Implementing Cisco SD-WAN

In This Course

- What is Cisco's Software Defined Wide-Area Network (SD-WAN) ?
- How Cisco SD-WAN Works Behind-the-Scenes
- Onboarding & Deploying Cisco SD-WAN Controllers, Routers & VPNs
- Implementing Cisco SD-WAN Templates & Policies
- Advanced Cisco SD-WAN Features
  - Direct Internet Access (DIA)
  - Application Aware Routing (AAR)
  - Security
  - QoS
  - TLOC Extension
  - Direct Cloud Access & Cloud OnRamp

Implementing Cisco SD-WAN

Course Introduction

Brian McGahan

CCIEx4 #8593 & CCDE #2013::13

In This Course

- What is Cisco's Software Defined Wide-Area Network (SD-WAN) ?
- How Cisco SD-WAN Works Behind-the-Scenes
- Onboarding & Deploying Cisco SD-WAN Controllers, Routers & VPNs
- Implementing Cisco SD-WAN Templates & Policies
- Advanced Cisco SD-WAN Features
  - Direct Internet Access (DIA)
  - Application Aware Routing (AAR)
  - Security
  - QoS
  - TLOC Extension
  - Direct Cloud Access & Cloud OnRamp

Course Prerequisites

- Course assumes a working knowledge of…
  - Cisco IOS (XE) syntax
  - Basic IP Routing with static & dynamic protocols (e.g. OSPF/BGP)
  - Differences between WAN technologies (e.g. Internet vs. MPLS)

- Course does not assume previous experience with…
  - IPsec VPNs
  - SD-WAN implementations
  - Cisco's SD-WAN Viptela OS

Implementing Cisco SD-WAN

SD-WAN Overview

In This Section

- Traditional WAN Routing Problems
- What is SD-WAN?
- How SD-WAN Can Fix Traditional Problems
- Cisco’s SD-WAN Solution Overview

Traditional WAN Routing Problems - Management

- In traditional networking, each router is managed separately
  - **Each device is typically manually configured from a Command Line Interface (CLI)**
    - E.g. IP addressing, routing protocols, & VPN configs are locally significant
  - **Change control becomes an issue** as the network starts to scale
    - E.g. 1000 branch sites = 1000 routers to manually configure for each change
  - **Bolt-on Network Management Systems (NMS) only offer a limited fix**
    - E.g. Cisco Prime, SolarWinds, etc.
    - Additional software licensing, increasing Total Cost of Ownership (TCO)
    - NMS is typically just for monitoring, not for configuring
      - Still might need something like Ansible to automate & manage configurations

Traditional WAN Routing Problems – Secure Connectivity

- In traditional networking, secure any-to-any connectivity is difficult at scale
  - Could use **static site-to-site IPsec tunnels**, but **hard to manage as network grows**
    - E.g. adding 100th site means 99 other sites need to be reconfigured
  - **Routers have upper limits on how many IPsec tunnels they can form**
    - E.g. Branch routers have much smaller CPUs than DC Edge routers
  - **Could use group encryption like GETVPN/GDOI, but this limits transport options**
    - E.g. NAT isn't supported, so Internet links are not supported
  - **Could use dynamic encryption, like DMVPN, but it has drawbacks as well**
    - Scaling up requires forklift upgrades of hub routers
    - Scaling out requires difficult routing and failover logic w/ daisy-chained hubs
    - DMVPN still has no visibility of underlying transport issues
      - E.g. service degradation (brownouts) can still occur without complicated IP SLA / PfR / iWAN configurations

Traditional WAN Routing Problems – Intelligent Routing

- In traditional networking, intelligent routing is difficult to implement
  - E.g. routing is normally based on a flat metric, not the real-time state of the network

- For example, a branch router with 2 ISPs typically uses default routes to both
  - Allows for basic failover and load balancing, but no application intelligence
  - E.g. static routes could prevent against link failure (blackout) but can't prevent against service degradation (brownout)
  - Could use something like IP SLA, but typically difficult to configure and scale

Traditional WAN Routing Problems – Service Reliability

- Traditionally, Direct Internet Access (DIA) could not meet reliability goals
  - Internet routing does not offer any end-to-end…
    - Bandwidth guarantees
    - Delay/Jitter guarantees
    - Mean Time to Repair (MTTR) SLA
- Traditionally, result was that expensive Private WAN circuits were needed
  - E.g. MPLS L2VPN/L3VPN
- Additionally, Private WAN typically does not offer Internet access
  - Separate Internet circuits were typically still required, driving up costs
  - E.g. collect all traffic at a central DC, then route it to the Internet

What is Software Defined WAN (SD-WAN)?

- Software Defined WAN (SD-WAN) uses centralized Controllers to decouple **WAN Edge Routers'** *data-plane* from their *control-plane* & *management-plane*
  - Data-plane is the forwarding of traffic between interfaces
  - Control-plane is the policy of how to forward this traffic, e.g. VPN, VoIP vs. Data, etc.
  - Management-plane is the configuration & monitoring of these policies

- SD-WAN is meant to fix previously mentioned problems with **WAN Edge Routing**
  - Note that this is for **WAN Edge** only, and does not apply to DC Core / LAN Core, etc.
  - DC Core (e.g. Cisco ACI) & LAN Core (e.g. Cisco SD-Access) can hand-off to SD-WAN

Solving Traditional WAN Routing Problems with Software Defined WAN (SD-WAN)

- SD-WAN offers some of the following benefits vs. traditional WAN routing
  - **Centralized management** - easier configuration, monitoring, & automation
    - We make changes on Controller, then it pushes changes to WAN Edge Routers
    - For example, allows for central Zero Touch Provisioning (ZTP) of new sites
  - **Secure connectivity** - transparent automation of IPsec config & key management
    - E.g. simple any-to-any encryption
  - **Intelligent routing** - traffic can be routed based on application performance parameters
    - E.g. choose lowest delay path for VoIP traffic based on current network conditions
  - **Cloud optimization** - SaaS traffic can be offloaded to closet cloud entry-point
    - E.g. don't go to the central Data Center to reach AWS/Azure/GCP; use local DIA
  - **Lower TCO** – SD-WAN is transport agnostic, which results in lower costs
    - E.g. MPLS isn't required; you could use multiple lower cost Internet circuits or a mix of MPLS + Internet and still meet reliability goals

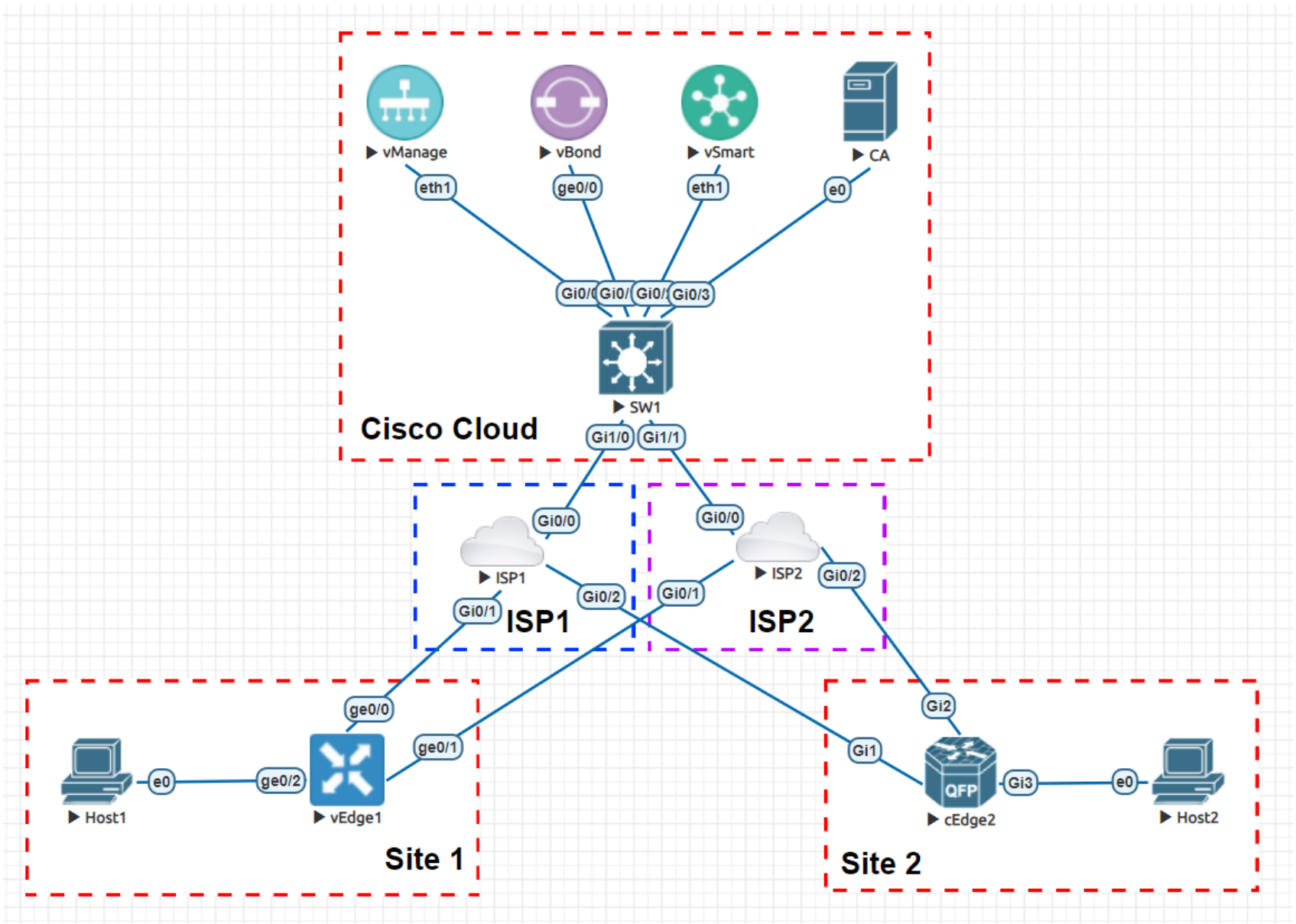
What is Cisco's Software Defined Wide Area Networking (SD-WAN) Solution?

- Cisco's (Viptela) SD-WAN solution offers all of these benefits, such as...
  - **Single Pane of Glass Management**
    - vManage Controller allows centralized configuration and management
  - **Simple Provisioning**
    - Automated onboarding of new sites with Zero Touch Provisioning (ZTP)
  - **Secure Connectivity**
    - Automatically uses IPsec w/certificates for encryption between sites
  - **Intelligent Routing**
    - Supports Application Aware Routing (AAR) + automatic failover
  - **Lower TCO**
    - Can use cheap Internet circuits instead of MPLS/VPLS, or a combination
  - **Cloud Friendly**
    - Simplifies SaaS/IaaS deployments by optimizing forwarding

How does Cisco's SD-WAN Solution Work?

- Cisco's SD-WAN Solution can be broken down into two main components
  - SD-WAN **Controllers**
  - **WAN Edge Routers**
- **Controllers** consist of three devices, each performing a different function
  - vManage NMS – GUI for configuring & managing the SD-WAN Solution
  - vBond Orchestrator – Automation engine used for onboarding new routers
  - vSmart Controller – Controls Routing & Policy decisions
- **WAN Edge Routers** consist of two categories of devices
  - vEdge Routers running Viptela software
  - cEdge Routers running Cisco IOS XE software
  - Both are available as hardware boxes and as Virtual Machines (VMs)

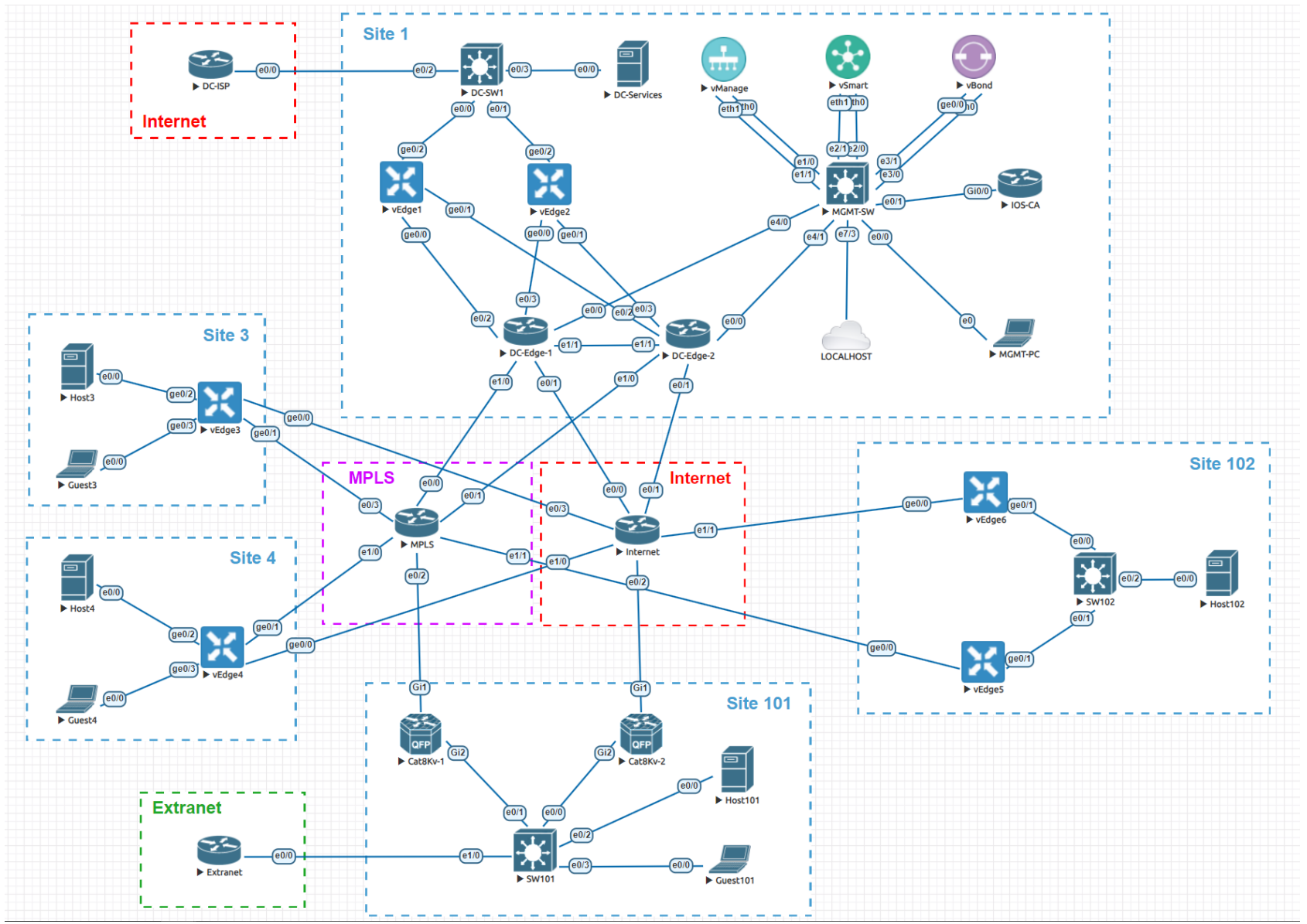
Example Simple Cisco SD-WAN Topology



Implementing Cisco SD-WAN

Cisco SD-WAN Behind the Scenes Example

Example Cisco SD-WAN Topology



How Cisco SD-WAN Works Behind the Scenes

- The first step in Cisco SD-WAN is to “onboard” the Controllers
  - Onboarding is the process of establishing secure tunnels between the devices
- vBond, vSmart, & vManage controllers use PKI Certificates to authenticate
  - For Cisco cloud-hosted controllers, PKI “just works”
  - Can also use internal Certificate Authority (CA) for on-prem hosted controllers
- Controllers then create DTLS (TLS over UDP) tunnels between each other
  - All control & management-plane traffic sent over DTLS is AES-256 encrypted
  - Uses UDP base port 12346 for transport, but can hop around for NAT

How Cisco SD-WAN Works Behind the Scenes (cont.)

- Next step is to “onboard” the WAN Edge Routers
  - I.e. establish secure tunnels from WAN Edge to the Controllers
- WAN Edge Router first establishes a DTLS tunnel to vBond
  - vBond is the “orchestrator”, and is in charge of new devices joining the SD-WAN
  - vBond has a list of authorized serial numbers, e.g. from Cisco Smart Licensing
  - WAN Edge knows vBond IP address either through manual config or through Zero Touch Provisioning (ZTP)
  - Once complete, vBond tells vSmart & vManage about new WAN Edge Router
- WAN Edge Router next establishes DTLS tunnels to vManage & vSmart

How Cisco SD-WAN Works Behind the Scenes (cont.)

- vManage can now manage the WAN Edge Router over DTLS
  - E.g. securely push a config template to the new WAN Edge Router

- vSmart can now exchange control-plane info with WAN Edge over DTLS
  - vSmart collects and distributes symmetric IPsec keys for all WAN Edge Routers
    - IKE/ISAKMP is not needed because a secure tunnel already exists
    - This makes the IPsec control-plane much more scalable
  - vSmart learns routes from WAN Edges w/ Overlay Management Protocol (OMP)
  - vSmart pushes routes & policy to other WAN Edge Routers via OMP
    - vSmart is analogous to a BGP Route Reflector
  - vSmart can control the routing policy for all WAN Edge Routers
    - e.g. hub-and-spoke, Application Aware Routing (AAR), etc.

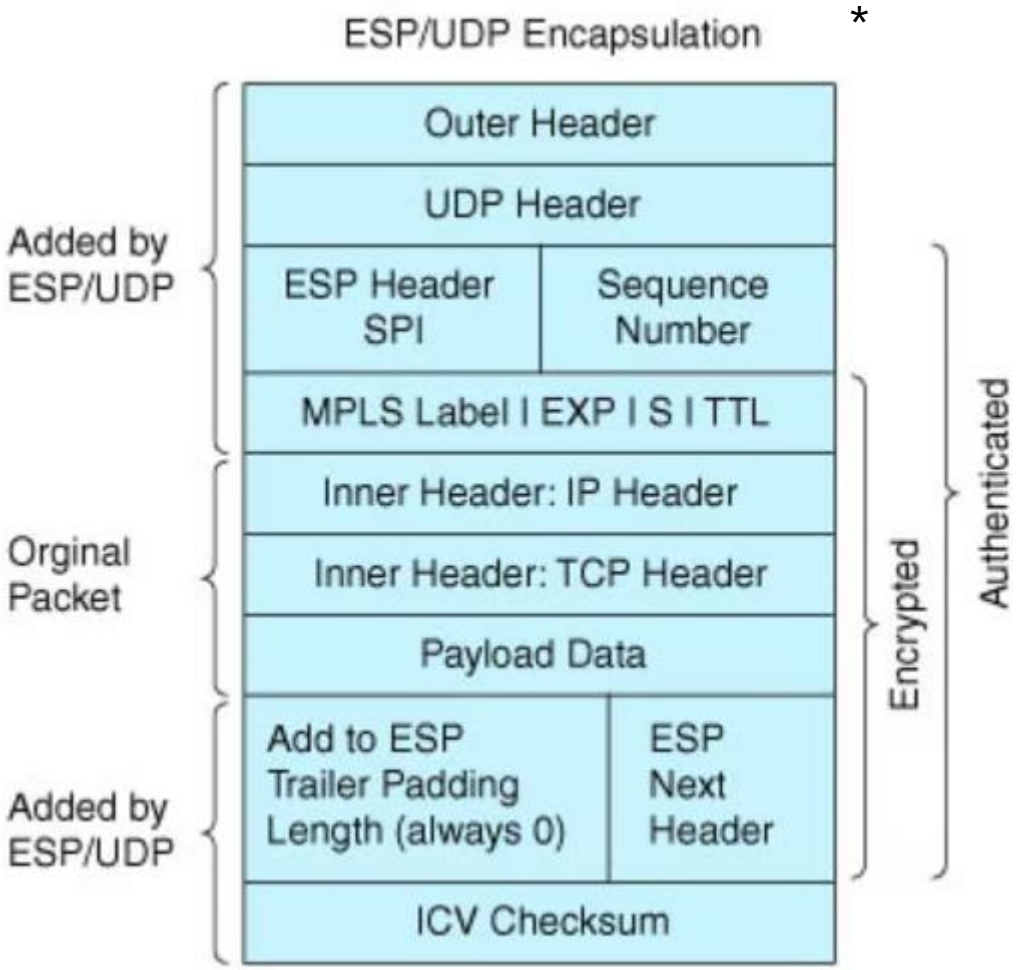
How Cisco SD-WAN Works Behind the Scenes (cont.)

- Final step is that WAN Edge Routers form IPsec tunnels between sites
  - IPsec keys are learned from vSmart over DTLS, so IKE/ISAKMP isn't needed
  - IPsec keys are global by default, but can be set to "pairwise"
    - Pairwise means one key per-peer, instead of same key for all
  - IPsec re-keying happens periodically, with new keys sent to vSmart for distribution
    - Rekey timer and replay-window can be modified if desired

- Default vSmart policy is a full-mesh of IPsec tunnels
  - vSmart learns about all Transport Locations (TLOCs) from OMP advertisements
    - TLOC is WAN Edge Router's IP towards WAN, plus other attributes like "color"
  - Result is if two WAN Edges have both ISP1 & ISP2, four tunnels form:
    - ISP1 to ISP1 / ISP2 to ISP2 / ISP1 to ISP2 / ISP2 to ISP1
  - vSmart policies are centrally controlled through vManage

How Cisco SD-WAN Works Behind the Scenes (cont.)

- IPsec data-plane uses modified ESP over UDP
  - Uses AES encryption & SHA authentication
  - MPLS label is added to carry the VPN number
    - E.g. VPN 1 can talk to VPN 1 by default
    - E.g. VPN 1 cannot talk to VPN 2 by default
  - UDP base port is 12346, but can jump around
    - E.g. two WAN Edges both behind NAT is supported in certain cases



*Security Configuration Guide for vEdge Routers, Cisco SD-WAN

How Cisco SD-WAN Works Behind the Scenes (cont.)

- Once IPsec data-plane is established between WAN Edge Routers, Bidirectional Forwarding Detection (BFD) is used to track tunnel states
  - Traditional IPsec uses Dead Peer Detection (DPD), which is slow
  - BFD is a lightweight way to quickly detect direct or indirect link failures
    - E.g. BFD tracks the entire transport path, not just your local link to ISP
  - BFD also measures link statistics like latency, loss, & jitter
    - Can be used later for Application Aware Routing (AAR)
  - Traffic is automatically re-routed around BFD failures by default
    - E.g. tracking the WAN “just works” out of the box

Cisco SD-WAN CLI Verifications

- Verify Control-Plane & Management-Plane (DTLS) tunnels
  - vBond - **`show orchestrator connections`**
  - vManage / vSmart / vEdge - **`show control connections`**
  - cEdge - **`show sdwan control connections`**
- Verify Data-Plane (IPsec) tunnels
  - vEdge - **`show bfd sessions`**
  - cEdge - **`show sdwan bfd sessions`**
- Verify Overlay (VPN) Routing
  - vEdge - **`show ip route [vpn 1]`**
  - cEdge - **`show ip route [vrf 1]`**

Cisco SD-WAN GUI Verifications

- View a WAN Edge Router's Running Configuration
  - vManage > Configuration > Devices > (3 dots) > Running Configuration
- SSH to a WAN Edge Router
  - vManage > Monitor > Devices > (3 dots) > SSH Terminal

Cisco SD-WAN GUI Verifications (cont.)

- Verify the Control Plane from vManage
  - vManage > Monitor > Devices > [device]
    - *Control Connections*
      - Equivalent of **`show [sdwan] control connections`**
    - *Real Time > BFD Sessions*
      - Equivalent of **`show [sdwan] bfd sessions`**
    - *Real Time > IP Routes*
      - Equivalent of **`show ip route [vrf *]`**
- Verify the Data Plane from vManage
  - vManage > Monitor > Devices > [device] > Troubleshooting > Ping | Traceroute
- Perform remote packet capture from vManage
  - **vManage > Administration > Settings > Data Stream > Enabled**
  - Monitor > Devices > [device] > Troubleshooting > Packet Capture

Implementing Cisco SD-WAN

Understanding Cisco SD-WAN Controller Onboarding

Cisco SD-WAN Controllers Overview

- Cisco's Viptela SD-WAN Solution uses three types of controllers
  - vManage – used for management
  - vSmart – used for routing & traffic policies
  - vBond – used for orchestration

- Starting v20.14.x, solution is now rebranded as Cisco Catalyst SD-WAN
  - vManage renamed to SD-WAN Manager
  - vSmart renamed to SD-WAN Controller
  - vBond renamed to SD-WAN Validator
  - Only a cosmetic change, all functionality in this course is still the same
    - v20.9 is current Suggested Release; v20.10 and above is Early Deployment (ED)

- Controllers can be hosted by Cisco (public cloud) or internally (private cloud)

Cisco SD-WAN Controller Functions

- **vBond** provides Orchestration Plane
  - Authenticates & authorizes SD-WAN WAN Edge Routers so secure control connections can be established (i.e. DTLS tunnels)

- **vSmart** runs the Control Plane
  - Learns routes from WAN Edge Routers and reflects them back
    - Like a BGP Route Reflector, but for Overlay Management Protocol (OMP)
  - Implements control-plane policies to affect traffic flows
  - Distributes centralized data plane polices to SD-WAN WAN Edge Routers
    - E.g. local Cloud OnRamp , Application Aware Routing (AAR), etc.

- **vManage** runs the Management Plane
  - “Single Pane of Glass” for centralized provisioning, troubleshooting, and monitoring
  - Provides a GUI for manual management and APIs for automation

Cisco SD-WAN Controller Hosting Options – Cisco Cloud

- Hosting SD-WAN Controllers in Cisco Cloud
  - Cisco provides you IP Addresses & credentials for vManage, vSmart, & vBond, which are pre-configured and reachable via the public Internet
  - Controllers + vEdge & cEdges already trust Cisco Certificate Authority (CA)
    - E.g. PKI “just works” automatically when using Cisco Cloud
  - Onboarding new WAN Edge Routers is simple with Zero Touch Provisioning (ZTP)
    - WAN Edge Router gets an IP address via DHCP from the ISP
    - Router contacts a pre-defined ZTP server in Cisco Cloud
    - Cisco Smart Licensing already has the serial numbers registered
    - E.g. onboarding “just works” automatically when using Cisco Cloud

Cisco SD-WAN Controller Hosting Options – Private Cloud

- Hosting SD-WAN Controllers in “Private” Cloud
  - vManage, vSmart, & vBond are virtual machines hosted internally
    - Could be on-prem, private colocation, public cloud, etc.
  - Controllers must be reachable from WAN Edge Routers
    - E.g. via the public Internet and/or Private WAN (e.g. MPLS)
  - Internal Certificate Authority (CA) is used for authentication
    - E.g. Controllers manually install/trust your internal Root CA & request cert signing
    - WAN Edge Routers must manually install the Root CA Cert
  - Onboarding WAN Edge Routers is more of a manual process in this case

Onboarding Overview

- “Onboarding” is the process of establishing secure tunnels between the devices
  - Cisco SD-WAN Controllers use PKI (Certificates) to authenticate each other
  - DTLS (TLS over UDP) tunnels are then established to exchange control-plane info

SD-WAN Controllers – Underlay Transport

- The first step in onboarding the Controllers is to establish IP transport between themselves & the WAN Edge Routers in the underlay network
  - E.g. over the Internet or MPLS facing link(s)

- Viptela OS uses VPN numbers to represent different routing table spaces
  - I.e. VPN numbers in Viptela OS are equivalent to VRFs in Cisco IOS

- VPN 0 is the "Transport VPN", and is used for links facing towards the WAN
  - VPN 0 is the underlay transport for the overlay IPsec tunnels
    - E.g. towards the public Internet or private MPLS
  - VPN 0 is equivalent to the global (default) VRF in Cisco IOS
  - All Controllers & WAN Edge Routers need reachability to each other in VPN 0
    - I.e. this is where DTLS tunnels are established for the control-plane

SD-WAN Controllers – Management VPN

- VPN 512 is the “Management VPN”, used for Out-of-Band Management
  - VPN 512 is equivalent to VRF “Mgmt-intf” in Cisco IOS
  - Can have a default route separate from the Transport VPN (0)

- Controllers & Edges don’t need VPN 512 if you only want to manage them in-band
  - **`allow-service`** command under the “tunnel-interface” in VPN 0 needs to be modified to allow for in-band management on the WAN facing links
    - E.g. SSH, NETCONF, HTTPS
  - **`allow-service`** doesn’t affect mgmt between Controllers or to WAN Edge Router
    - I.e. DTLS traffic is always allowed in between Controllers and in on WAN Edge

Onboarding Internally Hosted Cisco SD-WAN Controllers – Required CLI Config

- Onboarding Controllers starts with minimum CLI options
  - Host-name
  - System-ip
    - Does not need to be routable, just a unique Router-ID
  - Site-id
    - Devices in the same site don't form IPsec tunnels with each other
  - Organization-name
    - Must match the ORG in ***serialFile.viptela*** generated from Cisco Licensing Portal
  - vBond IP Address
    - I.e. Who is the Orchestrator?
  - VPN 0 - The "Transport VPN"
    - Interface(s), IP address(es), and routing towards the WAN
    - Unique tunnel "color" for each WAN link
      - Color can be used in routing decisions later

vBond Example Initial CLI Config

```
config t
!
system
 host-name vBond-1
 system-ip 172.17.101.103
 site-id 1
 organization-name VIPTELA.local
 vbond 150.1.1.103 local
!
vpn 0
 interface ge0/0
  ip address 150.1.1.103/24
  tunnel-interface
   encapsulation ipsec
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.1.1.254
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
!
commit and-quit
```

vSmart Example Initial CLI Config

```
config t
!
system
 host-name vSmart-1
 system-ip 172.17.101.102
 site-id 1
 organization-name VIPTELA.local
 vbond 150.1.1.103
!
vpn 0
 no interface eth0
 !
 interface eth1
  ip address 150.1.1.102/24
  tunnel-interface
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.1.1.254
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
!
commit and-quit
```

vManage Example Initial CLI Config

```
config t
!
system
 host-name vManage-1
 system-ip 172.17.101.101
 site-id 1
 organization-name VIPTELA.local
 vbond 150.1.1.103
!
vpn 0
 no interface eth0
 !
 interface eth1
  ip address 150.1.1.101/24
  tunnel-interface
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.1.1.254
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
!
commit and-quit
```

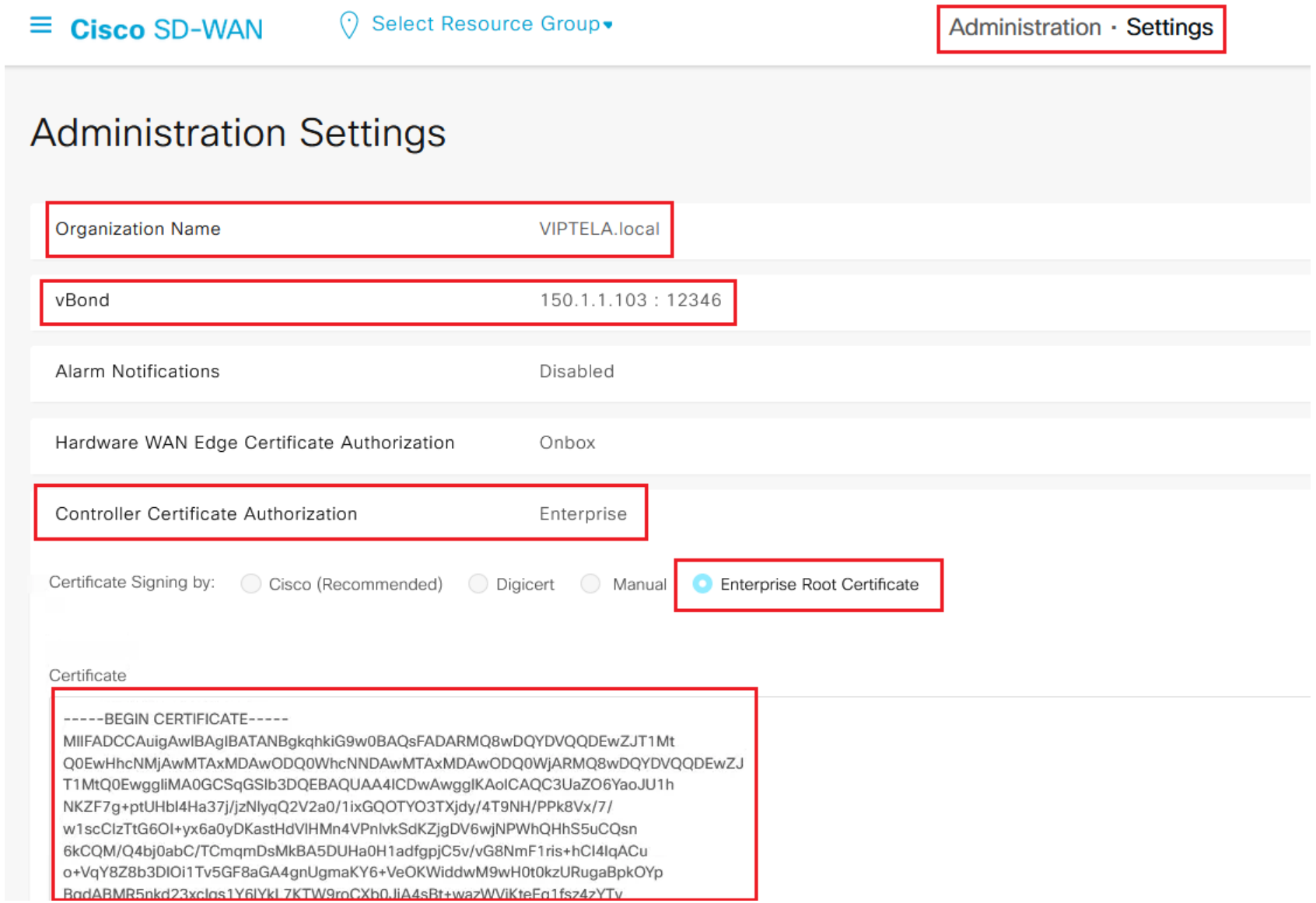
Enabling Enterprise Certificates on vManage

- vManage listens for HTTPS for GUI management via web browser
  - Allowed in Management VPN (512) by default
  - Can be allowed in Transport VPN (0) with **`allow-service [https | all]`** under CLI **`conf t ; vpn 0 ; interface [int] ; tunnel-interface`**

- vManage trusts Cisco's Certificate Authority (CA) by default
  - Allows for very simple Zero Touch Provisioning (ZTP)

- For internally hosted controllers, set to ***Enterprise Root Certificate*** on vManage
  - Administration > Settings
    - Set org, e.g. VIPTELA.local
    - Set vBond IP address
    - Set *Controller Certificate Authorization* to *Enterprise Root Certificate*
    - Paste your private Root CA Certificate

Enabling Enterprise Certificates on vManage – GUI Example



Trusting the Root CA Certificate on the SD-WAN Controllers

- All three Controllers must manually install your private Root CA Cert
  - Run **`vshell`** from Controllers CLI to drop to Linux shell
  - Copy the Root CA Cert to filesystem with SCP/TFTP
  - Could also use **`vi MyCA.crt`** and paste Cert in
  - When complete, **`exit`** vshell

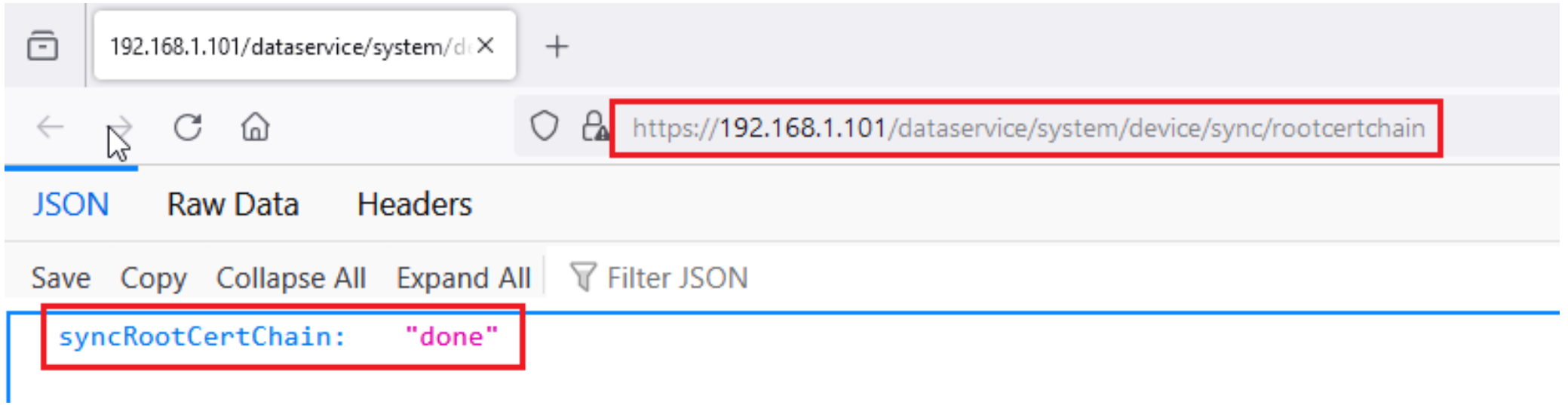
- Install the Root CA Cert from all Controllers CLI
  - **`vManage-1# request root-cert-chain install /home/admin/MyCA.crt`**

- Sync the Root CA Cert in the vManage database (required)
  - https://<vManage-ip-address>/dataservice/system/device/sync/rootcertchain

Trusting the Root CA Certificate on the SD-WAN Controllers – CLI Example

```
vManage-1# vshell
vManage-1:~$ more IOS-CA.crt
-----BEGIN CERTIFICATE-----
MIIFADCCAuigAwIBAgIBATANBgkqhkiG9w0BAQsFADARMQ8wDQYDVQQDEwZJT1Mt
Q0EwHhcNMjAwMTAxMDAwODQ0WhcNNDAwMTAxMDAwODQ0WjARMQ8wDQYDVQQDEwZJ
T1MtQ0EwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQC3UaZO6YaoJU1h
NKZF7g+ptUHbl4Ha37j/jzNlyqQ2V2a0/1ixGQOTYO3TXjdy/4T9NH/PPk8Vx/7/
w1scCIzTtG6OI+yx6a0yDKastHdVlHMn4VPnIvkSdKZjgDV6wjNPWhQHhS5uCQsn
6kCQM/Q4bj0abC/TCmqmDsMkBA5DUHa0H1adfgpjC5v/vG8NmF1ris+hCI4lqACu
o+VqY8Z8b3DIOi1Tv5GF8aGA4gnUgmaKY6+VeOKWiddwM9wH0t0kzURugaBpkOYp
<snip>

vManage-1:~$ exit
vManage-1# request root-cert-chain install /home/admin/IOS-CA.crt
Uploading root-ca-cert-chain via VPN 0
Copying ... /home/admin/IOS-CA.crt via VPN 0
Updating the root certificate chain..
Successfully installed the root certificate chain
```

Syncing the Root CA Cert in the vManage Database Example



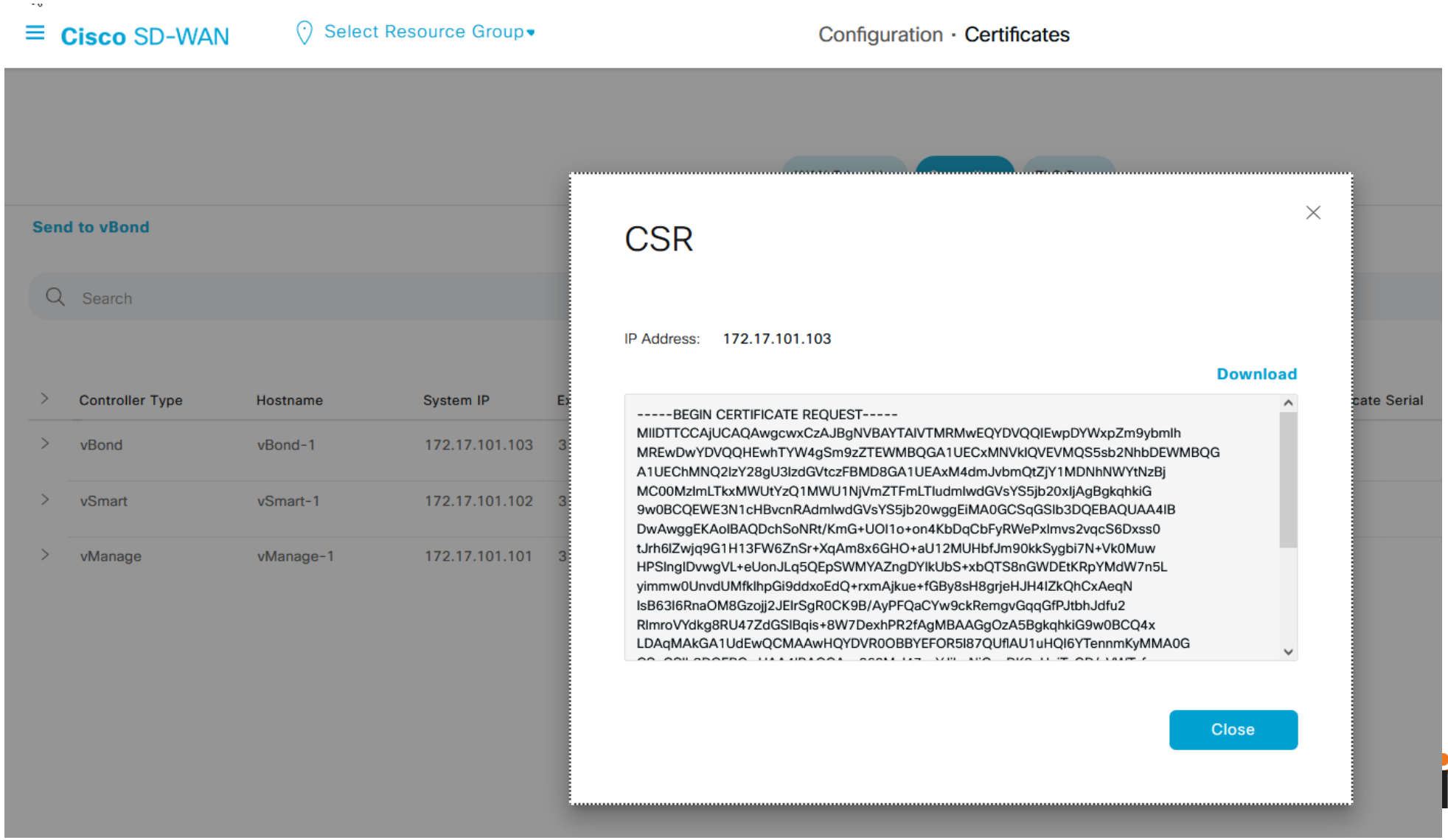
Signing the Controllers Certificates with Internal Certificate Authority (CA)

- All controllers must generate Certificate Signing Requests (CSRs), which are then signed by the internal Root CA

- First, add the new vSmart & vBond Controllers from vManage
  - vManage > Configuration > Devices > Controllers > Add Controller
  - Add vBond IP address, credentials, and Generate CSR
  - Add vSmart IP address, credentials, and Generate CSR

- Next, view the Certificate Signing Requests from vManage
  - vManage > Configuration > Certificates > Controllers > • • • > View CSR | Generate CSR

SD-WAN Controllers Certificate Signing Request Example

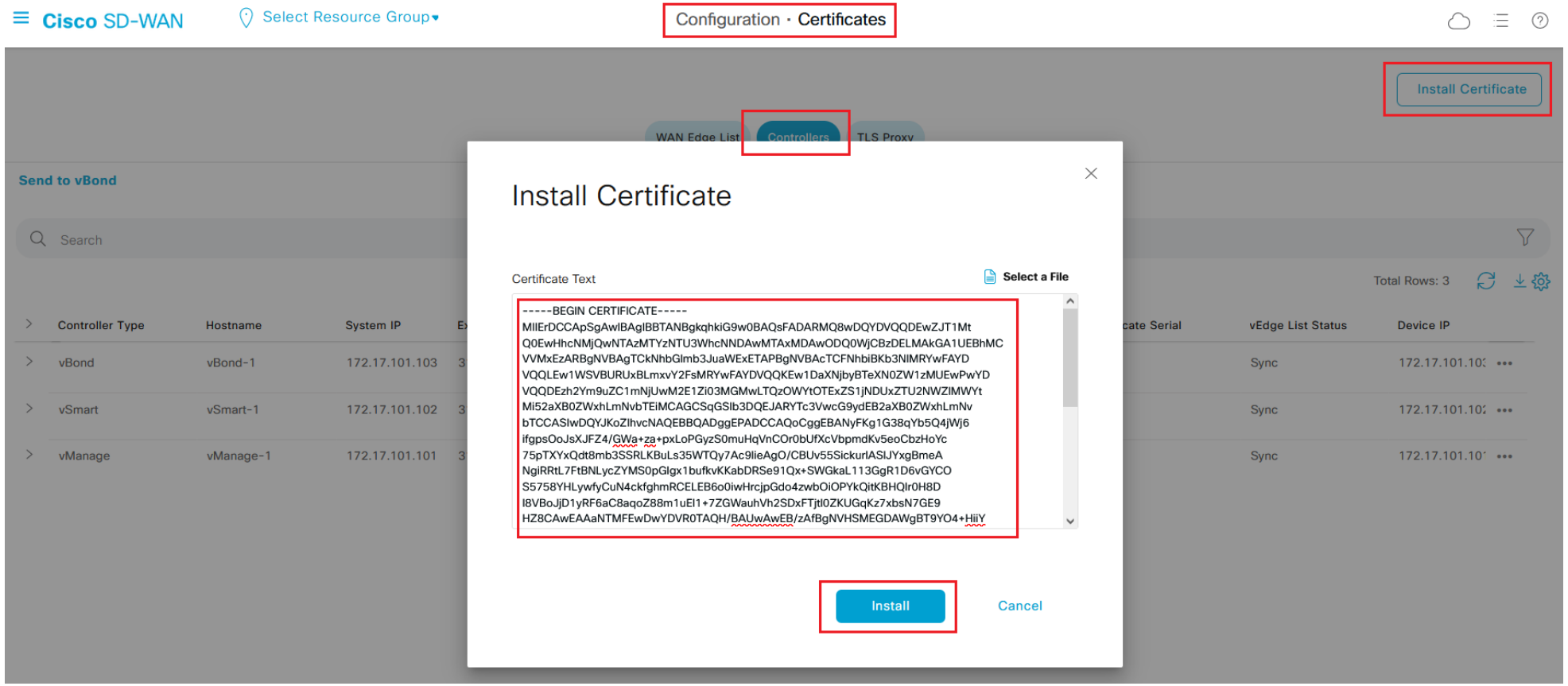


Using Cisco IOS as a Root CA Server Example – Signing the SD-WAN Controllers Certificates

```
IOS-CA#crypto pki server IOS-CA 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-----
MIIDTTCCAjUCAQAwgcwxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlh
<snip>
vFRHP1aMMUWAn6XutBKZCiq4mGKqkhIRuR8ln81EFW4Y
-----END CERTIFICATE REQUEST-----

% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIErDCCApSgAwIBAgIBBTANBgkqhkiG9w0BAQsFADARMQ8wDQYDVQQDEwZJT1Mt
<snip>
W6e1iHB8FZLcR7/8yin9nxSOw5xYGwe7USb/+0ZjmzVGByIYP/0zGkq7wL8R4G10
-----END CERTIFICATE-----

IOS-CA#
```

Installing the SD-WAN Controllers Signed Certificates

- vManage > Configuration > Certificates > Controllers > Install Certificate
  - Paste or upload each of the Controllers' Certificates granted from the Root CA

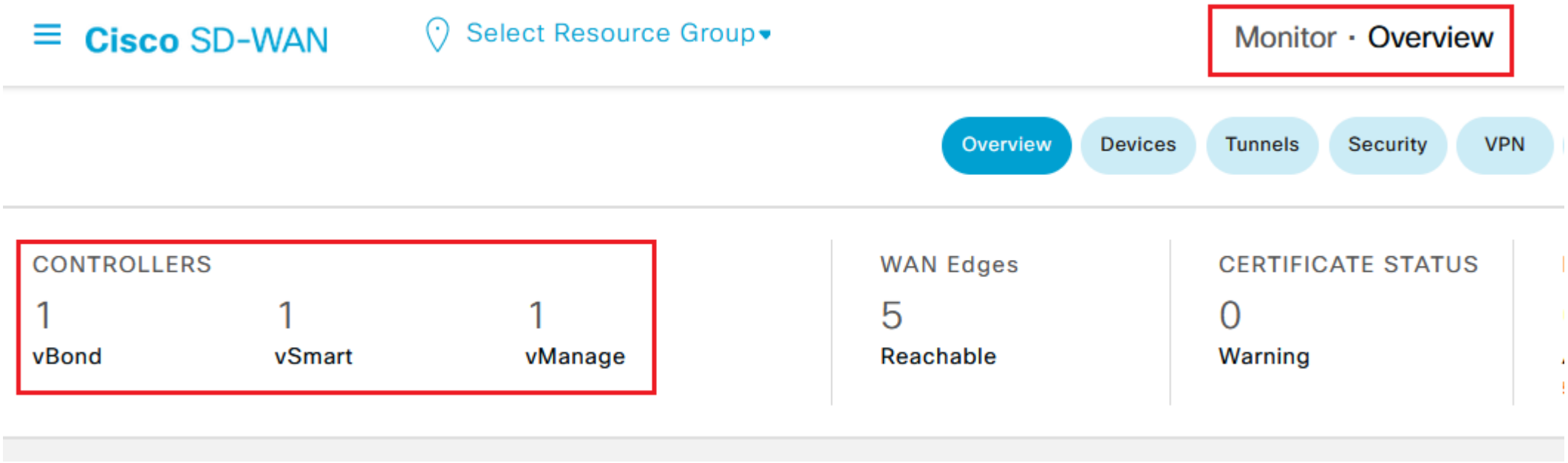


Verifying SD-WAN Controller Onboarding from CLI

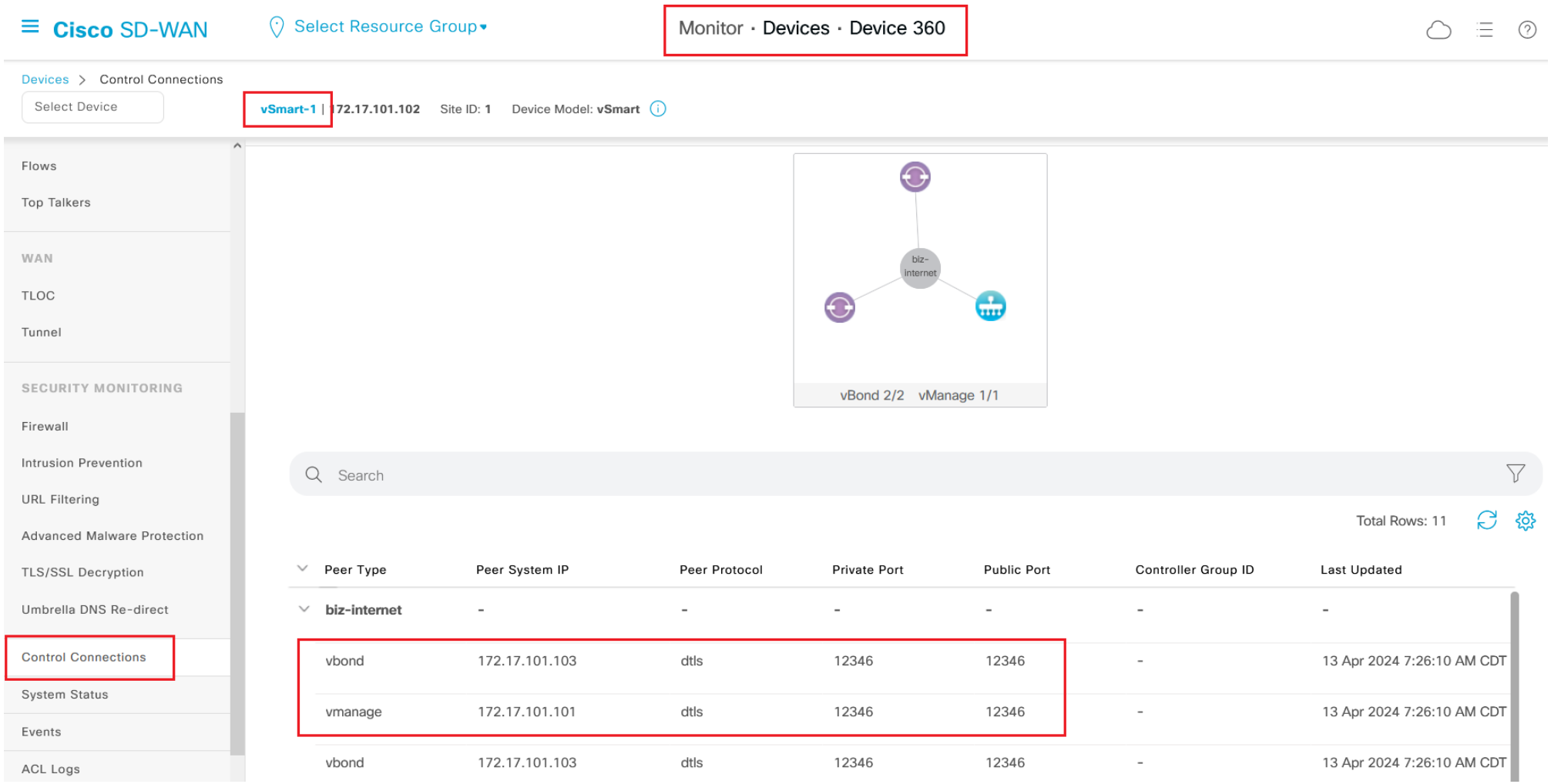
- Once PKI Authentication is complete, DTLS tunnels form between all Controllers
  - vManage forms one tunnel per-vCPU to vBond

```
vManage-1# show control connections | exclude vedge | tab

                                                    LOCAL
          PEER     SITE   DOMAIN   LOCAL           PRIVATE                    PUBLIC
INSTANCE  TYPE     ID     ID       PRIVATE IP      PORT      PUBLIC IP        PORT      <snip>
-------------------------------------------------------------------------------------
0         vsmart   1      1        150.1.1.101     12346     150.1.1.102      12346
0         vbond    0      0        150.1.1.101     12346     150.1.1.103      12346
1         vbond    0      0        150.1.1.101     12446     150.1.1.103      12346
2         vbond    0      0        150.1.1.101     12546     150.1.1.103      12346
3         vbond    0      0        150.1.1.101     12646     150.1.1.103      12346
4         vbond    0      0        150.1.1.101     12746     150.1.1.103      12346
5         vbond    0      0        150.1.1.101     12846     150.1.1.103      12346
6         vbond    0      0        150.1.1.101     12946     150.1.1.103      12346
7         vbond    0      0        150.1.1.101     13046     150.1.1.103      12346
```

Verifying SD-WAN Controller Onboarding from vManage GUI



Verifying SD-WAN Controller Onboarding from vManage GUI (cont.)



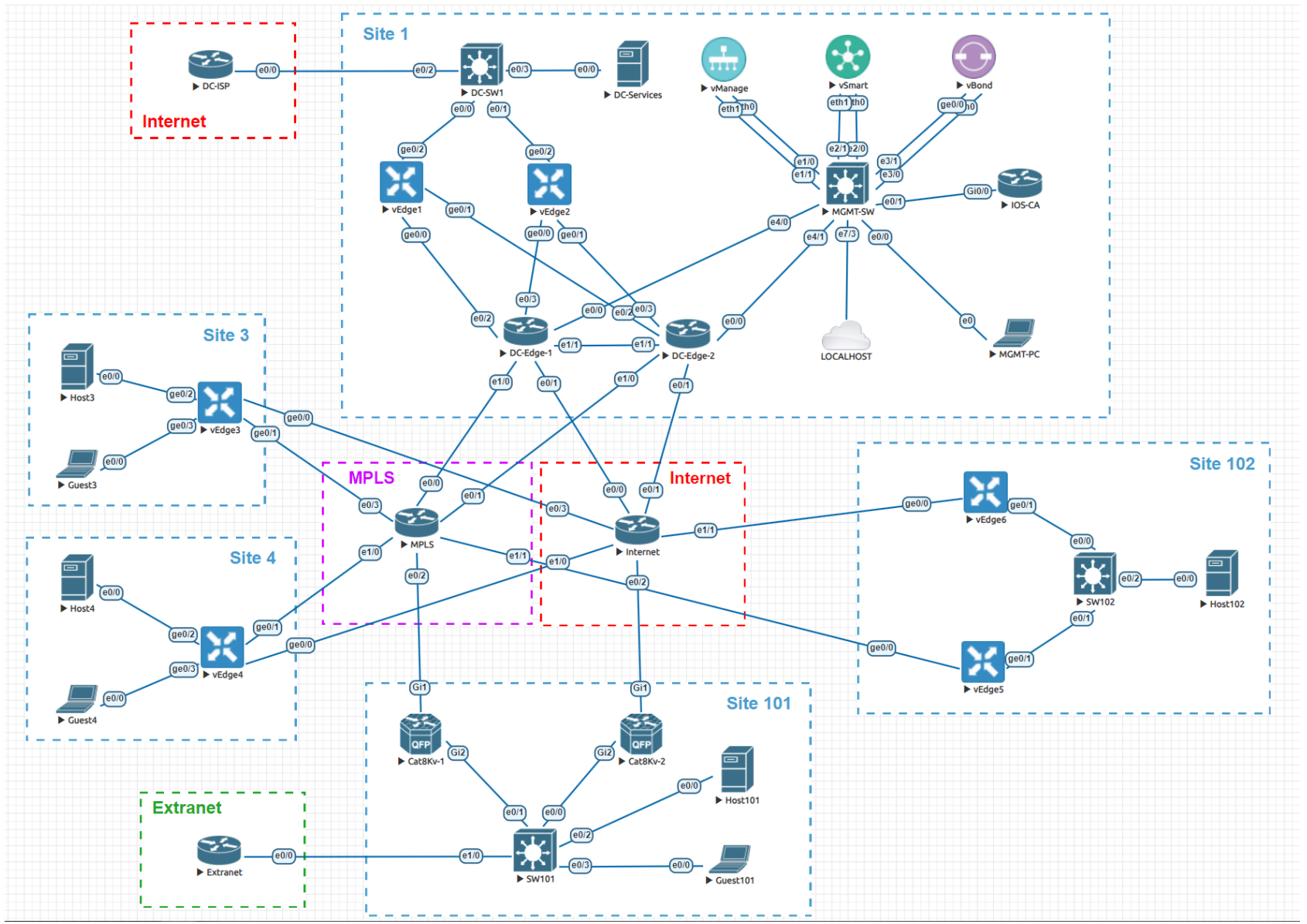
Verifying & Troubleshooting SD-WAN Controller Onboarding from CLI

- Verify that DTLS connections are up between Controllers
  - **`show control connections`** from vManage, vSmart, & vEdge
  - **`show orchestrator connections`** from vBond
- Troubleshoot failed onboarding of Controllers
  - **`show control connections-history`** from vManage, vSmart, & vEdge
  - **`show orchestrator connections-history`** from vBond
  - Check **`LOCAL ERROR`** & **`REMOTE ERROR`** fields against legend

Implementing Cisco SD-WAN

Cisco SD-WAN Controller Onboarding Example

Example Cisco SD-WAN Topology



vBond Example Initial CLI Config

```
config t
!
system
 host-name vBond-1
 system-ip 172.17.101.103
 site-id 1
 organization-name VIPTELA.local
 vbond 150.1.1.103 local
!
vpn 0
 interface ge0/0
  ip address 150.1.1.103/24
  tunnel-interface
   encapsulation ipsec
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.1.1.254
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
!
commit and-quit
```

vSmart Example Initial CLI Config

```
config t
!
system
 host-name vSmart-1
 system-ip 172.17.101.102
 site-id 1
 organization-name VIPTELA.local
 vbond 150.1.1.103
!
vpn 0
 no interface eth0
 !
 interface eth1
  ip address 150.1.1.102/24
  tunnel-interface
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.1.1.254
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
!
commit and-quit
```

vManage Example Initial CLI Config

```
config t
!
system
 host-name vManage-1
 system-ip 172.17.101.101
 site-id 1
 organization-name VIPTELA.local
 vbond 150.1.1.103
!
vpn 0
 no interface eth0
 !
 interface eth1
  ip address 150.1.1.101/24
  tunnel-interface
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.1.1.254
!
vpn 512
 interface eth0
  ip dhcp-client
  no shutdown
!
commit and-quit
```

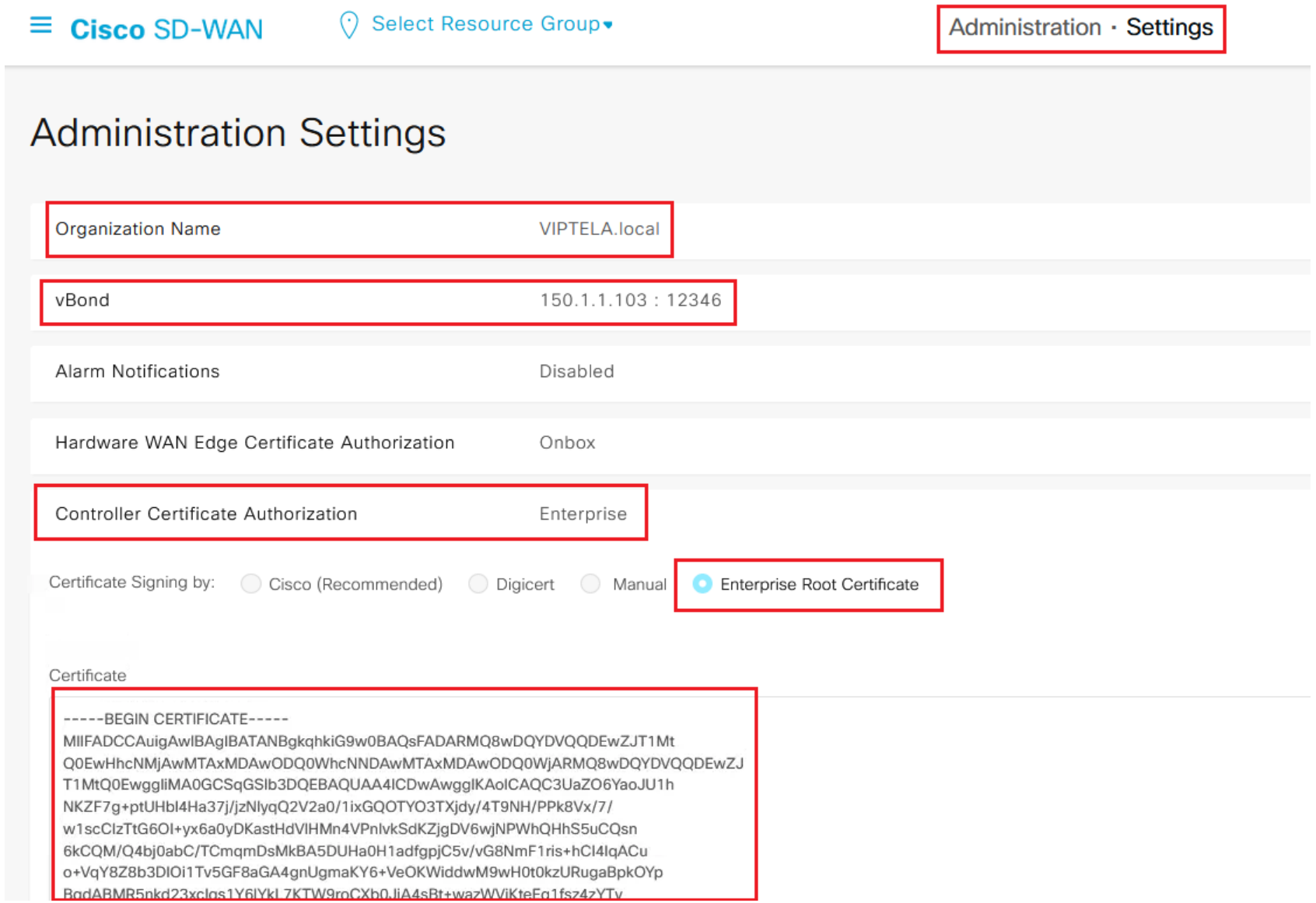
Enabling Enterprise Certificates on vManage

- vManage listens for HTTPS for GUI management via web browser
  - Allowed in Management VPN (512) by default
  - Can be allowed in Transport VPN (0) with **`allow-service [https | all]`** under CLI **`conf t ; vpn 0 ; interface [int] ; tunnel-interface`**

- vManage trusts Cisco's Certificate Authority (CA) by default
  - Allows for very simple Zero Touch Provisioning (ZTP)

- For internally hosted controllers, set to ***Enterprise Root Certificate*** on vManage
  - Administration > Settings
    - Set org, e.g. VIPTELA.local
    - Set vBond IP address
    - Set *Controller Certificate Authorization* to *Enterprise Root Certificate*
    - Paste your private Root CA Certificate

Enabling Enterprise Certificates on vManage – GUI Example



Trusting the Root CA Certificate on the SD-WAN Controllers

- All three Controllers must manually install your private Root CA Cert
  - Run **`vshell`** from Controllers CLI to drop to Linux shell
  - Copy the Root CA Cert to filesystem with SCP/TFTP
  - Could also use **`vi MyCA.crt`** and paste Cert in
  - When complete, **`exit`** vshell

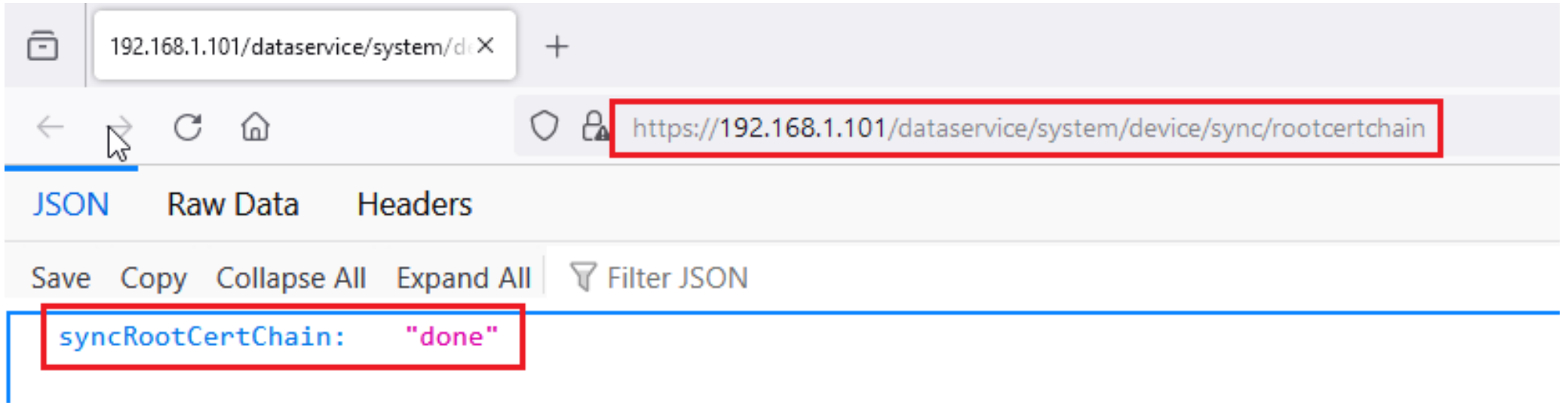
- Install the Root CA Cert from all Controllers CLI
  - **`vManage-1# request root-cert-chain install /home/admin/MyCA.crt`**

- Sync the Root CA Cert in the vManage database (required)
  - https://<vManage-ip-address>/dataservice/system/device/sync/rootcertchain

Trusting the Root CA Certificate on the SD-WAN Controllers – CLI Example

```
vManage-1# vshell
vManage-1:~$ more IOS-CA.crt
-----BEGIN CERTIFICATE-----
MIIFADCCAuigAwIBAgIBATANBgkqhkiG9w0BAQsFADARMQ8wDQYDVQQDEwZJT1Mt
Q0EwHhcNMjAwMTAxMDAwODQ0WhcNNDAwMTAxMDAwODQ0WjARMQ8wDQYDVQQDEwZJ
T1MtQ0EwggIiMA0GCSqGSIb3DQEBAQUAA4ICDwAwggIKAoICAQC3UaZO6YaoJU1h
NKZF7g+ptUHbl4Ha37j/jzNlyqQ2V2a0/1ixGQOTYO3TXjdy/4T9NH/PPk8Vx/7/
w1scCIzTtG6OI+yx6a0yDKastHdVlHMn4VPnIvkSdKZjgDV6wjNPWhQHhS5uCQsn
6kCQM/Q4bj0abC/TCmqmDsMkBA5DUHa0H1adfgpjC5v/vG8NmF1ris+hCI4lqACu
o+VqY8Z8b3DIOi1Tv5GF8aGA4gnUgmaKY6+VeOKWiddwM9wH0t0kzURugaBpkOYp
<snip>

vManage-1:~$ exit
vManage-1# request root-cert-chain install /home/admin/IOS-CA.crt
Uploading root-ca-cert-chain via VPN 0
Copying ... /home/admin/IOS-CA.crt via VPN 0
Updating the root certificate chain..
Successfully installed the root certificate chain
```

Syncing the Root CA Cert in the vManage Database Example



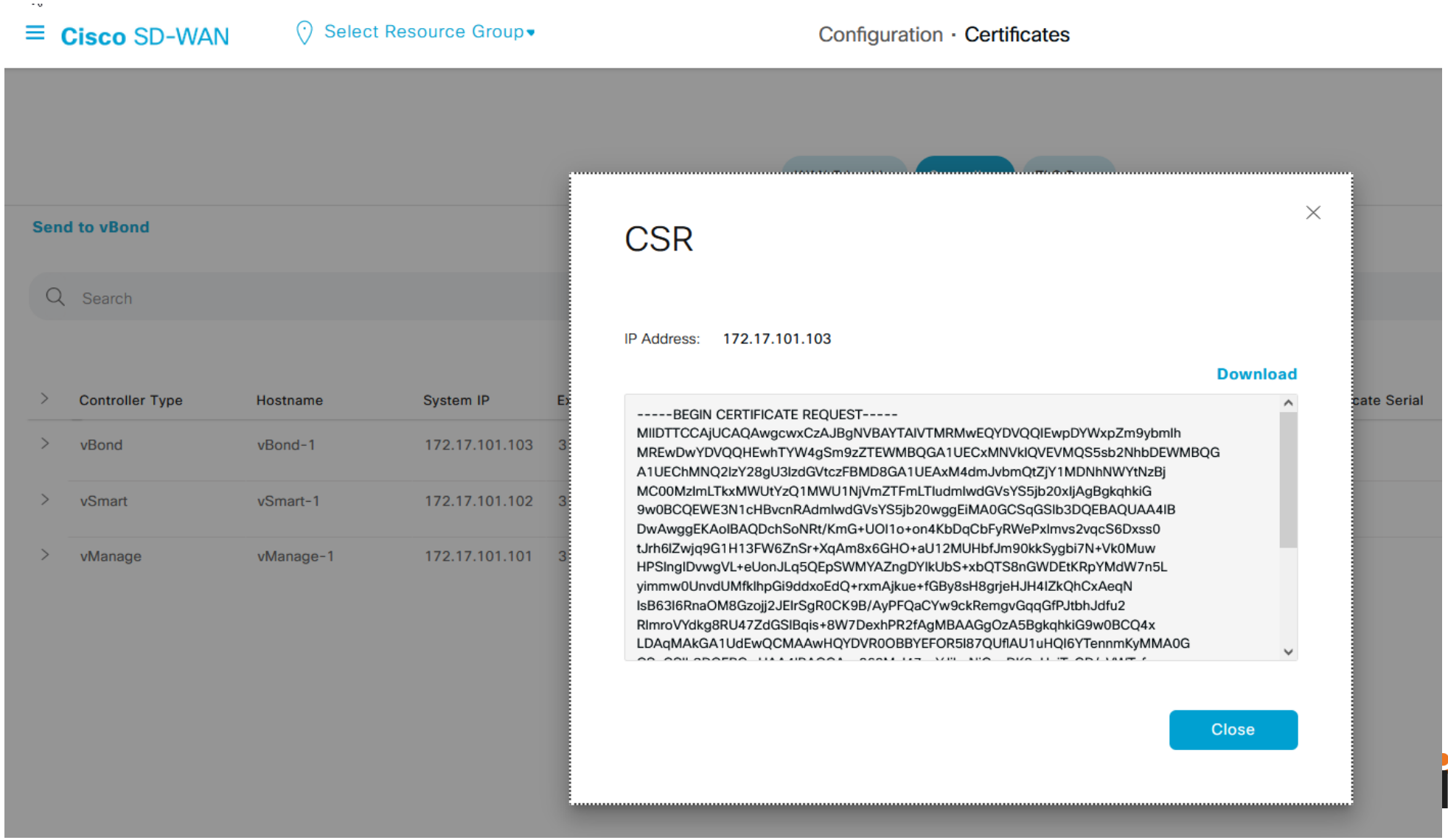
Signing the Controllers Certificates with Internal Certificate Authority (CA)

- All controllers must generate Certificate Signing Requests (CSRs), which are then signed by the internal Root CA

- First, add the new vSmart & vBond Controllers from vManage
  - vManage > Configuration > Devices > Controllers > Add Controller
  - Add vBond IP address, credentials, and Generate CSR
  - Add vSmart IP address, credentials, and Generate CSR

- Next, view the Certificate Signing Requests from vManage
  - vManage > Configuration > Certificates > Controllers > • • • > View CSR | Generate CSR

SD-WAN Controllers Certificate Signing Request Example

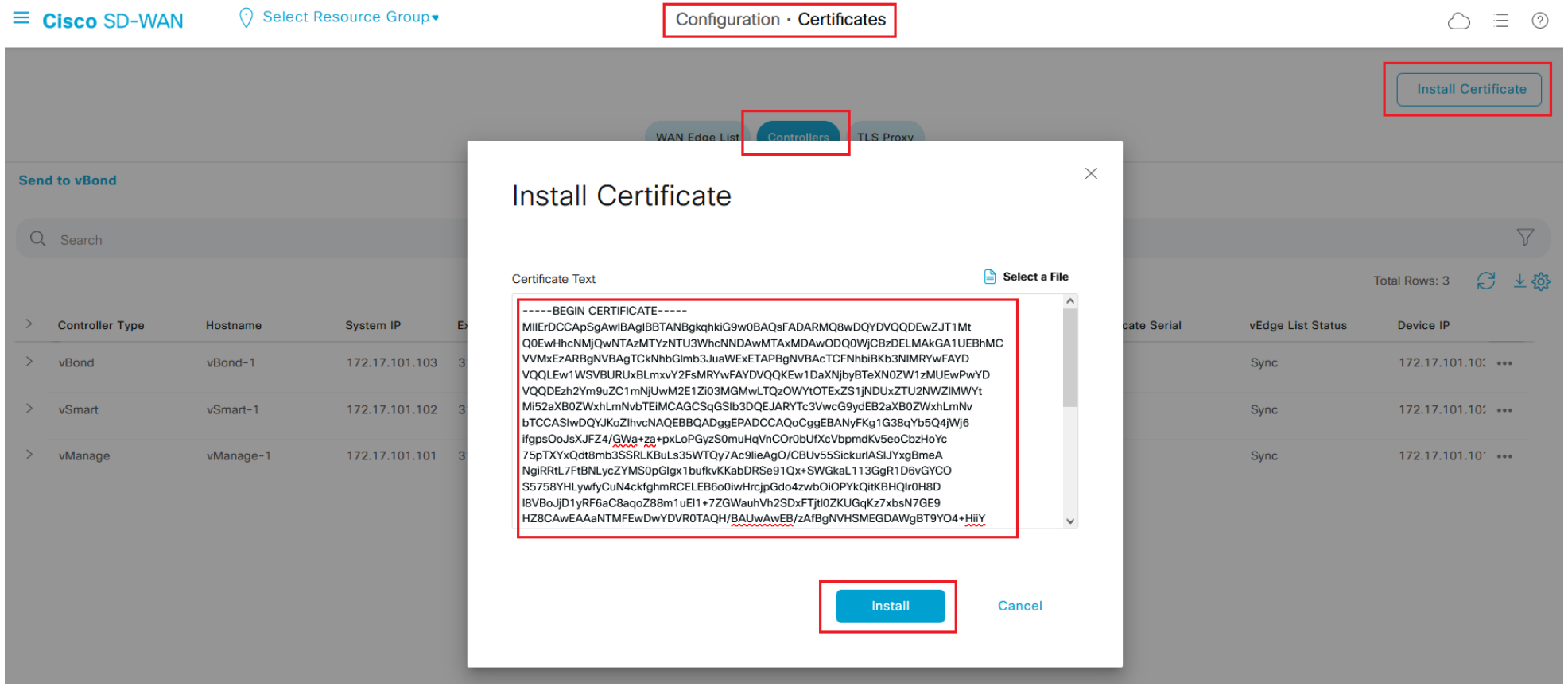


Using Cisco IOS as a Root CA Server Example – Signing the SD-WAN Controllers Certificates

```
IOS-CA#crypto pki server IOS-CA 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-----
MIIDTTCCAjUCAQAwgcwxCzAJBgNVBAYTAlVTMRMwEQYDVQQIEwpDYWxpZm9ybmlh
<snip>
vFRHP1aMMUWAn6XutBKZCiq4mGKqkhIRuR8ln81EFW4Y
-----END CERTIFICATE REQUEST-----

% Granted certificate:
-----BEGIN CERTIFICATE-----
MIIErDCCApSgAwIBAgIBBTANBgkqhkiG9w0BAQsFADARMQ8wDQYDVQQDEwZJT1Mt
<snip>
W6e1iHB8FZLcR7/8yin9nxSOw5xYGwe7USb/+0ZjmzVGByIYP/0zGkq7wL8R4G10
-----END CERTIFICATE-----

IOS-CA#
```

Installing the SD-WAN Controllers Signed Certificates

- vManage > Configuration > Certificates > Controllers > Install Certificate
  - Paste or upload each of the Controllers' Certificates granted from the Root CA

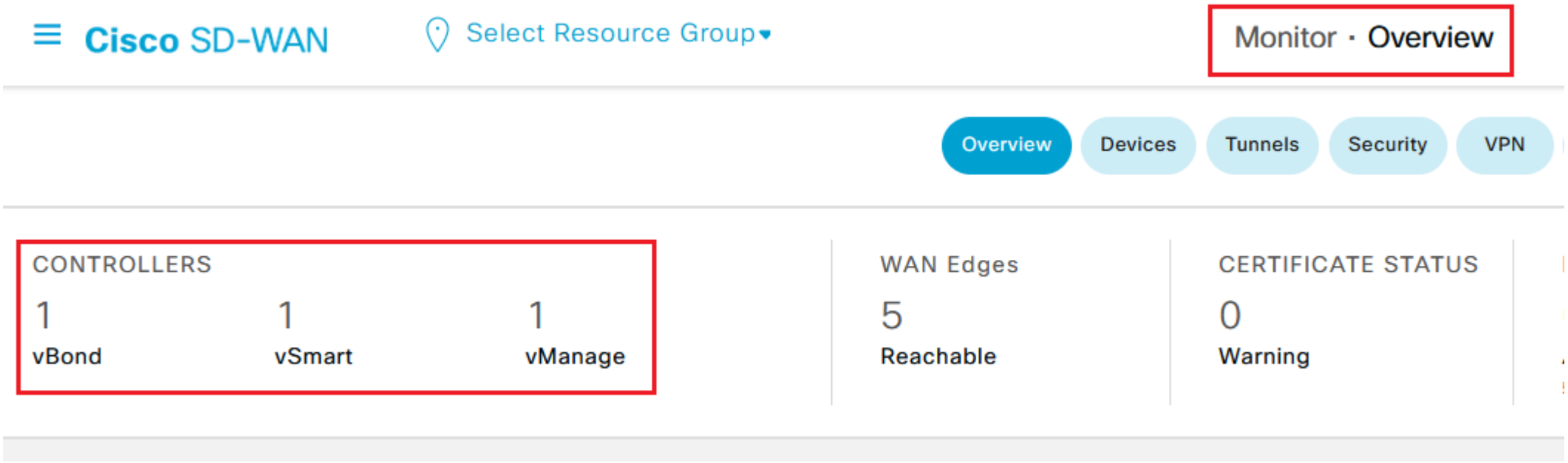


Verifying SD-WAN Controller Onboarding from CLI

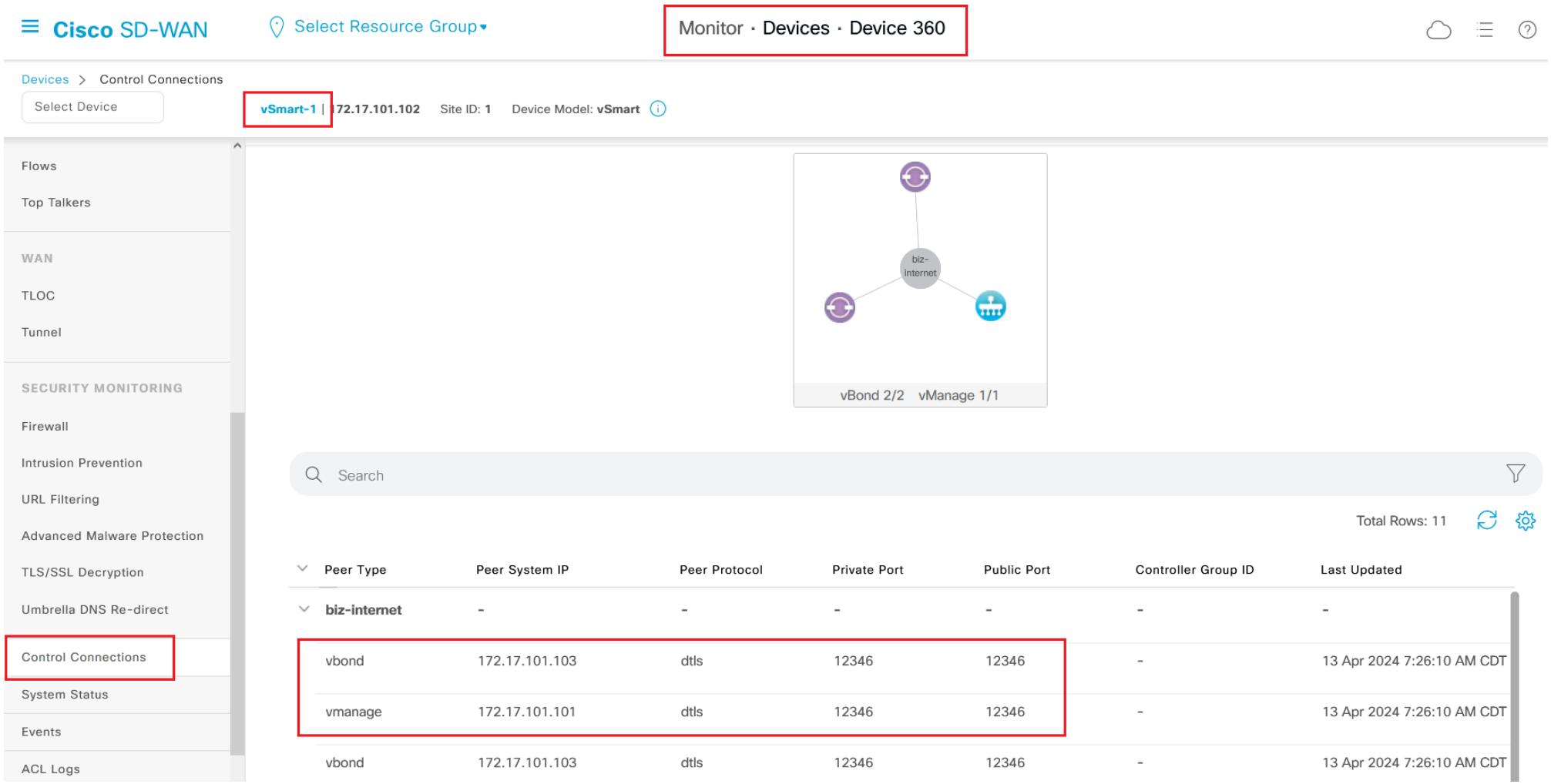
- Once PKI Authentication is complete, DTLS tunnels form between all Controllers
  - vManage forms one tunnel per-vCPU to vBond

```
vManage-1# show control connections | exclude vedge | tab

                                                    LOCAL
          PEER     SITE   DOMAIN   LOCAL           PRIVATE                    PUBLIC
INSTANCE  TYPE     ID     ID       PRIVATE IP      PORT      PUBLIC IP        PORT      <snip>
-------------------------------------------------------------------------------------
0         vsmart   1      1        150.1.1.101     12346     150.1.1.102      12346
0         vbond    0      0        150.1.1.101     12346     150.1.1.103      12346
1         vbond    0      0        150.1.1.101     12446     150.1.1.103      12346
2         vbond    0      0        150.1.1.101     12546     150.1.1.103      12346
3         vbond    0      0        150.1.1.101     12646     150.1.1.103      12346
4         vbond    0      0        150.1.1.101     12746     150.1.1.103      12346
5         vbond    0      0        150.1.1.101     12846     150.1.1.103      12346
6         vbond    0      0        150.1.1.101     12946     150.1.1.103      12346
7         vbond    0      0        150.1.1.101     13046     150.1.1.103      12346
```

Verifying SD-WAN Controller Onboarding from vManage GUI



Verifying SD-WAN Controller Onboarding from vManage GUI (cont.)



Verifying & Troubleshooting SD-WAN Controller Onboarding from CLI

- Verify that DTLS connections are up between Controllers
  - **`show control connections`** from vManage, vSmart, & vEdge
  - **`show orchestrator connections`** from vBond
- Troubleshoot failed onboarding of Controllers
  - **`show control connections-history`** from vManage, vSmart, & vEdge
  - **`show orchestrator connections-history`** from vBond
  - Check **`LOCAL ERROR`** & **`REMOTE ERROR`** fields against legend

Implementing Cisco SD-WAN

Onboarding Cisco SD-WAN vEdge Routers

Cisco SD-WAN Edge Routers Overview

- Cisco SD-WAN Edge Routers come in two variations
  - vEdge Routers running Viptela OS
    - End-of-Sale Jan 2023, but supported until Jan 2028
  - cEdge Routers running Cisco IOS XE SD-WAN

- Both vEdge & cEdge Routers come in physical and virtual form factors
  - E.g. vEdge-1000 vs. vEdge Cloud
  - E.g. Cisco ISR 4000 vs. Catalyst 8000v

- Features are similar between vEdge & cEdge, but different syntax

Onboarding Cisco SD-WAN vEdge Routers

- Onboarding vEdge Routers starts with minimum CLI options
  - Host-name
  - System-ip
    - Does not need to be routable, just a unique Router-ID
  - Site-id
    - Devices in the same site don't form IPsec tunnels with each other
  - Organization-name
    - Must match the ORG in ***serialFile.viptela*** generated from Cisco Licensing Portal
  - vBond IP Address
    - I.e. Who is the Orchestrator?
  - VPN 0 - The "Transport VPN"
    - Interface(s), IP address(es), and routing towards the WAN
    - Unique tunnel "color" for each WAN link
      - Color can be used in routing decisions later

Example vEdge Router Initial CLI Config

```
config
!
system
 host-name vEdge-2
 system-ip 172.17.2.2
 site-id 2
 organization-name VIPTELA.local
 vbond 150.1.1.103
!
vpn 0
 interface ge0/0
  ip address 150.11.1.0/31
  tunnel-interface
   encapsulation ipsec
   color biz-internet
   allow-service all
  no shutdown
 !
 ip route 0.0.0.0/0 150.11.1.1
!
commit and-quit
```

Installing a Private Root CA Certificate on the vEdge Router

- If not using Cisco Cloud for PKI, install the Root CA Cert:
  - **vEdge-2# vshell**
  - **vEdge-2:~$ vi MyCA.crt**
    - "i" to insert in vi
    - Paste the Root CA Certificate
    - "<esc> :wq" to save and quit
  - **vEdge-2:~$ exit**
  - **vEdge-2# request root-cert-chain install /home/admin/MyCA.crt**

Understanding the WAN Edge List

- vBond Orchestrator needs to know the list of Chassis Numbers & Serial Numbers of the WAN Edge Routers to authenticate & onboard them
  - Can be done automatically through vManage sync to Cisco Smart Licensing
  - Done manually by uploading the ***serialFile.viptela*** from Cisco Licensing Portal
    - vManage > Configuration > Devices > Upload WAN Edge List
    - Check the box to "*send to controllers*" to sync to vBond
- vBond must have the WAN Edge List synchronized with vManage
  - WAN Edge List not synced will result DTLS tunnel failure
    - I.e. onboarding fails if vBond can't authenticate the WAN Edge Router
  - Can be re-synced from vManage > Configuration > Certificates > WAN Edge List > Send to Controllers
  - Verified with **`show orchestrator valid-vedges`** from vBond CLI

Onboarding the vEdge Router from vManage

- Once the WAN Edge List is uploaded and synced to vBond, goto vManage > Configuration > Devices
  - Choose the appropriate Chassis Number from the list and then "Generate Bootstrap Configuration" from the ellipses on the right
    - For vEdge Cloud we can choose any used Chassis Number
  - Select "Cloud-Init" and click OK
  - Copy the UUID and OTP fields
- Enter the following command on the vEdge CLI:
  - **`request vedge-cloud activate chassis-number` *`UUID`* `token` *`OTP`***
    where *UUID* and *OTP* are the strings from the Bootstrap Configuration

vEdge Onboarding CLI Verifications

- Was the Certificate granted?
  - **`show control local-properties`**
  - Should show "*certificate-status Installed*" and "*token Invalid*"
- If not, what error code was generated?
  - **`show control connections-history`**
- Did the vEdge form DTLS tunnels to vSmart/vManage/vBond?
  - **`show control connections`**
- Have IPsec tunnels formed to the other sites?
  - **`show bfd sessions`**

Implementing Cisco SD-WAN

Onboarding Cisco SD-WAN cEdge Routers

Cisco SD-WAN Edge Routers Overview

- Cisco SD-WAN Edge Routers come in two variations
  - vEdge Routers running Viptela OS
    - End-of-Sale Jan 2023, but supported until Jan 2028
  - cEdge Routers running Cisco IOS XE SD-WAN

- Both vEdge & cEdge Routers come in physical and virtual form factors
  - E.g. vEdge-1000 vs. vEdge Cloud
  - E.g. Cisco ISR 4000 vs. Catalyst 8000v

- Features are similar between vEdge & cEdge, but different syntax

Understanding IOS XE SD-WAN Mode

- Cisco IOS XE routers don't run in SD-WAN Mode by default
  - Modes are mutually exclusive; can't run both at the same time
  - Enabling SD-WAN mode w/ **`controller-mode enable`** requires a reboot
  - In SD-WAN mode, router uses **`config-transaction`** instead of **`config t`**
    - Changes are saved with **`commit`**

```
router# controller-mode enable
Ensure the BOOT variable points to a valid image
Continue? [confirm]
% Warning: Bootstrap config file needed for Day-0 boot is missing
Do you want to abort? (yes/[no]): no
 Mode change success
%SYS-7-NV_BLOCK_INIT: Initialized the geometry of nvram
%SYS-5-RELOAD: Reload requested by Exec. Reload Reason: Enabling controller-mode.
```

Onboarding Cisco IOS XE SD-WAN cEdge Routers

- Onboarding IOS XE cEdge Routers starts with minimum CLI options
  - Host-name
  - System-ip
    - Does not need to be routable, just a unique Router-ID
  - Site-id
    - Devices in the same site don't form IPsec tunnels with each other
  - Organization-name
    - Must match the ORG in ***serialFile.viptela*** generated from Cisco Licensing Portal
  - vBond IP Address
    - I.e. Who is the Orchestrator?
  - Underlay Transport
    - WAN Interface(s), IP address(es), & routing in the default VRF
    - Unique tunnel "color" for each WAN link
      - Color can be used in routing decisions later

Example IOS XE SD-WAN cEdge Configuration

```
config-transaction
hostname Cat8Kv-1
system
 system-ip 172.17.101.1
 site-id 101
 organization-name VIPTELA.local
 vbond 150.1.1.103
!
interface GigabitEthernet1
 no shutdown
 ip address 10.101.1.1 255.255.255.252
!
ip route 0.0.0.0 0.0.0.0 10.101.1.2
!
interface Tunnel1
 ip unnumbered GigabitEthernet1
 tunnel source GigabitEthernet1
 tunnel mode sdwan
 no shutdown
!
sdwan
 interface GigabitEthernet1
  tunnel-interface
   encapsulation ipsec
   color biz-internet
   allow-service all
!
commit
```

Installing a Private Root CA Certificate on the IOS XE cEdge Router

- If not using Cisco Cloud for PKI, install the Root CA Cert:

```
Cat8Kv-1#copy tftp://192.168.223.127/MyCA.crt bootflash:
Destination filename [MyCA.crt]?
Accessing tftp://192.168.223.127/MyCA.crt...
Loading MyCA.crt from 192.168.223.127 (via GigabitEthernet3): !
[OK - 1245 bytes]

1245 bytes copied in 0.100 secs (12450 bytes/sec)

Cat8Kv-1#request platform software sdwan root-cert-chain install bootflash:MyCA.crt
Uploading root-ca-cert-chain via VPN 0
Copying ... /bootflash/MyCA.crt via VPN 0
Updating the root certificate chain..
Successfully installed the root certificate chain

Cat8Kv-1#
```

Understanding the WAN Edge List

- vBond Orchestrator needs to know the list of Chassis Numbers & Serial Numbers of the WAN Edge Routers to authenticate & onboard them
  - Can be done automatically through vManage sync to Cisco Smart Licensing
  - Done manually by uploading the ***serialFile.viptela*** from Cisco Licensing Portal
    - vManage > Configuration > Devices > Upload WAN Edge List
    - Check the box to "*send to controllers*" to sync to vBond
- vBond must have the WAN Edge List synchronized with vManage
  - WAN Edge List not synced will result DTLS tunnel failure
    - I.e. onboarding fails if vBond can't authenticate the WAN Edge Router
  - Can be re-synced from vManage > Configuration > Certificates > WAN Edge List > Send to Controllers
  - Verified with **`show orchestrator valid-vedges`** from vBond CLI

Onboarding an IOS XE cEdge Router from vManage

- Once the WAN Edge List is uploaded and synced to vBond, goto vManage > Configuration > Devices
  - Choose the appropriate Chassis Number from the list and then "Generate Bootstrap Configuration" from the ellipses on the right
    - For Catalyst 8000v we can choose any used Chassis Number
  - Select "Cloud-Init" and click OK
  - Copy the UUID and OTP fields
- Enter the following command on the IOS XE cEdge CLI from exec mode:
  - **`request platform software sdwan vedge_cloud activate chassis-number`** ***`UUID`*** **`token`** ***`OTP`***, where *UUID* and *OTP* are the strings from the Bootstrap Configuration

Verifying & Troubleshooting IOS XE cEdge Router Onboarding

- Was the Certificate granted?
  - **`show sdwan control local-properties`**
  - Should show "*certificate-status Installed*" and "*token Invalid*"
- If not, what error code was generated?
  - **`show sdwan control connection-history`**
- Did the cEdge form DTLS tunnels to vSmart/vManage/vBond?
  - **`show sdwan control connections`**
- Have IPsec tunnels formed to the other sites?
  - **`show sdwan bfd sessions`**
- Viewing the running config when in SD-WAN Mode:
  - **`show sdwan running-config`**

Implementing Cisco SD-WAN

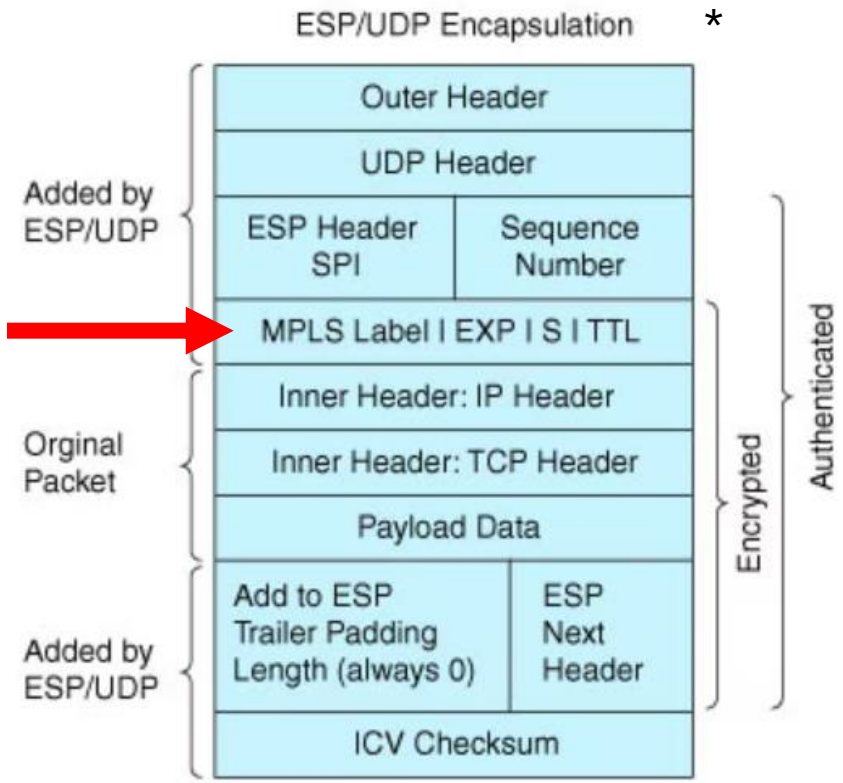
Implementing Cisco SD-WAN Service VPNs

What are Cisco SD-WAN Service VPNs?

- *Service VPNs* are used for traffic sent inside IPsec tunnels between SD-WAN sites
  - E.g. your private-to-private traffic
- *Service VPNs* provide a full-mesh of connectivity within the VPN by default
  - All VPN 1 sites can talk to all other VPN 1 sites, but not VPN 2 sites
  - All VPN 2 sites can talk to all other VPN 2 sites, but not VPN 1 sites
  - More granular routing policies can be defined using the vSmart later…
- The *Service VPN Number* is encoded as an *MPLS Label* in the custom IPsec header
  - Viptela OS (vEdge) uses VPNs 1-511 for Service VPNs
    - 0 reserved for Transport, 512 for MGMT
  - IOS XE (cEdge) uses VRF numbers for the same purpose
    - Global default VRF in IOS XE maps to VPN 0 in Viptela OS
    - VRF 1 maps to VPN 1, VRF 2 maps to VPN 2, etc.

Cisco SD-WAN Service VPNs Data-Plane

- *Service VPN Number* is encoded in the IPsec data-plane as an MPLS Label
  - When an SD-WAN Edge Router receives a packet in an IPsec tunnel, it uses the MPLS Label to find which routing table (VPN/VRF) to do a lookup in



*Security Configuration Guide for vEdge Routers, Cisco SD-WAN

Cisco SD-WAN Service VPNs Routing Control-Plane

- Cisco uses a custom protocol called *Overlay Management Protocol (OMP)* to exchange SD-WAN control-plane routing information
  - Extra attributes like *Service VPN Number* are encoded inside OMP Routes
- Result is that SD-WAN routing is like an MPLS L3VPN (BGP VPNv4 AFI)
  - In L3VPN, BGP VPNv4 routes include a prefix/len, Route Target (RT), & MPLS Label
  - Route Target (RT) defines which routing table the prefix belongs to
    - This is how customer info is segmented in the control-plane for multi-tenancy
    - RT is analogous to the *Service VPN Number* in our SD-WAN solution
  - BGP learned MPLS Label number is used in the data-plane encapsulation
    - Packets sent to the BGP VPNv4 prefix have the MPLS Label number inside
    - Receiving Edge Router uses the MPLS Label to map to the customer's VRF
    - Likewise in SD-WAN, MPLS Label maps to a *Service VPN Number* (VPN/VRF)

Routing over Service VPNs with Overlay Management Protocol (OMP)

- OMP is a BGP-like protocol which advertises a prefix with a set of attributes
  - Attributes include VPN Number, Transport Location (TLOC), Color, Site-ID, Tag, etc.
  - Attributes can be used later for routing policies like Application Aware Routing (AAR)
- OMP runs automatically between the WAN Edge Routers and vSmart Controller
  - Sent over DTLS tunnels from WAN Edge Routers to vSmart formed during onboarding
  - WAN Edge Routers redistribute routes from connected, static, BGP/OSPF/IS-IS/etc. into OMP, and then advertises them to vSmart Controller
    - Redistribute connected by default, plus OSPF Internal on vEdge (but not cEdge)
  - vSmart Controller acts like a BGP Route Reflector
    - WAN Edge Routers do not run OMP directly with each other, only with vSmart
    - vSmart receives routes from WAN Edge Routers, applies any configured policies, runs path selection rules, then reflects routes back to WAN Edge Routers

Example OMP Routes from vSmart Controller

```
vSmart-1# show omp routes | tab
Code:
C   -> chosen
<snip>
R   -> resolved
<snip>


                                        PATH                        ATTRIBUTE
VPN     PREFIX                  FROM PEER       ID      LABEL   STATUS  TYPE        TLOC IP         COLOR
-------------------------------------------------------------------------------------------------------------
1       192.168.3.0/24          172.17.101.1    66      1003    C,R     installed   172.17.101.1    mpls
1       192.168.33.0/24         172.17.3.3      66      1005    C,R     installed   172.17.3.3      mpls
                                172.17.3.3      68      1005    C,R     installed   172.17.3.3      biz-internet
1       192.168.44.0/24         172.17.4.4      66      1005    C,R     installed   172.17.4.4      mpls
                                172.17.4.4      68      1005    C,R     installed   172.17.4.4      biz-internet
1       192.168.101.0/24        172.17.101.1    66      1003    C,R     installed   172.17.101.1    mpls
                                172.17.102.1    68      1003    C,R     installed   172.17.102.1    biz-internet
1       192.168.102.0/24        172.17.5.5      66      1006    C,R     installed   172.17.5.5      mpls
                                172.17.6.6      68      1005    C,R     installed   172.17.6.6      biz-internet

vSmart-1#
```

Example Detailed OMP Route from vSmart Controller

```
vSmart-1# show omp route 192.168.33.0/24 | nomore
-----------------------------------------------------
omp route entries for vpn 1 route 192.168.33.0/24
             RECEIVED FROM:
peer              172.17.3.3
path-id           66
label             1005
status            C,R
<snip>
    Attributes:
     originator        172.17.3.3
     type              installed
     tloc              172.17.3.3, mpls, ipsec
     ultimate-tloc     not set
     domain-id         not set
     overlay-id        1
     site-id           33
     region-id         None
     region-path       not set
     affinity-group    None
     route-reoriginator   not set
     preference        not set
     tag               not set
     origin-proto      connected
     origin-metric     0
     as-path           not set
     community         not set
     unknown-attr-len not set
```

```
             RECEIVED FROM:
peer              172.17.3.3
path-id           68
label             1005
status            C,R
loss-reason       not set
lost-to-peer      not set
lost-to-path-id   not set
    Attributes:
     originator        172.17.3.3
     type              installed
     tloc              172.17.3.3, biz-internet, ipsec
     ultimate-tloc     not set
     domain-id         not set
     overlay-id        1
     site-id           33
     region-id         None
     region-path       not set
     affinity-group    None
     route-reoriginator   not set
     preference        not set
     tag               not set
     origin-proto      connected
     origin-metric     0
     as-path           not set
     community         not set
     unknown-attr-len not set
```

Configuring a Service VPN via the vEdge/cEdge CLI

vEdge:

```
config t
!
vpn 1
 interface ge0/1
  ip address 192.168.2.254/24
  no shutdown
!
commit
```

cEdge:

```
config-transaction
!
vrf definition 1
 rd 1:1
 !
 address-family ipv4
  route-target export 1:1
  route-target import 1:1
 exit-address-family
!
interface GigabitEthernet2
 vrf forwarding 1
 ip address 192.168.3.254 255.255.255.0
 no shutdown
!
commit
```

vEdge/cEdge CLI Verifications

- Are IPsec tunnels up to the other sites?
  - vEdge: **`show bfd sessions`**
  - cEdge: **`show sdwan bfd sessions`**
- Are we learning OMP routes from vSmart?
  - vEdge: **`show ip route [omp]`**
  - cEdge: **`show ip route vrf 1 [omp]`**
  - vEdge: **`show omp route`**
  - cEdge: **`show sdwan omp route`**
  - cEdge: **`show sdwan ip fib`**
- Do we have IP connectivity to the other sites?
  - vEdge: **`ping vpn 1 1.2.3.4 / traceroute vpn 1 1.2.3.4`**
  - cEdge: **`ping vrf 1 1.2.3.4 / traceroute vrf 1 1.2.3.4`**

Implementing Cisco SD-WAN

Understanding Cisco SD-WAN Templates

What are Cisco SD-WAN Device Templates?

- SD-WAN Device Templates are a way to standardize & automate configurations across SD-WAN Edge Routers (Devices)
- Device Templates can be attached to one or more Devices, which are then centrally controlled through vManage
  - Local CLI configuration is disabled once a Device is attached to a Template
- Templates can also be attached before onboarding new WAN Edge Routers
  - I.e. pre-provision the config before the device is onboarded
- Device Templates fall into two main categories:
  - CLI Templates
  - Feature Templates

What are Cisco SD-WAN CLI Templates?

- CLI Templates are a way to automate WAN Edge Routers using standard CLI syntax (IOS XE or Viptela OS) and variable replacements
- CLI Templates can be quickly built from the vManage GUI by using an already onboarded WAN Edge Router's config as an example
  - Create a CLI Template & use **"Load Running config from reachable device"** dropdown
  - Device's running-config will load in a **"Config Preview"** window
  - Highlight a value you want to become a variable, then click **"Create Variable"**
    - CLI Templates use double curly braces to define variables, e.g. **{{var}}**
- When a Device is Attached to a CLI Template, **"Device Specific Values"** are defined
  - You can manually enter the values via the GUI or download/upload a CSV

Cisco SD-WAN CLI Template for cEdge Router (IOS XE) Example

```
hostname {{hostname}}
 system
 system-ip {{System-IP}}
 site-id {{Site-ID}}
 !
 vrf definition {{VPN}}
 rd {{VPN}}:{{VPN}}
 address-family ipv4
  route-target export {{VPN}}:{{VPN}}
  route-target import {{VPN}}:{{VPN}}
  exit-address-family
!
interface GigabitEthernet1
 ip address {{WAN1-IPv4-Addr}}
 no shutdown
!
interface GigabitEthernet2
 vrf forwarding {{VPN}}
 ip address {{LAN-IPv4-Addr}}
 no shutdown
!
ip route 0.0.0.0 0.0.0.0 {{WAN1-Default-GW}}
!
interface Tunnel1
 no shutdown
 ip unnumbered GigabitEthernet1
 tunnel source GigabitEthernet1
 tunnel mode sdwan
!
sdwan
 interface GigabitEthernet1
  tunnel-interface
   encapsulation ipsec
   color {{WAN1-Color}}
   allow-service all
```

Example Completed Cisco SD-WAN CLI Template for cEdge Router (IOS XE)

```
hostname Cat8Kv-1
 system
 system-ip 172.17.101.1
 site-id 101
 !
 vrf definition 1
 rd 1:1
 address-family ipv4
  route-target export 1:1
  route-target import 1:1
  exit-address-family
!
interface GigabitEthernet1
 ip address 10.1.1.1 255.255.255.252
 no shutdown
!
interface GigabitEthernet2
 vrf forwarding 1
 ip address 192.168.1.254 255.255.255.0
 no shutdown
!
ip route 0.0.0.0 0.0.0.0 10.1.1.2
!
interface Tunnel1
 no shutdown
 ip unnumbered GigabitEthernet1
 tunnel source GigabitEthernet1
 tunnel mode sdwan
!
sdwan
 interface GigabitEthernet1
  tunnel-interface
   encapsulation ipsec
   color biz-internet
   allow-service all
```

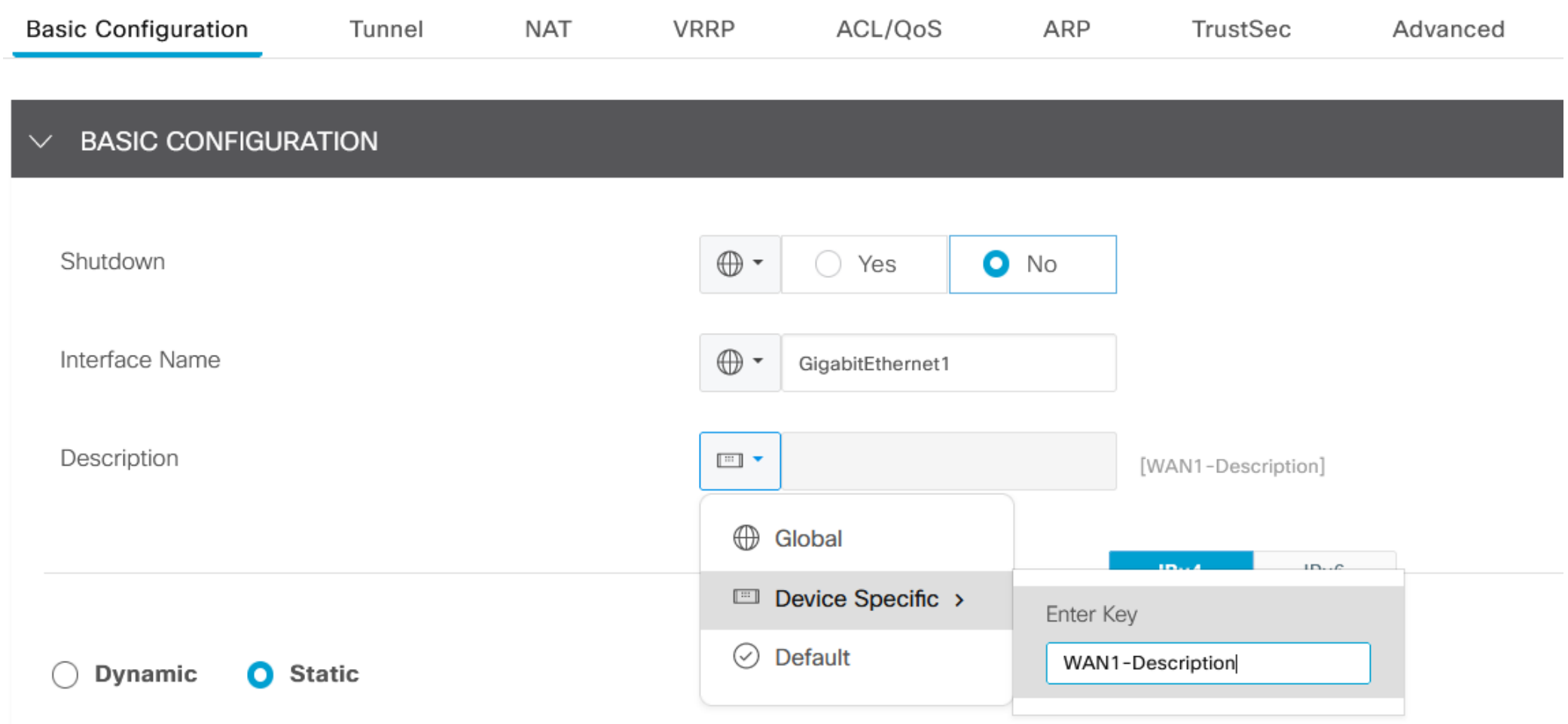
What are Cisco SD-WAN Feature Templates?

- Feature Templates are a way to automate WAN Edge Routers configuration using the vManage GUI in a modular and re-usable fashion
- Feature Templates transform our Intent into the required CLI commands
  - E.g. you can enable OSPF routing without needing to know the exact OSPF CLI
- Feature Templates are specific to Device Models
  - E.g. Viptela OS syntax is different than IOS XE syntax, so they have separate templates
- Feature Templates combine in different ways to form a Device Template
  - Feature Templates can be re-used between multiple Device Templates
  - Device Templates start with a list of default Feature Templates based on Device Type
  - Lots of default Feature Templates that you can **Copy** and customize

Feature Template Variables

- Like CLI Templates, Feature Templates use variable replacement to customize the WAN Edge Routers attached to the Device Template
  - E.g. different WAN Edge Routers have different IP addresses
- Feature Templates have **Global** variables and **Device Specific** variables
  - **Global** variables apply to all Devices attached to the Device Template
    - E.g. all routers use GigabitEthernet1 as the WAN interface
  - **Device Specific** variables must be manually defined for each Device attached
    - Variable names can be customized inside the Feature Template
    - Values are defined manually when you attach a Device to a Template through the GUI, or you can download/upload values in CSV format

Feature Template Global vs. Device Specific Variable Example



Attaching Devices to Templates

- After **Device Specific** variable values are defined, vManage allows you to preview the config & config-diff of changes being pushed to Devices
  - Config Diff shows highlighted deletions in red, and additions in green

- Once you approve the changes, vManage pushes the config over DTLS
  - If the configuration fails (i.e. syntax is rejected), it will automatically rollback
    - E.g. we entered a bad subnet mask for an interface in a Device Specific variable
  - If DTLS tunnel from Device to vManage goes down, auto-rollback of config after 5min
    - E.g. we entered the wrong default gateway IP address in a Device Specific variable, which broke connectivity from Device to vManage

Implementing Cisco SD-WAN

Cisco SD-WAN CLI Templates Example

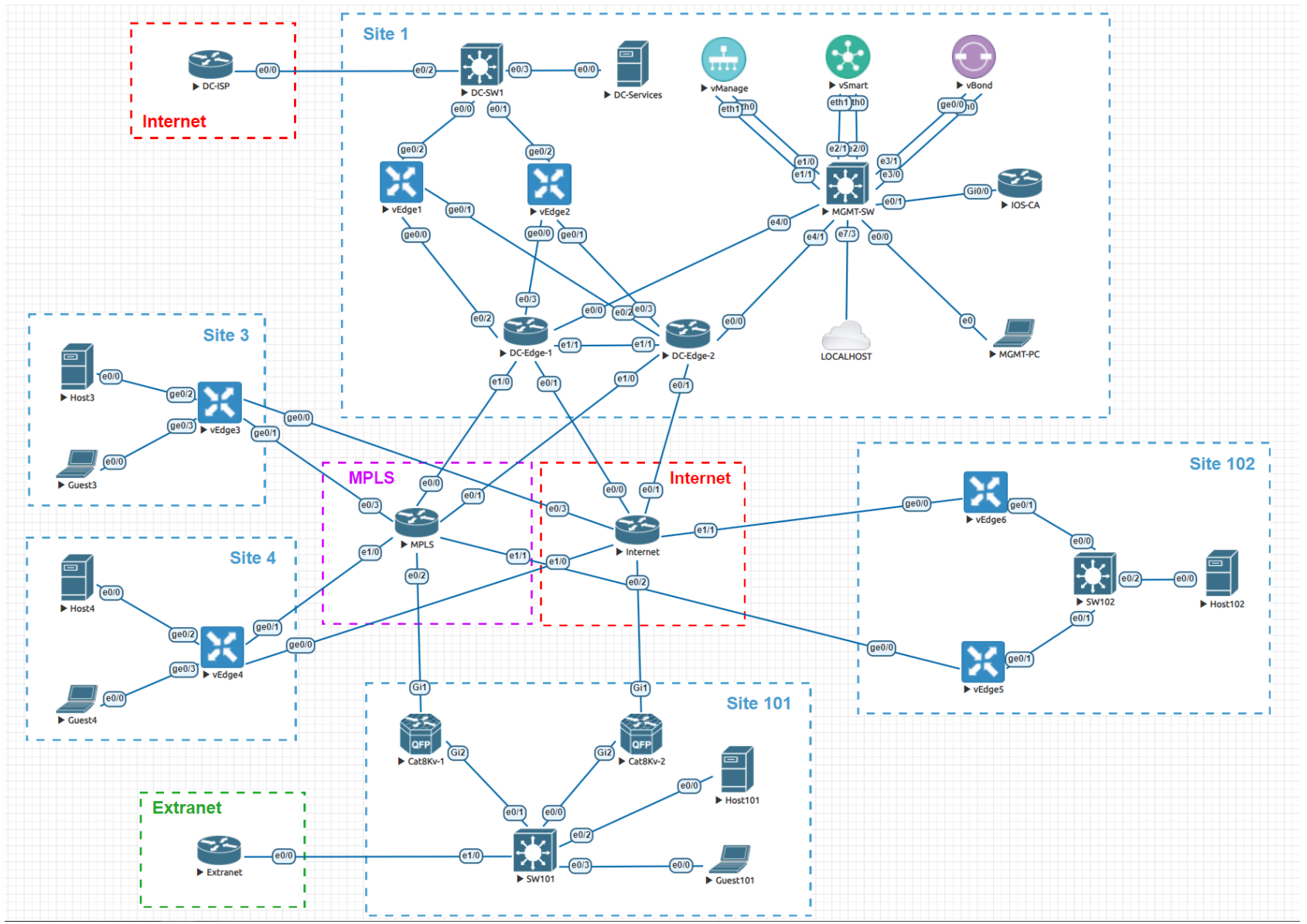
Cisco SD-WAN CLI Templates Example

- Create a CLI Template for cEdge (IOS XE) routers as follows:
  - Name the Device Template **`cEdge-Single-WAN-CLI`**
  - Use **`Cat8Kv-1`** as an example config
  - Create Device Specific Variables as needed...
    - Hostname
    - System-IP
    - Site-ID
    - IP Addresses
    - Default Gateway
    - Tunnel Color
  - Apply this Template to **`Cat8Kv-1`** & **`Cat8Kv-2`**

Example Device Specific Values for CLI Templates

Hostname	Cat8Kv-1	Cat8Kv-2
System-IP	172.17.101.1	172.17.101.2
Site-ID	101	101
WAN1-IPv4-Addr	10.101.1.1 255.255.255.252	150.101.2.1 255.255.255.252
WAN1-Default-GW	10.101.1.2	150.101.2.1
WAN1-Color	mpls	biz-internet
LAN1-IPv4-Addr	192.168.101.101 255.255.255.0	192.168.101.102 255.255.255.0

Example Cisco SD-WAN Topology



Implementing Cisco SD-WAN

Cisco SD-WAN Device & Feature Templates Example

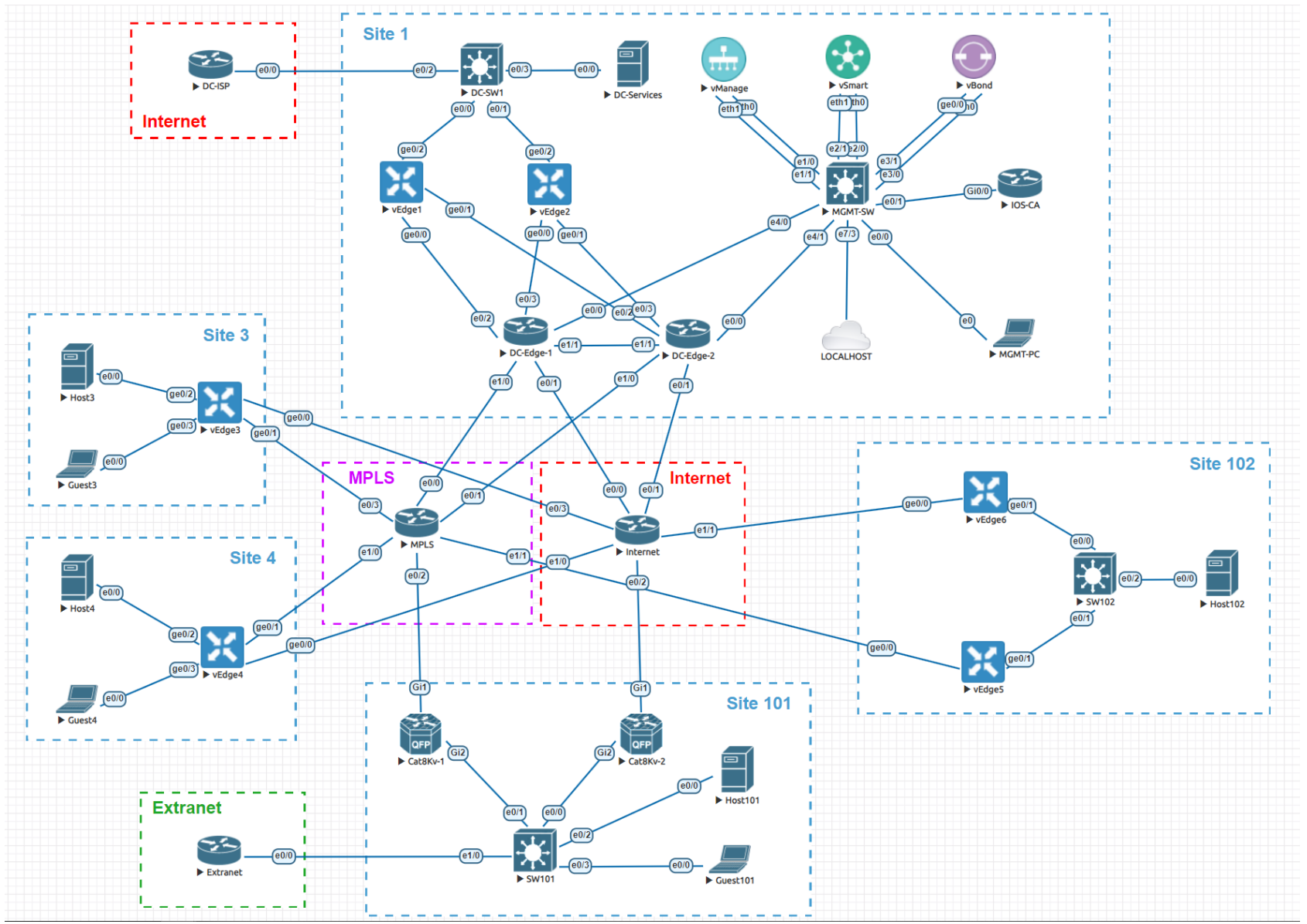
Cisco SD-WAN Device & Feature Templates Example

- Create a Device Template for cEdge (IOS XE) routers as follows:
  - Name the Device Template **`cEdge-Single-WAN`**
  - Create Feature Templates as needed
    - Transport VPN template
    - Service VPN template
    - Interface templates
  - Create Device Specific Variables as needed in these Feature Templates
    - Hostname
    - System-IP
    - Site-ID
    - IP Addresses
    - Default Gateway
    - Tunnel Color
  - Apply this Template to **`Cat8Kv-1`** & **`Cat8Kv-2`**

Example Device Specific Values for Device Templates

Hostname	Cat8Kv-1	Cat8Kv-2
System-IP	172.17.101.1	172.17.101.2
Site-ID	101	101
WAN1-IPv4-Addr	10.101.1.1/30	150.101.2.1/30
WAN1-Default-GW	10.101.1.2	150.101.2.1
WAN1-Color	mpls	biz-internet
LAN1-IPv4-Addr	192.168.101.101/24	192.168.101.102/24

Example Cisco SD-WAN Topology



Implementing Cisco SD-WAN

Advanced Cisco SD-WAN Device & Feature Templates Example

Advanced Cisco SD-WAN Device Templates – cEdge Example

- SD-WAN Edge Routers **`Cat8Kv-1`** & **`Cat8Kv-2`** are attached to the pre-configured template **`cEdge-Single-WAN-Template`**

- Modify the Template to enable EIGRP AS 65535 on the links towards the **`Extranet`** router, and redistribute as necessary.

- Once the config changes are applied to **`Cat8Kv-1`** & **`Cat8Kv-2`**, other VPN 1 sites should have IP reachability to 7.7.7.7/32 via the **`Extranet`** router

Advanced Cisco SD-WAN Device Templates – vEdge Example

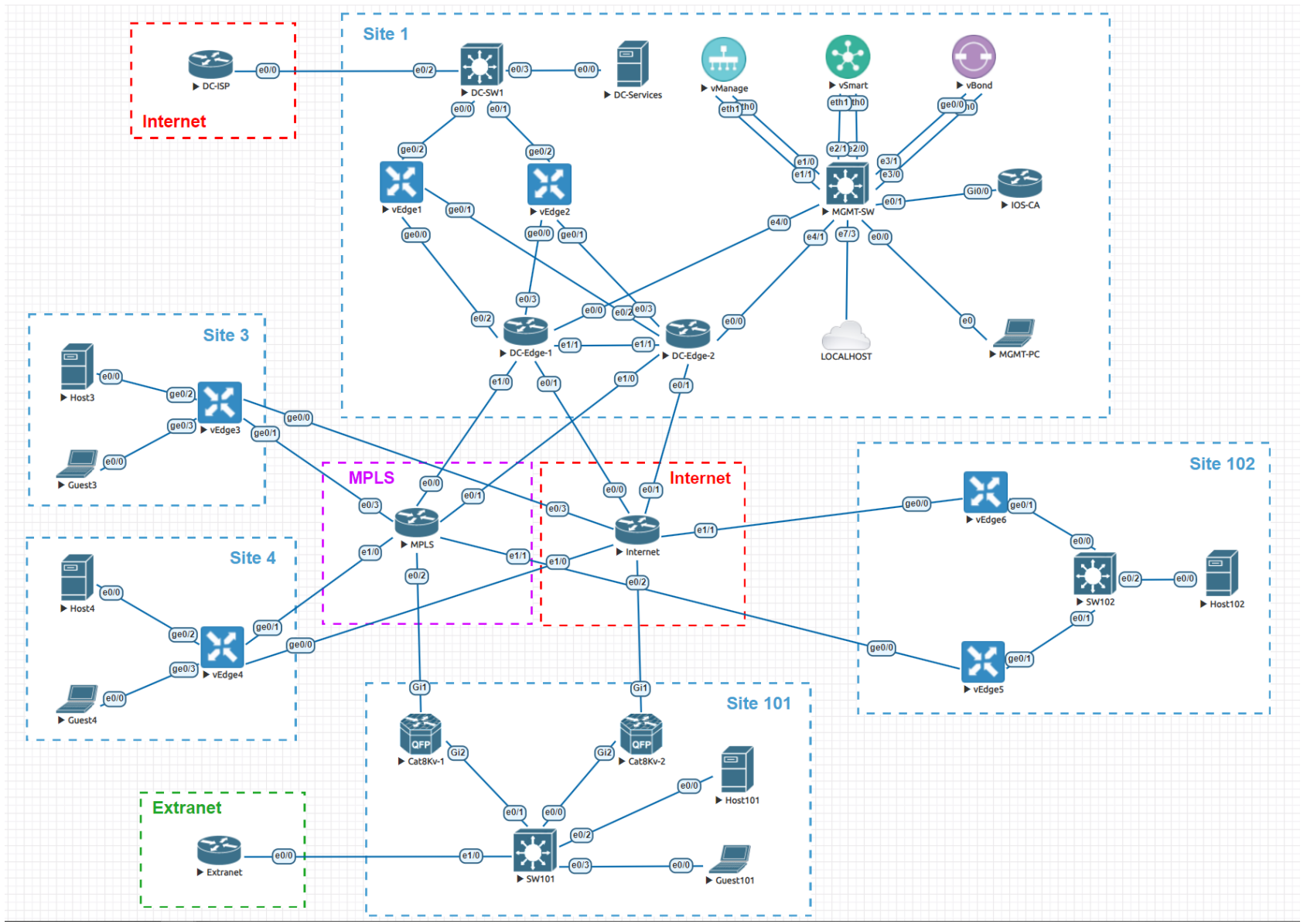
- SD-WAN Edge Routers **`vEdge-1`** & **`vEdge-2`** are attached to the pre-configured template **`vEdge-DC-Template`**

- Modify the Template to enable routing on these devices as follows:

  - Run **OSPF Area 1** on the sub-interface towards **`DC-Services`**, and redistribute where necessary. Once complete the other VPN 1 sites should have IP reachability to **4.4.4.4/32**

  - Run **BGP AS 12345** and peer to the **`DC-ISP`** address **10.9.9.9** in **AS 999**, and redistribute where necessary. Once complete the other VPN 1 sites should have IP reachability to **9.9.9.9/32**

Example Cisco SD-WAN Topology



Implementing Cisco SD-WAN

Cisco SD-WAN CLI Add-On Templates

CLI Add-On Templates Overview

- When an SD-WAN Edge Router is in vManage Mode (i.e. a template is attached), configuration from the local CLI is blocked

- CLI Add-On Templates gives you a way to further customize Device Templates by using CLI syntax
  - Final configuration pushed from vManage is a merge of the Device/Feature Templates & the CLI Add-On Template
  - In cases where the configuration overlaps, the CLI Add-On Template takes priority

- CLI Add-On Templates only apply to cEdge (IOS XE) routers, not vEdge

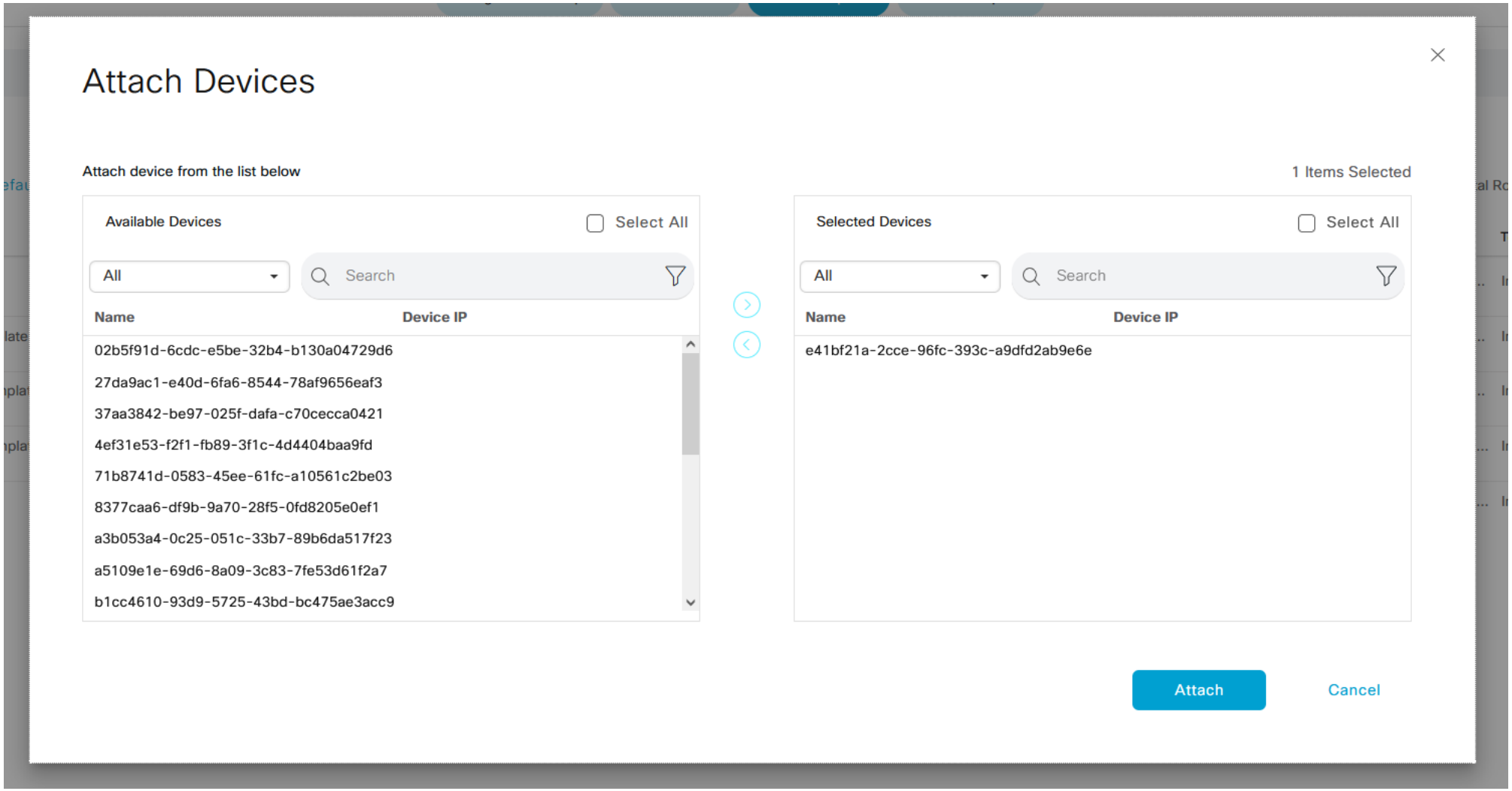
Implementing Cisco SD-WAN

Pre-Provisioning Cisco SD-WAN Templates

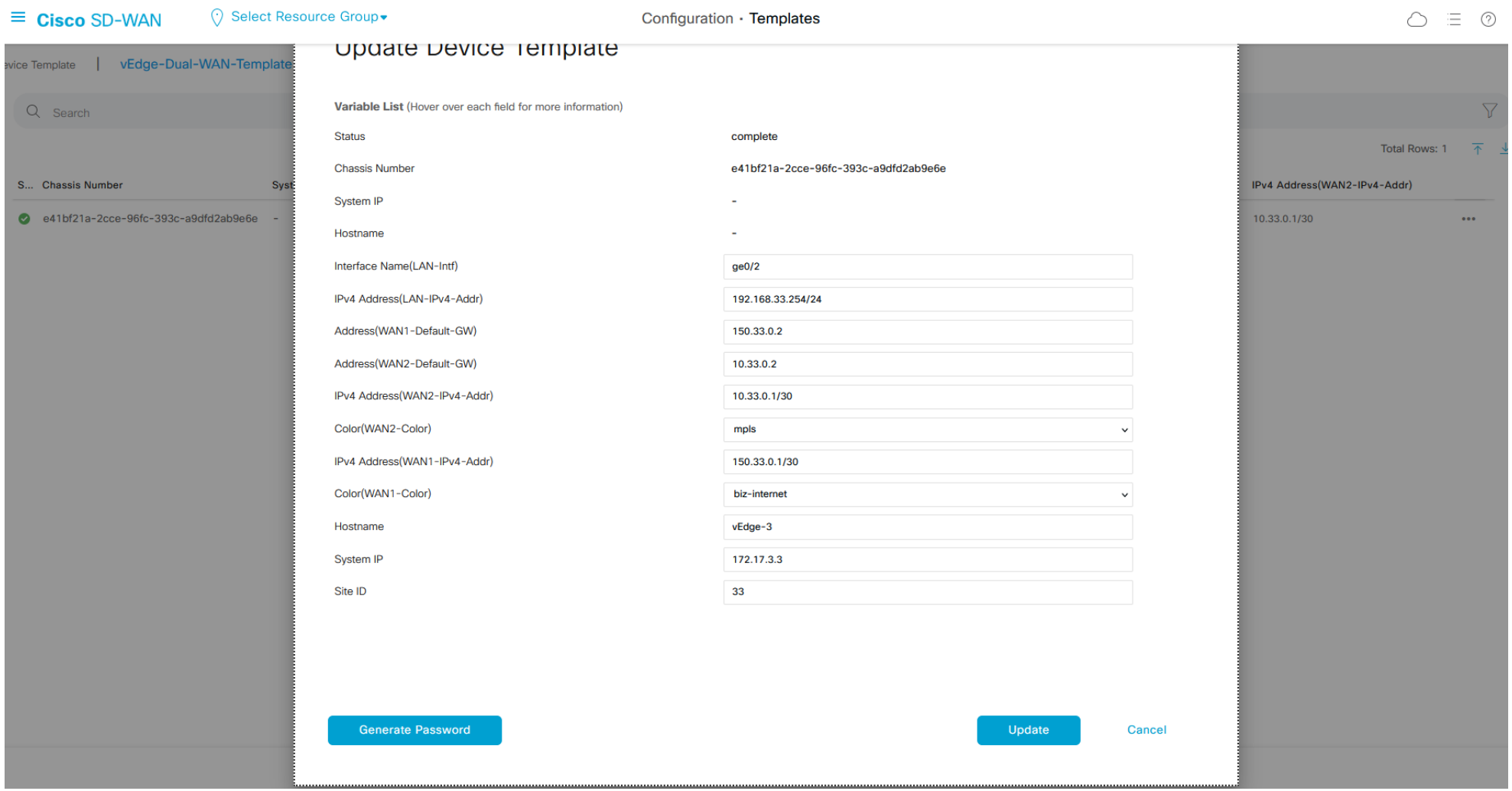
Pre-Provisioning SD-WAN Edge Routers with Templates

- Device & CLI Templates can be attached before onboarding occurs
  - I.e. When the device is onboarded, the config from Template is immediately applied
- Template is pre-provisioned by attaching to *Chassis Number* before onboarding
  - Can be used for both Zero Touch Provisioning (ZTP) and manual provisioning
  - Template's Device Specific variables are defined before WAN Edge Router onboards
  - Once Device is reachable from vManage (i.e. over DTLS), Template is applied

Attaching Device Template to Chassis Number for Pre-Provisioning

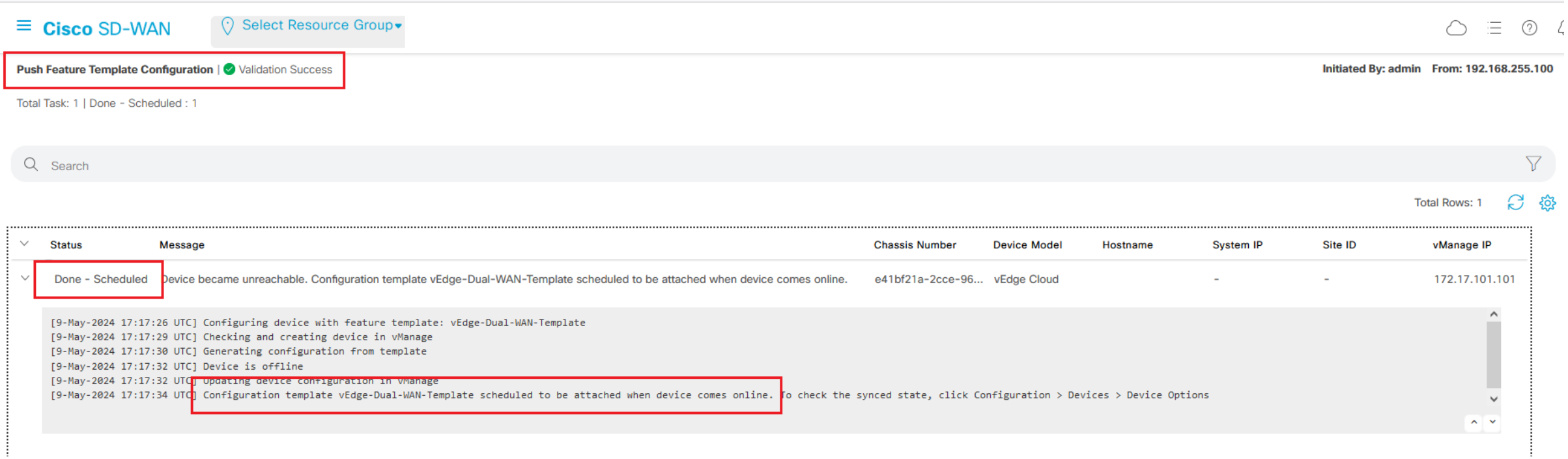


Defining Device Specific Variables for Pre-Provisioned Device

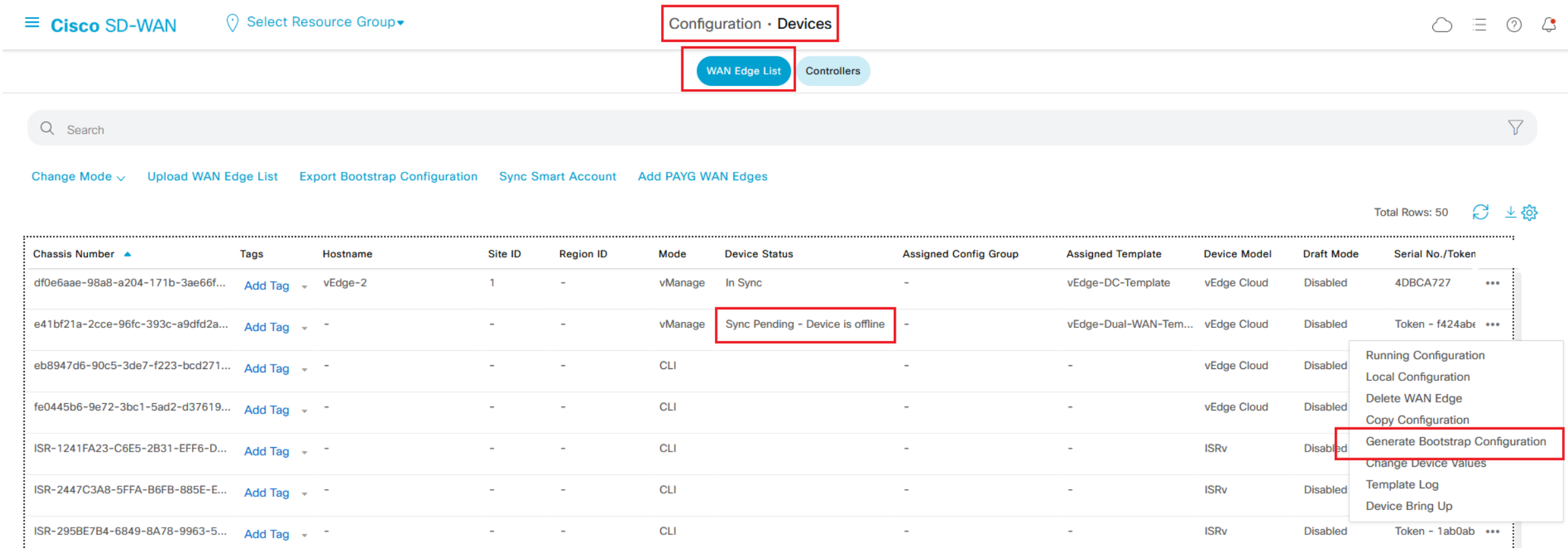


Scheduling Template Attachment to Pre-Provisioned Devices

- After attaching the Template, vManage schedules the configuration to be pushed once the Device becomes reachable (i.e. onboards)

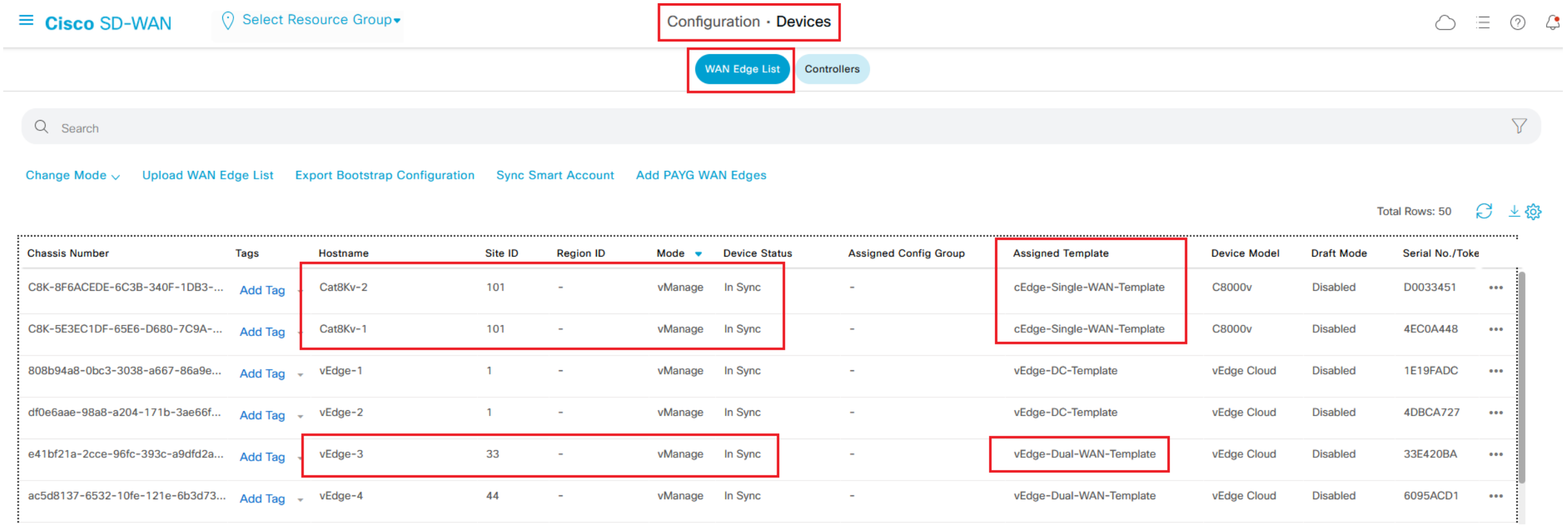


Template Pre-Provisioned Device Status Before Onboarding



Template Pre-Provisioned Device Status After Onboarding

- After onboarding, Template is applied and Device should be "In Sync"



Chassis Number	Tags	Hostname	Site ID	Region ID	Mode	Device Status	Assigned Config Group	Assigned Template	Device Model	Draft Mode	Serial No./Toke
C8K-8F6ACEDE-6C3B-340F-1DB3-...	Add Tag	Cat8Kv-2	101	-	vManage	In Sync	-	cEdge-Single-WAN-Template	C8000v	Disabled	D0033451
C8K-5E3EC1DF-65E6-D680-7C9A-...	Add Tag	Cat8Kv-1	101	-	vManage	In Sync	-	cEdge-Single-WAN-Template	C8000v	Disabled	4EC0A448
808b94a8-0bc3-3038-a667-86a9e...	Add Tag	vEdge-1	1	-	vManage	In Sync	-	vEdge-DC-Template	vEdge Cloud	Disabled	1E19FADC
df0e6aae-98a8-a204-171b-3ae66f...	Add Tag	vEdge-2	1	-	vManage	In Sync	-	vEdge-DC-Template	vEdge Cloud	Disabled	4DBCA727
e41bf21a-2cce-96fc-393c-a9dfd2a...	Add Tag	vEdge-3	33	-	vManage	In Sync	-	vEdge-Dual-WAN-Template	vEdge Cloud	Disabled	33E420BA
ac5d8137-6532-10fe-121e-6b3d73...	Add Tag	vEdge-4	44	-	vManage	In Sync	-	vEdge-Dual-WAN-Template	vEdge Cloud	Disabled	6095ACD1

Verifying Template is Assigned Using CLI

- CLI "commit" is not allowed when a device is in vManage mode
  - I.e. when a Template is attached, changes can't be made from the CLI

```
vEdge-3# config t
Entering configuration mode terminal
vEdge-3(config)# no vpn 1
vEdge-3(config)# commit and-quit
Aborted: 'system': This device is being managed by the vManage. Configuration
through the CLI is not allowed.
vEdge-3(config)#
```

```
Cat8Kv-1#config-transaction
admin connected from 127.0.0.1 using console on Cat8Kv-1
Cat8Kv-1(config)# hostname abc
Cat8Kv-1(config)# commit
Aborted: 'system is-vmanaged': This device is being managed by vManage,
configuration through CLI is not allowed.
Cat8Kv-1(config)#
```

Implementing Cisco SD-WAN

Understanding Cisco SD-WAN Policies

What are Cisco SD-WAN Policies?

- SD-WAN Policies are how we can encode our Intent into the network
  - E.g. I want VoIP to get the best service
- SD-WAN Policies can be used to affect both the control-plane and the data-plane
  - Traffic from A to B should forward through C
  - YouTube traffic should forward from A to D
  - VoIP should automatically use the lowest delay path

What Can SD-WAN Policies Do?

- SD-WAN Policies are used to apply features such as…
  - **Traffic Engineering & Application Aware Routing (AAR)**
    - Direct traffic based on the source, destination, application, QoS requirements, etc.
  - **VPN Membership**
    - Allows for Shared Services (i.e. route leaking) & Service Chaining
  - **Network Address Translation (NAT)**
    - Used for traffic not sent over the SD-WAN, e.g. to the Internet in the *Underlay*
  - **Access Control Lists (ACLs) & Security Policy**
    - E.g. filter out specific traffic with ACLs, IPS, SSL/TLS Proxy, URL Filtering, etc.
  - **Quality of Service (QoS)**
    - E.g. reserve bandwidth for specific applications
  - **Cflowd**
    - Similar to NetFlow – used to collect traffic statistics

Types of SD-WAN Policies

- SD-WAN Policies can be grouped into main two categories
  - **Centralized Policies**
    - Policies that apply across the entire network
  - **Localized Policies**
    - Policies that apply to an individual Edge Router

- Centralized and Localized Policies can be further grouped in 2 sub-categories
  - **Control Policies**
    - Also called *Topology Policies*
    - Used to affect the routing (OMP) control-plane
  - **Data Policies**
    - Also called *Traffic Policies*
    - Analogous to Policy Based Routing (PBR)

SD-WAN Control (Topology) Policies

- Control / Topology Policies are used to affect the routing (OMP) control-plane
- Example Control Policy Use Cases:
  - Prevent spoke-to-spoke communication by filtering routes
    - E.g. only advertise hub routes to spokes
  - Force spoke-to-spoke traffic to flow through the central Data Center (DC) first
    - E.g. force traffic to be inspected by centralized Firewalls at the DC
  - Prefer MPLS over public Internet
    - E.g. “color” preference
  - Isolate Guest Users from Corporate WAN
    - E.g. allow Internet access for guests, but not site-to-site SD-WAN traffic
  - Allow access to Shared Services
    - E.g. route leaking between VPNs
  - Filtering Redistribution
    - E.g. filter/modify routing attributes with a locally significant policy

SD-WAN Data (Traffic) Policies

- Data / Traffic Policies are used to override the control-plane routing policy
- Example Data Policy Use Cases:
  - Direct Internet Access (DIA)
    - E.g. don't use the SD-WAN if the destination is not internal
  - Direct Cloud Access
    - E.g. don't use the SD-WAN to reach Office 365, Google Apps, Salesforce, etc.
  - Cloud Based Firewall
    - E.g. redirect internal traffic to Cisco Umbrella for scrubbing
  - Application Aware Routing (AAR)
    - E.g. VoIP should prefer to use color MPLS unless delay goes over 100ms
  - Security Filtering
    - E.g. ACLs, ZBFW, IPS, URL Filtering, Malware Protection, DNS Security, etc.
  - Quality of Service (QoS)
    - E.g. mark an application's traffic as critical

Applying SD-WAN Centralized Policies through vSmart

- vSmart is the central point of policy application for SD-WAN Centralized Policies
  - vSmart receives OMP routes from Edges, applies policies, and advertises routes back
    - Similar logic to a BGP Route Reflector
  - *Centralized Policy application direction is from the perspective of vSmart*
    - Inbound policy affects routes received on vSmart from WAN Edge Routers
    - Outbound policy affects routes advertised from vSmart to WAN Edge Routers
- *Before applying any policies, the vSmart controller(s) must be in vManage mode*
  - vManage mode means that a Template is applied

Attaching Templates to vSmart

- To apply a Policy, the vSmart controller(s) must be in vManage mode
- vSmart in vManage mode implies a Template must be attached to vSmart
  - Same as WAN Edge Routers, could be a CLI Template or Device/Feature Template
- Using a CLI Template is a quick way to set the vSmart to vManage mode
  - **Configuration > Devices > Templates > Device Templates > Create Template > CLI Template > Device Model: vSmart > Load Running config dropdown > vSmart**
  - No variables needed unless you're applying the template to multiple vSmarts
    - E.g. each vSmart could have a separate unique CLI template if you wanted

Applying SD-WAN Localized Policies through Device Templates

- Some device specific policies need to be locally significant to the Edge Routers
  - E.g. Apply an ACL inbound on the Service VPN (LAN) interface
  - E.g. Redistribute OMP into BGP and set MED to 100

- **Localized Policies** are still centrally created & managed through vManage, but are **not applied through the vSmart controller**
  - Localized Policies are applied through Device Templates & Feature Templates
  - Device Templates are used to attach the Localized Policy to the Edge Router
    - E.g. here is the definition of the ACL named "**inside-in**"
  - Feature Templates define where the Localized Policy is actually applied
    - E.g. ACL "**inside-in**" applies to Feature Template "**cEdge-LAN**" inbound

Configuring Cisco SD-WAN Policies

- Configuring an SD-WAN Policy is a 3-step process
  - Define the Lists - what am I matching?
  - Define the Policy - what action am I taking?
  - Apply the Policy - who does the policy apply to, and in which direction?
- SD-WAN Policy logic is like a *Route-Map* in Cisco IOS
  - Policy is processed top-down until a match occurs
  - Once a match occurs, the defined actions are taken, and it exits the process
  - If no match occurs, Policy defaults to implicit deny at the end
- Order-of-operations in the Policy is significant
  - More specific matches should be at the top
  - E.g. if you match “ALL-IP” before “HTTPS”, “HTTPS” will have zero hits

Activating SD-WAN Centralized Policies

- After a Centralized Policy is created, it must be applied (Activated)
  - **Configuration > Policies > click 3 dots on right of policy > Activate**
  - Activating is pushing the Policy config to vSmart
- ***vSmart can only have one active SD-WAN Policy at a time***
- To modify the active Policy, first make a copy
  - **Configuration > Policies > click 3 dots on right of policy > Copy**
    - E.g. My-Policy-v002
  - Now you can modify the new copy
    - **Configuration > Policies > click 3 dots on right of *copy* > Edit**
- Activating the new Policy will automatically de-activate the old Policy
  - If the new Policy has unexpected results, you can re-activate the old Policy
  - E.g. My-Policy-v002 has a mistake, re-activate My-Policy-v001

Verifying SD-WAN Policies

- vSmart
  - **`show run [policy]`**
- vEdge & cEdge
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] bfd sessions`**
  - **`show ip route`**
  - **`show omp route`**
  - **`show [sdwan] policy service-path`**

Verifying SD-WAN Traffic Flows via CLI

- CLI command **`show [sdwan] policy service-path`** is used to show the forwarding result of any configured policies based on your input
  - E.g. what happens when a TCP packet from source IP 1.2.3.4 & port 34567 is received on interface Gig2 in VPN 1 with destination IP 5.6.7.8 & port 443?
  - Similar in logic to the IOS command **`show ip cef exact-route`**
  - Can also match App name & DSCP (QoS) markings

`show [sdwan] policy service-path` *`vpn-id`* **`vpn-id interface`** *`interface-name`* **`source-ip`** *`ip-address`* **`dest-ip`** *`ip-address`* **`protocol`** *`number`* **`source-port`** *`port-number`* **`dest-port`** *`port-number`* **`[all | app`** *`application-name`* **`| dscp`** *`value`***`]`**

Verifying SD-WAN Traffic Flows via CLI Example

```
vEdge-1# show policy service-path vpn 1 interface ge0/2.10 source-ip 4.4.4.4 dest-ip 192.168.33.1
protocol 6 source-port 34567 dest-port 443 all

Number of possible next hops: 2

Next Hop: IPsec
  Source: 150.11.1.0 12346 Destination: 150.33.0.1 12346 Color: custom1

Next Hop: IPsec
  Source: 150.22.1.0 12346 Destination: 150.33.0.1 12346 Color: custom2


Cat8Kv-1# show sdwan policy service-path vpn 1 interface GigabitEthernet2.101 source-ip 192.168.101.1
dest-ip 192.168.33.1 protocol 1 all

Number of possible next hops: 2

Next Hop: IPsec
  Source: 10.101.1.1 12346 Destination: 150.33.0.1 12346 Local Color: mpls Remote Color: biz-internet
Remote System IP: 172.17.3.3

Next Hop: IPsec
  Source: 10.101.1.1 12346 Destination: 10.33.0.1 12346 Local Color: mpls Remote Color: mpls
Remote System IP: 172.17.3.3
```

Implementing Cisco SD-WAN

SD-WAN Policy Example – Filtering Spoke to Spoke Traffic

SD-WAN CLI Verification Review

- Before we create a Policy, let's verify the default behavior of the SD-WAN
  - Did the WAN Edge Routers form IPsec tunnels between sites?
    - **vEdge-1# show bfd sessions**
    - **cEdge-1# show sdwan bfd sessions**
  - Did vSmart receive routes from the WAN Edge Routers?
    - **vSmart-1# show omp route**
  - Did the WAN Edge Routers receive routes from vSmart?
    - **vEdge-1# show ip route vpn 1**
    - **vEdge-1# show omp route vpn 1**
    - **cEdge-1# show ip route vrf 1**
    - **cEdge-1# show sdwan omp route vpn 1**
  - Do we have IP reachability to the destinations over the SD-WAN?
    - **vEdge-1# ping vpn 1 1.2.3.4**
    - **cEdge-1# ping vrf 1 1.2.3.4**

SD-WAN CLI Verification Review (cont.)

- In this example we will verify from WAN Edge Routers **vEdge-3** & **vEdge-4**
  - Routers have System-IPs 172.17.3.3 & 172.17.4.4 respectively
  - Routers have 2 WAN links each, one color "**mpls**", the other color "**biz-internet**"
  - By default, WAN Edge Routers form a full-mesh of IPsec tunnels out all colors
  - The result in this case is 4 IPsec tunnels:
    - 1) mpls to mpls
    - 2) mpls to biz-internet
    - 3) biz-internet to mpls
    - 4) biz-internet to biz-internet

```
vEdge-3# show bfd sessions | in "172.17.4.4|S"
                                  SOURCE TLOC   REMOTE TLOC                    DST PUBLIC  DST PUBLIC  <snip>
SYSTEM IP      SITE ID  STATE     COLOR         COLOR          SOURCE IP       IP          PORT
172.17.4.4     44       up        mpls          mpls           10.33.0.1       10.44.0.1   12346
172.17.4.4     44       up        mpls          biz-internet   10.33.0.1       150.44.0.1  12346
172.17.4.4     44       up        biz-internet  mpls           150.33.0.1      10.44.0.1   12346
172.17.4.4     44       up        biz-internet  biz-internet   150.33.0.1      150.44.0.1  12346
```

SD-WAN CLI Verification Review (cont.)

- Next, vSmart learns routes from **vEdge-3** & **vEdge-4**
  - Each Router advertises a LAN interface in VPN 1 via both "**mpls**" & "**biz-internet**"
  - *By default, vSmart does not filter any routing advertisements*, just reflects them back
  - Result is each Router has 2 routes to each prefix, one via **mpls**, one via **biz-internet**

```
vSmart-1# show omp route vpn 1 | tab | nomore | include "172.17.[3-4].[3-4]|P"
                                        PATH                        ATTRIBUTE
VPN     PREFIX           FROM PEER      ID      LABEL    STATUS     TYPE          TLOC IP      COLOR          ENCAP  <snip>
1       192.168.33.0/24  172.17.3.3     66      1005     C,R        installed     172.17.3.3   mpls           ipsec  <snip>
                         172.17.3.3     68      1005     C,R        installed     172.17.3.3   biz-internet   ipsec  <snip>
1       192.168.44.0/24  172.17.4.4     66      1005     C,R        installed     172.17.4.4   mpls           ipsec  <snip>
                         172.17.4.4     68      1005     C,R        installed     172.17.4.4   biz-internet   ipsec  <snip>

vEdge-3# show omp route vpn 1 | tab | nomore | include "172.17.4.4|P"
                                        PATH                        ATTRIBUTE
VPN     PREFIX           FROM PEER      ID      LABEL    STATUS     TYPE          TLOC IP      COLOR
1       192.168.44.0/24  172.17.101.102 13      1005     C,I,R      installed     172.17.4.4   mpls           <snip>
                         172.17.101.102 14      1005     C,I,R      installed     172.17.4.4   biz-internet   <snip>
                                                                                                             <snip>
vEdge-4# show omp route vpn 1 | tab | nomore | include "172.17.3.3|P"                                       <snip>
                                        PATH                        ATTRIBUTE                            <snip>
VPN     PREFIX           FROM PEER      ID      LABEL    STATUS     TYPE          TLOC IP      COLOR          <snip>
1       192.168.33.0/24  172.17.101.102 13      1005     C,I,R      installed     172.17.3.3   mpls           <snip>
                         172.17.101.102 14      1005     C,I,R      installed     172.17.3.3   biz-internet   <snip>
```

SD-WAN CLI Verification Review (cont.)

- Although we only have **2 routes**, we use all **4 tunnels** for multi-path routing
  - This is because we have 2 IPsec tunnels to each remote color

```
vEdge-3# show policy service-path vpn 1 interface ge0/2 source-ip 192.168.33.1 dest-ip 192.168.44.1 protocol 1 all
Number of possible next hops: 4
Next Hop: IPsec
  Source: 10.33.0.1 12346 Destination: 150.44.0.1 12346 Color: mpls
Next Hop: IPsec
  Source: 150.33.0.1 12346 Destination: 150.44.0.1 12346 Color: biz-internet
Next Hop: IPsec
  Source: 10.33.0.1 12346 Destination: 10.44.0.1 12346 Color: mpls
Next Hop: IPsec
  Source: 150.33.0.1 12346 Destination: 10.44.0.1 12346 Color: biz-internet

vEdge-4# show policy service-path vpn 1 interface ge0/2 source-ip 192.168.44.1 dest-ip 192.168.33.1 protocol 1 all
Number of possible next hops: 4
Next Hop: IPsec
  Source: 10.44.0.1 12346 Destination: 150.33.0.1 12346 Color: mpls
Next Hop: IPsec
  Source: 150.44.0.1 12346 Destination: 150.33.0.1 12346 Color: biz-internet
Next Hop: IPsec
  Source: 10.44.0.1 12346 Destination: 10.33.0.1 12346 Color: mpls
Next Hop: IPsec
  Source: 150.44.0.1 12346 Destination: 10.33.0.1 12346 Color: biz-internet
```

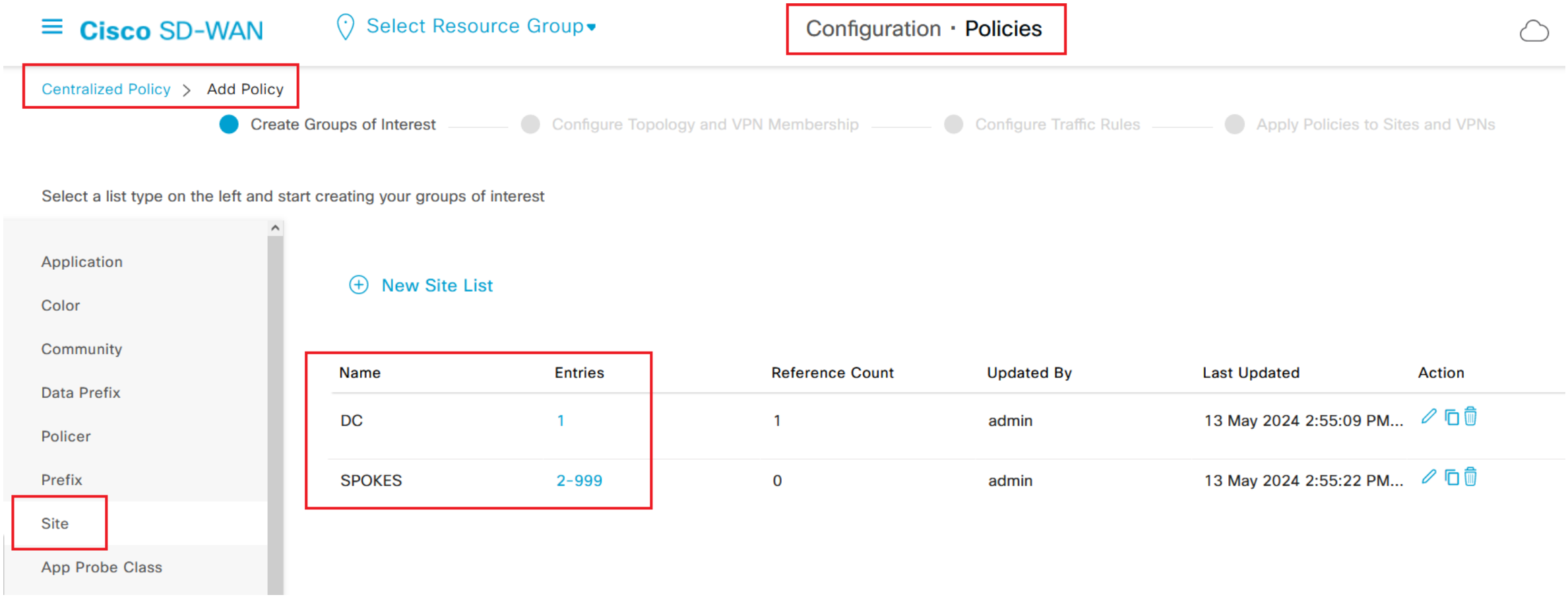
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic

- Next, let's create a Policy on vSmart to accomplish the following:
  - Allow Spokes to reach the Hub (DC) over IPsec tunnels
  - Prevent Spokes from learning each other's routes
  - Prevent Spokes from forming IPsec tunnels with each other

- First step is to classify the routes we want to apply policy to
  - DC routes can be grouped together by Site-ID 1
  - Spokes can also be grouped together by Site-ID "Not 1", e.g. 2-999

Classifying Based on Site-ID

- First, we create two "**Site Lists**", one for DC (Site 1) and one for Spokes (Site 2-999)

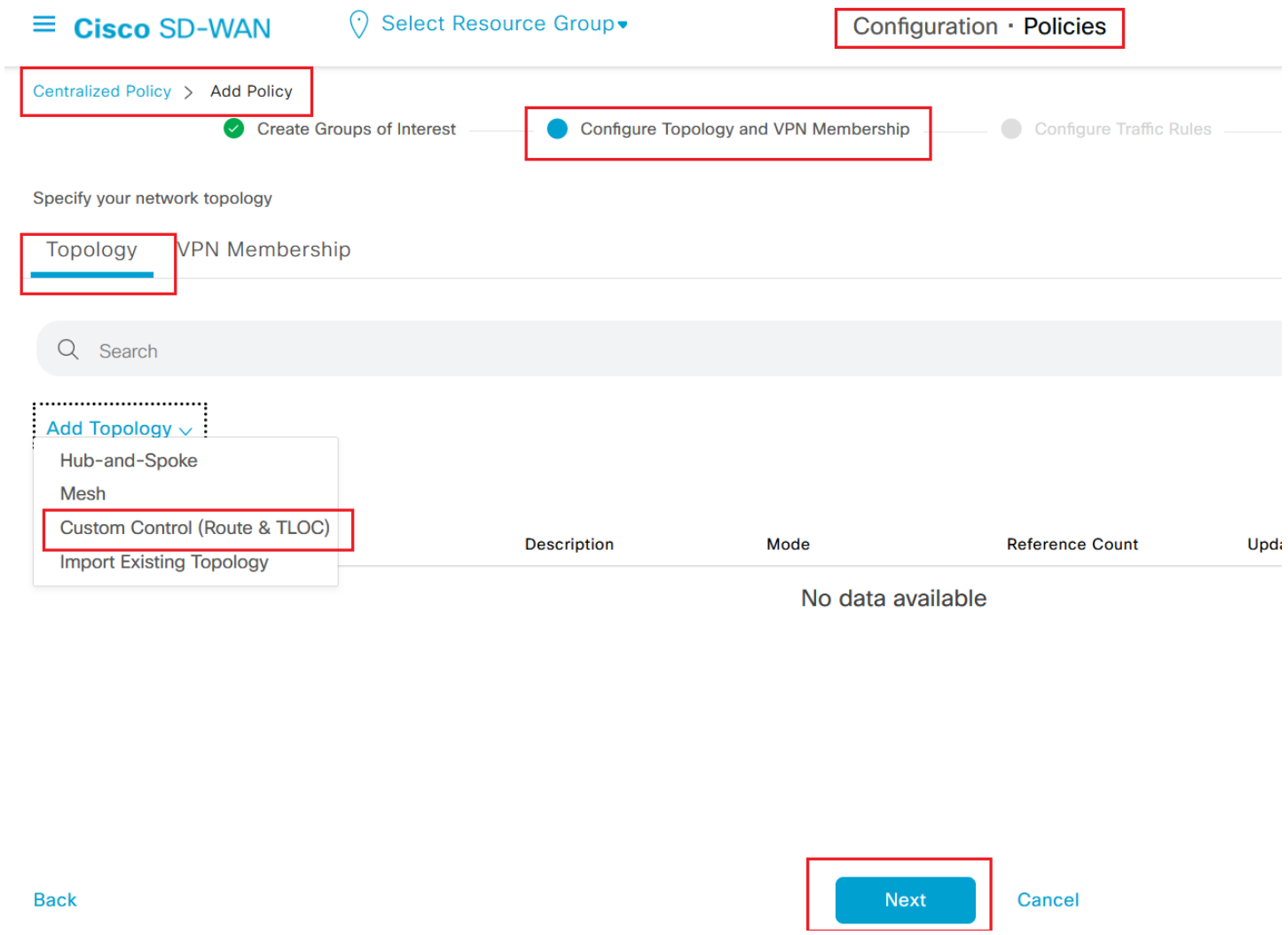


SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Next, we create a "**Custom Control**" Policy with the following logic:
  - IF Routes == ANY && Site-ID == DC, accept
  - IF TLOC == ANY && Site-ID == DC, accept
  - ELSE, reject

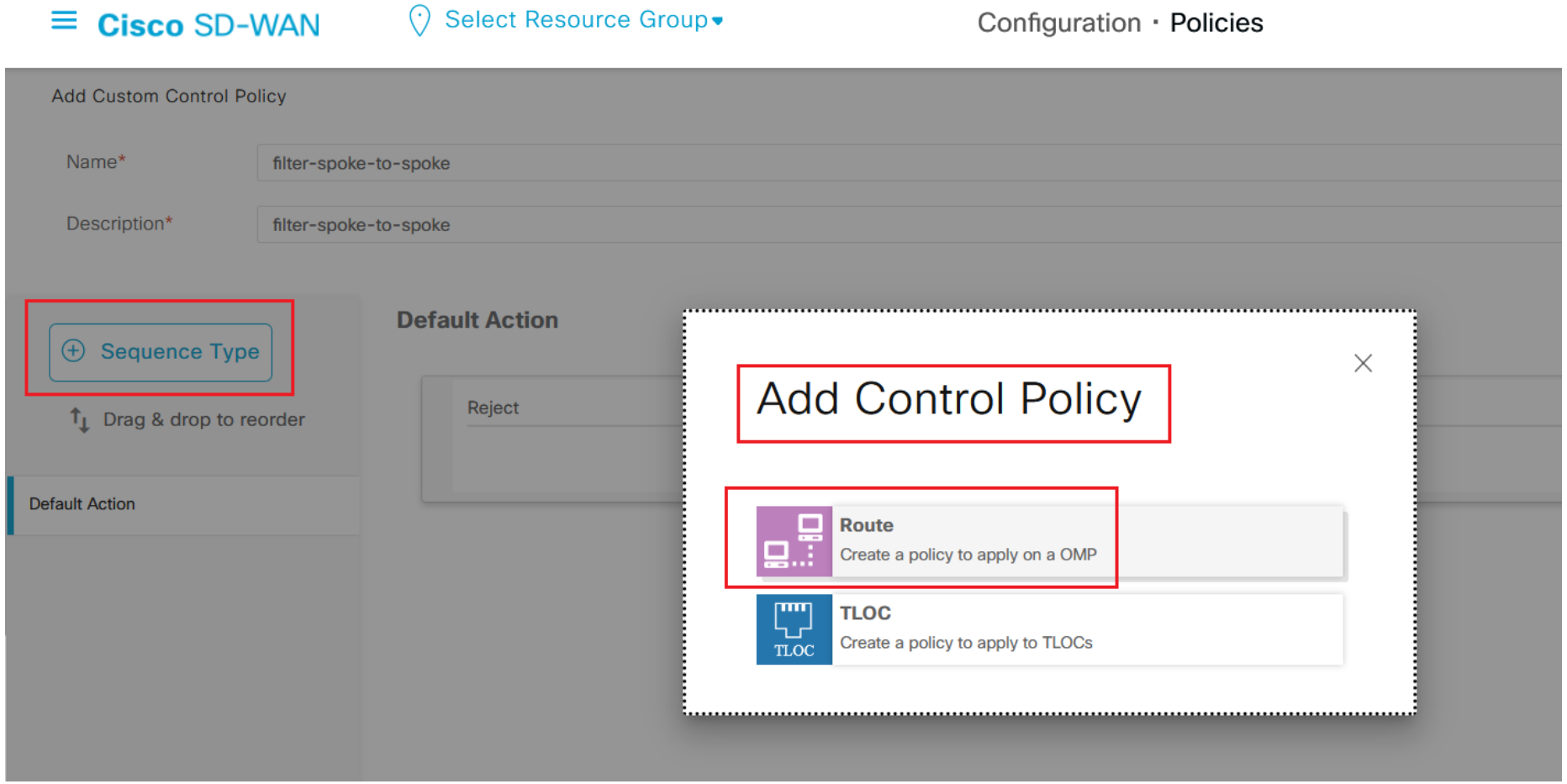
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Under “Topology” add “Custom Control (Route & TLOC)”



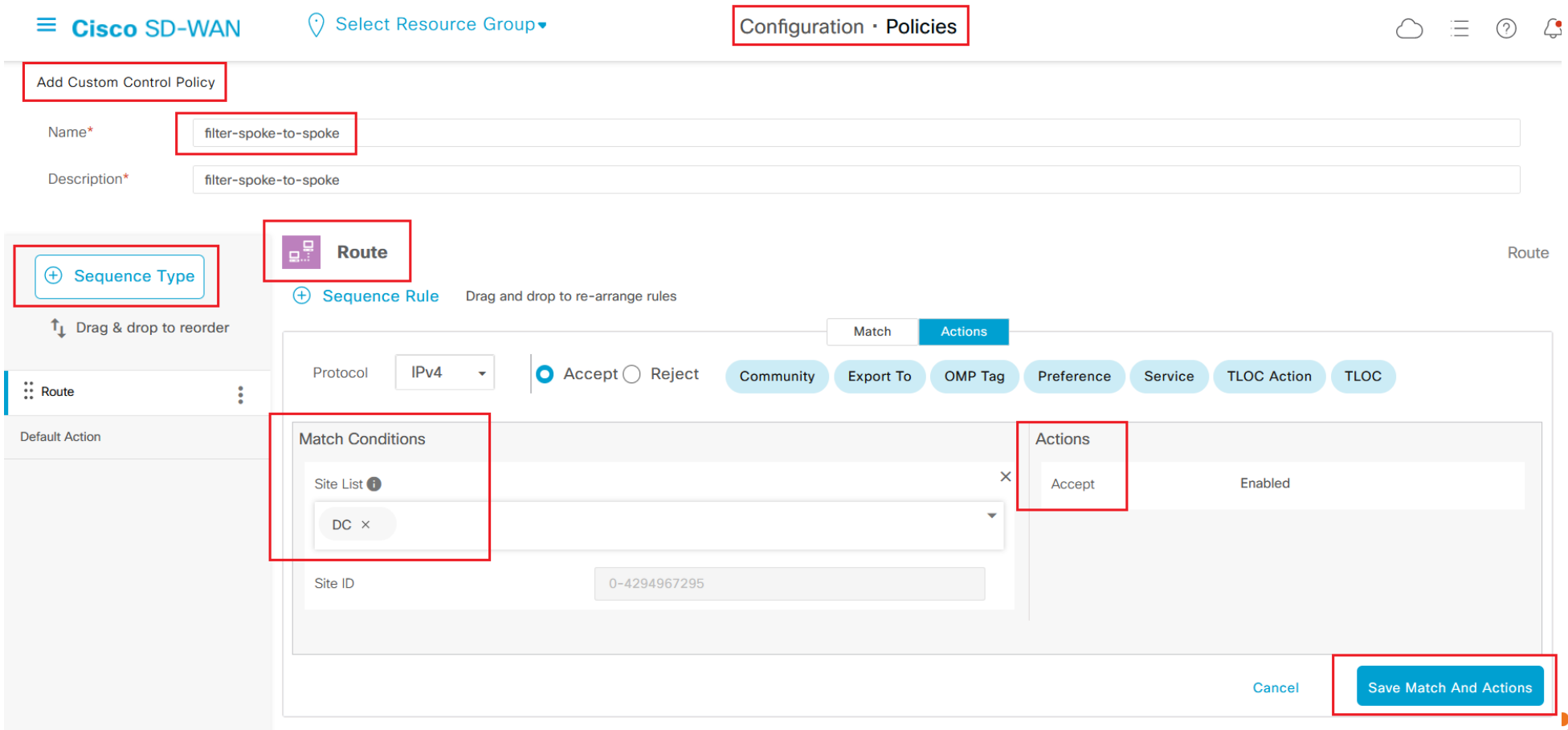
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Add a sequence of type "Route"



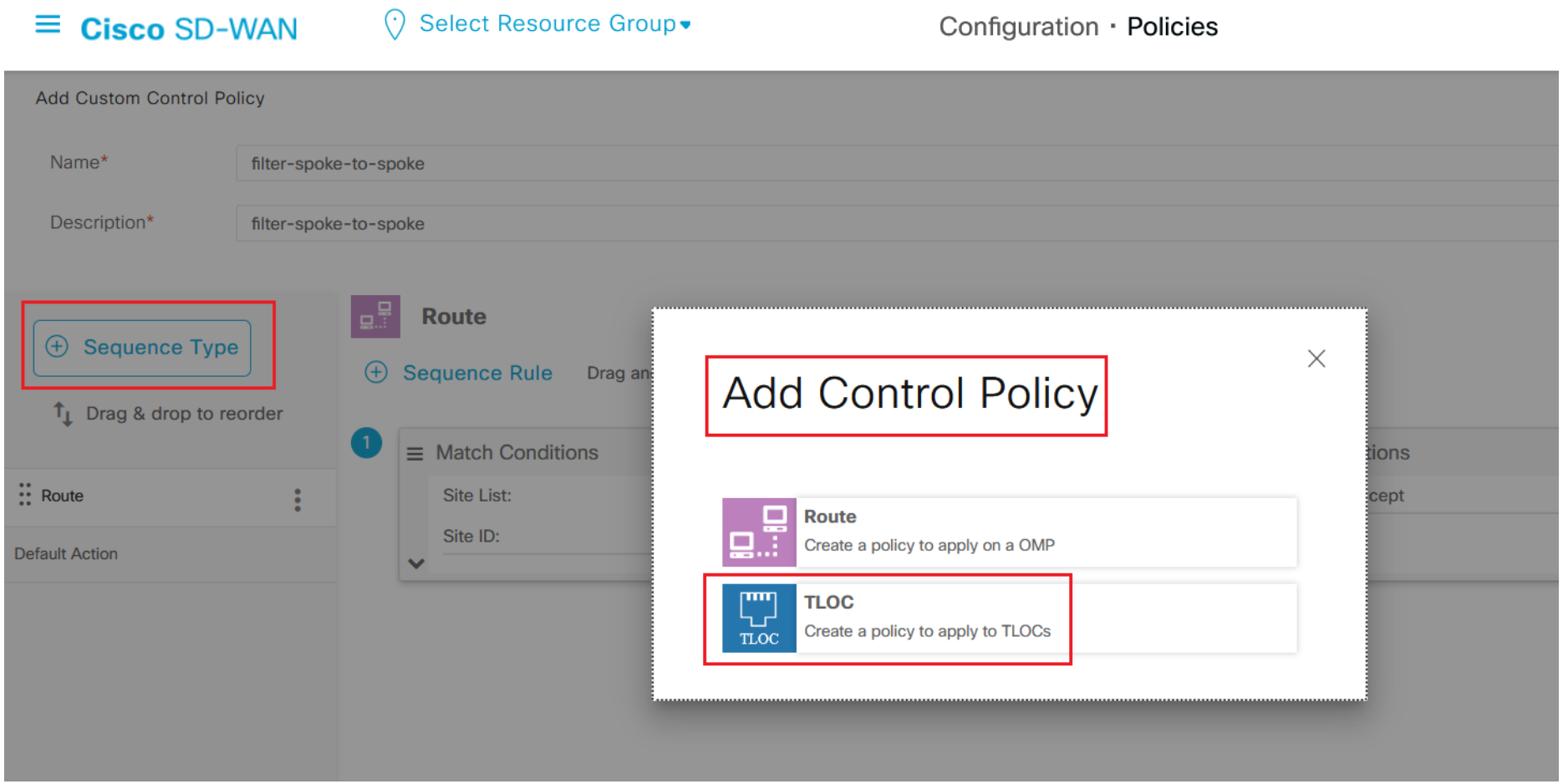
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Match the Site List "DC" and set the Actions to "Accept", then Save



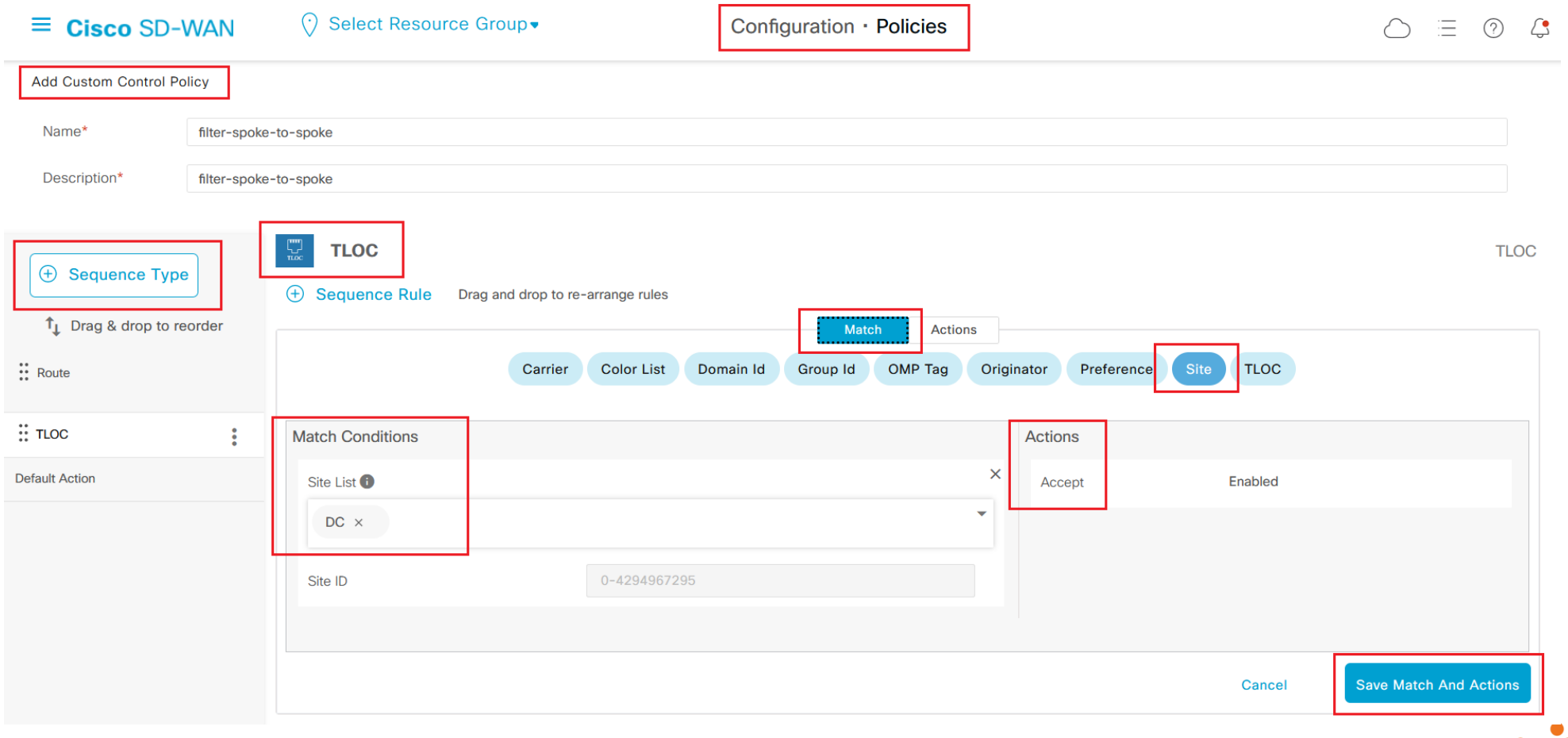
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

+ Add another sequence of type "TLOC"



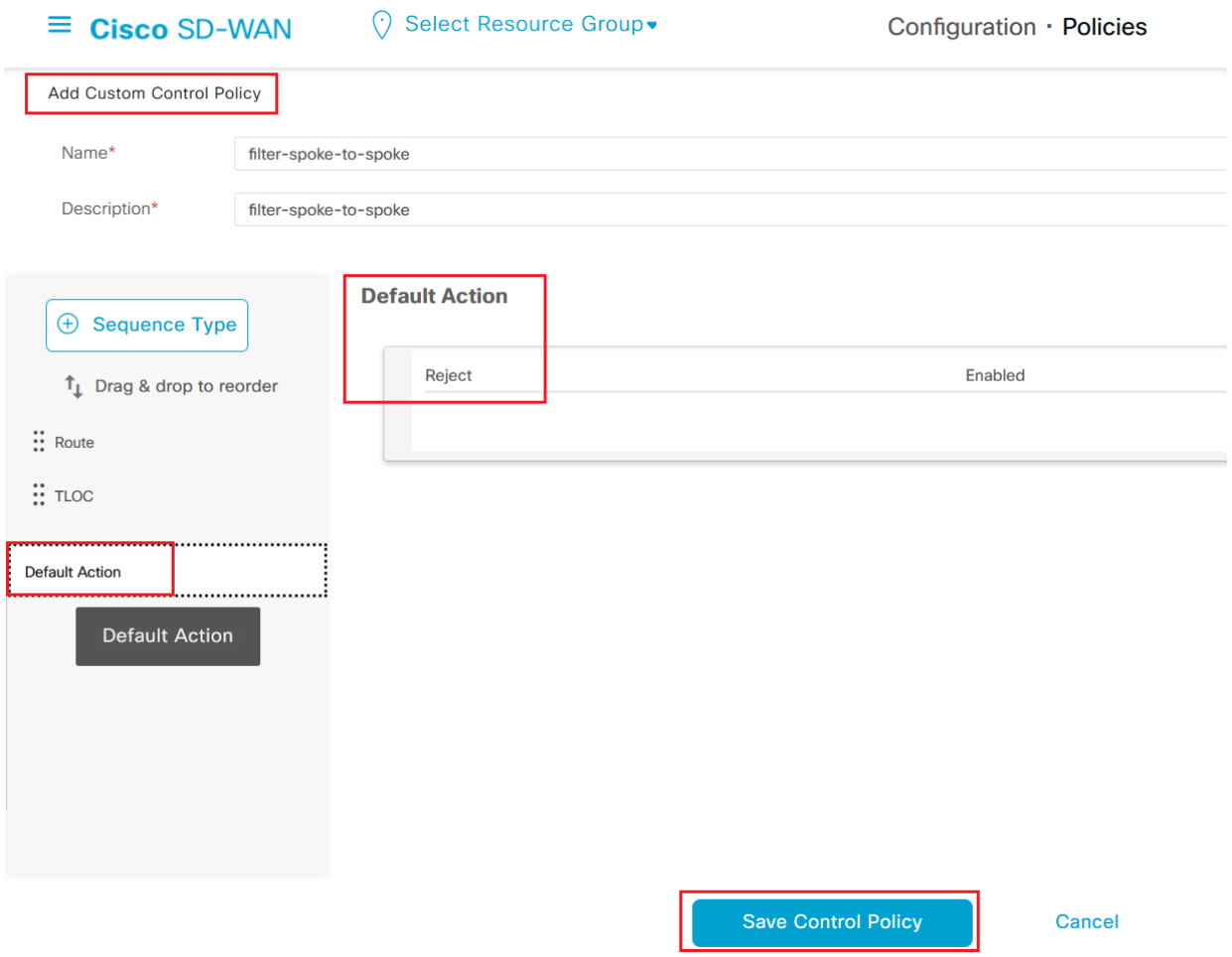
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Match the Site List "DC" and set the Actions to "Accept", then Save



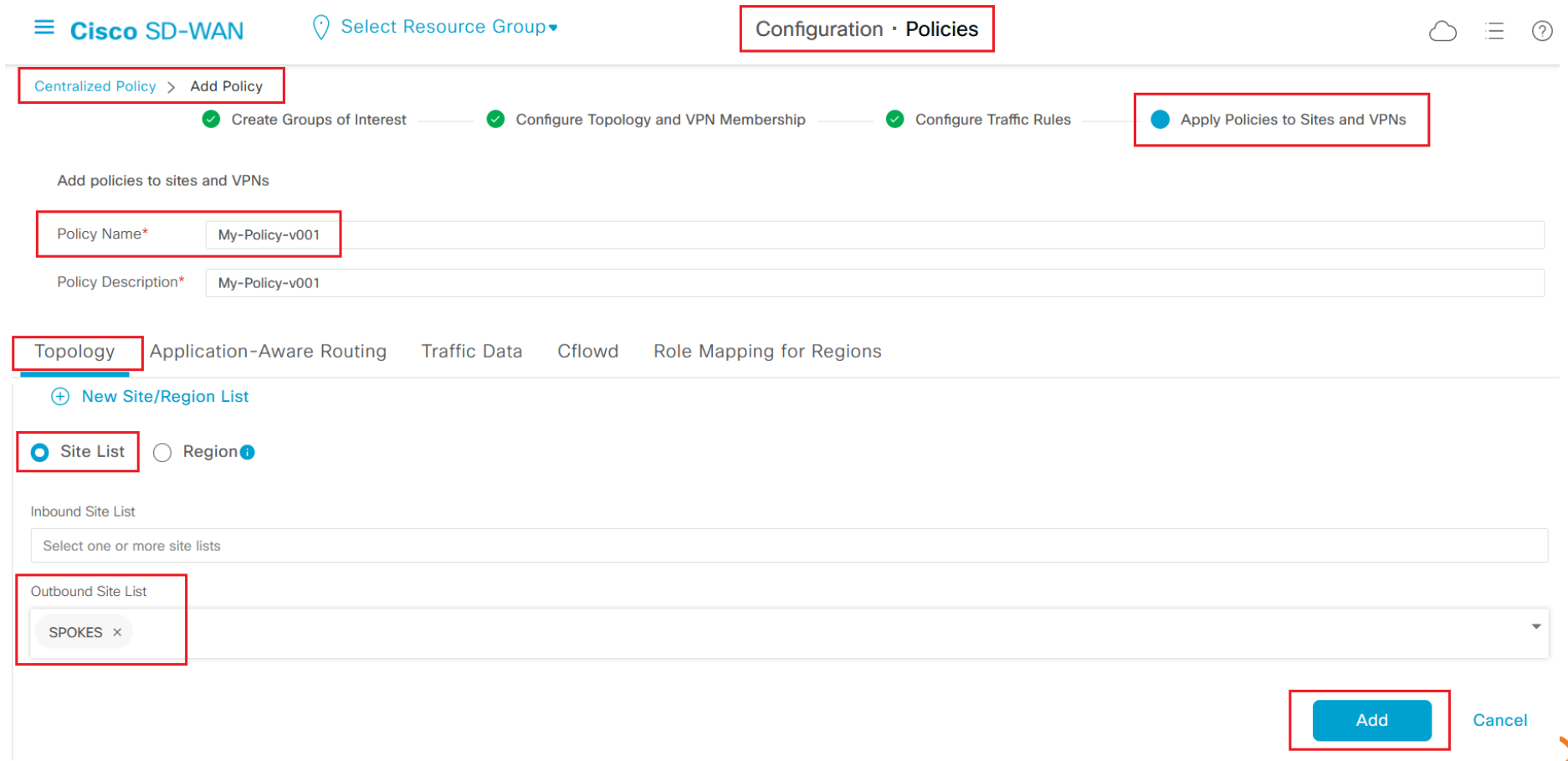
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Verify the Default Action is Reject (implicit deny at the end), then Save



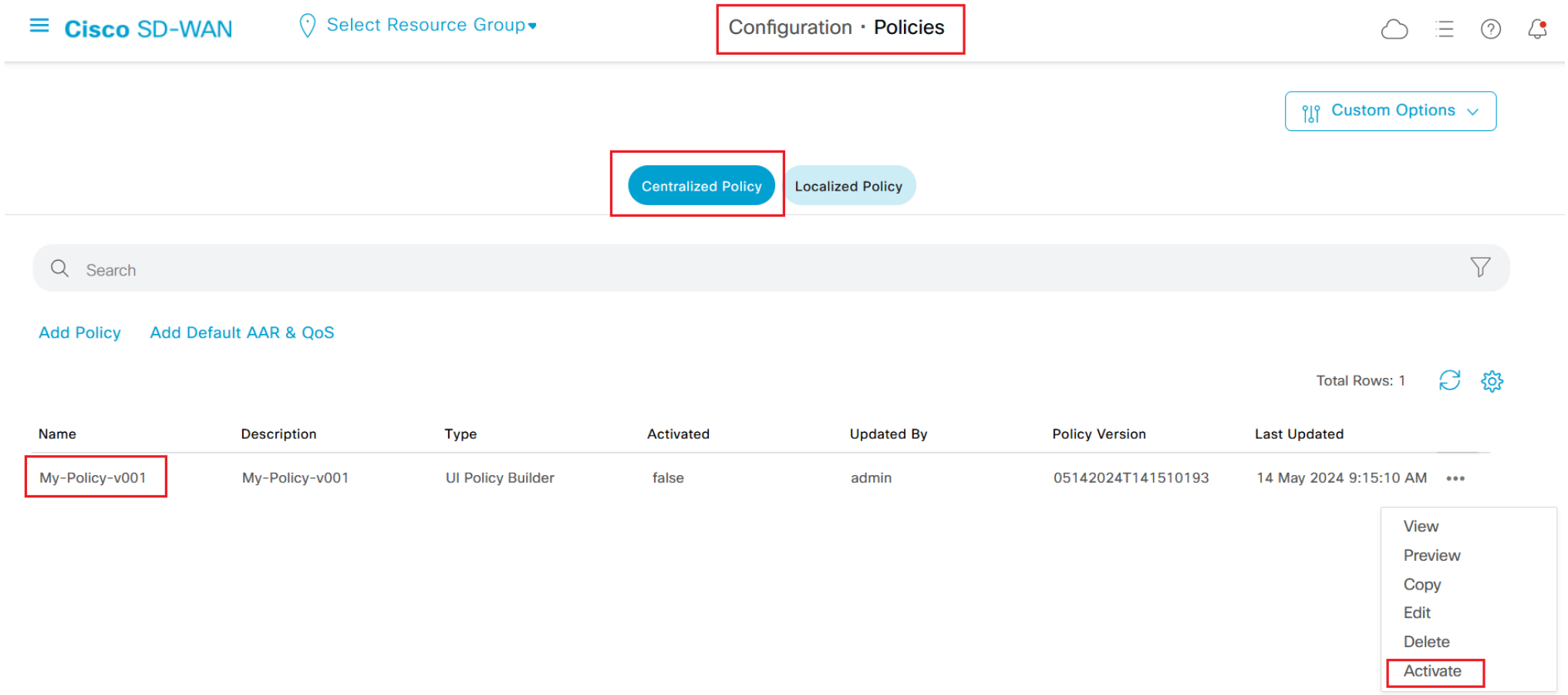
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Now we apply the Policy outbound to all "**Spokes**" Sites:
  - Recall this is ***outbound from the perspective of the vSmart***



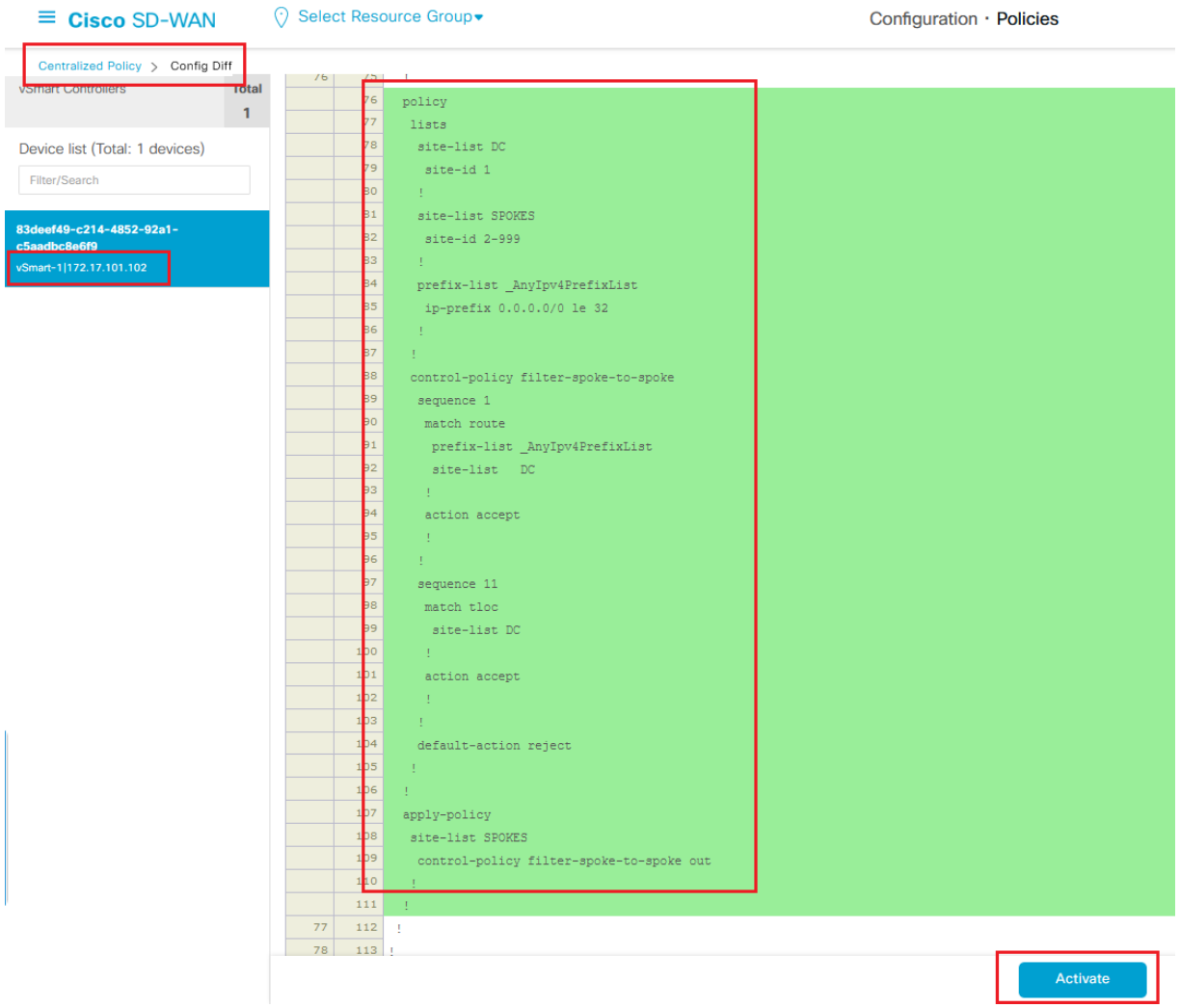
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- The final step is to "**Activate**" the Policy, which sends the config to vSmart



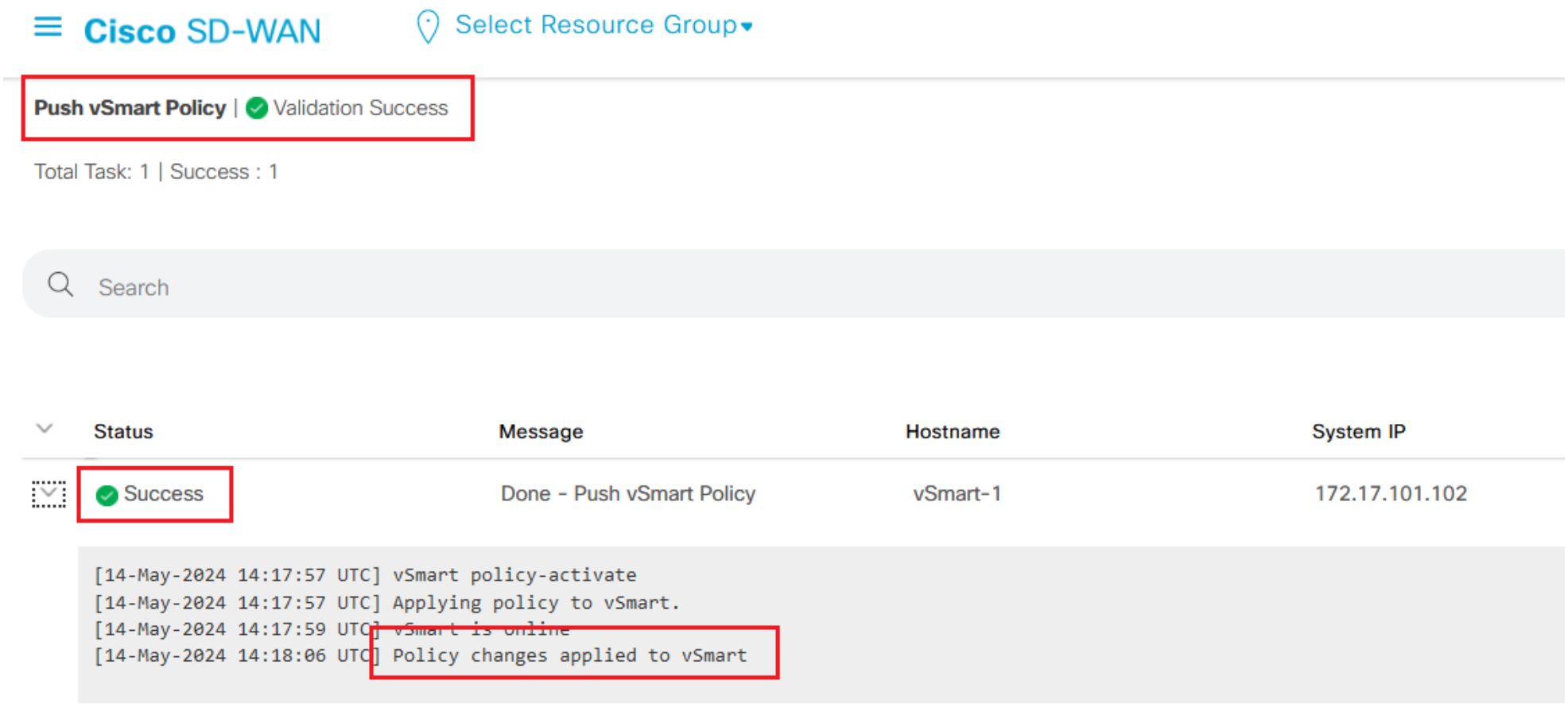
SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

+ Like with Templates, vSmart Policy activation can show you the config-diff



SD-WAN Policy Example: Filtering Spoke to Spoke Traffic (cont.)

- Finally, the Policy config is pushed to vSmart



Verifying Policies from vSmart CLI

```
vSmart-1# show run | begin ^policy
policy
 lists
  site-list DC
   site-id 1
  !
  site-list SPOKES
   site-id 2-999
  !
  prefix-list _AnyIpv4PrefixList
   ip-prefix 0.0.0.0/0 le 32
!
 control-policy filter-spoke-to-spoke
  sequence 1
   match route
    prefix-list _AnyIpv4PrefixList
    site-list   DC
   !
   action accept
   !
  !
  sequence 11
   match tloc
    site-list DC
   !
   action accept
   !
  !
  default-action reject
 !
!
apply-policy
 site-list SPOKES
  control-policy filter-spoke-to-spoke out
 !
!
vSmart-1#
```

Verifying Policy Results from WAN Edge Routers

- Spokes (**vEdge-3** & **vEdge-4**) now won't form IPsec tunnels with each other:

```
vEdge-3# show bfd sessions | in 172.16.4.4

vEdge-4# show bfd sessions | in 172.16.3.3
```

- Spokes will no longer have each other's routes:

```
vEdge-3# show omp route vpn 1 192.168.44.0/24
         show omp routes-table family ipv4 received-entries vpn 1 192.168.44.0/24
---------------------------------------------------------------^
syntax error: unknown argument
Error executing command: CLI command error -

vEdge-4# show ip route vpn 1 192.168.33.0/24
         show ip routes-table vpn 1 ipv4 192.168.33.0/24 *
------------------------------^
syntax error: unknown argument
Error executing command: CLI command error -
```

Verifying Policy Results from WAN Edge Routers (cont.)

- Spokes will still form IPsec tunnels with the DC (Site-ID 1)

```
vEdge-3# show bfd sessions
                                       SOURCE TLOC   REMOTE TLOC                DST PUBLIC   DST PUBLIC
SYSTEM IP    SITE ID   STATE   COLOR   COLOR         SOURCE IP    IP            PORT         ENCAP   <snip>
-------------------------------------------------------------------------------------------------<snip>
172.17.1.1   1         up      mpls    custom1       10.33.0.1    150.11.1.0    12346        ipsec   <snip>
172.17.1.1   1         up      mpls    custom2       10.33.0.1    150.22.1.0    12346        ipsec   <snip>
172.17.1.1   1         up      biz-internet custom1  150.33.0.1   150.11.1.0    12346        ipsec   <snip>
172.17.1.1   1         up      biz-internet custom2  150.33.0.1   150.22.1.0    12346        ipsec   <snip>
172.17.2.2   1         up      mpls    custom1       10.33.0.1    150.11.2.0    12346        ipsec   <snip>
172.17.2.2   1         up      mpls    custom2       10.33.0.1    150.22.2.0    12346        ipsec   <snip>
172.17.2.2   1         up      biz-internet custom1  150.33.0.1   150.11.2.0    12346        ipsec   <snip>
172.17.2.2   1         up      biz-internet custom2  150.33.0.1   150.22.2.0    12346        ipsec   <snip>
```

Verifying Policy Results from WAN Edge Routers (cont.)

- Result is that Spokes can reach routes from the DC, but not other Spokes:

```
vEdge-3# ping vpn 1 192.168.44.1
Ping in VPN 1
PING 192.168.44.1 (192.168.44.1) 56(84) bytes of data.
From 127.1.0.2 icmp_seq=1 Destination Net Unreachable
From 127.1.0.2 icmp_seq=2 Destination Net Unreachable
<snip>

vEdge-3# ping vpn 1 4.4.4.4
Ping in VPN 1
PING 4.4.4.4 (4.4.4.4) 56(84) bytes of data.
64 bytes from 4.4.4.4: icmp_seq=1 ttl=254 time=54.0 ms
64 bytes from 4.4.4.4: icmp_seq=2 ttl=254 time=51.3 ms
<snip>
```

Implementing Cisco SD-WAN

SD-WAN Policy Example – Traffic Engineering

Overlay Management Protocol (OMP) Review

- Cisco SD-WAN uses Overlay Management Protocol (OMP) for routing decisions
  - OMP is a custom (proprietary) protocol, but very similar in logic to BGP
  - vSmart Controller acts like a BGP Route Reflector (RR)
- WAN Edge Routers first send OMP updates to vSmart over DTLS
  - vSmart can apply user-defined routing policies on these ***inbound*** updates
- Next, vSmart performs OMP best-path selection on a per-prefix per-VPN basis
  - E.g. 192.168.0.0/24 in VPN 1 is a separate selection from 192.168.0.0/24 in VPN 2
  - OMP prefers routes with a higher “preference” value

Overlay Management Protocol (OMP) Review (cont.)

- Only “best” paths on vSmart are candidate to be advertised to other Edge Routers
  - vSmart will advertise up to 4 multi-paths by default
  - Can be modified with **`send-path-limit`** under **`omp`** global config mode on vSmart
- vSmart then reflects “best” routes back to other WAN Edge Routers
  - Can be modified with **`omp send-backup-paths`** in vSmart global config
  - vSmart can apply user-defined routing policies on these ***outbound*** updates

Cisco SD-WAN Traffic Engineering

- Traffic Engineering (i.e. modifying path selection) can be accomplished in multiple ways in Cisco's SD-WAN implementation
  - Centralized Policies on vSmart
    - Modify OMP attributes on routes received from WAN Edge Routers
      - I.e. an ***inbound*** policy from the perspective of vSmart
    - Modify OMP attributes on routes sent to WAN Edge Routers
      - I.e. an ***outbound*** policy from the perspective of vSmart
  - Localized Policies on WAN Edge Routers
    - E.g. modify attributes when BGP is redistributed into OMP, or vice-versa
    - Localized Policies are attached using Device Templates & Feature Templates, not through vSmart advertisements

Cisco SD-WAN Traffic Engineering with Centralized Policies Example

- Data Center (Site 1) devices **vEdge1** & **vEdge2** both have dual WAN connections
  - Links to **DC-Edge-1** are color “custom1”
  - Links to **DC-Edge-2** are color “custom2”
- By default, remote sites should see 4 paths to the DC routes
  - Paths 1 & 2 via **vEdge1** colors “custom1” & “custom2”
  - Paths 3 & 4 via **vEdge2** colors “custom1” & “custom2”
- Configure a Traffic Engineering policy as follows:
  - Load-share all traffic to the DC between **vEdge1** “custom1” & **vEdge2** “custom2”
  - Fallback to **vEdge1** “custom2” & **vEdge2** “custom1” if both primary paths are down

Cisco SD-WAN Traffic Engineering with Centralized Policies Example

```
vSmart-1# sh run | begin ^policy
policy
 lists
  tloc-list vEdge-1-custom1
   tloc 172.17.1.1 color custom1 encap ipsec
  !
  tloc-list vEdge-2-custom2
   tloc 172.17.2.2 color custom2 encap ipsec
  !
  site-list DC
   site-id 1
  !
  prefix-list _AnyIpv4PrefixList
   ip-prefix 0.0.0.0/0 le 32
  !
 !
 control-policy DC-Inbound-Traffic-Engineering
  sequence 1
   match route
    prefix-list _AnyIpv4PrefixList
    tloc-list   vEdge-1-custom1
   !
   action accept
    set
     preference 200
  !
  sequence 11
   match route
    prefix-list _AnyIpv4PrefixList
    tloc-list   vEdge-2-custom2
   !
   action accept
    set
     preference 200
    !
   !
  !
  default-action accept
 !
!
apply-policy
 site-list DC
  control-policy DC-Inbound-Traffic-Engineering in
 !
!
```

Verifying SD-WAN Policies

- vSmart
  - **`show run [policy]`**
- vEdge & cEdge
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] bfd sessions`**
  - **`show ip route`**
  - **`show omp route`**
  - **`show [sdwan] policy service-path`**

Implementing Cisco SD-WAN

SD-WAN Policy Example – Direct Internet Access (DIA)

Direct Internet Access (DIA) Overview

- Many SD-WAN designs have a default route in both the *Underlay* & the *Overlay*
  - *Underlay* default route is used to establish transport from the WAN Edge Routers to the Controllers, and to other sites' WAN Edge Routers over the Internet
    - E.g. transport for DTLS & IPsec tunnels over biz-internet, LTE, etc.
  - *Overlay* default route is used to centrally collect Internet traffic over the SD-WAN
    - E.g. send end-users' Internet traffic across the SD-WAN to the DC so it can be inspected by centralized firewalls
- Centralized Internet connectivity has multiple design issues
  - Central bandwidth/firewalls must be sized correctly to account for all remote sites
  - Increasing capacity may result in "forklift" upgrades
    - E.g. you need to buy a bigger router/firewall that can handle the increased flows
- DIA is designed to fix this problem by distributing Internet access to remote sites

How Direct Internet Access (DIA) Works

- Goal of DIA is to limit traffic sent over the SD-WAN to only internal destinations
  - E.g. traffic to internal private addresses (RFC 1918) should use the SD-WAN
- Traffic to external destinations (e.g. the Internet) is sent to the local *Underlay*
  - Leak the traffic from the Service VPN (e.g. VPN 1) into the Transport VPN (VPN 0)
  - Perform a source Network Address Translation (NAT) to the Underlay address(es)
    - E.g. NAT overload to the public IP of the WAN link
- Result is that traffic for the Internet is distributed amongst the remote sites
  - No need to send traffic to the DC first before going to the Internet
  - Reduces the load on centralized DC Internet connections and firewalls

Implementing Direct Internet Access (DIA)

- DIA is effectively a form of Policy Based Routing (PBR)
  - Overrides the normal routing lookup with a policy-based lookup…
    - IF traffic destination == RFC1918, use the SD-WAN
    - ELSE IF traffic destination != RFC1918, NAT to the Underlay (VPN 0)

- DIA is implemented using a Centralized Data (Traffic) Policy
  - Match destination RFC1918, set action Accept
    - I.e. perform normal forwarding for traffic going to internal addresses
  - Match destination ANY, set action Accept & NAT to VPN 0
    - I.e. leak Internet-bound traffic to the local Underlay; don't use the SD-WAN

- **Fallback** option can be used to prevent blackholes if local Internet link is down
  - E.g. fallback to the SD-WAN tunnel to DC over MPLS if local ISP is down

Cisco SD-WAN Direct Internet Access (DIA) Example

```
vSmart-1# sh run | begin ^policy
policy
 data-policy _Primary-VPN_DIA
  vpn-list Primary-VPN
    sequence 1
     match
      destination-data-prefix-list Internal
     !
     action accept
     !
    !
    sequence 11
     match
      destination-data-prefix-list Any
     !
     action accept
      nat use-vpn 0
      nat fallback
     !
    !
  default-action drop
 !
lists
  data-prefix-list Any
   ip-prefix 0.0.0.0/0
  !
  data-prefix-list Internal
   ip-prefix 10.0.0.0/8
   ip-prefix 172.16.0.0/12
   ip-prefix 192.168.0.0/16
  !
  site-list Spokes
   site-id 2-999
  !
  vpn-list Primary-VPN
   vpn 1
 !
!
apply-policy
 site-list Spokes
  data-policy _Primary-VPN_DIA from-service
```

Verifying SD-WAN Policies

- vSmart
  - **`show run [policy]`**
- vEdge & cEdge
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] bfd sessions`**
  - **`show ip route`**
  - **`show omp route`**
  - **`show [sdwan] policy service-path`**

Implementing Cisco SD-WAN

SD-WAN Policy Example – Isolating Guest User Access

Cisco SD-WAN Default VPN Forwarding Review

- Cisco SD-WAN automatically defaults to any-to-any connectivity over IPsec
  - All sites form tunnels to all other sites out all colors by default
    - E.g. internet to internet, mpls to mpls, internet to mpls, mpls to internet, etc.
  - All sites advertise connected routes into OMP with all TLOCs by default
    - E.g. Edge Router 172.17.1.1 has WAN links to Biz-Internet & MPLS
      - 192.168.1.0/24 in VPN 1 is via TLOC 172.17.1.1 color Biz-Internet
      - 192.168.1.0/24 in VPN 1 is via TLOC 172.17.1.1 color MPLS
  - Routes do not leak between VPNs by default
    - E.g. all VPN 1 sites can reach all other VPN 1 sites, but not VPN 2
    - E.g. all VPN 2 sites can reach all other VPN 2 sites, but not VPN 1

Cisco SD-WAN Guest User Access & VPN Membership Filtering

- In addition to private-to-private and private-to-Internet traffic over the SD-WAN, sites may also want to offer Guest-to-Internet connectivity

- First step in providing Guest access is to segment the traffic into different VPNs
  - E.g. Service VPN 1 for “Corp-VPN” and Service VPN 2 for “Guest-VPN”
  - This prevents Guest users from sending traffic to internal resources, and vice-versa
  - *This does not prevent Guests in Site 1 from accessing Guests in Site 2 by default*

- Next step is to filter the Guest network advertisements across the SD-WAN
  - By default, vSmart accepts/sends advertisements for all VPNs
  - “**VPN Membership**” filter is used to limit which VPNs are accepted/sent to which sites
  - E.g. match VPN list “Only-Corp-VPN”, set action accept, apply to site list “All-Sites”

VPN Membership Filter Example

```
vSmart-1# sh run | begin ^policy | nomore
policy
 lists
  vpn-list Only-Corp-VPN
   vpn 1
  !
  site-list All-Sites
   site-id 1-999
!
 vpn-membership vpnMembership_-969927050
  sequence 10
   match
    vpn-list Only-Corp-VPN
   !
   action accept
  !
  default-action reject
!
apply-policy
 site-list All-Sites
  vpn-membership vpnMembership_-969927050
```

Verifying VPN Membership Filters

```
vSmart-1# show omp route vpn 2 | tab
Code:
C   -> chosen
I   -> installed
Red -> redistributed
Rej -> rejected
<snip>
Inv -> invalid
<snip>

VPN    PREFIX              FROM PEER       ID     LABEL    STATUS     TYPE       TLOC IP      <snip>
-------------------------------------------------------------------------------------------<snip>
2      172.16.33.0/24      172.17.3.3      66     1006     Rej,R,Inv  installed  172.17.3.3   <snip>
                           172.17.3.3      68     1006     Rej,R,Inv  installed  172.17.3.3   <snip>
2      172.16.44.0/24      172.17.4.4      66     1006     Rej,R,Inv  installed  172.17.4.4   <snip>
                           172.17.4.4      68     1006     Rej,R,Inv  installed  172.17.4.4   <snip>
2      172.16.101.0/24     172.17.101.1    66     1004     Rej,R,Inv  installed  172.17.101.1 <snip>
                           172.17.101.1    68     1004     Rej,R,Inv  installed  172.17.101.1 <snip>
                           172.17.102.1    66     1004     Rej,R,Inv  installed  172.17.102.1 <snip>
                           172.17.102.1    68     1004     Rej,R,Inv  installed  172.17.102.1 <snip>
```

Guest Direct Internet Access (DIA)

- Guest-to-Internet traffic should use the local Underlay for Internet access
  - E.g. don't follow a default route across the SD-WAN to the DC for Guest Internet

- Solution is to combine VPN Membership filtering with the previous DIA example
  - Control Policy for Guest Route Filtering
    - IF VPN == Corp-VPN && Site == Any, action Accept
    - ELSE, action Reject
  - Data Policy for Guest Direct Internet Access (DIA)
    - IF VPN == Guest-VPN && traffic destination == Internal, action Reject
    - ELSE IF VPN == Guest-VPN && traffic destination != Internal, action NAT to VPN 0

Guest DIA with VPN Membership Filtering Example – Part 1

```
vSmart-1# sh run | begin ^policy | nomore
policy
 data-policy _Guest-VPN_Guest-DIA
  vpn-list Guest-VPN
   sequence 1
    match
     destination-data-prefix-list Internal
    !
    action drop
    !
   !
   sequence 11
    match
     destination-data-prefix-list Any
    !
    action accept
     nat use-vpn 0
    !
   !
   default-action drop
  !
 lists
  vpn-list Corp-VPN
   vpn 1
  !
  vpn-list Guest-VPN
   vpn 2
  !
  data-prefix-list Any
   ip-prefix 0.0.0.0/0
  !
  data-prefix-list Internal
   ip-prefix 10.0.0.0/8
   ip-prefix 172.16.0.0/12
   ip-prefix 192.168.0.0/16
  !
  site-list All-Sites
   site-id 1-999
  !
  site-list Spokes
   site-id 2-999
```

Guest DIA with VPN Membership Filtering Example – Part 2

```
vpn-membership vpnMembership_-969927050
  sequence 10
   match
    vpn-list Corp-VPN
   !
   action accept
   !
  !
  default-action reject
 !
!
apply-policy
 site-list All-Sites
  vpn-membership vpnMembership_-969927050
 !
 site-list Spokes
  data-policy _Guest-VPN_Guest-DIA from-service
```

Verifying SD-WAN Policies

- vSmart
  - **`show run [policy]`**
- vEdge & cEdge
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] bfd sessions`**
  - **`show ip route`**
  - **`show omp route`**
  - **`show [sdwan] policy service-path`**

Implementing Cisco SD-WAN

SD-WAN Policy Example – Extranets & VPN Route Leaking

Cisco SD-WAN VPN Route Leaking Overview

- By default, a full-mesh of connectivity exists within each Service VPN
  - Hosts in Service VPN 1 can reach all other sites with VPN 1, but not VPN 2
  - Hosts in Service VPN 2 can reach all other sites with VPN 2, but not VPN 1

- Route Leaking is the process of allowing selective access between VPN boundaries
  - E.g. copy route X from VPN 1 into VPN 2, and copy route Y from VPN 2 into VPN 1
  - Result is that X & Y have reachability to each other even though they're in separate VPNs
  - Implies that X & Y are unique networks, i.e. IP addresses do not overlap

Implementing Cisco SD-WAN VPN Route Leaking

- Cisco SD-WAN implements Route Leaking through Centralized Control Policies
  - Control Policy **In-From-Corp-VPN-Sites**
    - IF VPN == **Corp-VPN** && prefix-list == **Corp-VPN-Routes-to-Leak**, action Accept && export to VPN **Extranet-VPN**
    - ELSE Accept
      - I.e. don't filter other prefixes
  - Control Policy **In-From-Extranet-VPN-Sites**
    - IF VPN == **Extranet-VPN** && prefix-list == **Extranet-VPN-Routes-to-Leak**, action Accept && export to VPN **Corp-VPN**
    - ELSE Accept
      - I.e. don't filter other prefixes

Controlling Cisco SD-WAN VPN Route Leaking

- Route Leaking is applied inbound from the perspective of vSmart
  - SD-WAN sites advertise OMP routes outbound to vSmart
  - vSmart receives routes inbound and applies the policy (leak the routes into target VPN)
  - Leaked routes are then advertised outbound from vSmart to all sites by default

- Implies that additional filters may be needed to control where leaked routes are sent
  - E.g. in our case both the DC & Spoke sites learn about the Extranet routes
  - Filtering advertisements from vSmart outbound to Spokes would need a 3rd policy
    - E.g. when Extranet routes are leaked into Corp-VPN, set an OMP tag inbound
    - 3rd Policy out to Spokes, match this tag & set action reject, then default action accept

Cisco SD-WAN VPN Route Leaking Example – Part 1 (Define Lists)

```
vSmart-1# sh run | begin ^policy | nomore
policy
 lists
  vpn-list Corp-VPN
   vpn 1
  !
  vpn-list Extranet-VPN
   vpn 777
  !
  site-list DC
   site-id 1
  !
  site-list Extranet-Site
   site-id 101
  !
  site-list Spokes
   site-id 2-999
  !
  prefix-list DC-Shared-Services
   ip-prefix 4.4.4.4/32
  !
  prefix-list Extranet-Routes
   ip-prefix 7.7.7.7/32
  !
  prefix-list _AnyIpv4PrefixList
   ip-prefix 0.0.0.0/0 le 32
```

Cisco SD-WAN VPN Route Leaking Example – Part 2 (Define Policies)

```
control-policy Leak-Extranet-to-Corp-VPN
  sequence 1
   match route
    prefix-list Extranet-Routes
    vpn-list    Extranet-VPN
   !
   action accept
    set
     omp-tag 777
    !
    export-to
     vpn-list Corp-VPN
  !
  default-action accept
 !
```

```
control-policy Leak-DC-Services-to-Extranet
  sequence 1
   match route
    prefix-list DC-Shared-Services
    vpn-list    Corp-VPN
   !
   action accept
    export-to
     vpn-list Extranet-VPN
  !
  default-action accept
 !
control-policy Filter-Extranet-to-Spokes
  sequence 1
   match route
    omp-tag     777
    prefix-list _AnyIpv4PrefixList
   !
   action reject
  !
  default-action accept
 !
```

Cisco SD-WAN VPN Route Leaking Example – Part 3 (Apply Policies)

```
apply-policy
 site-list DC
  control-policy Leak-DC-Services-to-Extranet in
 !
 site-list Extranet-Site
  control-policy Leak-Extranet-to-Corp-VPN in
 !
 site-list Spokes
  control-policy Filter-Extranet-to-Spokes out
 !
```

Verifying Cisco SD-WAN VPN Route Leaking from vSmart CLI

```
vSmart-1# show omp route 4.4.4.4/32
Code:
C   -> chosen
<snip>
Ext -> extranet
<snip>
```

VPN	PREFIX	FROM PEER	PATH ID	LABEL	STATUS	ATTRIBUTE TYPE	TLOC IP	<snip>
								<snip>
1	4.4.4.4/32	172.17.1.1	78	1005	C,R	installed	172.17.1.1	<snip>
		172.17.1.1	79	1005	C,R	installed	172.17.1.1	<snip>
		172.17.2.2	78	1005	C,R	installed	172.17.2.2	<snip>
		172.17.2.2	79	1005	C,R	installed	172.17.2.2	<snip>
777	4.4.4.4/32	172.17.1.1	101	1005	C,R,Ext	original	172.17.1.1	<snip>
						installed	172.17.1.1	<snip>
		172.17.1.1	102	1005	C,R,Ext	original	172.17.1.1	<snip>
						installed	172.17.1.1	<snip>
		172.17.2.2	101	1005	C,R,Ext	original	172.17.2.2	<snip>
						installed	172.17.2.2	<snip>
		172.17.2.2	102	1005	C,R,Ext	original	172.17.2.2	<snip>
						installed	172.17.2.2	<snip>

Verifying SD-WAN Policies

- vSmart
  - **`show run [policy]`**
- vEdge & cEdge
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] bfd sessions`**
  - **`show ip route`**
  - **`show omp route`**
  - **`show [sdwan] policy service-path`**

Implementing Cisco SD-WAN

SD-WAN Policy Example – Service Insertion

Cisco SD-WAN Service Insertion / Service Chaining Overview

- Cisco SD-WAN **Service Insertion** is the process of automatically inserting external devices such as Firewalls, Intrusion Prevention Systems (IPS), Load Balancers, etc. into the SD-WAN data-plane based on a Policy
  - **Service Chaining** means combining multiple devices together, e.g. Firewall first, Load Balancer next, etc.

- Service Insertion in SD-WAN uses a combination of functions such as…
  - Feature Templates to define the device type (FW/IPS/etc.) and location
  - Centralized Control Policies with match criteria about when insertion will happen
    - Can also be a Data Policy or ACLs starting in vManage release 20.13.x
  - OMP routes & MPLS label values to trigger traffic redirection in the data-plane

How Cisco SD-WAN Service Insertion Works – Service Routes

- **Service Insertion** starts by advertising a "**Service Route**" to vSmart via OMP
  - **Service Route** originator is the SD-WAN Edge Router attached to the Firewall / IPS / LB
- The OMP **Service Route** is used to…
  - Describes the type of service (i.e. FW vs. IPS)
  - Assign an MPLS label locally unique to the originator of the **Service Route**
- vSmart uses the **Service Route** to trigger traffic redirection based on a Policy
  - E.g. IF Policy match == TRUE
    - THEN action == accept && **set service FW vpn 1**

How Cisco SD-WAN Service Insertion Works – Triggering Redirection

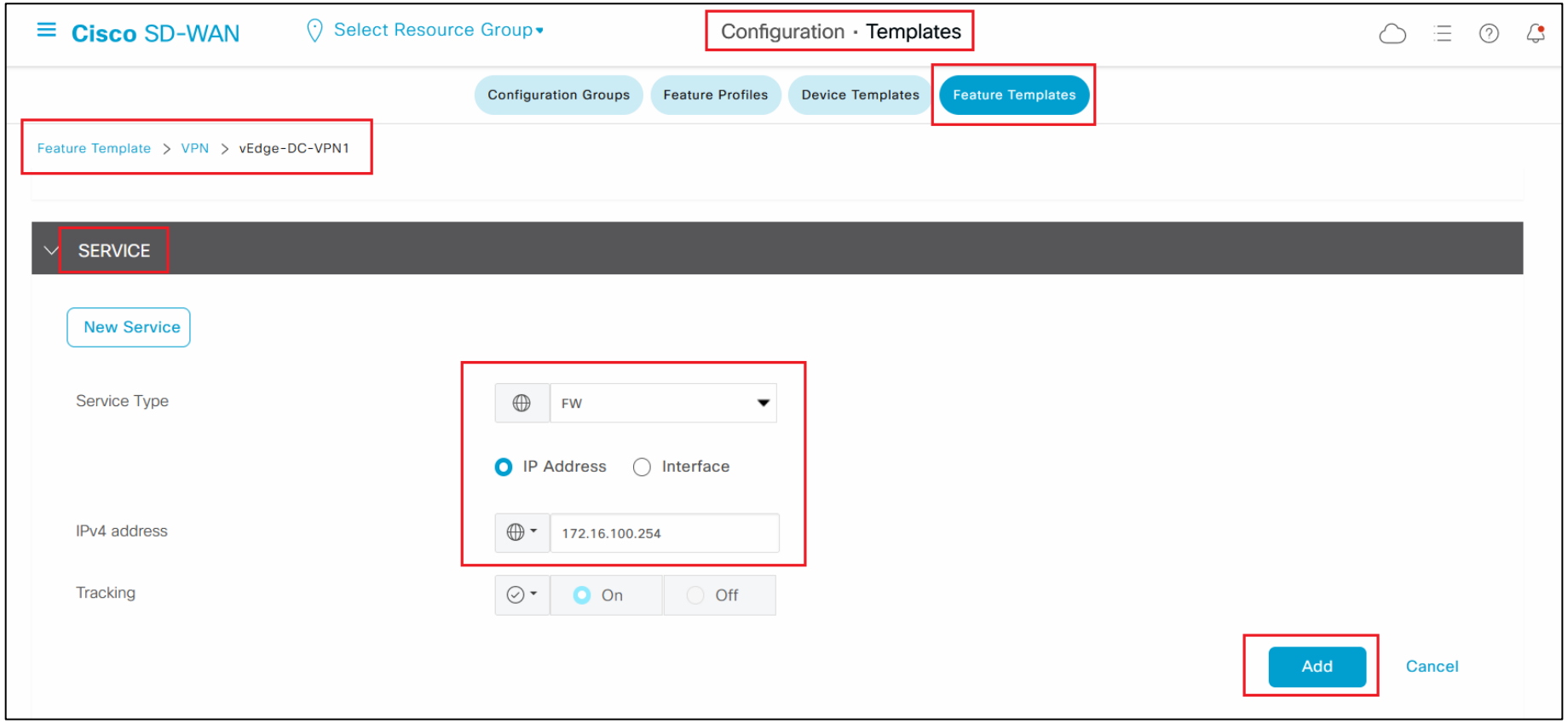
- Once the **Service Route** is learned by vSmart, redirection is triggered by a Policy
  - In our example, redirection will be based on matching a route (Control Policy)
  - In newest vManage code, redirection can also be based on a Data Policy and/or ACL
    - E.g. redirect only TCP Port 80 to the IPS

- When Policy match occurs and action is "**set service**", two key changes occur:
  - OMP TLOCs are re-written to the originator of the **Service Route**
    - E.g. change the next-hop to tunnel(s) towards the Edge Router attached to Firewall

  - MPLS label in OMP route is re-written to that of the **Service Route**
    - Data-plane packets use this new MPLS label when sending traffic to the re-written TLOCs, where traffic is redirected based on this locally unique label
      - E.g Firewall insertion has a different label than IPS insertion

Implementing SD-WAN Service Insertion – Advertising Network Service Availability

- First define where the FW/IPS/etc. is located in the SD-WAN fabric
  - Defined in the Feature Template for the Service VPN of the Edge Router(s) attached to the Firewall



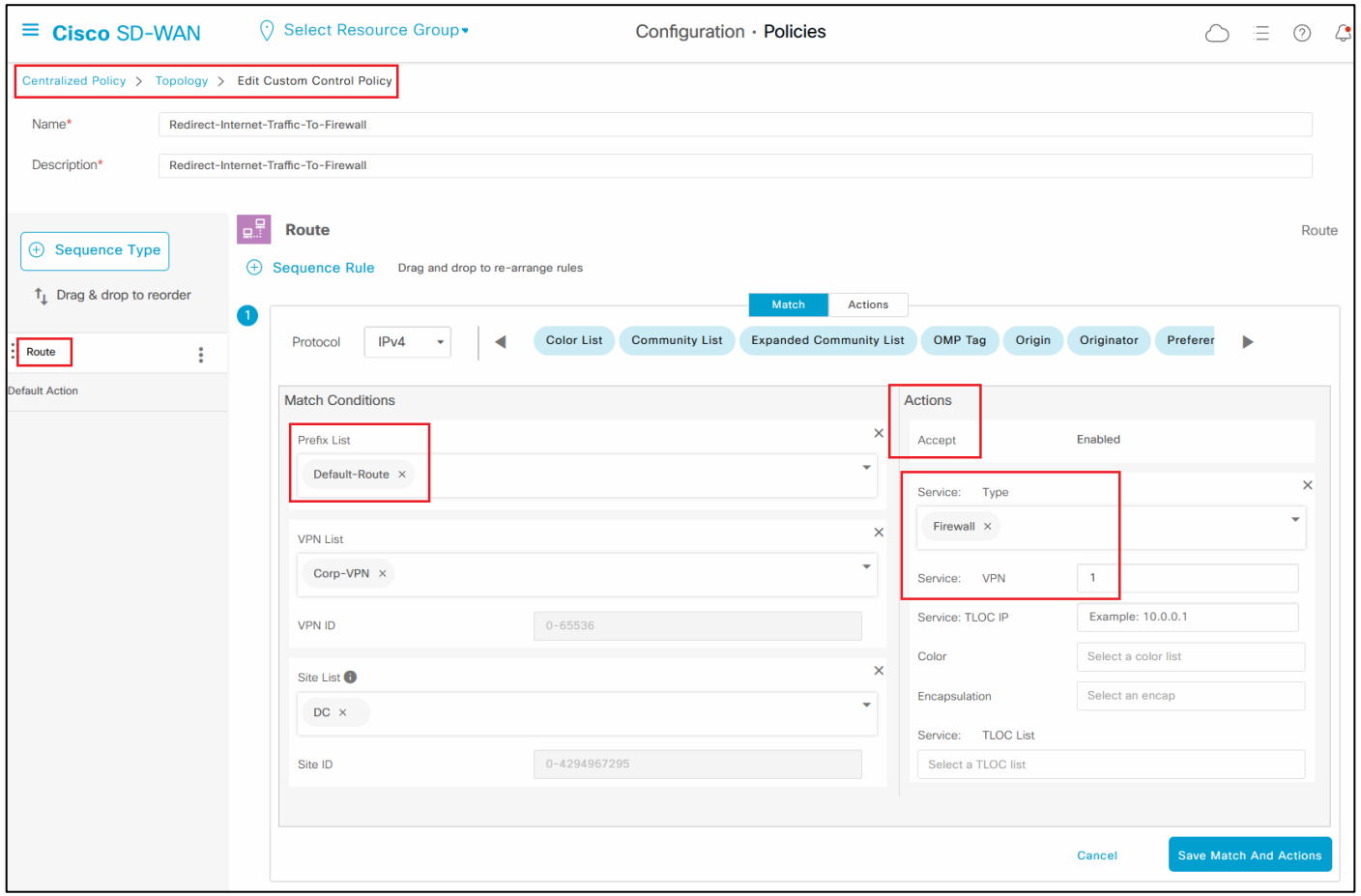
Implementing SD-WAN Service Insertion – Verifying Service Routes

- Once the Feature Template has the Service defined and is attached, the WAN Edge Routers advertise the Service Routes to vSmart
  - Service Route represents the TLOCs & MPLS Label that will be re-written to perform traffic redirection (i.e. Service Insertion)

```
vSmart-1# show omp services service FW
<snip>

ADDRESS                                                  PATH    REGION
FAMILY    VPN     SERVICE   ORIGINATOR   FROM PEER       ID      ID      LABEL     STATUS
-------------------------------------------------------------------------------------------
ipv4      1       FW        172.17.1.1   172.17.1.1      78      None    1006      C,I,R
                                         172.17.1.1      79      None    1006      C,I,R
          1       FW        172.17.2.2   172.17.2.2      78      None    1006      C,I,R
                                         172.17.2.2      79      None    1006      C,I,R
```

Implementing SD-WAN Service Insertion – Defining the Redirection Policy

- Using a Custom Control Policy, define what traffic should be redirected
  - E.g. traffic using the default route in the Corp-VPN from the DC site will be redirected to the firewall in VPN 1



Implementing SD-WAN Service Insertion – Example Control Policy

```
vSmart-1# show run | begin ^policy
policy
 lists
  vpn-list Corp-VPN
   vpn 1
  !
  site-list DC
   site-id 1
  !
  site-list Spokes
   site-id 2-999
  !
  prefix-list Default-Route
   ip-prefix 0.0.0.0/0
  !
 !
```

```
 control-policy Firewall-Redirect
  sequence 1
   match route
    prefix-list Default-Route
    site-list   DC
    vpn-list    Corp-VPN
   !
   action accept
    set
     service FW vpn 1
    !
   !
  !
  default-action accept
 !
!
apply-policy
 site-list Spokes
  control-policy Firewall-Redirect out
```

Implementing SD-WAN Service Insertion – Traffic Forwarding Before Redirection

```
Cat8Kv-1#show sdwan omp route vpn 1 0.0.0.0/0
<snip>
                                                          PATH                                PSEUDO
TENANT     VPN     ROUTE ADDR   <snip>  FROM PEER         ID      LABEL     STATUS      KEY     TLOC IP          COLOR
----------------------------<snip>-------------------------------------------------------------------------------
0          1       0.0.0.0      <snip>  172.17.101.102    25      1005      C,I,R       1       172.17.1.1       custom1
                                <snip>  172.17.101.102    47      1005      C,I,R       1       172.17.1.1       custom2
                                <snip>  172.17.101.102    49      1005      C,I,R       1       172.17.2.2       custom1
                                <snip>  172.17.101.102    51      1005      C,I,R       1       172.17.2.2       custom2

Cat8Kv-1#show ip route vrf 1 8.8.8.8

Routing Table: 1
% Network not in table

Cat8Kv-1#traceroute vrf 1 8.8.8.8 numeric
Type escape sequence to abort.
Tracing the route to 8.8.8.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.10.10.1 20 msec 18 msec 20 msec
  2 10.9.9.9 43 msec *  36 msec
```

Implementing SD-WAN Service Insertion – Traffic Forwarding After Redirection

```
Cat8Kv-1#show sdwan omp route vpn 1 0.0.0.0/0
<snip>
                                <snip>                    PATH                          PSEUDO                              <snip>
TENANT     VPN     ROUTE ADDR   <snip> FROM PEER          ID      LABEL     STATUS      KEY      TLOC IP       COLOR      <snip>
---------------------------------<snip>-------------------------------------------------------------------------------<snip>
0          1       0.0.0.0      <snip> 172.17.101.102     69      1006      C,I,R       1        172.17.1.1    custom1    <snip>
                                <snip> 172.17.101.102     70      1006      C,I,R       1        172.17.1.1    custom2    <snip>
                                <snip> 172.17.101.102     71      1006      C,I,R       1        172.17.2.2    custom1    <snip>
                                <snip> 172.17.101.102     72      1006      C,I,R       1        172.17.2.2    custom2    <snip>

Cat8Kv-1#show ip route vrf 1 8.8.8.8

Routing Table: 1
% Network not in table

Cat8Kv-1#traceroute vrf 1 8.8.8.8 numeric
Type escape sequence to abort.
Tracing the route to 8.8.8.8
VRF info: (vrf in name/id, vrf out name/id)
  1 10.10.10.1 27 msec 16 msec 20 msec
  2 172.16.100.254 62 msec 41 msec 41 msec
  3 150.100.2.1 42 msec 35 msec 37 msec
  4 150.22.0.2 34 msec *  38 msec
```

Implementing Cisco SD-WAN

SD-WAN Policy Example – ACL Filtering

Cisco SD-WAN Access-Lists (ACLs) Overview

- ACLs are implemented in Cisco SD-WAN using **Localized Data Policies**
  - ACLs are locally significant, not advertised through a vSmart policy
- vManage centrally controls the ACL definitions
  - **Configuration > Policies > Localized Policy > Access Control Lists**
- ACLs are pushed to devices through their **Device Templates**
  - **Device Template > Additional Templates > Policy**
- ACLs are applied at the port level under the Interface
  - Under **ACL/QoS** define the ingress/egress IPv4/IPv6 ACL
- Allows you to re-use the same ACL between multiple devices
  - If the ACL is updated, all devices using a Device Template referencing the ACL will automatically be updated
  - Helps to prevent drift of ACL configurations
    - I.e. we don't have to manage ACLs on a box-by-box basis

Verifying Access-Lists

- vEdge & cEdge
  - **show [sdwan] access-list-associations**
  - **show [sdwan] access-list-counters**
  - **show [sdwan] access-list-names**
  - **show [sdwan] access-list-policers**

Implementing Cisco SD-WAN

Understanding Cisco SD-WAN Security

Cisco SD-WAN Security Overview

- Cisco SD-WAN has several security features built-in...
  - Application-Aware Enterprise Firewall
    - I.e. Zone-Based Policy Firewall (ZFW/ZBPF)
  - Inline Intrusion Detection/Prevention System (IDS/IPS)
    - Snort-based IPS with dynamic signature updates published by Cisco Talos
  - URL Filtering
    - Security virtual image from Cisco software portal contains URL categories
  - Advanced Malware Protection (AMP) and Threat Grid
    - Threat Grid API allows uploading of files to malware analysis
  - DNS Security with Cisco Umbrella
    - Requires registration through Umbrella portal
- Security Policy is defined under **Configuration > Security**
- Security Policy is applied under **Device Template**
  - **Additional Templates > Security Policy**

Implementing Cisco SD-WAN

Understanding Cisco SD-WAN Quality of Service (QoS)

Cisco SD-WAN Quality of Service (QoS) Overview

- Quality of Service (QoS) in Cisco SD-WAN is implemented using a combination of **Centralized Policies** and **Localized Policies**
  - Centralized Data Policies are used to classify traffic
    - Match could be DSCP, L3 SRC/DST L4 Ports, Application, etc.
  - Localized Policies are used for traffic scheduling & congestion management
    - Map traffic classes to hardware queues
    - Choose scheduling technique for the queue
      - Low Latency Queuing (LLQ) or Weighted Round Robin (WRR)
    - Choose congestion management technique for the queue
      - Random Early Detection (RED) or Tail Drop
    - Choose bandwidth reservation
    - Choose buffer percentage

- Centralized Data (Traffic) Policy is first used to classify traffic
  - Match traffic, then action is set Traffic Class
  - Centralized Policy is then activated through vSmart
    - E.g. apply to Spokes, direction From Service
- Create Localized QoS Lists to map Traffic Class to Queue
  - Queue 0 is always the Low Latency Queue (LLQ)
  - Queue 0 automatically includes the control-plane
    - E.g. DTLS tunnels from WAN Edge to Controllers
- Localized Policy is then used to apply QoS scheduling/management
  - Create a QoS Map to match Traffic Class
  - Define LLQ bandwidth limit and/or bandwidth/buffer reservation for Queues
  - Apply Localized Policy to Device Template
  - Reference QoS Map from WAN Feature Template

Verifying QoS Policies

- vSmart
  - **show run [policy]**
- vEdge & cEdge
  - **show [sdwan] running-config**
  - **show [sdwan] policy from-vsmart**
  - **show policy-map interface**

Implementing Cisco SD-WAN

Understanding Cisco SD-WAN Application Aware Routing (AAR)

Cisco SD-WAN Application Aware Routing (AAR) Overview

- **Application Aware Routing (AAR)** is used to optimize traffic routing across the overlay SD-WAN tunnels based on App performance requirements
- SD-WAN Edge Routers sample real-time traffic conditions of all IPsec tunnels using **Bidirectional Forwarding Detection (BFD)**
  - E.g. mpls to mpls, biz-internet to biz-internet, etc.
  - System calculates average packet loss, latency, and jitter for each path
- Sampling results are then compared against App QoS parameters
  - E.g. VoIP should have no more than 50ms delay, 25ms jitter, and 5% loss
- Traffic is dynamically routed over QoS compliant tunnels based on a Policy
  - E.g. IF App == Voice && SLA-Class Voice-And-Video == Compliant
    - THEN Prefer Color MPLS
    - ELSE Fallback-to-Best-Path

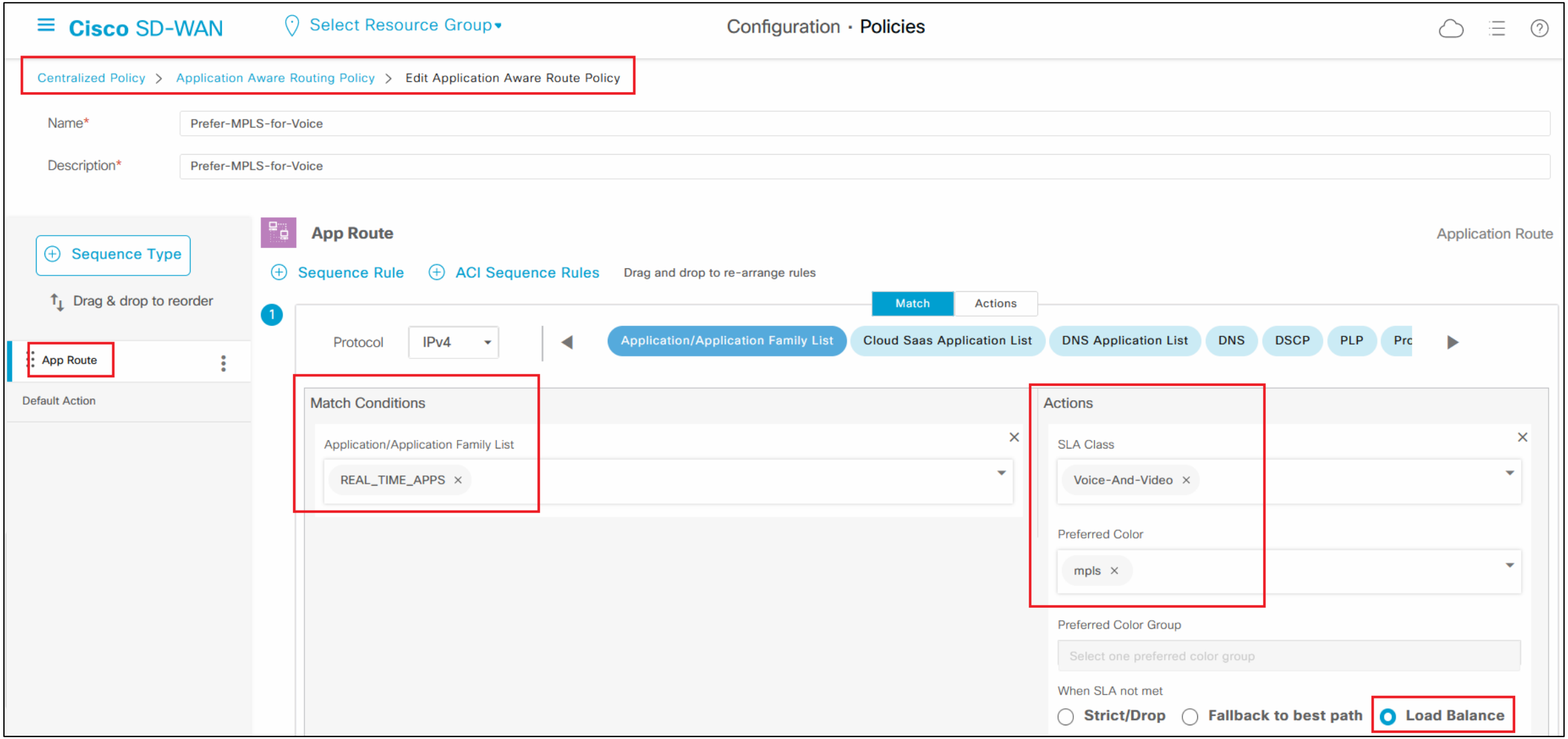
How Application Aware Routing (AAR) Works

- **Application Aware Routing (AAR)** is implemented through a **Centralized Data Policy**
  - Like previous Data Policy examples, AAR is effectively Policy Based Routing (PBR)
  - Match criteria is defined (e.g. App), then Action is to poll **SLA Class** & map traffic to tunnel(s)
- Match criteria for AAR can classify traffic up to Layer 7
  - vManage contains a variety of pre-defined App matches
    - E.g. Google Apps, Microsoft 365, VoIP, etc.
  - Custom matches can also be defined
    - E.g. TCP 80/443 + regex for *.myapp.com
- **SLA Class** defines the loss / latency / jitter parameters
  - Likewise a bunch of classes pre-defined in vManage
    - E.g. “Voice-and-Video”, “Best-Effort”, etc.

Application Aware Routing (AAR) Tunnel Mapping

- Traffic is ECMP load-balanced between all tunnels meeting the SLA by default
- If the SLA is not met (e.g. delay is too high), traffic is automatically re-routed
  - E.g. Use ECMP on all colors, but exclude the paths where SLA is not met
- **Preferred Color** can be defined in addition to SLA Class
  - E.g. Use MPLS first, but if the SLA is not met, use any other color (e.g. biz-internet)
- Additional branching logic can be defined
  - **Backup Preferred Color**
    - E.g. prefer MPLS, but fallback to just biz-internet instead of biz-internet & LTE
  - **Fallback to Best-Path**
    - E.g. prefer MPLS, but fallback to biz-internet or LTE, whichever is better
  - **Strict Mode**
    - E.g. prefer MPLS, but drop the packet if the SLA is not met

Defining an Application Aware Routing (AAR) Policy Example



Verifying Application Aware Routing (AAR) Policies from vSmart

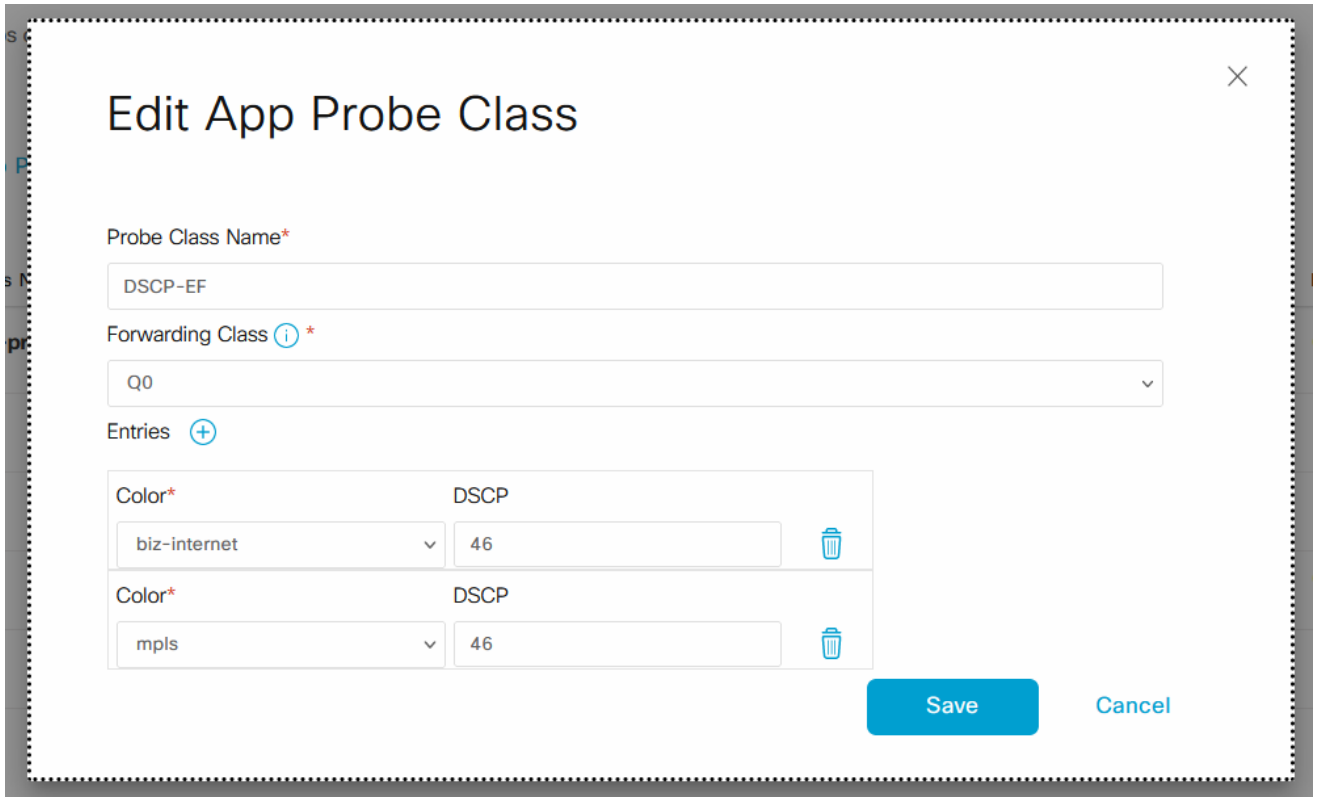
```
vSmart-1# sh run | begin ^policy
policy
 sla-class Voice-And-Video
  loss    2
  latency 45
  jitter  30
 !
 app-route-policy _Corp-VPN_Prefer-MPLS
  vpn-list Corp-VPN
   sequence 1
    match
     source-ip 0.0.0.0/0
     app-list  REAL_TIME_APPS
    !
    action
     sla-class Voice-And-Video preferred-color mpls
 !
lists
  vpn-list Corp-VPN
   vpn 1
  !
  app-list REAL_TIME_APPS
   app rtp
   app sccp
   app sip
   app sip_soap
   app skinny
   app uaudp_rtp
  !
  site-list Spokes
   site-id 2-999
!
apply-policy
 site-list Spokes
  app-route-policy _Corp-VPN_Prefer-MPLS
```

Application Aware Routing (AAR) App Probe Classes

- SD-WAN Edge Routers sample the loss/delay/jitter of tunnels using **BFD**
- **BFD** is marked as **DSCP CS6** (48) by default
  - **DSCP CS6** / IP Precedence 6 (Internetwork Control) **is higher priority than user traffic**
    - E.g. **DSCP EF** / IP Precedence 5 (Critical) for voice/video **is lower than BFD**
- Since BFD has a higher priority than user traffic, it may skew sampling results
  - E.g. users are having performance issues with “bulk-data”, but the problem is hidden since BFD is being processed with a higher priority during periods of congestion
- **App Probe Class** allows you to fix this by user defining…
  - Which output queue the BFD sampling packets are placed in
    - E.g. use the same queuing behavior of a VoIP phone call
  - What is the DSCP marking of the BFD sampling packets
    - E.g. use DSCP EF to simulate a VoIP phone call

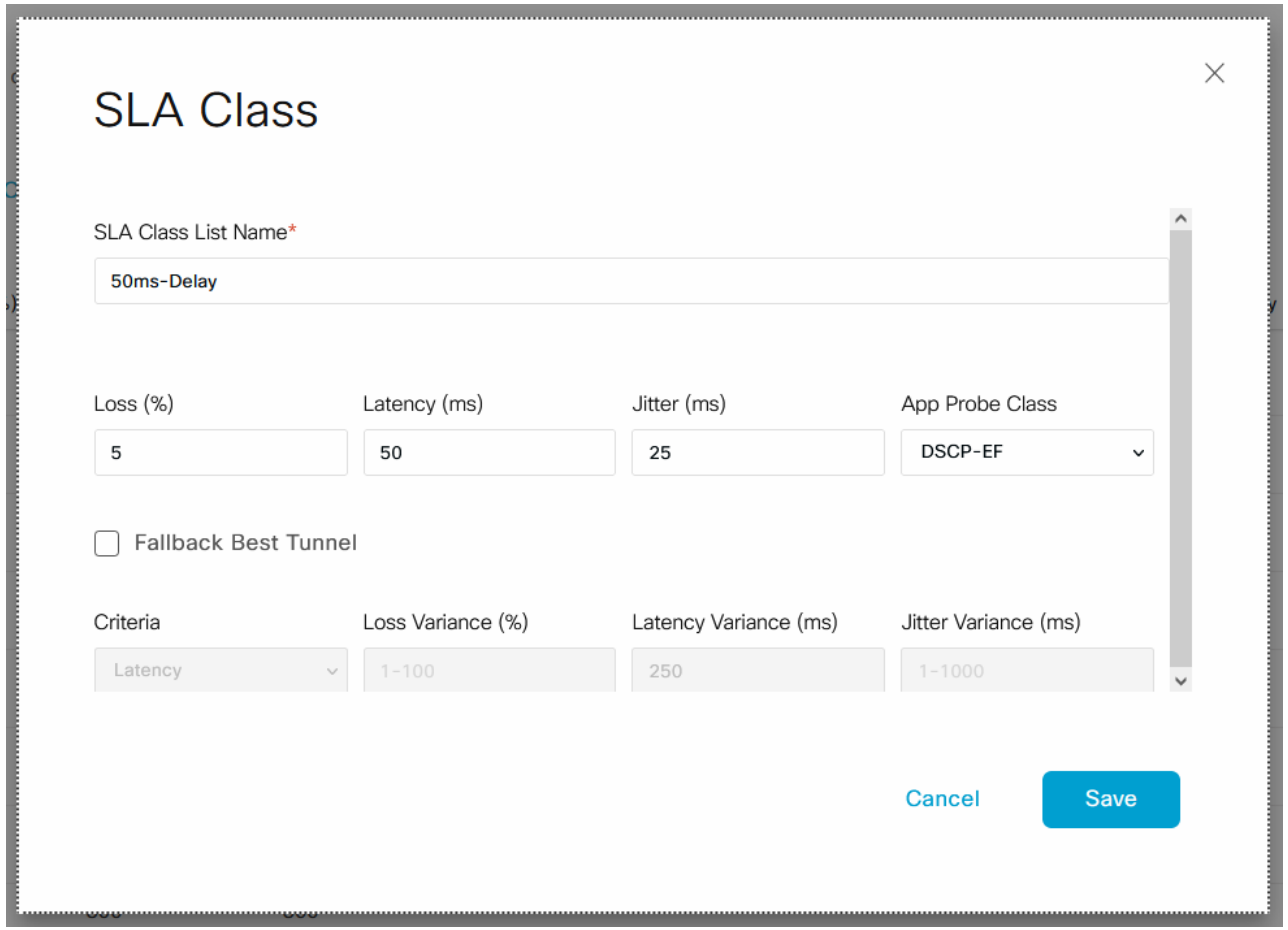
Defining an App Probe Class for Application Aware Routing (AAR)

- App Probe Class is defined under Centralized Policy Lists
  - Class-Map to define Queue number must first be defined under Localized Policy Lists
  - DSCP value is entered in decimal
    - E.g. DSCP EF (101110) = 46



Defining an SLA Class for Application Aware Routing (AAR)

- Custom **SLA Class** can be used to define loss, latency, and jitter limits
  - **SLA Class** calls **App Probe Class** which defines QoS marking of BFD sampling



Verifying Application Aware Routing (AAR)

- Verifying AAR from vManage
  - Monitor > Devices > [device] > Real Time > Device Options > App Route Statistics
  - Monitor > Devices > [device] > Real Time > Device Options > App Route SLA Class
  - Monitor > Devices > [device] > Troubleshooting > App Route Visualization
  - Monitor > Devices > [device] > Troubleshooting > Simulate Flows
- Verifying AAR from vSmart CLI
  - **`show run [policy]`**
- Verifying AAR from vEdge & cEdge CLI
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] policy app-route-policy-filter`**
  - **`show [sdwan] policy service-path`**

Implementing Cisco SD-WAN

SD-WAN Policy Example – Application Aware Routing (AAR)

SD-WAN Policy Example – Application Aware Routing (AAR)

- Configure an AAR Policy on the Spokes as follows:
  - Real-time traffic (e.g. VoIP) should only route over colors that have less than 50ms delay & jitter, or 2% packet loss
  - All other traffic should prefer to route over Biz-Internet, but failover to MPLS if packet loss exceeds 5%, or delay & jitter exceed 200ms

SD-WAN Policy Example – Application Aware Routing (AAR) Part 1

```
vSmart-1# sh run | begin ^policy
policy
 sla-class OTHER-TRAFFIC
  loss    5
  latency 200
  jitter  200
 !
 sla-class REALTIME-TRAFFIC
  loss    2
  latency 50
  jitter  50
 !
 app-route-policy _Corp_VPN_AAR-POLICY
  vpn-list Corp_VPN
   sequence 1
    match
     source-ip 0.0.0.0/0
     app-list  REAL_TIME_APPS
    !
    action
     sla-class REALTIME-TRAFFIC
    !
   !
   sequence 11
    match
     destination-data-prefix-list Any
    !
    action
     sla-class OTHER-TRAFFIC preferred-color biz-internet
   !
  !
```

SD-WAN Policy Example – Application Aware Routing (AAR) Part 2

```
!
 lists
  vpn-list Corp_VPN
   vpn 1
  !
  data-prefix-list Any
   ip-prefix 0.0.0.0/0
  !
  app-list REAL_TIME_APPS
   app rtp
   app sccp
   app sip
   app sip_soap
   app skinny
   app uaudp_rtp
  !
  site-list Spokes
   site-id 2-999
   !
  !
!
apply-policy
 site-list Spokes
  app-route-policy _Corp_VPN_AAR-POLICY
 !
!
```

Verifying Application Aware Routing (AAR)

- Verifying AAR from vManage
  - Monitor > Devices > [device] > Real Time > Device Options > App Route Statistics
  - Monitor > Devices > [device] > Real Time > Device Options > App Route SLA Class
  - Monitor > Devices > [device] > Troubleshooting > App Route Visualization
  - Monitor > Devices > [device] > Troubleshooting > Simulate Flows
- Verifying AAR from vSmart CLI
  - **`show run [policy]`**
- Verifying AAR from vEdge & cEdge CLI
  - **`show [sdwan] policy from-vsmart`**
  - **`show [sdwan] policy app-route-policy-filter`**
  - **`show [sdwan] policy service-path`**

Implementing Cisco SD-WAN

Cisco SD-WAN TLOC Extension

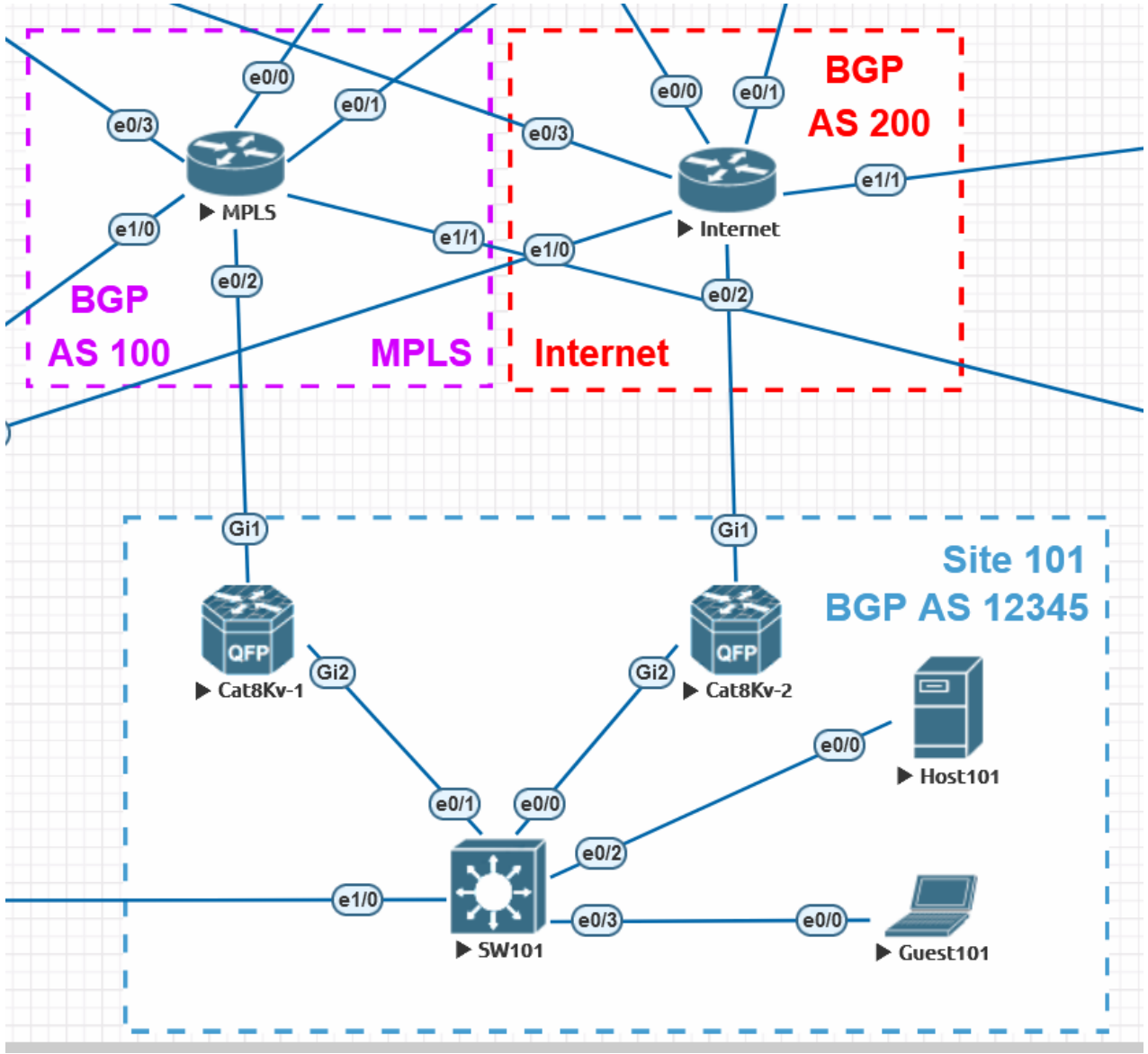
Cisco SD-WAN TLOC Review

- In Cisco SD-WAN a **Transport Locator** (**TLOC**) is an identifier used to represent a specific WAN transport interface on an SD-WAN Edge Router
  - I.e. an interface in VPN 0 on WAN Edge Router “cEdge-1”
- A **TLOC** is comprised of three main components
  - System IP
    - The unique identifier for each WAN Edge Router
  - Color
    - The type of transport, e.g. MPLS, Biz-Internet, LTE, etc.
  - Encapsulation
    - Either IPsec or GRE as the transport protocol
- By default, a full mesh of tunnels is established from and to all TLOCs
  - e.g. MPLS to MPLS, Internet to Internet, MPLS to Internet, Internet to MPLS, etc.
  - Can be limited to same-to-same color with **restrict** option under tunnel

Cisco SD-WAN TLOC Extension

- **TLOC Extension** is a feature used to add path diversity and redundancy in cases where multiple WAN Edge Routers in the same site only have a single WAN link each
  - E.g. in Site 101, Cat8Kv-1 has MPLS only, and Cat8Kv-2 has Biz-Internet only
- TLOC Extension allows a WAN Edge Router to share its TLOC with another Edge Router without adding additional physical WAN links
  - E.g. Cat8Kv-1 can share Cat8Kv-2's Biz-Internet link, without the need to physically dual-home to both MPLS and Biz-Internet providers
- TLOC Extension is accomplished by adding additional physical or logical East/West links between the WAN Edge Routers where new tunnels will terminate
  - E.g. Cat8Kv-1 indirectly connects to Biz-Internet through Cat8Kv-2

Example TLOC Extension Topology



Verifying TLOC Extension

- vSmart
  - **`show omp tlocs [system-ip]`**
- vEdge & cEdge
  - **`show [sdwan] omp tlocs`**
  - **`show [sdwan] bfd sessions`**

Implementing Cisco SD-WAN

Cisco SD-WAN Direct Cloud Access & Cloud OnRamp

What is Cisco SD-WAN Direct Cloud Access?

- Direct Cloud Access is a subset of Direct Internet Access (DIA) used to distribute access to AWS/Azure/GCP etc. to WAN Edge Routers using the local Internet breakout link
  - I.e. don't follow the default route over the SD-WAN to reach cloud providers

- Direct Cloud Access, like DIA, is implemented using a **Centralized Data Policy**
  - IF destination == AWS/Azure/GCP/etc. THEN NAT to VPN 0

What is Cisco SD-WAN Cloud OnRamp?

- Cisco SD-WAN **Cloud OnRamp** is a suite of features designed to optimize and simplify the integration of cloud applications and infrastructure with the SD-WAN
- Cloud OnRamp directly integrates with your cloud provider to automate VPN connectivity between the SD-WAN and the Cloud
  - E.g. vManage is configured with your AWS credentials
  - CSR1000v / Catalyst 8000v instances are automatically instantiated inside the Cloud, which are then used as Transit Gateways to reach your Virtual Private Cloud (VPC) over IPsec VPNs
  - Eliminates the need to manually configure IPsec VPNs on the cloud side
- Configured under **Configuration > Cloud onRamp** under vManage

Implementing Cisco SD-WAN

Course Conclusion

Recommended Resources

- Cisco Press SD-WAN Book
  - Cisco Software-Defined Wide Area Networks: Designing, Deploying and Securing Your Next Generation WAN with Cisco SD-WAN
- Cisco SD-WAN Documentation
  - https://www.cisco.com/c/en/us/td/docs/routers/sdwan/config/ios-xe-sdwan17.html
- Cisco DevNet Sandbox
  - Free virtual instances of SD-WAN devices for lab testing
  - https://devnetsandbox.cisco.com/