down menu.

Figure 10-21: *AddingTraffic Policy to Centralized Policy– Phase#2.*

Select the *NWKT-App-Aware-Policy* from the *Policy* drop-down menu and click the *Import* button.

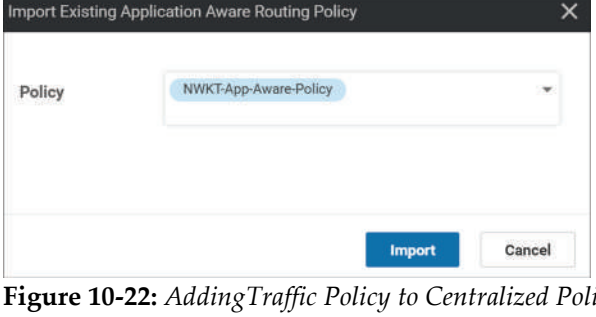


Figure 10-22: *AddingTraffic Policy to Centralized Policy– Phase#3.*

Figure 10-23 shows that our Traffic Policy is listed on the *Application Aware Routing* page.

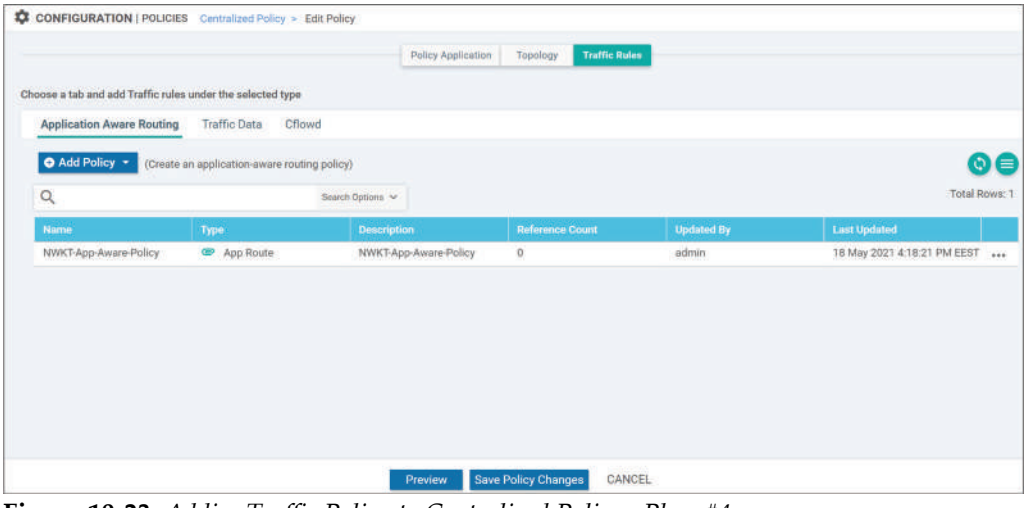


Figure 10-23: *AddingTraffic Policy to Centralized Policy– Phase#4.*

Next, select the *Policy Application* page. You can see that our previously created Traffic Rule *NWKT-App-Aware-Policy* is now shown on the *Application-Aware Routing* page. Now we need to attach the *Site List* and *VPN* to it. Click the *New Site List and VPN* button.

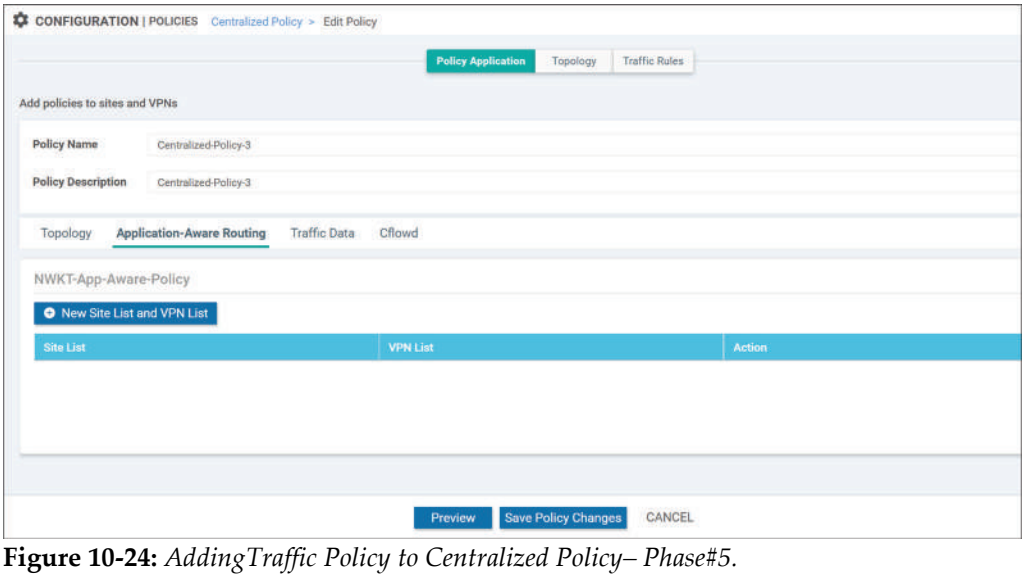


Figure 10-24: *AddingTraffic Policy to Centralized Policy– Phase#5.*

Select sites *DataCenter* and *Remote-Sites* from the *Select Site List* menu. Select *NWKT-VPN10* from the *Select VPN List* menu. Click the *Add* button (not shown in the figure). Note that I have configured the VPN list beforehand (VPN10/NWKT-VPN10) just like any other list even though the configuration is not shown.

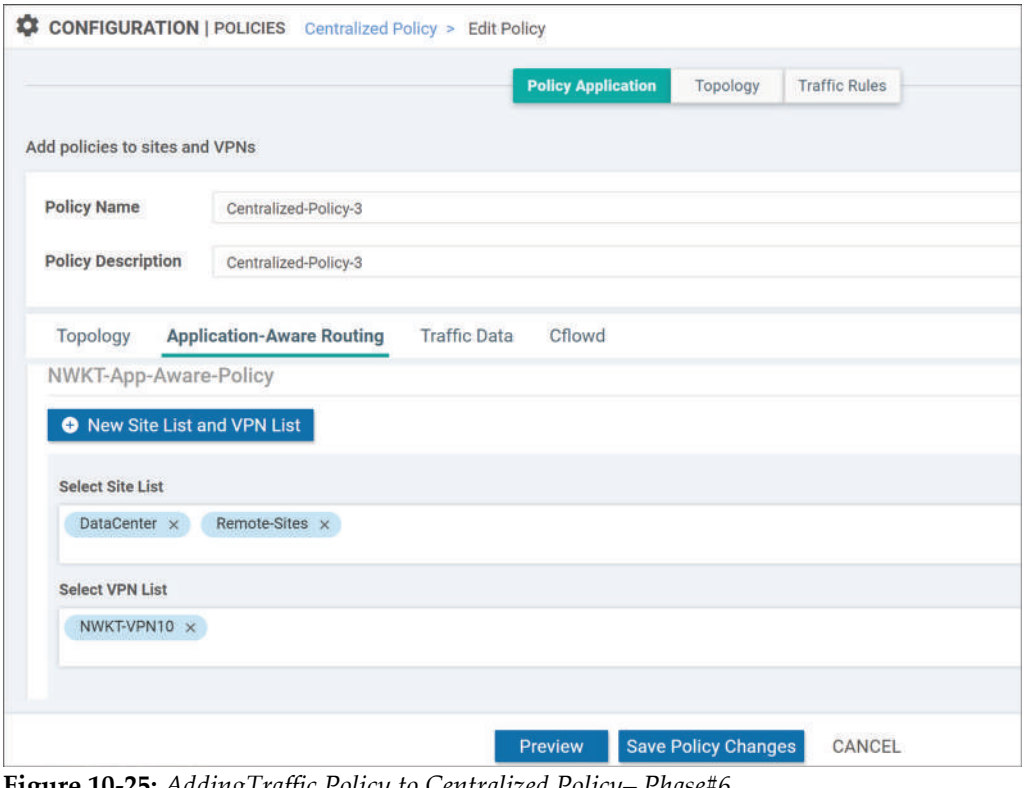


Figure 10-25: *AddingTraffic Policy to Centralized Policy– Phase#6.*

Figure 10-26 shows that our site lists and VPN 10 list are attached to NWKT-App-Aware-Policy.

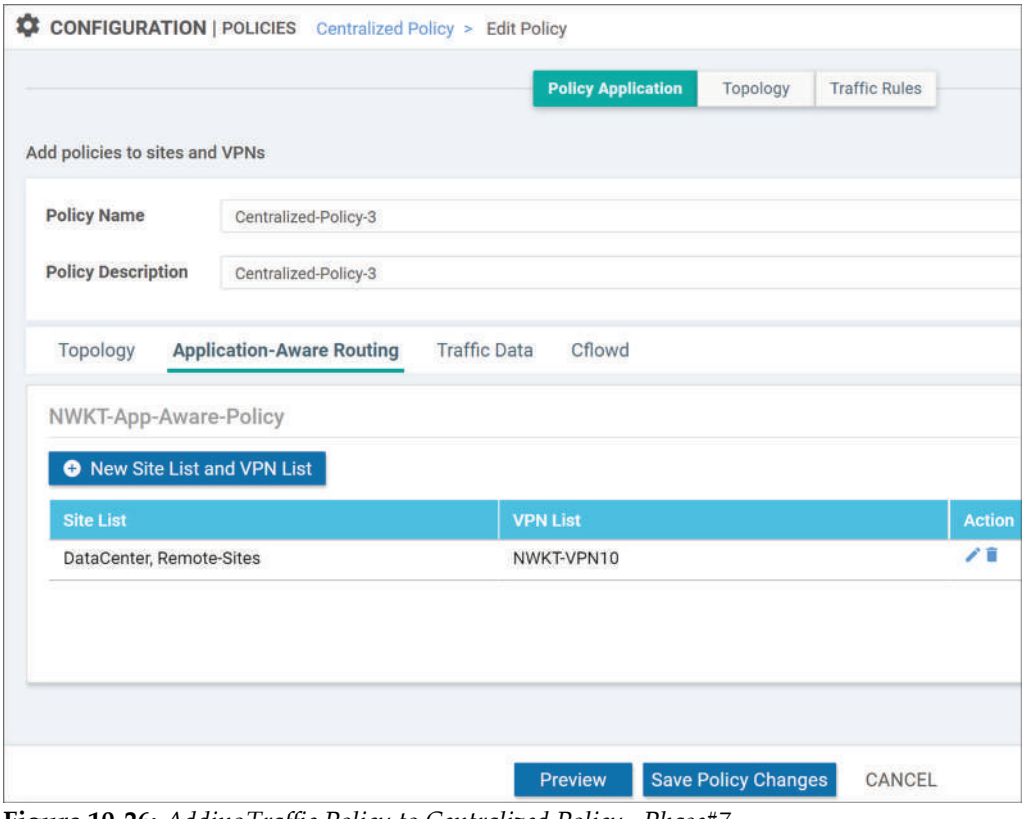


Figure 10-26: *AddingTraffic Policy to Centralized Policy– Phase#7.*

Before applying the policy you can check the policy configuration that will be sent to vSmart by clicking the *Preview* button. Example 10-3 shows the complete policy configuration. We have a ***sla-class*** *NWKT-Critical* where we set the thresholds for Packet Loss, Jitter, and Delay. Then we have ***app-route-policy*** which name is a combination of the VPN List name *NWKT-VPN* and the traffic rule policy name *NWKT-App-Aware-Policy*. This policy states that traffic towards subnet 172.16.30.0/24 within VPN10 will be sent over the tunnel going through the MPLS transport network. Then the app-route-policy is added to both site-list Remote-Sites and DataCenter. Note that we already have an existing Control Policy attached to the site list DataCenter that is used for preventing routing updates about site 30 internal networks from the WAN side.

```
viptela-policy:policy
  sla-class NWKT-Critical
   latency 10
   loss 3
   jitter 4
  !
 control-policy Route-Policy-to-DC
    sequence 1
     match route
      prefix-list Site30-Prefix-List
     !
     action reject
     !
    !
  default-action accept
 !
 app-route-policy _NWKT-VPN10_NWKT-App-Aware-Policy
  vpn-list NWKT-VPN10
    sequence 1
     match
      destination-ip 172.16.30.0/24
     !
     action
      sla-class NWKT-Critical  preferred-color mpls
     !
    !
 !
 lists
  prefix-list Site30-Prefix-List
   ip-prefix 172.16.30.0/24
  !
  site-list DataCenter
   site-id 30
  !
  site-list Remote-Sites
   site-id 10
   site-id 20
  !
  vpn-list NWKT-VPN10
   vpn 10
  !
 !
!
```

```
apply-policy
 site-list Remote-Sites
  app-route-policy _NWKT-VPN10_NWKT-App-Aware-Policy
 !
 site-list DataCenter
  control-policy Route-Policy-to-DC out
  app-route-policy _NWKT-VPN10_NWKT-App-Aware-Policy
 !
!
```

Example 10-3: *Complete Centralized Policy Configuration.*

After verifying the configuration, click the *Save Policy Changes* button and in *Activate Policy* pop-up window, click the *Activate* button.

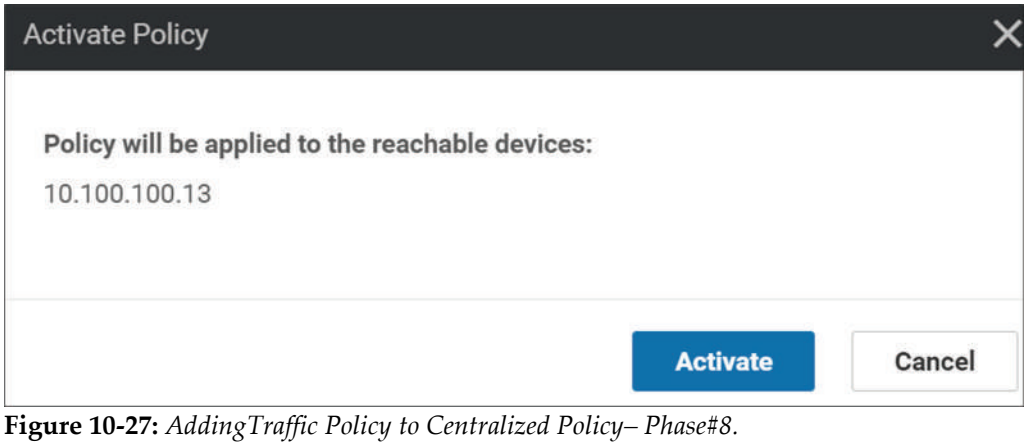


Figure 10-27: *AddingTraffic Policy to Centralized Policy– Phase#8.*

Figure 10-28 shows the progress of a policy activation process.

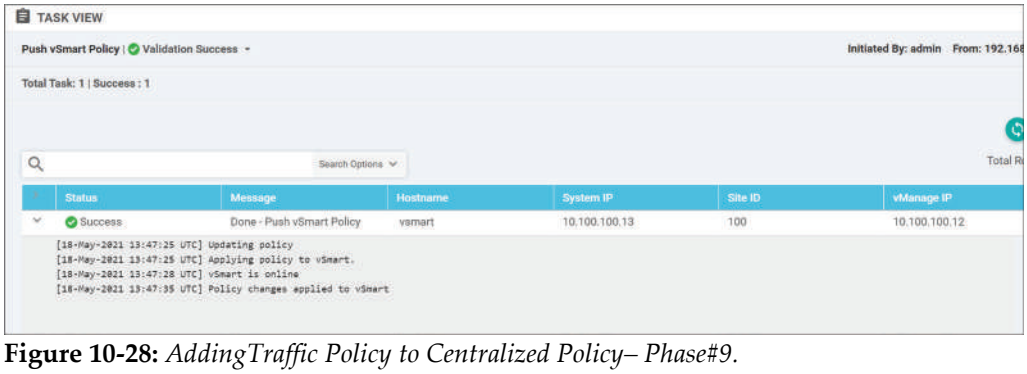


Figure 10-28: *AddingTraffic Policy to Centralized Policy– Phase#9.*

Chapter 11: Direct Cloud Access

Introduction

This chapter explains how a local Internet connection on remote sites can be used for accessing services hosted in public clouds. Figure 11-1 illustrates the example topology which we are going to use in this chapter. In vEdge-1, we have a Data-Policy that states that traffic from the VPN 10 subnet 172.16.10.0/24 to 10.100.0.13/32 (vSmart) will be sent over VPN 0. The traffic will be examined and allowed by the local Zone-Based Firewall (ZBF) and then routed to the Internet using NAT. All other traffic flows will be forwarded through the overlay tunnels using flow-based load-balancing. The main difference from the routing perspective between Direct Cloud Access (DCA) and Direct Internet Access (DIA) is that the default route points straight to the Internet in the DIA solution while in DCA we strictly define which subnets, IP addresses, or application traffic are using the local Internet. Note that in the DIA solution good practice is to filter out traffic flows towards Bogon IP addresses.

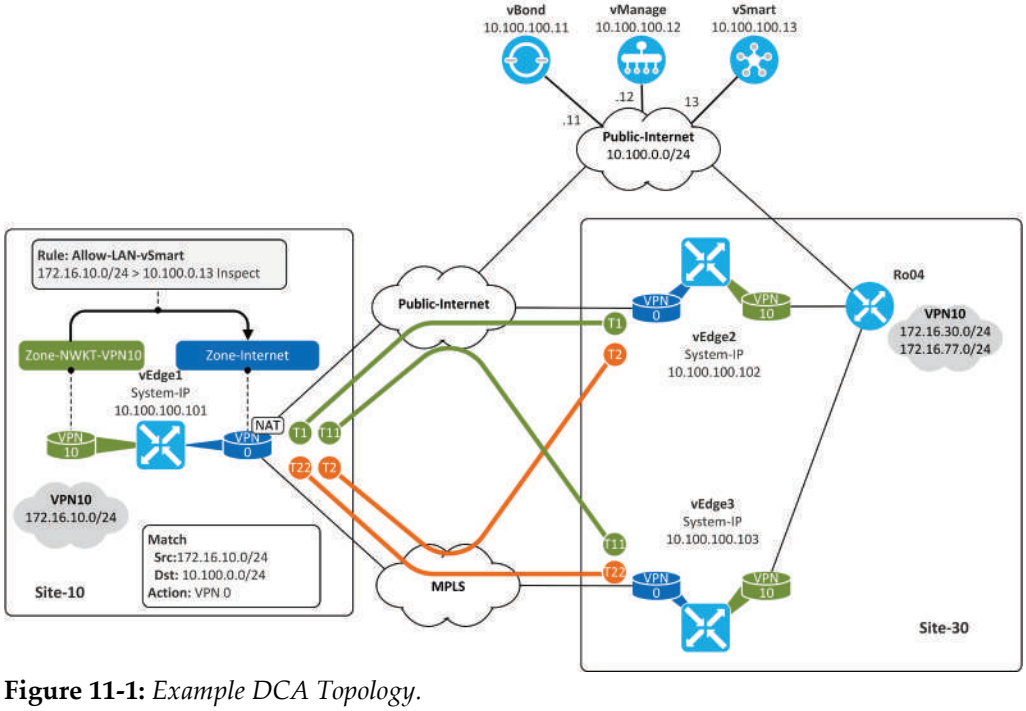


Figure 11-1: *Example DCA Topology.*

Data Policy

Building Blocks

Our example ***data-policy*** *_NWKT-VPN10_Direct-Cloud-Access* applies to Service VPN bind to ***vpn-list*** *NWKT-VPN10*. Traffic flows from the subnet 172.16.10.0/24 to the destination listed in the ***data-prefix-list*** *Cloud-Services* will be routed out from the VPN 0 using the NAT. The data-policy is added to vEdges listed in ***site-list*** *Remote-Sites*. Note that the Firewall policy is not part of the Data-Policy, it has its policy.

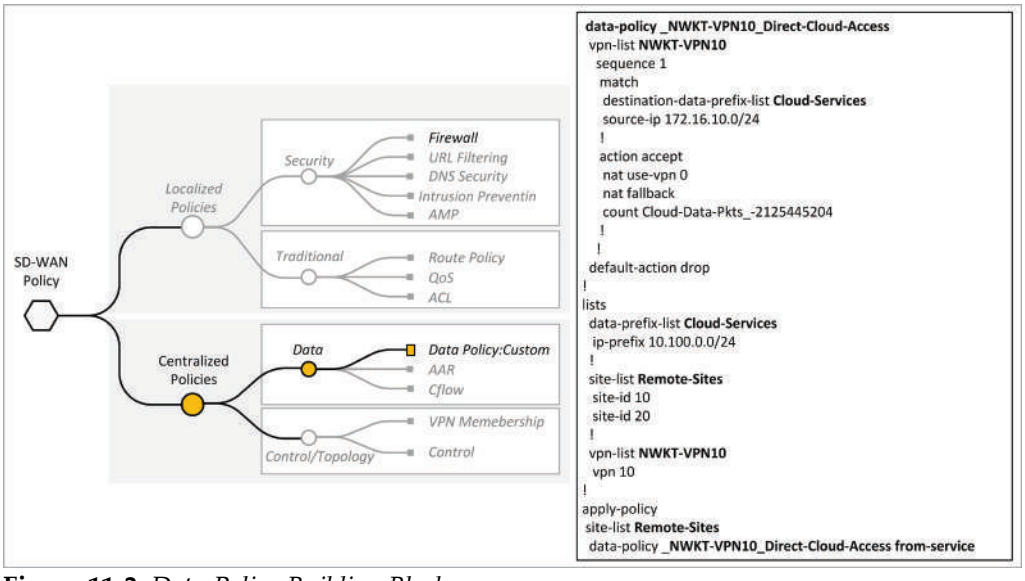


Figure 11-2: *Data-Policy Building Blocks.*

By simulating traffic flows from VPN 10 IP address 172.16.10.1 to destination IP 10.100.0.1 we can verify that without data policy vEdge-1 uses all four tunnels for flow-based load-balancing when forwarding data packets.

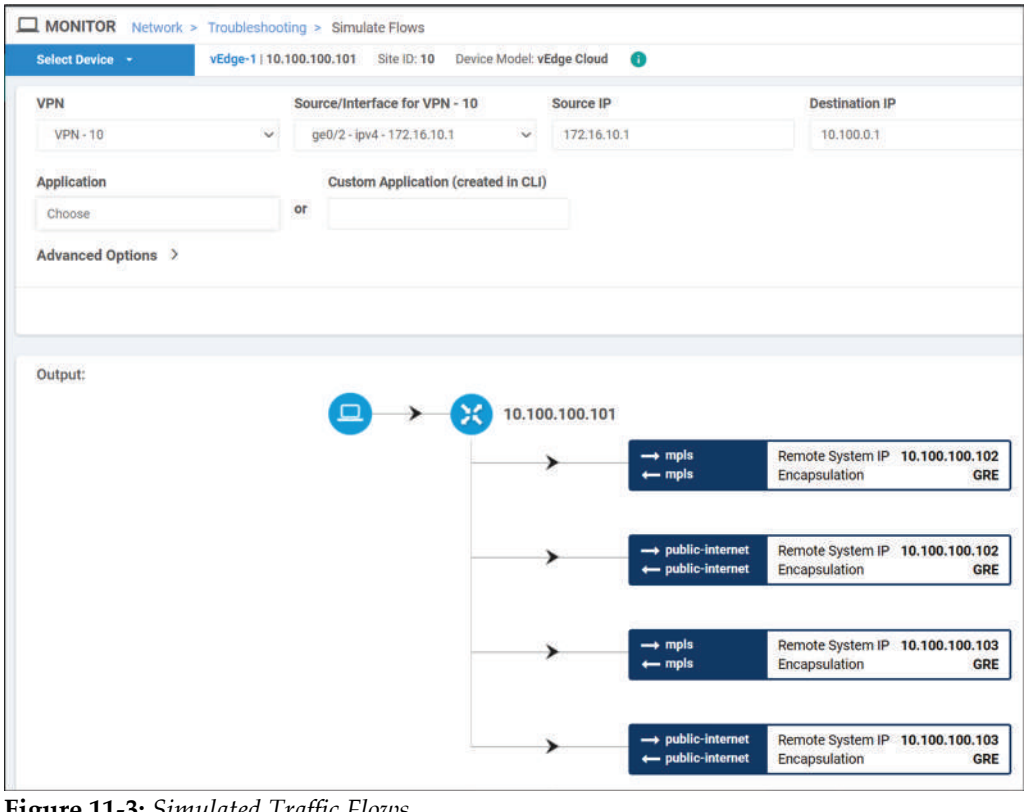


Figure 11-3: *Simulated Traffic Flows.*

Configuration Data Prefix List

Before starting to set up the data policy we need to configure the *Data Prefix List* where we define the cloud service subnets. Go to the *Configuration/Policies* page and open the *Custom Option* drop-down menu and select the *List* option under the *Centralized Policy*. This list is later applied to Data Policy. Note that *VPN List* for VPN 10 and *Site Lists* for Branch sites are already configured in previous chapters.

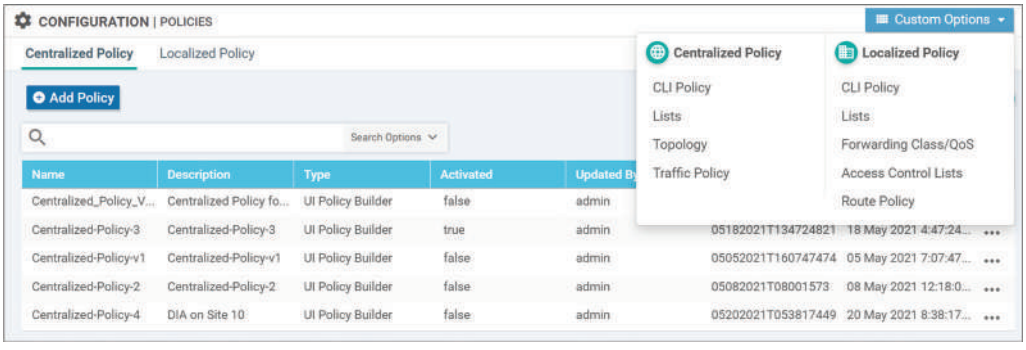


Figure 11-4: *Data Prefix List Configuration.*

Figure 11-5 shows the configured Data Prefix list Cloud-Service that has IP prefix 10.100.0.0/24 assigned to it.

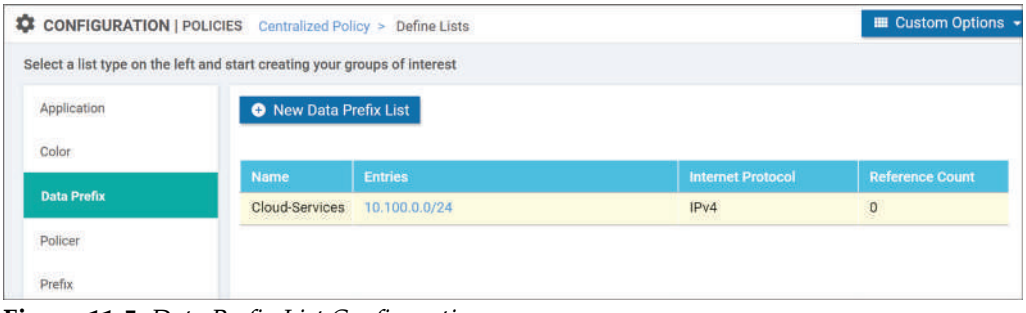


Figure 11-5: *Data Prefix List Configuration.*

Configuration Data Policy

Navigate back to the *Configuration/Policies* page and open the *Custom Option* drop-down menu and select the *Traffic Policy* option under the *Centralized Policy*.

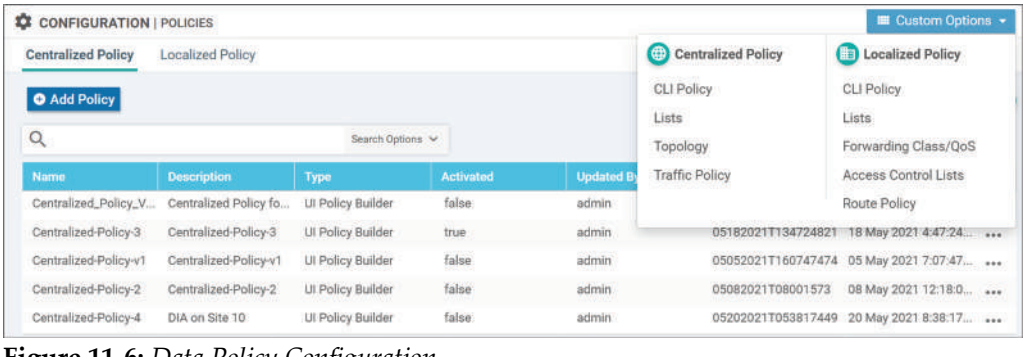


Figure 11-6: *Data Policy Configuration.*

Go to the *Traffic Data* page and select the *Create New* option from the *Add Policy* drop-down menu.

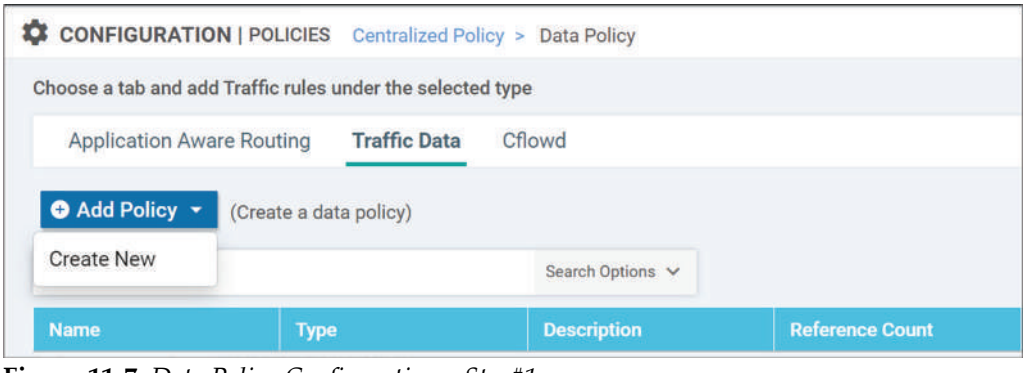


Figure 11-7: *Data Policy Configuration – Step#1.*

Click *Sequence Type* and select the *Custom* option from the *Add Data Policy* pop-up window.

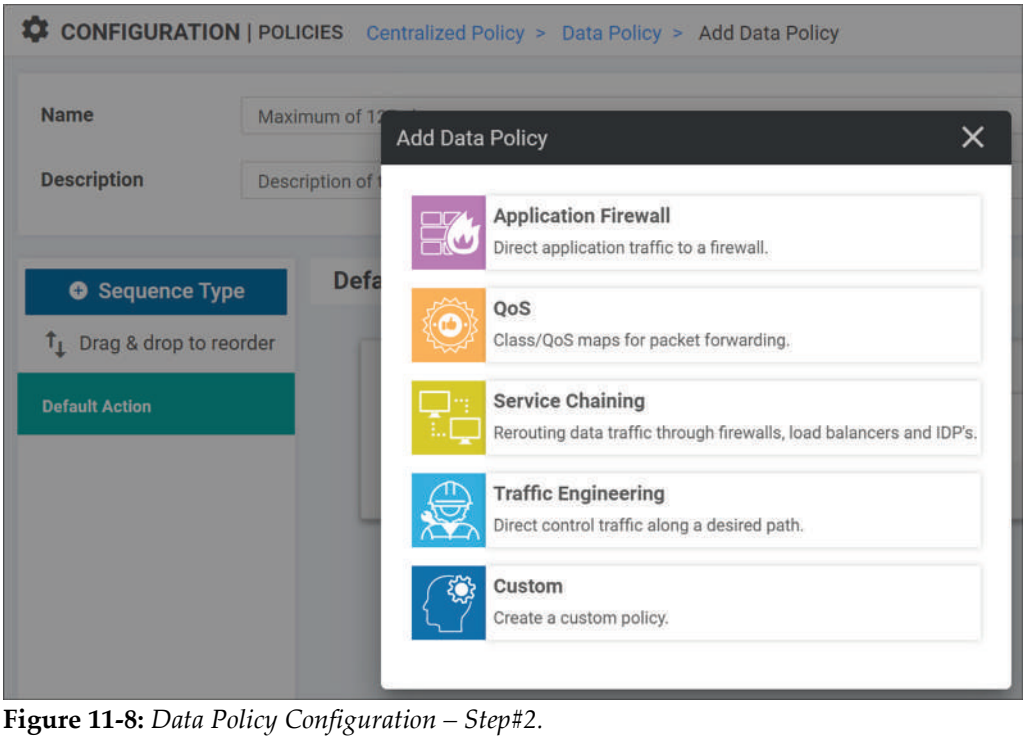


Figure 11-8: *Data Policy Configuration – Step#2.*

Fill in the *Name* and *Description* fields. Click the *Sequence Rule* button and select the previously created *Cloud-Service* Data Prefix List as a destination and set the source IP Prefix manually. Note that we could also use Data Prefix List as a source. Set the *NAT* as an action. You can't use NAT VPN with any other VPN Id than 0. The *Fallback* option allows the destination cloud service also to be used via Hub site if vEdge loses its Internet local connection. The Action Counter creates a counter that is used for tracking packets that match this Data Policy. When the Match and Action conditions are set, click the *Save Match And Actions* button.

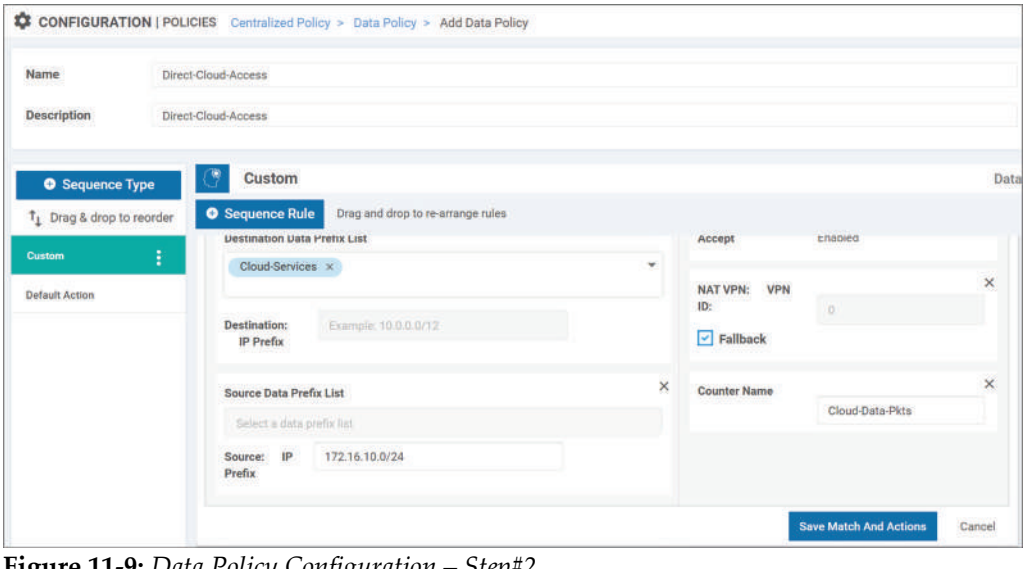


Figure 11-9: *Data Policy Configuration – Step#2.*

Figure 11-10 shows the complete Data Policy. Click the *Save Data Policy* button.

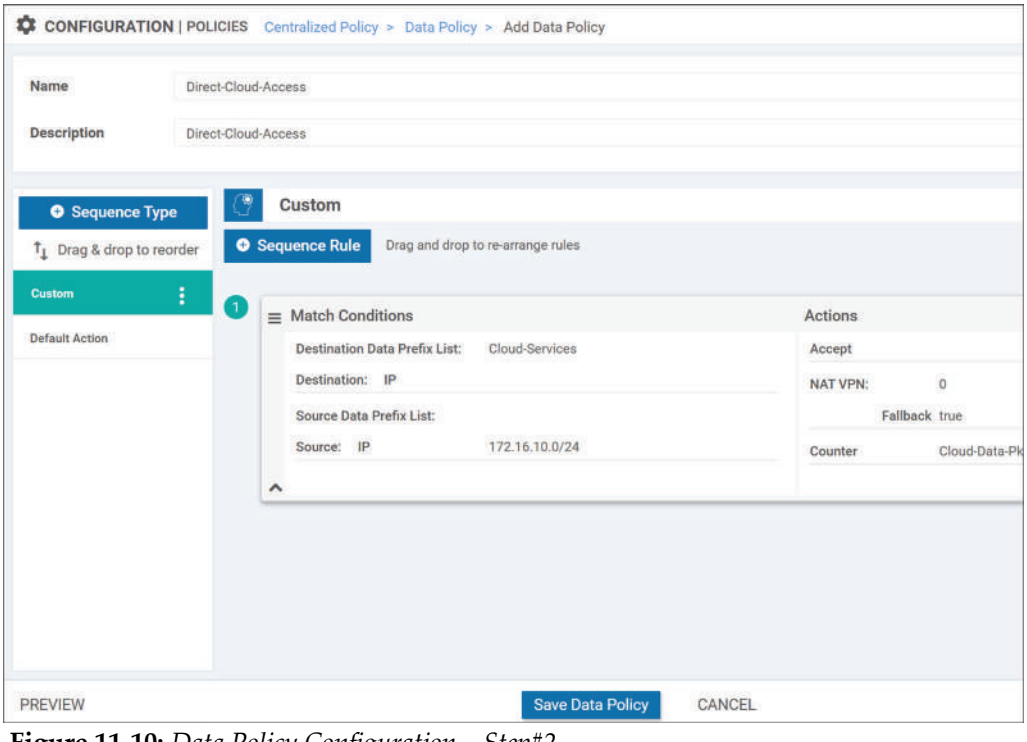


Figure 11-10: *Data Policy Configuration – Step#2.*

Set the Default Action to reject.

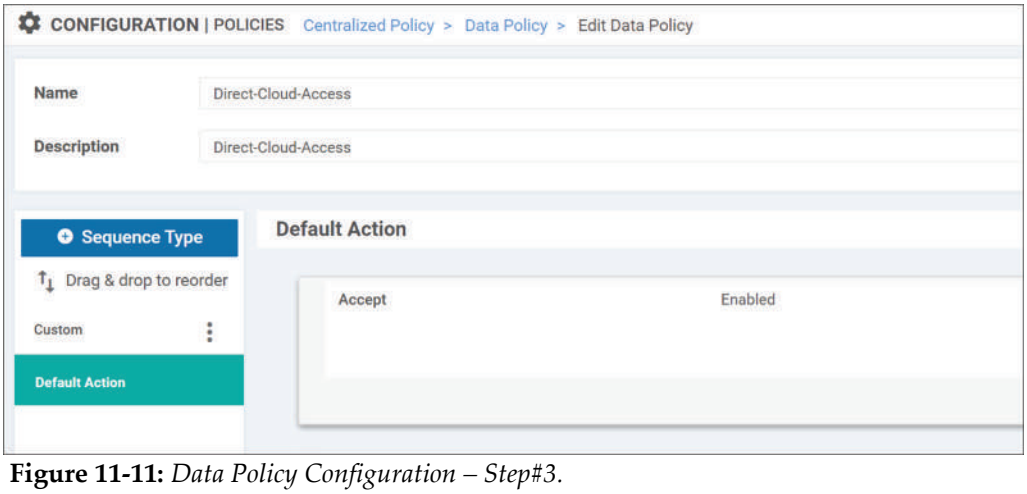


Figure 11-11: *Data Policy Configuration – Step#3.*

Figure 11-12 shows that our Data Policy is now listed on the *Traffic Data* page.

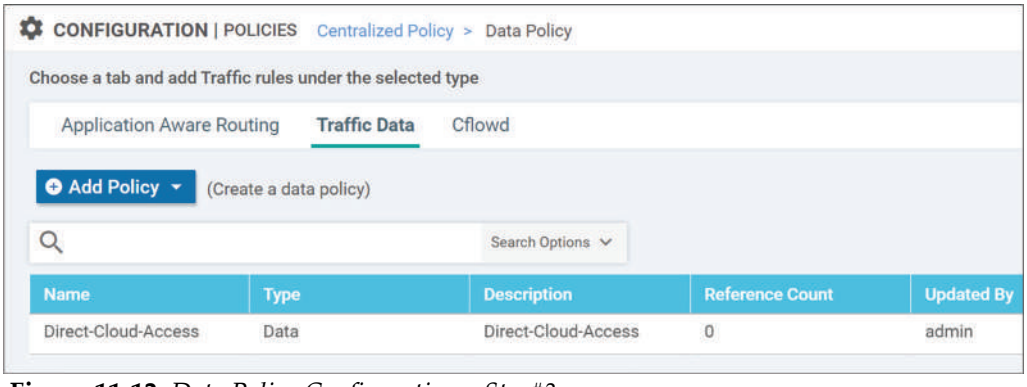


Figure 11-12: *Data Policy Configuration – Step#3.*

Applying the Data Policy to Centralized

You could add the Data Policy to the existing Centralized Policy or you can make a copy from the currently activated Centralized Policy and then add the new Data Policy to it, which is exactly what we are going to do. Open the Options menu […] at the end of the Centralized which you are going to copy and select the *Copy* option.

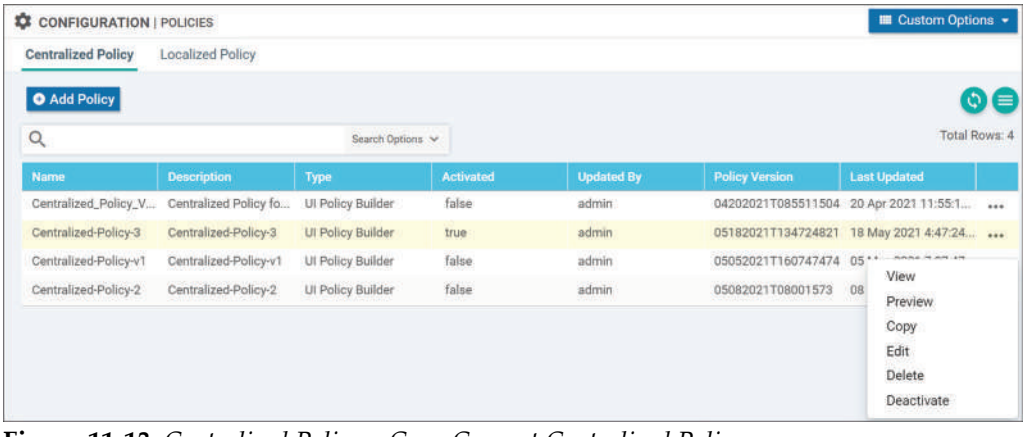


Figure 11-13: *Centralized Policy – Copy Current Centralized Policy.*

Select the Edit option from the copied Centralized Policy and choose the Edit option (figure 11-14).

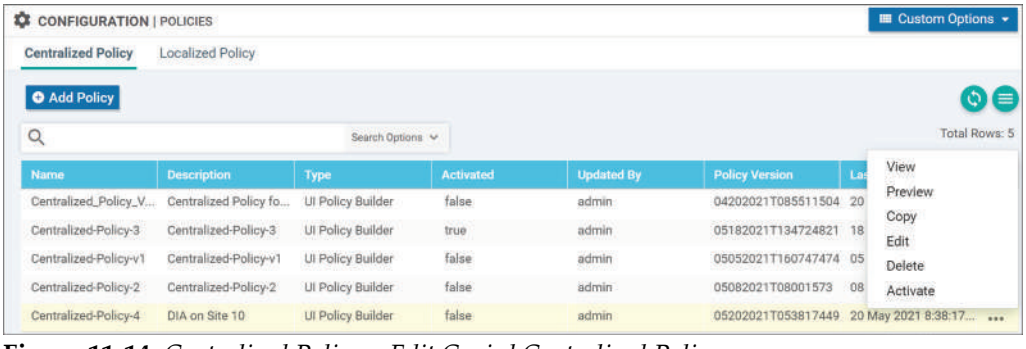


Figure 11-14: *Centralized Policy – Edit Copied Centralized Policy.*

Select the *Traffic Data* option from the Traffic *Rule Page* and select the *Import Existing* option from the *Add Policy* drop-down menu.

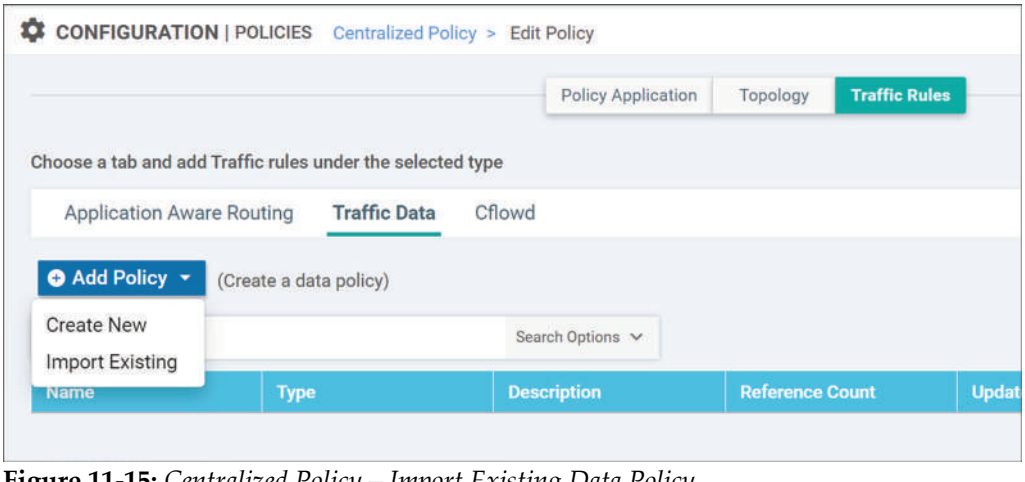


Figure 11-15: *Centralized Policy – Import Existing Data Policy.*

Select our previously created Data Policy *Direct-Cloud-Access* from the Policy drop-down menu and click the Import button.

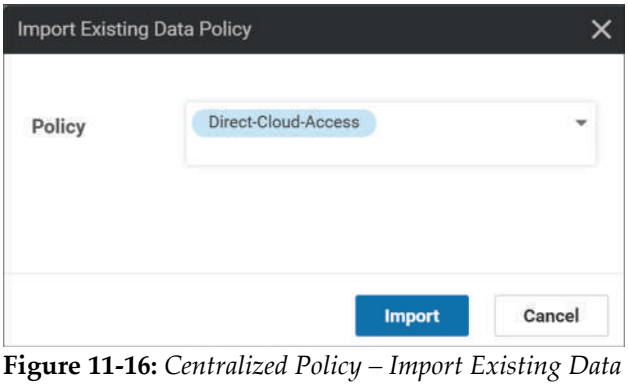


Figure 11-16: *Centralized Policy – Import Existing Data Policy.*

The Data Policy *Direct-Cloud-Access* is now listed on the *Traffic Data* page.

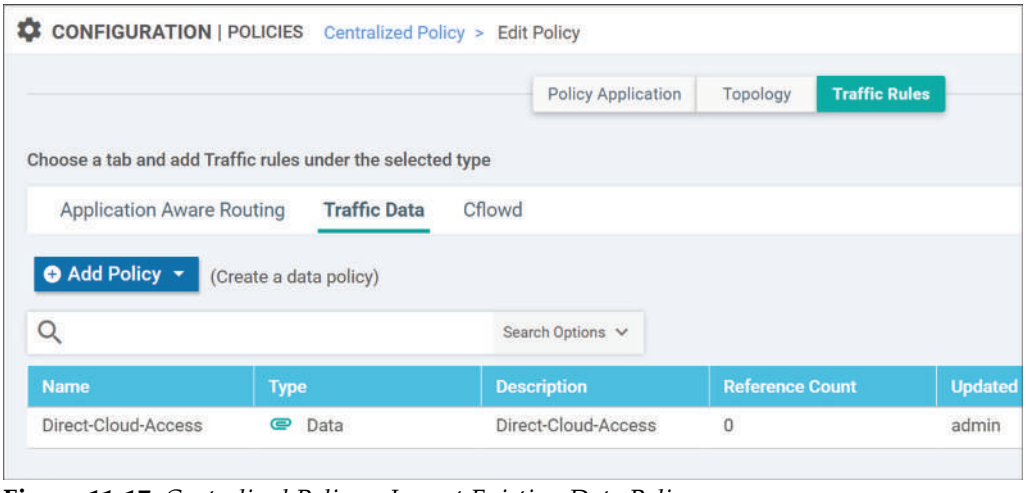


Figure 11-17: *Centralized Policy – Import Existing Data Policy.*

Click the *Policy Application* button and select *Traffic Data,* then click the *New Site List and VPN List* button.

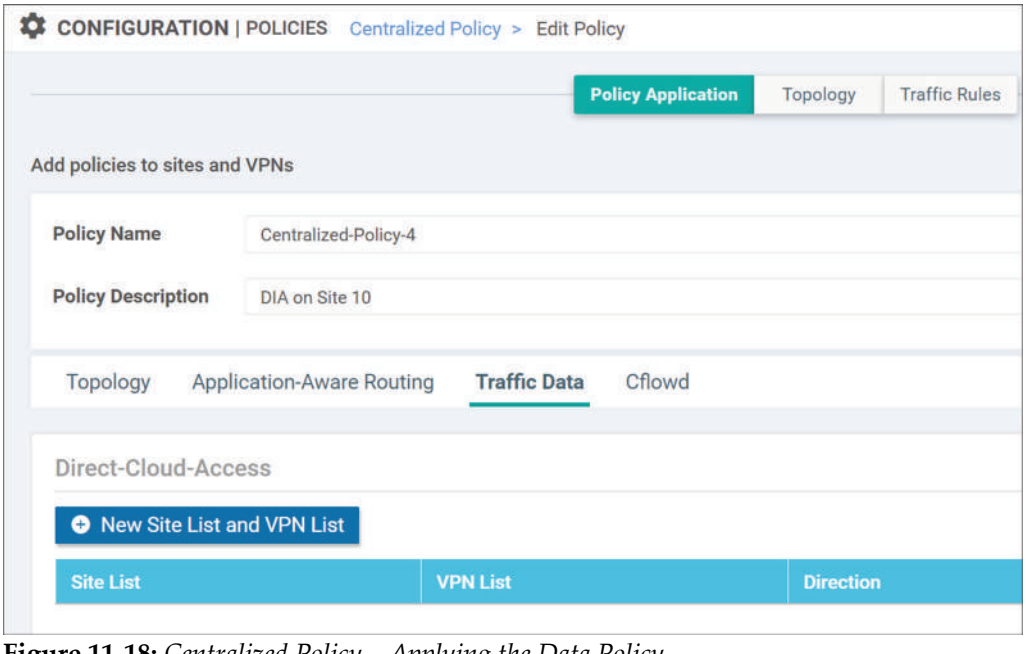


Figure 11-18: *Centralized Policy – Applying the Data Policy.*

Select pre-configured Site List *Remote-Sites* that includes sites 10 and 20. Select also pre-configured VPN List *NWKT-VPN10* that includes our Service VPN 10. No matter what Policy we are applying to Centralized Policy, we need to define the direction. The direction in our example is from VPN10 (Service VPN) to the Internet so the direction is *From Service*. Click the *Add* button (not shown in figure 11-19). When done, you can either preview the configuration or save policy changes.

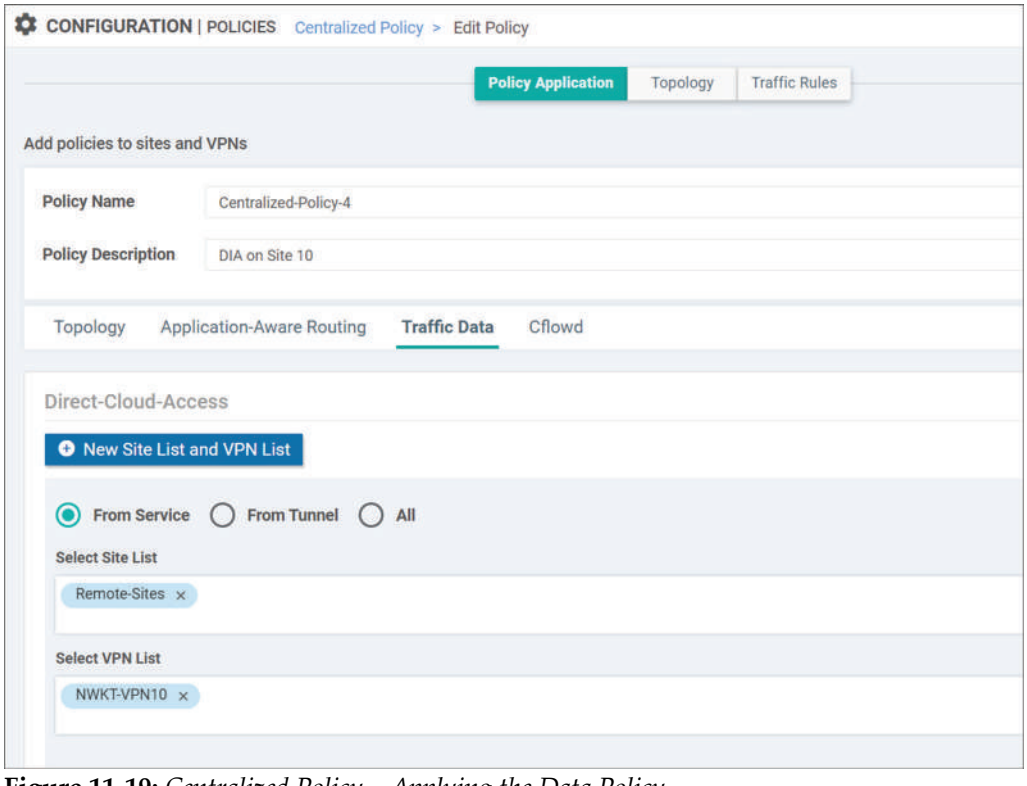


Figure 11-19: *Centralized Policy – Applying the Data Policy.*

As the last step, we need to deactivate the current Central Policy and then activate the new one. Figures from 11-20 to 11-24 illustrate the process.

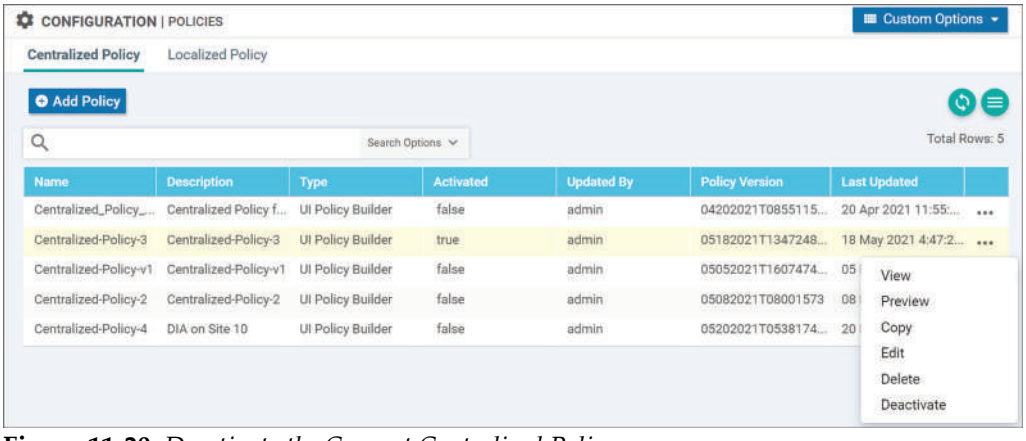


Figure 11-20: *Deactivate the Current Centralized Policy.*

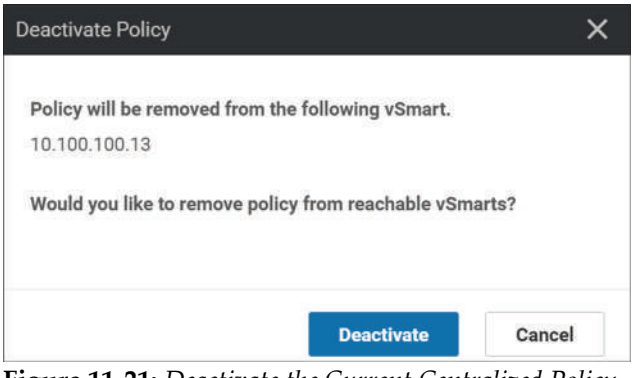


Figure 11-21: *Deactivate the Current Centralized Policy - Confirmation.*

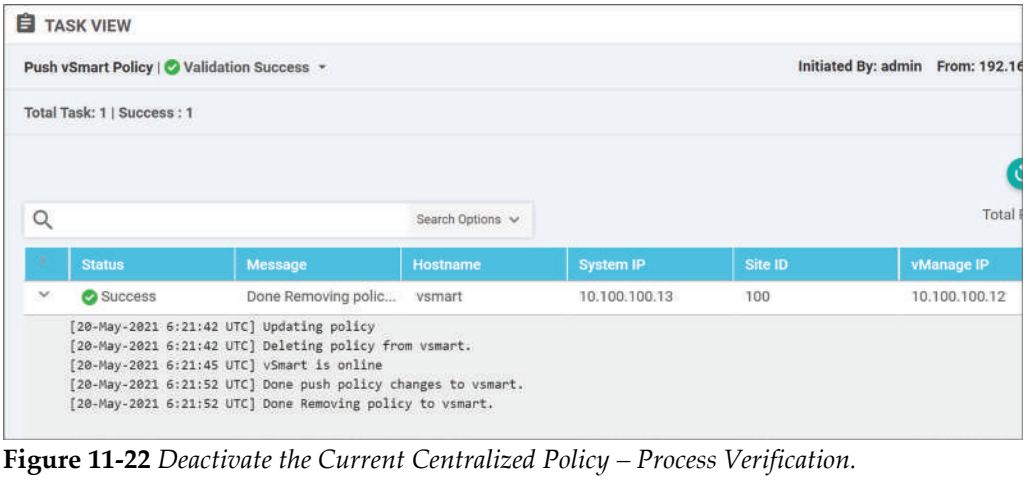


Figure 11-22 *Deactivate the Current Centralized Policy – Process Verification.*

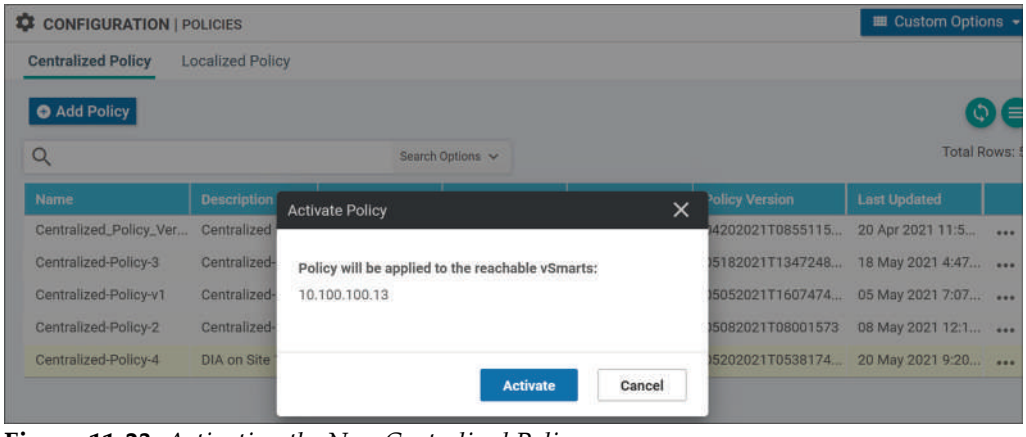


Figure 11-23: *Activating the New Centralized Policy.*

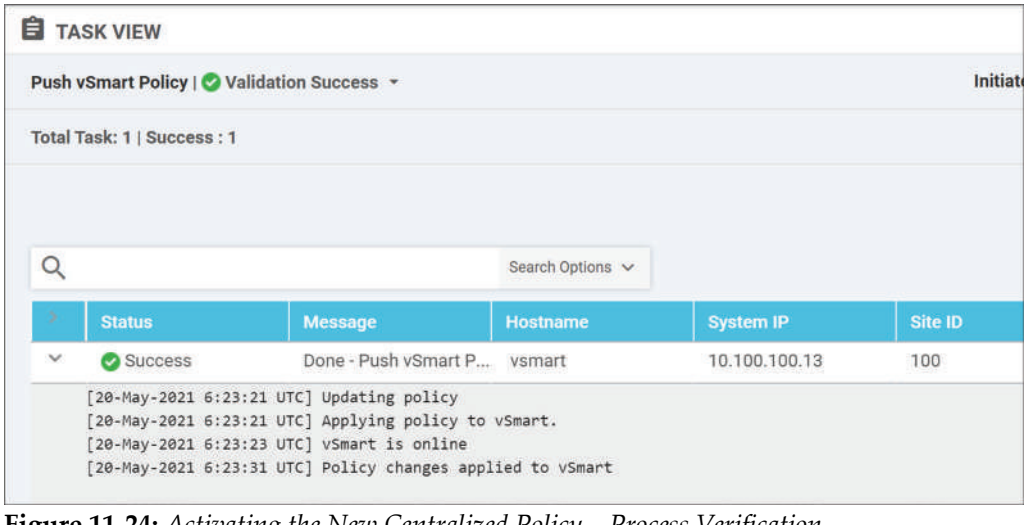


Figure 11-24: *Activating the New Centralized Policy – Process Verification.*

Data Policy Verification

By simulating traffic flows from VPN 10 IP address 172.16.10.1 to destination IP 10.100.0.1 we can verify that our Data Policy is working as expected (figure 11-25), the traffic is switched out from the interface ge0/0, which is the underlay connection to the Internet. Figure 11-24 in turn shows that traffic towards subnet 172.16.30.0/24 on site 30 is still tunneled. The reason why the traffic only goes over the MPLS transport is that the App-Route Policy we applied in chapter 10 is still effective.

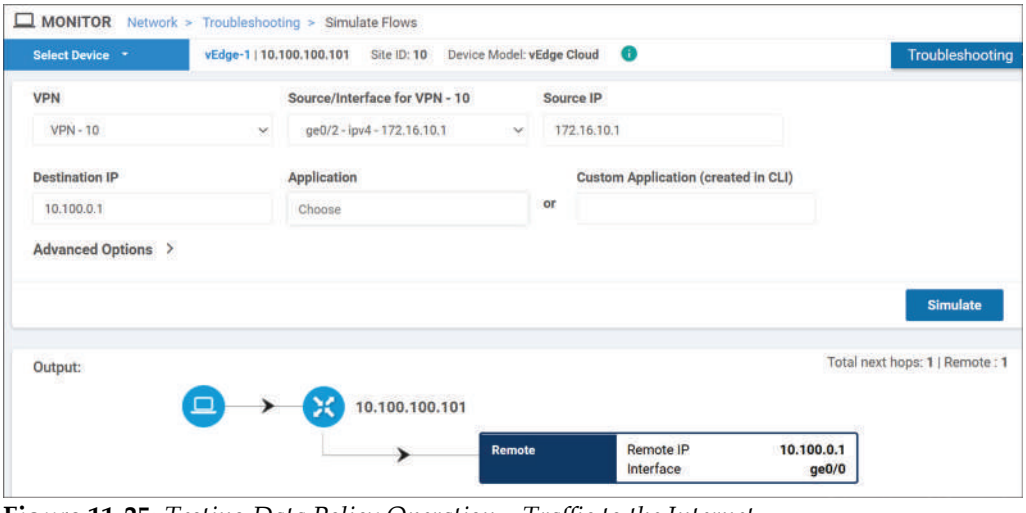


Figure 11-25: *Testing Data Policy Operation – Traffic to the Internet.*

Figure 11-26: *Testing Data Policy Operation – Traffic to the Site 30.*

You need to enable NAT (shown in figure 11-27) in the *Feature Template* attached to the *Device Template* that vEdge-1 is using. Without NAT the new Data Policy is not working and all the traffic from VPN10 to vSmart is forwarded to tunnels. The Device Template configuration is installed into vEdge devices while the Data Policy as a part of the Centralized Policy is installed into vSmart, not to vEdges.

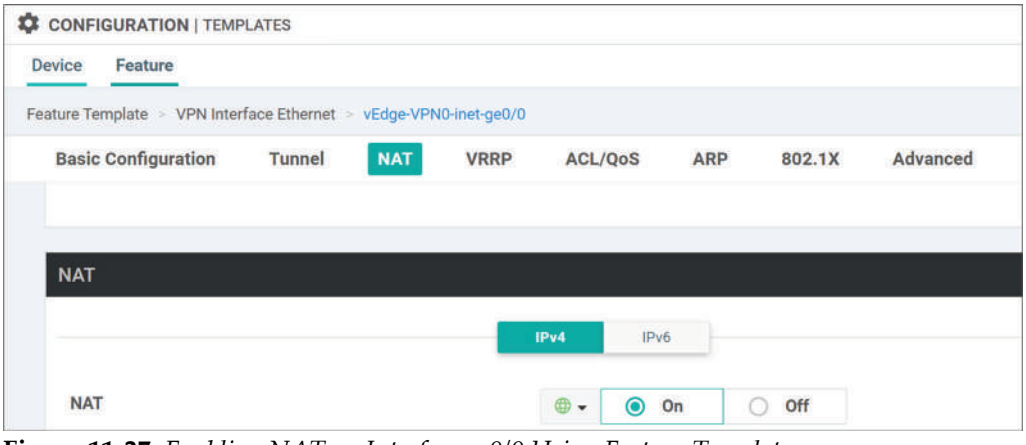


Figure 11-27: *Enabling NAT on Interface ge0/0 Using Feature Template.*

Zone-Based Firewall

Building Blocks

Zone-Based Firewalls will be implemented to vEdges just like any other Security Policies. That is why it is listed in the figure as *Localized Policy*. The Data Policy introduced in the previous section in turn is a Centralized Policy because it is applied to vSmart. The first thing to do is to define the *Security Zones* which at the time of writing are equivalent to VPN. Figure 11-26 shows that we have a ***zone-pair*** *ZP_Zone-NWKT-VPN10_Zo_13779244719* (auto-generated name) with ***source-zone*** Zone-NWKT-VPN10 (VPN10) and ***destination-zone*** *Zone-Internet* (VPN0). The *zone-policy* under the *zone-pair* binds the ***zone-based-policy*** *DIA-Policy* to zone-pair. The *zone-based-policy* is just a regular firewall rule that permits traffic from the subnet 172.16.10.0/24 to IP 10.100.0.13/32. The action Inspect states that return traffic is allowed without additional rule.

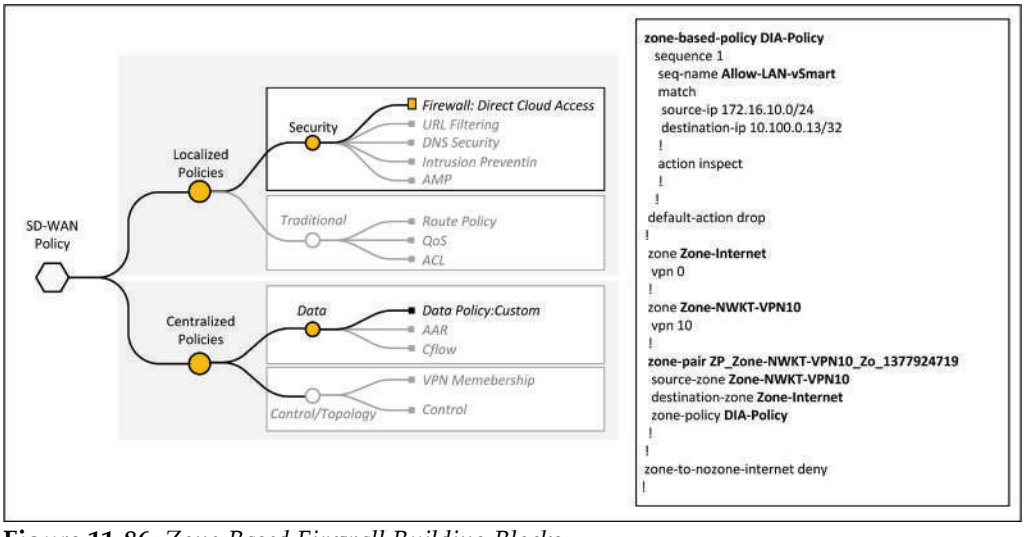


Figure 11-86: *Zone-Based Firewall Building Blocks.*

Configuration Zone Lists

The first thing to do is to create a new Zone-List. Go to the *Configuration/Security* window and select *List* from the *Custom Options* menu.

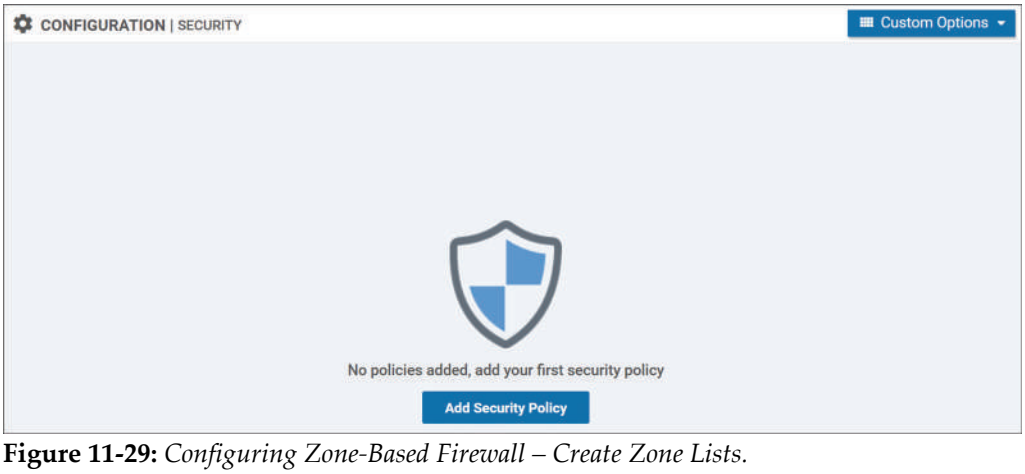


Figure 11-29: *Configuring Zone-Based Firewall – Create Zone Lists.*

You can create a new zone list by selecting *Zones* from the List view and then clicking the *New Zone List* button. Figure 11-30 shows the pre-configured zones *Zone-NWKT-VPN10* (VPN 10) and *Zone-Internet* (VPN 0).

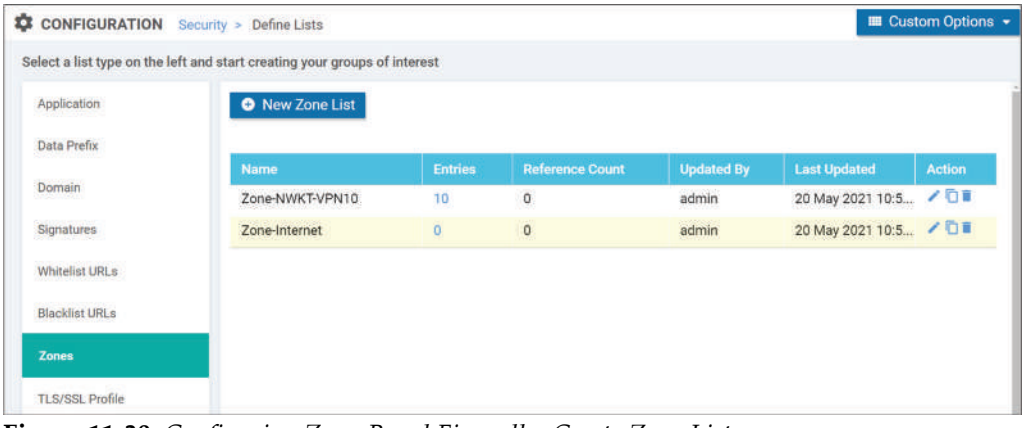


Figure 11-30: *Configuring Zone-Based Firewall – Create Zone Lists.*

Create Data Policy

Go back to the *Configuration/Security* window and click the *Add Security Policy* button.

Figure 11-31: *Configuring Zone-Based Firewall – Create Security Policy.*

Select the *Direct Cloud Access* option from the *Add Security Policy* pop-up window.

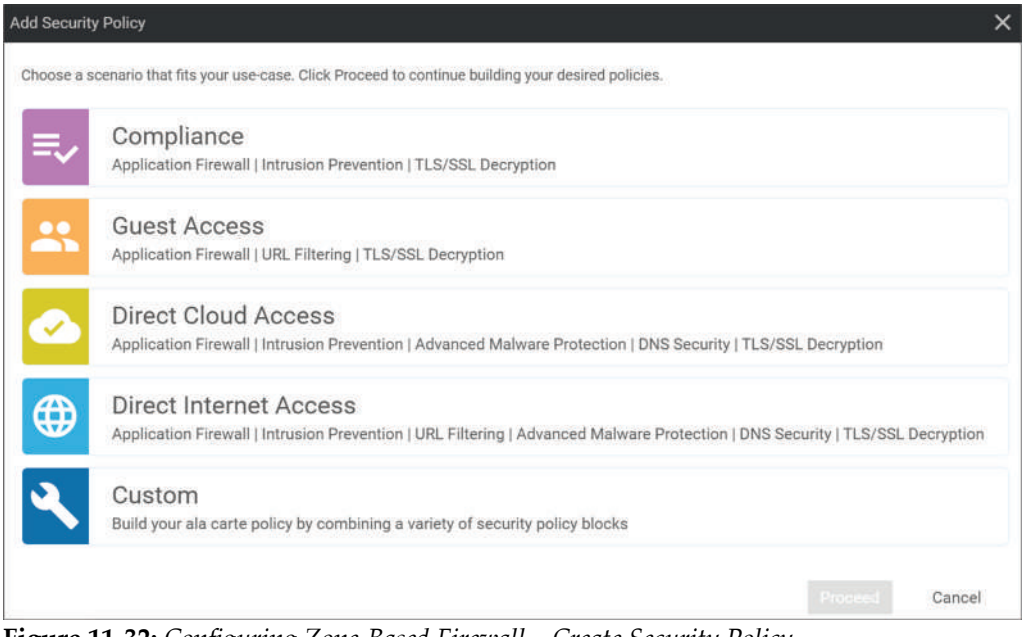


Figure 11-32: *Configuring Zone-Based Firewall – Create Security Policy.*

Select the *Create New* option from the *Add Firewall Policy* drop-down menu.

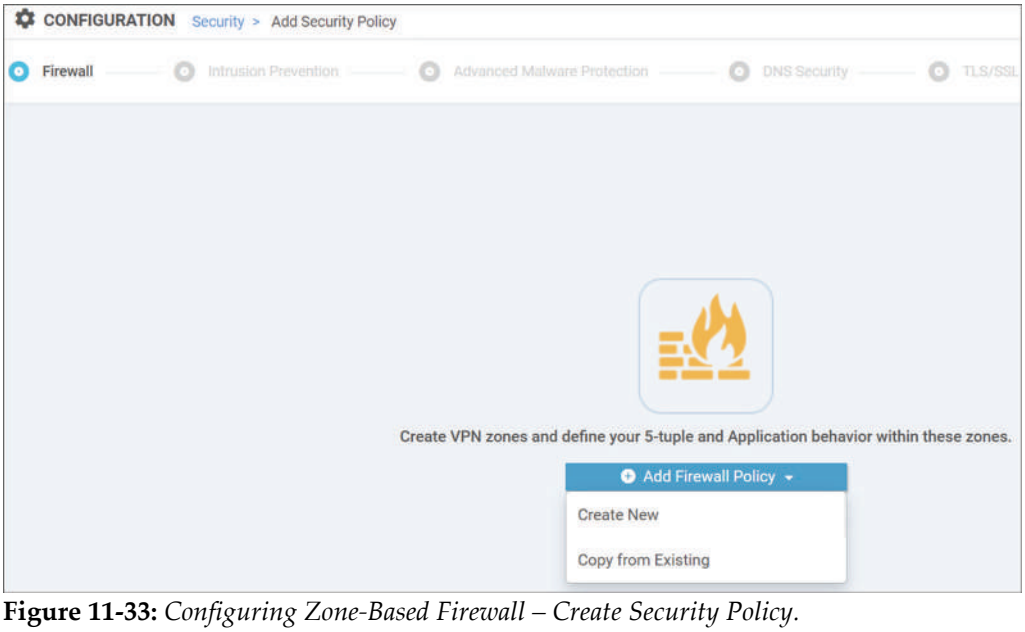


Figure 11-33: *Configuring Zone-Based Firewall – Create Security Policy.*

Click the *Apply Zone-Pairs* button.

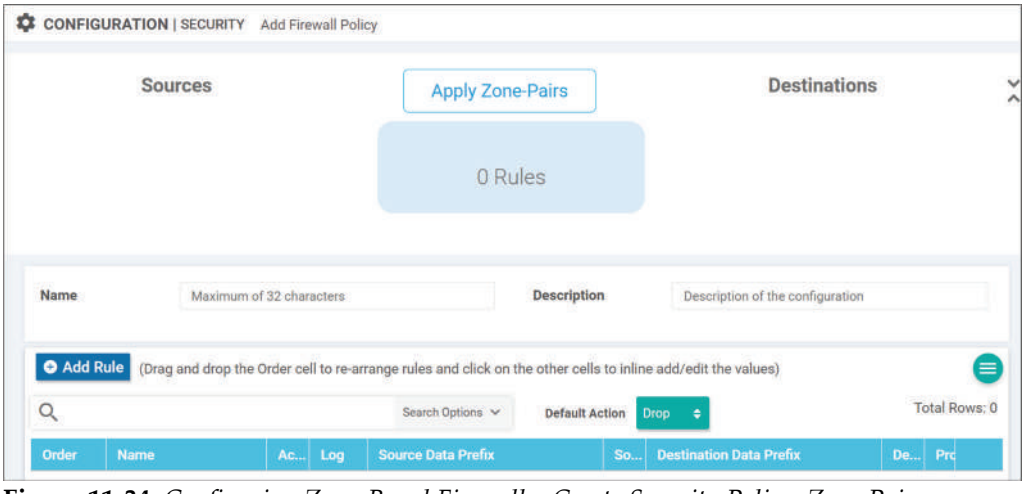


Figure 11-34: *Configuring Zone-Based Firewall – Create Security Policy: Zone Pair.*

Select *Zone-NWKT-VPN10* from the *Source Zone* drop-down menu and *Zone-Internet* from the *Destination Zone* drop-down menu.

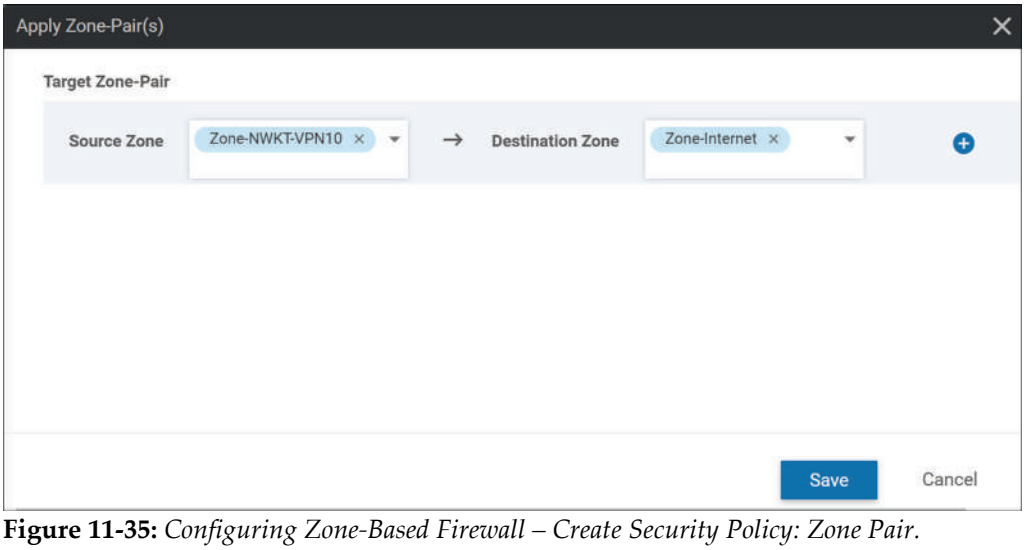


Figure 11-35: *Configuring Zone-Based Firewall – Create Security Policy: Zone Pair.*

Fill in the Firewall policy *Name* and *Description* fields. Click the *Add Rule* button to build a traffic filter between zones.

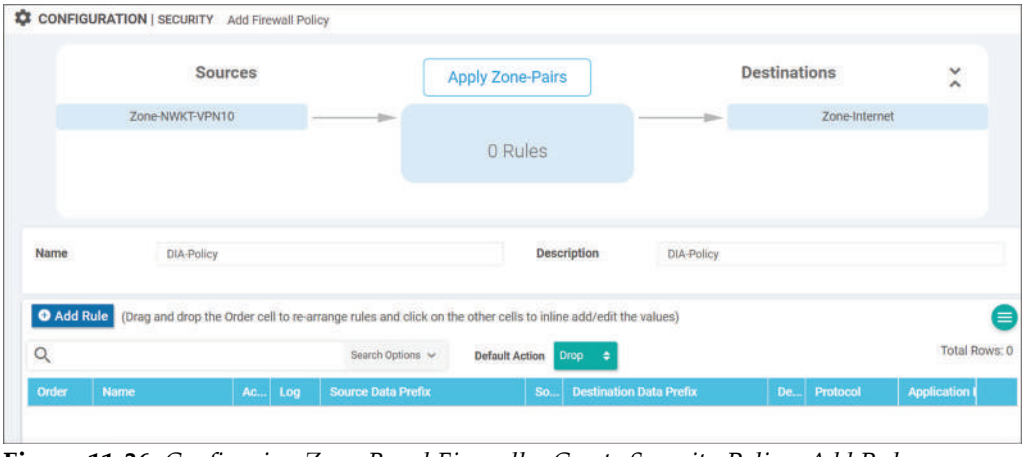


Figure 11-36: *Configuring Zone-Based Firewall – Create Security Policy: Add Rule.*

Figure 11-35 shows the rule that we are using between zones. Note that the Action field is not shown but it is set to *Inspect*. Save the policy.

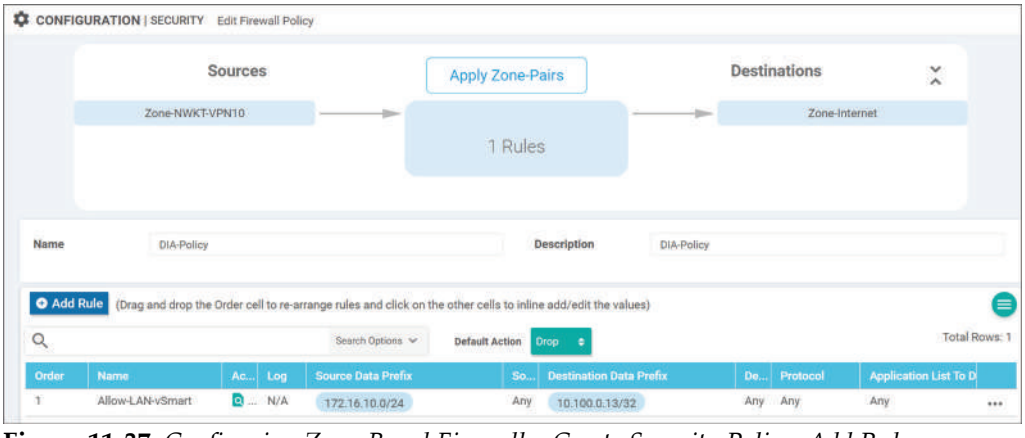


Figure 11-37: *Configuring Zone-Based Firewall – Create Security Policy: Add Rule.*

When the policy is done you can click the *Next* button (not shown in figure 11-38) until you landed on a *Policy Summary* page.

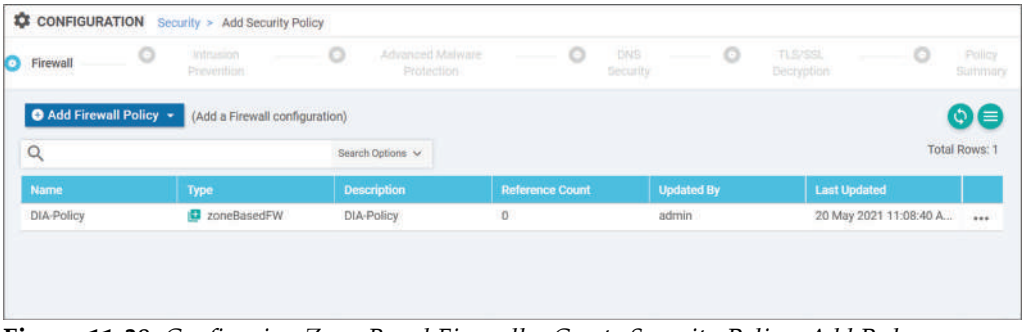


Figure 11-38: *Configuring Zone-Based Firewall – Create Security Policy: Add Rule.*

Fill in the *Name* and *Description* fields. Before saving the policy, you can preview the configuration. Note that when you save the policy, you don't implement it anywhere, that is done by editing a Feature Templates.

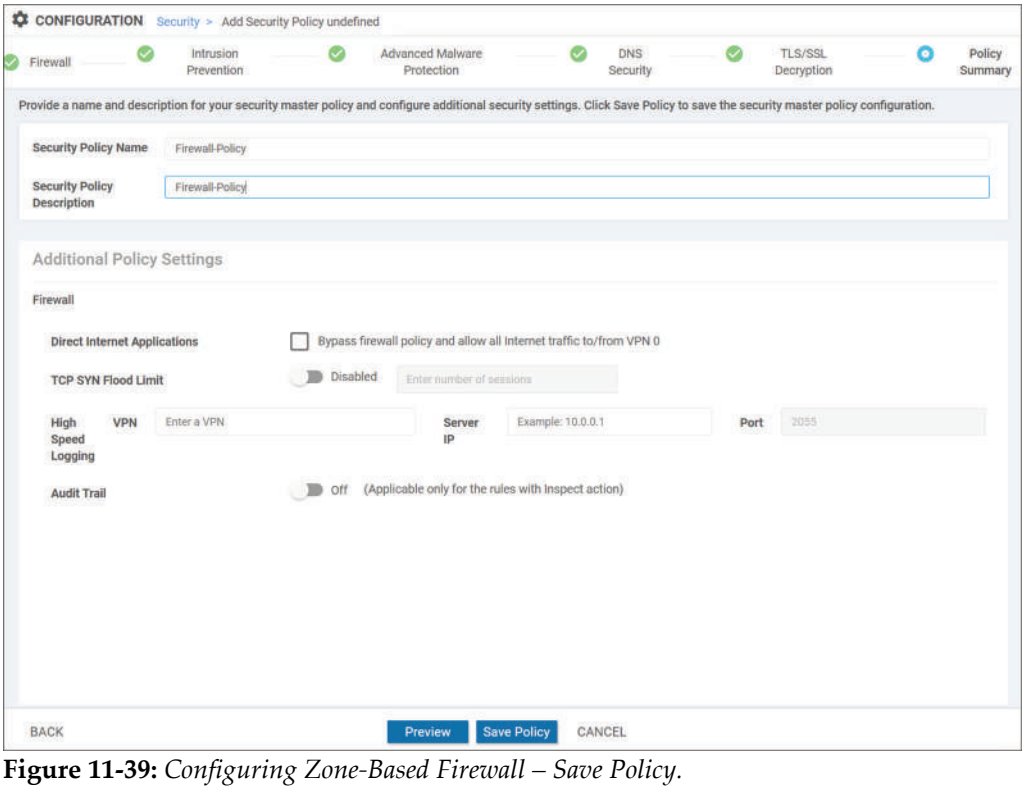


Figure 11-39: *Configuring Zone-Based Firewall – Save Policy.*

Figure 11-40 shows the complete Security Policy.

Figure 11-40: *Configuring Zone-Based Firewall – Save Policy.*

After saving the policy it is listed in the security policy view.

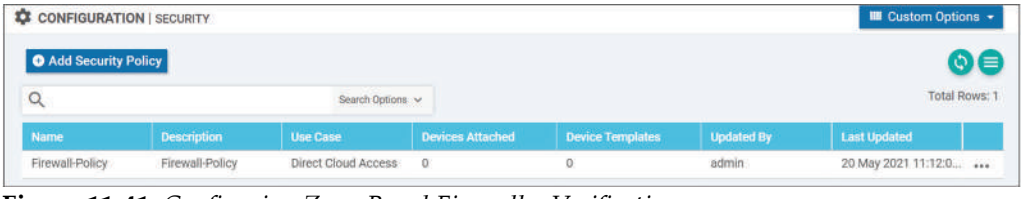


Figure 11-41: *Configuring Zone-Based Firewall – Verification.*

Apply FW Policy to Device Template

The Security Policies are applied to Device Templates. We want to add ZBF to vEdge-1 so we need to edit the *vEdge-Device-Template* shown in figure 11-42. Open the *Options* menu […] and select the *Edit* option.

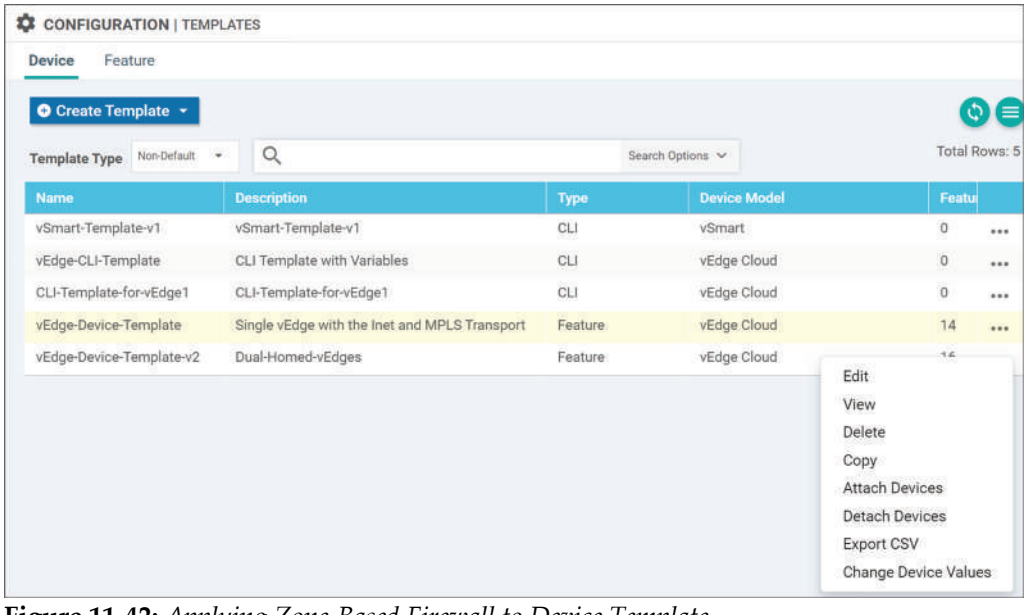


Figure 11-42: *Applying Zone-Based Firewall to Device Template.*

Go to the Additional Templates section and our *Firewall-Policy* from the *Security Policy* drop-down menu. Update change by clicking the *Update* button.

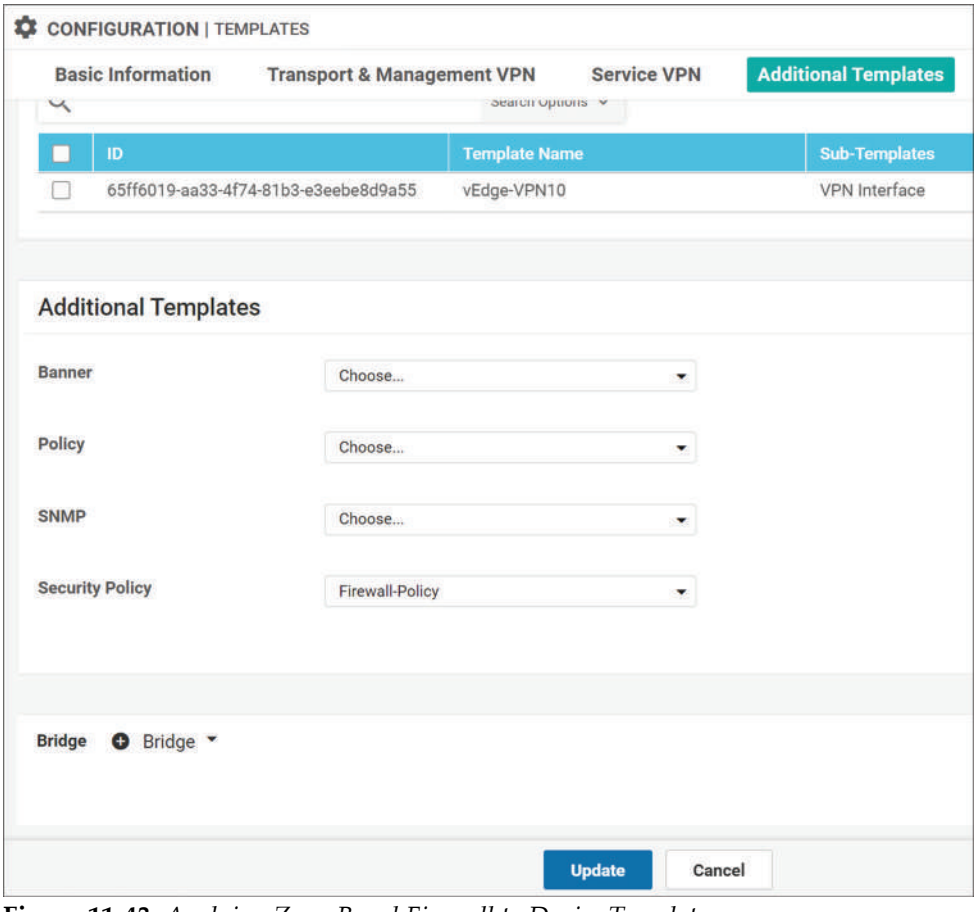


Figure 11-43: *Applying Zone-Based Firewall to Device Template.*

Figure 11-43 shows the successful update process.

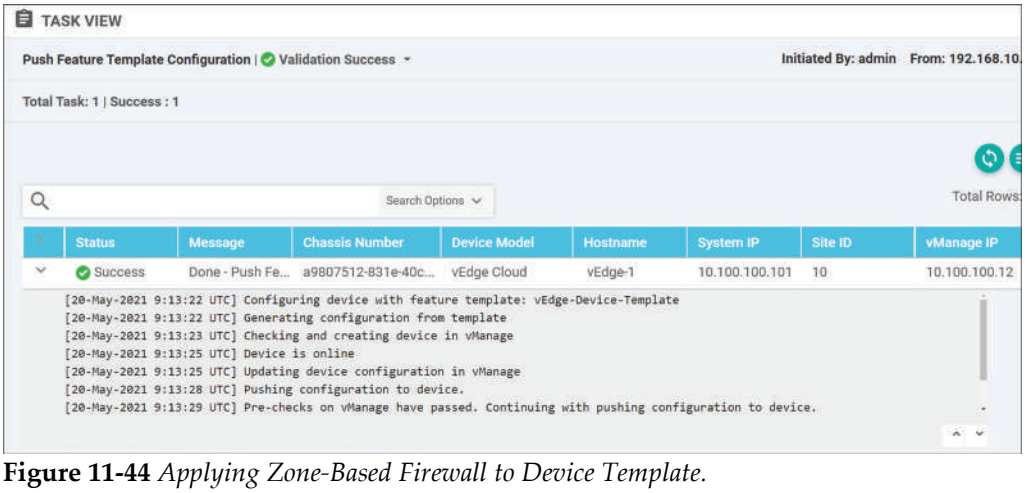


Figure 11-44 *Applying Zone-Based Firewall to Device Template.*

Example 11-1 shows the complete configuration pushed to vEdge-1 after updating the Device Template.

```
system
   device-model            vedge-cloud
   host-name               vEdge-1
   system-ip               10.100.100.101
   domain-id               1
   site-id                 10
   admin-tech-on-failure
   no route-consistency-check
   sp-organization-name    nwkt
   organization-name       nwkt
   console-baud-rate       115200
   vbond 10.100.0.11 port 12346
   aaa
    auth-order local radius tacacs
    usergroup basic
     task system read write
     task interface read write
    !
    usergroup netadmin
    !
    usergroup operator
     task system read
     task interface read
     task policy read
     task routing read
     task security read
    !
    user admin
     password
$6$siwKBQ==$wT2lUa9BSreDPI6gB8sl4E6PAJoVXgMbgv/whJ8F1C6sWdRazdxorYYTLrL6syiG6
qnLABTnrE96HJiKF6QRq1
    !
   !
   logging
    disk
     enable
    !
   !
   ntp
    server 10.100.0.14
     version 4
```

```
   exit
  !
 !
 omp
  no shutdown
  graceful-restart
  advertise connected
  advertise static
 !
 security
  ipsec
   authentication-type sha1-hmac ah-sha1-hmac
  !
 !
 vpn 0
  interface ge0/0
   description "Internet Transport"
   ip address 10.100.0.101/24
   nat
   !
   tunnel-interface
    encapsulation gre
    color public-internet restrict
    control-connections
    allow-service all
    no allow-service bgp
    allow-service dhcp
    allow-service dns
    allow-service icmp
    no allow-service sshd
    no allow-service netconf
    no allow-service ntp
    no allow-service ospf
    no allow-service stun
    allow-service https
   !
   no shutdown
  !
  interface ge0/1
   description "MPLS Transport"
   ip address 10.200.0.101/24
   tunnel-interface
    encapsulation gre
    color mpls restrict
    max-control-connections 0
```

```
    no control-connections
    allow-service all
    no allow-service bgp
    allow-service dhcp
    allow-service dns
    allow-service icmp
    no allow-service sshd
    no allow-service netconf
    no allow-service ntp
    no allow-service ospf
    no allow-service stun
    allow-service https
   !
   no shutdown
  !
  ip route 0.0.0.0/0 10.100.0.1
  ip route 0.0.0.0/0 10.200.0.1
!
vpn 10
 interface ge0/2
  ip address 172.16.10.1/24
  no shutdown
 !
!
vpn 512
 interface eth0
  description Management
  ip dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
policy
 zone Zone-Internet
  vpn 0
 !
 zone Zone-NWKT-VPN10
  vpn 10
 !
 zone-pair ZP_Zone-NWKT-VPN10_Zo_1377924719
  source-zone      Zone-NWKT-VPN10
  destination-zone Zone-Internet
  zone-policy      DIA-Policy
 !
 zone-based-policy DIA-Policy
```

```
    sequence 1
     seq-name Allow-LAN-vSmart
     match
      source-ip      172.16.10.0/24
      destination-ip 10.100.0.13/32
     !
     action inspect
     !
    !
    default-action drop
   !
   zone-to-nozone-internet deny
  !
 !
!
```

Example 11-1: *Complete Configuration of vEdge-1.*

Example 11-2 verifies that now we can only ping destination 10.100.0.13 but not the other IP addresses on the same subnet from site 10.

```
PC1-10> ping 10.100.0.13

84 bytes from 10.100.0.13 icmp_seq=1 ttl=63 time=2.753 ms
84 bytes from 10.100.0.13 icmp_seq=2 ttl=63 time=3.375 ms
^C
PC1-10> ping 10.100.0.12

10.100.0.12 icmp_seq=1 timeout
10.100.0.12 icmp_seq=2 timeout
10.100.0.12 icmp_seq=3 timeout
^C
```

Example 11-2: *Verification.*

And we are done.

Thanks for reading this book.

Appendix A: Device Configurations

Underlay Network Devices

Internet Router

```
INTERNET# sh run

version 9.3(5) Bios:version
hostname INTERNET
<snipped>

feature ospf
feature interface-vlan
<snipped>
ip route 10.100.32.0/24 10.100.0.102
ip route 172.16.10.0/24 10.100.0.40
vlan 1,100
vlan 100
  name Infra

ip prefix-list CONNECTED seq 5 permit 10.100.0.0/24
route-map CONNECTED permit 10
  match ip address prefix-list CONNECTED

interface Vlan100
  no shutdown
  ip address 10.100.0.1/24

interface Ethernet1/1
  switchport access vlan 100

interface Ethernet1/2
  switchport access vlan 100

interface Ethernet1/3
  switchport access vlan 100

interface Ethernet1/4
  switchport access vlan 100

interface Ethernet1/5
  switchport access vlan 100

interface Ethernet1/6
  switchport access vlan 100

interface Ethernet1/7
  switchport access vlan 100
<snipped>
router ospf INET-MPLS
  redistribute direct route-map CONNECTED
```

PE1-West

```
PE1-West#sh run
!
hostname PE1-West
!
license udi pid CSR1000V sn 9L6R8516KGG
!
interface Loopback0
 ip address 1.1.1.1 255.255.255.255
!
interface GigabitEthernet1
 ip address 10.200.0.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 ip address 10.1.2.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet3
 ip address 10.1.3.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
!
router bgp 1
 bgp log-neighbor-changes
 network 10.200.0.0 mask 255.255.255.0
 timers bgp 5 20
 neighbor 10.1.2.2 remote-as 2
 neighbor 10.1.3.3 remote-as 3
!
```

PE2-East

```
PE2-East#sh run

hostname PE2-East
!
!
interface Loopback0
 ip address 2.2.2.2 255.255.255.255
!
interface GigabitEthernet1
 ip address 10.200.1.1 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 ip address 10.1.2.2 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet3
 ip address 10.2.3.2 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
router bgp 2
 bgp log-neighbor-changes
 network 10.200.1.0 mask 255.255.255.0
 timers bgp 5 20
 redistribute static
 neighbor 10.1.2.1 remote-as 1
 neighbor 10.2.3.3 remote-as 3
!
ip route 10.100.23.0 255.255.255.0 10.200.1.103
!
```

P3-South

```
P3-South#sh run
!
hostname P3-South
!
!
interface Loopback0
 ip address 3.3.3.3 255.255.255.255
!
interface GigabitEthernet1
 ip address 10.1.3.3 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 ip address 10.2.3.3 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
!
router bgp 3
 bgp log-neighbor-changes
 timers bgp 5 15
 neighbor 10.1.3.1 remote-as 1
 neighbor 10.2.3.2 remote-as 2
!
```

Control Components

CA-Server

```
CA-Server#sh run
!
hostname CA-Server
!
!
no aaa new-model
clock calendar-valid
!
!
!
crypto pki server PKI
 database level complete
 database archive pkcs12 password 7 030752180500701E1D
 issuer-name cn=rootca.nwkt.local
 grant auto
 hash sha256
 database url flash:
!
crypto pki trustpoint TP-self-signed-3892705574
 enrollment selfsigned
 subject-name cn=IOS-Self-Signed-Certificate-3892705574
 revocation-check none
 rsakeypair TP-self-signed-3892705574
!
crypto pki trustpoint PKI
 revocation-check crl
 rsakeypair PKI
!
!
crypto pki certificate chain TP-self-signed-3892705574
crypto pki certificate chain PKI
 certificate ca 01
  30820316 308201FE A0030201 02020101 300D0609 2A864886 F70D0101 0B050030
  1C311A30 18060355 04031311 726F6F74 63612E6E 776B742E 6C6F6361 6C301E17
  0D323130 33313231 32353035 305A170D 32343033 31313132 35303530 5A301C31
  1A301806 03550403 1311726F 6F746361 2E6E776B 742E6C6F 63616C30 82012230
  0D06092A 864886F7 0D010101 05000382 010F0030 82010A02 82010100 9D0EFD28
  501B5D73 199B33B4 C9051BAE 4CBC3454 03A23DF0 0761E3E0 1F76C29B 9554F864
  9BD9521D A9634D5F F955AAC6 FA2E0E38 12ED3E7C D2DD74D5 ED3CAF3B 96F019C0
  85774100 D230CC30 E2D28C68 A296BE11 E673B853 780932BB 575719FA BFB2EFB0
  0D52C398 FAE3A4F8 76523808 F144953A 4D5CC492 44149694 732EA06F 422C52A6
  45FE7755 04250923 B736633F DAA4810E C8DB9839 2813B50C 0EB8193F 07B376F4
  5B26052F 1F711BC0 9C9C1545 6C03E879 A10E7760 F330ABC1 7D0CF50B F3919D8B
  A23B99B5 2CC13319 5FF0F28C 277BA7C4 C1F1B349 7044A730 7698F600 E4986B9C
  1DF1912D C5D2BF42 C8A903E7 932534F3 F5F6CD56 A8914987 6DE12671 02030100
  01A36330 61300F06 03551D13 0101FF04 05300301 01FF300E 0603551D 0F0101FF
  04040302 0186301F 0603551D 23041830 168014D2 C18924D1 7C547D28 43734EB4
  77F87F2D 77A1AF30 1D060355 1D0E0416 0414D2C1 8924D17C 547D2843 734EB477
  F87F2D77 A1AF300D 06092A86 4886F70D 01010B05 00038201 010072C0 921E1396
  E8B5C22D DBBBC867 33D178EE 9279E2E4 148C2D3C 42AC86E7 06F99465 F6A86663
```

```
  9E7A9BAC 8D27F5AB D524696D 89BF4526 2075BD58 ABD84A70 600B97B4 5AE64105
  877F07A9 51DC1F03 9513F3BD DE59D140 A84D58C8 0A5DE138 1A4386AE 425CC5E5
  327C9776 D64C58D9 15A3A61F 0E0305F6 8DE4430E BDC26FD6 E3397E2A 866C8569
  DF94243A 57F9BCA3 827B5460 04A12B38 2D29D0D8 15335662 1ED3AB84 35B96D22
  D26A828F 49A2A862 77A82D91 54F175A3 1D5CCAF8 D0B255D0 670522D1 936C7853
  AE2A6707 86094354 8E261801 2A054000 7E92C7A2 15C96C01 B766FD36 5E530450
  32F1F0FA DBFDBD29 917A2393 A554E597 217E546C D8749E17 BBE2
        quit
!
!
license udi pid CSR1000V sn 9YYPTPC2M42
diagnostic bootup level minimal
spanning-tree extend system-id
!
!
interface GigabitEthernet1
 ip address 10.100.0.14 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
!
virtual-service csr_mgmt
!
ip forward-protocol nd
ip http server
no ip http secure-server
ip ftp username ****
ip ftp password ****
ip route 0.0.0.0 0.0.0.0 10.100.0.1
!
ip scp server enable
!
!
!
tftp-server flash:PKI.ca
!
!
!
ntp master
!
```

vBond

```
vbond# sh run
system
 host-name               vbond
 system-ip               10.100.100.11
 site-id                 100
 admin-tech-on-failure
 no route-consistency-check
 organization-name       nwkt
 vbond 10.100.0.11 local
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  usergroup tenantadmin
  !
  user admin
   password
$6$meJoeGPXSl/jhFSK$hwZ5Uo3pLhiRLm9AD7eRAutUkRknCLgJyrhzFdJ74qaZyYS3yoKP2xrBw
Ao6slmRrOUiuI9ejmk9LmftljbIp/
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
```

```
omp
 no shutdown
 graceful-restart
 advertise connected
 advertise static
!
security
 ipsec
  authentication-type ah-sha1-hmac sha1-hmac
 !
!
vpn 0
 interface ge0/0
  ip address 10.100.0.11/24
  ipv6 dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.0.1
!
vpn 512
 interface eth0
  ip dhcp-client
  ipv6 dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
vbond#
```

vSmart

```
vsmart# sh run
system
 host-name             vsmart
 system-ip             10.100.100.13
 site-id               100
 admin-tech-on-failure
 organization-name     nwkt
 vbond 10.100.0.11
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
```

```
  usergroup tenantadmin
  !
  user admin
   password
$6$jKzSSqC2GCJveJV4$VxMCv59Qv2J.lDd2luqXXJ9dUuv3izVKXPEbE3b43AAry3n6ptI7DqunO
0y0TzxaUVRGAUZ7E/ySEiWdyt8/60
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
omp
 no shutdown
 graceful-restart
!
vpn 0
 interface eth0
  ip address 10.100.0.13/24
  ipv6 dhcp-client
  tunnel-interface
   allow-service all
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   allow-service netconf
   no allow-service ntp
   no allow-service stun
  !
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.0.1
!
vpn 512
 interface eth1
  ip dhcp-client
```

```
  no shutdown
 !
!
policy
 sla-class NWKT-Critical
  loss    3
  latency 10
  jitter  4
 !
 data-policy _NWKT-VPN10_Direct-Cloud-Access
  vpn-list NWKT-VPN10
   sequence 1
    match
     source-ip                    172.16.10.0/24
     destination-data-prefix-list Cloud-Services
    !
    action accept
     count Cloud-Data-Pkts_-651790820
     nat use-vpn 0
     nat fallback
    !
   !
   default-action accept
  !
 !
 app-route-policy _NWKT-VPN10_NWKT-App-Aware-Policy
  vpn-list NWKT-VPN10
   sequence 1
    match
     destination-ip 172.16.30.0/24
    !
    action
     sla-class NWKT-Critical preferred-color mpls
    !
   !
  !
 !
 lists
  vpn-list NWKT-VPN10
   vpn 10
  !
  data-prefix-list Cloud-Services
   ip-prefix 10.100.0.0/24
  !
  site-list DataCenter
   site-id 30
  !
  site-list Remote-Sites
   site-id 10
   site-id 20
  !
  prefix-list Site30-Prefix-List
   ip-prefix 172.16.30.0/24
  !
```

```
 !
 control-policy Route-Policy-to-DC
  sequence 1
   match route
    prefix-list Site30-Prefix-List
   !
   action reject
   !
  !
  default-action accept
 !
!
apply-policy
 site-list DataCenter
  control-policy Route-Policy-to-DC out
  app-route-policy _NWKT-VPN10_NWKT-App-Aware-Policy
 !
 site-list Remote-Sites
  data-policy _NWKT-VPN10_Direct-Cloud-Access from-service
  app-route-policy _NWKT-VPN10_NWKT-App-Aware-Policy
 !
!
vsmart#
```

vManage

```
vmanage# sh run
system
 host-name             vmanage
 system-ip             10.100.100.12
 site-id               100
 admin-tech-on-failure
 sp-organization-name  nwkt
 organization-name     nwkt
 vbond 10.100.0.11
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  usergroup tenantadmin
  !
  user admin
```

```
   password
$6$Stk7TwMMEy7Qi82x$j.WtL3.WseQgOhAPMtULUfaT9T5ihxYJJI.BXHJj.BzdPapd9TCElFF0M
Zm3daFrE2ClwX9DS5c0jTAASjiW8.
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
vpn 0
 interface eth0
  ip address 10.100.0.12/24
  ipv6 dhcp-client
  tunnel-interface
   allow-service all
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service stun
   allow-service https
  !
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.0.1
!
vpn 512
 interface eth1
  ip dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
vmanage#
```

Edge and LAN Devices

vEdge-1

```
vEdge-1# sh run
system
 host-name               vEdge-1
 system-ip               10.100.100.101
 site-id                 10
 admin-tech-on-failure
 no route-consistency-check
 sp-organization-name    nwkt
 organization-name       nwkt
 vbond 10.100.0.11
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  user admin
   password
$6$siwKBQ==$wT2lUa9BSreDPI6gB8sl4E6PAJoVXgMbgv/whJ8F1C6sWdRazdxorYYTLrL6syiG6
qnLABTnrE96HJiKF6QRq1
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
```

```
 !
!
omp
 no shutdown
 graceful-restart
 advertise connected
 advertise static
!
security
 ipsec
  authentication-type sha1-hmac ah-sha1-hmac
 !
!
vpn 0
 interface ge0/0
  description "Internet Transport"
  ip address 10.100.0.101/24
  nat
  !
  tunnel-interface
   encapsulation gre
   color public-internet restrict
   allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
   allow-service https
  !
  no shutdown
 !
 interface ge0/1
  description "MPLS Transport"
  ip address 10.200.0.101/24
  tunnel-interface
   encapsulation gre
   color mpls restrict
   max-control-connections 0
   no control-connections
   allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
```

```
   allow-service https
  !
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.0.1
 ip route 0.0.0.0/0 10.200.0.1
!
vpn 10
 interface ge0/2
  ip address 172.16.10.1/24
  no shutdown
 !
!
vpn 512
 interface eth0
  description Management
  ip dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
policy
 zone Zone-Internet
  vpn 0
 !
 zone Zone-NWKT-VPN10
  vpn 10
 !
 zone-pair ZP_Zone-NWKT-VPN10_Zo_1377924719
  source-zone      Zone-NWKT-VPN10
  destination-zone Zone-Internet
  zone-policy      DIA-Policy
 !
 zone-based-policy DIA-Policy
  sequence 1
   seq-name Allow-LAN-vSmart
   match
    source-ip      172.16.10.0/24
    destination-ip 10.100.0.13/32
   !
   action inspect
   !
  !
  default-action drop
 !
 zone-to-nozone-internet deny
!
```

vEdge-2

```
vEdge-2# sh run
system
 host-name               vEdge-2
 system-ip               10.100.100.102
 site-id                 30
 admin-tech-on-failure
 no route-consistency-check
 sp-organization-name    nwkt
 organization-name       nwkt
 vbond 10.100.0.11
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  user admin
   password
$6$siwKBQ==$wT2lUa9BSreDPI6gB8sl4E6PAJoVXgMbgv/whJ8F1C6sWdRazdxorYYTLrL6syiG6
qnLABTnrE96HJiKF6QRq1
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
omp
```

```
 no shutdown
 graceful-restart
 advertise bgp
 advertise static
!
security
 ipsec
  authentication-type sha1-hmac ah-sha1-hmac
 !
!
vpn 0
 interface ge0/0
  description "Internet Transport"
  ip address 10.100.0.102/24
  tunnel-interface
   encapsulation gre
   color public-internet restrict
   allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
   allow-service https
  !
  no shutdown
 !
 interface ge0/1
  description "MPLS Transport"
  ip address 10.100.23.2/24
  tunnel-interface
   encapsulation gre
   color mpls restrict
   max-control-connections 0
   no control-connections
   allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
   allow-service https
  !
  no shutdown
 !
 interface ge0/3
```

```
  ip address 10.100.32.2/24
  tloc-extension ge0/0
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.0.1
 ip route 0.0.0.0/0 10.100.23.3
 ip route 0.0.0.0/0 10.200.2.1
!
vpn 10
 router
  bgp 65030
   address-family ipv4-unicast
    redistribute omp
   !
   neighbor 10.2.4.4
    no shutdown
    remote-as 65030
    address-family ipv4-unicast
    !
   !
  !
 !
 interface ge0/2
  ip address 10.2.4.2/24
  no shutdown
 !
!
vpn 512
 interface eth0
  description Management
  ip dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
vEdge-2#
```

vEdge-3

```
vEdge-3# sh run
system
 host-name               vEdge-3
 system-ip               10.100.100.103
 site-id                 30
 admin-tech-on-failure
 no route-consistency-check
 sp-organization-name    nwkt
 organization-name       nwkt
 vbond 10.100.0.11
 aaa
  auth-order local radius tacacs
  usergroup basic
   task system read write
   task interface read write
  !
  usergroup netadmin
  !
  usergroup operator
   task system read
   task interface read
   task policy read
   task routing read
   task security read
  !
  user admin
   password
$6$siwKBQ==$wT2lUa9BSreDPI6gB8sl4E6PAJoVXgMbgv/whJ8F1C6sWdRazdxorYYTLrL6syiG6
qnLABTnrE96HJiKF6QRq1
  !
  user ciscotacro
   description CiscoTACReadOnly
   group       operator
   status      enabled
  !
  user ciscotacrw
   description CiscoTACReadWrite
   group       netadmin
   status      enabled
  !
 !
 logging
  disk
   enable
  !
 !
 ntp
  server 10.100.0.14
   version 4
  exit
 !
!
omp
```

```
 no shutdown
 graceful-restart
 advertise bgp
 advertise static
!
security
 ipsec
  authentication-type sha1-hmac ah-sha1-hmac
 !
!
vpn 0
 interface ge0/0
  description "Internet Transport"
  ip address 10.100.32.3/24
  nat
  !
  tunnel-interface
   encapsulation gre
   color public-internet restrict
   allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
   allow-service https
  !
  no shutdown
 !
 interface ge0/1
  description "MPLS Transport"
  ip address 10.200.1.103/24
  tunnel-interface
   encapsulation gre
   color mpls restrict
   max-control-connections 0
   no control-connections
   allow-service all
   no allow-service bgp
   allow-service dhcp
   allow-service dns
   allow-service icmp
   no allow-service sshd
   no allow-service netconf
   no allow-service ntp
   no allow-service ospf
   no allow-service stun
   allow-service https
  !
  no shutdown
```

```
 !
 interface ge0/3
  description    TLOC-Ext-to-MPLS
  ip address 10.100.23.3/24
  tloc-extension ge0/1
  no shutdown
 !
 ip route 0.0.0.0/0 10.100.32.2
 ip route 0.0.0.0/0 10.200.1.1
!
vpn 10
 router
  bgp 65030
   address-family ipv4-unicast
    network 172.16.30.0/24
    maximum-paths paths 2
    redistribute omp
   !
   neighbor 10.3.4.4
    no shutdown
    remote-as 65030
    address-family ipv4-unicast
    !
   !
  !
 !
 interface ge0/2
  ip address 10.3.4.3/24
  no shutdown
 !
!
vpn 512
 interface eth0
  description Management
  ip dhcp-client
  no shutdown
 !
 ip route 0.0.0.0/0 192.168.10.1
!
vEdge-3#
```

LAN-Ro04

```
Ro04#sh run
!
hostname Ro04
!
!
!
interface Loopback77
 ip address 172.16.77.1 255.255.255.0
!
interface Loopback172
 description ** LAN SIMULATION **
```

```
 ip address 172.16.30.1 255.255.255.0
!
interface GigabitEthernet1
 ip address 172.16.30.1 255.255.255.0
 shutdown
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet2
 ip address 10.2.4.4 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet3
 ip address 10.3.4.4 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
interface GigabitEthernet4
 ip address 10.100.0.40 255.255.255.0
 negotiation auto
 no mop enabled
 no mop sysid
!
router bgp 65030
 bgp router-id 4.4.4.4
 bgp log-neighbor-changes
 neighbor 10.2.4.2 remote-as 65030
 neighbor 10.3.4.3 remote-as 65030
 !
 address-family ipv4
  network 10.100.0.0 mask 255.255.255.0
  network 172.16.30.0 mask 255.255.255.0
  network 172.16.77.0 mask 255.255.255.0
  neighbor 10.2.4.2 activate
  neighbor 10.2.4.2 soft-reconfiguration inbound
  neighbor 10.3.4.3 activate
  maximum-paths ibgp 2
 exit-address-family
!
Ro04#
```

This book is part of **The Network Times** Handbook Series.

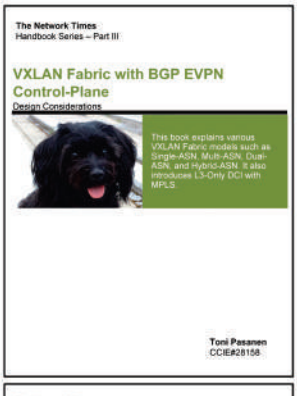
Virtual Extensible LAN (VXLAN):
A Practical guide to VXLAN solution

Amazon

Pages:	368
ISBN-10 :	170326536X
ISBN-13 :	978-1703265361
Paperback:	19:90€/21:00$
Kindle Edition:	7:70$

Leanpub.com

PDF Edition:	7:99$



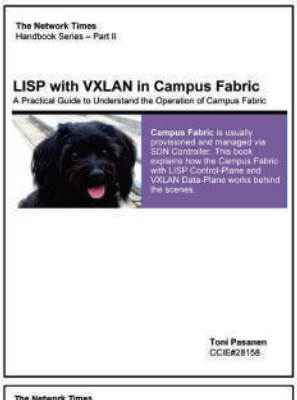
VXLAN Fabric with BGP EVPN Control-Plane:
Design Considerations

Amazon

Pages:	271
ISBN-13 :	979-8683023690
Paperback:	13:30€/15:00$
Kindle Edition:	6:60$

Leanpub.com

PDF Edition:	5:99$



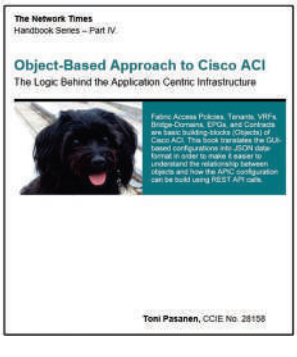
LISP Control-Plane in Campus Fabric:
A Practical Guide to Understand the Operation of Campus

Amazon

Paperback :	264 pages
ISBN-13 :	979-8615059186
Paperback:	14:54€/15:00$
Kindle Edition:	6:60$

Leanpub.com

PDF Edition:	4:99$



Object-Based Approach ti ACI
The Logic Behind the Application Centric Infrastructure

Amazon

Paperback :	228 pages
ISBN-13 :	979-8583070381
Paperback:	13:56€/10:00$
Kindle Edition:	5:50$

Leanpub.com

PDF Edition:	4:99$